PDFlib GmbH

PDFlib, PDFlib+PDI, PPS

Eine Bibliothek für dynamisches PDF

PDFlib 9.0.4

API-Referenz

Ausgabe für C, C++, Cobol, COM, Java, .NET, Objective-C,
Perl, PHP, Python, REALbasic/Xojo, RPG, Ruby

PDFlib GmbH
Franziska-Bilek-Weg 9, D-80339 München
www.pdflib.com
Tel. +49 • 89 • 452 33 84-0
Fax +49 • 89 • 452 33 84-99

Bei Fragen können Sie die PDFlib-Mailing-Liste abonnieren und sich deren Archiv ansehen unter: groups.yahoo.com/neo/groups/pdflib/info.

Vertriebsinformationen: sales@pdflib.com
Support für Inhaber einer kommerziellen PDFlib-Lizenz: support@pdflib.com (geben Sie bitte immer Ihre Lizenznummer an)

Inhaltsverzeichnis

1 Konzepte der PDFlib-Programmierung

1.1 Optionslisten

Optionslisten bieten eine ebenso leistungsstarke wie einfache Methode zur Steuerung von PDFlib-API-Funktionsaufrufen. Sie werden von vielen PDFlib-API-Methoden unterstützt, damit man es sich ersparen kann, unzählige Funktionsparameter zu übergeben. Optionslisten *(optlists)*Strings, die beliebig viele Optionen enthalten können. Sie unterstützen verschiedene Datentypen und zusammengesetzte Datenstrukturen wie Listen. In den meisten Sprachbindungen lassen sich Optionslisten problemlos durch Konkatenieren der erforderlichen Schlüsselwörter und Werte bilden.

Bindungen C-Sprachbindung: Sie können zur Erstellung von Optionslisten die Funktion *sprintf( )* nutzen.

.NET-Sprachbindung: C#-Programmierer sollten beachten, dass die StringBuild-Methode *AppendFormat( )* die geschweiften Klammern *{* und *}* zur Darstellung von Formatelementen verwendet, die durch die String-Darstellung der Argumente ersetzt werden. Dabei schreibt die *Append( )*-Methode keine bestimmte Bedeutung für die Klammerzeichen vor. Da in der Syntax von Optionslisten geschweifte Klammern verwendet werden, sollten Sie darauf achten, die Methoden *AppendFormat( )* oder *Append( )* korrekt anzuwenden.

1.1.1 Syntax

Formale Syntaxdefinition von Optionslisten. Optionslisten müssen nach folgenden Regeln konstruiert werden:

- Alle Elemente (Schlüssel und Werte) in einer Optionsliste müssen durch eines oder mehrere der folgenden Zeichen getrennt werden: Leerzeichen, Tabulator, Carriage Return, Newline oder durch ein Gleichheitszeichen '='.
- Das äußerste Paar von geschweiften Klammern ist nicht Teil des Elements. Aufeinanderfolgende geschweifte Klammern { } kennzeichnen ein leeres Element.
- Trennzeichen innerhalb des äußersten Paars von geschweiften Klammern teilen das Element nicht, sondern sind Teil des Elements. Deshalb muss ein Element, das Trennzeichen enthält, von geschweiften Klammern eingeschlossen werden.
- Wenn ein Element geschweifte Klammern enthält, müssen diese mit einem vorangestellten Backslash geschützt werden.
- Wenn ein Element eine Sequenz von einem oder mehreren Backslash-Zeichen vor einer geschweiften Klammer enthält, muss jeder Backslash in der Folge mit einem zusätzlichen Backslash geschützt werden.
- Optionslisten dürfen keine binären Null-Werte enthalten.

Eine Option kann einen Listenwert wie in der PDFlib-Referenz angegeben haben. Listenwerte enthalten ein oder mehrere Elemente (die selbst Listen sein können). Sie werden nach den oben genannten Regeln getrennt, mit dem einzigen Unterschied, dass das Gleichheitszeichen nicht mehr als Trennzeichen behandelt wird.

Hinweis *Optionsnamen (also Schlüssel) enthalten keine Binde-/Trennstriche. Beachten Sie dies, da die Tabellen mit den Optionsbeschreibungen manchmal lange Optionsnamen enthalten, die mit Trennstrich getrennt sind. Der Trennstrich muss weggelassen werden, wenn die Option in einer Optionsliste übergeben wird.*

Einfache Optionslisten. In vielen Fällen enthalten Optionslisten ein oder mehrere Schlüssel-/Wertpaare. Schlüssel und Wert sowie die Paare selbst müssen durch Leerzeichen, Tabulator, Carriage Return oder Newline getrennt werden. Sie können zwischen Schlüssel und Wert auch ein Gleichheitszeichen '=' setzen:

```
schlüssel=wert
schlüssel = wert
schlüssel wert
schlüssel1 = wert1  schlüssel2 = wert2
```

Zur Verbesserung der Lesbarkeit empfehlen wir, zwischen Schlüssel und Wert ein Gleichheitszeichen zu setzen und zwischen aufeinanderfolgende Schlüssel-/Wertpaare ein Leerzeichen.

Da Optionslisten von links nach rechts ausgewertet werden, kann eine Option in der Liste mehrfach übergeben werden. In diesem Fall hat das letzte Auftreten der Option Vorrang vor früheren Auftritten. Im folgenden Beispiel wird die erste Wertzuweisung an die Option *schlüssel* durch die zweite überschrieben, und *schlüssel* besitzt nach Verarbeitung der Optionsliste den Wert 2:

```
schlüssel=1 schlüssel=2
```

Listenwerte. Listenwerte bestehen aus mehreren Werten, die einfache Werte oder wiederum Listenwerte sein können. Listen werden mit *{* und *}* geklammert und müssen durch Leerzeichen, Tabulator, Carriage Return oder Newline getrennt werden, zum Beispiel:

```
dasharray={11 22 33}                        (Liste bestehend aus drei Zahlen)
position={ center bottom }                  (Liste bestehend aus zwei Schlüsselwörtern)
```

Eine Liste kann auch aus verschachtelten Listen bestehen. Die Listen müssen in diesem Fall durch Leerzeichen voneinander getrennt werden. Zwischen den aufeinanderfolgenden Klammerzeichen *}* und *{* muss ein Leerzeichen gesetzt werden; dieses kann bei Klammern des gleichen Typs weggelassen werden:

```
polylinelist={{10 20 30 40} {50 60 70 80}}    (Liste bestehend aus zwei Listen)
```

Wenn die Liste aus genau einer verschachtelten Liste besteht, dürfen die Klammern für die untergeordnete Liste nicht weggelassen werden:

```
polylinelist={{10 20 30 40}}          (Liste bestehend aus einer verschachtelten Liste)
```

Verschachtelte Optionslisten und Listenwerte. Einige Optionen des Typs »Liste« arbeiten mit dem Typ *Optionsliste* oder *Liste von Optionslisten*. Optionen des Typs *Optionsliste* enthalten eine oder mehrere verschachtelte Optionen. Optionen des Typs *Liste von Optionslisten* enthalten eine oder mehrere verschachtelte Optionslisten. Achten Sie bei verschachtelten Optionslisten auf die richtige Anzahl schließender Klammern. Im Folgenden finden Sie hierzu einige Beispiele.

Der Wert der Option *metadata* ist eine Optionsliste, die eine einzige Option *filename* enthält:

```
metadata={filename=info.xmp}
```

Der Wert der Option *fill* ist eine Liste von Optionslisten, die eine einzige Optionsliste enthält:

```
fill={{ area=table fillcolor={rgb 1 0 0} }}
```

Der Wert der Option *fill* ist eine Liste von Optionslisten, die zwei Optionslisten enthält:

```
fill={{ area=rowodd fillcolor={rgb 0 1 0} } { area=roweven fillcolor={rgb 1 0 0} }}
```

Die Liste enthält eine Optionsliste mit einem Wert, der Leerzeichen enthält:

```
attachments={{filename={foo bar.xml} }}
```

Liste bestehend aus drei Strings:

```
itemnamelist = { {Isaac Newton} {James Clark Maxwell} {Albert Einstein} }
```

Liste bestehend aus zwei Schlüsselwörtern:

```
position={left bottom}
```

Liste bestehend aus verschiedenen Typen (Float-Wert und Schlüsselwörtern):

```
position={10 bottom}
```

Liste bestehend aus einem Rechteck:

```
boxes={{10 20 30 40}}
```

Die Liste enthält zwei Polylinien mit Prozentwerten:

```
polygons = {{10 20 40 60 90 120}} {12 87 34 98 34% 67% 34% 7%}}
```

Häufige Fehler und Stolpersteine. In diesem Abschnitt werden einige häufige Fehler in Bezug auf die Syntax von Optionslisten aufgeführt.

Geschweifte Klammern sind keine Trennzeichen; das folgende Beispiel ist falsch:

```
schlüssel1 {wert1}schlüssel2 {wert2}        FALSCH!
```

Dies löst folgende Fehlermeldung aus: *Unknown option 'wert2'*. Ebenso ist folgendes Beispiel falsch, da die Trennzeichen fehlen:

```
schlüssel{wert}                             FALSCH!
schlüssel={{wert1}{wert2}}                  FALSCH!
```

Klammern müssen ausgeglichen sein; das folgende Beispiel ist falsch (siehe nächsten Abschnitt):

```
schlüssel={offene klammer {}                FALSCH!
```

Dies löst folgende Fehlermeldung aus: *Braces aren't balanced in option list 'key={offene klammer {}'*. Einem einzelnen Klammerzeichen als Teil eines Strings muss ein zusätzlicher Backslash vorangestellt werden:

```
schlüssel={geschlossene klammer \} und offene klammer \{}        RICHTIG!
```

Einem Backslash am Ende eines String-Werts muss ein zusätzlicher Backslash vorangestellt werden, wenn darauf ein schließendes Klammerzeichen folgt:

```
schlüssel={\wert\}                          FALSCH!
schlüssel={\wert\\}                         RICHTIG!
```

Nicht eingeschlossene String-Werte in Optionslisten. In den folgenden Fällen kann es zu Konflikten zwischen den Zeichen in einem Optionswert und Zeichen in der Syntax von Optionslisten kommen:

- Kennwörter können nicht ausgeglichene Klammern, Backslashes und andere Sonderzeichen enthalten
- Japanische SJIS-Dateinamen in Optionslisten (nur sinnvoll in nicht Unicode-fähigen Sprachbindungen)
- JavaScript-Code in Optionen ist wegen der Verwendung von geschweiften Klammern *{* und *}* problematisch

Um beliebigen Text oder Binärdaten zu übergeben, die nicht mit den Syntaxelementen von Optionslisten kollidieren, lassen sich nicht eingeschlossene Optionswerte zusammen mit einer Längenangabe in folgenden Syntaxvarianten angeben:

```
schlüssel[n]=wert
schlüssel[n]={wert}
```

Die Dezimalzahl *n* hat folgende Bedeutung:

- in Unicode-fähigen Sprachbindungen: die Anzahl der UTF-16-Code-Einheiten
- in nicht Unicode-fähigen Sprachbindungen: die Anzahl von Bytes, aus denen der String besteht

Die geschweiften Klammern um den String-Wert sind optional, aber dringend empfohlen. Sie sind für Strings beginnend mit einem Leerzeichen oder anderen Trennzeichen erforderlich. Geschweifte Klammern, Trennzeichen und Backslashes im String-Wert werden als solche interpretiert.
Das folgende Beispiel zeigt ein Kennwort bestehend aus 7 Zeichen mit Leerzeichen und Klammern. Der gesamte String ist von geschweiften Klammern umschlossen, die nicht Teil des Optionswerts sind:

```
kennwort[7]={ ab}c d}
```

Falls ein Optionswert in einer verschachtelten Optionsliste mit einer Längenangabe übergeben wird, muss die umgebende Optionsliste ebenfalls eine Längenangabe enthalten, zum Beispiel:

```
fitannotation[38]={inhalt[25]={dies ist eine klammer '}'}}
```

1.1.2 Einfache Datentypen

String. Strings sind einfache ASCII-Strings (bzw. EBCDIC-Strings auf EBCDIC-Plattformen) wie sie im Allgemeinen für nicht lokalisierbare Schlüsselwörter verwendet werden. Strings, die Leerzeichen oder Gleichheitszeichen '=' enthalten, müssen mit *{* und *}* geklammert werden:

```
kennwort={ secret string }              (Stringwert mit drei Leerzeichen)
inhalt={length=3mm}                     (Stringwert mit einem Gleichheitszeichen)
```

Vor den Zeichen *{* und *}* muss ein zusätzlicher Backslash \ stehen, wenn sie zum String gehören sollen:

```
kennwort={weird\}string}                (Stringwert mit einer rechten Klammer)
```

Einem Backslash vor der schließenden Klammer eines Elements muss ein zusätzlicher Backslash vorangestellt werden:

```
filename={C:\pfad\name\\}               (String endet mit einem einzelnen Backslash)
```

Ein leerer String wird durch ein Klammernpaar dargestellt:

```
{}
```

Content-Strings, Hypertext-Strings und Name-Strings: Diese können Unicode-Inhalt in verschiedenen Formaten enthalten. Einzelne Bytes können durch eine Escape-Sequenz ausgedrückt werden, sofern die Option *escapesequence* gesetzt ist. Einzelheiten über diese String-Typen sowie die Encodings für String-Optionen finden Sie im *PDFlib-Tutorial*.

Nicht Unicode-fähige Sprachbindungen: wenn die Optionsliste mit einem [EBCDIC]-UTF-8-BOM beginnt, wird jeder Content-, Hypertext- oder Name-String der Optionsliste als [EBCDIC-]UTF-8-String interpretiert.

Unichar. Ein Unichar ist ein einzelner Unicode-Wert, wobei mehrere syntaktische Varianten unterstützt werden: Dezimalwerte ≥ 10 (zum Beispiel 173), Hexadezimalwerte, die mit *x, X, ox, oX* oder *U+* beginnen *(xAD, oxAD, U+00AD)*, numerische, Character- oder Glyphnamen-Referenzen, aber ohne Auszeichnung durch '&' oder ';' *(shy, #xAD, #173)*. Alternativ können Buchstaben übergeben werden, zum Beispiel:

```
replacementchar=?                   (Buchstabe)
replacementchar=63                  (dezimal)
replacementchar=x3F                 (hexadezimal)
replacementchar=0x3F                (hexadezimal)
replacementchar=U+003F              (Unicode-Notation)
replacementchar=euro                (HTML-Character-Referenz)
replacementchar=.question           (Standard-Glyphnamen-Referenz)
replacementchar=.marina             (Fontspezifische Glyphnamen-Referenz)
```

Einzelne Zeichen, die Zahlen darstellen, werden als solche interpretiert und nicht als dezimale Unicode-Werte:

```
replacementchar=3                   (U+0033 DREI, nicht U+0003!)
```

Unichars müssen im hexadezimalen Bereich *0-0x10FFFF* (dezimal *0-1114111)* liegen. Einige Optionen sind jedoch auf den Bereich *0-0xFFFF (0-65535)* beschränkt. Dies ist in der jeweiligen Beschreibung der Option erwähnt.

Unicode-Bereich. Ein Unicode-Bereich kennzeichnet einen zusammenhängenden Bereich von Unicode-Zeichen durch Start- und Ende-Zeichen des Bereichs. Anfangs- und Endwerte eines Unicode-Bereichs müssen durch ein Minuszeichen '-' ohne Leerzeichen getrennt werden, zum Beispiel:

```
forcechars={U+03AC-U+03CE}
```

Boolean. Boolesche Optionen haben die Werte *true* oder *false*; wird bei einer Option Booleschen Typs kein Wert angegeben, wird von *true* ausgegangen. Als abkürzende Schreibweise kann *noname* statt *name=false* verwendet werden:

```
embedding            (Äquivalent zu embedding=true)
noembedding          (Äquivalent zu embedding=false)
```

Schlüsselwort. Eine Option vom Typ Schlüsselwort kann eine vordefinierte Liste von festen Schlüsselwörtern enthalten, zum Beispiel:

```
blendmode=overlay
```

Bei einigen Optionen kann der Wert entweder eine Zahl oder ein Schlüsselwort sein.

Zahl. Optionslisten unterstützen verschiedene numerische Typen.
Integers können dezimal oder hexadezimal sein. Positive Integer-Werte beginnend mit *x, X, ox* oder *oX* geben hexadezimale Werte an:

```
-12345
0
0xFF
```

Floats können dezimale Gleitkomma- oder Ganzzahlen enthalten; zur Trennung von Vor- und Nachkommastellen sind Punkt und Komma zulässig. Exponentialdarstellung wird ebenfalls unterstützt. Die folgenden Werte sind alle gleichbedeutend:

```
size = -123.45
size = -123,45
size = -1.2345E2
size = -1.2345e+2
```

Prozentangaben sind Zahlen mit einem Prozentzeichen % direkt hinter dem numerischen Wert. Manche Optionen erlauben negative Prozentwerte:

```
leading=120%
topoffset=-20.5%
```

Handle. Handles bezeichnen verschiedene Objekttypen, zum Beispiel Font-, Image-, ICC-Profil- oder Action-Handles. Technisch gesehen handelt es sich um Integer-Werte, die von einem früheren API-Aufruf zurückgegeben wurden. Von *PDF_load_image( )* wird zum Beispiel ein Image-Handle zurückgegeben. Handles müssen immer als opake Typen behandelt werden; sie dürfen nie direkt von der Anwendung geändert oder erzeugt werden (es dürfen also nur Handles verwendet werden, die von einer API-Funktion zu-

rückgegeben wurden). Handles gelten nur für den entsprechenden Objekttyp. Eine Option, die ein Image-Handle erwartet, darf nicht mit einem Graphic-Handle versehen werden, obwohl es sich bei beiden Handles um Integer-Typen handelt.

1.1.3 Datentypen Fontgröße und Aktion

Fontgröße. Eine Fontgröße kann auf verschiedene Arten festgelegt werden, wobei die Größe des Textes in absoluten Werten, relativ zu einer externen Bezugsgröße oder relativ zu einer bestimmten Fonteigenschaft angegeben werden kann. Sofern in der Beschreibung der Option nicht anders angegeben, muss die Fontgröße von 0 verschieden sein.

Im häufigsten Fall enthält die Fontgröße einen einzelnen Float-Wert, der sich auf die Einheiten im Benutzerkoordinatensystem bezieht:

```
fontsize=12
```

Eine zweite Variante enthält einen Prozentwert, wobei die Basis des Prozentwerts vom Kontext anhängt (zum Beispiel die Breite der Fitbox für *PDF_fit_textline( )):*

```
fontsize=8%
```

In einer dritten Variante wird die Fontgröße als Optionsliste angegeben, die ein Schlüsselwort oder eine Zahl enthalten muss. Das Schlüsselwort beschreibt die gewünschte Fontmetrik gemäß Tabelle 1.1, und die Zahl gibt die gewünschte Größe an. PDFlib berechnet die richtige Fontgröße, so dass die ausgewählte Text-Metrik dem übergebenen Wert entspricht:

```
fontsize={capheight 5}
```

Tabelle 1.1 Unteroptionen für Optionen vom Typ `fontsize`

Option	*Beschreibung*
ascender	*Die Zahl wird als Oberlänge interpretiert.*
bodyheight	*Die Zahl wird als minimaler Abstand zwischen den Grundlinien interpretiert, d.h. Unter- und Oberlänge benachbarter Zeilen berühren sich exakt, wenn dieser Wert als Zeilenabstand verwendet wird. Dies ist das Standardverhalten, wenn kein Schlüsselwort übergeben wird.*
capheight	*Die Zahl wird als Höhe eines Großbuchstabens interpretiert.*
xheight	*Die Zahl wird als Höhe eines Kleinbuchstabens interpretiert.*

Aktionslisten. Eine Aktionsliste legt eine oder mehrere Aktionen fest. Jeder Listeneintrag besteht aus einem Ereignis-Schlüsselwort (Auslöser) und einer Liste von Action-Handles, die mit *PDF_create_action( )* zurückgegeben wurden. Aktionen werden in der genannten Reihenfolge ausgeführt. Die Menge zulässiger Ereignisse (zum Beispiel *docopen)* und der Aktionstyp (zum Beispiel JavaScript) werden bei der jeweiligen Option beschrieben.

Liste mit einem einzelnen Auslöser bei drei Aktionen:

```
action={ activate={ 0 1 2 } }
```

Liste mit drei Auslösern und einer Aktion pro Auslöser:

```
action={ keystroke=0 format=1 validate=2 }
```

1.1.4 Datentyp Farbe

Farbräume im Überblick. Sie können die Farben zum Zeichnen und Füllen von Umrisslinien sowie für Textzeichen festlegen. Farben können in verschiedenen Farbräumen angegeben werden (jedes Listenelement beginnt mit dem entsprechenden Schlüsselwort von *PDF_setcolor( )* für den Farbraum und die Farboptionen):

- *gray:* Graustufen zwischen 0=schwarz und 1=weiß;
- *rgb:* RGB-Tripel, das heißt drei Werte zwischen 0 und 1 für den prozentualen Anteil an Rot, Grün und Blau; *(0, 0, 0)*=schwarz, *(1, 1, 1)*=weiß. Die häufig verwendeten RGB-Farbwerte im Bereich 0–255 müssen durch 255 geteilt werden, um sie wie für PDFlib erforderlich auf den Bereich 0–1 zu skalieren.
  Alternativ zu den numerischen RGB-Werten können Sie die RGB-Farben über ihre HTML-Namen oder hexadezimalen Werte ansprechen.
- *cmyk:* Vier CMYK-Werte zwischen 0 = keine Farbe und 1 = volle Farbe, dies entspricht Werten für Cyan, Magenta (Pink), Yellow (Gelb) und Black (Schwarz); *(0, 0, 0, 0)*=weiß, *(0, 0, 0, 1)*=schwarz. Beachten Sie den Unterschied zur RGB-Spezifikation.
- *iccbased* (nicht für *PDF_setcolor( )*) und *iccbasedgray/rgb/cmyk:* ICC-basierte Farben basieren auf einem ICC-Profil.
- *spotname:* Name einer vordefinierten Schmuckfarbe und Prozentwert für den Farbauftrag im Bereich 0=keine Farbe bis 1=maximale Intensität.
  Alternativ dazu Name einer benutzerdefinierten Schmuckfarbe, Prozentwert für den Farbauftrag und eine Alternativdarstellung in einem der oben erwähnten Farbräume.
- *spot:* Handle für eine vor- oder benutzerdefinierte Schmuckfarbe und ein Prozentwert für den Farbauftrag.
- *lab* erwartet geräteunabhängige Farben im Farbraum CIE L*a*b* mit D50-Normlicht. Farben werden über einen Helligkeitswert *(luminance)* zwischen 0–100 und zwei Farbwerte *a* und *b* im Bereich -128 und 127 festgelegt. Die Komponente *a* reicht von Grün bis Rot/Magenta (negative Werte sind grün, positive Werte sind magenta), und die Komponente *b* reicht von Blau bis Gelb (negative Werte sind blau, positive Werte sind gelb).
- *pattern:* Kachel-Pattern mit einem Objekt, das aus beliebigem Text, Vektorgrafik oder Rasterbildern zusammengesetzt ist. Pattern können mit *PDF_begin_pattern_ext( )* oder *PDF_shading_pattern( )* erstellt werden und werden durch ein Pattern-Handle ausgewählt.

Die Standardfarbe für das Zeichnen und Füllen von Umrisslinien ist schwarz. Der Farbraum für diese Standardfarbe wird automatisch gemäß der Farbanforderungen von PDF/X und PDF/A ausgewählt.

Hinweis (Weiche) Farbverläufe stellen einen kontinuierlichen Übergang zwischen zwei Farben her. Sie lassen sich mit PDF_shading( ) erzeugen.

Farboptionen. Farbräume können auf drei unterschiedliche Arten festgelegt werden: mit einem RGB-Farbnamen, hexadezimalen RGB-Werten oder einer flexiblen Optionsliste für Farben in einem beliebigen Farbraum.

Cookbook *Ein vollständiges Codebeispiel zur Verwendung von RGB-Farbräumen finden Sie im Cookbook-Topic* `color/web_colornames`.

Bei der ersten Art können alle gültigen Farbnamen von SVG 1.1 zur Definition einer RGB-Farbe oder einer sRGB-Farbe direkt angegeben werden, wenn das sRGB-ICC-Profil ausgewählt wurde, zum Beispiel:

```
strokecolor=pink
```

Bei Farbnamen wird nicht nach Groß- und Kleinschreibung unterschieden. Für eine Liste gültiger RGB-Farbnamen siehe:

www.w3.org/TR/SVG11/types.html#ColorKeywords

Auf eine zweite Art kann ein Hash-Zeichen '#' gefolgt von drei Paaren von hexadezimalen Ziffern *00-FF* übergeben werden, um einen RGB-Farbwert festzulegen, zum Beispiel:

```
strokecolor=#FFC0CB
```

Auf eine dritte Art kann eine Farboptionsliste aus einem Farbraum und einem Farbwert bestehen. Eine Farboptionsliste besteht aus einem Farbraum-Schlüsselwort und einer Liste von Float-Werten, deren Anzahl vom jeweiligen Farbraum abhängt. Farbraum-Schlüsselwörter werden dabei wie in *PDF_setcolor( )* festgelegt (siehe Abschnitt 8.1, »Festlegen von Farbe und Farbraum«, Seite 169). Tabelle 1.2 enthält genaue Beschreibungen mit Beispielen. Wie in den entsprechenden Funktionsbeschreibungen ausgeführt, unterstützen manche Optionen nur eine Teilmenge der Farbraum-Schlüsselwörter.

Cookbook *Ein vollständiges Codebeispiel hierzu finden Sie im Cookbook-Topic* `color/starter_color`.

Tabelle 1.2 Schlüsselwörter für die Farbdatentypen in Optionslisten

Schlüsselwort	Weitere Werte	Beispiel
gray	*Ein Float-Wert für den Graustufen-Farbraum*	`{ gray 0.5 }`
rgb	*Drei Float-Werte für den Farbraum RGB*	`{ rgb 1 0 0 }`
(kein Schlüsselwort)	*HTML-Farbname oder hexadezimale Werte für eine RGB-Farbe*	`pink` `#FFC0CB`
cmyk	*Vier Float-Werte für den Farbraum CMYK*	`{ cmyk 0 1 0 0 }`
lab	*Drei Float-Werte für den Farbraum Lab*	`{ lab 100 50 30 }`
spot	*Schmuckfarben-Handle und ein Float-Wert für den Farbauftrag*	`{ spot 1 0.8 }`
spotname	*(bis zu 63 Bytes; weniger Unicode-Zeichen je nach Format und Encoding) Schmuckfarbenname und ein Float-Wert für den Farbauftrag*	`{ spotname {PANTONE 281 U} 0.5 }`
spotname	*Ähnelt der einfachen Variante von* `spotname` *oben, kann aber zusätzlich einen Farbwert für die Alternativfarbe einer benutzerdefinierten Schmuckfarbe enthalten (d.h. einen Schmuckfarbnamen, der PDFlib unbekannt ist). Definieren mehrere Optionen denselben benutzerdefinierten Schmuckfarbennamen, müssen alle Definitionen konsistent sein (d.h. dieselbe Alternativfarbe festlegen).*	`{ spotname {PDFlib Blue} 0.5 { lab 100 50 30 } }`

Tabelle 1.2 Schlüsselwörter für die Farbdatentypen in Optionslisten

Schlüsselwort	Weitere Werte	Beispiel

iccbased	*ICC-Profil-Handle oder Schlüsselwort* `srgb`*, plus 1, 3 oder 4 Farbwerte je nach Typ des ICC-Profils (gray, RGB oder CMYK). Das Schlüsselwort* `srgb` *darf im Gültigkeitsbereich* `document` *nicht verwendet werden.*	`{ iccbased <handle> 0.5 }`
		`{ iccbased <handle> 0 0 0.75 }`
		`{ iccbased srgb 0 0 0.75 }`
		`{ iccbased <handle> 0 0 0.3 1 }`

iccbasedgray	*Einzelner Float-Wert, der sich auf ein ICC-Profil bezieht, das mit der Option* `iccprofilegray` *ausgewählt wurde*	`{ iccbasedgray 0.5 }`
iccbasedrgb	*Zwei Float-Werte, die sich auf ein ICC-Profil beziehen, das mit der Option* `iccprofilergb` *ausgewählt wurde*	`{ iccbasedrgb 1 0 0 }`
iccbasedcmyk	*Drei Float-Werte, die sich auf ein ICC-Profil beziehen, das mit der Option* `iccprofilecmyk` *ausgewählt wurde*	`{ iccbasedgray 0 1 0 0 }`
pattern	*Pattern-Handle*	`{ pattern 1 }`
none	*Legt fest, dass keine Farbe vorhanden ist.*	`none`

1.1.5 Geometrische Datentypen

Linie. Eine Linie ist eine Liste von vier Float-Werten, die die *x*- und *y*-Koordinaten des Start- und Endpunkts einer Geraden festlegen. Das Koordinatensystem für die Interpretation der Koordinaten (Standard- oder Benutzerkoordinatensystem) variiert je nach Option und wird deswegen bei der jeweiligen Option beschrieben:

```
line = {10 40 130 90}
```

Polylinie. Eine Polylinie ist eine Liste mit einer geraden Anzahl *n* von Float-Werten mit *n*>2. Jedes Paar in der Liste legt die *x*- und *y*-Koordinaten eines Punktes fest; diese Punkte werden durch Liniensegmente verbunden. Das Koordinatensystem für die Interpretation der Koordinaten (Standard- oder Benutzerkoordinatensystem) variiert je nach Option und wird deswegen bei der jeweiligen Option beschrieben:

```
polyline = {10 20 30 40 50 60}
```

Die folgenden Optionslisten sind gleichbedeutend:

```
polyline = {10 20 30r 40r 50r 60r}
polyline = {10 20 40 60 90 120}
```

Vierecke sind ein bestimmter Typ von Polylinien: dies sind Rechtecke, die rotiert werden können und für die genau vier Punkte festgelegt werden müssen.

Ein weiterer Spezialtyp sind Polygone: dies sind Polylinien, die automatisch durch ein Liniensegment geschlossen werden.

Rechtecke. Ein Rechteck besteht aus einer Liste von vier Float-Werten, die die *x*- und *y*-Koordinaten der linken unteren und der rechten oberen Ecke des Rechtecks festlegen. Das Koordinatensystem zur Interpretation der Rechteckkoordinaten (Standard- oder Benutzerkoordinatensystem) kann je nach Option unterschiedlich sein und wird deswegen bei der jeweiligen Option beschrieben. Manche Optionen akzeptieren Prozentwerte, wobei die Basis des Prozentwerts vom Kontext abhängt (zum Beispiel die Fitbox eines Textflows). Relative Koordinaten lassen sich durch Hinzufügen des Suffix *r* direkt

hinter einer Zahl angeben. Innerhalb einer Koordinatenliste bezieht sich eine relative Koordinate auf die vorige *x*- oder *y*-Koordinate. Relative Koordinaten am Anfang einer Liste beziehen sich auf den Ursprung, das heißt sie sind absolute Koordinaten, zum Beispiel:

```
cropbox={ 0 0 500 600 }
box={40% 30% 50% 70%}
```

Die folgenden Optionen sind gleichbedeutend:

```
box={12 34 56r 78r}
box={12 34 68 112}
```

Kreis. Ein Kreis besteht aus einer Liste von vier Float-Werten, wobei das erste Paar die *x*- und *y*-Koordinaten des Mittelpunkts festlegt und das zweite Paar die *x*- und *y*-Koordinaten eines beliebigen Punkts auf dem Kreis. Das Koordinatensystem zur Interpretation der Rechteckkoordinaten (Standard- oder Benutzerkoordinatensystem) kann je nach Option unterschiedlich sein und wird deswegen bei der jeweiligen Option beschrieben:

```
circle={200 325 200 200}
```

Kurvenliste. Eine Kurvenliste besteht aus zwei oder mehr verbundenen Bézier-Kurvensegmenten dritter Ordnung. Eine Bézierkurve wird durch vier Kontrollpunkte festgelegt. Der erste Kontrollpunkt ist der Startpunkt und der vierte Punkt ist der Endpunkt der Kurve. Mit dem zweiten und dritten Punkt wird die Form der Kurve bestimmt. In einer Kurvenliste dient der letzte Punkt eines Segments als erster Punkt für das nächste Segment.

```
curve={200 700 240 600 80 580 400 660 400 660 440 620}
```

Nach dem Zeichnen der Kurven wird der letzte Kontrollpunkt zum neuen aktuellen Punkt.

1.2 Gültigkeitsbereich von Funktionen

PDFlib-Anwendungen müssen sich an einige strukturelle Regeln halten, die sehr einfach zu verstehen sind. So wird man eine Seite zum Beispiel kaum schließen, bevor man sie geöffnet hat. PDFlib erzwingt die korrekte Reihenfolge von Funktionsaufrufen durch streng definierte Gültigkeitsbereiche (so genannte *scopes)*. Tabelle 1.3 definiert die verschiedenen Gültigkeitsbereiche. Zu jeder PDFlib-API-Funktion werden deren Gültigkeitsbereiche beschrieben. PDFlib löst eine Exception aus, wenn eine Funktion außerhalb ihres zulässigen Gültigkeitsbereichs aufgerufen wird. Der aktuelle Gültigkeitsbereich kann mit dem Schlüsselwort *scope* von *PDF_get_option( )* abgefragt werden.

Tabelle 1.3 Gültigkeitsbereiche für Funktionen

Name	Definition
path	*beginnt mit PDF_moveto( ), PDF_circle( ), PDF_arc( ), PDF_arcn( ), PDF_rect( ), PDF_ellipse( ) oder PDF_elliptical_arc( );* *endet mit einer der Funktionen aus Abschnitt 7.5, »Zeichnen und Beschneiden von Pfaden«, Seite 161*
page	*zwischen PDF_begin_page_ext( ) und PDF_end_page_ext( ), aber außerhalb von* `path`
template	*zwischen PDF_begin_template_ext( ) und PDF_end_template_ext( ), aber außerhalb von* `path`
pattern	*zwischen PDF_begin_pattern_ext( ) und PDF_end_pattern( ), aber außerhalb von* `path`
font	*zwischen PDF_begin_font( ) und PDF_end_font( ), aber außerhalb von* `glyph`
glyph	*zwischen PDF_begin_glyph_ext( ) und PDF_end_glyph( ), aber außerhalb von* `path`
document	*zwischen PDF_begin_document( ) und PDF_end_document( ), aber außerhalb von* `page`, `template`, `pattern` *und* `font`
object	*solange ein PDFlib-Objekt vorhanden, aber außerhalb von* `document` *ist; in C- und Cobol-Bindungen: zwischen PDF_new( ) und PDF_delete( ), aber außerhalb von* `document`

1.3 Logging

Mit der Logging-Funktion lassen sich API-Aufrufe protokollieren. Der Inhalt der Logdatei kann bei der Fehlersuche hilfreich sein oder vom Support der PDFlib GmbH benötigt werden. Die Logging-Optionen können auf folgende Arten übergeben werden:

- Als Optionsliste für die Option *logging* von *PDF_set_option( )*, zum Beispiel:

```
p.set_option("logging", "filename=trace.log remove")
```

- In der Umgebungsvariablen *PDFLIBLOGGING*. Wenn Sie diese Variante wählen, beginnt die Protokollierung mit dem ersten Aufruf einer API-Funktion.

Tabelle 1.4 Unteroptionen für die Option `logging`

Schlüssel	Beschreibung
(leere Liste)	*Protokollierung aktivieren*
disable	*(Boolean) Protokollierung deaktivieren*
enable	*(Boolean) Protokollierung aktivieren*
filename	*(String) Name der Logdatei (*`stdout` *und* `stderr` *werden gesondert behandelt). Bei CICS wird diese Option ignoriert, und die Logging-Ausgabe immer nach* `stderr` *geschrieben. Die Ausgabe wird an bereits vorhandene Inhalte angehängt. Standardwert:* `pdflog` *bei z/OS* `PDFlib.log` *bei OS X und iSeries* `\PDFlib.log` *bei Windows* `/tmp/PDFlib.log` *bei allen anderen Systemen* *Der Name der Logdatei kann auch in der Umgebungsvariablen PDFLIBLOGFILE übergeben werden.*
flush	*(Boolean) Bei* `true` *wird die Logdatei nach jeder Ausgabe geschlossen und bei der nächsten Ausgabe erneut geöffnet. Damit ist gewährleistet, dass die Ausgabe sicher in die Datei geschrieben wird. Dies kann bei der Verfolgung von Programmabstürzen mit unvollständiger Logdatei nützlich sein. Die Verarbeitungsgeschwindigkeit verringert sich jedoch erheblich. Bei* `false` *wird die Logdatei nur einmal geöffnet. Standardwert:* `false`
remove	*(Boolean) Bei* `true` *wird eine gegebenenfalls bereits vorhandene Logdatei gelöscht, bevor die neue Ausgabe geschrieben wird. Standardwert:* `false`
includepid	*(Boolean; nicht für MVS) Angabe der Prozess-ID im Logdateinamen. Dies sollte aktiviert werden, wenn mehrere Prozesse den selben Logdateinamen verwenden. Standardwert:* `false`
includetid	*(Boolean; nicht für MVS) Angabe der Thread-ID im Logdateinamen. Dies sollte aktiviert werden, wenn mehrere Threads im selben Prozess den selben Logdateinamen verwenden. Standardwert:* `false`
includeoid	*(Boolean; nicht für MVS) Angabe der Objekt-ID im Logdateinamen. Dies sollte aktiviert werden, wenn mehrere PDFlib-Objekte im selben Thread den selben Logdateinamen verwenden. Standardwert:* `false`
stringlimit	*(Integer) Limit für die Anzahl der Zeichen pro Zeile oder 0 für kein Limit. Standardwert: 0*
removeonsuccess	*(Boolean) Entfernt die generierte Logdatei in PDF_delete( ), sofern keine Exception ausgelöst wird. Dies kann für die Analyse gelegentlicher Probleme in Multi-Threaded-Anwendungen oder bei sporadisch auftretenden Problemen nützlich sein. Wir empfehlen, diese Option nach Bedarf mit* `includepid`/`includetid`/`includeoid` *zu kombinieren.*

Tabelle 1.4 Unteroptionen für die Option `logging`

Schlüssel	Beschreibung
classes	*(Optionsliste) Liste mit Optionen des Typs Integer, wobei jede Option eine Logging-Klasse und der zugehörige Wert die Granularität beschreibt. Die Granularität 0 deaktiviert eine Logging-Klasse und positive Zahlen aktivieren eine Klasse. Je höher die Granularität ist, desto detaillierter ist die Ausgabe. Unterstützte Optionen (Standardwert:* `{api=1 warning=1}`*):*
	api *Protokolliert alle API-Funktionsaufrufe mit Parametern und Rückgabewerten. Bei* `api=2` *wird ein Zeitstempel vor jeden API-Funktionsaufruf gestellt, und veraltete Funktionen und Optionen werden markiert. Bei* `api=3` *werden try/catch-Aufrufe protokoliert (nützlich für die Fehlersuche bei Problemen mit verschachtelten Exceptions).*
	filesearch *Protokolliert alle Versuche, Dateien via* `SearchPath` *oder PVF zu finden.*
	resource *Protokolliert alle Versuche, Ressourcen über die Windows-Registry, UPR-Definitionen oder Ergebnisse der Ressourcensuche zu finden.*
	tagging *Operationen mit Strukturelementen (Tags)*
	user *Benutzerdefinierte Logging-Ausgabe, die mit der Option* `userlog` *übergeben wurde*
	warning *Protokolliert alle PDFlib-Warnungen, d.h. Fehlerbedingungen, die ignoriert oder intern behandelt werden können. Ist* `warning=2`*, werden auch Meldungen von Funktionen protokolliert, die zwar keine Exception auslösen, aber den Meldungstext zur Abfrage mit PDF_get_errmsg( ) liefern, sowie die Ursache für alle gescheiterten Versuche, eine Datei zu öffnen (Suche einer Datei in searchpath).*

2 Allgemeine Funktionen

2.1 Verarbeitung von Exceptions

Tabelle 2.1 enthält die für diesen Abschnitt relevante Option. Diese Option wird von vielen Funktionen unterstützt, wie in den jeweiligen Beschreibungen der Optionslisten erwähnt. Sie kann auch als globale Option an *PDF_set_option( )* übergeben werden, siehe Abschnitt 2.3, »Globale Optionen«, Seite 25.

Tabelle 2.1 Option zur Verarbeitung von Exceptions für PDF_set_option( )

Schlüssel	Beschreibung
errorpolicy	*(Schlüsselwort) Steuert das Verhalten verschiedener Funktionen im Fehlerfall. Diese Einstellung kann in vielen Funktionen durch die Option* `errorpolicy` *überschrieben werden und dient als Standardwert der gleichnamigen Option. Unterstützte Schlüsselwörter (Standardwert:* `legacy`*):* **legacy** *(Veraltet) Das Verhalten der Funktionen entspricht dem Verhalten von PDFlib 6.* **return** *Im Fehlerfall kehrt die Funktion zurück. Funktionen wie zum Beispiel PDF_load_image( ), die einen Fehlercode zurückgeben können, geben -1 (in PHP: 0) zurück. Funktionen, die als Ergebnis einen String ausgeben (z.B. PDF_fit_table( )), geben den String* `_error` *zurück. Anwendungsentwickler müssen den Rückgabewert auf -1 (in PHP: 0) oder* `_error` *prüfen, um einen Fehler abzufangen. Im Fehlerfall kann eine detaillierte Beschreibung des Problems mit PDF_get_errmsg( ) abgefragt werden. Diese Einstellung wird für neue Anwendungen empfohlen.* **exception** *Tritt ein Fehler auf, löst die Funktion eine Exception aus. Die Exception muss im Client-Code durch die entsprechende sprachspezifische Methode abgefangen werden. Das bis dahin erzeugte PDF-Ausgabedokument ist unbrauchbar und muss gelöscht werden.*

C++ Java C# **int get_errnum( )**

Perl PHP **int get_errnum( )**

C **int PDF_get_errnum(PDF *p)**

Ermittelt die Nummer der zuletzt ausgelösten Exception oder die Ursache für einen gescheiterten Funktionsaufruf.

Rückgabe Fehlercode der letzten Fehlerbedingung.

Gültigkeit Zwischen einer von PDFlib ausgelösten Exception und dem Ende der Lebensdauer eines PDFlib-Objekts. Alternativ dazu kann diese Funktion aufgerufen werden, nachdem eine Funktion den Fehlercode -1 (in PHP: 0) zurückgegeben hat, aber bevor eine Funktion aufgerufen wird, die in diesem Abschnitt nicht beschrieben wird.

Bindungen In C++, Java, Objective-C, .NET, PHP und REALbasic/Xojo ist diese Funktion auch als Methode *get_errnum( )* im Objekt *PDFlibException* verfügbar.

C++ Java C# ***String get_errmsg( )***

Perl PHP ***string PDF_get_errmsg( )***

C ***const char *PDF_get_errmsg(PDF *p)***

Ermittelt den beschreibenden Text der zuletzt ausgelösten Exception oder die Ursache eines gescheiterten Funktionsaufrufs.

Rückgabe Beschreibung der letzten Fehlerbedingung als Text.

Gültigkeit Zwischen einer von PDFlib ausgelösten Exception und dem Ende der Lebensdauer eines PDFlib-Objekts. Alternativ dazu kann diese Funktion aufgerufen werden, nachdem eine Funktion den Fehlercode -1 (in PHP: 0) zurückgegeben hat, aber bevor eine Funktion aufgerufen wird, die in diesem Abschnitt nicht beschrieben ist.

Bindungen In C++, Java, Objective-C, .NET, PHP und REALbasic/Xojo ist diese Funktion auch als Methode *get_errmsg( )* im Objekt *PDFlibException* verfügbar.

C++ Java C# ***String get_apiname( )***

Perl PHP ***string PDF_get_apiname( )***

C ***const char *PDF_get_apiname(PDF *p)***

Ermittelt den Namen der API-Funktion, die die letzte Exception ausgelöst hat oder scheiterte.

Rückgabe Name der Funktion, die die letzte Exception ausgelöst hat oder zuletzt mit einem Fehlercode gescheitert ist.

Gültigkeit Zwischen einer von PDFlib ausgelösten Exception und dem Ende der Lebensdauer eines PDFlib-Objekts. Alternativ dazu kann diese Funktion aufgerufen werden, nachdem eine Funktion den Fehlercode -1 (in PHP: 0) zurückgegeben hat, aber bevor eine Funktion aufgerufen wird, die in diesem Abschnitt nicht beschrieben ist.

Bindungen In C++, Java, Objective-C, .NET, PHP und REALbasic/Xojo ist diese Funktion auch als Methode *get_apiname( )* im Objekt *PDFlibException* verfügbar.

C++ ***void *get_opaque( )***

C ***void *PDF_get_opaque(PDF *p)***

Holt den Zeiger auf Benutzerdaten, der in PDFlib abgelegt wurde.

Rückgabe Diese Funktion gibt den Zeiger auf Benutzerdaten zurück, der im Aufruf von *PDF_new2( )* übergeben und in PDFlib gespeichert wurde.

Details PDFlib gibt den Zeiger auf Benutzerdaten immer unverändert an den Client weiter. Dies kann in Multithread-Anwendungen für die Speicherung von privaten Thread-spezifischen Daten innerhalb eines PDFlib-Objekts verwendet werden. Es ist besonders nützlich bei der Thread-spezifischen Verarbeitung von Exceptions.

Gültigkeit beliebig

Bindungen Nur verfügbar in den C- und C++-Sprachbindungen.

2.2 Unicode-Konvertierung

C++ ***string convert_to_unicode(string inputformat, string input, string optlist)***

Java ***string convert_to_unicode(string inputformat, byte[ ] input, string optlist)***

Perl PHP ***string convert_to_unicode(string inputformat, string input, string optlist)***

C ***const char *PDF_convert_to_unicode(PDF *p,***
const char *inputformat, const char *input, int inputlen, int *outputlen, const char *optlist)

Konvertiert einen String mit beliebigem Encoding in einen Unicode-String mit wählbarem Format.

inputformat Unicode-Textformat oder Name des Encodings des zu konvertierenden Strings:

- Unicode-Textformate: *utf8, ebcdicutf8, utf16, utf16le, utf16be, utf32*
- Nur bei Angabe der Option *font: builtin, glyphid*
- Alle intern bekannten 8-Bit-Encodings, auf dem Host-System vorhandene Encodings sowie die CJK-Encodings *cp932, cp936, cp949, cp950*
- Das Schlüsselwort *auto* führt zu folgendem Verhalten: wenn der Eingabe-String einen UTF-8-BOM oder UTF-16-BOM enthält, wird er zur Bestimmung des korrekten Formats verwendet, ansonsten wird die aktuelle Codepage des Systems herangezogen.

input Nach Unicode zu konvertierender String (bei COM: Variant; bei REALbasic/Xojo: MemoryBlock).

inputlen (Nur C-Sprachbindung) Länge des zu konvertierenden Strings in Bytes. Bei *inputlen=0* muss ein null-terminierter String angegeben werden.

outputlen (Nur C-Sprachbindung) C-Zeiger auf einen Speicherplatz, an dem die Länge des zurückgegebenen Strings in Bytes abgelegt wird.

optlist Optionsliste, in der die Optionen für die Interpretation des zu konvertierenden Strings und für seine Konvertierung nach Unicode festgelegt werden:

- Optionen für Textfilter gemäß Tabelle 4.6: *charref, escapesequence*
- Optionen für Unicode-Konvertierung gemäß Tabelle 2.2: *bom, errorpolicy, font, inflate, outputformat*

Rückgabe Ein aus dem zu konvertierenden String gemäß den angegebenen Parametern und Optionen erzeugter Unicode-String. Wenn der zu konvertierende String nicht mit dem angegebenen Eingabeformat übereinstimmt (z.B. bei einem ungültigen UTF-8-String) wird bei *errorpolicy=return* ein leerer String zurückgegeben, und bei *errorpolicy=exception* wird eine Exception ausgelöst.

Details Diese Funktion ist für die allgemeine Unicode-Konvertierung von Strings nützlich, besonders wenn Sie in einer Umgebung ohne geeignete Unicode-Konverter arbeiten.

Gültigkeit beliebig

Bindungen C-Sprachbindung: Bis zu 10 zurückgegebene String-Einträge werden in einem Ring-Puffer abgelegt. Bei der Konvertierung von mehr als 10 Strings werden die Puffer wiederverwendet, dass heißt, wenn parallel auf mehr als 10 Strings zugegriffen werden soll, müssen diese kopiert werden. Zum Beispiel können bis zu 10 Aufrufe in dieser Funktion

als Parameter für eine *printf( )*-Anweisung verwendet werden, da die Rückgabe-Strings auf jeden Fall unabhängig voneinander sind, sofern nicht mehr als 10 Strings gleichzeitig verwendet werden.

Nicht Unicode-fähige Sprachbindungen: Diese Funktion kann zur Erstellung von Name-Strings und Optionslisten in nicht Unicode-fähigen Sprachbindungen verwendet werden. Damit wird der erforderliche BOM mit den Optionen *bom=optimize* und *outputformat=utf8* erzeugt.
C++-Sprachbindung: Die Parameter *inputformat* und *optlist* müssen als *wstrings* übergeben werden, *input* und die zurückgegebenen Daten müssen jedoch vom Typ *string* sein.

Tabelle 2.2 Optionen für PDF_convert_to_unicode( )

Option	Beschreibung
bom	*(Schlüsselwort; wird bei* `outputformat=utf32` *nicht ausgewertet) Richtlinie zum Hinzufügen eines Byte Order Mark (BOM) zum Ausgabe-String. Unterstützte Schlüsselwörter (Standardwert:* `none`*):* **add** *Hinzufügen eines BOM.* **keep** *Hinzufügen eines BOM, wenn der Eingabe-String einen BOM hat.* **none** *Kein Hinzufügen eines BOM.* **optimize** *Hinzufügen eines BOM, außer wenn* `outputformat=utf8` *oder* `ebcdicutf8` *und der Ausgabe-String nur Zeichen kleiner U+007F enthält.*
errorpolicy	*(Schlüsselwort) Verhalten bei einem Konvertierungsfehler (Standardwert: der Wert der globalen Option* `errorpolicy`*, siehe Tabelle 2.1):* **return** *Das Ersatzzeichen wird verwendet, wenn eine Character-Referenz nicht aufgelöst werden kann oder ein Code oder eine Glyph-ID nicht im angegebenen Font existiert. Bei einem Konvertierungsfehler wird ein leerer String ausgegeben.* **exception** *Bei einem Konvertierungsfehler wird eine Exception ausgelöst.*
font	*(Font-Handle; erforderlich bei* `inputformat=builtin` *und* `glyphid`*) Fontspezifische Konvertierung gemäß des angegebenen Fonts.*
inflate	*(Boolean; nur bei* `inputformat=utf8`*) Bei* `true` *löst ein ungültiger UTF-8-Eingabe-String keine Exception aus, vielmehr wird ein String mit ungewöhnlich vielen Bytes im angegebenen Ausgabeformat erzeugt. Dies kann bei der Fehlersuche hilfreich sein. Dieser String besteht aus einem Unicode-Zeichen, die der ASCII-Interpretation des Bytes im Eingabe-String entsprechen. Standardwert:* `falset`
outputformat	*(Schlüsselwort) Unicode-Textformat des konvertierten Strings:* `utf8`*,* `ebcdicutf8`*,* `utf16`*,* `utf16le`*,* `utf16be`*,* `utf32`*. Ein leerer String ist äquivalent zu* `utf16`*. Standardwert:* `utf16` *Unicode-fähige Sprachbindungen: das Ausgabeformat wird immer auf* `utf16` *gesetzt.* *C++-Sprachbindung: nur die folgenden Ausgabeformate sind erlaubt:* `ebcdicutf8`*,* `utf8`*,* `utf16`*,* `utf32`**.**

2.3 Globale Optionen

PDFlib bietet verschiedene globale Optionen für die Steuerung der Bibliothek und der Erstellung der PDF-Ausgabe. Diese Optionen behalten ihre Einstellungen über die Lebensdauer des PDFlib-Objekts oder bis sie explizit vom Client geändert werden.

C++ Java C# ***void set_option(String optlist)***

Perl PHP ***set_option(string optlist)***

C ***void PDF_set_option(PDF *p, const char *optlist)***

Setzt eine oder mehrere globale Optionen.

optlist Optionsliste mit globalen Optionen gemäß Tabelle 2.3. Folgende Optionen können verwendet werden:

- Optionen für Ressourcenverarbeitung und -kategorien gemäß Tabelle 2.3:
  Encoding, enumeratefonts, FontAFM, FontnameAlias, FontOutline, FontPFM, HostFont, ICCProfile, resourcefile, saveresources, searchpath
- Optionen zur Dateiverarbeitung und Lizenzierung gemäß Tabelle 2.3:
  avoiddemostamp, filenamehandling, license, licensefile
- Optionen für Textfilter gemäß Tabelle 2.3:
  charref,escapesequence, glyphcheck, stringformat, textformat
- Optionen für interaktive Elemente gemäß Tabelle 2.3:
  hypertextencoding, hypertextformat, usehypertextencoding, usercoordinates
- Andere Optionen gemäß Tabelle 2.3:
  asciifile, autospace,compress, kerning, logging, shutdownstrategy, usehostfonts, userlog
- Optionen zur Fehlerbehandlung gemäß Tabelle 2.1: *errorpolicy*
- Optionen für die Farbverarbeitung gemäß Tabelle 8.1:
  iccprofilecmyk, iccprofilegray, iccprofilergb, preserveoldpantonenames, spotcolorlookup

Details Außer bei den Optionen für Ressourcenkategorien überschreiben neue Werte die zuvor gesetzten Optionswerte.

Die folgenden Optionen liefern Standardwerte für die gleichnamigen Textoptionen (siehe Tabelle 4.6 und Tabelle 4.7):
charref, escapesequence, glyphcheck, kerning, textformat

Gleichzeitig ändern diese Optionen die gleichnamigen Optionen im aktuellen Textstatus. Um unerwünschte Nebeneffekte zu vermeiden, empfehlen wir deshalb, Optionen für Content-Strings nur in *PDF_set_text_option( )* zu setzen.

Gültigkeit beliebig, aber bei einigen Optionen sind die Gültigkeitsbereiche eingeschränkt

Tabelle 2.3 Globale Optionen für PDF_set_option( )

Option	Beschreibung
asciifile	*(Boolean; nur unterstützt bei i5/iSeries und zSeries). Es werden Textdateien (PFA, AFM, UPR, Encodings) in ASCII-Encoding erwartet. Standardwert:* `true` *bei i5/iSeries;* `false` *bei zSeries*
autospace	*Bei* `true` *und wenn der aktuelle Font eine Glyphe für U+0020 enthält, fügt PDFlib am Ende eines ausgegebenen Texts automatisch ein Leerzeichen an. Dies kann bei der Generierung von Tagged PDF nützlich sein. Beachten Sie, dass sich durch das Anfügen eines Leerzeichens die aktuelle Textposition ändert. Standardwert:* `false`

Tabelle 2.3 Globale Optionen für PDF_set_option( )

Option	Beschreibung
avoiddemostamp	*(Boolean) Bei* true *wird eine Exception ausgelöst, wenn kein gültiger Lizenzschlüssel gefunden wurde; bei* false *wird auf jeder Seite ein Demostempel aufgedruckt. Diese Option muss vor dem ersten Aufruf von PDF_begin_document( ) gesetzt werden. Standardwert:* false
charref	*(Boolean) Bei* true *werden numerische Referenzen, Character-Referenzen und Glyphnamen-Referenzen für alle Content-, Name- und Hypertext-Strings ersetzt. Um das Ersetzen von Character-Referenzen an unerwünschten Stellen zu vermeiden (z.B. in Dateinamen) empfehlen wir, diese Option für Content-Strings nur in PDF_set_text_option( ) zu setzen; siehe PDFlib-Tutorial für weitere Informationen. Standardwert:* false
compress	*(Integer) Kompressionsstufe von 0=keine Kompression, 1=maximale Geschwindigkeit, etc. bis zu 9=maximale Kompression. Diese Option hat keinen Einfluss auf Rasterbilddaten, die im Pass-Through-Modus bearbeitet werden. Standardwert: 6. Gültigkeitsbereich: beliebig außer* object
Encoding	*(Liste mit Paaren von Name-Strings) Liste von Schlüssel-/Wertpaaren für eine Ressourcendefinition durch Leer- oder Gleichheitszeichen '=' getrennt (siehe PDFlib-Tutorial für weitere Informationen). Mehrfache Aufrufe fügen der internen Liste neue Einträge hinzu.*
enumeratefonts	*(Boolean) Bei* true *durchsucht PDFlib alle Verzeichnisse, die über die Ressource* SearchPath *angesteuert werden können, nach Fontdateien. Dies kann bei einer großen Anzahl von Fonts erhebliche Zeit in Anspruch nehmen und sollte daher mit Vorsicht verwendet werden. Die erzeugte Ressourcenliste kann in einer Datei mit der Option* saveresources *gespeichert werden. Wir empfehlen, die Ressourcenliste nur zu erzeugen und zu speichern, wenn sich die Anzahl der zugänglichen Fonts geändert hat, und nicht bei jedem generierten Dokument oder für jedes PDFlib-Objekt.* *Für jede gültige Fontdatei ermittelt PDFlib die Namen für* font-family, font-weight *und* font-style *und synthetisiert daraus einen API-Fontnamen gemäß folgendem Schema:* <font-family>[,<font-weight>][,font-style] *PDFlib erzeugt eine* FontOutline*-Ressource im Format* <fontname>=<pathname>*, die den künstlichen Fontnamen mit dem vollständigen Pfadnamen des Fonts verknüpft. Für PostScript-Type-1-Fonts wird ebenfalls die entsprechende Ressource* FontAFM *oder* FontPFM *erzeugt. Zusätzlich zum API-Fontnamen erzeugt PDFlib eine Ressource* FontnameAlias *mit dem PostScript-Namen des Fonts, wenn er vom künstlichen Namen abweicht:* <PostScript fontname>=<artificial fontname> *Als Ergebnis kann der Font entweder über den künstlichen Namen oder über seinen PostScript-Namen geladen werden. Standardwert:* false
escapesequence	*(Boolean) Bei* true *werden Escape-Sequenzen in allen Content-, Name- und Hypertext-Strings ersetzt. Um das Ersetzen von Escape-Sequenzen an unerwünschten Stellen zu vermeiden (z.B. in Dateinamen) empfehlen wir, diese Option für Content-Strings nur in PDF_set_text_option( ) zu setzen; Standardwert:* false

Tabelle 2.3 Globale Optionen für PDF_set_option( )

Option	Beschreibung
filename-handling	*(Schlüsselwort) Encoding für Dateinamen. Alle als Funktionsparameter übergebenen Dateinamen ohne einen UTF-8-BOM werden in nicht Unicode-fähigen Sprachbindungen gemäß dieser Option interpretiert, um für das Dateisystem ungültige Zeichen zu vermeiden und um eine Unicode-Version des Dateinamens zu erstellen. Ein Fehler tritt auf, wenn der Dateiname Zeichen außerhalb des angegebenen Encodings enthält. Standardwert:* `unicode` *bei Windows und OS X,* `auto` *bei i5/iSeries, sonst* `honorlang`: **ascii** *7-Bit-ASCII* **basicebcdic** *Grundlegendes EBCDIC gemäß Codepage 1047, aber nur Unicode-Werte <= U+007E* **basicebcdic_37** *Grundlegendes EBCDIC gemäß Codepage 0037, aber nur Unicode-Werte <= U+007E* **honorlang** *(Nicht bei i5/iSeries) Die Umgebungsvariablen LC_ALL, LC_CTYPE und LANG werden interpretiert. Der in* `LANG` *angegene Codeset, sofern vorhanden, wird auf Dateinamen angewendet.* **legacy** *Verwendung des Encodings* `host` *zur Interpretation des Dateinamens* **unicode** *Unicode-Encoding im (EBCDIC-)UTF-8-Format* **alle gültigen Encoding-Namen** *Jedes von PDFlib erkannte Encoding außer CMaps,* `glyphid` *und* `builtin` *(siehe Tabelle 4.2)*
FontAFM	*(Liste mit Paaren von Name-Strings) Liste von Schlüssel-/Wertpaaren für eine Ressourcendefinition durch Leer- oder Gleichheitszeichen '=' getrennt (siehe PDFlib-Tutorial für weitere Informationen). Mehrfache Aufrufe fügen der internen Liste neue Einträge hinzu.*
Fontname-Alias	*(Liste mit Paaren von Name-Strings) Liste von Schlüssel-/Wertpaaren für eine Ressourcendefinition durch Leer- oder Gleichheitszeichen '=' getrennt (siehe PDFlib-Tutorial für weitere Informationen). Mehrfache Aufrufe fügen der internen Liste neue Einträge hinzu.*
FontOutline	*(Liste mit Paaren von Name-Strings) Liste von Schlüssel-/Wertpaaren für eine Ressourcendefinition durch Leer- oder Gleichheitszeichen '=' getrennt (siehe PDFlib-Tutorial für weitere Informationen). Mehrfache Aufrufe fügen der internen Liste neue Einträge hinzu.*
FontPFM	*(Liste mit Paaren von Name-Strings) Liste von Schlüssel-/Wertpaaren für eine Ressourcendefinition durch Leer- oder Gleichheitszeichen '=' getrennt (siehe PDFlib-Tutorial für weitere Informationen). Mehrfache Aufrufe fügen der internen Liste neue Einträge hinzu.*
glyphcheck	*(Schlüsselwort) Siehe Tabelle 4.6 für weitere Informationen. Wir empfehlen, diese Option für Content-Strings nur in PDF_set_text_option( ) zu setzen; siehe PDFlib-Tutorial für weitere Informationen. Standardwert:* `replace`
HostFont	*(Liste mit Paaren von Name-Strings) Liste von Schlüssel-/Wertpaaren für eine Ressourcendefinition durch Leer- oder Gleichheitszeichen '=' getrennt (siehe PDFlib-Tutorial für weitere Informationen). Mehrfache Aufrufe fügen der internen Liste neue Einträge hinzu.*
hypertext-encoding	*(String; nur für nicht Unicode-fähige Sprachbindungen) Bestimmt das Encoding für Hypertext-Strings. Ein leerer String ist äquivalent zu* `unicode`*. Standardwert:* `auto`
hypertext-format	*(Schlüsselwort; nur für nicht Unicode-fähige Sprachbindungen) Bestimmt das Format für Hypertext-Strings. Unterstützte Schlüsselwörter sind* `bytes`, `utf8`, `ebcdicutf8`, `utf16`, `utf16le`, `utf16be` *und* `auto`*. Standardwert:* `auto`
ICCProfile	*(Liste mit Paaren von Name-Strings) Liste von Schlüssel-/Wertpaaren für eine Ressourcendefinition durch Leer- oder Gleichheitszeichen '=' getrennt (siehe PDFlib-Tutorial für weitere Informationen). Mehrfache Aufrufe fügen der internen Liste neue Einträge hinzu.*
iccprofilecmyk **iccprofilegray** **iccprofilergb**	*(ICC-Profil-Handle) ICC-Profil, das einen Farbraum vom Typ CMYK, Gray oder RGB für die Verwendung mit den Farboptionen* `iccbasedcmyk`/`gray`/`rgb` *festlegt. Standardwert: kein ICC-Farbraum*
kerning	*(Boolean) Bei* `true` *wird das Unterschneiden bei Fonts aktiviert, die mit der Option* `readkerning` *geöffnet wurden; anderenfalls wird das Unterschneiden ausgeschaltet. Standardwert:* `true`

Tabelle 2.3 Globale Optionen für PDF_set_option( )

Option	Beschreibung
license	*(String) Lizenzschlüssel für PDFlib, PDFlib+PDI oder PPS (siehe PDFlib-Tutorial für weitere Informationen). Der Schlüssel kann vor dem ersten Aufruf von PDF_begin_document( ) gesetzt werden. Mit der Option* `avoiddemostamp` *lässt sich sicherstellen, dass ein fehlender Lizenzschlüssel nicht versehentlich zu einem Demostempel führt.*
licensefile	*(Name-String) Setzt den Namen der Datei mit dem Lizenzschlüssel (siehe PDFlib-Tutorial für weitere Informationen). Dieser kann nur einmal vor dem ersten Aufruf von PDF_begin_document( ) festgelegt werden.*
logging	*(Optionsliste) Logging-Optionen gemäß Tabelle 1.4*
maxfile-handles	*(Nicht unterstützt; nur für Windows implementiert) Neue maximale Anzahl für die Anzahl der gleichzeitig geöffneten Dateien (in der C-Laufzeitumgebung). Die Anzahl muss größer oder gleich 20 und kleiner oder gleich 2048 sein. Wird der neue Wert von der C-Laufzeitumgebung nicht akzeptiert, so wird eine Exception ausgelöst. Gültigkeitsbereich:* `object`
resourcefile	*(Name-String) Relativer oder absoluter Name der von PDFlib verwendeten UPR-Ressourcendatei. Diese wird unmittelbar vor dem ersten Zugriff auf eine Ressource geladen. Bereits vorhandene Ressourcen bleiben erhalten. Ihre Werte werden gegebenenfalls von den neuen Werten überschrieben.*
saveresources	*(Optionsliste) Speichert die aktuelle Ressourcenliste in einer Datei. Die folgende Option wird unterstützt:* **filename** *Name der Ressourcendatei, in der die Ressourcenliste gespeichert wird. Standardwert:* `pdflib.upr`
searchpath	*(Liste von Name-Strings) Ein oder mehrere relative oder absolute Pfadnamen eines Verzeichnisses mit den einzulesenden Dateien.* `searchpath` *kann mehrfach gesetzt werden; die Einträge werden in einer Liste zusammengefasst, die von hinten nach vorne abgearbeitet wird (siehe PDFlib-Tutorial für weitere Informationen). Wir empfehlen, selbst bei einem einzelnen Eintrag doppelte geschweifte Klammern zu verwenden, um Probleme bei Verzeichnissen mit Leerzeichen im Namen zu vermeiden. Eine leere String-Liste (d.h.* {{}} *) entfernt alle bestehenden Searchpath-Einträge einschließlich der Standardeinträge. Bei Windows kann der Searchpath auch über die Registry gesetzt werden. Standardwert: plattformabhängig, siehe PDFlib-Tutorial*
shutdown-strategy	*(Integer) Methode für die Freigabe von globalen Ressourcen, die einmal für alle PDFlib-Objekten alloziert werden. Jede globale Ressource wird initialisiert, sobald sie zum ersten Mal benötigt wird. Diese Option muss in einem PDFlib-Prozess für alle Objekte auf denselben Wert eingestellt werden, da sonst das Verhalten undefiniert ist (Standardwert: 0):* **0** *Ein Verweiszähler verfolgt, wie viele PDFlib-Objekte die globalen Ressourcen nutzen. Wenn das letzte PDFlib-Objekt gelöscht ist, werden die Ressourcen freigegeben.* *1* *Die Ressourcen werden bis zum Ende des Prozesses gehalten. Damit lässt sich die Leistung leicht steigern, er wird aber auch mehr Speicher benötigt, nachdem das letzte PDFlib-Objekt gelöscht wurde.*

Tabelle 2.3 Globale Optionen für PDF_set_option( )

Option	Beschreibung
stringformat	*(Schlüsselwort; nur für nicht Unicode-fähige Sprachbindungen) Setzt das Format für die Interpretation aller Strings im API, das heißt, Name-Strings, Content-Strings, Hypertext-Strings und Optionslisten. Unterstützte Schlüsselwörter (Standardwert:* `legacy`*):* ***ebcdicutf8*** *(Nur für i5/iSeries und zSeries) Alle Strings und Optionslisten werden im Format EBCDIC-UTF-8 mit oder ohne BOM erwartet.* ***legacy*** *Name-Strings, Content-Strings, Hypertext-Strings und Optionslisten werden gemäß der Optionen* `textformat`, `hypertextformat` *und* `hypertextencoding` *behandelt.* ***utf8*** *(Nicht für i5/iSeries und zSeries) Alle Strings und Optionslisten werden im UTF-8-Format mit oder ohne BOM erwartet. Die Optionen* `textformat`, `hypertextformat` *und* `hypertextencoding` *sind unzulässig. Die Textflow-Option* `fixedtextformat` *wird immer auf* `true` *gesetzt. Bestehende CJK-CMaps können nicht zum Laden von Fonts verwendet werden. In der C-Sprachbindung werden Name-Strings als Funktionsparamter immer noch als UTF-16-Strings interpretiert, sofern die Option* `length` *mit einem Wert größer 0 übergeben wird. Verwenden Sie PDF_convert_to_unicode( ) für die Konvertierung von 8-Bit-Encodings nach UTF-8.*
user-coordinates	*(Boolean) Bei* `false` *werden die Koordinaten für Hypertext-Rechtecke im Standardkoordinatensystem erwartet; anderenfalls wird das aktuelle Benutzerkoordinatensystem verwendet. Standardwert:* `false`
userlog	*String, der in die Logdatei geschrieben wird*
usehostfonts	*(Boolean) Bei* `true` `werden` *Host-Fonts mit in die Fontsuche einbezogen. Standardwert:* `true`
usehypertext-encoding	*(Boolean; nur für nicht Unicode-fähige Sprachbindungen) Bei* `true` *wird das in der Option* `hypertext-encoding` *festgelegte Encoding auch für Name-Strings verwendet. Bei* `false` *ist das Encoding für Name-Strings ohne UTF-8-BOM gleich* `host`*. Standardwert:* `false`
textformat	*(Schlüsselwort; nur für nicht Unicode-fähige Sprachbindungen) Setzt das Format für die Interpretation von Content-Strings. Unterstützte Schlüsselwörter:* `bytes`, `utf8`, `ebcdicutf8` *(nur für i5/iSeries und zSeries),* `utf16`, `utf16le`, `utf16be` *und* `auto`*. Standardwert:* `auto`

C++ Java C# ***double get_option(String keyword, String optlist)***

Perl PHP ***float get_option(string keyword, string optlist)***

C ***double PDF_get_option(PDF *p, const char *keyword, const char *optlist)***

Ermittelt eine Option oder einen anderen Wert.

keyword Schlüsselwort, das die zu ermittelnde Option festlegt. Die unten angegebenen Schlüsselwörter werden unterstützt; siehe unter *PDF_set_option( )*, *PDF_set_text_option( )* und *PDF_set_graphics_option( )* für ihre jeweilige Bedeutung. Für Schlüsselwörter, für die es keine entsprechende Option gibt, siehe Tabelle 2.4:

- Schlüsselwörter für den String-Index des *n*-ten Eintrags der angegebenen Ressource, wobei *n* der Option *resourcenumber* entspricht:
  Encoding, FontAFM, FontnameAlias, FontOutline, FontPFM, HostFont, ICCProfile, searchpath
- Schlüsselwörter für Boolesche Optionswerte geben 1 für *true* zurück oder 0 für *false:*
  asciifile, autospace,avoiddemostamp, charref, decorationabove, escapesequence, glyphcheck, fakebold, kerning, overline, pdi, preserveoldpantonenames, spotcolorlookup, strikeout, tagged, topdown, underline, usercoordinates, usehostfonts, usehypertextencoding
- Schlüsselwörter für Optionswerte vom Typ *integer* und *float*:
  charspacing, compress, ctm_a, ctm_b, ctm_c, ctm_d, ctm_d, ctm_f, currentx, currenty, icccomponents, flatness, font, fontsize, horizscaling, iccprofilecmyk, iccprofilegray, iccprofilergb,

italicangle, leading, linecap, linejoin, linewidth, major, minor, miterlimit, pageheight, pagewidth, revision, scope, textrendering, textrise, textx, texty, underlineposition, underlinewidth, wordspacing

- Schlüsselwörter, die einen String-Index für einen Optionswert oder -1 zurückgeben, wenn der String-Wert nicht vorhanden ist:
  cliprule, errorpolicy, filenamehandling, fillrule, glyphcheck, hypertextencoding, hypertextformat, resourcefile, scope, textformat
- Schlüsselwörter für die Abfrage des aktuellen Strukturelements (nur im Modus Tagged PDF):
  activeitemid, activeitemindex, activeitemisinline, activeitemkidcount, activeitemname, activeitemstandardname

Tabelle 2.4 Zusätzliche Schlüsselwörter für PDF_get_option( )

Schlüsselwort	Beschreibung
activeitemid	*(Integer) ID des aktuell aktiven Strukturelements. Das Schlüsselwort kann mit PDF_activate_item() oder der Unteroption* `parent` *von PDF_begin_item() und der Option* `tag` *verwendet werden. -1 wird zurückgegeben, wenn noch kein Stammelement vorhanden ist. Gültigkeitsbereich:* `document`, `page`
activeitemindex	*(Integer) Null-basierter Index des aktuell aktiven Strukturelements innerhalb seines übergeordneten Elements. Das Schlüsselwort kann mit der Tag-Option* `index` *verwendet werden. -1 wird zurückgegeben, wenn es sich bei dem aktuellen Element um ein Pseudo-Element oder das Stammelement handelt oder wenn noch kein Stammelement vorhanden ist. Gültigkeitsbereich:* `document`, `page`
activeitemisinline	*(Integer) 1, falls das aktuell aktive Strukturelement ein Inline-Element ist, anderenfalls 0. Gültigkeitsbereich:* `document`, `page`
activeitemkidcount	*(Integer) Anzahl der bis zu diesem Punkt erzeugten untergeordneten Elemente des aktuell aktiven Strukturelements (ohne Pseudo-Elemente). -1 wird zurückgegeben, wenn noch kein Stammelement vorhanden ist. Gültigkeitsbereich:* `document`, `page`
activeitemname	*Gibt den String-Index für den Typnamen des aktuell aktiven Strukturelements oder Pseudo-Elements zurück oder -1, wenn noch kein Stammelement vorhanden ist. Gültigkeitsbereich:* `document`, `page`
activeitemstandardname	*Gibt den String-Index für den Standardnamen des Elementtyps zurück, für den es eine Rollenzuordnung auf das aktuell aktive Element gibt, oder -1, wenn noch kein Stammelement vorhanden ist oder das aktuelle Element ein benutzerdefiniertes Element ist, für das keine Rollenzuordnung vorhanden ist. Wenn keine Rollenzuordnung aktiv ist, wird der ursprüngliche Typname zurückgegeben. Gültigkeitsbereich:* `document`, `page`
ctm_a ***ctm_b*** ***ctm_c*** ***ctm_d*** ***ctm_e*** ***ctm_f***	*(Float) Komponenten der aktuellen Transformationsmatrix (CTM) für Vektorgrafik. Gültigkeitsbereich:* `page`, `pattern`, `template`, `glyph`, `path`
currentx ***currenty***	*(Float) Gibt die x- bzw. y-Koordinate (in Einheiten des aktuellen Koordinatensystems) des aktuellen Punkts zurück. Gültigkeitsbereich:* `page`, `pattern`, `template`, `glyph`, `path`
icccomponents	*(Integer) Gibt die Anzahl der Farbkomponenten des ICC-Profils zurück, das durch das von der Option* `iccprofile` *übergebene Handle bezeichnet wird.*
major ***minor*** ***revision***	*(Integer) Gibt die Major-Versionsnummer, die Minor-Versionsnummer bzw. die Revisionsnummer von PDFlib zurück. Gültigkeitsbereich: beliebig,* `null`[1]
pageheight ***pagewidth***	*(Float) Ermittelt die Größe der aktuellen Seite (Größe der MediaBox), des Templates oder der Glyphe. Gültigkeitsbereich: beliebig außer* `object`

Tabelle 2.4 Zusätzliche Schlüsselwörter für PDF_get_option( )

Schlüsselwort	Beschreibung
pdi	*(Integer) Gibt 1 zurück, falls der PDI-Sourcecode beim Erzeugen der zugrunde liegenden Bibliothek integriert wurde. Dies ist für alle von PDFlib GmbH ausgelieferten kombinierten Binärpakete erfüllt, die PDFlib, PDFlib+PDI und PPS enthalten (unabhängig vom Lizenzschlüssel). Anderenfalls wird 0 zurückgegeben. Gültigkeitsbereich: beliebig,* `null`[1]
scope	*(Integer) Gibt den String-Index für den Namen des aktuellen Gültigkeitsbereichs aus (siehe Tabelle 1.3)*
textx ***texty***	*(Float) Gibt die* x- *bzw.* y*-Koordinate der aktuellen Textposition aus. Gültigkeitsbereich:* `page`, `pattern`, `template`, `glyph`

*1. C-Sprachbindung: Kann mit NULL oder 0 als PDF * Argument aufgerufen werden*

optlist Optionsliste mit einer Option gemäß Tabelle 2.5.

Rückgabe Der Wert der Option, die durch *keyword* ausgewählt wurde. Wenn kein Wert für das erforderliche Schlüsselwort vorhanden ist, gibt die Funktion -1 zurück. Wenn das erforderliche Schlüsselwort Text produziert, wird ein String-Index zurückgegeben und der entsprechende String muss mit *PDF_get_string( )* ermittelt werden.

Gültigkeit beliebig, aber bei einigen Schlüsselwörtern sind die Gültigkeitsbereiche eingeschränkt.

Tabelle 2.5 Optionen für PDF_get_option()

Option	Beschreibung
textstate	*(Boolean) Bei* `true` *wird der Wert der folgenden Optionen aus dem Textstatus ermittelt, anderenfalls aus den globalen Optionen (Standardwert:* `false`*):* `charref, escapesequence, predefcmap, kerning, textformat`
iccprofile	*(ICC-Profil-Handle) ICC-Profil für die Verwendung mit dem Schlüsselwort* `icccomponents`
resource-number	*(Integer) Nummer der zu ermittelnden Ressource; Ressourcen werden bei 1 beginnend durchnummeriert Standardwert: 1*

C++ Java C# **String get_string(int idx, String optlist)**

Perl PHP **string get_string(int idx, string optlist)**

C **const char *PDF_get_string(PDF *p, int idx, const char *optlist)**

Ermittelt einen String-Wert.

idx String-Index, der von einer der Funktionen *PDF_get_option()* oder *PDF_info_*()* zurückgegeben wird, oder -1, falls eine Option angegeben wurde.

optlist Optionsliste mit Optionen gemäß Tabelle 2.6.

Rückgabe Der Wert eines mit *idx* oder *optlist* ausgewählten Strings.

Gültigkeit Abhängig von der erforderlichen Option.

Bindungen C-Sprachbindung: Der zurückgegebene String ist bis zum nächsten Aufruf einer API-Funktion gültig.

Tabelle 2.6 Option für PDF_get_string()

Schlüsselwort	Beschreibung
version	*(Boolean) Gibt die vollständige PDFlib-Version als String im Format* `<major>.<minor>.<revision>` *aus, an den gegebenenfalls noch weitere Kennungen angefügt sind* (`beta`, `rc`, *etc.). Gültigkeitsbereich: beliebig, null*[1]

*1. C-Sprachbindung: Kann mit NULL oder 0 als PDF * Argument aufgerufen werden*

C++ Java C# **void set_parameter(String key, String value)**

Perl PHP **set_parameter(string key, string value)**

C **void PDF_set_parameter(PDF *p, const char *key, const char *value)**

Veraltet, verwenden Sie stattdessen *PDF_set_option( )*, *PDF_set_text_option( )* und *PDF_set_graphics_option( )*.

C++ Java C# **void set_value(String key, double value)**

Perl PHP **set_value(string key, float value)**

C **void PDF_set_value(PDF *p, const char *key, double value)**

Veraltet, verwenden Sie stattdessen *PDF_set_option( )*, *PDF_set_text_option( )* und *PDF_set_graphics_option( )*.

C++ Java C# **String get_parameter(String key, double modifier)**

Perl PHP **string get_parameter(string key, float modifier)**

C **const char * PDF_get_parameter(PDF *p, const char *key, double modifier)**

Veraltet, verwenden Sie stattdessen *PDF_get_option( )* und *PDF_get_string( )*.

C++ Java C# **double get_value(String key, double modifier)**

Perl PHP **float get_value(string key, float modifier)**

C **double PDF_get_value(PDF *p, const char *key, double modifier)**

Veraltet, verwenden Sie stattdessen *PDF_get_option( )*.

2.4 Erstellen und Löschen von PDFlib-Objekten

C ***PDF *PDF_new(void)***

Erzeugt ein neues PDFlib-Objekt.

Details Diese Funktion erzeugt ein neues PDFlib-Objekt, wobei es die PDFlib-internen Standardroutinen zur Fehlerbehandlung und Speicherallozierung verwendet.

Rückgabe Handle auf ein PDFlib-Objekt, das in nachfolgenden PDFlib-Aufrufen verwendet werden kann. Schlägt diese Funktion aufgrund von mangelndem Speicher fehl, wird NULL zurückgegeben oder eine Exception ausgelöst.

Gültigkeit *null;* mit dieser Funktion beginnt der Gültigkeitsbereich *object*, der immer mit einem entsprechenden Aufruf von *PDF_delete( )* beendet werden muss.

Bindungen C-Sprachbindung: Wenn Sie die PDFlib-DLL dynamisch zur Laufzeit laden möchten, verwenden Sie *PDF_new_dl( )*. *PDF_new_dl( )* gibt einen Zeiger auf eine *PDFlib_api*-Struktur zurück, die Zeiger auf alle PDFlib-API-Funktionen enthält. Wenn die DLL nicht geladen werden kann oder Major- bzw. Minor-Versionsnummer nicht passen, wird NULL zurückgegeben.
Andere Sprachbindungen: Diese Funktion ist nicht verfügbar, da sie im PDFlib-Konstruktor verborgen ist.

C ***PDF *PDF_new2(void (*errorhandler)(PDF *p, int errortype, const char *msg),***
void* (*allocproc)(PDF *p, size_t size, const char *caller),
void* (*reallocproc)(PDF *p, void *mem, size_t size, const char *caller),
void (*freeproc)(PDF *p, void *mem),
void *opaque)

Erzeugt ein neues PDFlib-Objekt mit benutzerdefinierten Routinen zur Fehlerbehandlung und Speicherverwaltung.

errorhandler Zeiger auf eine benutzerdefinierte Fehlerbehandlungsfunktion. Die Fehlerbehandlungsfunktion wird in *PDF_TRY/PDF_CATCH*-Blöcken ignoriert.

allocproc Zeiger auf eine benutzerdefinierte Speicherallozierungsfunktion.

reallocproc Zeiger auf eine benutzerdefinierte Speicherreallozierungsfunktion.

freeproc Zeiger auf eine benutzerdefinierte Speicherfreigabefunktion.

opaque Zeiger auf Benutzerdaten, der mit *PDF_get_opaque( )* abgefragt werden kann.

Rückgabe Handle auf ein PDFlib-Objekt, das dann in nachfolgenden PDFlib-Aufrufen verwendet werden kann. Schlägt diese Funktion aufgrund von mangelndem Speicher fehl, wird NULL (in C) zurückgegeben oder eine Exception (in C++) ausgelöst.

Details Diese Funktion erzeugt ein neues PDFlib-Objekt mit benutzerdefinierten Funktionen zur Fehlerbehandlung und Speicherverwaltung. Im Gegensatz zu *PDF_new( )* kann der Aufrufer bei Bedarf eigene Prozeduren zur Fehlerbehandlung und Speicherverwaltung übergeben. Die Funktionszeiger für den Error-Handler oder die Speicherverwaltungsroutinen können NULL sein. Es werden dann die PDFlib-Standardroutinen verwendet.

Es müssen entweder alle drei oder es darf keine Speicherverwaltungsroutine übergeben werden.

Gültigkeit *null;* mit dieser Funktion beginnt der Gültigkeitsbereich *object*, der immer mit einem entsprechenden Aufruf von *PDF_delete( )* beendet werden muss.

Bindungen C++: Diese Funktion ist indirekt über den PDFlib-Konstruktor verfügbar. Fehlt ein Funktionsargument, wird als Standardwert NULL übergeben. Alle übergebenen Funktionen müssen C-Funktionen sein; C++-Methoden sind nicht erlaubt.

C *void PDF_delete(PDF *p)*

Löscht ein PDF-Objekt und gibt alle zugehörigen internen Ressourcen frei.

Details Diese Funktion löscht ein PDFlib-Objekt und gibt alle zum Dokument gehörenden PDFlib-internen Ressourcen frei. Die Funktion darf für ein PDF-Objekt nur einmal aufgerufen werden. *PDF_delete( )* sollte außerdem zur Bereinigung nach einer Exception aufgerufen werden. *PDF_delete( )* selbst löst unter keinen Umständen eine Exception aus. Bei der Generierung mehrerer PDF-Dokumente muss *PDF_delete( )* nicht nach jedem einzelnen Dokument aufgerufen werden, sondern dies kann auch nach der Generierung aller PDF-Dokumente erfolgen.

Gültigkeit *beliebig*; nach diesem Aufruf sind keine weiteren Aufrufe von API-Funktionen mit dem gleichen PDFlib-Objekt mehr erlaubt.

Bindungen C: Wurde die PDFlib-DLL mit *PDF_new_dl( )* dynamisch zur Laufzeit geladen, so verwenden Sie *PDF_delete_dl( )* zum Löschen des PDFlib-Objekts.

C++: Diese Funktion ist indirekt über den PDFlib-Destruktor verfügbar.
Java: Diese Funktion wird automatisch vom Wrapper-Code aufgerufen. Um Unzulänglichkeiten im Finalizer-System von Java zu umgehen, kann sie jedoch auch explizit aus dem Client-Code heraus aufgerufen werden.
Objective-C: Diese Funktion wird aufgerufen, wenn die Methode *release* des PDFlib-Objekts aufgerufen wird.
Perl und PHP: Diese Funktion wird automatisch aufgerufen, wenn das PDFlib-Objekt den Gültigkeitsbereich verlässt.

2.5 PDFlib Virtual File System (PVF)

Cookbook *Ein vollständiges Codebeispiel hierzu finden Sie im Cookbook-Topic* `general/starter_pvf`.

C++ ***void create_pvf(string filename, const void *data, size_t size, string optlist)***

Java C# ***void create_pvf(String filename, byte[ ] data, String optlist)***

Perl PHP ***create_pvf(string filename, string data, string optlist)***

C ***void PDF_create_pvf(PDF *p,***
const char *filename, int len, const void *data, size_t size, const char *optlist)

Erzeugt eine benannte virtuelle, schreibgeschützte Datei aus Daten im Speicher.

filename (Name-String) Name der virtuellen Datei; dies ist ein beliebiger String, der in weiteren PDFlib-Aufrufen zum Referenzieren der virtuellen Datei verwendet werden kann. Der Name der virtuellen Datei wird von der *SearchPath-Methode* berücksichtigt, wenn nur Schrägstriche '/' als Trennzeichen bei Verzeichnis- oder Dateinamen verwendet werden.

len (Nur C-Sprachbindung) Länge von *filename* (in Bytes). Ist *len*=0, muss ein null-terminierter String übergeben werden.

data Verweis auf die Daten, die den Inhalt der virtuellen Datei bilden sollen. In C und C++ handelt es sich um einen Zeiger auf einen Speicherbereich. In Java ist es ein Byte-Array. In Perl, Phyton und PHP ist es ein String. In COM ist es eine Variante. In REALbasic/Xojo ist es ein MemoryBlock.

size (Nur C- und C++-Sprachbindung) Länge des Speicherblocks mit den Daten in Bytes.

optlist Optionsliste entsprechend Tabelle 2.7. Folgende Option kann verwendet werden: *copy*.

Details Der Name der virtuellen Datei kann an alle API-Funktionen übergeben werden, die Eingabedateien verarbeiten. Benutzen Sie die Option *createpvf* von *PDF_begin_document( )*, um eine PVF-Datei mit der generierten PDF-Ausgabe zu erstellen. Manche Funktionen sperren die virtuelle Datei, solange die Daten verwendet werden. Virtuelle Dateien werden solange im Speicher gehalten, bis sie mit *PDF_delete_pvf( )* explizit oder durch *PDF_delete( )* automatisch gelöscht werden.

PVF-Dateien werden für jedes PDFlib-Objekt getrennt gespeichert. Sie lassen sich nicht von verschiedenen PDFlib-Objekten gemeinsam nutzen, können aber zur Erstellung mehrerer Dokumente mit demselben PDFlib-Objekt verwendet werden. Arbeiten Threads mit verschiedenen PDFlib-Objekten, brauchen sie den PVF-Gebrauch nicht zu synchronisieren. Verweist *filename* auf eine bereits existierende virtuelle Datei, wird eine Exception ausgelöst. Diese Funktion überprüft nicht, ob *filename* bereits für eine normale auf der Festplatte liegende Datei verwendet wird.

Wird die Option *copy* nicht angegeben, darf der Aufrufer die übergebenen Daten erst nach dem erfolgreichen Aufruf von *PDF_delete_pvf( )* ändern oder freigeben (löschen). Bei Nichtbeachtung dieser Regel droht einen Absturz.

Gültigkeit beliebig

Tabelle 2.7 Option für PDF_create_pvf( )

Option	Beschreibung
copy	*(Boolean) PDFlib erzeugt sofort eine interne Kopie der übergebenen Daten. Damit kann der Aufrufer die übergebenen Daten unmittelbar nach dem Aufruf löschen. Die Option* copy *ist in den Sprachbindungen für COM, .NET und Java automatisch auf* true *gesetzt (der Standardwert für andere Bindungen ist* false*). In anderen Sprachbindungen werden die Daten nur kopiert, wenn die Option* copy *explizit übergeben wird.*

C++ Java C# ***int delete_pvf(String filename)***

Perl PHP ***int PDF_delete_pvf(string filename)***

C ***int PDF_delete_pvf(PDF *p, const char *filename, int len)***

Löscht eine benannte virtuelle Datei und gibt die zugehörigen Datenstrukturen (nicht jedoch den eigentlichen Inhalt) frei.

filename (Name-String; wird gemäß der globalen Option *filenamehandling* interpretiert; siehe Tabelle 2.3) Name der virtuellen Datei, der in *PDF_create_pvf( )* übergeben wurde.

len (Nur C-Sprachbindung) Länge von *filename* (in Bytes). Ist *len*=0, muss ein null-terminierter String übergeben werden.

Rückgabe -1 (in PHP: 0), falls die zugehörige virtuelle Datei existiert und gesperrt ist, sonst 1.

Details Ist die Datei nicht gesperrt, werden die zu *filename* gehörigen Datenstrukturen sofort von PDFlib gelöscht. Verweist *filename* nicht auf eine virtuelle Datei, beendet sich diese Funktion kommentarlos. Nach einem erfolgreichen Aufruf kann *filename* wieder verwendet werden. Virtuelle Dateien werden durch *PDF_delete( )* automatisch gelöscht.

Das genaue Verhalten hängt davon ab, ob das zugehörige *PDF_create_pvf( )* mit der Option *copy* aufgerufen wurde: Ist dies der Fall, werden sowohl die administrativen Datenstrukturen der Datei als auch die eigentlichen Daten freigegeben; anderenfalls wird die Freigabe des Inhalts dem Client überlassen.

Gültigkeit beliebig

C++ Java C# ***double info_pvf(string filename, string keyword)***

Perl PHP ***float info_pvf(string filename, string keyword)***

C ***double PDF_info_pvf(PDF *p, const char *filename, int len, const char *keyword)***

Liefert die Eigenschaften einer virtuellen Datei oder des PDFlib Virtual File System (PVF).

filename (Name-String) Der Name der virtuellen Datei. Bei *keyword=filecount* kann der Dateiname leer sein.

len (nur C-Sprachbindung) Länge von *filename* in Bytes. Bei *len=0* muss ein null-terminierter String übergeben werden.

keyword Schlüsselwort gemäß Tabelle 2.8.

Details Diese Funktion gibt verschiedene Eigenschaften einer virtuellen Datei oder des PDFlib Virtual File System (PVF) zurück. Die Eigenschaft wird mit *keyword* ausgewählt.

Gültigkeit beliebig

Tabelle 2.8 Schlüsselwörter für PDF_info_pvf( )

Schlüsselwort	Beschreibung
filecount	*Gesamtzahl der Dateien im PDFlib Virtual File System, die für das aktuelle PDFlib-Objekt verwaltet werden. Die Option* `filename` *wird ignoriert.*
exists	*Gibt 1 aus, wenn die Datei im PDFlib Virtual File System existiert (und nicht gelöscht wurde), andernfalls 0.*
size	*(Nur für bestehende virtuelle Dateien) Gibt die Größe der entsprechenden virtuellen Datei in Bytes aus.*
iscopy	*(Nur für bestehende virtuelle Dateien) Gibt 1 aus, wenn die Option* `copy` *bei Erstellung der angegebenen virtuellen Datei übergeben wurde, anderenfalls 0.*
lockcount	*(Nur für bestehende virtuelle Dateien) Anzahl der Sperren für die angegebene virtuelle Datei, die von PDFlib-Funktionen intern gesetzt wurden. Die Datei kann nur gelöscht werden, wenn der Zähler für die Sperren auf 0 steht.*

2.6 PDF Object Creation API (POCA)

Objekttypen und eingefrorene Objekte. POCA bietet eine Low-Level-Schnittstelle zur Erstellung von PDF-Objekten. POCA unterstützt die folgenden Objekttypen:

- einfache Objekttypen: *boolean, integer, name, float, string;*
- Container-Objekttypen: *array, dictionary, stream;*
- spezielle Typen für PDFlib-Blöcke: *percentage, color*.

Die generierten PDF-Objekte lassen sich folgendermaßen verwenden:

- mit der Option *dpm von PDF_begin/end_dpart( )* lässt sich Document Part Metadata (DPM) für PDF/VT erstellen;
- mit der Option *blocks* von *PDF_begin/end_page_ext( )* lassen sich PDFlib-Blöcke für die Verwendung mit PPS erstellen;
- mit der Option *richmediaargs* von *PDF_create_action( )* lassen sich Argumentlisten für Rich-Media-Annotationen für ActionScript oder JavaScript erstellen.

Bei der Übergabe eines PDF-Container-Objekts an eine der obigen Optionen friert das Container-Objekt selbst sowie alle davon direkt oder indirekt referenzierten Objekte ein, das heißt, der komplette vom Container erstellte Objektbaum wird eingefroren. Eingefrorene Objekte können erneut mit den obigen Optionen verwendet werden, jedoch nicht mehr mit *PDF_poca_insert( )* oder *PDF_poca_remove( )* modifiziert werden.

C++ Java C# ***int poca_new(String optlist)***

Perl PHP ***int poca_new(string optlist)***

C ***int PDF_poca_new(PDF *p, const char *optlist)***

Erstellt ein neues PDF-Container-Objekt vom Typ *dictionary, array* oder *stream* und füllt es mit Objekten.

optlist Optionsliste für die Erstellung und Befüllung eines Containers.

- Optionen für die Erstellung eines Containers gemäß Tabelle 2.9: *containertype, usage*
- Optionen für die Befüllung des Containers mit Objekten gemäß Tabelle 2.11: *direct, hypertextencoding, index, key, type, value, values*

Rückgabe Handle für einen POCA-Container, der solange verwendet werden kann, bis er mit *PDF_poca_delete( )* gelöscht wird.

Details Diese Funktion erstellt ein leeres PDF-Container-Objekt vom angegebenen Container-Typ. Der Container kann entweder sofort im selben Aufruf oder in einem späteren Aufruf von *PDF_poca_insert( )* befüllt werden.

PDF/VT Ein Handle für POCA-Container für ein Objekt vom Typ *dictionary* mit *usage=dpm* kann als Document Part Metadata (DPM) mit der Option *dpm* von *PDF_begin/end_dpart( )* übergeben werden.

Gültigkeit beliebig

Tabelle 2.9 Optionen für PDF_poca_new( )

Option	Beschreibung
container-type	*(Schlüsselwort; erforderlich) Container-Typ:* `dict`, `array` *oder* `stream`. *Nicht angegebene Array-Slots sowie Array-Slots, die entfernt wurden, ohne ein neues Objekt einzufügen, enthalten in der PDF-Ausgabe das Schlüsselwort* `null`. *Hinweis:* `containertype=stream` *ist noch nicht implementiert.*
usage	*(Schlüsselwort; erforderlich) Gibt den Kontext an, in dem der neue Container verwendet werden soll. Diese Option überprüft, ob der Container für die gewünschte Verwendung geeignet ist:* **blocks** *(Nur relevant für* `containertype=dict`*; nur für PPS) Das Block-Dictionary (der Container, der an die Option* `blocks` *von PDF_begin/end_page_ext( ) übergeben wird) muss eine oder mehrere PDFlib-Blockdefinitionen enthalten. Die Option* `usage=blocks` *muss ebenfalls an alle Container-Objekte übergeben werden, die direkt oder indirekt in das neue Dictionary eingefügt werden.* **dpm** *(Nur relevant für* `containertype=dict`*) Alle Schlüssel im neuen Dictionary oder einem darin enthaltenen Dictionary müssen aus ASCII-Zeichen bestehen und die Regeln für ein XML NMTOKEN einhalten. Damit wird gewährleistet, dass das Dictionary als Document Part Metadata (DPM) für PDF/VT verwendet werden kann. Die Option* `usage=dpm` *muss ebenfalls an alle Container-Objekte übergeben werden, die direkt oder indirekt in das neue Dictionary eingefügt werden.* **richmediaargs** *(Nur relevant für* `containertype=array`*) Das Array kann Objekte vom Typ* `string`, `integer`, `float` *oder* `Boolean` *enthalten. Wir empfehlen jedoch folgendes, um Parameter von PDF an Flash zu übergeben: bei einem Parameter für einen ActionScript-Funktionsparameter vom Typ* `string`, `number` *oder* `int` *verwenden Sie in POCA* `type=string` *(d.h. Zahlen müssen in Strings eingebettet werden); wird der Parameter als Typ* `Boolean` *deklariert, verwenden Sie in POCA* `type=boolean` *(d.h. Boolesche Werte dürfen nicht in Strings eingebettet werden). Die POCA-Typen* `integer` *und* `float` *sollten nicht verwendet werden, da Acrobat sie nicht korrekt an ActionScript übergibt.*

C++ Java C# **void poca_delete(int container, String optlist)**

Perl PHP **poca_delete(int container, string optlist)**

C **void PDF_poca_delete(PDF *p, int container, const char *optlist)**

Löscht ein PDF-Container-Objekt.

container Gültiger Handle für einen POCA-Container, der mit *PDF_poca_new( )* erstellt wurde.

optlist Optionsliste gemäß Tabelle 2.10. Die folgende Option kann verwendet werden: *recursive*

Details Der Container wird gelöscht und kann nicht mehr verwendet werden. Wenn der Container von einem anderen Dictionary oder Array referenziert wurde, werden alle Dictionary-Verweise auf den gelöschten Container ebenfalls gelöscht und alle Array-Verweise auf den gelöschten Container durch das Objekt *null* ersetzt. POCA-Container-Objekte werden durch *PDF_end_document( )* nicht automatisch gelöscht.

Gültigkeit beliebig; muss immer paarweise mit dem entsprechenden Aufruf von *PDF_poca_new( )* verwendet werden.

Tabelle 2.10 Optionen für PDF_poca_delete( )

Option	Beschreibung
recursive	*(Boolean) Bei* `true` *wird das Container-Objekt selbst sowie alle davon referenzierten Objekte gelöscht. Dies kann als Abkürzung nützlich sein, um einen gesamten Objektbaum zu löschen, der nicht mehr benötigt wird. Standardwert:* `false`

C++ Java C# **void poca_insert(int container, String optlist)**

Perl PHP **poca_insert(int container, string optlist)**

C **void PDF_poca_insert(PDF *p, int container, const char *optlist)**

Fügt ein einfaches oder ein Container-Objekt in ein PDF-Container-Objekt ein.

container Gültiger Handle für POCA-Container, der mit *PDF_poca_new( )* ermittelt wurde. Eingefrorene Container sind nicht erlaubt, da sie nicht mehr modifiziert werden können (siehe »Objekttypen und eingefrorene Objekte«, Seite 39).

optlist Optionsliste gemäß Tabelle 2.11. Die folgenden Optionen können verwendet werden: *direct, hypertextencoding, index, key, type, value, values*

Details Diese Funktion fügt ein Objekt in einen Container ein. Die Reihenfolge, in der die Objekte eingefügt werden, hat dabei keine Bedeutung. Eingefügte Container können danach befüllt werden; sie müssen zum Zeitpunkt des Einfügens nicht vollständig sein.

Das Einfügen eines Objekts in einen Container darf keine Schleife von direkten Objekten innerhalb des Objektgraphen erzeugen. Ein direkt eingefügtes Dictionary darf zum Beispiel keinen direkten Verweis auf seinen Container enthalten. Zur Erstellung von zyklischen Verweisen verwenden Sie *direct=false,* um indirekte Objekte zu erzeugen, die beliebige andere Objekte referenzieren können.

Gültigkeit beliebig

Tabelle 2.11 Optionen für PDF_poca_new( ), PDF_poca_insert( ) und PDF_poca_remove( )

Option	Beschreibung
direct[1]	*(Boolean; nur bei* `type=array` *und* `dict`*; nicht ausgewertet bei anderen Typen) Bei* `true` *wird das Objekt direkt in den Container eingefügt; bei* `false` *wird ein indirektes PDF-Objekt erzeugt und ein Verweis auf das indirekte PDF-Objekt in den Container eingefügt. Mit indirekten Objekten lässt sich im generierten PDF Platz sparen, wenn ein Objekt mehrfach verwendet wird. Standardwert:* `true`
hypertext-encoding	*(Schlüsselwort) Legt das Encoding für die Optionen* `key`, `value` *und* `values` *fest. Ein leerer String ist äquivalent zu* `unicode`*. Standardwert: Wert der globalen Option* `hypertextencoding`
index	*(Integer; nur für Container mit* `type=array`*; erforderlich für PDF_poca_remove( )) Gibt den null-basierten Index an, bei dem die Werte in das Array eingefügt oder gelöscht werden. Mit dem Wert -1 kann ein Element als neues letztes Element eingefügt werden. Das Array wächst entsprechend, um ein Element mit dem angegebenen Index aufzunehmen. Wenn das Array bereits einen Wert beim angegebenen Index enthält, wird dieser mit dem neuen Wert überschrieben. Standardwert für PDF_poca_new( ) und PDF_poca_insert( ): -1*
key	*(Hypertext-String; nur für Container mit* `type=dict` *und* `stream`*; erforderlich für* `type=dict`*) Der Schlüssel, unter dem der Wert in den Dictionary-Container oder in das mit dem Stream-Container assoziierte Dictionary eingefügt wird. Der Schlüssel darf keine führenden Schrägstriche '/' enthalten. Der Schlüssel muss konform sein zu den in der Dictionary-Option* `usage` *angegebenen Bedingungen. Enthält das Dictionary bereits einen Eintrag mit dem gleichen Schlüssel, wird er mit dem neuen Wert überschrieben.* *Bei* `type=stream` *muss der Schlüssel von* `Length` *und* `Filter` *verschieden sein.*

Tabelle 2.11 Optionen für PDF_poca_new( ), PDF_poca_insert( ) und PDF_poca_remove( )

Option	Beschreibung
type[1]	*(Schlüsselwort; erforderlich außer für Stream-Container ohne die Option* key*) Gibt den Typ des eingefügten Objekts an:* array, boolean, dict, integer, name, float, stream, string, percentage, color *Die folgenden Typen sind nicht erlaubt, wenn der Container mit* usage=dpm *erzeugt wurde:* name *(verwenden Sie stattdessen* type=string*),* stream *Die folgenden Typen sind nur erlaubt, wenn der Container mit* usage=blocks *erzeugt wurde:* color, percentage
value[1]	*(Datentyp gemäß der Option* type*; genau eine der Optionen* value *und* values *muss übergeben werden) Gibt den Wert des eingefügten Objekts abhängig vom Container-Typ und der Option* type *an:* *Bei Array- und Dictionary-Containern:* *Bei* type=boolean *muss der Wert den Datentyp* string *haben und einen der Strings* true *oder* false *enthalten.* *Bei* type=string *oder* name *muss der Wert den Datentyp* Hypertext string *haben und muss das Ziel direkt enthalten. Werte für* type=name *sind in der UTF-8-Darstellung auf 127 Bytes begrenzt und dürfen keinen führenden Schrägstrich '/' enthalten.* *Bei* type=integer *muss der Wert den Datentyp* integer *haben und muss das Ziel direkt enthalten.* *Bei* type=float *muss der Wert den Datentyp* float *oder* integer *haben und muss das Ziel direkt enthalten.* *Bei* type=array, dict *oder* stream *muss der Wert den Datentyp POCA-Container-Handle haben (d.h. mit PDF_poca_new( ) erstellt worden sein) und muss den eingefügten Container angeben. Das eingefügte Objekt muss mit der gleichen Option* usage *erstellt worden sein wie der Container.* *Bei* type=percentage *muss der Wert den Datentyp* number *haben. Er wird als Prozentwert interpretiert und muss ein Prozentzeichen enthalten (z.B.* 50%*). Er wird als Block-Datentyp* percentage *in die PDF-Ausgabedatei geschrieben.* *Bei* type=color *muss der Wert den Datentyp* color *haben (siehe Tabelle 1.2). Er wird als Block-Datentyp* percentage *in die PDF-Ausgabedatei geschrieben. Die folgenden Farbraum-Schlüsselwörter sind nicht erlaubt:* iccbased, iccbasedgray, iccbasedrgb, iccbasedcmyk, pattern *Um beliebige Strings mit dieser Option zu übergeben, empfehlen wir die Syntax für Optionslisten, siehe Abschnitt »Nicht eingeschlossene String-Werte in Optionslisten«, Seite 10.*
values[1]	*(Liste von einem oder mehreren Werten gemäß der Option* type*; nur für Container mit* type=array*; genau eine der Optionen* value *und* values *muss übergeben werden) Gibt einen oder mehrere Werte desselben Typs an, die an der Position in dem Array eingefügt werden, die in der Option* index *festgelegt wurde. Siehe die Option* value *bezüglich der Bedingungen für bestimmte Typen. Enthält die angegebene Liste nur ein einziges Element, ist das Ergebnis äquivalent zu der Option* value*. Enthält die Liste mehr als ein Element, werden alle Elemente aus der Liste nacheinander in das Array eingefügt, gegebenenfalls werden dabei bestehende Elemente überschrieben. Das Array wächst entsprechend, um alle Elemente in der angegebenen Liste aufzunehmen.*

1. Nur für PDF_poca_new( ) und PDF_poca_insert( )

C++ Java C# **void poca_remove(int container, String optlist))**

Perl PHP **poca_remove(int container, string optlist)**

C **void PDF_poca_remove(PDF *p, int container, const char *optlist)**

Entfernt ein einfaches oder ein Container-Objekt aus einem PDF-Container-Objekt.

container Gültiger Handle für POCA-Dictionary oder für ein Array, die mit *PDF_poca_new( )* erstellt wurden. Eingefrorene Container sind nicht erlaubt, da sie nicht mehr modifiziert werden können (siehe »Objekttypen und eingefrorene Objekte«, Seite 39).

optlist Die folgenden Optionen von *PDF_poca_insert( )* können gemäß Tabelle 2.11 verwendet werden: *hypertextencoding, index, key*

Details Diese Funktion entfernt ein Objekt aus einem Container vom Typ Array oder Dictionary. Wenn das angegebene Objekt sich nicht in dem Container befindet, hat diese Funktion keinen Effekt.

Gültigkeit beliebig

3 Dokument- und Seitenfunktionen

3.1 Dokumentfunktionen

C++ Java C# ***int begin_document(String filename, String optlist)***

Perl PHP ***int begin_document(string filename, string optlist)***

C ***int PDF_begin_document(PDF *p, const char *filename, int len, const char *optlist)***

C++ ***void begin_document_callback(size_t (*writeproc) (PDF *p, void *data, size_t size), string optlist)***

C ***void PDF_begin_document_callback(PDF *p,***
size_t (*writeproc) (PDF *p, void *data, size_t size), const char *optlist)

Erstellt anhand verschiedener Optionen eine neue PDF-Datei.

filename (Name-String; wird gemäß der globalen Option *filenamehandling* interpretiert, siehe Tabelle 2.3) Absoluter oder relativer Name der zu generierenden PDF-Ausgabedatei. Ist *filename* leer, wird das PDF-Dokument direkt im Arbeitsspeicher und nicht in einer Datei erzeugt. In diesem Fall müssen die generierten PDF-Daten vom Client mit der Funktion *PDF_get_buffer( )* abgeholt werden. Unter Windows können UNC-Pfade oder Netzwerkfreigaben verwendet werden.

len (Nur C-Sprachbindung) Länge von *filename* (in Bytes). Ist *len* gleich 0, muss ein null-terminierter String übergeben werden.

writeproc (Nur C und C++) C-Callback-Funktion, die von PDFlib aufgerufen wird, um die generierten PDF-Daten vollständig oder teilweise zu übertragen. *writeproc* muss als C-Funktion deklariert sein; C++-Methoden sind nicht erlaubt.

optlist Optionsliste mit Dokumentoptionen:

- Allgemeine Optionen: *errorpolicy* (siehe Tabelle 2.1) und *hypertextencoding* (siehe Tabelle 2.3)
- Dokumentoptionen gemäß Tabelle 3.1. Einige der Optionen können auch in *PDF_end_document( )* angegeben werden: in diesem Fall haben sie Vorrang vor gleichnamigen Optionen in *PDF_begin_document( )*:
  associatedfiles, attachments, autoxmp, destination, groups, labels, linearize, metadata, moddate, objectstreams, openmode, optimize, pagelayout, portfolio, search, uri, viewerpreferences
- Optionen für PDF-Kompatibilität und -Standards gemäß Tabelle 3.2:
  compatibility, limitcheck, nodenamelist, pdfa, pdfua, pdfvt, pdfx, recordlevel, usestransparency
- Optionen für Tagged PDF gemäß Tabelle 3.3:
  checktags, lang, rolemap, structuretype, tag, tagged
- Sicherheitsoptionen gemäß Tabelle 3.4:
  attachmentpassword, masterpassword, permissions, userpassword
- Optionen für die Verarbeitung der Ausgabe gemäß Tabelle 3.5:
  createoutput, createpvf, filemode, flush, inmemory, recordsize, removefragments, tempdirname, tempfilenames

Rückgabe -1 (in PHP: 0) im Fehlerfall, sonst 1. Ist *filename* leer, ist die Funktion in jedem Fall erfolgreich und gibt nie den Fehlercode -1 (in PHP: 0) zurück.

Details Diese Funktion erzeugt eine neue PDF-Datei mit dem Namen *filename*. PDFlib versucht, eine Datei mit dem übergebenen Namen zu öffnen und schließt die Datei, sobald das PDF-Dokument fertig ist.

PDF_begin_document_callback( ) öffnet ein neues PDF-Dokument über den Aufruf einer vom Client übergebenen Callback-Funktion, um die PDF-Ausgabedaten zu übergeben, ohne dabei in eine Datei zu schreiben. Die in *writeproc* übergebene Callback-Funktion muss die Anzahl der geschriebenen Bytes zurückgeben. Entspricht der Rückgabewert nicht von PDFlib übergebenen Argument *size*, wird eine Exception ausgelöst. Die Frequenz der *writeproc*-Aufrufe lässt sich mit der Option *flush* einstellen.

PDF/VT Die folgende Option ist nicht erlaubt: *groups*.

Gültigkeit *object;* mit dieser Funktion beginnt der Gültigkeitsbereich *document*, sofern die Datei erfolgreich geöffnet werden konnte. Diese Funktion muss immer paarweise mit *PDF_end_document( )* aufgerufen werden.

Bindungen ASP: Um vollständige Pfadnamen zur Übergabe an diese Funktion zu erzeugen, sollten Sie die *MapPath*-Funktion verwenden.

C, C++, Java, JScript: Achten Sie darauf, die Sonderbedeutung des Gegenschrägstrichs im Pfadseparator korrekt aufzuheben. Folgendes Beispiel bezeichnet eine Datei auf einem Netzlaufwerk: *\\\\malik\\rp\\foo.pdf*.
PDF_begin_document_callback( ) ist nur in C und C++ verfügbar.

C++ Java C# ***void end_document(String optlist)***

Perl PHP ***end_document(string optlist)***

C ***void PDF_end_document(PDF *p, const char *optlist)***

Schließt das generierte PDF-Dokument unter Anwendung verschiedener Optionen.

optlist Optionsliste mit Optionen zur Dokumentverarbeitung:
- Allgemeine Option: *hypertextencoding* (siehe Tabelle 2.3)
- Dokumentoptionen gemäß Tabelle 3.1. Optionen in *PDF_end_document( )* haben Vorrang vor gleichnamigen Optionen in *PDF_begin_document( )*. Folgende Optionen können verwendet werden:
  action, associatedfiles, attachments, autoxmp, destination, destname, labels, metadata, moddate, openmode, pagelayout, portfolio, search, uri, viewerpreferences.

Details Diese Funktion beendet die Generierung des PDF-Dokuments, gibt alle dokumentspezifischen Ressourcen frei und schließt die Ausgabedatei, sofern das PDF-Dokument mit *PDF_begin_document( )* geöffnet wurde. Diese Funktion muss nach abgeschlossener Seitengenerierung durch den Client auf jeden Fall aufgerufen werden, unabhängig davon, auf welche Art das PDF-Dokument geöffnet wurde.

Wurde das Dokument direkt im Speicher und nicht in einer Datei generiert, bleibt der Dokumentpuffer auch nach dieser Funktion erhalten, so dass er mit *PDF_get_buffer( )* ausgelesen werden kann. Er wird freigegeben, sobald *PDF_begin_document( )* erneut aufgerufen wird oder das PDFlib-Objekt seine Gültigkeit verliert.

Gültigkeit *document;* mit dieser Funktion wird der Gültigkeitsbereich *document* beendet; diese Funktion muss immer paarweise mit einer der Funktionen *PDF_begin_document( )* oder *PDF_begin_document_callback( )* aufgerufen werden.

