

PDFlib GmbH 慕尼黑, 德国

[www.pdflib.com](http://www.pdflib.com)



适用于  
COM, .NET, 和 REALbasic

版权所有 © 1997–2006 PDFlib GmbH 和 Thomas Merz。保留所有权利。  
在此授予 PDFlib 用户许可，可以内部使用为目的复制本手册的印刷或数字副本。

PDFlib GmbH  
Tal 40, 80331 München, Germany  
www.pdflib.com

电话: +49 • 89 • 29 16 46 87  
传真: +49 • 89 • 29 16 46 86

如果您有任何疑问，请查看 [groups.yahoo.com/group/pdflib](http://groups.yahoo.com/group/pdflib) 上的 PDFlib 邮件列表及相关存档。

许可联系人: [sales@pdflib.com](mailto:sales@pdflib.com)  
商业 PDFlib 许可证持有人支持: [support@pdflib.com](mailto:support@pdflib.com) (请提供您的许可证号)

本出版物及其所含信息“按原样”提供；如有更改，恕不另行通知，且不应视为 PDFlib GmbH 所做的承诺。PDFlib GmbH 不对存在的任何错误和不准确性承担任何责任，不对本出版物做出任何形式（明示、默示或法定）的保证，并明确声明：不对其适销性、特定用途的实用性以及有关第三方权利的无侵权做出任何形式的保证。

PDFlib 及 PDFlib 徽标是 PDFlib GmbH 的注册商标。在此授予 PDFlib 许可证持有者在其产品文档中使用 PDFlib 名称和徽标的权利。但是，并不要求使用 PDFlib 名称和徽标。

Adobe、Acrobat 和 PostScript 是 Adobe Systems Inc. 的商标。AIX、IBM、OS/390、WebSphere、iSeries 和 zSeries 是 International Business Machines Corporation 的商标。ActiveX、Microsoft、OpenType 和 Windows 是 Microsoft Corporation 的商标。Apple、Macintosh 和 TrueType 是 Apple Computer, Inc. 的商标。Unicode 及 Unicode 徽标是 Unicode, Inc. 的商标。Unix 是 The Open Group 的商标。Java 和 Solaris 是 Sun Microsystems, Inc. 的商标。HKS 是 HKS 品牌联名的注册商标：Hostmann-Steinberg、K+E Printing Inks、Schmincke。其他公司的产品和服务名称可能是其他公司的商标或服务标志。

软件应用程序或用户文档中显示的 PANTONE® 颜色可能不符合 PANTONE 确定的标准。有关准确的颜色，请参阅最新版的 PANTONE 颜色文献。PANTONE® 及其他 Pantone, Inc. 商标规 Pantone, Inc. © Pantone, Inc. 所有，2003。  
Pantone, Inc. 是颜色数据和 / 或软件的版权所有人，并已授予 PDFlib GmbH 使用许可，但仅限于同 PDFlib 软件一起分发。除非作为执行 PDFlib 软件的一部分，否则不得将 PANTONE 颜色数据和 / 或软件复制到另一此案或存储设备中。

PDFlib 包含下列经过修改的第三方软件组件：  
ICCLib，版权所有 © 1997-2002 Graeme W. Gill  
GIF 图像解码器，版权所有 © 1990-1994 David Koblas  
PNG 图像参考库 (libpng)，版权所有 © 1998-2004 Glenn Randers-Pehrson  
Zlib 压缩库，版权所有 © 1995-2002 Jean-loup Gailly and Mark Adler  
TIFFlib 图像库，版权所有 © 1988-1997 Sam Leffler，版权所有 © 1991-1997 Silicon Graphics, Inc.  
Eric Young 编写的密码软件，版权所有 © 1995-1998 Eric Young (eay@cryptsoft.com)  
Independent JPEG Group 的 JPEG 软件，版权所有 © 1991-1998, Thomas G. Lane

PDFlib 包含 RSA Security, Inc. 的 MD5 消息摘要算法。



作者: Thomas Merz  
设计和插图: Alessio Leonardi  
质量控制 (手册): Katja Schnelle Romaus、Kurt Stützer  
质量控制 (软件): 软件开发组

# 目录

## o 应用 PDFlib 许可证密钥 9

### 1 前言 11

- 1.1 PDFlib 编程 11
- 1.2 PDFlib 6 中的主要新增功能 12
- 1.3 PDFlib 功能 13
- 1.4 不同产品中的功能的可用性 15

## 2 PDFlib 语言绑定 19

- 2.1 概述 19
- 2.2 Cobol 绑定 19
- 2.3 COM 绑定 20
  - 2.3.1 COM 绑定的工作原理 20
  - 2.3.2 安装 PDFlib COM 版本 20
  - 2.3.3 无提示安装 21
  - 2.3.4 部署方案 21
  - 2.3.5 COM 的错误处理 22
  - 2.3.6 将 PDFlib 与 Active Server Pages 一起使用 22
  - 2.3.7 将 PDFlib 与 Visual Basic 一起使用 24
  - 2.3.8 将 PDFlib 与 Windows Script Host 一起使用 25
  - 2.3.9 将 PDFlib 与 Borland Delphi 一起使用 27
  - 2.3.10 将 PDFlib COM 版本与 .NET 一起使用 28
- 2.4 C 绑定 30
- 2.5 C++ 绑定 31
- 2.6 Java 绑定 31
- 2.7 .NET 绑定 31
  - 2.7.1 安装 PDFlib .NET 版本 31
  - 2.7.2 无提示安装 31
  - 2.7.3 .NET 的错误处理 32
  - 2.7.4 将 PDFlib 与 C# 一起使用 32
  - 2.7.5 将 PDFlib 与 VB.NET 一起使用 33
  - 2.7.6 将 PDFlib 与 ASP.NET 一起使用 35
- 2.8 Perl 绑定 36
- 2.9 PHP 绑定 36
- 2.10 Python 绑定 36
- 2.11 REALbasic 绑定 36
  - 2.11.1 安装 PDFlib REALbasic 版本 36
  - 2.11.2 用 REALbasic 编写的“Hello world”示例 37
  - 2.11.3 REALbasic 的错误处理 37
- 2.12 Ruby 绑定 38
- 2.13 RPG 绑定 38

## 2.14 Tcl 绑定 38

# 3 PDFlib 编程 39

## 3.1 常规编程 39

### 3.1.1 PDFlib 程序结构和函数范围 39

### 3.1.2 参数 39

### 3.1.3 异常处理 40

### 3.1.4 PDFlib 虚拟文件系统 (PVF) 42

### 3.1.5 资源配置和文件搜索 42

### 3.1.6 大文件支持 45

## 3.2 页面说明 46

### 3.2.1 坐标系统 46

### 3.2.2 页面大小和坐标限制 48

### 3.2.3 路径 49

### 3.2.4 模板 49

## 3.3 使用颜色 50

### 3.3.1 颜色和色彩空间 50

### 3.3.2 图案和平滑着色 51

### 3.3.3 专色 51

### 3.3.4 色彩管理和 ICC 配置文件 53

## 3.4 超文本元素 56

### 3.4.1 创建超文本元素的示例 56

### 3.4.2 文本域的格式化选项 59

# 4 文本处理 63

## 4.1 字体和编码概述 63

### 4.1.1 支持的字体格式 63

### 4.1.2 编码 63

### 4.1.3 支持 Unicode 标准 64

## 4.2 字体格式详细信息 65

### 4.2.1 Windows 和 Mac 上的宿主字体支持 65

### 4.2.2 PostScript 字体 66

### 4.2.3 TrueType 和 OpenType 字体 67

### 4.2.4 用户定义的 (Type 3) 字体 68

## 4.3 字体嵌入和子集化 69

### 4.3.1 PDFlib 搜索字体的方法 69

### 4.3.2 字体嵌入 70

### 4.3.3 字体子集化 71

## 4.4 编码详细信息 72

### 4.4.1 8 位编码 72

### 4.4.2 符号字体和特定于字体的编码 75

### 4.4.3 TrueType 和 OpenType 字体的字形 ID 寻址 75

### 4.4.4 欧元符号字形 76

## 4.5 Unicode 支持 76

### 4.5.1 页面内容和超文本的 Unicode 76

### 4.5.2 内容字符串、超文本字符串和名称字符串 77

### 4.5.3 支持 Unicode 的语言中的字符串处理 78

### 4.5.4 不支持 Unicode 的语言中的字符串处理 78

### 4.5.5 字符引用 79

- 4.5.6 Unicode 兼容的字体 80
- 4.6 文本规格和文本变体 81
  - 4.6.1 字体和字符规格 81
  - 4.6.2 字距调整 82
  - 4.6.3 文本变体 83
- 4.7 中文、日文和韩文文本 84
  - 4.7.1 Acrobat 和 PDF 中支持的 CJK 84
  - 4.7.2 标准 CJK 字体和 CMap 84
  - 4.7.3 自定义 CJK 字体 87
- 4.8 放置并调整单行文本 88
  - 4.8.1 简单的文本放置 89
  - 4.8.2 在文本框中放置文本 89
  - 4.8.3 对齐文本 90
- 4.9 多行 textflow 91
  - 4.9.1 在限定框中放置 textflow 92
  - 4.9.2 段落格式设置选项 93
  - 4.9.3 内嵌选项列表和宏 94
  - 4.9.4 制表位 96
  - 4.9.5 编号列表 96
  - 4.9.6 控制字符、字符映射和符号字体 97
  - 4.9.7 断字 99
  - 4.9.8 控制换行算法 100
  - 4.9.9 使用 textflow 格式化 CJK 文本 103

## 5 导入和放置对象 105

- 5.1 导入光栅图像 105
  - 5.1.1 基本图像处理 105
  - 5.1.2 受支持的图像文件格式 106
  - 5.1.3 图像蒙版和透明度 108
  - 5.1.4 图像着色 109
  - 5.1.5 多页图像文件 110
  - 5.1.6 OPI 支持 110
- 5.2 使用 PDI (PDF 导入库) 生成 PDF 页 110
  - 5.2.1 PDI 功能和应用程序 111
  - 5.2.2 与 PDFlib 一起使用 PDI 函数 111
  - 5.2.3 可接受的 PDF 文档 112
- 5.3 放置图像和导入的 PDF 页 113
  - 5.3.1 缩放、方向和旋转 113
  - 5.3.2 调整页面尺寸 115

## 6 数据变量和块 119

- 6.1 安装 PDFlib 块增效工具 119
- 6.2 PDFlib 块概念的概述 120
  - 6.2.1 文档设计和程序代码的完整分离 120
  - 6.2.2 块属性 121
  - 6.2.3 为什么不使用 PDF 表单域? 122
- 6.3 建立 PDFlib 块 122
  - 6.3.1 利用 PDFlib 块增效工具交互式建立块 122

6.3.2	编辑块属性	125
6.3.3	页和文件之间的块拷贝	125
6.3.4	将 PDF 表单域转换为 PDFlib 块	126
6.4	用于自动化处理的标准属性	128
6.4.1	一般属性	129
6.4.2	Text 属性	130
6.4.3	Image 属性	133
6.4.4	PDF 属性	134
6.4.5	自定义属性	134
6.5	询问块名及其属性	135
6.6	PDFlib 块规格	136
6.6.1	PDFlib 块的 PDF 对象结构	136
6.6.2	利用 pdfmarks 生成 PDFlib 块	138
7	生成各种风格的 PDF	141
7.1	Acrobat 和 PDF 版本	141
7.2	加密 PDF	142
7.2.1	PDF 安全性的优缺点	142
7.2.2	用 PDFlib 保护文档	143
7.3	网页优化的（线性化的）PDF	144
7.4	PDF/X	145
7.4.1	PDF/X 系列标准	145
7.4.2	生成符合 PDF/X 标准的输出	145
7.4.3	用 PDI 导入 PDF/X 文档	148
7.5	标签 PDF	148
7.5.1	用 PDFlib 生成标签 PDF	149
7.5.2	用直接 Text Output 和 Textflows 创建标签 PDF	150
7.5.3	启动复杂布局的项目	151
7.5.4	在 Acrobat 中使用标签 PDF	154
8	用于 PDFlib、PDI 和 PPS 的 API 参考	157
8.1	数据类型	157
8.1.1	PDFlib 数据类型	157
8.1.2	选项列表	157
8.2	常规函数	159
8.2.1	设置	159
8.2.2	文档和页面	160
8.2.3	参数处理	168
8.2.4	PDFlib 虚拟文件系统 (PVF) 函数	169
8.2.5	异常处理	170
8.2.6	实用程序函数	171
8.3	文本函数	171
8.3.1	字体处理	171
8.3.2	用户定义 (Type 3) 字体	174
8.3.3	编码定义	175
8.3.4	简单文本输出	176
8.3.5	使用文本流的多行文本	181

8.4 图形函数 189

8.4.1 图形状态函数 189

8.4.2 保存和恢复图形状态 192

8.4.3 坐标系统转换函数 193

8.4.4 显式图形状态 194

8.4.5 路径构建 195

8.4.6 路径绘制和剪切 198

8.4.7 图层参数 199

8.5 颜色函数 202

8.5.1 设置颜色和色彩空间 202

8.5.2 图案和着色 205

8.6 图像和模板函数 207

8.6.1 图像 207

8.6.2 模板 212

8.6.3 缩览图 213

8.7 PDF 导入函数 (PDI) 213

8.7.1 文档和页面 214

8.7.2 其他 PDI 处理 216

8.7.3 PDI 参数处理 217

8.8 块填充函数 (PPS) 219

8.9 超文本函数 222

8.9.1 动作 222

8.9.2 命名目标 224

8.9.3 注释 225

8.9.4 表单域 229

8.9.5 书签 233

8.9.6 文档信息域 234

8.9.7 不常用的超文本参数和函数 235

8.10 标签 PDF 的结构函数 236

A 相关文献 241

B PDFlib 快速参考 243

C 修订历史记录 248

索引 249





# o 应用 PDFlib 许可证密钥

无论您是否持有 PDFlib 的商业许可证，由 PDFlib GmbH 提供的所有二进制版本的 PDFlib、PDFlib+PDI 和 PPS 都可以作为功能完整的评估版本使用。然而，未经许可的版本将在所有生成的页面上显示 [www.pdfliib.com](http://www.pdfliib.com) 演示图章（提示标记）。那些真正对 PDFlib 许可证感兴趣并希望消除评估期间或原型演示的提示标记的公司，可以将带有简要说明的有关公司和项目的详细信息提交到 [sales@pdfliib.com](mailto:sales@pdfliib.com)，以便申请临时许可证密钥（我们保留拒绝评估密钥的权利，例如匿名请求）。

在您购买许可证密钥之后，必须应用该密钥才能消除演示图章。您可以准备一个密钥文件，在运行时调用 PDFlib 函数从文件中获取许可证密钥；或在 Windows 中使用注册表。若使用 Windows 安装程序，您可以在产品安装时键入一个许可证密钥。

在运行中提供许可证密钥 ▶在脚本或程序中添加一行代码以便在运行时设置许可证密钥。

在实例化 PDFlib 对象之后，紧接着必须设置 *license* 参数。具体的句法将取决于您使用的编程语言：COM/VBScript 和 REALbasic：

```
oPDF.set_parameter "license", "...your licnese key..."
```

▶ .NET/C#:

```
p.set_parameter ("license", "...your licnese key...");
```

使用许可证文件 ▶按照下面的格式在文本文件中输入许可证密钥（您可以使用所有 PDFlib 分发中包含的许可证文件模板 *licensekeys.txt*）：

```
PDFlib license file 1.0
# Licensing information for PDFlib GmbH products
PDFlib 6.0.3 ...your license key...
```

许可证文件可以包含多个 PDFlib GmbH 产品的许可证密钥，每一行对应于一个许可证密钥。接下来，您必须通过以下两种方式之一向 PDFlib 通知有关许可证文件的情况：

在实例化 PDFlib 对象之后，紧接着按如下方式设置 *licensefile* 参数：

▶ COM/VBScript 和 REALbasic:

```
oPDF.set_parameter "licensefile", "/path/to/licensekeys.txt"
```

▶ .NET/C#:

```
p.set_parameter("licensefile", "/path/to/licensekeys.txt");
```

或者，您可以设置环境变量 *PDFLIBLICENSEFILE* 指向相应的许可证文件。在 Windows 使用系统控制面板。

Windows 注册表 在 Windows 中您可以在下列注册表条目中输入许可证文件名：

HKLM\Software\PDFlib\PDFLIBLICENSEFILE

请注意，虽然 PDFlib、PDFlib+PDI 和 PDFlib 个性化设置服务器 (PPS) 是在一个包装中发送的，但是它们是需要不同许可证密钥的不同产品。虽然 PDFlib+PDI 许可证密钥也对 PDFlib 有效，但是反过来则行不通，PPS 许可证密钥对 PDFlib+PDI 和 PDFlib 有效。所有许可证密钥都依赖于平台，只能在所购买的密钥针对的平台上使用。

**累积各个 CPU 密钥** 若您分为多次购买了多个 CPU 许可证（而不是一次性购买了所有这些 CPU 许可证），则可以通过在许可文件中相继输入这些密钥来累积所有密钥。可以针对各个许可证密钥多次调用 `set_parameter()` 函数。但是，Windows 注册表和 Windows 安装程序不能够用来累积许可证密钥。

**更新和升级** 若您已购买一个更新（从产品的旧版本到同一产品的新版本的更改）或升级（从 PDFlib 到 PDFlib+PDI 或 PPS 的更改，或者，从 PDFlib+PDI 到 PPS 的更改），则必须应用您所接收的用于更新或升级的新的许可证密钥。不得再使用先前产品的旧的许可证密钥。

**评估未经许可的功能** 您可以使用未应用任何许可证密钥的软件完全评估所有功能。不过，一旦您应用了特定产品的有效许可证密钥，则将不再可以使用更高类别的功能。例如，若您已安装有效的 PDFlib 许可证密钥，则 PDI 功能将不再可以用于测试。同样，在安装 PDFlib+PDI 许可证密钥后，PPS 功能（块函数）将不再可用。

若您已安装某个产品的许可证密钥，则可以用 o（零）虚拟许可证密钥替换该密钥来启用更高的产品类别的功能以进行评估。这将启用先前已禁用的功能，并重新激活所有页的演示图章。

**许可选项** 对于在一个或多个服务器上使用 PDFlib，以及将自己的产品与 PDFlib 一起重新分发，可以使用不同的许可选项。我们还提供支持 and 源代码约定。可以在 PDFlib 分发中找到许可证详细信息和 PDFlib 购买订单。若您有兴趣获得商业 PDFlib 许可证，或您有任何问题，请与我们联系：

PDFlib GmbH, Licensing Department

Tal 40, 80331 München, Germany

[www.pdflib.com](http://www.pdflib.com), 电话 +49 · 89 · 29 16 46 87, 传真 +49 · 89 · 29 16 46 86

许可证联系人: [sales@pdflib.com](mailto:sales@pdflib.com)

对 PDFlib 被许可方的支持: [support@pdflib.com](mailto:support@pdflib.com)

# 1 前言

## 1.1 PDFlib 编程

什么是 **PDFlib**? PDFlib 是一个库文件,它允许您按照 Adobe 的可移植文档格式 (PDF) 生成文件。PDFlib 充当您自己的程序的后端。当您(程序员)负责抽取要处理的数据时,PDFlib 承担生成 PDF 代码的任务并以图形方式表示数据。尽管您仍必须设置文本和图表对象的格式并安排版式,但 PDFlib 可让您不用再关注 PDF 的内部细节问题。我们的二进制程序包在一个库中提供了不同的产品:

- ▶ PDFlib 包含创建 PDF 输出(包括文本、矢量图形和图像以及超文本元素)所必需的所有函数。
- ▶ PDFlib+PDI 包括所有 PDFlib 函数,以及用于在已生成的输出中包括现有 PDF 文档页的 PDF 输入库(PDI)。
- ▶ PDFlib Personalization Server (PPS) 包括 PDFlib+PDI 以及用于自动填充 PDFlib 块的附加函数。块是页上的占位矩形,可以用文本、图像或 PDF 页进行填充。可以使用适用于 Adobe Acrobat (Mac 或 Windows) 的 PDFlib 块增效工具交互式创建块,并使用 PPS 自动进行填充。该增效工具包含在 PPS 中。

如何使用 **PDFlib**? PDFlib 可在多种平台上使用,包括 Unix、Windows、Mac 和基于 EBCDIC 的系统(如 IBM eServer iSeries 和 zSeries)。虽然 PDFlib 自身是用 C 语言编写的,但也可以从多种其他语言或编程环境访问它。这种情况称为“语言绑定”。这些语言绑定覆盖了所有当前 Web 和独立应用程序环境。应用程序编程接口(API)简单易学,且对于所有的绑定都是相同的。目前,支持以下的语言绑定:

- ▶ COM (用于 Visual Basic、带有 VBScript 或 JScript 的 Active Server Page、Borland Delphi、Windows Script Host 和其他环境)
- ▶ ANSI C
- ▶ ANSI C++
- ▶ Cobol (IBM eServer zSeries)
- ▶ Java (包括 servlet)
- ▶ .NET (用于 C#、VB.NET、ASP.NET 和其他环境)
- ▶ PHP 超文本处理器
- ▶ Perl
- ▶ Python
- ▶ REALbasic
- ▶ RPG (IBM eServer iSeries)
- ▶ Ruby
- ▶ Tcl

使用 **PDFlib** 可以做什么? PDFlib 的主要目标是在您自己的软件内部或在万维网上动态创建 PDF。类似于在 Web 服务器上动态创建 HTML 页,您可以使用 PDFlib 程序动态创建 PDF 以反映用户输入或一些其他的动态数据(例如从 Web 服务器的数据库中检索的数据)。PDFlib 方法具有下面几个优点:

- ▶ 可以在生成数据的应用程序中直接集成 PDFlib,消除复杂的创建路径 application-PostScript-Acrobat Distiller-PDF。
- ▶ 这一直接简单的方法意味着,PDFlib 是生成 PDF 的最快方法,完全适用于 Web。
- ▶ PDFlib 的多线程安全及其可靠内存和错误处理可支持实现高性能的服务器应用程序。

► PDFlib 可用于多种操作系统和开发环境。

**有关使用 PDFlib 的要求** PDFlib 可让用户不用费心了解 PDF 规范即可生成 PDF。尽管 PDFlib 会尽力向用户隐藏 PDF 的技术细节，但是对 PDF 有一个大致了解还是有用的。为了更好地使用 PDFlib，应用程序程序员最好应熟悉 PostScript 的基本图形模型（以及 PDF）。不过，对于已使用过用于屏幕显示或打印的任何图形 API 的有相当经验的应用程序程序员，掌握本手册中描述的 PDFlib API 应无困难。

**本手册的内容** 本手册描述由 PDFlib 提供的 API。本手册并不试图解释 Acrobat 功能。有关进一步的参考，请参阅本手册末尾所引用的 Acrobat 产品文献和资料。PDFlib 分发包含有关调用 PDFlib 功能的附加示例。

**实例编码** T 本手册包含了大量编码片断。虽然 PDFlib 支持多种编程语言，本手册只提供 Visual Basic 的实例编码。使用其他编程语言的用户需将这些实例编码转换为所用的语言句法。

## 1.2 PDFlib 6 中的主要新增功能

下面的列表讨论 PDFlib 6 中最重要的新增功能或改进功能。

**编程改进** 先前版本中的许多限制已经被取消。例如，可以按照任意顺序创建页，可以在现有页之间插入新页，可以在以后将更多内容添加到现有页中。

**图层** 虽然 PDF 的图层功能（在 Acrobat 6 中引入）对 CAD 和工程设计应用程序很重要，但是它也可以用于效果出色的交互文档、多语言文档等。PDFlib 支持在 PDF 1.5 中可用的所有图层控制功能，包括在 Acrobat 中无法访问的各种控制。

**Unicode** PDFlib 6 允许在所有相关区域（例如文件名称、页内容、超文本、表域等）中使用 Unicode 字符串，从而改善了对 Unicode 标准的支持。这对于欧洲和北美洲之外的用户特别重要。

**文本格式** 新的 textflow 格式化程序提供强大而易于使用的功能，可以通过各种选项格式化文本。Unicode 文本、锯齿形或分散对齐形文本、任意字体更改、多行正文文本或发票中的大型表 — 新的 textflow 功能可以处理所有常见的格式化任务。

**图像处理** TIFF 图像处理得到了扩展，可以涵盖先前不受支持的 TIFF 风格，例如压缩的 JPEG TIFF 或 Lab 和 YCbCr 色彩空间。由于 PDF 1.5 支持 16 位颜色深度，因此，每色彩分量采用 16 位表示的 TIFF 和 PNG 图像现在可以在 PDF 中转换为 16 位色彩。PDFlib 6.0.2 添加了对 JPEG2000 图像的支持。

**标签 PDF** 根据美国康复法案第 508 节 (section 508) 和其他国家的相应准则，标签 PDF 是残疾人士访问 PDF 的关键。PDFlib 是第一个程序库从总体上支持标记 PDF 的生成。使用这一新功能，用户可以非常容易的从动态数据里生成标签 PDF。PDFlib 生成的文档将支持所有 Acrobat 标签 PDF 的功能如：页重组、朗读和高质量的输出 RTF、HTML、XML 等不同的文件格式。textflow 格式器与这一功能的组合能迅速的生成大量的标签 PDF。第一次，在网络服务器上动态生成的 PDF 文档可以满足上面提到的残疾人士可访问性准则。

**PDF/X 印前标准** PDFlib 6 是市场上第一款、根据有关印前的最新的 PDF/X 2003 版本标准 (PDF/X-1a:2003、PDF/X-2:2003 和 PDF/X-3:2003) 支持生成和处理 PDF 文档的软件。PDF/X

对于在印前行业中的文件交换起到了重要的作用。为了在图片艺术行业中实现可靠的数据交换，全世界越来越多的出版商采用 PDF/X 作为数据交换的标准。新的 2003 版本更新、增强并统一了 PDF/X 标准系列。

开放印前作业界面 (OPI) 图片艺术行业中的一些工作流程仍依赖于 PostScript 时期的 OPI 标准，并使用嵌入在 PDF 文档中的 OPI 信息。PDFlib 6 通过提供用于将 OPI 信息添加到导入的图像的选项，可以支持这一标准。

线性 PDF PDFlib 6 可生成线性 PDF，也称作“网页最优化 PDF”。当用户在网上浏览具有这一特征的 PDF 时，它将逐页下载（又称 byteserving）从而显著提高用户的使用经验。

用于可变数据处理的 PDFlib 块 为建立 PDF 模版而设计的 PDFlib 块用户接口被扩充并流线性化。PDFlib 块现在能用新增的 textflow 格式器填充多行文本。受益于此，PDFlib Personalization Server (PPS) 将不再局限于原有的简单的信件合并，它可以应用于更复杂的排版和文本格式。

表单 PDF 所有的表域类型可以通过 Javascript 或其它动作 (actions) 生成或加强。因此 PDF 表单可根据用户输出和数据库信息由 PDFlib 动态生成。

超文本 PDFlib 的超文本功能经过了扩展，可以完全支持用于书签、操作和注释的所有 PDF 选项。可以创建页标签以将符号名称或罗马数字附加到页上，如 i, ii, iii... 或 A-1, A-2 等。

REALbasic 作为所支持的一系列编程环境中的新成员，PDFlib 6 引入了在 Mac 和 Windows 上对于 REALbasic 的一种新的语言绑定。REALbasic 是一种用于开发多平台应用程序的语言。用于 REALbasic 的 PDFlib 6 可顺畅地集成到 RB 的对象模块中，支持 Unicode 字符串，并可让开发人员从 REALbasic 内部访问所有的 PDFlib 功能。

### 1.3 PDFlib 功能

表 1.1 列出了用于生成和导入 PDF 的主要 PDFlib 功能。PDFlib 6 中新功能或改进功能已被标记出来。

表 1.1 有关 PDFlib、PDFlib+PDI 和 PDFlib Personalization Server (PPS) 的功能列表

主题	功能
PDF 输出	直接保存在内存中（用于 Web 服务器）或保存在磁盘文件上的任意长度的 PDF 文档
	文本、矢量图形、图像数据和文件附件的压缩
	挂起 / 继续 <sup>1</sup> 和插入页 <sup>1</sup> 等支持不按顺序创建页的功能
PDF 风格	PDF 1.3、1.4、1.5 和 1.6（Acrobat 4、5、6 和 7） 线性 PDF（网页最优化 PDF），用于 Web 逐页下载（又称 byteserving） <sup>1</sup>
PDF 输入	从现有 PDF 文档（仅 PDFlib+PDI 和 PPS）导入页
块	使用用于文本、图像和 PDF 数据（仅 PPS）的 PDFlib 块个性化设置 PDF 使用 Acrobat 的 PDFlib 块增效工具以创建 PDFlib 块（仅 PPS），重新设计的用户界面 <sup>1</sup>
图形	常见矢量图形基元：直线、曲线、弧线、矩形等。 阴影平滑（色彩混和）、图案填充和描边 利用模板有效重用文本或矢量图形 用于文本镂空、压印等的图形状态参数 透明度（不透明度）和混和模式 图层 <sup>1</sup> ：可以选择性地启用或禁用的可选页内容

表 1.1 有关 PDFlib、PDFlib+PDI 和 PDFlib Personalization Server (PPS) 的功能列表（续）

主题	功能
字体	TrueType（ <i>ttf</i> 和 <i>ttc</i> ）和 PostScript Type 1 字体（ <i>pfm</i> 和 <i>pfa</i> 以及 Mac 上的 <i>lwf</i> ）
	带有 PostScript 或 TrueType 轮廓的 OpenType 字体（ <i>ttf</i> 和 <i>otf</i> ）
	AFM 和 PFM PostScript 字库量度文件
	字体嵌入
	直接使用已在 Windows 或 Mac 主机系统上安装的字体的 TrueType 和 OpenType 字体的子集
文本输出	用于点阵字体或自定义徽标的用户自定义的 (Type 3) 字体
	使用不同字体显示的文本输出：下划线文本、上划线文本和删除线文本
	用于 PostScript 字体、TrueType 字体和 OpenType 字体的字距调整
	用于高级拣字程序的 TrueType 和 OpenType 字形 ID 寻址
	用于标准 CJK 字体的比例宽度
国际化	用于页内容、超文本 <sup>1</sup> 和文件名称 <sup>1</sup> 的 Unicode；UTF-8 和 UCS-2 格式，little-endian 和 big-endian
	在 COM、Java、.NET、REALbasic、Tcl 中完全集成对 Unicode 字符串的处理
	支持各种编码（国际标准和供应商特定的代码页）
	从系统（Windows、IBM eServer iSeries 和 zSeries）中获取代码页
	标准的 CJK 字体和对中文、日文和韩文文本的 CMap 支持
	带有 Unicode 编码的 TrueType 和 OpenType 格式的自定义 CJK 字体
	将 Unicode 信息嵌入 PDF 中以用于 Acrobat 中的正确文本提取
图像	嵌入 BMP、GIF、PNG、TIFF <sup>1</sup> 、JPEG、JPEG2000 和 CCITT 光栅图像
	自动检测图像文件格式（文件格式探查）
	透明（蒙版）的图像，包括软蒙版
	图像蒙版（应用了单一色彩的透明图像）
	使用专色为图像着色
	图像插值处理（使低分辨率图像平滑）
色彩	灰度、RGB、CMYK、CIE L*a*b* 色彩
	内置 PANTONE® 和 HKS® 专色表
	用户自定义的专色
色彩 <i>fb,il</i>	带 ICC 配置文件 (ICC profile) 的基于 ICC 的色彩：接受图像中已嵌入的配置文件或对图像应用外部描述文件
	文本、图形和光栅图像的渲染方法
	默认的灰度、RGB 和 CMYK 色彩空间以便重新映射设备相关的色彩
	生成符合 PDF/X-1、PDF/X-1a、PDF/X-2 <sup>1</sup> 和 PDF/X-3 标准的输出，包括 2003 风格 <sup>1</sup>
印前	嵌入 ICC 配置文件 (ICC profile) 所要输出的设备的输出条件或引用标准输出内容
	从导入的 PDF 文档（仅 PDFlib+PDI 和 PPS）复制输出内容
	为导入的图像创建 OPI 1.3 和 OPI 2.0 信息 <sup>1</sup>
	分色信息 (PlateColor) <sup>1</sup>
格式设置	textflow 格式设置 <sup>1</sup> ：使用断字、字体和色彩更改、各种对齐方法、控制命令，将任意数量的文本格式化到一个或多个矩形区域内
	文本行位置和格式设置
	灵活多变的图像位置和格式设置
安全	使用 40 位或 128 位加密生成输出
	使用权限设置生成输出