Tabelle 3.1 Dokumentoptionen für PDF_begin_document( ) und PDF_end_document( )

Option	Beschreibung
action[1]	*(Aktionsliste; bei PDF/A nicht zulässig) Liste mit Dokumentaktionen für eines oder mehrere der folgenden Ereignisse (Standardwert: leere Liste):* ***open*** *Aktionen beim Öffnen des Dokuments. Beachten Sie, dass wegen der Ausführungsreihenfolge in Acrobat für* open*-Aktionen kein JavaScript auf Dokumentebene zulässig ist.* ***didprint/didsave/willclose/willprint/willsave*** *JavaScript-Aktionen, die nach dem Drucken/nach dem Speichern/vor dem Schließen/vor dem Drucken/vor dem Speichern des Dokuments durchgeführt werden.*
associated-files[1]	*(Liste mit Asset-Handles; nur für PDF 2.0 und PDF/A-3) Asset-Handles für assoziierte Dateien gemäß PDF/A-3. Die Dateien müssen mit PDF_load_asset( ) geladen werden und* `type=attachment` *haben.*
attachments	*(Liste mit Optionslisten oder Liste mit Asset-Handles; nicht für PDF/X-1a/3 und PDF/A-1; PDF/A-2: nur PDF/A-1- und PDF/A-2-Dokumente können angehängt werden; PDF/A-3: nicht erlaubt, verwenden Sie stattdessen* `associatedfiles`*) Definiert Dateianhänge auf Dokumentebene, die mit PDF_load_asset( ) und* `type=attachment` *geladen wurden. Dateianhänge können sowohl in PDF_begin_document( ) als auch in PDF_end_document( ) angegeben werden. Asset-Handles können jedoch nur an PDF_end_document( ) übergeben werden. Unterstützte Unteroptionen: siehe Tabelle 13.8*
autoxmp	*(Boolean; wird im PDF/A-Modus und im PDF/X-3/4/5-Modus immer auf* `true` *gesetzt) Bei* `true` *erzeugt PDFlib aus Dokument-Infofeldern XMP-Dokumentmetadaten (siehe Abschnitt 14.2, »XMP-Metadaten«, Seite 277). Standardwert:* `false`
destination	*(Optionsliste; wird ignoriert, wenn mit* `action` *eine open-Aktion definiert wurde) Optionsliste zur Festlegung einer Dokumentöffnen-Aktion gemäß Tabelle 12.14.*
destname[1]	*(Hypertext-String; wird ignoriert, wenn die Option* `destination` *angegeben wurde) Name eines mit PDF_add_nameddest( ) definierten Ziels, das als Dokumentöffnen-Aktion verwendet wird.*
groups[2]	*(String-Liste; nicht erlaubt im PDF/VT-Modus oder wenn eine Document Part Hierarchy erzeugt wird) Definiert die Namen und die Reihenfolge der in einem Dokument verwendeten Seitengruppen. Seitengruppen halten Seiten zusammen, was zum Beispiel beim Festlegen von Namen für Seitenbereiche nützlich ist. Seiten können einer Seitengruppe zugeordnet und innerhalb dieser Gruppe referenziert werden. Sind Seitengruppen für ein Dokument definiert, so müssen alle Dokumentseiten auch einer Seitengruppe zugeordnet werden.*
labels	*(Liste mit Optionslisten) Liste mit einer oder mehreren Optionslisten gemäß Tabelle 3.6, die symbolische Seitennamen festlegen. Diese Namen werden als Seiten-Labels (statt der Seitennummer) in der Statusleiste von Acrobat angezeigt. Die Kombination der Werte* `style`/`prefix`/`start` *muss innerhalb eines Dokuments eindeutig sein. Standardwert: keine Seitennamen*
linearize[2]	*(Boolean; nicht für PDF_begin_document_callback( )) Bei* `linearize=true` *wird das Ausgabedokument linearisiert. Auf z/OS kann diese Option nicht mit einem leeren Dateinamen kombiniert werden. Standardwert:* `false`
metadata	*(Optionsliste) Übergibt XMP-Dokument-Metadaten (siehe Abschnitt 14.2, »XMP-Metadaten«, Seite 277). Einzelne XMP-Properties können mit Dokument-Infofeldern überschrieben werden, die mit PDF_set_info( ) gesetzt werden. Im PDF/A-Modus müssen die übergebenen XMP-Metadaten zusätzlichen Anforderungen genügen (siehe PDFlib-Tutorial).*
moddate	*(Boolean) Falls* `true`*, wird das Dokument-Infofeld* `ModDate` *(modification date) erzeugt, was bei der Arbeit mit bestimmten Preflight-Tools nützlich sein kann. Standardwert:* `false`

Tabelle 3.1 Dokumentoptionen für PDF_begin_document( ) und PDF_end_document( )

Option	Beschreibung
objectstreams[2]	*(Liste mit Schlüsselwörtern; PDF 1.5; wird immer auf* `false` *gesetzt, wenn* `optimize` *oder* `linearize` *auf* `true` *gesetzt ist) Generiert komprimierte Object-Streams, die die Größe der Ausgabedateien deutlich reduzieren (Standardwert:* `{other nodocinfo}`*):* **bookmarks** *Lesezeichen-Objekte komprimieren* **docinfo** *Dokument-Infofelder komprimieren* **dpartarrays** *Auf Document Part Hierarchy bezogene Dictionaries komprimieren* **dpartdicts** *Auf Document Part Hierarchy bezogene Arrays komprimieren* **fields** *Formularfelder komprimieren* **names** *Objekte für benannte Ziele komprimieren* **none** *Keine komprimierten Object-Streams erzeugen (außer für Kategorien, die explizit nach dieser Option aktiviert wurden)* **other** *Alle Kategorien komprimieren, die nicht explizit nach diesem Schlüsselwort deaktiviert wurden und andere Objekttypen ohne eigene Schlüsselwörter komprimieren* **pages** *Objekte komprimieren, die den Seitenbaum umfassen* **poca** *Alle einfachen Objekte komprimieren, die mit der POCA-Schnittstelle erzeugt wurden* **tags** *Tags für markierten Inhalt komprimieren* **xref** *Komprimierten xref-Stream erzeugen. Diese Kategorie wird automatisch aktiviert, wenn mindestens noch eine weitere Kategorie aktiviert ist.* *Außer bei* `none` *und* `other` *kann allen Schlüsselwörtern das Präfix* `no` *vorangestellt werden (z.B.* `nodocinfo`*), um die Kompression für die angegebene Kategorie zu deaktivieren. Wenn mindestens ein Schlüsselwort mit* `no` *übergeben wird, wird das Schlüsselwort* `other` *der Liste vorangestellt.*
openmode	*(Schlüsselwort) Legt die Anzeige des Dokuments beim Öffnen fest. Standardwert:* `bookmarks`*, sofern das Dokument Lesezeichen enthält, sonst* `none`*.* **none** *Nur Dokumentfenster anzeigen* **bookmarks** *Lesezeichen-Fenster anzeigen* **thumbnails** *Seiten-Fenster anzeigen* **fullscreen** *Im Vollbildmodus öffnen (funktioniert im Browser nicht)* **layers** *(PDF 1.5) Ebenen-Fenster anzeigen* **attachments** *(PDF 1.6) Dateianlagen-Fenster anzeigen*
optimize[2]	*(Boolean) Bei* `true` *wird das Ausgabedokument erst generiert und dann in einem weiteren Durchgang optimiert. Die Optimierung entfernt Objekte, die überflüssigerweise mehrfach vorkommen. Im Allgemeinen hat die Optimierung außer nicht sinnvollem Client-Code keine Auswirkungen (z.B. mehrfaches Laden eines Rasterbildes oder ICC-Profils statt Wiederverwendung des Handles). Auf z/OS kann diese Option nicht mit der direkten Generierung im Speicher kombiniert werden (d.h. einem leeren Dateinamen). Standardwert:* `false`
pagelayout	*(Schlüsselwort) Seitenlayout beim Öffnen des Dokuments (Standardwert:* `default`*):* **default** *Standardeinstellung von Acrobat* **singlepage** *Nur einzelne Seiten anzeigen* **onecolumn** *Seiten fortlaufend anzeigen* **twocolumnleft** *Doppelseiten anzeigen, ungerade Seiten links* **twocolumnright** *Doppelseiten anzeigen, ungerade Seiten rechts* **twopageleft** *(PDF 1.5) Doppelseiten anzeigen, ungerade Seiten links* **twopageright** *(PDF 1.5) Doppelseiten anzeigen, ungerade Seiten rechts*
portfolio[1]	*(Optionsliste; PDF 1.7) Unteroptionen zur Erzeugung eines PDF-Portfolios gemäß Tabelle 12.17*

Tabelle 3.1 Dokumentoptionen für PDF_begin_document( ) und PDF_end_document( )

Option	Beschreibung
search	*(Optionsliste; nicht in ISO 32000-1) Mit dieser Option wird Acrobat veranlasst, beim Öffnen des Dokuments einen Suchindex zu verwenden. Folgende Unteroptionen werden unterstützt:* ***filename*** *(Hypertext-String; erforderlich) Dateiname des Suchindex* ***indextype*** *(Name-String) Typ des Index; muss für Acrobat gleich* PDX *sein. Standardwert:* PDX
uri	*(String) Setzt den Basis-URL des Dokuments. Dies kann nützlich sein, wenn ein Dokument mit relativen Webverknüpfungen auf andere Dokumente an einen neuen Ort verschoben wird. Das Anpassen des Basis-URL sorgt dafür, dass relative Links weiterhin funktionieren. Standardwert: kein URI*
viewer-preferences	*(Optionsliste) Optionsliste mit verschiedenen Viewer-Voreinstellungen gemäß Tabelle 3.7. Standardwert: leer*

1. Nur für PDF_end_document( )
2. Nur für PDF_begin_document( ) und PDF_begin_document_callback( )

Tabelle 3.2 Optionen für PDF-Kompatibilität und -Standards in PDF_begin_document( )

Option	Beschreibung
compatibility	*(Schlüsselwort; wird ignoriert, wenn eine der Optionen* pdfa, pdfua, pdfvt *oder* pdfx *mit einem von* none *verschiedenen Wert verwendet wird) Setzt die PDF-Version des Dokuments auf eines der unten angegebenen Schlüsselwörter. Diese Option beeinflusst, welche Features zur PDF-Erzeugung zur Verfügung stehen und welche PDF-Dokumente mit PDFlib+PDI importiert werden können (Standardwert:* 1.7*):* ***1.4*** *PDF 1.4 benötigt Acrobat 5 oder höher.* ***1.5*** *PDF 1.5 benötigt Acrobat 6 oder höher.* ***1.6*** *PDF 1.6 benötigt Acrobat 7 oder höher.* ***1.7*** *PDF 1.7 ist in ISO 32000-1 spezifiziert und benötigt Acrobat 8 oder höher.* ***1.7ext3*** *PDF 1.7 extension level 3 benötigt Acrobat 9 oder höher.* ***1.7ext8*** *PDF 1.7 extension level 8 benötigt Acrobat X oder höher.* ***2.0*** *PDF 2.0 ist in ISO 32000-2 spezifiziert.*
limitcheck	*Bei* true *wird der Höchstwert für die Anzahl von indirekten PDF-Objekten (8 388 607) berücksichtigt. Standardwert:* true
nodenamelist	*(Liste aus Name-Strings; erforderlich für* pdfvt=PDF/VT-1 *und* pdfvt=PDF/VT-2*) Namen für alle Level der Document Part Hierarchy. Alle Namen müssen aus ASCII-Zeichen bestehen und müssen die Regeln für ein XML NMTOKEN einhalten. Der erste String legt den Namen für Level 0 in der Document Part Hierarchy fest.*
pdfa	*(Schlüsselwort) Setzt die PDF/A-Kompatibilitätsstufe auf einen der folgenden Werte (Standardwert:* none*):* PDF/A-1a:2005, PDF/A-1b:2005 *(impliziert* compatibility=1.4*)* PDF/A-2a, PDF/A-2b, PDF/A-2u *(impliziert* compatibility=1.7*)* PDF/A-3a, PDF/A-3b, PDF/A-3u *(impliziert* compatibility=1.7*)* none PDF/A1-a:2005, PDF/A-2a *und* PDF/A-3a *implizieren* tagged=true*. PDF/A kann gleichzeitig auch konform zu anderen Standards sein, und zwar wie folgt:* pdfx=PDF/X-1a:2003, PDF/X-3:2003, PDF/X-4 pdfvt=PDF/VT-1 pdfua=PDF/UA-1 *Werden mehrere Optionen für PDF-Standards angegeben, wird der Wert für die niedrigste Kompatibilitätsstufe verwendet.*

Tabelle 3.2 Optionen für PDF-Kompatibilität und -Standards in PDF_begin_document( )

Option	Beschreibung
pdfua	*(Schlüsselwort) Setzt die PDF/UA-Konformitätsstufe auf einen der folgenden Werte (Standardwert:* none*):* ***PDF/UA-1*** *Impliziert* compatibility=1.7 *und* tagged=true*.* ***none*** *Keine PDF/UA-Ausgabe*
pdfvt	*(Schlüsselwort) Setzt die PDF/VT-Konformitätsstufe auf einen der folgenden Werte (Standardwert:* none*):* ***PDF/VT-1*** *Impliziert* pdfx=PDF/X-4*; jeder andere Wert für die Option* pdfx *ist ein Fehler.* ***PDF/VT-2*** *In der Option* pdfx *muss einer der folgenden Werte angegeben werden:* PDF/X-4p*,* PDF/X-5g*,* PDF/X-5pg*; jeder andere Wert für die Option* pdfx *ist ein Fehler.* ***none*** *Keine PDF/VT-Ausgabe*
pdfx	*(Schlüsselwort) Setzt die PDF/X-Konformitätsstufe auf einen der folgenden Werte (Standardwert:* none*):* PDF/X-1a:2003*(impliziert* compatibility=1.4*)* PDF/X-3:2003*(impliziert* compatibility=1.4*)* PDF/X-4, PDF/X-4p[1]*(impliziert* compatibility=1.6*)* PDF/X-5g, PDF/X-5pg[1]*(impliziert* compatibility=1.6*)* none
recordlevel	*(Nicht negativer Integer; nur relevant, wenn eine Document Part Hierarchy erzeugt wird) Null-basierter Level der Document Part Hierarchy, die den Datensätzen für die Empfänger (recipient records) entspricht.*
uses-transparency	*(Boolean; nur für PDF/VT) Bei* false *enthält keine der Seiten im generierten Dokument transparente Objekte. PDFlib löst eine Exception aus, wenn diese Zusicherung verletzt wird. Diese Option darf nur bei Dokumenten ohne Transparenz auf* false *gesetzt werden. Sie vereinfacht die Erzeugung von gekapselten XObjects für PDF/VT, da alle XObjects bedingungslos als gekapselt markiert werden. Standardwert:* true

1. Im PDFlib-Tutorial finden Sie einen wichtigen Hinweis zu den Problemen mit referenzierten ICC-Profilen in Acrobat.

Tabelle 3.3 Optionen für Tagged PDF in PDF_begin_document( )

Option	Beschreibung
checktags	*(Schlüsselwort; muss im PDF/UA-Modus auf* strict *gesetzt sein) Legt fest, ob die Verschachtelungsregeln für Strukturelemente (siehe PDFlib-Tutorial) auf solche Elemente angewendet werden sollen, die mit PDF_begin_item( ) oder der Option* tag *verschiedener Funktionen erzeugt wurden. Diese Option ist nur als Migrationshilfe vorgesehen. Sie hat keine Auswirkungen auf die Tags in importierten Seiten (siehe Option* checktags *von PDF_open_pdi_document( )). Unterstützte Schlüsselwörter (Standardwert:* strict*):* ***none*** *Verschachtelungsregeln für Tags werden nicht angewandt. Diese Einstellung kann zu einer ungültigen Strukturhierarchie führen und wird deshalb nicht empfohlen.* ***relaxed*** *Ähnlich wie* strict*, außer dass einige Regeln nicht angewendet werden (siehe PDFlib-Tutorial).* ***strict*** *Wenn ein Tag die Verschachtelungsregeln verletzt, wird eine Exception ausgelöst.*
lang	*(String; empfehlenswert bei* tagged=true*) Setzt die Sprache des Dokuments als Sprachcode aus zwei Zeichen gemäß ISO 639 (Beispiele: DE, EN, FR, JA). Optional kann ein Minuszeichen und ein zwei Zeichen langer Ländercode gemäß ISO 3166 folgen (Beispiele: EN-US, EN-GB, ES-MX). Es wird nicht zwischen Groß- und Kleinschreibung unterschieden.* *Die Sprache kann für einzelne Einheiten auf allen Ebenen des Strukturbaums geändert werden, sollte aber anfangs für das Gesamtdokument gesetzt werden.* *PDF/UA: Die natürliche Sprache muss mit dieser Option oder Unteroption* lang *der einzelnen Strukturelemente angegeben werden.*

Tabelle 3.3 Optionen für Tagged PDF in PDF_begin_document( )

Option	Beschreibung
rolemap	*(Liste mit String-Listen; das erste Element in jeder String-Liste ist ein Name-String, das zweite Element ist ein String; nur für Tagged PDF; erforderlich, wenn benutzerdefinierte Elementtypen verwendet werden) Legt die Zuordnung von benutzerdefinierten Elementtypen zu Standard-Elementtypen fest. Jede Teilliste enthält den Namen eines Standard- oder benutzerdefinierten Elementtyps und den Namen des Standard-Elementtyps, dem der erste Typ zugeordnet wird. Inline- und Pseudo-Elementtypen sind nicht für den zweiten Eintrag in einer Teilliste erlaubt. Namen für Standard-Elementtypen können auch anderen Standard-Elementtypen zugeordnet werden, um bestehenden Elementtypen eine andere Bedeutung zuzuordnen. Indirekte Zuordnungen sind erlaubt, das heißt, ein benutzerdefinierter Typ kann auf einen anderen benutzerdefinierten Typ gemappt werden, der wiederum einem Standardtyp zugeordnet wird. Paare mit identischen Einträgen werden ignoriert. Für weitere Informationen siehe Abschnitt 14.3, »Tagged PDF«, Seite 278. Benutzerdefinierte Namen für Elementtypen dürfen nicht mit dem reservierten Präfix* `Plib` *beginnen.* *Bei PDF/UA ist es ist nicht erlaubt, Standard-Elementtypen neu zuzuordnen.*
structuretype	*(Schlüsselwort; nur für PDF/UA) Dokumentstruktur-Typ. Unterstützte Schlüsselwörter (Standardwert:* `weak`*):* **strong** *Das Dokument ist stark strukturiert, das heißt, der Strukturbaum spiegelt die logische Gliederung des Dokuments wider. Der einzig erlaubte Strukturtyp für Überschriften ist* `H`*, während* `H1`*,* `H2` *usw. nicht erlaubt sind. Jeder Knoten im Strukturbaum enthält höchstens ein* `H`*-Tag sowie ein oder mehrere Absatz-Tags* `P`*.* **weak** *Das Dokument ist schwach strukturiert, das heißt, der Strukturbaum ist nur wenige Ebenen tief, alle Überschriften, Absätze usw. sind unmittelbar untergeordnet. Die logische Struktur kann mit Überschriften-Tags wie* `H1`*,* `H2` *ausgedrückt werden, wobei* `H` *nicht zulässig ist. Überschriften dürfen keine direkten Nachkommen haben.*
tag	*(Optionsliste) Tagging-Optionen gemäß Tabelle 14.4. Das angegebene Strukturelement bildet die Wurzel des Strukturbaums und wird automatisch in PDF_end_document( ) geschlossen. Für die Unteroption* `tagname` *sind nur Gruppierungselemente erlaubt.*
tagged	*(Boolean; PDF 1.4) Bei* `tagged=true` *wird die Ausgabe als Tagged PDF generiert. Vom Client müssen im Tagged-PDF-Modus korrekte Strukturinformationen übergeben werden (siehe Abschnitt 14.3, »Tagged PDF«, Seite 278). Ist der Modus* `PDF/A-1a:2005`*,* `PDF/A-2a`*,* `PDF/A-3a` *oder* `PDF/UA-1` *aktiv, wird diese Option automatisch auf* `true` *gesetzt. Standardwert:* `false`

Tabelle 3.4 Sicherheitsoptionen für PDF_begin_document(); nicht erlaubt für PDF/A und PDF/X

Option	Beschreibung
attachmentpassword[1]	*(String*[2]*; PDF 1.6; wird ignoriert, wenn* `userpassword` *oder* `masterpassword` *gesetzt sind; kann nicht mit den Optionen* `linearize` *und* `optimize` *kombiniert werden; nicht für PDF/A und PDF/X) Dateianlagen werden mit dem übergebenen String als Kennwort verschlüsselt. Das übrige Dokument bleibt unverschlüsselt. Auf EBCDIC-Plattformen wird das Kennwort in EBCDIC-Encoding oder EBCDIC-UTF-8 erwartet.*
masterpassword[1]	*(String; für* `permissions` *erforderlich; nicht für PDF/A und PDF/X) Master-Kennwort für das Dokument. Ist es leer, so wird kein Master-Kennwort angewandt. Auf EBCDIC-Plattformen wird das Kennwort in EBCDIC-Encoding oder EBCDIC-UTF-8 erwartet. Standardwert: nicht vorhanden*
permissions	*(Schlüsselwortliste; nicht für PDF/A und PDF/X) Liste der Zugriffsberechtigungen für das Ausgabedokument. Sie besteht aus beliebig vielen der folgenden Schlüsselwörter (Standardwert: leere Liste):* **noprint** *Acrobat verbietet das Drucken des Dokuments.* **nohiresprint** *Acrobat verbietet das Drucken des Dokuments in hoher Auflösung. Ist* `noprint` *nicht angegeben, so ist der Ausdruck auf die Funktion »Als Bild drucken« beschränkt, die die Seite in niedriger Auflösung druckt.* **nomodify** *Acrobat verbietet das Bearbeiten und Beschneiden von Seiten sowie das Hinzufügen und Ändern von Formularfeldern.* **noassemble** *(Impliziert* `nomodify`*) Acrobat verbietet das Einfügen, Löschen und Drehen von Seiten sowie das Anlegen von Lesezeichen und Miniaturseiten (Thumbnails).* **noannots** *Acrobat verbietet das Hinzufügen oder Ändern von Kommentaren oder Formularfeldern.* **noforms** *(Impliziert* `nomodify` *und* `noannots`*) Acrobat verbietet das Ausfüllen von Formularfeldern.* **nocopy** *Acrobat verbietet das Kopieren oder Extrahieren von Text oder Grafik; der barrierefreie Zugang wird durch* `noaccessible` *gesteuert.* **noaccessible** *(In PDF 2.0 nicht mehr unterstützt; bei PDF/UA nicht erlaubt) Acrobat verbietet die Extraktion von Text oder Grafik für den barrierefreien Zugang (zum Beispiel durch einen Screenreader).* **plainmetadata** *(PDF 1.5) Beim Verschlüsseln des Dokuments bleiben die XMP-Dokumentmetadaten unverschlüsselt.*
userpassword[1]	*(String; nicht für PDF/A und PDF/X) Benutzerkennwort für das Dokument. Ist* `userpassword` *leer, wird kein Benutzerkennwort verwendet. Auf EBCDIC-Plattformen wird das Kennwort in EBCDIC-Encoding oder EBCDIC-UTF-8 erwartet. Standardwert: nicht vorhanden*

1. Um beliebige Strings mit dieser Option zu übergeben, siehe die Syntax für Optionslisten in »Nicht eingeschlossene String-Werte in Optionslisten«, Seite 10.
2. In Kennwörtern sind Zeichen außerhalb des Winansi-Encodings für `compatibility=1.7ext3` *oder höher erlaubt.*

Tabelle 3.5 Optionen zur Verarbeitung der Ausgabe für PDF_begin_document( )

Option	Beschreibung
createoutput	*(Boolean) Bei* `false` *wird der Parameter* `filename` *ignoriert und keine Ausgabedatei oder Speicherbereich erzeugt. Diese Option impliziert* `compress=0`, `linearize=false` *und* `optimize=false`. *Standardwert:* `true`
createpvf	*(Boolean) Bei* `true` *wird die PDF-Datei direkt im Speicher und nicht auf der Festplatte erzeugt. Der übergebene Dateiname ist der Name einer virtuellen Datei, die mit dem Aufruf von PDF_end_document( ) erzeugt wird. In diesem Fall kann PDF_get_buffer( ) nicht aufgerufen werden, um die PDF-Ausgabe abzuholen; stattdessen kann der Name der erzeugten PVF-Datei an andere PDFlib-Funktionen übergeben werden. Dies kann für die Erzeugung von Dokumenten für ein PDF-Portfolio nützlich sein. Standardwert:* `false`
filemode	*(String, nur für z/OS und USS) Parameter-String für den Dateimodus (file mode) der erzeugten PDF-Datei und eventuell erzeugter Temporärdateien (z.B. mit der Option* `linearize`*). Der angegebene String wird an den Standard-Dateimodus* `»wb,«` *angehängt. Die Option* `recordsize` *muss konsistent zum Inhalt dieser Option sein. Beispiel:* `recfm=fb,lrecl=80,space=(cyl,(1,5)`. *Standardwert: leer oder* `recfm=v` *bei Ausgabe ohne feste Blockgröße*
flush	*(Schlüsselwort; nur für PDF_begin_document_callback( )) Setzt die Flushing-Strategie (Standardwert:* `page`*):* ***none*** *Nur einmal am Ende des Dokuments flushen* ***page*** *Am Ende jeder Seite flushen* ***content*** *Nach allen Fonts, Rasterbildern, Dateianhängen und Seiten flushen* ***heavy*** *Immer leeren, sobald der interne 64 KB große Dokumentpuffer voll ist*
inmemory	*(Boolean; nicht für PDF_begin_document_callback( )) Sind* `inmemory` *sowie die Option* `linearize` *oder* `optimize` *gleich* `true`*, erzeugt PDFlib bei der Linearisierung keine temporären Dateien, sondern verarbeitet das Dokument direkt im Speicher. Auf manchen Systemen (insbesondere z/OS) lässt sich die Leistung damit erheblich steigern. Es ist jedoch Speicher in der doppelten Dokumentgröße erforderlich. Bei* `inmemory=false` *wird bei der Linearisierung und Optimierung eine temporäre Datei angelegt. Standardwert:* `false`
recordsize	*(Integer; nur z/OS und USS) Record-Größe für die Ausgabedatei sowie für alle temporären Dateien, die für die Optionen* `linearize` *und* `optimize` *erzeugt werden. Standardwert: 0 (Ausgabe ohne feste Blockgröße)*
remove-fragments	*Bei* `true` *wird in PDF_delete( ) die unfertige PDF-Datei gelöscht, die nach einer Exception übriggeblieben ist. Solche PDF-Fragmente sind als Dokumente ohnehin unbrauchbar. Diese Option hat keine Auswirkungen, wenn ein leerer Dateiname angegeben wird, also für die PDF-Erzeugung im Speicher. Standardwert:* `false`
tempdirname	*(String; nicht für PDF_begin_document_callback( )) Name des Verzeichnisses, in dem temporäre Dateien gespeichert werden, die für die Optionen* `linearize` *und* `optimize` *benötigt werden. Ist diese Option leer, werden temporäre Dateien im aktuellen Verzeichnis abgelegt. Ist die Option* `tempfilenames` *vorhanden, so wird* `tempdirname` *ignoriert. Standardwert: nicht vorhanden*
temp-filenames	*(Liste mit zwei Strings; nur für z/OS und USS) Vollständige Namen zweier temporärer Dateien, die für die Optionen* `linearize` *und* `optimize` *benötigt werden. Ist* `tempfilenames` *leer, generiert PDFlib selbst eindeutige Namen. Die temporären Dateien müssen nach PDF_end_document( ) explizit gelöscht werden. Wird diese Option übergeben, darf der Parameter* `filename` *nicht leer sein. Standardwert: nicht vorhanden*

Tabelle 3.6 Unteroptionen für `labels` *in PDF_begin/end_document( ) und* `label` *in PDF_begin/end_page_ext( )*

Option	Beschreibung
group	*(String; nur für PDF_begin_document( ); erforderlich, wenn das Dokument Seitengruppen verwendet, aber sonst nicht zulässig) Das Label wird auf alle Seiten der festgelegten sowie aller nachfolgenden Gruppen angewandt, bis ein neues Label zum Einsatz kommt. Der Gruppenname muss mit der Option* `groups` *in PDF_begin_document( ) definiert worden sein.*
hypertext-encoding	*(Schlüsselwort) Legt den Zeichensatz für die Option* `prefix` *fest. Ein leerer String wird als* `unicode` *interpretiert. Standardwert: Wert der globalen Option* `hypertextencoding`**.**
pagenumber	*(Integer; nur für PDF_end_document( ); erforderlich, wenn das Dokument keine Seitengruppen verwendet, aber sonst nicht zulässig) Das Label wird auf die festgelegte Seite sowie solange auf alle nachfolgenden Seiten angewandt, bis ein neues Label zum Einsatz kommt.*
prefix	*(Hypertext-String) Das Präfix für alle Labels des Bereichs. Standardwert: nicht vorhanden*
start	*(Integer >= 1) Numerischer Wert für das erste Label des Bereichs. Nachfolgende Seiten des Bereichs werden sequentiell ab diesem Wert durchnummeriert. Standardwert: 1*
style	*(Schlüsselwort) Der zu verwendende Nummerierungsstil. Standardwert:* `none` **none** *keine Seitennummern; Labels bestehen nur aus dem Präfix.* **D** *Dezimale arabische Ziffern (1, 2, 3, ...)* **R** *Große römische Zahlen (I, II, III, ...)* **r** *Kleine römische Zahlen (i, ii, iii, ...)* **A** *Großbuchstaben (A, B, C, ..., AA, BB, CC, ...)* **a** *Kleinbuchstaben (a, b, c, ..., aa, bb, cc, ...)*

Tabelle 3.7 Unteroptionen für die Option `viewerpreferences` *in PDF_begin_document( ) und PDF_end_document( )*

Option	Beschreibung
centerwindow	*(Boolean) Bei* `true` *wird das Dokumentfenster am Bildschirm zentriert. Standardwert:* `false`
direction	*(Schlüsselwort) Leserichtung des Dokuments, die sich auf das Blättern in der Doppelseitenansicht und auf die erste Seite (links/rechts) bei doppelseitigem Layout in Acrobat auswirkt. (Standardwert:* `l2r`*):* **l2r** *Von links nach rechts* **r2l** *Von rechts nach links (einschließlich der Schriftsysteme mit vertikaler Laufrichtung)*
displaydoctitle	*(Boolean; im PDF/UA-Modus ist nur* `true` *erlaubt) Legt fest, ob in der Titelleiste von Acrobat das Dokument-Infofeld »Titel«* (`true`) *oder der Dateiname* (`false`) *angezeigt wird. Standardwert:* `true` *für PDF/UA; anderenfalls* `false`
duplex	*(Schlüsselwort; PDF 1.7) Layout-Optionen für den Druckdialog (Standardwert:* `none`*):* **DuplexFlipShortEdge** *Beidseitiger Druck mit Wenden an der kurzen Papierseite* **DuplexFlipLongEdge** *Beidseitiger Druck mit Wenden an der langen Papierseite* **none** *Keine Layout-Optionen festgelegt* **Simplex** *Einseitiger Druck*
fitwindow	*(Boolean) Legt fest, ob das Dokumentfenster an die Größe der ersten Seite angepasst wird. Standardwert:* `false`
hidemenubar[1]	*(Boolean) Legt fest, ob die Menüleiste von Acrobat ausgeblendet wird. Standardwert:* `false`
hidetoolbar[1]	*(Boolean) Legt fest, ob die Werkzeugleisten von Acrobat ausgeblendet werden. Standardwert:* `false`

Tabelle 3.7 Unteroptionen für die Option viewerpreferences *in PDF_begin_document( ) und PDF_end_document( )*

Option	Beschreibung
hidewindowui[1]	*(Boolean) Legt fest, ob die Fenstersteuerelemente von Acrobat ausgeblendet werden. Standardwert:* false
nonfullscreenpagemode	*(Schlüsselwort; nur relevant, wenn die Option* openmode *auf* fullscreen *gesetzt ist) Legt fest, wie das Dokument beim Beenden des Vollbildmodus angezeigt wird (Standardwert:* none*):* **bookmarks** *Lesezeichen-Fenster anzeigen* **thumbnails** *Seiten-Fenster anzeigen* **layers** *Ebenen-Fenster anzeigen* **none** *Nur Dokumentfenster anzeigen*
numcopies	*(Integer von 1-5, PDF 1.7) Anzahl der Kopien für den Druckdialog. Standardwert: abhängig vom Viewer*
picktraybypdfsize	*(Boolean; PDF 1.7; keine Auswirkung bei OS X) Bestimmt, ob der Papierschacht im Druckdialog anhand der PDF-Seitengröße ausgewählt wird. Standardwert: abhängig vom Viewer*
printscaling	*(Schlüsselwort; PDF 1.6) Option zur Seitenskalierung. Diese wird im Druckdialog für das Dokument automatisch ausgewählt. Standardwert:* appdefault **none** *Seitengröße wird nicht verändert; dies ist sinnvoll, wenn der Seiteninhalt in Originalgröße gedruckt werden soll.* **appdefault** *Die in Acrobat eingestellte Seitenskalierung wird verwendet.*
printpagerange	*(Liste aus Integer-Paaren; PDF 1.7) Seitennummern für den Druckdialog; jedes Paar bezeichnet die Start- und Endseite eines zu druckenden Seitenbereichs (erste Seite ist 1). Standardwert: abhängig vom Viewer*
printarea **printclip** **viewarea** **viewclip**	*(Schlüsselwort; für PDF/X sind nur* media *und* bleed *zulässig) Schlüsselwort zur Auswahl einer Seitengrößenangabe (Box), die den Seitenbereich beschreibt, der beim Betrachten des Dokuments am Bildschirm oder beim Drucken angezeigt oder beschnitten wird. Acrobat ignoriert diese Einstellung zwar, sie kann aber für andere Anwendungen sinnvoll sein. Unterstützte Schlüsselwörter (Standardwert:* crop*):* **art** *ArtBox wird verwendet.* **bleed** *BleedBox wird verwendet.* **crop** *CropBox wird verwendet.* **media** *MediaBox wird verwendet.* **trim** *TrimBox wird verwendet.*

1. Ab Acrobat 8 wird die Kombination von hidemenubar*,* hidetoolbar *und* hidewindowui *nicht mehr unterstützt. Das bedeutet, dass alle Elemente der Bedienoberfläche versteckt werden. Die Menüleiste bleibt weiterhin sichtbar, wenn alle drei Elemente auf* hidden *gesetzt sind.*

3.2 PDF-Dokumente aus dem Speicher holen

Wenn ein nicht leerer Parameter *filename* an *PDF_begin_document( )* übergeben wurde, schreibt PDFlib PDF-Dokumente in die angegebene Datei auf dem Datenträger. Alternativ dazu können PDF-Dokumentdaten im Arbeitsspeicher generiert werden, wenn der Parameter *filename* leer ist. In diesem Fall müssen die PDF-Dokumentdaten mit *PDF_get_buffer( )* aus dem Speicher geholt werden. Dies ist besonders nützlich beim Versenden von PDF von einem Webserver.

C++ ***const char *get_buffer(long *size)***

Java C# ***byte[ ] get_buffer( )***

Perl PHP ***string get_buffer( )***

C ***const char * PDF_get_buffer(PDF *p, long *size)***

Holt den Inhalt des PDF-Ausgabepuffers.

size (Nur C- und C++-Sprachbindung) C-Zeiger auf eine Speicherstelle, an der die Länge der zurückgegebenen Daten in Bytes abgelegt wird.

Rückgabe Ein Puffer mit binären PDF-Daten zur Weiterverarbeitung durch den Client. Die Funktion gibt einen sprachspezifischen Datentyp für Binärdaten zurück. Der zurückgegebene Puffer muss vom Client vor dem Aufruf anderer PDFlib-Funktionen verwendet werden.

Details Diese Funktion holt den gesamten Puffer mit den PDF-Daten oder einen Teil davon. Wird diese Funktion zwischen Seitenbeschreibungen aufgerufen, gibt sie die bislang generierten PDF-Daten zurück. Werden die PDF-Daten direkt im Speicher erzeugt, muss sie mindestens nach *PDF_end_document( )* aufgerufen werden und gibt dann den Rest des PDF-Dokuments zurück. Sie kann aber auch früher aufgerufen werden, um nur einen Teil der Daten abzuholen. Wird diese Funktion nur ein einziges Mal, und zwar nach *PDF_end_document( )* aufgerufen, enthält der Rückgabepuffer garantiert das vollständige PDF-Dokument in einem Stück.

Da die PDF-Ausgabe binäre Zeichen enthält, muss die Client-Software auf nicht druckbare Zeichen inklusive Null vorbereitet sein.

Gültigkeit *object*, *document*, also nach *PDF_end_page_ext( )* und vor *PDF_begin_page_ext( )* oder nach *PDF_end_document( )* und vor *PDF_delete( )*. Diese Funktion darf nur verwendet werden, wenn ein leerer Dateiname an *PDF_begin_document( )* übergeben wurde.
Wurde die Option *linearize* in *PDF_begin_document( )* auf *true* gesetzt, beschränkt sich der Gültigkeitsbereich auf *object*. Das bedeutet, dass diese Funktion nur nach *PDF_end_document( )* aufgerufen werden kann.

Bindungen C und C++: Der *size*-Parameter wird nur von C- und C++-Clients verwendet.

COM: Die meisten COM-Clients benutzen einen Variantentyp für den Pufferinhalt. Bei JavaScript mit COM ist es nicht möglich, die Länge des zurückgegebenen Varianten-Arrays abzufragen. Bei anderen Sprachbindungen als COM ist dies möglich.

Andere Sprachbindungen: Ein Objekt passender Länge wird zurückgegeben, und der *size*-Parameter muss weggelassen werden.

3.3 Seitenfunktionen

C++ Java C# ***void begin_page_ext(double width, double height, String optlist)***

Perl PHP ***begin_page_ext(float width, float height, string optlist)***

C ***void PDF_begin_page_ext(PDF *p, double width, double height, const char *optlist)***

Fügt unter Anwendung verschiedener Optionen eine neue Seite zum Dokument hinzu.

width, height Die Parameter *width* und *height* geben die Größe der neuen Seite in Punkt an (oder in *user units*, falls die Option *userunits* angegeben wurde). Sie können von den gleichnamigen Optionen überschrieben werden (in diesem Fall kann für diese Parameter der Dummy-Wert 0 verwendet werden). Eine Liste üblicher Seitenmaße finden Sie in Tabelle 3.8. Im PDFlib-Tutorial finden Sie eine Liste mit Seitenmaß-Begrenzungen für verschiedene Acrobat-Versionen. Für weitere Informationen siehe auch Tabelle 3.9 (Optionen *width* und *height)*.

Tabelle 3.8 Übliche Seitenmaße in Punkt[1]

Format	Breite	Höhe	Format	Breite	Höhe	Format	Breite	Höhe
A0	2380	3368	A4	595	842	letter	612	792
A1	1684	2380	A5	421	595	legal	612	1008
A2	1190	1684	A6	297	421	ledger	1224	792
A3	842	1190				11 x 17	792	1224

1. Informationen über ISO-, japanische und US-amerikanische Standardformate finden Sie unter: www.cl.cam.ac.uk/~mgk25/iso-paper.html

optlist Optionsliste mit Seitenoptionen gemäß Tabelle 3.9. Diese Optionen haben niedrigere Priorität als gleichnamige Optionen in *PDF_end_page_ext( )*:
action, artbox, associatedfiles, bleedbox, blocks, cropbox, defaultcmyk, defaultgray, defaultrgb, duration, group, height, label, mediabox, metadata, pagenumber*, rotate, separationinfo, taborder, topdown, transition,* transparencygroup*, trimbox, userunit, viewports, width*

Details Diese Funktion setzt für die neue Seite alle Parameter für Text, Grafik und Farbzustand auf die Standardwerte zurück und erstellt ein Koordinatensystem gemäß der Option *topdown*.

PDF/VT Die folgenden Optionen sind nicht erlaubt: *group, pagenumber.*

Gültigkeit *document;* mit dieser Funktion beginnt der Gültigkeitsbereich *page*; diese Funktion muss immer paarweise mit *PDF_end_page_ext( )* aufgerufen werden.

C++ Java C# **void end_page_ext(String optlist)**

Perl PHP **end_page_ext(string optlist)**

C **void PDF_end_page_ext(PDF *p, const char *optlist)**

Beendet eine Seite unter Anwendung verschiedener Optionen.

optlist Optionsliste gemäß Tabelle 3.9. Die Optionen in *PDF_end_page_ext( )* haben Vorrang vor den gleichnamigen Optionen in *PDF_begin_page_ext( )*. Folgende Optionen können verwendet werden:
action, artbox, associatedfiles, bleedbox, blocks, cropbox, defaultcmyk, defaultgray, defaultrgb, duration, group, height, label, mediabox, metadata, rotate, taborder, transition, transparencygroup, trimbox, userunit, viewports, width.

Gültigkeit *page;* mit dieser Funktion endet der Gültigkeitsbereich *page*; diese Funktion muss immer paarweise mit *PDF_begin_page_ext( )* aufgerufen werden. Im Modus Tagged PDF müssen alle Inline- und Pseudo-Elemente vor dem Aufruf dieser Funktion geschlossen werden.

Tabelle 3.9 Seitenoptionen für PDF_begin_page_ext( ) und PDF_end_page_ext( )

Option	Beschreibung
action	*(Aktionsliste; nicht für PDF/A) Liste mit Seitenaktionen für ein oder mehrere der folgenden Ereignisse (Standardwert: leere Liste):* **open** *Aktionen beim Öffnen der Seite* **close** *Aktionen beim Schließen der Seite*
artbox **bleedbox** **cropbox**	*(Rechteck) Bestimmt die ArtBox, BleedBox bzw. CropBox für die aktuelle Seite. Die Koordinaten werden im Standardkoordinatensystem festgelegt. Standardwert: keine ArtBox, BleedBox bzw. CropBox*
associatedfiles[1]	*(Liste aus Asset-Handles; nur für PDF 2.0 und PDF/A-3) Asset-Handles für assoziierte Dateien gemäß PDF/A-3. Die Dateien müssen mit PDF_load_asset( ) geladen werden und* `type=attachment` *haben.*
blocks	*(POCA-Container-Handle; kann an PDF_begin_page_ext( ) oder PDF_end_page_ext( ), aber nicht an beide Funktionen für dieselbe Seite übergeben werden; nur für PPS) Handle für einen Dictionary-Container, der mit PDF_poca_new( ) erzeugt wurde. Er enthält PDFlib-Blockdefinitionen für den PDFlib Personalization Server (PPS). Die angegebenen Blöcke werden an die Seite angehängt. Das Dictionary muss mit der Option* `usage=blocks` *erzeugt worden sein. Standardwert: keine Blöcke*
defaultcmyk **defaultgray** **defaultrgb**	*(ICC-Handle oder Schlüsselwort) Setzt anhand des übergebenen ICC-Profil-Handles einen Standardfarbraum für Graustufen, RGB oder CMYK für die Seite. Die Option* `defaultrgb` *unterstützt auch das Schlüsselwort* `srgb`**.** *Der angegebene Farbraum wird zur Umsetzung geräteabhängiger Graustufen, RGB oder CMYK auf der Seite verwendet (nicht für den Inhalt von Templates auf der Seite).*
duration	*(Float) Setzt für die aktuelle Seite deren Anzeigedauer in Sekunden, sofern* `openmode=fullscreen` *gesetzt ist (siehe Tabelle 3.1). Standardwert: 1*
group[1]	*(String; erforderlich, wenn das Dokument Seitengruppen enthält, aber sonst nicht zulässig; nicht zulässig im PDF/VT-Modus oder wenn eine Document Part Hierarchy erzeugt wird) Name der Seitengruppe, zu der die Seite gehören soll. Dieser Name kann verwendet werden, um Seiten in einer Gruppe zusammenzufassen oder sie mit PDF_resume_page( ) zu adressieren. Der Gruppenname muss mit der Option* `groups` *in PDF_begin_document( ) definiert worden sein.*

Tabelle 3.9 Seitenoptionen für PDF_begin_page_ext( ) und PDF_end_page_ext( )

Option	Beschreibung
height	*(Float oder Schlüsselwort; nicht zulässig, wenn die Option* topdown *gleich* true *ist) Größe der neuen Seite in Punkt (oder user units, falls die Option* userunits *gesetzt wurde). Für Seiten im Querformat setzen Sie* width > height *oder verwenden die Option* rotate*. PDFlib erzeugt anhand von* width *und* height *die MediaBox der Seite. Diese lässt sich auch explizit mit der Option* mediabox *setzen. Die Optionen* width *und* height *überschreiben die gleichnamigen Parameter.* *Folgende symbolische Seitenformatnamen können durch Anfügen von* .width *oder* .height *als Schlüsselwörter verwendet werden (zum Beispiel* a4.width *oder* a4.height*):* a0, a1, a2, a3, a4, a5, a6, b5, letter, legal, ledger, 11x17
label	*(Optionsliste) Optionsliste gemäß Tabelle 3.6 zur Festlegung von symbolischen Seitennamen. Der Seitenname wird als Seitenbezeichnung in der Statusleiste von Acrobat angezeigt. Das festgelegte Nummerierungsschema wird für die aktuelle und alle nachfolgenden Seiten verwendet, bis es explizit geändert wird. Die Kombination aus* style/prefix/start *-Werten muss innerhalb des Dokuments eindeutig sein.*
mediabox	*(Rechteck; nicht zulässig, wenn die Option* topdown *gleich* true *ist) Ändert die MediaBox für die aktuelle Seite. Die Koordinaten werden im Standardkoordinatensystem festgelegt. Die MediaBox wird standardmäßig anhand der Parameter* width *und* height *angelegt. Die Option* mediabox *überschreibt die Optionen bzw. Parameter* width *und* height*.*
metadata	*(Optionsliste) Metadaten zur Seite (siehe Abschnitt 14.2, »XMP-Metadaten«, Seite 277).*
pagenumber[1]	*(Integer; nicht zulässig im PDF/VT-Modus oder wenn eine Document Part Hierarchy erzeugt wird) Ist diese Option auf den Wert* n *gesetzt, so wird die Seite vor der bereits vorhandenen Seite* n *in der mit der Option* group *festgelegten Seitengruppe eingefügt (oder im Dokument, wenn keine Seitengruppen existieren). Ist diese Option nicht angegeben, wird die Seite am Ende der Gruppe eingefügt.*
rotate	*(Integer) Legt die Seitendrehung fest. Die Drehung wirkt sich auf die Anzeige der Seite aus, lässt aber das Koordinatensystem unverändert. Mögliche Werte sind 0, 90, 180, 270. Standardwert: 0*
separation-info[1]	*(Optionsliste) Optionsliste mit Angaben zur Farbseparation der aktuellen Seite. Diese Angaben werden von Acrobat ignoriert, sind aber in Fremdsoftware unter Umständen nützlich, um separierte Seiten in einem Workflow mit Vorabseparation zu erkennen und korrekt anzuzeigen.* **pages** *(Integer; erforderlich für die erste von mehreren separierten Seiten eines Satzes, aber nicht zulässig auf den Folgeseiten) Anzahl der Seiten, die zu einem Satz von separierten Seiten gehören, der die Farbdaten einer Farbseite enthält. Alle Seiten des Satzes müssen in der Datei aufeinanderfolgen.* **spotname** *(String; erforderlich, sofern nicht* spotcolor *übergeben wurde) Name der Farbe für die aktuelle Seite.* **spotcolor** *(Schmuckfarben-Handle) Farben-Handle für die Farbe der aktuellen Seite.*
taborder	*(Schlüsselwort; PDF 1.5; in PDF/UA ist nur* structure *erlaubt) Legt die Tabulatorreihenfolge für Formularfelder und Anmerkungen auf der Seite fest (Standardwert:* structure *im Tagged-PDF-Modus ab PDF 1.5; anderenfalls:* none*):* **column** *Formularfelder und Anmerkungen werden spaltenweise von oben nach unten durchlaufen, wobei die Spalten gemäß der Unteroption* direction *der Option* viewerpreferences *von PDF_begin/end_document( ) angeordnet sind.* **none** *Die Tabulatorreihenfolge ist nicht definiert.* **structure** *Formularfelder und Anmerkungen werden in der Reihenfolge durchlaufen, in der sie im Strukturbaum erscheinen.* **row** *Formularfelder und Anmerkungen werden zeilenweise von oben nach unten durchlaufen, wobei die Richtung, in der eine Zeile durchlaufen wird, durch die Unteroption* direction *der Option* viewerpreferences *von PDF_begin/end_document( ) festgelegt ist.*
topdown[1]	*(Boolean) Bei* topdown=true *wird zu Beginn einer Seite der Ursprung des Koordinatensystems in die linke obere Ecke der Seite gelegt, und die y-Koordinaten wachsen nach unten; anderenfalls wird das Standardkoordinatensystem verwendet. Standardwert:* false

Tabelle 3.9 Seitenoptionen für PDF_begin_page_ext( ) und PDF_end_page_ext( )

Option	Beschreibung
transition	*(Schlüsselwort) Bestimmt für die aktuelle Seite den Seitenübergang, der für Präsentationen als Spezialeffekt bei der Anzeige des PDFs im Vollbildmodus von Acrobat verwendet wird, sofern* `openmode=fullscreen` *gesetzt ist (siehe Tabelle 3.1). Standardwert:* `replace`
	split *Die Seite wird wie ein Vorhang aufgezogen und gibt die nächste frei.*
	blinds *Split-Effekt mit mehreren Linien, die den Eindruck einer Jalousie erwecken*
	box *Ein Rechteck vergrößert sich und legt die neue Seite frei.*
	wipe *Eine einzelne Linie wischt über die alte Seite und legt dabei die neue frei.*
	dissolve *Die alte Seite löst sich mosaikartig auf und legt dabei die neue frei.*
	glitter *Wie bei* `dissolve`*, nur sind die Mosaiksteinchen nicht gleichmäßig verteilt, sondern bewegen sich von einer Bildschirmseite zur gegenüberliegenden.*
	replace *Die neue Seite ersetzt die alte ohne Übergangseffekt.*
	fly *(PDF 1.5) Die neue Seite fliegt in die alte hinein.*
	push *(PDF 1.5) Die neue Seite schiebt die alte hinaus.*
	cover *(PDF 1.5) Die neue Seite gleitet in den Bildschirm hinein und bedeckt die alte Seite.*
	uncover *(PDF 1.5) Die alte Seite gleitet aus dem Bildschirm heraus und deckt die neue Seite auf.*
	fade *(PDF 1.5) Die alte Seite verblasst und gibt die Sicht auf die neue Seite frei.*
transparency-group	*(Optionsliste oder Schlüsselwort; nicht für PDF/A-1 und PDF/X-1/3; Einschränkungen für PDF/A-2/3 und PDF/X-4/5) Erzeugt eine Transparenzgruppen für die Seite. Folgende Schlüsselwörter werden unterstützt (Standardwert:* `auto`*):*
	auto *Enthält eine Seite, eine importierte Seite, eine Grafik oder ein Template transparente Objekte, wird automatisch die Option* `transparencygroup` *mit einem geeigneten Farbraum erzeugt; anderenfalls wird keine Transparenzgruppe erzeugt.*
	none *(Bei PDF/A-2/3 ohne Druckausgabebedingung unzulässig, wenn Transparenz auf der Seite verwendet wird) Es wird keine Transparenzgruppe für die Seite erzeugt.*
	Mit den folgenden Unteroptionen kann explizit eine Transparenzgruppe erzeugt werden:
	colorspace *(Schlüsselwort oder ICC-Profil-Handle; erforderlich für PDF/A-2/3 ohne Druckausgabebedingung, wenn Transparenz auf der Seite verwendet wird) Legt den Farbraum der Transparenzgruppe fest (Standardwert:* `none`*):*
	DeviceCMYK *PDF/A-2/3 und PDF/X-4/5: nur erlaubt mit einer CMYK-Druckausgabebedingung oder wenn die Option* `defaultcmyk` *angegeben wurde.*
	DeviceGray *PDF/A-2/3 und PDF/X-4/5: nur erlaubt mit einer Graustufen- oder CMYK-Druckausgabebedingung oder wenn die Option* `defaultgray` *angegeben wurde.*
	DeviceRGB *PDF/A-2/3 und PDF/X-4/5: nur erlaubt mit einer RGB-Druckausgabebedingung oder wenn die Option* `defaultrgb` *angegeben wurde.*
	none *(Bei PDF/A-2/3 ohne Druckausgabebedingung unzulässig, wenn Transparenz auf der Seite verwendet wird) Für die Transparenzgruppe wird kein Farbraum ausgewiesen.*
	srgb *Schlüsselwort zur Auswahl des sRGB-Farbraums*
	isolated *(Boolean) Legt fest, ob die Transparenzgruppe isoliert ist. Standardwert:* `false`
	knockout *(Boolean) Legt fest, ob die Transparenzgruppe eine Knockout-Gruppe ist. Standardwert:* `false`
trimbox	*(Rechteck) Bestimmt die TrimBox für die aktuelle Seite. Die Koordinaten werden im Standardkoordinatensystem festgelegt. Standardwert: keine TrimBox*

Tabelle 3.9 Seitenoptionen für PDF_begin_page_ext( ) und PDF_end_page_ext( )

Option	Beschreibung
userunit	*(Float oder Schlüsselwort; PDF 1.6) Zahl zwischen 1 und 75 000, die die Größe einer »user unit« in Punkt bestimmt, oder eines der Schlüsselwörter* `mm`*,* `cm` *oder* `m`*, das in die entsprechende Größe umgerechnet wird. User units ändern den Seiteninhalt nicht; sie stellen lediglich einen Hinweis an Acrobat dar, der beim Drucken der Seite oder Einsatz der Messwerkzeuge berücksichtigt wird. Standardwert: 1 (d.h. eine Einheit entspricht einem Punkt)*
viewports	*(Liste von Optionslisten; PDF 1.7ext3) Legt einen oder mehrere georeferenzierte Bereiche (Viewports) auf der Seite fest; für weitere Informationen siehe Abschnitt 12.7, »Features für Geodaten«, Seite 258.* *Viewports erlauben die Verwendung verschiedener Georeferenzen in unterschiedlichen Bereichen auf einer Seite (festgelegt mit der Option* `georeference`*), zum Beispiel bei mehreren Karten. Die Reihenfolge der Optionslisten in der* `viewports`*-Liste ist ausschlaggebend bei übereinanderliegenden Viewports: der letzte Viewport, der einen Punkt enthält, wird für diesen Punkt verwendet.*
width	*(Float oder Schlüsselwort; nicht zulässig, wenn die Option* `topdown` *gleich* `true` *ist) Siehe Option* `height`*.*

1. Nur für PDF_begin_page_ext( )

C++ Java C# ***void suspend_page(String optlist)***

Perl PHP ***suspend_page(string optlist)***

C ***void PDF_suspend_page(PDF *p, const char *optlist)***

Unterbricht die Ausgabe der aktuellen Seite. Sie kann später wieder aufgenommen werden.

optlist Optionsliste für zukünftige Verwendung.

Details Der komplette Grafikzustand (Grafik, Farbe, Text usw.) sowie der Ebenenzustand der aktuellen Seite werden intern gespeichert. Soll mit der Ausgabe fortgefahren werden, kann sie mit *PDF_resume_page( )* wieder aufgenommen werden. Wurde die Ausgabe einer Seite unterbrochen, muss sie erst wieder aufgenommen werden, bevor die Seite geschlossen werden kann.

Gültigkeit *page;* mit dieser Funktion beginnt der Gültigkeitsbereich *document*. Diese Funktion muss immer paarweise mit *PDF_resume_page( )* auftreten. Im Modus Tagged PDF müssen alle Inline- und Pseudo-Elemente vor dem Aufruf dieser Funktion geschlossen werden.

C++ Java C# ***void resume_page(String optlist)***

Perl PHP ***resume_page(string optlist)***

C ***void PDF_resume_page(PDF *p, const char *optlist)***

Nimmt eine unterbrochene Seitenausgabe wieder auf, um weiteren Inhalt hinzuzufügen.

optlist Optionsliste gemäß Tabelle 3.10. Folgende Optionen können verwendet werden: *group, pagenumber.*

Details Die Seitenausgabe muss mit *PDF_suspend_page( )* unterbrochen worden sein. Die Ausgabe wird wieder aufgenommen, um weitere Inhalte anzufügen. Wurde die Ausgabe einer Seite unterbrochen, muss sie erst wieder aufgenommen werden, um die Seite schließen zu können, auch wenn keine weiteren Inhalte hinzukommen.

Beachten Sie im Modus Tagged PDF, dass bei Wiederaufnahme einer Seite ihre Strukturelemente nicht wiederhergestellt werden. Stattdessen wird das Element, das beim Aufruf von *PDF_resume_page( )* aktiv war, zum aktuellen Element für die folgenden Seiteninhalte. Sie sollten *PDF_activate_item( )* verwenden, um ein bestimmtes Strukturelement auf der Seite zum übergeordneten Element für nachfolgend erzeugte Inhalte zu machen.

Gültigkeit *document;* mit dieser Funktion beginnt der Gültigkeitsbereich *page*; diese Funktion muss immer paarweise mit *PDF_suspend_page( )* aufgerufen werden.

Tabelle 3.10 Optionen für PDF_resume_page( )

Option	Beschreibung
group	*(String; erforderlich, wenn das Dokument Seitengruppen enthält, sonst nicht zulässig) Name der Seitengruppe der wieder aufgenommenen Seite. Der Gruppenname muss mit der Option* `groups` *in PDF_begin_document( ) definiert worden sein.*
pagenumber	*(Integer) Nummer der Seite in der Seitengruppe, die mit der Option* `group` *festgelegt wurde (oder Nummer der Seite im Dokument, wenn keine Seitengruppen vorhanden sind), die wieder aufgenommen wird. Fehlt diese Option, wird die letzte Seite der Gruppe wieder aufgenommen.*

3.4 Ebenen

Cookbook *Ein vollständiges Codebeispiel hierzu finden Sie im Cookbook-Topic* `graphics/starter_layer`.

C++ Java C# ***int define_layer(String name, String optlist)***

Perl PHP ***int define_layer(string name, string optlist)***

C ***int PDF_define_layer(PDF *p, const char *name, int len, const char *optlist)***

Erzeugt eine neue Ebenendefinition (ab PDF 1.5).

name (Hypertext-String) Name der Ebene

len (Nur C-Bindung) Länge von *name* (in Bytes). Ist *len* gleich *0*, muss ein null-terminierter String übergeben werden.

optlist Optionsliste mit Ebenen-Einstellungen:
- Allgemeine Optionen: *hypertextencoding* und *hypertextformat* (siehe Tabelle 2.3)
- Optionen zur Steuerung von Ebenen gemäß Tabelle 3.11:
  creatorinfo, defaultstate, initialexportstate, initialprintstate, initialviewstate, intent, language, onpanel, pageelement, printsubtype, removeunused, zoom.

Rückgabe Ein Ebenen-Handle, das in Aufrufen von *PDF_begin_layer( )* und *PDF_set_layer_dependency( )* innerhalb des umgebenden Gültigkeitsbereichs *document* verwendet werden kann.

Details PDFlib gibt eine Warnung aus, wenn eine Ebene zwar definiert, aber im Dokument nicht benutzt wurde. Ebenen, die auf mehreren Seiten vorkommen, sollten nur einmal definiert werden (z.B. vor dem Anlegen der ersten Seite). Denn wird *PDF_define_layer( )* mehrmals auf mehreren Seiten aufgerufen, summieren sich die Ebenendefinitionen (selbst wenn sie denselben Namen haben), und dies ist in der Regel nicht erwünscht.

PDF/A PDF/A-1: Diese Funktion darf nicht aufgerufen werden.
PDF/A-2/3: Einige Optionen sind eingeschränkt.

PDF/X PDF/X-1/2/3: Diese Funktion darf nicht aufgerufen werden.
PDF/X-4/5: Einige Optionen sind eingeschränkt.

PDF/UA Einige Optionen sind eingeschränkt.

Gültigkeit beliebig, außer *object*

Tabelle 3.11 Optionen für PDF_define_layer( )

Option	Beschreibung
creatorinfo	*(Optionsliste; nicht für PDF/A-2/3, PDF/X-4/5 und PDF/UA) Beschreibt den Inhalt und die zu dessen Erstellung benutzte Anwendung. Zur Verwendung dieser Option sind die beiden folgenden Einträge erforderlich:* ***creator*** *(Hypertext-String) Name der zur Ebenen-Erstellung benutzten Anwendung* ***subtype*** *(String) Art des Inhalts, zum Beispiel* `Artwork` *oder* `Technical`.
defaultstate	*(Boolean) Legt fest, ob die Ebene standardmäßig sichtbar ist oder nicht. Standardwert:* `true`
initial-exportstate	*(Boolean; nicht für PDF/A-2/3, PDF/X-4/5 und PDF/UA) Legt den für die Ebene empfohlenen Exportzustand fest. Bei* `true` *bezieht Acrobat die Ebene in die Konvertierung oder den Export in ältere PDF-Versionen oder andere Dokumentformate mit ein. Standardwert:* `true`

Tabelle 3.11 Optionen für PDF_define_layer( )

Option	Beschreibung
initial-printstate	*(Boolean; nicht für PDF/A-2/3, PDF/X-4/5 und PDF/UA) Legt den für die Ebene empfohlenen Druckzustand fest. Bei* `true` *bezieht Acrobat die Ebene in das Drucken des Dokuments mit ein. Standardwert:* `true`
initial-viewstate	*(Boolean; nicht für PDF/A-2/3, PDF/X-4/5 und PDF/UA) Legt den für die Ebene empfohlenen Anzeigezustand fest. Bei* `true` *zeigt Acrobat die Ebene beim Öffnen des Dokuments mit an. Standardwert:* `true`
intent	*(Schlüsselwort) Beabsichtigte Verwendung der Grafik:* `View` *oder* `Design`*. Standardwert:* `View`
language	*(Optionsliste; nicht für PDF/A-2/3, PDF/X-4/5 und PDF/UA) Legt die Sprache für die Ebene fest:* **lang** *(String; erforderlich) Sprache und eventuell Ländercode im Format, das für die Option* `lang` *in Tabelle 3.1 beschrieben wird* **preferred** *(Boolean) Bei* `true` *wird diese Ebene auch verwendet, wenn ihre Spracheinstellung nur teilweise zur Systemsprache passt. Standardwert:* `false`
onpanel	*(Boolean; nicht für PDF/A-2/3, PDF/X-4/5 und PDF/UA) Bei* `false` *ist der Ebenen-Name im Ebenen-Fenster von Acrobat nicht sichtbar und kann somit vom Benutzer nicht verändert werden. Standardwert:* `true`
pageelement	*(Schlüsselwort; nicht für PDF/A-2/3, PDF/X-4/5 und PDF/UA) Gibt an, dass der Ebeneninhalt von der Seiteneinteilung/Paginierung abhängt:* `HF` *für Header/Footer (Kopf-/Fußzeile),* `FG` *für Foreground Image oder Graphic (Vordergrundbild oder -grafik),* `BG` *für Background Graphic (Hintergrundbild oder -grafik) und* `L` *für Logo.*
printsubtype	*(Optionsliste; nicht für PDF/A-2/3, PDF/X-4/5 und PDF/UA) Legt fest, ob die Ebene zum Druck vorgesehen ist:* **subtype** *(Schlüsselwort) Art des Ebeneninhalts:* `Trapping`, `PrintersMarks` *oder* `Watermark`. **printstate** *(Boolean) Bei* `true` *aktiviert Acrobat den Ebeneninhalt beim Drucken.*
removeunused	*(Boolean) Falls diese Option den Wert* `true` *hat und die Ebene nicht auf einer Seite benutzt wird, so wird sie nicht in der Ebenen-Liste dieser Seite aufgeführt. Eine Ebene gilt als benutzt, falls sie mindestens einmal auf dieser Seite an PDF_begin_layer( ) übergeben wurde. Standardwert:* `false`, *sofern die Ebene nicht mit* `listmode=visiblepages` *in eine Standardvariante aufgenommen wurde.*
zoom	*(Liste aus Float- oder Prozentwerten; nicht für PDF/A-2/3, PDF/X-4/5 und PDF/UA) Ein oder zwei Werte, die die Sichtbarkeit der Ebene abhängig vom Zoomfaktor festlegen (1.0 entspricht einem Zoomfaktor von 100 Prozent). Ein einzelner Wert wird als der maximale Zoomfaktor benutzt, bei dem die Ebene noch angezeigt wird; zwei Werte legen den minimalen und maximalen Zoomfaktor für die Ebenen-Anzeige fest. Mit dem Schlüsselwort* `maxzoom` *kann der größtmögliche Zoomfaktor angegeben werden.*

C++ Java C# ***void set_layer_dependency(String type, String optlist)***

Perl PHP ***set_layer_dependency(string type, string optlist)***

C ***void PDF_set_layer_dependency(PDF *p, const char *type, const char *optlist)***

Definiert die Beziehungen von Ebenen und Varianten (ab PDF 1.5).

type Art der Beziehung oder Abhängigkeit gemäß Tabelle 3.12.

Tabelle 3.12 Beziehungen und Abhängigkeiten bei Ebenen

Typ	Beschreibung, Optionen
GroupAllOn	*Die Ebene in der Option* `depend` *wird eingeblendet, wenn alle Ebenen in der Option* `group` *sichtbar sind. Optionen für diesen Typ:* `depend`, `group`
GroupAnyOn	*Die Ebene in der Option* `depend` *wird eingeblendet, wenn irgendeine der Ebenen in der Option* `group` *sichtbar ist. Optionen für diesen Typ:* `depend`, `group`

Tabelle 3.12 Beziehungen und Abhängigkeiten bei Ebenen

Typ	Beschreibung, Optionen
GroupAllOff	*Die Ebene in der Option* depend *wird eingeblendet, wenn alle Ebenen in der Option* group *ausgeblendet sind. Optionen für diesen Typ:* depend, group
GroupAnyOff	*Die Ebene in der Option* depend *wird eingeblendet, wenn irgendeine Ebene in der Option* group *ausgeblendet ist. Optionen für diesen Typ* depend, group
Lock	*(PDF 1.6) Die Ebenen in der Option* group *werden gesperrt, d.h. dass sich ihr Status in Acrobat nicht interaktiv ändern lässt. Optionen für diesen Typ:* group
Parent	*Definiert eine hierarchische Beziehung zwischen der Ebene in der Option* parent *und den Ebenen in der Option* children*. Wird die übergeordnete Ebene auf unsichtbar gesetzt, werden auch die untergeordneten Ebenen unsichtbar. Jede Ebene darf höchstens eine* parent*-Ebene haben. Optionen für diesen Typ:* children, parent
Radiobtn	*Legt eine Beziehung in der Art eines Optionsfelds (Radiobutton) zwischen den Ebenen in der Option* group *fest. Das bedeutet, dass jeweils höchstens eine Ebene der Gruppe eingeblendet ist; dies ist insbesondere bei mehrsprachigen Ebenen nützlich. Optionen für diesen Typ:* group
Title	*Die in der Option* parent *angegebene Ebene dient nicht zur unmittelbaren Steuerung des Seiteninhalts, sondern als hierarchischer Separator für die Ebenen in der Option* children*. Optionen für diesen Typ:* children, parent
Variant	*Legt eine Dokumentvariante fest, das heißt, eine Kombination aus mehreren Ebenen. In späteren Aufrufen von PDF_set_layer_dependency( ) kann die Option* variantname *erneut übergeben werden, um die Abhängigkeitsregeln für diese Konfiguration festzulegen. Optionen für diesen Typ:* basestate, defaultvariant, includelayers, invisiblelayers, visiblelayers

optlist Optionsliste für die Beziehungen zwischen Ebenen:

- Allgemeine Option: *hypertextencoding*(siehe Tabelle 2.3)
- Optionen für Ebenen-Abhängigkeiten gemäß Tabelle 3.13:
  basestate, children, createorderlist, defaultvariant, depend, group, includelayers, invisiblelayers, listmode, parent, variantname, visiblelayers.

Details Ebenen-Beziehungen legen die Darstellung von Ebenen-Namen im Ebenen-Navigationsfenster von Acrobat sowie die Sichtbarkeit einer oder mehreren Ebenen fest, wenn der Benutzer sie aktiviert oder deaktiviert.

Varianten bestehen aus einer festgelegten Kombination von Ebenen, um die Sicherheit bei der Produktion zu verbessern. Statt einzelner Ebenen kann der Benutzer lediglich eine Variante aktivieren oder deaktivieren. Enthält ein Dokument Varianten, werden in Acrobat 9 nicht die einzelnen Ebenen-Namen angezeigt, sondern nur die Namen der Ebenen-Varianten. Ebenen-Varianten werden nur in Acrobat 9 dargestellt, und nur für PDF/X-Dokumente. Acrobat X und höher stellt keine Ebenen-Varianten dar. Aus diesem Grund wird der Einsatz von Ebenen-Varianten nicht empfohlen.

Um eine Abhängigkeit bei Ebenen-Varianten festzulegen, bei denen nicht alle betroffenen Ebenen Teil derselben Variante sind, muss die Abhängigkeit festgelegt werden, bevor die Standard-Variante gesetzt wird.

PDF/A PDF/A-1: Diese Funktion darf nicht aufgerufen werden.
PDF/A-2/3: Einige Optionen sind eingeschränkt.

PDF/X PDF/X-1/2/3: Diese Funktion darf nicht aufgerufen werden.
PDF/X-4/5: Einige Optionen sind eingeschränkt.

Ebenen-Varianten waren im früheren Standard PDF/X-4:2008 erforderlich, direkte Steuerung der Ebenen (ohne Varianten) ist im Nachfolger PDF/X-4:2010 erlaubt und wird von PDFlib auch unterstützt.

PDF/UA Einige Optionen sind eingeschränkt.

Gültigkeit alle außer *object;* Beziehungen zwischen Ebenen sollten erst festgelegt werden, nachdem alle Ebenen definiert wurden.

Tabelle 3.13 Optionen für PDF_set_layer_dependency( )

Option	Beschreibung
basestate	*(Schlüsselwort; nur für* type=Variant*; nicht für PDF/A-2/3, PDF/X-4/5 und PDF/UA) Legt die Sichtbarkeit aller Ebenen fest, die nicht explizit in den Optionen* visiblelayers *und* invisiblelayers *angegeben wurden. Unterstützte Schlüsselwörter (Standardwert:* on*):* **on** *Alle Ebenen der ausgewählten Variante sind sichtbar.* **off** *Alle Ebenen der ausgewählten Variante sind unsichtbar.* **unchanged** *Der Status aller Ebenen bleibt für die ausgewählte Variante unverändert.*
children	*(Liste aus Ebenen-Handles; nur bei* type=Parent *oder* Title*) Eine oder mehrere Ebenen-Handles, die die der* parent*-Ebene untergeordneten Ebenen festlegen.*
createorder-list	*(Boolean; nur für* type=Variant *und* defaultvariant=true*) Bei* true *zeigt Acrobat die Namen aller Ebenen an. Der Wert* true *bedeutet folgendes (Standardwert:* true*):* ▸ *Acrobat 9 zeigt im Ebenen-Fenster vorhandene Ebenen-Varianten, jedoch keine Ebenen-Namen an. Preflight meldet PDF/X-4-Validierungsfehler für Dokumente mit* createorderlist=true*, da dies bei PDF/X-4:2008 nicht erlaubt ist.* ▸ *Acrobat X und höher zeigt im Ebenen-Fenster Ebenen-Namen, jedoch keine Ebenen-Varianten an und validiert erfolgreich Dokumente mit* createorderlist=true*, da dies bei PDF/X-4:2010 erlaubt ist.*
default-variant	*(Boolean; nur für* type=Variant*) Bei* true *handelt es sich bei der angegebenen Variante um die Standardvariante, das heißt, sie wird beim Öffnen des Dokuments aktiviert. Genau eine Variante muss als Standardvariante definiert werden. Standardwert:* false
depend	*(Ebenen-Handle; nur bei* type=GroupAllOn*,* GroupAnyOn*,* GroupAllOff *oder* GroupAnyOff*) Ebene, die von den Ebenen in der Option* group *gesteuert wird.*
group	*(Liste aus Ebenen-Handles; nur bei* type=GroupAllOn*,* GroupAnyOn*,* GroupAllOff*,* GroupAnyOff*,* Lock *und* Radiobtn*) Ein oder mehrere Ebenen-Handles, die eine Gruppe bilden. Bei* type=Lock *werden alle Ebenen in der Gruppe gesperrt.*
includelayers	*(Liste aus Ebenen-Handles; nur für* type=Variant*) Legt die Ebenen fest, die zu der Variante gehören. Standardwert: alle bislang im Dokument definierten Ebenen*
invisiblelayers	*(Liste aus Ebenen-Handles; nur für* type=Variant*) Legt eine Liste von Ebenen fest, die anfänglich für die ausgewählten Varianten unsichtbar sind. Eine Ebene darf nicht gleichzeitig in der Variantenliste* visiblelayers *und* invisiblelayers *aufgeführt werden. Bei* defaultvariant=true *überschreibt diese Option die Option* defaultstate *von PDF_define_layer( ). Standardwert (abhängig von der Option* basestate*): alle Ebenen in der Liste* includelayers *bei* basestate=off*; leere Liste bei* basestate=on
listmode	*(Schlüsselwort; nur für* type=Variant*) Legt fest, welche Ebenen-Namen im Ebenen-Fenster von Acrobat angezeigt werden. Unterstützte Schlüsselwörter (Standardwert:* visiblepages*):* **allpages** *Die Namen aller Ebenen auf allen Seiten werden angezeigt.* **visiblepages** *Die Namen aller Ebenen der aktuell sichtbaren Seite(n) werden angezeigt. Dies impliziert den Standardwert* removeunused=true *für alle zu der Variante gehörende Ebenen.* *Bei Acrobat wirkt sich dies nur aus, wenn* defaultvariant=true*.*
parent	*(Ebenen-Handle; nur bei* type=Parent *oder* Title*) Die Ebene, die den Ebenen in der Option* children *übergeordnet ist.*

Tabelle 3.13 Optionen für PDF_set_layer_dependency( )

Option	Beschreibung
variantname	*(Hypertext-String; erforderlich für* `type=Variant`*) Name der gewählten Variante. Bei* `type=Variant` *darf jeder Variantenname nur einmal angegeben werden. Standardwert, wenn* `type` *verschieden ist von* `Variant`*: die mit* `defaultvariant` *festgelegte Standardvariante*
visiblelayers	*(Liste aus Ebenen-Handles; nur für* `type=Variant`*) Legt eine Liste von Ebenen fest, die zunächst in der ausgewählten Variante sichtbar sind. Eine Ebene darf nicht gleichzeitig in einer Variantenliste* `visiblelayers` *und* `invisiblelayers` *aufgeführt sein. Bei* `defaultvariant=true` *überschreibt diese Option die Option* `defaultstate` *von PDF_define_layer( ). Standardwert (abhängig von der Option* `basestate`*): alle Ebenen in der Liste* `includelayers` *bei* `basestate=on`*; leere Liste bei* `basestate=off`

C++ Java C# ***void begin_layer(int layer)***
Perl PHP ***begin_layer(int layer)***
C ***void PDF_begin_layer(PDF *p, int layer)***

Beginnt eine Ebene für die nachfolgende Ausgabe auf der Seite (ab PDF 1.5).

layer Ebenen-Handle, das von *PDF_define_layer( )* zurückgegeben wurde.

Details Alle Inhalte, die nach diesem Aufruf und vor dem nächsten Aufruf von *PDF_begin_layer( )* oder *PDF_end_layer( )* auf der Seite ausgegeben werden, gehören zur angegebenen Ebene. Die Sichtbarkeit der Inhalte hängt von den Ebenen-Einstellungen ab.

Diese Funktion aktiviert die übergebene Ebene und deaktiviert dabei gegebenenfalls eine andere aktive Ebene.

Ebenen für Anmerkungen, Bilder, SVG-Grafiken, Templates und Formularfelder lassen sich mit der Option *layer* der jeweiligen Funktion steuern.

Gültigkeit *page*

C++ Java C# ***void end_layer( )***
Perl PHP ***end_layer( )***
C ***void PDF_end_layer(PDF *p)***

Deaktiviert alle aktiven Ebenen (ab PDF 1.5).

Details Inhalte, die nach diesem Aufruf auf der Seite ausgegeben werden, gehören keiner Ebene an. Alle Ebenen müssen am Seitenende geschlossen werden.

Um von Ebene A nach Ebene B zu wechseln, reicht ein Aufruf von *PDF_begin_layer( )* aus; Ebene A braucht nicht explizit mit *PDF_end_layer( )* geschlossen zu werden. *PDF_end_layer( )* ist nur notwendig, um unbedingte Inhalte (die stets sichtbar sind) zu erstellen und alle Ebenen am Seitenende zu schließen.

Gültigkeit *page*

4 Font- und Textfunktionen

4.1 Fontverarbeitung

C++ Java C# ***int load_font(String fontname, String encoding, String optlist)***

Perl PHP ***int load_font(string fontname, string encoding, string optlist)***

C ***int PDF_load_font(PDF *p, const char *fontname, int len, const char *encoding, const char *optlist)***

Sucht nach einem Font und bereitet ihn zur späteren Verwendung vor.

fontname (Name-String) Name des Fonts. Er kann alternativ über die Option *fontname* angegeben werden, die diesen Parameter überschreibt. Für weitere Informationen siehe Tabelle 4.2.

len (Nur C-Sprachbindung) Länge von *fontname* in Bytes. Ist *len* = 0, muss ein null-terminierter String übergeben werden.

encoding Name des Encodings. Er kann auch mit der Option *encoding* angegeben werden, die diesen Parameter überschreibt. Für weitere Informationen siehe Tabelle 4.2. Beachten Sie folgende Probleme bezüglich Encodings:

- Ein 8-Bit-Encoding wurde übergeben, aber der Font enthält keine passenden Glyphen für dieses Encoding bzw. ist ein Standard-CJK-Font.
- Das Encoding *builtin* wurde übergeben, aber der Font enthält kein internes Encoding. Dieses Problem tritt nur bei TrueType-Fonts auf.
- Eine vordefinierte CMap wurde übergeben, passt aber nicht zum Font.

optlist Optionsliste mit den folgenden Optionen:

- Allgemeine Option: *errorpolicy* (siehe Tabelle 2.1)
- Optionen zum Laden von Fonts gemäß Tabelle 4.2:

ascender, autosubsetting, capheight, descender, dropcorewidths, embedding, encoding, fallbackfonts, fontname, initialsubset, keepfont, keepnative, linegap, metadata, optimizeinvisible, preservepua, readfeatures, readkerning, readselectors, readshaping, replacementchar, simplefont, skipembedding, skipposttable, subsetlimit, subsetminsize, subsetting, unicodemap, vertical, xheight

Rückgabe Font-Handle zur späteren Verwendung in *PDF_info_font()*, in Textausgabe-Funktionen sowie in der Option *font* zur Textdarstellung. Die Funktion liefert den Fehlercode -1 (in PHP: 0) zurück oder löst eine Exception aus, wenn die angeforderte Font/Encoding-Kombination nicht geladen werden kann. Dies ist zum Beispiel dann der Fall, wenn ein Font, Metrikdaten oder eine Encoding-Datei nicht gefunden oder nicht zugeordnet werden konnte. Dieses Verhalten lässt sich mit der Option *errorpolicy* steuern.

Gibt die Funktion einen Fehlercode zurück, so kann die Fehlerursache mit *PDF_get_errmsg()* abgefragt werden. Anderenfalls kann der von dieser Funktion zurückgegebene Wert beim Aufruf anderer Fontfunktionen als Font-Handle verwendet werden. Der zurückgegebene Wert – das Font-Handle – hat für den Benutzer keinerlei inhaltliche Bedeutung. Das zurückgegebene Font-Handle ist gültig, bis der Font mit *PDF_close_font()* geschlossen wird. Das Beenden des Dokuments mit *PDF_end_document()* schließt alle geöffneten Font-Handle. dies ist jedoch nicht der Fall, wenn die Option *keepfont* überge-

ben wurde oder der Font im Gültigkeitsbereich *object* geladen wurde (das heißt, außerhalb eines Dokuments).

Details Diese Funktion bereitet einen Font zur späteren Verwendung vor.

Mehrfache Aufrufe: Ein erneuter Aufruf dieser Funktion mit dem gleichen Fontnamen, Encoding und gleichen Optionen liefert das Font-Handle des ersten Aufrufs zurück. Ausnahmen: wenn eine der folgenden Optionen im ersten, aber nicht im nachfolgenden Aufruf angegeben wurde, wird das zweite Font-Handle dennoch identisch mit dem ersten Font-Handle sein: *embedding, readkerning, replacementchar, fallbackfonts, metadata*.

Ebenso wird die Option *initialsubset* beim Vergleich der Fonts ignoriert. Wenn der Font zum Beispiel zunächst ohne und dann mit *initialsubset* geladen wurde, wird ein Handle für den ersten Font zurückgegeben und *initialsubset* bleibt ohne Auswirkung.

Der Versuch, einen Font erneut zu laden, schlägt fehl, wenn im ersten Aufruf *embedding=false* und im zweiten Aufruf *embedding=true* gesetzt ist. Diese Situation weist in der Regel auf ein Problem in der Anwendung hin.

Indirektes Laden von Fonts: zusätzlich zum expliziten Laden eines Fonts über *PDF_load_font( )* können mit einigen Funktionen (z.B. *PDF_add/create_textflow( )* oder *PDF_fill_textblock( )*) Fonts implizit geladen werden, für die der Fontname und das Encoding in einer Optionsliste angegeben wurden. Wenn der Font bereits früher geladen wurde, wird ein neues Font-Handle erstellt.

Manche Textausgabe-Features sind für bestimmte Encodings nicht verfügbar (siehe Tabelle 4.1).

In nicht Unicode-fähigen Sprachbindungen verhält sich die Option *textformat=auto* wie folgt (beachten Sie, dass in beiden Fällen alle UTF-Formate erlaubt sind):

- Wide-Character-Encodings: Text im geladenen Font wird im Textformat *utf16* erwartet (bei *encoding=glyphid* werden Surrogatwerte nicht interpretiert).
- Byte- und Multibyte-Encodings: Text im geladenen Font wird im Textformat *bytes* erwartet.

PDF/A Alle Fonts müssen eingebettet werden.

PDF/UA Alle Fonts müssen eingebettet werden.

PDF/X Alle Fonts müssen eingebettet werden.

Tabelle 4.1 Verfügbarkeit von PDFlib-Features für verschiedene Encodings

Feature	Unicode und Unicode CMaps	8-Bit-Encodings	ältere CMaps, cp936 usw.	glyphid
Textflow	*ja*	*ja*	*ja*[1]	*ja*
Glyphenersetzung	*ja*[2]	*ja*	*ja*[1]	–
Fallback-Fonts	*ja*[2]	*ja*	*ja*[1]	–
Shaping	*ja*[2]	–	*ja*[1]	*ja*
OpenType-Layout-Features	*ja*	–	*ja*[1]	*ja*

1. Dieses Feature ist für CJK-Fonts mit `keepnative=true` *nicht verfügbar.*
2. Dieses Feature ist für Standard-CJK-Fonts mit Unicode-CMaps oder `keepnative=true` *nicht verfügbar.*

Gültigkeit beliebig

Tabelle 4.2 Font-Optionen für PDF_load_font( ) und implizites Laden von Fonts

Option	Beschreibung
ascender	*(Integer zwischen -2048 und 2048) Die entsprechende typografische Eigenschaft wird auf den definierten Wert gesetzt. Gegebenenfalls im Font vorhandene Werte werden überschrieben. Dies ist nützlich, wenn der Font keine diesbezüglichen Informationen enthält (zum Beispiel Type-3-Fonts). Standardwert: der gegebenenfalls im Font vorhandene Wert, sonst ein Schätzwert (der mit PDF_info_font( ) abgefragt werden kann)*
autocidfont	*(Boolean) Veraltet und aufgrund von internen Änderungen am Font-Engine nicht länger funktionsfähig*
auto-subsetting	*(Boolean) Entscheidet dynamisch, ob Fontuntergruppen auf Basis der Optionen* `subsetlimit` *und* `subsetminsize` *sowie der tatsächlich genutzten Glyphen gebildet werden. Diese Option wird ignoriert, wenn die Option* `subsetting` *übergeben wird. Standardwert:* `true`
capheight	*(Integer zwischen -2048 und 2048) Siehe* `ascender` *oben.*
descender	*(Integer zwischen -2048 und 2048) Siehe* `ascender` *oben.*
dropcore-widths	*(Boolean; nicht unterstützt; wird für PDF/A, PDF/UA und PDF/X auf* `false` *gesetzt) Die Breiten für nicht eingebettete PDF-Standardfonts werden im generierten PDF nicht ausgegeben. Dies reduziert etwas die Größe der Ausgabedatei, kann aber bei manchen Zeichen zu einer falschem Textdarstellung führen. Es wird deshalb dringend empfohlen, diese Option auf dem Standardwert zu belassen. Standardwert:* `false`
embedding	*(Boolean; muss für PDF/A, PDF/UA und PDF/X gleich* `true` *sein; wird bei SING- und Type-3-Fonts ignoriert, da diese immer eingebettet werden) Steuert, ob der Font eingebettet wird. Soll ein Font eingebettet werden, muss neben den Metrikdaten auch die Zeichenbeschreibungsdatei vorhanden sein (dies spielt bei TrueType- und OpenType-Fonts keine Rolle). Die tatsächlichen Zeichenbeschreibungen werden dann in die PDF-Ausgabe aufgenommen. Wird ein Font nicht eingebettet, werden nur allgemeine Fontinformationen in die PDF-Ausgabe geschrieben.* *Standardwert: Im Allgemeinen* `false`*, aber* `true` *für TrueType-/OpenType-Fonts mit Encodings, die eine Konvertierung in einen CID-Font verlangen. PDFlib bettet solche Fonts zwar automatisch ein, dies lässt sich aber mit* `embedding=false` *verhindern. In diesem Fall muss der Font auf dem System installiert sein, auf dem die PDF-Dokumente betrachtet oder gedruckt werden.* *Die Option* `embedding=false` *wird ignoriert, wenn der gleiche Font bereits früher mit* `embedding=true` *geladen wurde. Das Verhalten beim Einbetten von Fonts mit unsichtbarem Text kann mit der Option* `optimizeinvisible` *geändert werden, selbst bei* `embedding=true`**.** *Die Fonteinbettung kann auch mit der Option* `skipembedding` *gesteuert werden.*
encoding	*(String; erforderlich für implizites Laden von Fonts, falls die Option* `Font` *zur Textdarstellung nicht angegeben wurde) Encoding, in dem Texteingabe für diesen Font interpretiert wird:* *Wide-Character-Encodings:* ► `unicode` *oder der Name einer Unicode-CMap* ► `Identity-H` *oder* `Identity-V` *für CID-Adressierung* ► `glyphid`*: alle Glyphen im Font können über ihre fontspezifische ID adressiert werden* *Byte- und Multibyte-Encodings:* ► *eines der vordefinierten 8-Bit-Encodings* `winansi`, `macroman`, `macroman_apple`, `ebcdic`, `ebcdic_37`, `pdfdoc`, `iso8859-X` *oder* `cpXXXX` *und ältere, nicht Unicode-fähige CMaps* ► *(nicht für Unicode-fähige Sprachbindungen)* `cp932`, `cp936`, `cp949` *oder* `cp950` *für CJK-Codepages* ► `host` *oder* `auto` *oder der Name eines benutzerdefinierten Encodings oder ein dem Betriebssystem bekanntes Encoding* ► `builtin` *zur Auswahl des fontinternen Encodings (meist für Symbolfonts);* *PDF_load_font( ): diese Option kann alternativ als Funktionsparameter übergeben werden.* *PDF_fill_textblock( ): diese Option ist erforderlich, es sei denn, der String im Parameter* `text` *ist leer und die Property* `defaulttext` *wird verwendet oder die Option* `font` *wurde übergeben.*

Tabelle 4.2 Font-Optionen für PDF_load_font( ) und implizites Laden von Fonts

Option	Beschreibung
fallbackfonts	*(Liste von Optionslisten gemäß Tabelle 4.3) Bestimmt einen oder mehrere Fallback-Fonts für den geladenen Font. Jeder Fallback-Font muss durch ein Font-Handle in der Unteroption* `font` *definiert werden oder durch geeignete Unteroptionen für das implizite Laden von Fonts. Fallback-Fonts werden bei manchen Kombinationen von Fonttypen und Encodings nicht unterstützt (siehe Tabelle 4.1).* *Bei* `glyphcheck=replace` *und wenn der Text ein Zeichen enthält, das nicht im 8-Bit-Encoding des Basisfonts enthalten ist, oder der Basisfont keine Glyphe für das Zeichen enthält oder die Glyphenersetzung mit der Unteroption* `forcechars` *erzwungen wird, sucht PDFlib in allen angegebenen Fallback-Fonts der Reihe nach nach einer Glyphe für dieses Zeichen. Wird eine passende Glyphe in einem der Fallback-Fonts gefunden, wird das Zeichen durch diese Glyphe dargestellt; anderenfalls wird die übliche Methode der Glyphenersetzung angewandt.*
fontname	*(Name-String; erforderlich für implizites Laden von Fonts außer für PDF_fill_textblock( ), wenn die Option* `font` *zur Textdarstellung nicht angegeben wurde) Richtiger Name oder Aliasname des Fonts (es wird nach Groß-/Kleinschreibung unterschieden). Dieser Name wird verwendet, um die Fontdaten zu finden. Bei Windows können Fontstilnamen durch Komma getrennt an den Fontnamen angehängt werden (für weitere Informationen siehe PDFlib-Tutorial). Wenn* `fontname` *mit dem Zeichen '@' beginnt, wird der Font in vertikaler Schreibrichtung angewandt.* *PDF_load_font( ): Der Fontname kann alternativ als Funktionsparameter angegeben werden.*
fontstyle	*(Schlüsselwort; veraltet) Steuert die Erzeugung künstlicher Fontstile. Mögliche Schlüsselwörter sind* `normal`, `bold`, `italic` *und* `bolditalic`. *Mit diesem Font erzeugter Text wird jeweils mit den Optionen zur Textdarstellung* `fakebold` *und/oder* `italicangle` *formatiert. Sofern kein anderer Wert für* `italicangle` *gesetzt wurde, wird -12 verwendet.* *Wird diese Option auf einen der PDF-Standardfonts angewandt, so wird ein entsprechender fetter, kursiver oder fettkursiver Fontschnitt gewählt und kein unechter Fontstil erzeugt. Ist kein geeigneter Font verfügbar (zum Beispiel um* `bold` *auf Times-Bold anzuwenden), so wird die Option ignoriert. Standardwert:* `normal`
initialsubset	*(Liste von Unichars oder Unicode-Bereichen oder Liste von Schlüsselwörtern; nur relevant bei* `embedding=true` *und* `subsetting=true`*) Legt die Unicode-Werte fest, für die Glyphen in die ursprüngliche Fontuntergruppe aufgenommen werden. Damit lässt sich die Größe von PDF-Dateien reduzieren. Dabei werden nach wie vor identische Untergruppen gebildet. dies erleichtert spätere Optimierungen nach dem konkatenieren mehrerer Dokumente. Die Unicode-Werte können explizit als Unichars oder Unicode-Bereiche oder implizit als Namen eines 8-Bit-Encodings angegeben werden. Unichars und Unicode-Bereiche haben Vorrang vor Encoding-Namen. Unterstützte Schlüsselwörter (Standardwert:* `empty`*):* ***empty*** *Die ursprüngliche Fontuntergruppe ist leer; der Inhalt der Untergruppe wird durch den Text im Dokument bestimmt.* ***beliebiger Name für 8-Bit-Encoding*** *Alle in dem Encoding gefundenen Unicode-Werte werden in die ursprüngliche Untergruppe aufgenommen. Glyphen für zusätzliche Zeichen werden der Untergruppe automatisch hinzugefügt, wenn der Text im Dokument oder die Textoptionen* `features` *und* `shaping` *dies erfordern.*
keepfont	*(Boolean; bei Type-3-Fonts unzulässig) Bei* `false` *wird der Font automatisch in PDF_end_document( ) gelöscht. Bei* `true` *kann der Font auch in nachfolgenden Dokumenten verwendet werden, bis PDF_close_font( ) aufgerufen wird. Standardwert:* `true`*, wenn PDF_load_font( ) im Gültigkeitsbereich* `object` *aufgerufen wird; sonst* `false`

Tabelle 4.2 Font-Optionen für PDF_load_font( ) und implizites Laden von Fonts

Option	Beschreibung
keepnative	*(Boolean; nur relevant bei Standard-CJK-Fonts mit einer Nicht-Unicode-CMap; wird bei anderen Fonts ignoriert; ist bei* `embedding=true` *immer gleich* `false`*) Bei* `false` *wird Text in diesem Font bei der PDF-Ausgabe nach Unicode konvertiert (mit Adressierung über die Glyph-ID und Encoding* `Identity-H`*). Der Text, der an API-Funktionen übergeben wird, muss zur ausgewählten CMap passen (zum Beispiel Shift-JIS). Der Font kann jedoch mit Textflow und allen einfachen Textausgabe-Funktionen verwendet werden (jedoch nicht in Formularfeldern).* *Bei* `true` *wird Text mit diesem Font in seinem nativen Format gemäß der festgelegten CMap in die PDF-Ausgabe geschrieben. Der Font kann in Formularfeldern und allen einfachen Textausgabe-Funktionen, nicht jedoch mit Textflow verwendet werden. Standardwert:* `false` *bei TrueType-Fonts oder bei* `embedding=true`*; sonst* `true`
linegap	*(Integer zwischen -2048 und 2048) Siehe* `ascender` *oben.*
metadata	*(Optionsliste) Metadaten für den Font (siehe Abschnitt 14.2, »XMP-Metadaten«, Seite 277)*
monospace	*(Integer im Bereich 1...2048; nicht für PDF/A und PDF/UA; veraltet) Alle Glyphen des Fonts werden äquidistant in der Breite geschrieben, die durch den Integer-Wert gegeben ist (gemessen im Font-Koordinatensystem: 1000 Einheiten entsprechen der Fontgröße). Diese Option ist nur für Standard-CJK-Fonts empfehlenswert und wird für PDF-Standardfonts nicht unterstützt; sie wird ignoriert, wenn der Font eingebettet ist. Standardwert: Ist kein Wert gegeben, werden die Metrikdaten des Fonts verwendet.*
optimize-invisible	*(Boolean; nicht für PDF/X-1/2/3) Bei* `true` *werden Fonts, die ausschließlich für unsichtbaren Text verwendet werden (also* `textrendering=3`*) nicht eingebettet, selbst dann nicht, wenn* `embedding=true`*. Dies kann nützlich sein, um Fonteinbettung für PDF/A-Ausgabe bei unsichtbarem Text mit OCR-Ergebnissen zu vermeiden. Auch wenn der Font nicht eingebettet ist, müssen Fontdateien wie üblich konfiguriert werden, da PDFlib erst am Ende des Dokuments über das Nicht-Einbetten entscheidet. Standardwert:* `false`
preservepua	*(Boolean) Bei* `true` *behalten Zeichen, die einem Unicode-Wert in der Private Use Area (PUA) im Font zugeordnet sind, in der PDF-Ausgabe ihren PUA-Wert. Dies kann nützlich sein, wenn die PUA-Zeichen lokal definierte Einheiten wie Japanische Gaiji-/EUDC-Zeichen enthalten. Bei* `false` *werden PUA-Zeichen in der PDF-Ausgabe auf U+FFFD (Unicode-Ersatzzeichen) in der ToUnicode-CMap abgebildet. Standardwert:* `false`
readfeatures	*(Boolean; nur relevant bei TrueType- und OpenType-Fonts und* `encoding=unicode`*,* `glyphid` *oder Unicode-CMaps) Legt fest, ob die Feature-Tabellen eines TrueType- oder OpenType-Fonts aus dem Font gelesen werden. Das Anwenden von OpenType-Features auf den Text wird dabei von der Option* `features` *gesteuert (siehe Tabelle 5.10). Wird* `readfeatures` *auf* `false` *gesetzt und falls keine OpenType-Features erforderlich sind, lässt sich das Laden des Fonts beschleunigen. Standardwert:* `true`
readkerning	*(Boolean) Steuert, ob Kerning-Werte aus dem Font gelesen werden. Das Anwenden von Kerning-Werten auf den Text wird mit der Textoption* `kerning` *gesteuert (siehe Tabelle 4.7). Wird diese Option auf* `false` *gesetzt und falls keine OpenType-Features erforderlich sind, lässt sich das Laden des Fonts beschleunigen. Standardwert:* `true`
readselectors	*(Boolean; nur relevant bei TrueType- und OpenType-Fonts) Bei* `true` *werden die Variantenselektoren aus dem Font gelesen, sofern vorhanden. Dies ist für die automatische Ersetzung von ideografischen Variantensequenzen (IVS) innerhalb von Unicode-Text erforderlich. Standardwert:* `true`
readshaping	*(Boolean; nur relevant bei TrueType- und OpenType-Fonts und den Encodings* `unicode` *und* `glyphid`*) Legt fest, ob die Shaping-Tabellen eines TrueType- oder OpenType-Fonts gelesen werden, was beim Shaping komplexer Schriftsysteme erforderlich ist. Das Shaping von Text wird dabei von der Option* `shaping` *gesteuert (siehe Tabelle 5.10). Wird diese Option auf* `false` *gesetzt und ist Shaping nicht erforderlich, lässt sich Speicherplatz sparen. Standardwert:* `true`

Tabelle 4.2 Font-Optionen für PDF_load_font( ) und implizites Laden von Fonts

Option	Beschreibung
replacementchar	*(Unichar oder Schlüsselwort; nur relevant bei* `glyphcheck=replace`*; wird bei Fonts ignoriert, die mit einer Nicht-Unicode-CMap oder dem Encoding* `glyphid` *geladen werden) Glyphen, die im gewählten Font nicht verfügbar sind und sich nicht durch Fallback-Fonts ersetzen lassen, oder typografisch ähnliche Zeichen werden durch den hier angegebenen Unicode-Wert ersetzt. Enthält der Font keine Glyphe für das angegebene Unicode-Zeichen, wird das Verhalten von* `auto` *angewendet. Mit U+0000 wird das durch den Font bestimmte Symbol für eine fehlende Glyphe festgelegt. Für Symbolfonts, die mit* `encoding=builtin` *geladen werden, muss statt des Unicode-Werts der Byte-Code angegeben werden. Die folgenden Schlüsselwörter können anstelle eines Unicode-Zeichens verwendet werden (Standardwert:* `auto`*):* **auto** *Das erste Zeichen aus der folgenden Liste, für das eine Glyphe im Font vorhanden ist, wird als Ersatzzeichen verwendet: U+00A0 (NO-BREAK SPACE; umbruchgeschütztes Leerzeichen), U+0020 (SPACE, Leerzeichen), U+0000 (Symbol für fehlende Glyphe)* **drop** *Für das Zeichen wird keine Ausgabe erzeugt.* **error** *Eine Exception wird ausgelöst, wenn kein typografisch ähnliches Zeichen gefunden wurde. Dies kann zur Vermeidung von nicht lesbarer Textausgabe verwendet werden.*
simplefont	*(Boolean; nur relevant für TrueType-Fonts mit 8-Bit-Encoding) Ist diese Option auf* `true` *und* `embedding=false` *gesetzt, wird der Font als einfacher Font statt als CID-Font ausgegeben. Für einige PDF-Viewer (vor allem Acrobat) ist dies zur erfolgreichen Fontersetzung erforderlich, falls der Font nicht auf dem Viewer-System installiert ist. In Formularfeldern kann der Font jedoch eventuell nicht verwendet werden. Standardwert:* `false`
skipembedding	*(Liste von Schlüsselwörtern; nur relevant bei* `embedding=true`*) Wenn der Font eine oder mehrere in der Liste angegebene Bedingungen erfüllt, werden Probleme bei der Fonteinbettung ignoriert (das heißt, wenn trotz* `embedding=true` *der Font aus irgendeinem Grund nicht eingebettet werden kann). Dies ist nützlich, wenn Fonteinbettung zwar gewünscht, aber ein nicht eingebetteter Font gegenüber einem fehlenden Font vorzuziehen ist, wenn die Einbettung nicht möglich ist. Unterstützte Schlüsselwörter:* **latincore** *Der Font ist in der Gruppe der 14 lateinischen Standardfonts enthalten (für weitere Informationen siehe PDFlib-Tutorial).* **standardcjk** *Der Font ist in der Gruppe der Standard-CJK-Fonts enthalten (für weitere Informationen siehe PDFlib-Tutorial).* **fstype** *Der Font ist ein TrueType- oder OpenType-Font und erlaubt keine Einbettung gemäß dem Flag* `fsType` *in der OS/2-Tabelle des Fonts.* **metricsonly** *Nur die PFM- oder AFM-Metrikdaten stehen für den Font zur Verfügung, die Zeichenbeschreibungsdatei (PFA, PFB) fehlt jedoch.* *Standardwert: leere Liste* *Bei PDF/A, PDF/X und PDF/UA ist nur eine leere Liste erlaubt.*
skippostable	*(Boolean; nicht unterstützt; nur für TrueType- und OpenType-Fonts relevant) Bestimmt, ob die Tabelle* `post` *von TrueType-/OpenType-Fonts zur Ermittlung der Glyphnamen geparst wird. Wenn Sie diese Option auf* `true` *setzen, wird der Font unter Umständen schneller geladen, Glyphnamen-Referenzen funktionieren für den Font dann aber nur auf Glyphen mit Standardnamen (dies betrifft vorwiegend Symbolfonts und meist keine Textfonts). Standardwert:* `false`
subsetlimit	*(Float oder Prozentwert; wird bei Type-3-Fonts ignoriert) Deaktiviert die Bildung von Fontuntergruppen, wenn der prozentuale Anteil der im Dokument verwendeten Zeichen an der Gesamtzahl der im Font vorhandenen Zeichen den gegebenen Prozentsatz übersteigt. Standardwert: 100%*
subsetminsize	*(Float; wird bei Type-3-Fonts ignoriert) Deaktiviert die Bildung von Fontuntergruppen, wenn die originale Fontdatei kleiner als der angegebene Wert in KB ist. Standardwert: 50*

Tabelle 4.2 Font-Optionen für PDF_load_font( ) und implizites Laden von Fonts

Option	Beschreibung
subsetting	*(Boolean) Steuert, ob Fontuntergruppen gebildet werden. Um Untergruppen für einen Type-3-Font bilden zu können, muss dieser in zwei Durchläufen definiert und die Option* `subsetting` *im ersten Aufruf von PDF_load_font( ) übergeben werden (siehe PDFlib-Tutorial). Standardwert: Untergruppenbildung wird basierend auf den Einstellungen* `subsetlimit`/`subsetminsize` *automatisch aktiviert.*
unicodemap	*(Boolean; darf für PDF/A-1a/2a/2u/3a/3u und PDF/UA nicht* `false` *sein) Steuert die Generierung von To-Unicode-CMaps. Diese Option wird im Tagged-PDF-Modus ignoriert. Standardwert:* `true`
vertical	*(Boolean; nur für TrueType- und OpenType-Fonts; wird ignoriert, wenn eine vordefinierte CMap angegeben ist und wird immer auf* `true` *gesetzt, wenn der Fontname mit @ beginnt) Bei* `true` *wird der Font für vertikale Textausgabe vorbereitet.*
xheight	*(Integer zwischen -2048 und 2048) Siehe* `ascender` *oben.*

Tabelle 4.3 Unteroptionen für die Option `fallbackfonts` *von PDF_load_font( )*

Option	Beschreibung
Font-Optionen	*Wird der Font implizit angegeben (das heißt über die Optionen* `fontname` *und* `encoding`*, im Gegensatz zu der Option* `font`*), können alle Optionen zum Laden von Fonts gemäß Tabelle 4.2 außer* `fallbackfonts` *als Unteroptionen übergeben werden. Mit einer Nicht-Unicode-CMap geladene Fonts können nicht als Fallback-Fonts verwendet werden.*
font	*(Font-Handle) Von PDF_load_font( ) ohne die Option* `fallbackfonts` *zurückgegebenes Font-Handle. Mit dieser Option werden alle Optionen zum Laden von Fonts ignoriert (einschließlich* `fontname` *und* `encoding`*). Der Font darf kein Standard-CJK-Font mit* `embedding=false` *und* `keepnative=true` *sein.*
fontsize	*(Float oder Prozentwert) Größe des Fallback-Fonts in Benutzerkoordinaten oder als Prozentwert der aktuellen Fontgröße. Mit dieser Option kann die Größe des Fallback-Fonts angepasst werden, wenn die Designgröße des Fallback-Fonts nicht mit der des Basisfonts übereinstimmt. Standardwert: 100%*
forcechars	*(Liste von Unichars oder Unicode-Bereichen oder einzelnes Schlüsselwort) Bestimmt die Zeichen, die immer mit den Glyphen aus dem Fallback-Font dargestellt werden (unabhängig von der Einstellung* `glyphcheck`*). Der Fallback-Font muss die Glyphen für die angegebenen Zeichen enthalten (wenn einzelne Zeichen angegeben sind) oder wenigstens Glyphen für die ersten und letzten Zeichen im gewählten Unicode-Bereich. Für diese Option können auch Unicode-Werte angegeben werden, wenn ein 8-Bit-Encoding für den Basisfont angegeben wurde.* *Eines der folgenden Schlüsselwörter kann übergeben werden:* **gaiji** *Der Fallback-Font muss einen SING-Font referenzieren und dieses Schlüsselwort kann als Abkürzung für den Unicode-Wert der Hauptglyphe des SING-Fonts verwendet werden.* **all** *Alle Glyphen im Fallback-Font werden als Ersatz für die entsprechenden Zeichen im Basisfont verwendet, selbst wenn das Zeichen im Basisfont vorhanden ist.*
textrise	*(Float oder Prozentwert) Wert für vertikalen Textversatz (siehe Tabelle 4.7). Prozentangaben basieren auf der für den Fallback-Font angegebenen Größe (also nach Anwenden der Unteroption* `fontsize`*, sofern vorhanden). Mit dieser Option kann die Textposition im Fallback-Font angepasst werden, wenn die Designgröße des Fallback-Fonts nicht mit der des Basisfonts übereinstimmt. Standardwert: 0*

C++ Java C# **void close_font(int font)**

Perl PHP **close_font(int font)**

C **void PDF_close_font(PDF *p, int font)**

Schließt ein offenes Font-Handle, das im Dokument noch nicht verwendet wurde.

font Mit *PDF_load_font()* zurückgegebenes Font-Handle, das im Dokument bislang weder verwendet noch geschlossen wurde.

Details Mit dieser Funktion wird ein Font-Handle geschlossen und alle auf den Font bezogenen Ressourcen freigegeben. Nach diesem Aufruf darf das Font-Handle nicht mehr verwendet werden. Normalerweise werden Fonts am Ende des Dokuments automatisch geschlossen. Das Schließen eines Font kann jedoch in folgenden Situationen nützlich sein:

- Nach Abfrage von Fonteigenschaften mit *PDF_info_font()* wurde festgestellt, dass der Font im aktuellen PDF-Dokument nicht verwendet wird.
- Ein Font wurde über Dokumentgrenzen hinweg erhalten (mit der Option *keepfont* von *PDF_load_font()*), wird aber nun nicht länger benötigt und soll daher gelöscht werden.

Wurde der Font im aktuellen Dokument bereits verwendet, darf er nicht geschlossen werden.

Gültigkeit beliebig

C++ Java C# **double info_font(int font, String keyword, String optlist)**

Perl PHP **float info_font(int font, string keyword, string optlist)**

C **double PDF_info_font(PDF *p, int font, const char *keyword, const char *optlist)**

Fragt Einzelheiten zu einem geladenen Font ab.

font Font-Handle, das von *PDF_load_font()* zurückgegeben wurde. Für manche Schlüsselwörter kann auch -1 (in PHP: 0) übergeben werden.

keyword Schlüsselwort zur Abfrage der gewünschten Information gemäß Tabelle 4.5. Die folgenden Schlüsselwörter können verwendet werden:

- Schlüsselwörter für die Zuordnung von Glyphen: *cid, code, glyphid, glyphname, unicode*
- Fontmetriken: *ascender, capheight, descender, italicangle, linegap, xheight*
- Fontdateien, -name und -typ: *cidfont, familyname, fontfile, fontname, fonttype, metricsfile, outlineformat, singfont, standardfont, supplement*
- Technische Fontinformation: *feature, featurelist, hostfont, kerningpairs, numglyphs, shapingsupport, vertical*
- Schlüsselwörter für ideografische Variantenselektoren: *maxuvsunicode, minuvsunicode, selector, selectorlist*
- Beziehung zwischen Font und Encoding: *codepage, codepagelist, encoding, fallbackfont, keepnative, maxcode, numcids, numusableglyphs, predefcmap, replacementchar, symbolfont, unicodefont, unmappedglyphs*
- Ergebnisse der Fontverarbeitung für das aktuelle Dokument: *numusedglyphs, usedglyph, willembed, willsubset*
- Prüfung der Farbkompatibilität für Type-3-Fonts und PDF/A oder PDF/X: *checkcolorspace*

optlist Optionsliste, die das gewählte Schlüsselwort zusätzlich qualifiziert. Die folgenden Optionen können verwendet werden:

- Schlüsselwort-spezifische Optionen, die mit dem zugehörigen Schlüsselwort in Tabelle 4.5 beschrieben werden.
- Optionen zum Zuordnen von Glyphen gemäß Tabelle 4.4: *cid, code, glyphid, glyphname, selector, unicode*.
  Diese Optionen definieren den Quellwert für die Schlüsselwörter zum Zuordnen *cid, code, glyphid, glyphname* und *unicode*. Die Optionen zum Zuordnen schließen sich gegenseitig aus. Die Optionen *code, glyphname* und *unicode* können mit der Option *encoding* kombiniert werden.

Table 4.4 Optionen für die Angabe von Glyphen in PDF_info_font( )

Option	Beschreibung
cid	*(Zahl) CID-Wert der Glyphe; nur sinnvoll, wenn* `cidfont=1`
code	*(Zahl im Bereich 0...255; nur für Fonts mit 8-Bit-Encoding) Encoding-Position*
glyphid	*(Zahl im Bereich 0...65535) Interne Glyph-ID*
glyphname	*(String) Name einer Glyphe; nicht sinnvoll, wenn* `cidfont=1`
selector	*(Unichar) Unicode-Wert eines Variantenselektors im Bereich* `U+0xFE00..U+FE0F` *oder* `U+E0100..U+E01EF`. *Alle vom Schlüsselwort* `selector` *zurückgegebenen Werte können hier übergeben werden.*
unicode	*(Unichar) Unicode-Zeichen*

Rückgabe Wert einer mit *keyword* und gegebenenfalls weiteren Optionen definierten Font- oder Encoding-Eigenschaft. Bei nicht unterstützten Kombinationen von Schlüsselwort und Optionen wird -1 (in PHP: 0) zurückgegeben. Wenn das erforderliche Schlüsselwort Textausgabe produziert, wird ein String-Index zurückgegeben und der zugehörige String muss mit *PDF_get_string( )* abgerufen werden.

Details Diese Funktion liefert Information aus den folgenden Quellen:

- Wird ein gültiges Font-Handle übergeben, werden Informationen aus dem Font ausgegeben, zum Beispiel Fontmetrik, -name oder -typ oder der *unicode*-Wert für eine bestimmte *glyphid*.
- Bei *font = -1* (in PHP: 0) und wenn die Option *encoding* übergeben wird, werden Informationen über dieses Encoding ausgegeben, zum Beispiel der *unicode*-Wert für einen *code* im Encoding.
- Bei *font = -1* (in PHP: 0) und wenn die Option *encoding* nicht übergeben wird, werden Informationen aus PDFlib-internen Tabellen ausgegeben, zum Beispiel der *unicode*-Wert für einen bestimmten *glyphname*.

Gültigkeit beliebig

Tabelle 4.5 Schlüsselwörter und Optionen für PDF_info_font( )

Option	Beschreibung
ascender	*Metrikwert für die Oberlänge. Unterstützte Optionen (Standardwert:* `fontsize=1000`*):* ***faked*** *(Boolean) 1, wenn der Wert geschätzt wurde, da er in der Font- oder Fontmetrikdatei nicht verfügbar ist, sonst 0* ***fontsize*** *(Fontgröße) Der Wert wird auf die angegebene Fontgröße skaliert.*
capheight	*Metrikwert für die Versalhöhe. Siehe* `ascender`.
cid	*CID für die angegebene Glyphe oder -1, falls keine Glyphe vorhanden ist. Unterstützte Optionen:* `cid`*,* `glyphid`*,* `unicode`*,* `selector`
cidfont	*1, wenn der Font als CID-Font eingebettet wird, sonst 0*
code	*Zahl zwischen 0 und 255 für eine Encoding-Position oder -1, falls keine passende Position im Font oder Encoding gefunden wurde (wenn die Option* `encoding` *übergeben wurde und* `font=-1` *(in PHP: 0)). Unterstützte Optionen sind die Optionen zum Zuordnen* `code`*,* `glyphid`*,* `glyphname`*,* `unicode` *sowie die folgende:* ***encoding*** *(String) Name eines 8-Bit-Encodings*
codepage	*Prüft, ob der Font eine bestimmte Codepage unterstützt. Die Information wird aus den OS/2-Tabellen von TrueType-/OpenType-Fonts entnommen, sofern vorhanden. Unterstützte Option:* ***name*** *(String; erforderlich) Name einer Codepage in der Form cpXXXX, wobei XXXX die Dezimalzahl der Codepage darstellt (z.B.* `cp437`*,* `cp1252`*)* *Die folgenden Rückgabewerte geben an, ob die angegebene Codepage vom Font unterstützt wird:* ***-1*** *Unbekannt, da der Font kein TrueType- oder OpenType-Font ist.* ***0*** *Die angegebene Codepage wird vom Font nicht unterstützt.* ***1*** *Die angegebene Codepage wird vom Font unterstützt.*
codepagelist	*String-Index einer Liste aller vom Font unterstützten Codepages, jeweils durch ein Leerzeichen getrennt (in der Form* `cpXXXX`*), oder -1, wenn die Liste der Codepages unbekannt ist, da der Font kein TrueType- oder OpenType-Font ist (siehe* `codepage`*).*
check-colorspace	*(Nur relevant für Type-3-Fonts mit* `colorized=true`*) 1, wenn der Font auf der aktuellen Seite verwendet werden kann, ohne eine Verletzung bezüglich des Farbraums bei PDF/A oder PDF/X zu riskieren; sonst 0*
descender	*Metrikwert für die Unterlänge. Siehe* `ascender`.
encoding	*String-Index des Encoding- oder CMap-Namens des Fonts. Unterstützte Optionen (Standardwert:* `actual`*):* ***api*** *(Boolean) Bei* `true` *wird der im API ausgewählte Name des Encodings angegeben.* ***actual*** *(Boolean) Bei* `true` *wird der Name des tatsächlich für den Font verwendeten Encodings angegeben.*
fallbackfont	*Handle des Basis- oder Fallback-Fonts, der zur Darstellung des in der Option* `unicode` *angegebenen Zeichens verwendet wird. Damit lässt sich prüfen, welcher Font in der Reihe der Fallback-Fonts die Glyphe bereitstellt, die für das angegebene Zeichen verwendet wird. Kann das Zeichen weder im Basis- noch in den Fallback-Fonts dargestellt werden, wird -1 zurückgegeben. Unterstützte Option:* `unicode`
familyname	*String-Index des Namens der Fontfamilie oder -1, wenn nicht verfügbar*

Tabelle 4.5 Schlüsselwörter und Optionen für PDF_info_font( )

Option	Beschreibung
feature	*Prüft, ober der Font eine bestimmte, von PDFlib unterstützte OpenType-Feature-Tabelle enthält. Unterstützte Optionen:* ***language*** *(Schlüselwort; nur, wenn* `script` *übergeben wird) Name der Sprache. Standardwert:* _none ***name*** *(Schlüsselwort; erforderlich) Vierbuchstabiger Name einer OpenType-Feature-Tabelle, z.B.* `liga` *(Standard-Ligaturen),* `ital` *(kursive Formen in CJK-Fonts),* `vert` *(vertikale Textausgabe). Das Feature* `kern` *kann nicht abgefragt werden.* ***script*** *(Schlüsselwort) Bestimmt den Namen des Schriftsystems. Standardwert:* _none *Wird für* `language`, `name` *oder* `script` *ein unbekanntes Schlüsselwort übergeben, so wird eine Exception ausgelöst; für Listen bekannter Schlüsselwörter siehe das PDFlib Tutorial. Die folgenden Rückgabewerte geben an, ob die angegebene OpenType-Feature-Tabelle im Font vorhanden ist und von PDFlib unterstützt wird:* ***-1*** *Der Font enthält keine Feature-Tabellen.* ***0*** *Das Feature ist für das angegebene Schriftsystem und die Sprache nicht im Font verfügbar oder ist PDFlib nicht bekannt.* ***1*** *Das Feature ist für das angegebene Schriftsystem und die Sprache verfügbar.*
featurelist	*String-Index einer Liste aller Features, durch Leerzeichen getrennt, die im Font vorhanden sind und von PDFlib unterstützt werden, oder -1, wenn keine Feature-Tabellen vorhanden sind.*
fontfile	*String-Index des Pfadnamens der Zeichenbeschreibungsdatei des Fonts oder -1, falls nicht verfügbar*
fontname	*String-Index des Fontnamens oder -1, falls nicht verfügbar. Unterstützte Optionen (Standardwert:* `acrobat`*):* ***api*** *(Boolean) Bei* `true` *wird der im API festgelegte Fontname zurückgeliefert.* ***full*** *(Boolean) Bei* `true` *wird der Eintrag /FontName im PDF-Fontdeskriptor zurückgeliefert.* ***acrobat*** *(Boolean) Bei* `true` *wird der in Acrobat angezeigte Fontname zurückgeliefert.*
fontstyle	*String-Index für den Wert der Option* `fontstyle` (`normal`, `bold`, `italic` *oder* `bolditalic`)
fonttype	*String-Index des Fonttyps oder -1, falls nicht verfügbar. Bekannte Fonttypen sind:* `Multiple Master`, `OpenType`, `TrueType`, `TrueType (CID)`, `Type 1`, `Type 1 (CID)`, `Type 1 CFF`, `Type 1 CFF (CID)`, `Type 3`
glyphid	*Zahl zwischen 0 und 65535 für eine fontinterne Glyph-ID (GID) der angegebenen Glyphe oder -1, wenn keine passende Glyphe gefunden wurde. Unterstützte Optionen sind die Optionen zum Zuordnen* `cid`, `code`, `glyphid`, `glyphname`, `unicode`, `selector`.
glyphname	*String-Index des Namens der Glyphe oder -1, wenn diese Glyphe weder im Font noch im angegebenen Encoding gefunden wurde (wenn die Option* `encoding` *übergeben wurde und* `font=-1` *(in PHP: 0)). Unterstützte Optionen sind die Optionen zum Zuordnen* `code`, `glyphid`, `glyphname`, `unicode`, *sowie die folgende:* ***encoding*** *(String) Name eines 8-Bit-Encodings*
hostfont	*1, wenn der Font ein Systemfont ist, sonst 0*
italicangle	*Neigung des Fonts* (`ItalicAngle` *im PDF-Fontdeskriptor)*
keepnative	*Rückgabewert der Option* `keepnative`
kerningpairs	*Anzahl der Kerning-Paare im Font*
linegap	*Metrikwert für den* `linegap`*-Wert. Siehe* `ascender`.
maingid	*Glyph-ID der Hauptglyphe (Eintrag* `mainGID` *der SING-Tabelle)*
maxcode	*Höchster Codewert im Encoding des Fonts, insbesondere: 0xFF für Single-Byte-Encodings,* `numglyphs-1` *für* `encoding=glyphid`, *sonst der höchste Unicode-Wert im Encoding.*
metricsfile	*String-Index des Pfadnamens der Fontmetrikdatei (AFM oder PFM) oder -1, falls nicht verfügbar*

Tabelle 4.5 Schlüsselwörter und Optionen für PDF_info_font( )

Option	Beschreibung
maxuvs-unicode	*Höchster Unicode-Wert, der in einer gültigen ideografischen Variantensequenz (IVS) enthalten sein kann.*
minuvs-unicode	*Niedrigster Unicode-Wert, der in einer gültigen ideografischen Variantensequenz (IVS) enthalten sein kann.*
monospace	*(Veraltet) Wert der Option* `monospace` *oder 0, wenn sie nicht übergeben wurde*
numcids	*Anzahl der CIDs, wenn der Font eine Standard-CMap verwendet, sonst -1*
numglyphs	*Anzahl der Glyphen im Font (einschließlich der Glyphe* `.notdef`*). Da GIDs bei 0 beginnen, ist der höchstmögliche GID-Wert um eins kleiner als* `numglyphs`*.*
numusable-glyphs	*Anzahl der Glyphen im Font, die mit dem in PDF_load_font( ) übergebenen Encoding erreicht werden können*
numused-glyphs	*Anzahl der Glyphen, die bislang im generierten Text verwendet wurden*
outlineformat	*Fontformat; eins von* `PFA`, `PFB`, `LWFN`, `TTF`, `OTF`*. Für TTC- und WOFF-Fonts wird das Schlüsselwort für das zugrundeliegende Fontformat zurückgegeben, z.B.* `TTF`*. Für CEF-Fonts wird der String* `OTF` *zurückgegeben.*
predefcmap	*String-Index des Namens einer vordefinierten CMap, die als Encoding für den Font angegeben wurde, oder -1, falls nicht verfügbar.*
replacement char	*Unicode-Wert des in der Option* `replacementchar` *angegebenen Zeichens. Für Symbolfonts, die mit* `encoding=builtin` *geladen wurden, wird anstelle des Unicode-Werts der Code zurückgegeben.*
selector	*Unicode-Wert des Variantenselektors mit der in der Option* `index` *angegebenen Nummer. Wird die Option* `index` *nicht angegeben oder ist der angegebene Selektor nicht im Font verfügbar, wird -1 zurückgegeben. Unterstützte Option:* *index (Nicht negativer Integer) Index eines Selektors*
selectorlist	*String-Index eines Strings mit einer Liste von Unicode-Werten aller Variantenselektoren im Font, durch Leerzeichen getrennt. Jeder Wert wird in der Form* `hhhhh` *dargestellt, wobei* `h` *eine hexadezimale Ziffer darstellt.*
shaping-support	*1, wenn der Font Shaping unterstützt und die Option* `readshaping` *beim Laden des Fonts übergeben wurde, sonst 0*
singfont	*1, wenn der Font ein SING-Font (gaiji) ist, sonst 0*
standardfont	*1, wenn der Font ein PDF-Standardfont oder ein Standard-CJK-Font ist, sonst 0*
supplement	*Supplement-Nummer der Character-Collection für Fonts mit einer Standard-CMap, sonst 0*
symbolfont	*1, wenn der Font ein Symbolfont ist, sonst 0 (Symbolflag im PDF-Fontdeskriptor)*
unicode	*Unicode-UTF-32-Wert für die angegebene Glyphe oder -1, wenn der Unicode-Wert weder im Font noch im Encoding gefunden wurde (wenn die Option* `encoding` *übergeben wurde und* `font=-1` *(in PHP: 0)). Unterstützte Optionen sind die Optionen zum Zuordnen* `cid`, `code`, `glyphid`, `glyphname`, `unicode`, *sowie die folgende:* ***encoding*** *(String) Name eines 8-Bit-Encodings*
unicodefont	*1, wenn die Font/Encoding-Kombination Unicode-Zuordnungen für die Glyphen liefert, sonst 0. Bei CJK-Fonts mit Nicht-Unicode-CMaps und* `keepnative=true` *wird 0 zurückgegeben.*
unmapped-glyphs	*Anzahl der Glyphen in der Font/Encoding-Kombination, die auf Unicode-PUA-Werte abgebildet wurden, unabhängig davon, ob der PUA-Wert bereits im Font vorhanden ist oder von PDFlib zugewiesen wurde.*
usedglyph	*1, wenn die angegebene Glyph-ID im Text verwendet wurde, sonst 0. Unterstützte Option:* `glyphid`
vertical	*1, wenn der Font für vertikale Textausgabe vorgesehen ist, sonst 0*

Tabelle 4.5 Schlüsselwörter und Optionen für PDF_info_font( )

Option	Beschreibung
weight	*Fontgewicht zwischen 100 und 900; 400=normal, 700=fett*
willembed	*1, wenn der Font eingebettet wird (über die Option* `embedding` *oder erzwungene Fonteinbettung), sonst 0*
willsubset	*1, wenn eine Fontuntergruppe erzeugt wird (bei* `autosubsetting=true` *muss das durch* `subsetlimit` *definierte Limit erreicht werden, damit eine Untergruppe gebildet wird), sonst 0*
xheight	*Metrikwert für die x-Höhe. Siehe* `ascender`.

4.2 Optionen für Textfilterung und Textdarstellung

In diesem Abschnitt kennzeichnet der Begriff *Text* Content-Strings, also Text mit einer definierten Darstellung (z.B. Font, Farbe). Für Name-Strings und Hypertext-Strings (z.B. Dateinamen) kann dagegen keine Darstellung festgelegt werden; für weitere Informationen siehe PDFlib-Tutorial.

Textoptionen können mit *PDF_set_text_option( )*, *PDF_fit/info_textline( )*, *PDF_fill_textblock( )* und *PDF_add/create_textflow( )* verwendet werden. Textoptionen gelten auch für Tabellenzellen und Textblöcke. Die folgenden Gruppen von Textoptionen stehen zur Verfügung:

- Optionen zur Textfilterung gemäß Tabelle 4.6;
- Optionen zur Textdarstellung gemäß Tabelle 4.7;
- Optionen für Shaping und typografische Optionen gemäß Tabelle 5.10 (nicht für *PDF_set_text_option( )*).

Tabelle 4.6 Optionen zur Textfilterung für PDF_set_text_option( ), PDF_fit/info_textline( ), PDF_fill_textblock( ) und PDF_add/create_textflow( )

Option	Beschreibung
actualtext	*(Boolean; nur für PDF_set_text_option( ), PDF_fit_textline( ) und PDF_fill_textblock( )) Bei* `true` *erzeugt PDFlib markierten Inhalt mit einem geeigneten* `ActualText`*, falls der oder die Unicode-Werte aus der ToUnicode-CMap nicht korrekt sind und kein benutzerdefinierter Wert für* `Alt` *oder* `ActualText` *übergeben wurde. Dadurch wird die korrekte Textextraktion für Glyphen gewährleistet, die zur Darstellung mehrerer ähnlich aussehender Unicode-Werte in einem Font verwendet werden, z.B. Ohm und Omega. Standardwert:* `true`
charref	*(Boolean) Bei* `true` *werden numerische Referenzen, Entity-Referenzen und Glyphnamen-Referenzen in Content-Strings ersetzt.*[1] *Standardwert: die globale Option* `charref`
escape-sequence	*(Boolean) Bei* `true` *werden Escape-Sequenzen in Content-Strings ersetzt.*[1] *Standardwert: die globale Option* `escapesequence`
glyphcheck	*(Schlüsselwort) Richtlinie für die Überprüfung von Glyphen: was passiert, wenn ein Code im Text keiner Glyphe im ausgewählten Font zugeordnet werden kann (Standardwert: die globale Option* `glyphcheck`*)*[1]*:* **error** — *Bei nicht verfügbaren Glyphen wird eine Exception ausgelöst. Mit PDF_get_errmsg( ) kann eine detaillierte Fehlermeldung abgefragt werden.* **none** — *Keine Prüfung.* `notdef`*-Glyphen lösen im Modus PDF/A, PDF/UA oder PDF/X-4/5 eine Exception aus; anderenfalls können* `notdef`*-Glyphen in der Textausgabe erscheinen.* **replace** — *Versucht, nicht verfügbare Glyphen mit typografisch ähnlichen Zeichen aus den Basis- und Fallback-Fonts zu ersetzen und Ligaturen zu zerlegen. Wird keine passende Glyphe gefunden, wird das Zeichen gemäß der Option* `replacementchar` *behandelt.*

Tabelle 4.6 Optionen zur Textfilterung für PDF_set_text_option( ), PDF_fit/info_textline( ), PDF_fill_textblock( ) und PDF_add/create_textflow( )

Option	Beschreibung
normalize	*(Schlüsselwort; wird bei* encoding=glyphid *und Nicht-Unicode CMaps ignoriert) Texteingabe wird gemäß einer der Unicode-Normalformen normalisiert (Standardwert:* none*):* ***none*** *Keine Normalisierung. Dies ist das Standardverhalten; der Client ist für die Anlieferung von Text verantwortlich, der mit Glyphen aus dem gewählten Font dargestellt werden kann.* ***nfc*** *Normalform C (NFC): kanonische Zerlegung gefolgt von kanonischer Zusammensetzung. NFC ersetzt kombinierende Sequenzen durch vorkombinierte Zeichen. Dies ist für Workflows mit kombinierenden Sequenzen nützlich, da Fonts im Allgemeinen nur Glyphen für vorkombinierte Zeichen enthalten. Ohne NFC-Normalisierung gibt PDFlib statt des vorkombinierten Zeichens eine Sequenz von mehreren Zeichen aus.* ***nfkc*** *Normalform KC (NFKC): Kompatibilitätszerlegung gefolgt von kanonischer Zusammensetzung. Dies ist für Workflows nützlich, die auf die Bedeutung von Zeichen und nicht auf ihre unterschiedliche Formatierung ausgerichtet sind, wie zum Beispiel die Konvertierung geformter arabischer Zeichen in ihre Grundform, die Auflösung von Ligaturen und Brüchen, das Ersetzen von vertikalen durch horizontale Formen oder von breiten durch normal breite Zeichen.* ***nfd*** *Normalform D (NFD): kanonische Zerlegung* ***nfkd*** *Normalform KD (NFKD): Kompatibilitätszerlegung* *Da NFD und NFKD kombinierende Sequenzen erzeugen können, sind sie für den Einsatz mit PDFlib weniger geeignet.*
textformat	*(Schlüsselwort; nur für nicht Unicode-kompatible Sprachbindungen) Legt fest, in welchem Format Content-Strings an die Textausgabe-Funktionen übergeben werden sollen. Unterstützte Schlüsselwörter:* bytes, utf8, ebcdicutf8 *(nur bei i5/iSeries und zSeries),* utf16, utf16le, utf16be *und* auto.[1] *Standardwert: die globale Option* textformat

1. Der Wert kann von einem späteren Aufruf von PDF_set_option( ) mit der gleichen Option überschrieben werden.

Tabelle 4.7 Optionen zur Textdarstellung für PDF_set_text_option( ), PDF_fit/info_textline( ), PDF_fill_textblock( ) und PDF_add/create_textflow( )

Option	Beschreibung
charspacing	*(Float oder Prozentwert) Setzt den Zeichenabstand, das heißt, wie stark sich die aktuelle Position nach der Platzierung eines einzelnen Zeichens in einem String ändert. Float-Werte geben Einheiten des aktuellen Benutzerkoordinatensystems an; Prozentangaben basieren auf* fontsize*. Um einen größeren Zeichenabstand zu erreichen, verwenden Sie positive Werte im horizontalen Ausgabemodus und negative Werte im vertikalen Ausgabemodus. Standardwert: 0*
dasharray	*(Liste aus zwei nicht negativen Floats oder Schlüsselwort) Strichlängen und Zwischenräume für Umrisslinien und Textdekoration. Mit dem Schlüsselwort* none *lassen sich durchgezogene Linien zeichnen.*
decoration-above	*(Boolean) Bei* true *wird die mit den Optionen* underline, strikeout *und* overline *aktivierte Textdekoration über dem Text gezeichnet und sonst unter den Text. Wird die Reihenfolge der Textausgabe geändert, wirkt sich dies auf die Sichtbarkeit der Dekorationslinien aus, man kann also steuern, ob der Text die Linien überdruckt oder umgekehrt. Standardwert:* false
fakebold	*(Boolean) Bei* true *wird Fettschrift durch Zeichnen der Umrisslinien von Glyphen oder mehrfaches Überdrucken simuliert. Standardwert:* false
fillcolor	*(Farbe) Füllfarbe des Textes.*[1] *Standardwert für Funktionen zur einfachen Textausgabe und PDF_fit_textline( ) mit* inittextstate=false*: die entsprechende Option im aktuellen Grafikzustand.* *Standardwert für Textflow und PDF_fit_textline( ) mit* inittextstate=true*:* {gray 0} *(im PDF/A-Modus:* {lab 0 0 0}*)*

Tabelle 4.7 Optionen zur Textdarstellung für PDF_set_text_option( ), PDF_fit/info_textline( ), PDF_fill_textblock( ) und PDF_add/create_textflow( )

Option	Beschreibung
font	*(Font-Handle) Handle für den zu verwendenden Font. Mit dieser Option werden alle Optionen zum Laden von Fonts ignoriert (einschließlich* `fontname` *und* `encoding`*). Die Verwendung der Option* `font` *ist etwas schneller als das implizite Laden eines Fonts mit den Optionen* `fontname` *und* `encoding`*.* *Standardwert: der implizit geladene Font, falls verfügbar, ansonsten der mit PDF_setfont( ) ausgewählte Font für einfache Textausgabe und PDF_fit_textline( ) mit* `inittextstate=false`*. Anderenfalls ist kein Font verfügbar, was zu einem Fehler führt.*
fontsize	*(Fontgröße) Größe des aktuellen Fonts in den aktuellen Benutzerkoordinaten. Bei PDF_fit_textline( ) beziehen sich die Prozentangaben auf die Breite der Box (bei* `orientate=north` *und* `south`*) oder auf die Höhe der Box (bei* `orientate=east` *und* `west`*). Bei PDF_set_text_option( ) und Textflow beziehen sich die Prozentangaben auf die Größe des vorangegangenen Textes.* *Standardwert: PDF_setfont( ) setzt den Standardwert nur für Funktionen zur einfachen Textausgabe und PDF_fit_textline( ) mit* `inittextstate=false`*. Anderenfalls ist keine Fontgröße verfügbar, was zu einem Fehler führt.*
gstate	*(Gstate-Handle) Handle für einen mit PDF_create_gstate( ) erzeugten Grafikzustand. Der Grafikzustand wirkt sich auf den gesamten, mit dieser Funktion erzeugten Text aus. Standardwert: kein Grafikzustand (das heißt, die aktuellen Einstellungen werden verwendet).*
horizscaling	*(Float oder Prozentwert; darf nicht 0 sein) Horizontale Skalierung von Text auf den gegebenen Prozentwert. Legt anhand eines Prozentwertes fest, ob und wie stark der Text gestaucht oder gestreckt wird. Die Textskalierung bezieht sich immer auf die horizontale Koordinate. Standardwert: 100%*
inittextstate	*(Boolean; nur für PDF_fit_textline( )) Bei* `true` *werden alle Optionen zur Textdarstellung mit den Standardwerten belegt. Bei* `false` *werden die aktuellen Werte des Textzustands verwendet. Standardwert:* `false`
italicangle	*(Float; nicht unterstützt bei vertikalem Textausgabemodus) Bestimmt den Neigungswinkel von kursivem Text in Grad (von -90° bis 90°). Negative Werte können zur Simulation von kursivem Text verwendet werden, wenn nur der reguläre Font verfügbar ist. Dies lässt sich insbesondere bei CJK-Fonts nutzen. Standardwert: 0*
kerning	*(Boolean) Bei* `true` *wird die Unterschneidung (Kerning) für Fonts aktiviert, die mit der Option* `readkerning` *geladen wurden; ansonsten wird sie deaktiviert.[2] Standardwert: die globale Option* `kerning`
leading	*(Float oder Prozentwert) Setzt den Zeilenabstand, der als Abstand zwischen den Grundlinien aufeinanderfolgendem mehrzeiligem Text definiert ist, entweder als absoluter Wert in Benutzerkoordinaten oder als Prozentwert von* `fontsize`*. Wird der Zeilenabstand mit der Fontgröße gleichgesetzt, stoßen die Zeilen fast aneinander. Ober- und Unterlängen aufeinanderfolgender Zeilen überlappen aber in der Regel nicht (Bei* `leading=0` *werden die Zeilen überdruckt). Standardwert: 100%* *Der Zeilenabstand für PDF_add/create_textflow( ) wird folgendermaßen bestimmt: Sind am Anfang einer Zeile Optionslisten vorhanden, wird der Zeilenabstand anhand der letzten relevanten Option bemessen (*`font`*,* `fontsize`*,* `leading` *usw.). Sind in der selben Zeile weitere Optionslisten vorhanden, werden für den Zeilenabstand relevante Optionen nur berücksichtigt, wenn* `fixedleading=false` *ist. Sind in der Zeile keine Optionslisten vorhanden, wird der letzte verfügbare Wert für den Zeilenabstand verwendet.*
overline	*(Boolean) Bei* `true` *wird eine Linie oberhalb des Textes gezogen. Standardwert:* `false`

Tabelle 4.7 Optionen zur Textdarstellung für PDF_set_text_option( ), PDF_fit/info_textline( ), PDF_fill_textblock( ) und PDF_add/create_textflow( )

Option	Beschreibung
shadow	*(Optionsliste; nur für PDF_fit_textline( ), PDF_fill_textblock( ), PDF_add/create_textflow( )) Erzeugt einen Schatteneffekt für den Text (Standardwert: kein Schatten):* ***disable*** *(Boolean; nur für PDF_add/create_textflow( )) Bei* `true` *wird ein zuvor festgelegter Schatten deaktiviert. Standardwert:* `false` ***fillcolor*** *(Farbe) Füllfarbe des Schattens. Standardwert:* {`gray 0.8`} ***gstate*** *(Gstate-Handle) Mit PDF_create_gstate( ) erzeugter Grafikzustand, der auf den Schatten angewandt wird. Standardwert: kein Grafikzustand* ***offset*** *(Liste aus Floats oder Prozentwert) Bestimmt den Schatten-Offset vom Referenzpunkt des Textes in Benutzerkoordinaten oder als Prozentwert der Fontgröße. Standardwert:* {`5% -5%`} ***strokecolor*** *(Farbe; nur wenn* `textrendering` *auf Umrisslinien zeichnen gesetzt ist) Umrisslinie des Schattens. Standardwert: aktuelle Farbe für Umrisslinien* ***strokewidth*** *(Float, Prozentwert oder Schlüsselwort; nur wenn* `textrendering` *auf Umrisslinien zeichnen gesetzt ist) Linienstärke für die Textumrisslinien im Schatten (in Benutzerkoordinaten oder als Prozentwert der Fontgröße). Das Schlüsselwort* `auto` *oder der äquivalente Wert 0 verwenden einen eingebauten Standardwert. Standardwert: aktuelle Strichstärke, sofern für den Haupttext ebenfalls Umrisslinien zeichnen gesetzt ist, sonst* `auto` ***textrendering*** *(Integer) Modus für die Textdarstellung des Schattens. Standardwert: aktueller Wert von* `textrendering`
strikeout	*(Boolean) Bei* `true` *wird eine Linie durch den Text gezogen; siehe auch* `decorationabove`. *Standardwert:* `false`
strokecolor	*(Farbe; nur bei Umrisslinien, siehe* `textrendering`) *Farbe, in der Umrisslinien von Text gezeichnet werden. Standardwert: siehe* `fillcolor`
strokewidth	*(Float, Prozentwert oder Schlüsselwort; nur wenn* `textrendering` *auf Umrisslinien zeichnen gesetzt ist. Linienstärke für die Umrisslinien (in Benutzerkoordinaten oder als Prozentwert von* `fontsize`). *Das Schlüsselwort* `auto` *oder der äquivalente Wert 0 verwenden einen eingebauten Standardwert. Standardwert:* `auto`
tagtrailing-hyphen	*(Unichar oder Schlüsselwort; nur relevant bei Tagged PDF) Wenn das letzte Zeichen im Text (nach möglichen Glyphenersetzungen) gleich dem angegebenen Unicode-Wert ist, wird es als* `Span` *mit* `ActualText` *und dem weichen Trennzeichen U+00AD ausgezeichnet, wenn der Font dies erfordert. Es wird kein* `autospace` *hinzugefügt. Das Schlüsselwort* `none` *bewirkt, dass weiche Trennzeichen nicht ausgezeichnet werden. Standardwert: U+00AD*

Tabelle 4.7 Optionen zur Textdarstellung für PDF_set_text_option( ), PDF_fit/info_textline( ), PDF_fill_textblock( ) und PDF_add/create_textflow( )

Option	Beschreibung
textrendering	*(Integer) Legt den Modus für die Textdarstellung fest. Nur* `textrendering=3` *hat eine Auswirkung auf Type-3-Fonts (Standardwert: 0):* **0**  *Text füllen* **1**  *Umrisslinien zeichnen* **2**  *Umrisslinien zeichnen, füllen* **3** *Unsichtbarer Text* **4**  *Text füllen, zum Beschneidungspfad hinzufügen* **5**  *Umrisslinien zeichnen und zum Beschneidungspfad hinzufügen* **6**  *Umrisslinien zeichnen, füllen und zum Beschneidungspfad hinzufügen* **7**  *Text zum Beschneidungspfad hinzufügen* *Verhalten bei* `textrendering=4/5/6/7` *(Clipping-Modi):* ▸ *Wenn die Option* `textpath` *angegeben wurde, erfolgt nach PDF_fit_textflow( ), PDF_fit_table( ), PDF_fill_textblock( ) und PDF_fit_textline( ) kein Clipping.* ▸ *Clipping-Bereiche können über mehrere Aufrufe von einfachen Textausgabe-Funktionen hinweg erweitert werden, nicht jedoch über mehrere Aufrufe von PDF_fit_textline( ).* ▸ *PDF_fit_textline( ): Werden* `fillcolor` *und* `strokecolor` *angegeben, bleiben sie auch nach dem Funktionsaufruf aktiv.*
textrise	*(Float oder Prozentwert) Vertikaler Textversatz, der den Abstand zwischen der gewünschten Textposition und der Grundlinie festlegt. Positive Werte von* `textrise` *verschieben den Text nach oben. Der Textversatz bezieht sich immer auf die vertikale Koordinate. Dies kann zum Hoch- und Tiefstellen nützlich sein. Prozentwerte basieren auf* `fontsize`*. Standardwert: 0*
underline	*(Boolean) Bei* `true` *wird eine Linie unterhalb des Textes gezeichnet. Standardwert:* `false`
underline-position	*(Float, Prozentwert oder Schlüsselwort) Position der gezeichneten Linie für unterstrichenen Text relativ zur Grundlinie (absoluter Wert oder prozentual zur Fontgröße; ein typischer Wert ist -10%). Das Schlüsselwort* `auto` *legt einen fontspezifischen Wert fest, der aus der Metrik- oder Outline-Datei abgerufen wird. Standardwert:* `auto`
underline-width	*(Float, Prozentwert oder Schlüsselwort) Strichstärke für unterstrichenen Text (absoluter Wert oder Prozentwert der Fontgröße). Das Schlüsselwort* `auto` *oder der Wert 0 verwenden einen fontspezifischen Wert aus der Metrik- oder Outline-Datei, falls verfügbar; ansonsten 5%. Standardwert:* `auto`
wordspacing	*(Float oder Prozentwert) Wortabstand, der angibt, wie stark sich die aktuelle Position nach der Platzierung eines Wortes in einer Textzeile ändert. Anders ausgedrückt wird die aktuelle Position nach jedem Leerzeichen (U+0020) horizontal verschoben. Der Abstand wird in Einheiten des Textkoordinatensystems oder als Prozentwert der Fontgröße angegeben. Standardwert: 0*

1. Der Wert kann von einem nachfolgenden Aufruf von PDF_setcolor( ) für Funktionen zur einfachen Textausgabe und PDF_fit_textline( ) mit `inittextstate=false` *überschrieben werden.*
2. Der Wert kann von einem nachfolgenden Aufruf von PDF_set_option( ) mit der gleichen Option überschrieben werden.

C++ Java C# ***void set_text_option(String optlist)***

Perl PHP ***set_text_option(string optlist)***

C ***void PDF_set_text_option(PDF *p, const char *optlist)***

Setzt eine oder mehrere Optionen für Textfilter oder Textdarstellung bei Funktionen zur einfachen Textausgabe.

optlist Optionsliste mit folgenden Font- und Textoptionen:

- Textfilter-Optionen gemäß Tabelle 4.6:
  charref, escapesequence, glyphcheck, textformat
- Optionen zur Textdarstellung gemäß Tabelle 4.7:
  actualtext, charspacing, dasharray, decorationabove, fakebold, fillcolor, font, fontsize, gstate, horizscaling, inittextstate, italicangle, kerning, leading, overline, strikeout, strokecolor, strokewidth, tagtrailinghyphen, textrendering, textrise, underline, underlineposition, underlinewidth, wordspacing

Details Die Werte der Textoptionen sind für alle Funktionen zur einfachen Textausgabe und *PDF_fit_textline( )* mit *inittextstate=false* relevant. *PDF_set_text_option( )* sollte nicht mit *PDF_setfont( )* und *PDF_setcolor( )* vermischt werden.

Alle Textoptionen werden am Anfang einer Seite, eines Patterns, eines Templates oder einer Glyphenbeschreibung auf ihre Standardwerte zurückgesetzt und behalten ihre Werte bis ans Ende des aktuellen Gültigkeitsbereichs *page*, *pattern, template* oder *glyph*. Textoptionen lassen sich jedoch mit der Option *inittextstate* zurücksetzen.

Gültigkeit *page, pattern, template, glyph*

4.3 Einfache Textausgabe

Die Funktionen in diesem Abschnitt können für die Low-Level-Textausgabe verwendet werden. Für fortgeschrittene Textausgabe empfehlen wir, die leistungsfähigeren Textline- und Textflow-Funktionen zu verwenden (siehe Abschnitt 5.1, »Einzeilige Textausgabe mit Textlines«, Seite 97 und Abschnitt 5.2, »Mehrzeilige Textausgabe mit Textflows«, Seite 104).

C++ Java C# ***void PDF_setfont(int font, double fontsize)***

Perl PHP ***setfont(int font, float fontsize)***

C ***void PDF_setfont(PDF *p, int font, double fontsize)***

Setzt den aktuellen Font in der angegebenen Größe.

font Font-Handle, das von *PDF_load_font( )* zurückgegeben wurde.

fontsize Größe des Fonts, gemessen in Einheiten des aktuellen Benutzerkoordinatensystems. Die Fontgröße darf nicht gleich 0 sein; bei negativen Werten erscheint der Text gespiegelt bezogen auf die aktuelle Transformationsmatrix.

Details Mit dieser Funktion werden Font und Fontgröße für die Funktionen zur einfachen Textausgabe (z.B. *PDF_show( )*) und *PDF_fit_textline( )* gesetzt. Sie ist fast gleichbedeutend mit dem Aufruf von *PDF_set_text_option( )* mit der Optionsliste *font=<Font> fontsize=<Fontgröße>*. Anders jedoch als bei *PDF_set_text_option( )* wird bei dieser Funktion die Textoption *leading* auf *fontsize* gesetzt.

Der Font muss auf jeder Seite erneut gesetzt werden, und zwar bevor eine der Funktionen zur einfachen Textausgabe aufgerufen wird. Die Fonteinstellungen gelten immer nur für die aktuelle Seite.

Wir empfehlen, statt *PDF_setfont( )* besser *PDF_set_text_option( )* zu verwenden.

Gültigkeit *page, pattern, template, glyph*

C++ Java C# ***void set_text_pos(double x, double y)***

Perl PHP ***set_text_pos(float x, float y)***

C ***void PDF_set_text_pos(PDF *p, double x, double y)***

Setzt die aktuelle Position für einfache Textausgabe auf der Seite.

x, y Neue aktuelle Textposition

Details Zu Beginn jeder Seite wird die Textposition auf den Standardwert *(0, 0)* gesetzt. Der aktuelle Punkt für die Grafikausgabe und die aktuelle Textposition werden getrennt verwaltet.

Gültigkeit *page, pattern, template, glyph*

C++ Java C# ***void show(String text)***

Perl PHP ***show(string text)***

C ***void PDF_show(PDF *p, const char *text)***

C ***void PDF_show2(PDF *p, const char *text, int len)***

Gibt Text im aktuellen Font und in der aktuellen Größe an der aktuellen Textposition aus.

text (Content-String) Auszugebender Text. In C darf *text* in *PDF_show( )* keine Nullbytes enthalten, da ein null-terminierter String erwartet wird; für Strings, die Nullbytes enthalten können, verwenden Sie *PDF_show2( )*.

len (Nur für *PDF_show2( ))* Länge von *text* (in Bytes). Bei *len* = *0* muss ein null-terminierter String übergeben werden.

Details Der Font muss zuvor mit *PDF_setfont( )* oder *PDF_set_text_option( )* eingestellt worden sein. Die aktuelle Textposition wird ans Ende des ausgegebenen Textes verschoben.

Gültigkeit *page, pattern, template, glyph*

Bindungen *PDF_show2( )* ist nur in C verfügbar, da in allen anderen Sprachbindungen Strings beliebigen Inhalts an *PDF_show( )* übergeben werden können.

C++ Java C# ***void show_xy(String text, double x, double y)***

Perl PHP ***show_xy(string text, float x, float y)***

C ***void PDF_show_xy(PDF *p, const char *text, double x, double y)***

C ***void PDF_show_xy2(PDF *p, const char *text, int len, double x, double y)***

Gibt *text* im aktuellen Font an der angegebenen Textposition aus.

text (Content-String) Auszugebender Text. In C darf *text* in *PDF_show_xy( )* keine Nullzeichen enthalten, da ein null-terminierter String erwartet wird; für Strings, die Nullzeichen enthalten können, verwenden Sie *PDF_show_xy2( )*.

x,y Position im Benutzerkoordinatensystem, an der Text ausgegeben wird.

len (Nur für *PDF_show_xy2( ))* Länge von *text* (in Bytes). Bei *len* = *0* muss ein null-terminierter String übergeben werden.

Details Der Font muss zuvor mit *PDF_setfont( )* oder *PDF_set_text_option( )* eingestellt worden sein. Die aktuelle Textposition wird ans Ende des ausgegebenen Textes verschoben.

Gültigkeit *page, pattern, template, glyph*

Bindungen *PDF_show_xy2( )* ist nur in C verfügbar, da in allen anderen Sprachbindungen Strings beliebigen Inhalts an *PDF_show_xy( )* übergeben werden können.

C++ Java C# **void continue_text(String text)**

Perl PHP **continue_text(string text)**

C **void PDF_continue_text(PDF *p, const char *text)**

C **void PDF_continue_text2(PDF *p, const char *text, int len)**

Gibt Text in der nächsten Zeile aus.

text (Content-String) Auszugebender Text. Wird ein Leerstring übergeben, wird die Textposition in die nächste Zeile verschoben. In C darf *text* in *PDF_continue_text( )* keine Nullbytes enthalten, da ein null-terminierter String erwartet wird; für Strings, die Nullbytes enthalten können, verwenden Sie *PDF_continue_text2( )*.

len (Nur für *PDF_continue_text2( ))* Länge von *text* (in Bytes). Bei *len = 0* muss ein null-terminierter String wie in *PDF_continue_text( )* übergeben werden.

Details Die Textpositionierung *(x-* und *y*-Position) und der Zeilenabstand werden durch die Textoption *leading* (das mit *PDF_set_text_option( )* gesetzt werden kann) und dem letzten Aufruf von *PDF_show_xy( )* oder *PDF_set_text_pos( )* bestimmt. Die aktuelle Position wird ans Ende des auszugebenden Texts verschoben; die *x*-Position für nachfolgende Aufrufe dieser Funktion wird nicht geändert.

Gültigkeit *page, pattern, template, glyph;* diese Funktion sollte im vertikalen Textausgabemodus nicht verwendet werden.

Bindungen *PDF_continue_text2( )* ist nur in C verfügbar, da in allen anderen Sprachbindungen Strings beliebigen Inhalts an *PDF_continue_text( )* übergeben werden können.

C++ Java C# **double stringwidth(String text, int font, double fontsize)**

Perl PHP **float stringwidth(string text, int font, float fontsize)**

C **double PDF_stringwidth(PDF *p, const char *text, int font, double fontsize)**

C **double PDF_stringwidth2(PDF *p, const char *text, int len, int font, double fontsize)**

Gibt die Breite von *text* in einem beliebigen Font zurück.

text (Content-String) Text, dessen Breite abgefragt werden soll. In C darf *text* in *PDF_stringwidth( )* keine Nullbytes enthalten, da ein null-terminierter String erwartet wird; für Strings, die Nullbytes enthalten können, verwenden Sie *PDF_stringwidth2( )*.

len (Nur für *PDF_stringwidth2( ))* Länge von *text* (in Bytes). Bei *len = 0* muss ein null-terminierter String übergeben werden.

font Font-Handle, das von *PDF_load_font( )* zurückgegeben wurde.

fontsize Größe des Fonts, gemessen in Einheiten des Benutzerkoordinatensystems.

Rückgabe Breite von *text* in einem beliebigen mit *PDF_load_font( )* gewählten Font sowie der in *fontsize* übergebenen Fontgröße. Der Wert, der für die Breite zurückgegeben wird, kann negativ sein (zum Beispiel, wenn eine negative horizontale Skalierung eingestellt wurde). Bei vertikaler Textausgabe wird die Breite der breitesten Glyphe zurückgegeben (mit *PDF_info_textline( )* lässt sich die aktuelle Texthöhe ermitteln).
Wurde ein Wert für *charspacing* angegeben, wird er nach der letzten Glyphe ebenfalls angewendet (dies unterscheidet sich vom Verhalten bei *PDF_info_textline( )*).

Details Bei der Berechnung der Textbreite werden die aktuellen Werte folgender Textparameter herangezogen: *horizscaling*, *kerning*, *charspacing* und *wordspacing*.

Gültigkeit alle außer *object*

Bindungen *PDF_stringwidth2( )* ist nur in C verfügbar, da in allen anderen Sprachbindungen Strings beliebigen Inhalts an *PDF_stringwidth( )* übergeben werden können.

C++ Java C# ***void xshow(String text, const double *xadvancelist)***

C ***void PDF_xshow(PDF *p, const char *text, int len, const double *xadvancelist)***

Veraltet; verwenden Sie stattdessen *PDF_fit_textline( )* mit der Option *xadvancelist*.

4.4 Benutzerdefinierte (Type-3-)Fonts

Cookbook *Ein vollständiges Codebeispiel hierzu finden Sie im Cookbook-Topic* `fonts/starter_type3font`.

C++ Java C# ***void begin_font(String fontname,***
double a, double b, double c, double d, double e, double f, String optlist)

Perl PHP ***begin_font(string fontname,***
float a, float b, float c, float d, float e, float f, string optlist)

C ***void PDF_begin_font(PDF *p, char *fontname, int reserved,***
double a, double b, double c, double d, double e, double f, const char *optlist)

Beginnt die Definition eines Type-3-Fonts.

fontname (Name-String) Name, unter dem der Font registriert wird und später in *PDF_load_font( )* verwendet werden kann.

reserved (Nur C-Sprachbindung) Reserviert, muss gleich 0 sein.

a, b, c, d, e, f (Wird im zweiten Durchlauf der Fontdefinition für Type-3-Fontuntergruppen ignoriert) Elemente der Fontmatrix, die das Koordinatensystem definiert, in dem die Glyphen gezeichnet werden. Die sechs Werte definieren eine Matrix wie in PostScript und PDF. Um entartete Transformationen zu vermeiden, darf *a*d* nicht gleich *b*c* sein. Eine gängige Fontmatrix für ein 1000 *x* 1000 Koordinatensystem ist *[0.001, 0, 0, 0.001, 0, 0]*.

optlist (Wird im zweiten Durchlauf der Fontdefinition für Type-3-Fontuntergruppen ignoriert.) Optionsliste gemäß Tabelle 4.8. Folgende Optionen können verwendet werden: *colorized*, *familyname*, *stretch*, *weight*, *widthsonly*

Details Der Font kann eine beliebige Anzahl von Glyphen enthalten. Er kann bis zum Ende des aktuellen Gültigkeitsbereichs *document* benutzt werden.

Gültigkeit alle außer *object;* mit dieser Funktion beginnt der Gültigkeitsbereich *font;* diese Funktion muss immer paarweise mit *PDF_end_font( )* aufgerufen werden. Beim zweiten Durchlauf der Definition von Type-3-Fonts mit Untergruppen ist nur der Gültigkeitsbereich *document* erlaubt.

Tabelle 4.8 Optionen für PDF_begin_font( )

Option	Beschreibung
colorized	*(Boolean) Bei* `true` *kann der Font die Farbe einzelner Zeichen explizit festlegen. Bei* `false` *werden alle Zeichen in der (zum Zeitpunkt der Fontverwendung) aktuellen Farbe gezeichnet. Außerdem dürfen die Glyphdefinitionen keinerlei Farboperatoren oder von Masken verschiedene Bilder enthalten. Standardwert:* `false`
familyname[1]	*(String; PDF 1.5)s Name der Fontfamilie*
stretch[1]	*(Schlüsselwort; PDF 1.5) Streckung des Fonts. Mögliche Schlüsselwörter sind* `ultracondensed`, `extracondensed`, `condensed`, `semicondensed`, `normal`, `semiexpanded`, `expanded`, `extraexpanded`, `ultraexpanded`. *Standardwert:* `normal`

Tabelle 4.8 Optionen für PDF_begin_font( )

Option	Beschreibung
weight[1]	*(Integer oder Schlüsselwort; PDF 1.5) Fontgewicht: 100*=thin, *200*=extralight, *300*=light, *400*=normal, *500*=medium, *600*=semibold, *700*=bold, *800*=extrabold, *900*=black. *Standardwert:* normal
widthsonly	*(Boolean) Bei* true *(Erster Durchlauf bei der Untergruppenbildung) werden nur die Font- und Zeichenmetriken definiert. Zwischen PDF_begin_glyph( ) und PDF_end_glyph( ) sollten keine anderen API-Funktionen aufgerufen werden, da sie keinen Effekt auf die PDF-Ausgabe haben und auch keine Exception auslösen. Bei* widthonly=false *(Zweiter Durchlauf bei der Untergruppenbildung) können die eigentlichen Zeichenbeschreibungen definiert werden. Diese schrittweise Definition ermöglicht PDFlib die Untergruppenbildung von Type-3-Fonts. Standardwert:* false

1. Diese Optionen sollten bei der Erzeugung von Tagged PDF unbedingt verwendet werden und werden ansonsten ignoriert.

C++ Java C# ***void end_font( )***
Perl PHP ***end_font( )***
C ***void PDF_end_font(PDF *p)***

Beendet eine Type-3-Fontdefinition.

Gültigkeit *font;* mit dieser Funktion endet der Gültigkeitsbereich *font;* diese Funktion muss immer paarweise mit *PDF_begin_font( )* aufgerufen werden.

C++ Java C# ***void begin_glyph_ext(int uv, String optlist)***
Perl PHP ***begin_glyph_ext(int uv, string optlist)***
C ***void PDF_begin_glyph_ext(PDF *p, int uv, const char *optlist)***

Beginnt eine Type-3-Glyphdefinition.

uv Unicode-Wert für die Glyphe. Jeder Unicode-Wert kann nur für eine einzige Glyphenbeschreibung verwendet werden. Die Glyphe mit dem Unicode-Wert 0 erhält immer die Glyph-ID 0 und den Glyphnamen *.notdef*, unabhängig davon, ob die Glyphe angegeben wurde oder nicht.

Bei *uv*=-1 wird der Unicode-Wert von der Option *glyphname* gemäß der internen Glyphnamenliste von PDFlib abgeleitet. Ist ein Glyphname unbekannt, werden aufeinanderfolgende PUA-Werte zugewiesen (beginnend bei U+E000). Dieser Wert lässt sich mit *PDF_info_font( )* abfragen.

optlist Optionsliste gemäß Tabelle 4.9. Die folgenden Optionen können verwendet werden: *boundingbox*, ***code***, *glyphname*, *width*

Details Die Glyphen eines Fonts können mittels Text-, Vektorgrafik- und Rasterbild-Funktionen definiert werden. Rasterbilder dürfen aber nur eingesetzt werden, wenn die Option *colorized* des Fonts auf *true* gesetzt ist oder sie mit der Option *mask* geöffnet wurden. Diese Funktion setzt alle Parameter für Text, Grafik und Farbzustand auf ihre Standardwerte zurück.

Da der Grafikzustand der Seite beim Zeichnen der Glyphe vollständig übernommen wird, sofern die Option *colorize* auf *true* gesetzt ist, sollten in der Glyphdefinition alle relevanten Aspekte des Grafikzustands (zum Beispiel die Strichstärke mit *linewidth)* explizit eingestellt werden.

Gültigkeit *font;* mit dieser Funktion beginnt der Gültigkeitsbereich *glyph;* diese Funktion muss immer paarweise mit *PDF_end_glyph( )* aufgerufen werden. Bei *widthsonly=true* in *PDF_begin_font( )* werden alle API-Funktionsaufrufe zwischen *PDF_begin_glyph( )* und *PDF_end_glyph( )* ignoriert.

Tabelle 4.9 Optionen für PDF_begin_glyph_ext( )

Option	Beschreibung
boundingbox	*(List von 4 Floats; wird im zweiten Schritt der Fontdefinition für Type-3-Fontuntergruppen ignoriert, sofern die Option* `colorized` *des Fonts auf* `true` *gesetzt ist) Ist die Option* `colorized` *des Fonts gleich* `false` *(Standardwert), bezeichnen diese vier Parameter die Koordinaten der linken unteren und der rechten oberen Ecke der Boundingbox der Glyphe. Die Boundingbox-Werte müssen korrekt gesetzt sein, um Probleme beim PostScript-Druck zu vermeiden. Standardwert:* `{0 0 0 0}`
code	*(Integer) Slot-Nummer der Glyphe, das heißt ihr Byte-Code im Encoding* `builtin` *des Typ-3-Fonts. Standardmäßig werden die Glyphen nacheinander in der Reihenfolge ihrer Erstellung nummeriert (beginnend mit 0).*
glyphname	*(String) Name der Glyphe. Der Name für die Glyphe 0 mit Unicode 0 wird auf* `.notdef` *gesetzt. Standardwert:* `G<i>` *für Glyphe* `<i>=1,2,3,...`
width	*(Float; erforderlich; wird im zweiten Schritt der Fontdefinition für Type-3-Fontuntergruppen ignoriert) Laufweite der Glyphe im Glyphen-Koordinatensystem, das durch die Fontmatrix vorgegeben ist.*

C++ Java C# ***void end_glyph( )***
Perl PHP ***end_glyph( )***
C ***void PDF_end_glyph(PDF *p)***

Beendet eine Type-3-Glyphdefinition.

Gültigkeit *glyph;* mit dieser Funktion geht der Gültigkeitsbereich *glyph* in den Gültigkeitsbereich *font* über; diese Funktion muss immer paarweise mit *PDF_begin_glyph( )* auftreten.

C++ Java C# ***void begin_glyph(String glyphname, double wx, double llx, double lly, double urx, double ury)***
Perl PHP ***begin_glyph(string glyphname, float wx, float llx, float lly, float urx, float ury)***
C ***void PDF_begin_glyph(PDF *p,***
char *glyphname, double wx, double llx, double lly, double urx, double ury)

Veraltet, verwenden Sie stattdessen *PDF_begin_glyph_ext( )*.

4.5 Benutzerdefinierte 8-Bit-Encodings

C++ Java C# ***void encoding_set_char(String encoding, int slot, String glyphname, int uv)***

Perl PHP ***encoding_set_char(string encoding, int slot, string glyphname, int uv)***

C ***void PDF_encoding_set_char(PDF *p, const char *encoding, int slot, const char *glyphname, int uv)***

Fügt einen Glyphnamen und/oder Unicode-Wert zu einem benutzerdefinierten 8-Bit-Encoding hinzu.

encoding Name des Encodings. Dieser Name muss in *PDF_load_font()* verwendet werden. Er darf weder ein integriertes noch ein bereits verwendetes Encoding bezeichnen.

slot Position des zu definierenden Zeichens, wobei *0 <= slot <= 255* gilt. Jede Position darf innerhalb eines bestimmten Encodings nur einmal belegt werden.

glyphname Name des Zeichens

uv Unicode-Wert des Zeichens

Details Diese Funktion ist nur bei Anwendungen erforderlich, die mit speziellen benutzerdefinierten 8-Bit-Encodings arbeiten. Sie kann mehrfach aufgerufen werden, und es können maximal 256 Zeichenpositionen in einem Zeichensatz definiert werden. Dies ist schrittweise solange möglich, bis das Encoding das erste Mal verwendet wird. Danach jedoch wird beim Versuch, weitere Zeichen hinzuzufügen, eine Exception ausgelöst. Es müssen nicht alle Zeichenpositionen festgelegt werden; nicht definierte Positionen werden mit *.notdef* und U+0000 belegt.

Es sind drei verschiedene Kombinationen aus Glyphnamen und Unicode-Wert möglich:

- *glyphname* wird übergeben, *uv=0:* dies entspricht einer Encoding-Datei ohne Unicode-Werte;
- *uv* wird übergeben, *glyphname* jedoch nicht: dies entspricht einer Codepage-Datei;
- *glyphname* und *uv* werden übergeben: dies entspricht einer Encoding-Datei mit Unicode-Werten.

Wir empfehlen dringend, jeden Glyphnamen/Unicode-Wert in einem Encoding nur einmal zu übergeben (mit Ausnahme von *.notdef*/U+0000). Wird die Position 0 verwendet, sollte sie das Zeichen *.notdef* enthalten.

Soll das Encoding zusammen mit Type 3-Fonts verwendet werden, sollten Sie die Encoding-Positionen nur mit Glyphnamen angeben.

Das definierte Encoding kann bis zum Ende des aktuellen Gültigkeitsbereichs *object* verwendet werden.

Gültigkeit beliebig

5 Text- und Tabellenformatierung

5.1 Einzeilige Textausgabe mit Textlines

Cookbook *Ein vollständiges Codebeispiel finden Sie im Cookbook-Topic* `text_output/starter_textline`.

C++ Java C# ***void fit_textline(String text, double x, double y, String optlist)***
Perl PHP ***fit_textline(string text, float x, float y, string optlist)***
C ***void PDF_fit_textline(PDF *p, const char *text, int len, double x, double y, const char *optlist)***

Platziert eine einzelne Textzeile unter Berücksichtigung verschiedener Optionen am Referenzpunkt *(x, y)*.

text (Content-String) Auszugebender Text.

len (Nur C-Sprachbindung) Länge von *text* (in Bytes). Bei *len = 0* muss ein null-terminierter String übergeben werden.

x, y Koordinaten des Referenzpunkts im Benutzerkoordinatensystem, an dem der Text platziert wird. Die Platzierung lässt sich durch verschiedene Optionen genauer steuern. Für Informationen zum Algorithmus für die Objekteinpassung siehe Abschnitt 6.1, »Objekteinpassung«, Seite 133.

optlist Optionsliste mit Font-, Text- und Formatierungsoptionen. Folgende Optionen können verwendet werden:

- Allgemeine Option: *errorpolicy* (siehe Tabelle 2.1)
- Optionen zum impliziten Laden von Fonts gemäß Tabelle 4.2 (das heißt, die Option *font* wird in der Gruppe der Optionen zur Textdarstellung nicht übergeben): *ascender, autosubsetting, capheight, descender, embedding, encoding fallbackfonts, fontname, fontname, keepnative, linegap, metadata, monospace, readfeatures, replacementchar, subsetlimit, subsetminsize, subsetting, unicodemap, vertical, xheight*
- Optionen zur Textfilterung gemäß Tabelle 4.6: *charref, escapesequence, glyphcheck, normalize, textformat*
- Optionen zur Textdarstellung gemäß Tabelle 4.7: *actualtext, charspacing, dasharray, decorationabove, fakebold, fillcolor, font, fontsize, gstate, horizscaling, inittextstate, italicangle, kerning, leading, overline, shadow, strikeout, strokecolor, strokewidth, textrendering, textrise, underline, underlineposition, underlinewidth, wordspacing*
- Optionen für die Textline-Formatierung gemäß Tabelle 5.1: *justifymethod, leader, textpath, xadvancelist*
- Optionen für das Shaping und typografische Optionen gemäß Tabelle 5.3: *features, language, script, shaping*
- Optionen zur Objekteinpassung gemäß Tabelle 6.1: *alignchar, blind, boxsize, fitmethod, margin, matchbox, orientate, position, rotate, stamp, showborder, shrinklimit*
- Option zum vereinfachten Tagging von Strukturelementen gemäß Tabelle 14.5 (nur zulässig im Gültigkeitsbereich *page*): *tag*

Details Beim Standardwert *inittextstate=false* wird die Darstellung der Textausgabe durch die aktuellen Optionen für Text- und Grafikzustand gesteuert, sofern diese nicht explizit durch Optionen überschrieben werden.

Bei *inittextstate=true* werden die Standardwerte der Optionen für Text- und Grafikzustand für die Textausgabe verwendet, sofern sie nicht explizit mit Optionen überschrieben werden. Die Optionen für einzeilige Textausgabe haben nach diesem Aufruf von *PDF_fit_textline( )* keine Auswirkung auf die Textausgabe mehr.

Der aktuelle Text- und Grafikzustand ändert sich durch diese Funktion nicht (insbesondere der aktuelle Font bleibt unbeeinflusst). Die Optionen *textx/texty* dagegen werden ans Ende der generierten Textausgabe verschoben.

Der Referenzpunkt für *PDF_continue_text( )* wird nicht auf den Anfang des Textes gesetzt. Um *PDF_continue_text( )* nach *PDF_fit_textline( )* zu verwenden, müssen Sie den Ausgangspunkt mit *PDF_info_textline( )* und den Schlüsselwörtern *startx/starty* abfragen und dann die Textposition mit *PDF_set_text_pos( )* setzen.

Gültigkeit *page, pattern, template, glyph*

Tabelle 5.1 Zusätzliche Optionen für PDF_fit_textline( )

Option	Beschreibung
justifymethod	*(Liste von Schlüsselwörtern; nur relevant bei* `fitmethod=auto` *und* `stamp=none`; *erfordert* `boxsize`; *wird bei vertikaler Schreibrichtung ignoriert) Durch verschiedene Formatierungsmethoden wird sichergestellt, dass der Text ohne Änderung der Fontgröße in die Fitbox passt. Ein oder mehrere der folgenden Schlüsselwörter können übergeben werden; wenn mehrere Schlüsselwörter vorhanden sind, wird die Ausrichtung in der folgenden Reihenfolge in abnehmender Priorität angewendet:* `wordspacing`, `charspacing`, `horizscaling` *(Standardwert:* `none`*):* ***charspacing*** *Ausrichtung mit einem geeigneten* `charspacing`*-Wert.* ***horizscaling*** *Ausrichtung mit einem geeigneten* `horizscaling`*-Wert.* ***none*** *Keine Ausrichtung* ***wordspacing*** *Ausrichtung mit einem geeigneten* `wordspacing`*-Wert. Enthält der Text keine Leerzeichen, wird die Ausrichtung mit Wortabstand nicht angewendet.*
leader	*(Optionsliste; wird ignoriert, wenn* `boxsize` *nicht angegeben oder die Breite der Box gleich 0 ist) Definiert Führungszeichen (zum Beispiel Punkte) und Formatierungsoptionen. Führungszeichen werden kontinuierlich zwischen dem Rand der Textbox und dem Text eingefügt.* *Für eine Liste der unterstützten Unteroptionen siehe Tabelle 5.4. Standardwert: keine Führungszeichen*

Tabelle 5.1 Zusätzliche Optionen für PDF_fit_textline( )

Option	Beschreibung
textpath	*(Optionsliste) Erzeugt Text auf einem Pfad. Über das Pfadende hinausgehender Text wird nicht dargestellt. Bei* `showborder=true` *wird der in Liniensegmente umgesetzte Pfad mit der aktuellen Linienstärke und -farbe gezeichnet. Die Optionen in Tabelle 5.2 sowie die folgenden Optionen von PDF_draw_path( ) werden unterstützt:* `align`, `attachmentpoint`, `boxsize`, `fitmethod`, `orientate`, `position`, `scale` *(siehe Tabelle 6.1)* `close`, `round`, `subpaths` *(siehe Tabelle 7.7)* `bboxexpand`, `boundingbox` *(siehe Tabelle 7.7)* *Die folgenden Optionen von PDF_fit_textline( ) haben bei Text auf einem Pfad eine andere Bedeutung:* **matchbox** *Für jede Glyphe wird eine eigene Box erzeugt.* **position** *Der erste Wert gibt den Ausgangspunkt des Textes relativ zur Länge des Pfades an* (`left`/`center`/`right`). *Ist der Text länger als der Pfad, beginnt der Pfad immer bei* `startoffset`. *Der zweite Wert legt die vertikale Position jeder Glyphe relativ zum Pfad fest, das heißt, welcher Teil der Glyphbox den Pfad berührt* (`bottom`/`center`/`top`). **rotate** *Legt für jede Glyphe den Drehwinkel fest.* *Die folgenden auf die Fitbox bezogenen Optionen werden ignoriert:* `boxsize`, `margin`, `fitmethod`, `orientate`, `alignchar`, `showborder`, `stamp`, `leader` *Kerning sowie Text mit älteren CJK-Encodings werden bei Text auf einem Pfad nicht unterstützt.*
xadvancelist	*(Float-Liste) Legt den horizontalen Versatz der Glyphen im Text in Benutzerkoordinaten fest. Die Länge der Liste muss kleiner oder gleich der Anzahl der Glyphen im Text sein. Die hier übergebenen Werte werden statt der Standard-Glyphenbreiten verwendet. Andere Einstellungen wie die Unterschneidung (Kerning) oder der Zeichenabstand werden nicht beeinflusst.*

Tabelle 5.2 Zusätzliche Unteroptionen für die Option `textpath` *von PDF_fit_textline( )*

Option	Beschreibung
path	*(Pfad-Handle; erforderlich) Pfad, der als Grundlinie für die Textausgabe verwendet wird. Standardmäßig wird der Text an der linken Seite des Pfades platziert und der Pfad als Grundlinie für den Text verwendet. Wenn das zweite Schlüsselwort in der Option* `position` *jedoch auf* `top` *gesetzt ist, wird der Text auf der anderen Seite des Pfades platziert, und der obere Rand des Textes berührt den Pfad. Die Parameter* x *und* y *von PDF_fit_textline( ) werden als Referenzpunkte für den Pfad verwendet.*
rotate	*(Float) Dreht den Pfad, wobei der Referenzpunkt als Mittelpunkt und der übergebene Wert als Drehwinkel in Grad benutzt wird. Standardwert: 0*
scale	*(Liste aus einem oder zwei Floats) Skaliert den Pfad, wobei der Referenzpunkt als Mittelpunkt und die übergebenen Werte als horizontale und vertikale Skalierungsfaktoren benutzt werden. Wird nur ein Wert angegeben, wird er für beide Richtungen verwendet. Standardwert:* `{1 1}`
startoffset	*(Float oder Prozentwert) Der Versatz zum Ausgangspunkt des Textes entlang des Pfades in Benutzerkoordinaten oder als Prozentwert der Pfadlänge. Standardwert: 0*
tolerance	*(Float oder Prozentwert) Gibt an, um wie viel die letzte Glyphe auf dem Pfad über ihn hinausreichen darf. Der Wert wird in Benutzerkoordinaten oder als Prozentwert der Fontgröße angegeben. Standardwert: 25%*

Tabelle 5.2 Zusätzliche Unteroptionen für die Option textpath *von PDF_fit_textline()*

Option	Beschreibung
subpaths	*(Liste von Integers oder einzelnes Schlüsselwort) Liste mit der Anzahl der zu zeichnenden Teilpfade. Das Schlüsselwort* all *bedeutet alle Teilpfade. Standardwert:* all
close	*(Boolean) Bei* true *wird jeder Teilpfad mit einer geraden Linie geschlossen. Standardwert: Wert, der bei der Erzeugung des Pfades angegeben wurde, oder* false*, wenn dort kein Wert angegeben wurde*
round	*(Float) Für jeden Teilpfad werden benachbarte* line*-Eckpunkte an ihrem Verbindungspunkt durch einen Kreisbogen mit den Liniensegmenten als Tangenten und mit dem angegeben Radius abgerundet. Ist der Radius negativ, wird der Bogen so geschwungen, dass die Ecken kreisförmig nach innen gebogen sind. Bei* close=true *und wenn keine Linie vom letzten zum ersten Punkt angegeben wurde, werden die erste und die schließende Linie ebenfalls abgerundet. Bei* round=0 *wird nicht gerundet. Standardwert: Wert, der bei der Erzeugung des Pfades angegeben wurde, oder* 0*, wenn dort kein Wert angegeben wurde*

Tabelle 5.3 Optionen für das Shaping und typografische Optionen für PDF_fit/info_textline(), PDF_add/create_textflow() und PDF_fill_textblock()

Option	Beschreibung
features	*(Liste von Schlüsselwörtern) Legt fest, welche typografischen Features eines OpenType-Fonts auf einen Text angewendet werden, unter Berücksichtigung der Optionen* script *und* language*. Schlüsselwörter für nicht im Font vorhandene Features werden ignoriert. Die folgenden Schlüsselwörter werden unterstützt:* **_none** *Keines der Features im Font wird angewendet. Als Ausnahme muss das Feature* vert *explizit mit dem Schlüsselwort* novert *deaktiviert werden.* **<name>** *Das Feature wird durch Angabe seines OpenType-Namens (bestehend aus vier Zeichen) aktiviert. Einige gängige Feature-Namen sind* liga*,* ital*,* tnum*,* smcp*,* swsh*,* zero*. Eine vollständige Liste aller unterstützten Features mit den jeweiligen Beschreibungen finden Sie im PDFlib-Tutorial.* **no<name>** *Mit dem Präfix* no *vor dem Feature-Namen lässt sich das Feature deaktivieren (z.B.* noliga*).* *Standardwert:* _none *für horizontalen Textausgabemodus,* vert *für vertikalen Textausgabemodus.*
language	*(Schlüsselwort; nur relevant, wenn* script *übergeben wird) Der Text wird entsprechend der angegebenen Sprache verarbeitet, was sich auf die Optionen* features *und* shaping *auswirkt. Eine vollständige Liste aller Schlüsselwörter finden Sie im PDFlib-Tutorial, zum Beispiel* ARA *(Arabisch),* JAN *(Japanisch),* HIN *(Hindi). Standardwert:* _none *(Sprache nicht definiert)*
script	*(Schlüsselwort; erforderlich bei* shaping=true*) Der Text wird entsprechend des angegebenen Schriftsystems verarbeitet, was sich auf die Optionen* features*,* shaping *und* advancedlinebreak *auswirkt. Die gängigsten Schlüsselwörter für Schriftsysteme sind:* _none *(Schriftsystem nicht definiert),* latn*,* grek*,* cyrl*,* armn*,* hebr*,* arab*,* deva*,* beng*,* guru*,* gujr*,* orya*,* taml*,* thai*,* laoo*,* tibt*,* hang*,* kana*,* han*. Eine vollständige Liste aller Schlüsselwörter finden Sie im PDFlib-Tutorial. Das Schlüsselwort* _auto *wählt das Schriftsystem aus, zu dem die Mehrzahl der Zeichen im Text gehört, wobei* latn *und* _none *ignoriert werden.* _auto *ist nur bei* shaping *relevant und wird bei* features *und* advancedlinebreak *ignoriert. Standardwert:* _none
shaping	*(Boolean) Bei* true *werden Shaping für komplexe Schriftsysteme und bidirektionale Neuanordnung gemäß den Optionen* script *und* language *auf den Text angewendet. Die Option* script *muss einen von* _none *verschiedenen Wert haben und der Font muss bestimmten Bedingungen genügen (siehe PDFlib-Tutorial). Shaping wird nur bei Zeichen aus demselben Font angewendet. Shaping von linksläufigem Text ist nur für Textlines, aber nicht für Textflows verfügbar. Standardwert:* false

Tabelle 5.4 Unteroptionen für die Option `leader` *von PDF_fit_textline( ) und PDF_add/create_textflow( ) sowie für Inline-Optionen in PDF_create_textflow( )*

Option	Beschreibung
Optionen zum Laden von Fonts	*Wird der Font implizit angegeben (das heißt über die Optionen* `fontname` *und* `encoding`*, im Gegensatz zu der Option* `font`*), können alle Optionen zum Laden von Fonts gemäß Tabelle 4.2 als Unteroptionen übergeben werden.*
alignment	*(Ein oder zwei Schlüsselwörter) Textline (Textzeile): Das erste Schlüsselwort legt die Ausrichtung der Führungszeichen zwischen dem linken Rand der Fitbox und der Textzeile fest; das zweite Schlüsselwort bestimmt die Ausrichtung der Führungszeichen zwischen der Textzeile und dem rechten Rand der Fitbox. Wird nur ein Schlüsselwort angegeben, so definiert es die Ausrichtung zwischen der Textzeile und dem rechten Rand der Fitbox. Unterstützte Schlüsselwörter (Standardwert für Textline:* `{none grid}`*; Standardwert für Textflow:* `grid`*):* **center** *Textline: Die Führungszeichen werden bündig zwischen der Textzeile und dem Rand der Fitbox ausgerichtet.* *Textflow: Die Führungszeichen werden mittig zwischen dem letzten Textfragment (oder dem Zeilenanfang, falls kein Text vorhanden ist) und der Tabulatorposition (oder dem Zeilenende, falls kein Tabulator vorhanden ist) ausgerichtet.* **grid** *PDFlib rastet die Position der Führungszeichen auf dem nächsten Vielfachen der halben Breite der Führungszeichen links oder rechts der Textzeile ein. Dabei ergibt sich unter Umständen eine Lücke zwischen der Textzeile und den Führungszeichen, die maximal 50% der Breite der Führungszeichen einnimmt.* **justify** *Textline: Die Führungszeichen werden gleichmäßig zwischen der Textzeile und dem Rand der Fitbox verteilt, wobei ein geeigneter Zeichenabstand festgelegt wird.* *Textflow: Die Führungszeichen werden zwischen dem letzten Textfragment (oder dem Zeilenanfang, falls kein Text vorhanden ist) und der Tabulatorposition (oder dem Zeilenende, falls kein Tabulator vorhanden ist) ausgerichtet, wobei ein geeigneter Zeichenabstand festgelegt wird.* **left** *Die Führungszeichen werden wiederholt beginnend am linken Rand der Fitbox bzw. am Ende der Textzeile. Dabei kann sich eine Lücke am Anfang der Textzeile bzw. am rechten Rand der Fitbox ergeben.* **none** *Keine Führungszeichen* **right** *Die Führungszeichen werden wiederholt beginnend am rechten Rand der Fitbox bzw. am Anfang der Textzeile. Dabei kann sich eine Lücke am Ende der Textzeile bzw. am linken Rand der Fitbox ergeben.*
fillcolor	*(Farbe) Farbe der Führungszeichen. Standardwert: Farbe der Textzeile*
font	*(Font-Handle) Handle des Font für die Führungszeichen. Standardwert: Font der Textzeile*
fontsize	*(Fontgröße) Größe der Führungszeichen. Standardwert: Fontgröße der Textzeile*
text	*(Content-String) Text, aus dem die Führungszeichen bestehen. Standardwert: U+002E '.' (Punkt)*
yposition	*(Float oder Schlüsselwort) Definiert die vertikale Position der Führungszeichen relativ zur Grundlinie als numerischen Wert oder als eines der Schlüsselwörter* `fontsize`, `ascender`, `xheight`, `baseline`, `descender` *oder* `textrise`*. Standardwert:* `baseline`

C++ Java C# ***double info_textline(String text, String keyword, String optlist)***

Perl PHP ***float info_textline(string text, string keyword, string optlist)***

C ***double PDF_info_textline(PDF *p, const char *text, int len, const char *keyword, const char *optlist)***

Formatiert eine Textzeile, ohne Textausgabe zu erzeugen, und ermittelt die daraus resultierenden Metrikdaten.

text (Content-String) Inhalt der Textzeile.

len (Nur C-Sprachbindung) Länge des Texts in Bytes oder 0 für null-terminierte Strings.

keyword Schlüsselwort zur Anforderung der gewünschten Informationen:

- Schlüsselwörter zur Abfrage der Ergebnisse der Objekteinpassung gemäß Tabelle 6.3: *boundingbox, fitscalex, fitscaley, height, objectheight, objectwidth, width*
- Zusätzliche Schlüsselwörter gemäß Tabelle 5.5: *angle, ascender, capheight, descender, endx, endy, pathlength, perpendiculardir, replacedchars, righttoleft, scalex, scaley, scriptlist, startx, starty, textwidth, textheight, unmappedchars, wellformed, writingdirx, writingdiry, xheight*

optlist Optionsliste mit Optionen für *PDF_fit_textline( )*. Optionen, die für die jeweiligen Schlüsselwörter nicht relevant sind, werden ignoriert.

Rückgabe Wert der mittels *keyword* ausgewählten Metrikinformation für den Text.

Details Diese Funktion führt alle Berechnungen durch, die zur Platzierung des Texts anhand der übergebenen Optionen erforderlich sind, generiert aber keinerlei Ausgabe auf der Seite. Als Referenzpunkt für den Text wird {0 0} verwendet.

Bei *errorpolicy=return* gibt diese Funktion im Fehlerfall 0 zurück. Bei *errorpolicy=exception* löst diese Funktion im Fehlerfall eine Exception aus (auch für das Schlüsselwort *wellformed)*.

Gültigkeit beliebig außer *object*

Tabelle 5.5 Schlüsselwörter für PDF_info_textline( )

Schlüsselwort	Beschreibung
angle	*Rotationswinkel der Grundlinie in Grad, d.h. die Drehung des Texts*
ascender capheight descender	*Entsprechender typografischer Wert in Benutzerkoordinaten*
endx, endy	*x/y-Koordinaten der Text-Endposition in Benutzerkoordinaten*
pathlength	*(Nur für Text auf einem Pfad) Länge des Pfades, den der Text von seinem Ausgangspunkt bis zum Endpunkt einnimmt. Dieser Wert kann auch dann abgefragt werden, wenn PDF_fit_textline( ) im Modus* `blind` *aufgerufen wurde. Dieser Wert kann an die Option* `startoffset` *von PDF_fit_textline( ) übergeben werden, um die Beschriftung des Pfades mit weiterem Text fortzusetzen.*
perpendiculardir	*Einheitsvektor senkrecht zu* `writingdir`*; für horizontalen Standardtext entspricht dies (0, 1), für vertikalen Text (1, 0)*
replacedchars	*Anzahl der Zeichen, die durch eine leicht abweichende Glyphenform aus der internen Liste typografisch ähnlicher Zeichen oder durch einen Fallback-Font ersetzt wurden, weil sie auf keinen Code im aktuellen Encoding und auf keine Glyphe im Font abgebildet werden können. Dieser Wert kann nur von 0 verschieden sein, wenn* `glyphcheck=replace`*.*

Tabelle 5.5 Schlüsselwörter für PDF_info_textline()

Schlüsselwort	Beschreibung
righttoleft	*1, wenn für den Text linksläufige Ausgabe global eingestellt ist, und 0 bei rechtsläufiger Textausgabe. Die globale Ausgaberichtung wird basierend auf den ersten Zeichen und eventuell im Text vorhandenen Richtungsmarkern bestimmt (z.B. U+202D oder &LRO; LEFT-TO-RIGHT OVERRIDE).*
scalex, scaley	*Veraltet, verwenden Sie stattdessen* `fitscalex/fitscaley`
scriptlist	*String-Liste mit den Namen aller Schriftsysteme im Text, jeweils durch Leerzeichen getrennt. Dies kann zur Vorbereitung des Textes für das Shaping nützlich sein. Die Namen sind nach Häufigkeit in absteigender Reihenfolge sortiert.* `_none` *und* `_latn` *sind für das Shaping nicht relevant und werden ignoriert. Enthält der Text nur Zeichen aus den Schriftsystemen* `_none` *und* `_latn`, *wird -1 zurückgegeben.*
startx, starty	*x/y-Koordinaten der Textanfangsposition im Benutzerkoordinatensystem*
textwidth, textheight	*Breite und Höhe des Textes. Die Höhe ist abhängig von der Matchbox-Definition von* `boxheight`, *die auf den Standardwert* {`capheight none`} *eingestellt ist.*
unknownchars	*Bei* `glyphcheck=none`: *Anzahl der übersprungenen Zeichen. Hierzu werden auch Character-Referenzen gezählt, die nicht aufgelöst werden konnten, sowie Zeichen, die auf keinen Code im aktuellen Encoding und auf keine Glyphe im Font abgebildet werden konnten.* *Bei* `glyphcheck=replace`: *Anzahl der Zeichen, die durch das festgelegte Ersetzungszeichen ersetzt wurden (Option* `replacementchar`). *Hierzu werden auch Zeichen gezählt, die auf keinen Code im aktuellen Encoding und auf keine Glyphe im Font abgebildet und nicht durch typografisch ähnliche Zeichen ersetzt werden konnten.*
unmappedchars	*Anzahl aller Zeichen, die übersprungen oder ersetzt wurden, das heißt, die Summe aus* `replacedchars` *und* `unknownchars`.
wellformed	*1, wenn der Text gemäß dem gewählten Font/Encoding (und* `textformat`, *sofern anwendbar) wohlgeformt ist, sonst 0.*
writingdirx writingdiry	*x/y-Koordinaten der dominanten Schreibrichtung des Texts, die einen Einheitsvektor von* (`startx`, `starty`) *bis* (`endx`, `endy`) *darstellen. Für rechtsläufigen horizontalen Standardtext entspricht dies* `(1, 0)` *und für vertikalen Text* (`0, -1`) *und für linksläufigen Text* (`-1, 0`). *Die Schreibrichtung wird auf Grundlage der Optionen* `shaping` *und* `vertical` *sowie der im Text eingestellten Eigenschaften für die Schreibrichtung bestimmt.*
xheight	*x-Höhe in Benutzerkoordinaten*

5.2 Mehrzeilige Textausgabe mit Textflows

Cookbook *Ein vollständiges Codebeispiel finden Sie im Cookbook-Topic* `text_output/starter_textflow`.

C++ Java C# ***int add_textflow(int textflow, String text, String optlist)***

Perl PHP ***int add_textflow(int textflow, string text, string optlist)***

C ***int PDF_add_textflow(PDF *p, int textflow, const char *text, int len, const char *optlist)***

Erzeugt ein Textflow-Objekt oder fügt Text und Optionen zu einem vorhandenen Textflow hinzu.

textflow Textflow-Handle, das von einem früheren Aufruf von *PDF_create_textflow( )* oder *PDF_add_textflow( )* zurückgegeben wurde, oder -1 (in PHP: 0) beim Erstellen eines neuen Textflows.

text (Content-String) Inhalt des Textflows. Der Text darf keine Inline-Optionen enthalten.

len (Nur C-Sprachbindung) Länge des Texts in Bytes oder 0 für null-terminierte Strings.

optlist Optionsliste mit den folgenden Textflow-Optionen:

- Allgemeine Option: *errorpolicy* (siehe Tabelle 2.1)
- Optionen zum impliziten Laden von Fonts gemäß Tabelle 4.2 (das heißt, die Option *font* wird in der Gruppe der Optionen zur Textdarstellung nicht übergeben):
  ascender, autosubsetting, capheight, descender, embedding, encoding fallbackfonts, fontname, fontname, keepnative, linegap, metadata, monospace, readfeatures, replacementchar, subsetlimit, subsetminsize, subsetting, unicodemap, xheight
- Optionen zur Textfilterung gemäß Tabelle 4.6:
  charref, escapesequence, glyphcheck, normalize, textformat
- Optionen zur Textdarstellung gemäß Tabelle 4.7:
  charspacing, dasharray, decorationabove, fakebold, fillcolor, font, fontsize, gstate, horizscaling, inittextstate, italicangle, kerning, leading, overline, shadow, strikeout, strokecolor, strokewidth, textrendering, textrise, underline, underlineposition, underlinewidth, wordspacing
- Optionen für das Shaping und typografische Optionen gemäß Tabelle 5.3:
  features, language, script, shaping
- Optionen zur Textflow-Formatierung gemäß Tabelle 5.6:
  alignment, avoidemptybegin, fixedleading, hortabmethod, hortabsize, lastalignment, leader, leftindent, minlinecount, parindent, rightindent, ruler, tabalignment
- Optionen zur Steuerung des Algorithmus für den Zeilenumbruch gemäß Tabelle 5.7:
  adjustmethod, advancedlinebreak, avoidbreak, locale, maxspacing, minspacing, nofitlimit, shrinklimit, spreadlimit
- Optionen, die als Befehle wirken, gemäß Tabelle 5.8:
  comment, mark, matchbox, nextline, nextparagraph, restore, resetfont, return, save, space
- Optionen für die Textsemantik gemäß Tabelle 5.9:
  charclass, charmapping, hyphenchar, tabalignchar

Rückgabe Textflow-Handle, das in Funktionsaufrufen für Textflow verwendet werden kann. Das Handle ist bis zum Ende des zugehörenden Gültigkeitsbereichs *document* oder bis zum Aufruf von *PDF_delete_textflow( )* mit diesem Handle gültig.

Ist der Parameter *textflow* gleich -1 (in PHP: 0), so wird ein neuer Textflow erzeugt und dessen Handle zurückgegeben. Anderenfalls wird das Handle zurückgegeben, das im Parameter *textflow* übergeben wurde. Die Funktion gibt im Fehlerfall standardmäßig -1 (in PHP: 0) zurück. Das Verhalten lässt sich mit der Option *errorpolicy* ändern. Im Fehlerfall kann das im Parameter *textflow* übergebene Handle in keinen Funktionsaufrufen mehr verwendet werden (außer in *PDF_delete_textflow( ),* sofern es nicht -1 war).

Details Diese Funktion analysiert den übergebenen Text und erzeugt daraus eine interne Datenstruktur. Sie bestimmt Textabschnitte (zum Beispiel Wörter), die später bei der Formatierung berücksichtigt werden, konvertiert den Text nach Möglichkeit nach Unicode, ermittelt mögliche Zeilenumbrüche und berechnet die Breite von Textabschnitten anhand von Font- und Textoptionen.

Im Gegensatz zu *PDF_create_textflow( ),* das den gesamten Textinhalt und alle Optionen in einem einzigen Aufruf erwartet, können bei *PDF_add_textflow( )* die Textinhalte und Optionen eines Textflows in mehreren getrennten Aufrufen übergeben werden. Diese Funktion fügt den übergebenen Text und die Optionsliste zu einem neuen oder bereits vorhandenen Textflow hinzu. Die in *optlist* festgelegten Optionen werden vor der Verarbeitung von *text* ausgewertet. Sowohl *text* als auch *optlist* können leer sein.

Bei *textflow=-1* (in PHP: 0) entspricht diese Funktion im Wesentlichen *PDF_create_textflow( )* mit dem einzigen Unterschied, dass der Text nicht nach Inline-Optionen durchsucht wird. Man braucht deshalb mit einer Inline-Option weder das Startzeichen für Inline-Optionslisten umzudefinieren noch die Länge des Textes festzulegen (dies ist auch nicht erforderlich bei nicht-Unicode-Text oder UTF-16-Text).

Diese Funktion bereitet Text und Optionen nur vor und gibt ihn nicht in das generierte PDF-Dokument aus. Verwenden Sie *PDF_fit_textflow( ), PDF_fit_table( )* oder *PDF_fill_textblock( ),* um Textausgabe mit dem vorbereiteten Textflow-Handle zu generieren.

Standardmäßig bewirken die Zeichen *U+000B* (VT), *U+2028* (LS), *U+000A* (LF), *U+000D* (CR), CRLF, *U+0085* (NEL), *U+2029* (PS) und *U+000C* (FF) in Unicode-kompatiblen Fonts eine neue Zeile. Diese Kontrollzeichen werden für Symbolfonts mit *encoding=builtin* nicht ausgewertet. Diese Zeichen, mit Ausnahme von VT und LS, veranlassen zugleich die Erzeugung eines neuen Absatzes, so dass die Option *parindent* zur Anwendung kommt. FF bewirkt die sofortige Unterbrechung der Textplatzierung in die Fitbox (die Funktion *PDF_fit_textflow( )* beendet sich mit dem Rückgabe-String *_nextpage).*

Ein horizontaler Tabulator (HT) verändert die Startposition für nachfolgenden Text. Mit den Optionen *hortabmethod* und *hortabsize* wird diese Änderung im Detail gesteuert.

Weiche Trennzeichen *(soft hyphen, SHY)* werden bei einem Zeilenumbruch durch das in der Option *hyphenchar* festgelegte Zeichen ersetzt.

Vertikale Schreibrichtung wird nicht unterstützt.

Gültigkeit beliebig außer *object*

Tabelle 5.6 Zusätzliche Formatierungsoptionen für PDF_add/create_textflow( ) und Inline-Optionen in PDF_create_textflow( )

Option	Beschreibung
alignment	*(Schlüsselwort) Legt die Formatierung für die Zeilen eines Absatzes fest (Standardwert:* `left`*):* ***left*** *linksbündig, beginnend bei* `leftindent`+`parindent` *(in der ersten Zeile eines Absatzes) bzw. bei* `leftindent` *(in allen anderen Zeilen)* ***center*** *mittig zwischen* `leftindent` *und* `rightindent` ***right*** *rechtsbündig, bei* `rightindent` *endend* ***justify*** *links- und rechtsbündig (Blocksatz)*
avoid-emptybegin	*(Boolean) Bei* `true` *werden Leerzeilen am Anfang einer Fitbox gelöscht. Standardwert:* `false`
fixedleading	*(Boolean) Bei* `true` *wird der beim ersten Zeichen einer Zeile geltende Zeilenabstand verwendet. Anderenfalls wird der größte Zeilenabstand in der Zeile verwendet.* `fixedleading` *wird auf jeden Fall auf* `true` *gesetzt, wenn die Option* `wrap` *von PDF_fit_textflow( ) oder die Unteroption* `createwrapbox` *der Option* `matchbox` *genutzt wird, damit der Text um bestimmte Bereiche herumfließt. Standardwert:* `false`
hortab-method	*(Schlüsselwort) Legt die Interpretation von horizontalen Tabulatoren im Text fest. Liegt die berechnete Position links der aktuellen Textposition, so wird der Tabulator ignoriert (Standardwert:* `relative`*):* ***relative*** *Die Position wird um den in* `hortabsize` *festgelegten Betrag vorgerückt.* ***typewriter*** *Die Position wird auf das nächste Vielfache von* `hortabsize` *vorgerückt.* ***ruler*** *Die Position wird auf den n-ten in der Option* `ruler` *verfügbaren Tabulatorwert gesetzt, wobei n die Anzahl der bislang in der Textzeile vorgekommenen Tabs bezeichnet. Ist n größer als die Anzahl der Tabulatorpositionen, kommt die Methode* `relative` *zum Einsatz.*
hortabsize	*(Float oder Prozentwert) Legt die Breite eines horizontalen Tabulators fest[1]. Die Interpretation wird von der Option* `hortabmethod` *gesteuert. Standardwert: 7.5%*
lastalignment	*(Schlüsselwort) Bestimmt die Formatierung der letzten Zeile eines Absatzes. Neben allen Schlüsselwörtern der Option* `alignment` *wird das folgende Schlüsselwort unterstützt (Standardwert:* `auto`*):* ***auto*** *Es wird der Wert der Option* `alignment` *verwendet. Nur bei* `justify` *wird* `left` *verwendet.*
leader	*(Optionsliste) Legt die Führungszeichen fest (zum Beispiel Punkte). Führungszeichen werden bis zur nächsten Tabulatorposition oder, wenn kein Tabulatorzeichen vorhanden ist, bis zum Zeilenende eingefügt. Sie werden nur innerhalb einer Zeile und nicht über mehrere Zeilen hinweg ausgegeben. Für eine Liste der unterstützten Unteroptionen siehe Tabelle 5.4. Standardwert: kein Führungszeichen*
leading	*(Float oder Prozentwert) Zeilenabstand[2]. Der Wert wird wie folgt ermittelt: Befinden sich am Zeilenanfang Optionslisten, ergibt sich der Zeilenabstand aus der letzten relevanten Option (*`font`*,* `fontsize`*,* `leading` *usw.). Befinden sich weitere Optionslisten in der Zeile, werden diese nur bei* `fixedleading=false` *berücksichtigt. Befinden sich keine Optionslisten in der Zeile, wird der* `leading`*-Wert herangezogen. Standardwert: 100%*
leftindent	*(Float oder Prozentwert) Bestimmt den linken Einzug von Textzeilen[1]. Wird* `leftindent` *innerhalb der Zeile angegeben und befindet sich die definierte Position links der aktuellen Textposition, so wird die Option für die aktuelle Zeile ignoriert. Standardwert: 0*
minlinecount	*(Integer) Minimale Zeilenanzahl im letzten Absatz der Fitbox. Passen so viele Zeilen nicht mehr vollständig in die Fitbox, so werden sie zur Platzierung in der nächsten Fitbox einbehalten. Mit dem Wert 2 lassen sich einzelne »verwaiste« Zeilen am Ende der Fitbox verhindern. Standardwert: 1*
parindent	*(Float oder Prozentwert) Legt den linken Einzug der ersten Zeile eines Absatzes fest[1]. Der Wert wird zu* `leftindent` *addiert. Wird diese Option innerhalb der Zeile angegeben, so wirkt sie wie ein Tabulator. Standardwert: 0*
rightindent	*(Float oder Prozentwert) Bestimmt den rechten Einzug von Textzeilen[1]. Standardwert: 0*

Tabelle 5.6 Zusätzliche Formatierungsoptionen für PDF_add/create_textflow( ) und Inline-Optionen in PDF_create_textflow( )

Option	Beschreibung
ruler	*(Liste aus Floats oder Prozentwerten) Liste der absoluten Tabulatorpositionen für* `tabmethod=ruler`[1]*. Die Liste darf maximal 32 nicht-negative Einträge in aufsteigender Reihenfolge enthalten. Standardwert: ganzzahlige Vielfache von* `tabsize`
tabalignment	*(Schlüsselwortliste) Ausrichtung für Tabulatoren. Jeder Listeneintrag definiert die Ausrichtung des entsprechenden Eintrags in der Option* `ruler` *(Standardwert:* `left`*):*
	center *Text wird mittig an der Tabulatorposition ausgerichtet.*
	decimal *Das erste* `tabalignchar`*-Zeichen wird linksbündig an der Tabulatorposition ausgerichtet. Ist kein* `tabalignchar`*-Zeichen vorhanden, wird rechtsbündig ausgerichtet.*
	left *Text wird linksbündig an der Tabulatorposition ausgerichtet.*
	right *Text wird rechtsbündig an der Tabulatorposition ausgerichtet.*

1. In Benutzerkoordinaten oder als Prozentwert der Breite der Fitbox
2. In Benutzerkoordinaten oder als Prozentwert der Fontgröße

Tabelle 5.7 Zusätzliche Optionen zur Steuerung des Algorithmus für den Zeilenumbruch für PDF_add/create_textflow( ) und Inline-Optionen in PDF_create_textflow( )

Option	Beschreibung
adjustmethod	*(Schlüsselwort) Methode zur Anpassung von Textteilen, die nicht mehr in die Zeile passen, nachdem der Wortabstand gemäß der Optionen* `minspacing` *und* `maxspacing` *vergrößert oder verkleinert wurde. Unterstützte Schlüsselwörter (Standardwert:* `auto`*):*
	auto *Folgende Methoden werden in der angeführten Reihenfolge angewandt:* `shrink`, `spread`, `nofit`, `split`.
	clip *Wie* `nofit`*, nur dass der längere Teil am rechten Rand der Fitbox (unter Berücksichtigung der Option* `rightindent`*) abgeschnitten wird.*
	nofit *Das letzte Wort wird in die nächste Zeile verschoben, sofern die verbleibende (kurze) Zeile nicht kürzer als der in der Option* `nofitlimit` *festgelegte Prozentwert ist. Auch Absätze im Blocksatz sehen bei dieser Methode leicht ausgefranst aus.*
	shrink *Passt ein Wort nicht in die Zeile, wird der Text gestaucht, bis das Wort hineinpasst, sofern die Option* `shrinklimit` *dies zulässt. Anderenfalls kommt die Methode* `nofit` *zum Einsatz.*
	split *Das letzte Wort wird nicht in die nächste Zeile verschoben, sondern zwangsweise getrennt. Bei Textfonts (aber nicht bei Symbolfonts oder* `hyphenchar=none`*) wird ein Trennzeichen eingefügt.*
	spread *Das letzte Wort wird in die nächste Zeile verschoben. Der Rest der (kurzen) Zeile wird im Blocksatz ausgerichtet, indem der Zeichenabstand innerhalb der Wörter vergrößert wird, sofern die Option* `spreadlimit` *dies zulässt. Kann kein Blocksatz erreicht werden, kommt die Methode* `nofit` *zum Einsatz.*
advanced-linebreak	*(Boolean) Aktiviert einen Algorithmus für fortgeschrittenen Zeilenumbruch bei komplexen Schriftsystemen. Dies ist für den Zeilenumbruch bei solchen Schriftsystemen erforderlich, die Wortgrenzen nicht mit Leerzeichen kennzeichnen, wie zum Beispiel bei Thai. Die Optionen* `locale` *und* `script` *werden ausgewertet. Standardwert:* `false`
avoidbreak	*(Boolean) Bei* `true` *wird jeder Zeilenumbruch (z.B. bei Leerzeichen) nach Möglichkeit vermieden, bis* `avoidbreak` *auf* `false` *zurückgesetzt wird. Obligatorische Zeilenumbrüche (z.B. bei einer neuen Zeile) and Methoden, die durch* `adjustmethod` *definiert wurden, werden weiterhin durchgeführt. Insbesondere kann* `adjustmethod=split` *weiterhin die Silbentrennung aktivieren. Standardwert:* `false`

Tabelle 5.7 Zusätzliche Optionen zur Steuerung des Algorithmus für den Zeilenumbruch für PDF_add/create_textflow( ) und Inline-Optionen in PDF_create_textflow( )

Option	Beschreibung
locale	*(Schlüsselwort) Bestimmt die Spracheinstellung für lokalisierte Zeilenumbruch-Methoden, wenn* `advancedlinebreak=true`. *Das Schlüsselwort besteht aus einer oder mehreren Komponenten, wobei die optionalen Komponenten durch einen Unterstrich '_' getrennt sind. (Die Syntax unterscheidet sich geringfügig von NLS / POSIX locale IDs):* ► *Ein erforderlicher Sprachcode aus zwei oder drei Kleinbuchstaben gemäß ISO 639-2 (siehe www.loc.gov/standards/iso639-2), z.B.* `en`, *(Englisch),* `de` *(Deutsch),* `ja` *(Japanisch). Dieser unterscheidet sich von der Option* `language`. ► *Ein optionaler Ländercode aus zwei Großbuchstaben nach ISO 3166 (siehe www.iso.org/iso/country_codes/iso_3166_code_lists), z. B.* `DE` *(Deutschland),* `CH` *(Schweiz),* `GB` *(Großbritannien)* *Mit dem Schlüsselwort* _none *wird keine Sprache eingestellt. Bei manchen Schriftsystemen, z.B. Thai, ist für den erweiterten Zeilenumbruch eine Spracheinstellung erforderlich: Standardwert:* _none *Beispiele: Thai,* `de_DE`, `en_US`, `en_GB`
maxspacing **minspacing**	*(Float oder Prozentwert; nur relevant, wenn die Zeile mindestens ein Leerzeichen U+0020 enthält und bei* `alignment=justify`*) Bestimmt den maximalen bzw. minimalen Abstand zwischen Wörtern (in Benutzerkoordinaten oder als prozentualer Anteil der Breite eines Leerzeichens). Der berechnete Wortabstand wird durch die hier übergebenen Werte begrenzt, aber der Wert der Option* `wordspacing` *wird noch addiert. Standardwerte:* `minspacing=50%`, `maxspacing=500%`
nofitlimit	*(Float oder Prozentwert; nur relevant bei* `alignment=justify`*) Minimale Zeilenlänge bei der Methode* `nofit`[1]*. Standardwert: 75*
shrinklimit	*(Prozentwert) Untere Grenze für das Stauchen von Text mit* `adjustmethod=shrink`. *Der berechnete Stauchungsfaktor wird durch den hier übergebenen Wert begrenzt, aber noch mit dem Wert der Option* `horizscaling` *multipliziert. Standardwert: 85%*
spreadlimit	*(Float oder Prozentwert) Obere Grenze für den Abstand zwischen zwei Zeichen bei der Methode* `spread`[1]*; der berechnete Abstand wird zum Wert der Option* `charspacing` *addiert. Standardwert: 0*

1. In Benutzerkoordinaten oder als Prozentwert der Breite der Fitbox

Tabelle 5.8 Zusätzliche Befehlsoptionen für PDF_add/create_textflow( ) und Inline-Optionen in PDF_create_textflow( )

Option	Beschreibung
comment	*(String) Beliebiger Text, der ignoriert wird; nützlich zur Kommentierung von Optionslisten oder Makros*
mark	*(Integer) Speichert die übergebene Zahl intern als Marke. Die zuletzt gespeicherte Marke lässt sich dann mit PDF_info_textflow( ) und dem Schlüsselwort* `lastmark` *abfragen. Dies kann nützlich sein, um zu ermitteln, welche Textabschnitte bereits auf der Seite platziert wurden.*
matchbox	*(Optionsliste) Optionsliste mit detaillierten Angaben zur Matchbox gemäß Tabelle 6.4*
nextline **nextparagraph**	*(Boolean) Erzeugt eine neue Zeile oder einen neuen Absatz.*
resetfont	*(Boolean) Setzt die Optionen* `font` *und* `fontsize` *auf die letzten Werte zurück, die von den aktuellen Einstellungen abweichen (anderer Font oder Fontgröße). Dies kann zum Beispiel nach dem Einfügen von kursivem Text sinnvoll sein. Die Option* `font` *hat Vorrang vor dieser Option. Dieser Befehl ist erst nach dem ersten Wechsel einer fontspezifischen Option sinnvoll und wird sonst ignoriert.*
restore	*(Boolean) Bei* `true` *werden die Werte aller Text- und Textflow-Optionen auf die mit dem letzten* `save`*-Befehl gesicherten Werte zurückgesetzt. Eine innerhalb einer save/restore-Sequenz erzeugte Matchbox bleibt danach erhalten. Standardwert:* `false`

Tabelle 5.8 Zusätzliche Befehlsoptionen für PDF_add/create_textflow( ) und Inline-Optionen in PDF_create_textflow( )

Option	Beschreibung
return	*(String; darf nicht mit einem Unterstrich _ beginnen) Beendet PDF_fit_textflow( ) mit dem übergebenen String als Rückgabewert.*
save	*(Boolean) Bei* `true` *werden die Werte aller Text- und Textflow-Optionen gesichert, außer denen der zustandsfreien Optionen* `nextline`, `nextparagraph`, `resetfont`, `return`, `space` *und* `textlen`. *Save/restore-Paare können beliebig tief verschachtelt werden. Standardwert:* `false`
space	*(Float oder Prozentwert) Die Textposition wird horizontal um den angegebenen Wert[1] vorgerückt.*

1. In Benutzerkoordinaten oder als Prozentwert der Fontgröße

Tabelle 5.9 Zusätzliche Textsemantik-Optionen für PDF_add/create_textflow( ), Inline-Optionen in PDF_create_textflow( )

Option	Beschreibung
charclass	*(Liste von Paaren; das jeweils erste Element eines Paares ist ein Schlüsselwort und das zweite Element entweder ein Unichar oder eine Liste von Unichars; der Unichar muss < 0xFFFF sein; wird bei* `advancedlinebreak=true` *ignoriert) Um das Zeilenumbruchverhalten des oder der Zeichen zu bestimmen, werden die angegebenen Unichars durch ein Schlüsselwort kategorisiert:* **letter** *Verhalten wie bei Buchstaben, zum Beispiel* `a B` **punct** *Verhalten wie bei einem Satzzeichen, zum Beispiel* `+ / ; :` **open** *Verhalten wie bei einer öffnenden Klammer, zum Beispiel* `[` **close** *Verhalten wie bei einer schließenden Klammer, zum Beispiel* `]` **default** *setzt alle Zeichenkategorien auf die PDFlib-internen Standardwerte zurück* *Beispiel:* `charclass={ close » open « letter {/ : =} punct & }`

charmapping	*(Liste aus Paaren von zwei Unichars oder einem Unichar und einer Liste von Unichars und Integers; Unichars müssen < 0xFFFF sein) Mit dieser Option lassen sich einzelne Zeichen durch andere Zeichen in beliebiger Wiederholung ersetzen. Die Optionsliste enthält ein oder mehrere Paare von Unichars, wobei das erste Zeichen eines Paares durch das zweite Zeichen ersetzt wird. Das zweite Element eines Paares kann auch eine Optionsliste sein, die aus einem Unichar und einem Zähler besteht:*
	Zähler > 0 *Das Ersatzzeichen wird die angegebenen Male wiederholt.*
	Zähler < 0 *Eine Reihe mehrfach auftretender Zeichen wird auf den Absolutwert der angegebenen Anzahl reduziert.*
	Zähler=0 *Das Zeichen wird gelöscht.*
	Beispiele:
	`charmapping={ hortab space CRLF space LF space CR space }`
	`charmapping={ shy {shy 0} }`
	`charmapping={ hortab {space 4} }`

hyphenchar	*(Unichar < 0xFFFF oder Schlüsselwort) Das Zeichen, das ein weiches Trennzeichen bei Zeilenumbrüchen ersetzt. Durch den Wert 0 bzw. das Schlüsselwort* `none` *werden Trennzeichen generell unterbunden. Standardwert: U+00AD (soft hyphen), sofern im Font verfügbar, anderenfalls U+002D (hyphen-minus)*
tabalignchar	*(Unichar < 0xFFFF) Das Zeichen, an dem dezimale Tabulatoren ausgerichtet werden. Standardwert: U+002E (.)*

Makros für Textflow-Optionen. Optionslisten für Textflüsse (entweder im Parameter *optlist* von *PDF_create_textflow( ) bzw. PDF_add_textflow( )* oder inline im Text für *PDF_create_textflow( )*) können Makrodefinitionen oder -aufrufe gemäß Tabelle 5.10 enthalten. Makros sind nützlich, um mehrfach auftretende Optionen, zum Beispiel für Fontnamen oder Absatzeinrückung, einmal an zentraler Stelle zu definieren. Vor dem Parsen einer Optionsliste wird jedes enthaltene Makro durch die in der Makrodefinition festgelegte Optionsliste ersetzt. Die daraus resultierende Optionsliste wird dann geparst. Das folgende Beispiel zeigt eine Makrodefinition für zwei Makros:

```
<macro {
        comment { Die folgenden Makros werden als Absatzformate verwendet }
        H1 {fontname=Helvetica-Bold encoding=winansi fontsize=14 }
        Text {fontname=Helvetica encoding=winansi fontsize=12 }
}>
```

Diese Makros könnten wie folgt in einer Optionsliste verwendet werden:

```
<&H1>Kapitel 1
<&Text>Dieses Kapitel beschäftigt sich mit ...
```

Für die Definition und Verwendung von Makros gelten die folgenden Regeln:

- Makros können beliebig verschachtelt werden (d.h. in Makrodefinitionen können andere Makros aufgerufen werden).
- Makros dürfen nicht in der Optionsliste vorkommen, in der sie definiert werden. Bei *PDF_create_textflow( )* kann auf eine Inline-Optionsliste, in der das Makro definiert wird, direkt eine neue Inline-Optionsliste folgen, in der das Makro verwendet wird. Bei *PDF_add_textflow( )* ist ein Funktionsaufruf zur Makrodefinition und einer zur Verwendung des Makros erforderlich (da *PDF_add_textflow( )* nicht mehrere Optionslisten gleichzeitig akzeptiert).
- Bei Makronamen wird nicht zwischen Groß- und Kleinschreibung unterschieden.
- Nicht definierte Makros lösen eine Exception aus.
- Makros können jederzeit umdefiniert werden.

Tabelle 5.10 Makrodefinitionen und -aufrufe in Optionslisten für PDF_add/create_textflow( ) und PDF_fit_textflow( )

Option	*Beschreibung*
macro	*(Liste aus Paaren) Jedes Paar beschreibt den Namen und die Definition eines Makros wie folgt (beachten Sie, dass zwischen dem Makronamen und seiner Definition kein Gleichheitszeichen '=' stehen darf):* ***name*** *(String) Name des Makros, der später in Makroaufrufen verwendet werden kann. Bereits definierte Makros können umdefiniert werden. Der Sondername* `comment` *wird ignoriert.* ***suboptlist*** *Optionsliste, durch die Makroname beim Aufruf des Makros wortwörtlich ersetzt wird. Führende und schließende Leerzeichen werden ignoriert.*
&name	*Das Makro mit dem angegebenen Namen wird expandiert, und der Makroname (einschließlich des Zeichens* `&`*) durch den Inhalt des Makros ersetzt, also durch die für das Makro definierte Suboptionsliste (ohne die umgebenden Klammern). Da der Makroname durch ein Leerzeichen,* `{ , } , =` *oder* `&` *begrenzt ist, dürfen diese Zeichen nicht Bestandteil des Makronamens sein.* *Verschachtelte Makros werden ohne Einschränkung expandiert. Makros, die in String-Optionen enthalten sind, werden ebenfalls expandiert. Aus der Makroexpansion muss sich eine gültige Optionsliste ergeben.*

C++ Java C# ***int create_textflow(String text, String optlist)***

Perl PHP ***int create_textflow(string text, string optlist)***

C ***int PDF_create_textflow(PDF *p, const char *text, int len, const char *optlist)***

Erzeugt ein Textflow-Objekt aus Textinhalt, Inline-Optionen und explizit übergebenen Optionen.

text (Content-String) Inhalt des Textflows, der Text in verschiedenen Encodings, Makros (siehe Abschnitt »Makros für Textflow-Optionen«, Seite 110) und Inline-Optionslisten gemäß Tabelle 5.6 und Tabelle 5.11 enthalten kann (siehe auch »Inline-Optionslisten für Textflows«, Seite 112). Auch wenn *text* ein leerer String ist, wird ein gültiges Textflow-Handle zurückgegeben.

len (Nur C-Sprachbindung) Länge des Texts in Bytes oder 0 für null-terminierte Strings.

optlist Optionsliste mit Textflow-Optionen. Die Optionen in *optlist* werden vor den Inline-Optionslisten in *text* ausgewertet. Inline-Optionen haben somit Vorrang vor den Optionen, die in *optlist* übergeben werden. Folgende Optionen können verwendet werden:

- Allgemeine Optionen: *errorpolicy* (siehe Tabelle 2.1)
- Alle Optionen von *PDF_add_textflow( )*, siehe Optionsliste von *PDF_add_textflow( )*
- Optionen, die die Verarbeitung von Inline-Optionslisten gemäß Tabelle 5.11 steuern: *begoptlistchar, endoptlistchar, fixedtextformat, textlen*

Rückgabe Textflow-Handle, das in Aufrufen von *PDF_add_textflow( )*, *PDF_fit_textflow( )*, *PDF_info_textflow( )* und *PDF_delete_textflow( )* verwendet werden kann. Das Handle ist bis zum Ende des zugehörenden Geltungsbereichs *document* gültig oder bis zum Aufruf von *PDF_delete_textflow( )* mit dem Handle. Im Fehlerfall gibt die Funktion den Fehlercode -1 (in PHP: 0) zurück. Dieses Verhalten lässt sich mit der Option *errorpolicy* ändern.

Details Diese Funktion bereitet anhand von Textinhalt und Optionen den Textflow vor. Im Gegensatz zu *PDF_add_textflow( )* kann der Text Inline-Optionen enthalten. Die Ermittlung von Inline-Optionen kann für einzelne Textabschnitte oder den gesamten Text deaktiviert werden, indem die Option *textlen* im Parameter *optlist* übergeben wird (siehe »Inline-Optionslisten für Textflows«, Seite 112).

Diese Funktion erzeugt keinerlei Ausgabe im generierten PDF-Dokument, sondern bereitet den Text nur gemäß der übergebenen Optionen vor. Zur Generierung der eigentlichen Ausgabe verwenden Sie *PDF_fit_textflow( )* mit dem erhaltenen Handle.

Für weitere Informationen zu Sonderzeichen, Zeilenumbruch usw. siehe die Rubrik *Details* von *PDF_add_textflow( )*.

Gültigkeit beliebig außer *object*

Tabelle 5.11 Weitere Optionen zur Verarbeitung von Inline-Optionslisten für PDF_create_textflow( )

Option	Beschreibung
begoptlistchar	*(Unichar < 0xFFFF oder Schlüsselwort) Zeichen, mit dem Inline-Optionslisten beginnen. Das standardmäßig definierte Zeichen muss unter Umständen umdefiniert werden, wenn es in seiner ursprünglichen Bedeutung im Text enthalten ist (siehe »Inline-Optionslisten für Textflows«, Seite 112). Ist* `textlen` *nicht angegeben, muss das Zeichen* `begoptlistchar` *im Text im selben Textformat und Encoding wie der vorangehende Text kodiert sein. Der Unicode-Wert von* `begoptlistchar` *muss also so gewählt werden, dass er im Encoding des vorangehenden Texts enthalten ist. Mit dem Schlüsselwort* `none` *lässt sich die Interpretation von Optionslisten vollständig deaktivieren. Standardwert: U+003C (<)*
endoptlistchar	*(Unichar < 0xFFFF; U+007D '}' ist nicht zulässig) Unicode-Wert des Zeichens, mit dem Inline-Optionslisten beendet werden. Standardwert: U+003F (>)*
fixedtext-format	*(Boolean; wird in Unicode-fähigen Sprachen ignoriert; wird bei* `stringformat=utf8` *auf* `true` *gesetzt; diese Option ist in Inline-Optionslisten sinnlos und kann nur im Parameter* `optlist` *verwendet werden) Bei* `true` *verwenden alle Textfragmente und Inline-Optionslisten dasselbe Textformat. Zur Auswahl stehen* `utf8`, `utf16`, `utf16be` *und* `utf16le`*. Diese Einstellung ist sinnvoll, wenn Text und Inline-Optionen aus derselben Quelle stammen.* *Bei* `false` *müssen Inline-Optionslisten einschließlich der Trennzeichen in* `textformat=bytes` *kodiert sein, unabhängig vom für den eigentlichen Textinhalt verwendeten Format. Dies ermöglicht zum Beispiel die Kombination von UTF-16-Text mit ASCII-kodierten Inline-Optionslisten (der Text kann zum Beispiel aus einer Unicode-Datenbank stammen, während die Inline-Optionen aus ASCII-Text innerhalb der Anwendung bestehen). Standardwert:* `false`
textlen	*(Integer oder Schlüsselwort; erforderlich bei Textteilen mit* `fixedtextformat=false` *und* `textformat=utf16xx` *in nicht Unicode-fähigen Sprachen) Anzahl der Bytes oder (in Unicode-fähigen Sprachen) Zeichen vor der nächsten Inline-Optionsliste (siehe »Inline-Optionslisten für Textflows«, Seite 112). Die Zeichen werden gezählt, bevor Character-Referenzen aufgelöst werden, zum Beispiel* `<textlen=8>&#x2460;<...>`*. Das Schlüsselwort* `all` *bezeichnet den gesamten verbleibenden Text. Standardwert: der Text wird nach dem nächsten durch* `begoptlistchar` *definierten Zeichen durchsucht.*

Inline-Optionslisten für Textflows. Der im Parameter *text* für *PDF_create_textflow( )* (aber nicht für *PDF_add_textflow( )*) übergebene Inhalt kann eine beliebige Anzahl von Optionslisten (Inline-Optionen) mit Textflow-Optionen gemäß Tabelle 5.11 enthalten. All diese Optionen können auch im Parameter *optlist* von *PDF_create_textflow( )* und *PDF_add_textflow( )* übergeben werden. Ein und dieselbe Option kann in einer Optionsliste mehrmals vorkommen; gültig ist dann nur der zuletzt eingestellte Wert.

Inline-Optionslisten müssen in die Zeichen eingeschlossen sein, die in den Optionen *begoptlistchar* und *endoptlistchar* festgelegt werden (standardmäßig die Zeichen < und >). Es kann problematisch werden, wenn das Zeichen, das eine Inline-Optionsliste einleitet, auch im Text verwendet werden soll. Zur Behebung dieses Problems gibt es verschiedene Möglichkeiten, deren Einsatz davon abhängt, ob der Text Inline-Optionslisten enthält oder nicht. Bei *PDF_add_textflow( )* tritt das Problem generell nicht auf, da dort Text und Optionen vollständig voneinander getrennt sind.

Enthält der Text keine Inline-Optionslisten, können Sie die Interpretation von Inline-Optionslisten auf eine der folgenden Arten vollständig deaktivieren:

- Setzen Sie im Parameter *optlist* für *PDF_create_textflow( )* die Option *begoptlistchar=none*.
- Setzen Sie im Parameter *optlist* für *PDF_create_textflow( )* die Option *textlen* auf die Gesamtlänge des Textes.

Enthält der Text Inline-Optionslisten, können Sie den Konflikt zwischen tatsächlichem Textinhalt und dem mit *begoptlistchar* definierten Zeichen zur Einleitung einer Inline-Optionsliste auf eine der folgenden Arten vermeiden:

- Ersetzen Sie das Zeichen < überall im eigentlichen Text durch die entsprechende numerische oder Entity-Referenz (*<* oder *<)* und benutzen Sie das Zeichen < nur zum Einleiten von Inline-Optionslisten:

```
A&lt;B<fontname=Helvetica encoding=winansi>
```

- Legen Sie mit der Option *begoptlistchar* im Parameter *optlist* für *PDF_create_textflow( )* ein Zeichen (zum Beispiel *$)* fest, das im Text nicht verwendet wird, und starten Sie Inline-Optionslisten mit diesem Zeichen:

```
<begoptlistchar=$>A<B$fontname=Helvetica encoding=winansi>
```

- Legen Sie in jeder Inline-Optionsliste mit der Option *textlen* die Länge des nächsten Textfragments fest (bis zum Anfang der nächsten Inline-Optionsliste):

```
<textlen=3>A<B<fontname=Helvetica encoding=winansi>
```

Hinweis *Schließt eine Inline-Optionsliste unmittelbar an die vorangehende Optionsliste an, wird davon ausgegangen, dass sie ein Textfragment der Länge 0 enthalten. Dies ist von Bedeutung, wenn in der ersten Optionsliste die Option* `textlen` *übergeben wird.*

C++ Java C# ***String fit_textflow(int textflow, double llx, double lly, double urx, double ury, String optlist)***

Perl PHP ***string fit_textflow(int textflow, float llx, float lly, float urx, float ury, string optlist)***

C ***const char *PDF_fit_textflow(PDF *p,***
int textflow, double llx, double lly, double urx, double ury, const char *optlist)

Gibt den nächsten Abschnitt eines Textflows aus.

textflow Textflow-Handle, das von *PDF_create_textflow( )* oder *PDF_add_textflow( )* zurückgegeben wurde.

llx, lly, urx, ury *x*- und *y*-Koordinaten der linken unteren und rechten oberen Ecke des Ausgaberechtecks *(Fitbox)* in Benutzerkoordinaten. Die Ecken können auch in umgekehrter Reihenfolge festgelegt werden. Nicht rechteckig begrenzte Bereiche lassen sich mit der Option *wrap* füllen.

optlist Optionsliste mit Verarbeitungsoptionen. Folgende Optionen können verwendet werden:

- Textflow-Optionen gemäß Tabelle 5.12:
  avoidwordsplitting, blind, createfittext, createlastindent, exchangefillcolors, exchangestrokecolors, firstlinedist, fitmethod, fontscale, lastlinedist[1]*, linespreadlimit, maxlines, minfontsize, orientate, returnatmark, rewind, rotate, showborder, showtabs, stamp, truncatetrailingwhitespace, verticalalign*[1]*, wrap*
- Matchbox-Option gemäß Tabelle 6.1: *matchbox*
- Option zum vereinfachten Tagging von Strukturelementen gemäß Tabelle 14.5 (nur zulässig im Gültigkeitsbereich *page*): *tag*

Rückgabe String mit der Ursache für das Beenden der Funktion:

- *_stop:* der Textfluss wurde vollständig verarbeitet. War der Text leer, wird *_stop* trotzdem übergeben, auch wenn *return* oder die Option *mark/returnatmark* übergeben wurde.
- *_nextpage:* die nächste Seite wird erwartet (verursacht durch das *formfeed*-Zeichen U+000C). Zur Platzierung des restlichen Texts ist ein weiterer Aufruf von *PDF_fit_textflow( )* erforderlich.
- *_boxfull:* die Fitbox ist voll, die maximale mit der Option *maxlines* festgelegte Zeilenanzahl ist erreicht oder *fitmethod=auto* und *minfontsize* wurde spezifiziert, der Text passte jedoch nicht vollständig in die Box. Zur Platzierung des restlichen Texts ist ein weiterer Aufruf von *PDF_fit_textflow( )* erforderlich.
- *_boxempty:* die Fitbox enthält nach der Verarbeitung keinerlei Text. Dies kann passieren, wenn die Fitbox so klein ist, dass kein Text hineinpasst oder eine Wrapping-Box größer als die Fitbox war. Um Endlosschleifen zu vermeiden, sollten keine weiteren Aufrufe von *PDF_fit_textflow( )* mit derselben Fitbox erfolgen.
- *_mark#:* Die Option *returnatmark* wurde auf Nummer # gesetzt und die Marke mit der in dieser Option festgelegten Nummer platziert.
- Ein anderer String: der vom Befehl *return* in einer Inline-Optionsliste übergebene String.