表 1.1 有关 *PDFlib*、*PDFlib+PDI* 和 *PDFlib Personalization Server (PPS)* 的功能列表（续）

主题	功能
	导入加密的文档（要求许可口令；仅 <i>PDFlib+PDI</i> 和 <i>PPS</i> ）
超文本	使用所有字段选项和 <i>JavaScript</i> <sup>1</sup> 创建表域 <sup>1</sup>
	创建用于书签、注释、页面打开 / 关闭和其他事件的动作 <sup>1</sup>
	使用各种选项和控件创建书签 <sup>1</sup>
	支持页面转换效果，例如渐变和马赛克
	创建所有 <i>PDF</i> 注释类型 <sup>1</sup> ，例如 <i>PDF</i> 链接、启动链接（其他文档类型）、 <i>Web</i> 链接
	文档信息：标准字段（标题、主题、作者、关键字）以及没有数量限制的用户自定义的信息字段
	为链接、书签和文档打开动作的指定的目标
	浏览器首选项（隐藏菜单栏等） <sup>1</sup>
	创建页标签（页的符号名称） <sup>1</sup>
标签 <i>PDF</i>	创建标签 <i>PDF</i> <sup>1</sup> ，及辅助功能、页重排和改进内容重规划的结构信息
	轻松地格式化用于标签 <i>PDF</i> 的大量文本 <sup>1</sup>
编程	支持 <i>Cobol</i> 、 <i>COM</i> 、 <i>C</i> 、 <i>C++</i> 、 <i>Java</i> 、 <i>.NET</i> 、 <i>Perl</i> 、 <i>PHP</i> <sup>1</sup> 、 <i>Python</i> 、 <i>REALbasic</i> <sup>1</sup> 、 <i>RPG</i> 、 <i>Ruby</i> 和 <i>Tcl</i> 的语言绑定
	支持多线程服务器应用程序中的多线程安全和可靠部署
	支持在内存中提供数据的虚拟文件系统，例如数据库中的映像

1. *PDFlib 6* 中的新增功能或显著改进

1.4 不同产品中的功能的可用性

表 1.2 详述了不同的产品中的功能的可用性。

表 1.2 不同产品中的功能的可用性

功能	API 函数和参数	<i>PDFlib Lite</i> (开放源码)	<i>PDFlib</i>	<i>PDFlib+PDI</i>	<i>PDFlib Personalization Server (PPS)</i>
基本 <i>PDF</i> 生成	(除以下列选项之外的全部)	X	X	X	X
语言绑定	<i>C</i> 、 <i>C++</i> 、 <i>Java</i> 、 <i>Perl</i> 、 <i>Tcl</i> 、 <i>PHP</i> 、 <i>Python</i> 、 <i>Ruby</i>	X	X	X	X
语言绑定	<i>Cobol</i> 、 <i>COM</i> 、 <i>.NET</i> 、 <i>REALbasic</i> 、 <i>RPG</i>	–	X	X	X
支持 <i>EBCDIC</i> 系统		–	X	X	X
口令保护和权限设置	带有 <i>userpassword</i> 、 <i>masterpassword</i> 、 <i>permissions</i> 选项的 <i>begin_document()</i>	–	X	X	X
线性 <i>PDF</i>	带有 <i>linearize</i> 选项的 <i>begin_document()</i>	–	X	X	X
字体子集	带有 <i>subsetting</i> 选项的 <i>load_font()</i>	–	X	X	X
<i> Kerning</i>	带有 <i>kerning</i> 选项的 <i>load_font()</i>	–	X	X	X
访问 <i>Mac</i> 和 <i>Windows</i> 宿主字体	<i>load_font()</i>	–	X	X	X
访问 <i>Windows</i> 、 <i>iSeries</i> 、 <i>zSeries</i> 上的系统编码	<i>load_font()</i>	–	X	X	X

表 1.2 不同产品中的功能的可用性（续）

功能	API 函数和参数	PDFlib Lite (开放源码)	PDFlib	PDFlib+PDI	PDFlib Personalization Server (PPS)
Unicode 编码和 ToUnicode CMaps	带有 <code>encoding = unicode</code> 、 <code>autocidfont</code> 和 <code>unicodemap</code> 参数的 <code>load_font()</code>	-	X	X	X
数字和字符实体引用	<code>fit_textline()</code> 中的 <code>charref</code> 选项， <code>charref</code> 参数	-	X	X	X
带 Unicode CMap 的标准 CJK 字体的比例字形宽度	带有 <code>UCS2-compatible CMap</code> 的 <code>load_font()</code>	-	X	X	X
字形 ID 寻址	带有 <code>encoding = glyphid</code> 的 <code>load_font()</code>	-	X	X	X
基于 PostScript OpenType 字体的扩展编码	<code>load_font()</code>	-	X	X	X
Textflow	<code>create_textflow()</code> 、 <code>delete_textflow()</code> 、 <code>fit_textflow()</code> 、 <code>info_textflow()</code>	-	X	X	X
专色	<code>makespotcolor()</code>	-	X	X	X
分色	带有 <code>separationinfo</code> 选项的 <code>begin_page_ext()</code>	-	X	X	X
表域	<code>create_field()</code> 、 <code>create_fieldgroup()</code> 、 带有 <code>type=SetOCGState</code> 的 <code>create_action()</code>	-	X	X	X
JavaScript 动作	带有 <code>type=JavaScript</code> 的 <code>create_action()</code>	-	X	X	X
图层	<code>define_layer()</code> 、 <code>begin_layer()</code> 、 <code>end_layer()</code> 、 <code>set_layer_dependency()</code> 、带有 <code>type=SetOCGState</code> 的 <code>create_action()</code>	-	X	X	X
标签 PDF	<code>begin_item()</code> 、 <code>end_item()</code> 、 <code>activate_item()</code> 、带有 <code>tagged</code> 和 <code>lang</code> 选项的 <code>begin_document()</code>	-	X	X	X
JPEG2000	带有 <code>imagetype=jpeg2000</code> 的 <code>load_image()</code>	-	X	X	X
支持 PDF/X	<code>process_pdi()</code> 、 带 <code>pdfx</code> 选项的 <code>begin_document()</code>	-	X	X	X
支持 ICC 配置文件 (ICC profile)	<code>load_iccprofile()</code> 、带有 <code>iccbasedgray/rgb/cmyk</code> 的 <code>setcolor()</code> 、带有 <code>honoriccprofile</code> 选项、 <code>honoriccprofile</code> 参数的 <code>load_image()</code> 、带有 <code>defaultgray/rgb/cmyk</code> 选项的 <code>begin/end_page_ext()</code>	-	X	X	X
CIE L*a*b* 色彩空间	带有 <code>type = lab</code> 的 <code>setcolor()</code> ； <code>Lab TIFF</code> 图像	-	X	X	X
支持开放印前作业界面 (OPI)	带有 <code>OPI-1.3/OPI-2.0</code> 选项的 <code>load_image()</code>	-	X	X	X
PDF 导入 (PDI)	<code>open_pdi()</code> 、 <code>open_pdi_callback()</code> 、 <code>open_pdi_page()</code> 、 <code>fit_pdi_page()</code> 、 <code>process_pdi()</code>	-	-	X	X
查询现有 PDF 的信息	<code>get_pdi_value()</code> 、 <code>get_pdi_parameter()</code>	-	-	X	X
使用块进行变量数据处理和 个性化设置	<code>fill_textblock()</code> 、 <code>fill_imageblock()</code> 、 <code>fill_pdfblock()</code>	-	-	-	X



表 1.2 不同产品中的功能的可用性（续）

功能	API 函数和参数	PDFlib Lite (开放源码)	PDFlib	PDFlib+PDI	PDFlib Personalization Server (PPS)
查询标准和自定义块属性	<code>get_pdi_value()</code> 、带有 <code>vdp/Blocks</code> 密钥的 <code>get_pdi_parameter()</code>	-	-	-	X
用于 Acrobat 的 PDFlib 块 增效工具	交互式创建用于 PPS 的 PDFlib 块	-	-	-	X



# 2 PDFlib 语言绑定

## 2.1 概述

可用性和平台 所有 PDFlib 功能在所有平台上和所有语言绑定（有少数几种例外情况已在本手册中注明）中都可用。表 2.1 列出了我们用于测试的语言 / 平台组合。

表 2.1 测试所使用的语言 and 平台组合

语言	Unix (Linux、Solaris、HP-UX、Mac OS X、AIX、IRIX a.o.)	Windows NT4SP2 或以上平台	IBM eServer iSeries 和 zSeries
Cobol	–	–	ILE Cobol
COM	–	ASP (PWS、IIS 4、5、6) WSH (VBScript 5、JScript 5) Visual Basic 6.0, Borland Delphi 5 – 7	–
ISO/ANSI C	gcc 3/4、HP C、IBM C 6、Sun Workshop 6 和其他 ISO C 编译器	Microsoft Visual C++ 6、VS .NET Metrowerks CodeWarrior 8 Borland C++ Builder 6	IBM c89 SAS C for MVS
ISO C++	gcc 3/4 和其他 ISO C++ 编译器	Microsoft Visual C++ 6、VS .NET Metrowerks CodeWarrior 8	IBM c89
Java	JDK 1.1.8、1.2.2、1.3、1.4、1.5	Sun JDK 1.1.8、1.2.2、1.3、1.4、1.5 ColdFusion MX	JDK 1.3.1
.NET	–	.NET Framework 1.1: C#、VB.NET、ASP.NET 等 (也测试了 Visual Studio 2005 Beta 2 与 .NET Framework 2.0)	–
Perl	Perl 5.6 – 5.8	Perl 5.6 – 5.8	–
PHP	PHP 4.3.X、4.4.X、5.0.X	PHP 4.3.X、4.4.X、5.0.X	–
Python	Python 1.6, 2.0 – 2.3	Python 1.6, 2.0 – 2.3	–
REALbasic	适用于 Mac OS Classic、Mac OS X 和 Windows 的 REALbasic 5.5、2005 或更高版本		–
RPG	–	–	ILE RPG
Ruby	Ruby 1.8	Ruby 1.8	–
Tcl	Tcl 8.3.2 和 8.4.4	Tcl 8.3.2 和 8.4.4	–

嵌入系统上的 PDFlib 应当指出，在嵌入系统上也可以使用 PDFlib，并且 PDFlib 已被移植到 Windows CE、QNX 和 EPOC 环境以及自定义嵌入系统中。为了在受限制的环境下使用，可以对某些功能进行配置以减少 PDFlib 的资源需求。若您对详细信息感兴趣，请通过 [sales@pdflib.com](mailto:sales@pdflib.com) 与我们联系。

## 2.2 Cobol 绑定

(只有通用版的 PDFlib 手册中包括此小节。)

## 2.3 COM 绑定

### 2.3.1 COM 绑定的工作原理

COM (组件对象模型)<sup>1</sup>是一种用于互操作的软件组件的独立于语言的标准。PDFlib 的 COM 实现方式是一个基于 PDFlib 内核构建的 DLL。此包装 DLL 调用 PDFlib 内核函数并负责与基础 COM 机制、注册表和类型库问题以及 COM 异常处理通信。从技术上来说, PDFlib COM 包装具有以下特点 (即使您对所有这些术语不熟悉也不用担心, 因为使用 PDFlib 并不需要掌握这些知识):

- ▶ PDFlib 将用作一个 Win32 进程内 COM 服务器组件 (也称作 “自动化服务器”), 此组件没有任何用户界面。
- ▶ PDFlib 是一个 “多线程” 组件, 也就是说, 它既可以视为一个单元线程组件, 也可以视为一个自由线程组件。另外, PDFlib 聚合了一个自由线程的封送处理程序。简单地说, 客户端可以直接使用 PDFlib 对象提高性能 (而不是通过代理 / 存根对)。
- ▶ PDFlib 二进制 *pdflib\_com.dll* 是一种带有类型库的自行注册的 DLL。
- ▶ PDFlib 是无状态的, 也就是说, 使用方法参数而不使用属性。
- ▶ PDFlib 的双重接口支持早期绑定和后期绑定。
- ▶ PDFlib 支持详细错误信息。

注: PDFlib 无法识别 *MTS (Microsoft Transaction Server)*, 尽管可以使用 *MTS* 对其进行部署。

### 2.3.2 安装 PDFlib COM 版本

PDFlib 可以在所有支持 COM 组件的环境中部署。我们将在一些环境中演示我们的示例:

- ▶ Visual Basic
- ▶ 使用 JScript 的 Active Server Pages (ASP)
- ▶ 使用 VBScript 的 Windows Script Host (WSH)
- ▶ Borland Delphi

Active Server Pages 和 Windows Script Host 都支持 JScript 和 VBScript。不过, 因为脚本几乎都一样, 所以在这里我们不演示所有组合。另外, 可以使用许多其他识别 COM 的开发环境, 例如: Microsoft Visual C++、Borland C++ Builder 和 PowerBuilder。PDFlib 也可用于 Visual Basic for Applications (VBA)。

注: 尽管 PDFlib COM 版本可以与 *ColdFusion 5* 一起正常工作, 但是我们还是建议将 *PDFlib Java* 版本和 *ColdFusion MX* 一起使用。

有关安装 PDFlib 的要求 安装 PDFlib 的过程很简单。请注意以下事项:

- ▶ 若在 NTFS 分区上安装, 则所有 PDFlib 用户必须具有对安装目录的读取权限以及对 ...\\PDFlib 6.0.2\\bin\\pdflib\_com.dll 的执行权限。
- ▶ 安装程序必须具有对系统注册表的写入权限。管理员或高级用户组权限通常就足够了。

1. 有关 COM 的更多信息, 请参见 [www.microsoft.com/com](http://www.microsoft.com/com)

**PDFlib MSI 安装程序** PDLlib 可以用作一个 MSI 包（Microsoft 的 Windows Installer），它支持安装、修复和卸载功能。若要从 MSI 包安装 PDLlib，只需双击 *.msi* 文件，或右击该文件并选择“安装”。

若您的系统不支持 MSI（如 Windows NT），则可以从 Microsoft 获取免费的 MSI 支持。另外，您可以使用标准 zip 压缩包。

**PDFlib COM 安装程序的作用** 为 PDLlib COM 组件提供的安装程序将自动处理与将 PDLlib 用于 COM 相关的所有问题。为了保持完整性，以下将说明使用 PDLlib 所要求的运行时环境（安装例程将负责设置此环境）：

- ▶ 将 PDLlib COM DLL *pdflib\_com.dll* 复制到安装目录。
- ▶ PDLlib COM DLL 必须在 Windows 注册表注册。安装程序使用自行注册 PDLlib DLL 以完成注册。
- ▶ 若获许可版本的 PDLlib 已安装，则已在系统中输入许可证密钥。

### 2.3.3 无提示安装

若 PDLlib 必须作为另外的软件包重新分发，或必须在由一些工具（如 SMS）管理的大量计算机上部署，则在每台计算机上手动安装 PDLlib 可能会比较麻烦。在这种情况下，PDLlib 也可以自动安装，而不需要任何用户干预。

**使用 MSI 安装程序进行无提示安装** MSI 安装程序支持无提示安装。例如，您可以从带有以下命令的命令行安装 PDLlib，而不需要任何用户干预：

```
msiexec.exe /I PDLlib-6.0.2-com-dotnet.msi /qn
```

请查阅 Microsoft Windows 安装程序文档中有关命令行选项的完全列表。

### 2.3.4 部署方案

**在 ISP 的服务器上部署 PDLlib COM** 通常在 Internet 服务提供商 (ISP) 管理的服务器上安装软件比在本地计算机上安装软件要难一些，因为当用户需要安装一些新的软件时，ISP 经常都会非常不乐意。PDLlib 对 ISP 非常友好，因为它既不会弄乱 Windows 目录，也不会损坏注册表：

- ▶ 仅需要一个 DLL，只要使用 *regsvr32* 实用工具将该 DLL 适当注册，则其可以位于任意目录中。
- ▶ 默认情况下，仅创建少数专用注册表项，这些注册表项位于 *HKEY\_LOCAL\_MACHINE\SOFTWARE\PDFlib* 注册表配置单元下。如需要，可以手动创建这些项。
- ▶ 若确实需要，甚至可以使用 PDLlib，而不通过任何专用注册表项。用户必须通过对 *set\_parameter()* 函数使用适当的调用以设置 *SearchPath*、*resourcefile* 和 *license* 参数来补偿这些项。不过，请注意 COM 机制本身将仍需要一些注册表项以便安装 PDLlib。

请参见下面的部分获取一个文件列表。

**重新发布 PDLlib COM 组件** 购买了可重新发布的运行库许可证并希望将 PDLlib COM 组件连同他们自己的产品一起重新发布的开放人员，必须提供完整的 PDLlib 安装并作为其产品的安装过程的一部分运行 PDLlib 安装程序，或必须执行以下所有过程：

- ▶ 在他们自己的安装中集成 PDLlib 安装文件（也请参见第 21 页上的第 2.3.3 节“无提示安装”）。通过查阅 PDLlib 安装目录可以轻松确定 PDLlib 要求的文件列表，因为这是 PDLlib 安装程序放置任何文件的唯一位置。
- ▶ 维护必要的 PDLlib 注册表项。可以通过在提供的注册文件模板 *pdflib.reg* 中完成各个项，然后在您自己的产品安装过程期间使用此注册文件来完成此操作。

- ▶ 必须为自我注册调用 *pdflib\_com.dll*（例如，使用 *regsvr32* 实用工具）。
- ▶ 在运行时提供您的许可证密钥，使用 PDFlib 的 *set\_parameter()* 函数，提供 *license* 作为第一参数，并提供实际许可证密钥作为第二参数（也请参见第 9 页上的第 o 节“应用 PDFlib 许可证密钥”）：

```
oPDF.set_parameter("license", "... 您的许可证密钥 ...")
```