Gibt es mehrere Gründe für das Beenden der Funktion, so wird derjenige gewählt, der in obiger Liste zuerst aufgeführt ist. Der Rückgabe-String ist bis zum nächsten Aufruf der Funktion gültig.

Details Anders als bei *PDF_fit_textline( )* wird die von dieser Funktion erzeugte Textausgabe vom aktuellen Text- und Grafikzustand nicht beeinflusst. Die Textdarstellung lässt sich mit *fillcolor*, *strokecolor* und anderen Darstellungsoptionen für *PDF_create_textflow( )* oder *PDF_add_textflow( )* steuern (siehe Tabelle 4.7). Nach der Rückkehr von dieser Funktion ist der Textzustand unverändert. Die Optionen *textx/texty* werden jedoch auf das Ende des generierten Ausgabetextes gesetzt (sofern die Option *blind* nicht auf *true* gesetzt wurde).

Gültigkeit *page, pattern, template, glyph*

Tabelle 5.12 Optionen für PDF_fit_textflow( )

Option	Beschreibung
avoidwordsplitting	*(Boolean) Bei* `true` *und* `fitmethod=auto` *wird versucht, den Text durch Reduzierung der Fontgröße und unter Vermeidung von Silbentrennung vollständig in der Fitbox unterzubringen (siehe* `adjustmethod`*).*
blind	*(Boolean) Bei* `true` *wird zwar keine Ausgabe generiert, aber alle Berechnungen durchgeführt. Die Formatierungsergebnisse können mit PDF_info_textflow( ) abgefragt werden. Standardwert:* `false`
createfittext	*(Boolean) Bei* `true` *wird der in der aktuellen Fitbox platzierte Text im Arbeitsspeicher gespeichert und kann später mit einem Aufruf von PDF_info_textflow( ) sowie dem Schlüsselwort* `fittext` *wieder abgerufen werden. Standardwert:* `true`

Tabelle 5.12 Optionen für PDF_fit_textflow( )

Option	Beschreibung
createlast-indent	*(Optionsliste) Reserviert am Ende der letzten Zeile Platz in der Fitbox und erzeugt optional eine Matchbox, die zum Füllen des reservierten Platzes verwendet werden kann. Der reservierte Platz kann zum Beispiel für zusätzliche Punkte, ein Bild oder einen Link auf den Folgetext verwendet werden. Unterstützte Unteroptionen:* **rightindent** *(Float oder Prozentwert) Zusätzliche rechte Einrückung der letzten Textzeile in der Fitbox in Benutzerkoordinaten oder als Prozentwert der Breite der Fitbox. Der Wert wird zum Wert der Option* `rightindent` *von PDF_add/create_textflow( ) addiert. Standardwert: 0* **matchbox** *(Optionsliste gemäß Tabelle 6.4) Erzeugt eine Matchbox am Ende der letzten Zeile. Wird die Matchbox-Option* `boxwidth` *nicht angegeben, wird als Breite der Box der Wert von* `rightindent` *verwendet. Bei* `boxwidth=0` *wird keine Box erzeugt.*
exchange-fillcolors	*(Liste mit einer geraden Anzahl von Farben) Jedes Paar in der Liste stellt eine originale Füllfarbe und eine Ersatzfarbe dar. Immer wenn in der Fitbox für den Textflow die Originalfarbe angegeben ist, wird sie durch die Ersatzfarbe ersetzt. Dies kann für die Anpassung an den Hintergrunde nützlich sein.* *Zum Beispiel:* `exchangefillcolors={{gray 0} white Orchid DeepPink {rgb 1 0 1} MediumBlue}`
exchange-strokecolors	*(Liste mit einer geraden Anzahl von Farben) Jedes Paar in der Liste stellt eine Originalfarbe und eine Ersatzfarbe für Umrisslinien dar. Immer wenn in der Fitbox für den Textflow die Originalfarbe angegeben ist, wird sie durch die Ersatzfarbe ersetzt. Dies kann für die Anpassung an den Hintergrund nützlich sein.*
firstlinedist[1]	*(Float, Prozentwert oder Schlüsselwort) Abstand zwischen dem oberen Rand der Fitbox und der Grundlinie der ersten Textzeile. Angegeben wird er in Benutzerkoordinaten, als Schlüsselwort oder als Prozentwert der Fontgröße, d.h. bei* `fixedleading=true` *der Größe des ersten in der Zeile auftretenden Fonts und bei* `fixedleading=false` *der höchsten in der Zeile auftretenden Fontgröße (Standardwert:* `leading`*):* **leading** *Für die erste Zeile ermittelter Zeilenabstand; diakritische Zeichen wie* Ä *berühren dabei den oberen Rand der Fitbox.* **ascender** *Für die erste Zeile ermittelte Oberlänge; Zeichen mit großer Oberlänge wie d oder* h *berühren dabei den oberen Rand der Fitbox.* **capheight** *Für die erste Zeile ermittelte Versalhöhe; hohe Großbuchstaben wie* H *berühren dabei den oberen Rand der Fitbox.* **xheight** *Für die erste Zeile ermittelte x-Höhe; Kleinbuchstaben wie* x *berühren dabei den oberen Rand der Fitbox.* *Ist* `fixedleading=false`*, wird der größte Wert verwendet, der für Zeilenabstand, Oberlänge, x-Höhe und Versalhöhe in der ersten Zeile ermittelt wurde.*
fitmethod	*(Schlüsselwort) Bestimmt die Methode zum Einpassen des Textes in die Fitbox (Standardwert:* `clip`*):* **auto** *PDF_fit_textflow( ) wird solange im blind-Modus, jeweils reduzierter Fontgröße und anderen fontbezogenen Optionen aufgerufen (siehe* `fontscale`*), bis der Text in die Fitbox passt (siehe auch* `minfontsize`*).* **clip** *Der Text wird am unteren Rand der Fitbox abgeschnitten.* **nofit** *Der Text darf über den unteren Rand der Fitbox hinausreichen.*
fontscale	*(Positiver Float oder Prozentwert) Der Wert von* `fontsize` *sowie die Absolutwerte (nicht die Prozentwerte) von* `leading`*,* `minspacing`*,* `maxspacing`*,* `spreadlimit` *und* `space` *werden mit dem übergebenen Skalierungsfaktor oder Prozentwert multipliziert. Standardwert: 1, wenn* `rewind=0`*, sonst der Wert, der mit dem entsprechenden Aufruf von PDF_fit_textflow( ) übergeben wird*
gstate	*(Gstate-Handle) Handle für einen mit PDF_create_gstate( ) erzeugten Grafikzustand. Der Grafikzustand wirkt sich auf den gesamten mit dieser Funktion erzeugten Text aus. Wurde bereits ein anderer Grafikzustand an PDF_add/create_textflow( ) übergeben, werden beide Grafikzustände zusammengeführt. Standardwert: kein Grafikzustand (das heißt, die aktuellen Einstellungen werden verwendet).*

Tabelle 5.12 Optionen für PDF_fit_textflow( )

Option	Beschreibung
lastlinedist[1]	*(Float, Prozentwert oder Schlüsselwort; wird ignoriert bei* `fitmethod=nofit`*) Der kleinste Abstand zwischen der Grundlinie der letzten Textzeile und dem unteren Rand der Fitbox. Angegeben wird er in Benutzerkoordinaten, als Schlüsselwort oder als Prozentsatz der Fontgröße, d.h. bei* `fixedleading=true` *der Größe des ersten in der Zeile auftretenden Fonts und bei* `fixedleading=false` *der höchsten in der Zeile auftretenden Fontgröße. Standardwert: 0, d.h. der untere Rand der Fitbox wird als Grundlinie verwendet und die normalen Unterlängen reichen aus der Fitbox hinaus. Unterstütztes Schlüsselwort:* **descender** *Für die letzte Zeile ermittelte Unterlänge; Zeichen mit Unterlängen wie* `g` *oder* `j` *berühren dabei den unteren Rand der Fitbox. Ist* `fixedleading=false`*, wird der größte Wert verwendet, der in der letzten Zeile für die Unterlänge ermittelt wurde.*
linespread-limit	*(Float oder Prozentwert; nur für* `verticalalign=justify`*) Größter Wert in Benutzerkoordinaten oder als Prozentsatz des Zeilenabstands, um den der Zeilenabstand bei vertikaler Ausrichtung erhöht wird. Standardwert: 200%*
maxlines	*(Integer oder Schlüsselwort) Maximale Anzahl der Zeilen in der Fitbox oder das Schlüsselwort* `auto`*, bei dem möglichst viele Zeilen in der Fitbox platziert werden. Nach der Platzierung der maximalen Anzahl von Zeilen gibt PDF_fit_textflow( ) den String* `_boxfull` *zurück. Standardwert:* `auto`
minfontsize	*(Float oder Prozentwert) Minimal zulässige Fontgröße bei der Skalierung von Text zum Einpassen in die Fitbox; diese ist insbesondere bei* `fitmethod=auto` *relevant. Diese Grenze wird in Benutzerkoordinaten oder prozentual zur Höhe der Fitbox angegeben. Ist die Grenze erreicht und passt der Text immer noch nicht, so wird der String* `_boxfull` *zurückgegeben. Standardwert: 0.1%*
mingapwidth	*(Float oder Prozentwert) Minimale horizontale Breite für die Einpassung von Text zwischen Formen (z.B. zwischen Wrapping-Konturen) in Benutzerkoordinaten oder als Prozentwert der Fontgröße. Dies kann zur Vermeidung unschöner Formatierungsergebnisse bei sehr kleinen Lücken zwischen Wrapping-Konturen nützlich sein. Standardwert: 10%*
orientate	*(Schlüsselwort) Ausrichtung des Texts bei der Platzierung (Standardwert:* `north`*):* **north** *nach oben* **east** *nach rechts* **south** *nach unten* **west** *nach links*
returnatmark	*(Integer) PDF_fit_textflow( ) kehrt vorzeitig an der Textposition zurück, an der die Option* `mark` *mit der übergebenen Nummer definiert wurde. Der Return-String ist* `_mark#`*, wobei # die in dieser Option spezifizierte Nummer bezeichnet.*
rewind	*(Integer: -2, -1, 0 oder 1) Der Zustand des übergebenen Textflusses wird auf den Zustand vor einem bestimmten Aufruf von PDF_fit_textflow( ) mit demselben Textflow-Handle zurückgesetzt (Standardwert:* `0`*):* **1** *Zustand vor dem ersten Aufruf von PDF_fit_textflow( )* **0** *Kein Zurücksetzen* **-1** *Zustand vor dem letzten Aufruf von PDF_fit_textflow( )* **-2** *Zustand vor dem vorletzten Aufruf von PDF_fit_textflow( )*
rotate	*(Float) Dreht das Koordinatensystem, wobei die linke untere Ecke der Fitbox als Mittelpunkt und der übergebene Wert als Drehwinkel in Grad benutzt wird. Dabei werden die Box und der Text gedreht. Die Drehung wird gesetzt, nachdem der Text platziert wurde. Standardwert: 0*
showborder	*(Boolean) Bei* `true` *wird die Umrandung der Fitbox und aller Wrapping-Boxen gezeichnet (mit dem aktuellen Grafikzustand). Dies kann bei Entwicklung und Fehlersuche nützlich sein. Standardwert:* `false`

Tabelle 5.12 Optionen für PDF_fit_textflow( )

Option	Beschreibung
showtabs	*(Schlüsselwort) Tabulatoren und linke Einrückungen werden zur Hilfe bei der Fehlersuche mit vertikalen Linien veranschaulicht. Die Linien werden gemäß des Grafikzustands gezeichnet, der vor dem Aufruf von PDF_fit_textflow( ) aktiviert war (Standardwert:* `none`*):* **none** *Es werden keine Linien gezeichnet.* **fitbox** *Linien werden über die volle Höhe der Fitbox gezeichnet.* **validarea** *Linien werden nur in der Höhe gezeichnet, in der sie relevant sind.*
stamp	*(Schlüsselwort) Mit dieser Option lässt sich ein diagonaler Stempel in der Fitbox erstellen. Zeilenumbrüche sollten explizit festgelegt werden (z.B. mit* `newline`*-Zeichen oder der Option* `newline`*) Enthält der Text keine expliziten Zeilenumbrüche, wird ein einzeiliger Stempel erzeugt. Der Stempeltext wird dabei so groß wie möglich, aber nicht größer als die angegebene Fontgröße ausgegeben. Unterstützte Schlüsselwörter (Standardwert:* `none`*):* **ll2ur** *Der Stempel verläuft diagonal von der linken unteren in die rechte obere Ecke.* **ul2lr** *Der Stempel verläuft diagonal von der linken oberen in die rechte untere Ecke.* **none** *Es wird kein Stempel erstellt.*
truncate-trailing-whitespace	*(Boolean) Behandlung von Fitboxen, die nur aus Weißraum bestehen, das heißt, der Text in der Fitbox beginnt mit Weißraum und enthält bis zum Ende des Textflows nichts als Weißraum. Bei* `true` *wird der Weißraum entfernt und die Fitbox als leer behandelt, der Rückgabewert ist in diesem Fall* `_stop`*. Bei* `false` *wird der Weißraum wie normaler Text verarbeitet. Die Funktion liefert dann möglicherweise einen anderen Wert als* `_stop` *zurück (abhängig von der Menge des Weißraums), und der Weißraum wird von den Schlüsselwörtern* `textendx/y` *und anderen Schlüsselwörtern von PDF_info_textflow( ) berücksichtigt.* `truncatetrailingwhitespace=false` *kann nützlich sein, wenn der Originaltext ohne die Entfernung von Weißraum verarbeitet werden muss. Standardwert:* `true`

Tabelle 5.12 Optionen für PDF_fit_textflow( )

Option	Beschreibung
verticalalign[1]	*(Schlüsselwort) Vertikale Ausrichtung des Texts in der Fitbox; die Optionen* `firstlinedist` *und* `lastlinedist` *werden entsprechend berücksichtigt. Unterstützte Schlüsselwörter (Standardwert:* `top`*):*
	top *Die Formatierung beginnt in der ersten Zeile und setzt sich nach unten fort. Füllt der Text die Fitbox nicht vollständig aus, bleibt Weißraum unter dem Text.*
	center *Der Text wird vertikal in der Fitbox zentriert. Füllt der Text die Fitbox nicht vollständig aus, bleibt Weißraum über und unter dem Text.*
	bottom *Die Formatierung beginnt in der letzten Zeile und setzt sich nach oben fort. Füllt der Text die Fitbox nicht vollständig aus, bleibt Weißraum über dem Text.*
	justify *Der Text wird am oberen und unteren Rand der Fitbox ausgerichtet. Dazu wird der Zeilenabstand bis zur durch* `linespreadlimit` *festgelegten Grenze erhöht. Die Höhe der ersten Zeile wird nur bei* `firstlinedist=leading` *vergrößert.*
wrap	*(Optionsliste gemäß Tabelle 5.13) Der Text spart die Bereiche aus, die mit den in Tabelle 5.13 aufgeführten Unteroptionen festgelegt werden. Dies kann zur Platzierung von Grafiken innerhalb des Textflows verwendet werden, um die der Text herumfließen soll, oder zum Füllen beliebig begrenzter Bereiche mit Text. Die Fitbox wird gemäß der Even-Odd-Regel gefüllt, wobei am Rand der Fitbox begonnen wird.* *Standardmäßig enthalten die definierten Bereiche keinen Text (außer an Überlappungen), d.h. der Text fließt um die Bereiche herum. Dies kann zur Platzierung von Grafiken innerhalb des Bereichs verwendet werden.* *Mit den Optionen* `addfitbox` *und* `inversefill` *lässt sich der gegenteilige Effekt erzielen: Die festgelegten Bereiche werden mit Text gefüllt, und der Rest der Fitbox bleibt leer. Damit lassen sich beliebig begrenzte Bereiche (und nicht nur Rechtecke) mit Text füllen.* *Absolute Koordinatenwerte werden im Benutzerkoordinatensystem interpretiert; Prozentwerte werden im Koordinatensystem der Fitbox interpretiert, d.h. die linke untere Ecke der Fitbox ist (0, 0) und die rechte obere Ecke ist (100, 100). Es können maximal 256 Prozentwerte übergeben werden. Beispiele:* *Ausschluss einer Box mit relativen Koordinaten:* `wrap={ boxes={{120r 340r 50r 60r}} }` *(äquivalent zu* `wrap={ boxes={{120 340 170 400}} }` *Ausschluss des rechten oberen Viertels der Fitbox:* `wrap={ boxes={{50% 50% 100% 100%}} }` *Füllen eines Dreiecks:* `wrap={ addfitbox polygons={{50% 80% 30% 40% 70% 40% 50% 80%}} }` *Ausschluss eines Bildbereichs anhand einer Matchbox namens* `image1`*:* `wrap={ usematchboxes={{ image1 }}}`

1. Die Optionen `firstlinedist`, `lastlinedist` *und* `verticalalign` *beziehen sich selbst bei vorhandenen Wrapping-Elementen immer auf die Fitbox. Insbesondere beim inversen Füllen, wenn also Wrapping-Elemente mit Text gefüllt werden, verwendet der Textflow-Formatierer daher nicht die Boundingbox der Wrapping-Elemente, um den Abstand zwischen dem Text, den Rändern der Fitbox und der Position der Textbox gemäß der Option* `verticalalign` *zu ermitteln. Dies kann zu unerwarteten Ergebnissen führen, vor allem, wenn die äußeren Ecken der Wrapping-Elemente die Fitbox nicht berühren. Dies kann fast immer dadurch vermieden werden, indem nur Wrapping-Elemente übergeben werden, die die Fitbox berühren.*

Tabelle 5.13 Unteroptionen für die Option wrap *von PDF_fit_textflow( )*

Option	Beschreibung
addfitbox	*(Boolean) Bei* true *wird die Fitbox zum Wrapping-Bereich hinzugefügt, so dass die mit anderen Wrapping-Optionen definierten Bereiche mit Text gefüllt werden, anstatt dass sie von Text ausgespart bleiben. Standardwert:* false
beziers	*(Liste aus zwei oder mehr Bézierkurven) Bézierkurven, die zum Wrapping-Bereich hinzugefügt werden.*
boxes	*(Liste aus Rechtecken) Ein oder mehrere Rechtecke, die zum Wrapping-Bereich hinzugefügt werden.*
circles	*(Liste aus Kreisen) Ein oder mehrere Kreise, die zum Wrapping-Bereich hinzugefügt werden.*
creatematchboxes	*(Liste aus Optionslisten) Erzeugt Matchboxen aus einem oder mehreren Rechtecken in der Option* boxes*. Jede Optionsliste bezieht sich auf einen Eintrag in der Option* boxes *(die Reihenfolge ist relevant) und steuert die Erzeugung einer Matchbox. Alle relevanten Matchbox-Optionen aus Tabelle 6.4 können verwendet werden. Eine Unteroption kann leer sein; in diesem Fall wird keine Matchbox für die entsprechende Wrapping-Box erzeugt.*
fillrule	*(Schlüsselwort) Legt die Methode fest, mit der das Innere von überlappenden Bereichen ermittelt wird (Standardwert:* evenodd*) . Für weitere Informationen siehe Tabelle 7.1:* **evenodd** *Die Even-Odd-Regel wird verwendet.* **winding** *Die Nonzero-Winding-Number-Regel wird verwendet. Hiermit können Sie das Innere überlappender Kreise einbeziehen (um zum Beispiel »Doughnut-Löcher« zu vermeiden) oder die Vereinigungsmenge überlappender Bereiche (statt deren Schnittmenge).*
inversefill	*(Boolean) Bei* true *beginnt die Ermittlung des Wrapping-Bereichs dort, wo die Textzeile zum ersten Mal den Rand eines Wrapping-Elements innerhalb der Fitbox schneidet. Bei* false *beginnt die Ermittlung am Rand der Fitbox. Bei* fillrule=evenodd *besitzt die Option* inversefill=true *dieselbe Wirkung wie* addfitbox=true*. Bei* fillrule=winding *führt die Option* addfitbox=true *zu einer leeren bzw. vollen Fitbox (je nachdem, ob* inversefill=false *bzw.* true *ist).*
lineheight	*(Liste aus zwei Elementen, die jeweils ein positiver Float-Wert oder ein Schlüsselwort sind) Definiert die vertikale Ausdehnung der Textzeile, die zur Berechnung der Überlappungen von Wrapping-Bereichen verwendet wird. Zwei Schlüsselwörter/Floats legen die Ausdehnung über und unter der Grundlinie fest. Unterstützte Schlüsselwörter:* none *(keine Ausdehnung),* xheight, descender, capheight, ascender, fontsize, leading, textrise *Standardwert:* {ascender descender}
usematchboxes	*(Liste aus String-Listen) Das jeweils erste Listenelement entspricht einem Name-String zur Festlegung einer Matchbox. Das zweite Element ist entweder ein Integer zur Festlegung der Nummer des gewünschten Rechtecks oder das Schlüsselwort* all *für alle Rechtecke der ausgewählten Matchbox. Fehlt das zweite Element, so wird* all *als Standardwert verwendet. Die Boundingboxen der Rechtecke werden zum Wrapping-Bereich hinzugefügt.*
offset	*(Float oder Prozentwert) Horizontaler Abstand zwischen dem Text und der Kontur des Wrapping-Bereichs in Benutzerkoordinaten oder als Prozentsatz der Fitboxbreite. Dieser kann zur horizontalen Vergrößerung des Wrapping-Bereichs verwendet werden. Standardwert: 0*
paths	*(Liste aus Optionslisten) Ein oder mehrere Pfadobjekte, die zum Wrapping-Bereich hinzugefügt werden. Unterstützte Unteroptionen:* **path** *(Pfad-Handle; erforderlich) Handle für den Pfad, der zum Wrapping-Bereich hinzugefügt wird.* **refpoint** *(Liste aus zwei Floats oder Prozentwerten) Koordinaten des Referenzpunkts für den Pfad in Benutzerkoordinaten oder als Prozentwert der Breite und Höhe der Fitbox. Standardwert:* {0 0} *Die folgenden Optionen von PDF_draw_path( ) können ebenfalls verwendet werden (siehe Tabelle 6.1 und Tabelle 7.7):* align, attachmentpoint, boxsize, close, fitmethod, orientate, boxsize, round, scale, subpaths

Tabelle 5.13 Unteroptionen für die Option `wrap` *von PDF_fit_textflow( )*

Option	Beschreibung
polygons	*(Liste aus Polylinien) Eine oder mehrere (nicht unbedingt geschlossene) Polylinien, die zum Wrapping-Bereich hinzugefügt werden.*

C++ Java C# **double info_textflow(int textflow, String keyword)**

Perl PHP **float info_textflow(int textflow, string keyword)**

C **double PDF_info_textflow(PDF *p, int textflow, const char *keyword)**

Ermittelt den aktuellen Zustand eines Textflows nach einem Aufruf von *PDF_fit_textflow( )*.

textflow Textflow-Handle, das von *PDF_add/create_textflow( )* oder von *PDF_fill_textblock( )* mit der Option *textflowhandle* zurückgegeben wurde.

keyword Schlüsselwort für die abzufragende Information gemäß Tabelle 5.14.

Rückgabe Der Wert des mit *keyword* abgefragten Textflow-Parameters. Im Gegensatz zu den Optionen *textx* und *texty* gibt diese Funktion auch im Blind-Modus korrekte geometrische Informationen zurück. Wenn das angegebene Schlüsselwort Text liefert, wird ein String-Index zurückgegeben und der entsprechende String muss mit *PDF_get_string( )* abgerufen werden.

Gültigkeit beliebig außer *object*

Tabelle 5.14 Schlüsselwörter für PDF_info_textflow( )

Schlüsselwort	Beschreibung
boundingbox	*Handle des Pfades in Benutzerkoordinaten oder -1 (0 in PHP), der die Boundingbox des Textflows enthält.* `firstlinedist` *und* `lastlinedist` *werden berücksichtigt.*
boxlinecount	*Anzahl der Zeilen in der letzten Fitbox*
firstparalinecount	*Anzahl der Zeilen im ersten Absatz der Fitbox*
firstlinedist	*Abstand zwischen der ersten Grundlinie des Texts und einer fiktiven Grundlinie darüber (bei* `verticalalign=top` *entspricht diese dem oberen Rand der Fitbox)*
fittext	*String-Index für den Text, der im letzten Aufruf von PDF_fit_textflow( ) platziert wurde. Hiermit lässt sich die Textmenge ermitteln, die in der Fitbox platziert werden konnte. Der String wird folgendermaßen normalisiert: UTF-16-Encoding in Unicode-fähigen Sprachbindungen, sonst (EBCDIC-)UTF-8; Zeilenumbrüche werden mit U+000A markiert und horizontale Tabulatoren werden durch ein Leerzeichen U+0020 ersetzt.*
fontscale	*Der Wert von* `fontscale` *nach dem letzten Aufruf von PDF_fit_textflow( ) mit* `fitmethod=auto`.
lastfont	*Font-Handle, das in der letzten Textzeile der Fitbox verwendet wird*
lastfontsize	*Fontgröße, die in der letzten Textzeile der Fitbox verwendet wird*
lastlinedist	*Abstand zwischen der letzten Grundlinie des Texts und einer fiktiven Grundlinie darunter (bei* `verticalalign=bottom` *entspricht diese dem unteren Rand der Fitbox), wobei von einem unveränderten Zeilenabstand ausgegangen wird*
lastmark	*Nummer der letzten Marke im letzten Textflow-Abschnitt, der in der letzten Fitbox platziert wurde (Marken lassen sich mit der Option* `mark` *setzen)*
lastparalinecount	*Anzahl der Zeilen im letzten Absatz der Fitbox*

Tabelle 5.14 Schlüsselwörter für PDF_info_textflow( )

Schlüsselwort	Beschreibung
leading	*Aktueller Wert der Option* `leading`*, der sich durch den Text und die Optionen im Textflow ergibt*
leftlinex[1] **leftliney**[1]	*x- und y-Koordinaten der Zeile in der letzten Fitbox, die am weitesten links beginnt, in aktuellen Benutzerkoordinaten*
maxlinelength	*Länge der längsten Textzeile in der zuletzt gefüllten Fitbox*
maxliney[1]	*y-Koordinate der Grundlinie der längsten Textzeile in der zuletzt gefüllten Fitbox in aktuellen Benutzerkoordinaten*
minlinelength	*Länge der kürzesten Textzeile in der zuletzt gefüllten Fitbox*
minliney[1]	*y-Koordinate der Grundlinie der kürzesten Textzeile in der zuletzt gefüllten Fitbox in aktuellen Benutzerkoordinaten*
returnreason	*String-Index für die Rückkehrursache des letzten direkten oder indirekten Aufrufs von PDF_fit_textflow( ). Dies ist bei der Abfrage des Ergebnisses von indirekten Textflow-Aufrufen nützlich, die intern von PDF_fill_textblock( ) durchgeführt werden.*
rightlinex[1], **rightliney**[1]	*x- und y-Koordinaten der Zeile in der letzten Fitbox, die am weitesten rechts endet, in aktuellen Benutzerkoordinaten*
split	*Zeigt an, ob in der letzten Fitbox Wörter zwangsweise getrennt wurden:* **0** *Kein Wort wurde zwangsweise getrennt.* **1** *Mindestens ein Wort wurde zwangsweise getrennt.*
textendx, textendy	*x- bzw. y-Koordinate der aktuellen Position nach Platzierung des Texts in aktuellen Benutzerkoordinaten*
textheight	*Höhe der Boundingbox des gesamten Texts (unter Berücksichtigung von* `firstlinedist` *und* `lastlinedist`*) in aktuellen Benutzerkoordinaten*
textwidth	*Breite der Boundingbox des gesamten Texts in aktuellen Benutzerkoordinaten*
used	*Prozentualer Anteil (0...100) des bereits platzierten Texts*
x1, y1, ... , x4, y4	*Koordinaten der Boundingbox des gesamten Texts (unter Berücksichtigung von* `firstlinedist` *und* `lastlinedist`*) in aktuellen Benutzerkoordinaten*

1. Ist `rotate` *ungleich 0, so bezieht sich dieser Wert auf das gedrehte System.*

C++ Java C# **void delete_textflow(int textflow)**

Perl PHP **delete_textflow(int textflow)**

C **void PDF_delete_textflow(PDF *p, int textflow)**

Löscht einen Textflow und alle damit verbundenen Datenstrukturen.

textflow Textflow-Handle, das von *PDF_create_textflow( ) oder PDF_add_textflow( )* zurückgegeben wurde.

Details Am Ende des umschließenden Gültigkeitsbereichs *document* werden alle nicht mit dieser Funktion gelöschten Textflows automatisch gelöscht. Es kann die Geschwindigkeit Ihrer Anwendung jedoch erheblich beeinträchtigen, wenn Sie viele Textflows generieren, *PDF_delete_textflow( )* aber nicht benutzen.

Gültigkeit beliebig

5.3 Tabellenformatierung

Cookbook *Ein vollständiges Codebeispiel hierzu finden Sie im Cookbook-Topic* `tables/starter_table`.

C++ Java C# ***int add_table_cell(int table, int column, int row, string text, string optlist)***

Perl PHP ***int add_table_cell(int table, int column, int row, string text, string optlist)***

C ***int PDF_add_table_cell(PDF *p,***
int table, int column, int row, const char *text, int len, const char *optlist)

Fügt eine Zelle zu einer neuen oder bereits vorhandenen Tabelle hinzu.

table Gültiges Tabellen-Handle, das von einem früheren Aufruf von *PDF_add_table_cell( )* zurückgegeben wurde, oder -1 (in PHP: 0) beim ersten Aufruf. Das Tabellen-Handle darf nicht bereits in *PDF_fit_table( )* verwendet worden sein, d.h. dass alle Tabelleninhalte definiert werden müssen, bevor die Tabelle auf der Seite platziert wird.

column, row Nummer der Spalte und der Zeile, die die Zelle enthalten. Erstreckt sich die Zelle über mehrere Spalten und/oder Zeilen, muss die Nummer der am weitesten links liegenden Spalte und die Nummer der obersten Zeile übergeben werden. Die erste Spalte/Zeile hat die Nummer 1.

text (Content-String) Text zum Füllen der Zelle. Ist *text* nicht leer, so wird er zum Füllen der Zelle mit *PDF_fit_textline( )* verwendet.

len (Nur C-Sprachbindung) Länge von *text* (in Bytes). Ist *len* = 0, muss ein null-terminierter String übergeben werden.

optlist Optionsliste mit Details zur Formatierung der Tabellenzelle gemäß Tabelle 5.15. Folgende Optionen können verwendet werden:

- Allgemeine Option: *errorpolicy* (siehe Tabelle 2.1)
- Spalten- und Zeilendefinition: *colwidth, colscalegroup, minrowheight, return, rowheight, rowjoingroup, rowscalegroup*
- Zellendefinition: *avoidwordsplitting, colspan, margin, marginleft, marginbottom, marginright, margintop, rowspan*
- Cell content formatting: *continuetextflow, repeatcontent*
- Statischer Zelleninhalt gemäß Tabelle 5.16: *fitgraphics, fitimage, fitpath, fitpdipage, fittextflow, fittextline, graphics, image, matchbox, path, pdipage, textflow*
- Interaktiver Zelleninhalt gemäß Tabelle 5.17 (nur im Gültigkeitsbereich *scope*): *annotationtype, fieldname, fieldtype, fitannotation, fitfield*
- Option zum vereinfachten Tagging von Strukturelementen gemäß Tabelle 14.5: *tag*

Rückgabe Tabellen-Handle, das in nachfolgenden Tabellenfunktionen verwendet werden kann. Bei *errorpolicy=return* muss der Aufrufer den Rückgabewert auf -1 (in PHP: 0) überprüfen, da dies auf einen Fehler hinweist. Im Fehlerfall wird nur die letzte Zellendefinition verworfen; es wird zwar kein Inhalt zur Tabelle hinzugefügt, das Tabellen-Handle ist aber noch gültig. Das zurückgegebene Tabellen-Handle darf nicht in mehreren PDF-Ausgabedokumenten verwendet werden.

Details Eine Tabellenzelle kann mit Bildern, Grafiken, importierten PDF-Seiten, Textflows oder Textzeilen gefüllt werden. In einem Funktionsaufruf können auch mehrere Inhaltstypen für eine Zelle festgelegt werden.

Im PDFlib-Tutorial wird der Algorithmus für die Tabellenformatierung sowie die Breiten- und Höhenberechnung beschrieben.

PDF/UA Mit den Optionen *path* oder *image* übergebene Vektorgrafiken und Rasterbilder müssen mit der Option *tag* als *Artifact* oder *Figure* ausgezeichnet werden.

Gültigkeit beliebig außer *object*

Tabelle 5.15 Formatierungsoptionen für PDF_add_table_cell( )

Option	Beschreibung
avoidwordsplitting	*(Boolean; nur bei Textflow-Zellen relevant) Bei* `true` *überprüft der Tabellenformatierer, ob im Textflow Wörter zwangsweise getrennt werden müssten, um den Text in die Tabellenzelle einzupassen. Ist dies der Fall, so wird die Zellenbreite erhöht, um den Text gegebenenfalls ohne zwangsweise Trennung platzieren zu können. Standardwert:* `true`
checkwordsplitting	*Veraltet, verwenden Sie stattdessen* `avoidwordsplitting`
colscalegroup[1]	*(String) Name einer Spaltengruppe, zu der die Spalte hinzugefügt wird. Wenn eine Spalte in der Gruppe vergrößert werden muss, um langen Text vollständig aufzunehmen, werden alle Spalten in der Gruppe einheitlich mitskaliert. Erstreckt sich eine Zelle über mehrere Spalten, so bilden die betroffenen Spalten automatisch eine Skalierungsgruppe.*
colspan	*(Integer) Anzahl der Spalten, über die sich eine Zelle erstreckt. Standardwert: 1*
colwidth[1]	*(Float oder Prozentwert) Breite der im Parameter* `column` *übergebenen Spalte in Benutzerkoordinaten*[2] *oder als prozentualer Anteil der Breite der ersten Fitbox der Tabelle (siehe PDF_fit_table( )). Alle Spaltenbreiten einer Tabelle dürfen entweder nur mit Benutzerkoordinaten oder nur mit Prozentwerten definiert werden, aber nicht mit beiden gemischt. Wenn sich eine Spalte über mehrere Textzellen erstreckt, wird die Spaltenbreite automatisch angepasst. Rasterbilder, Grafiken und PDF-Seiten in Tabellenzellen haben keinen Einfluss auf die Spaltenbreiten. Standardwert: siehe Option* `colwidthdefault` *von PDF_fit_table( )*
continuetextflow	*(Boolean; nur bei Textflows relevant) Bei* `true` *können die in der Option* `textflow` *angegebenen Textflow-Inhalte in einer anderen Zelle fortgesetzt werden, sofern diese mit demselben Textflow-Handle und* `continuetextflow=true` *gefüllt wird. Die einzelnen Textflow-Bestandteile werden in der Reihenfolge der hinzugefügten Zellen platziert. PDFlib passt die Größe der Zelle nicht an den gesamten Textflow an und die Option* `avoidwordsplitting` *wird ignoriert. Deshalb sollte eine angemessene Zellengröße definiert werden.* *Bei* `false` *wird der Textflow erneut gestartet. Standardwert:* `false`
margin marginleft marginbottom marginright margintop	*(Float oder Prozentwert) Linke/untere/rechte/obere Zellenränder in Benutzerkoordinaten (müssen größer oder gleich 0 sein) oder als Prozentwert der Zellenbreite bzw. Zellenhöhe (muss kleiner 100% sein). Die Ränder definieren die innere Zellenbox, die als Fitbox des Zelleninhalts dient. Standardwert für* `margin: 0`*; Standardwert für alle anderen:* `margin`
minrowheight[1]	*(Float oder Prozentwert) Für den Fall, dass eine Tabellenzeile nicht mehr vollständig in einer Tabelleninstanz platziert werden kann, lässt sich mit dieser Option festlegen, ob die Zeile geteilt werden darf und wie klein die Teilstücke werden dürfen. Die minimale Höhe eines Teilstücks kann in Benutzerkoordinaten oder als Prozentwert der Zeilenhöhe angegeben werden. Standardwert: 100%, d.h. keine Teilung der Zeile*

Tabelle 5.15 Formatierungsoptionen für PDF_add_table_cell( )

Option	Beschreibung
repeatcontent	*(Boolean) Bestimmt, ob der Inhalt einer Tabellenzelle wiederholt wird, wenn eine Zelle oder Zeile zwischen verschiedenen Tabelleninstanzen aufgeteilt wird. Standardwert:* `true` *Aufteilung einer Zelle: Wenn die letzten Zeilen einer Zelle nicht in die Fitbox passen, wird die Zelle gespalten. Bei* `repeatcontent=true` *wird der Inhalt der Zelle in der nächsten Tabelleninstanz wiederholt, außer bei Textflows (die nicht wiederholt werden). Anderenfalls wird der Inhalt nicht wiederholt.* *Aufteilung einer Zeile: Wenn die letzte Body-Zeile nicht in die Fitbox passt, wird sie in der Regel komplett, ohne Teilung in der nächsten Tabelleninstanz platziert. Durch Angabe eines niedrigeren Werts bei* `minrowheight` *kann die letzte Body-Zeile so geteilt werden, dass nur der angegebene Prozentsatz des Inhalts in der ersten Instanz und der Rest der Zeile in der nächsten Instanz platziert wird. Bei* `repeatcontent=true` *wird der Inhalt der Zelle in der nächsten Tabelleninstanz wiederholt, außer bei Textflows (die nicht wiederholt werden). Anderenfalls wird der Inhalt nicht wiederholt.*
return[1]	*(String) PDF_fit_table( ) stoppt nach der Platzierung der festgelegten Tabellenzeile und gibt den angegebenen String zurück. Der String darf nicht mit einem Unterstrich '_' beginnen. Gehört die festgelegte Zeile zu einer Zeilenverbindungsgruppe, so muss es die letzte Zeile in der Gruppe sein; anderenfalls tritt ein Fehler auf.*
rowheight[1]	*(Float oder Prozentwert) Höhe der im Parameter* `row` *übergebenen Tabellenzeile in Benutzerkoordinaten*[2] *oder als prozentualer Anteil an der Höhe der ersten Fitbox der Tabelle (siehe PDF_fit_table( )). Bei Prozentangaben ist diese Option erforderlich, und alle Zeilenhöhen müssen als Prozentwerte festgelegt werden. Bei Textflow-Zellen wird die hier definierte Zeilenhöhe bei Bedarf an den Textflow angepasst.*
rowscale-group[1]	*(String) Name einer Zeilenskalierungsgruppe, zu der die Tabellenzeile hinzugefügt wird. Wenn eine Zeile in der Gruppe vergrößert werden muss, um langen Text vollständig aufzunehmen, werden alle Zeilen in der Gruppe einheitlich mitskaliert. Erstreckt sich eine Zelle über mehrere Zeilen, bilden die betroffenen Zeilen automatisch eine Skalierungsgruppe.*
rowjoin-group[1]	*(String) Name einer Zeilenverbindungsgruppe, zu der die Tabellenzeile hinzugefügt wird. Alle Zeilen in einer Gruppe werden in derselben Tabelleninstanz ausgegeben. Die Zeilen in einer Gruppe müssen fortlaufend durchnummeriert sein. Erstreckt sich eine Zelle über mehrere Tabellenzeilen, so bilden die betroffenen Zeilen nicht automatisch eine Zeilenverbindungsgruppe.*
rowspan	*(Integer) Anzahl der Tabellenzeilen, über die sich die Zelle erstreckt. Standardwert: 1*

1. Die letzte Einstellung dieser Option hat Vorrang; frühere Einstellungen für dieselbe Tabellenzeile oder -spalte werden ignoriert.
2. Genauer gesagt das Koordinatensystem, das wirksam ist, wenn PDF_fit_table( ) zur Platzierung der ersten Tabelleninstanz aufgerufen wird.

Tabelle 5.16 Optionen für statischen Zelleninhalt in PDF_add_table_cell( ) sowie Unteroptionen für die Option `caption` *von PDF_fit_table( )*

Option	Beschreibung
fitgraphics	*(Optionsliste; nur für Grafiken relevant) Optionsliste für PDF_fit_graphics( ). Anhand dieser Optionsliste wird mit der Option* `graphics` *übergebene Grafik in der Zelle platziert. Die linke untere Ecke der Fitbox wird als Referenzpunkt verwendet.* *Standardwert:* `fitmethod=meet position=center`*. Diese Optionsliste wird den benutzerspezifischen Optionen vorangestellt.*[1]
fitimage	*(Optionsliste; nur für Bilder und Templates relevant) Optionsliste für PDF_fit_image( ). Anhand dieser Optionsliste wird das mit der Option* `image` *übergebene Bild oder Template in der Zelle platziert. Die linke untere Ecke der Fitbox wird als Referenzpunkt verwendet.* *Standardwert:* `fitmethod=meet position=center`*. Diese Optionsliste wird den benutzerspezifischen Optionen vorangestellt.*[1]

Tabelle 5.16 Optionen für statischen Zelleninhalt in PDF_add_table_cell( ) sowie Unteroptionen für die Option `caption` *von PDF_fit_table( )*

Option	Beschreibung
fitpath	*(Optionsliste; nur für Pfadobjekte relevant) Optionsliste für PDF_draw_path( ). Anhand dieser Optionsliste wird das mit der Option* `path` *übergebene Pfadobjekt innerhalb seiner Boundingbox in der Zelle platziert. Die linke untere Ecke der Fitbox wird als Referenzpunkt verwendet.* *Standardwert:* `fitmethod=meet position=center`*. Diese Optionsliste wird den benutzerspezifischen Optionen vorangestellt.*[1]
fitpdipage	*(Optionsliste; nur für PDI-Seiten relevant; nur wenn PDI verfügbar ist) Optionsliste für PDF_fit_pdi_page( ). Anhand dieser Optionsliste wird die mit der Option* `pdipage` *übergebene Seite in der Zelle platziert. Die linke untere Ecke der Fitbox wird als Referenzpunkt verwendet.* *Standardwert:* `fitmethod=meet position=center`*. Diese Optionsliste wird den benutzerspezifischen Optionen vorangestellt.*[1]
fittextflow	*(Optionsliste; nur für Textflows relevant) Optionsliste für PDF_fit_textflow( ). Anhand dieser Optionsliste wird der mit der Option* `textflow` *übergebene Textflow in der Zelle platziert. Die innere Zellenbox wird als Fitbox verwendet.* *Standardwert:* `verticalalign=center lastlinedist=descender`*. Diese Optionsliste wird der benutzerspezifischen Optionsliste vorangestellt.*
fittextline	*(Optionsliste; nur für Textzeilen relevant) Optionsliste für PDF_fit_textline( ). Anhand dieser Optionsliste wird der mit dem Parameter* `text` *übergebene Text in der Zelle platziert. Die linke untere Ecke der Fitbox wird als Referenzpunkt verwendet. Nicht definierte Optionen werden durch entsprechende Standardwerte ersetzt; der aktuelle Textzustand wird nicht berücksichtigt.* *Standardwert:* `fitmethod=meet position=center`*. Diese Optionsliste wird den benutzerspezifischen Optionen vorangestellt.*[1]
graphics	*(Grafik-Handle) Die zum Handle gehörende Grafik wird in der Fitbox platziert.*
image	*(Image-Handle) Das zum Handle gehörende Bild oder Template wird in der Fitbox platziert.*
matchbox	*(Optionsliste) Optionsliste mit detaillierten Angaben zur Matchbox gemäß Tabelle 6.4*
path	*(Pfad-Handle) Das Pfadobjekt innerhalb seiner Boundingbox wird gemäß der Option* `fitpath` *in der Fitbox platziert.*
pdipage	*(Seiten-Handle) Die zum Handle gehörende importierte PDI-Seite wird in der Fitbox platziert.*
text	*(Content-String) Text, der mit PDF_fit_textline( ) gemäß der Option* `fittextline` *platziert werden soll. Bei PDF_add_table_cell( ) kann der Wert dieser Option alternativ über den Funktionsparameter* `text` *angegeben werden.*
textflow	*(Textflow-Handle) Der zum Handle gehörende Textflow wird in der Fitbox platziert. Die Option* `continuetextflow` *steuert das Verhalten eines Textflow-Handles, das in mehreren Zellen vorkommt. Das Textflow-Handle darf nicht außerhalb der Tabelle verwendet werden.*

1. Die Größe der Box wird automatisch berechnet; jede Option `boxsize` *in der übergebenen Optionsliste wird ignoriert.*

C++ Java C# ***String fit_table(int table, double llx, double lly, double urx, double ury, String optlist)***

Perl PHP ***string fit_table(int table, float llx, float lly, float urx, float ury, string optlist)***

C ***const char *PDF_fit_table(PDF *p,***
int table, double llx, double lly, double urx, double ury, const char *optlist)

Platziert eine Tabelle vollständig oder teilweise auf der Seite.

table Gültiges Tabellen-Handle, das von einem Aufruf von *PDF_add_table_cell( )* zurückgegeben wurde.

Tabelle 5.17 Optionen für interaktiven Zelleninhalt für PDF_add_table_cell( ) sowie Unteroptionen für die Option `caption` *(nur im Gültigkeitsbereich* page*)*

Optionen	Beschreibung
annotation-type	*(String) Bestimmt den Typ der Anmerkung, der gemäß Tabelle 12.3 in die Tabellenzelle eingefügt wird.*
fieldname	*(Hypertext-String) Name des Formularfelds für* `fieldtype`
fieldtype	*(String) Bestimmt den Typ eines Formularfelds, das gemäß Tabelle 12.5 in die Tabellenzelle eingefügt wird. Formularfeld-Gruppen sollten außerhalb von Tabellen definiert werden.*
fitannotation	*(Optionsliste) Optionen für Anmerkungen für* `annotationtype` *gemäß Tabelle 12.4*
fitfield	*(Optionsliste) Optionen für Formularfelder für* `fieldtype` *gemäß Tabelle 12.8*

llx, lly, urx, ury Koordinaten der linken unteren und rechten oberen Ecke des Zielrechtecks (Fitbox) in Benutzerkoordinaten, in das die Tabelleninstanz platziert wird. Die Ecken können auch in umgekehrter Reihenfolge angegeben werden.

optlist Optionsliste mit detaillierten Angaben zur Platzierung gemäß Tabelle 5.18. Folgende Optionen können verwendet werden:

- Allgemeine Option: *errorpolicy* (siehe Tabelle 2.1)
- Optionen zur Objekteinpassung gemäß Tabelle 6.1: *fitmethod, position, showborder*
- Allgemeine Tabellenoptionen:
  blind, colwidthdefault, horshrinklimit, rewind, rowheightdefault, vertshrinklimit
- Tabelleninhalt: *header, footer*
- Tabellendekoration: *fill, firstdraw, gstate, stroke*
- Darstellungshilfe bei Entwicklung und Fehlersuche: *debugshow, showcells, showgrid*
- Option zum vereinfachten Tagging von Strukturelementen gemäß Tabelle 14.5 (nur im Gültigkeitsbereich *page* erlaubt): *tag*. Mit dieser Option lässt sich das automatische Anbringen von Tabellen-Tags aktivieren (für weitere Informationen siehe PDFlib-Tutorial).

Rückgabe String, der die Ursache des Funktionsendes anzeigt:

- *_stop:* Alle Tabellenzeilen wurden verarbeitet.
- *_boxfull:* Es sind noch nicht alle Zeilen platziert, aber es ist nicht mehr genug Platz in der Fitbox der Tabelle; zur Verarbeitung der übrigen Zeilen ist ein weiterer Aufruf von *PDF_fit_table( )* erforderlich.
- *_error:* Ein Fehler ist aufgetreten; mit *PDF_get_errmsg( )* erhalten Sie detaillierte Informationen zum Problem.
- Jeder andere String: String, der in der Option *return* in *PDF_add_table_cell( )* übergeben wurde.

Das Verhalten im Fehlerfall kann mit der Option *errorpolicy* geändert werden.

Details Platziert die Tabelle auf der Seite. Die Tabellenzellen müssen mit früheren Aufrufen von *PDF_add_table_cell( )* gefüllt worden sein. Passt die Tabelle nicht vollständig in die Fitbox, so wird die erste Tabelleninstanz platziert; abhängig vom Rückgabewert können mit nachfolgenden Aufrufen dieser Funktion weitere Instanzen platziert werden. Die Inhalte einer Tabellenzelle werden in folgender Reihenfolge platziert:

- Flächenfüllung: Die mit der Option *fill* festgelegten Bereiche werden in folgender Reihenfolge gefüllt: *table, colother, colodd, coleven, col#, collast, rowother, rowodd, roweven, row#, rowlast, header, footer.*

- Matchbox-Schattierung: Einzelne Zellen, die über eine Matchbox definiert wurden.
- Inhalte: Die übergebenen Zelleninhalte werden in folgender Reihenfolge platziert: Bild, Grafik, importierte PDF-Seite, Pfadobjekte, Textflow, Textzeile, Anmerkungen, Formularfelder.
- Matchbox: Einzelne Bereiche, die über eine Matchbox definiert wurden.
- Lineatur: Die mit der Option *stroke* definierten Linien werden entsprechend der Unteroptionen *linecap* und *linejoin* der Option *stroke* in folgender Reihenfolge gezeichnet: *other, horother, hor#, horlast, vertother, vert#, vertlast, frame* (die Reihenfolge der horizontalen und vertikalen Linien lässt sich mit der Option *firstdraw* ändern). Zellen, die sich über mehrere Tabellenzeilen und -spalten erstrecken, werden nicht von Linien zerschnitten. Auch werden keine Linien um Zellen gezeichnet, die eine Matchbox enthalten, für die selbst Randlinien gezeichnet werden (außer die Matchbox verwendet die innere Zellenbox). Die Tabellenrandlinien *verto, horo, vertN* und *horN* werden unterdrückt, wenn *frame* angegeben ist.
- Benannte Matchboxen: Diese können mit anderen Elementen wie Anmerkungen, Formularfeldern, Bildern, Grafiken usw. außerhalb der Tabellenfunktionen gefüllt werden.

Mit der Option *tag* lässt sich das automatische Anbringen von Tabellen-Tags aktivieren (für weitere Informationen siehe PDFlib-Tutorial).

Gültigkeit In der Regel *page, pattern, template, glyph;* enthält die Tabelle jedoch Formularfelder oder Anmerkungen, gilt der jeweilige Gültigkeitsbereich dieser Tabelleninhalte. Eine Tabelle mit Formularfeldern oder Anmerkungen kann zum Beispiel nicht auf einem Template platziert werden.

PDF/UA Wenn das automatische Anbringen von Tabellen-Tags aktiviert ist, wir d die Tabellendekoration (Rahmen und Schattierung) automatisch als Artefakt ausgezeichnet.

Tabelle 5.18 Optionen für PDF_fit_table( )

Option	Beschreibung
blind	*(Boolean) Bei* `true` *werden alle Berechnungen durchgeführt, es wird aber keine Ausgabe generiert. Die Formatierungsergebnisse können mit PDF_info_table( ) abgefragt werden. Standardwert:* `false`
caption	*(Optionsliste) Erzeugt eine Fitbox für eine Beschriftung relativ zur berechneten Fitbox und füllt sie mit unterschiedlichen Inhaltstypen. Die folgenden Optionen können übergeben werden (Standardwert: keine Beschriftung):* **fitbox** *(Liste aus vier Floats oder Prozentwerten mit absoluten oder relativen Koordinaten; erforderlich) Koordinaten von zwei diagonalen Ecken einer Box in Benutzerkoordinaten. Ein Prozentwert oder ein relativer Wert gibt den Abstand zu der entsprechenden Ecke der Tabelleninstanz an* `{llx lly urx ury}`*. Prozentwerte, die* `llx` *oder* `urx` *entsprechen, beziehen sich auf die Breite der Tabelleninstanz; Prozentwerte, die* `lly` *oder* `ury` *entsprechen, beziehen sich auf die Höhe der Tabelleninstanz. Die Fitbox wird nicht automatisch an die Größe des Inhalts angepasst. Die angegebene Matchbox beschreibt die Fitbox; dies kann zum Zeichnen der Fitbox für die Beschriftung verwendet werden oder um die Matchbox mit PDF_info_matchbox( ) abzurufen.* *Beispiele für die Verwendung der Option* `fitbox`*:* *Fitbox am Anfang der Tabelleninstanz mit einer Höhe von 20:* `fitbox={0r 100% 0r 20r}` *Fitbox rechts der Tabelleninstanz mit einer Breite von 20 und einem Abstand von 20% zum Ende der Tabelleninstanz:* `fitbox={100% 20% 20r 0r}` *Zusätzlich werden folgende Optionen unterstützt:* ▸ *Optionen für statischen Zelleninhalt gemäß Tabelle 5.16:* `fitgraphics, fitimage, fitpath, fitpdipage, fittextflow, fittextline, graphics, image, matchbox, path, pdipage, textflow` ▸ *Optionen für interaktiven Zelleninhalt gemäß Tabelle 5.17 (nur im Gültigkeitsbereich* `page`*):* `annotationtype, fieldname, fieldtype, fitannotation, fitfield` ▸ *Option zum vereinfachten Tagging von Strukturelementen gemäß Tabelle 14.5:* `tag`*. Damit kann ein übergeordnetes Element der Beschriftungsinhalte eingefügt werden oder ein Gruppierungselement als Container für mehrere Elemente, die zusammen die Beschriftung bilden.*
colwidth-default	*(Float oder Schlüsselwort; nur im ersten Aufruf von PDF_fit_table( ) für eine bestimmte Tabelle relevant) Standardbreite für Spalten ohne Textlines oder Textflow und für die die Option* `colwidth` *von PDF_add_table_cell( ) nicht angegeben wird. Die Standardbreite kann als absoluter Wert oder als Schlüsselwort angegeben werden. Der Wert 0 (null) ist äquivalent zum Schlüsselwort* `distribute`*. Die folgenden Schlüsselwörter werden unterstützt (Standardwert:* `auto`*):* **auto** *Spalten mit unbestimmter Breite, die nur aus Textline-Zellen bestehen, erhalten die Breite des Textes. Die verbleibende Breite der Fitbox wird unter allen Zeilen mit Textflow oder anderen Inhalten aufgeteilt. Die Tabelle umfasst die gesamte Breite der Fitbox.* **distribute** *Die Breite der Fitbox wird gleichmäßig auf alle Spalten mit unbestimmter Breite und Spalten ohne Textlines verteilt. Die Tabelle umfasst die gesamte Breite der Fitbox, sofern sie nur Textlines enthält.* **minimum** *Spalten mit unbestimmter Breite, die nur aus Textline-Zellen bestehen, haben die Breite des Textes, das heißt die kleinstmögliche Breite, um den Text vollständig aufzunehmen.* *Um Spalten mit minimaler Breite zu erstellen, können Sie einen niedrigen Wert (z.B. 1) angeben. Die Breite aller Spalten, die Textline- oder Textflow-Zellen enthalten, wird automatisch angepasst (siehe PDFlib-Tutorial).*
debugshow	*(Boolean) Bei* `true` *werden alle Fehler, die sich auf zu hohe oder breite Tabellen oder zu kleine Tabellenzellen beziehen, unterdrückt und stattdessen protokolliert. Die Tabelleninstanz wird zur Unterstützung bei der Fehlersuche generiert, die Tabelle ist jedoch unbrauchbar. Standardwert:* `false`

Tabelle 5.18 Optionen für PDF_fit_table( )

Option	Beschreibung
fill	*(Liste aus Optionslisten) Mit dieser Option können Sie Tabellenzeilen oder -spalten mit Farbe füllen (mit der Option* `matchbox` *werden einzelne Zellen mit Farbe versehen, siehe Abschnitt 6.2, »Matchboxen«, Seite 141):* **area** *(Schlüsselwort) Zu füllende Tabellenbereiche:* **col#** *Spalte Nummer # in der Tabelle* **collast** *letzte Spalte* **coleven** *alle Spalten mit gerader Nummer (entsprechend* `col` *in PDF_add_table_cell( ))* **colodd** *alle Spalten mit ungerader Nummer* **colother** *alle hier nicht festgelegten Spalten* **row#** *Zeile Nummer # in der Tabelle* **rowlast** *letzte Body-Zeile in der Tabelleninstanz* **roweven** *alle Zeilen mit gerader Nummer (entsprechend* `row` *in PDF_add_table_cell( ))* **rowodd** *alle Zeilen mit ungerader Nummer* **header** *alle Zeilen, die zu den Kopfzeilen gehören* **footer** *alle Zeilen, die zu den Fußzeilen gehören* **rowother** *alle hier nicht festgelegten Body-Zeilen* **table** *vollständiger Tabellenbereich (d.h. alle Zeilen in der Tabelle)* *Die folgenden Optionen zur Grafikdarstellung können gemäß Tabelle 7.1 verwendet werden:* ***fillcolor, shading*** *Beispiele:* *Alle Tabellenzeilen mit roter Farbe füllen:* `fill = { {area=table fillcolor=red} }` *Ungerade Zeilen mit grüner und gerade Zeilen mit roter Farbe füllen:* `fill = { {area=rowodd fillcolor=green} {area=roweven fillcolor=red} }` *Mit* `fillcolor=none` *kann die Schattierung eines Tabellenbereichs unterdrückt werden.*
firstdraw	*(Schlüsselwort) Legt die Reihenfolge fest, in der horizontale und vertikale Linien erzeugt werden (Standardwert:* `vertlines`*):* **horlines** *Horizontale Linien werden zuerst erzeugt.* **vertlines** *Vertikale Linien werden zuerst erzeugt.*
footer	*(Integer) Anzahl der abschließenden Zeilen in der Tabellendefinition, die als Fußzeilen am Ende jeder Tabelleninstanz wiederholt werden. Standardwert: 0 (keine Fußzeilen)*
gstate	*(Gstate-Handle) Handle für einen mit PDF_create_gstate( ) erzeugten Grafikzustand. Der Grafikzustand wirkt sich auf die mit dieser Funktion erzeugte gesamte Tabellendekoration aus. Zelleninhalte sind davon nicht betroffen. Standardwert: kein Grafikzustand (das heißt, die aktuellen Einstellungen werden verwendet).*
header	*(Integer) Anzahl der ersten Zeilen in der Tabellendefinition, die als Kopfzeilen am Anfang jeder Tabelleninstanz wiederholt werden. Standardwert: 0 (keine Kopfzeilen)*
horshrinklimit	*(Float oder Prozentwert) Untere Grenze für den horizontalen Stauchungsfaktor, der zum Stauchen der Tabelle verwendet wird, damit diese in die Fitbox passt (sofern ein Prozentwert angegeben wird) oder die absolute Differenz zwischen der Tabellenbreite und der Breite der Fitbox (wenn ein Float angegeben wird). Standardwert: 50%*
rewind	*(Integer: -1, 0 oder 1) Die Tabelle wird in den Zustand zurückgesetzt, den sie vor einem anderen Aufruf von PDF_fit_table( ) hatte. Gegenwärtig werden folgende Werte unterstützt (Standardwert: 0):* **1** *In den Zustand vor dem ersten Aufruf von PDF_fit_table( ) zurücksetzen.* **0** *Tabelle nicht zurücksetzen.* **-1** *In den Zustand vor dem letzten (d.h. dem aktuellen Aufruf vorangehenden) Aufruf von PDF_fit_table( ) zurücksetzen.*

Tabelle 5.18 Optionen für PDF_fit_table( )

Option	Beschreibung
rowheight-default	*(Float oder Schlüsselwort; nur im ersten Aufruf von PDF_fit_table( ) für eine bestimmte Tabelle erforderlich) Standardhöhe von Zeilen, für die die Option* `rowheight` *von PDF_add_table_cell( ) nicht angegeben wurde. Die Standardhöhe kann als absoluter Wert oder als Schlüsselwort angegeben werden. Wird ein Float-Wert angegeben, wird dieser als Standardzeilenhöhe verwendet, sofern er nicht kleiner als die Höhe der Textbox ist. Der Wert 0 (null) ist äquivalent zum Schlüsselwort* `distribute`*. Die folgenden Schlüsselwörter werden unterstützt (Standardwert:* `auto`*):* **auto** *Zeilen, die nur aus Textline-Zellen bestehen, erhalten die doppelte Höhe der Textbox. Die verbleibende Höhe des Fitbox wird unter allen Zeilen mit Textflow oder andere Zellen aufgeteilt. Die Tabelle umfasst die gesamte Höhe der Fitbox.* **distribute** *Die Höhe des Fitbox wird gleichmäßig auf alle Zeilen mit unbestimmter Höhe verteilt. Die Tabelle umfasst die gesamte Höhe der Fitbox.* **minimum** *Zeilen mit unbestimmter Höhe, die nur aus Textline-Zellen bestehen, erhalten die Höhe der Textbox, das heißt, die kleinstmögliche Höhe, um den Text vollständig aufzunehmen. Mit den Optionen* `boxsize` *oder* `margin` *lässt sich die Höhe von Textline-Zellen vergrößern.* *Um Spalten mit minimaler Höhe zu erstellen, können Sie einen niedrigen Wert (z.B. 1) angeben. Die Höhe aller Zeilen, die Textline- oder Textflow-Zellen enthalten, wird automatisch angepasst (siehe PDFlib-Tutorial).*
showcells	*(Boolean) Bei* `true` *wird der Rand der inneren Zellenboxen anhand des aktuellen Grafikzustands gezeichnet. Im Gültigkeitsbereich* `page` *und wenn PDF/A nicht aktiviert ist, wird jede Zelle noch zusätzlich mit einer Anmerkung zu den Zelleninhalten dargestellt. Dies kann zur Analyse bei Problemen mit Tabellen nützlich sein. Standardwert:* `false`
showgrid	*(Boolean) Bei* `true` *werden alle Spalten- und Zeilenränder gezeichnet. Standardwert:* `false`
stroke	*(Liste aus Optionslisten) Mit dieser Option können Linien an den Zellenränder gezeichnet werden:* **line** *(Schlüsselwort) Zu zeichnende Tabellenlinien:* **vert#** *vertikale Linie am rechten Rand der Spalte Nummer # in der Tabelle;* `vert0` *entspricht dem linken Tabellenrand* **vertfirst** *erste vertikale Linie (entspricht* `vert0`*)* **vertlast** *letzte vertikale Linie* **vertother** *alle hier nicht festgelegten vertikalen Linien* **hor#** *horizontale Linie am Ende von Zeile Nummer # in der Tabelle;* `row0` *entspricht dem oberen Rand* **horfirst** *erste horizontale Linie in der Tabelleninstanz* **horlast** *letzte horizontale Line in der Tabelleninstanz* **horother** *alle hier nicht festgelegten horizontalen Linien* **frame** *äußerer Tabellenrand* **other** *alle nicht festgelegten Linien* *Die folgenden Optionen zur Grafikdarstellung können gemäß Tabelle 7.1 verwendet werden:* ***dasharray***, ***dashphase***, ***linecap***, ***linejoin***, ***linewidth***, ***strokecolor*** *Beispiele:* *Alle Linien mit schwarzer Farbe und Strichstärke 1 zeichnen:* `stroke = {line=other}` *Äußeren Tabellenrand mit Strichstärke 0.5 zeichnen:* `stroke = { {line=frame linewidth=0.5} }` *Äußeren Tabellenrand mit Strichstärke 0.5 und alle anderen Linien mit Linienstärke 0.1 zeichnen:* `stroke = { {line=frame linewidth=0.5} {line=other linewidth=0.1} }` *Mit* `strokecolor=none` *kann das Zeichnen von Linien für einen Tabellenbereich unterdrückt werden.*
vertshrink-limit	*(Float oder Prozentwert) Untere Grenze für den vertikalen Stauchungsfaktor, der zum Stauchen der Tabelle verwendet wird, damit diese in die Fitbox passt (sofern ein Prozentwert angegeben wird) oder die absolute Differenz zwischen der Tabellenhöhe und der Höhe der Fitbox (wenn ein Float angegeben wird). Standardwert: 90%*

C++ Java C# ***double info_table(int table, String keyword)***

Perl PHP ***float info_table(int table, string keyword)***

C ***double PDF_info_table(PDF *p, int table, const char *keyword)***

Fragt tabellenspezifische Informationen zur zuletzt platzierten Tabelleninstanz ab.

table Gültiges Tabellen-Handle, das von einem Aufruf von *PDF_add_table_cell( )* zurückgegeben wurde. Das Tabellen-Handle muss bereits in einem Aufruf von *PDF_fit_table( )* verwendet worden sein, da die zurückgegebenen Werte nur nach der Platzierung einer Tabelleninstanz auf der Seite aussagekräftig sind.

keyword Schlüsselwort zur Anforderung der gewünschten Information:

- Schlüsselwörter zur Abfrage der Ergebnisse der Objekteinpassung gemäß Tabelle 6.3: *boundingbox, fitscalex, fitscaley, height, objectheight, objectwidth, width, x1, y1, x2, y2, x3, y3, x4, y4*
- Zusätzliche Schlüsselwörter gemäß Tabelle 5.19: *firstbodyrow, horboxgap, horshrinking, lastbodyrow, returnreason, rowcount, rowsplit, tableheight, tablewidth, vertboxgap, vertshrinking, xvertline#, yhorline#*

Rückgabe Wert eines durch *keyword* bezeichneten Tabellenparameters. Diese Funktion gibt auch im Blind-Modus korrekte geometrische Informationen zurück. Erzeugt das entsprechende Schlüsselwort Text, wird ein String-Index zurückgegeben und der entsprechende String muss mit *PDF_get_string( )* abgerufen werden.

Gültigkeit beliebig außer *object*

Tabelle 5.19 Schlüsselwörter für PDF_info_table( )

Schlüsselwort	Beschreibung
firstbodyrow	*Nummer der ersten Body-Zeile in der zuletzt platzierten Tabelleninstanz*
horboxgap	*Differenz zwischen der Breite der Tabelleninstanz und der Breite der Fitbox. Musste die Tabelle gestaucht werden, bezeichnet der Wert die Abweichung von der Breite der Fitbox (d.h. einen negativen Wert).*
horshrinking	*Horizontaler Stauchungsfaktor als prozentualer Anteil der berechneten Tabellenbreite. Musste die Tabelle horizontal gestaucht werden, bezeichnet der Wert den Prozentsatz, um den die Tabelle horizontal gestaucht wurde, und sonst 100.*
lastbodyrow	*Nummer der letzten Body-Zeile in der zuletzt platzierten Tabelleninstanz*
returnreason	*String-Index der Rückkehrursache*
rowcount	*Anzahl der Zeilen in der zuletzt platzierten Tabelleninstanz (einschließlich der Kopf- und Fußzeilen)*
rowsplit	*1, wenn die letzte Zeile aufgeteilt wurde, sonst 0*
tableheight tablewidth	*Breite und Höhe der gesamten Tabelle*
vertboxgap	*Differenz zwischen der Höhe der zuletzt generierten Tabelleninstanz und der Höhe der Fitbox. Musste die Tabelle gestaucht werden, bezeichnet der Wert die Abweichung von der Höhe der Fitbox (d.h. einen negativen Wert).*
vertshrinking	*Vertikaler Stauchungsfaktor als prozentualer Anteil an der berechneten Tabellenhöhe. Musste die Tabelle gestaucht werden, bezeichnet der Wert den Prozentsatz, um den die Tabelle vertikal gestaucht wurde, und sonst 100.*

Tabelle 5.19 Schlüsselwörter für PDF_info_table( )

Schlüsselwort	Beschreibung
xvertline#	*x-Koordinate der vertikalen Linie Nummer #.* `xvertline0` *bezeichnet den linken Tabellenrand.*
yhorline#	*x-Koordinate der horizontalen Line Nummer #.* `yhorline0` *bezeichnet den oberen Tabellenrand.*

C++ Java C# **void delete_table(int table, String optlist)**

Perl PHP **delete_table(int table, string optlist)**

C **void PDF_delete_table(PDF *p, int table, const char *optlist)**

Löscht eine Tabelle und alle damit verbundenen Datenstrukturen.

table Gültiges Tabellen-Handle, das von einem Aufruf von *PDF_add_table_cell( )* zurückgegeben wurde.

optlist Optionsliste mit Löschoptionen gemäß Tabelle 5.20.

Details Wird eine Tabelle nicht mit dieser Funktion gelöscht, so wird sie automatisch am Ende des umgebenden Gültigkeitsbereichs *document* gelöscht.

Gültigkeit beliebig

Tabelle 5.20 Option für PDF_delete_table( )

Option	Beschreibung
keephandles	*(Boolean) Bei* `false` *werden alle Handles, die an die Optionen* `textflow`, `image`, `graphics` *und* `pdipage` *von PDF_add_table_cell( ) übergeben wurden, automatisch gelöscht. Standardwert:* `false`

6 Objekteinpassung und Matchboxen

6.1 Objekteinpassung

Der Algorithmus von PDFlib für die Objekteinpassung platziert ein rechteckiges Grafikobjekt relativ zu einem Punkt, einer horizontalen oder vertikalen Linie oder einem Rechteck. Der Algorithmus für die Objekteinpassung ist in verschiedenen Funktionen implementiert:

- *PDF_fit_textline( )*, *PDF_info_textline( )*
- *PDF_fit_image( )*, *PDF_info_image( )*
- *PDF_fit_graphics( )*, *PDF_info_graphics( )*
- *PDF_fit_pdi_page( )*, *PDF_info_pdi_page( )*
- *PDF_draw_path( )*, *PDF_info_path( )*
- *PDF_add_table_cell( )* (über Optionslisten für die Optionen *fitgraphics, fitimage, fitpdipage, fitpath, fittextline)*
- *PDF_fit_table( )*
- *PDF_fill_*block( )*

Hinweis *Die Optionen zur Einpassung von Textflow weichen leicht von den in diesem Kapitel dargestellten Optionen ab und werden in Abschnitt 5.2, »Mehrzeilige Textausgabe mit Textflows«, Seite 104 beschrieben.*

In Tabelle 6.1 werden Optionen zur Objekteinpassung aufgelistet, die an die Funktionen zur Objekteinpassung übergeben werden können. Nicht alle Optionen stehen für alle Funktionen zur Verfügung, und das Verhalten einiger Optionen kann je nach Funktion leicht variieren; für weitere Informationen siehe Tabelle 6.1. Für die Objekteinpassung können folgende Optionen verwendet werden:

alignchar, boxsize, dpi, fitmethod, margin, matchbox, minfontsize, orientate, position, refpoint, rotate, scale, stamp, showborder, shrinklimit

Objektbox. Der Algorithmus berechnet immer das kleinste umschließende Rechteck des platzierten Objekts. Dieses Rechteck wird *Objektbox* genannt. Es kann entsprechend dem Typ des Objekts geändert werden:

- Textlines *(PDF_fit/info_textline( )*, Blöcke vom Typ *Textline*, also einzeiliger Text, Tabellenzellen): Die Breite ist die Breite des Text-Strings (in horizontalem Ausgabemodus) oder die Breite der breitesten Glyphe (in vertikalem Ausgabemodus). Die Standardhöhe der Textbox ist die Versalhöhe *(capheight)* des gewählten Fonts. Dies kann mit der Unteroption *boxheight* der Option *matchbox* geändert werden. Nach der letzten Glyphe wird kein Zeichenabstand mehr angewendet.
- Bilder und Templates *(PDF_fit/info_image( )*, Blöcke vom Typ *Image*, Tabellenzellen): mit der Unteroption *clipping* der Option *matchbox* lassen sich bestimmte Teile des Objekts als Objektbox definieren. Für Bilder vom Typ TIFF und JPEG mit einem Beschneidungspfad wird das kleinste umschließende Rechteck mit Kanten parallel zu den Achsen des Koordinatensystems als Objektbox verwendet, sofern die Unteroption *innerbox* der Option *matchbox* gesetzt ist.

- Grafik *(PDF_fit/info_graphics( ))*: mit der Unteroption *clipping* der Option *matchbox* können bestimmte Teile des Objekts als Objektbox definiert werden. Die Objektbox wird durch die Breite und Höhe der SVG-Grafik oder durch *forcedwidth* und *forcedheight* festgelegt. Sind diese Werte gleich 0, gilt Folgendes: ist *fitmethod* verschieden von *nofit* oder ist die Fitbox nicht definiert, wird die Größe der Objektbox mit *fallbackwidth* und *fallbackheight* festgelegt. Ist *fitmethod=nofit* und ist die Fitbox definiert, wird die Größe der Objektbox durch die Fitbox bestimmt.
- Importierte PDF-Seiten *(PDF_fit/info_pdi_page( )*, PDF-Blöcke, Tabellenzellen): die Optionen in *PDF_open_pdi_page( )* werden berücksichtigt. Bei *cloneboxes=true* wird die sichtbare Box verwendet (also die CropBox, wenn vorhanden, und sonst die MediaBox). Mit der Unteroption *clipping* der Option *matchbox* können bestimmte Teile des Objekts als Objektbox definiert werden.
- Pfadobjekte *(PDF_draw/info_path( ),* Tabellenzellen): das kleinste den Pfad umschließende Rechteck mit Kanten parallel zu den Achsen des Koordinatensystems wird als Objektbox verwendet. Die Objektbox wird nur berechnet, wenn die Werte der Optionen *boxsize* und *position* verschieden von null sind. Die Optionen *linewidth* und *miterlimit* werden ignoriert.
- Tabelleninstanzen *(PDF_fit_table( ))*: das kleinste die Tabelleninstanz umschließende Rechteck mit Kanten parallel zu den Achsen des Koordinatensystems wird als Objektbox verwendet.

Referenzpunkt. Der *Referenzpunkt* wird als Anker zur Platzierung der Objektbox verwendet. Er wird folgendermaßen bestimmt:

- In *PDF_fit_*( )* und *PDF_draw_path( ):* die Funktionsparameter *x* und *y*;
- In *PDF_info_*( ):* der Punkt *(0, 0); PDF_info_path( )* unterstützt zusätzlich die Option *refpoint* zur Angabe des Referenzpunkts.
- *PDF_add_table_cell( ), PDF_fit_table( )* und *PDF_fill_*block( ):* die linke untere Ecke der Tabellenzelle, der Tabelleninstanz oder des PDFlib-Blocks; *PDF_fill_*block( )* unterstützt zusätzlich die Option *refpoint* zur Angabe des Referenzpunkts.

Fitbox und Referenz-Liniensegment. Das Rechteck, in dem die Objektbox platziert wird, wird *Fitbox* genannt. Ihr Referenzpunkt *(x, y)* liegt an ihrer linken unteren Ecke und ihre Größe wird durch die beiden Werte der Option *boxsize* bestimmt:

```
linke untere Ecke = (x, y)
rechte obere Ecke = (x + boxsize[0], y + boxsize[1])     (bei topdown=false)
rechte obere Ecke = (x + boxsize[0], y - boxsize[1])     (bei topdown=true)
```

Zusätzlich zu der oben angegebenen Definition kann die Fitbox folgendermaßen geändert werden:

- Textlines: die Fitbox kann mit der Option *margin* verkleinert werden;
- Tabellenzellen: die Fitbox wird durch die innere Zellenbox definiert, das heißt, die durch die Optionen *margin** geänderte Zellenbox*;*
- Tabelleninstanzen: die Fitbox wird durch die Parameter *llx/lly/urx/ury* definiert;
- PDFlib-Blöcke: die Fitbox wird standardmäßig durch die Blockeigenschaft *Rect* definiert, kann aber mit den Optionen *refpoint* und/oder *boxsize* geändert werden.

In den letzten drei obigen Fällen ist die Fitbox immer verfügbar; ansonsten ist sie nur verfügbar, wenn in der Option *boxsize* zwei von null verschiedene Werte übergeben wurden.

Bei *boxsize[0]=0* wird die Box zu einer vertikalen Linie. Der Algorithmus zur Objekteinpassung platziert die Objektbox relativ zu diesem Liniensegment. Bei *boxsize[1]=0* wird die Box auf ähnliche Art relativ zu dem daraus resultierenden horizontalen Liniensegment platziert. Das vertikale oder horizontale Liniensegment wird *Referenz-Liniensegment* genannt.

Platzieren der Objektbox. Die Objektbox kann auf verschiedene Arten platziert werden:

- Ist keine Fitbox verfügbar, wird das Objekt relativ zum Referenzpunkt platziert (nicht bei Tabellenzellen, Tabelleninstanzen und PDFlib-Blöcken): Die linke untere Ecke der Objektbox kommt auf dem Referenzpunkt zu liegen. Mit der Option *position* können andere Punkte in der Objektbox ausgewählt werden. Beispielsweise wird mit *position=center* der Mittelpunkt der Objektbox am Referenzpunkt platziert.
  Die Option *scale* wird bei Rasterbildern, Grafiken, Templates, Pfadobjekten und importierten PDF-Seiten ausgewertet; die Option *dpi* wird bei Rasterbildern ausgewertet. Die Option *fitmethod* wird in diesem Fall ignoriert.
  Pfadobjekte: bei *position={0 0}* wird die Boundingbox nicht berechnet, und der Ursprung des Pfadobjekts kommt auf dem Referenzpunkt zu liegen.
- Relativ zu einem Referenz-Liniensegment (nicht bei Tabellenzellen, Tabelleninstanzen und PDFlib-Blöcken): dies funktioniert ähnlich wie bei der Platzierung eines Objekts relativ zum Referenzpunkt wie oben beschrieben. Zusätzlich wird mit der Option *position* auch ein Punkt auf dem Liniensegment definiert, der als Referenzpunkt dient.
- Relativ zur Fitbox: Mit der Option *fitmethod* wird festgelegt, ob und auf welche Weise die Objektbox in die Fitbox eingepasst wird. Bei *fitmethod=nofit* wird das Ergebnis nicht in die Grenzen der Fitbox eingepasst. Mit anderen Werten von *fitmethod* können Details des Algorithmus für die Objekteinpassung gemäß Tabelle 6.2 festgelegt werden.
  In diesem Fall werden die Optionen *scale* und *dpi* ignoriert, die Optionen *margin*, *shrinklimit* und *showborder* werden berücksichtigt.
  Die linke untere Ecke der Objektbox kommt auf der linken unteren Ecke der Fitbox zu liegen. Mit der Option *position* können andere Punkte in der Objektbox und gleichzeitig der entsprechende Punkt in der Fitbox ausgewählt werden. Beispielsweise wird mit *position=center* der Mittelpunkt der Objektbox am Mittelpunkt der Fitbox platziert.

Tabelle 6.1 Optionen zur Objekteinpassung in verschiedenen Funktionen

Option	Beschreibung
align	*(Liste aus zwei Floats; nur für Pfadobjekte) Die Koordinaten eines Richtungsvektors in Benutzerkoordinaten, der die Drehung des Pfadobjekts festlegt. Die x-Achse des Koordinatensystems des Pfadobjekts wird am angegebenen Vektor ausgerichtet. Die beiden Koordinaten dürfen nicht gleichzeitig den Wert 0 annehmen. Der berechnete Drehwinkel wird zum Drehwinkel addiert, der mit der Option* `orientate` *festgelegt wurde. Standardwert:* {1 0}*, das heißt, keine zusätzliche Drehung*
alignchar	*(Unichar < 0xFFFF oder Schlüsselwort; nur für Textlines) Wird das angegebene Zeichen im Text gefunden, so wird es mit der linken unteren Ecke am Referenzpunkt ausgerichtet. Bei horizontalem Text mit* `orientate=north` *oder* `south` *definiert der erste in der Option* `position` *übergebene Wert die Position. Bei horizontalem Text mit* `orientate=west` *oder* `east` *definiert der zweite in der Option* `position` *übergebene Wert die Position.* *Ist diese Option vorhanden, kann der formatierte Text über die Fitbox hinausreichen. Diese Option wird ignoriert, wenn das festgelegte Ausrichtungszeichen im Text nicht vorkommt. Ist das angegebene Zeichen im Font oder Encoding nicht verfügbar, so wird bei* `glyphcheck=error` *eine Exception ausgelöst. Bei anderen* `glyphcheck`*-Werten wird die Option* `alignchar` *ignoriert, sollte das Zeichen nicht verfügbar sein.* *Beim Wert 0 und dem Schlüsselwert* `none` *erfolgt keine Ausrichtung. Die in* `fitmethod` *angegebene Methode wird auch dann angewandt, wenn der Text wegen der erzwungenen Positionierung des Ausrichtungszeichens nicht innerhalb der Fitbox platziert werden kann. Standardwert:* `none`
attachment-point	*(String; nur für Pfadobjekte) Name des Anschlusspunkts. Das Pfadobjekt wird so platziert, dass der angegebene Anschlusspunkt auf dem Referenzpunkt zu liegen kommt. Wenn* `fitmethod` *ungleich* `nofit` *ist, wird das Objekt gemäß der angegebenen Methode in der Fitbox platziert. Standardwert: Ursprung des Pfadobjekts*
blind	*(Boolean) Bei* `true` *wird keine Ausgabe generiert, aber alle Berechnungen werden durchgeführt. Die Formatierungsergebnisse können mit der entsprechenden Funktion PDF_info_*() abgefragt werden. Standardwert:* `false`
boxsize	*(Liste aus zwei Floats; nicht für Tabellen) Breite und Höhe der Fitbox relativ zu der das Objekt platziert wird (und gegebenenfalls gemäß der Option* `rotate` *gedreht wird). Die linke untere Ecke der Fitbox kommt auf dem Referenzpunkt (x, y) zu liegen. Die Platzierung des Objekts wird durch die Optionen* `position` *und* `fitmethod` *gesteuert. Bei* `width=0` *wird nur die Höhe berücksichtigt. Bei* `height=0` *wird nur die Breite berücksichtigt. In diesen beiden Fällen wird die Option* `fitmethod` *ignoriert und das Objekt relativ zur senkrechten Strecke von* `(x, y)` *nach* `(x, y+height)` *(oder* `(x, y-height)` *bei* `topdown`*-Systemen) bzw. zur waagrechten Strecke* `(x, y)` *nach* `(x+width, y)` *so platziert, wie es durch den entsprechenden Wert der Option* `position` *vorgegeben ist.* *Standardwert für Blöcke: Breite und Höhe der Blockeigenschaft* `Rect` *Standardwert für alle anderen Funktionen zur Objekteinpassung:* {0 0}
dpi	*(Liste aus zwei Floats oder Schlüsselwörtern; nur für Rasterbilder) Ein oder zwei Werte, die die gewünschte Bildauflösung in Pixeln pro Zoll in horizontaler und vertikaler Richtung angeben. Die Anzahl der Pixel im Bild (Downsampling) wird durch die Option nicht verändert. Wird nur ein einzelner Wert übergeben, wird er für beide Richtungen verwendet. Beim Wert null wird die interne Auflösung des Bilds verwendet, sofern vorhanden, und sonst 72 dpi. Alternativ dazu kann das Schlüsselwort* `internal` *übergeben werden. Die durch diese Option gegebene Auflösung bezieht sich auf das aktuelle Benutzerkoordinatensystem; wurde dieses skaliert, weicht die tatsächliche physikalische Auflösung von den hier übergebenen Werten ab. Zusätzlich zu den* `dpi`*-Werten wird die Option* `scale` *angewendet.* *Wird die Option* `fitmethod` *mit einem der Schlüsselwörter* `auto`, `meet`, `slice` *oder* `entire` *übergeben, wird mit den* `dpi`*-Werten nur das Seitenverhältnis, aber nicht die absolute Größe des Bildes festgelegt. Standardwert:* `internal`
fitmethod	*(Schlüsselwort) Methode zum Einpassen des Objekts in die Fitbox. Für unterstützte Schlüsselwörter siehe Tabelle 6.2. Wurde keine Fitbox angegeben, werden alle Schlüsselwörter außer* `nofit` *ignoriert.* *Standardwert:* `clip` *für Textflow;* `meet` *für Tabellen, Pfadobjekte und die Option* `reference`*; sonst* `nofit`

Tabelle 6.1 Optionen zur Objekteinpassung in verschiedenen Funktionen

Option	Beschreibung
margin	*(Float-Liste; nur für Textlines) Ein oder zwei Float-Werte für einen zusätzlichen horizontalen und vertikalen Rand um die Fitbox. Standardwert: 0*
matchbox	*(Optionsliste; nur für Pfadobjekte) Optionsliste zur Erzeugung einer Matchbox gemäß Tabelle 6.4*
minfontsize	*(Float oder Prozentwert; nur für Textflow) Erlaubte minimale Fontgröße, wenn Text mit* `fitmethod=auto` *verkleinert wird, falls* `shrinklimit` *überschritten wurde. Der Grenzwert wird in Benutzerkoordinaten oder als Prozentwert der Höhe der Fitbox angegeben. Ist er erreicht, wird der Text mit der in* `minfontsize` *angegebenen Fontgröße erzeugt. Standardwert: 0.1%*
orientate	*(Schlüsselwort oder Float; nicht für Tabellen) Legt die gewünschte Ausrichtung des Objekts in Benutzerkoordinaten fest. Standardwert:* `north`. *Nur für Pfadobjekte (nicht aber für andere Objekttypen) können beliebige Drehwinkel festgelegt werden (in Grad). Die Boundingbox des Pfadobjekts wird berechnet, nachdem das Pfadobjekt gedreht wurde. Alle Funktionen unterstützen die folgenden Schlüsselwörter (die entsprechenden Winkel sind in Klammern angegeben):* ***north*** *nach oben (0)* ***east*** *nach rechts (270)* ***south*** *nach unten (180)* ***west*** *nach links (90)*
position	*(Float- oder Schlüsselwort-Liste) Ein oder zwei Werte, die die Position des Objekts relativ zum Referenzpunkt, zum Referenz-Liniensegment oder zur Fitbox festlegen. Die Werte legen dabei eine Position innerhalb der Objektbox fest. Diese Position wird horizontal definiert in Prozent der Boxbreite (erster Wert) und vertikal in Prozent der Boxhöhe (zweiter Wert). Diese Position fällt mit dem Referenzpunkt, einem Punkt auf dem Referenz-Liniensegment oder einem Punkt innerhalb der Fitbox zusammen. Obwohl es sich hierbei um Prozentwerte handelt, müssen sie ohne Prozentzeichen angegeben werden. Negative Werte sind erlaubt. Sind die beiden Werte gleich, so kann ein Wert weggelassen werden.* *Standardwert:* `{0 100}` *für Tabellen,* `center` *für die Option* `reference`*, sonst* `{0 0}`*. Beispiele:* `{0 0}` *Die linke untere Ecke der Objektbox fällt mit dem Referenzpunkt, dem Startpunkt auf dem Referenz-Liniensegment oder der linken untere Ecke der Fitbox zusammen.* `{100 100}` *Die rechte obere Ecke der Objektbox fällt mit dem Referenzpunkt, dem Endpunkt auf dem Referenz-Liniensegment oder der rechten oberen Ecke der Fitbox zusammen.* *Statt der Werte 0, 50 und 100 können die Schlüsselwörter* `left`, `center` *und* `right` *(in x-Richtung) oder* `bottom`, `center` *und* `top` *(in y-Richtung) verwendet werden. Wird nur ein einziges Schlüsselwort angegeben, wird für die andere Richtung das entsprechende Schlüsselwort verwendet.* *Beispiele:* `{left center}` *oder* `{0 50}` *linksbündig* `{center}` *oder* `{50 50}` *zentriert* `{right center}` *oder* `{100 50}` *rechtsbündig* *Nur für Textlines: Das Schlüsselwort* `auto` *kann als erster Wert in der Liste verwendet werden. Es bedeutet* `right` *bei linksläufiger Textausgabe (z.B. bei arabischem oder hebräischem Text) und sonst* `left`.
refpoint	*(Float-Liste; nur für PDF_fill_*block( ) und PDF_info_path( )) Legt den Referenzpunkt in Benutzerkoordinaten zur Einpassung des Blockinhalts oder Pfades fest.* *Standardwert für PDF_fill_*block( ): Linke untere Ecke des Rechtecks, das durch die Blockeigenschaft* `Rect` *definiert wurde* *Standardwert für PDF_info_path( ):* `{0 0}`

Tabelle 6.1 Optionen zur Objekteinpassung in verschiedenen Funktionen

Option	Beschreibung
rotate	*(Float; nicht für Tabellen und Pfadobjekte) Dreht das Koordinatensystem, wobei der Referenzpunkt als Mittelpunkt und der übergebene Wert als Drehwinkel in Grad benutzt wird. Dabei werden die Fitbox und das Objekt gedreht. Die Drehung wird zurückgenommen, sobald das Objekt platziert wurde. Standardwert: 0* *Textlines in Tabellenzellen: wenn die Option* `rotate` *mit einem Wert verschieden von 0 angegeben wurde, versucht der Tabellenformatierer, die Boundingbox des gedrehten Textes in die Zellenbox gemäß den Optionen* `fitmethod` *und* `position` *einzupassen. Ist* `fitmethod` *verschieden von* `auto`*, wird die Zelle gegebenenfalls entsprechend vergrößert.*
scale	*(Float-Liste; nicht für Textlines und Textflow) Skaliert das Objekt in horizontaler und vertikaler Richtung mit den übergebenen Skalierungsfaktoren (keine Prozentwerte) und dem Referenzpunkt als Mittelpunkt. Sind beide Faktoren gleich, braucht nur ein einziger Float-Wert übergeben zu werden. Bei negativen Werten wird gespiegelt. Diese Option wird ignoriert, falls die Option* `fitmethod` *mit einem der Schlüsselwörter* `meet`, `slice` *oder* `entire` *angegeben wurde. Standardwert:* `{1 1}`
stamp	*(Schlüsselwort; nur für Textlines; wird nur in Kombination mit* `boxsize` *berücksichtigt) Mit dieser Option lässt sich ein diagonaler Stempel von maximaler Größe in dem mit der Option* `boxsize` *festgelegten Rechteck erstellen. Der Text wird dabei diagonal in der Fitbox platziert. Die Größe der Textbox wird dabei so gewählt, dass sie die Fitbox so weit möglich ausfüllt unter gleichzeitiger Beibehaltung der Proportionen der Textbox (das heißt, der Stempeltext wird dabei so groß wie möglich ausgegeben). Die Optionen* `fontsize`, `fitmethod` *und* `position` *werden ignoriert. Die Optionen* `orientate=west` *und* `=east` *sind nicht sinnvoll (nur* `north` *und* `south` *werden berücksichtigt). Unterstützte Schlüsselwörter (Standardwert:* `none`*):* **ll2ur** *Der Stempel verläuft diagonal von der linken unteren in die rechte obere Ecke.* **ul2lr** *Der Stempel verläuft diagonal von der linken oberen in die rechte untere Ecke.* **none** *Es wird kein Stempel erstellt.*
showborder	*(Boolean) Bei* `true` *wird unter Anwendung des aktuellen Grafikzustands ein Rahmen um die Fitbox gezeichnet. Wird ein Stempel erstellt, wird seine Boundingbox ebenfalls umrandet. Dies kann im Entwicklungsstadium und bei der Fehlersuche nützlich sein. Standardwert:* `false`
shrinklimit	*(Float oder Prozentwert; nur für Textlines) Untere Grenze, bis zu der Text beim Einpassen mit* `fitmethod=auto` *maximal gestaucht werden darf. Standardwert: 0.75*

Tabelle 6.2 Schlüsselwörter für die Option fitmethod *in verschiedenen Funktionen; die Abbildungen zeigen die typische Auswirkung der Schlüsselwörter auf eine Textzeile, alle Beispielen verwenden denselben Wert für die Option* fontsize.

Schlüsselwort	Beschreibung
auto	*Diese Methode versucht, die Objektbox automatisch in die Fitbox einzupassen:* *Falls das Objekt in die Box passt, wird die Methode* nofit *angewandt, das heißt, das Objekt wird ohne Skalierung platziert.* *Falls das Objekt größer ist als die Box, wird die Größe folgendermaßen proportional reduziert:* *Textlines: Der Skalierungsfaktor wird so berechnet, dass das Objekt horizontal (verzerrt) in die Box passt. Ist dieser Faktor kleiner als der in der Option* shrinklimit *definierte Wert, kommt die Methode* meet *zum Tragen. Dabei wird die Fontgröße so lange reduziert, bis der Text in die Box passt oder bis der Wert von* infontsize *erreicht ist.* *Andere Objekttypen: das Verhalten ist identisch zur Methode* meet.
clip	*Positioniert das Objekt und beschneidet es an den Rändern der Box.* *PDF_fit_table( ): die berechnete Tabellenbox wird am unteren Rand der Fitbox logisch beschnitten und kann in der nächsten Fitbox fortgeführt werden. Logisches Clipping verhält sich ähnlich wie PDF_fit_textflow( ), aber nicht wie grafisches Clipping mit PDF_fit_image( ) usw. Die Tabellenbox wird gemäß der Option* position *in der Fitbox platziert.*
entire	*Skaliert die Objektbox so, dass sie die Fitbox vollständig ausfüllt. In der Regel wird das Objekt durch dieses Verfahren verzerrt. Die Option* position *wird ignoriert.* *PDF_fit_table( ): ähnlich wie* clip. *Ist die Tabellenbox kleiner als die Fitbox, werden die Zellen der Tabellenbox (nicht aber ihr Inhalt) gleichmäßig vergrößert, bis die Tabellenbox die Fitbox vollständig ausfüllt.*
meet	*Positioniert das Objekt gemäß der Option* position *und skaliert es so, dass es unter Beibehaltung seiner Proportionen vollständig in die Fitbox hineinpasst. Dabei kommen mindestens zwei Ränder der Objektbox genau auf den Rändern der Fitbox zu liegen.* *PDF_fit_table( ): ähnlich wie* clip. *Ist die Tabellenbox kleiner als die Fitbox, werden die Zellen der Tabellenbox (nicht aber ihr Inhalt) gleichmäßig vergrößert, bis die horizontalen oder vertikalen Tabellenränder die Fitbox vollständig bedecken.*
nofit	*Platziert lediglich das Objekt. Die Option* scale *wird auf Rasterbilder und Grafiken angewendet, die Option* dpi *wird nur auf Rasterbilder angewendet.* *PDF_fit_table( ): Die Tabelle wird für eine virtuelle Fitbox mit unbegrenzter Höhe berechnet. Die Tabellenbox wird gemäß der Option* position *in der Fitbox platziert. Die Standardgröße von Spalten und Zeilen richtet sich nach der angegebenen Höhe der Fitbox.* fitmethod=nofit *wird zur Formatierung von Tabellen im* blind*-Modus empfohlen.*
slice	*Positioniert das Objekt gemäß der Option* position *und skaliert es so, dass es die Fitbox unter Beibehaltung seiner Proportionen vollständig ausfüllt, wobei gewährleistet ist, dass der Text in mindestens einer Dimension vollständig in der Box enthalten ist. In der Regel ragt der Text in der anderen Dimension teilweise aus der Box hinaus und wird deshalb beschnitten.* *PDF_fit_table( ): ähnlich wie* clip. *Ist die Tabellenbox kleiner als die Fitbox, werden die Zellen der Tabellenbox (nicht aber ihr Inhalt) gleichmäßig und unter Beibehalten ihrer Proportionen vergrößert, bis die Tabellenbox die Fitbox vollständig ausfüllt. Die Tabellenbox wird gemäß der Option* position *in der Fitbox platziert. Die Teile der Tabellenbox, die über die Fitbox hinausragen, werden grafisch an den Rändern der Fitbox beschnitten.*

Allgemeine Schlüsselwörter zur Abfrage von Ergebnissen der Objekteinpassung. Die Ergebnisse der Objekteinpassung können abgefragt werden, ohne das Objekt tatsächlich auf der Seite zu platzieren. Damit kann vor dem eigentlichen Erzeugen des Seiteninhalts die Formatierung geprüft werden. Formatierungsergebnisse lassen sich durch Übergabe der Optionen zur Objekteinpassung an die entsprechende Funktion *PDF_info_*()* für das Objekt abfragen. In Tabelle 6.3 werden die entsprechenden Schlüsselwörter aufgeführt. Die Formatierungsergebnisse für *PDF_info_path()* werden relativ zum Referenzpunkt angegeben.

Tabelle 6.3 Allgemeine Schlüsselwörter zur Abfrage von Ergebnissen der Objekteinpassung mit PDF_info_image(), PDF_info_graphics(), PDF_info_path(), PDF_info_pdi_page(), PDF_info_table(), PDF_info_textline()

Schlüsselwort	Beschreibung
boundingbox	*Pfad-Handle für die Boundingbox des Objekts*
fitscalex, fitscaley	*Skalierungsfaktoren, die sich aus dem Einpassen des Objekts in eine Box ergeben haben*
height	*Höhe des Objekts in Benutzerkoordinaten*
objectheight, objectwidth	*Größe des Objekts nach Verarbeitung aller Optionen, die für das Laden oder Erzeugen des Objekts relevant sind. Diese Größe wird vom Algorithmus für die Objekteinpassung verwendet.*
width	*Breite des Objekts in Benutzerkoordinaten*
x1, y1, x2, y2, x3, y3, x4, y4	*Position der i-ten Ecke der Boundingbox des Objekts (i=1, 2, 3, 4) in Benutzerkoordinaten gemäß den übergebenen Optionen*

6.2 Matchboxen

Matchboxen werden mit keiner besonderen Funktion definiert, sondern mit der Option *matchbox* für die Funktion festgelegt, die das betreffende Element erzeugt:

- Textlines mit *PDF_fit_textline( )*, *PDF_fill_textblock( )* mit der Eigenschaft *textflow=false:* die Matchbox beschreibt die Boundingbox der Textzeile.
- Textflow mit *PDF_add/create_textflow( )*, *PDF_fill_textflow( )*, *PDF_fill_textblock( )* mit der Eigenschaft *textflow=true:* die Matchbox beschreibt die Boundingbox der generierten Textausgabe. Matchbox-Angaben in *PDF_fill_textbloc( )* können nicht als Startpunkt für die Inline-Textdekoration verwendet werden, sondern nur für die Erzeugung einer Matchbox für den gesamten Text.
- Importierte PDF-Seiten mit *PDF_fit_pdi_page( )*, *PDF_fill_pdf_block( ):* die Matchbox beschreibt die Boundingbox der platzierten Seite.
- Rasterbilder und Templates mit *PDF_fit_image( )*, *PDF_fill_image_block( ):* die Matchbox beschreibt die Boundingbox des platzierten Bilds oder Templates.
- Grafik mit *PDF_fit_graphics( ):* die Matchbox beschreibt die Boundingbox der platzierten Grafik.
- Tabellenzellen: *PDF_add_table_cell( ):* die Matchbox beschreibt die Boundingbox der platzierten Tabellenzelle.

Matchboxen werden mit der Option *matchbox* dieser Funktionen definiert. Diese erwartet eine Optionsliste, die die folgenden Unteroptionen enthalten kann:

- Optionen zur Grafikdarstellung gemäß Tabelle 7.1:
  borderwidth, dasharray, dashphase, fillcolor, gstate, linecap, linejoin, shading, strokecolor
- Optionen zur Steuerung von Matchboxen gemäß Tabelle 6.4;
- Option zum vereinfachten Tagging von Strukturelementen gemäß Tabelle 14.5 (nur erlaubt im Gültigkeitsbereich *page): tag*

Detaillierte Angaben über die Rechtecke, die zur einer Matchbox gehören, lassen sich mit *PDF_info_matchbox( )* abfragen.

Tabelle 6.4 Unteroptionen für die Option `matchbox` *in verschiedenen Funktionen*

Option	*Beschreibung*
boxheight	*(Liste aus zwei Elementen, die einen positiven Float-Wert, einen Prozentwert der Fontgröße oder ein Schlüsselwort darstellen; nur für Textzeilen und Textflow) Vertikale Ausdehnung der Textbox. Für die Ausdehnung über und unter der Grundlinie können zwei numerische Werte oder Schlüsselwörter angegeben werden:* `none` *(keine Ausdehnung),* `xheight`, `descender`, `capheight`, `ascender`, `fontsize`, `leading`, `textrise` *Bei Textflow werden die Werte verwendet, die dem Text am Anfang der Matchbox entsprechen.* *Standardwert:* `{capheight none}`
boxwidth	*(Float oder Prozentwert; nur für Textflow) Breite der Matchbox in Benutzerkoordinaten oder als Prozentwert der Breite der Fitbox. Wird diese Option übergeben, wird ein horizontaler Abstand der festgelegten Breite zwischen der Option* `matchbox` *und dem nächsten Textfragment bzw. der Spezifikation* `matchbox end` *eingefügt. Beachten Sie, dass bei* `alignment=justify` *die Breite der Box ebenso komprimiert werden kann wie der Text (siehe Option* `shrinklimit`*). Standardwert: 0*

Tabelle 6.4 Unteroptionen für die Option `matchbox` *in verschiedenen Funktionen*

Option	Beschreibung
clipping	*(Rechteck oder 4 Prozentwerte; nur für Rasterbilder, Grafik und importierte PDF-Seiten; wird ignoriert, wenn die Option* `innerbox` *angegeben wird) Koordinaten der linken unteren und rechten oberen Ecke eines Rechtecks innerhalb des Rasterbildes, der Grafik oder der Seite, die den anzuzeigenden Teilbereich festlegen. Die Angaben sind abhängig vom Objekttyp. Standardwert:* `{0% 0% 100% 100%}`*:* ▸ *Bei Bildern kann das Clipping-Rechteck in Pixeln oder als prozentualer Anteil an der Breite/Höhe angegeben werden.* ▸ *Bei Grafiken kann das Clipping-Rechteck in Benutzerkoordinaten oder als prozentualer Anteil an der Breite/Höhe der Grafikbox angegeben werden.* ▸ *Bei PDF-Seiten kann das Clipping-Rechteck in Standardeinheiten oder als prozentualer Anteil an der Cropbox der Seite angegeben werden.*
create-wrapbox	*(Boolean; nur für Textflow) Bei* `true` *werden das oder die Rechtecke, aus denen die Matchbox besteht, nach ihrer Berechnung als Wrapping-Bereiche in den Textflow eingefügt. Alle Zeilen, die auf die Zeilen mit der Matchbox folgen, fließen um die Matchbox-Rechtecke herum. Standardwert:* `false`
doubleadapt	*Bei* `true` *werden Start- und Endpunkt der zweiten Linie an die erste Linie angepasst. Anderenfalls wäre die zweite Linie um den Wert von* `doubleoffset` *kürzer oder länger. Standardwert:* `true`
doubleoffset	*(Float) Ist der Wert verschieden von 0, werden die Linien um den Rand des inneren Matchbox-Rechtecks verdoppelt. Die zweite Linie hat den angegebenen Abstand von der ursprünglichen Linie. Ist der Abstand positiv, wird die Linie außerhalb des Matchbox-Rechtecks gezeichnet und bei einem negativen Abstand innerhalb des Matchbox-Rechtecks. Standardwert: 0 (das heißt, eine einzelne Linie)*
drawleft drawbottom drawright drawtop	*(Boolean) Bei* `true` *wird der entsprechende Rechteckrand gezeichnet, sofern die Option* `borderwidth` *auf einen Wert größer 0 gesetzt wurde. Standardwert:* `true`
end	*(Boolean; nur für Textflow) Legt das Ende der Matchbox fest. Bei* `true` *werden alle anderen Unteroptionen für die aktuelle Matchbox-Definition ignoriert. Matchboxen in Textflows lassen sich nicht verschachteln. Die Breite einer Textflow-Matchbox definiert sich durch die Option* `boxwidth` *(sofern vorhanden) und die Ausdehnung des Texts, der in die Optionen* `matchbox` *und* `matchbox=end` *eingeschlossen ist. Wurde die Unteroption* `end` *nicht angegeben, so endet die Matchbox nach dem letzten zum Textflow gehörenden Zeichen.*
exceedlimit	*(Float oder Prozentwert; nur für Textflow) Obere Grenze für den Teil der Matchbox, der über den unteren oder rechten Rand der Fitbox hinausreichen darf, in Benutzerkoordinaten oder als Prozentwert der Höhe der Matchbox. Wird der angegebene Grenzwert überschritten, gibt* PDF_fit_textflow( ) `_boxfull` *zurück; der restliche Text und die Matchbox können in der nächsten Fitbox fortgeführt werden. Standardwert: 0, das heißt, die Matchbox muss vollständig in die Box passen.*
innerbox	*(Boolean; nur für Tabellenzellen sowie TIFF- und JPEG-Bilder) Tabellenzellen: Bei* `true` *wird die Zellenbox um die für die Zelle mit* `margin` *definierten Ränder verkleinert; anderenfalls wird die ganze Zellenbox verwendet.* *TIFF- und JPEG-Bilder: Ist* `innerbox=true` *und enthält das Bild einen Beschneidungspfad, wird die Boundingbox des Clipping-Pfades statt des gesamten Bildes verwendet.* *Standardwert:* `false`
margin	*(Float oder Prozentwert) Zusätzlicher Rand für das Matchbox-Rechteck in Benutzerkoordinaten (muss größer oder gleich 0 sein) oder als prozentualer Anteil an der Breite oder Höhe des Rechtecks (muss kleiner als 100% sein). Diese Option wird bei jeder Kante ignoriert, für die die Option* `offset*` *übergeben wurde. Standardwert: 0*

Tabelle 6.4 Unteroptionen für die Option matchbox *in verschiedenen Funktionen*

Option	*Beschreibung*
name	*(Name-String) Name der Matchbox. Gibt es bereits eine Matchbox dieses Namens, so wird ein weiteres Rechteck unter diesem Namen erstellt. Das bedeutet, dass eine Matchbox aus mehreren Rechtecken bestehen kann. Der Name kann in PDF_info_matchbox( ) verwendet werden. Verschiedene Funktionen unterstützen die Option* usematchbox*, um eines oder mehrere Rechtecke einer Matchbox zu referenzieren und zum Beispiel mit PDF_create_annotation( ) ein interaktives Element hinzuzufügen. Matchbox-Namen können bis zum Ende der aktuellen Seite verwendet werden. Der Name »*«* (Asterisk; einzelner Stern) sollte nicht als Matchbox-Name verwendet werden. Standardwert: kein Name*
offsetleft ***offsetbottom*** ***offsetright*** ***offsettop***	*(Float oder Prozentwert) Benutzerdefinierter Offset von der linken/rechten/unteren/oberen Kante des berechneten Rechtecks und der gewünschten Box. Die Werte werden in Benutzerkoordinaten oder als prozentualer Anteil der Rechteckbreite (für* offsetleft/offsetright*) bzw. Höhe (für* offsetbottom/offsettop*) angegeben. Negative Werte sind zulässig und können zur Erweiterung der Matchbox verwendet werden. Standardwert für* offsetleft/offsetbottom: margin*; Standardwert für* offsetright/offsettop: -margin
openrect	*(Boolean; nur für Textflow und Tabellenzellen) Textflow: Bei* true *und wenn ein Matchbox-Rechteck aufgeteilt werden muss (z.B. wegen eines Fontwechsels oder Zeilenumbruchs), dann werden weder der rechte Rand des ersten Rechtecks noch der linke Rand des zweiten Rechtecks gezeichnet. Tabellenzellen: Bei* true *und wenn eine Tabellenzeile aufgeteilt und in der nächsten Tabelleninstanz fortgeführt wird, dann werden weder der untere Rand des ersten Teils noch der obere Rand des zweiten Teils gezeichnet. Standardwert:* false
round	*(Float) Benachbarte Linien eines Matchbox-Rechtecks werden durch einen Kreisbogen mit den Liniensegmenten als Tangenten und mit dem angegeben Radius abgerundet. Ist der Radius negativ, werden die Bogensegmente nach innen geschwungen und die Tangenten werden senkrecht zu den Liniensegmenten der Box. Standardwert: 0 (nicht gerundet)*

C++ Java C# ***double info_matchbox(String boxname, int num, String keyword)***

Perl PHP ***float info_matchbox(string boxname, int num, string keyword)***

C ***double PDF_info_matchbox(PDF *p, const char *boxname, int len, int num, const char *keyword)***

Fragt Informationen über eine Matchbox auf der aktuellen Seite ab.

boxname (Name-String) Name einer Matchbox, die unter diesem Namen auf der aktuellen Seite erstellt wurde. Der Name muss bei der Definition der Matchbox mit der Unteroption *name* der Option *matchbox* erstellt worden sein. Alternativ dazu kann mit dem Namen '*' (ein einzelner Stern) Information über alle Matchboxen auf der Seite abgefragt werden. Mit einem leeren *boxname* kann Information über alle Matchbox-Rechtecke auf der aktuellen Seite abgefragt werden.

len (Nur C-Sprachbindung) Länge von *name* in Bytes. Ist *len*=0, muss ein null-terminierter String übergeben werden.

num Positive Nummer einer Matchbox oder eines Rechtecks (beginnend mit der Nummer 1).

keyword Schlüsselwort zur Auswahl der abzufragenden Information gemäß Tabelle 6.3.

Rückgabe Wert eines durch *keyword* festgelegten Matchbox-Parameters. Existiert keine Matchbox mit dem angegebenen Namen oder kein Matchbox-Rechteck mit der angegebenen Nummer, wird für die Schlüsselwörter *boundingbox*, *name* und *rectangle* -1 (in PHP:0) zurückgeliefert und für alle anderen Schlüsselwörter der Wert 0. Gibt das erforderliche

Schlüsselwort Text aus, wird ein String-Index zurückgegeben und der betreffende String muss mit *PDF_get_string( )* abgefragt werden.

Abhängig vom aktuellen Gültigkeitsbereich gibt die Funktion Informationen über die Matchboxen auf der aktuellen Seite, Pattern, Template oder Glyphenbeschreibung zurück.

Details Benannte Matchboxen innerhalb eines Textflows können erst nach dem Aufruf von *PDF_fit_textflow( )* abgefragt werden. Im Blind-Modus erstellte Matchboxen können nicht abgefragt werden.

Rechtecke für die Schlüsselwörter *boundingbox, exists, height, name, rectangle, width, x1, y1,...,x4, y4* werden folgendermaßen ausgewählt:

- Wenn *boxname* den Namen einer Matchbox enthält: Wählen Sie das *num*-te Rechteck der angegebenen benannten Matchbox auf der aktuellen Seite.
- Wenn *boxname=*:* Wählen Sie das erste Rechteck der *num*-ten benannten Matchbox auf der aktuellen Seite.
- Wenn *boxname* leer ist: Wählen Sie das *num*-te von einer benannten Matchbox erzeugte Rechteck auf der aktuellen Seite.

Gültigkeit alle außer *document* und *object*

Tabelle 6.5 Schlüsselwörter für PDF_info_matchbox( )

Schlüsselwort	Beschreibung
boundingbox	*Pfad-Handle mit der Boundingbox des ausgewählten Rechtecks in Benutzerkoordinaten oder -1 (in PHP: 0), falls das angegebene Rechteck nicht vorhanden ist. Die Boundingbox unterscheidet sich vom Rechteck, wenn die Matchbox gedreht wurde.*
count	*(Der Parameter* num *wird ignoriert)* *Wenn* boxname *den Namen einer Matchbox bezeichnet: Anzahl der zur Matchbox gehörenden Rechtecke* *Bei* boxname=*: *Anzahl der Matchboxen mit mindestens einem Rechteck* *Ist* boxname *leer: Gesamtzahl der von einer benannten Matchbox erzeugten Rechtecke*
exists	*1, wenn das ausgewählte Rechteck existiert, sonst 0*
height[1]	*Höhe des ausgewählten Rechtecks in Benutzerkoordinaten*
name	*String-Index für den Namen der Matchbox, für die das ausgewählten Rechteck erzeugt wurde. Der zugehörige String kann mit PDF_get_string( ) abgefragt werden.*
rectangle	*Handle des Pfadobjekts mit dem ausgewählten Rechteck in Benutzerkoordinaten oder -1 (in PHP: 0), wenn das Rechteck nicht gefunden werden konnte*
width[1]	*Breite des ausgewählten Rechtecks in Benutzerkoordinaten*
x1, y1, ... , x4, y4[1]	*Position der i-ten Ecke des ausgewählten Rechtecks in Benutzerkoordinaten (i=1, 2, 3, 4). Im Koordinatensystem des entsprechenden zu platzierenden Elements (Bild, Text usw.) bezeichnet* x1, y1 *die linke untere,* x2, y2 *die rechte untere,* x3, y3 *die rechte obere und* x4, y4 *die linke obere Ecke.*

1. Dieses Schlüsselwort wird bei boxname=* *ignoriert.*

7 Grafikfunktionen

Cookbook Ein vollständiges Codebeispiel finden Sie im Cookbook-Topic `graphics/starter_graphics`.