### 2.3.5 COM 的错误处理

根据 COM 约定完成 PDFlib COM 组件的错误处理：当发生 PDFlib 异常时，将引发 COM 异常并提供错误的明文说明。另外，将释放 PDFlib 对象分配的内存。

可以在 PDFlib 客户端中利用客户端环境支持处理 COM 错误的任意方式，捕捉和处理 COM 异常。有关 PDFlib 的异常机制的更详细的讨论可以在第 40 页上的第 3.1.3 节“异常处理”中找到。

### 2.3.6 将 PDFlib 与 Active Server Pages 一起使用

有关 **Active Server Pages** 的特殊注意事项 当使用外部文件（例如图像文件）时，必须使用 ASP 的 *MapPath* 功能以便将本地磁盘上的路径名称映射到可以在 ASP 脚本内使用的路径。若您对 *MapPath* 不熟悉，可以查阅 PDFlib 提供的 ASP 示例以及 ASP 文档。不要在 ASP 脚本中使用绝对路径名称，因为没有 *MapPath*，这些名称可能不起作用。

注：UNC 路径名称在 IIS 中不起作用。

除非使用用于生成 PDF 的内核方法，否则包含 ASP 脚本的目录必须具有执行权限和写入权限（提供的 ASP 示例使用内核 PDF 生成）。

通过在 ASP 页面上的实际脚本代码的外部实例化对象，您可以提高 COM 对象（如 **Active Server Pages** 上的 *PDFlib\_com*）的性能，并有效提供对象会话范围，而不是提供页面范围。更加确切地说，不使用 *CreateObject*（如下一节中的示例所示）。

```
<%@ LANGUAGE = "JavaScript" %>
<%
    var oPDF;
    oPDF = Server.CreateObject("PDFlib_com.PDF");
    oPDF.begin_document("", "");
    ...
```

使用带有 *RUNAT*、*ID*、和 *ProgID* 属性的 *OBJECT* 标记以创建 *PDFlib\_com* 对象：

```
<OBJECT RUNAT=Server ID=oPDF ProgID="PDFlib_com.PDF"> </OBJECT>

<%@ LANGUAGE = "JavaScript" %>
<%
    oPDF.begin_document("", "");
    ...
```

通过对 *global.asa* 文件使用此技术，使用 *Scope=Application* 属性，从而为对象提供应用程序范围，您可以更大程度地提高性能。

其他 ASP 示例可以在 PDFlib 分发中找到。

使用 **JScript** 的 **Active Server Pages (ASP)** 的“Hello world”示例 与其他示例不同的是，我们在 ASP 示例中未创建 PDF 输出文件，而是在内存中生成 PDF 数据并通过 HTTP 将该数据直接发送到客户端。此技术更加适合 Web 服务器环境。

```

<%@ LANGUAGE = "JavaScript" %>
<%
    var exc;
    try {
        var font;
        var oPDF;

        oPDF = Server.CreateObject("PDFlib_com.PDF");

        if (oPDF == null) {
            Response.write("Couldn't create PDFlib object!");
            Response.end();
        }

        // Generate a PDF in memory; insert a file name to create PDF on disk
        if (oPDF.begin_document("", "") == -1) {
            Response.write("Error:" + oPDF.get_errmsg());
            Response.end();
        }

        oPDF.set_info("Creator", "hello.js.asp");
        oPDF.set_info("Author", "Thomas Merz");
        oPDF.set_info("Title", "Hello, world (COM/ASP/JScript!)");

        oPDF.begin_page_ext(595, 842, "");

        font = oPDF.load_font("Helvetica-Bold", "unicode", "");

        oPDF.setfont(font, 24);
        oPDF.set_text_pos(50, 700);
        oPDF.show("Hello, world!");
        oPDF.continue_text("(says COM/ASP/JScript)");
        oPDF.end_page_ext("");

        oPDF.end_document("");

        Response.Expires = 0;
        Response.Buffer = true;
        Response.ContentType = "application/pdf";
        // The following header is suggested in order to prevent problems with IE
        Response.Addheader("Content-Disposition", "inline; filename=" + "hello.pdf");

        Response.BinaryWrite(oPDF.get_buffer());
        Response.End();

    } catch (exc) {
        Response.write("Error:" + exc.description);
        Response.End();
    }
%>

```

利用 **JScript** 进行错误处理 JScript 5.0<sup>1</sup> 将结构化异常处理添加到类似于 C++ 或 Java 的语言，所不同的是 JScript 异常不能类型化，一条子句只能处理一个异常。通过一个 *try ... catch* 子句检测异常并进行相应处理：

1. JScript 5.0 可以与 Microsoft Internet Explorer 5.0 和 Microsoft Internet Information Services 5.0 一起使用

```

try {
    ...some PDFlib instructions...
} catch (exc) {
    Response.write("Error:" + exc.description);
    Response.end();
}

```

## 2.3.7 将 PDFlib 与 Visual Basic 一起使用

有关 **Visual Basic** 的特别注意事项 谈到利用外部 COM 组件，**Visual Basic** 支持早期（编译时）和后期（运行时）绑定。尽管两种类型的绑定可能都适合 **PDFlib**，但是强烈建议使用早期绑定。可通过执行以下步骤达到此目的：

- ▶ 通过单击“项目”->“引用”并选择 *PDFlib\_com* 控件，创建一个从 VB 项目到 **PDFlib** 的引用。
- ▶ 声明 *PDFlib\_com.PDF* 类型的对象变量，而不是一般类型 *Object*：

```

Dim oPDF As PDFlib_com.PDF
Set oPDF = CreateObject("PDFlib_com.PDF") ' or: Set oPDF = New PDFlib_com.PDF

```

创建引用并使用早期绑定具有以下一些优势：

- ▶ **VB** 可以检查代码中的拼写错误。
- ▶ 可以使用 **IntelliSense**（自动语句完成）功能和区分上下文的帮助。
- ▶ **VB** 对象浏览器显示所有 **PDFlib** 方法，以及它们的参数和一个简短的说明。
- ▶ **VB** 程序使用早期绑定机制的运行速度比使用后期绑定要快得多。

使用 **Visual Basic** 可以直接进行 **PDFlib** 编程，但有一个例外。由于 **Visual Basic 6** 中的一个 **Microsoft** 确认的 **Bug**（请原谅：一个“问题”），**PDFlib** 功能无法直接使用，因为 **VB** 用自身的一些内置方法错误地重写 **PDFlib** 方法名称。例如，用 **VB 6** 无法成功编译以下内容：

```
oPDF.circle 10, 10, 30
```

为了解决此问题，**PDFlib API** 提供了以下等效方法：

```
pcircle（等同于 circle）
pscale（等同于 scale）
```

另外，您可以使用以下 **Microsoft** 技术支持建议的语法：

```
oPDF.[circle] 10, 10, 30
```

将关键方法名称放入括号内似乎可行。此问题仅影响以下一些 **PDFlib** 功能：

```
circle
scale
```

在 **PDFlib COM** 组件中使用 *integer* 数据类型时，该类型是一个有符号的 32 位数。在 **Visual Basic** 中，该类型对应于 *long* 数据类型。因此，当 **PDFlib API** 引用需要一个 *int* 类型变量时，**Visual Basic** 程序员应将其转换为 *long*（尽管提供 *int* 值，**VB** 也会正确转换）。

用 **Visual Basic** 编写的“Hello world”示例

```
Attribute VB_Name = "hello"
```

```
Option Explicit
```

```
Sub main()
```



```

Dim ret As Long, font As Long
Dim oPDF As PDFlib_com.PDF

On Error GoTo ErrExit

Set oPDF = New PDFlib_com.PDF

ret = oPDF.begin_document("hello.pdf", "")
If (ret = -1) Then
    MsgBox "Error: " & oPDF.get_errmsg
End
End If

oPDF.set_info "Creator", "hello.bas"
oPDF.set_info "Author", "Thomas Merz"
oPDF.set_info "Title", "Hello, world (COM/VB)!"

oPDF.begin_page_ext 595, 842, ""

font = oPDF.load_font("Helvetica-Bold", "unicode", "")

oPDF.setfont font, 24

oPDF.set_text_pos 50, 700
oPDF.show "Hello, world!"
oPDF.continue_text "(says COM/VB)"

oPDF.end_page_ext ""
oPDF.end_document ""

Set oPDF = Nothing
End

ErrExit:
    MsgBox "PDFlib exception " & Err.Description

End Sub

```

利用 **Visual Basic** 进行错误处理 当发生错误时，**Visual Basic** 程序可以对错误进行检测并作出反应。使用 *On Error GoTo* 子句实现在 **Visual Basic** 中捕获异常：

```

Sub main()
    Dim oPDF As PDFlib_com.PDF
    On Error GoTo ErrExit

    ...some PDFlib instructions...

End

ErrExit:
    MsgBox Hex(Err.Number) & ": " & Err.Description
End Sub

```

## 2.3.8 将 PDFlib 与 Windows Script Host 一起使用

Windows Script Host<sup>1</sup> 支持 JScript 和 VBScript。由于 JScript 示例已在第 22 页上的第 2.3.6 节“将 PDFlib 与 Active Server Pages 一起使用”中讨论。为此，在这里我们仅将列举 VBScript 的“Hello world”示例。

## 使用 VBScript 的 Windows Script Host (WSH) 的 “Hello world” 示例

Option Explicit

Dim font

Dim oPDF

Set oPDF = WScript.CreateObject("PDFlib\_com.PDF")

if (oPDF.begin\_document("hello.pdf", "") = -1) then

    WScript.Echo "Error: " & oPDF.get\_errmsg

    WScript.Quit(1)

end if

oPDF.set\_info "Creator", "hello.vbs"

oPDF.set\_info "Author", "Thomas Merz"

oPDF.set\_info "Title", "Hello, world (COM/WSH)!"

oPDF.begin\_page\_ext 595, 842, ""

font = oPDF.load\_font("Helvetica-Bold", "unicode", "")

oPDF.setfont font, 24

oPDF.set\_text\_pos 50, 700

oPDF.show "Hello, world!"

oPDF.continue\_text "(says COM/VBS)"

oPDF.end\_page\_ext ""

oPDF.end\_document ""

Set oPDF = Nothing

利用 **VBScript** 进行错误处理 遗憾的是，VBScript 没有用来捕获错误的任何方法，只能忽略错误。为此，VBScript 必须定时检查 *ERR* 对象以便查看在对 COM 组件先前的调用中是否出错。VBScript 缺少 *On Error GoTo* 子句有个主要缺点，即脚本代码不是与对错误检查例程的调用混乱，就是后面的错误可能在第一个错误和错误检查例程的下个调用之间发生：

On Error Resume Next

Err.Clear

...some PDFlib instructions...

CheckPDFError

...more PDFlib instructions...

Sub CheckPDFError()

    If Err.number <> 0 then

        WScript.Echo "Error: " & Err.description

        Err.Clear

    End If

end Sub

1. WSH 提供了命令行形式 (*cscript.exe*) 和窗口形式 (*wscript.exe*)。在 *Microsoft Internet Explorer 5*、*Windows 98*、*2000* 和 *XP* 中都包含 WSH。有关更多信息，请参见 [msdn.microsoft.com/scripting](http://msdn.microsoft.com/scripting)。

## 2.3.9 将 PDFlib 与 Borland Delphi 一起使用

有关 **Borland Delphi 5-7<sup>1</sup>** 的特别注意事项 为了将 PDFlib 与 Delphi 5-7 一起使用，您必须在项目中引用 PDFlib 类型库，如下所示：在“项目”->“导入类型库”中，从列表中选择 PDFlib（或者，如果列表中没有 PDFlib，则使用“添加”以选择磁盘上的 *pdflib\_com.dll*），选择“调色板页”中的任意项，然后单击“安装”。此操作只需对每个 Delphi 安装执行一次，而不用对每个项目都执行一次。

**Borland Delphi 2005 和 2006.** 理论上来说，以下的流程是可行的：*Component, Add Component, Import a Type Library*, 选择 *PDFlib\_com*, *Add unit to ... project*。然而在我们的测试中，这种做法只在第一次有效。在随后 PDFlib 调用中，Delphi 产生 “*Field X does not have a corresponding component. Remove the declaration?*” 信息。由于我们尚未找到可靠的解决办法，PDFlib 暂时不支持 Delphi 2005 和 2006。**Borland Delphi** 的 “Hello world” 示例

```
unit hello;

interface

uses Windows, Classes, Forms, Dialogs, ExtCtrls, Controls, StdCtrls,
    PDFlib_com_TLB, OleServer;

type
    TForm1 = class(TForm)
        Panel1: TPanel;
        InsertBtn: TButton;
        pdf: TPDF;
        procedure InsertBtnClick(Sender: TObject);
    end;

var
    Form1: TForm1;

implementation

{$R *.DFM}

procedure TForm1.InsertBtnClick(Sender: TObject);
var
    font: Integer;
begin
    if pdf.begin_document('hello.pdf', '') = -1 then begin
        ShowMessage('Error: ' + pdf.get_errmsg());
        Exit;
    end;

    pdf.set_info('Creator', 'hello.pas');
    pdf.set_info('Author', 'Rainer Schaaf');
    pdf.set_info('Title', 'Hello World (Delphi)');

    pdf.begin_page_ext(595,842,'');

    font := pdf.load_font('Helvetica-Bold', 'winansi', '');
end;
```

1. 请参见 [www.borland.com/delphi](http://www.borland.com/delphi)

```

pdf.setfont(font, 24);
pdf.set_text_pos(50, 700);
pdf.show('Hello World');
pdf.continue_text('(says Delphi)');
pdf.end_page_ext('');

pdf.end_document('');
end;

end.

```

利用 **Borland Delphi** 进行错误处理 Delphi 支持结构化异常处理，可以用于处理从 OLE 对象（如 PDFlib）引发的异常。为了从 Delphi 代码捕获 PDFlib 异常，请使用以下基本结构：

```

uses SysUtils;

...

try
    ...some PDFlib instructions...
except
    On E: Exception do begin
        ShowMessage(E.Message);
    end;
end;

```

### 2.3.10 将 PDFlib COM 版本与 .NET 一起使用

作为 PDFlib.NET 版本的一个替代（请参见第 31 页上的第 2.7 节 “.NET 绑定”），PDFlib 的 COM 版本也可以与 .NET 一起使用。首先，您必须通过使用 *tlbimp.exe* 实用程序创建 PDFlib COM 版本的 .NET 程序集：

```
tlbimp pdflib_com.dll /namespace:pdflib_com /out:Interop.pdflib_com.dll
```

您可以在 .NET 应用程序内使用此程序集。若您对源自 Visual Studio .NET 内的 *pdflib\_com.dll* 添加引用，则将自动创建程序集。

将 **PDFlib COM** 版本与 **VB.NET** 一起使用 以下代码片段显示如何将 PDFlib COM 版本与 VB.NET 一起使用：

```

Imports PDFlib_com
...
Dim p As PDFlib_com.IPDF
...
p = New PDF()
...
buf = p.get_buffer()

```

将 **PDFlib COM** 版本与 **C#** 一起使用 以下代码片段显示如何将 PDFlib COM 版本与 C# 一起使用：

```

using PDFlib_com;
...
static PDFlib_com.IPDF p;
...
p = New PDF();

```

```
...
buf = (byte[])p.get_buffer();
```

代码的其余部分与用于 PDFlib 的 .NET 版本相同。请注意，在 C# 中，您必须转换 *get\_buffer()* 的结果，因为这里不存在 COM 对象返回的 VARIANT 数据类型的转换（类似的转换对于 *create\_pfv()* 是必要的）

与 PDFlib COM 版本一起使用的 ASP.NET 和 VB 的 “Hello world” 示例

```
<%@ Import Namespace="System" %>
<%@ Import Namespace="System.Runtime.InteropServices" %>
<%@ Import Namespace="PDFlib_com" %>
<html>
  <script language="VB" runat="server">
    Private Sub Page_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
      Dim p As PDFlib
      Dim font As Integer
      Dim buf() As Byte

      Try
        p = New PDFlib()

        ' Generate a PDF in memory; insert a file name to create PDF on disk
        if (p.begin_document("", "") = -1) then
          Response.write("Error: " & p.get_errmsg())
          Response.end
        end if

        p.set_info("Creator", "hello.vb.aspx")
        p.set_info("Author", "Rainer Schaaf")
        p.set_info("Title", "Hello, world (ASP.NET/VB)!")

        p.begin_page_ext(595, 842, "")

        font = p.load_font("Helvetica-Bold", "unicode", "")

        p.setfont(font, 24)
        p.set_text_pos(50, 700)
        p.show("Hello, world!")
        p.continue_text("(says .NET/VB)")

        p.end_page_ext("")
        p.end_document("")

        buf = p.get_buffer()

        Response.Buffer = true
        Response.ContentType = "application/pdf"
        Response.AppendHeader("Content-Disposition", "inline; filename=hello.pdf")
        Response.AppendHeader("Content-Length", buf.Length.ToString)
        Response.BinaryWrite(buf)
        Response.End()

      Catch err As PDFlibException
        ' caught exception thrown by PDFlib
        Response.Write("PDFlib exception occurred in hello.vb.aspx sample:<br><b>")
        Response.Write(String.Format("[{0}] {1}: {2}",
          err.get_errnum, err.get_apiname, err.get_errmsg))
      End Try
    End Sub
  </script>
</html>
```

```
        Finally
            p = nothing
        End Try

    End Sub

</script>
</html>
```