7.1 Optionen zur Grafikdarstellung

Die Optionen zur Grafikdarstellung in Tabelle 7.1 können mit den folgenden Funktionen verwendet werden (beachten Sie, dass nicht alle Funktionen alle Optionen unterstützten; für weitere Informationen siehe die Funktionsbeschreibungen):

- *PDF_set_graphics_option( )*
- *PDF_create_gstate( )* (nur *flatness, linecap, linejoin, linewidth, miterlimit)*
- *PDF_add_path_point( )* und *PDF_draw_path( )*
- Die Option *fill* von *PDF_fit_table( )* (nur *fillcolor, shading)* und die Option *stroke* von *PDF_fit_table( )* (nur *dasharray, dashphase, linecap, linejoin, linewidth, strokecolor)*
- Die Option *matchbox* in verschiedenen Funktionen

Tabelle 7.1 Optionen zur Grafikdarstellung

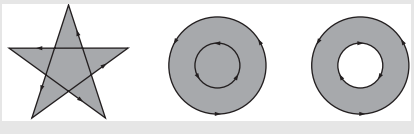
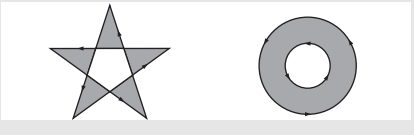
Option	Beschreibung
cliprule	*(Schlüsselwort) Clipping-Regel, die das Flächeninnere für das Clipping festlegt; für mögliche Schlüsselwörter siehe* `fillrule`. *Standardwert: Wert der Option* `fillrule`
borderwidth	*(Float; nur für Matchboxen) Strichstärke für die Rechteckränder. Ist* `borderwidth` *größer 0, so werden alle Rechteckränder gezeichnet. Um den oberen, unteren, linken oder rechten Rand nicht zu zeichnen, setzen Sie die entsprechende Option* `drawtop`, `drawbottom`, `drawleft` *oder* `drawright` *auf* `false`. *Standardwert: 0*
dasharray	*(Liste aus zwei nicht negativen Floats oder Schlüsselwort) Liste aus 2 bis 12 alternierenden Werten für Strichlängen und Zwischenräume einer gestrichelten Linie zum Zeichnen von Pfaden (gemessen in Benutzerkoordinaten). Die Array-Werte dürfen nicht negativ sein. Sie werden zyklisch wiederverwendet, bis der Pfad vollständig gezeichnet ist. Das Schlüsselwort* none *kann zum Zeichnen von durchgezogenen Linien verwendet werden. Standardwert:* none
dashphase	*(Float) Abstand vom Anfang des Pfades, nach dem der erste Strich beginnt. Standardwert: 0*
fillcolor	*(Farbe) Füllfarbe der Fläche. Standardwert: in der Regel* `{gray 0}` *(im PDF/A-Modus:* `{lab 0 0 0}`*), aber* none *für Tabellen und Matchboxen*
fillrule	*(Schlüsselwort) Füllregel, die das Flächeninnere für das Clipping und Füllen festlegt (Standardwert:* `winding`*):* **winding** *Die Nonzero-Winding-Number-Regel wird angewendet. Bei einfachen Formen entspricht das Ergebnis des Füllens der intuitiven Erwartung. Bei Formen mit mehreren Pfaden ist die Pfadrichtung relevant.*  **evenodd** *Die Even-Odd-Regel wird angewendet. Bei einfachen Formen liefert sie die gleichen Ergebnisse wie* `winding`*, bei komplexen Formen, besonders bei sich überschneidenden Pfaden, weichen die Ergebnisse jedoch ab.* 
flatness	*(Float > 0) Positive Zahl, die den maximalen Abstand (in Gerätepixeln) zwischen einem Kreisbogen oder einer Kurve und einer Näherung aus geraden Liniensegmenten beschreibt. Standardwert: 1*

Tabelle 7.1 Optionen zur Grafikdarstellung

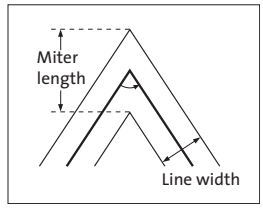
Option	Beschreibung
gstate	*(Gstate-Handle) Mit PDF_create_gstate( ) erzeugtes Handle für einen Grafikzustand. Standardwert: kein Grafikzustand (das heißt, die aktuellen Einstellungen werden verwendet)*
initgraphics-state	*(Boolean; nur für PDF_set_graphics_option( )) Bei* `true` *werden alle Optionen zur Grafikdarstellung mit den Standardwerten initialisiert. Der aktuelle Beschneidungspfad ist davon nicht betroffen. Bei* `false` *werden die aktuellen Werte für den Grafikzustand verwendet. Standardwert:* `false`
linecap	*(Integer oder Schlüsselwort) Art der Linienenden eines Pfads (Standardwert:* `projecting` *in PDF_fit_table( ), sonst* `butt`*):*  ***butt*** *(Äquivalenter Wert: 0) Abgeschnittene Linienenden: Die Linie wird am Endpunkt des Pfads rechtwinklig abgeschnitten.* ***round*** *(Äquivalenter Wert: 1) Abgerundete Linienenden: Ein Halbkreis mit einem der Strichstärke entsprechenden Durchmesser wird um den Endpunkt gezeichnet und gefüllt.*  ***projecting*** *(Äquivalenter Wert: 2) Hervorstehende rechtwinklige Linienenden: Das Linienende wird um die halbe Strichstärke verlängert und rechtwinklig abgeschnitten.* 
linejoin	*(Integer oder Schlüsselwort) Form der Ecken in Pfaden (Standardwert:* `miter`*):* ***miter*** *(Äquivalenter Wert: 0) Spitze Verbindungen (miter joins): Die beiden Pfadsegmente werden durchgezogen, bis die äußeren Ecken sich berühren. Steht die resultierende Spitze zu weit hervor, was durch die* `miterlimit`*-Einstellung festgelegt ist, wird stattdessen eine abgeschnittene Verbindung verwendet.*  ***round*** *(Äquivalenter Wert: 1) Abgerundete Verbindungen (round joins): Ein gefüllter Kreis mit einem der Strichstärke entsprechenden Durchmesser wird um den Punkt gezeichnet, in dem sich die Liniensegmente treffen. Das Ergebnis ist eine abgerundete Ecke.*  ***bevel*** *(Äquivalenter Wert: 2) Stumpfe Verbindungen (bevel joins): Die beiden Pfadsegmente werden nur durchgezogen, bis die inneren Ecken sich berühren (siehe Beschreibung der* `linecap`*-Einstellung). Die verbleibende Aussparung wird mit einem Dreieck gefüllt* 
linewidth	*(Float > 0) Strichstärke. Standardwert: 1*
miterlimit	*(Float >= 1) Legt den Grenzwert für das Abschneiden spitzer Liniensegmente fest (Standardwert: 10; dies entspricht einem Winkel von ungefähr 11,5 Grad)* *Ist der* `linejoin`*-Stil auf 0 gesetzt für eine spitze Verbindung (miter join), ergibt sich eine sehr dünne Spitze, wenn die beiden Liniensegmente in einem schmalen Winkel zusammenlaufen. Diese Spitze wird abgeschnitten, das heißt, die spitze Verbindung (miter join) wird durch eine stumpfe (bevel join) ersetzt, wenn das Verhältnis zwischen Spitzenbreite und Strichstärke den Wert der* `miterlimit`*-Einstellung übersteigt.* 
shading	*(Optionsliste gemäß Tabelle 7.2; nur für Matchboxen und Tabellen) Legt den Farbverlauf für Matchbox-Rechtecke oder Tabellenbereiche fest, unter Verwendung der Werte der Option* `fillcolor` *(wenn vorhanden) oder der aktuellen Füllfarbe als Ausgangsfarbe.*
strokecolor	*(Farbe) Farbe zum Zeichnen eines Pfads. Standardwert: in der Regel* `{gray 0}` *(im PDF/A-Modus:* `{lab 0 0 0}`*), aber* `none` *für Tabellen und Matchboxen*

Tabelle 7.2 Unteroptionen für die Grafikoption `shading`

Option	Beschreibung
antialias	*(Boolean) Legt fest, ob Antialiasing (Kantenglättung) für den Farbverlauf aktiviert wird. Standardwert:* `false`
domain	*(Liste aus 2 Floats) Zwei Zahlen, die Grenzwerte einer Variablen* `t` *festlegen. Die Variable bewegt sich linear zwischen diesen beiden Werten, während der Farbverlauf sich zwischen Start- und Endpunkt der Achse verändert. Standardwert:* `{0 1}`
end	*(Liste aus 2 Floats oder Prozentwerten)* `x-` *und* `y`*-Koordinaten des Endpunkts der Farbverlaufsachse (*`type=axial`*) oder eines Punktes auf einem Kreisbogen zur Berechnung des Radius (*`type=radial`*) in Benutzerkoordinaten oder als Prozentwert der Breite und Höhe des Rechtecks. Standardwert:* `{100% 100%}`
endcolor	*(Farbe; erforderlich) Farbe für den Endpunkt*
N	*(Float) Exponent der Farbübergangsfunktion; muss > 0 sein. Standardwert: 1*
start	*(Liste aus 2 Floats oder Prozentwerten)* `x-` *und* `y`*-Koordinaten des Startpunkts für die Farbverlaufsachse (*`type=axial`*) oder des Mittelpunkts des Farbverlaufskreises (*`type=radial`*) in Benutzerkoordinaten oder als Prozentwert der Breite und Höhe des Rechtecks. Standardwert:* `{0% 0%}`
type	*(Schlüsselwort) Farbverlaufstyp:* `axial` *(linearer Farbverlauf) oder* `radial` *(kreisförmiger Farbverlauf). Standardwert:* `axial`

C++ Java C# ***void set_graphics_option(String optlist)***

Perl PHP ***set_graphics_option(string optlist)***

C ***void PDF_set_graphics_option(PDF *p, const char *optlist)***

Setzt eine oder mehrere Optionen zur Grafikdarstellung.

optlist Optionsliste mit Optionen zur Grafikdarstellung gemäß Tabelle 7.1. Die folgenden Optionen können verwendet werden:

cliprule, dasharray, dashphase, fillcolor, fillrule, flatness, gstate, initgraphicsstate, linecap, linejoin, linewidth, miterlimit, strokecolor

Details Die Optionen zur Grafikdarstellung legen den Grafikzustand für die folgenden Funktionsgruppen fest:

- Funktionen für explizites Zeichnen, z.B. *PDF_stroke( )*, *PDF_fill( )*
- Funktionen für implizites Zeichnen, z.B. die Option *showborder* von *PDF_fit_textline( )*, *PDF_fit_textflow( )*
- mit den Funktionen zur einfachen Textausgabe erzeugte Textausgabe, wenn in den Textoptionen keine Farbe angegeben wurde, z.B. *PDF_show( )*

Am Anfang einer Seite, eines Patterns, Templates oder einer Glyphenbeschreibung werden alle Optionen zur Grafikdarstellung auf ihre Standardwerte zurückgesetzt. Sie behalten ihre Werte bis zum Ende des aktuellen Gültigkeitsbereichs *page*, *pattern*, *template* oder *glyph* bei. Diese Optionen können jedoch auch mit der Option *initgraphicsstate* zurückgesetzt werden.

Mit einem nachfolgenden Aufruf von *PDF_setcolor( )* werden alle Werte von *fillcolor* und/oder *strokecolor* überschrieben. Mit einem nachfolgenden Aufruf von *PDF_setlinewidth( )* werden alle *linewidth*-Werte überschrieben.

Gültigkeit *page, pattern, template, glyph*

7.2 Grafikzustand

C++ Java C# **void setlinewidth(double width)**
Perl PHP **setlinewidth(float width)**
C **void PDF_setlinewidth(PDF *p, double width)**

Setzt die aktuelle Strichstärke.

width Strichstärke in Einheiten des Benutzerkoordinatensystems.

Details Mit dieser Funktion wird die Strickstärke im Grafikzustand (siehe *PDF_set_graphics_option( )*) sowie die Strichstärke im Textzustand (siehe *PDF_set_text_option( )*) gesetzt. Am Anfang jeder Seite wird *width* auf den Standardwert 1 gesetzt.

Gültigkeit *page, pattern, template, glyph*

C++ Java C# **void save( )**
Perl PHP **save( )**
C **void PDF_save(PDF *p)**

Speichert den aktuellen Grafikzustand auf einem Stack.

Details Der Grafikzustand umfasst Optionen, die alle Arten von Grafikobjekten betreffen. Ein Speichern des Grafikzustands wird von PDF nicht gefordert; dies ist nur notwendig, wenn die Anwendung später wieder auf einen definierten Zustand zurückgreifen möchte (zum Beispiel auf ein benutzerdefiniertes Koordinatensystem), ohne erneut explizit alle relevanten Optionen einstellen zu müssen. Beim Speichern und Wiederherstellen des Grafikzustands werden folgende Optionen berücksichtigt:

- Optionen zur Grafikdarstellung:
  Beschneidungspfad, Koordinatensystem, aktuelle Position, Flatness, Linien-Endenstil *(linecap)*, Strichmuster *(dash)*, Art der Verbindungslinien *(linejoin)*, Strichstärke *(line width)*, Grenzwert für das Abschneiden spitzer Liniensegmente *(miter limit)*
- Farboptionen: Linien- und Füllfarben
- Grafikoptionen, die mit expliziten Grafikzuständen mit *PDF_set_gstate( )* gesetzt wurden;
- Textposition und folgende textbezogene Optionen:
  charspacing, decorationabove, fakebold, font, fontsize, horizscaling, italicangle, leading, strokewidth, textrendering, textrise, underlineposition, underlinewidth, wordspacing

Paare aus *PDF_save( )* und *PDF_restore( )* können verschachtelt werden. Obwohl es für diese Art der Verschachtelung in der PDF-Spezifikation keinen Grenzwert gibt, dürfen Anwendungen nicht mehr als 26 Verschachtelungsebenen nutzen. Anderenfalls können Probleme beim Drucken auftreten, die sich aus Einschränkungen bei der von PDF-Viewern erzeugten PostScript-Ausgabe ergeben. Außerdem sind einige Verschachtelungsebenen für den internen Gebrauch durch PDFlib reserviert.

Der *save/restore*-Mechanismus bezieht die meisten textbezogenen Optionen ein (siehe oben). Die folgenden Optionen werden nicht berücksichtigt:
fillrule, *kerning*, *underline*, *overline*, *strikeout*.

Gültigkeit *page, pattern, template, glyph;* diese Funktion muss immer paarweise mit *PDF_restore( )* aufgerufen werden. Genauer gesagt müssen *PDF_save( )* und *PDF_restore( )* paarweise innerhalb der Seite, des Patterns, des Templates oder der Glyphdefinition auftreten.

C++ Java C# ***void restore( )***
Perl PHP ***restore( )***
C ***void PDF_restore(PDF *p)***

Stellt den zuletzt im Stack gespeicherten Grafikzustand wieder her.

Details Der entsprechende Grafikzustand muss innerhalb derselben Seite, desselben Patterns, desselben Templates oder derselben Glyphdefinition gespeichert worden sein.

Gültigkeit *page, pattern, template, glyph;* diese Funktion muss immer paarweise mit *PDF_save( )* aufgerufen werden. Genauer gesagt müssen *PDF_save( )* und *PDF_restore( )* paarweise innerhalb der Seite, des Patterns, des Templates oder der Glyphdefinition auftreten.

C++ Java C# ***int create_gstate(String optlist)***
Perl PHP ***int create_gstate(string optlist)***
C ***int PDF_create_gstate(PDF *p, const char *optlist)***

Erzeugt ein Grafikzustandsobjekt unter Berücksichtigung verschiedener Optionen.

optlist Optionsliste mit Grafikzustandsoptionen:
- Optionen zur Grafikdarstellung gemäß Tabelle 7.1:
  flatness, linecap, linejoin, linewidth, miterlimit
- Grafikzustandsoptionen gemäß Tabelle 7.3:
  alphaisshape, blendmode, opacitystroke, overprintfill, overprintmode, overprintstroke, renderingintent, smoothness, softmask, strokeadjust, textknockout

Rückgabe Grafikzustands-Handle, das in späteren Aufrufen von *PDF_set_gstate( )* innerhalb des umgebenden Gültigkeitsbereichs *document* verwendet werden kann.

Details Die Optionsliste kann beliebig viele Grafikzustandsoptionen enthalten.

Gültigkeit beliebig außer *object*

C++ Java C# ***void set_gstate(int gstate)***

Perl PHP ***set_gstate(int gstate)***

C ***void PDF_set_gstate(PDF *p, int gstate)***

Aktiviert ein Grafikzustandsobjekt.

gstate Handle für ein Grafikzustandsobjekt, das von *PDF_create_gstate( )* zurückgegeben wurde.

Details Es werden alle im Grafikzustandsobjekt enthaltenen Optionen eingestellt. Wird diese Funktion mehrfach aufgerufen, so werden die neuen Optionen für den Grafikzustand zu den aktuellen hinzugefügt. Nicht explizit neu gesetzte Optionen behalten ihre aktuellen Werte bei. Am Anfang jeder Seite werden alle Grafikzustandsoptionen auf ihre Standardwerte zurückgesetzt.

Gültigkeit *page, pattern, template, glyph*

Tabelle 7.3 Optionen für PDF_create_gstate( )

Option	Beschreibung
alphaisshape	*(Boolean) Alpha-Quellen werden als Form* (`true`) *oder Deckkraft* (`false`) *verwendet. Standardwert:* `false`
blendmode	*(Schlüsselwortliste; bei PDF/X-1/3 oder PDF/A-1 muss der Wert* `Normal` *verwendet werden) Name des Farbmischmodus. Es können mehrere Farbmischmodi festgelegt werden. Mögliche Werte:* `Color`, `ColorDodge`, `ColorBurn`, `Darken`, `Difference`, `Exclusion`, `HardLight`, `Hue`, `Lighten`, `Luminosity`, `Multiply`, `None`, `Normal`, `Overlay`, `Saturation`, `Screen`, `SoftLight`. *Standardwert:* `None`
opacityfill	*(Float; im PDF/A-1-Modus muss der Wert 1 verwendet werden) Deckkraft für das Füllen von Flächen im Bereich 0...1. Der Wert 0 bedeutet vollständig transparent; 1 bedeutet völlig undurchsichtig.*
opacitystroke	*(Float; im PDF/A-1- Modus muss der Wert 1 verwendet werden) Deckkraft für das Durchziehen von Linien im Bereich 0...1. Der Wert 0 bedeutet vollständig transparent; 1 bedeutet völlig undurchsichtig.*
overprintfill	*(Boolean) Overprint (Überdrucken) für Fülloperationen. Standardwert:* `false`
overprintmode	*(Integer) Overprint-Modus, der das Verhalten beim Überdrucken von Elementen in DeviceCMYK-Farben ändert, falls eine Farbkomponente den Wert 0 hat. Der Overprint-Modus wirkt sich auf Text- und Vektorelemente in DeviceCMYK-Farben aus, jedoch nicht auf Rasterbilder oder Farbverlaufsmuster. Der Modus 0 bedeutet, dass jede Farbkomponente vorhandene Markierungen ersetzt (»Vordergrundfarbe setzt sich durch«). Der Modus 1 bedeutet, dass die Farbkomponente 0 die entsprechende Komponente unverändert lässt (in Acrobat Distiller »Überdruckenstandard ist nicht Null« genannt). PDF/A-2/3:* `overprintmode=1` *ist nicht erlaubt, wenn der aktuelle Farbraum ein ICC-basierter CMYK-Farbraum ist und* `overprintfill` *oder* `overprintstroke` *gleich* `true` *ist. Standardwert: 0*
overprintstroke	*(Boolean) Overprint (Überdrucken) für Zeichen-Operationen. Standardwert:* `false`
renderingintent	*(Schlüsselwort) Beim Gamut-Mapping verwendeter Rendering-Intent; mögliche Schlüsselwörter:* `Auto`, `AbsoluteColorimetric`, `RelativeColorimetric`, `Saturation`, `Perceptual`
smoothness	*(Float) Maximaler Fehler einer linearen Interpolation für einen Farbverlauf; muss >= 0 und <= 1 sein*

Tabelle 7.3 Optionen für PDF_create_gstate( )

Option	Beschreibung
softmask	*(Optionsliste oder Schlüsselwort) Aktuelle Transparenzmaske mit Maskenform oder Deckkraft für die transparente Darstellung. Unterstützte Optionen und Schlüsselwörter (Standardwert:* none*)*:
	backdropcolor *(Liste mit einem, drei oder vier Floats; nur für* `type=luminosity`*) Farbe, die als Hintergrund bei der Zusammensetzung des Templates für die Transparenzgruppe verwendet wird. Die Anzahl der Float-Werte hängt von der Unteroption* `colorspace` *der Option* `transparencygroup` *ab, die zur Erzeugung des Templates für die Transparenzgruppe verwendet wurde (1 für DeviceGray, 3 für DeviceRGB, 4 für DeviceCMYK). Standardwert: Schwarz im jeweiligen Farbraum*
	none *(Schlüsselwort) Keine Transparenzmaske; dies ist erforderlich, um Transparenzmasken zu deaktivieren, die noch von einem bereits gesetzten Grafikzustand aktiv sind.*
	template *(Template-Handle; erforderlich) Template für Transparenzgruppen, das mit PDF_begin_template_ext( ) und der Option* `transparencygroup` *erzeugt wurde. Bei* `type=luminosity` *muss das Template mit der Unteroption* `colorspace` *und einem Wert ungleich* none *erzeugt worden sein.*
	type *(Schlüsselwort; erforderlich) Methode, um Maskenwerte aus dem Template für Transparenzgruppen abzuleiten:*
	alpha *Verwendet den Alpha-Wert der Transparenzgruppe und ignoriert die Farbe.*
	luminosity *Konvertiert die Farbe der Transparenzgruppe in einen einzelnen Luminanzwert.*
strokeadjust	*(Boolean) Automatische Anpassung der Strichstärke. Standardwert:* `false`
textknockout	*(Boolean) Beim Überblenden werden die Zeichen in einem Textobjekt als getrennte Objekte* (`false`) *oder als ein einziges Objekt* (`true`) *behandelt. Standardwert:* `true`

C++ Java C# **void setdash(double b, double w)**
Perl PHP **setdash(float b, float w)**
C **void PDF_setdash(PDF *p, double b, double w)**

Veraltet, verwenden Sie stattdessen *PDF_set_graphics_option( )*.

C++ Java C# **void setdashpattern(String optlist)**
Perl PHP **setdashpattern(string optlist)**
C **void PDF_setdashpattern(PDF *p, const char *optlist)**

Veraltet, verwenden Sie stattdessen *PDF_set_graphics_option( )*.

C++ Java C# **void setflat(double flatness)**
Perl PHP **setflat(float flatness)**
C **void PDF_setflat(PDF *p, double flatness)**

Veraltet, verwenden Sie stattdessen *PDF_set_graphics_option( )*.

C++ Java C# **void setlinejoin(int linejoin)**
Perl PHP **setlinejoin(int linejoin)**
C **void PDF_setlinejoin(PDF *p, int linejoin)**

Veraltet, verwenden Sie stattdessen *PDF_set_graphics_option( )*.

C++ Java C# **void setlinecap(int linecap)**
Perl PHP **setlinecap(int linecap)**
C **void PDF_setlinecap(PDF *p, int linecap)**

Veraltet, verwenden Sie stattdessen *PDF_set_graphics_option( )*.

C++ Java C# **void setmiterlimit(double miter)**
Perl PHP **setmiterlimit(float miter)**
C **void PDF_setmiterlimit(PDF *p, double miter)**

Veraltet, verwenden Sie stattdessen *PDF_set_graphics_option( )*.

C++ Java C# **void initgraphics( )**
Perl PHP **initgraphics( )**
C **void PDF_initgraphics(PDF *p)**

Veraltet, verwenden Sie stattdessen *PDF_set_graphics_option( )*.

7.3 Transformation des Koordinatensystems

Alle Transformationsfunktionen *(PDF_translate( ), PDF_scale( ), PDF_rotate( ), PDF_align( ), PDF_skew( ), PDF_concat( ), PDF_setmatrix( )* und die Option *initgraphicsstate* von *PDF_set_graphics_option( ))* ändern das Koordinatensystem, das beim Zeichnen nachfolgender Objekte verwendet wird. Sie haben keinerlei Einfluss auf bereits auf der Seite vorhandene Objekte.

C++ Java C# ***void translate(double tx, double ty)***
Perl PHP ***translate(float tx, float ty)***
C ***void PDF_translate(PDF *p, double tx, double ty)***

Verschiebt den Ursprung des Koordinatensystems.

tx, ty Der neue Ursprung des Koordinatensystems liegt im Punkt *(tx, ty)*, wobei sich dessen Werte auf das alte Koordinatensystem beziehen.

Gültigkeit *page, pattern, template, glyph*

C++ Java C# ***void scale(double sx, double sy)***
Perl PHP ***scale(float sx, float sy)***
C ***void PDF_scale(PDF *p, double sx, double sy)***

Skaliert das Koordinatensystem.

sx, sy Skalierungsfaktor in *x*- und *y*-Richtung.

Details Diese Funktion skaliert das Koordinatensystem um die Faktoren *sx* und *sy*. Ein negativer Skalierungsfaktor bewirkt eine Spiegelung. Eine Einheit in *x*-Richtung im neuen Koordinatensystem entspricht *sx* Einheiten in *x*-Richtung im alten Koordinatensystem; Entsprechendes gilt für die *y*-Koordinaten.

Gültigkeit *page, pattern, template, glyph*

Bindungen COM: Um einen Fehler in VB zu umgehen, ist diese Funktion auch unter dem Namen *pscale* verfügbar.

C++ Java C# ***void rotate(double phi)***
Perl PHP ***rotate(float phi)***
C ***void PDF_rotate(PDF *p, double phi)***

Dreht das Koordinatensystem.

phi Rotationswinkel in Grad.

Details Die Winkelmessung beginnt auf der positiven *x*-Achse des aktuellen Koordinatensystems und erfolgt gegen den Uhrzeigersinn. Die neuen Koordinatenachsen ergeben sich aus einer Drehung der alten Koordinatenachsen um *phi* Grad.

Gültigkeit *page, pattern, template, glyph*

C++ Java C# **void align(double dx, double dy)**

Perl PHP **align(float dx, float dy)**

C **void PDF_align(PDF *p, double dx, double dy)**

Richtet das Koordinatensystem mit einem relativen Vektor aus.

dx, dy Die Koordinaten eines Richtungsvektors *dx* und *dy* dürfen nicht beide gleich 0 sein.

Details Dreht das Koordinatensystem so, dass die *x*-Achse des neuen Koordinatensystems mit dem Vektor *(dx, dy)* und die *y*-Achse mit dem Vektor *(-dy, dx)* ausgerichtet wird. Dies ist äquivalent zu *PDF_rotate( )* mit *phi=180° / pi * atan2(dy/dx)*.

Gültigkeit *page, pattern, template, glyph*

C++ Java C# **void skew(double alpha, double beta)**

Perl PHP **skew(float alpha, float beta)**

C **void PDF_skew(PDF *p, double alpha, double beta)**

Schert das Koordinatensystem.

alpha, beta Scherungswinkel in *x*- und *y*-Richtung in Grad.

Details Bei der Scherung wird das Koordinatensystem in *x*- und *y*-Richtung um die angegebenen Winkel geneigt. *alpha* wird dabei von der positiven *x*-Achse des aktuellen Koordinatensystems ausgehend gegen den Uhrzeigersinn gemessen und *beta* von der positiven *y*-Achse ausgehend im Uhrzeigersinn. Beide Winkel müssen im Bereich -360° < *alpha, beta* < 360° liegen und dürfen nicht -270°, -90°, 90° oder 270° sein.

Gültigkeit *page, pattern, template, glyph*

C++ Java C# **void concat(double a, double b, double c, double d, double e, double f)**

Perl PHP **concat(float a, float b, float c, float d, float e, float f)**

C **void PDF_concat(PDF *p, double a, double b, double c, double d, double e, double f)**

Wendet eine Transformationsmatrix auf das aktuelle Koordinatensystem an.

a, b, c, d, e, f Elemente einer Transformationsmatrix. Die sechs Werte bilden eine Matrix wie in PostScript oder PDF (siehe die entsprechenden Referenzen). Um entartete Transformationen zu vermeiden, darf *a*d* nicht gleich *b*c* sein.

Details Diese Funktion erlaubt die allgemeinste Art von Transformationen. Wenn Sie mit Transformationsmatrizen keine Erfahrung haben, sollten Sie stattdessen *PDF_translate( ), PDF_scale( ), PDF_rotate( )* und *PDF_skew( )* verwenden. Am Anfang jeder Seite wird das Koordinatensystem auf das aktuelle Koordinatensystem zurückgesetzt (das heißt, die aktuelle Transformationsmatrix entspricht der Einheitsmatrix *[1, 0, 0, 1, 0, 0]*) .

Gültigkeit *page, pattern, template, glyph*

C++ Java C# ***void setmatrix(double a, double b, double c, double d, double e, double f)***

Perl PHP ***setmatrix(float a, float b, float c, float d, float e, float f)***

C ***void PDF_setmatrix(PDF *p, double a, double b, double c, double d, double e, double f)***

Setzt die aktuelle Transformationsmatrix explizit.

a, b, c, d, e, f Siehe *PDF_concat( )*.

Details Diese Funktion entspricht im Wesentlichen *PDF_concat( )* mit dem Unterschied, dass sie die aktuelle Transformationsmatrix verwirft und durch eine neue Matrix ersetzt.

Gültigkeit *page, pattern, template, glyph*

7.4 Pfadkonstruktion

Hinweis *Achten Sie darauf, immer eine der Funktionen aus Abschnitt 7.5, »Zeichnen und Beschneiden von Pfaden«, Seite 161, aufzurufen, nachdem Sie die Funktionen im vorliegenden Abschnitt verwendet haben. Sonst hat der konstruierte Pfad keinerlei Auswirkungen und nachfolgende Operationen führen unter Umständen zu einer Exception.*

PDF/UA Vektorgrafiken müssen bei einem Aufruf von *PDF_begin_item( )* als *Artifact* oder *Figure* ausgezeichnet werden.

C++ Java C# ***void moveto(double x, double y)***
Perl PHP ***moveto(float x, float y)***
C ***void PDF_moveto(PDF *p, double x, double y)***

Setzt den aktuellen Punkt zur Grafikausgabe.

x, y Koordinaten des neuen aktuellen Punkts.

Details Am Anfang jeder Seite wird der aktuelle Punkt auf den Standardwert *undefined* gesetzt. Die aktuelle Grafikposition sowie die aktuelle Textposition werden getrennt verwaltet.

Gültigkeit *page, pattern, template, path, glyph;* mit dieser Funktion beginnt der Gültigkeitsbereich *path*.

C++ Java C# ***void lineto(double x, double y)***
Perl PHP ***lineto(float x, float y)***
C ***void PDF_lineto(PDF *p, double x, double y)***

Zeichnet eine Linie zwischen dem aktuellen und einem anderen Punkt.

x, y Koordinaten des zweiten Linienendpunkts.

Details Diese Funktion ergänzt den aktuellen Pfad um eine Linie zwischen dem aktuellen Punkt und *(x, y)*. Der aktuelle Punkt muss zuvor gesetzt worden sein. Der Punkt *(x, y)* wird zum neuen aktuellen Punkt.

Die Linie wird um die »ideale« Linie herum zentriert, das heißt, dass auf jeder Seite der die beiden Endpunkte verbindenden Linie die halbe Strichstärke gezeichnet wird. Die Strichstärke wird dabei durch den Wert der Option *linewidth* bestimmt. Das Verhalten an den Endpunkten hängt von der Einstellung der Option *linecap* ab.

Gültigkeit *path*

C++ Java C# ***void curveto(double x1, double y1, double x2, double y2, double x3, double y3)***
Perl PHP ***curveto(float x1, float y1, float x2, float y2, float x3, float y3)***
C ***void PDF_curveto(PDF *p, double x1, double y1, double x2, double y2, double x3, double y3)***

Zeichnet eine Bézier-Kurve ausgehend vom aktuellen Punkt, wobei drei Kontrollpunkte benutzt werden.

x1, y1, x2, y2, x3, y3 Koordinaten der drei Kontrollpunkte.

Details Der aktuelle Pfad wird um eine Bézier-Kurve zwischen dem aktuellen Punkt und *(x3, y3)* ergänzt, wobei *(x1, y1)* und *(x2, y2)* als Kontrollpunkte dienen. Der aktuelle Punkt muss zuvor gesetzt worden sein. Der Endpunkt der Kurve wird zum neuen aktuellen Punkt.

Gültigkeit *path*

C++ Java C# ***void circle(double x, double y, double r)***

Perl PHP ***circle(float x, float y, float r)***

C ***void PDF_circle(PDF *p, double x, double y, double r)***

Zeichnet einen Kreis.

x, y Koordinaten des Kreismittelpunkts.

r Kreisradius.

Details Diese Funktion ergänzt den aktuellen Pfad um einen Kreis, der einen vollständigen Teilpfad darstellt. Der Punkt *(x + r, y)* wird zum neuen aktuellen Punkt. Die resultierende Figur ist in Benutzerkoordinaten kreisförmig. Wurde das Koordinatensystem in *x*- und *y*-Richtung unterschiedlich skaliert, ist die resultierende Kurve eine Ellipse. Der Kreis wird gegen den Uhrzeigersinn gezeichnet.

Gültigkeit *page, pattern, template, glyph, path;* mit dieser Funktion beginnt der Gültigkeitsbereich *path*.

Bindungen COM: um einen Fehler in VB 6 zu umgehen, ist diese Funktion auch unter dem Namen *pcircle* verfügbar.

C++ Java C# ***void arc(double x, double y, double r, double alpha, double beta)***

Perl PHP ***arc(float x, float y, float r, float alpha, float beta)***

C ***void PDF_arc(PDF *p, double x, double y, double r, double alpha, double beta)***

Zeichnet ein Kreissegment gegen den Uhrzeigersinn.

x, y Koordinaten des Mittelpunkts des Kreissegments.

r Radius des Kreissegments. *r* darf nicht negativ sein.

alpha, beta Anfangs- und Endwinkel des Kreissegments in Grad.

Details Diese Funktion ergänzt den aktuellen Pfad um ein gegen den Uhrzeigersinn gezeichnetes Kreissegment, das bei *alpha* Grad beginnt und bei *beta* Grad endet. Sowohl bei *PDF_arc( )* als auch bei *PDF_arcn( )* werden die Winkel gegen den Uhrzeigersinn gemessen, und zwar ausgehend von der positiven *x*-Achse des aktuellen Koordinatensystems. Ist der aktuelle Punkt gesetzt, dann wird von dort eine weitere gerade Linie zum Startpunkt des Kreissegments gezeichnet. Der Endpunkt des Kreissegments wird zum neuen aktuellen Punkt.

Das Kreissegment ist in Benutzerkoordinaten kreisförmig. Wurde das Koordinatensystem in *x*- und *y*-Richtung unterschiedlich skaliert, hat die resultierende Kurve eine elliptische Form.

Gültigkeit *page, pattern, template, glyph, path;* mit dieser Funktion beginnt der Gültigkeitsbereich *path*.

C++ Java C# ***void arcn(double x, double y, double r, double alpha, double beta)***

Perl PHP ***arcn(float x, float y, float r, float alpha, float beta)***

C ***void PDF_arcn(PDF *p, double x, double y, double r, double alpha, double beta)***

Zeichnet ein Kreissegment im Uhrzeigersinn.

Details Mit Ausnahme der Zeichenrichtung verhält sich diese Funktion genau wie *PDF_arc( )*. Beachten Sie, dass dies auch impliziert, dass die Winkel ausgehend von der positiven *x*-Achse gegen den Uhrzeigersinn gemessen werden.

C++ Java C# ***void circular_arc(double x1, double y1, double x2, double y2)***

Perl PHP ***circular_arc(float x1, float y1, float x2, float y2)***

C ***void PDF_circular_arc(PDF *p, double x1, double y1, double x2, double y2)***

Zeichnet ein durch drei Punkte definiertes Kreissegment.

x1, y1 Koordinaten eines beliebigen Punkts auf dem Kreissegment.

x2, y2 Koordinaten des Endpunkts des Kreissegments.

Details Diese Funktion ergänzt den aktuellen Pfad um ein Kreissegment. Das Kreissegment beginnt am aktuellen Punkt, verläuft durch *(x1, y1)* und endet bei *(x2, y2)*. Der aktuelle Punkt muss zuvor gesetzt worden sein. Der Endpunkt der Kurve wird zum neuen aktuellen Punkt.

Das Kreissegment ist in Benutzerkoordinaten kreisförmig. Wurde das Koordinatensystem in *x*- und *y*-Richtung unterschiedlich skaliert, ist die resultierende Kurve eine Ellipse.

Scope *path*

C++ Java C# ***void ellipse(double x, double y, double rx, double ry)***

Perl PHP ***ellipse(float x, float y, double rx, double ry)***

C ***void PDF_ellipse(PDF *p, double x, double y, double rx, double ry)***

Zeichnet eine Ellipse.

x, y Koordinaten des Mittelpunkts der Ellipse.

rx, ry *x*- und *y*-Radien der Ellipse.

Details Diese Funktion ergänzt den aktuellen Pfad um eine Ellipse, die einen vollständigen Teilpfad darstellt. Der Punkt *(x + rx, y)* wird zum neuen aktuellen Punkt. Die Ellipse wird gegen den Uhrzeigersinn gezeichnet.

Gültigkeit *page, pattern, template, glyph, path;* mit dieser Funktion beginnt der Gültigkeitsbereich *path*.

C++ Java C# ***void elliptical_arc(double x, double y, double rx, double ry, String optlist)***

Perl PHP ***elliptical_arc(float x, float y, double rx, double ry, string optlist)***

C ***void PDF_elliptical_arc(PDF *p, double x, double y, double rx, double ry, const char *optlist)***

Zeichnet ein elliptisches Kreissegment ausgehend vom der aktuellen Punkt.

x, y Koordinaten des Endpunkts des elliptischen Kreissegments.

rx, ry *x*- und *y*-Radien der Ellipse. Mindestens einer dieser Werte muss größer als der halbe Abstand zwischen dem aktuellen Punkt und *(x, y)* sein.

optlist Optionsliste mit Optionen zur Konstruktion von elliptischen Kreissegmenten gemäß Tabelle 7.4.

Details Diese Funktion ergänzt den aktuellen Pfad um ein elliptisches Kreissegment. Das Kreissegment beginnt am aktuellen Punkt und endet bei *(x, y)*. Der aktuelle Punkt muss zuvor nicht gesetzt worden sein. Der Endpunkt des Kreissegments wird zum neuen aktuellen Punkt. Zwei der vier möglichen Kreissegmente sind ≤ 180° (die kleinen Kreissegmente), die anderen beiden Kreissegmente sind ≥ 180° (die großen Kreissegmente).

Gültigkeit *page, pattern, template, glyph, path;* mit dieser Funktion beginnt der Gültigkeitsbereich *path*.

Tabelle 7.4 Optionen für PDF_elliptical_arc( )

Option	Beschreibung
clockwise	*(Boolean) Bei* `true` *wird ein Kreissegment im Uhrzeigersinn erzeugt; anderenfalls wird ein gegen den Uhrzeigersinn gezeichnetes Kreissegment erzeugt. Standardwert:* `false`
largearc	*(Boolean) Bei* `true` *wird ein großes Kreissegment erzeugt; anderenfalls wird eine kleines Kreissegment erzeugt. Standardwert:* `false`
rectify	*(Boolean) Bei* `true` *werden zu kleine Radien soweit vergrößert, dass ein elliptisches Kreissegment konstruiert werden kann; anderenfalls wird eine Exception ausgelöst. Standardwert:* `false`
xrotate	*(Float) Drehwinkel für die Ellipse, d.h., der Winkel der x-Achse der Ellipse relativ zum aktuellen Koordinatensystem der x-Achse in Grad. Start- und Endpunkt des Kreissegments bleiben dabei unverändert. Standardwert: 0*

C++ Java C# ***void rect(double x, double y, double width, double height)***

Perl PHP ***rect(float x, float y, float width, float height)***

C ***void PDF_rect(PDF *p, double x, double y, double width, double height)***

Zeichnet ein Rechteck.

x, y Koordinaten der linken unteren Ecke des Rechtecks.

width, height Breite und Höhe des Rechtecks.

Details Diese Funktion ergänzt den aktuellen Pfad um ein Rechteck, das einen vollständigen Teilpfad darstellt. Der aktuelle Punkt muss zuvor nicht gesetzt worden sein. Der Punkt *(x, y)* wird zum neuen aktuellen Punkt. Die Linie wird um die »ideale« Linie herum zentriert, das heißt, dass auf jeder Seite der die beiden Endpunkte verbindenden Linie die

halbe Strichstärke gezeichnet wird. Die Strichstärke wird durch den Wert der Option *linewidth* bestimmt. Das Rechteck wird gegen den Uhrzeigersinn gezeichnet.

Details

Gültigkeit *page, pattern, template, glyph, path;* mit dieser Funktion beginnt der Gültigkeitsbereich *path*.

C++ Java C# ***void closepath( )***

Perl PHP ***closepath( )***

C ***void PDF_closepath(PDF *p)***

Schließt den aktuellen Pfad.

Details Diese Funktion schließt den aktuellen Teilpfad, das heißt, sie zeichnet eine Linie zwischen dem aktuellen Punkt und dem Startpunkt des Teilpfads.

Gültigkeit *path*

7.5 Zeichnen und Beschneiden von Pfaden

Hinweis *Die meisten in diesem Abschnitt beschriebenen Funktionen leeren den Pfad und lassen den aktuellen Punkt undefiniert. Auf diese Funktionen folgende Zeichenoperationen müssen den aktuellen Punkt daher explizit setzen (zum Beispiel mit PDF_moveto( )).*

C++ Java C# ***void stroke( )***
Perl PHP ***stroke( )***
C ***void PDF_stroke(PDF *p)***

Zeichnet den Pfad mit der aktuellen Strichstärke und Linienfarbe und leert ihn.

Gültigkeit *path;* mit dieser Funktion endet der Gültigkeitsbereich *path*.

C++ Java C# ***void closepath_stroke( )***
Perl PHP ***closepath_stroke( )***
C ***void PDF_closepath_stroke(PDF *p)***

Schließt den Pfad und zeichnet ihn.

Details Diese Funktion schließt den aktuellen Teilpfad (ergänzt um eine gerade Linie zwischen dem aktuellen Punkt und dem Startpunkt des Pfads) und zeichnet den kompletten aktuellen Pfad mit der aktuellen Strichstärke und Linienfarbe.

Gültigkeit *path;* mit dieser Funktion endet der Gültigkeitsbereich *path*.

C++ Java C# ***void fill( )***
Perl PHP ***fill( )***
C ***void PDF_fill(PDF *p)***

Füllt das Pfadinnere mit der aktuellen Füllfarbe.

Details Diese Funktion füllt das Innere des aktuellen Pfads mit der aktuellen Füllfarbe. Das Innere des Pfads wird durch einen von zwei Algorithmen bestimmt (siehe Option *fillrule)*. Offene Pfade werden vor dem Füllen geschlossen.

Gültigkeit *path;* mit dieser Funktion endet der Gültigkeitsbereich *path*.

C++ Java C# ***void fill_stroke( )***
Perl PHP ***fill_stroke( )***
C ***void PDF_fill_stroke(PDF *p)***

Zeichnet und füllt den Pfad mit der aktuellen Zeichenfarbe und der aktuellen Füllfarbe.

Gültigkeit *path;* mit dieser Funktion endet der Gültigkeitsbereich *path*.

C++ Java C# **void closepath_fill_stroke( )**

Perl PHP **closepath_fill_stroke( )**

C **void PDF_closepath_fill_stroke(PDF *p)**

Schließt, zeichnet und füllt den Pfad.

Details Diese Funktion schließt den aktuellen Teilpfad (ergänzt um eine gerade Linie zwischen dem aktuellen Punkt und dem Startpunkt des Pfads) und zeichnet und füllt den kompletten aktuellen Pfad.

Gültigkeit *path;* mit dieser Funktion endet der Gültigkeitsbereich *path*.

C++ Java C# **void clip( )**

Perl PHP **clip( )**

C **void PDF_clip(PDF *p)**

Verwendet den aktuellen Pfad als Beschneidungspfad und schließt ihn.

Details Diese Funktion verwendet die Schnittmenge des aktuellen Pfads und des aktuellen Beschneidungspfads als Beschneidungspfad für weitere Operationen. Am Anfang jeder Seite wird dieser auf den Standardwert gesetzt, nämlich die Seitengröße. Der Beschneidungspfad unterliegt der Klammerung mit *PDF_save( )*/*PDF_restore( )*. Er kann nur mittels *PDF_save( )*/*PDF_restore( )* vergrößert werden. Der Clipping-Bereich wird gemäß dem mit der Option *cliprule* ausgewählten Algorithmus bestimmt.

Gültigkeit *path;* mit dieser Funktion endet der Gültigkeitsbereich *path*.

C++ Java C# **void endpath( )**

Perl PHP **endpath( )**

C **void PDF_endpath(PDF *p)**

Beendet den aktuellen Pfad, ohne ihn zu zeichnen oder zu füllen.

Details Diese Funktion hat keine sichtbaren Auswirkungen auf die Seite. Sie generiert einen unsichtbaren Pfad auf der Seite.

Gültigkeit *path;* mit dieser Funktion endet der Gültigkeitsbereich *path*.

7.6 Pfadobjekte

C++ Java C# **int add_path_point(int path, double x, double y, String type, String optlist)**

Perl PHP **int add_path_point(int path, float x, float y, string type, string optlist)**

C **int PDF_add_path_point(PDF *p, int path, double x, double y, const char *type, const char *optlist)**

Ergänzt ein neues oder vorhandenes Pfadobjekt um eine Punkt oder einen Pfad.

path Gültiges Pfad-Handle, das von einem anderen Aufruf von PDF_add_path_point( ) oder -1 (in PHP: 0) zurückgeben wurde, um einen neuen Pfad zu erzeugen.

x, y Koordinaten des neuen aktuellen Punkts. Bei *polar=false* kennzeichnen die beiden Werte die kartesischen Koordinaten *(x, y)* des Punkts. Bei *polar=true* kennzeichnen die beiden Werte den Radius *r* und den Winkel *phi* des Punkts (in Grad oder als Radiant abhängig von der Option *radians)*. Dieser Punkt wird zum neuen aktuellen Punkt für *type=circle, circular, elliptical, ellipse, move, line, curve, rect*.

type Art des Punkts gemäß Tabelle 7.5.

Tabelle 7.5 Punktarten für PDF_add_path_point( )

Schlüsselwort	Beschreibung
addpath	*Fügt die in der Option* svgpath *angegebene Pfaddefinition als vollständigen Teilpfad hinzu, mit* (x, y) *als Ursprung.*
circle	*Ergänzt den Pfad um einen Kreis als vollständigen Teilpfad, mit* (x, y) *als Mittelpunkt und* radius *als Größe.*[1]
circular	*Fügt einen Kreis zwischen dem aktuellen Punkt und* (x, y) *ein, mit dem zuvor definierten Kontrollpunkt als erforderlichem dritten Kreispunkt. Ist der neue Punkt mit dem aktuellen Punkt identisch, wird ein Kreis mit einem Durchmesser zwischen dem aktuellen Punkt und dem Kontrollpunkt erstellt.*[2]
control	*Kontrollpunkt für eine Bézierkurve oder einen Kreisbogen.*[2]
curve	*Fügt eine Bézierkurve mit den zuvor erzeugten Kontrollpunkten zwischen dem aktuellen Punkt und dem neuen Punkt ein. Mindestens ein Kontrollpunkt muss übergeben werden. Ist nur ein Kontrollpunkt verfügbar, wird er als zweiter Kontrollpunkt für die Kurve verwendet, und der erste Kontrollpunkt wird als Spiegelung des zweiten Kontrollpunkts am Endpunkt der vorigen Bézierkurve konstruiert.*[2]
ellipse	*Ergänzt den Pfad um eine Ellipse als vollständigen Teilpfad, mit* (x, y) *als Mittelpunkt und den Werten in der Option* radius *als Größe.*[1] *Die Ellipse kann mit der Option* xrotate *gedreht werden.*
elliptical	*Fügt einen elliptischen Bogen zwischen dem aktuellen Punkt und* (x, y) *ein. Größe und Richtung der Ellipse werden durch die Optionen* radius, xrotate, largearc *und* clockwise *definiert. Wird als* radius *nur ein einziger Wert übergeben, wird ein Kreisbogen erstellt. In diesem Fall wird automatisch ein entsprechender Punkt auf dem Kreisbogen erstellt. Werden in der Option* radius *zwei Werte übergeben, wird eine Reihe von Bézierkurven erzeugt.*[2]
line	*Fügt ein Liniensegment zwischen dem aktuellen Punkt und* (x, y) *ein.*[2]
move	*Beginnt einen neuen Teilpfad. Teilpfade werden durchnummeriert (1, 2, ...). Der erste Teilpfad beginnt am Ursprung.*
pathref	*Fügt eine Referenz auf den Pfad in der Option* path *als vollständigen Teilpfad zum Pfad hinzu, mit* (x, y) *als Ursprung. Da der Pfad referenziert und nicht kopiert wird, werden nachfolgende Änderungen bei* path *beim Zeichnen des Pfades dargestellt.*

Tabelle 7.5 Punktarten für PDF_add_path_point( )

Schlüsselwort	Beschreibung
rect	*Fügt ein Rechteck als vollständigen Teilpfad zum Pfad hinzu, mit* `(x, y)` *als Mittelpunkt des Rechtecks und* `width` *und* `height` *als Größe.*[1] *Die Ecken des Rechtecks können vor dem Zeichnen des Pfads mit der Optionen* `round` *gerundet werden. Alternativ dazu können sie mit der Option* `radius` *elliptisch gerundet werden.*

1. Nach dem Pfad wird automatisch ein neuer Punkt mit `type=move` *und den selben Koordinaten und Optionen zur Grafikdarstellung erzeugt.*
2. Für diese Typen sind keine Optionen zur Grafikdarstellung und für Pfadoperationen erlaubt.

optlist Optionsliste mit Optionen zur Pfadkonstruktion:

- Optionen für Pfadberechnungen und die Benennung von Punkten gemäß Tabelle 7.6: *name, polar, radians, relative*
- Optionen für Pfadoperationen gemäß Tabelle 7.6: *close, fill, round, stroke*
- Optionen zur Ergänzung von Pfaddefinitionen gemäß Tabelle 7.6: *path, svgpath*
- Optionen zur Konstruktion von Pfadelementen gemäß Tabelle 7.6: *clockwise, height, largearc, radius, rectify, width, xrotate*
- Optionen zur Grafikdarstellung gemäß Tabelle 7.1 (nur für *type=addpath, circle, ellipse, move, rect* oder *pathref)*: *dasharray, dashphase, fillcolor, fillrule, flatness, gstate, linecap, linejoin, linewidth, miterlimit, strokecolor*

Rückgabe Pfad-Handle, das verwendet werden kann, bis es mit *PDF_delete_path( )* gelöscht wird.

Details Pfadobjekt, das als Container für Vektorgrafiken dient. Das Pfadobjekt kann schrittweise mit Pfaden und Teilpfaden befüllt werden. Neue Pfadelemente können dabei durch Angabe einzelner Pfadknoten oder durch Hinzufügen von Pfaddefinitionen erzeugt werden, die über ein Pfad-Handle oder eine SVG-Pfadbeschreibungen angegeben werden. Der generierte Pfad kann später mit *PDF_draw_path( )* und anderen Funktionen verwendet werden.

Ein Pfadobjekt kann eine beliebige Anzahl von Pfaden enthalten. Jeder Pfad kann wiederum einen oder mehrere Teilpfade enthalten, die in der Option *subpaths* von *PDF_draw_path( )* zum Zeichnen ausgewählt werden können. Alle Pfade werden gemäß der angegebenen Optionen separat geschlossen, gefüllt, gezeichnet und gerundet.

Eine Operation mit einem der Typen *addpath, circle, ellipse, move, rect* oder *pathref* beginnt einen neuen Teilpfad. Optionen zur Grafikdarstellung und für Pfadoperationen (z.B. *stroke*, *fill)* können nur für *type=addpath, circle, ellipse, move, rect* oder *pathref* geändert werden. In diesem Fall wird automatisch ein neuer Pfad innerhalb des Pfadobjekts gestartet. Objekte vom Typ *circle, ellipse, elliptical* und *rect* werden gegen den Uhrzeigersinn gezeichnet, dies lässt sich aber mit der Option *clockwise* umkehren.

Gültigkeit beliebig

Tabelle 7.6 Optionen für PDF_add_path_point( )

Option	Beschreibung
clockwise	*(Boolean; nur bei* type=circle, elliptical, rect*) Bei* true *wird das Objekt im Uhrzeigersinn gezeichnet; anderenfalls gegen den Uhrzeigersinn. Standardwert:* false
close	*(Boolean; nur bei* type=move*) Bei* true *wird der Teilpfad mit einer geraden Linie geschlossen. Standardwert: siehe Fußnote*[1]
fill	*(Boolean; nur für* type=move*) Bei* true *wird der Teilpfad geschlossen und gefüllt. Standardwert: siehe Fußnote*[1]
height	*(Float; nur bei* type=rect; *in diesem Fall erforderlich) Höhe des Rechtecks*
largearc	*(Boolean; nur bei* type=elliptical*) Bei* true *wird ein großes elliptisches Kreissegment erzeugt; anderenfalls ein kleines elliptisches Kreissegment. Standardwert:* false
name	*(String) Name des Punkts. Standardwert:* p<i> *(z.B.* p1*), wobei* i *die fortlaufende Nummer des übergebenen Punkts ist.*
path	*(Pfad-Handle; nur bei* type=pathref*) Ergänzt den aktuellen Pfad um eine Referenz auf den angegebenen Pfad. Die Koordinaten des hinzugefügten Pfads beziehen sich auf den aktuellen Punkt als Ursprung. Optionen zur Grafikdarstellung und die Option* name *werden ignoriert.*
polar	*(Boolean) Bei* true *sind die Parameter* (x, y) *Polarkoordinaten mit Angabe von Radius* r *und Winkel* phi*, anderenfalls kartesische Koordinaten mit Angabe von* x- *und* y-*Werten. Standardwert:* false
radians	*(Boolean) Bei* true *werden die Winkel für Polarkoordinaten im Bogenmaß angegeben und sonst in Grad. Standardwert:* false
radius	*(Ein oder zwei Floats; erforderlich bei* type=circle, ellipse *und* elliptical; *auch erlaubt bei* type=rect*) Der erste Wert gibt den Radius des Kreises oder den x-Radius der Ellipse an. Der zweite Float-Wert, falls vorhanden, gibt den y-Radius der Ellipse an. Der erste Wert wird als Standardwert für den zweiten Wert verwendet.* *Bei* type=rect *geben die Werte die x- und y-Radien der elliptischen Bögen in den Ecken des Rechtecks an. Die elliptischen Bögen werden sofort erzeugt. Standardwert:* 0
rectify	*(Boolean; nur bei* type=ellipse *und* elliptical*) Bei* true *werden zu kleine Radien so angepasst, dass der elliptische Kreisbogen konstruiert werden kann; anderenfalls wird eine Exception ausgelöst. Standardwert:* false
relative	*(Boolean) Bei* true *sind* (x, y) *relativ zum aktuellen Punkt und sonst relativ zum aktuellen Ursprung. Standardwert: siehe Fußnote*[1]
round	*(Float; nur bei* type=move*) Für jeden Teilpfad werden benachbarte* line*-Eckpunkte durch einen Kreisbogen mit den Liniensegmenten als Tangenten und mit dem angegebenen Radius abgerundet. Ist der Radius negativ, werden die Bogensegmente nach innen geschwungen, so dass die Ecken abgerundet sind. Bei* close=true *und wenn zwischen letztem und erstem Punkt keine explizite Linie angegeben wurde, werden die erste und die schließende Linie ebenfalls gerundet. Bei* round=0 *wird nicht gerundet. Standardwert: Wert, der beim Konstruieren des Pfades angegeben wurde, oder* 0*, wenn kein Wert angegeben wurde. Die Kreisbögen werden erzeugt, wenn der Pfad gezeichnet wird. Standardwert: siehe Fußnote*[1]
stroke	*(Boolean; nur bei* type=move*) Bei* true *wird der Teilpfad gezeichnet. Standardwert: siehe Fußnote*[1]
svgpath	*(String; nur bei* type=addpath*) String mit einer Pfadbeschreibung in SVG-Syntax gemäß* www.w3.org/TR/SVG11/paths.html#PathData*. Erweitert den aktuellen Pfad um den angegebenen Pfad. Die Koordinaten des angegebenen SVG-Pfads beziehen sich auf den aktuellen Punkt als Ursprung. Für den SVG-Pfad können Optionen zur Grafikdarstellung angegeben werden. Die Option* rectify *wird für den eingefügten SVG-Pfad berücksichtigt. Wenn der Pfad aus einer SVG-Datei mit einem Top-Down-Koordinatensystem stammt, muss er gespiegelt werden (auch wenn PDFlib im Top-Down-Modus arbeitet). Dies kann mit der Option* scale={1 -1} *von PDF_draw_path ( ) erreicht werden.*

Tabelle 7.6 Optionen für PDF_add_path_point( )

Option	Beschreibung
width	*(Float; nur bei* `type=rect`*; in diesem Fall erforderlich) Breite des Rechtecks*
xrotate	*(Float; nur bei* `type=ellipse` *und* `elliptical`*) Drehwinkel für die Ellipse in aktuellen Benutzerkoordinaten (siehe Option* `radians`*), das heißt, der Winkel der x-Achse der Ellipse relativ zum aktuellen Koordinatensystem der x-Achse in Grad. Start- und Endpunkt des Kreissegments bleiben unverändert. Diese Option wird ignoriert, wenn als* `radius` *nur ein einziger Wert übergeben wird. Standardwert: 0*

1. Der Standardwert wird in PDF_draw_path( ), PDF_info_path( ), der Option `textpath` *von PDF_fit_textline( ), der Option* `wrap` *von PDF_fit_textflow( ) oder der Option* `fitpath` *von PDF_add_table_cell( ) angegeben.*

C++ Java C# ***void draw_path(int path, double x, double y, String optlist)***

Perl PHP ***draw_path(int path, float x, float y, string optlist)***

C ***void PDF_draw_path(PDF *p, int path, double x, double y, const char *optlist)***

Zeichnet ein Pfadobjekt.

path Gültiges Pfad-Handle, das von *PDF_add_path_point( )* oder einer anderen Funktion (z.B. *PDF_info_image( )* mit dem Schlüsselwort *boundingbox*) zurückgegeben wurde.

x, y Koordinaten des Referenzpunkts in Benutzerkoordinaten. Der Referenzpunkt wird von verschiedenen Optionen verwendet und gibt die Position des Ursprungs für das Pfadobjekt in aktuellen Benutzerkoordinaten an. Dies bewirkt eine Verschiebung des Pfadobjekts.
Ist die Option *boxsize* angegeben, wird *(x, y)* zur linken unteren Ecke der Fitbox, in die das Pfadobjekt hineinpasst (siehe Tabelle 6.1).

optlist Optionsliste mit Optionen zum Zeichnen von Pfaden:

- Optionen zur Objekteinpassung gemäß Tabelle 6.1:
  align, attachmentpoint, boxsize, fitmethod, orientate, position, scale
- Optionen für Pfadoperationen und für die Auswahl von Teilpfaden gemäß Tabelle 7.7: *clip, close, fill, round, stroke, subpaths*
- Optionen für das Anpassen der Boundingbox gemäß Tabelle 7.7:
  bboxexpand, boundingbox
- Optionen zur Grafikdarstellung für die Optionen *fill* und *stroke* gemäß Tabelle 7.1:
  dasharray, dashphase, fillcolor, flatness, gstate, linecap, linejoin, linewidth, miterlimit, strokecolor
- Optionen zur Grafikdarstellung gemäß Tabelle 7.1 für die Option *clip* gemäß Tabelle 7.1: *cliprule, fillrule*
- Option zum vereinfachten Tagging von Strukturelementen gemäß Tabelle 14.5 (nur erlaubt im Gültigkeitsbereich *page*): *tag*

Details Der oder die Pfade werden am Referenzpunkt *(x, y)* platziert und dann gemäß der angegebenen Optionen gezeichnet, gefüllt oder als Beschneidungspfad verwendet. Diese Funktion ändert den aktuellen Grafikzustand nur, wenn die Option *clip* verwendet wird. Die Optionen zur Grafikdarstellung und für Pfadoperationen überschreiben die Standardeinstellungen, außer den Optionen zur Grafikdarstellung, die in *PDF_add_path_point( )* für einen Teilpfad angegeben wurden.

PDF/UA Alle Pfadobjekte müssen als *Artifact* oder *Figure* ausgezeichnet werden, entweder mit der Option *tag* oder mit einem vorausgehenden Aufruf von *PDF_begin_item( )*.

Gültigkeit *page, pattern, template, glyph*

Tabelle 7.7 Pfadoptionen für PDF_draw_path( ) zur Steuerung aller Teilpfade in einem Pfadobjekt

Option	Beschreibung
bboxexpand	*(Liste aus Floats; wird bei Angabe der Option* boundingbox *ignoriert) Ein oder zwei Floats, die eine Erweiterung der automatisch berechneten Boundingbox bewirken (das kleinste umschließende Rechteck des Pfadobjekts). Standardwert:* {0 0}
boundingbox	*(Rechteck) Rechteck im Koordinatensystem des Pfadobjekts, das als Boundingbox verwendet wird, um das Pfadobjekt in die Fitbox einzupassen. Standardwert: das kleinste umschließende Rechteck des Pfadobjekts, gegebenenfalls gemäß der Option* bboxexpand *erweitert*
clip	*(Boolean) Bei* true *wird der Pfad geschlossen und als Beschneidungspfad verwendet. Standardwert:* false
close	*(Boolean) Bei* true *wird jeder Teilpfad mit einer geraden Linie geschlossen. Standardwert: Wert, der beim Konstruieren des Pfades angegeben wurde, oder* false*, wenn kein Wert angegeben wurde*
fill	*(Boolean; überschreibt* clip*) Bei* true *wird jeder Pfad gefüllt. Standardwert: Wert, der beim Konstruieren des Pfades angegeben wurde, oder* false*, wenn kein Wert angegeben wurde*
round	*(Float) Für jeden Teilpfad werden benachbarte* line*-Eckpunkte durch einen Kreisbogen mit den Liniensegmenten als Tangenten und mit dem angegebenen Radius abgerundet. Ist der Radius negativ, werden die Bogensegmente nach innen geschwungen, so dass die Ecken abgerundet sind. Bei* close=true *und wenn zwischen letztem und erstem Punkt keine explizite Linie angegeben wurde, werden die erste und die schließende Linie ebenfalls gerundet. Bei* round=0 *wird nicht gerundet. Standardwert: Wert, der beim Konstruieren des Pfads angegeben wurde, oder* 0*, wenn kein Wert angegeben wurde*
stroke	*(Boolean; überschreibt* clip*) Bei* true *wird der Pfad gezeichnet. Standardwert:* false
subpaths	*(Liste aus Integers oder einzelnes Schlüsselwort) Liste mit den Nummern der Teilpfade, die gezeichnet werden sollen; der erste Teilpfad hat die Nummer 1. Das Schlüsselwort* all *beschreibt alle Teilpfade. Standardwert:* all

C++ Java C# ***double info_path(int path, String keyword, String optlist)***

Perl PHP ***float info_path(int path, string keyword, string optlist)***

C ***double PDF_info_path(PDF *p, int path, const char *keyword, const char *optlist)***

Fragt die Ergebnisse für das Zeichnen eines Pfadobjekts ab, ohne dieses selbst zu zeichnen.

path Gültiges Pfad-Handle, das mit *PDF_add_path_point( )* oder einer anderen Funktion (z.B. *PDF_info_image( )* mit dem Schlüsselwort *boundingbox*) zurückgegeben wurde.

keyword Schlüsselwort mit den erforderlichen Informationen:

- Schlüsselwörter zur Abfrage von Ergebnissen der Objekteinpassung gemäß Tabelle 6.3:
  boundingbox, fitscalex, fitscaley, height, objectheight, objectwidth, width, x1, y1, x2, y2, x3, y3, x4, y4
- Zusätzliche Schlüsselwörter gemäß Tabelle 7.8:
  bboxwidth, bboxheight, numpoints, px, py

optlist Optionsliste mit Optionen für das Zeichnen von Pfaden:

- Alle Optionen von *PDF_draw_path( )* gemäß Tabelle 7.7
- Zusätzliche Option zur Objekteinpassung gemäß Tabelle 6.1: *refpoint*
- Zusätzliche Option gemäß Tabelle 7.9: *name*

Tabelle 7.8 Schlüsselwörter für PDF_info_path( )

Schlüsselwort	Beschreibung
bboxwidth, bboxheight	*Breite und Höhe der Boundingbox für den Pfad*
numpoints	*Anzahl der übergebenen Punkte. Die Option* subpaths *wird ignoriert.*
px, py	x- *oder* y*-Koordinate des Pfadpunkts (im Benutzerkoordinatensystem), der in der Option* name *festgelegt wurde. Die Option* subpaths *wird ignoriert.*

Rückgabe Wert einer mit *keyword* ausgewählten Pfadeigenschaft.

Details Diese Funktion führt die gleichen Berechnungen durch wie *PDF_add_path_point( )*, erzeugt jedoch keine sichtbare Ausgabe auf der Seite.

Gültigkeit beliebig

Tabelle 7.9 Optionen für PDF_info_path( )

Option	Beschreibung
name	*Name eines Pfadpunkts für die Schlüsselwörter* px *oder* py*. Ein Standardname (z.B.* p1*) kann auch dann verwendet werden, wenn der Name explizit in PDF_add_path_point( ) angegeben wurde.*

C++ Java C# ***void delete_path(int path)***

Perl PHP ***delete_path(int path)***

C ***void PDF_delete_path(PDF *p, int path)***

Löscht ein Pfadobjekt.

path Gültiges Pfad-Handle, das von *PDF_add_path_point( )* oder einer anderen Funktion (z.B. *PDF_info_image( )* mit dem Schlüsselwort *boundingbox* zurückgegeben wird.

Details Löscht das Pfadobjekt und alle damit verbundenen internen Datenstrukturen. Beachten Sie, dass Pfadobjekte in *PDF_end_document( )* nicht automatisch gelöscht werden.

Gültigkeit beliebig

8 Farbfunktionen

8.1 Festlegen von Farbe und Farbraum

Cookbook *Ein vollständiges Codebeispiel hierzu finden Sie im Cookbook-Topic* `color/starter_color`.

Tabelle 8.1 enthält alle für diesen Abschnitt relevanten globalen Optionen (siehe Abschnitt 2.3, »Globale Optionen«, Seite 25).

Tabelle 8.1 Farbrelevante Optionen für PDF_set_option( )

Option	Beschreibung
preserveold-pantonenames	*Bei* `false` *werden die alten Namen der Pantone-Schmuckfarben in die entsprechenden neuen Farbnamen konvertiert. Bei* `true` *bleiben sie unverändert. Standardwert:* `false`
spotcolorlookup	*Bei* `false` *ermittelt PDFlib Schmuckfarbnamen nicht aus der internen Datenbank. Dies kann als Behelfslösung sinnvoll sein, damit die übergebenen kundenspezifischen Definitionen bekannter Schmuckfarbnamen mit den von anderen Anwendungen verwendeten Definitionen übereinstimmen. Diese Funktionalität sollte nach Möglichkeit nicht oder nur mit Vorsicht verwendet werden. Standardwert:* `true`

C++ Java C# ***void setcolor(String fstype, String colorspace, double c1, double c2, double c3, double c4)***

Perl PHP ***setcolor(string fstype, string colorspace, float c1, float c2, float c3, float c4)***

C ***void PDF_setcolor(PDF *p,***
const char *fstype, const char *colorspace, double c1, double c2, double c3, double c4)

Setzt den aktuellen Farbraum und die aktuelle Farbe für den Grafik- und Textzustand.

fstype Einer der Werte *fill, stroke* oder *fillstroke*, die festlegen, ob die Farbe zum Zeichnen, Füllen oder für beides eingestellt wird.

colorspace Festlegung des Farbraums, der für die übergebenen Farbwerte oder einen RGB-Farbwert verwendet werden soll, durch Angabe eines Namens oder eines hexadezimalen Werts:

- Erste Form: Festlegung des Farbraums durch einen der Werte *gray, rgb, cmyk, spot, pattern, iccbasedgray, iccbasedrgb, iccbasedcmyk* oder *lab*.
- Zweite Form: RGB-Farbname (z.B. *pink)* oder ein Hash-Zeichen gefolgt von sechs hexadezimalen Ziffern (z.B. *#FFCoCB)*.

c1, c2, c3, c4 Farbkomponenten für den ausgewählten Farbraum. Die Interpretation dieser Werte hängt vom Parameter *colorspace* ab:

- *gray: c1* legt einen Graustufenwert fest.
- *rgb: c1, c2, c3* legen die Werte für Rot, Grün und Blau fest.
- *cmyk:c1, c2, c3, c4* legen die Werte für Cyan, Magenta, Gelb und Schwarz fest.
- *iccbasedgray: c1* legt einen Graustufenwert fest.
- *iccbasedrgb: c1, c2, c3* legen die Werte für Rot, Grün und Blau fest.
- *iccbasedcmyk: c1, c2, c3, c4* legen die Werte für Cyan, Magenta, Gelb und Schwarz fest.
- *spot: c1* enthält ein Schmuckfarben-Handle, das von *PDF_makespotcolor( )* erzeugt wurde, und *c2* legt den Farbauftrag mit einem Wert zwischen 0 und 1 fest.

- *lab: c1, c2* und *c3* legen Farbwerte im Farbraum CIE L*a*b* fest, wobei von einer Lichtquelle vom Typ D50 ausgegangen wird. *c1* definiert die Helligkeit L* *(luminance)* als Wert zwischen 0 und 100; *c2* und *c3* legen a* und b* (Buntwerte) als Werte zwischen -128 und 127 fest.
- *pattern: c1* enthält das Handle für ein Füllmuster, das von *PDF_begin_pattern_ext( )* oder *PDF_shading_pattern( )* zurückgegeben wurde. Wird das Füllmuster zum Füllen oder Zeichnen eingestellt, wird die aktuelle Füll- bzw. Zeichenfarbe darauf angewendet. Der aktuelle Farbraum darf kein anderer Pattern-Farbraum sein.

Details Alle Farbwerte für die Farbräume *gray, rgb* und *cmyk* sowie für den Farbauftrag des Farbraums *spot* müssen Zahlen im Bereich 0–1 sein. Nicht verwendete Parameter sollten auf 0 gesetzt werden. Für weitere Informationen zu Farbräumen und Farbwerten siehe »Farboptionen«, Seite 14.

Am Anfang jeder Seite werden die Zeichen- und Füllfarbenwerte für die Farbräume *gray, rgb* und *cmyk* auf den Standardwert schwarz gesetzt. Für Schmuck- und Füllmusterfarbe gibt es keine Standardwerte.

Beim Einsatz der Farbräume *iccbasedgray/rgb/cmyk* muss das zugehörige ICC-Profil gesetzt worden sein, bevor die Optionen *iccprofilegray/rgb/cmyk* verwendet werden können.

Diese Funktion ist äquivalent zu *PDF_set_text_option( )* und *PDF_set_graphics_option( )* mit den Optionen *fillcolor* und/oder *strokecolor*. *PDF_setcolor( )* überschreibt die Werte dieser Optionen.

PDF/A *colorspace=gray* erfordert eine Druckausgabebedingung (beliebigen Typs) oder die Option *defaultgray* in *PDF_begin_page_ext( )*.

colorspace=rgb erfordert eine RGB- Druckausgabebedingung oder die Option *defaultrgb* in *PDF_begin_page_ext( )*.

colorspace=cmyk erfordert eine CMYK-Druckausgabebedingung oder die Option *defaultcmyk* in *PDF_begin_page_ext( )*.

PDF/X PDF/X-1a: *colorspace=rgb*, *iccbasedgray/rgb/cmyk* und *lab* sind nicht zulässig.

PDF/X-3: Bei *iccbasedgray/rgb/cmyk* und *lab* muss in der Ausgabebedingung ein ICC-Profil angegeben sein (ein Standardname reicht in diesem Fall nicht aus).

PDF/X-3/4/5: *colorspace=gray* erwartet eine Graustufen- oder CMYK-Druckausgabebedingung oder die Option *defaultgray* in *PDF_begin_page_ext( )*.

colorspace=rgb erwartet eine RGB-Druckausgabebedingung oder die Option *defaultrgb* in *PDF_begin_page_ext( )*.

colorspace=cmyk erwartet eine CMYK-Druckausgabebedingung oder die Option *defaultcmyk* in *PDF_begin_page_ext( )*.

PDF/UA Informationen sollten nicht allein durch Farbe oder Kontrast übermittelt werden.

Gültigkeit *page, pattern* (nur falls *paintype=colored), template, glyph* (nur wenn die Option *colorized* des Type-3-Fonts gleich *true* ist), *document;* eine Füllmusterfarbe kann nicht innerhalb ihrer eigenen Definition verwendet werden. Die Farbe im Gültigkeitsbereich *document* einzustellen, kann zur Definition von Schmuckfarben mit *PDF_makespotcolor( )* sinnvoll sein.

C++ Java C# ***int makespotcolor(String spotname)***

Perl PHP ***int makespotcolor(String spotname)***

C ***int PDF_makespotcolor(PDF *p, const char *spotname, int reserved)***

Wählt eine integrierte Schmuckfarbe über den Namen aus oder erzeugt aus der aktuellen Füllfarbe eine Schmuckfarbe.

spotname Name einer integrierten Schmuckfarbe oder beliebiger Name für die zu definierende Schmuckfarbe. Dieser Name ist auf eine Länge von maximal 126 Byte beschränkt.

Der spezielle Schmuckfarbenname *All* kann verwendet werden, um eine Farbe auf alle Farbauszüge zu bringen. Dies kann zum Beispiel für Schnittmarken sinnvoll sein. Der Schmuckfarbenname *None* produziert Ausgabe, die auf keinem Auszug sichtbar ist.

reserved (Nur C-Sprachbindung) Reserviert, muss gleich 0 sein.

Rückgabe Ein Farben-Handle, das in nachfolgenden Aufrufen von *PDF_setcolor( )* oder den Optionen *fillcolor/strokecolor* von *PDF_set_text_option(), PDF_set_graphics_option()* und anderen Funktionen im ganzen Dokument verwendet werden kann. Schmuckfarben-Handles können auf allen Seiten, aber nicht in anderen Dokumenten verwendet werden. Die Anzahl der Schmuckfarben in einem Dokument ist nicht begrenzt.

Details Falls *spotname* in den internen Farbtabellen von Pantone- und HKS-Farben gefunden wurde und die Option *spotcolorlookup* gleich *true* ist (Standardwert), werden dieser Name und die zugehörigen internen Alternativwerte verwendet. Anderenfalls werden die Farbwerte (CMYK oder andere) der aktuellen Füllfarbe zum Erstellen einer neuen Schmuckfarbe benutzt. Diese Alternativwerte werden nur zur Bildschirmanzeige oder für den Low-End-Druck verwendet. Zur Erstellung von Farbseparationen wird statt der Alternativwerte der übergebene Schmuckfarbenname verwendet.

Wurde *spotname* bereits in einem früheren Aufruf von *PDF_makespotcolor( )* verwendet, ist der Rückgabewert wieder derselbe und gibt somit nicht die aktuelle Farbe wieder.

PDF/X PANTONE®-Farben werden im PDF/X-1a-Modus nicht unterstützt.

Gültigkeit *page, pattern, template, glyph* (nur wenn die Option *colorized* des Type-3-Fonts gleich *true* ist), *document;* die aktuelle Füllfarbe darf keine Schmuckfarbe und kein Füllmuster sein, falls eine benutzerdefinierte Schmuckfarbe angelegt wird.

8.2 ICC-Profile

C++ Java C# ***int load_iccprofile(String profilename, String optlist)***

Perl PHP ***int load_iccprofile(string profilename, string optlist)***

C ***int PDF_load_iccprofile(PDF *p, const char *profilename, int len, const char *optlist)***

Sucht nach einem ICC-Profil und bereitet es zur späteren Verwendung vor.

profilename (Name-String) Name einer *ICCProfile*-Ressource oder Name einer Festplatten- oder virtuellen Datei.

len (Nur C-Sprachbindung) Länge von *profilename* (in Bytes). Ist *len* = 0, muss ein nullterminierter String übergeben werden.

optlist Optionsliste zur Steuerung der Profilverarbeitung:

- Allgemeine Option: *errorpolicy* (siehe Tabelle 2.1)
- Optionen zur Profilverarbeitung gemäß Tabelle 8.2: *description, embedprofile, metadata, urls, usage*

Tabelle 8.2 Optionen für PDF_load_iccprofile( )

Option	Beschreibung
description	*(String; nur bei* `usage=outputintent` *und Nicht-Standard-Ausgabebedingungen) Klartextbeschreibung des ICC-Profils, die gemeinsam mit der Druckausgabebedingung verwendet wird.*
embedprofile	*(Boolean; nur bei PDF/X-1a/3 und* `usage=outputintent`*; wird bei PDF/X-4 und PDF/X-5g auf* `true` *gesetzt) Bettet ein ICC-Profil selbst dann ein, wenn als* `profilename` *eine Standard-Druckausgabebedingung für PDF/X-1a/3 übergeben wurde. Standardwert:* `false`
metadata	*(Optionsliste; wird bei* `usage=outputintent` *in PDF/X-4p und PDF/X-5pg ignoriert) Liefert Metadaten zum Profil (siehe Abschnitt 14.2, »XMP-Metadaten«, Seite 277)*
urls	*(Liste von einem oder mehreren Strings; nur erforderlich bei PDF/X-4p und PDF/X-5pg) Liste von URLs, die angibt, wo eine referenzierte ICC-basierte Druckausgabebedingung erhalten werden kann. Sender und Empfänger sollten sinnvolle URLs einrichten. Die Strings sind frei wählbar, müssen jedoch gültiger URL-Syntax entsprechen.*
usage	*(Schlüsselwort) Beschreibt, wie das ICC-Profil verwendet werden soll. Unterstützte Schlüsselwörter (Standardwert:* `iccbased`*):* ***iccbased*** *Das ICC-Profil wird als ICC-basierter Farbraum für Text oder Grafik verwendet oder auf ein Bild angewandt, als Standard-Farbraum oder als Mischfarbraum für eine Transparenzgruppe verwendet.* ***outputintent*** *Das ICC-Profil wird zur Definition einer Druckausgabebedingung für PDF/A oder PDF/X verwendet.*

Rückgabe Profil-Handle, das in nachfolgenden Aufrufen von *PDF_load_image( )* oder mit profilspezifischen Optionen verwendet werden kann. Bei *errorpolicy=return* muss der Aufruf auf den Rückgabewert -1 (in PHP: 0) überprüft werden, der einen Fehler anzeigt. Das zurückgegebene Profil-Handle kann nicht über mehrere PDF-Dokumente hinweg eingesetzt werden. Es gibt keine Obergrenze für die Anzahl der in einem Dokument enthaltenen ICC-Profile. Scheitert der Funktionsaufruf, so können Sie die Fehlerursache mit *PDF_get_errmsg( )* abfragen.

Details Bei *usage=iccbased* wird das benannte Profil gemäß der Suchstrategie ermittelt. Je nach Verwendungszweck müssen ICC-Profile die im PDFlib-Tutorial aufgeführten Bedingungen erfüllen. Das *sRGB*-Profil ist in jedem Fall intern verfügbar und muss nicht gesondert definiert werden.

PDF/A Die Ausgabebedingung für das generierte Dokument kann mit dieser Funktion gesetzt werden oder durch Kopieren der Ausgabebedingung eines importierten Dokuments mit *PDF_process_pdi( )*. Werden im Dokument nur geräteunabhängige Farben verwendet, so ist keine Druckausgabebedingung erforderlich.

PDF/X Die Ausgabebedingung für das generierte Dokument muss entweder mit dieser Funktion gesetzt werden oder durch Kopieren der Ausgabebedingung eines importierten Dokuments mit *PDF_process_pdi( )*.

PDF/X-1/3: Einer der folgenden Standardnamen für Druckausgabebedingungen kann ohne Einbettung des entsprechenden ICC-Profils verwendet werden:

```
CGATS TR 001, CGATS TR 002, CGATS TR 003, CGATS TR 005, CGATS TR 006,
FOGRA30, FOGRA31, FOGRA32, FOGRA33, FOGRA34, FOGRA35, FOGRA36, FOGRA38, FOGRA39
FOGRA40, FOGRA41, FOGRA42, FOGRA43, FOGRA44, FOGRA45, FOGRA46, FOGRA47,
IFRA26, IFRA30,
EUROSB104, EUROSB204,
JC200103, JC200104, JCN2002, JCW2003
```

PDF/X-4: Zur Erzeugung eines PDF-Dokuments muss ein Profil für die Druckausgabebedingung verfügbar sein, welches dann eingebettet wird.

PDF/X-4/5: Ein Profil für die CMYK-Druckausgabebedingung (das heißt, geladen mit *usage=outputintent)* kann im selben Dokument nicht für einen ICC-basierten Farbraum verwendet werden (das heißt, geladen mit *usage=iccbased)*. Dies ist eine Anforderung des PDF/X-Standards und bezieht sich nur auf CMYK-Profile, nicht jedoch auf Graustufen- oder RGB-Profile. Besteht die Anforderung, das gleiche CMYK-Profil wie in der Druckausgabebedingung auch als ICC-basierte Farbe zu verwenden (z.B. für das Tagging eines Bildes), können Sie das ICC-Profil einfach weglassen, da das Profil für die Druckausgabebedingung bei PDF/X von vorneherein verwendet wird.

PDF/X-4p/5pg: Das Profil wird nicht eingebettet, sondern extern referenziert. Es muss sowohl bei der Generierung als auch später zum Betrachten und Drucken des Dokuments verfügbar sein. Im PDFlib-Tutorial finden Sie einen wichtigen Hinweis zu Acrobat-Problemen mit referenzierten ICC-Profilen.

Wenn eine der obigen Standardausgabebedingungen bei PDF/X-4 oder PDF/X-5 verwendet wird, muss das entsprechende ICC-Profil mit der Ressource *ICCProfile* konfiguriert werden.

Gültigkeit *document;* die Druckausgabebedingung sollte direkt nach *PDF_begin_document( )* gesetzt werden. Bei *usage=iccbased* sind auch folgende Gültigkeitsbereiche zulässig: *page, pattern, template, glyph*.

8.3 Füllmuster (Patterns) und Farbverläufe

C++ ***int begin_pattern_ext(double width, double height, string optlist)***
Perl PHP ***int begin_pattern_ext(float width, float height, string optlist)***
C ***int PDF_begin_pattern_ext(PDF *p, double width, double height, const char *optlist)***

Leitet die Definition eines Patterns mit Optionen ein.

width, height Die Abmessungen der Boundingbox des Patterns im Koordinatensystem des Patterns.

optlist Optionsliste mit Pattern-Eigenschaften gemäß Tabelle 8.3:
boundingbox, painttype, tilingtype, topdown, transform, xstep, ystep

Rückgabe Ein Pattern-Handle, das in nachfolgenden Aufrufen von *PDF_setcolor( )* und für die Optionen *fillcolor* und *strokecolor* innerhalb des umschließenden Gültigkeitsbereichs *document* verwendet werden kann.

Details Diese Funktion setzt alle Text-, Grafik- und Farbzustandsparameter auf ihre Standardwerte zurück. Die Option *transform* definiert die Zuordnung des Koordinatensystems des Patterns auf das der Seite, des Templates oder der Glyphenbeschreibung, in denen das Pattern verwendet wird. Alle Text- und Grafikfunktionen dürfen innerhalb einer Pattern-Definition verwendet werden. Hypertext-Funktionen und Funktionen zum Öffnen von Rasterbildern, Grafik und PDF-Seiten sind jedoch nicht erlaubt. Farbfunktionen könne abhängig vom Wert der Option *painttype* verwendet werden.

Gültigkeit beliebig außer *object*; mit dieser Funktion beginnt der Gültigkeitsbereich *pattern*; diese Funktion muss immer paarweise mit *PDF_end_pattern( )* aufgerufen werden.

Tabelle 8.3 Optionen für PDF_begin_pattern_ext( )

Option	Beschreibung
boundingbox	*(Rechteck) Koordinaten der linken, unteren, rechten und oberen Ecke der Boundingbox der Pattern-Zelle. Die Boundingbox kann zum Beschneiden der Pattern-Zelle oder zur Erzeugung von Weißraum um die sichtbaren Pattern-Elemente herum verwendet werden. Standardwert:* `{0 0 width height}`
painttype	*(Schlüsselwort) Gibt an, ob das Pattern eigene Farbspezifikationen enthält oder die aktuelle Zeichen- oder Füllfarbe angewandt wird, wenn es zum Zeichnen oder Füllen benutzt wird (Standardwert:* `colored`*):* ***colored*** *(Äquivalent zu* `painttype=1` *in der veralteten Funktion PDF_begin_pattern( )) Das Pattern wird mit einem oder mehreren Aufrufen von PDF_setcolor( ) oder den Optionen* `fillcolor`*/*`strokecolor` *eingefärbt. Innerhalb des Patterns können Bilder, PDF-Seiten oder Grafiken platziert werden.* ***uncolored*** *(Äquivalent zu* `painttype=2` *in der veralteten Funktion PDF_begin_pattern( )) Das Pattern enthält keine Farbspezifikation. Stattdessen wird die aktuelle Zeichen- oder Füllfarbe angewandt, wenn es zum Zeichnen oder Füllen benutzt wird. Bildmasken können verwendet werden; Bilder, platzierte PDF-Seiten und Grafiken dürfen nicht verwendet werden. Vor der Verwendung des Patterns muss mit PDF_setcolor( ) oder den Optionen* `fillcolor`*/*`strokecolor` *die aktuelle Farbe mit einem Farbraum gesetzt werden, der selbst nicht auf einem anderen Pattern basiert.*

Tabelle 8.3 Optionen für PDF_begin_pattern_ext()

Option	Beschreibung
tilingtype	*(Schlüsselwort) Steuert die Anpassung des Abstands von Kachel-Füllmustern (Standardwert:* `constantspacing`*):* ***constantspacing*** *Pattern-Zellen werden mit konsistenten Abständen dargestellt, das heißt, mit einer ganzen Zahl von Gerätepixeln. Der PDF-Viewer muss das Pattern dieser Zellen durch Anpassung der Werte* `xstep`*,* `ystep` *und der Transformationsmatrix dazu eventuell etwas verzerren.* ***nodistortion*** *Die Pattern-Zelle wird nicht verzerrt, aber der Abstand zwischen den Zellen kann horizontal oder vertikal bis zu einem Gerätepixel variieren, wenn das Pattern gezeichnet wird. Dadurch wird der durchschnittliche gewünschte Abstand erzielt, jedoch nicht unbedingt für jede einzelne Zelle.* ***fastertiling*** *Pattern-Zellen werden mit konsistenten Abständen dargestellt, genau wie bei* `constanttiling`*, jedoch ist eine zusätzliche Verzerrung für eine effizientere Implementierung erlaubt.*
topdown	*(Boolean) Bei* `true` *wird der Ursprung des Koordinatensystems am Anfang der Füllmusterdefinition in die linke obere Ecke gelegt, und die y-Koordinaten wachsen nach unten; anderenfalls wird das Koordinatensystem des Füllmusters direkt verwendet. Standardwert:* `false`
transform	*(Transformationsliste) Liste mit der Transformationsmatrix, die das Koordinatensystem des Füllmusters dem Standard-Koordinatensystem der Seite, des Templates oder der Glyphenbeschreibung zuordnet, in denen das Füllmuster verwendet wird. Die Verkettung der Pattern-Matrix mit der der Seite, des Templates oder der Glyphenbeschreibung bildet das Koordinatensystem des Pattern dar, in dem alle Grafikobjekte interpretiert werden.* *Die Liste enthält Paare aus einem Schlüsselwort und einer Float-Liste gemäß Tabelle 8.3., wobei jedes Paar eine Transformation definiert. Interpretation und Länge der Zahlenliste hängen von der Transformation ab. Die Transformationen werden in der angegebenen Reihenfolge angewendet. Die Elemente innerhalb eines Paares können durch Gleichheitszeichen '=' getrennt werden. Standardwert: keine Transformationen* *Beispiel:* `transform={rotate=45 translate={100 0}}`
xstep	*(Float) Horizontaler Abstand zwischen den Pattern-Zellen in den Pattern-Koordinaten. Standardwert:* `width`
ystep	*(Float) Vertikaler Abstand zwischen den Pattern-Zellen in Pattern-Koordinaten. Standardwert:* `height`

Tabelle 8.4 Schlüsselwörter und Float-Listen für die Option `transform` *von PDF_begin_pattern_ext()*

Schlüsselwort	Beschreibung
align	*Drehung um den Richtungsvektor* `{dx dy}`
matrix	*Legt eine Transformationsmatrix durch ihre sechs Komponenten fest* `{a b c d e f}`.
rotate	*Drehung um* `{phi}`*, wobei der Winkel* `phi` *in Grad gegen den Uhrzeigersinn von der positiven* x*-Achse der Pattern-Koordinaten gemessen wird.*
scale	*Skalierung um* `{sx sy}`*. Wird* `sy` *nicht übergeben, wird der gleiche Wert wie für* `sx` *verwendet.*
skew	*Scherung des Koordinatensystems um* `{alpha beta}`*.* `alpha` *wird dabei von der positiven x-Achse des aktuellen Koordinatensystems ausgehend gegen den Uhrzeigersinn gemessen, und* `beta` *von der positiven y-Achse ausgehend im Uhrzeigersinn. Beide Winkel müssen im Bereich -360° <* `alpha, beta` *< 360° liegen und dürfen nicht -270°, -90°, 90° oder 270° sein.*
translate	*Verschiebung um* `{tx ty}`*.*

C++ Java C# **void end_pattern( )**

Perl PHP **end_pattern( )**

C **void PDF_end_pattern(PDF *p)**

Beendet eine Pattern-Definition.

Gültigkeit *pattern;* mit dieser Funktion endet der Gültigkeitsbereich *pattern*; diese Funktion muss immer paarweise mit *PDF_begin_pattern_ext( )* aufgerufen werden.

C++ Java C# **int begin_pattern(double width, double height, double xstep, double ystep, int painttype)**

Perl PHP **int begin_pattern(float width, float height, float xstep, float ystep, int painttype)**

C **int PDF_begin_pattern(PDF *p,**
double width, double height, double xstep, double ystep, int painttype)

Veraltet, verwenden Sie stattdessen *PDF_begin_pattern_ext( )*.

C++ Java C# **int shading_pattern(int shading, String optlist)**

Perl PHP **int shading_pattern(int shading, string optlist)**

C **int PDF_shading_pattern(PDF *p, int shading, const char *optlist)**

Definiert ein Farbverlaufsmuster anhand eines Farbverlaufsobjekts.

shading Shading-Handle, das von *PDF_shading( )* erzeugt wurde.

optlist Optionsliste für die Grafikeigenschaften des Farbverlaufsmusters gemäß Tabelle 7.1: *gstate*.

Rückgabe Pattern-Handle, das in nachfolgenden Aufrufen von *PDF_setcolor( )* und für die Optionen *fillcolor* und *strokecolor* innerhalb des umschließenden Gültigkeitsbereichs *document* verwendet werden kann.

Details Diese Funktion dient zum Füllen beliebiger Objekte mit einem Farbverlauf. Dazu muss mit *PDF_shading( )* zunächst ein Shading-Handle erstellt und dann mit *PDF_shading_pattern( )* auf Basis dieses Farbverlaufs ein Pattern definiert werden. Das Pattern-Handle wiederum wird an *PDF_setcolor( )* oder die Optionen *fillcolor* und *strokecolor* übergeben, die die aktuelle Farbe auf das Farbverlaufsmuster setzen.

Gültigkeit beliebig außer *object*

C++ Java C# **void shfill(int shading)**

Perl PHP **shfill(int shading)**

C **void PDF_shfill(PDF *p, int shading)**

Füllt einen Bereich mit einem Farbverlauf, der auf einem Farbverlaufsobjekt basiert.

shading Farbverlaufsobjekt, das mit *PDF_shading( )* erzeugt wurde.

Details Diese Funktion ermöglicht die Verwendung von Farbverläufen ohne *PDF_shading_pattern( )*, *PDF_setcolor( )* oder die Optionen *fillcolor* und *strokecolor*. Sie funktioniert aber nur bei einfachen Formen, bei denen die Geometrie des zu füllenden Objekts derjenigen des Farbverlaufs entspricht. Da der aktuelle Clipping-Bereich mit dem Farbverlauf

gefüllt wird (was von den Farbverlaufsoptionen *extendo* und *extend1* beeinflusst wird), wird diese Funktion in der Regel in Kombination mit *PDF_clip( )* verwendet.

Gültigkeit *page, pattern* (falls *painttype=colored), template, glyph* (falls die Option *colorized* des Type-3-Fonts gleich *true* ist)

C++ Java C# ***int shading(String shtype, double x0, double y0, double x1, double y1, double c1, double c2, double c3, double c4, String optlist)***

Perl PHP ***int shading(string shtype, float x0, float y0, float x1, float y1, float c1, float c2, float c3, float c4, string optlist)***

C ***int PDF_shading(PDF *p, const char *shtype, double x0, double y0, double x1, double y1, double c1, double c2, double c3, double c4, const char *optlist)***

Definiert einen Farbverlauf zwischen der aktuellen Füllfarbe und der übergebenen Farbe.

shtype Typ des Farbverlaufs: *axial* für lineare und *radial* für kreisförmige Verläufe.

x0, y0, x1, y1 Bei linearen Farbverläufen bilden *(x0, y0)* und *(x1, y1)* die Koordinaten des Anfangs- und Endpunkts des Farbverlaufs. Bei kreisförmigen Farbverläufen legen sie die Mittelpunkte der Anfangs- und Endkreise fest.

c1, c2, c3, c4 Farbwerte des Endpunkts des Farbverlaufs, die im Farbraum der aktuellen Füllfarbe auf dieselbe Art wie die Farbparameter in *PDF_setcolor( )* und den Optionen *fillcolor* und *strokecolor* interpretiert werden. Ist die aktuelle Füllfarbe eine Schmuck_farbe, wird *c1* ignoriert und *c2* enthält den Farbauftrag.

optlist Optionsliste mit Farbverlaufseigenschaften gemäß Tabelle 8.5. Folgende Optionen können verwendet werden:
antialias, boundingbox, domain, extendo, extend1, N, r0, r1, startcolor

Rückgabe Shading-Handle, das innerhalb des umgebenden Gültigkeitsbereichs *document* in späteren Aufrufen von *PDF_shading_pattern( )* und *PDF_shfill( )* verwendet werden kann.

Details Die aktuelle Füllfarbe wird als Startfarbe verwendet; sie darf nicht auf einem Pattern basieren.

Gültigkeit beliebig außer *object*

Tabelle 8.5 Optionen für PDF_shading( )

Option	Beschreibung
antialias	*(Boolean) Legt fest, ob Antialiasing (Kantenglättung) für den Farbverlauf aktiviert wird. Standardwert:* `false`
boundingbox	*(Rechteck) Rechteck, das die Boundingbox für den Farbverlauf in Benutzerkoordinaten festlegt. Die Boundingbox wird als temporärer Beschneidungspfad verwendet, wenn der Farbverlauf aufgetragen wird (zusätzlich zum aktuellen Beschneidungspfad, sofern vorhanden). Mit dieser Option lässt sich der Farbverlauf ohne PDF_clip( ) beschneiden.*
domain	*(Liste mit zwei Floats) Zwei Zahlen, die die Grenzwerte einer Variablen* `t` *festlegen. Die Variable bewegt sich linear zwischen diesen beiden Werten, wenn der Farbverlauf vom Start- und Endpunkt der Achse wandert. Standardwert:* `{0 1}`
extendo	*(Boolean) Legt fest, ob der Farbverlauf über den Startpunkt hinaus verlängert wird. Standardwert:* `false`

Tabelle 8.5 Optionen für PDF_shading( )

Option	Beschreibung
extend1	*(Boolean) Legt fest, ob der Farbverlauf über den Endpunkt hinaus verlängert wird. Standardwert:* `false`
N	*(Float) Exponent der Farbübergangsfunktion; muss > 0 sein. Standardwert: 1*
r0	*(Float; nur erforderlich bei kreisförmigen Farbverläufen) Radius des Anfangskreises*
r1	*(Float; nur erforderlich bei kreisförmigen Farbverläufen) Radius des Endkreises*
startcolor	*(Farbe) Farbe des Startpunkts. Mit dieser Option ist die Funktion nicht mehr von der aktuellen Farbe abhängig. Standardwert: die aktuelle Füllfarbe*

9 Rasterbild-, SVG- und Template-Funktionen

9.1 Rasterbilder

Cookbook *Ein vollständiges Codebeispiel hierzu finden Sie im Cookbook-Topic* `images/starter_image`.

C++ Java C# ***int load_image(String imagetype, String filename, String optlist)***
Perl PHP ***int load_image(string imagetype, string filename, string optlist)***
C ***int PDF_load_image(PDF *p,***
const char *imagetype, const char *filename, int len, const char *optlist)

Öffnet eine auf der Festplatte liegende oder virtuelle Bilddatei unter Anwendung verschiedener Optionen.

imagetype Wenn Sie den String *auto* übergeben, versucht PDFlib, den Typ der Bilddatei automatisch zu erkennen (dies ist bei Bildern vom Typ *CCITT* und *raw* nicht möglich). Es beschleunigt die Verarbeitung etwas, wenn Sie das Bildformat explizit mit einem der Strings *bmp, ccitt, gif, jbig2, jpeg, jpeg2000* (PDF 1.5 und höher), *png*, *raw* oder *tiff* übergeben. Weitere Informationen zu Bildformaten finden Sie im PDFlib-Tutorial.

filename (Name-String; wird gemäß der globalen Option *filenamehandling* interpretiert, siehe Tabelle 2.3) Name der Bilddatei, die geöffnet werden soll. Es muss sich um eine lokale oder virtuelle Datei handeln, da PDFlib keine Bilddaten von URLs bezieht.

Wurde keine Datei mit dem angegebenen Namen gefunden und ist *imagetype=auto*, so versucht PDFlib automatisch, die Dateinamenserweiterung zu ermitteln; alle in der folgenden Liste enthaltenen Erweiterungen werden versuchsweise (in Klein- und in Großbuchstaben) an den in *filename* übergebenen Dateinamen angehängt und eine Datei dieses Namens in den im Searchpath enthaltenen Verzeichnissen gesucht:

```
.bmp, .ccitt, .g3, .g4, .fax, .gif, .jbig2, .jb2, .jpg, .jpeg, .jpx, .jp2, .jpf, .jpm,
.png, .raw, .tif, .tiff
```

len (Nur C-Sprachbindung) Länge von *filename* (in Bytes). Ist *len* = *0*, muss ein null-terminierter String übergeben werden.

optlist Optionsliste mit Bildeigenschaften gemäß Tabelle 9.1. Folgende Optionen können verwendet werden:

- Allgemeine Optionen: *errorpolicy* (siehe Tabelle 2.1) und *hypertextencoding* (siehe Tabelle 2.3)
- Farboptionen: *colorize, honoriccprofile, iccprofile, invert, renderingintent*
- Optionen für Beschneidungspfade, Masken und Transparenzmasken: *alphachannelname, clippingpathname, downsamplemask, honorclippingpath, ignoremask, mask, masked*
- Besondere PDF-Funktionen für Rasterbilder: *createtemplate, interpolate*
- Option zur Bildanalyse ohne PDF-Ausgabe: *infomode*

- Optionen zur Verarbeitung der Bilddaten:
  cascadedflate, ignoreorientation, page, passthrough
- Optionen für CCITT-, JBIG2- und Raw-Bilder gemäß Tabelle 9.2:
  bitreverse, bpc, components, copyglobals, height, imagehandle, inline, K, width

Rückgabe Handle für das Bild (oder für das Template bei *createtemplate=true)*, das in nachfolgenden Aufrufen von Bildfunktionen verwendet werden kann. Bei *errorpolicy=return* muss der Anwendungsentwickler den Rückgabewert auf -1 (in PHP: 0) überprüfen, da dies auf einen Fehler hinweist. Das zurückgelieferte Image-Handle kann nicht über mehrere PDF-Dokumente hinweg verwendet werden. Scheitert der Funktionsaufruf, so können Sie die Fehlerursache mit *PDF_get_errmsg( )* abfragen.

Details Diese Funktion öffnet und analysiert eine Rasterbilddatei in einem der unterstützten Dateiformate, das mit dem Parameter *imagetype* ausgegeben wurde, und kopiert alle relevanten Bilddaten in das Ausgabedokument. Die Funktion hat keinen sichtbaren Einfluss auf die Ausgabe. Um das importierte Bild tatsächlich im generierten Ausgabedokument zu platzieren, muss *PDF_fit_image( )* aufgerufen werden. Ein Bild sollte für jedes generierte Dokument nur einmal geöffnet werden, da die Bilddaten sonst mehrfach ins Ausgabedokument kopiert werden. Wenn die Anwendung dies nicht verhindern kann, können Sie redundante Daten mit der Option *optimize* von *PDF_begin_document( )* entfernen.

PDFlib öffnet die Bilddatei mit dem in *filename* übergebenen Namen, verarbeitet den Inhalt und schließt sie wieder, bevor dieser Funktionsaufruf zurückkehrt. Selbst wenn Bilder innerhalb eines Dokuments mit *PDF_fit_image( )* mehrfach platziert werden, muss die zugehörige Bilddatei nicht erneut geöffnet werden.

PDF/A Einige Optionen sind eingeschränkt:

Bei Graustufenbildern muss eine Druckausgabebedingung beliebigen Typs oder die Option *defaultgray* von *PDF_begin_page_ext( )* festgelegt werden.

Bei RGB- Bildern muss eine RGB- Druckausgabebedingung oder die Option *defaultrgb* von *PDF_begin_page_ext( )* festgelegt werden.

Bei CMYK- Bildern muss eine CMYK- Druckausgabebedingung oder die Option *defaultcmyk* von *PDF_begin_page_ext( )* festgelegt werden.

PDF/A-2/3: JPEG-2000-Bilder müssen bestimmte Bedingungen erfüllen; für weitere Informationen siehe das PDFlib-Tutorial.

PDF/X Einige Optionen sind eingeschränkt.

PDF/X-1a: RGB-Bilder sind nicht erlaubt.

PDF/X-1a/3: JBIG2-Bilder sind nicht erlaubt.

PDF/X-3/4/5: Graustufenbilder setzen voraus, dass die Option *defaultgray* in *PDF_begin_page_ext( )* gesetzt wurde, sofern das in der Druckausgabebedingung festgelegte Gerät nicht mit Graustufen oder CMYK arbeitet.

Bei RGB-Bildern muss die Option *defaultrgb* in *PDF_begin_page_ext( )* gesetzt worden sein, sofern das in der Druckausgabebedingung festgelegte Gerät nicht mit RGB arbeitet.

Bei CMYK-Bildern muss die Option *defaultcmyk* in *PDF_begin_page_ext( )* gesetzt worden sein, sofern das in der Druckausgabebedingung festgelegte Gerät nicht mit CMYK arbeitet.

JPEG-2000-Bilder müssen bestimmte Bedingungen erfüllen; für weitere Informationen siehe das PDFlib-Tutorial.

PDF/VT Dieser Aufruf kann fehlschlagen, wenn die Option *usestransparency=false* in *PDF_begin_document( )* angegeben wurde, das importierte Bild aber dennoch Transparenz enthält.

PDF/UA Bilder sollten als *Figure* oder *Artifact* ausgezeichnet werden.

Gültigkeit beliebig außer *object;* diese Funktion sollte immer paarweise mit *PDF_close_image( )* aufgerufen werden.

Tabelle 9.1 Optionen für PDF_load_image( )

Option	Beschreibung
alphachannelname	*(Name-String; nur für TIFF-Bilder; wird bei* `ignoremask=true` *ignoriert) Liest den Alphakanal mit dem angegebenen Namen aus der Bilddatei und weist ihn dem Bild als Transparenzmaske zu. Der benannte Kanal muss in der Bilddatei vorhanden sein. Standardwert: der erste Alphakanal im Bild*
cascadedflate	*(Boolean; nur bei* `imagetype=jpeg`*) Bei* `true` *wird eine zusätzliche Ebene von Flate-Kompression auf die JPEG-komprimierten Bilddaten angewandt. In manche Fällen lässt sich dadurch die Ausgabedatei verkleinern, z.B. bei Bildern mit großen gleichfarbigen Flächen. Bei den meisten Typen von Bildinhalten führt diese Option jedoch nicht zu einer kleineren, sondern manchmal sogar zu einer größeren Ausgabedatei. Standardwert:* `false`
clippingpathname	*(String; nur für* `imagetype=tiff` *und* `jpeg`*; wird bei* `honorclippingpath=false` *ignoriert) Liest den Pfad mit dem angegebenen Namen aus der Bilddatei und verwendet ihn als Beschneidungspfad. Der benannte Pfad muss in der Bilddatei vorhanden sein. Mit* `Work Path` *kann ein mit Photoshop erstellter temporärer Pfad angesprochen werden. Standardwert: Name des Pfades, der in der Bilddatei als Beschneidungspfad angegeben ist*
colorize	*(Schmuckfarben-Handle; wird ignoriert, wenn die Option* `iccprofile` *übergeben wird) Färbt das Bild anhand eines Schmuckfarben-Handle ein, das mittels PDF_makespotcolor( ) erzeugt wurde. Das Bild muss ein Schwarzweiß- oder Graustufenbild sein.*
createtemplate	*(Boolean) Bei* `true` *wird kein normales XObject vom Typ Image, sondern ein PDF Image XObject generiert, das in ein Form XObject (in PDFlib: Template) eingebettet ist. Dies kann zur Erstellung von Templates für Formularfeld-Icons nützlich sein, die nur aus einem Bild bestehen. Für das generierte Template wird ein Handle zurückgegeben. Standardwert:* `false`
downsamplemask	*(Boolean; nur bei* `type=TIFF` *und* `PNG`*) Bei* `true` *wird ein 16-Bit-Alphakanal auf einen 8-Bit-Alphakanal reduziert, um einen Fehler in Acrobat 7/8/9 zu umgehen. Dies kann nützlich sein, um 16-Bit-Masken in älteren Acrobat-Versionen korrekt anzuzeigen. Standardwert:* `false`
honorclippingpath	*(Boolean; nur bei* `imagetype=tiff` *und* `jpeg`*) Liest den Beschneidungspfad (sofern vorhanden) aus der Bilddatei ein und wendet ihn auf das Bild an. Standardwert:* `true`
honoriccprofile	*(Boolean; nur bei* `imagetype=jpeg`*,* `png` *und* `tiff`*; wird auf* `false` *gesetzt, wenn die Option* `colorize` *übergeben wird) Falls das Bild ein eingebettetes ein ICC-Profil oder einen ICC-Verweis im Exif-Marker enthält, wird dieses Profil auf das Bild angewendet. Standardwert:* `true`
iccprofile	*(ICC-Handle oder Schlüsselwort) Handle eines ICC-Profils, das auf das Bild angewandt werden soll. Mit dem Schlüsselwort* `srgb` *wird der sRGB-Farbraum ausgewählt. Standardwert: eingebettetes Profil (oder äquivalente Exif-Information in der Bilddatei), sofern im Bild vorhanden und* `honoriccprofile=true`
ignoremask	*(Boolean; muss in PDF/X-1/3 und PDF/A-1 bei Bildern mit einem Alphakanal auf* `true` *gesetzt werden) Alle in der Bilddatei vorhandenen Transparenzinformationen und Alphakanäle werden ignoriert. Standardwert:* `false`
ignoreorientation	*(Boolean; nur bei* `imagetype=tiff` *und* `jpeg`*) Ignoriert im Bild vorhandene Orientierungsangaben. Dies kann zur Korrektur falscher Orientierungsangaben in den Bilddaten dienen. Standardwert:* `false`

Tabelle 9.1 Optionen für PDF_load_image()

Option	Beschreibung
infomode	*(Boolean) Bei* `true` *wird das Bild geladen, ohne Pixeldaten in die Ausgabe zu schreiben. Bildeigenschaften können mit PDF_info_image() abgefragt werden, das Bild kann allerdings nicht mit PDF_fit_image() oder anderen Funktionen auf der Seite platziert werden. Mit dieser Option lässt sich das Bild prüfen, ohne die PDF-Ausgabe zu beeinflussen. Bei* `false` *werden die Pixeldaten sofort in die PDF-Ausgabe geschrieben. Standardwert:* `false`
interpolate	*(Boolean; muss bei PDF/A gleich* `false` *sein) Aktiviert die Interpolation des Bilds, um das Aussehen auf Bildschirm und Ausdruck zu verbessern. Dies ist insbesondere bei Rasterbildern nützlich, die als Glyphenbeschreibungen in Type-3-Fonts dienen. Standardwert:* `false`
invert	*(Boolean; nicht bei* `imagetype=jpeg2000`*, außer bei* `mask=true`*) Invertiert das Bild (durch Vertauschung von hellen und dunklen Farben). Dies kann bei manchen Bildern als Behelfslösung dienen, wenn sie von Anwendungen unterschiedlich interpretiert werden. Standardwert:* `false`
mask	*(Boolean; nur für Bilder mit einer Farbkomponente, einschließlich indizierter Farbe) Das Bild soll als Maske verwendet werden; erforderlich bei Masken mit 1-Bit-Pixeltiefe und optional bei Masken mit mehr als einem Bit pro Pixel. Standardwert:* `false`*. Es gibt zwei Anwendungsfälle für Masken:* ▸ *Maskieren eines anderen Bildes: Das zurückgegebene Image-Handle wird später beim Öffnen eines anderen Bildes verwendet und mit der Option* `masked` *übergeben.* ▸ *Platzieren eines eingefärbten transparenten Bildes: Pixelwerte von 0 im Bild werden als transparent interpretiert, und Werte von 1 werden mit der aktuellen Füllfarbe eingefärbt.* *Diese Option erzwingt* `ignoremask=true`*, da ein als Maske verwendetes Bild nicht selbst eine Maske enthalten kann.*
masked	*(Image-Handle; wird ignoriert, wenn das Bild einen Alphakanal enthält und* `ignoremask=false` *ist) Image-Handle für ein Bild, das als Maske auf das aktuelle Bild angewandt wird. Das Image-Handle wurde von einem vorangehenden Aufruf von PDF_load_image() zurückgegeben und noch nicht geschlossen. Im PDF/A-1- und PDF/X-1/3-Modus ist diese Option nur mit 1-Bit-Masken zulässig.*
page	*(Integer; nur bei* `imagetype=gif`*,* `jbig2` *und* `tiff`*; muss bei anderen Formaten 1 sein) Das Bild mit der angegebenen Nummer wird aus einer mehrseitigen Bilddatei extrahiert. Das erste Bild hat die Nummer 1. Der Aufruf schlägt fehl, wenn die entsprechende Seite nicht in der Bilddatei gefunden werden kann. Standardwert: 1*
passthrough	*(Boolean; nur bei* `imagetype=tiff` *oder* `jpeg`*) Steuert die Verarbeitung von TIFF- und JPEG-Bilddaten.* ***Tiff-Bilder*** *(Standardwert:* `true`*): Bei* `true` *werden komprimierte TIFF-Bilddaten möglichst direkt an die PDF-Ausgabe weitergereicht. Enthält ein TIFF-Bild beschädigte oder unvollständige Bilddaten, kann es sinnvoll sein, diese Option auf* `false` *zu setzen.* ***JPEG-Bilder*** *(Standardwert:* `false`*): Bei* `false` *werden die JPEG-Bilddaten von PDFlib durch Transkodierung bereinigt, um Kompatibilität zu Acrobat zu erreichen. Bei* `true` *werden JPEG-Bilddaten direkt in die PDF-Ausgabe kopiert. Diese Option wird bei Multiscan- und manchen CMYK-JPEG-Bildern ignoriert. Wenn Sie diese Option auf* `true` *setzen, wird die Bearbeitung zwar etwas schneller, aber einige selten vorkommende JPEG-Varianten werden in Acrobat nicht mehr korrekt angezeigt.*

Tabelle 9.1 Optionen für PDF_load_image( )

Option	Beschreibung
rendering-intent	*Rendering-Intent für Bilder (Standardwert:* `Auto`*):* ***Auto*** *Legt keinen Rendering-Intent in der PDF-Datei fest, sondern verwendet den Standard-Intent des Ausgabegeräts. Typische Verwendung: unbekannte Fälle* ***AbsoluteColorimetric*** *Es findet keine Korrektur des Weißpunkts des Gerätes statt (zum Beispiel Papierweiß). Farben außerhalb des Farbraums werden auf die nächstliegenden Werte innerhalb des Gerätefarbraums abgebildet. Typischer Einsatz: exakte Reproduktion von Volltonfarben; sollte nur in diesem Fall verwendet werden* ***RelativeColorimetric*** *Die Farbdaten werden in den Gerätefarbraum hineinskaliert, wobei die Weißpunkte bei leichter Farbverschiebung aufeinander abgebildet werden. Typische Verwendung: Vektorgrafiken* ***Saturation*** *Die Farbsättigung bleibt erhalten, wobei sich die Farbwerte verschieben können. Typischer Einsatz: Geschäftsgrafiken* ***Perceptual*** *Die Farbrelationen bleiben erhalten, indem Farben innerhalb und außerhalb des Farbraums gleichermaßen verändert werden, um eine gefällige Darstellung zu erhalten. Typische Verwendung: gescannte Bilder*
template	*Veraltet; verwenden Sie stattdessen* `createtemplate`

Tabelle 9.2 Optionen für PDF_load_image( ) mit `imagetype=ccitt`, `jbig2` *oder* `raw`

Option	Beschreibung
bitreverse	*(Boolean; nur bei* `imagetype=ccitt`*) Bei* `true` *wird die Bitreihenfolge eines jeden Bytes in den komprimierten Daten umgekehrt. Standardwert:* `false`
bpc	*(Integer; nur bei* `imagetype=raw` *und dann erforderlich) Anzahl der Bits pro Komponente; muss 1, 2, 4, 8 oder 16 sein. In PDF 1.4 nur 1, 2, 4 oder 8).*
components	*(Integer; nur bei* `imagetype=raw` *und dann erforderlich) Anzahl der Bildkomponenten (Kanäle); muss 1, 3 oder 4 sein.*
copyglobals	*(Schlüsselwort; nur bei* `imagetype=jbig2`*) Legt fest, welche globalen Segmente in einem JBIG2-Stream in das PDF-Dokument kopiert werden. Enthält der JBIG2-Stream keine globalen Segmente, hat diese Option keine Auswirkungen (Standardwert:* `current`*):* ***all*** *Kopiert die globalen Segmente aller Seiten im JBIG2-Stream in das PDF-Dokument. Dies sollte verwendet werden, wenn mehr als eine Seite aus dem selben JBIG2-Stream importiert wird. Sollen später weitere Seiten aus dem selben JBIG2-Stream verwendet werden, sollte die Option* `imagehandle` *verwendet werden.* ***current*** *Kopiert nur die für die aktuelle Seite erforderlichen globalen Segmente im JBIG2-Stream (das heißt, die in der Option* `page` *angegebene Seite) in das PDF. Dies sollte verwendet werden, wenn keine weiteren Seiten aus dem selben JBIG2-Stream mehr importiert werden.*
height	*(Integer; nur bei* `imagetype=raw` *oder* `ccitt` *und dann erforderlich) Bildhöhe in Pixeln*
imagehandle	*(Image-Handle; nur bei* `imagetype=jbig2`*) Fügt eine Referenz auf ein globales Segment hinzu, das zu einem anderen Bild gehört, das aus dem selben JBIG2-Stream erzeugten wurde. Dieses Bild muss zuvor mit der Option* `copyglobals=all` *geladen worden sein. Wird ein Bild referenziert, das aus einer anderen Datei als dem aktuellen JBIG2-Stream erzeugt wurde, führt dies zu einem Fehler. Das angegebene Image-Handle darf noch nicht geschlossen worden sein. Standardwert: kein Image-Handle, das heißt, es wird ein neues PDF-Objekt mit allen erforderlichen globalen Segmenten für die aktuelle Seite erzeugt.*

Tabelle 9.2 Optionen für PDF_load_image( ) mit `imagetype=ccitt`, `jbig2` *oder* `raw`

Option	Beschreibung
inline	*(Boolean; nur bei* `imagetype=ccitt`, `jpeg` *und* `raw`**;** *nicht erlaubt, wenn eine der Optionen* `iccprofile` *oder* `colorize` *übergeben wurde) Bei* `true` *wird das Bild direkt in den Content-Stream der Seiten-, Pattern-, Template- oder Glyphenbeschreibung eingefügt. Mit dieser Option wird implizit PDF_fit_image( ) und PDF_close _image( ) aufgerufen (siehe PDFlib-Tutorial). Diese Option wird für Bitmap-Glyphen von Type-3-Fonts empfohlen und sollte ansonsten nicht verwendet werden. Wird diese Option übergeben, darf PDF_close_image( ) nicht aufgerufen werden. Standardwert:* `false`
K	*(Integer; nur bei* `imagetype=ccitt`*) CCITT-Kompressionsparameter zur Auswahl des Kompressionsverfahrens (Standardwert: 0):* **-1** *G4-Kompression* **0** *eindimensionale G3-Kompression (G3-1D)* **1** *gemischte ein- und zweidimensionale Kompression (G3, 2-D)*
width	*(Integer; nur bei* `imagetype=raw` *oder* `ccitt` *und dann erforderlich) Breite des Bilds in Pixeln*

C++ Java C# ***void close_image(int image)***

Perl PHP ***close_image(int image)***

C ***void PDF_close_image(PDF *p, int image)***

Schließt ein Bild oder Template.

image Gültiges Image- oder Template-Handle, das mit *PDF_load_image( )* oder *PDF_begin_template_ext( )* erzeugt wurde.