## 2.4 C 绑定

（只有通用版的 **PDFlib** 手册中包括此小节。）

## 2.5 C++ 绑定

(只有通用版的 PDFlib 手册中包括此小节。)

## 2.6 Java 绑定

(只有通用版的 PDFlib 手册中包括此小节。)

## 2.7 .NET 绑定

Microsoft .NET 结构<sup>1</sup>支持多种基于公共语言运行库 (CLR) 的编程语言，该公共语言运行库提供用于执行程序的公共环境。

PDFlib 的 .NET 版本支持所有相关的 .NET 概念。从技术上讲，PDFlib .NET 版本是一个 C++ 类（带有未托管 PDFlib 核心库的托管包装），它在 .NET Framework 控制下运行。将其打包成一个带有强名称的静态程序集。PDFlib 程序集 (*pdflib\_dotnet.dll*) 包含实际库和元信息。

注：PDFlib.NET 要求 .NET framework 1.1 或更高版本。它不使用 framework 1.0。若您必须使用 Framework 1.0，我们建议按照第 28 页上的第 2.3.10 节“将 PDFlib COM 版本与 .NET 一起使用”中记录的那样将 PDFlib COM 组件作为 .NET 程序集使用。

### 2.7.1 安装 PDFlib .NET 版本

PDFlib.NET 可以在所有支持 .NET Framework 1.1 或更改版本的环境中进行开发。我们将在以下一些环境中演示我们的示例：

- ▶ C#
- ▶ Visual Basic .NET
- ▶ ASP.NET with VB .NET

另外，存在一些其他的 .NET 的开发环境。PDFlib 也使用这些环境，尽管我们不提供编程示例。

安装 PDFlib.NET 的要求 安装 PDFlib 是一个简单且直接的过程。请注意以下内容：

- ▶ 显然，必须在安装 PDFlib.NET 之前安装 .NET Framework。
- ▶ PDFlib.NET 自身不需要任何特定的 Windows 版本或服务包。

PDFlib.NET MSI 安装程序将在计算机上交交互式安装 PDFlib 程序集，附加的数据文件，文档和示例。同时，安装程序对 PDFlib 进行注册使之易于在 Visual Studio .NET 的 *Add Reference* 对话框 .NET tab 标签中被引用 (reference)。

### 2.7.2 无提示安装

MSI 安装程序也支持无提示安装。例如，您利用以下命令从命令行安装 PDFlib，而不通过任何用户干预：

```
msiexec.exe /I PDFlib-6.0.2-com-dotnet.msi /qn
```

请审阅 Microsoft Windows 安装程序文档，以获取一个命令行选项的完整列表。

也支持一个称作“xcopy 部署”的过程。通过使用 *xcopy* 命令或 FTP 转换，您可以将 PDFlib 程序集 (*pdflib\_dotnet.dll*) 简单复制到服务器。

1. 请参见 [msdn.microsoft.com/vstudio](http://msdn.microsoft.com/vstudio)

### 2.7.3 .NET 的错误处理

PDFlib.NET 支持 .NET 异常，并在发生运行时问题时，将引发带有详细错误消息的异常。客户端负责捕获此异常并对其作出适当的反应。否则，.NET Framework 将捕获此异常，通常还将终止应用程序。

为了传达与异常相关的信息，PDFlib 用成员 *get\_errnum*、*get\_errmsg* 和 *get\_apiname* 定义其自身的异常类 *PDFlib\_dotnet.PDFlibException*。

### 2.7.4 将 PDFlib 与 C# 一起使用

有关 C# 的特别注意事项 为了在您的 C# 项目中使用 PDFlib.NET，您必须按如下所示在 Visual C# .NET 中对 PDFlib.NET 程序集创建引用：项目、添加引用 ...、浏览 ... 并从安装目录中选择 *PDFlib\_dotnet.dll*。

通过命令行编译器，您可以如以下示例中那样引用 PDFlib.NET：

```
csc.exe /r:...\bin\PDFlib_dotnet.dll hello.cs
```

#### C# 的 “Hello world” 示例

```
using System;
using System.Text;
using PDFlib_dotnet;

class Hello {
    static void Main(string[] args) {

        PDFlib p;
        int font;

        p = new PDFlib();

        try
        {
            if (p.begin_document("hello.pdf", "") == -1)
            {
                Console.WriteLine("Error: {0}\n", p.get_errmsg());
                return;
            }

            p.set_info("Creator", "hello.cs");
            p.set_info("Author", "Rainer Schaaf");
            p.set_info("Title", "Hello, world (.NET/C#)!");

            p.begin_page_ext(595, 842, "");

            font = p.load_font("Helvetica-Bold", "unicode", "");

            p.setfont(font, 24);
            p.set_text_pos(50, 700);
            p.show("Hello, world!");
            p.continue_text("(says .NET/C#)");
            p.end_page_ext("");

            p.end_document("");
        }

        catch (PDFlibException e)
```



```

    {
        // caught exception thrown by PDFlib
        Console.WriteLine("PDFlib exception occurred in hello sample:\n");
        Console.WriteLine("[{0}] {1}: {2}\n",
            e.get_errnum(), e.get_apiname(), e.get_errmsg());
        return;
    }
}

```

利用 **C#** 进行错误处理 客户端代码可以用常见的 *try...catch* 构造处理由 PDFlib 引发的 .NET 异常:

```

try {
    ...some PDFlib instructions...
} catch (PDFlibException e)
{
    // caught exception thrown by pdflib
    Console.WriteLine("PDFlib exception occurred in hello sample:\n");
    Console.WriteLine("[{0}] {1}: {2}\n",
        e.get_errnum(), e.get_apiname(), e.get_errmsg());
}

```

## 2.7.5 将 PDFlib 与 VB.NET 一起使用

有关 **Visual Basic .NET** 的特别注意事项 为了在您的 VB.NET 项目中使用 PDFlib.NET, 您必须按如下所示在 Visual Basic .NET 中对 PDFlib.NET 程序集创建引用: 项目、添加引用 ...、浏览 ... 并从安装目录中选择 *PDFlib\_dotnet.dll*。

通过命令行编译器, 您可以如以下示例中那样引用 PDFlib.NET:

```
vbc.exe /r:...\bin\pdflib_dotnet.dll hello.vb
```

### Visual Basic .NET 的 “Hello world” 示例

```

Imports System
Imports System.Text
Imports PDFlib_dotnet

Class hello
    Public Shared Sub Main()
        Dim p As PDFlib_dotnet.PDFlib
        Dim font As Integer

        p = New PDFlib()

        Try
            If (p.begin_document("hello.pdf", "") = -1) Then
                Console.WriteLine("Error: {0}", p.get_errmsg())
            End If

            p.set_info("Creator", "hello.vb")
            p.set_info("Author", "Rainer Schaaf")
            p.set_info("Title", "Hello, world (VB.NET)!")

            p.begin_page_ext(595, 842, "")

```

```

font = p.load_font("Helvetica-Bold", "unicode", "")

p.setfont(font, 24)
p.set_text_pos(50, 700)
p.show("Hello, world!")
p.continue_text("(says .NET/VB)")

p.end_page_ext("")
p.end_document("")

Catch e As PDFlibException
    Console.WriteLine("PDFlib exception occurred in hello sample:")
    Console.WriteLine("{0} {1}: {2}",
        e.get_errnum(), e.get_apiname(), e.get_errmsg())
Finally
    p = Nothing
End Try

End Sub
End Class

```

利用 **Visual Basic .NET** 进行错误处理 Visual Basic .NET 支持两种不同类别的异常处理：

- ▶ 现代结构化异常处理（也在其他现代语言中使用，如 C#）
- ▶ 传统结构化异常处理（已成为 Visual Basic 6.0 中唯一的异常处理方式）

尽管客户端通过使用两种类别的异常处理，能够处理由 PDFlib 引发的 .NET 异常，但是语法是不同的。建议使用结构化异常处理来捕获异常。该异常处理使用 *try...catch* 子句：

```

Try
    ...some PDFlib instructions...
Catch e As PDFlibException
    Console.WriteLine("PDFlib exception occurred in hello sample:")
    Console.WriteLine("{0} {1}: {2}",
        e.get_errnum(), e.get_apiname(), e.get_errmsg())
End Try

```

使用传统结构化异常处理捕获异常是通过一个 *On Error GoTo* 子句实现的：

```

Imports Microsoft.VisualBasic

Public Shared Sub Main()
    On Error GoTo ErrExit
    ...some PDFlib instructions...
    Exit Sub

ErrExit:
    Console.WriteLine("PDFlib exception caught: {0}", Err.Description)
End Sub

```

## 2.7.6 将 PDFlib 与 ASP.NET 一起使用

为 **ASP.NET** 安装 **PDFlib.NET** 为了在您的 ASP.NET 脚本中使用 PDFlib.NET, 您必须使 PDFlib.NET 程序集对 ASP 可用。通过将 *PDFlib\_dotnet.dll* 放置在 IIS 安装的 *bin* 子目录中 (若它不存在, 则您必须手动创建), 或放置在 Web 应用程序的 *bin* 子目录中, 可以实现这一点。

```
C:\Inetpub\wwwroot\bin\pdflib_dotnet.dll
```

或

```
C:\Inetpub\wwwroot\WebApplicationX\bin\pdflib_dotnet.dll
```

有关 **ASP.NET** 的特别注意事项 当使用外部文件 (例如图像文件) 时, 必须使用 ASP 的 *MapPath* 功能以便将本地磁盘上的路径名称映射到可以在 ASP 脚本内使用的路径。若您对 *MapPath* 不熟悉, 可以查阅 PDFlib 提供的 ASP.NET 示例以及 ASP.NET 文档。不要使用 ASP.NET 脚本中的绝对路径名称, 因为没有 *MapPath*, 这些名称可能不起作用。

除非使用用于生成 PDF 的核内方法, 否则包含 ASP.NET 脚本的目录必须具有执行权限和写入权限 (提供的 ASP 示例使用核内 PDF 生成)。

### ASP.NET 和 VB 的 “Hello world” 示例

```
<%@ Import Namespace="System" %>
<%@ Import Namespace="PDFlib_dotnet" %>
<html>
  <script language="VB" runat="server">
    Private Sub Page_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
      Dim p As PDFlib
      Dim font As Integer
      Dim buf() As Byte

      Try
        p = New PDFlib()

        ' Generate a PDF in memory; insert a file name to create PDF on disk
        if (p.begin_document("", "") = -1) then
          Response.write("Error: " & p.get_errmsg())
          Response.end
        end if

        p.set_info("Creator", "hello.vb.aspx")
        p.set_info("Author", "Rainer Schaaf")
        p.set_info("Title", "Hello, world (ASP.NET/VB)!")

        p.begin_page_ext(595, 842, "")

        font = p.load_font("Helvetica-Bold", "unicode", "")

        p.setfont(font, 24)
        p.set_text_pos(50, 700)
        p.show("Hello, world!")
        p.continue_text("says .NET/VB")

        p.end_page_ext("")
        p.end_document("")
      End Try
    End Sub
  </script>
</html>
```

```

        buf = p.get_buffer()

        Response.Buffer = true
        Response.ContentType = "application/pdf"
        Response.AppendHeader("Content-Disposition", "inline; filename=hello.pdf")
        Response.AppendHeader("Content-Length", buf.Length.ToString)
        Response.BinaryWrite(buf)
        Response.End()

    Catch err As PdfFlibException
        ' caught exception thrown by PdfFlib
        Response.Write("PdfFlib exception occurred in hello.vb.aspx sample:<br><b>")
        Response.Write(String.Format("[{0}] {1}: {2}",
            err.get_errnum, err.get_apiname, err.get_errmsg))
    Finally
        p = nothing
    End Try

End Sub

</script>
</html>

```

利用 **ASP.NET** 进行错误处理 请审阅第 32 页上的第 2.7.4 节 “将 PdfFlib 与 C# 一起使用” 和第 33 页上的第 2.7.5 节 “将 PdfFlib 与 VB.NET 一起使用” 中相应的段落。

## 2.8 Perl 绑定

(只有通用版的 PdfFlib 手册中包括此小节。)

## 2.9 PHP 绑定

(只有通用版的 PdfFlib 手册中包括此小节。)

## 2.10 Python 绑定

(只有通用版的 PdfFlib 手册中包括此小节。)

## 2.11 REALbasic 绑定<sup>1</sup>

### 2.11.1 安装 PdfFlib REALbasic 版本

在 Mac 和 Windows 上， PdfFlib 支持 REALbasic 开放环境。

**Mac 上的 REALbasic** REALbasic 的 PdfFlib 插件必须复制到 REALbasic 应用程序所在的相同文件夹中的一个称作 *Plugins* 的文件夹。尽管 Mac 上的 REALbasic 的 PdfFlib 是在单一包中交付的，但它包含对以下平台的支持：

- ▶ Mac OS Classic
- ▶ Mac OS X (Carbon)
- ▶ Windows

<sup>1</sup> 请参见 [www.realbasic.com](http://www.realbasic.com)

这意味着您可以使用 Mac 版本为 Mac 和 Windows 构建应用程序。独立应用程序一旦生成，REALbasic 将选择 PDFlib 插件的适当的部分并仅包含生成的应用程序中的平台特定的部分。

**Windows 上的 REALbasic** REALbasic 的 PDFlib 插件必须复制到 REALbasic 应用程序所在的相同文件夹中的一个称作 *Plugins* 的文件夹。Windows 上的 REALbasic 的 PDFlib 仅支持 Windows 开发。

其他 **REALbasic** 类 PDFlib 插件将两个新类添加到 REALbasic 的对象层次结构：

- ▶ 此 *PDFlib* 类包含所有 PDFlib API 方法。
- ▶ 派生自 *RuntimeException* 的 *PDFlibException* 类可以用于处理由 PDFlib 引发的异常（有关更详细的信息，请参见第 37 页上的第 2.11.3 节“REALbasic 的错误处理”）。

PDFlib 可以用于创建 GUI 应用程序以及控制台应用程序。由于 PDFlib 不是控件，所以它不能在 REALbasic 的控制调色板中安装新的图标。不过，当 PDFlib 插件可用时，REALbasic 将识别 PDFlib 类及其相关的方法。例如，完全用于 PDFlib API 方法的语句完成和参数检查。

### 2.11.2 用 REALbasic 编写的“Hello world”示例

```
Dim ret, rbfont As Integer
Dim p As PDFlib

p = New PDFlib()

ret = p.begin_document("hello.pdf", "")

If (ret = -1) Then
    MsgBox("Error: " + p.get_errmsg())
    Exit
End If

p.set_info("Creator", "hello.rb")
p.set_info("Author", "Rainer Schaaf")
p.set_info("Title", "Hello, world (REALbasic)!")

p.begin_page_ext(595, 842, "")

rbfont = p.load_font("Helvetica-Bold", "unicode", "")

p.setfont(rbfont, 24)

p.set_text_pos(50, 700)
p.show("Hello, world!")
p.continue_text("(says REALbasic)")

p.end_page_ext("")
p.end_document("")

Exception e As PDFlibException
MsgBox("PDFlib exception occurred in hello sample:" + CHR(10) + _
    "[" + Str(e.get_errnum()) + "] " + e.get_apiname() + ": " + e.get_errmsg())
Return
```

### 2.11.3 REALbasic 的错误处理

发生异常时，PDFlib 将引发 *PDFlibException* 类的本地 REALbasic 异常。利用标准 REALbasic 技术可以处理 PDFlib 异常：使用 *try/catch* 块（建议使用此技术，但它要求 REALbasic 5.5 或

更高版本)，或在 `Exception` 块中处理这些异常。在以下代码片段中，将演示在 `Exception` 块中处理这些异常：

```
Exception err As PDFlibException
  MsgBox("PDFlib exception occurred in hello sample: [" + _
    Str(err.get_errnum()) + "] " + err.get_apiname() + ": " + err.get_errmsg())
```

如此示例中所示，`REALbasic` 开发人员通过使用 `PDFlibException` 方法访问详细的错误信息以便检索错误号码、错误消息和引发此异常的 `API` 函数的名称。

## 2.12 Ruby 绑定

（只有通用版的 `PDFlib` 手册中包括此小节。）

## 2.13 RPG 绑定

（只有通用版的 `PDFlib` 手册中包括此小节。）

## 2.14 Tcl 绑定

（只有通用版的 `PDFlib` 手册中包括此小节。）

# 3 PDFlib 编程

## 3.1 常规编程

### 3.1.1 PDFlib 程序结构和函数范围

PDFlib 应用程序必须遵守某些非常易于理解的结构规则。按照这些限制编写应用程序可做到简单明了。例如，在关闭一个文档前，您不需要考虑先打开该文档。由于 PDFlib API 几乎是完全仿效文档 / 页范例，所以用自然方式生成文档通常可得到标准格式的 PDFlib 客户端程序。