Details Diese Funktion wirkt sich nur auf die in PDFlib verwaltete interne Bildstruktur aus. Wurde das Bild aus einer Datei geöffnet, bleibt die eigentliche Bilddatei von dieser Funktion unberührt, da sie bereits am Ende von *PDF_load_image( )* geschlossen wurde. Ein mit dieser Funktion geschlossenes Image-Handle kann nicht weiter verwendet werden.

Gültigkeit beliebig außer *object*; diese Funktion darf nur paarweise mit einem entsprechenden Aufruf von *PDF_load_image( )* aufgerufen werden (sofern nicht die Option *inline* verwendet wurde) oder von *PDF_begin_template_ext( )*.

C++ Java C# ***void fit_image(int image, double x, double y, String optlist)***

Perl PHP ***fit_image(int image, float x, float y, string optlist)***

C ***void PDF_fit_image(PDF *p, int image, double x, double y, const char *optlist)***

Platziert ein Bild oder Template unter Anwendung verschiedener Optionen auf der Seite.

image Gültiges Image- oder Template-Handle, das mit *PDF_load_image( )* oder *PDF_begin_template_ext( )* erzeugt wurde. Das Bild darf nicht mit *infomode=true* geladen worden sein.

x, y Koordinaten des Referenzpunkts im Benutzerkoordinatensystem, an dem das Bild oder Template platziert werden soll. Die Platzierung wird über verschiedene Optionen gesteuert.

optlist Optionsliste mit Optionen zur Objekteinpassung und Bildverarbeitung. Folgende Optionen werden unterstützt:

- Optionen zur Objekteinpassung gemäß Tabelle 6.1:
  boxsize, blind, dpi, fitmethod, matchbox, orientate, position, rotate, scale, showborder
- Optionen zur Bildverarbeitung gemäß Tabelle 9.3:
  adjustpage, gstate, ignoreclippingpath, ignoreorientation
- Option zum vereinfachten Tagging von Strukturelementen gemäß Tabelle 14.5 (nur erlaubt im Gültigkeitsbereich *page): tag*

Details Das Bild oder Template (die im Folgenden unterschiedslos »Objekt« genannt werden) wird relativ zum Referenzpunkt *(x, y)* platziert. Standardmäßig wird die linke untere Ecke des Objekts am Referenzpunkt platziert. Dieses Verhalten kann durch die Optionen *orientate, boxsize, position* und *fitmethod* geändert werden. Standardmäßig wird ein Bild gemäß seiner Auflösung skaliert. Dieses Verhalten lässt sich durch die Optionen *dpi, scale* und *fitmethod* modifizieren.

PDF/UA Rasterbilder sollten als *Figure* oder *Artifact* ausgezeichnet werden, entweder mit der Option *tag* oder mit einem vorausgehenden Aufruf von *PDF_begin_item( )*.

Gültigkeit *page, pattern* (nur, wenn für das Pattern *painttype=colored* gesetzt wurde oder das Bild eine Maske ist), *template, glyph* (nur, wenn die Option *colorized* des Type-3-Fonts gleich *true* oder das Bild eine Maske ist); diese Funktion kann auf beliebigen Seiten beliebig oft aufgerufen werden, solange das Image-Handle noch nicht mit *PDF_close_image( )* geschlossen wurde.

Tabelle 9.3 Optionen für die Verarbeitung von Grafiken, Bildern und Templates mit PDF_fit_graphics( ), PDF_fit_pdi_page( ), PDF_fill_imageblock( ), PDF_fill_graphicsblock( ) und PDF_fill_pdfblock( )

Option	Beschreibung
adjustpage	*(Boolean; nur im Gültigkeitsbereich* page *wirksam; nicht zulässig, wenn die Option* topdown *in PDF_begin_page_ext( ) übergeben wurde; nicht bei PDF_fill_*block( )) Passt die Größe der aktuellen Seite an das Objekt an, so dass die rechte obere Ecke der Seite auf die rechte obere Ecke des Objekts plus den Funktionsparametern (x, y) zu liegen kommt. Die Media-Box wird angepasst, und alle anderen Boxeinträge werden auf ihre Standardwerte zurückgesetzt. Mit der Option* position=0 *sind folgende Fälle nützlich:* *x >= 0 und y >= 0* *Das Objekt wird von einem weißen Rand umgeben. Dieser Rand hat die Dicke* y *in horizontaler Richtung und die Dicke* x *in vertikaler Richtung.* *x < 0 und y < 0* *Vom Bild werden horizontale und vertikale Streifen abgeschnitten.* *Standardwert:* false
gstate	*(Gstate-Handle) Handle für einen mit PDF_create_gstate( ) erzeugten Grafikzustand. Der Grafikzustand wirkt sich auf alle mit dieser Funktion erzeugten grafischen Elemente aus. Standardwert: kein Grafik-Handle (das heißt, die aktuellen Einstellungen werden verwendet)*
ignore-clippingpath	*(Boolean; nur für TIFF- und JPEG-Bilder) Ein eventuell in der Bilddatei vorhandener Beschneidungspfad wird ignoriert. Standardwert:* false*, d.h. der Beschneidungspfad wird angewandt.*
ignore-orientation	*(Boolean; nur für TIFF- und JPEG-Bilder) Ignoriert eventuell im Bild vorhandene Orientierungsangaben. Dies kann zur Korrektur falscher Orientierungsangaben dienen. Standardwert: Wert der Option* ignoreorientation *in PDF_load_image( )*

C++ Java C# ***double info_image(int image, String keyword, String optlist)***

Perl PHP ***float info_image(int image, string keyword, string optlist)***

C ***double PDF_info_image(PDF *p, int image, const char *keyword, const char *optlist)***

Formatiert ein Bild oder Template und fragt Metrik und andere Eigenschaften ab.

image Gültiges Bild- oder Template-Handle, das mit *PDF_load_image( )* oder *PDF_begin_template_ext( )* erzeugt wurde.

keyword Schlüsselwort zur Abfrage der gewünschten Information:

- Schlüsselwörter zur Abfrage der Ergebnisse der Objekteinpassung gemäß Tabelle 6.3: *boundingbox, fitscalex, fitscaley, height, objectheight, objectwidth, width, x1, y1, x2, y2, x3, y3, x4, y4*
- Zusätzliche Schlüsselwörter gemäß Tabelle 9.4: *clippingpath, checkcolorspace, filename, iccprofile, imageheight, imagemask, imagetype, imagewidth, infomode, mirroringx, mirroringy, orientation, resx, resy, strips, transparent, xid*

optlist Folgende Optionen werden unterstützt:

- Optionen für *PDF_fit_image( )*. Optionen, die für die Bestimmung des Werts des angegebenen Schlüsselworts nicht relevant sind, werden ignoriert.
- Option, um zwischen Bild und Template zu wechseln: *useembeddedimage*

Rückgabe Der Wert der Bildeigenschaft, die durch *keyword* ausgewählt wurde. Ist die ausgewählte Eigenschaft nicht in der Bilddatei vorhanden, gibt die Funktion den Wert 0 zurückgegeben. wurde ein Objekt-Handle angefordert (z.B. *clippingpath),* gibt diese Funktion ein Handle für das Objekt zurück, oder -1 (in PHP: 0), falls das Objekt nicht vorhanden ist. Wenn das erforderliche Schlüsselwort Text produziert, wird ein String-Index zurückgegeben und der entsprechende String muss mit *PDF_get_string( )* ermittelt werden.

Details Diese Funktion führt alle Berechnungen durch, die zur Platzierung des Bildes gemäß den übergebenen Optionen erforderlich sind, generiert aber keinerlei Ausgabe auf der Seite. Als Referenzpunkt für den Text wird *{0 0}* verwendet.

Gültigkeit beliebig außer *object*

Tabelle 9.4 Schlüsselwörter für PDF_info_image( )

Schlüsselwort	Beschreibung
clippingpath	*Pfad-Handle des Bild-Beschneidungspfads oder -1 (in PHP: 0), falls kein Beschneidungspfad vorhanden ist*
checkcolorspace	*1, falls das Bild oder Template ohne farbbezogene Verletzung des PDF/A- oder PDF/X-Standards auf der aktuellen Seite platziert werden kann; sonst 0. Bei dieser Prüfung wird der Default-Farbraum der Seite berücksichtigt, der beim Laden des Bilds nicht geprüft wird.*
filename	*String-Index für den Namen der Bilddatei (gegebenenfalls einschließlich des Suchpfades) oder -1 für Templates*
iccprofile	*Handle für das im Bild eingebettete ICC-Profil oder -1 (in PHP: 0), falls kein Profil vorhanden ist*
imageheight	*Bilder: Höhe in Pixeln* *Templates: benutzerdefinierte Höhe oder automatisch bestimmte Höhe für die Option* `reference`
imagemask	*Image-Handle auf die dem Bild zugeordnete Maske oder -1 (in PHP: 0), falls keine Maske zugeordnet ist*

Tabelle 9.4 Schlüsselwörter für PDF_info_image( )

Schlüsselwort	Beschreibung
imagetype	*String-Index für das Bildformat:* `bmp`, `ccitt`, `gif`, `jbig2`, `jpeg`, `jpeg2000`, `png`, `raw`, `tiff` *für Rasterbilder. Wurde das zum übergebenen Handle gehörige Objekt mit PDF_begin_template_ext( ) erzeugt, wird als String* `template` *zurückgegeben.*
imagewidth	*Bilder: Breite in Pixeln* *Templates: benutzerdefinierte Breite oder automatisch bestimmte Breite für die Option* `reference`
infomode	*1, wenn das Bild mit der Option* `infomode` *geladen wurde, sonst 0*
mirroringx, mirroringy	*Horizontale oder vertikale Spiegelung des Bildes gemäß der übergebenen Optionen (als 1 oder -1 ausgegeben)*
orientation	*Wert für die Orientierung des Bildes. Bei TIFF-Bildern mit Orientation-Tag wird der Wert dieses Tags zurückgegeben; anderenfalls wird 1 zurückgegeben. PDFlib kompensiert automatisch alle Orientation-Werte ungleich 1.*
resx, resy	*Horizontale bzw. vertikale Auflösung des Bilds. Ist der Rückgabewert positiv, so enthält er die Bildauflösung in Pixeln pro Zoll (dpi). Ist der Rückgabewert 0, so ist die Bildauflösung unbekannt. Ist der Rückgabewert negativ, kann über beide Werte zusammen das Seitenverhältnis nicht-quadratischer Pixel ermittelt werden; ihre absoluten Werte sind dann aber bedeutungslos.*
strips	*Anzahl der Bildstreifen (kann nur bei bestimmten mehrstreifigen TIFF-Bildern von 1 verschieden sein)*
transparent	*1, wenn das Bild Transparenz enthält (Alphakanal mit mehr als 1 Bit), sonst 0. Wurde ein Alphakanal aus der originalen Bilddatei eingelesen oder das Bild mit der Option* `mask` *geladen, wird von Transparenz ausgegangen.*
xid	*(Nur bei PDF/VT) String-Index für den Eintrag* `GTS_XID` *des Bilds oder Templates oder -1, falls kein* `GTS_XID`*-Wert zugewiesen wurde. Der String* `GTS_XID` *kann in der Metadaten-Eigenschaft* `CIP4/Summary/Content/Referenced` *für DPM verwendet werden.*

Tabelle 9.5 Option für PDF_info_image( )

Option	Beschreibung
useembedded-image	*(Boolean; wird bei* `createtemplate=false` *ignoriert) Bei* `true` *werden Angaben zu dem im Template eingebettete Bild zurückgeliefert, ansonsten Angaben zum Template selbst. Bei* `useembeddedimage=true` *sind einige Schlüsselwörter nur für das Originalbild, aber nicht für das generierte Template relevant (z.B.* `dpi`*). Insbesondere sollten die Werte nicht für die Platzierung des Templates verwendet werden, das für das Bild erstellt wurde. Standardwert:* `false`

9.2 SVG-Grafik

Cookbook *Ein vollständiges Codebeispiel hierzu finden Sie im Cookbook-Topic* `graphics/starter_svg`.

C++ Java C# ***int load_graphics(String type, String filename, String optlist)***

Perl PHP ***int load_graphics(string type, string filename, string optlist)***

C ***int PDF_load_graphics(PDF *p, const char *type, const char *filename, int len, const char *optlist)***

Öffnet eine auf der Festplatte liegende oder virtuelle Vektorgrafikdatei unter Anwendung verschiedener Optionen.

type Typ der Vektorgrafikdatei. Mit dem Schlüsselwort *auto* kann der Dateityp automatisch ermittelt werden. Es ist äquivalent zu *svg*, was SVG-Grafik bedeutet.

filename (Name-String; wird gemäß der globalen Option *filenamehandling* interpretiert, siehe Tabelle 2.3) Name der Grafikdatei, die geöffnet werden soll. Es muss sich um eine Datei auf der Festplatte oder um eine virtuelle Datei handeln, da PDFlib keine Grafikdaten von URLs bezieht.

Wird keine Datei mit dem angegebenen Namen gefunden, versucht PDFlib, die Dateinamenserweiterung zu ermitteln; alle in der folgenden Liste enthaltenen Erweiterungen werden versuchsweise (in Klein- und in Großbuchstaben) an den in *filename* übergebenen Dateinamen angehängt und eine Datei dieses Namens in den im Searchpath enthaltenen Verzeichnissen gesucht:

```
.svg, .svgz
```

len (Nur C-Sprachbindung) Länge of *filename* (in Bytes). Bei *len* = *0* muss ein null-terminierter String übergeben werden.

optlist Optionsliste mit Optionen für Grafikeigenschaften. Folgende Optionen können verwendet werden:

- Allgemeine Optionen: *errorpolicy* (siehe Tabelle 2.1) und *hypertextencoding* (siehe Tabelle 2.3)
- Fontoptionen gemäß Tabelle 9.6:
  defaultfontfamily, defaultfontoptions, fallbackfontfamily, fallbackfontoptions
- Größenoptionen gemäß Tabelle 9.6:
  fallbackheight, fallbackwidth, forcedheight, forcedwidth
- Bildoptionen gemäß Tabelle 9.6: *defaultimageoptions, fallbackimage*
- Weitere Optionen zur SVG-Verarbeitung gemäß Tabelle 9.6: *errorconditions, lang*
- Bestimmte PDF-Features gemäß Tabelle 9.6: *devicergb, templateoptions*

Rückgabe Grafik-Handle, das in nachfolgenden Aufrufen für Grafikfunktionen verwendet werden kann. Bei *errorpolicy=return* muss der Anwendungsentwickler den Rückgabewert auf -1 (in PHP: 0) überprüfen, da dies auf einen Fehler hinweist. Das zurückgegebene Grafik-Handle kann über mehrere PDF-Dokumente hinweg verwendet werden. Scheitert der Funktionsaufruf, so können Sie die Fehlerursache mit *PDF_get_errmsg( )* abfragen.

Details Diese Funktion öffnet und analysiert eine Grafikdatei in einem der unterstützten Dateiformate, das mit dem Parameter *type* angegeben wurde. Die Grafikdaten werden so lange im Speicher gehalten, bis die Grafik mit *PDF_close_graphics( )* geschlossen wird oder bis zum Ende des Lebenszyklus des PDFlib-Objekt. Diese Funktion hat keinen sichtbaren

Einfluss auf die Ausgabe. Um die importierte Grafik tatsächlich im generierten Ausgabedokument zu platzieren, muss *PDF_fit_graphics( )* aufgerufen werden. Eine Grafik sollte für jedes generierte Dokument nur einmal geöffnet werden, da die Grafikdaten sonst mehrfach ins Ausgabedokument kopiert werden.

PDFlib öffnet die Grafikdatei mit dem in *filename* übergebenen Namen, verarbeitet den Inhalt und schließt sie wieder, bevor dieser Funktionsaufruf zurückkehrt.

Fonteinbettung (besonders relevant bei PDF/A, PDF/X und PDF/UA): für alle in der Grafik verwendeten Fonts (oder geeignete Standardfonts) müssen Font-Outlinedateien konfiguriert werden. Dies lässt sich mit der Option *enumeratefonts* leicht bewerkstelligen (siehe Abschnitt 2.3, »Globale Optionen«, Seite 25).

PDF/A Alle erforderlichen Fonts müssen eingebettet werden (siehe oben). Die Option *devicergb* ist nur erlaubt, wenn die Option *defaultrgb* von *PDF_begin_page_ext( )* gesetzt ist oder wenn es sich bei der Druckausgabebedingung um ein RGB-Gerät handelt.
PDF/A-1: Grafiken mit Transparenz sind nicht erlaubt.
PDF/A-2a/3a: enthält die Grafik Text mit PUA-Zeichen, muss die Option *tag* mit einer geeigneten Option *ActualText* übergeben werden.

PDF/X Alle erforderlichen Fonts müssen eingebettet werden (siehe oben).
PDF/X-1a: diese Funktion darf nicht aufgerufen werden.
PDF/X-3: Grafiken mit Transparenz sind nicht erlaubt.
PDF/X-3/4/5: die Option *devicergb* ist nur erlaubt, wenn die Option *defaultrgb* von *PDF_begin_page_ext( )* gesetzt ist oder wenn es sich bei der Druckausgabebedingung um ein RGB-Gerät handelt.

PDF/VT Dieser Aufruf kann fehlschlagen, wenn die Option *usestransparency=false* in *PDF_begin_document( )* angegeben wurde, die importierte Grafik aber dennoch Transparenz enthält.

PDF/UA Grafiken sollten als *Figure* oder *Artifact* ausgezeichnet werden. Alle erforderlichen Fonts müssen eingebettet werden (siehe oben). Enthält die Grafik Text mit PUA-Zeichen, muss die Option *tag* mit der Unteroption *ActualText* übergeben werden.

Gültigkeit beliebig; im Gültigkeitsbereich *object* nicht erlaubt, falls *templateoptions* angegeben wurde

Tabelle 9.6 Optionen für PDF_load_graphics( )

Option	Beschreibung
defaultfont-family	*(Name-String) Name der Fontfamilie, die verwendet wird, wenn ein Font für einen Text in der Grafikdatei entweder nicht angegeben oder nicht verfügbar ist. Standardwert:* `Arial Unicode MS` *sofern vorhanden, sonst* `Helvetica`
defaultfont-options	*(Optionsliste) Optionen zum Laden von Fonts gemäß Tabelle 4.2. Wird für Text in einer Grafikdatei ein Font benötigt, der noch nicht geladen wurde, werden die hier angegebenen Optionen an PDF_load_font( ) übergeben. Standardwert:* `{subsetting embedding skipembedding={latincore standardcjk} }`
defaultimage-options	*(Optionsliste) Optionen zum Laden von Bildern gemäß Tabelle 9.1. Bei der Verarbeitung von eingebetteten oder externen Bildern werden die hier angegebenen Optionen an PDF_load_image( ) übergeben. Standardwert:* `{ }`

Tabelle 9.6 Optionen für PDF_load_graphics( )

Option	Beschreibung
devicergb	*(Boolean; bei PDF/A und PDF/X-3/4/5 ist diese Option nur erlaubt, wenn die Option* defaultrgb *an PDF_begin_page_ext( ) übergeben wurde oder wenn es sich bei der Druckausgabebedingung um ein RGB-Gerät handelt) Bei* true *werden Grafik- und Textfarben in SVG im DeviceRGB-Farbraum und nicht im sRGB-Farbraum interpretiert. Eingebettete Rasterbilder werden mit* honoriccprofile=false *verarbeitet. Standardwert:* false
error-conditions	*(Optionsliste) Liste mit Bedingungen, die einen Fehler auslösen (Standardwert: leere Liste):* **attributes** *(String-Liste) Ein Fehler tritt auf, wenn eins der angegebenen SVG-Attribute zwar in der Grafik vorhanden, aber nicht von PDFlib unterstützt wird (siehe PDFlib-Tutorial für eine Liste der unterstützten Attribute). Standardmäßig werden nicht unterstützte Attribute ignoriert.* **elements** *(String-Liste) Ein Fehler tritt auf, wenn eins der angegebenen SVG-Elemente zwar in der Grafik vorhanden, aber nicht von PDFlib unterstützt wird (siehe PDFlib-Tutorial für eine Liste der unterstützten Elemente). Standardmäßig werden nicht unterstützte Elemente ignoriert.* **references** *(Liste mit Schlüsselwörtern) Ein Fehler tritt auf, wenn einer der folgenden Verweistypen nicht aufgelöst oder ausgeführt werden kann (standardmäßig werden alle Typen außer* image *ignoriert und eine Warnung wird ausgegeben):* **image** *Verweis auf eine Bild- bzw. Grafikdatei; für das Standardverhalten siehe Option* fallbackimage **internal** *interner Verweis auf ein SVG-Element* **external** *Verweis auf andere Dateitypen als Bild- oder Grafikdateien* **fontfamily** *Verweis auf eine Fontfamilie* **font** *Verweis auf einen vollständigen Fontnamen mit Angabe von Fontfamilie, Gewicht und Stil*
fallback-fontfamily	*(Name-String) Name der Fontfamilie für die Erzeugung des Fallback-Fonts für jeden Font, zusätzlich zu den eventuell bereits in der Grafikdatei angegebenen Fallback-Fonts. Standardwert: leer*
fallback-fontoptions	*(Optionsliste) Optionen, die auf die mit der Option* fallbackfontfamily *erzeugten Fallback-Fonts angewendet werden. Die folgenden Optionen können gemäß Tabelle 4.3, Seite 75 verwendet werden:* fontsize, forcechars, textrise. *Standardwert: leere Liste*
fallback-height	*(Float; wird bei* forcedheight *ignoriert) Höhe der SVG-Grafik (in Benutzerkoordinaten) für die Objekteinpassung. Standardwert: im SVG-Attribut* viewBox *übergebene Höhe, wenn vorhanden, sonst 1000*
fallback-image	*(Optionsliste) Legt fest, wie der Platz behandelt werden soll, der für die Fitbox eines Grafik- oder Bildelements reserviert ist, wenn das Element nicht verfügbar ist (Farben werden im sRGB-Farbraum interpretiert, wenn* devicergb=false *und sonst im RGB-Farbraum). Standardwert: ein graues, halbtransparentes Schachbrettmuster wird gezeichnet:* **fillcolor** *(RGB-Farbe oder Schlüsselwort) Füllfarbe für eine Fläche mit Schachbrettmuster (bei* gridsize > 0*), wobei die Hälfte der Quadrate mit der angegebenen Farbe gefüllt wird und die andere Hälfte transparent ist, oder Füllfarbe für die gesamte Fläche bei* gridsize=0*. Bei* none *wird die Fläche nicht ausgefüllt. Standardwert:* LightGrey **gridsize** *(Float oder Prozentwert) Breite eines Schachbrettquadrats in Standardkoordinaten oder als Prozentwert der Breite der Fitbox. Prozentwerte werden so gerundet, dass eine ganzzahlige Anzahl von Quadraten in die Fläche passt. Bei* gridsize=0 *wird die gesamte Fläche mit der Farbe gefüllt, wodurch das Schachbrettmuster verschwindet. Standardwert: 10* **image** *(Image- oder Template-Handle) Image oder Template, das mit* fitmethod=entire *in der Fitbox platziert wird. Standardwert: kein Bild oder Template* **opacity** *(Float zwischen 0...1) Transparenz der Schachbrettquadrate oder der Innenfläche. Standardwert: 0,5* **strokecolor** *(RGB-Farbe oder Schlüsselwort) Linienfarbe für die Umrandung der Fläche oder der einzelnen Quadrate. Bei* none *wird keinerlei Umrandung gezeichnet. Standardwert:* Red
fallback-width	*(Float; wird bei* forcedwidth *ignoriert) Breite der SVG-Grafik (in Benutzerkoordinaten) für die Objekteinpassung. Standardwert: im SVG-Attribut* viewBox *übergebene Breite, wenn vorhanden, sonst 1000*

Tabelle 9.6 Optionen für PDF_load_graphics( )

Option	Beschreibung
forcedheight	*(Float) Die Höhe der SVG-Grafik (wenn vorhanden) wird ignoriert und stattdessen der angegebene Wert im Standard-Koordinatensystem verwendet. Standardwert: Höhe der Grafik*
forcedwidth	*(Float) Die Breite der SVG-Grafik (wenn vorhanden) wird ignoriert und stattdessen der angegebene Wert im Standard-Koordinatensystem verwendet. Standardwert: Breite der Grafik*
lang	*(String) Natürliche Sprache für die Grafikdatei, die z.B. in einem SVG-Element vom Typ* `switch` *verwendet werden kann. Das Format der Sprachspezifikation ist mit der Option* `lang` *von PDF_begin_document( ) identisch (siehe Tabelle 3.3). Standardwert: die in der Umgebungsvariable* `LANG` *angegebene Sprachkennung*
template-options	*(Optionsliste) Erzeugt ein Template (PDF Form XObject) gemäß der übergebenen Optionsliste. Diese Option wird empfohlen, wenn die Grafik mehrfach platziert wird oder Template-Features erforderlich sind (z.B. für PDF/VT). Die übergebene Optionsliste (die leer sein kann) wird für PDF_begin_template_ext( ) verwendet. Die folgenden allgemeinen Optionen für XObjects können verwendet werden (siehe Tabelle 9.10):* `associatedfiles`, `iconname`, `layer`, `metadata`, `pdfvt`, `transparencygroup`. *Breite und Höhe des Templates werden anhand der Größe der Grafik berechnet. Das Template wird in PDF_close_graphics( ) oder am Ende des Dokuments in die PDF-Ausgabe geschrieben (letzteres nur, sofern PDF_fit_graphics( ) mindestens einmal für diese Grafikdatei aufgerufen wurde).*

C++ Java C# ***void close_graphics(int graphics)***

Perl PHP ***close_graphics(int graphics)***

C ***void PDF_close_graphics(PDF *p, int graphics)***

Schließt eine Vektorgrafik.

graphics Gültiges Grafik-Handle, das mit *PDF_load_graphics( )* erzeugt wurde.

Details Die zugehörige interne Grafikstruktur von PDFlib wird gelöscht. Wurde die Option *templateoptions* in *PDF_load_graphics( )* angegeben, wird vor dem Schließen der Grafik das entsprechende PDF-Template erzeugt. Wurde die Grafik aus einer Datei geladen, ist die eigentliche Grafikdatei von diesem Aufruf nicht betroffen, da sie bereits am Ende des entsprechenden Aufrufs von *PDF_load_graphics( )* geschlossen wurde. Ein Grafik-Handle kann nach dem Schließen durch diese Funktion nicht mehr verwendet werden.

Gültigkeit beliebig; im Gültigkeitsbereich *object* nicht erlaubt, falls *templateoptions* im entsprechenden Aufruf von *PDF_load_graphics( )* angegeben wurde und die Grafik mindestens einmal platziert wurde; diese Funktion darf nur paarweise mit *PDF_load_graphics( )* aufgerufen werden.

C++ Java C# ***void fit_graphics(int graphics, double x, double y, String optlist)***

Perl PHP ***fit_graphics(int graphics, float x, float y, string optlist)***

C ***void PDF_fit_graphics(PDF *p, int graphics, double x, double y, const char *optlist)***

Platziert eine Vektorgrafik in einem Content-Stream unter Anwendung verschiedener Optionen.

graphics Gültiges Grafik-Handle, das mit *PDF_load_graphics( )* erzeugt wurde.

x, y Koordinaten des Referenzpunkts im Benutzerkoordinatensystem, an dem die Grafik platziert wird.

optlist Optionsliste mit Optionen zur Grafikplatzierung und -verarbeitung. Folgende Optionen werden unterstützt:

- Optionen zur Objekteinpassung gemäß Tabelle 6.1:
  blind, boxsize, fitmethod, matchbox, orientate, position, refpoint, rotate, scale, showborder
- Optionen für die Grafikverarbeitung gemäß Tabelle 9.3: *adjustpage, gstate*
- Option für die Verarbeitung interaktiver Links in Grafiken gemäß Tabelle 9.7: *convertlinks*
- Option zum vereinfachten Tagging von Strukturelementen gemäß Tabelle 14.5 (nur erlaubt im Gültigkeitsbereich *page*): *tag*

Details Die Grafik wird relativ zum Referenzpunkt *(x, y)* platziert. Standardmäßig wird die linke untere Ecke des Objekts am Referenzpunkt platziert. Mit den Optionen *orientate, boxsize, position* und *fitmethod* kann dieses Verhalten geändert werden. Standardmäßig wird eine Grafik gemäß ihrer intern angegebenen Größe skaliert. Mit den Optionen *scale* und *fitmethod* kann dieses Verhalten geändert werden.

Wir empfehlen, vor dem Aufruf von *PDF_fit_graphics( )* mit dem Schlüsselwort *fittingpossible* von *PDF_info_graphics( )* zu prüfen, ob *PDF_fit_graphics( )* erfolgreich sein wird, um im Fehlerfall eine Exception zu vermeiden.

PDF/UA Grafik, die Vektorgrafiken oder Rasterbilder enthält, muss als *Figure* oder *Artifact* ausgezeichnet werden.

Gültigkeit *page, pattern* (wenn für das Pattern *painttype=colored* gesetzt wurde), *template, glyph* (wenn die Option *colorized* des Type-3-Fonts auf *true* gesetzt ist); die Funktion kann beliebig oft auf beliebig vielen Seiten aufgerufen werden, solange das Grafik-Handle nicht mit *PDF_close_graphics( )* geschlossen wird.

Tabelle 9.7 Zusätzliche Option für PDF_fit_graphics( )

Option	Beschreibung
convertlinks	*(Boolean) Bei* `true` *werden interaktive Links in der Grafikdatei in interaktive Anmerkungen vom Typ* `Link` *im PDF konvertiert. Unabhängig von dieser Einstellung werden in den folgenden Fällen keine Links erstellt (Standardwert:* `true`*):* ▸ *die Funktion wird in einem anderen Gültigkeitsbereich als* page *aufgerufen* ▸ *die Option* `templateoptions` *wurde an PDF_load_graphics( ) übergeben* ▸ *Tagged-PDF-Modus: beim aktuell aktiven Strukturelement handelt es sich um ein Artefakt;* ▸ *PDF/X: die Anmerkung vom Typ* `link` *befindet sich innerhalb der BleedBox (oder TrimBox bzw. ArtBox, falls keine BleedBox vorhanden ist).*

C++ Java C# ***double info_graphics(int graphics, String keyword, String optlist)***

Perl PHP ***float info_graphics(int graphics, string keyword, string optlist)***

C ***double PDF_info_graphics(PDF *p, int graphics, const char *keyword, const char *optlist)***

Formatiert eine Vektorgrafik und fragt Metrik und andere Eigenschaften ab.

graphics Gültiges Grafik- oder Template-Handle, das mit *PDF_load_graphics( )* erzeugt wurde.

keyword Schlüsselwort zur Abfrage der gewünschten Information:

- Schlüsselwörter zur Abfrage der Ergebnisse der Objekteinpassung gemäß Tabelle 6.3:
  boundingbox, fitscalex, fitscaley, height, objectheight, objectwidth, width, x1, y1, x2, y2, x3,

y3, x4, y4

- Zusätzliche Schlüsselwörter gemäß Tabelle 9.8:
  description, filename, fittingpossible, graphicswidth, graphicsheight, istemplate, metadata, title, type, xid

optlist Optionsliste mit Optionen für *PDF_fit_graphics( )*. Für den Wert der erforderlichen Schlüsselwörter nicht relevante Optionen werden ignoriert.

Rückgabe Wert einer durch *keyword* festgelegter Grafikeigenschaft. Wird außerhalb einer Seite eine geometrische Eigenschaft abgefragt, gibt die Funktion -1 (in PHP: 0) zurück. Wurde ein Objekt-Handle angefordert, gibt diese Funktion ein Handle für das Objekt zurück oder -1 (in PHP: 0), wenn das Objekt nicht verfügbar ist. Wenn das erforderliche Schlüsselwort Textausgabe produziert, wird ein String-Index zurückgegeben und der zugehörige String muss mit *PDF_get_string( )* abgerufen werden.

Details Diese Funktion führt alle Berechnungen durch, die zur Platzierung der Grafik anhand der übergebenen Optionen erforderlich sind, generiert aber keinerlei Ausgabe auf der Seite. Als Referenzpunkt für die Grafik wird *{0 0}* verwendet.

Gültigkeit beliebig

Tabelle 9.8 Schlüsselwörter für PDF_info_graphics( )

Schlüsselwort	Beschreibung
description	*String-Index für die Inhalte des Elements* `desc` *des äußersten* `svg`*-Elements, sofern vorhanden, oder des äußersten* `g`*-Elements, sofern vorhanden. Ansonsten -1. Der String kann Markup enthalten.*
filename	*String-Index für den Namen der Grafikdatei (einschließlich eines eventuellen Suchpfad-Verzeichnisses)*
fittingpossible	*Prüft, ob die Grafik mit PDF_fit_graphics( ) im aktuellen Kontext platziert werden kann. Der Wert 1 wird zurückgegeben, wenn die Grafik platziert werden kann. Der Wert 0 wird zurückgegeben, wenn die Platzierung aus einem der folgenden Gründe fehlschlägt (das heißt, wenn PDF_fit_graphics( ) eine Exception auslösen würde):* ▸ *internes Problem in der Grafikdatei* ▸ *Konflikt mit den aktuell eingestellten Standards (z.B. keine Transparenz erlaubt, Fonteinbettung erforderlich, aber keine Fontdatei verfügbar)* *Wird 0 zurückgegeben, kann die Ursache für das Problem mit PDF_get_errmsg( ) abgefragt werden. Da das Ergebnis nur im aktuellen Kontext gültig ist, sollte diese Überprüfung unmittelbar vor dem Platzieren der Seite durchgeführt werden.*
graphicswidth, graphicsheight	*Breite und Höhe der Grafik im Standardkoordinatensystem gemäß den Angaben in der Grafikdatei. Sind keine Werte in der Grafikdatei verfügbar, wird 0 zurückgegeben.*
istemplate	*1, wenn die Option* `templateoptions` *übergeben wurde, sonst 0*
metadata	*String-Index für die Inhalte des Elements* `metadata` *des äußersten* `svg`*-Elements, sofern vorhanden, oder des äußersten* `g`*-Elements, sofern vorhanden. Ansonsten -1. Der String kann Markup enthalten.*
title	*String-Index für die Inhalte des Elements* `title` *des äußersten* `svg`*-Elements oder -1, wenn dieses Element nicht vorhanden ist. Der String kann Markup enthalten.*

Tabelle 9.8 Schlüsselwörter für PDF_info_graphics( )

Schlüsselwort	***Beschreibung***
type	*String-Index für das Format der Grafik: immer* svg
xid	*(Nur bei PDF/VT) String-Index für den Eintrag* GTS_XID *des Templates, das für die Grafik erzeugt wurde, oder -1, wenn kein Template erzeugt wurde oder dem Template kein* GTS_XID*-Wert zugewiesen wurde. Der String* GTS_XID *kann in der Metadaten-Eigenschaft* CIP4/Summary/Content/Referenced *für DPM verwendet werden.*

9.3 Templates

Hinweis *Wir verwenden den Begriff »Template« als Synonym für PDF-Form-XObjects. Die in diesem Abschnitt beschriebenen Template-Funktionen beziehen sich nicht auf die variablen Daten in der PDFlib-Blockverarbeitung. Zum Füllen von Blöcken, die mit dem PDFlib Block-Plugin erstellt wurden, verwenden Sie PDF_fill_*block( ) (siehe Kapitel 11, »Funktionen zum Füllen von Blöcken (PPS)«, Seite 219).*

C++ Java C# ***int begin_template_ext(double width, double height, String optlist)***

Perl PHP ***int begin_template_ext(float width, float height, string optlist)***

C ***int PDF_begin_template_ext(PDF *p, double width, double height, const char *optlist)***

Beginnt eine Template-Definition.

width, height Die Größe der Boundingbox des Templates in Punkt. Die Parameter *width* und *height* können 0 sein. In diesem Fall müssen sie in *PDF_end_template_ext( )* übergeben werden. Sofern die Option *postscript* nicht angegeben wurde, müssen beide Werte von 0 verschieden sein.

optlist Optionsliste für die Eigenschaften des Templates:

- Folgende Option von *PDF_begin_page_ext( )* kann verwendet werden (siehe Tabelle 3.9): *topdown*
- Folgende allgemeine Optionen für XObjects können verwendet werden (siehe Tabelle 9.10):
  associatedfiles, iconname, layer, metadata, OPI-1.3, OPI-2.0, pdfvt, reference, transparencygroup
- Folgende Transformationsoption kann verwendet werden (siehe Tabelle 8.3): *transform*
- Folgende Option für das Einbinden von PostScript-Code kann verwendet werden (siehe Tabelle 9.9): *postscript*

Rückgabe Template-Handle, das in nachfolgenden Aufrufen von *PDF_fit_image( )*, *PDF_info_image( )* und *PDF_end_template_ext( )* verwendet werden kann, oder -1 (in PHP: 0) im Fehlerfall.

Details Diese Funktion setzt alle Text-, Grafik- und Farbzustandsparameter auf ihre Standardwerte zurück und richtet ein Koordinatensystem entsprechend der Option *topdown* ein. Hypertext-Funktionen dürfen innerhalb einer Template-Definition nicht verwendet werden. Alle Text-, Grafik- und Farbfunktionen sind hingegen einsetzbar.

Größe des Templates: im einfachsten Fall werden Breite und Höhe in *PDF_begin_template_ext( )* übergeben. Falls diese noch nicht bekannt sind, können sie auch mit Null angegeben werden. In diesem Fall müssen sie im entsprechenden Aufruf von *PDF_end_template_ext( )* übergeben werden.

Wurde die Option *reference* übergeben, wird die Größe automatisch anhand der Größe der PDF-Zielseite ermittelt, und es müssen keine Werte angegeben werden. Werden *width* und *height* jedoch trotzdem angegeben, werden sie verwendet und automatisch an das Seitenverhältnis der Zielseite angepasst.

PDF/A Die Option *postscript* darf nicht verwendet werden.

PDF/X Die Option *postscript* darf nicht verwendet werden.

Gültigkeit beliebig außer *object;* mit dieser Funktion beginnt der Gültigkeitsbereich *template*; diese Funktion muss immer paarweise mit *PDF_end_template_ext( )* aufgerufen werden.

Tabelle 9.9 Option für PDF_begin_template_ext( )

Option	Beschreibung
postscript	*(Optionsliste; nicht für PDF/A und PDF/X) Erzeugt statt eines PDF-Form-XObjects ein PostScript-XObject. Diese Option sollte nur in Szenarien mit genauer Kontrolle über die Nachbearbeitung der erzeugten PDF-Dokumente verwendet werden; für den Import von EPS-Grafiken ist diese Option nicht geeignet. Das PostScript-XObject wird sofort geschrieben. Obwohl sich der Gültigkeitsbereich zu* `template` *ändert, sind Funktionsaufrufe für die Erzeugung von Grafikausgabe bis zum entsprechenden Aufruf von PDF_end_template_ext( ) nicht erlaubt. Wird diese Option übergeben, sind keine weiteren Optionen erlaubt. Unterstützte Unteroption:* **filename** *(Name-String; erforderlich) Name einer auf der Festplatte liegenden oder virtuellen Datei mit PostScript-Code. Der PostScript-Code sollte mit einem Leerzeichen beendet werden. Dieser Code wird ohne Validierung in das PostScript-XObject eingefügt. Für die PostScript-Inhalte sind die Benutzer verantwortlich.*

C++ Java C# **void end_template_ext(double width, double height)**

Perl PHP **end_template_ext(float width, float height)**

C **void PDF_end_template_ext(PDF *p, double width, double height)**

Beendet eine Template-Definition.

width, height Die Abmessungen der Boundingbox des Templates in Punkt. Ist *width* oder *height* gleich 0, wird der in *PDF_begin_template_ext( )* übergebene Wert verwendet. Anderenfalls wird der in *PDF_begin_template_ext( )* übergebene Wert überschrieben. Wurde im entsprechenden Aufruf von *PDF_begin_template_ext( )* jedoch die Option *reference* angegeben, werden die an *PDF_end_template_ext( )* übergebenen Werte ignoriert.

Gültigkeit *template;* mit dieser Funktion endet der Gültigkeitsbereich *template*; diese Funktion muss immer paarweise mit *PDF_begin_template_ext( )* aufgerufen werden.

9.4 Allgemeine XObject-Optionen

PDF-XObjects werden für importierte Bilder, importierte Vektorgrafiken (sofern die Optionsliste *templateoptions* übergeben wurde), importierte PDF-Seiten und Templates erzeugt. Die Optionen in diesem Abschnitt beziehen sich auf die folgenden Funktionen, mit denen XObjects erzeugt werden können:

- *PDF_load_image( )*
- *PDF_load_graphics( )* mit der Option *templateoptions*
- *PDF_open_pdi_page( )*
- *PDF_begin_template_ext( )*

Die folgenden XObject-Optionen können verwendet werden (siehe Tabelle 9.10):
associatedfiles, georeference, iconname, layer, metadata, OPI-1.3, OPI-2.0, pdfvt, reference, transparencygroup

PDF/A Einige Optionen sind eingeschränkt.

PDF/X Einige Optionen sind eingeschränkt. Die Option *reference* ist für PDF/X-5g/5pg relevant.

PDF/VT Die Option *pdfvt* ist für PDF/VT relevant.

Tabelle 9.10 Allgemeine XObject-Optionen für PDF_load_image( ), PDF_open_pdi_page( ) und PDF_begin_template_ext( ) sowie PDF_load_graphics( ) mit der Option `templateoptions`

Option	*Beschreibung*
associatedfiles	*(Liste von Asset-Handles; nur bei PDF 2.0 und PDF/A-3) Asset-Handles für Dateianhänge gemäß PDF/A-3. Die Dateien müssen mit PDF_load_asset( ) und* `type=attachment` *geladen worden sein.*
georeference	*(Optionsliste; PDF 1.7ext3; nur bei PDF_load_image( )) Beschreibung eines geografischen Koordinatensystems für die Nutzung von Geodaten, die dem XObject zugeordnet ist; für weitere Informationen siehe Abschnitt 12.7, »Features für Geodaten«, Seite 258.*
iconname	*(Hypertext-String) Weist dem XObject einen Namen zu, über den es via JavaScript angesteuert werden kann. Dies ist zum Beispiel nützlich, um die Seite einem Formularfeld als Symbol zuzuweisen.* *PDF_load_image( ): diese Option wird ignoriert, wenn* `inline=true`*; erzwingt* `createtemplate=true`
layer	*(Ebenen-Handle; PDF 1.5) Layer, zu dem das XObject gehören soll, sofern vor der Platzierung des Objekts keine weitere Ebene mit PDF_begin_layer( ) aktiviert wurde. Wird PDF_begin_layer( ) zur Aktivierung einer Ebene aufgerufen, bevor das XObject platziert wurde, wird dadurch die Option* `layer` *des Objekts überschrieben. Um eine Überschreibung der Option* `layer` *zu vermeiden, sollten Sie vor der Platzierung des Objekts PDF_end_layer( ) aufrufen.*
metadata	*(Optionsliste) Metadaten für das XObject (siehe Abschnitt 14.2, »XMP-Metadaten«, Seite 277).*
OPI-1.3	*(Optionsliste; nicht bei PDF_open_pdi_page( ); nicht bei PDF/A und PDF/X) Optionsliste mit OPI 2.0-PostScript-Kommentaren als Optionsnamen; folgende Einträge sind erforderlich:* `ALDImageFilename` *(String),* `ALDImageDimensions` *(Integer-Liste),* `ALDImageCropRect` *(Rechteck mit Integers),* `ALDImagePosition` *(Float-Liste)* *Die Unteroption* `normalizefilename` *steuert die Verarbeitung von Dateinamen: bei* `true`*, werden Dateinamen so normalisiert, wie es in der PDF-Referenz vorgeschrieben ist. Bei* `false` *werden sie unverändert in die Ausgabe übernommen. Letzteres kann bei der Arbeit mit bestimmten OPI-Servern nützlich sein, die nicht korrekt mit normalisierten Dateinamen umgehen können. Standardwert:* `false`
OPI-2.0	*(Optionsliste; nicht bei PDF_open_pdi_page( ); nicht bei PDF/A und PDF/X) Optionsliste mit OPI 2.0-PostScript-Kommentaren als Optionsnamen; der folgende Eintrag ist erforderlich:* `ImageFilename` *(String)* *Folgende Einträge sollten entweder gemeinsam vorhanden oder nicht vorhanden sein:* `ImageCropRect` *(Rechteck),* `ImageDimensions` *(Float-Liste)* *Außerdem wird die Unteroption* `normalizefilename` *unterstützt (siehe OPI-1.3).*

Tabelle 9.10 Allgemeine XObject-Optionen für PDF_load_image( ), PDF_open_pdi_page( ) und PDF_begin_template_ext( ) sowie PDF_load_graphics( ) mit der Option `templateoptions`

Option	Beschreibung
pdfvt	*(Optionsliste; nur bei PDF/VT) PDF/VT-Unteroptionen für das XObject gemäß Tabelle 9.12*
reference	*(Optionsliste; nicht bei PDF_load_image( ); benötigt Acrobat 9 oder höher für die korrekte Seitenanzeige; nicht erlaubt bei PDF/A, PDF/X-1/2/3/4, PDF/VT-1 und PDF/UA; in PDFlib-Quellcode-Paketen nicht verfügbar) Referenziert eine Seite in einem externen PDF-Dokument (dem »Ziel«-Dokument). Die mit PDF_open_pdi_page( ) geöffnete Seite oder das mit PDF_begin_template_ext( ) erzeugte Template oder die mit PDF_load_graphics( ) geladene Grafik werden als Proxy für diese Referenz verwendet. Abhängig von der Konfiguration des PDF-Viewers und der Verfügbarkeit des Zieldokuments wird entweder der interne Proxy oder das externe Ziel angezeigt und gedruckt. Für unterstützte Unteroptionen siehe Tabelle 9.11.* *PDF_open_pdi_page( ): Das referenzierte PDF-Dokument muss lokal zur Verfügung stehen und muss die mit den Unteroptionen* `pagelabel` *oder* `pagenumber` *adressierten Seiten enthalten. Für das Zieldokument darf kein Benutzer- oder Master-Kennwort erforderlich sein. Die Größe der referenzierten Seite wird gemäß der Unteroption* `pdiusebox` *der Option* `reference` *bestimmt. Sie kann mit den Schlüsselwörtern* `imagewidth`/`imageheight` *von PDF_info_image( ) abgefragt werden. Die Seitengrößen von Proxy- und Zielseite müssen kompatibel sein, das heißt, die mit der Option* `pdiusebox` *ausgewählten Seitenboxen müssen übereinstimmen, damit beide Seiten an der selben Stelle auf der Seite platziert werden können.* *PDF_begin_template_ext( ): Wurden* `width` *und* `height` *mit dem Wert 0 übergeben, kann die Größe des Templates mit den Schlüsselwörtern* `imagewidth`/`imageheight` *von PDF_info_image( ) abgefragt werden. Wurden* `width` *und* `height` *mit Werten ungleich 0 übergeben, können auch folgende Unteroptionen verwendet werden (siehe Tabelle 6.1):* `fitmethod`, `position`. *PDF_load_graphics( ): die Grafik wird an die Größe der Zielseite angepasst; es können auch folgende Unteroptionen verwendet werden (siehe Tabelle 6.1):* `fitmethod`, `position`. *PDF/X-5g/5pg: das Ziel muss zu einem der folgenden Standards konform sein: PDF/X-1a:2003, PDF/X-3:2003, PDF/X-4, PDF/X-4p, PDF/X-5g oder PDF/X-5pg. Es muss für dieselbe Druckausgabebedingung vorbereitet worden sein.* *PDF/VT-2: das Ziel muss zu einem der folgenden Standards konform sein: PDF/X-1a:2003, PDF/X-3:2003, PDF/X-4, PDF/X-4p oder PDF/VT-1. Es muss für dieselbe Druckausgabebedingung vorbereitet worden sein.* *Für erforderliche Acrobat-Konfiguration zur Anzeige referenzierter Seiten siehe das PDFlib-Tutorial.*
transparency group	*(Optionsliste oder Schlüsselwort; nicht bei PDF_load_image( ); nicht bei PDF/A-1 und PDF/X-1/3; bei PDF/A-2/3 and PDF/X-4/5 bestehen Einschränkungen) Erzeugt eine Transparenzgruppe für die importierte Seite oder Grafik oder das generierte Template. Folgende Schlüsselwörter werden unterstützt (Standardwert:* `auto`*):* **auto** *Enthält eine importierte Seite eine Transparenzgruppe, wird sie in das erzeugte Form XObject kopiert. Wenn dies jedoch zu Konflikten mit einem CMYK-basierten ICC-Profil in einer PDF/X-4-Druckausgabebedingung führt, wird sie ignoriert (in diesem Fall ist auch keine Transparenzgruppe erforderlich).* *Für Grafiken und Templates wird keine Transparenzgruppe erzeugt.* **none** *Es wird keine Transparenzgruppe erzeugt, auch nicht für eine importierte Seite, die bereits eine solche enthält.* *Mit folgenden Unteroptionen kann explizit eine Transparenzgruppe erzeugt werden:* **colorspace** *(Schlüsselwort oder ICC-Profil-Handle; erforderlich bei einem Wert ungleich* `none`*, falls das Template für die Option* `softmask` *von PDF_create_gstate( ) mit* `type=luminosity` *verwendet wird) Farbmischraum (Standardwert:* `none`*):* **DeviceCMYK** *PDF/A-2/3 und PDF/X-4/5: nur erlaubt mit einer CMYK-Druckausgabebedingung* **DeviceGray** *PDF/A-2/3 und PDF/X-4/5: nur erlaubt mit einer Graustufen- oder CMYK-Druckausgabebedingung* **DeviceRGB** *PDF/A-2/3 und PDF/X-4/5: nur erlaubt mit einer RGB-Druckausgabebedingung* **none** *Für die Transparenzgruppe wird kein Farbraum ausgegeben.* **srgb** *Schlüsselwort für die Auswahl des sRGB-Farbraums* **isolated** *(Boolean) Legt fest, ob die Transparenzgruppe isoliert ist. Standardwert:* `false` **knockout** *(Boolean) Legt fest, ob die Transparenzgruppe eine Knockout-Gruppe ist. Standardwert: false*

Tabelle 9.11 Unteroptionen für die Option `reference` *von PDF_begin_template_ext( ), PDF_open_pdi_page( ) sowie PDF_load_graphics( ) mit der Option* `templateoptions`

Option	Beschreibung
filename	*(Name-String; erforderlich) Name der Datei mit dem Zieldokument. Der Name wird im PDF-Dokument gespeichert und vom PDF-Viewer verwendet. Er wird auch verwendet, um das Zieldokument auf der Festplatte zu finden (das PDF-Dokument muss also vorhanden sein), sofern die Option* `target` *nicht angegeben wurde. Sie sollten einfache Namen ohne Verzeichnisangaben verwenden.*
hypertextencoding	*(Schlüsselwort) Legt das Encoding für die Option* `pagelabel` *fest. Ein leerer String ist äquivalent zu* `unicode`*. Standardwert: Wert der globalen Option* `hypertextencoding`
pagelabel	*(Hypertext-String; darf nicht mit* `pagenumber` *kombiniert werden) Seiten-Label der zu referenzierenden Seite*
pagenumber	*(Integer) Nummer der referenzierten Seite. Die erste Seite hat die Nummer 1. Standardwert: 1 (dies kann mit* `pagelabel` *überschrieben werden).*
pdiusebox	*(Schlüsselwort; wird bei PDF/X-5g/5pg auf* `media` *gesetzt) Legt fest, welche Box-Abmessungen zur Bestimmung der Größe der Zielseite verwendet werden. Standardwert:* `media` *im Modus PDF/X-5g/5pg, sonst* `crop`*.* ***media*** *MediaBox wird verwendet (sie ist immer vorhanden)* ***crop*** *CropBox wird verwendet, wenn vorhanden, sonst MediaBox* ***bleed*** *BleedBox wird verwendet, wenn vorhanden, sonst CropBox* ***trim*** *TrimBox wird verwendet, wenn vorhanden, sonst CropBox* ***art*** *ArtBox wird verwendet, wenn vorhanden, sonst CropBox*
strongref	*(Boolean; wird bei PDF/X-5g/5pg auf* `true` *gesetzt) Bei* `true` *erzeugt PDFlib über den ID-Eintrag des Zieldokuments einen starken Verweis auf das Zieldokument. Der Verweis wird fehlerhaft (das heißt, der PDF-Viewer verwendet den Proxy) wenn das Zieldokument durch ein anderes Dokument ersetzt wird. Ist der Austausch von Zieldokumenten geplant, muss diese Option auf* `false` *gesetzt werden, und das lokale Ziel und das Ziel, das letztlich für die Darstellung des Dokuments verwendet wird, müssen identische Seitenboxen und Rotationswerte haben. Standardwert:* `true`
target	*(Handle für PDF-Dokument) Handle auf das Zieldokument, das mit PDF_open_pdi_document( ) erzeugt wurde. Das PDF-Zieldokument muss mit der Option* `repair=none` *und ohne die Option* `password` *geöffnet worden sein. Ein Dokument-Handle in Kombination mit dem Dateinamen anzugeben, kann in folgenden Situationen sinnvoll sein:* ► *Wenn viele generierte Dokumente das selbe PDF-Zieldokument referenzieren, muss das Zieldokument nur einmal geöffnet werden, und die Ergebnisse können intern gecached werden.* ► *Der Dateiname des lokalen Zieldokuments weicht vom Namen ab, der im PDF-Dokument für das Zieldokument gespeichert ist.*

Tabelle 9.12 Unteroptionen für die Option pdfvt *von* PDF_load_image(), PDF_open_pdi_page(), PDF_begin_template_ext() *sowie* PDF_load_graphics() *mit der Option* templateoptions

Option	Beschreibung
environment	*(Hypertext-String; erforderlich bei* scope=stream *oder* scope=global*) Legt einen PDF/VT-Umgebungskontext fest, das heißt einen Identifikator, der von einem PDF/VT-Prozessor verwendet werden kann, um eine Schnittstelle für die Verwaltung zugehöriger XObjects bereitzustellen. Beispielsweise könnten Kundenname oder Auftragsbezeichnung hier verwendet werden.*
scope	*(Schlüsselwort) PDF/VT-Gültigkeitsbereich des XObjects (nicht zu verwechseln mit dem Gültigkeitsbereich einer PDFlib-Funktion) (Standardwert:* unknown*):* **unknown** *Der Gültigkeitsbereich des XObjects ist unbekannt.* **singleuse** *Das XObject wird nur einmal in der PDF/VT-Datei referenziert.* **record** *(Nur erlaubt, wenn die Option* recordlevel *von* PDF_begin_document() *angegeben wurde) Das XObject wird auf den Seiten, die zu einem Record gehören, mehrfach referenziert, wird aber nicht innerhalb anderer Records referenziert.* **file** *Das XObject wird mehrfach in der PDF/VT-Datei referenziert. Wurde die Option* recordlevel *angegeben, sollte* scope=file *nur verwendet werden, wenn das XObject in mehr als einem Record referenziert wird (und sonst* scope=record*).* **stream** *(Erfordert die Option* environment*; nur erlaubt bei Dokumenten innerhalb eines PDF/VT-2s-Streams) Das XObject oder ein äquivalentes XObject wird mehrfach in einem PDF/VT-2s-Stream referenziert, in dem die PDF/VT-Datei liegt.* **global** *(Erfordert die Option* environment*) Das XObject oder ein äquivalentes XObject wird in mehr als einer PDF/VT-Datei oder einem PDF/VT-2s-Stream referenziert.*
xid	*(String; nur bei* PDF_begin_template_ext()*, da für alle anderen Typen von XObjects automatisch Identifikatoren erstellt werden) Eindeutiger Identifikator für das Form XObject, das für das Template erstellt wurde. Dieser Identifikator sollte unbedingt wie im Format von ISO 16612-2:2010, Abschnitt 6.7.2 angegeben werden, also ein URI mit dem Schema* uuid *und einer 128-Bit-Zahl gemäß RFC 4122. Für Template-Definitionen, die äquivalente Form-XObjects gemäß PDF/VT erzeugen (das heißt Templates, die die gleiche grafische Ausgabe erzeugen), sollten die Identifikatoren identisch sein. Nicht äquivalente Templates müssen einen anderen oder gar keinen Identifikator haben.* *Wir empfehlen dringend, für Templates die Option* xid *mit* scope=stream *oder* scope=global *zu übergeben, damit Form XObjects über Dokumentgrenzen hinweg gecached werden können.* *Beispiel für* xid *im empfohlenen Format:* uuid:1228c416-48f2-e817-ad69-8206e41dca2d

10 PDF-Import- (PDI) und pCOS-Funktionen

Hinweis *Alle in diesem Kapitel beschriebenen Funktionen benötigen die PDF-Importbibliothek PDI, die Bestandteil von PDFlib+PDI und PDFlib Personalization Server (PPS) ist, aber nicht in PDFlib enthalten ist. Bitte informieren Sie sich zum Erwerb von PDI gegebenenfalls auf unserer Webseite.*

10.1 Dokumentfunktionen

Cookbook *Ein vollständiges Codebeispiel finden Sie im Cookbook-Topic* `pdf_import/starter_pdfmerge`.

C++ Java C# ***int open_pdi_document(String filename, String optlist)***

Perl PHP ***int open_pdi_document(string filename, string optlist)***

C ***int PDF_open_pdi_document(PDF *p, const char *filename, int len, const char *optlist)***

Öffnet ein auf der Festplatte oder in einer PVF-Datei liegendes PDF-Dokument und bereitet es zur späteren Verwendung vor.

filename (Name-String; wird gemäß der globalen Option *filenamehandling* interpretiert, siehe Tabelle 2.3) Name der PDF-Datei.

optlist Optionsliste mit Optionen zum Öffnen des PDF-Dokuments:

- Allgemeine Option: *errorpolicy* (siehe Tabelle 2.1)
- PDF-Dokumentoptionen gemäß Tabelle 10.1:
  checkoutputintentprofile, infomode, inmemory, password, repair, requiredmode, shrug
- Optionen zur Verarbeitung von Tagged PDF gemäß Tabelle 10.1:
  checktags, usetags
- Optionen zur Verarbeitung von Ebenen gemäß Tabelle 10.1:
  parentlayer, parenttitle, uselayers

len (Nur C-Sprachbindung) Länge von *filename* (in Bytes). Ist *len* = 0, muss ein null-terminierter String übergeben werden.

Rückgabe PDI-Dokument-Handle, das zur Verarbeitung einzelner Seiten des Dokuments oder zur Abfrage von Dokumenteigenschaften verwendet werden kann. Ein Rückgabewert von -1 (in PHP: 0) zeigt an, dass das PDF-Dokument nicht geöffnet werden konnte. Es kann eine beliebige Anzahl von PDF-Dokumenten gleichzeitig geöffnet werden. Der Rückgabewert kann bis zum Ende des umgebenden Gültigkeitsbereichs *document* verwendet werden. Scheitert der Funktionsaufruf, so kann die Fehlerursache mit *PDF_get_errmsg( )* abgefragt werden.

Das Verhalten im Fehlerfall kann mit der Option *errorpolicy* geändert werden.

Details Das Dokument wird standardmäßig nicht akzeptiert, wenn eine oder mehrere der folgenden Bedingungen zutreffen:

- Das Dokument ist beschädigt und konnte nicht repariert werden (oder *repair=none* wurde angegeben).
- Das Dokument ist verschlüsselt und es wurde kein passendes Master-Kennwort in der Option *password* übergeben. Mit der Option *shrug* kann unter bestimmten Bedin-

gungen der Seitenimport aus geschützten Dokumenten aktiviert werden (siehe PDFlib-Tutorial).

Außer bei der ersten Bedingung kann das Dokument mit der Option *infomode* in jedem Fall geöffnet werden. Dies kann nützlich sein, um mit den Funktionen *PDF_pcos_get_*()* Informationen über das PDF-Dokument abzufragen, zum Beispiel über Verschlüsselung, Dokument-Infofelder usw.

Mit der Funktion *PDF_get_errmsg()* erhalten Sie in Form einer detaillierten Fehlermeldung genauere Informationen über die Art des mit dem importierten PDF zusammenhängenden Problems (PDF-Dateiname falsch, fehlerhafte PDF-Daten usw.).

Gültigkeit beliebig; im Gültigkeitsbereich *object* kann ein PDI-Dokument-Handle nur in den Funktionen *PDF_pcos_get_*()* verwendet werden.

Tabelle 10.1 Optionen für PDF_open_pdi_document()

Option	Beschreibung
checktags	*(Schlüsselwort) Legt fest, ob die Verschachtelungsregeln für Strukturelemente (siehe PDFlib-Tutorial) in PDF_open_pdi_page() auch auf importierte Strukturelemente angewendet werden. Unterstützte Schlüsselwörter (Standardwert:* `none`*):* **none** *Verschachtelungsregeln für Tags werden nicht überprüft. Dies ist die Standardeinstellung, da viele Dokumente aus der realen Welt diese Regeln verletzen und sonst nicht importiert werden könnten.* **relaxed** *Ähnlich wie* `strict`*, es werden aber einige Regeln nicht überprüft (siehe PDFlib-Tutorial).* **strict** *Verletzt ein importiertes Tag die Verschachtelungsregeln, schlägt der Aufruf von PDF_open_pdi_page() fehl.*
checkoutput-intentprofile	*(Boolean, nur relevant für PDF/A und PDF/X) Bei* `true` *wird die Anzahl der Farbkomponenten der Druckausgabebedingung mit der Anzahl der Komponenten im zugehörigen ICC-Profil verglichen. Damit lassen sich inkonsistente Eingabedokumente vermeiden. Wird diese Option auf* `false` *gesetzt, reduziert sich der erforderliche Speicherbedarf. Diese Einstellung sollte aber nur verwendet werden, wenn die Eingabedokumente garantiert konsistente Druckausgabeprofile enthalten. Standardwert:* `true`
infomode	*(Boolean) Bei* `true` *wird das Dokument so geöffnet, dass Informationen mit der pCOS-Schnittstelle abfragbar sind, die Seiten sich aber nicht mit PDF_open_pdi_page() in das Ausgabedokument importieren lassen. Folgende Dokumenttypen lassen sich mit* `infomode=true` *öffnen: verschlüsselte PDF-Dokumente mit unbekanntem Kennwort (Ausnahme: PDF 1.6-Dokumente, die mit der Distiller-Einstellung »Komprimierung auf Objektebene: Maximum« erstellt wurden). Standardwert:* `false`*, wenn* `requiredmode=full`*, sonst* `true`
inmemory	*(Boolean) Bei* `true` *lädt PDI die Datei vollständig in den Speicher und verarbeitet sie dort. Auf manchen Systemen (insbesondere z/OS) lässt sich die Leistung damit erheblich steigern, es ist jedoch mehr Speicher erforderlich. Bei* `false` *wird das Dokument stückweise von der Festplatte gelesen. Standardwert:* `false`
parentlayer	*(Ebenen-Handle; wird ignoriert, wenn das Eingabedokument keine Ebenen enthält oder bei* `uselayers=false`*) Fügt alle aus dem Dokument importierten Ebenendefinitionen als untergeordnete Elemente in die angegebene Ebene ein. Wurde die angegebene Ebene im Ausgabedokument aktiviert, wird sie als übergeordnete Ebene verwendet; anderenfalls wird sie nur als Titel (Separator) verwendet. Standardwert: kein übergeordneter Titel*
parenttitle	*(Hypertext-String; wird ignoriert, wenn das Eingabedokument keine Ebenen enthält oder bei* `uselayers=false`*) Fügt eine Titel-Ebene ein, die zwar die Sichtbarkeit von Seitenelementen nicht direkt steuert, aber als hierarchischer Separator für die importierten Ebenendefinitionen dient. Standardwert: kein übergeordneter Titel*

Tabelle 10.1 Optionen für PDF_open_pdi_document( )

Option	Beschreibung
password	*(String) Master-Kennwort, das zum Öffnen eines geschützten PDF-Dokuments erforderlich ist. Bei* `infomode=true` *reicht das Benutzerkennwort (das auch leer sein kann) zur Abfrage von Dokumentinformationen aus. Wurde für ein verschlüsseltes Dokument kein Kennwort übergeben, lässt sich mit dem Dokument-Handle nur der Verschlüsselungszustand abfragen. Mit der Option* `shrug` *kann der Seitenimport aus geschützten Dokumenten unter bestimmten Bedingungen aktiviert werden (siehe PDFlib-Tutorial).*
repair	*(Schlüsselwort) Legt fest, wie beschädigte PDF-Eingabedokumente behandelt werden. Die Reparatur eines Dokuments benötigt zwar mehr Zeit als das normale Parsen, ermöglicht aber die Verarbeitung bestimmter beschädigter PDFs. Beachten Sie, dass sich manche Dokumente trotz Reparatur nicht verarbeiten lassen. Unterstützte Schlüsselwörter (Standardwert:* `auto`*):* ***auto*** *Es wird nur versucht, das Dokument zu reparieren, wenn beim Öffnen Probleme auftreten.* ***force*** *Es wird in jedem Fall versucht, das Dokument zu reparieren.* ***none*** *Es wird keinesfalls versucht, das Dokument zu reparieren; bei Problemen im PDF-Dokument scheitert der Funktionsaufruf.*
requiredmode	*(Schlüsselwort) Minimaler pCOS-Modus* (`minimum`/`restricted`/`full`)*, der beim Öffnen akzeptiert wird. Der Funktionsaufruf scheitert, wenn der resultierende pCOS-Modus niedriger als der erforderliche Modus ist. Ist der Aufruf erfolgreich, so ist sichergestellt, dass der resultierende Modus mindestens dem in dieser Option festgelegten Modus entspricht. Er kann jedoch auch höher sein; so ergibt sich zum Beispiel aus* `requiredmode=minimum` *für ein unverschlüsseltes Dokument der Modus* `full`*. Standardwert:* `full`
shrug	*(Boolean) Bei* `true` *wird das Feature »shrug« für den Seitenimport aus geschützten Dokumenten unter bestimmten Bedingungen aktiviert (siehe PDFlib-Tutorial). Mit der Option* `shrug` *versichern Sie, dass Sie die Urheberrechte des Dokuments beachten. Standardwert:* `false`
uselayers	*(Boolean; nur relevant bei einem Eingabedokument mit Ebenen) Bei* `true` *werden alle Ebenendefinitionen von den importierten Seiten importiert. Diese Option wirkt sich nur auf die Ebenendefinitionen, nicht aber auf die eigentlichen Inhalte der Ebenen aus, da PDI immer die Inhalte aller Ebenen auf einer Seite importiert. Standardwert:* `true`
usetags	*(Boolean; nur relevant bei Eingabedokumenten vom Typ Tagged PDF; muss im PDF/UA-Modus auf* `true` *gesetzt werden) Bei* `true` *wird die Strukturhierarchie des importierten Dokuments eingelesen, so dass Strukturelement-Tags später zusammen mit den Seiten importiert werden können. Standardwert:* `true`

C ***int PDF_open_pdi_callback(PDF *p, void *opaque, size_t filesize,***
size_t (*readproc)(void *opaque, void *buffer, size_t size),
int (*seekproc)(void *opaque, long offset), const char *optlist)

Öffnet ein vorhandenes PDF-Dokument aus einer benutzerdefinierten Datenquelle und bereitet es zur späteren Verwendung vor.

opaque Zeiger auf Benutzerdaten, die dem zu öffnenden PDF-Dokument zugeordnet sind. Er wird als erster Parameter an die Callback-Funktionen übergeben und kann beliebig genutzt werden. PDI verwendet diesen Zeiger selbst nicht.

filesize Größe des vollständigen PDF-Dokuments in Bytes.

readproc Callback-Funktion, die *size* Bytes in den als *buffer* angegebenen Speicher kopiert. Ist das Ende des Dokuments früher erreicht, kann die Funktion entsprechend weniger Bytes kopieren. Die Funktion muss die Anzahl der tatsächlich kopierten Bytes zurückgeben.

seekproc Callback-Funktion, die die aktuelle Leseposition im Dokument setzt. *offset* bezeichnet die Position ausgehend vom Dokumentanfang (0 bezeichnet dabei das erste Byte). Bei Erfolg muss die Funktion 0 zurückgeben, anderenfalls -1.

optlist Optionsliste mit Optionen zum Öffnen des PDF-Dokuments; alle Optionen von *PDF_open_pdi_document( )* können verwendet werden.

Rückgabe PDI-Dokument-Handle, das zur Verarbeitung einzelner Dokumentseiten oder zur Abfrage von Dokumenteigenschaften verwendet werden kann. Der Rückgabewert -1 zeigt an, dass das PDF-Dokument nicht geöffnet werden konnte. Es können gleichzeitig beliebig viele PDF-Dokumente geöffnet werden. Der Rückgabewert kann bis zum Ende des umgebenden Gültigkeitsbereichs *document* verwendet werden. Scheitert der Funktionsaufruf, so kann die Fehlerursache mit *PDF_get_errmsg( )* abgefragt werden.

Details Diese Funktion bietet eine besondere Schnittstelle für Applikationen, die Teile einer PDF-Datei aus einer Datenquelle beziehen, statt das PDF-Dokument auf der Festplatte oder im Speicher zur Verfügung zu stellen.

Gültigkeit beliebig; im Gültigkeitsbereich *object* kann ein PDI-Dokument-Handle nur zur Abfrage von Informationen aus einem PDF-Dokument verwendet werden.

Bindungen Nur in der C-Sprachbindung verfügbar.

C++ Java C# ***void close_pdi_document(int doc)***

Perl PHP ***close_pdi_document(int doc)***

C ***void PDF_close_pdi_document(PDF *p, int doc)***

Schließt alle geöffneten PDI-Seiten-Handles sowie das PDF-Importdokument.

doc Gültiges PDF-Dokument-Handle, das mit *PDF_open_pdi_document( )* erzeugt wurde.

Details Diese Funktion schließt ein PDF-Importdokument und gibt alle Ressourcen frei, die sich auf das Dokument beziehen. Alle noch geöffneten Dokumentseiten werden geschlossen. Das Dokument-Handle darf nach Aufruf dieser Funktion nicht mehr benutzt werden. Ein zu importierendes PDF-Dokument sollte nicht geschlossen werden, wenn noch weitere Seiten zu importieren sind. Sie können es zwar beliebig oft öffnen und schließen, dies kann die PDF-Ausgabedaten aber unnötig vergrößern.

Gültigkeit beliebig

10.2 Seitenfunktionen

C++ Java C# ***int open_pdi_page(int doc, int pagenumber, String optlist)***
Perl PHP ***int open_pdi_page(int doc, int pagenumber, string optlist)***
C ***int PDF_open_pdi_page(PDF *p, int doc, int pagenumber, const char* optlist)***

Bereitet eine Seite zur späteren Verwendung mit *PDF_fit_pdi_page( )* vor.

doc Gültiges Dokument-Handle, das mit *PDF_open_pdi_document( )* erzeugt wurde.

pagenumber Nummer der zu öffnenden Seite. Die erste Seite hat die Seitennummer 1.

optlist Optionsliste mit Seitenoptionen:
- Allgemeine Optionen: *errorpolicy* (siehe Tabelle 2.1) und *hypertextencoding* (siehe Tabelle 2.3)
- PDF-Seitenoptionen gemäß Tabelle 10.2.:
  boxexpand, checktransgroupprofile, clippingarea, cloneboxes, forcebox, pdiusebox, usetags
- Allgemeine XObject-Optionen (siehe Tabelle 9.10):
  associatedfiles, iconname, layer, metadata, pdfvt, reference, transparencygroup

Rückgabe PDI-Seiten-Handle, das zur Platzierung von Seiten mit *PDF_fit_pdi_page( )* verwendet werden kann. Ein Rückgabewert von -1 (in PHP: 0) zeigt an, dass die Seite nicht geöffnet werden konnte. Scheitert der Funktionsaufruf, so kann die Fehlerursache mit *PDF_get_errmsg( )* abgefragt werden. Der Rückgabewert kann bis zum Ende des umgebenden Gültigkeitsbereichs *document* verwendet werden. War die Option *infomode* beim Öffnen des Dokuments mit *PDF_open_pdi_document( )* gleich *true*, kann das Handle nicht mit *PDF_fit_pdi_page( )* verwendet werden.

Das Verhalten im Fehlerfall kann mit der Option *errorpolicy* geändert werden.

Details Diese Funktion kopiert alle Daten der importierten Seite in das Ausgabedokument, hat aber keinen sichtbaren Effekt auf die Ausgabe. Um die importierte Seite tatsächlich im generierten Dokument zu platzieren, müssen Sie *PDF_fit_pdi_page( )* verwenden. Diese Funktion schlägt in folgenden Fällen fehl:
- Das Dokument verwendet eine PDF-Version, die zum aktuellen PDF-Dokument nicht kompatibel ist. Für PDF-Versionen bis PDF 1.6 sind alle Versionen bis einschließlich derselben Version kompatibel. PDF 1.7, PDF 1.7ext3, PDF 1.7ext8 und PDF 2.0 sind bezüglich Seitenimport mit PDI alle untereinander kompatibel. Im PDF/A-Modus wird jedoch die PDF-Versionsnummer des Eingabedokuments ignoriert, da bei PDF/A die PDF-Versionsheader ignoriert werden müssen.
- Das Dokument ist zur Konformitätsstufe PDF/A, PDF/X, PDF/VT oder PDF/UA des aktuellen Ausgabedokuments nicht kompatibel oder verwendet eine inkompatible Druckausgabebedingung.
- Enthält das Dokument eine inkonsistente PDF/A- oder PDF/X-Druckausgabebedingung, werden keine Seiten importiert.

Mit *PDF_get_errmsg( )* erhalten Sie genauere Informationen über ein Problem beim PDF-Import (fehlerhafte PDF-Daten usw.).

Enthält die importierte Seite referenzierte XObjects, kopiert *PDF_open_pdi_page( )* sowohl den Proxy als auch den Verweis in das Zieldokument.

Es kann eine beliebige Anzahl von Seiten gleichzeitig geöffnet sein. Wird dieselbe Seite mehrfach geöffnet, werden dafür unterschiedliche Handles zurückgegeben und jedes Handle muss genau einmal geschlossen werden.

PDF/A Das importierte Dokument muss zur PDF/A-Konformitätsstufe des aktuellen Ausgabedokuments kompatibel sein (siehe PDFlib-Tutorial), sowie zur Druckausgabebedingung.

PDF/X Das importierte Dokument muss zur PDF/X-Konformitätsstufe des aktuellen Ausgabedokuments kompatibel sein (siehe PDFlib-Tutorial) und muss die selbe Druckausgabebedingung verwenden wie das generierte Dokument.

PDF/X-4/5: Verwendet die importierte Seite ein CMYK-ICC-Profil, das mit dem Druckausgabeprofil des generierten Dokuments identisch ist, so wird die Seite zurückgewiesen.

PDF/VT Das importierte Dokument muss zur PDF/VT-Konformitätsstufe des aktuellen Ausgabedokuments kompatibel sein (siehe PDFlib-Tutorial) und muss die gleiche Druckausgabebedingung verwenden wie das generierte Dokument. Document Part Metadata (DPM) im importierten Dokument wird nicht importiert. Falls die Option *usestransparency=false* in *PDF_begin_document( )* angegeben wurde und die importierte Seite dennoch transparente Objekte enthält, schlägt der Aufruf fehl.

PDF/UA Das importierte Dokument muss zu PDF/UA konform sein. Die *Rolemap* des importierten Dokuments muss zu der Zuordnung kompatibel sein, die in der Option *rolemap* von *PDF_begin_document( )* übergeben wurde (siehe PDFlib-Tutorial). In der Option *rolemap* (oder zuvor importierten Dokumenten) und der *Rolemap* des importierten Dokuments dürfen benutzerdefinierte Elementtypen daher nicht unterschiedlichen Standardtypen zugeordnet werden.

Die Überschriftenstruktur der importierten Seite muss zum Strukturtyp des generierten Dokuments kompatibel sein, das heißt, ist *structuretype=weak* darf nur *H1*, *H2*, usw. (aber nicht *H)* auf der Seite verwendet werden; ist *structuretype=strong* darf nur *H* (aber nicht *H1*, *H2*, usw.) auf der importierten Seite verwendet werden. Seiten mit sowohl unnummerierten als auch nummerierten Überschriften werden zurückgewiesen.

Gültigkeit beliebig außer *object*

Tabelle 10.2 Optionen für PDF_open_pdi_page( )

Option	Beschreibung
boxexpand	*(Float oder Liste aus vier Floats) Erweitert die mit der Option* `pdiusebox` *ausgewählte Seitenbox auf allen vier Seiten um den gleichen Betrag (wenn ein einziger Wert angegeben wird) oder um den jeweiligen Betrag auf der linken/rechten/oberen/unteren Seite (wenn vier Werte angegeben werden). Negative Werte sind erlaubt, um die Seite zu verkleinern. Mit dieser Option können Inhalte, die außerhalb aller Seitenboxen der importierten Seite liegen, platziert oder Seitenränder hinzugefügt werden. Standardwert: 0*
checktransgroupprofile	*(Boolean, nur relevant für PDF/A und PDF/X) Bei* `true` *und wenn die importierte Seite eine Transparenzgruppe, wird ihr Farbraum auf Konsistenz und Kompatibilität mit dem erzeugten Ausgabedokument geprüft. Damit lassen sich inkonsistente Eingabedokumente und Farbraumkonflikte vermeiden, die zu nicht konformer PDF/X- oder PDF/A-Ausgabe führen könnten. Die Einstellung dieser Option auf* `false` *reduziert Speicherplatz, sollte aber nur verwendet werden, wenn sichergestellt ist, dass die importierte Seite eine konforme Transparenzgruppe enthält (sofern vorhanden). Standardwert:* `true`

Tabelle 10.2 Optionen für PDF_open_pdi_page( )

Option	Beschreibung
clippingarea	*(Schlüsselwort) Legt fest, welche Seitenbox der importierten Seite zum Beschneiden verwendet wird. Außerhalb dieser Box liegende Inhalte sind nach dem Platzieren der importierten Seite auf der neuen Seite nicht mehr sichtbar. Unterstützte Schlüsselwörter (Standardwert:* pdiusebox*):* **art** *ArtBox, falls vorhanden, sonst CropBox* **bleed** *BleedBox, falls vorhanden, sonst CropBox* **crop** *CropBox, falls vorhanden, sonst MediaBox* **media** *MediaBox (immer vorhanden)* **pdiusebox** *In der Option* pdiusebox *angegebene Box* **trim** *TrimBox, falls vorhanden, sonst CropBox*
cloneboxes	*(Boolean; nicht erlaubt bei* boxexpand*,* forcebox *oder* pdiusebox*; muss der Option* cloneboxes *von PDF_fit_pdi_page( ) entsprechen) Bei* true *wird die Seite zum Klonen der Box mit der Option* cloneboxes *von PDF_fit_pdi_page( ) vorbereitet. Standardwert:* false
forcebox	*(Rechteck) Setzt die Seitenbox auf die angegebenen Werte. Mit dieser Option werden die Optionen* pdiusebox *und* boxexpand *überschrieben. Mit dieser Option können Inhalte platziert werden, die außerhalb aller Seitenboxen der importierten Seite liegen. Enthält die importierte Seite das Schlüsselwort* /Rotate*, müssen die Werte sorgfältig gewählt werden. In diesem Fall ist die Option* boxexpand *vorzuziehen, da sie unabhängig vom Schlüsselwort* /Rotate *funktioniert. Standardwert: die mit der Option* pdiusebox *ausgewählte Box*
ignore-pdfversion	*(Boolean) Bei* true *wird die PDF-Versionsnummer des PDF-Eingabedokuments ignoriert, d.h. Seiten aus Dokumenten mit einer höheren PDF-Version als das aktuelle PDF-Ausgabedokument können importiert werden. Dies kann PDF-Dokumente mit einer höheren PDF-Version nützlich sein, die trotzdem vollständig kompatibel zu PDF 1.7 sind. Standardwert: false, im PDF/A- und PDF/X-Modus jedoch* true
pdiusebox	*(Schlüsselwort; nicht erlaubt bei* cloneboxes*) Legt fest, welche der folgenden Box-Angaben zur Ermittlung der Größe einer importierten Seite verwendet werden. Die Größe der Box wird zur Skalierung in PDF_fit_pdi_page( ) verwendet. Mit dieser Box werden auch die sichtbaren Inhalte einer Seite festgelegt, sofern sie nicht mit der Option* clippingarea *geändert wurden. Standardwert:* crop **art** *ArtBox, falls vorhanden, sonst CropBox* **bleed** *BleedBox, falls vorhanden, sonst CropBox* **crop** *CropBox, falls vorhanden, sonst MediaBox* **media** *MediaBox (immer vorhanden)* **trim** *TrimBox, falls vorhanden, sonst CropBox*
usetags	*(Boolean; nur relevant für Ein- und Ausgabe mit Tags und wenn das Dokument mit* usetags=true *geöffnet wurde) Bei* true *werden die Struktur-Tags der importierten Seite in die Strukturhierarchie des generierten Ausgabedokuments kopiert. In diesem Fall kann PDF_fit_pdi_page( ) nur im Gültigkeitsbereich* page *aufgerufen werden. Standardwert:* true

C++ Java C# **void close_pdi_page(int page)**

Perl PHP **close_pdi_page(int page)**

C **void PDF_close_pdi_page(PDF *p, int page)**

Schließt das Seiten-Handle und gibt alle Ressourcen frei, die sich auf die Seite beziehen.

page Gültiges PDF-Seiten-Handle (keine Seitennummer!), das mit *PDF_open_pdi_page( )* erzeugt wurde.

Details Diese Funktion schließt die Seite, die mit dem Seiten-Handle *page* verknüpft ist, und gibt alle zugehörigen Ressourcen frei. *page* darf nach diesem Aufruf nicht mehr verwendet werden.

Gültigkeit beliebig außer *object*

C++ Java C# ***void fit_pdi_page(int page, double x, double y, String optlist)***

Perl PHP ***fit_pdi_page(int page, float x, float y, string optlist)***

C ***void PDF_fit_pdi_page(PDF *p, int page, double x, double y, const char *optlist)***

Platziert eine importierte PDF-Seite auf der Ausgabeseite unter Berücksichtigung verschiedener Optionen.

page Gültiges PDF-Seiten-Handle (keine Seitennummer!), das mit *PDF_open_pdi_page( )* erzeugt wurde. Die Option *infomode* muss beim Öffnen des Dokuments auf *false* gesetzt gewesen sein. Das Seiten-Handle darf nicht bereits geschlossen worden sein.

x, y Koordinaten des Referenzpunkts im Benutzerkoordinatensystem, an dem die Seite platziert wird. Die Platzierung wird über verschiedene Optionen gesteuert.

optlist Optionsliste mit Seitenoptionen:

- Optionen zur Objekteinpassung gemäß Tabelle 6.1: *blind, boxsize, fitmethod, matchbox, orientate, position, rotate, scale, showborder*
- Optionen zur Seitenverarbeitung gemäß Tabelle 9.3: *adjustpage, gstate*
- Option *cloneboxes* gemäß Tabelle 10.3
- Option zum vereinfachten Tagging von Strukturelementen gemäß Tabelle 14.5 (nur zulässig im Gültigkeitsbereich *page*): *tag*

Details Diese Funktion arbeitet ähnlich wie *PDF_fit_image( )*, arbeitet aber mit importierten PDF-Seiten. Falls eine importierte Seite *ExtGState*-Objekte enthält, sollten Sie folgende Option für *PDF_begin/end_page_ext( )* verwenden, um die Qualität der Ausgabe zu verbessern: *transparencygroup={colorspace=DeviceRGB}*.

Eine mit Tags versehene Seite (das heißt, Tagged PDF wird erzeugt und die Seite wird mit *usetags=true* aus Tagged PDF importiert) kann nicht mehr als einmal platziert werden.

Bei Tagged PDF sollten Sie vor dem Aufruf von *PDF_fit_pdi_page( )* die Funktion *PDF_info_pdi_page( )* mit dem Schlüsselwort *fittingpossible* aufrufen, um zu prüfen, ob *PDF_fit_pdi_page( )* erfolgreich sein wird (um im Fehlerfall eine Exception zu vermeiden).

Gültigkeit *page, pattern, template, glyph;* wurde eine Seite aus einem mit Tags versehenen PDF-Dokument mit *usetags=true* geladen, kann diese Funktion nur im Gültigkeitsbereich *page* aufgerufen werden.

C++ Java C# ***double info_pdi_page(int page, String keyword, String optlist)***

Perl PHP ***float info_pdi_page(int page, string keyword, string optlist)***

C ***double PDF_info_pdi_page(PDF *p, int page, const char *keyword, const char *optlist)***

Führt alle Berechnungen zur Formatierung einer PDI-Seite durch und ermittelt die daraus resultierende Metrik.

page Gültiges Seiten-Handle, das mit *PDF_open_pdi_page( )* erzeugt wurde.

Tabelle 10.3 Zusätzliche Option für PDF_fit_pdi_page( )

Option	Beschreibung
cloneboxes	*(Boolean; nicht erlaubt, wenn die Option* topdown *in PDF_begin_page_ext( ) angegeben wurde; muss der Option* cloneboxes *von PDF_open_pdi_page( ) entsprechen; nur im Gültigkeitsbereich* page*).* *Bei* true *hat die Option folgende Auswirkungen (Standardwert:* false*):* ► *Alle in der importierten Seite vorhandenen Einträge von Rotate, MediaBox, TrimBox, ArtBox, BleedBox und CropBox werden in die aktuelle Ausgabeseite kopiert.* ► *Die Seiteninhalte werden so platziert, dass die Eingabeseite dupliziert wird; Sie können die Position oder Größe der platzierten Seite nicht ändern. Die Parameter* x*,* y *sowie die folgenden Optionen werden daher ignoriert:* adjustpage*,* boxsize*,* fitmethod*,* orientate*,* position*,* rotate*,* scale*. Das Duplizieren der Eingabeseite ist nur möglich, wenn das Standardkoordinatensystem beim Aufruf von PDF_fit_pdi_page( ) aktiv ist.* ► *Mit der Option* cloneboxes *erzeugte Seitenboxen überschreiben die Optionen* artbox*,* bleedbox*,* cropbox*,* trimbox*,* mediabox *und* rotate *sowie die Parameter* width *und* height *von PDF_begin_page_ext( ).*

keyword Schlüsselwort zur Abfrage der gewünschten Information:

- Schlüsselwörter zur Abfrage der Ergebnisse der Objekteinpassung gemäß Tabelle 6.3: *boundingbox, fitscalex, fitscaley, height, objectheight, objectwidth, width, x1, y1, x2, y2, x3, y3, x4, y4*
- Seitenbezogene Schlüsselwörter gemäß Tabelle 10.4: *mirroringx, mirroringy, pageheight, pagewidth, rotate, xid*
- Schlüsselwörter für Tagged PDF gemäß Tabelle 10.4: *fittingpossible, lang, topleveltag, topleveltagcount*

optlist Optionsliste mit Optionen zu Skalierung und Platzierung:

- Allgemeine Option: *errorpolicy* (siehe Tabelle 2.1)
- Optionen zur Objekteinpassung gemäß Tabelle 6.1 (diese Optionen werden ignoriert, falls die PDF-Seite mit der Option *cloneboxes* von *PDF_open_pdi_page( )* geöffnet wurde): *boxsize, fitmethod, matchbox, orientate, position, rotate, scale*
- Optionen für die Seitenverarbeitung gemäß Tabelle 9.3 sind nicht sinnvoll; sie können zwar zur Vereinheitlichung von Optionslisten übergeben werden, werden aber für *PDF_fit_pdi_page( )* und *PDF_info_pdi_page( )* ignoriert: *adjustpage, gstate*
- Option zum vereinfachten Tagging von Strukturelementen gemäß Tabelle 14.5: *tag*
- Option zur Auswahl eines der übergeordneten Strukturelemente der Seite zur Abfrage von Informationen: *index*

Rückgabe Wert der Seiteneigenschaft, die durch *keyword* ausgewählt wurde. Ist die ausgewählte Eigenschaft für die Seite nicht vorhanden, gibt die Funktion den Wert 0 zurück. Wurde ein Objekt-Handle angefordert (z.B. *clippingpath),* gibt diese Funktion ein Handle für das Objekt zurück oder -1 (in PHP: 0), falls das Objekt nicht vorhanden ist. Wenn das erforderliche Schlüsselwort Text liefert, wird ein String-Index zurückgegeben und der entsprechende String muss mit *PDF_get_string( )* ermittelt werden.

Details Diese Funktion führt alle Berechnungen durch, die zur Platzierung der importierten Seite gemäß den übergebenen Optionen erforderlich sind, generiert aber keinerlei Ausgabe auf der Seite. Als Referenzpunkt für die Seite wird *{0 0}* verwendet. Wurde die Option *cloneboxes* von *PDF_open_pdi_page( )* übergeben, wird die Seite an der selben Stelle platziert wie die Originalseite (relativ zu den Seitenboxen).

PDF/UA Die Prüfung von *fittingpossible* fällt strenger aus als im Nicht-PDF/UA-Modus.

Gültigkeit beliebig außer *object*

Tabelle 10.4 Schlüsselwörter für PDF_info_pdi_page( )

Schlüsselwort	Beschreibung
fittingpossible	*(Nur relevant bei Tagged-PDF-Ausgabe) 0, wenn die Seite aus einem der folgenden Gründe nicht platziert werden kann (und PDF_fit_pdi_page( ) daher eine Exception auslösen würde):* ► *Eins der übergeordneten Tags auf der Seite ist gemäß der Verschachtelungsregeln für Strukturelemente unter dem aktuell aktiven Tag nicht erlaubt.* ► *Die Seite ist nicht mit Tags versehen oder enthält keine Strukturelemente und das aktive Tag erlaubt keinen direkten Inhalt als untergeordnetes Element.* ► *Die Seite wurde bereits platziert.* ► *PDF/UA mit schwacher Dokumentstruktur: bei der Nummerierung der Überschriftenebenen gibt es eine Lücke zwischen dem aktuellen Strukturelement und seinem übergeordneten Elementen einerseits und der importierten Teilhierarchie andererseits.* *Der Wert 1 wird zurückgegeben, falls die Seite im aktuellen Kontext platziert werden kann. Die Option* tag *von PDF_fit_pdi_page( ) kann übergeben werden und wird beim Ergebnis berücksichtigt. Es wird nur die Unteroption* tagname *der Option* tag *ausgewertet; weitere Unteroptionen sollten nicht angegeben werden.* *Da das Ergebnis nur im aktuellen Kontext gültig ist, sollte dieses Schlüsselwort unmittelbar vor dem geplanten Platzieren der Seite verwendet werden.*
lang	*String-Index für das Attribut* Lang *eines der Top-Level-Strukturelemente der importierten Seite oder -1, wenn kein Attribut* Lang *gefunden werden konnte. Mit der Option* index *kann eins von mehreren Top-Level-Elementen ausgewählt werden.*
mirroringx, mirroringy	*Horizontale oder vertikale Spiegelung der Seite (angegeben als 1 oder -1) gemäß der übergebenen Optionen*
pageheight, pagewidth	*Höhe und Breite der Originalseite in Punkt*
rotate	*Bei* cloneboxes=true*: Rotationswinkel der importierten Seite in Grad, also der Wert des Schlüsselworts* Rotate *der Seite. Mögliche Werte sind 0, 90, 180 und 270.* *Bei* cloneboxes=false*: immer 0*
topleveltag	*String-Index für den Namen eines der Top-Level-Strukturelemente auf der importierten Seite, wenn diese mit* usetags=true *geöffnet wurde und als Strukturelement ausgezeichneten Inhalt enthält, anderenfalls -1 (z.B. bei einer als* Artifact *ausgezeichneten Seite). Mit der Option* index *kann bei mehr als einem Top-Level-Element das entsprechende ausgewählt werden. Handelt es sich bei dem Tag um ein benutzerdefiniertes Element mit einer Rollenzuordnung im importierten Dokument, wird der entsprechende Name des Standardelements und nicht der des benutzerdefinierten Elements ausgegeben.*
topleveltagcount	*Anzahl der Strukturelemente auf oberster Ebene der Strukturhierarchie der importierten Seite. Mit den Schlüsselwörtern* lang *und* topleveltag *können Informationen über ein mit der Option* index *ausgewähltes Element abgefragt werden. 0 wird zurückgeben, falls keine Strukturelemente importiert wurden, entweder, weil die Seite keine Tags enthält oder keine Strukturelemente bzw. keine mit Tags versehenen Inhalte enthält.*
xid	*(Nur bei PDF/VT) String-Index für den Eintrag* GTS_XID *der Seite oder -1, wenn kein* GTS_XID*-Wert zugewiesen wurde. Der String* GTS_XID *kann in der Metadaten-Eigenschaft* CIP4/Summary/Content/Referenced *für DPM verwendet werden.*

Tabelle 10.5 Option für PDF_info_pdi_page( )

Option	Beschreibung
index	*(Integer; nur relevant für die Schlüsselwörter* `lang` *und* `topleveltag`*) Wählt eins der Top-Level-Strukturelemente der Seite, dessen Attribute abgefragt werden sollen. Der Wert muss im Bereich* `0..(toplevelcount-1)` *liegen. Standardwert: 0*

10.3 Weitere PDI-Funktionen

C++ Java C# ***int process_pdi(int doc, int page, String optlist)***

Perl PHP ***int process_pdi(int doc, int page, string optlist)***

C ***int PDF_process_pdi(PDF *p, int doc, int page, const char* optlist)***

Verarbeitet bestimmte Elemente eines importierten PDF-Dokuments.

doc Gültiges PDF-Dokument-Handle, das mit *PDF_open_pdi_document( )* erzeugt wurde.

page Verlangt *optlist* ein Seiten-Handle (siehe Tabelle 10.6), muss *page* ein gültiges mit *PDF_open_pdi_page( )* erzeugtes PDF-Seiten-Handle sein (keine Seitennummer!). Das Seiten-Handle darf nicht bereits geschlossen worden sein. Erfordert *optlist* kein Seiten-Handle, muss *page* gleich -1 sein.

optlist Optionsliste mit Optionen für die PDI-Verarbeitung:
- Allgemeine Option: *errorpolicy* (siehe Tabelle 2.1)
- Optionen für die PDI-Verarbeitung gemäß Tabelle 10.6: *action*, *block*

Rückgabe Wurde die Funktion erfolgreich ausgeführt, wird der Wert 1 zurückgegeben, im Fehlerfall der Fehlercode -1 (in PHP: 0). Bei *errorpolicy=exception* löst diese Function im Fehlerfall eine Exception aus. Wurden auf der Eingabeseite keine Blöcke für *action=copyallblocks* gefunden, wird 1 zurückgegeben.

PDF/A Eine Druckausgabebedingung kann mit dieser Funktion und der Option *copyoutputintent* oder mit *PDF_load_iccprofile( )* gesetzt werden. Werden im Dokument nur geräteunabhängige Farben verwendet, ist keine Druckausgabebedingung erforderlich.

PDF/X Eine Druckausgabebedingung muss mit dieser Funktion und der Option *copyoutputintent* oder mit *PDF_load_iccprofile( )* gesetzt werden.

Gültigkeit *document* bei *action=copyoutputintent*,
page bei *action=copyallblocks* und *action=copyblock*

Tabelle 10.6 Optionen für PDF_process_pdi( )

Option	Beschreibung
action	*(Schlüsselwort; erforderlich; diese Option erfordert kein Seiten-Handle) Legt die Art der PDF-Verarbeitung fest:* ***copyoutputintent*** *(Keine Auswirkung, falls das Ausgabedokument weder PDF/X- noch PDF/A-konform ist) Kopiert die Druckausgabebedingung für PDF/A oder PDF/X des importierten Dokuments in das Ausgabedokument. Der zweite und nachfolgende Versuche, eine Druckausgabebedingung zu kopieren, werden ignoriert. Enthält das Dokument mehr als eine Druckausgabebedingung, so wird die erste verwendet. Standard-Druckausgabebedingungen (ohne eingebettetes ICC-Profil) lassen sich mit dieser Methode nicht kopieren.* *Sind Ein- und Ausgabedokument PDF/X-4p/5pg-konform, wird der Verweis auf die externe Druckausgabebedingung kopiert. Die Option* `action=copyoutputintent` *ist nicht erlaubt, falls zwar die Eingabe, nicht aber die Ausgabe PDF/X-4p/5pg-konform ist.* ***copyallblocks*** *(Nur in PPS verfügbar) Kopiert alle PDFlib-Blöcke gemäß der Option* `block` *von einer Seite des Eingabedokuments auf die aktuelle Ausgabeseite.* ***copyblock*** *(Nur in PPS verfügbar) Kopiert einen PDFlib-Block gemäß der Option* `block` *von einer Seite des Eingabedokuments auf die aktuelle Ausgabeseite.*
block	*(Optionsliste; erforderlich bei* `action=copyallblocks` *und* `action=copyblock`*) Legt Details für den Kopierprozess der Blöcke fest. Die folgenden Unteroptionen werden unterstützt:* ***blockname*** *(Name-String; nur bei* `action=copyblock` *und dann erforderlich) Name des Blocks* ***outputblockname*** *(Name-String; nur bei* `action=copyblock`*) Name, unter dem der Block auf der Ausgabeseite gespeichert wird. Standardwert: Wert der Option* `blockname` ***pagenumber*** *(Integer; erforderlich) Die Seitenzahl (beginnend bei 1) der Seite des Eingabedokuments, auf der sich der Block befindet.*

10.4 pCOS-Funktionen

Alle pCOS-Funktionen arbeiten mit Pfaden, die das Zielobjekt im PDF-Dokument adressieren. Eine ausführliche Beschreibung finden Sie in der *pCOS-Pfadreferenz*.

Cookbook *Ein vollständiges Codebeispiel für die Verwendung von pCOS mit PDFlib+PDI finden Sie im Cookbook-Topic* `pdf_import/starter_pcos`. *Eine große Auswahl an pCOS-Programmierbeispielen finden Sie im pCOS-Cookbook.*

Hinweis *Im Evaluierungsmodus verarbeitet pCOS Eingabedokumente bis maximal 1 MB bzw. 10 Seiten. Folgende Elemente können auch in größeren Dokumenten im Evaluierungsmodus abgefragt werden: Seitenzahl und -größe, Blockeigenschaften sowie alle universellen Pseudo-Objekte.*

C++ Java C# ***double pcos_get_number(int doc, string path)***

Perl PHP ***double pcos_get_number(long doc, string path)***

C ***double PDF_pcos_get_number(PDF *p, int doc, const char *path, ...)***

Liefert den Wert eines pCOS-Pfades vom Typ *Zahl* oder *Boolean*.

doc Gültiges Dokument-Handle, das mit *PDF_open_pdi_document( )* erzeugt wurde.

path Vollständiger pCOS-Pfad für ein Zahl- oder Boolean-Objekt.

Weitere Parameter (Nur C-Sprachbindung) Es können ein oder mehrere zusätzliche Parameter übergeben werden, sofern der Parameter *key* entsprechende Platzhalter enthält *(%s* für Strings oder *%d* für Integers; *%%* wird für ein einzelnes Prozentzeichen verwendet). Diese Parameter ersparen es Ihnen, komplexe Pfade mit mehreren Zahlen oder Strings explizit zu formatieren. Der Client muss sicherstellen, dass die Anzahl und der Typ der Platzhalter mit den zusätzlich übergebenen Parametern übereinstimmt.

Rückgabe Numerischer Wert des durch den pCOS-Pfad bezeichneten Objekts. Bei Booleschen Werten wird 1 zurückgegeben, wenn sie *true* sind, und anderenfalls 0.

Gültigkeit beliebig

C++ Java C# ***string pcos_get_string(int doc, string path)***

Perl PHP ***string pcos_get_string(long doc, string path)***

C ***const char *PDF_pcos_get_string(PDF *p, int doc, const char *path, ...)***

Liefert den Wert eines pCOS-Pfades vom Typ *Name*, *String* oder *Boolean*.

doc Gültiges Dokument-Handle, das mit *PDF_open_pdi_document( )* erzeugt wurde.

path Vollständiger pCOS-Pfad für ein String-, Name- oder Boolean-Objekt.

Weitere Parameter (Nur C-Sprachbindung) Es können ein oder mehrere zusätzliche Parameter übergeben werden, sofern der Parameter *key* entsprechende Platzhalter enthält *(%s* für Strings oder *%d* für Integers; *%%* wird für ein einzelnes Prozentzeichen verwendet). Diese Parameter ersparen es Ihnen, komplexe Pfade mit mehreren Zahlen oder Strings explizit zu formatieren. Der Client muss sicherstellen, dass die Anzahl und der Typ der Platzhalter mit den zusätzlich übergebenen Parametern übereinstimmt.

Rückgabe String mit dem Wert des durch den pCOS-Pfad bezeichneten Objekts. Bei Booleschen Werten wird einer der Strings *true* oder *false* zurückgegeben.

Details Diese Funktion löst eine Exception aus, wenn pCOS nicht im vollständigen Modus läuft und der Typ des Objekts *string* ist. Unter bestimmten Bedingungen können einige Objekte jedoch auch im eingeschränkten pCOS-Modus abgefragt werden; Einzelheiten hierzu finden Sie in der pCOS-Pfadreferenz.

Diese Funktion geht davon aus, dass die Strings, die aus dem PDF-Dokument abgefragt werden, Text-Strings sind. String-Objekte, die Binärdaten enthalten, sollten mit *PDF_pcos_get_stream( )* abgefragt werden, das die Daten unverändert zurückliefert.

Gültigkeit beliebig

Bindungen C-Sprachbindung: Der String wird im UTF-8-Format (bei zSeries und i5/iSeries: EBCDIC-UTF-8) ohne BOM zurückgegeben. Die zurückgegebenen Strings werden in einem Ring-Puffer mit bis zu 10 Einträgen gespeichert. Werden mehr als 10 Strings abgefragt, werden die Puffer wiederverwendet. Clients müssen die Strings deshalb kopieren, wenn sie auf mehr als 10 String gleichzeitig zugreifen möchten. Bis zu 10 Aufrufe dieser Funktion können zum Beispiel als Parameter für eine *printf( )*-Anweisung verwendet werden, da die Rückgabe-Strings unabhängig voneinander sind, sofern nicht mehr als 10 Strings gleichzeitig verwendet werden.

C++-Sprachbindung: Der String wird in der standardmäßig aktiven *wstring*-Konfiguration des C++-Wrappers als *wstring* zurückgegeben. Im Kompatibilitätsmodus *string* bei zSeries und i5/iSeries wird das Ergebnis im EBCDIC-UTF-8-Format ohne BOM zurückgegeben.

Bindungen COM: das Ergebnis wird als Unicode-String im UTF-16-Format zurückgegeben. Ist kein Text verfügbar, wird ein leerer String zurückgegeben.

Java, .NET und Python: das Ergebnis wird als Unicode-String zurückgegeben. Ist kein Text verfügbar, wird ein Null-Objekt zurückgegeben.

Perl- und PHP-Sprachbindungen: das Ergebnis wird als UTF-8-String zurückgegeben. Ist kein Text verfügbar, wird ein leerer String zurückgegeben.

RPG-Sprachbindung: das Ergebnis wird als EBCDIC-UTF-8-String zurückgegeben.

C++ Java C# ***const unsigned char *pcos_get_stream(int doc, int *length, string optlist, string path)***

Perl PHP ***string pcos_get_stream(long doc, string optlist, string path)***

C ***const unsigned char *PDF_pcos_get_stream(PDF *p, int doc, int *length, const char *optlist, const char *path, ...)***

Ermittelt den Inhalt eines pCOS-Pfades vom Typ *stream, fstream* oder *string*.

doc Gültiges Dokument-Handle, das mit *PDF_open_pdi_document( )* erzeugt wurde.

length (Nur C- und C++-Sprachbindung) Zeiger auf eine Variable zur Speicherung der Länge der zurückgegebenen Streamdaten in Byte.

optlist Optionsliste mit Optionen zur Abfrage von Streamdaten gemäß Tabelle 10.7.

path Vollständiger pCOS-Pfad für ein Stream- oder String-Objekt.

Weitere Parameter (Nur C-Sprachbindung) Es können ein oder mehrere zusätzliche Parameter übergeben werden, sofern der Parameter *key* entsprechende Platzhalter enthält *(%s* für Strings oder *%d* für Integers; *%%* wird für ein einzelnes Prozentzeichen verwendet). Diese Parameter ersparen es Ihnen, komplexe Pfade mit mehreren Zahlen oder Strings explizit zu formatieren. Der Client muss sicherstellen, dass die Anzahl und der Typ der Platzhalter mit den zusätzlich übergebenen Parametern übereinstimmt.

Rückgabe Die im Stream bzw. String enthaltenen unverschlüsselten Daten. Die zurückgegebenen Daten sind leer (in C: NULL), wenn der Stream bzw. String leer ist oder wenn Inhalte verschlüsselter Anhänge in einem unverschlüsselten Dokument abgefragt werden und für die angehängten Dokumente kein Kennwort übergeben wurde.

Ist das Objekt vom Typ *stream*, werden eventuell vorhandene Filter entfernt (d.h. die eigentlichen Rohdaten werden zurückgegeben), sofern nicht *keepfilter=true*. Ist das Objekt vom Typ *fstream* oder *string*, werden die Daten so zurückgegeben, wie sie im PDF vorgefunden wurden, mit Ausnahme der Filter ASCII85 und ASCIIHex, die entfernt werden.

Details Diese Funktion löst eine Exception aus, wenn pCOS nicht im vollständigen Modus läuft. Eine Ausnahme bildet das Objekt */Root/Metadata*, das auch im eingeschränkten pCOS-Modus abrufbar ist, sofern *nocopy=false* oder *plainmetadata=true*. Eine Exception wird zudem ausgelöst, wenn *path* nicht auf ein Objekt vom Typ *stream, fstream* oder *string* zeigt.

Ungeachtet ihres Namens kann diese Funktion auch zur Abfrage von Objekten vom Typ *string* eingesetzt werden. Im Gegensatz zu *PDF_pcos_get_string( ),* das das Objekt als Text-String behandelt, verändert diese Funktion die zurückgegebenen Daten in keiner Weise. Binäre Daten-Strings werden in PDF selten verwendet und lassen sich nicht zuverlässig automatisch erkennen. Benutzer müssen deshalb selbst darauf achten, bei der Abfrage von String-Objekten die für Binärdaten oder Text geeignete Funktion zu verwenden.

Gültigkeit beliebig

Bindungen COM: Client-Programme verwenden zur Speicherung des Stream-Inhalts meist den Typ Variant. Bei JavaScript mit COM ist es nicht möglich, die Länge des zurückgegebenen Variant-Arrays abzufragen (bei anderen Sprachen mit COM ist dies aber möglich).

C- und C++-Sprachbindungen: Der zurückgegebene Datenpuffer kann bis zum nächsten Aufruf dieser Funktion verwendet werden.

Python: das Ergebnis wird als 8-Bit-String zurückgegeben (Python 3: *bytes)*.

Hinweis *Mit dieser Funktion lassen sich eingebettete Fonts aus PDF extrahieren. Beachten Sie, dass Fonts den Lizenzvereinbarungen der jeweiligen Hersteller unterliegen und ohne explizite Genehmigung des Rechteinhabers nicht weiterverwendet werden dürfen. Kontaktieren Sie gegebenenfalls den Anbieter des Fonts, um Lizenzfragen zu klären.*

Tabelle 10.7 Optionen für PDF_pcos_get_stream( )

Option	Beschreibung
convert	*(Schlüsselwort; wird bei Streams ignoriert, die mit nicht unterstützten Filtern komprimiert sind) Steuert, ob die String- oder Stream-Inhalte konvertiert werden (Standardwert:* none*):* ***none*** *Inhalte werden als Binärdaten ohne Konvertierung behandelt.* ***unicode*** *Inhalte werden als Textdaten behandelt (d.h. genauso wie in PDF_pcos_get_string( )), und nach Unicode normalisiert. In nicht Unicode-fähigen Sprachbindungen werden Daten ins Format UTF-8 ohne BOM formatiert.* *Diese Option wird für den selten verwendeten PDF-Datentyp »Text-Stream« benötigt (er kann z.B. für JavaScript verwendet werden, obwohl der Großteil der JavaScripts in String- und nicht in Stream-Objekten enthalten ist).*
keepfilter	*(Boolean; nur empfohlen bei Streams mit Bilddaten; wird bei Streams ignoriert, die mit nicht unterstützten Filtern komprimiert sind) Bei* `true` *sind die Streamdaten mit dem im Pseudo-Objekt* `filterinfo` *des Bildes angegebenen Filter komprimiert. Bei* `false` *sind die Streamdaten nicht komprimiert. Standardwert:* `true` *bei nicht unterstützten Filtern, sonst* `false`

11 Funktionen zum Füllen von Blöcken (PPS)

Der PDFlib Personalization Server (PPS) bietet spezielle Funktionen zur Verarbeitung von variablen Datenblöcken vom Typ *Text, Image, Graphics* und *PDF*. Diese Blöcke müssen in der importierten PDF-Seite enthalten sein, werden aber nicht in die generierte Ausgabe übernommen. Die importierte Seite muss vor dem Einsatz jeglicher Blockfunktionen mit *PDF_fit_pdi_page( )* auf der Ausgabeseite platziert worden sein. Bei der Berechnung der Blockposition auf der Seite berücksichtigen die Blockfunktionen die Skalierungsoptionen, die beim letzten Aufruf von *PDF_fit_pdi_page( )* übergeben wurden.

Hinweis *Alle in diesem Abschnitt erläuterten Blockfunktionen benötigen den PDFlib Personalization Server (PPS). Außerdem ist das PDFlib Block-Plugin für Adobe Acrobat erforderlich, um Blöcke in PDF-Templates zu generieren. Alternativ können Blöcke auch mit PPS selbst erstellt werden.*

Cookbook *Ein vollständiges Codebeispiel zum Füllen von Blöcken mit PPS finden Sie im Cookbook-Topic* `blocks/starter_block`.

11.1 Optionen für Rechtecke

In Tabelle 11.1 werden Rechteckoptionen für *PDF_fill_textblock( )*, *PDF_image_block( )*, *PDF_graphics_block( )* und *PDF_fill_pdfblock( )* aufgeführt. Spezifische Optionen für einen bestimmten Blocktyp *(Textline/Textflow, Image, Graphics* oder *PDF)* werden in den folgenden Abschnitten beschrieben. Fast alle Blockeigenschaften können mit gleichnamigen Optionen überschrieben werden, außer den folgenden:

`Name, Description, Subtype, Type.`

Tabelle 11.1 Rechteckoptionen für die Funktionen PDF_fill_*block( )

Option	Beschreibung
Rect	*(Rechteck) Blockkoordinaten im Koordinatensystem der Block-PDF-Datei. Das Block-Rechteck kann mit den Optionen* `refpoint` *und* `boxsize` *angegeben werden (in Benutzerkoordinaten).*
Status	*(Schlüsselwort) Beschreibt, wie der Block verarbeitet wird (Standardwert:* `active`*):* **active** *Der Block wird entsprechend seiner Eigenschaften komplett verarbeitet.* **ignore** *Der Block wird ignoriert.* **ignoredefault** *Wie* `active`*, außer dass die Eigenschaften* `defaulttext/image/pdf/graphics` *ignoriert werden, d.h. der Block bleibt leer, wenn keine Inhalte verfügbar sind. Damit können Sie sicherstellen, dass die Block-Vorgabewerte nicht zum serverseitigen Füllen der Blöcke verwendet werden, obwohl der Block Vorgabewerte zur Erzeugung der Vorschau enthalten kann. Es kann auch verwendet werden, um die Vorgabewerte für die Block-Vorschau zu deaktivieren, ohne die Vorgabewerte aus den Blockeigenschaften zu entfernen.* **static** *Es wird kein variabler Inhalt platziert, sondern der Vorgabewert für Text, Rasterbild, PDF oder Grafik wird verwendet, sofern vorhanden.*
backgroundcolor	*(Farbe) Füllfarbe für den Block; diese Farbe wird vor dem Füllen des Blocks angewendet. Damit lassen sich vorhandene Seiteninhalte verdecken. Standardwert:* none

Tabelle 11.1 Rechteckoptionen für die Funktionen PDF_fill_*block( )

Option	Beschreibung
bordercolor	*(Farbe) Farbe für die Blockränder; diese Farbe wird vor dem Füllen des Blocks angewendet. Standardwert:* none
linewidth	*(Float; muss größer* 0 *sein) Breite der Linie, mit der das Blockrechteck gezeichnet wird; wird nur verwendet, wenn* bordercolor *gesetzt ist. Standardwert: 1*

11.2 Textline- und Textflow-Blöcke

C++ Java C# ***int fill_textblock(int page, String blockname, String text, String optlist)***

Perl PHP ***int fill_textblock(int page, string blockname, string text, string optlist)***

C ***int PDF_fill_textblock(PDF *p,***
int page, const char *blockname, const char *text, int len, const char *optlist)

Füllt einen Block des Typs *Textline* oder *Textflow* mit variablen Daten unter Berücksichtigung seiner Eigenschaften.

page Gültiges PDF-Seiten-Handle für eine Seite mit PDFlib-Blöcken. Die PDF-Eingabeseite mit den Blöcken muss zuvor auf der Seite platziert worden sein, entweder direkt mit *PDF_fit_pdi_page( )*, indirekt in einer Tabellenzelle mit *PDF_fit_table( )* oder als Inhalt eines PDF-Blocks mit *PDF_fill_pdfblock( )*.

blockname (Name-String) Name des Blocks.

text (Content-String) Text, der in den Block eingetragen werden soll, oder ein Leerstring, wenn der in den Blockeigenschaften definierte Standardtext verwendet werden soll. Wird die Option *textflowhandle* übergeben und enthält sie ein gültiges Textflow-Handle, so wird dieser Parameter ignoriert.

len (Nur C-Sprachbindung) Länge von *text* (in Bytes). Ist *len = 0*, muss ein null-terminierter String übergeben werden.

optlist Optionsliste mit Optionen zum Füllen von Text-Blöcken. Folgende Optionen können verwendet werden:

- Allgemeine Option: *errorpolicy* (siehe Tabelle 2.1)
- Rechteck-Optionen für Funktionen zum Füllen von Blöcken gemäß Tabelle 11.1: *Rect, Status, backgroundcolor, bordercolor, linewidth*
- Optionen zur Objekteinpassung (siehe Abschnitt 6.1, »Objekteinpassung«, Seite 133)
- Textline-Blöcke, das heißt die Eigenschaft bzw. Option *textflow* ist auf *false* gesetzt: alle Textline-Optionen (siehe Abschnitt 5.1, »Einzeilige Textausgabe mit Textlines«, Seite 97)
- Textflow-Blöcke, das heißt die Eigenschaft bzw. Option *textflow* ist auf *true* gesetzt: alle Optionen für *PDF_add/create_textflow( )* (siehe Abschnitt 5.2, »Mehrzeilige Textausgabe mit Textflows«, Seite 104) sowie alle Optionen für *PDF_fit_textflow( )* (siehe Tabelle 5.12)
- Optionen für Text-Blöcke gemäß Tabelle 11.2: *textflow, textflowhandle*
- Option für Standardinhalt: *defaulttext* (siehe PDFlib-Tutorial)

Rückgabe -1 (in PHP: 0), falls kein Block mit dem angegebenen Namen auf der Seite existiert, der Block nicht gefüllt werden konnte (zum Beispiel wegen Fontproblemen), vom falschen Typ ist oder eine aktuellere PDFlib-Version zur Verarbeitung benötigt; 1, falls der Block erfolgreich verarbeitet werden konnte. Wurde die Option *textflowhandle* übergeben, so wird ein gültiges Textflow-Handle zur Verwendung in weiteren Aufrufen zurückgegeben.