PDFlib 使用严格的作用界定范围系统强制推行正确的函数调用顺序。函数说明指定一个函数的允许范围。从不同范围调用函数将触发 PDFlib 异常。若库客户端提供错误的参数，则 PDFlib 也将引发异常。

第 8 章中的函数说明引用到这些范围；范围定义可以在表 3.1 中找到。图 3.1 说明范围的嵌套。若在允许的范围之外调用函数，则 PDFlib 将引发异常。您可以使用 *scope* 参数查询当前范围。

表 3.1 函数范围定义

范围名称	定义
路径	由 <i>moveto()</i> 、 <i>circle()</i> 、 <i>arc()</i> 、 <i>arcn()</i> 或 <i>rect()</i> 中的一个函数开始；由第 198 页上的第 8.4.6 节 “路径绘制和剪切” 中的任何函数终止。
页	在 <i>begin_page()</i> 和 <i>end_page()</i> 之间，但是在路径范围之外
模板	在 <i>begin_template()</i> 和 <i>end_template()</i> 之间，但是在路径范围之外
图案	在 <i>begin_pattern()</i> 和 <i>end_pattern()</i> 之间，但是在路径范围之外
字体	在 <i>begin_font()</i> 和 <i>end_font()</i> 之间，但是在字形范围之外
字形	在 <i>begin_glyph()</i> 和 <i>end_glyph()</i> 之间，但是在路径范围之外
文档	在 <i>begin_document()</i> 和 <i>end_document()</i> 之间，但是在页、模板、图案和字体范围之外
对象	PDFlib 对象的生存时间中的任何时间，但是在文档范围之外

### 3.1.2 参数

PDFlib 的操作可以被各种全局参数控制。这些参数将在 PDFlib 对象的整个生存期内保留其设置，或者直到设置被客户端明确地更改。以下函数可以用于处理参数：

- ▶ *set\_parameter()* 可以用于为字符串类型参数赋值。
- ▶ *set\_value()* 可以用于为参数设置数值。
- ▶ *get\_parameter()* 可以用于查询字符串类型的参数。
- ▶ *get\_value()* 可以用于查询数值参数的值。

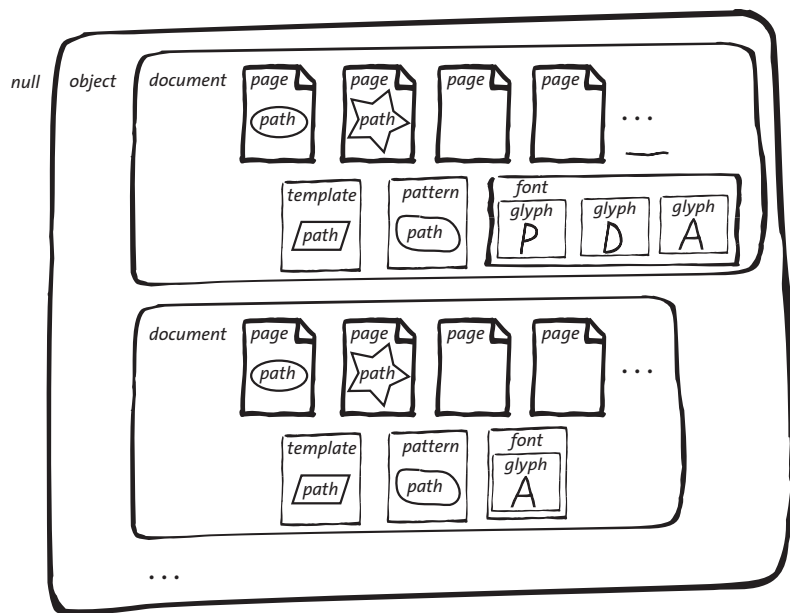


图 3.1  
范围的嵌套

有关参数名称和可能值的详细信息，见第 8 章。

### 3.1.3 异常处理

有一类型的错误在许多语言中有很好的理由被称作异常 — 它们仅仅是异常，并且在程序的生存期间不会经常发生。一般策略是对可能时常出错的函数调用使用常规的错误报告机制（读取：特殊错误返回代码），而对那些极少出现的且在一定条件下会弄乱代码的情况使用特殊异常机制。这正是 PDFlib 遵循的处理方法：可预期一些操作将相当频繁地出错，例如：

- ▶ 试图打开对其没有相应权限的输出文件
- ▶ 试图用错误的文件名打开输入 PDF 文件
- ▶ 试图打开损坏的图像文件

如 API 参考中所述，PDFlib 通过返回一个特殊的值（此值通常为 -1）来指示类似错误。其他事件可能被认为是有害的，但是它们将很少发生，例如：

- ▶ 虚拟内存不足
- ▶ 范围冲突（例如，打开某一文档前关闭该文档）
- ▶ 提供 PDFlib API 函数的错误参数（例如，试图使用负半径绘制圆形）

当 PDFlib 检测到这种情形时，将引发一个异常而不是将一个特殊的错误返回值传递给调用方。必须懂得一点，即在发生异常后无法完成生成的 PDF 文档。发生异常后，只可以安全调用以下方法之一：`delete()`、`get_apiname()`、`get_errnum()` 和 `get_errmsg()`。发生异常后，调用任何其他 PDFlib 方法可能导致意外结果。异常（或传递给 C 错误处理程序的数据）将包含以下信息：

- ▶ 唯一的错误号（请参见表 3.2）；
- ▶ 引发异常的 PDFlib API 函数的名称；
- ▶ 包含问题的详细信息的描述性文本；

**禁用异常** 可以禁用一些异常。这些异常分为两类：非致命错误（警告）以及根据客户端首选项设置可能会也可能不会认定为错误的错误。



表 3.2 PDFlib 异常号的范围

错误范围	原因
1000 - 1999	(PDCORE 库): 内存、I/O、变量、参数 / 值、选项
2000 - 3999	(PDFlib 库): 配置、范围、图形和文本、颜色、图像、字体、编码、PDF/X、超文本、标签 PDF、图层
4000 - 4999	(PDF 导入库 PDI): 配置和参数, 损坏的 PDF (文件、对象或流级别)

警告通常是指 PDFlib 代码中需要您密切关注和调查的一些问题。若出现的错误是非致命的时, 程序可继续运行。为此, 您可以使用以下函数调用取消显示警告:

```
oPDF.set_parameter "warning", "false"
```

除全局 *warning* 参数以外, 一些函数也支持一些专用选项来启用或禁用各个函数调用的警告。建议的策略是在开发周期内启用警告 (同时密切关注可能的警告), 而在生产系统中禁用警告。

某些操作对于一些客户端可能是严重的, 而其他的客户端可能已准备好处理这种情况。在这种情况下, 相应的 PDFlib API 函数的行为会根据参数发生更改。对于加载字体、图像、导入的 PDF 文档和 ICC 配置文件, 此差别已经得到实现。例如, 如果由于某个配置问题无法加载某一字体, 一个客户端可能会完全放弃加载, 而另外一个客户端可能会选择加载其他的字体。在参数或选项 *fontwarning* 设置为 *true* 的情况下, 当无法加载字体时, 将引发异常。否则, 函数将返回一个错误代码。可以按如下所示设置参数:

```
oPDF.set_parameter "fontwarning", "false"
```

查询函数调用失败的原因 正如以上提到的那样, 发生异常时, 必须总是放弃生成的 PDF 输出文档。不过, 有些客户端可能更愿意通过调整一些参数继续处理文档。例如, 当不能加载某一特别的字体时, 大多数客户端将放弃文档, 而其他的客户端可能更愿意使用不同的字体。可以使用 *fontwarning* 等参数实现此差别。在这种情况下, 可能需要检索已作为异常的一部分的错误消息。一旦某个函数调用失败, 则可能立即调用函数 *get\_errnum()*、*get\_errmsg()* 和 *get\_apiname()*。例如, 当函数调用返回 -1 错误值。

以下代码片段总结了有关异常处理的不同策略。示例试图加载并嵌入某一字体, 但假定该字体不可用。

若 *fontwarning* 参数或选项为 *true* (默认设置), 则必须放弃文档:

```
font = oPDF.load_font("MyFontName", "winansi", "fontwarning=true");
' unless an exception was thrown the font handle is valid;
' when an exception occurred the PDF output cannot be continued
```

若 *fontwarning* 参数或选项为 *false*, 则必须检查返回值的有效性。若返回值指示失败, 则查询失败的原因以便对这种情况进行正确地处理:

```
font = oPDF.load_font("MyFontName", "winansi", "fontwarning=false");
if (font == -1) Then
' font handle is invalid; find out what happened.
errmsg = oPDF.get_errmsg();
' Log error message
' Try a different font or give up
...
End If
' font handle is valid; continue as usual
```

### 3.1.4 PDFlib 虚拟文件系统 (PVF)

除磁盘文件以外，一种名为“PDFlib 虚拟文件系统” (PDFlib Virtual File System, PVF) 的工具允许客户端直接在内存中提供数据而不需要任何磁盘文件参与。这种方式带来了性能上的优势，可用于从数据库捕获的数据（这些数据甚至不存在于单独的磁盘文件上）以及其他一些客户端在内存中已具有所需的数据（作为处理的结果）的情况。

PVF 基于命名的虚拟只读文件的概念，这类文件与常规文件名称一样，可以与任何 API 函数一起使用。PVF 甚至可以在 UPR 配置文件中使用。客户端可以使用任意方法生成虚拟文件名称。显然，必须选择虚拟文件名称以避免与常规磁盘文件发生名称冲突。为此，建议遵循以下的虚拟文件名称的分层命名约定（*filename* 是指客户端选择的在相应类别中唯一的名称）。还建议保留标准文件名称后缀：

- ▶ 光栅图像文件：/pvf/image/filename
- ▶ 字型和规格文件（建议将实际字体名称用作文件名称的基础部分）：/pvf/font/filename
- ▶ ICC 配置文件：/pvf/iccprofile/filename
- ▶ 编码和代码页：/pvf/codepage/filename
- ▶ PDF 文档：/pvf/pdf/filename

当搜索一个命名的文件时，PDFlib 将首先检查提供的文件名称是否引用已知的虚拟文件，然后试图打开磁盘上相应的命名文件。

**虚拟文件的生存时间** 一些函数将立即使用虚拟文件中提供的数据，而其他函数将仅读取文件的部分内容，并在稍后使用其他数据片段。为此，必须密切关注虚拟文件的生存时间。PDFlib 将在每个虚拟文件上放置一个内部锁，并仅在不再需要相应内容时移除该锁。除非客户端请求 PDFlib 立即复制数据（使用 *create\_pvf()* 中的 *copy* 选项），否则，只有当 PDFlib 不再锁定虚拟文件内容时，客户端才能对虚拟文件内容进行修改、删除或释放。PDFlib 将使用 *delete()* 自动删除所有虚拟文件。不过，必须总是由客户端释放实际文件内容（包含虚拟文件的数据）。

**不同的策略** PVF 支持不同的管理虚拟文件内存的方法。这是由于：在接受虚拟文件名称的 API 调用之后，PDFlib 可能需要访问虚拟文件的内容，但从不在 *close()* 之后需要访问内容。记住，调用 *delete\_pvf()* 不会释放实际的文件内容（除非使用了 *copy* 选项），而只是释放用于 PVF 文件名称管理的相应的数据结构。这就产生了以下策略：

- ▶ 最小化内存使用：建议在接受虚拟文件名称的 API 调用之后立即调用 *delete\_pvf()*，并在 *close()* 之后再次调用。第二次调用是必需的，因为有可能在第一次调用时 PDFlib 未完成数据访问，故拒绝对虚拟文件解除锁定。当然，在有些情况下，第一次调用就已将数据释放，即使如此第二次调用也不会有任何害处。客户端仅在 *delete\_pvf()* 成功的情况下，才可能释放文件内容。
- ▶ 通过重用虚拟文件优化性能：有些客户端可能希望重用各种输出文档中的一些数据（例如，字体定义），和避免多次重复创建 / 删除相同的文件内容。在这种情况下，如果更多 PDF 输出文档将使用此虚拟文件，建议不要调用 *delete\_pvf()*。
- ▶ 简化编程：如果不需要考虑内存的使用情况，则客户端可选择不调用 *delete\_pvf()*。在这种情况下，PDFlib 将在 *delete()* 中内部删除所有挂起的虚拟文件。

在所有情况下，客户端仅当 *delete\_pvf()* 成功返回时，或在 *delete()* 之后释放相应的数据。

### 3.1.5 资源配置和文件搜索

在大多数高级应用程序中，PDFlib 需要访问字体文件、编码定义、ICC 配置文件等资源。为了使 PDFlib 的资源处理功能与平台无关并实现自定义，可以提供一份配置文件用于描述可用资源以及相应的磁盘文件的名称。除了静态配置文件之外，可以利用 *set\_parameter()* 在运行时添加资源从而实现动态配置。对于配置文件，我们挖掘出在 Display PostScript 时代

问世并仍被几个系统使用的简单文本格式 (*Unix PostScript Resource, UPR*)。不过，我们已经根据自己的需要扩展了原始的 UPR 格式。下面将描述 PDFlib 使用的 UPR 文件格式。有一种名为 *makespres*（经常作为 X Window 系统的一部分进行分发）的实用工具，此工具可用于从 PostScript 字库外形和量度文件自动生成 UPR 文件。

**资源类别** 在表 3.3 中已列出 PDFlib 支持的资源类别。其他资源类别将被忽略。这些值被视为名称字符串；可以用 ASCII 或 UTF-8（带 BOM）对其进行编码。Unicode 值可能对带有 *HostFont* 资源的本地化的字体名称很有用。

表 3.3 PDFlib 支持的资源类别

种类	格式	说明
<i>SearchPath</i>	<i>value</i>	包含数据文件的目录的相对或绝对路径名称
<i>FontAFM</i>	<i>key=value</i>	用 <i>AFM</i> 格式编写的 <i>PostScript</i> 字库量度文件
<i>FontPFM</i>	<i>key=value</i>	用 <i>PFM</i> 格式编写的 <i>PostScript</i> 字库量度文件
<i>FontOutline</i>	<i>key=value</i>	<i>PostScript</i> 、 <i>TrueType</i> 或 <i>OpenType</i> 字库外形文件
<i>Encoding</i>	<i>key=value</i>	包含 8 位编码或代码页表的文本文件
<i>HostFont</i>	<i>key=value</i>	系统上安装的字体的名称
<i>ICCProfile</i>	<i>key=value</i>	<i>ICC</i> 配置文件 的名称
<i>StandardOutputIntent</i>	<i>key=value</i>	<i>PDF/X</i> 的标准输出条件的名称

应避免冗余的资源项。例如，一个字体的量度数据不应包含多个项。另外，在 UPR 文件中配置的字体名称应与实际的字体名称完全相同以避免混淆（尽管 PDFlib 不强制此限制）。

在 Mac OS Classic 中，冒号字符 “:” 必须作为目录分隔符使用。必须使用完整路径（包括卷名）指定基于资源的 *PostScript Type 1* 字体（*LWFN* 字体）的字体名称，例如：

Foo-Italic=Classic:Data:Fonts:FooIta

**UPR 文件格式** UPR 文件是结构非常简单的文本文件，可以用文本编辑器轻松编写或自动生成。首先，让我们来了解一下句法问题：

- ▶ 每行最多有 255 个字符。
- ▶ 反斜线符号 “\” 转义换行符。它可能用于扩展行。
- ▶ 单独的句点字符 “.” 用作节终结符。
- ▶ 所有项都要区分大小写。
- ▶ 可以用百分比 “%” 字符引入注释行并在行的结尾终止。
- ▶ 除了资源名称和文件名称中的空白字符外，其余的将被忽略。

UPR 文件由以下各部分组成：

- ▶ 用于标识文件的特殊行。它具备以下形式：

PS-Resources-1.0

- ▶ 列出文件中描述的所有资源类别的可选节。每行描述一种资源类别。该列表由带有单个句点字符的行终止。稍后将描述可用的资源类别。
- ▶ 在文件的开头列出的针对每一个资源类别的节。每一节由显示资源类别的行开始，后跟任意数量的描述可用资源的行。该列表由带有单个句点字符的行终止。每个资源数据行都包含资源的名称（必须用引号将等号引起来）。若资源需要一个文件名称，则必须在等号后面添加此名称。当 PDFlib 搜索资源项中列出的文件时，将应用 *SearchPath*（请参见以下内容）。

文件搜索和 **SearchPath** 资源类别 PDFlib 从磁盘文件中读取各种数据项，例如光栅图像、字型和规格信息、编码定义、PDF 文档和 ICC 配置文件。除了相对或绝对路径名称之外，您还可以使用不带任何路径说明的文件名称。**SearchPath** 资源类别可以用于为包含必需的数据文件的目录指定路径名称的列表。当 PDFlib 必须打开一个文件时，它先会完全按照所提供的文件名称试图打开该文件。若此尝试失败，PDFlib 将接连尝试打开 **SearchPath** 资源类别中指定的目录下的文件，直到成功为止。**SearchPath** 项可以进行累积，并且将按相反的顺序对其进行搜索（即，先搜索后来设置的路径，再搜索早先设置的路径）。此功能可以用于让 PDFlib 应用程序不再依赖特定于平台的文件系统架构。为了禁用搜索，您可以在 PDFlib 函数中使用指定完整的路径名称。请注意 **SearchPath** 资源类别的下列特定于平台的功能：

- ▶ 在 Windows 上，PDFlib 将使用注册表中的项初始化 **SearchPath**。以下注册表项可能包含一个包含路径名称的列表，其中的路径名称由分号 “;” 字符隔开：

```
HKLM\SOFTWARE\PDFlib\PDFlib\6.0.3\SearchPath
```

- ▶ COM 和 .NET 安装程序将用以下目录初始化 **SearchPath** 注册表项（或类似的目录，若您已在别处安装 PDFlib）：

```
C:\Program Files\PDFlib\PDFlib 6.0.3\resource
```

示例 **UPR** 文件 以下列表提供了一个 UPR 配置文件示例：

```
PS-Resources-1.0
SearchPath
/usr/local/lib/fonts
C:/psfonts/pfm
C:/psfonts
/users/kurt/my_images
.
FontAFM
Code-128=Code_128.afm
.
FontPFM
Corporate-Bold=corpb__.pfm
.
Mistral=c:/psfonts/pfm/mist____.pfm
.
FontOutline
Code-128=Code_128.pfa
ArialMT=Arial.ttf
.
Encoding
myencoding=myencoding.enc
.
ICCPProfile
highspeedprinter=cmykhigspeed.icc
.
```

搜索 **UPR** 资源文件 若仅要使用内建资源（例如，PDF 核心字体、内置编码、sRGB ICC 配置文件）和系统资源（宿主字体），则不需要 **UPR** 配置文件，这是因为 PDFlib 在不使用任何附加配置的情况下，也可以找到所有必需的资源。

若要使用其他资源，您可以通过调用 **set\_parameter()** 指定这些资源（请参见以下内容），或者在 **UPR** 资源文件中指定这些资源。当请求第一个资源时，PDFlib 自动读取此文件。详细过程如下所示：

- ▶ 若定义环境变量 **PDFLIBRESOURCE**，则 PDFlib 将其值作为要读取的 **UPR** 文件的名称。若无法读取此文件，则将引发异常。

- ▶ 若未定义环境变量 *PDFLIBRESOURCE*，则 PDFlib 尝试打开带有以下名称的文件：

```
upr (在 MVS 上；预期为数据集)
pdflib/<version>/fonts/pdflib.upr (在 IBM eServer iSeries 上)
pdflib.upr (在 Windows、Unix 以及所有其他系统上)
```

即使无法读取此文件，也不会引发异常。

- ▶ 在 Windows 上，PDFlib 将额外尝试读取注册表项

```
HKLM\SOFTWARE\PDFlib\PDFlib\6.0.3\resourcefile
```

此项的值（将由 PDFlib 安装程序创建，但也可以通过其他方式创建）将作为要使用的资源文件的名称。若无法读取此文件，则将引发异常。

- ▶ 通过明确地设置 *resourcefile* 参数，客户端可以在运行时强制 PDFlib 读取资源文件：

```
oPDF.set_parameter "resourcefile", "c:\path\to\pdflib.upr"
```

可以时常任意重复此调用；资源项将进行累积。

运行时对资源进行配置 除了使用 UPR 文件进行配置以外，还能够通过 *set\_parameter()* 函数在源代码内直接配置各个资源。此函数采用一个类别名称和相应的资源项作为参数，这正如同它出现在 UPR 资源文件中此类别的相关节中一样，例如：

*end\_document()* 之后）。交错进行 PDF 数据的生产和消耗有几个好处。首先，由于不是所有数据都必须保存在内存中，从而降低了内存要求。其次，此方案可以增强性能，这是因为当下一个块区还在生成中时，第一个数据块区可以通过一个慢速链接传输。不过，只有当整个文档完成后，才能知道生成数据的总长度。

主动的 PDF 文件输出到内存的生成的接口 为了在内存中生成 PDF 数据，只需对 *begin\_document()* 提供一个空文件名称，并用 *get\_buffer()* 提取数据：

```
oPDF.begin_document ("", "")
...create document...
oPDF.close

' Fetch the buffer with the PDF and write to a file
' This is rather pointless in VB but useful in ASP
Open "file.pdf" For Binary Access Write As #1
Put #1, , oPDF.get_buffer
Close #1
Set oPDF = Nothing
```

注：缓冲区中的 PDF 数据必须作为二进制数据进行处理。

### 3.1.6 大文件支持

在本节中，术语“大文件”是指大小超过 2 GB 的文件。虽然对于普通用户来说似乎不需要如此大的文件，但是实际上有些企业应用程序需要创建或处理如包含大量发票或结算单的单个大文件。在这类情况下，文件大小可能会超过 2 GB 的限制。

PDFlib 支持输出大文件，也就是说，它可以创建大于 2 GB 的 PDF 输出。PDI 也支持处理大输入文件。不过，对大文件的支持仅适用于操作系统其本身支持大文件。显然，所使用的文件系统也必须支持大文件。请注意，Acrobat 6 及更早的版本不能处理大文件。不过，Acrobat 7 完全可以处理大文件。

注：除 PDF 以外的导入文件（例如字体和图像）不能超过 2 GB 限制。用 `get_buffer()` 接口获取的 PDF 输出片段也需要遵从此限制。最终，PDF 输出文件通常限于  $10^{10}$  个字节，即大约 9.3 GB。

## 3.2 页面说明

### 3.2.1 坐标系统

在 PDFlib 中使用 PDF 的默认坐标系统。默认坐标系统（或默认用户空间）的原点位于页面的左下角，并使用 DTP 点作为单位：

$1\text{ pt} = 1/72\text{ inch} = 25.4/72\text{ mm} = 0.3528\text{ mm}$

第一个坐标向右增加，第二个坐标向上增加。PDFlib 客户端程序可以通过旋转、缩放、平移或斜移更改默认用户空间，从而产生新的用户坐标。这些转换对应的函数分别为 `rotate()`、`scale()`、`translate()` 和 `skew()`。若已转换用户空间，则必须根据新的坐标系统提供图形和文本函数中的所有坐标。在每一页的开头，坐标系统都会重置为默认坐标系统。

使用公制坐标 通过缩放坐标系统可以轻松使用公制坐标。缩放因子从上面给定的 DTP 点的定义派生：

`oPDF.[scale] 28.3465, 28.3465`

完成此调用后，PDFlib 将以厘米为单位解释所有坐标（超文本功能除外，具体请参见以下内容），这是由于  $72/2.54 = 28.3465$ 。

用于超文本元素的坐标 PDF 总是需要用于超文本函数的坐标，例如在默认坐标系统而不是在（可能已经过转换的）用户坐标系统中用于创建文本注释、链接和文件注释的矩形坐标。由于这样非常麻烦，因此 PDFlib 对 PDF 需要的格式提供了用户坐标的自动转换。通过将 `usercoordinates` 参数设置为 `true` 可以激活此自动转换：

`oPDF.set_parameter "usercoordinates", "true"`

由于 PDF 仅支持其边缘平行于页边缘的超文本矩形，因此，当坐标系统已经通过缩放、旋转、平移或斜移完成转换后，必须修改所提供的矩形。在这种情况下，PDFlib 将计算其边缘平行于页边缘的最小包容矩形，将其转换到默认坐标，并使用得到的值替换提供的坐标。

总体而言，当 `usercoordinates` 参数已设置为 `true` 时，您可以对页面内容和超文本元素使用同一坐标系统。

使坐标可视化 为了帮助 PDFlib 用户使用 PDF 的坐标系统，PDFlib 分发中包含 PDF 文件 `grid.pdf`，此文件可让一些普通页面大小的坐标可视化。在透明的材料上打印适当大小的页可能会为准备 PDFlib 开发提供有用的工具。

Acrobat 6 或 7（仅限完全版本，而不是免费的 Reader）也带有一个有用的实用工具。在 Acrobat 7.0（标准版）中，只需选择“视图”->“导航标签”->“信息”，即可显示测量调板。请注意，显示的坐标参照页面左上角的原点，而不是 PDF 默认的左下角的原点。若想改变显示单位，选择“编辑”->“首选项”->“单位”，在 pica、点、毫米、厘米和英寸中选择单位。您也可以到“视图”->“导航标签”->“信息”，在“选项”中选择单位。

不要被看起来好像页面尺寸存在错误的 PDF 打印输出误导。由于以下一些常见原因，这些打印输出可能是错误的：

- ▶ 已经在 Acrobat 的打印对话框中选中“缩小以适合打印机页边距”选项，此设置将导致对打印输出进行缩放。
- ▶ 非 PostScript 打印机驱动程序并不总是能够保留所打印对象的精确大小。

旋转对象 必须懂得一点，即一旦对象已经被绘制在页面上，就不能修改。尽管 PDFlib 提供了用于旋转、平移、缩放和斜移坐标系统的函数，这些函数也不能影响页面上现有的对象，而只可以影响随后绘制的对象。可以使用 `fit_textline()`、`fit_image()` 和 `fit_pdi_page()` 函数中的 *orientate* 选项，轻松地将文本、图像和导入的 PDF 页旋转 90 度的整数倍角度。

通过提供常规的坐标转换函数，可以完成任意角度的旋转。以下示例生成一些水平文本并旋转坐标系统以便显示旋转的文本。保存 / 还原嵌套功能使得在完成垂直文本输出后能轻松地在原坐标系统中继续水平文本的输出：

```
oPDF.set_text_pos 50, 600
oPDF.show "This is horizontal text"
textx = oPDF.get_value("textx", 0)           ' determine text position
texty = oPDF.get_value("texty", 0)           ' determine text position

oPDF.save
    oPDF.translate textx, texty                ' move origin to end of text
    oPDF.rotate 45                             ' rotate coordinates
    oPDF.set_text_pos 18, 0                    ' provide for distance from horizontal text
    oPDF.show "rotated text"
oPDF.restore

oPDF.continue_text "horizontal text continues"
```

使用自上而下的坐标 与 PDF 的自下而上的坐标系统不同，一些图形环境使用自上而下的坐标系统，有些开发人员也偏爱这一坐标系统。可以使用 PDFlib 的转换函数轻松地建立这样的坐标系统。不过，由于转换也将影响文本输出，因此需要附加的调用以避免文本显示时出现镜像效果。

为了便于使用自上而下的坐标，PDFlib 支持一个特殊的模式，在这种模式下将对所有相关的坐标进行不同的解释：从原有的以页面左小角为原点 (0, 0)，y 坐标向上增加的 PDF 默认坐标系统；改为以页面左上角为原点 (0, 0)，y 坐标向下增加的 PDF 默认坐标系统。使用 *topdown* 参数可以激活此自上而下的坐标系统：

```
oPDF.set_parameter "topdown", "true"
```

每页可以建立不同的坐标系统，但对 *topdown* 参数的设置不能在页描述内进行（仅可以在页与页之间）。*topdown* 功能是为了让 PDFlib 用户在自上而下的坐标系统中自如地工作而设计的。为了完整性，下面将列出建立自上而下的坐标系统对不同坐标值的影响。

绝对坐标在未经任何修改的情况下在用户坐标系统上使用：

- ▶ 这包括在函数说明中指定为坐标的所有函数参数。一些示例：*moveto()* 中的 *x, y*；*circle()* 中的 *x, y*；*rect()* 中的 *x, y*（但不是 *width* 和 *height*！）；*create\_annotation()* 中的 *llx, lly, urx, ury*。

以下的相对坐标值将在内部修改以匹配自上而下系统：

- ▶ 文本（字体大小为正数）将面向页面顶端；
- ▶ 当手册中谈到到矩形、方框等的左下角时，这将被解释为您在页面上实际看到的图形的左下角；
- ▶ 当指定一个旋转角度时，旋转中心仍是用户坐标系统的原点 (0, 0)。顺时针方向旋转的可视化结果将仍是顺时针方向的。

### 3.2.2 页面大小和坐标限制

**标准页格式** 为了方便 PDFlib 用户，表 3.4 列出了常见的标准页面大小<sup>1</sup>。页面大小的格式名可以被用于 `begin/end_page_ext()` 中的 `width` 和 `height` 选项。它们称作 `<format>.width` 和 `<format>.height`，此处的 `<format>` 是表 3.4 中列出的格式之一（用小写字母表示，例如 `a4.width`）。

表 3.4 以点为单位的常见页面大小尺寸

格式	宽度	高度	格式	宽度	高度	格式	宽度	高度
<i>a0</i>	<i>2380</i>	<i>3368</i>	<i>a4</i>	<i>595</i>	<i>842</i>	<i>letter</i>	<i>612</i>	<i>792</i>
<i>a1</i>	<i>1684</i>	<i>2380</i>	<i>a5</i>	<i>421</i>	<i>595</i>	<i>legal</i>	<i>612</i>	<i>1008</i>
<i>a2</i>	<i>1190</i>	<i>1684</i>	<i>a6</i>	<i>297</i>	<i>421</i>	<i>ledger</i>	<i>1224</i>	<i>792</i>
<i>a3</i>	<i>842</i>	<i>1190</i>	<i>a5</i>	<i>501</i>	<i>709</i>	<i>11x17</i>	<i>792</i>	<i>1224</i>

**页面大小限制** 虽然 PDF 和 PDFlib 没有对可用的页面大小作任何的限制，但是 Acrobat 在页面大小方面却受到设计结构的限制。请注意，其他 PDF 解释程序也许能够处理更大或更小的文档格式。若超出 Acrobat 的页面大小极限，则 PDFlib 将引发一个非致命的异常。Acrobat 的页面大小极限显示在表 3.5 中。

表 3.5 Acrobat 的最大页面尺寸和最小页面尺寸

PDF 浏览器	最小页面尺寸	最大页面尺寸
Acrobat 4 及以上版本	1/24" = 3 pt = 0.106 cm	200" = 14400 pt = 508 cm

**不同的页面大小方框** 尽管许多 PDFlib 开发人员仅指定页面的宽度和高度，但是一些高级应用程序（特别是用于印前工作的应用程序）可能需要指定一个或多个额外的 PDF 的方框项。PDFlib 支持所有 PDF 的方框项。下面的项（在某些环境下可能有用）可以由 PDFlib 客户端指定（从 PDF reference 中获取定义）：

- ▶ **MediaBox**：用于指定页面的宽度和高度，即我们通常所说的页面大小。
- ▶ **CropBox**：页面内容中要被裁剪下的区域；Acrobat 使用此尺寸进行屏幕显示和打印。
- ▶ **TrimBox**：已完成的（可能是已裁剪下的）页面的预期尺寸；
- ▶ **ArtBox**：页面中有意义内容的范围。应用程序软件很少使用它；
- ▶ **BleedBox**：当在生产环境中输出时，页面内容中要被裁剪下的区域。它可能包括附加的出血区域以处理生成过程中的不精确之处。

PDFlib 除了在输出文件中记录这些值以外，将不会使用其中的任何值。默认情况下，PDFlib 根据指定的页面宽度和高度生成 **MediaBox**，而不会生成任何其他项。以下代码片段将启动一个新页，并设置裁剪框的四个值：

```
' start a new page with custom CropBox
oPDF.begin_page_ext 595, 842, "cropbox {10 10 500 800}"
```

**文档中页面的数目** 在 PDFlib 中，对于文档中生成的页面的数目没有限制。PDFlib 生成的 PDF 结构可让 Acrobat 在带有数以万计的页面的文档中高效地导航。

<sup>1</sup> 有关 ISO、日本和美国标准格式的更多信息可以在下面的 URL 上找到：  
[home.inter.net/eds/paper/papersize.html](http://home.inter.net/eds/paper/papersize.html)、[www.cl.cam.ac.uk/~mgk25/iso-paper.html](http://www.cl.cam.ac.uk/~mgk25/iso-paper.html)



输出准确性和坐标范围 PDFlib 的数字输出精确度是根据 PDF 和其支持的环境的要求，并结合对优化文件输出大小的考虑而仔细挑选的。正如表 3.6 中详细描述的一样，PDFlib 的精确度取决于坐标的绝对值。虽然大多数开发人员无需为此担心，但要求苛刻的应用程序在缩放操作中应当注意不要超出 PDF 的内置坐标极限。