Stellt sich das PDF-Dokument als beschädigt heraus, löst die Funktion je nach Einstellung der Option *errorpolicy* entweder eine Exception aus oder gibt -1 zurück.

Details Der übergebene Text wird in den Block eingetragen, wobei die Eigenschaften des Blocks berücksichtigt werden. Ist *text* leer, verwendet die Funktion den Standardtext des Blocks, sofern dieser vorhanden ist und *Status* nicht gleich *ignoredefault* ist, oder beendet sich ohne weiteres. Dies kann sinnvoll sein, um Blockeigenschaften wie die Füll- oder Zeichenfarbe zu nutzen.

Auswahl eines Fonts: Wird weder die Option *font* noch das implizite Laden von Fonts mit Optionen angewendet, wird der Font unter Berücksichtigung der Blockeigenschaften implizit geladen. Da das Encoding für den Font nur in einer Option, aber nicht als Blockeigenschaft angegeben werden kann, wird es standardmäßig folgendermaßen ausgewählt:

- *builtin* bei Symbolfonts und wenn *charref=false* und *textformat=auto* oder *bytes* (letzteres ist nur bei nicht Unicode-fähigen Sprachen relevant).
- sonst *unicode*.

Wird *defaulttext* verwendet, sollten die Optionen *encoding*, *charref* und *textformat* vermieden werden.

Besondere Vorsicht ist bei der Option *embedding* geboten: wird der Font unter Berücksichtigung der Blockeigenschaften implizit geladen, wird er nicht automatisch eingebettet. Soll der Font eingebettet werden, muss die Option *embedding* angegeben werden.

Verkettung von Textflow-Blöcken: Passt ein Textflow nicht vollständig in einen Block, können mit der Option *textflowhandle* mehrere Blöcke so verkettet werden, dass jeder einen Teil des Textflows aufnimmt:

- Dazu muss im ersten Aufruf der Wert -1 (in PHP: 0) übergeben werden. Das intern erzeugte Textflow-Handle wird von *PDF_fill_textblock( )* zurückgegeben und muss vom Benutzer gespeichert werden.
- Im nächsten Aufruf kann das im vorigen Schritt zurückgegebene Textflow-Handle mit der Option *textflowhandle* übergeben werden (der im Parameter *text* enthaltene Text wird in diesem Fall ignoriert und sollte leer sein). Der Block wird mit dem Rest des Textflows gefüllt.
- Dieser Vorgang lässt sich mit weiteren Textflow-Blöcken wiederholen.
- Das zurückgegebene Textflow-Handle kann an *PDF_info_textflow( )* übergeben werden, um sich über den Effekt des Blockfüllens, zum Beispiel die Endposition des Texts, zu informieren.

Der Vorgang lässt sich beliebig oft wiederholen. Danach muss der Benutzer das Textflow-Handle mit *PDF_delete_textflow( )* löschen.

PDF/UA Blockdekoration, das heißt Rahmen und Flächenfüllung unter Anwendung der Eigenschaften *backgroundcolor, bordercolor* oder *linewidth,* wird automatisch als *Artifact* ausgezeichnet.

Gültigkeit *page, pattern, template, glyph*

Tabelle 11.2 Zusätzliche Optionen für PDF_fill_textblock( )

Option	Beschreibung
textflow	*(Boolean) Steuert die ein- oder mehrzeilige Verarbeitung. Mit dieser Eigenschaft kann zwischen Textline- und Textflow-Blöcken umgeschaltet werden:* ***false*** *Der Text kann nur eine einzelne Zeile umfassen und wird mit PDF_fit_textline( ) verarbeitet.* ***true*** *Der Text kann mehrere Zeilen umfassen und wird mit PDF_fit_textflow( ) verarbeitet.* *Standardwert je nach Blocktyp:* `true` *für Textflow-Blöcke und* `false` *für Textline-Blöcke*
textflow-handle	*(Textflow-Handle; nur für PDF_fill_textblock( ) mit* `textflow=true`*) Diese Option kann zur Verkettung von Textflow-Blöcken verwendet werden. Für den ersten Block der Verkettung muss der Wert -1 (in PHP: 0) übergeben werden; der von dieser Funktion zurückgegebene Wert kann als Textflow-Handle in weiteren Aufrufen für andere zur Verkettung gehörende Blöcke übergeben werden. Diese Option ändert den Standardwert für* `fitmethod` *zu* `clip`*. Beachten Sie, dass alle Eigenschaften in den Eigenschaftengruppen für Textvorbereitung, Textformatierung und Textdarstellung der Blöcke ignoriert werden, wenn* `textflowhandle` *übergeben wird, da die entsprechenden bei der Erzeugung des Textflows verwendeten Werte angewendet werden.*

11.3 Image-Blöcke

C++ Java C# ***int fill_imageblock(int page, String blockname, int image, String optlist)***

Perl PHP ***int fill_imageblock(int page, string blockname, int image, string optlist)***

C ***int PDF_fill_imageblock(PDF *p, int page, const char *blockname, int image, const char *optlist)***

Füllt einen Block des Typs *Image* mit variablen Daten unter Berücksichtigung seiner Eigenschaften.

page Gültiges PDF-Seiten-Handle für eine Seite mit PDFlib-Blöcken. Die PDF-Eingabeseite mit den Blöcken muss zuvor auf der Seite platziert worden sein, entweder direkt mit *PDF_fit_pdi_page( )*, indirekt in einer Tabellenzelle mit *PDF_fit_table( )* oder als Inhalt eines PDF-Blocks mit *PDF_fill_pdfblock( )*.

blockname (Name-String) Name des Blocks.

image Gültiges Image-Handle für das Bild, das in den Block gefüllt werden soll, oder -1, falls das in den Blockeigenschaften definierte Standardbild verwendet werden soll.

optlist Optionsliste mit Optionen zum Füllen von Image-Blöcken. Folgende Optionen können verwendet werden:

- Allgemeine Option: *errorpolicy* (siehe Tabelle 2.1)
- Rechteck-Optionen für Funktionen zum Füllen von Blöcken gemäß Tabelle 11.1: *Rect, Status, backgroundcolor, bordercolor, linewidth*
- Optionen zur Objekteinpassung (siehe Abschnitt 6.1, »Objekteinpassung«, Seite 133)
- Optionen für die Bildverarbeitung gemäß Tabelle 9.3
- Option für Standardinhalt: *defaultimage* (siehe PDFlib-Tutorial)

Rückgabe -1 (in PHP: 0), falls kein Block mit dem angegebenen Namen auf der Seite existiert, der Block nicht gefüllt werden konnte, vom falschen Typ ist oder eine aktuellere PDFlib-Version zur Verarbeitung benötigt; 1, falls der Block erfolgreich verarbeitet werden konnte. Mit *PDF_get_errmsg( )* erhalten Sie detaillierte Informationen zu dem Problem.

Details Das Bild, auf das mit dem übergebenen Image-Handle verwiesen wird, wird im Block platziert, wobei die Eigenschaften des Blocks berücksichtigt werden. Ist *image* gleich -1 (in PHP: 0), verwendet die Funktion das Standardbild des Blocks, sofern dieses vorhanden und *Status* nicht gleich *ignoredefault* ist, oder beendet sich ohne weiteres.

Stellt sich das PDF-Dokument als beschädigt heraus, löst die Funktion abhängig von der Option *errorpolicy* entweder eine Exception aus oder gibt -1 zurück.

PDF/UA Alle Rasterbilder müssen mit einem vorausgehenden Aufruf von *PDF_begin_item( )* als *Artifact* oder *Figure* ausgezeichnet werden.

Blockdekoration, das heißt Rahmen und Flächenfüllung unter Anwendung der Eigenschaften *backgroundcolor, bordercolor* oder *linewidth,* wird automatisch als *Artifact* ausgezeichnet.

Gültigkeit *page, pattern, template, glyph*

11.4 PDF-Blöcke

C++ Java C# ***int fill_pdfblock(int page, String blockname, int contents, String optlist)***

Perl PHP ***int fill_pdfblock(int page, string blockname, int contents, string optlist)***

C ***int PDF_fill_pdfblock(PDF *p, int page, const char *blockname, int contents, const char *optlist)***

Füllt einen Block des Typs *PDF* mit variablen Daten unter Berücksichtigung seiner Eigenschaften.

page Gültiges PDF-Seiten-Handle für eine Seite mit PDFlib-Blöcken. Die PDF-Eingabeseite mit den Blöcken muss zuvor auf der Seite platziert worden sein, entweder direkt mit *PDF_fit_pdi_page( )*, indirekt in einer Tabellenzelle mit *PDF_fit_table( )* oder als Inhalt eines PDF-Blocks mit *PDF_fill_pdfblock( )*.

blockname (Name-String) Name des Blocks.

contents Gültiges PDF-Seiten-Handle für die PDF-Seite, die in den Block eingetragen werden soll, oder -1, falls die in den Blockeigenschaften definierte Standard-PDF-Seite verwendet werden soll.

optlist Optionsliste mit Optionen zum Füllen von PDF-Blöcken. Folgende Optionen können verwendet werden:

- Allgemeine Option: *errorpolicy* (siehe Tabelle 2.1)
- Rechteck-Optionen für Funktionen zum Füllen von Blöcken gemäß Tabelle 11.1: *Rect, Status, backgroundcolor, bordercolor, linewidth*
- Optionen zur Objekteinpassung (siehe Abschnitt 6.1, »Objekteinpassung«, Seite 133)
- Optionen für die Bildverarbeitung gemäß Tabelle 9.3
- Optionen für Standardinhalt: *defaultpdf, defaultpdfpage* (siehe PDFlib-Tutorial)

Rückgabe -1 (in PHP: 0), falls kein Block mit dem angegebenen Namen auf der Seite existiert, der Block nicht gefüllt werden konnte, vom falschen Typ ist oder eine aktuellere PDFlib-Version zur Verarbeitung benötigt; 1, falls der Block erfolgreich verarbeitet werden konnte. Mit *PDF_get_errmsg( )* erhalten Sie detaillierte Informationen zu dem Problem.

Details Die PDF-Seite, auf die mit dem übergebenen Seiten-Handle *contents* verwiesen wird, wird im Block platziert, wobei die Eigenschaften des Blocks berücksichtigt werden. Ist *contents* gleich -1 (in PHP: 0), verwendet die Funktion die Standard-PDF-Seite des Blocks, sofern diese vorhanden und *Status* nicht gleich *ignoredefault* ist, oder beendet sich ohne weiteres.

Stellt sich das PDF-Dokument als beschädigt heraus, löst die Funktion abhängig von der Option bzw. dem Parameter *errorpolicy* entweder eine Exception aus oder gibt -1 zurück.

PDF/UA Blockdekoration, das heißt Rahmen und Flächenfüllung unter Anwendung der Eigenschaften *backgroundcolor, bordercolor* oder *linewidth,* wird automatisch als *Artifact* ausgezeichnet.

Gültigkeit *page, pattern, template, glyph*

11.5 Graphics-Blöcke

C++ Java C# ***int fill_graphicsblock(int page, String blockname, int contents, String optlist)***

Perl PHP ***int fill_graphicsblock(int page, string blockname, int contents, string optlist)***

C ***int PDF_fill_graphicsblock(PDF *p, int page, const char *blockname, int contents, const char *optlist)***

Füllt einen Block des Typs *Graphics* mit variablen Daten unter Berücksichtigung seiner Eigenschaften.

page Gültiges PDF-Seiten-Handle für eine Seite mit PDFlib-Blöcken. Die PDF-Eingabeseite mit den Blöcken muss zuvor auf der Seite platziert worden sein, entweder direkt mit *PDF_fit_pdi_page( )*, indirekt in einer Tabellenzelle mit *PDF_fit_table( )* oder als Inhalt eines PDF-Blocks mit *PDF_fill_pdfblock( )*.

blockname (Name-String) Name des Blocks.

graphics Gültiges Grafik-Handle für die Grafik, die in den Block eingetragen werden soll, oder -1, falls die in den Blockeigenschaften definierte Standardgrafik verwendet werden soll.

optlist Optionsliste mit Optionen zum Füllen von Graphics-Blöcken. Folgende Optionen können verwendet werden:

- Allgemeine Option: *errorpolicy* (siehe Tabelle 2.1)
- Rechteck-Optionen für Funktionen zum Füllen von Blöcken gemäß Tabelle 11.1: *Rect, Status, backgroundcolor, bordercolor, linewidth*
- Optionen zur Objekteinpassung (siehe Abschnitt 6.1, »Objekteinpassung«, Seite 133)
- Optionen für die Grafikverarbeitung gemäß Tabelle 9.3.
- Option für Standardinhalt: *defaultgraphics* (siehe PDFlib-Tutorial)

Rückgabe -1 (in PHP: 0), falls kein Block mit dem angegebenen Namen auf der Seite existiert, der Block nicht gefüllt werden konnte, vom falschen Typ ist oder eine aktuellere PDFlib-Version zur Verarbeitung benötigt; 1, falls der Block erfolgreich verarbeitet werden konnte. Mit *PDF_get_errmsg( )* erhalten Sie detaillierte Informationen zu dem Problem.

Details Die Grafik, auf die mit dem übergebenen Grafik-Handle verwiesen wird, wird im Block platziert, wobei die Eigenschaften des Blocks berücksichtigt werden. Ist *graphics* gleich -1 (in PHP: 0), verwendet die Funktion die Standardgrafik des Blocks, sofern diese vorhanden und *Status* nicht gleich *ignoredefault* ist, oder beendet sich ohne weiteres.

Stellt sich das PDF-Dokument als beschädigt heraus, löst die Funktion abhängig von der Option *errorpolicy* entweder eine Exception aus oder gibt -1 zurück.

PDF/UA Blockdekoration, das heißt Rahmen und Flächenfüllung unter Anwendung der Eigenschaften *backgroundcolor, bordercolor* oder *linewidth,* wird automatisch als *Artifact* ausgezeichnet.

Gültigkeit *page, pattern, template, glyph*

12 Interaktive Funktionen

12.1 Lesezeichen

C++ Java C# ***int create_bookmark(String text, String optlist)***

Perl PHP ***int create_bookmark(string text, string optlist)***

C ***int PDF_create_bookmark(PDF *p, const char *text, int len, const char *optlist)***

Erstellt ein Lesezeichen unter Anwendung verschiedener Optionen.

text (Hypertext-String) Lesezeichentext.

len (Nur C-Sprachbindung) Länge von *text* (in Bytes). Ist *len* = *0*, muss ein null-terminierter String übergeben werden.

optlist Optionsliste mit Lesezeicheneigenschaften. Folgende Optionen können verwendet werden:

- Allgemeine Optionen: *errorpolicy* (siehe Tabelle 2.1), *hypertextencoding* und *hypertextformat* (siehe Tabelle 2.3)
- Optionen zur Steuerung von Lesezeichen gemäß Tabelle 12.1:
  action, destination, destname, fontstyle, index, item, open, parent, textcolor

Rückgabe Handle für das erstellte Lesezeichen, das in nachfolgenden Aufrufen in der Option *parent* verwendet werden kann.

Details Diese Funktion fügt ein PDF-Lesezeichen mit dem übergebenen *text* hinzu. Wird die Option *destination* nicht übergeben, zeigt das Lesezeichen auf die aktuelle Seite (bzw. auf die zuletzt erzeugte Seite, wenn es im Gültigkeitsbereich *document*, oder auf die erste Seite, wenn es vor der ersten Seite verwendet wird).

Falls Lesezeichen erstellt werden, wird die Option *openmode* für *PDF_begin/end_document( )* auf den Standardwert *bookmarks* gesetzt, sofern kein anderer Wert angegeben wurde.

PDF/UA Für PDF/UA wird das Erstellen von Lesezeichen empfohlen.

Gültigkeit *document, page*

Tabelle 12.1 Optionen für PDF_create_bookmark( )

Option	Beschreibung
action	*(Aktionsliste) Liste der Lesezeichenaktionen für das folgende Ereignis (Standardwert:* `GoTo`*-Aktion mit dem Ziel, das mit der Option* `destination` *festgelegt wird):* **activate** *Aktionen, die bei Aktivierung des Lesezeichens durchgeführt werden sollen. Alle Arten von Aktionen sind zulässig.*
destination	*(Optionsliste; wird ignoriert, wenn eine* `activate`*-Aktion festgelegt wurde) Optionsliste zur Festlegung des Lesezeichenziels gemäß Tabelle 12.10. Standardwert:* `{type fitwindow page 0}`*, falls* `destination`*,* `destname` *oder* `action` *nicht vorhanden sind.*

Tabelle 12.1 Optionen für PDF_create_bookmark( )

Option	Beschreibung
destname	*(Hypertext-String; kann leer sein; wird ignoriert, wenn die Option* `destination` *angegeben wurde) Name eines mit PDF_add_nameddest( ) definierten Ziels. Die Aktionen* `destination` *und* `destname` *haben Vorrang vor dieser Option. Wenn* `destname` *ein leerer String (d.h. {}) ist und weder* `destination` *noch* `action` *angegeben sind, ist dem Lesezeichen keine Aktion zugeordnet. Dies kann sinnvoll sein, wenn das Lesezeichen lediglich als Separator dient.*
fontstyle	*(Schlüsselwort) Legt den Fontstil des Lesezeichentexts fest:* `normal`, `bold`, `italic`, `bolditalic`. *Standardwert:* `normal`
index	*(Integer) Index, an dem das Lesezeichen unterhalb des übergeordneten Lesezeichens eingefügt wird. Anhand von Werten zwischen 0 und der Anzahl der Lesezeichen auf gleicher Ebene wird das Lesezeichen an der gewünschten Position unterhalb des übergeordneten Lesezeichens eingefügt. Mit -1 lässt sich das Lesezeichen auf der aktuellen Ebene an letzter Stelle eintragen. Standardwert: -1. Bei später eingefügten oder wieder aufgenommenen Seiten werden die Lesezeichen so platziert, als wären alle Seiten in physischer Reihenfolge erstellt worden (die Lesezeichen geben also die Seitenreihenfolge wieder).*
item	*(Item-Handle oder Schlüsselwort; das Handle darf kein aktives Strukturelement referenzieren, und auch kein Inline- oder Pseudo-Element; nur für Tagged PDF) Handle für ein Strukturelement, das dem Lesezeichen zugeordnet wird. Der Wert 0 bezieht sich immer auf das oberste Element (root) des Strukturbaums. Der Wert* `-1` *und das äquivalente Schlüsselwort* `current` *beziehen sich auf das aktuell aktive Element.*
open	*(Boolean) Bei* `false` *sind untergeordnete Lesezeichen nicht sichtbar. Bei* `true` *werden sie vollständig ausgeklappt. Standardwert:* `false`
parent	*(Lesezeichen-Handle) Das neue Lesezeichen wird dem Lesezeichen untergeordnet, das durch das Handle festgelegt ist. Bei* `parent=0` *wird ein neues Lesezeichen auf oberster Ebene erstellt. Standardwert: 0*
textcolor	*(Farbe) Legt die Farbe des Lesezeichentexts fest. Unterstützte Farbräume sind:* `none`, `gray`, `rgb`. *Standardwert:* `rgb {0 0 0}` *(=schwarz)*

12.2 Anmerkungen

C++ Java C# ***void create_annotation(double llx, double lly, double urx, double ury, String type, String optlist)***

Perl PHP ***create_annotation(float llx, float lly, float urx, float ury, string type, string optlist)***

C ***void PDF_create_annotation(PDF *p,***
double llx, double lly, double urx, double ury, const char *type, const char *optlist)

Erzeugt eine Anmerkung auf der aktuellen Seite.

llx, lly, urx, ury *x*- und *y*-Koordinaten der linken unteren und der rechten oberen Ecke des Anmerkungsrechtecks in Standardkoordinaten (wenn die Option *usercoordinates* gleich *false* ist) oder in Benutzerkoordinaten (wenn *usercoordinates* gleich *true* ist). Acrobat positioniert die linke obere Ecke der Anmerkung an der linken oberen Ecke des angegebenen Rechtecks.

Beachten Sie, dass die Koordinaten für Anmerkungen und die Parameter der Funktion *PDF_rect( )* unterschiedlich sind. *PDF_create_annotation( )* erwartet die Parameter für zwei Ecken, während an *PDF_rect( )* die Koordinaten einer Ecke sowie die Breite und Höhe übergeben werden.

Wurde die Option *usematchbox* übergeben, so werden die Parameter *llx/lly/urx/ury* ignoriert.

type Typ der Anmerkung gemäß Tabelle 12.2. In der Tabelle sind Anmerkungen vom Typ Markup durch eine Fußnote gekennzeichnet, da bestimmte Optionen ausschließlich für solche Anmerkungen gelten.

Tabelle 12.2 Anmerkungstypen

Typ	***Hinweise; spezifische Optionen für diesen Typ (zusätzlich zu den allgemeinen Optionen)***
3D	*(PDF 1.6) 3D-Modell:* `3Dactivate`, `3Ddata`, `3Dinteractive`, `3Dshared`, `3Dinitialview`
Circle[1]	`cloudy`, `createrichtext`, `inreplyto`, `interiorcolor`, `replyto`
File-Attachment[1]	`attachment`, `calloutline`, `createrichtext`, `iconname`, `inreplyto`, `replyto`
FreeText[1]	`alignment`, `calloutline`, `cloudy`, `createrichtext`, `endingstyles`, `fillcolor`, `font`, `fontsize`, `inreplyto`, `orientate`, `replyto`
Highlight[1]	`createrichtext`, `inreplyto`, `polylinelist`, `replyto`
Ink[1]	`createrichtext`, `inreplyto`, `polylinelist`, `replyto`
Line[1]	`captionoffset`, `captionposition`, `createrichtext`, `endingstyles`, `interiorcolor`, `inreplyto`, `leaderlength`, `leaderoffset`, `line`, `showcaption`, `replyto`
Link	`destination`, `destname`, `highlight`
Movie	*(Anmerkung vom Typ Movie oder Sound)* `attachment`, `movieposter`, `playmode`, `showcontrols`, `soundvolume`, `windowposition`, `windowscale`
Polygon[1]	*(PDF 1.5; Ecken sind durch gerade Linien verbunden):* `cloudy`, `createrichtext`, `inreplyto`, `polylinelist`, `replyto`
PolyLine[1]	*(PDF 1.5; ähnlich wie Polygone, außer dass die ersten und letzten Ecken nicht miteinander verbunden sind)* `createrichtext`, `endingstyles`, `inreplyto`, `interiorcolor`, `polylinelist`, `replyto`
Popup	`open`, `parentname`

Tabelle 12.2 Anmerkungstypen

Typ	*Hinweise; spezifische Optionen für diesen Typ (zusätzlich zu den allgemeinen Optionen)*
RichMedia	*(PDF 1.7ext3)* `richmedia`
Square[1]	`cloudy`, `createrichtext`, `inreplyto`, `interiorcolor`, `replyto`
Squiggly[1]	*(Unterringeln-Anmerkung)* `createrichtext`, `inreplyto`, `polylinelist`, `replyto`
Stamp[1]	`createrichtext`, `iconname`, `inreplyto`, `orientate`, `replyto`
StrikeOut[1]	`createrichtext`, `inreplyto`, `polylinelist`, `replyto`
Text[1]	*(In Acrobat wird dieser Anmerkungstyp* `Notiz` *genannt)* `createrichtext`, `iconname`, `inreplyto`, `open`, `replyto`
Underline[1]	`createrichtext`, `inreplyto`, `polylinelist`, `replyto`

1. Markup-Anmerkung; dies ist relevant für die Option `createrichtext`.

optlist Optionsliste mit Anmerkungseigenschaften:

- Allgemeine Option: *hypertextencoding* (siehe Tabelle 2.3)
- Folgende Optionen gemäß Tabelle 12.3 werden für alle Anmerkungstypen unterstützt:
  action, annotcolor, borderstyle, cloudy, contents, createdate, custom, dasharray, display, layer, linewidth, locked, lockedcontents, name, opacity, popup, readonly, rotate, subject, template, title, usematchbox, usercoordinates, zoom
- Folgende typspezifische Optionen gemäß Tabelle 12.3 werden nur für manche Anmerkungstypen gemäß Tabelle 12.2 unterstützt:
  alignment, calloutline, captionoffset, captionposition, createrichtext, destname, endingstyles, fillcolor, font, fontsize, highlight, iconname, inreplyto, interiorcolor, leaderlength, leaderoffset, line, movieposter, open, orientate, movieposter, playmode, polylinelist, replyto, showcaption, showcontrols, soundvolume, windowposition, windowscale
- Optionen für *type=3D* gemäß Tabelle 13.4:
  3Dactivate, 3Ddata, 3Dinteractive, 3Dshared, 3Dinitialview
- Option *richmedia* für *type=RichMedia* gemäß Tabelle 13.4 mit entsprechenden Unteroptionen in Tabelle 13.7.
- Option zum vereinfachten Tagging von Strukturelementen gemäß Tabelle 14.5*: tag*

Details Bei Tagged PDF erzeugt diese Funktion automatisch ein geeignetes *OBJR*-Element für die erzeugte Anmerkung. Sie müssen vor dem Aufruf dieser Funktion ein neues Container-Element vom Typ *Link* oder *Annot* erzeugen (siehe PDFlib-Tutorial).

PDF/A PDF/A-1: es sind nur folgende Anmerkungstypen zulässig:
Circle, FreeText, Highlight, Ink, Line, Link, Popup, Square, Squiggly, Stamp, StrikeOut, Text, Underline
PDF/A-2/3: nur *type=Link* ist zulässig.
Einige Optionen sind eingeschränkt, siehe Tabelle 12.3.

PDF/X Anmerkungen sind nur zulässig, wenn sie vollständig außerhalb der BleedBox liegen (oder TrimBox/ArtBox, wenn keine BleedBox vorhanden ist).
PDF/X-1a/3: der Anmerkungstyp *FileAttachment* ist nicht zulässig.

PDF/UA Wenn diese Funktion zur Erzeugung einer sichtbaren Anmerkung aufgerufen wird, muss ein nicht-gruppierendes Strukturelement mit *PDF_begin_item( )* oder der Option *tag* erzeugt werden.

Für sichtbare Anmerkungen muss die Option *contents* oder die Option *tag* mit der Unteroption *ActualText* übergeben werden.

Gültigkeit *page*

Tabelle 12.3 Optionen für PDF_create_annotation()

Option	Beschreibung
action	*(Aktionsliste) Liste aus Anmerkungsaktionen für folgende Ereignisse (Standardwert: leere Liste). Alle Aktionstypen sind zulässig.* **activate** *(Nur für* type=Link*) Aktionen, die bei Aktivierung der Anmerkung durchgeführt werden.* **close** *(PDF 1.5) Aktionen, die beim Schließen der Seite mit der Anmerkung durchgeführt werden.* **open** *(PDF 1.5) Aktionen, die beim Öffnen der Seite mit der Anmerkung durchgeführt werden.* **invisible** *(PDF 1.5) Aktionen, die durchgeführt werden, wenn die Seite mit der Anmerkung nicht mehr sichtbar ist.* **visible** *(PDF 1.5) Aktionen, die durchgeführt werden, wenn die Seite mit der Anmerkung wieder sichtbar wird.*
alignment	*(Schlüsselwort; nur für* type=FreeText*) Ausrichtung von Text in der Anmerkung:* left, center, right. *Standardwert:* left
annotcolor	*(Farbe) Farbe des Hintergrunds des geschlossenen Anmerkungssymbols, der Titelleiste des Popup-Fensters der Anmerkung und des Randes einer Verknüpfungsanmerkung. Unterstützte Farbräume:* none *(nicht für* type=Square, Circle*),* gray, rgb *und (in PDF 1.6)* cmyk. *Standardwert:* white *für* type=Square, Circle, *sonst* none *PDF/A-1: diese Option kann nur verwendet werden, wenn eine Druckausgabebedingung für RGB festgelegt wurde. Außerdem muss* gray- *oder* rgb*-Farbe verwendet werden.*
attachment	*(Asset-Handle; nur für* type=FileAttachment *und* Movie*; erforderlich) Dateianhang, der mit PDF_load_asset() und* type=attachment *geladen wurde. Unterstützte Unteroptionen: siehe Tabelle 13.6.* *Für* type=FileAttachment*: Datei, die zur Anmerkung gehört* *Für* type=Movie*: Mediendatei in einem der folgenden Formate: AVI- oder QuickTime-Movie, WAV- oder AIFF-Sound. Beachten Sie, dass der Anhang mit* type=Attachment *und* external=true *in PDF_load_asset() geladen werden muss.*
borderstyle	*(Schlüsselwort) Stil des Anmerkungsrandes oder der Linie der Anmerkungstypen* polygon, PolyLine, Line, Square, Circle, Ink*:* solid, beveled, dashed, inset, underline. *Beachten Sie, dass die Stile* beveled, inset *und* underline *in Acrobat nicht zuverlässig funktionieren. Standardwert:* solid
calloutline	*(Liste mit vier oder sechs Floats; PDF 1.6;nur für* type=FreeText*) Liste mit 4 oder 6 Float-Werten, die eine Linie zu einer FreeText-Anmerkung beschreibt (Callout). Sechs Zahlen* {x1 y1 x2 y2 x3 y3} *stellen die Anfangs-, Mittel- und Endkoordinaten der Linie dar. Vier Zahlen* {x1 y1 x2 y2} *stellen die Anfangs- und Endkoordinaten der Linie dar. Die Koordinaten werden in Standardkoordinaten interpretiert (wenn die Option* usercoordinates *gleich* false *ist) oder in Benutzerkoordinaten (bei* true*).* *Der Anfangspunkt wird mit dem Symbol ausgezeichnet, das im ersten Schlüsselwort der Option* endingstyles *angegeben wurde.*
captionoffset	*(2 Floats; nur für* type=Line*; PDF 1.7) Versatz der Beschriftung von ihrer normalen Position. Der erste Wert gibt den horizontalen Versatz entlang der Anmerkungszeile von ihrem Mittelpunkt an, wobei ein positiver Wert eine Verschiebung nach rechts und ein negativer Wert eine Verschiebung nach links bedeutet. Der zweite Wert gibt den vertikalen Versatz senkrecht zur Anmerkungszeile an, wobei ein positiver Wert eine Verschiebung nach oben und ein negativer Wert eine Verschiebung nach unten bedeutet. Standard:* {0, 0}*, d.h. kein Versatz von der normalen Position*
caption-position	*(Schlüsselwort; nur für* type=Line*; PDF 1.7) Position der Beschriftung der Anmerkung. Diese Option wird bei* showcaption=false *ignoriert. Unterstützte Schlüsselwörter (Standardwert:* Inline*):* **Inline** *Die Beschriftung wird innerhalb der Zeile zentriert.* **Top** *Die Beschriftung wird oberhalb der Zeile positioniert.*

Tabelle 12.3 Optionen für PDF_create_annotation( )

Option	Beschreibung
cloudy	*(Float; nur für* type=Circle, FreeText, Polygon *und* Square; *PDF 1.5) Legt die Intensität des »Wolken«-Effekts beim Zeichnen des Polygons fest. Mögliche Werte sind 0 (kein Effekt), 1 und 2. Wird diese Option verwendet, dann wird die Option* borderstyle *ignoriert. Standardwert: 0*
contents	*(String für* type=FreeText; *sonst Hypertext-String mit maximal 65535 Bytes; PDF/UA: erforderlich, sofern kein* ActualText *in der Option* tag *übergeben wurde, und immer erforderlich für* type=Link) *Text, der in der Anmerkung angezeigt werden soll, oder (wenn dies nicht der Fall ist) ein den Inhalt beschreibender Text in lesbarer Form. Mit Carriage-Return oder Line-Feed-Zeichen kann ein neuer Absatz begonnen werden.*
createdate	*(Boolean; PDF 1.5) Bei* true *wird für die Anmerkung ein Datum/Zeiteintrag angelegt. Standardwert:* true *für Markup-Anmerkungen, sonst* false
createrichtext	*(Optionsliste; nur für Markup-Amerkungen; die Option* contents *muss leer sein; PDF 1.5) Erzeugt formatierten Text (Rich Text) aus einem Textflow. Dies kann für die Erstellung von formatiertem Text für Anmerkungen nützlich sein. Unterstützte Unteroptionen:* **textflow** *(Textflow-Handle) Textflow, der als formatierter Text an die Anmerkung angehängt wird. Wurde das Textflow-Handle vor dem Aufruf von PDF_create_annotation( ) an PDF_fit_textflow/table( ) übergeben, wird nur der verbleibende Text für die Anmerkung verwendet. Ist kein weiterer Text mehr verfügbar, wird der Textflow vom Anfang erneut gestartet. Die Verwendung eines Textflows für eine Anmerkung hat auf nachfolgende Aufrufe von PDF_fit_textflow/table( ) keine Auswirkung.* **userunit** *(Schlüsselwort) Maßeinheit für Einzelwerte (z.B. firstlinedist, fontsize):* cm *(Zentimeter),* in *(Inches),* mm *(Millimeter) oder* pt *(Punkte). Standardwert:* pt *Folgende Textflow-Optionen werden bei der Erzeugung von Rich Text ausgewertet; alle anderen werden ignoriert:* nextline, alignment, fillcolor, underline, strikeout, font, fontsize, textrise, *Optionen zur Textformatierung* *Beachten Sie, dass die Optionen* font *und* alignment *in Acrobat nicht den erwarteten Effekt haben.*
custom	*(Liste aus Optionslisten; nur für erfahrene Anwender) Mit dieser Option lassen sich beliebig viele private Einträge ins Annotation-Dictionary einfügen. Dies kann bei Spezialanwendungen hilfreich sein, zum Beispiel für Verarbeitungsanweisungen für Digitaldrucker. Diese Option erfordert Kenntnisse des PDF-Dateiformats und der Zielanwendung. Unterstützte Unteroptionen:* **key** *(String; erforderlich) Name des Dictionary-Keys (ohne das Zeichen /). Jeder Nicht-Standard-PDF-Key kann angegeben werden und außerdem folgende Standard-Keys:* Contents, Name *(Option* iconname), NM *(Option* name) *und* Open. *In diesem Fall werden die entsprechenden Optionen ignoriert.* **type** *(Schlüsselwort; erforderlich) Typ des entsprechenden Wertes:* boolean, name *oder* string **value** *(Hypertext-String bei* type=string, *sonst String; erforderlich) Wert, wie er in der PDF-Ausgabe erscheint; PDFlib wendet automatisch die für Strings und Namen erforderlichen Auszeichnungen an.*
dasharray	*(Liste aus zwei nicht negativen Floats; nur für* borderstyle=dashed). *Länge der Striche und Lücken bei einem gestrichelten Rand in Standardeinheiten (siehe Tabelle 7.1). Standardwert: 3 3*
destination	*(Optionsliste; nur für* type=Link*; wird ignoriert, wenn eine* activate*-Aktion festgelegt wurde) Optionsliste gemäß Tabelle 12.10, die das anzuspringende Ziel definiert*
destname	*(Hypertext-String; nur für* type=Link*; wird ignoriert, wenn die Option* destination *angegeben wurde) Name eines mit PDF_add_nameddest( ) definierten Ziels. Aktionen, die mit der Option* destination *oder* destname *von PDF_create_action( ) erstellt wurden, haben Vorrang vor dieser Option.*
display	*(Schlüsselwort; wird bei PDF/A auf* visible *gesetzt) Sichtbarkeit auf Bildschirm und Papier:* visible, hidden, noview, noprint. *Standardwert:* visible

Tabelle 12.3 Optionen für PDF_create_annotation()

Option	Beschreibung
endingstyles	*(Schlüsselwortliste; nur für* type=FreeText, Line, PolyLine*) Liste mit zwei Schlüsselwörtern, die die Art der Linienenden festlegen: das zweite Schlüsselwort wird bei* type=FreeText *ignoriert (Standardwert:* {none none}*):* none, square, circle, diamond, openarrow, closedarrow *Zusätzlich für PDF 1.5:* butt, ropenarrow, rclosedarrow *Zusätzlich für PDF 1.6:* slash
filename	*Veraltet; verwenden Sie stattdessen* attachment
fillcolor	*(Farbe; nur für* type=FreeText*) Füllfarbe des Texts. Unterstützte Farbräume:* none, gray, rgb *und (in PDF 1.6)* cmyk*. Standardwert:* {gray 0} *(=schwarz)* *PDF/A-1: diese Option kann nur verwendet werden, wenn eine Druckausgabebedingung für RGB festgelegt wurde. Außerdem muss* gray- *oder* rgb*-Farbe verwendet werden.*
font	*(Font-Handle; nur für* type=FreeText*; erforderlich) Legt den Font für die Anmerkung fest.*
fontsize	*(Fontgröße; nur für* type=FreeText*; erforderlich) Fontgröße in Standard- oder Benutzerkoordinaten abhängig von der Option* usercoordinates*. Der Wert 0 oder das Schlüsselwort* auto *bedeuten, dass Acrobat die Fontgröße dem Rechteck anpasst.*
highlight	*(Schlüsselwort; nur für* type=Link*) Highlight-Modus der Anmerkung, wenn sie vom Benutzer angeklickt wird:* none, invert, outline, push*. Standardwert:* invert
iconname	*(String; nur für* type=Text, Stamp, FileAttachment*) Name eines Symbols, mit dem die Anmerkung angezeigt wird (zum Anlegen einer Anmerkung ohne sichtbares Symbol setzen Sie* opacity=0*):* *Für* type=Text *(Standardwert:* note*):* comment, key, note, help, newparagraph, paragraph, insert *Für* type=Stamp*, allerdings funktionieren diese in Acrobat nicht zuverlässig, deshalb empfehlen wir stattdessen die Option* template *(Standardwert:* draft*):* approved *(genehmigt),* experimental *(in Versuchsphase),* notapproved *(nicht genehmigt),* asis *(wie gesehen),* expired *(abgelaufen),* notforpublicrelease *(nicht zur Veröffentlichung),* confidential *(vertraulich),* final *(endgültige Version),* sold *(verkauft),* departmental *(abteilungsintern),* forcomment *(zum Kommentar),* topsecret *(geheim),* draft *(Entwurf),* forpublicrelease *(zur Veröffentlichung).* *Für* type=FileAttachment *(Standardwert:* pushpin*):* graph *(Diagramm)*, pushpin *(Anlage)*, paperclip *(Büroklammer)*, tag *(Tag)* *Mit der Option* template *lassen sich benutzerdefinierte Icons erstellen.*
inreplyto	*(Hypertext-String; PDF 1.5; nur für Markup-Anmerkungen; erforderlich, wenn die Option* replyto *angegeben wird) Name der Anmerkung (siehe Option* name*), auf die diese Anmerkung eine Antwort darstellt. Beide Anmerkungen müssen sich auf der selben Seite des Dokuments befinden. Die Beziehung der beiden Anmerkungen zueinander muss in der Option* replyto *festgelegt werden.*
interiorcolor	*(Farbe; nur für* type=Line, PolyLine, Square, Circle*) Farbe für die Linienenden, das Rechteck bzw. die Ellipse der Anmerkung. Unterstützte Farbräume:* none, gray, rgb *und (in PDF 1.6)* cmyk*. Standardwert:* none *PDF/A-1: diese Option kann nur verwendet werden, wenn eine Druckausgabebedingung für RGB festgelegt wurde. Außerdem muss* gray- *oder* rgb*-Farbe verwendet werden.*
layer	*(Ebenen-Handle; PDF 1.5) Ebene, zu der die Anmerkung gehört. Die Anmerkung ist nur sichtbar, wenn auch die entsprechende Ebene sichtbar ist.*

Tabelle 12.3 Optionen für PDF_create_annotation( )

Option	Beschreibung
leaderlength	*(Liste aus ein oder zwei Floats; der zweite Float darf nicht negativ sein; nur für* `type=Line`*; PDF 1.6) Länge der Führungslinien in Standardkoordinaten (wenn die Option* `usercoordinates` *gleich* `false` *ist) oder in Benutzerkoordinaten (bei* `true`*). Führungslinien sind Hilfslinien, die durch die Endpunkte der Linie gehen und senkrecht auf ihr stehen. Die Länge wird durch zwei Zahlen angegeben (Standardwert:* `{0 0}`*):* *Die erste Zahl gibt die Erweiterung von jedem Endpunkt der Linie senkrecht zur Linie selbst an. Bei einem positiven Wert erscheinen die Führungslinien im Uhrzeigersinn vom Anfangspunkt in Richtung Endpunkt aus gesehen (wie in der Option* `line` *angegeben); bei einem negativen Wert in der entgegengesetzten Richtung.* *Der zweite Wert, der weggelassen werden kann, gibt die Länge der Erweiterung der Führungslinie auf der entgegengesetzten Seite der Führungslinie an. Wenn der erste Wert 0 ist, wird ein positiver zweiter Wert ignoriert.*
leaderoffset	*(Nicht negativer Float; nur für* `type=Line`*; PDF 1.7) Versatz der Führungslinie, berechnet aus dem Leerraum zwischen den Endpunkten der Anmerkung und dem Anfang der Führungslinien in Standardkoordinaten (wenn die Option* `usercoordinates` *gleich* `false` *ist) oder in Benutzerkoordinaten (bei* `true`*). Standardwert: 0*
line	*(Linie; nur für* `type=Line`*; erforderlich) Liste aus vier Zahlen x1, y1, x2, y2 für die Anfangs- und Endpunkte der Linie in Standardkoordinaten (wenn die Option* `usercoordinates` *gleich* `false` *ist) oder in Benutzerkoordinaten (bei* `true`*).*
linewidth	*(Integer) Breite des Anmerkungsrandes oder der Linie für die Anmerkungstypen* `Line`*,* `PolyLine`*,* `Polygon`*,* `Square`*,* `Circle`*,* `Ink` *in Standardeinheiten (=Punkt). Bei* `linewidth=0` *ist kein Rand sichtbar. Standardwert: 1*
locked	*(Boolean) Bei* `true` *können die Anmerkungseigenschaften in Acrobat nicht bearbeitet werden (z.B. Position und Größe). Die Inhalte lassen sich weiterhin bearbeiten. Standardwert:* `false`
locked-contents	*(Boolean; PDF 1.7) Bei* `true` *können die Anmerkungsinhalte in Acrobat nicht bearbeitet werden. Die Anmerkung kann aber gelöscht oder ihre Eigenschaften geändert werden (z.B. Position und Größe). Standardwert:* `false`
mimetype	*Veraltet, verwenden Sie stattdessen* `attachment`*.*
movieposter	*(Schlüsselwort; nur für* `type=Movie`*) Schlüsselwort, das ein Standbild zur Darstellung des Movies auf der Seite definiert. Unterstützte Schlüsselwörter:* `auto` *(das Standbild wird der Movie-Datei entnommen),* `none` *(es wird kein Standbild angezeigt). Standardwert:* `none`
name	*(Hypertext-String) Name, der die Anmerkung eindeutig identifiziert. Der Name ist bei einigen Aktionen erforderlich und muss auf der Seite eindeutig sein. Standardwert: nicht vorhanden*
opacity	*(Float- oder Prozentwert; nicht für PDF/A-1) Wert der konstanten Transparenz (constant opacity) (0-1 oder 0%-100%), der beim Zeichnen der Anmerkung verwendet wird. Standardwert: 1*
open	*(Boolean; nur für* `type=Text`*,* `Popup`*) Bei* `true` *wird die Anmerkung anfangs in geöffnetem Zustand angezeigt. Standardwert:* `false`
orientate	*(Schlüsselwort; nur für* `type=FreeText`*,* `Stamp`*) Legt die gewünschte Orientierung der Anmerkung innerhalb ihres Rechtecks fest. Unterstützte Schlüsselwörter:* `north` *(nach oben),* `east` *(nach rechts),* `south` *(auf dem Kopf stehend),* `west` *(nach links). Standardwert:* `north`
parentname	*(String; nur für* `type=PopUp`*) Name der in der Anmerkungshierarchie übergeordneten Popup-Anmerkung. Wird diese Option übergeben, überschreiben die Optionen* `contents`*,* `annotcolor` *und* `title` *der übergeordneten Anmerkung die zugehörigen Werte der Popup-Anmerkung.*
playmode	*(Schlüsselwort; nur für* `type=Movie`*) Modus zum Abspielen des Movies oder Sounds. Unterstützte Schlüsselwörter:* `once` *(einmal abspielen und beenden),* `open` *(abspielen und den Balken mit der Movie-Steuerung geöffnet lassen),* `repeat` *(bis zum Beenden wiederholt von Anfang bis Ende abspielen),* `palindrome` *(bis zum Beenden kontinuierlich vorwärts und rückwärts abspielen). Standardwert:* `once`

Tabelle 12.3 Optionen für PDF_create_annotation()

Option	Beschreibung
polylinelist	*(Liste mit einer oder mehreren Polylinien oder Vierecken; nur für* type=Polygon, PolyLine, Ink, Highlight, Underline, Squiggly, Strikeout*) Die Koordinaten werden in Standardkoordinaten interpretiert (wenn die Option* usercoordinates *gleich* false *ist) oder in Benutzerkoordinaten (bei* true*). Standardwert: Polyline, die die Eckpunkte des Anmerkungsrechtecks miteinander verbindet.* ***type=Polygon, PolyLine, Ink*** *Einzelne Liste mit einer Polylinie aus* n *Segmenten (Mindestanzahl:* n=2*).* ***sonst*** *Liste mit* n *Unterlisten aus jeweils 8 Float-Werten für* n *Vierecke (Mindestanzahl:* n=1*). Jedes Viereck umschließt ein Wort oder eine Gruppe zusammenhängender Wörter im Text, der der Anmerkung zugrunde liegt. Die Vierecke müssen in Zickzack-Reihenfolge angegeben werden (rechts oben, links oben, rechts unten, links unten*
popup	*(String) Name einer Popup-Anmerkung, mit der der Text eingegeben oder bearbeitet werden kann. Standardwert: nicht vorhanden*
readonly	*(Boolean) Bei* true *wird jede Interaktion zwischen Benutzer und Anmerkung unterbunden. Die Anmerkung lässt sich zwar anzeigen oder drucken, reagiert aber weder auf Mausklicks noch ändert sie ihr Aussehen bei Mausbewegungen. Standardwert:* false
replyto	*(Schlüsselwort; PDF 1.6; nur für Markup-Anmerkungen) Legt die Beziehung (den Antworttyp) zwischen dieser Anmerkung und der in der Option* inreplyto *angegebenen Anmerkung fest. Unterstützte Schlüsselwörter (Standardwert:* reply*):* ***reply*** *Die Anmerkung wird als Antwort auf die in der Option* inreplyto *angegebene Anmerkung behandelt.* ***group*** *Die Anmerkung muss mit der in der Option* inreplyto *angegebenen Anmerkung gruppiert werden.*
richmedia	*(Optionsliste; erforderlich für* type=RichMedia*) Optionen für Rich Media gemäß Tabelle 13.7*
rotate	*(Boolean; darf für Text-Anmerkungen im PDF/A-Modus nicht auf* true *gesetzt werden) Bei* true *dreht sich die Anmerkung zusammen mit der Seite. Anderenfalls bleibt die Anmerkung unverändert. Diese Option wird für die Symbole von Text-Anmerkungen ignoriert. Standardwert:* false *für Text-Anmerkungen im PDF/A-Modus, sonst* true
showcaption	*(Boolean; nur für* type=Line*; PDF 1.6) Bei* true *wird der in den Optionen* contents *oder* createrichtext *angegebene Text als Beschriftung für die Linie wiederholt. Standardwert:* false
showcontrols	*(Boolean; nur für* type=Movie*) Bei* true *wird beim Abspielen des Movies oder Sounds ein Steuerbalken angezeigt. Standardwert:* false
soundvolume	*(Float; nur für* type=Movie*) Die Anfangslautstärke, mit der das Movie abgespielt wird, im Bereich von -1.0 bis 1.0. Höhere Werte bedeuten eine größere Lautstärke; negative Werte dämpfen den Ton. Standardwert: 1.0*
subject	*(Hypertext-String; PDF 1.5) Text mit einer Kurzbeschreibung des Themas, das die Anmerkung behandelt. Standardwert: nicht vorhanden*

Tabelle 12.3 Optionen für PDF_create_annotation( )

Option	Beschreibung
template	*(Optionsliste) Visuelle Darstellung der Anmerkung für ein oder mehrere Zustände. Dies kann z.B. für benutzerdefinierte Stempel nützlich sein. Unterstützte Unteroptionen:* ***normal/rollover/down*** *(Template-Handle oder Schlüsselwort) Template, das für die Darstellung der Anmerkung als normale Anmerkung, Maus-Rollover oder als Mausklick verwendet wird. Mithilfe des Schlüsselworts* `viewer` *erzeugt Acrobat beim Zeichnen der Seite die entsprechende Darstellung. Standardwert für* `normal`*:* `viewer`*; Standardwert für* `rollover` *und* `down`*: der Wert von* `normal` ***fitmethod*** *(Schlüsselwort) Methode zum Einpassen des Templates in das Anmerkungsrechteck. Ist* `fitmethod` *von* `entire` *verschieden, wird das Anmerkungsrechteck an die Template-Box angepasst (Standardwert:* `entire`*):* ***nofit*** *Platziert das Template ohne jegliche Skalierung oder Beschneidung.* ***meet*** *Platziert das Template gemäß der Option* `position` *und passt es vollständig in das Rechteck ein, indem es unter Beibehaltung seiner Seitenproportionen skaliert wird. Normalerweise passen mindestens zwei Kanten des Templates auf die entsprechenden Kanten des Rechtecks.* ***entire*** *Platziert das Template gemäß der Option* `position` *und skaliert es so, dass es das Rechteck vollständig bedeckt. Normalerweise wird das Template dadurch verzerrt.* ***position*** *(Liste aus Floats oder Schlüsselwörtern) Ein oder zwei Werte für die Position des Referenzpunkts* (`x`, `y`) *innerhalb des Templates, wobei* {`0 0`} *die linke untere Ecke des Templates und* {`100 100`} *die rechte obere Ecke angibt. Die Werte werden als Prozentwerte der Breite und Höhe des Templates angegeben. Sind die beiden Prozentwerte gleich, so genügt die Angabe eines einzigen Float-Wertes.* *Statt der Werte 0, 50 und 100 können die äquivalenten Schlüsselwörter* `left`, `center` *und* `right` *(in x-Richtung) oder* `bottom`, `center` *und* `top` *(in y-Richtung) angegeben werden. Wird nur ein einziges Schlüsselwort angegeben, wird für die andere Richtung das entsprechende Schlüsselwort verwendet. Standardwert:* {`left bottom`}.
title	*(Hypertext-String; bei Movie-Anmerkungen empfohlen) Text, der in der Titelleiste des Fensters einer aktiven und geöffneten Popup-Anmerkung erscheint. Der String entspricht in Acrobat dem Feld »Verfasser«. Er darf maximal 255 Einbyte-Zeichen oder 126 Unicode-Zeichen enthalten. In der Praxis sind aber nicht mehr als 32 Zeichen ratsam. Standardwert: nicht vorhanden*
usematchbox	*(String-Liste) Die Parameter* `llx`/`lly`/`urx`/`ury` *werden ignoriert, und stattdessen wird die angegebene Matchbox verwendet. Das erste Element in der Optionsliste ist ein Name-String zur Angabe einer Matchbox. Das zweite Element ist entweder ein Integer, der die Nummer des gewünschten Rechtecks bezeichnet, oder das Schlüsselwort* `all` *für alle zur gewählten Matchbox gehörenden Rechtecke. Fehlt das zweite Element, so wird* `all` *als Standardwert verwendet.* *Existiert die Matchbox oder das angegebene Rechteck auf der aktuellen Seite nicht, kehrt die Funktion zurück, ohne eine Anmerkung zu erstellen. Wenn bei der Erzeugung von Anmerkungen in Tabellenzellen Matchboxen verwendet werden, muss PDF_create_annotation( ) nach PDF_fit_table( ) aufgerufen werden, um sicherzustellen, dass die Größe der Matchbox bereit berechnet wurde.*
user-coordinates	*(Boolean) Bei* `false` *werden die Koordinaten für Anmerkungen sowie die Fontgröße im Standardkoordinatensystem erwartet; anderenfalls wird das aktuelle Benutzerkoordinatensystem verwendet. Standardwert: Wert der globalen Option* `usercoordinates`
window-position	*(Liste aus 2 Floats oder Schlüsselwörtern; nur für* `type=Movie`*) Relative Position des Fensters am Bildschirm bei einem überlagerten Wiedergabefenster. Die beiden Werte legen die Position des überlagerten Wiedergabefensters am Bildschirm fest, wobei* {`0 0`} *die linke untere Ecke des Bildschirms und* {`100 100`} *die rechte obere Ecke bezeichnet. Die Schlüsselwörter* `left`, `center`, `right` *(in horizontaler Bildschirmrichtung) oder* `bottom`, `center`, `top` *(in vertikaler Bildschirmrichtung) können als Äquivalente für die Werte 0, 50 und 100 verwendet werden. Standardwert:* {`center center`}

Tabelle 12.3 Optionen für PDF_create_annotation( )

Option	*Beschreibung*
windowscale	*(Float oder Schlüsselwort; nur für* `type=Movie`*) Zoom-Faktor, mit dem das Movie in einem überlagerten Wiedergabefenster abgespielt wird. Ist diese Option nicht vorhanden, wird das Movie im Anmerkungsrechteck abgespielt. Der Wert dieser Option ist entweder ein Zoom-Faktor für das Movie oder das folgende Schlüsselwort (Standardwert: Option ist nicht vorhanden, d.h. das Movie wird im Anmerkungsrechteck abgespielt):* ***fullscreen*** *Das Movie wird im gesamten verfügbaren Bildschirmbereich abgespielt.*
zoom	*(Boolean; darf für Text-Anmerkungen im PDF/A-Modus nicht auf* `true` *gesetzt werden) Bei* `true` *wird die Anmerkung in gleichem Maße wie die Seite vergrößert. Bei* `false` *bleibt die Größe der Anmerkung unverändert. Diese Option wird bei den Symbolen für Text-Anmerkungen ignoriert. Standardwert:* `false` *im PDF/A-Modus, sonst* `true`

12.3 Formularfelder

Cookbook *Ein vollständiges Codebeispiel finden Sie im Cookbook-Topic* `webserver/starter_webform`.

C++ Java C# ***void create_field(double llx, double lly, double urx, double ury, String name, String type, String optlist)***

Perl PHP ***create_field(float llx, float lly, float urx, float ury, string name, string type, string optlist)***

C ***void PDF_create_field(PDF *p, double llx, double lly, double urx, double ury, const char *name, int len, const char *type, const char *optlist)***

Erstellt unter Anwendung verschiedener Optionen ein Formularfeld auf der aktuellen Seite.

llx, lly, urx, ury *x*- und *y*-Koordinaten der linken unteren und rechten oberen Ecke des Feldrechtecks in Standardkoordinaten (wenn die Option *usercoordinates* gleich *false* ist) oder in Benutzerkoordinaten (bei *true*).

Beachten Sie, dass sich die Koordinaten für Formularfelder von den Parametern der Funktion *PDF_rect( )* unterscheiden. Während *PDF_create_field( )* explizit die Koordinaten für zwei Ecken erwartet, werden an *PDF_rect( )* die Koordinaten einer Ecke sowie die Breite und die Höhe übergeben.

name (Hypertext-String) Name des Formularfelds, dem eventuell die Namen einer oder mehrerer Gruppen vorangestellt werden, die mit *PDF_create_fieldgroup( )* erstellt wurden. Gruppennamen müssen durch einen Punkt ».« voneinander sowie vom Feldnamen getrennt werden. Feldnamen müssen auf der Seite eindeutig sein und dürfen nicht mit einem Punkt ».« enden.

len (Nur C-Sprachbindung) Länge von *text* (in Bytes). Ist *len* = *0*, muss ein null-terminierter String übergeben werden.

type Feldtyp gemäß Tabelle 12.4.

Tabelle 12.4 Formularfeldtypen

Typ	*Icon*	*Optionen für diesen Typ (zusätzlich zu den allgemeinen Optionen)*
pushbutton (Schaltfläche)		`buttonlayout`, `caption`, `captiondown`, `captionrollover`, `charspacing`, `fitmethod`, `icon`, `icondown`, `iconrollover`, `position`, `submitname`
checkbox (Kontrollkästchen)		`currentvalue`, `itemname`
radiobutton (Optionsfeld)		`buttonstyle`, `currentvalue`, `itemname`, `toggle`, `unisonselect` *Dem Namen muss ein Gruppenname vorangestellt werden, da Radiobuttons immer zu einer Gruppe gehören. Bei allen anderen Feldtypen ist die Gruppenzugehörigkeit optional.*
listbox (Listenfeld)		`charspacing`, `currentvalue`, `itemnamelist`, `itemtextlist`, `multiselect`, `sorted`, `topindex`
combobox (Kombinationsfeld)		`commitonselect`, `charspacing`, `currentvalue`, `editable`, `itemnamelist`, `itemtextlist`, `sorted`, `spellcheck`

Tabelle 12.4 Formularfeldtypen

Typ	*Icon*	*Optionen für diesen Typ (zusätzlich zu den allgemeinen Optionen)*
textfield (Textfeld)		`comb, charspacing, currentvalue, fileselect, maxchar, multiline, password, richtext, scrollable, spellcheck`
		*Textfelder werden auch für Barcodes verwendet:* `barcode`
signature (Digitales Unterschriftsfeld)		`charspacing, lockmode`

optlist Optionsliste mit Feldeigenschaften:

- Allgemeine Optionen: *errorpolicy* (siehe Tabelle 2.1), *hypertextencoding* und *hypertextformat* (siehe Tabelle 2.3)
- Optionen für Feldeigenschaften gemäß Tabelle 12.5. Folgende Optionen werden für alle Feldtypen unterstützt:
  action, alignment, backgroundcolor, barcode, bordercolor, borderstyle, calcorder, dasharray, defaultvalue, display, exportable, fieldtype, fillcolor, font, fontsize, highlight, layer, linewidth, locked, orientate, readonly, required, strokecolor, taborder, tooltip, usercoordinates
- Die in Tabelle 12.4 aufgeführten Optionen werden nur für bestimmte Feldtypen unterstützt. Sie werden ebenfalls in Tabelle 12.5 erläutert.
- (Nicht für *PDF_create_fieldgroup( )*) Option zum vereinfachten Tagging von Strukturelementen gemäß Tabelle 14.5*: tag*

Details Die Tabulatorreihenfolge der Felder auf der Seite (d.h. die Reihenfolge, in der sie beim Drücken der Tabulatortaste den Fokus erhalten) bestimmt sich standardmäßig durch die Reihenfolge der Aufrufe von *PDF_create_field( )*. Mit der Option *taborder* lässt sich eine andere Reihenfolge festlegen. Die Tabulatorreihenfolge ist nach dem Anlegen der Felder nicht mehr änderbar. Das Tab-Verhalten lässt sich jedoch mit der Option *taborder* von *PDF_begin/end_page_ext( )* ändern.

In Acrobat können Textfelder mit einem Format versehen werden (Zahl, Prozentwert etc.). Dies ist nicht im PDF-Standard spezifiziert, sondern wird mit benutzerdefinierten JavaScripts implementiert. Sie können diese Funktionalität nutzen, indem Sie einem Feld JavaScript-Aktionen zuordnen, die auf die vordefinierten (aber nicht standardisierten) JavaScript-Funktionen von Acrobat verweisen (siehe PDFlib-Tutorial).

Bei Tagged PDF erzeugt diese Funktion automatisch ein geeignetes *OBJR*-Element für das erzeugte Formularfeld. Sie müssen vor dem Aufruf dieser Funktion ein neues Container-Element vom Typ *Form* erzeugen (siehe PDFlib-Tutorial).

PDF/A Diese Funktion ist nicht zulässig.

PDF/X Formularfelder sind nur erlaubt, wenn sie vollständig außerhalb der BleedBox liegen (oder der TrimBox/ArtBox, wenn keine BleedBox vorhanden ist).

PDF/UA Wenn diese Funktion aufgerufen wird, muss ein Strukturelement vom Typ *Form* mit *PDF_begin_item( )* oder der Option *tag* erzeugt werden. Die Option *tooltip* ist erforderlich.

Gültigkeit *page*

C++ Java C# **void create_fieldgroup(String name, String optlist)**

Perl PHP **create_fieldgroup(string name, string optlist)**

C **void PDF_create_fieldgroup(PDF *p, const char *name, int len, const char *optlist)**

Erstellt eine Formularfeldgruppe unter Anwendung verschiedener Optionen.

name (Hypertext-String) Name der Formularfeldgruppe; diesem kann der Name einer anderen Gruppe vorangestellt sein. Feldgruppen lassen sich beliebig verschachteln. Gruppennamen müssen durch einen Punkt ».« getrennt werden. Sie müssen innerhalb des Dokuments eindeutig sein und dürfen nicht mit einem Punkt ».« enden.

len (Nur C-Sprachbindung) Länge von *text* (in Bytes). Ist *len* = 0, muss ein null-terminierter String übergeben werden.

optlist Optionsliste mit Feldeigenschaften für *PDF_create_field( )*.

Details Feldgruppen sind nützlich, um den Inhalt eines Feldes in einem oder mehreren anderen Feldern wiederzugeben. Wird ein Feldgruppenname als Präfix für einen mit *PDF_create_field( )* erzeugten Namen übergeben, gehört das neue Feld zu dieser Gruppe. Die für eine Gruppe in *optlist* übergebenen Feldeigenschaften werden von allen zur Gruppe gehörenden Feldern geerbt.

PDF/A Diese Funktion ist nicht zulässig.

PDF/UA Die Option *tooltip* ist erforderlich. Der Font *ZapfDingbats* ist für *type=radiobutton* und *checkbox* erforderlich. Da bei PDF/UA alle Fonts eingebettet werden müssen, muss eine einbettbare Fontdatei für *ZapfDingbats* konfiguriert werden.

Gültigkeit beliebig außer *object*

Tabelle 12.5 Optionen für Feldeigenschaften in PDF_create_field( ) und PDF_create_fieldgroup( )

Option	Beschreibung
action	*(Aktionsliste) Liste der Feldaktionen für eines oder mehrere der folgenden Ereignisse. Das Ereignis* activate *ist für alle Feldtypen zulässig, die anderen Ereignisse sind nicht erlaubt bei* type=pushbutton, checkbox, radiobutton. *Standardwert: leere Liste*
	activate *Aktionen, die bei Aktivierung des Felds durchgeführt werden.*
	blur *Aktionen, die durchgeführt werden, wenn das Feld den Eingabefokus verliert.*
	calculate *JavaScript-Aktionen, die durchgeführt werden, um den Feldwert neu zu berechnen, wenn sich der Wert eines anderen Felds ändert.*
	close *(PDF 1.5) Aktionen, die durchgeführt werden, wenn die Seite mit dem Feld geschlossen wird.*
	down *Aktionen, die durchgeführt werden, wenn die Maustaste im Feldbereich gedrückt wird.*
	enter *Aktionen, die durchgeführt werden, wenn die Maus in den Feldbereich hineinbewegt wird.*
	exit *Aktionen, die durchgeführt werden, wenn die Maus aus dem Feldbereich hinausbewegt wird.*
	focus *Aktionen, die durchgeführt werden, wenn das Feld den Eingabefokus erhält.*
	format *JavaScript-Aktionen, die durchgeführt werden, bevor das Feld zur Anzeige seines aktuellen Wertes formatiert wird. Damit lässt sich der Feldwert vor der Formatierung ändern.*
	invisible *(PDF 1.5) Aktionen, die durchgeführt werden, wenn die Seite mit dem Feld nicht mehr sichtbar ist.*
	keystroke *JavaScript-Aktionen, die durchgeführt werden, wenn der Benutzer Text in ein Text- oder Kombinationsfeld eingibt oder die Auswahl in einem Listenfeld mit Rollbalken ändert.*
	open *(PDF 1.5) Aktionen, die durchgeführt werden, wenn die Seite mit dem Feld geöffnet wird.*
	up *Aktionen, die durchgeführt werden, wenn die Maustaste im Feldbereich losgelassen wird (damit wird ein Feld in der Regel aktiviert).*
	validate *JavaScript-Aktionen, die bei einer Änderung des Feldwertes durchgeführt werden. Damit kann der neue Wert auf Gültigkeit geprüft werden.*
	visible *(PDF 1.5) Aktionen, die durchgeführt werden, wenn die Seite mit dem Feld sichtbar wird.*
alignment	*(Schlüsselwort) Ausrichtung des Texts im Feld:* left, center, right. *Standardwert:* left
backgroundcolor bordercolor	*(Farbe) Farbe des Feldhintergrunds bzw. Feldrands. Unterstützte Farbräume:* none, gray, rgb, cmyk. *Standardwert:* none
barcode	*(Optionsliste; nur für* type=textfield; *impliziert* readonly; *PDF 1.7ext3) Erzeugt gemäß der Optionen in Tabelle 12.6 ein Barcode-Feld. Das Feld sollte die Option* action *mit einem Skript für das Ereignis* calculate *bereitstellen, das den Barcode-Inhalt basierend auf dem Inhalt anderer Felder bestimmt oder einen statischen Wert liefert:* action={calculate=...}. *Der Barcode wird in Acrobat 9 oder höher angezeigt, jedoch nicht in Adobe Reader. Acrobat 9 und X stürzen ab, wenn das erste Feld auf einer Seite ein Barcode-Feld ist. Um dieses Problem zu umgehen, müssen Sie zuerst ein anderes Feld erstellen, bevor Sie das Barcode-Feld hinzufügen. Das erste Feld kann ein einfaches Dummy-Textfeld mit einer Breite und Höhe von Null sein, um den Absturz zu verhindern.*
borderstyle	*(Schlüsselwort) Stil des Feldrands:* solid *(durchgehend),* beveled *(Relief),* dashed *(unterbrochen),* inset *(eingedrückt) oder* underline *(unterstrichen). Standardwert:* solid
buttonlayout	*(Schlüsselwort; nur für* type=pushbutton*) Relative Position der in* caption *übergebenen Schaltflächenbeschriftung zum in* icon *übergebenen Schaltflächensymbol, sofern beide auch festgelegt wurden:* below *(unterhalb),* above *(oberhalb),* right *(rechts),* left *(links),* overlaid *(übereinander). Standardwert:* right
buttonstyle	*(Schlüsselwort; nur für* type=radiobutton *und* checkbox*) Legt das Symbol für das Feld fest:* check *(Häkchen),* cross *(Kreuz),* diamond *(Karo),* circle *(Kreis),* star *(Stern),* square *(Quadrat). Standardwert:* check

Tabelle 12.5 Optionen für Feldeigenschaften in PDF_create_field( ) und PDF_create_fieldgroup( )

Option	Beschreibung
calcorder	*(Integer; nur bei vorhandener JavaScript-Aktion für das Ereignis* `calculate`*) Legt die Berechnungsreihenfolge des Feldes relativ zu anderen Feldern fest. Felder mit niedrigen Nummern werden vor Feldern mit höheren Nummern berechnet. Standardwert: 10 plus der auf der aktuellen Seite größte verwendete* `calcorder`*-Wert (anfangs 10)*
caption	*(Content-String; nur für* `type=pushbutton`*; eine der Optionen* `caption` *oder* `icon` *muss für Schaltflächen übergeben werden) Beschriftung, die angezeigt wird, wenn die Schaltfläche keinen Eingabefokus hat. Sie wird in dem in der Option* `font` *angegebenen Font angezeigt. Verwenden Sie einen leeren String (d.h.* `caption { }`*), wenn Sie keine Beschriftung und kein Symbol wünschen. Standardwert: nicht vorhanden*
captiondown	*(Content-String; nur für* `type=pushbutton`*) Beschriftung, die angezeigt wird, wenn die Schaltfläche aktiviert ist. Sie wird in dem in der Option* `font` *angegebenen Font angezeigt. Standardwert: nicht vorhanden*
captionrollover	*(Content-String; nur für* `type=pushbutton`*) Beschriftung, die angezeigt wird, wenn die Schaltfläche den Eingabefokus hat. Sie wird in dem in der Option* `font` *angegebenen Font angezeigt. Standardwert: nicht vorhanden*
charspacing	*(Float; nicht für* `type=radiobutton, checkbox`*) Wortabstand für Text im Feld in Einheiten des aktuellen Benutzerkoordinatensystems. Standardwert: 0*
comb	*(Boolean; nur für* `type=textfield`*; PDF 1.5) Bei* `true` *und wenn die Optionen* `multiline`*,* `fileselect` *und* `password` *gleich* `false` *sind und die Option* `maxchar` *mit einem Integer-Wert übergeben wurde, wird das Feld in eine durch* `maxchar` *bestimmte Anzahl von Unterfeldern gleichen Abstandes zur Aufnahme einzelner Zeichen unterteilt. Standardwert:* `false`
commitonselect	*(Boolean; nur für* `type=listbox, combobox`*; PDF 1.5) Bei* `true` *wird die Selektion eines Listeneintrags sofort festgeschrieben. Bei* `false` *geschieht dies erst beim Verlassen des Feldes. Standardwert:* `false`
currentvalue	*(Nicht für* `type=pushbutton, signature`*) Anfangswert des Feldes. Typ und Standardwert hängen vom Feldtyp ab:* ***checkbox, radiobutton*** *(String) Jeder von* `Off` *verschiedene String bedeutet, dass der Button aktiviert ist. Der String* `Off` *bedeutet, dass der Button deaktiviert ist. Diese Option sollte für den ersten Button gesetzt werden. Standardwert:* `Off` ***textfield, combobox*** *(Content-String) Inhalt des Felds. Er wird in dem in der Option* `font` *angegebenen Font angezeigt. Standardwert: leer* ***listbox*** *(Integer-Liste) Indizes der in* `itemtextlist` *selektierten Einträge. Standardwert: nicht vorhanden*
dasharray	*(Liste aus zwei nicht negativen Floats; nur für* `borderstyle=dashed`*). Länge der Striche und Lücken eines gestrichelten Randes in Standardeinheiten (siehe Tabelle 7.1). Standardwert: 3 3*
defaultvalue	*Der Feldwert nach der Aktion* `ResetForm`*. Typen und Standardwerte entsprechen der Option* `currentvalue`*. Ausnahme: bei Listenfeldern ist nur ein einzelner Integer-Wert erlaubt.*
display	*(Schlüsselwort) Sichtbarkeit auf Bildschirm und Papier;* `visible` *(sichtbar),* `hidden` *(unsichtbar),* `noview` *(unsichtbar, aber Drucken möglich),* `noprint` *(sichtbar, aber Drucken nicht möglich). Standardwert:* `visible`
editable	*(Boolean; nur für* `type=combobox`*) Bei* `true` *kann der momentan im Feld ausgewählte Text editiert werden. Standardwert:* `false`
exportable	*(Boolean) Das Feld wird exportiert, wenn die Aktion* `SubmitForm` *durchgeführt wird. Standardwert:* `true`

Tabelle 12.5 Optionen für Feldeigenschaften in PDF_create_field( ) und PDF_create_fieldgroup( )

Option	Beschreibung
fieldtype	*(Schlüsselwort; nur für* PDF_create_fieldgroup( )*) Typ der in der Gruppe enthaltenen Felder:* mixed, pushbutton, checkbox, radiobutton, listbox, combobox, textfield *oder* signature. *Außer bei* fieldtype=mixed *darf die Gruppe nur Felder des angegebenen Typs enthalten. Wurde ein bestimmter Feldtyp festgelegt, so wird der aktuelle Wert in allen Feldern der Gruppe gleichzeitig angezeigt, auch wenn sich die Felder auf unterschiedlichen Seiten befinden. Bei* fieldtype=radiobutton *muss die Option* unisonselect *übergeben werden. Die Optionen* itemtextlist, itemnamelist, currentvalue *und* defaultvalue *müssen in den Optionen für* PDF_create_fieldgroup( ) *und nicht in den Optionen für* PDF_create_field( ) *angegeben werden. Standardwert:* mixed
fileselect	*(Boolean; nur für* type=textfield*) Bei* true *wird der Text im Feld wie ein Dateiname behandelt. Standardwert:* false
fillcolor	*(Farbe) Füllfarbe für Text. Unterstützte Farbräume:* gray, rgb, cmyk. *Standardwert:* {gray 0} *(=schwarz)*
fitmethod	*(Schlüsselwort; nur für* type=pushbutton*) Methode zur Platzierung eines Templates, das mit den Optionen* icon, icondown *und* iconrollover *in der Schaltfläche übergeben wird. Unterstützte Schlüsselwörter (Standardwert:* meet*):* **auto** *wie* meet*, wenn das Template in die Schaltfläche hineinpasst, sonst* clip **nofit** *wie* clip **clip** *Template wird nicht skaliert, sondern am Feldrand beschnitten* **meet** *Template wird proportional skaliert, so dass es in die Schaltfläche hineinpasst* **slice** *wie* meet **entire** *Template wird skaliert, so dass es die Schaltfläche vollständig ausfüllt*
font	*(Font-Handle) Legt den Font für das Feld fest. Acrobat kann auch Zeichen anzeigen, die nicht im Encoding des Fonts enthalten sind. Sie können zum Beispiel* encoding=winansi *verwenden und dennoch Unicode-Zeichen außerhalb von* winansi *übergeben. Die Fontverwendung hängt vom Feldtyp ab:* ▸ *Felder mit* type=listing, combobox *oder* textfield*: diese Option ist erforderlich.* ▸ *Für* type=pushbutton *ist diese Option nur erforderlich, wenn eine oder mehrere der Optionen* caption, captionrollover *oder* captiondown *angegeben werden.* ▸ *Felder mit* type=radiobutton *oder* checkbox *verwenden immer ZapfDingbats.*
fontsize	*(Fontgröße) Fontgröße in Benutzerkoordinaten. Der Wert 0 oder das Schlüsselwort* auto *bedeutet, dass Acrobat die Fontgröße automatisch an das Rechteck anpasst. Standardwert:* auto
highlight	*(Schlüsselwort) Hervorheben-Modus des Felds, wenn es angeklickt wird:* none *(kein),* invert *(negativ),* outline *(Umrandung),* push *(Schaltfläche). Standardwert:* invert
icon	*(Template-Handle[1]; nur für* type=pushbutton*; eine der Optionen* caption *oder* icon *muss für Schaltflächen übergeben werden) Handle für ein Template, das angezeigt wird, wenn die Schaltfläche keinen Eingabefokus besitzt. Standardwert: nicht vorhanden*
icondown	*(Template-Handle[1]; nur für* type=pushbutton*) Handle für ein Template, das angezeigt wird, solange die Schaltfläche aktiviert ist. Standardwert: nicht vorhanden*
iconrollover	*(Template-Handle[1]; nur für* type=pushbutton*) Handle für ein Template, das angezeigt wird, solange die Schaltfläche den Eingabefokus hat. Standardwert: nicht vorhanden*
itemname	*(Hypertext-String; nur für* type=radiobutton, checkbox*; muss verwendet werden, wenn der Exportwert kein Latin-1-String ist.) Exportwert des Felds. Bei mehreren zu einer Gruppe gehörenden Radiobuttons können die in* itemname *übergebenen Werte identisch sein. Standardwert: Feldname*
itemnamelist	*(Hypertext-String; nur für* type=listbox, combobox*) Exportwerte der Listeneinträge. Mehrere Einträge können denselben Exportwert besitzen. Standardwert: nicht vorhanden*
itemtextlist	*(Liste aus Content-Strings; nur für* type=listbox *oder* combobox *und dann erforderlich) Textinhalte für alle Listeneinträge. Wenn die beiden Optionen* itemnamelist *und* itemtextlist *übergeben werden, so müssen sie dieselbe Anzahl von Strings enthalten.*

Tabelle 12.5 Optionen für Feldeigenschaften in PDF_create_field( ) und PDF_create_fieldgroup( )

Option	Beschreibung
layer	*(Layer-Handle; PDF 1.5) Ebene, zu der das Feld gehört. Das Feld ist nur sichtbar, wenn auch die Ebene sichtbar ist.*
linewidth	*(Integer) Linienbreite des Feldrands in Standardeinheiten. Standardwert: 1*
locked	*(Boolean) Bei* `true` *lassen sich die Feldeigenschaften in Acrobat nicht bearbeiten. Standardwert:* `false`
lockmode	*(Schlüsselwort; nur für* `type=signature`*; PDF 1.5) Bezeichnet die Felder, die gesperrt werden sollen, sobald das Feld unterzeichnet ist:* ***all*** *Alle Felder im Dokument werden gesperrt.*
maxchar	*(Integer oder Schlüsselwort; nur für* `type=textfield`*) Maximale Anzahl der Zeichen im Feld oder das Schlüsselwort* `unlimited`*, wenn keine Beschränkung existiert. Standardwert:* `unlimited`
multiline	*(Boolean; nur für* `type=textfield`*) Bei* `true` *wird der Text gegebenenfalls in mehrere Zeilen umbrochen. Standardwert:* `false`
multiselect	*(Boolean; nur für* `type=listbox`*) Bei* `true` *können mehrere Listeneinträge selektiert werden. Standardwert:* `false`
orientate	*(Schlüsselwort) Ausrichtung der Inhalte innerhalb des Feldrechtecks:* `north`, `west`, `south`, `east`. *Standardwert:* `north`
password	*(Boolean; nur für* `type=textfield`*) Bei* `true` *wird der Text bei der Eingabe mit Punkten oder Sternchen angezeigt. Standardwert:* `false`
position	*(Float- oder Schlüsselwort-Liste; nur für* `type=pushbutton`*) Ein oder zwei Werte für die Position eines Templates innerhalb des Feldrechtecks, das mit einer der Optionen* `icon`*... übergeben wurde. {0 0} bezeichnet dabei die linke untere Ecke der Textbox und {100 100} die rechte obere Ecke. Die Werte werden in Prozent der Breite und Höhe des Feldrechtecks angegeben. Sind die beiden Prozentwerte gleich, so genügt die Angabe eines einzigen Float-Wertes.* *Statt der Werte 0, 50 und 100 können dementsprechend die Schlüsselwörter* `left`, `center`*, und* `right` *(in x-Richtung) oder* `bottom`, `center` *und* `top` *(in y-Richtung) angegeben werden. Wird nur ein einziges Schlüsselwort angegeben, wird für die andere Richtung das entsprechende Schlüsselwort verwendet. Standardwert:* `{center}`*. Beispiele:* `{0 50}` *oder* `{left center}` *linksbündig ausgerichtetes Template* `{50 50}` *oder* `{center}` *mittig ausgerichtetes Template* `{100 50}` *oder* `{right center}` *rechtsbündig ausgerichtetes Template*
readonly[2]	*(Boolean) Bei* `true` *ist keine Eingabe in das Feld erlaubt. Standardwert:* `false`
required	*(Boolean) Bei* `true` *muss das Feld beim Senden des Formulars einen Wert enthalten. Standardwert:* `false`
richtext	*(Boolean; nur für* `type=textfield`*; PDF 1.5) Erlaubt Rich-Text-Formatierung. Bei* `true` *muss* `fontsize` *ungleich 0 sein und* `fillcolor` *darf nicht mit dem Farbraum* `cmyk` *definiert sein. Standardwert:* `false`
scrollable	*(Boolean; nur für* `type=textfield`*) Bei* `true` *wird zu langer Text in den unsichtbaren Bereich außerhalb des Feldes bewegt. Bei* `false` *wird keine Eingabe mehr akzeptiert, sobald der Text das Feld vollständig ausfüllt. Standardwert:* `true`
sorted	*(Boolean; nur für* `type=listbox` *und* `combobox`*) Bei* `true` *werden die Listeneinträge sortiert. Standardwert:* `false`
spellcheck	*(Boolean; nur für* `type=textfield` *und* `combobox`*) Bei* `true` *wird die Rechtschreibprüfung für das Feld aktiviert. Standardwert:* `true`
strokecolor	*(Farbe) Farbe zum Zeichnen des Texts. Unterstützte Farbräume:* `gray`, `rgb`, `cmyk`. *Standardwert:* `{gray 0}` *(=schwarz).*
submitname	*(Hypertext-String; nur empfohlen für* `type=pushbutton`*) URL-kodierter String der Internet-Adresse, an die das Formular gesendet wird. Standardwert: nicht vorhanden*

Tabelle 12.5 Optionen für Feldeigenschaften in PDF_create_field( ) und PDF_create_fieldgroup( )

Option	Beschreibung
taborder	*(Integer) Legt die Tabulatorreihenfolge des Feldes relativ zu anderen Feldern fest. Felder mit niedrigen Nummern werden vor Feldern mit höheren Nummern betreten. Standardwert: 10 plus dem maximalen auf der Seite verwendeten* `taborder`*-Wert (und 10 für das erste Feld auf der Seite); im Standardfall richtet sich die Tabulatorreihenfolge also nach der Erstellungsreihenfolge.*
toggle	*(Boolean; nur für PDF_create_fieldgroup( ) und* `type=radiobutton`*) Bei* `true` *lässt sich ein Radiobutton innerhalb einer Gruppe durch Anklicken sowohl aktivieren als auch deaktivieren. Bei* `false` *kann er durch Anklicken nur aktiviert werden, eine Deaktivierung ist nur durch Anklicken eines anderen Buttons möglich. Standardwert:* `false`
tooltip[2]	*(Hypertext-String; bei PDF/UA darf der Wert nicht leer sein) Text, der im Tooltip des Feldes erscheint und auch von Screenreadern verwendet wird. Bei Radiobuttons und -gruppen verwendet Acrobat immer den* `tooltip`*-Wert des ersten Buttons der Gruppe, die anderen werden ignoriert. Standardwert: nicht vorhanden*
topindex	*(Integer; nur für* `type=listbox`*) Index des ersten sichtbaren Eintrags. Das erste Element hat den Index 0. Standardwert: 0*
unisonselect	*(Boolean; nur für PDF_create_fieldgroup( ),* `type=radiobutton` *und PDF 1.5) Bei* `true` *werden Radiobuttons mit gleichem Feld- oder Elementnamen gleichzeitig selektiert. Standardwert:* `false`
user-coordinates	*(Boolean) Bei* `false` *werden Feldkoordinaten im Standardkoordinatensystem erwartet; anderenfalls wird das aktuelle Benutzerkoordinatensystem verwendet. Standardwert: Wert der globalen Option* `user-coordinates`

1. Templates für Symbole lassen sich mit PDF_begin_template_ext( ) erzeugen; besteht das Symbol nur aus einem Bild, so können Sie das Template mit der Option `template` *von PDF_load_image( ) erstellen.*
2. Für `type=radiobutton` *sollte diese Option nicht mit PDF_create_field( ), sondern nur mit PDF_create_fieldgroup( ) verwendet werden.*

Tabelle 12.6 Unteroptionen für die Option `barcode` *von PDF_create_field( ) und PDF_create_fieldgroup( )*

Option	Beschreibung
caption	*(Hypertext-String) Die Beschriftung wird unterhalb des Barcodes platziert. Standardmäßig verwendet Acrobat den URL des Dokuments mit dem Präfix* `file:`**.**
dataprep	*(Integer) Mögliche Datenvorbereitung. Unterstützte Werte (Standardwert: 0):* **0** *Keine Kompression vor dem Kodieren der Daten im Barcode.* **1** *Kompression mit dem Flate-Algorithmus vor dem Kodieren der Daten.*
ecc	*(Integer; erforderlich für* `symbology=PDF417` *und* `QRCode`*) Koeffizient für die Fehlerkorrektur, wobei höhere Werte durch Redundanz eine bessere Fehlerkorrektur bedeuten, aber einen größeren Barcode erfordern. Für* `symbology=PDF417` *müssen die Werte im Bereich 0-8 liegen; für* `symbology=QRCode` *müssen die Werte im Bereich 0-3 liegen.*
resolution	*(Positiver Integer) Auflösung in dpi, mit der der Barcode dargestellt wird (Standardwert: 300)*
symbology	*(Schlüsselwort; erforderlich) Zu verwendende Barcode-Technologie:* **PDF417** *PDF417-Barcode gemäß ISO 15438* **QRCode** *QR-Code-2005-Barcode gemäß ISO 18004* **DataMatrix** *Data-Matrix-Barcode gemäß ISO 16022*
xsymheight	*(Integer; nur für* `symbology=PDF417` *und dann erforderlich) Vertikaler Abstand zwischen zwei Barcode-Modulen in Pixeln. Das Verhältnis zwischen* `xsymheight/xsymwidth` *muss ganzzahlig sein. Der erlaubte Bereich dafür ist 1-4.*
xsymwidth	*(Integer; erforderlich) Horizontaler Abstand zwischen zwei Barcode-Modulen in Pixeln*

12.4 Aktionen

C++ Java C# ***int create_action(String type, String optlist)***

Perl PHP ***int create_action(string type, string optlist)***

C ***int PDF_create_action(PDF *p, const char *type, const char *optlist)***

Erzeugt eine Aktion, die auf verschiedene Objekte und Ereignisse angewendet werden kann.

type Typ der Aktion gemäß Tabelle 12.7.

Tabelle 12.7 Aktionstypen

Typ	*Beschreibung; für diesen Typ erlaubte Optionen*
GoTo	*Gehe zu einem Ziel im aktuellen Dokument:* `destination`, `destname`
GoTo3DView	*(PDF 1.6) Darstellung einer 3D-Animation festlegen:* `3Dview`, `target`
GoToE	*(PDF 1.6) Gehe zu einem Ziel in einem eingebetteten Dokument:* `destination`, `destname`, `filename`, `newwindow`, `targetpath`
GoToR	*Gehe zu einem Ziel in einem anderen (entfernten) Dokument:* `destination`, `destname`, `filename`, `newwindow`
Hide	*Notiz oder Formularfeld ein-/ausblenden:* `hide`, `namelist`
ImportData	*Formulardaten aus einer Datei importieren:* `filename`
JavaScript	*Skript mit JavaScript-Code ausführen:* `script`, `scriptname`
Launch	*Datei öffnen bzw. Anwendung ausführen:* `defaultdir`, `filename`, `newwindow`, `operation`, `parameters`
Movie	*Externe Sound- oder Movie-Datei in einem überlagerten Wiedergabefenster oder innerhalb des Rechtecks einer Movie-Anmerkung abspielen:* `operation`, `target`
Named	*Acrobat-Menübefehl ausführen, der durch seinen Namen identifiziert wird:* `menuname`
ResetForm	*Bestimmte oder alle Formularfelder auf ihre Standardwerte zurücksetzen:* `exclude`
RichMedia-Execute	*(PDF 1.7ext3) Einen Befehl an eine Anmerkung vom Typ* `RichMedia` *senden:* `functionname`, `instance`, `richmediaargs`, `target`
SetOCGState	*(PDF 1.5) Ebenen ein- und ausblenden:* `layerstate`, `preserveradio`
SubmitForm	*Formulardaten an einen Uniform Resource Locator, also eine Internet-Adresse senden (beachten Sie, dass in Acrobat kein Senden mit Basic Authentication möglich ist):* `canonicaldate`, `exclude`, `exportmethod`, `submitemptyfields`, `url`
Trans	*(PDF 1.5) Seitenanzeige mit visuellen Effekten aktualisieren. Dies kann zur Steuerung der Anzeige innerhalb einer Folge mehrerer Aktionen nützlich sein:* `duration`, `transition`
URI	*Uniform Resouce Identifier auflösen, d.h. Webverknüpfung öffnen:* `ismap`, `url`

optlist Optionsliste mit Aktionseigenschaften:

- Allgemeine Optionen: *errorpolicy* (siehe Tabelle 2.1) und *hypertextencoding* (siehe Tabelle 2.3)
- Folgende typspezifische Optionen gemäß Tabelle 12.8 werden für bestimmte Aktionstypen gemäß Tabelle 12.7 unterstützt:
  3Dview, canonicaldate, defaultdir, destination, destname, duration, exclude, export-

method, filename, functionname, hide, instance, ismap, layerstate, menuname, namelist, newwindow, operation, parameters, preserveradio, richmediaargs, script, scriptname, submitemptyfields, target, targetpath, transition, url

Rückgabe Aktions-Handle, mit dem Aktionen an Objekte im Dokument geknüpft werden können. Es kann bis zum Ende des umschließenden Geltungsbereichs *document* verwendet werden.

Details Diese Funktion erstellt eine Einzelaktion. Objekte (wie Seiten, Ereignisse für Formularfelder oder Lesezeichen) können auch mit mehreren Aktionen versehen werden, aber jede Aktion muss zuvor mit einem eigenen Aufruf von *PDF_create_action( )* generiert werden. Eine Aktion kann an verschiedene Objekte geknüpft werden. Wir empfehlen, die bestehenden Handles wiederzuverwenden, wenn eine Aktion mit den gleichen Optionen bereits zuvor erstellt wurde.

PDF/A Es sind nur folgende Aktionstypen zulässig:
GoTo, GoToE, GoToR, Named, RichMediaExecute, SubmitForm, URI

PDF/X Diese Funktion ist nicht zulässig.

PDF/UA Die Option *ismap=true* ist nicht zulässig.

Gültigkeit beliebig außer *object*. Das zurückgegebene Handle kann bis zum nächsten Aufruf von *PDF_end_document( )* verwendet werden.

Tabelle 12.8 Optionen für Aktionseigenschaften mit PDF_create_action( )

Option	Beschreibung
3Dview	*(Schlüsselwort oder 3D-View-Handle;* `GoTo3DView`*; erforderlich) Bestimmt die Darstellung einer 3D-Anmerkung; Eins der Schlüsselwörter* `first`, `last`, `next`, `previous` *(mit Bezug auf die entsprechenden Einträge in der Option* `views` *der Anmerkung) oder* `default` *(mit Bezug auf die Option* `defaultview` *der Anmerkung), oder ein 3D-View-Handle, das mit PDF_create_3dview( ) erstellt wurde.*
canonical-date	*(Boolean;* `SubmitForm`*) Bei* `true` *werden gesendete Formularfeldwerte, die ein Datum darstellen, in ein Standardformat konvertiert. Ob oder wann ein Feld ein Datum darstellt, wird nicht explizit im Feld selbst definiert, sondern im verarbeitenden JavaScript-Code. Standardwert:* `false`
defaultdir	*(String;* `Launch`*) Setzt das Standardverzeichnis der auszuführenden Anwendung. Diese Option wird nur von Acrobat unter Windows unterstützt. Standardwert: nicht vorhanden*
destination	*(Optionsliste;* `GoTo`*,* `GoToE`*,* `GoToR`*; erforderlich, sofern nicht* `destname` *übergeben wird) Optionsliste zur Definition des Sprungziels gemäß Tabelle 12.10.*
destname	*(Hypertext-String)* `GoTo` *(erforderlich, sofern nicht* `destination` *übergeben wird): Name eines Ziels, das mit PDF_add_nameddest( ) definiert wurde. Das Ziel kann vor oder nach Einfügen des Verweises erstellt werden.* `GoToR`*,* `GoToE` *(erforderlich, sofern nicht* `destination` *übergeben wird): Name eines Ziels im entfernten oder eingebetteten Dokument.*
duration	*(Float;* `Trans`*) Bestimmt die Dauer eines Übergangseffekts in Sekunden für die aktuelle Seite. Standardwert: 1*

Tabelle 12.8 Optionen für Aktionseigenschaften mit PDF_create_action()

Option	Beschreibung
exclude	*(Boolean)* `SubmitForm`: *Bei* `true` *werden in der Option* `namelist` *die auszuschließenden Felder aufgeführt; es werden alle Felder im Dokument gesendet, mit Ausnahme der in* `namelist` *aufgeführten und der Felder, deren Option* `exportable` *auf* `false` *gesetzt ist. Bei* `false` *werden in der Option* `namelist` *die zu sendenden Felder aufgeführt. Feldgruppen werden mit all ihren Elementen gesendet. Standardwert:* `false` `ResetForm`: *Bei* `true` *werden in der Option* `namelist` *die auszuschließenden Felder aufgeführt; es werden alle Felder im Dokument zurückgesetzt mit Ausnahme der in* `namelist` *aufgeführten. Bei* `false` *werden in der Option* `namelist` *die zurückzusetzenden Felder aufgeführt. Feldgruppen werden mit all ihren Elementen zurückgesetzt. Standardwert:* `false`
exportmethod	*(Schlüsselwortliste;* `SubmitForm`*) Steuert, auf welche Art Formularfeldnamen und -werte gesendet werden. (Standardwert:* `fdf`*):* ***html, fdf, xfdf, pdf*** *im Format HTML, FDF, XFDF oder PDF* ***annotfields*** *(Nur für* `fdf`*) Enthält alle Anmerkungen und Felder.* ***coordinate*** *(Nur für* `html`*) Die Koordinaten des Mausklicks, der die Aktion* `submitform` *auslöste, werden in den Formulardaten mitgesendet. Die Koordinatenwerte beziehen sich auf die linke obere Ecke des Feldrechtecks.* ***exclurl*** *(Nur für* `fdf`*) Im gesendeten FDF ist der URL-String nicht enthalten.* ***getrequest*** *(Nur für* `html` *und* `pdf`*) Senden mit HTTP GET; anderenfalls HTTP POST* ***onlyuser*** *(Nur für* `fdf` *und* `annotfields`*) Beim Senden werden nur Anmerkungen berücksichtigt, deren Name mit dem Namen des aktuellen Benutzers übereinstimmt, der vom entfernten Server, zu dem das Feld gesendet werden soll, festgelegt wird.* ***updates*** *(Nur für* `fdf`*) Bezieht die Inhalte aller inkrementellen Updates des zugrundeliegenden PDF-Dokuments ein* *Beispiel für kombinierte Optionen:* `exportmethod {fdf updates onlyuser}`
filename	*(Hypertext-String)* `GoToR`, `Launch` *(erforderlich): Name einer externen (PDF- oder anderen) Datei oder Anwendung, die beim Auslösen der Aktion geöffnet wird. Die Notation für UNC-Dateinamen ist* `\\server\volume`. `ImportData` *(erforderlich): Name der externen Datei mit den Formulardaten.* `GoToE`: *Name des Stammdokuments des Ziels relativ zum Stammdokument der Quelle. Fehlt dieser Eintrag, haben Quelle und Ziel das selbe Stammdokument*
functionname	*(Hypertext-String;* `RichMediaExecute`*; erforderlich) String mit dem Script-Befehl in Form eines einfachen ActionScript- oder JavaScript-Funktionsnamens (kein vollständiges Script). Wenn die in der Option* `instance` *angegebene Zielinstanz Flash-Inhalte enthält, handelt es sich bei dem Befehls-String um einen ActionScript-Aufruf* `ExternalInterface` *an den Skript-Engine-Kontext der Zielinstanz. Handelt es sich bei der Zielinstanz um ein 3D-Modell, wird der Aufruf im globalen Kontext der Anmerkungsinstanz der 3D-JavaScript-Engine ausgeführt.*
hide	*(Boolean;* `Hide`*) Blendet Anmerkungen aus* (`true`) *oder ein* (`false`)*. Standardwert:* `true`
instance	*(Integer;* `RichMediaExecute`*) Index der Optionsliste (bei 1 beginnend) in der Unteroption* `instances` *der Unteroption* `configuration` *der Option* `richmedia` *von PDF_create_annotation() zur Angabe einer Flash- oder 3D-Instanz einer Rich-Media-Annotation, für die das Script ausgeführt wird. Standardwert: 1*
ismap	*(Boolean;* `URI`*; bei PDF/UA ist* `true` *nicht zulässig) Bei* `true` *werden die Koordinaten der Mausposition bei der Auflösung des URL zum Ziel-URI hinzugefügt. Standardwert:* `false`

Tabelle 12.8 Optionen für Aktionseigenschaften mit PDF_create_action( )

Option	Beschreibung
layerstate	*(Optionsliste;* `SetOCGState`*; erforderlich) Liste von Paaren, wobei jedes Paar aus einem Schlüsselwort und einem Ebenen-Handle besteht. Folgende Schlüsselwörter werden unterstützt:* ***on*** *Ebene aktivieren* ***off*** *Ebene deaktivieren* ***toggle*** *Zustand der Ebene umkehren. Bei dieser Option sollte die Option* `preserveradio` *auf* `false` *gesetzt werden.*
menuname	*(String;* `Named`*; erforderlich) Name des auszuführenden Menübefehls. Im PDF/A-Modus sind nur die Werte* `nextpage`*,* `prevpage`*,* `firstpage` *und* `lastpage` *zulässig. Sonst sind auch andere Namen erlaubt. Ein vollständiges Codebeispiel zur Ermittlung der Namen für andere Menübefehle finden Sie im Cookbook-Topic* `interactive/acrobat_menu_items`*.*
namelist	*(String-Liste;* `Hide`*; erforderlich) Namen (einschließlich Gruppennamen) der Anmerkungen oder Felder, die aus- oder eingeblendet werden sollen.* (`SubmitForm`) *Namen (einschließlich Gruppennamen) der Formularfelder, die abhängig von der Option* `exclude` *in den Sendevorgang einbezogen oder davon ausgeschlossen werden sollen. Standardwert: Alle Felder außer denjenigen, deren Option* `exportable` *auf* `false` *gesetzt ist, werden gesendet.* (`ResetForm`) *Namen (einschließlich Gruppennamen) der Formularfelder, die abhängig von der Option* `exclude` *zurückgesetzt oder nicht zurückgesetzt werden. Standardwert: Alle Felder werden zurückgesetzt.*
newwindow	*(Boolean;* `GoToE`**,** `GoToR`*) Legt fest, ob das Zieldokument in einem eigenen Fenster geöffnet wird. Bei* `newwindow=false` *wird das aktuelle Dokument im selben Fenster durch das Zieldokument ersetzt.* `Launch`**:** *Dieser Eintrag wird ignoriert, wenn die Datei kein PDF-Dokument ist. Standardwert: Acrobat verhält sich so, wie es in den Grundeinstellungen festgelegt ist.*
operation	*(Schlüsselwort;* `Launch`*) Legt die Operation fest, die auf das in der Option* `filename` *festgelegte Dokument angewandt werden soll. Diese Option wird nur von Acrobat unter Windows unterstützt. Bezieht sich die Option* `filename` *auf eine Anwendung und nicht auf ein Dokument, so wird diese Option ignoriert und die Anwendung gestartet. Unterstützte Schlüsselwörter (Standardwert:* `open`*):* ***open*** *Dokument öffnen* ***print*** *Dokument drucken* *(Schlüsselwort;* `Movie`*) Legt die Operation fest, die auf das Movie oder den Sound angewandt wird. Unterstützte Schlüsselwörter (Standardwert:* `play`*):* ***play*** *Abspielen des Movies in dem Modus starten, der in der Option* `playmode` *der Movie-Anmerkung festgelegt ist. Das Movie wird vor dem Abspielen an den Anfang zurückgesetzt, falls es angehalten wurde.* ***stop*** *Abspielen des Movies beenden* ***pause*** *Abspielen des Movies anhalten* ***resume*** *Mit dem Abspielen des Movies fortfahren*
parameters	*(String;* `Launch`*) Parameter-String, der an die in der Option* `filename` *angegebene Anwendung übergeben wird. Diese Option wird von Acrobat nur unter Windows unterstützt. Parameter können durch Leerzeichen voneinander getrennt werden, die Parameter selbst dürfen keine Leerzeichen enthalten. Diese Option ist wirkungslos, wenn sich die Option* `filename` *auf ein Dokument bezieht. Standardwert: nicht vorhanden*
preserve-radio	*(Boolean;* `SetOCGState`*) Bei* `true` *bleibt die Radiobutton-Relation zwischen verschiedenen Ebenen erhalten. Standardwert:* `true`
richmediaargs	*(POCA-Container-Handle;* `RichMediaExecute`*) Handle für einen Array-Container, der eine beliebige Anzahl von Argumenten für den Befehl enthält. Gültige Argumente sind Objekte vom Typ String, Integer, Float oder Boolean. Das Array muss mit der Option* `usage=richmediaargs` *erzeugt worden sein. Standardwert: keine Argumente*

Tabelle 12.8 Optionen für Aktionseigenschaften mit PDF_create_action( )

Option	Beschreibung
script	*(Hypertext-String;* `JavaScript`*; erforderlich) String mit dem auszuführenden JavaScript-Code. Um mit dieser Option beliebige Argumente zu übergeben, kann die Syntax für Optionslisten nützlich sein, die im Abschnitt »Nicht eingeschlossene String-Werte in Optionslisten«, Seite 10 erläutert wird.*
scriptname	*(Hypertext-String;* `JavaScript`*) Wenn vorhanden, wird das JavaScript in der Option* `script` *mit dem Namen in der Option* `scriptname` *als JavaScript-Code auf Dokumentebene eingefügt. Wird in einem Dokument derselbe Scriptname mehrmals übergeben, so wird nur das letzte Script verwendet. JavaScripts auf Dokumentebene werden nach dem Laden des Dokuments in Acrobat ausgeführt. Dies kann bei Skripten in Formularfeldern nützlich sein.*
submit-emptyfields	*(Boolean;* `SubmitForm`*) Bei* `true` *werden alle durch die Optionen* `namelist` *und* `exclude` *festgelegten Felder gesendet, unabhängig davon, ob sie einen Wert besitzen oder nicht. Bei Feldern ohne Wert wird nur der Feldname übertragen. Bei* `false` *wird ein Feld nur gesendet, wenn es einen Wert enthält. Standardwert:* `false`
target	*(String;* `GoTo3DView`*,* `Movie`*;* `RichMediaExecute`*; erforderlich) Name der 3D-, Movie- oder Rich-Media-Annotation, der in der Option* `name` *von PDF_create_annotation( ) definiert wurde*
targetpath	*(Optionsliste;* `GoToE`*; erforderlich, sofern* `filename` *nicht angegeben wird) Ziel-Optionsliste (siehe Tabelle 12.9) mit Pfadinformationen für das Zieldokument. Jede Ziel-Optionsliste legt ein Element im vollständigen Pfad zum Ziel fest und kann weitere Ziel-Optionslisten mit zusätzlichen Elementen enthalten.*
transition	*(Schlüsselwort;* `Trans`*) Bestimmt den Übergangseffekt; eine Liste der Schlüsselwörter finden Sie in Tabelle 3.9. Standardwert:* `replace`
url	*(String;* `URI` *und* `SubmitForm`*; erforderlich) Uniform Resource Locator in 7-Bit-ASCII oder EBCDIC (aber nur ASCII-Zeichen) für das Verknüpfungsziel (für* `type=URI`*) oder die Adresse des Scripts auf dem Webserver, das für den Sendevorgang zuständig ist (für* `type=SubmitForm`*). Er kann auf eine beliebige lokale oder Webressource verweisen und muss mit einer Protokollidentifikation beginnen (zum Beispiel* `http://`*).*

Tabelle 12.9 Unteroptionen für die Option `targetpath` *von PDF_create_action( )*

Option	Beschreibung
annotation	*(Hypertext-String; erforderlich bei* `relation=child` *und wenn das Ziel mit einer Anmerkung für einen Dateianhang verknüpft ist) Gibt den Namen der Dateianhangs-Anmerkung des Ziels auf der mit* `pagenumber` *oder* `destname` *angegebenen Seite an.*
destname	*(Hypertext-String; erforderlich, sofern* `pagenumber` *nicht übergeben wurde und nicht* `relation=child` *und das Ziel nicht mit einer Anmerkung für einen Dateianhang verknüpft ist) Legt ein benanntes Ziel für eine Seite im aktuellen Dokument fest, die die Dateianhangs-Anmerkung des Ziels enthält. Wurde* `pagenumber` *angegeben, wird diese Option ignoriert.*
name	*(Hypertext-String; erforderlich bei* `relation=child` *und wenn sich das Ziel in der* `attachments`*-Liste befindet; sonst darf der String nicht vorhanden sein; wird bei Angabe von* `annotation` *ignoriert) Name des Ziels in der* `attachments`*-Liste von PDF_begin/end_document( ).*
pagenumber	*(Integer; erforderlich sofern* `destname` *nicht übergeben wurde und nicht* `relation=child` *und das Ziel nicht mit einer Anmerkung für einen Dateianhang verknüpft ist; wird bei Angabe von* `destname` *ignoriert) Gibt die Nummer der Seite im aktuellen Dokument an, die die Dateianhangs-Anmerkung des Ziels enthält.*
relation	*(Schlüsselwort; erforderlich) Beziehung zwischen aktuellem Dokument und dem Ziel (das lediglich ein Zwischenziel sein kein). Unterstützte Schlüsselwörter:* **parent** *Das Ziel ist dem aktuellen Dokument übergeordnet.* **child** *Das Ziel ist dem aktuellen Dokument untergeordnet.*
targetpath	*(Optionsliste) Ziel-Optionsliste gemäß Tabelle 12.9 mit zusätzlichen Pfadinformationen zum Zieldokument. Wird diese Option nicht angegeben, wird das aktuelle Dokument zur Zieldatei, die das Ziel enthält.*

12.5 Benannte Ziele

C++ Java C# **void add_nameddest(String name, String optlist)**

Perl PHP **add_nameddest(string name, string optlist)**

C **void PDF_add_nameddest(PDF *p, const char *name, int len, const char *optlist)**

Erzeugt im aktuellen Dokument auf einer beliebigen Seite ein benanntes Ziel.

name (Hypertext-String) Name des Ziels für Verknüpfungen, Lesezeichen oder andere Auslöser. Namen für benannte Ziele müssen im Dokument eindeutig sein. Wird derselbe Name für ein Dokument mehrmals übergeben, so wird nur die letzte Definition verwendet und die übrigen werden ignoriert.

len (Nur C-Sprachbindung) Länge von *name* (in Bytes). Ist *len* = 0, muss ein null-terminierter String übergeben werden.

optlist Optionsliste zur Festlegung des benannten Ziels. Eine leere Liste entspricht *{type=fitwindow page=0}*. Folgende Optionen können verwendet werden:

- Allgemeine Optionen: *errorpolicy* (siehe Tabelle 2.1), *hypertextencoding* und *hypertextformat* (siehe Tabelle 2.3)
- Optionen für die Steuerung benannter Ziele gemäß Tabelle 12.10: *bottom, group, left, page, right, top, type, zoom*

Details Das Ziel muss in *optlist* festgelegt werden und kann sich auf einer beliebigen Seite im aktuellen Dokument befinden. Der übergebene *name* kann mit der Option *destname* in *PDF_create_action( )*, *PDF_create_annotation( )*, *PDF_create_bookmark( )* und *PDF_begin/end_document( )* verwendet werden. Auf diese Weise lassen sich die Definition und die Benutzung eines Ziels in zwei getrennte Schritte aufspalten.

Alternativ dazu können Definition und Benutzung des benannten Ziels auch kombiniert werden, sofern das Ziel zum Zeitpunkt seiner Benutzung bereits bekannt ist. Dazu verwenden Sie in obigen Funktionen die Option *destination*. *PDF_add_nameddest( )* ist in diesem Fall nicht erforderlich.

Gültigkeit beliebig außer *object*

Tabelle 12.10 Optionen zur Festlegung des Ziels in PDF_add_nameddest( ) sowie für die Option `destination` *in PDF_create_action( ), PDF_create_annotation( ), PDF_create_bookmark( ) und PDF_begin/end_document( ).*

Option	Beschreibung
bottom	*(Float; nur relevant bei* `type=fitrect`*) y-Koordinate der Seite, die am unteren Fensterrand platziert wird. Standardwert: 0*
group	*(String; erforderlich, wenn die Option* page *angegeben wurde und das Dokument Seitengruppen verwendet; sonst nicht zulässig) Name der Seitengruppe, zu der die Zielseite gehört.*
left	*(Float; nur relevant bei* `type=fixed`*,* `fitheight`*,* `fitrect` *oder* `fitvisibleheight`*) x-Koordinate der Seite, die am linken Fensterrand platziert wird. Standardwert: 0*
page	*(Integer) Seitennummer der Zielseite (erste Seite ist 1). Die Seite muss im Zieldokument vorhanden sein. Seite 0 bedeutet die aktuelle Seite im Gültigkeitsbereich* page *und Seite 1 im Gültigkeitsbereich* document*. Standardwert: 0*
right	*(Float; nur relevant bei* `type=fitrect`*) x-Koordinate der Seite, die am rechten Fensterrand platziert wird. Standardwert: 1000*

Tabelle 12.10 Optionen zur Festlegung des Ziels in PDF_add_nameddest( ) sowie für die Option `destination` *in PDF_create_action( ), PDF_create_annotation( ), PDF_create_bookmark( ) und PDF_begin/end_document( ).*

Option	Beschreibung
top	*(Float; nur relevant bei* `type=fixed`*,* `fitwidth`*,* `fitrect` *oder* `fitvisiblewidth`*) y-Koordinate der Seite, die am oberen Fensterrand platziert wird. Standardwert: 1000*
type	*(Schlüsselwort) Legt die Platzierung des Dokumentfensters auf der Zielseite fest. Unterstützte Schlüsselwörter (Standardwert:* `fitwindow`*):* **fitheight** *Passt die Seitenhöhe in das Fenster ein und platziert die Seite mit der x-Koordinate* `left` *am linken Fensterrand.* **fitrect** *Passt das durch* `left`*,* `bottom`*,* `right` *und* `top` *festgelegte Rechteck in das Fenster ein.* **fitvisible** *Passt den sichtbaren Seiteninhalt (die ArtBox) in das Fenster ein.* **fitvisibleheight** *Passt den sichtbaren Seiteninhalt in das Fenster ein und platziert die Seite mit der x-Koordinate* `left` *am linken Fensterrand.* **fitvisiblewidth** *Passt den sichtbaren Seiteninhalt in das Fenster ein und platziert die Seite mit der y-Koordinate* `top` *am oberen Fensterrand.* **fitwidth** *Passt die Seitenbreite in das Fenster ein und platziert die Seite mit der y-Koordinate* `top` *am oberen Fensterrand.* **fitwindow** *Passt die Seite vollständig in das Fenster ein.* **fixed** *Verwendet die durch die Optionen* `left`*,* `top` *und* `zoom` *festgelegte Ansicht. Fehlt eine der Optionen, wird ihr aktueller Wert verwendet.*
zoom	*(Float oder Prozentwert; nur relevant bei* `type=fixed`*) Zoomfaktor (1 bedeutet 100%), mit dem der Seiteninhalt angezeigt wird. Ist diese Option nicht vorhanden oder gleich 0, so wird der Zoomfaktor beibehalten, der bei der Aktivierung der Verknüpfung aktuell war.*

12.6 PDF-Pakete und Portfolios

Portfolio-Features sind mit folgenden Funktionen und Optionen implementiert:

- Portfolios können mit der Option *portfolio* von *PDF_end_document( )* erzeugt werden. Für weitere Informationen zu dieser Funktion siehe Abschnitt 3.1, »Dokumentfunktionen«, Seite 45; Die Option *portfolio* wird in Tabelle 12.13 beschrieben.
- Einem Portfolio können mit *PDF_add_portfolio_folder( )* und *PDF_add_portfolio_file( )* Dateien und Dateiverzeichnisse hinzugefügt werden. Diese Funktionen werden im Folgenden beschrieben.
- Aktionen für die Navigation innerhalb eines Portfolios können mit *PDF_create_action( )* und *type=GoToE* erzeugt werden (siehe Abschnitt 12.4, »Aktionen«, Seite 246).

C++ Java C# ***int add_portfolio_folder(int parent, String, foldername, String optlist)***

Perl PHP ***int add_portfolio_folder(int parent, string foldername, string optlist)***

C ***int PDF_add_portfolio_folder(PDF *p, int parent, const char *foldername, int len, const char *optlist)***

Fügt ein Verzeichnis zu einem neuen oder bestehenden Portfolio hinzu (erfordert PDF 1.7ext3).

parent Übergeordnetes Verzeichnis, das mit einem von einem früheren Aufruf von *PDF_add_portfolio_folder( )* zurückgegebenen Verzeichnis-Handle angegeben wird, oder -1 (in PHP: 0) für das Stammverzeichnis.

foldername (Hypertext-String aus 1-255 Zeichen; die Zeichen / \ : * " < > | dürfen nicht verwendet werden; das letzte Zeichen darf kein Punkt '.' sein) Name des Verzeichnisses. Zwei Verzeichnisse mit demselben *parent*-Verzeichnis dürfen nach der Normalisierung nicht denselben Namen haben. Der Name des Stammverzeichnisses wird bei Acrobat ignoriert.

len (Nur C-Sprachbindung) Länge von *foldername* (in Bytes). Ist *len* = 0, muss ein nullterminierter String übergeben werden.

optlist Optionsliste mit Portfolio-Eigenschaften. Folgende Optionen können verwendet werden:

- Allgemeine Optionen: *errorpolicy* (siehe Tabelle 2.1), *hypertextencoding* und *hypertextformat* (siehe Tabelle 2.3)
- Optionen für Verzeichniseigenschaften gemäß Tabelle 13.6: *description, thumbnail*
- Metadaten-Option gemäß Tabelle 12.11: *fieldlist*

Rückgabe Handle, das in *PDF_add_portfolio_folder( )* oder *PDF_add_portfolio_file( )* verwendet werden kann.

Details Die generierte Verzeichnisstruktur wird zur Erzeugung eines PDF-Portfolios für das aktuelle Dokument verwendet. Nach *PDF_end_document( )* wird die Verzeichnisstruktur gelöscht. Diese Funktion darf nicht verwendet werden, falls die Option *attachments* an *PDF_begin_document( )* übergeben wurde.

Gültigkeit beliebig außer *object*

Tabelle 12.11 Optionen für PDF_add_portfolio_folder( ) und PDF_add_portfolio_file( )

Option	Beschreibung
fieldlist	*(List mit Optionslisten) Legt Metadaten-Felder für die Datei oder das Verzeichnis fest. Jede Liste bezieht sich auf ein Feld in der Unteroption* `schema` *der Option* `portfolio` *von PDF_end_document( ). Für unterstützte Unteroptionen siehe Tabelle 12.12.*

C++ Java C# **int add_portfolio_file(int folder, String filename, String optlist)**

Perl PHP **int add_portfolio_file(int folder, string filename, string optlist)**

C **int PDF_add_portfolio_file(PDF *p, int folder, const char *filename, int len, const char *optlist)**

Fügt einem Portfolio-Verzeichnis oder PDF-Paket eine Datei hinzu (erfordert PDF 1.7).

folder Verzeichnis-Handle, das von einem früheren Aufruf von *PDF_add_portfolio_folder( )* zurückgegeben wurde, oder -1 (in PHP: 0) für das Stammverzeichnis. Alle Verzeichnisse außer dem Stammverzeichnis erfordern PDF 1.7ext3.

filename (Name-String; wird gemäß der globalen Option *filenamehandling* interpretiert, siehe Tabelle 2.3) Name einer lokalen oder virtuellen Datei, die dem ausgewählten Verzeichnis des PDF-Portfolios hinzugefügt wird. Mit der Option *createpvf* von *PDF_begin_document( )* können Sie Dokumente im Arbeitsspeicher erzeugen und in ein PDF-Portfolio einfügen, ohne temporäre Dateien auf der Festplatte zu erzeugen.

Beachten Sie, dass Acrobat anhand der Dateinamenserweiterung nach der passenden Anwendung für eine Datei sucht. Wenn ein Dateiname mit der passenden Erweiterung wegen äußerer Einschränkungen nicht verwendet werden kann, können Sie stattdessen eine PVF-Datei erzeugen (die beliebige Dateinamen unterstützt).

len (Nur C-Sprachbindung) Länge von *filename* (in Bytes). Ist *len* = 0, muss ein null-terminierter String übergeben werden.

optlist Optionsliste mit Dateieigenschaften:

- Allgemeine Optionen: *errorpolicy* (siehe Tabelle 2.1) und *hypertextformat* (siehe Tabelle 2.3)
- Optionen für Dateieigenschaften gemäß Tabelle 13.6: *description, filename, mimetype, name, password, relationship, thumbnail*
- Metadaten-Option gemäß Tabelle 12.11: *fieldlist*

Rückgabe Wurde die Datei erfolgreich hinzugefügt, wird der Wert 1 zurückgegeben, oder im Fehlerfall der Fehlercode -1 (in PHP: 0). Bei *errorpolicy=exception* löst diese Funktion im Fehlerfall eine Exception aus. PDF-Dokumente werden geöffnet, um das Änderungs- und Erstellungsdatum auszulesen. Das Dokument wird auch dann in das PDF-Portfolio übernommen, wenn das PDF-Dokument nicht geöffnet werden kann (z.B. weil kein Kennwort übergeben wurde).

Details Die ausgewählte Datei wird zum angegebenen Verzeichnis eines PDF-1.7ext3-Portfolios oder eines PDF 1.7-Pakets hinzugefügt. Ist PDI verfügbar, werden die PDF-Dokumente nach Möglichkeit geöffnet und ihr Erstellungs- und Änderungsdatum wird in das Portfolio geschrieben. Diese Funktion darf nicht verwendet werden, wenn die Option *attachments* an *PDF_begin_document( )* übergeben wurde.

PDF/A PDF/A-1: diese Funktion ist nicht zulässig.
PDF/A-2: *filename* muss sich auf ein PDF/A-1- oder PDF/A-2-Dokument beziehen. Einige Optionen sind eingeschränkt, siehe Tabelle 13.6.
PDF/A-3: es können beliebige Dateitypen hinzugefügt werden. Die Option *relationship* ist erforderlich. Zu einem Paket hinzugefügte Dateien sind implizit mit dem gesamten Dokument verknüpft.

Gültigkeit beliebig außer *object*

Tabelle 12.12 Unteroptionen der Option `fieldlist` *von PDF_add_portfolio_folder( ) und PDF_add_portfolio_file( )*

Option	*Beschreibung*
key	*(String; erforderlich) Name des Felds, der sich auf* `key` *in der Unteroption* `schema` *der Optionsliste* `portfolio` *von PDF_end_document( ) beziehen muss. Der Name muss eindeutig sein.*
prefix	*(Hypertext-String) Präfix-String, der dem Feldwert bei der Anzeige vorangestellt wird. Acrobat verwendet diesen Eintrag nur bei* `type=text`*. Standardwert: nicht vorhanden*
type	*(Schlüsselwort) Datentyp des Felds. Unterstützte Schlüsselwörter (Standardwert:* `text`*):* ***text*** *Textfeld: der Wert des Felds wird als Hypertext-String gespeichert.* ***date*** *Datumsfeld: der Wert des Felds wird als PDF-Datums-String gespeichert.* ***number*** *Nummernfeld: der Wert des Felds wird als PDF-Nummer gespeichert.*
value	*(Hypertext-String; erforderlich) Wert eines Felds in der Unteroption* `schema` *der Optionsliste* `portfolio` *von PDF_end_document( ). Der Datentyp muss in der Option* `type` *angegeben werden und muss der zugehörigen Unteroption* `type` *der Unteroption* `schema` *der Option* `portfolio` *entsprechen.*

Tabelle 12.13 Unteroptionen der Option `portfolio` *von PDF_end_document( )*

Option	*Beschreibung*
coversheet	*(Hypertext-String) Name des Deckblatts des Portfolios, die an der Benutzeroberfläche angezeigt wird. Standardwert: das Dokument, welches das Portfolio enthält*
coversheet-folder	*(Verzeichnis-Handle) Name des Verzeichnisses im Portfolio, das die in der Option* `coversheet` *angegebene Datei enthält. Kommt die Datei mit dem bei* `coversheet` *angegebenen Namen in mehreren Portfolio-Verzeichnissen vor und wurde kein Verzeichnis* `coversheetfolder` *angegeben, wird die zuerst auftretende Datei verwendet. Standardwert: nicht vorhanden*
initialview	*(Schlüsselwort) Ansicht beim Öffnen des Portfolios. Unterstützte Schlüsselwörter (Standardwert:* `detail`*):* ***custom*** *(PDF 1.7ext3; erfordert die Option* `navigator`*) Das Portfolio wird als benutzerdefinierte Flash-Navigation dargestellt.* ***detail*** *Das Portfolio wird in Detailansicht dargestellt, die Informationen in der Option* `schema` *werden in mehrspaltigem Layout dargestellt. Dieser Modus bietet die umfassendste Information (Acrobat: »Ansicht oben«).* ***hidden*** *Das Portfolio ist anfänglich versteckt, Benutzer können sich aber mit einer expliziten Aktion eine Dateiliste anzeigen lassen (Acrobat: »Ansicht minimieren«).* ***tile*** *Die einzelnen Dateien des Portfolios werden als Kacheln dargestellt, wobei für jede Datei ein kleines Icon und ein Teil der Informationen aus der Option* `schema` *angezeigt werden. Dieser Modus bietet eine schnelle Übersicht über die Dateianhänge (Acrobat: »Ansicht links«).*

Tabelle 12.13 Unteroptionen der Option portfolio *von PDF_end_document( )*

Option	Beschreibung
navigator	*(Optionsliste; PDF 1.7ext3; erforderlich bei* initialview=custom*) Bettet eine benutzerdefinierte Flash-Navigation in das Portfolio ein. Um diese beim Öffnen des Dokuments zu aktivieren, verwenden Sie* view=custom*, denn sonst kann der Navigator nur beim Bearbeiten des Portfolios verwendet werden. Für unterstützte Unteroptionen siehe Tabelle 12.14.* *Die Werte von* category*,* description*,* icon *und* name *werden in Acrobat benutzt, um den Navigator in der Liste der verfügbaren Portfolio-Layouts anzuzeigen.*
schema	*(Liste mit Optionslisten) Metadaten-Schema für das Portfolio: jede Optionsliste definiert ein Feld mit einem eindeutigen Namen, der einem Schlüssel in der* fieldlist *einer Datei oder eines Verzeichnisses entspricht, oder aber einem Standardfeld. Diese Felder legen fest, wie ein Portfolio in Acrobat angezeigt wird (Standardwert: Acrobat zeigt Dateinamen und -größe, Änderungsdatum und, sofern vorhanden, eine Beschreibung an):* **editable** *(Boolean) Legt fest, ob der Wert des Feldes in Acrobat editierbar ist. Standardwert:* false **key** *(String; erforderlich) Interner Feldname, muss eindeutig sein.* *Mit folgenden Namen (für benutzerdefinierte Felder unzulässig) können neue Label an vordefinierte Felder zugewiesen werden:* _creationdate*,* _description*,* _filename*,* _moddate*,* _size*.* **label** *(Hypertext-String; erforderlich) Dem Benutzer angezeigtes Textfeld-Label.* **order** *(Integer) Relative Reihenfolge der Felder in der Benutzeroberfläche (1,2,3,...)* **type** *(Schlüsselwort) Datentyp des Felds. Folgende Datentypen können für benutzerdefinierte Felder in der Option* fieldlist *verwendet werden (Standardwert:* text*):* **text** *Hypertext-String* **date** *PDF-Datums-String* **number** *Nummer* **visible** *(Boolean) Sichtbarkeit des Felds beim Öffnen der Benutzeroberfläche. Standardwert:* true*; sind jedoch benutzerdefinierte Felder vorhanden, versteckt Acrobat vordefinierte Felder, sofern sie nicht explizit als sichtbar gekennzeichnet sind.*
sort	*(Liste mit Optionslisten, wobei jede Liste einen String und ein optionales Schlüsselwort enthält) Legt die Reihenfolge fest, in welcher die in der Option* schema *angegebenen Felder in der Benutzeroberfläche angezeigt werden. Jede Teilliste enthält den Feldnamen (erforderlich) und ein Schlüsselwort (optional). Unterstützte Schlüsselwörter (Standardwert:* ascending*):* **ascending** *Feldwerte werden in aufsteigender Reihenfolge sortiert* **descending** *Feldwerte werden in absteigender Reihenfolge sortiert* *Mit dieser Liste werden in Acrobat die Felder im Portfolio sortiert. In der Liste sind zusätzliche Felder zur Sortierung vorgesehen, wobei die zusätzlichen Felder benutzt werden, um gleich sortierte Felder anzuordnen: wenn mehrere Felder in der Option* schema *den gleichen Wert für das erste Feld in der Liste haben, werden die Werte für aufeinanderfolgende Felder in der Liste für die Sortierung verwendet, bis eine eindeutige Reihenfolge feststeht oder bis keine Feldnamen mehr vorhanden sind. Standardwert: keine Sortierung*
split	*(Optionsliste; PDF 1.7ext3) Ausrichtung und Position des Teilungsbalkens eines Fensters. Der Standardwert ist abhängig von der Option* initialview*: Der Wert* detail *(oder kein Wert) bedeutet horizontale Teilung und* tile *bedeutet vertikale Teilung. Bei* initalview=hidden *wird das Fenster nicht geteilt. Unterstützte Unteroptionen:* **direction** *(Schlüsselwort) Ausrichtung des Teilungsbalkens. Unterstützte Schlüsselwörter:* **horizontal** *Das Fenster wird horizontal geteilt.* **vertical** *Das Fenster wird vertikal geteilt.* **none** *Das Fenster wird nicht geteilt. Das gesamte Fenster wird in der Dateinavigation angezeigt.* **position** *(Prozentwert) Ausgangsposition des Teilungsbalkens, angegeben als Prozentwert des verfügbaren Fensterbereichs. Erlaubte Werte liegen im Bereich 0 bis 100. Bei* direction=none *wird dieser Eintrag ignoriert. Standardwert: abhängig vom Betrachter*

Tabelle 12.14 Unteroptionen der Unteroption navigator *der Option* portfolio *von PDF_add_portfolio_folder( ) und PDF_add_portfolio_file( )*

Option	Beschreibung
apiversion	*(String; erforderlich) Version des Navigator-API, die für die Navigator-SWF-Datei erforderlich ist, angegeben als String im Format* m[.n[.p[.q]]]*, wobei* m, n, p *und* q *nicht negative ganze Zahlen sind. Werden für* n, p *und* q *keine Werte angegeben, wird der Standardwert 0 gesetzt. Folgende Einträge werden empfohlen:* *für Acrobat 9:* 9.0.0.0 *für Acrobat X:* 9.5.0.0
assets	*(Liste mit Optionslisten, erforderlich) Assets für die Implementierung des Navigators, z.B. eine Flash-Datei, ein Icon und andere Ressourcen wie Bilder oder XML-Dateien. Unterstützte Unteroptionen:* **asset** *(Asset-Handle; erforderlich) Handle für ein mit PDF_load_asset( ) geladenes Asset.* **name** *(Hypertext-String mit 1-255 Zeichen; die Zeichen : * " < > | sind unzulässig; das letzte Zeichen darf kein Punkt '.' sein; erforderlich) Name eines Assets zur Identifizierung im Flash-Code.*
category	*(Hypertext-String) Kategorie, in der der Navigator angezeigt wird*
description	*(Hypertext-String) Beschreibung des Navigators*
flash	*(Hypertext-String; erforderlich) Name eines Assets in der Optionsliste* assets *mit* type=Flash*, der den SWF-Code enthält, mit dem das Portfolio-Layout implementiert wird.*
icon	*(Hypertext-String) Name eines Assets in der Optionsliste* assets *mit* type=JPEG *oder* PNG*, das ein Icon für den Navigator enthält. Es wird jedoch der Einsatz von Bildern im PNG-Format empfohlen, da Bilder im JPEG-Format in Acrobat nicht zuverlässig funktionieren. Die besten Ergebnisse in Acrobat erzielt man mit einer Größe von 42x42 Pixeln.*
id	*(String; erforderlich bei* version*) String, der eine eindeutige ID für den Navigator in Form eines URI darstellt. Soll ein Versionierungsschema implementiert werden, muss dieselbe* id *über alle Versionen des Navigators hinweg verwendet werden. Wird keine* id *angegeben, generiert PDFlib einen URN auf Basis einer maschinell erzeugten GUID.*
loadtype	*(Schlüsselwort) Methode zum Laden der Navigator-SWF. Unterstützte Schlüsselwörter (Standardwert ist das Schlüsselwort* default*):* **module** *Navigator-SWF wird als Adobe-Flex-2-Module geladen.* **default** *Navigator-SWF wird als normale SWF-Datei geladen.*
locale	*(String) String mit einem Sprachcode gemäß Unicode Technical Standard #35. Beispiele:* en_GB, de_DE, zh_Hans
name	*(Hypertext-String; erforderlich) Name des Portfolios*
strings	*(Liste mit Hypertext-Stringpaaren) Lokalisierte Strings für den Navigator. Jedes Paar besteht aus einem Identifikator für den lokalisierten String und dem lokalisierten String selbst.*
version	*(String; erfordert die Option* id*) Version des Navigators, angegeben als String im Format* m[.n[.p[.q]]]*, wobei* m, n, p *und* q *nicht negative ganze Zahlen sind. Werden für* n, p *und* q *keine Werte angegeben, wird der Standardwert 0 gesetzt.*

12.7 Features für Geodaten

Geodaten-Features sind mit folgenden Funktionen und Optionen implementiert:

- Mit der Option *viewports* von *PDF_begin/end_page_ext( )* können einer Seite georeferenzierte Bereiche zugewiesen werden.
- Mit der Option *georeference* von *PDF_load_image( )* kann einem Bild ein erdbasiertes Koordinatensystem zugewiesen werden.

Optionen für Geodaten-Features werden in den folgenden Tabellen beschrieben.

Tabelle 12.15 Unteroptionen für die Option `viewports` *von PDF_begin/end_page_ext( )*

Option	Beschreibung
boundingbox	*(Rechteck; erforderlich) Rechteck in Standardkoordinaten, das die Position des Viewports auf der Seite angibt.*
georeference	*(Optionsliste; erforderlich) Beschreibung eines mit dem Viewport verbundenen Weltkoordinatensystems für geodätische Messungen fest; für unterstützte Optionen siehe Tabelle 12.16.*
hypertextencoding	*(Schlüsselwort) Encoding für die Option* `name`*. Ein leerer String ist äquivalent zu* `unicode`*. Standardwert: Wert der globalen Option* `hypertextencoding`
name	*(Hypertext-String) Beschreibender Titel des Viewports (Kartenname). Acrobat zeigt den Viewport-Namen in der Benutzeroberfläche jedoch nicht an.*

Tabelle 12.16 Unteroptionen für die Option `georeference` *von PDF_load_image( ) und die Unteroption* `georeference` *der Option* `viewports` *von PDF_begin/end_page_ext( )*

Option	Beschreibung
angularunit	*(Schlüsselwort) Gewünschtes Winkelmaß für die Anzeige (Standardwert:* `deg`*):* ***degree*** *Grad* ***grad*** *Gon (1/400 des gesamten Kreises bzw. 0,9 Grad)*
areaunit	*(Schlüsselwort) Gewünschtes Flächenmaß für die Anzeige (Standardwert:* `sqm`*):* ***sqm*** *Quadratmeter* ***ha*** *Hektar (10.000 Quadratmeter)* ***sqkm*** *Quadratkilometer* ***sqft*** *Quadratfuß* ***a*** *Acre* ***sqmi*** *Quadratmeilen* *Die gewünschte Einheit wird nur verwendet, wenn folgende Einstellung in Acrobat deaktiviert wird: »Voreinstellungen, Messen (Geo), Standard-Flächeneinheit verwenden«.*
bounds	*(Polylinie mit zwei oder mehr Punkten) Flächengrenze, für die die Geodaten-Transformation gilt (bei Karten nennt man diese begrenzende Polylinie Kartenrahmen oder Neatline). Die Punkte werden relativ zur* `boundingbox` *eines Seiten-Viewports oder zum Umfang eines Bildes ausgedrückt. Standardwert:* `{0 0 0 1 1 1 1 0}`*, d.h., der gesamte Viewport oder die Bildfläche wird für die Karte verwendet.*
displaysystem	*(Optionsliste) Koordinatensystem gemäß Tabelle 12.17 für die Anzeige von Positionswerten, z.B. Längen- und Breitengrad. Mit diesem Eintrag können die Koordinaten in einem anderen System dargestellt werden als in der Option* `coords` *für die Darstellung der Karte angegeben.*

Tabelle 12.16 Unteroptionen für die Option georeference *von* PDF_load_image() *und die Unteroption* georeference *der Option* viewports *von* PDF_begin/end_page_ext()

Option	Beschreibung
linearunit	*(Schlüsselwort) Gewünschtes Längenmaß für die Anzeige (Standardwert:* m*):* ***m*** *Meter* ***km*** *Kilometer* ***ft*** *internationaler Fuß* ***usft*** *US survey foot* ***mi*** *internationale Meile* ***nm*** *nautische Meile* *Die gewünschte Einheit wird nur verwendet, wenn folgende Einstellung in Acrobat deaktiviert wird: »Voreinstellungen, Messen (Geo), Standard-Entfernungseinheit verwenden«.*
mappoints	*(Liste mit zwei oder mehr Float-Paaren; erforderlich) Liste mit Zahlen, wobei jedes Paar einen Punkt in einem 2D-Einheitsquadrat bestimmt. Das Einheitsquadrat wird an die rechteckige Begrenzung des Seiten-Viewports oder des Bildes angepasst, die die Optionsliste* georeference *enthalten. Die* mappoints*-Liste muss die gleiche Anzahl an Punkten haben wie die* worldpoints*-Liste; jeder Punkt stellt eine Kartenposition im Einheitsquadrat dar, die der räumlichen Position in der* worldpoints*-Liste entspricht.*
worldpoints	*(Liste mit zwei oder mehr Float-Paaren; erforderlich) Liste mit Koordinatenpaaren, wobei jedes Paar die Weltkoordinaten des entsprechenden Punkts in der Option* mappoints *angibt. Die Anzahl der Paare muss mit der Anzahl der Paare in der Option* mappoints *übereinstimmen. Die Koordinatenwerte basieren auf dem in der Option* worldsystem *angegebenen Koordinatensystem: bei* type=geographic *müssen die Werte für Längen- und Breitengrad in Grad angegeben werden. Bei* type=projected *müssen projizierte x- und y-Werte angegeben werden.*
worldsystem	*(Optionsliste; erforderlich) Weltkoordinatensystem (zur Interpretation von* worldpoints*) gemäß Tabelle 12.17.*

Tabelle 12.17 Unteroptionen für die Unteroptionen mapsystem *und* displaysystem *der Option* georeference *von* PDF_load_image() *und der Unteroption* georeference *der Option* viewports *von* PDF_begin/end_page_ext()

Option	Beschreibung
epsg	*(Integer; es muss genau eine der Optionen* epsg *oder* wkt *übergeben werden) Koordinatensystem als EPSG-Referenzcode. Beachten Sie, dass Acrobat 9 keine EPSG-Codes für* type=geographic *unterstützt; verwenden Sie in diesem Fall* wkt*.*
type	*(Schlüsselwort; erforderlich) Typ des Koordinatensystems:* ***geographic*** *geografisches Koordinatensystem (unterstützt nur* wkt*)* ***projected*** *projiziertes Koordinatensystem (unterstützt* wkt *und* epsg*)*
wkt	*(String mit bis zu 1024 ASCII-Zeichen; es muss genau eine der Optionen* epsg *oder* wkt *übergeben werden) Koordinatensystem als String im Format »Well Known Text« (WKT). WKT wird für benutzerdefinierte Koordinatensysteme ohne EPSG-Code empfohlen und scheint in Acrobat 9 bei* type=geographic *erforderlich zu sein.*

13 Multimedia-Funktionen

13.1 3D-Modelle

Cookbook *Ein vollständiges Codebeispiel hierzu finden Sie im Cookbook-Topic* `multimedia/starter_3d`.

3D-Features sind mit folgenden Funktionen und Optionen implementiert:

- 3D-Daten können mit *PDF_load_3ddata( )* geladen werden. Diese Funktion wird im Folgenden beschrieben.
- 3D-Views können mit *PDF_create_3dview( )* erzeugt werden. Diese Funktion wird im Folgenden beschrieben.
- 3D-Anmerkungen können mit *PDF_create_annotation( )* und *type=3D* erzeugt werden. Diese Funktion wird in Abschnitt 12.2, »Anmerkungen«, Seite 229 beschrieben. Die Optionen für diese Funktion zur Steuerung von 3D-Anmerkungen finden Sie in Tabelle 13.4 unten.
- Aktionen für die Steuerung von 3D-Anmerkungen können mit *PDF_create_action( )* und *type=3Dview* erzeugt werden (siehe Abschnitt 12.4, »Aktionen«, Seite 246).

C++ Java C# ***int load_3ddata(String filename, String optlist)***

Perl PHP ***int load_3ddata(string filename, string optlist)***

C ***int PDF_load_3ddata(PDF *p, const char *filename, int len, const char *optlist)***

Öffnet eine auf der Festplatte liegende oder virtuelle 3D-Modelldatei (erfordert PDF 1.6).

filename (Name-String; wird gemäß der globalen Option *filenamehandling* interpretiert, siehe Tabelle 2.3) Name einer auf der Festplatte liegenden oder virtuellen Datei mit einem 3D-Modell.

len (Nur C-Sprachbindung) Länge von *filename* (in Bytes). Ist *len* = 0, muss ein null-terminierter String übergeben werden.

optlist Optionsliste mit Eigenschaften des 3D-Modells:

- Allgemeine Optionen: *errorpolicy* (siehe Tabelle 2.1), *hypertextencoding* (siehe Tabelle 2.3)
- Optionen mit Eigenschaften für 3D-Modelle gemäß Tabelle 13.1: *defaultview, script, type, views*

Rückgabe 3D-Handle, das zur Erstellung von 3D-Anmerkungen mit *PDF_create_annotation( )* verwendet werden kann. Es kann bis zum Ende des umgebenden Gültigkeitsbereichs *document* verwendet werden. Bei *errorpolicy=return* muss der Aufrufer den Rückgabewert auf -1 (in PHP: 0) überprüfen, da dies auf einen Fehler hinweist.

Details Die Datei muss 3D-Daten im Format PRC oder U3D enthalten.

Gültigkeit beliebig außer *object*. Das zurückgegebene Handle kann bis zum nächsten Aufruf von *PDF_end_document( )* verwendet werden.

Tabelle 13.1 Optionen für PDF_load_3ddata( )

Option	Beschreibung
defaultview	*(Schlüsselwort oder 3D-View-Handle) Legt die Anfangsdarstellung der 3D-Anmerkung fest; eines der Schlüsselwörter* `first` *oder* `last` *(bezogen auf die entsprechenden Einträge in der Option* `views`*) oder ein 3D-View-Handle, das mit PDF_create_3dview( ) erstellt wurde. Standardwert:* `first`
script	*(Hypertext-String) String mit JavaScript-Code, der bei der Instantiierung des 3D-Modells ausgeführt wird. Standardwert: kein Skript*
type	*(Schlüsselwort) Legt den 3D-Datentyp fest (Standardwert:* `U3D`*):* **PRC** *(PDF 1.7ext3) Die Daten werden im Format Product Representation Compact (PRC) (ISO 14739-1) übergeben* **U3D** *Die Daten werden im Format Universal 3D File (U3D) in den folgenden Varianten (siehe* `www.ecma-international.org` *) übergeben:* *PDF 1.6, erfordert Acrobat 7.0.7 oder höher: ECMA-363, Universal 3D File Format (U3D), 1. Ausgabe;* *PDF 1.6, erfordert Acrobat 8.1 oder höher: ECMA-363, Universal 3D File Format (U3D), 3. Ausgabe;* *Beachten Sie, dass es bei Acrobat 9.3.x und 9.4.x (aber nicht bei Acrobat 8 und X) Probleme mit der Darstellung von U3D-Modellen gibt und folgende Fehlermeldung ausgegeben wird: »A 3D data parsing error has occurred«.*
views	*(Liste aus 3D-View-Handles) Liste aus vordefinierten Views für das 3D-Modell. Jedes Listenelement entspricht einem 3D-View-Handle, das mit PDF_create_3dview( ) erstellt wurde. Die bei der Erstellung von Views mit PDF_create_3dview( ) verwendete Option* `type` *muss mit der Option* `type` *in PDF_load_3ddata( ) übereinstimmen. Standardwert: leere Liste*

C++ Java C# ***int create_3dview(String username, String optlist)***

Perl PHP ***int create_3dview(string username, string optlist)***

C ***int PDF_create_3dview(PDF *p, const char *username, int len, const char *optlist)***

Erzeugt einen 3D-View (erfordert PDF 1.6).

username (Hypertext-String) Name der Benutzerschnittstelle für den 3D-View.

len (Nur C-Sprachbindung) Länge von *username* (in Bytes). Ist *len = 0*, muss ein null-terminierter String übergeben werden.

optlist Optionsliste mit Eigenschaften des 3D-Views:

- Allgemeine Optionen: *errorpolicy* (siehe Tabelle 2.1), *hypertextencoding* (siehe Tabelle 2.3)
- Optionen für Eigenschaften des 3D-Views gemäß Tabelle 13.2: *background, camera2world, cameradistance, lighting, name, rendermode, type, U3Dpath*

Rückgabe 3D-View-Handle, das bis zum Ende des umgebenden Gültigkeitsbereichs *document* verwendet werden kann. Bei *errorpolicy=return* muss der Aufrufer den Rückgabewert auf -1 (in PHP: 0) überprüfen, da dies auf einen Fehler hinweist.

Details Das 3D-View-Handle kann 3D-Modellen mit der Option *views* in *PDF_load_3ddata( )* zugeordnet. Es kann auch verwendet werden, um 3D-Anmerkungen mit *PDF_create_annotation( )* oder 3D-bezogene Aktionen mit *PDF_create_action( )* zu erstellen.

Gültigkeit beliebig außer *object*. Das zurückgegebene Handle kann bis zum nächsten Aufruf von *PDF_end_document( )* verwendet werden.

Tabelle 13.2 Optionen für PDF_create_3dview( )

Option	Beschreibung
background	*(Optionsliste) Legt den Hintergrund für das 3D-Modell fest:* ***fillcolor*** *(Farbe) Hintergrundfarbe, definiert im Farbraum RGB. Standardwert: weiß* ***entire*** *(Boolean) Bei* `true` *bezieht sich der Hintergrund auf die gesamte Anmerkung; anderenfalls bezieht er sich nur auf das Rechteck, das mit der Option* `3Dbox` *der Anmerkung festgelegt wurde. Standardwert:* `false`
camera2world	*(Liste aus 12 Floats) 3D-Transformationsmatrix, die die Position und Orientierung der Kamera in Weltkoordinaten festlegt (siehe Beschreibung unten). Standardwert: der im 3D-Modell definierte erste View*
camera-distance	*(Float; darf nicht negativ sein; wird nur berücksichtigt, wenn* `camera2world` *festgelegt ist) Abstand zwischen der Kamera und dem Zentrum des Orbits. Für weitere Informationen siehe die Beschreibung des* `C0`*-Schlüssels im Abschnitt 13.6.4 »3D Views« des ISO-Standards 32000-1. Standardwert: ist in den 3D-Daten definiert*
lighting	*(Optionsliste; PDF 1.7) Legt das Beleuchtungsschema für das 3D-Modell fest. Folgende Option wird unterstützt:* ***type*** *(Schlüsselwort) Legt das Beleuchtungsschema fest. Unterstützte Schlüsselwörter (Standardwert:* `Artwork`*):* ***Artwork*** *Das Licht wird im 3D-Modell angegeben.* ***None*** *kein Licht; im 3D-Modell angegebenes Licht wird ignoriert.* ***White*** *Drei hellgraue gerichtete Lichter, ohne Umgebungslicht* ***Day*** *Drei hellgraue gerichtete Lichter, ohne Umgebungslicht* ***Night*** *Ein gelbes, ein aquafarbenes und ein blaues gerichtetes Licht, ohne Umgebungslicht* ***Hard*** *Drei graue gerichtete Lichter, mittleres Umgebungslicht* ***Primary*** *Ein rotes, ein grünes und ein blaues gerichtetes Licht, ohne ambiente Komponente (Umgebungslicht)* ***Blue*** *Drei blaue gerichtete Lichter, ohne Umgebungslicht* ***Red*** *Drei rote gerichtete Lichter, ohne Umgebungslicht* ***Cube*** *Sechs graue gerichtete Lichter ausgerichtet an den Hauptachsen, ohne Umgebungslicht* ***CAD*** *Drei graue gerichtete Lichter und ein an der Kamera angebrachtes Licht, ohne Umgebungslicht* ***Headlamp*** *Einzelnes, an der Kamera angebrachtes, gerichtetes Licht, niedriges Umgebungslicht*
name	*(Hypertext-String) Name des 3D-View, der in GoTo-Aktionen verwendet werden kann. Dies ist ein optionaler interner Name, der getrennt vom erforderlichen Parameter* `username` *behandelt wird.*
rendermode	*(Optionsliste; PDF 1.7) Legt den Rendering-Modus für die Anzeige des 3D-Modells fest. Unterstützte Unteroptionen sind inTabelle 13.3 aufgeführt.*
type	*(Schlüsselwort; erforderlich, falls der View in PDF_load_3ddata( ) mit* `type=PRC` *verwendet wird) Legt den 3D-Datentyp fest (Standardwert:* `U3D`*):* ***PRC*** *Der View wird in PDF_load_3ddata( ) mit* `type=PRC` *verwendet.* ***U3D*** *Der View wird in PDF_load_3ddata( ) mit* `type=U3D` *verwendet.*
U3Dpath	*(Hypertext-String; wird ignoriert, wenn* `camera2world` *angegeben ist; nur bei* `type=U3D`*) Name, mit dem ein View-Knoten innerhalb eines 3D-Modells angesprochen wird.*

Kameraposition. Die Position der Kamera kann mit der Option *camera2world* angegeben werden. Alternativ dazu kann JavaScript-Code verwendet werden, um die Kamera zum Modell zu positionieren und auszurichten. Im PDFlib Cookbook finden Sie hierzu den entsprechenden Beispielcode.

Folgende Werte für gebräuchliche Kamerapositionen können an die Option *camera2world* übergeben werden. *x*, *y* und *z* sind geeignete Werte, die die Position der Kamera beschreiben. Diese Werte sollten die unten genannten Bedingungen erfüllen:
Ansicht von vorne:

```
{-1 0 0 0 0 1 0 1 0 x y z}  x klein, y groß und negativ, z klein
```

Ansicht von links:

```
{ 0 1 0 0 0 1 1 0 0 x 0 z}  x groß und negativ, z klein
```

Ansicht von oben:

```
{-1 0 0 0 1 0 0 0 -1 x 0 z}  x klein, z groß und positiv
```

Ansicht von hinten:

```
{ 1 0 0 0 0 1 0 -1 0 x y z}  x klein, y groß und positiv, z klein
```

Ansicht von unten:

```
{-1 0 0 0 -1 0 0 0 1 x 0 z}  x klein, z groß und negativ
```

Ansicht von rechts:

```
{ 0 -1 0 0 0 1 -1 0 0 x 0 z}  x groß und positiv, z klein
```

Isometrische Ansicht, das heißt, die Richtung der Projektion schneidet alle drei Achsen im gleichen Winkel. Es gibt genau acht solcher Ansichten, eine in jedem Oktanten:

```
{0.707107 -0.707107 0 -0.5 -0.5 0.707107 -0.5 -0.5 -0.707107 x y z}
x, y, z groß und positiv
```

Die Werte *x*, *y* und *z* sollten abhängig von der Position und Größe des Modells gewählt werden. »Groß« bedeutet, die Werte sollten deutlich größer als die Größe des Modells sein, um genügend Abstand zwischen Kamera und Modell zu lassen. Ist der Wert zu groß, erscheint das Modell sehr klein und wird beim Drehen der Ansicht schnell aus dem Blickfeld verschwinden. Ist der Wert zu klein, passt das Modell eventuell nicht vollständig in die Ansicht hinein. »Klein« bedeutet, dass der absolute Wert im Vergleich zum großen Wert klein sein sollte und die Größe des Modells nicht nennenswert überschreiten sollte.

Tabelle 13.3 Unteroptionen für die Option `rendermode` *von PDF_create_3dview()*

Option	Beschreibung
crease	*(Float im Bereich 0...180) Crease-Wert*
facecolor	*(RGB-Farbe oder Schlüsselwort; nur für* `type=Illustration`*) Flächenfarbe; diese Farbe wird von verschiedenen Rendering-Modi verwendet. Das Schlüsselwort* `backgroundcolor` *bezieht sich auf die aktuelle Hintergrundfarbe. Standardwert:* `backgroundcolor`
opacity	*(Float im Bereich 0...1) Opazität für einige Rendering-Modi. Standardwert: 0.5*

Tabelle 13.3 Unteroptionen für die Option `rendermode` *von PDF_create_3dview( )*

Option	Beschreibung
rendercolor	*(RGB-Farbe) Hilfsfarbe. Diese Farbe wird von verschiedenen Rendering-Modi verwendet. Standardwert: schwarz*
type	*(Schlüsselwort; PDF 1.7) Legt den Rendering-Modus für die Darstellung des 3D-Modells fest (Standardwert:* `Artwork`*):* ***Artwork*** *Der Rendering-Modus wird im 3D-Modell angegeben; alle anderen Unteroptionen der Option* `rendermode` *werden ignoriert.* ***Solid*** *Zeigt beleuchtete geometrischen Formen mit Oberflächenstruktur an.* ***SolidWireframe*** *Zeigt beleuchtete geometrischen Formen (Dreiecke) mit Oberflächenstruktur und mit einfarbigen, darübergelegten Kanten an.* ***Transparent*** *Zeigt beleuchtete geometrischen Formen (Dreiecke) mit Oberflächenstruktur und mit einer zusätzlichen, transparenten Ebene an.* ***TransparentWireframe*** *Zeigt beleuchtete geometrischen Formen (Dreiecke) mit Oberflächenstruktur und mit einer zusätzlichen, transparenten Ebene an.* ***BoundingBox*** *Zeigt beleuchtete geometrischen Formen (Dreiecke) mit Oberflächenstruktur und mit einer zusätzlichen, transparenten Ebene und einfarbigen, undurchsichtigen, darübergelegten Kanten an.* ***TransparentBoundingBox*** *Zeigt die Flächen von Bounding-Boxen für jeden Knoten an, ausgerichtet an den Achsen des lokalen Koordinatenraums für den Knoten, mit einer zusätzlichen transparenten Ebene.* ***TransparentBoundingBoxOutline*** *Zeigt die Kanten und Flächen von Bounding-Boxen für jeden Knoten an, ausgerichtet an den Achsen des lokalen Koordinatenraums für diesen Knoten, mit einer zusätzlichen transparenten Ebene.* ***Wireframe*** *Zeigt die Kanten und Flächen von Bounding-Boxen für jeden Knoten an, ausgerichtet an den Achsen des lokalen Koordinatenraums für diesen Knoten, mit einer zusätzlichen transparenten Ebene.* ***ShadedWireframe*** *Zeigt nur die Kanten an, fügt ihre Farbe zwischen ihren beiden Eckpunkten ein und wendet Beleuchtung an.* ***HiddenWireframe*** *Zeigt die Kanten einfarbig an, entfernt die Rückansicht und verdeckte Kanten.* ***Vertices*** *Zeigt nur einfarbige Eckpunkte an.* ***ShadedVertices*** *Zeigt nur Eckpunkte in ihrer Farbe an und wendet Beleuchtung an.* ***Illustration*** *Zeigt Silhouetten-Kanten mit Oberflächen an, entfernt verdeckte Linien.* ***SolidOutline****i* *Zeigt Silhouetten-Kanten mit beleuchteten Oberflächen und Oberflächenstruktur an, entfernt verdeckte Linien.* ***ShadedIllustration*** *Zeigt Silhouetten-Kanten mit beleuchteten Oberflächen und Oberflächenstruktur an und einen zusätzlichen emissiven Term zum Entfernen von schlecht ausgeleuchteten Bereichen des Modells.*

Tabelle 13.4 3D-Optionen für PDF_create_annotation( ) mit `type=3D`

Option	Beschreibung
3Dactivate	*(Optionsliste; nur für* `type=3D`*) Legt fest, wann die 3D-Anmerkung aktiviert werden soll, sowie ihren Zustand bei der Aktivierung/Deaktivierung. Unterstützte Optionen:* **enable** *(Schlüsselwort) Bestimmt, wann die Animation aktiviert wird (Standardwert:* `click`*):* **open** *Beim Öffnen der Seite aktivieren* **visible** *Aktivieren, wenn die Seite sichtbar wird* **click** *Anmerkung muss durch ein Skript oder eine Benutzeraktion explizit aktiviert werden.* **enablestate** *(Schlüsselwort) Anfangszustand der Animation (Standardwert:* `play`*):* **pause** *Das 3D-Modell wird instantiiert, aber die Skriptanimationen werden deaktiviert.* **play** *Das 3D-Modell wird instantiiert; Skriptanimationen werden aktiviert, falls vorhanden.* **disable** *(Schlüsselwort) Legt fest, wann die Animation deaktiviert wird (Standardwert:* `invisible`*):* **close** *Beim Schließen der Seite deaktivieren* **invisible** *Deaktivieren, wenn die Seite unsichtbar wird* **click** *Anmerkung muss durch ein Skript oder eine Benutzeraktion explizit deaktiviert werden.* **disablestate** *(Schlüsselwort) Zustand der Animation bei der Deaktivierung (Standardwert:* `reset`*):* **pause** *Das 3D-Modell kann dargestellt werden, aber Animationen werden deaktiviert.* **play** *Das 3D-Modell kann dargestellt werden und Animationen werden aktiviert.* **reset** *Anfangszustand des 3D-Modell, bevor es in irgendeiner Weise genutzt wurde.* **modeltree** *(Boolean; PDF 1.6) Bei* `true` *wird das Navigationsfenster »Modellhierarchie« bei Aktivierung der Anmerkung geöffnet (Standardwert:* `false`*)* **toolbar** *(Boolean; PDF 1.6) Bei* `true` *wird die 3D-Werkzeugleiste (am oberen Rand der Anmerkung) angezeigt, wenn die Anmerkung aktiviert wird (Standardwert:* `true`*)*
3Ddata	*(3D-Handle; nur für* `type=3D`*; erforderlich) 3D-Handle, das mit PDF_load_3ddata( ) erzeugt wurde.*
3Dinteractive	*(Boolean; nur für* `type=3D`*) Bei* `true` *ist das 3D-Modell zur interaktiven Nutzung vorgesehen. Bei* `false` *wird es mit JavaScript manipuliert. Standardwert:* `true`
3Dshared	*(Boolean; nur für* `type=3D`*) Bei* `true` *werden die in der Option* `3Ddata` *festgelegten 3D-Daten indirekt referenziert. Referenzieren mehrere 3D-Anmerkungen dieselben Daten, so nutzen sie dieselbe Laufzeitinstanz des Modells. Änderungen sind dann in allen Anmerkungen gleichzeitig sichtbar. Standardwert:* `false`
3Dinitialview	*(Schlüsselwort oder 3D-View-Handle) Legt die Anfangsdarstellung des 3D-Modells fest; Eines der Schlüsselwörter* `first`*,* `last`*, (bezogen auf die entsprechenden Einträge in der Option* `views` *von PDF_load_3ddata( )) oder* `default` *(bezogen auf die Option* `defaultview` *des Modells) oder ein mit PDF_create_3dview( ) erzeugtes 3D-View-Handle. Standardwert:* `default`

13.2 Asset und Rich Media Features (Flash)

Multimedia-Features sind mit folgenden Funktionen und Optionen implementiert:

- Multimedia-Assets für Rich-Media-Anmerkungen können mit *PDF_load_asset( )* geladen werden. Mit dieser Funktion können auch Assets geladen werden, die als Dateianhänge verwendet werden. Diese Funktion wird im Folgenden beschrieben.
- Rich-Media-Anmerkungen können mit *PDF_create_annotation( )* und *type=RichMedia* erzeugt werden. Diese Funktion wird in Abschnitt 12.2, »Anmerkungen«, Seite 229 beschrieben. Die entsprechenden Unteroptionen der Option *richmedia* finden Sie hingegen in Tabelle 13.7 und den folgenden Tabellen.
- Aktionen zur Steuerung von Rich-Media-Anmerkungen können mit *PDF_create_action( )* und *type=RichMediaExecute* erzeugt werden (siehe Abschnitt 12.4, »Aktionen«, Seite 246).

Mit *PDF_load_asset( )* geladene Assets können auch mit *PDF_create_annotation( )* und *type=FileAttachment* oder *Movie* verwendet werden.

C++ Java C# ***int load_asset(String type, String filename, String optlist)***

Perl PHP ***int load_asset(string type, string filename, string optlist)***

C ***int PDF_load_asset(PDF *p, const char *type, const char *filename, int len, const char *optlist)***

Lädt ein Rich-Media-Asset oder einen Dateianhang aus einer Datei auf der Festplatte oder einer virtuellen Datei.

type Schlüsselwort, das den Typ des geladenen Assets gemäß Tabelle 13.5 angibt.

Tabelle 13.5 Asset-Typen

Typ	*erlaubter Inhalt*	*Asset kann mit welcher Funktion und Option/Unteroption verwendet werden?*
3D	*U3D und PRC*	*PDF_create_annotation( ):* richmedia/configurations/instances/asset
Attachment[1]	*PDF/A-2: nur PDF/A-1 und PDF/A-2; sonst: beliebig*	*PDF_end_document( ):* attachments *PDF_create_annotation( ):* attachment *Nur PDF/A-3: PDF_end_document( ), PDF_begin/end_page_ext( ), PDF_begin/end_dpart( ), PDF_begin_template_ext( ), PDF_load_image( ), PDF_open_pdi_page( ), PDF_load_graphics( ):* associatedfiles
Flash	*Shockwave (*.swf)*	*PDF_create_annotation( ):* richmedia/configurations/instances/asset *PDF_end_document( ):* portfolio/navigator/flash
Generic	*beliebig*	*PDF_create_annotation( ):* richmedia/assets *PDF_end_document( ):* portfolio/navigator/assets
JavaScript	*Textdatei mit ECMA-Script edition 3*[2]	*PDF_create_annotation( ):* richmedia/activate/script
JPEG	*JPEG-Bild*	*PDF_end_document( ):* portfolio/navigator/icon
PNG	*PNG-Bild*	*PDF_end_document( ):* portfolio/navigator/icon
Sound	*MP3 etc.*	*PDF_create_annotation( ):* richmedia/configurations/instances/asset
Video	*FLV, QuickTime, F4V, H.264 etc.*	*PDF_create_annotation( ):* richmedia/configurations/instances/asset

1. type=Attachment wird implizit für PDF_add_portfolio_folder/file( ) und als Unteroption für die Option attachments von PDF_begin/end_document( ) angenommen
2. In ISO 8859-1 oder UTF-16 LE oder BE mit BOM

filename (Name-String; wird gemäß der globalen Option *filenamehandling* interpretiert, siehe Tabelle 2.3) Name einer auf der Festplatte liegenden oder virtuellen Datei, die in die PDF-Datei eingebettet werden soll. Unicode-Dateinamen werden unterstützt, erfordern zur korrekten Wiedergabe in Acrobat aber PDF 1.7.

len (Nur C-Sprachbindung) Länge von *filename* (in Bytes). Ist *len* = *0*, muss ein null-terminierter String übergeben werden.

optlist Optionsliste mit folgenden möglichen Optionen:
- Allgemeine Option für alle Typen: *errorpolicy* (siehe Tabelle 2.1)
- Bei *type=Attachment* werden zusätzliche Optionen für Anhänge gemäß Tabelle 13.6 unterstützt:
  description, documentattachment, external, filename, mimetype, name, password, relationship, thumbnail

Rückgabe Asset-Handle für Rich Media oder Dateianhänge, das mit den in Tabelle 13.5 aufgeführten Funktionen bis zum Ende des umschließenden Gültigkeitsbereichs *document* verwendet werden kann. Das zurückgegebene Asset-Handle kann nicht in mehreren PDF-Dokumenten gleichzeitig verwendet werden.

Bei *errorpolicy=return* muss der Aufrufer den Rückgabewert auf -1 (in PHP: 0) prüfen, da dies auf einen Fehler hinweist. Schlägt die Funktion fehl, so kann die Fehlerursache mit *PDF_get_errmsg( )* abgefragt werden.

Details Diese Funktion kann mit allen PDF-Kompatibilitätsstufen verwendet werden, wenn *type=Attachment* gesetzt ist. Für alle anderen Typen ist PDF 1.7ext3 erforderlich.

PDF/A PDF/A-1: Diese Funktion ist nicht zulässig.
PDF/A-2: *filename* muss sich auf ein PDF/A-1- oder PDF/A-2-Dokument beziehen. Einige Optionen sind eingeschränkt.
PDF/A-3: Einige Optionen sind eingeschränkt. Jeder Anhang muss genau einem Teil des Dokuments zugeordnet sein, das heißt, das generierte Asset-Handle muss an genau eine Option *associatedfiles* übergeben werden.

PDF/X PDF/X-1a/3: Diese Funktion ist nicht zulässig.

Gültigkeit beliebig außer *object*

Tabelle 13.6 Optionen für PDF_load_asset( ) mit `type=Attachment`*, für PDF_add_portfolio_folder/file( ) und als Unteroptionen für die Option* `attachments` *von PDF_begin/end_document( )*

Option	Beschreibung
description	*(Hypertext-String; PDF 1.6; empfohlen für PDF/A-2/3 und PDF/UA) Beschreibender Text für die Datei.*
document-attachment	*(Boolean; nur für PDF/A-3 und PDF 2.0 und nur wenn die Option* `relationship` *angegeben wird; nicht für PDF_add_portfolio_file/folder( )) Speichert die zugehörige Datei auch als Anhang auf Dokumentebene (eingebettete Datei). Dies kann für die Auflistung der Anhänge in der Benutzeroberfläche von PDF-Viewern nützlich sein, die keine Benutzeroberfläche für Associated Files bieten (dies gilt für Acrobat 9/X/XI). Standardwert:* `false`
external	*(Boolean; muss für PDF/A auf* `false` *gesetzt werden; nicht für die Option* `attachments` *und PDF_add_portfolio_file/folder( )) Bei* `true` *werden die Dateiinhalte nicht in das PDF eingebettet, sondern es wird ein Verweis auf eine externe Datei erzeugt. Die Option* `external=true` *ist erforderlich für Movies, damit sie in Acrobat abgespielt werden können. Standardwert:* `false`

Tabelle 13.6 Optionen für PDF_load_asset( ) mit `type=Attachment`*, für PDF_add_portfolio_folder/file( ) und als Unteroptionen für die Option* `attachments` *von PDF_begin/end_document( )*

Option	Beschreibung
filename	*(Name-String) Name der Datei. Die Inhalte müssen konform sein zu den inTabelle 13.5 aufgeführten Anforderungen gemäß dem angegebenen Typ. Unicode-Dateinamen werden unterstützt, erfordern jedoch PDF 1.7 für die korrekte Darstellung in Acrobat. Diese Option kann alternativ über den Funktionsparameter* `filename` *von PDF_load_asset( ) und PDF_add_portfolio_file/folder( ) übergeben werden, ist aber erforderlich, wenn sie mit der Option* `attachments` *von PDF_begin/end_document( ) verwendet wird. Es wird nur der Hauptbestandteil von* `filename` *ohne Verzeichnisangaben in die PDF-Ausgabe geschrieben.*
mimetype	*(String; erforderlich bei PDF/A-3; nicht für PDF_add_portfolio_folder( )) MIME-Typ der Datei. Ist der MIME-Typ unbekannt, muss bei PDF/A-3 der String* `application/octet-stream` *verwendet werden.*
name	*(Hypertext String; nicht für PDF_add_portfolio_folder( )) Name des Anhangs. Standardwert:* `filename` *ohne Pfadangabe*
password	*(String aus bis zu 127 Zeichen; nicht für PDF/A-2; nur wenn PDI verfügbar ist; nicht für PDF_add_portfolio_folder( )) PDF-Master-Kennwort wird benötigt, um die Datumseinträge eines geschützten PDF-Dokuments abzufragen.*
relationship	*(Hypertext-String; nur für PDF 2.0 und PDF/A-3; erforderlich für PDF/A-3; nicht für PDF_add_portfolio_folder( )) Beziehung zwischen der Datei und dem ihr zugeordneten Dokumentbestandteil. Der Wert kann ein beliebiger String sein, im PDF/A-3-Standard sind allerdings folgende vordefinierte Schlüsselwörter aufgeführt:* ***Alternative*** *Bei der Datei handelt es sich um eine alternative Darstellung (z.B. Audio).* ***Data*** *Bei der Datei handelt es sich um Informationen für eine visuelle Darstellung (z.B. CSV-Daten für eine Tabelle oder eine Grafik).* ***Source*** *Bei der Datei handelt es sich um das Originalmaterial (z.B. das dem Dokument zugeordnete Textverarbeitungsdokument; das einem Bild zugeordnete Spreadsheet).* ***Supplement*** *Bei der Datei handelt es sich um eine ergänzende Darstellung der ursprünglichen Quelle oder Daten, die leichter zu verarbeiten sind (z.B. MathML-Darstellung einer Gleichung in einem Bild).* ***Unspecified*** *Die Beziehung ist unbekannt oder kann mit den anderen Schlüsselwörtern nicht beschrieben werden.*
thumbnail	*(Image-Handle) Bild, das als Miniaturansicht für die Datei verwendet wird. Das Handle muss mit PDF_load_image( ) erzeugt worden sein, und das Bild muss alle für PDF_add_thumbnail( ) aufgeführten Bedingungen erfüllen. Acrobat ignoriert die Miniaturansicht bei Dateianhängen.*

Tabelle 13.7 Unteroptionen für die Option `richmedia` *von* PDF_create_annotation( ) *mit* `type=RichMedia`

Option	Beschreibung
activate	*(Optionsliste) Optionsliste gemäß Tabelle 13.8, die den Darstellungsstil, das Standardverhalten für Skripte, die Standardinformation für Views und den Animationsstil festlegt, wenn die Anmerkung aktiviert wird.*
assets	*(Liste aus Optionslisten; erforderlich) Benanntes Asset, das von Flash-Inhalten referenziert werden kann:* **asset** *(Asset-Handle; erforderlich) Handle für ein mit* PDF_load_asset( ) *geladenes Asset.* **name** *(Hypertext-String mit 1-255 Zeichen; die Zeichen : * " < > | sind nicht zulässig; das letzte Zeichen darf kein Punkt '.' sein; erforderlich) Name des Assets, mit dem das Asset im Flash-Code identifiziert werden kann.*
configuration	*(Optionsliste, erforderlich) Die Konfigurations-Optionsliste kann eine oder mehrere Instanz-Optionslisten enthalten. Mit der Option* `type` *kann das Verhalten für die Sammlung dieser Instanzen beschrieben werden. Zum Beispiel kann zum Betrachten einer FLV-Videodatei eine SWF-Datei erforderlich sein. Obwohl die erste Instanz in der Konfiguration eine SWF-Datei sein kann, ist die Anmerkung für die Video-Wiedergabe gedacht und sollte deshalb vom Typ* `Video` *sein. Die Option* `type` *lässt Rückschlüsse auf die vom Autor beabsichtigte Verwendung des Assets zu, was zu besseren Entscheidungen führt, wenn inhaltsspezifische Benutzeroberflächen während des Erstellungs- oder Bearbeitungsprozesses dargestellt werden. Unterstützte Optionen:* **instances** *(Liste aus Optionslisten; erforderlich) Jede Liste gibt eine einzelne Instanz eines Assets mit Einstellungen an, um das Modell einer Anmerkung zu füllen. Unterstützte Optionen:* **asset** *(Hypertext-String; erforderlich) Name eines in der Option* `assets` *angegebenen Rich-Media-Assets. Es dürfen hier nur die Namen von Rich-Media-Assets vom Typ* `3D`, `Flash`, `Sound` *oder* `Video` *angegeben werden.* **params** *(Optionsliste; nur für* `type=Flash`*) Die einem Flash-Asset zugeordneten Parameter gemäß Tabelle 13.10.* **name** *(Hypertext-String) Eindeutiger Name der Konfiguration* **type** *(Schlüsselwort) Primärer Inhaltstyp für die Konfiguration. Gültige Schlüsselwörter sind* `3D`, `Flash`, `Sound` *und* `Video`*. Standardwert: Der Typ der Szene wird durch einen Verweis auf den im ersten Element der Optionsliste* `instances` *angegebenen Dateityp bestimmt.*
deactivate	*(Optionsliste) Legt die Art der Deaktivierung fest (Neustart oder Pause):* **condition** *(Schlüsselwort) Legt fest, wann die Anmerkung deaktiviert wird (Standardwert:* `clicked`*):* **clicked** *Die Anmerkung wird explizit durch eine Benutzeraktion oder ein Skript deaktiviert.* **closed** *Die Anmerkung wird deaktiviert, sobald die Seite mit der Anmerkung den Fokus als aktuelle Seite verliert.* **invisible** *Die Anmerkung wird deaktiviert, sobald die Seite mit der Anmerkung nicht mehr sichtbar ist.*
views	*(Liste aus 3D-View-Handles) Von* PDF_create_3dview( ) *zurückgegebene 3D-View-Handles. Werden keine Views angegeben, so werden für die Komponenten eines 3D-Views die Standardwerte verwendet, einschließlich der Darstellungs-/Beleuchtungs-Modi, der Hintergrundfarbe und der Kameradaten. Standardwert: leere Liste*

Tabelle 13.8 Unteroptionen für die Unteroption `activate` *der Option* `richmedia` *von PDF_create_annotation()*

Option	Beschreibung
animation	*(Optionsliste) Bevorzugte Methode zur Steuerung von Keyframe-Animationen im 3D-Modell. Unterstützte Optionen:* **playcount** *(Integer) Nicht negative Zahl, die angibt, wie oft die Animation abgespielt wird. Eine negative Zahl bedeutet, dass die Animation unendlich oft wiederholt wird. Standardwert: -1* **speed** *(Positiver Float) Ein Wert größer als eins verkürzt die Dauer des Abspielens der Animation bzw. beschleunigt die Animation. Damit können Autoren die gewünschte Geschwindigkeit von Animationen einstellen, ohne deren Inhalte erneut zu bearbeiten. Standardwert: 1* **style** *(Schlüsselwort) Legt den Animationsstil fest (Standardwert:* `none`*):* **none** *Keyframe-Animationen sollten nicht direkt von der Viewer-Anwendung abgespielt werden. Dieser Wert wird von Dokumenten verwendet, die Animationen durch alternative Methoden wie JavaScript abspielen. Die verbleibenden Unteroptionen der Option* `animation` *werden ignoriert.* **linear** *Keyframe-Animationen werden linear vom Anfang bis zum Ende abgespielt. Dies führt zu einem wiederholten Durchspielen der Animation, wie bei einer Gehbewegung.* **oscillating** *Keyframe-Animationen pendeln entlang ihrer Zeitachse. Dies führt zu einem Abspielen in Vorwärts- und Rückwärts-Richtung, wie bei explodierenden oder kollabierenden Teilen.*
condition	*(Schlüsselwort) Legt fest, wann die Anmerkung aktiviert wird (Standardwert:* `clicked`*):* **clicked** *Die Anmerkung wird durch eine Benutzeraktion oder ein Script explizit aktiviert.* **opened** *Die Anmerkung wird aktiviert, sobald die Seite, auf der sich die Anmerkung befindet, den Fokus als aktuelle Seite erhält.* **visible** *Die Anmerkung wird aktiviert, sobald ein Teil der Seite sichtbar wird, auf der sich die Anmerkung befindet.*
presentation	*(Optionsliste) Legt fest, wie Anmerkungen und Elemente der Benutzeroberfläche angelegt und gezeichnet werden: für unterstützte Optionen siehe Tabelle 13.9.*
scripts	*(Liste aus Hypertext-Strings) Liste mit Namen von Rich-Media-Assets mit* `type=JavaScript`*, die mit PDF_load_asset() eingebettet und in der Unteroption* `assets` *der Option* `richmedia` *angegeben wurden. Eine leere Liste bedeutet, dass kein Skript ausgeführt wird. Standardwert: leere Liste*
view	*(Schlüsselwort oder 3D-View-Handle) Legt den Aktivierungs-View von 3D-Rich-Media fest. Das Handle muss auch in der Unteroption* `views` *der Optionsliste* `richmedia` *enthalten sein. Wurde die Option* `views` *nicht angegeben, werden für die Komponenten eines 3D-Views die Standardwerte verwendet. Unterstütztes Schlüsselwort (Standardwert:* `first`*):* **first** *Das erste Element in der Unteroption* `views` *der Optionsliste* `richmedia`

Tabelle 13.9 Unteroptionen für die Unteroption `presentation` *der Unteroption* `activate` *der Option* `richmedia` *von PDF_create_annotation( )*

Option	Beschreibung
navigationpane	*(Boolean) Standardverhalten der Navigationsleiste in der Benutzeroberfläche. Bei* `true` *wird die Navigationsleiste bei der ersten Aktivierung des Inhalts eingeblendet. Standardwert:* `false`
passcontextclick	*(Boolean) Gibt an, ob ein Kontext-Klick auf eine Rich-Media-Anmerkung an die Laufzeitumgebung des Media-Players übergeben wird oder vom PDF-Viewer verarbeitet wird. Bei* `false` *verarbeitet der PDF-Viewer den Kontext-Klick. Bei* `true` *ist das Kontextmenü sowie eventuell generierte benutzerdefinierte Elemente der Laufzeitumgebung des Media Players sichtbar und nicht das Kontextmenü des PDF-Viewers. Standardwert:* `false`
style	*(Schlüsselwort) Legt fest, wie Rich Media dargestellt wird (Standardwert:* `embedded`*):* ***embedded*** *eingebettet in der PDF-Seite* ***windowed*** *in einem separaten in der Optionsliste* `window` *angegebenen Fenster*
toolbar	*(Boolean) Standardverhalten der interaktiven Werkzeugleiste, die dieser Anmerkung zugeordnet ist. Bei* `true` *wird eine Werkzeugleiste angezeigt, sobald die Anmerkung aktiviert wird und sich im Fokus befindet. Standardwert:* `true` *bei* `type=3D`*, sonst* `false`
transparent	*(Boolean) Gibt an, ob die Seiteninhalte durch die transparenten Bereiche von Rich-Media-Inhalten hindurch dargestellt werden. Bei* `true` *wird das Rich-Media-Modell mit Hilfe eines Alphakanals mit den Seiteninhalten kombiniert. Bei* `false` *wird das Rich-Media-Modell über einem undurchsichtigen Hintergrund gezeichnet, bevor es mit den Seiteninhalten zusammengesetzt wird. Standardwert:* `false`
window	*(Optionsliste) Größe und Position des gleitenden Fensters bei* `style=windowed`*. Koordinaten werden abhängig von der Option* `usercoordinates` *in Benutzer- oder Standardkoordinaten interpretiert. Das Verhalten ist ähnlich, als wenn die Anmerkungsoptionen* `zoom` *und* `rotate` *auf* `false` *gesetzt werden. Unterstützte Unteroptionen:* ***heightdefault*** *(Float) Standardhöhe des Fensters. Standardwert: 216 (in Standardkoordinaten)* ***widthdefault*** *(Float) Standardbreite des Fensters. Standardwert: 288 (in Standardkoordinaten)*

Tabelle 13.10 Unteroptionen für die Unteroption `params` *der Unteroption* `instances` *der Unteroption* `configurations` *der Option* `richmedia` *von PDF_create_annotation( )*

Option	Beschreibung
binding	*(Schlüsselwort) Legt fest, an welche Einheit die Flash-Inhalte gebunden werden (Standardwert:* `none`*):* ***background*** *Flash-Inhalte werden an den Hintergrund gebunden und werden hinter allen 3D-Modell-Inhalten oder Flash-Vordergrundelementen in der aktiven Anmerkung wiedergegeben. Für eine Rich-Media-Anmerkung kann es eine aktive Instanz mit der Optionsliste* `params` *und* `binding=background` *geben. Wird mehr als eine Instanz angegeben, wird die für den Hintergrund zuletzt angegebene Instanz verwendet.* ***foreground*** *Flash-Inhalte sind an den Vordergrund gebunden und werden vor dem 3D-Modell und dem Hintergrund des Flash-Inhalts in der aktiven Anmerkung wiedergegeben. Ist für mehr als eine Instanz* `binding=foreground` *gesetzt, werden die Flash-Inhalte von hinten nach vorne wiedergegeben, wobei jede Instanz mit Hilfe eines Alphakanals mit vorigen Instanzen kombiniert wird.* ***material*** *Flash-Inhalte werden an Material gebunden, das Bestandteil von 3D-Inhalten ist. Wird das Material innerhalb von 3D-Szenengeometrie angewendet, scheint sich das Flash-Objekt sich der Oberfläche des Objekts anzupassen.* ***none*** *Flash-Inhalte sind nicht gebunden und bei der Wiedergabe unsichtbar.*
cuepoints	*(Liste aus Optionslisten) Ein Video kann Cue-Points enthalten, die im Video-Stream kodiert sind oder durch ein zugehöriges ActionScript im Flash-Inhalt erstellt werden. In jeder Optionsliste wird ein Zustand festgelegt, der einen Cue-Point mit einer Aktion verbindet, der an die Anwendung übergeben werden kann oder mit der das Aussehen verändert werden kann. Cue-Points im Flash-Inhalt werden mit den in den Optionen* `name` *oder* `time` *angegebenen Werten auf die Cue-Points in der PDF-Datei abgestimmt.* ***action*** *(Aktionsliste) Aktion, die ausgeführt wird, falls dieser Cue-Point ausgelöst wird, d.h., der Flash-Inhalt hat den passenden Cue-Point während der Wiedergabe erreicht. Unterstützter Auslöser für die Aktion:* ***activate*** *Aktionen, die ausgeführt werden, falls dieser Cue-Point ausgelöst wird, d.h., der Flash-Inhalt hat den passenden Cue-Point während der Wiedergabe erreicht.* ***name*** *(Text String; erforderlich) Name des Cue-Points zum Abgleich mit dem Cue-Point im Flash-Inhalt und für die Anzeige.* ***time*** *(Float; erforderlich) Zeitwert des Cue-Points in Millisekunden zum Abgleich mit dem Cue-Point im Flash-Inhalt und für die Anzeige. Beachten Sie, dass Acrobat 9 und X den Zeitstempel ignorieren. Da dieses Verhalten jedoch für zukünftige Versionen von Acrobat nicht garantiert werden kann, sollte ein korrekter Zeitstempel übergeben werden, sofern bekannt. Ansonsten kann ein Dummy-Wert übergeben werden.* ***type*** *(Schlüsselwort) Typ des Cue-Points:* ***Event*** *Ein Ereignis ist ein generischer Cue-Point ohne besondere Bedeutung außer, dass eine entsprechende Aktion ausgelöst wird.* ***Navigation*** *Ein Navigation-Cue-Point ist ein in einem Flash-Film (FLV) kodiertes Ereignis. Ein Stopp nach einem Kapitel kann so kodiert werden, dass, wenn der Benutzer zu einem Kapitel gehen oder ein Kapitel überspringen will, ein Navigation-Cue-Point verwendet wird, um die Position des Kapitels anzuzeigen.*
flashvars	*(Hypertext-String) Formatierte Namenspaare werden an den Flash-Player-Kontext übergeben, wenn dieser aktiviert wird. Für Informationen zum Format siehe das Dokument »Using FlashVars to pass variables to a SWF«, TechNote tn_16417, abrufbar unter www.adobe.com/go/tn_16417. Standardwert: an den Flash-Player werden keine Daten übergeben*
materialname	*(Hypertext-String; erforderlich bei* `binding=material`*) Materialname, an den Inhalte gebunden werden sollen.*

Tabelle 13.10 Unteroptionen für die Unteroption params *der Unteroption* instances *der Unteroption* configurations *der Option* richmedia *von PDF_create_annotation( )*

Option	Beschreibung
settings	*(Hypertext-String) Text-String zum Speichern von Einstellungen, die einer Flash-Instanz zugeordnet sind. Dieser Wert wird vom ActionScript-Befehl* multimedia_loadSettingsString *im* ExternalInterface *übergeben. Standardwert:* undefined

14 Dokumentaustausch

14.1 Dokument-Infofelder

C++ Java C# **void set_info(String key, String value)**

Perl PHP **set_info(string key, string value)**

C **void PDF_set_info(PDF *p, const char *key, const char *value)**

C **void PDF_set_info2(PDF *p, const char *key, const char *value, int len)**

Trägt *value* in das Dokument-Infofeld *key* ein.

key (Name-String) Name eines Standard-Infofelds oder eines benutzerdefinierten Felds, der beliebig gewählt sein kann (siehe Tabelle 14.1). Die Anzahl der benutzerdefinierten Felder ist nicht beschränkt. Wenn Sie mit benutzerdefinierten Dokument-Infofeldern arbeiten, empfehlen wir einen Blick auf den Dublin Core Metadata Element-Set.[1]

value (Hypertext-String) String, der dem Parameter *key* zugewiesen wird. Durch Acrobat ist die Länge von *value* auf maximal 255 Bytes beschränkt.

len (Nur *PDF_set_info2( )* und C-Sprachbindung) Länge von *value* (in Bytes). Ist *len = 0,* muss ein null-terminierter String übergeben werden.

Details Der übergebene Wert wird nur für das aktuelle und nicht für alle im selben Gültigkeitsbereich *object* erstellten Dokumente verwendet. Wurde die Option *autoxmp* an *PDF_begin/end_document( )* übergeben, erzeugt PDFlib automatisch entsprechende XMP-Dokumentmetadaten aus den Infoeinträgen, die an *PDF_set_info( )* übergeben wurden.

Dokument-Infofelder überschreiben die zugehörigen Properties in den XMP-Dokumentmetadaten, die mit der Option *metadata* von *PDF_begin/end_document( )* übergeben werden.

PDF/X Infofelder mit *key=Title* und *key=Creator* dürfen nicht mit leeren Werten übergeben werden. Bei PDF/X-4 und PDF/X-5 kann alternativ dazu die Option *metadata* von *PDF_begin_document( )* mit den Properties *dc:title* und *xmp:CreatorTool* XMP übergeben werden.

Für das Infofeld *Trapped* sind nur die Werte *True* und *False* zulässig.

PDF/UA Ein Infofeld mit *key=Title* darf nicht mit leeren Werten übergeben werden. Alternativ dazu kann die Option *metadata* von *PDF_begin_document( )* mit der XMP-Property *dc:title* übergeben werden.

Gültigkeit beliebig; wird diese Funktion im Gültigkeitsbereich *object* verwendet, so werden die übergebenen Werte nur für das nächste Dokument verwendet.

1. Siehe dublincore.org

Tabelle 14.1 Schlüssel für Dokument-Infofelder

Schlüssel	Beschreibung
Subject	Thema des Dokuments
Title	Titel des Dokuments
Creator	Programm, mit dem das Dokument erstellt wurde (im Gegensatz zum Producer der PDF-Ausgabe, der immer PDFlib ist). In Acrobat wird dieser Eintrag als »Anwendung« angezeigt.
Author	Verfasser des Dokuments
Keywords	Stichwörter für den Inhalt des Dokuments
Trapped	Gibt an, ob die Datei Überfüllungsinformationen enthält. Erlaubt sind die Werte `True`, `False` und `Unknown`.
Beliebiger Name	Benutzerdefiniertes Dokument-Infofeld. PDFlib unterstützt beliebig viele Felder. Der Name eines benutzerdefinierten Felds sollte nur einmal übergeben werden. Siehe auch die Option `moddate` für PDF_begin/end_document( ). Bei benutzerdefinierten Dokument-Infofelder nsind folgende Zeichen unzulässig, wenn (über die Optionen `autoxmp` oder `metadata`) XMP-Metadaten erstellt werden: `& \ < > "` und Leerzeichen Für die Standardidentifikation verwendete Felder sind unzulässig.

14.2 XMP-Metadaten

XMP-Metadaten können für ein gesamtes Dokument oder einzelne Seiten, Fonts, ICC-Profile, Bilder, Templates und importierten PDF-Seiten übergeben werden. In Tabelle 14.2 werden die Unteroptionen der Option *metadata* verschiedener Funktionen erläutert.

Tabelle 14.2 Unteroptionen für die Option `metadata` *in PDF_begin/end_document( ), PDF_begin/end_page_ext( ), PDF_load_font( ), PDF_load_iccprofile( ), PDF_load_image( ), PDF_begin_template_ext( ), PDF_open_pdi_page( ) sowie die Option* `templateoptions` *von PDF_load_graphics( )*

Option	Beschreibung
compress	*(Boolean; nicht für PDF_begin/end_document( )) Komprimiert den XMP-Metadaten-Stream in der PDF-Ausgabe. Wird die Option nur in PDF_begin_page_ext( ) übergeben, aber nicht in PDF_end_page_ext( ), hat dieser Wert Vorrang vor dem Standardwert. Standardwert:* `false` *PDF/A-1 und PDF/X:* `compress=true` *ist nicht zulässig.*
inputencoding	*(Schlüsselwort) Encoding zur Interpretation der in* `filename` *übergebenen Metadaten. Standardwert:* `unicode`
inputformat	*(Schlüsselwort) Format der in* `filename` *übergebenen Metadaten. Standardwert:* `utf8`*, jedoch* `bytes`*, falls* `inputencoding` *einem 8-Bit-Encoding entspricht*
keepxmp	*(Boolean; nur für PDF_load_image( ) und PDF_load_graphics( ); kann nicht mit* `filename` *kombiniert werden) XMP-Metadaten in einer Bild- oder Grafikdatei bleiben erhalten, d.h. an das entsprechende Bild im PDF-Dokument angehängt. XMP-Metadaten werden in den Bildformaten TIFF, JPEG und JPEG 2000 sowie in SVG-Grafiken berücksichtigt. Sind in einer Bild- oder Grafikdatei keine XMP-Metadaten vorhanden, hat diese Option keine Wirkung. Standardwert:* `false`
filename	*(Name-String; erforderlich, sofern* `keepxmp` *nicht übergeben wird) Name einer Datei mit syntaktisch korrekten XMP-Metadaten. Er wird gemäß der globalen Option* `filenamehandling` *interpretiert, siehe Tabelle 2.3.*

PDF/A Die XMP-Identifikationseigenschaften für PDF/A werden automatisch erstellt.

PDFlib passt relevante Einträge in benutzerdefinierten XMP-Streams an die Standard-Dokument-Infofelder an (analog zum *autoxmp*-Modus, der XMP an Dokument-Infofelder anpasst). PDFlib erzeugt jedoch keine zusätzlichen XMP-Einträge für benutzerdefinierte Dokument-Infofelder. Für weitere PDF/A-Anforderungen an XMP-Dokumentmetadaten siehe das PDFlib-Tutorial. Die folgende Validierung wird auf Dokument-Metadaten angewendet:

- PPDF/A-1: XMP muss konform sein zu XMP 2004 oder über eine Beschreibung des Extension-Schemas verfügen;
- PDF/A-2/3: XMP muss konform sein zu XMP 2005 oder über eine Beschreibung des Extension-Schemas verfügen;

PDF/X Die XMP-Identifikationseigenschaften für PDF/X-4/5 werden automatisch erzeugt.

PDF/VT Die XMP-Identifikationseigenschaften für PDF/VT werden automatisch erzeugt.

PDF/UA Die XMP-Identifikationseigenschaften für PDF/UA werden automatisch erzeugt.

14.3 Tagged PDF

Um Tagged PDF zu erzeugen, muss die Option *tagged* in PDF_begin_document( ) auf *true* gesetzt werden. Der Tagged-PDF-Modus wird für die Standards PDF/A-1a/2a/3a und PDF/UA automatisch aktiviert. Wir empfehlen bei der Erzeugung von Tagged PDF dringend, die Regeln für PDF/UA einzuhalten.

Cookbook *Ein vollständiges Codebeispiel finden Sie im Cookbook-Topic* `pdfua/starter-pdfua1/`.

C++ Java C# ***int begin_item(String tag, String optlist)***

Perl PHP ***int begin_item(string tag, string optlist)***

C ***int PDF_begin_item(PDF *p, const char *tag, const char *optlist)***

Öffnet ein Strukturelement oder einen anderen Dokumentbestandteil für Tagged PDF.

tagname Name des Strukturelementtyps des Elements. Folgende Gruppen von Elementtypen werden gemäß Tabelle 14.3 unterstützt (für weitere Informationen siehe das PDFlib-Tutorial):

- Standard-Elementtypen (für weitere Informationen zu den Standard-Elementtypen siehe das PDFlib-Tutorial)
- Pseudo-Elementtypen, bei denen es sich nicht um Strukturelemente handelt
- Der Name *Plib_custom_tag* weist auf die Verwendung eines benutzerdefinierten Elementtyps hin (dies entspricht der Option *customtag=true);* der eigentliche Tag-Name muss in der Option *tagname* übergeben werden. Für benutzerdefinierte Elementtypen muss die Dokumentoption *rolemap* übergeben werden.

Der Tag-Name kann alternativ in der Option *tagname* angegeben werden, die diesen Parameter überschreibt.

Tabelle 14.3 Standard-, Pseudo- und benutzerdefinierte Elementtypen für PDF_begin_item( ), PDF_begin_mc( ) und PDF_mc_point( ) sowie der Option `tag` *verschiedener Funktionen*

Kategorie	Tags
Standard-Strukturelementtypen	
Gruppierung	`Document`, `Part`, `Art`, `Sect`, `Div`, `BlockQuote`, `Caption`, `TOC`, `TOCI`, `Index`, `NonStruct`, `Private`
Überschrift und Absatz	`P`, `H`, `H1`, `H2`, `H3`, `H4`, `H5`, `H6`, `H7`, `H8` *usw. (BLSEs)*
Label und Liste	`L`, `LI`, `Lbl`, `LBody` *(BLSEs)*
Tabelle	`Table` *(BLSE),* `TR`, `TH`, `TD`, `THead`[1], `TBody`[1], `TFoot`[1] *(Tabellen-Tags können mit PDF_fit_table( ) automatisch erzeugt werden, siehe PDFlib-Tutorial)*
Inline-Level	`Span`, `Quote`, `Note`, `Reference`, `BibEntry`, `Code`, `Link`, `Annot`[1] *(ILSEs, können aber mit der Option* `inline=false` *in BLSEs umgewandelt werden)*
Illustration	`Figure`, `Formula`, `Form` *(ILSE)*
Japanisch	`Ruby`[1] *(zur Gruppierung),* `RB`[1], `RT`[1], `RP`[1], `Warichu`[1] *(zur Gruppierung),* `WT`[1], `WP`[1]

Tabelle 14.3 Standard-, Pseudo- und benutzerdefinierte Elementtypen für PDF_begin_item( ), PDF_begin_mc( ) und PDF_mc_point( ) sowie der Option `tag` *verschiedener Funktionen*

Kategorie	Tags	
Pseudo-Elementtypen		
keine Strukturelemente	***Artifact***	*Artefakt, das vom tatsächlichen Seiteninhalt zu unterscheiden ist.*
	ASpan	*(Accessibility Span; wird als* Span *ins PDF geschrieben, ist aber vom Inline-Element* Span *zu unterscheiden) Kann verwendet werden, um Inhalte mit Attributen für die Zugänglichkeit zu verknüpfen, die zu keinem Strukturelement gehören oder nur einen Teil eines Strukturelements darstellen.*
	ReversedChars	*(Nicht empfohlen) Legt Text in umgekehrter Zeichenreihenfolge in einer von rechts nach links geschriebenen Sprache fest.*
	Clip	*(Nicht empfohlen) Legt eine markierte Beschneidungssequenz fest. Dabei handelt es sich um eine Sequenz, die nur aus Beschneidungspfaden oder Text mit* `textrendering=7` *besteht und keine sichtbaren Grafiken oder PDF_save( )/PDF_restore( ) enthält.*
Benutzerdefinierte Elementtypen		
benutzerdefinierte Elemente	*Der Tag-Name* `Plib_custom_tag` *muss im Parameter* `tagname` *übergeben werden. Der eigentliche Tag-Name, der in die PDF-Ausgabe geschrieben wird, muss in der Option* `tagname` *übergeben werden. Benutzerdefinierte Elementtypen erfordern die Dokumentoption* `rolemap`.	

1. Erfordert PDF 1.5 oder höher

optlist Optionsliste mit Elementeigenschaften. Alle vererbbaren Einstellungen werden an die untergeordneten Elemente vererbt und brauchen deshalb nicht wiederholt zu werden. Alle Elementeigenschaften müssen hier eingestellt werden, da sie später nicht mehr veränderbar sind. Folgende Optionen können verwendet werden:

- Allgemeine Option: *hypertextencoding* (siehe Tabelle 2.3)
- Optionen zur Steuerung von Tags und zur Barrierefreiheit gemäß Tabelle 14.4: *ActualText, Alt, customtag, E, inline, Lang, tagname, Placement, Title*
- Auf Artefakte bezogene Optionen gemäß Tabelle 14.4: *artifactsubtype, artifacttype, Attached*
- Tabellenoptionen gemäß Tabelle 14.4: *ColSpan, Headers, id, RowSpan, Scope, Summary*
- Geometrieoptionen gemäß Tabelle 14.4: *BBox, Height, usercoordinates, Width*
- Optionen für die Beziehung von Elementen gemäß Tabelle 14.4: *index, parent*
- Option für das Anbringen eines Lesezeichens gemäß Tabelle 14.4: *bookmark*
- Option für die Angabe einer Listeneigenschaft gemäß Tabelle 14.4: *ListNumbering*
- Option für das Einfügen eines verschachtelten Strukturelements gemäß Tabelle 14.4: *tag*

Rückgabe Element-Handle, das in nachfolgenden elementspezifischen Aufrufen verwendet werden kann.

Details Diese Funktion beginnt ein neues Strukturelement oder Artefakt (beides *Element* genannt). Das Element wird standardmäßig dem aktuell aktiven Element untergeordnet, kann jedoch mit den Optionen *parent* und *index* an einer anderen Position im Strukturbaum platziert werden. Strukturelemente können beliebig tief verschachtelt werden. Außer Pseudo- und Inline-Elementtypen sind Strukturelemente nicht an die Seite ge-

bunden, auf der sie geöffnet wurden, sondern können auf einer beliebigen Anzahl von Seiten fortgesetzt werden. Wir empfehlen, auf leere Strukturelemente zu verzichten.

Strukturelemente und *Alt/ActualText*-Attribute müssen gemäß den Regeln im PDFlib-Tutorial verschachtelt werden. Einige Elemente für die Textdekoration werden automatisch als *Artifact* ausgezeichnet; für weitere Informationen siehe PDFlib-Tutorial.

PDF/A Obwohl das Anbringen von Tags für PDF/A-1a/2a/3a erforderlich ist, gibt es keine besonderen Anforderungen bezüglich Tag-Nutzung oder Verschachtelung.

PDF/UA Alle Texte erfordern eine natürliche Sprachspezifikation.
Bild- und Grafikinhalte müssen als *Artifact* oder *Figure* ausgezeichnet werden.
Für manche Elementtypen und Optionen gelten zusätzliche Regeln (siehe PDFlib-Tutorial).

Gültigkeit *page;* für Gruppierungselemente auch *document;* diese Funktion muss immer paarweise mit *PDF_end_item()* aufgerufen werden. Sie ist nur im Tagged-PDF-Modus zulässig.

Tabelle 14.4 Optionen für Struktur- und Pseudo-Tags in PDF_begin_item() PDF_begin_mc() und PDF_mc_point() sowie für vereinfachtes Tagging mit der Option tag *verschiedener Funktionen*

Option	Beschreibung
ActualText	*(Hypertext-String; nicht für Pseudo-Tags außer in PDF 1.5 für* ASpan*; wird diese Option im PDF-1.4-Modus verwendet, muss die Option* inline *auf* false *gesetzt sein) Alternativtext für den Dokumentbestandteil einschließlich seiner untergeordneten Elemente.*
Alt	*(Hypertext-String; nicht für Pseudo-Tags außer in PDF 1.5 für* ASpan*; wird diese Option im PDF-1.4-Modus verwendet, muss die Option* inline *auf* false *gesetzt sein) Alternative Beschreibung für den Dokumentbestandteil einschließlich seiner untergeordneten Elemente. Sie sollte für Bilder usw. übergeben werden, die sich nicht als Text interpretieren lassen.*
artifact-subtype	*(Schlüsselwort; nur für* tagname=Artifact *und* artifacttype=Pagination*; PDF 1.7) Untertyp des Artefakts:* Header, Footer, Watermark
artifacttype	*(Schlüsselwort; nur für* tagname=Artifact*) Bezeichnet den Artefakttyp des Dokumentbestandteils:* **Pagination** *Paginierungseigenschaften wie Kolumnentitel oder Seitenzahlen* **Layout** *Kosmetische typografische oder Layout-Elemente wie Fußnotenlinien oder Tabellenschattierung* **Page** *Produktionshilfen wie Beschnittmarken oder Farbbalken* **Background** *(PDF 1.7) Über die gesamte Länge und/oder Breite einer Seite oder eines Strukturelements verlaufende Bilder oder farbige Bereiche.*
Attached	*(Schlüsselwortliste; nur für* tagname=Artifact *und* artifacttype=Pagination *oder* Background *mit ganzseitigen Hintergrund-Artefakten) Legt die Kanten der Seite fest, falls dies der Fall ist, an die das Artefakt logisch angehängt ist. Die Liste enthält ein bis vier der Schlüsselwörter* Top, Bottom, Left *und* Right*. Sind sowohl* Left *als auch* Right *bzw. sowohl* Top *als auch* Bottom *vorhanden, handelt es sich um ein Artefakt über die gesamte Seitenbreite bzw. über die gesamte Seitenhöhe.*
BBox	*(Rechteck; nur für* tagname=Artifact , Figure, Form, Table*; erforderlich für* artifacttype=Background*, sonst optional, aber für das Umfließen von Text ratsam) Die Größe der Bounding-Box des Elements in Standardkoordinaten (bei* usercoordinates=false*) oder in Benutzerkoordinaten (bei* usercoordinates=true*). PDFlib erzeugt automatisch einen* BBox*-Eintrag für importierte Bilder, Grafiken und PDF-Seiten (mit* tagname=Figure *oder* Artifact*), für Formularfelder (mit* tagname=Form *oder* Artifact*) und für Tabellen, die mit dem Tabellengenerator erzeugt wurden (mit* tagname=Table *or* Artifact*).*
bookmark	*(Lesezeichen-Handle; nicht für Inline- und Pseudo-Tags) Handle für ein Lesezeichen, das mit dem Strukturelement verknüpft ist.*
ColSpan	*(Integer; nur für* tagname=TH *und* TD*) Anzahl der von einer Zelle belegten Tabellenspalten. Standardwert: 1*

Tabelle 14.4 Optionen für Struktur- und Pseudo-Tags in PDF_begin_item( ) PDF_begin_mc( ) und PDF_mc_point( ) sowie für vereinfachtes Tagging mit der Option `tag` *verschiedener Funktionen*

Option	Beschreibung
customtag	*(Boolean; erfordert die Dokumentoption* `rolemap`*) Bei* `true` *ist der in der Option* `tagname` *übergebene Name des Elementtyps ein benutzerdefinierter Elementtyp, der mit der Option* `rolemap` *einem Standardelementtyp zugeordnet werden muss. Standardwert:* `false` *Wird diese Option auf* `true` *gesetzt, ist dies äquivalent zur Übergabe des Parameters* `tagname=Plib_custom_tag`.
E	*(Hypertext-String; nicht für Pseudo-Tags außer* `ASpan`*; für Struktur-Tags ist PDF 1.5 erforderlich) Abkürzungsergänzung für den Dokumentbestandteil. Sie sollte zur Erklärung von Abkürzungen und Akronymen übergeben werden. Die Sprachausgabe von Acrobat sieht diese Ergänzung auch ohne explizite Wortgrenze als eigenes Wort an.*
Headers	*(String-Liste; nur für* `tagname=TH` *und* `TD`*; PDF 1.5) Jeder String in der Liste ist ein Identifikator für eine Tabellenkopfzelle (Element* `TH`*), der mit der Zelle verbunden ist. Die Identifikatoren müssen der Zielzelle mit der Option* `id` *zugewiesen worden sein. Wenn mehrere Kopfzellen referenziert werden, sollten Zeilenkopfzellen vor Spaltenkopfzellen aufgeführt werden. Innerhalb dieser Gruppen sollten Kopfzellen von spezifisch nach allgemein sortiert werden.*
Height	*(Float; nur für* `tagname=Figure, Form, Formula, Table, TD, TH`*) Höhe des Elements in Standardkoordinaten (bei* `usercoordinates=false`*) oder in Benutzerkoordinaten (bei* `usercoordinates=true`*)*
id	*(String; nicht für Inline- und Pseudo-Elemente außer* `Note`*; erforderlich für* `tagname=Note` *in PDF/UA) Weist dem Element einen Identifikator zu. Der String muss innerhalb aller Strukturelemente eindeutig sein.*
index	*(Integer; nicht für Pseudo-Tags) Auf Null basierender Index, an dem das Element im übergeordneten Elements eingefügt wird. Bereits vorhandene untergeordnete Elemente werden von hier aus nach oben verschoben. Es können Werte zwischen 0 und der aktuellen Anzahl an untergeordneten Elementen übergeben werden. Mit -1 lässt sich das Element an letzter Position eintragen. Äquivalent dazu kann die aktuelle Anzahl an Elementen als Index angegeben werden. Standardwert: -1*
inline	*(Boolean; nur für* `tagname=Code, BibEntry, Note, Quote, Reference, Span`*) Bei* `true` *wird der Dokumentbestandteil* `inline` *geschrieben und kein Strukturelement erzeugt. Standardwert:* `true`
Lang	*(String; nicht für Pseudo-Tags außer* `ASpan`*) Sprachidentifikation für den Dokumentbestandteil in dem Format, das in Tabelle 3.3 für die Option* `lang` *beschrieben wird. Damit lässt sich die für das Dokument eingestellte Sprache für einzelne Dokumentbestandteile überschreiben.* *PDF/UA: die natürliche Sprache muss in dieser Option oder in der Option* `lang` *von PDF_begin_document( ) übergeben werden.*

Tabelle 14.4 Optionen für Struktur- und Pseudo-Tags in PDF_begin_item( ) PDF_begin_mc( ) und PDF_mc_point( ) sowie für vereinfachtes Tagging mit der Option `tag` *verschiedener Funktionen*

Option	Beschreibung
List-Numbering	*(Schlüsselwort; nur für* `tagname=L`*; erforderlich für PDF/UA; muss bei PDF/UA für Listen, in denen kein untergeordnetes* `LI`*-Element ein* `Lbl`*-Element enthält, auf* `None` *gesetzt werden) Nummerierungssystem für den Inhalt der* `Lbl`*-Elemente in einer nummerierten Liste oder das Symbol, das vor jedem Listenelement in einer nicht nummerierten Liste steht (Standardwert:* `None`*):* **Circle** *Nicht ausgefüllter Kreis* **Decimal** *Arabische Dezimalzahlen (1–9, 10–99, ...)* **Disc** *Ausgefüllter Kreis* **LowerAlpha** *Kleinbuchstaben (a, b, c, ...)* **LowerRoman** *Kleine römische Ziffern (i, ii, iii, iv, ...)* **None** *Keine automatische Nummerierung;* `Lbl`*-Elemente (sofern vorhanden) enthalten beliebigen Text ohne Nummerierung. In diesem Fall sollten die als Listenlabels verwendeten Grafiken als Artefakte ausgezeichnet werden.* **Square** *Ausgefüllte Quadrate* **UpperAlpha** *Großbuchstaben (A, B, C, ...)* **UpperRoman** *Große römische Ziffern (I, II, III, IV, ...)*
parent	*(Element-Handle; nicht für Pseudo-Tags) Element-Handle des übergeordneten Elements, das von einem früheren Aufruf von PDF_begin_item( ) oder dem Schlüsselwort* `activeitemid` *von PDF_get_option( ) zurückgegeben wurde. Der Wert 0 bezeichnet die Wurzel des Strukturbaums. -1 bezeichnet das übergeordnete Element des auf der aktuellen Seite zuletzt geöffneten Elements.* `parent=-1` *öffnet also ein Element auf gleicher Ebene wie das aktuelle Element. Standardwert: -1*
Placement	*(Schlüsselwort; nicht für Pseudo-Elemente und die Inline-Elemente* `Span`,`Quote`, `Note`, `Reference`, `BibEntry` *und* `Code`*) Positionierung des Elements bezogen auf den umgebenden Referenzbereich. Dies ist relevant, wenn das Dokument neu formatiert oder in andere Formate exportiert wird. Die Option* `Placement=Block` *wird für solche Elemente vom Typ* `Figure`, `Formula`, `Form`, `Link`, `Annot` *empfohlen, die Gruppierungselementen untergeordnet sind (Standardwert:* `Inline`*):* **Before** *So platziert, dass die Kante »vor« dem reservierten Rechteck des Elements mit der Kante des nächsten umgebenden Referenzbereichs zusammenfällt.* **Block** *In Blockformatierungsrichtung platziert innerhalb des umgebenden Referenzbereichs oder des übergeordneten BLSEs.* **End** *So platziert, dass die Kante »nach« dem reservierten Rechteck des Elements mit des Kante des nächsten umgebenden Referenzbereichs zusammenfällt.* **Inline** *In Inline-Formatierungsrichtung platziert innerhalb des übergeordneten BLSEs.* **Start** *So platziert, dass die »Start«-Kante des reservierten Rechtecks des Elements mit der Kante des nächsten umgebenden Referenzbereichs zusammenfällt.*
RowSpan	*(Integer; nur für* `tagname=TH` *und* `TD`*) Anzahl der Tabellenzeilen, die von einer Zelle belegt werden*
Scope	*(Schlüsselwort; nur für* `tagname=TH`*; PDF 1.5; bei PDF/UA erforderlich) Eines der Schlüsselwörter* `Row`, `Column` *oder* `Both`*, die anzeigen, ob sich die Tabellenkopfzelle auf alle Zellen in derselben Zeile, alle Zellen in derselben Spalte oder auf beides erstreckt.*
Summary	*(Hypertext-String; nur für* `tagname=Table`*; PDF 1.7) Zusammenfassung von Struktur und Zweck der Tabelle*
tag	*(Optionsliste) Zusätzliches Strukturelement, das als untergeordnetes Element des aktuellen Elements hinzugefügt wird. Alle Optionen gemäß Tabelle 14.4 werden als Unteroptionen unterstützt. Tags können beliebig tief verschachtelt werden.*

Tabelle 14.4 Optionen für Struktur- und Pseudo-Tags in PDF_begin_item( ) PDF_begin_mc( ) und PDF_mc_point( ) sowie für vereinfachtes Tagging mit der Option `tag` *verschiedener Funktionen*

Option	Beschreibung
tagname	*(Name-String; außer in PDF_add_table_cell( ) ist diese Option für vereinfachtes Tagging mit der Option* `tag` *erforderlich) Name eines Standard-Elementtyps oder eines Pseudo-Elementtyps gemäß Tabelle 14.3 bzw. Name eines benutzerdefinierten Tags. Der Wert dieser Option kann alternativ mit dem Funktionsparameter* `tagname` *übergeben werden.* *Geben Sie die Option* `customtag=true` *oder den Parameter* `tagname=Plib_custom_tag` *zur Übergabe von benutzerdefinierten Tag-Namen an. In diesem Fall kann die Option* `tagname` *einen beliebigen Namen eines benutzerdefinierten Tags enthalten, für den eine Zuordnung auf einen Standard-Elementtyp mit der Option* `rolemap` *von PDF_begin_document( ) definiert sein muss. Namen für benutzerdefinierte Elementtypen sind auf 127 winansi-Zeichen oder eine Sequenz von Unicode-Zeichen beschränkt, die maximal 127 UTF-8-Bytes erzeugt. Namen für benutzerdefinierte Elementtypen dürfen nicht mit dem reservierten Präfix* `Plib` *beginnen.*
Title	*(Hypertext-String; nicht für Inline- und Pseudo-Tags; für Überschriften in PDF/UA empfohlen) Name des Strukturelements. Der Name kann beim anzeigen oder Bearbeiten des Strukturbaums in Acrobat nützlich sein.*
user-coordinates	*(Boolean) Bei* `false` *werden* `BBox`, `Width` *und* `Height` *in Standardkoordinaten erwartet; anderenfalls werden Benutzerkoordinaten verwendet. Standardwert: Wert der globalen Option* `usercoordinates`
Width	*(Float; nur für* `tagname=Figure`, `Form`, `Formula`, `Table`, `TD`, `TH`*) Breite des Elements in Standardkoordinaten (bei* `usercoordinates=false`*) oder in Benutzerkoordinaten (bei* `usercoordinates=true`*)*

C++ Java C# ***void end_item(int id)***

Perl PHP ***end_item(int id)***

C ***void PDF_end_item(PDF *p, int id)***

Schließt ein Strukturelement oder einen anderen Dokumentbestandteil.

id Element-Handle, das von *PDF_begin_item( )* zurückgegeben wurde.

Details Alle Inline-Elemente müssen vor dem Seitenende und alle regulären Elemente vor dem Dokumentende geschlossen werden. Um den Speicherbedarf zu verringern, sollten reguläre Elemente aber unmittelbar nach Beendigung geschlossen werden. Ein Element kann nur geschlossen werden, wenn zuvor all seine untergeordneten Elemente geschlossen wurden. Nach dem Schließen eines Elements wird sein übergeordnetes Element zum aktiven Element.

Gültigkeit *page* für Inline-Elemente; für Gruppierungselemente auch *document;* diese Funktion muss immer paarweise mit *PDF_begin_item( )* aufgerufen werden. Sie ist nur im Tagged-PDF-Modus zulässig.

Vereinfachtes Tagging. Strukturelemente und Artefakte können mit Paaren aus *PDF_begin/end_item( )* erzeugt werden. Alternativ dazu steht vereinfachtes Tagging mit der Option *tag* in folgenden Funktionen zur Verfügung (siehe Tabelle 14.5):

- *PDF_add_table_cell( )* und die zugehörigen Optionen von *PDF_add_table_cell( ): fitgraphics, fitimage, fitpath, fitpdipage, fittextline, fittextflow, fitannotation, fitfield*
- *PDF_begin_document( ):* mit vereinfachtem Tagging kann das Stammelement der Strukturhierarchie erzeugt werden
- *PDF_create_annotation( )*
- *PDF_create_field( )*

- *PDF_draw_path( )*
- *PDF_fit_graphics( )*
- *PDF_fit_image( )*
- *PDF_fit_pdi_page( )* und *PDF_info_pdi_page( )*
- *PDF_fit_table( )*; vereinfachtes Tagging bewirkt automatisches Erstellen von Tabellen-Tags
- *PDF_fit_textflow( )*
- *PDF_fit_textline( )*
- die Option *matchbox* verschiedener Funktionen

Vereinfachtes Tagging kann nicht zur Erzeugung von Gruppierungselementen verwendet werden, außer wenn es mit *PDF_begin_document( )* verwendet wird (da alle anderen Funktionen direkten Inhalt erzeugen, was für Gruppierungselemente nicht zulässig ist). Für weitere Informationen zum vereinfachten Tagging siehe das PDFlib-Tutorial.

Tabelle 14.5 Optionen zum vereinfachten Tagging in PDF_add_table_cell( ) und den zugehörigen Optionen `fit*` *in PDF_add_table_cell( ), PDF_create_annotation( ), PDF_create_field( ), PDF_draw_path( ), PDF_fit_graphics( ), PDF_fit_image( ), PDF_fit_pdi_page( ), PDF_fit_table( ), PDF_fit_textflow( ), PDF_fit_textline( ) sowie der Option* `matchbox` *für verschiedene Funktionen*

Option	Beschreibung
tag	*(Optionsliste) Erzeugt ein Strukturelement oder Artefakt für den platzierten Inhalt. Die in Tabelle 14.4 aufgeführten Unteroptionen können verwendet werden.*

C++ Java C# ***void activate_item(int id)***

Perl PHP ***activate_item(int id)***

C ***void PDF_activate_item(PDF *p, int id)***

Aktiviert ein zuvor erzeugtes Strukturelement oder einen anderen Dokumentbestandteil.

id Element-Handle, das von *PDF_begin_item( )* zurückgegeben und noch nicht geschlossen wurde. Pseudo- und Inline-Elemente lassen sich nicht aktivieren.

Details Es erhöht die Flexibilität beim effizienten Erstellen von Tagged-PDF-Seiten, wenn man ein Strukturelement suspendiert und später wieder aktiviert. Dies kann sinnvoll sein, wenn mehrere Zweige des Strukturbaums auf der Seite parallel verlaufen, zum Beispiel bei mehrspaltigen Layouts oder Texteinschüben, die den Haupttext unterbrechen.

Während sich mit den Tagging-Optionen *parent* und *index* (siehe Tabelle 14.4) Strukturelemente an einer bestimmten Stelle im Strukturbaum platzieren lassen, kann mit *PDF_activate_item( )* weiterer Inhalt zu einem zuvor erstellten Strukturelement hinzugefügt werden.

Um Probleme in Acrobat zu vermeiden, sollten unmittelbar nach dem Aufruf von *PDF_activate_item( )* nur andere Strukturelemente, aber keine direkten Inhalte hinzugefügt werden.

Gültigkeit *document, page;* diese Funktion ist nur im Tagged-PDF-Modus zulässig.

14.4 Markierter Inhalt

C++ Java C# ***void begin_mc(String tagname, String optlist)***
Perl PHP ***begin_mc(string tagname, string optlist)***
C ***void PDF_begin_mc(PDF *p, const char *tagname, const char *optlist)***

Beginnt eine markierte Inhaltssequenz mit optionalen Eigenschaften.

tagname Name der markierten Inhaltssequenz. Folgende Tags werden unterstützt:
- Alle Inline- und Pseudo-Tags in Tabelle 14.3.
- Der Tag-Name *Plib_custom* kann für benutzerdefinierte Einträge und Eigenschaften verwendet werden.
- Der Tag-Name *Plib* ist reserviert.

optlist Folgende Optionen für markierten Inhalt werden unterstützt:
- Optionen für Standardeigenschaften von markiertem Inhalt gemäß Tabelle 14.4.
- Die Tags *Plib_custom* und *Plib* unterstützen zusätzlich die Option in Tabelle 14.6.

Details Es wird eine Sequenz mit markiertem Inhalt mit den angegebenen Tags und Eigenschaften gestartet. Werden keine Optionen übergeben, wird eine Sequenz ohne Eigenschaften erzeugt. Sequenzen mit markiertem Inhalt können beliebig tief verschachtelt werden. Für das korrekte Verschachteln von Sequenzen aus *PDF_begin/end_item()* und *PDF_begin/end_mc()* sind die Benutzer verantwortlich.

Gültigkeit *page, pattern, template, glyph;* diese Funktion muss immer paarweise mit *PDF_end_mc()* im selben Gültigkeitsbereich aufgerufen werden.

Tabelle 14.6 Option für benutzerdefinierte Eigenschaften von Tags mit PDF_begin_mc() und PDF_mc_point()

Option	Beschreibung	
properties	*(Liste aus Optionslisten; nur für* `tagname=Plib` *und* `tagname=Plib_custom`*) Jede Liste enthält drei Optionen, die eine benutzerdefinierte Eigenschaft festlegen:*	
	key	*(String; erforderlich) Name der Eigenschaft*
	type	*(Schlüsselwort; erforderlich) Typ des Eigenschaftswerts:* `boolean`*,* `name` *oder* `string`
	value	*(Hypertext-String bei* `type=string`*, sonst String; erforderlich) Wert der Eigenschaft*

C++ Java C# ***void end_mc()***
Perl PHP ***end_mc()***
C ***void PDF_end_mc(PDF *p)***

Beendet die zuletzt geöffnete Sequenz mit markiertem Inhalt.

Details Alle Sequenzen mit markiertem Inhalt müssen vor dem Aufruf von *PDF_end_page_ext()* geschlossen werden.

Gültigkeit *page, pattern, template, glyph;* diese Funktion muss immer paarweise mit *PDF_end_mc()* im selben Gültigkeitsbereich aufgerufen werden.

C++ Java C# ***void mc_point(String tagname, String optlist)***

Perl PHP ***mc_point(string tagname, string optlist)***

C ***void PDF_mc_point(PDF *p, const char *tagname, const char *optlist)***

Fügt einen markierten Inhaltspunkt mit optionalen Eigenschaften hinzu.

tagname Name des markierten Inhaltspunkts. Folgende Tags werden unterstützt:
- Alle Inline-Level- und Pseudo-Tags in Tabelle 14.3.
- Der Tag-Name *Plib_custom* kann für benutzerdefinierte Einträge verwendet werden.
- Der Tag-Name *Plib* ist reserviert.

optlist Folgende Optionen können verwendet werden:
- Optionen für Standardeigenschaften des markierten Inhaltspunkts gemäß Tabelle 14.4.
- Die Tags *Plib_custom* und *Plib* unterstützen zusätzlich die Option in Tabelle 14.6.

Details Es wird ein markierter Inhaltspunkt mit den angegebenen Tags und Eigenschaften erzeugt. Werden keine Optionen übergeben, wird ein markierter Inhaltspunkt ohne Eigenschaften erzeugt.

Gültigkeit *page, pattern, template, glyph*

14.5 Document Part Hierarchy

C++ Java C# ***void begin_dpart(String optlist)***

Perl PHP ***begin_dpart(string optlist)***

C ***void PDF_begin_dpart(PDF *p, const char *optlist)***

Öffnet einen neuen Knoten in der Document Part Hierarchy (erfordert PDF/VT oder PDF 2.0).

optlist Optionsliste mit Optionen für die Document Part Hierarchy gemäß Tabelle 14.7: *associatedfiles, dpm*

PDF/VT Der erste Aufruf von *PDF_begin_dpart( )* erzeugt implizit den Wurzelknoten der Document Part (DPart-) Hierarchy. Der mehrmalige Aufruf von *PDF_begin_dpart( )* auf oberster Ebene führt zu einem Fehler.

Ein Aufruf von *PDF_begin_dpart( )* gefolgt von einem Aufruf von *PDF_begin_page_ext( )* definiert den Anfang eines Seitenbereichs, eines Dokumentbestandteils. Alle folgenden Seiten gehören bis zum nächsten Aufruf von *PDF_begin_dpart( )* zum selben Dokumentbestandteil. Alle Aufrufe zusammen erzeugen die Document Part Hierarchy als Strukturbaum, der zwei Arten von knoten enthalten kann:

- Innere Knoten enthalten ein oder mehrere untergeordnete Knoten. Die untergeordneten Knoten können innere Knoten oder Blattknoten sein. Ein innerer Knoten kann allerdings nicht beide Knotentypen gleichzeitig enthalten.
- Blattknoten beschreiben die Seite(n) in einem Bereich. Blattknoten können keine untergeordneten Knoten enthalten.

Wenn *PDF_end_document( )* aufgerufen wird, müssen die Aufrufe von *PDF_begin_dpart( )* und *PDF_end_dpart( )* ausgeglichen sein. Soll für das Dokument eine Document Part Hierarchy erzeugt werden, muss diese Funktion mindestens einmal vor dem ersten Aufruf von *PDF_begin_page_ext( )* aufgerufen werden. Wird *PDF_begin_dpart( )* mehrfach aufgerufen, wird die Document Part Hierarchy dadurch tiefer verschachtelt. Die erzeugte Verschachtelungstiefe der Document Part Hierarchy (d.h. die höchste Anzahl an Verschachtelungsebenen von *PDF_begin_dpart( )*) muss mit der Länge der Liste übereinstimmen, die mit der Dokumentoption *nodenamelist* übergeben wurde.

Gültigkeit *document;* diese Funktion muss immer paarweise mit *PDF_end_dpart( )* aufgerufen werden.

Tabelle 14.7 Optionen für PDF_begin/end_dpart( )

Option	Beschreibung
associatedfiles	*(Liste von Asset-Handles; nur bei PDF 2.0 und PDF/A-3) Asset-Handles für assoziierte Dateien gemäß PDF/A-3. Die Dateien müssen mit PDF_load_asset( ) und* `type=attachment` *geladen worden sein.*
dpm	*(POCA-Container-Handle; kann an PDF_begin_dpart( ) oder an PDF_end_dpart( ) übergeben werden, aber nicht an beide Funktionen für den selben Dokumentbestandteil) Handle für einen mit PDF_poca_new( ) erzeugten Dictionary-Container, der Document Part Metadata für den neuen Knoten enthält. Das Dictionary muss mit der Option* `usage=dpm` *erzeugt worden sein.*

C++ Java C# **void end_dpart(String optlist)**

Perl PHP **end_dpart(string optlist)**

C **void PDF_end_dpart(PDF *p, const char *optlist)**

Schließt einen Knoten in der Document Part Hierarchy (erfordert PDF/VT oder PDF 2.0).

optlist Optionsliste mit Optionen für die Document Part Hierarchy gemäß Tabelle 14.7: *associatedfiles, dpm*

PDF/VT Der erste Aufruf von *PDF_end_dpart( )* nach *PDF_end_page_ext( )* definiert implizit das Ende des Seitenbereichs, der zu der Document Part Hierarchy gehört. Die Aufrufe von *PDF_begin_dpart( )* und *PDF_end_dpart( )* müssen ausgeglichen sein, wenn *PDF_end_document( )* aufgerufen wird.

Gültigkeit *document;* diese Funktion muss immer paarweise mit *PDF_end_dpart( )* aufgerufen werden.

A Liste aller Funktionen

Um zur Funktionsbeschreibung zu gelangen, klicken Sie auf die jeweilige Funktion.

Grafikzustand
set_graphics_option
setlinewidth
save
restore
create_gstate
set_gstate

Koordinatenumwandlung
translate
scale
rotate
align
skew
concat
setmatrix

Pfad konstruieren
moveto
lineto
curveto
circle
arc
arcn
circular_arc
ellipse
elliptical_arc
rect
closepath

Pfad zeichnen und beschneiden
stroke
closepath_stroke
fill
fill_stroke
closepath_fill_stroke
clip
endpath

Pfadobjekte
add_path_point
draw_path
info_path
delete_path

PDI
open_pdi_document
open_pdi_callback (nur C/C++)
close_pdi_document
open_pdi_page
close_pdi_page
fit_pdi_page
info_pdi_page
process_pdi

pCOS
pcos_get_number
pcos_get_string
pcos_get_stream

Blockverarbeitung (PPS)
fill_textblock
fill_imageblock
fill_pdfblock
fill_graphicsblock

Interaktive Funktionen
create_action
add_nameddest
create_annotation
create_field
create_fieldgroup
create_bookmark
add_portfolio_folder
add_portfolio_file

Multimedia
load_3ddata
create_3dview
load_asset

Dokumentaustausch
set_info
begin_item
end_item
activate_item
begin_mc
end_mc
mc_point

B Liste aller Optionen und Schlüsselwörter

Dieser Index enthält eine alphabetische Liste aller Optionen und Schlüsselwörter mit den Funktionen, in denen sie verwendbar sind. Um zu einer Beschreibung zu springen, klicken Sie auf die Seitenzahl.

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

C Änderungen an diesem Handbuch

Datum	Änderungen
12. Dezember 2014	▸ Deutsche Übersetzung des Handbuchs für PDFlib 9.0.4
16. Mai 2014	▸ Deutsche Übersetzung des Handbuchs für PDFlib 9.0.3
16. Dezember 2013	▸ Deutsche Übersetzung des Handbuchs für PDFlib 9.0.2
13. Februar 2008	▸ Deutsche Übersetzung des Handbuchs für PDFlib 7.0.3
24. April 2007	▸ Deutsche Übersetzung des Handbuchs für PDFlib 7.0.1
1. Februar 2007	▸ Deutsche Übersetzung und Umstrukturierung des Handbuchs für PDFlib 7.0.0
13. März 2006	▸ Deutsche Übersetzung des Handbuchs für PDFlib 6.0.3 mit kleineren Erweiterungen und Korrekturen und einem neuen Abschnitt über Ruby
14. September 2005	▸ Deutsche Übersetzung des Handbuchs für PDFlib 6.0.2 mit kleineren Erweiterungen und Korrekturen
19. November 2004	▸ Deutsche Übersetzung des Handbuchs für PDFlib 6.0.1 mit kleineren Erweiterungen und Korrekturen ▸ Sprachabhängige Funktionsprototypen in Kapitel 8 ▸ Beispiele für Hypertext-Erstellung in Kapitel 3
19. August 2004	▸ Deutsche Übersetzung des Handbuchs für PDFlib 6.0.0
18. Juni 2004	▸ Erweiterungen für PDFlib 6
6. Oktober 2003	▸ Deutsche Übersetzung des Handbuchs für PDFlib 5.0.2
15. September 2003	▸ Kleinere Änderungen für PDFlib 5.0.2, Ergänzung der Blockspezifikation
30. Mai 2003	▸ Deutsche Übersetzung des Handbuchs für PDFlib 5.0.1
6. August 2002	▸ Deutsche Übersetzung des Handbuchs für PDFlib 4.0.3, Ergänzungen für .NET
14. Juni 2002	▸ Kleinere Änderungen für PDFlib 4.0.3 und Erweiterungen für die .NET-Bindung
26. Januar 2002	▸ Kleinere Änderungen für PDFlib 4.0.2 und Erweiterungen für die IBM-eServer-Edition
17. Mai 2001	▸ Kleinere Änderungen für PDFlib 4.0.1
1. April 2001	▸ Dokumentiert PDI- und andere Funktionen von PDFlib 4.0.0
5. Februar 2001	▸ Dokumentiert Template- und CMYK-Funktionen in PDFlib 3.5.0
22. Dezember 2000	▸ ColdFusion-Dokumentation und Erweiterungen für PDFlib 3.03; separate ActiveX-Edition des Handbuchs
8. August 2000	▸ Delphi-Dokumentation und kleinere Erweiterungen für PDFlib 3.02
1. Juli 2000	▸ Erweiterungen und Klarstellungen für PDFlib 3.01
20. Februar 2000	▸ Änderungen für PDFlib 3.0
2. August 1999	▸ Kleinere Änderungen und Erweiterungen für PDFlib 2.01
29. Juni 1999	▸ Eigene Abschnitte für die jeweiligen Sprachbindungen ▸ Erweiterungen für PDFlib 2.0
1. Februar 1999	▸ Kleinere Änderungen für PDFlib 1.0 (keine öffentliche Freigabe)
10. August 1998	▸ Erweiterungen für PDFlib 0.7 (nur für einen Kunden)

Datum	*Änderungen*
8. Juli 1998	► *Erste Beschreibung der PDFlib-Skriptunterstützung in PDFlib 0.6*
25. Februar 1998	► *Kleine Erweiterungen am Handbuch für PDFlib 0.5*
22. September 1997	► *Erste öffentliche Version von PDFlib 0.4 und dieses Handbuchs*

Index

Einen Index aller Optionen und Schlüsselwörter finden Sie in Anhang B, Seite 291.

Q

R

S

T

U

V

W

X

Z

PDFlib GmbH

PDFlib GmbH
Franziska-Bilek-Weg 9
D-80339 München
www.pdflib.com
Tel. +49 • 89 • 452 33 84-0
Fax +49 • 89 • 452 33 84-99

Bei Fragen können Sie die PDFlib-Mailing-Liste abonnieren
und sich deren Archiv ansehen unter groups.yahoo.com/neo/groups/pdflib/info

Vertriebsinformationen
sales@pdflib.com

Support
support@pdflib.com *(geben Sie bitte immer Ihre Lizenznummer an)*