表 3.6 输出准确性和坐标范围

绝对值	输出
0 ... 0.000015	0
0.000015 ... 32767.999999	四舍五入为四位小数位数
32768 ... $2^{31}-1$	四舍五入为下一个整数
$\geq 2^{31}$	将引发异常

### 3.2.3 路径

路径是由任意数目的直线、矩形或曲线构成的一种形状。路径可以包含一些断开的部分（称作子路径）。有一些操作可用于路径处理（请参见第 198 页上的第 8.4.6 节“路径绘制和剪切”）：

- ▶ 描边就是使用客户端提供的绘制参数（例如颜色、线宽）沿着路径绘制线条。
- ▶ 填充就是使用客户端提供的填充参数对路径包围的整个区域涂色。
- ▶ 剪切就是以当前剪切区域（默认情况下为页面大小）与路径包围区域的相交区域替换当前剪切区域，从而减少用于后续绘制操作的成像区域。
- ▶ 断然终止路径将导致不可见的路径，不过该路径仍存在于 PDF 文件中。这种需求极少。

在没有应用任何上述操作的情况下构造路径是错误的。PDFlib 的范围界定系统确保客户端遵从此限制。如果您需要设置路径的任何外观属性（例如，颜色、线宽），则必须在开始执行任何绘图操作之前进行设置。这些规则可以总结为“不要在更改路径描述内改变外观”。

仅仅构造路径不会在页面上显示任何内容；您必须填充或描边路径才能获取可视结果：

```
oPDF.moveto 100, 100
oPDF.lineto 200, 100
oPDF.stroke
```

大多数图形函数都利用了当前点的概念，当前点可看作是用于绘图的笔的位置。

### 3.2.4 模板

PDF 中的模板 PDFlib 支持一种在技术上称为“*form XObjects*”的 PDF 功能。不过，由于此术语与交互式表格存在冲突，因此我们将此功能称为“模板”。可以将 PDFlib 模板看作一个离页缓冲区，文本、矢量和图像操作将被重定向到此离页缓冲区中（而不在常规页上操作）。模板完成后，就可以像光栅图像一样使用，并可以在任意页上放置任意次数。与图像一样，模板也可以被几何转换（例如缩放或斜移）。当在多个页面上（或在同一页面上多次）使用同一模板时，用于构造模板的实际 PDF 运算符只需在 PDF 文件中包含一次，因此可减小 PDF 输出文件的大小。模板适用于在一些页面上反复出现的元素，例如保持不变的背景、公司徽标或由 CAD 和地理绘图软件绘制的图形元素。使用模板的其他典型示例包括裁剪和注册标志或自定义亚洲字形。

在 PDFlib 中使用模板 模板仅可以在页面描述以外被定义，但可以在页面描述内使用。不过，模板也可以包含其他模板。但不能包含自身。可以通过使用 `fit_image()` 函数在页面上引用已定义的模板，如同在页面上放置图像一样（请参见第 113 页上的第 5.3 节“放置图像和导入的 PDF 页”）。PDFlib 中的常规模板惯用法如下所示：

```
' define the template
lTemplate = oPDF.begin_template(template_width, template_height);
...place marks on the template using text, vector, and image functions...
oPDF.end_template
...
oPDF.begin_page page_width, page_height
' use the template
oPDF.fit_image lTemplate, 0.0, 0.0, ""
...more page marking operations...
oPDF.end_page
...
oPDF.close_image lTemplate
```

可以在模板上使用所有文本、图形和颜色函数。不过，构造模板下列函数禁止使用：

- ▶ 第 207 页上的第 8.6 节“图像和模板函数”中的函数，但 *fit\_image()* 和 *close\_image()* 除外。由于图像可以在模板定义之外打开，并可以在模板（但是未打开）内自由使用，故这并不是一个很大的限制。
- ▶ 第 225 页上的第 8.50 节“*add\_nameddest()* 的目标选项以及 *create\_action()*、*create\_annotation()*、*create\_bookmark()* 和 *begin/end\_document()* 中的目标选项”中的函数。超文本元素必须总是在所出现的文档页上进行定义，而不能作为模板的一部分生成。

第三方软件中的模板支持 模板（form XObjects）是 PDF 规范的一个不可分的一部分，可以使用 Acrobat 完全查看和打印。不过，不是所有的 PDF 用户都准备好处理此构造。例如，Acrobat 增效工具 Enfocus PitStop 5.0 只能移动模板，而不能访问模板内的单个元素。另一方面，Adobe Illustrator 9 和 10 完全支持模板。

## 3.3 使用颜色

### 3.3.1 颜色和色彩空间

PDFlib 客户端可以指定用于填充和描边路径内部和文本字符的颜色。可以使用几个色彩空间指定颜色：

- ▶ 介于 0（表示黑色）和 1（表示白色）之间的灰度值；
- ▶ RGB 三原色组合，即用于指定红色 (red)、绿色 (green) 和蓝色 (blue) 的百分比的介于 0 和 1 之间的三个值；其中，(0, 0, 0) 表示黑色，(1, 1, 1) 表示白色。
- ▶ 介于 0（表示无颜色）和 1（表示完全色）之间的四个 CMYK 值，分别表示青色 (cyan)、洋红色 (magenta)、黄色 (yellow) 和黑色 (black) 值；其中，(0, 0, 0, 0) 表示白色，(0, 0, 0, 1) 表示黑色。请注意，这不同于 RGB 规范。
- ▶ CIE L\*a\*b\* 色彩空间内独立于设备的颜色由一个在 0-100 范围内的亮度值和两个在 -127 到 128 范围内的颜色值指定（请参见第 53 页上的第 3.3.4 节“色彩管理和 ICC 配置文件”）。
- ▶ 在 ICC 配置文件的帮助下，指定基于 ICC 的颜色（请参见第 53 页上的第 3.3.4 节“色彩管理和 ICC 配置文件”）。
- ▶ 专色（分色色彩空间）：一个预定义或任意命名的自定义颜色表示以上色彩空间的某一颜色；通常用于为打算在使用一种或多种自定义颜色的胶印机上印刷文档作准备。色调值（百分比）范围是从 0（表示无颜色）到 1（表示专色最大亮度）。有关专色名称的列表，请参见第 51 页上的第 3.3.3 节“专色”。
- ▶ 图案：平铺由任意文本、矢量或图像图形组成的对象（请参见第 51 页上的第 3.3.2 节“图案和平滑着色”）。

- ▶ 着色（平滑混合）提供基于另一色彩空间的两种颜色之间的逐渐过渡（请参见第 51 页上的第 3.3.2 节“图案和平滑着色”）。
- ▶ 索引色彩空间本身不是真正的色彩空间，而是其他色彩空间的有效编码。当导入索引（基于调色板）图像时，将自动生成索引色彩空间。

描边和填充操作的默认颜色是黑色。

### 3.3.2 图案和平滑着色

作为对纯色的补充，图案和着色是可以用于填充或描边任意对象的特殊种类的颜色。

图案 图案是任意数目的绘图操作组合成的单一实体。这组对象可以被任意的复制（或平铺）直至整个区域被填充或描边。使用图案涉及以下步骤：

- ▶ 首先，必须在 *begin\_pattern()* 和 *end\_pattern()* 之间定义图案。大多数图形运算符都可以用于定义图案。
- ▶ 由 *begin\_pattern()* 返回的图案句柄可以通过使用 *setcolor()* 将图案设置为当前颜色。

图案定义可能会也可能不会包含其自身的颜色规范，这取决于 *begin\_pattern()* 的 *painttype* 参数。若 *painttype* 为 1，则图案定义包含其自身的颜色规范并且外观总是保持相同；若 *painttype* 为 2，则图案定义禁止包含任何颜色规范。反之，用当前填充或描边颜色进行填充和描边。

注：也可以基于平滑着色定义图案（请参见以下内容）。

平滑着色 平滑着色也称作“颜色混合”或“渐变”，提供从一种颜色到另一种颜色的连续过渡。且这两种颜色必须属于同一色彩空间。PDFlib 支持用于平滑着色的两种不同种类的几何形状：

- ▶ 轴向着色沿直线定义；
- ▶ 辐射着色在两个圆形之间定义。

着色是两种颜色之间的过渡。总是将第一种颜色作为当前填充颜色；第二种颜色在 *shading()* 的 *c1*、*c2*、*c3* 和 *c4* 参数中提供。这些数值将根据第一种颜色的色彩空间 *setcolor()* 的描边做解释。

调用 *shading()* 将返回处理某个着色对象的句柄，可以按以下两种方式使用该句柄：

- ▶ 使用 *shfill()* 填充一个区域。当要填充的对象的几何形状与着色的几何形状相同时，可以使用此方法。与其名称相反，此函数不仅将填充对象内部，而且还会影响外部。使用 *clip()* 可以修改此行为。
- ▶ 定义一个可用于填充更多复杂对象的着色图案。这就需要调用 *shading\_pattern()* 以创建基于着色的图案，并用该图案填充或描边任意对象。

### 3.3.3 专色

PDFlib 支持专色（技术上称为 PDF 中的分色色彩空间，尽管“分色”这个术语通常也与印刷色一起使用），专色可以用于印刷超出印刷混合色范围之外的自定义颜色。专色是根据名称指定的，在 PDF 中，总是附有一种与专色接近但不完全类似的替代颜色。Acrobat 将使用此替代颜色进行屏幕显示和输出到不支持专色的设备（例如办公用的普通打印机）。在印刷时，除了文档中可能使用的任何印刷色之外，还将应用要求的专色。这就需要通过一种称作“分色”的过程对 PDF 文件进行后处理。

注：在已打开“叠印预览”的情况下，使用 Acrobat 5 在屏幕上无法正确显示一些专色。但这些专色可以正确地分色和打印。

PDFlib 支持各种内置专色库，也支持自定义（用户定义的）专色。当使用 `makespotcolor()` 请求一个专色名称时，PDFlib 首先将检查在其内置库中是否可以找到请求的专色。如果能够找到，PDFlib 将对替代颜色使用内置值。否则，将假定该专色为用户定义的颜色，客户端必须提供适当的替代颜色值（通过当前颜色）。专色可以减淡，即可以与 0 和 1 之间的百分比一起使用。

默认情况下，不能使用自定义替代值重新定义内置专色。不过，可以使用 `spotcolorlookup` 参数更改此行为。这对实现与早期应用程序（可能使用了不同的颜色定义）的兼容性，以及对于无法处理 Pantone 颜色的 PDFlib 的 Lab 替代值的工作流来说，会很有用。

当选定 PDF/X 规范等级后，PDFlib 将为内置专色自动生成适合的替代颜色（请参见第 145 页上的第 7.4 节“PDF/X”）。对于自定义专色，用户需要负责提供与选定的 PDF/X 规范等级兼容的替代颜色。

注：内置专色数据和相应的商标已经由 *PDFlib GmbH* 从各自的商标所有者那里得到了授权，可以在 *PDFlib* 软件中使用。

**PANTONE® 专色** PANTONE 专色很有名，并在全世界范围内被广泛使用。PDFlib 完全支持 PANTONE MATCHING SYSTEM®（总计有 20 000 个色板）。可以使用以下数字颜色库中的所有色板名称（括号中提供了一些示例色板名称）：

- ▶ PANTONE solid coated (PANTONE 185 C)
- ▶ PANTONE solid uncoated (PANTONE 185 U)
- ▶ PANTONE solid matte (PANTONE 185 M)
- ▶ PANTONE process coated (PANTONE DS 35-1 C)
- ▶ PANTONE process uncoated (PANTONE DS 35-1 U)
- ▶ PANTONE process coated EURO (PANTONE DE 35-1 C)
- ▶ PANTONE pastel coated (PANTONE 9461 C)
- ▶ PANTONE pastel uncoated (PANTONE 9461 U)
- ▶ PANTONE metallic coated (PANTONE 871 C)
- ▶ PANTONE solid to process coated (PANTONE 185 PC)
- ▶ PANTONE solid to process coated EURO (PANTONE 185 EC)
- ▶ PANTONE hexachrome® coated (PANTONE H 305-1 C)
- ▶ PANTONE hexachrome® uncoated (PANTONE H 305-1 U)
- ▶ PANTONE solid in hexachrome coated (PANTONE 185 HC)



PDFlib 商业用户可以向我们请求一个列出了全部 Pantone 专色名称的文本文件。

专色名称区分大小写；如示例中所示，使用大写字母。也将接受旧式颜色名称前缀 CV、CVV、CVU、CVC 和 CVP，并将其更改为相应的新颜色名称，除非 `preserveoldpantonenames` 参数为 `true`。如示例中所示，必须总是在色板名称中提供 PANTONE 前缀。通常，必须根据以下方案构造 PANTONE 颜色名称：

PANTONE <id> <paperstock>

此处，<id> 是颜色的标识符（例如 185），<paperstock> 是使用的纸张的缩写（例如，C 表示涂层纸）。在构成色板名称的各个部分之间必须使用一个空格字符。在请求一个以 PANTONE 前缀开头的专色名称时，如果该名称不代表一个有效 PANTONE 颜色，则将导致一个非致命的异常（通过将 `warning` 参数设置为 `false` 可以禁止此异常）。以下代码片段演示如何使用色调值为 70% 的 PANTONE 专色：

```
spot = oPDF.makespotcolor("PANTONE 281 U", 0)
oPDF.setcolor "fill", "spot", spot, 0.7, 0, 0
```

注：此处显示的 **PANTONE®** 专色可能与 **PANTONE** 标识的标准不符。请参考最新的 **PANTONE** 专色出版物获取准确的颜色。 **PANTONE®** 以及其他 **Pantone, Inc.** 商标为 **Pantone, Inc.** 所有。版权所有 © **Pantone, Inc.**, 2003。

注： **PANTONE®** 专色在 **PDF/X-1:2001**、 **PDF/X-1a:2001** 和 **PDF/X-1a:2003** 标准模式下不受支持。

**HKS® 专色** **HKS** 专色系统在德国和其他欧洲国家被广泛使用。**PDFlib** 完全支持 **HKS** 专色。可以使用以下数字颜色库 (*Farbfächer*) 中的所有色板名称（括号中提供了一些示例色板名称）：



- ▶ **HKS K (Kunstdruckpapier)** 用于光面艺术纸，88 种专色 (**HKS 43 K**)
- ▶ **HKS N (Naturpapier)** 用于天然纸，86 种专色 (**HKS 43 N**)
- ▶ **HKS E (Endlospapier)** 用于连续静物，88 种专色 (**HKS 43 E**)
- ▶ **HKS Z (Zeitungspapier)** 用于新闻纸，50 种专色 (**HKS 43 Z**)

**PDFlib** 商业用户可以向我们索取一个列出了全部 **HKS** 专色名称的文本文件。

专色名称区分大小写；如示例中所示，使用大写字母。如示例中所示，必须总是在色板名称中提供 **HKS** 前缀。通常，必须根据以下方案构造 **HKS** 专色名称：

**HKS** <id> <paperstock>

此处，<id> 是颜色的标识符（例如 **43**），<paperstock> 是使用的纸张的缩写（例如，**N** 表示天然纸）。在构成色板名称的 **HKS**、<id> 和 <paperstock> 各个部分之间必须使用一个空格字符。在请求一个以 **HKS** 前缀开头的专色名称时，如果该名称不代表一个有效 **HKS** 专色，则将导致一个非致命的异常（通过将 **warning** 参数设置为 **false** 可以禁用此异常）。以下代码片段演示如何使用色调值为 70% 的 **HKS** 颜色：

```
spot = oPDF.makespotcolor("HKS 38 E", 0)
oPDF.setcolor "fill", "spot", spot, 0.7, 0, 0
```

**用户定义的专色** 除了以上详细介绍的内置专色之外，**PDFlib** 还支持自定义专色。可以为这些自定义专色指定一个任意名称（此名称不得与任何内置颜色的名称发生冲突）和一个替代颜色。替代颜色将用于屏幕预览或低质量打印，但是不能用于高质量分色。客户端需要负责为自定义专色提供合适的替代颜色。

**PDFlib** 没有提供单独的为新的专色设置替代颜色的 **PDFlib** 函数，但可使用当前填充颜色完成。除了使用附加调用设置替代颜色以外，定义和使用自定义专色与使用内置专色的方法类似：

```
oPDF.setcolor "fill", "cmyk", 0.2, 1.0, 0.2, 0      ' define alternate CMYK values
spot = oPDF.makespotcolor("CompanyLogo", 0)        ' derive a spot color from it
oPDF.setcolor "fill", "spot", spot, 1, 0, 0         ' set the spot color
```

### 3.3.4 色彩管理和 ICC 配置文件

**PDFlib** 支持一些色彩管理概念，包括独立于设备的颜色、渲染方法和 **ICC** 配置文件。

**独立于设备的 CIE L\*a\*b\* 颜色** 通过对 **setcolor()** 提供色彩空间名称 **lab**，可以在 **CIE 1976 L\*a\*b\*** 色彩空间中指定独立于设备的颜色值。可以用范围在 0-100 的亮度值和两个范围在 -127 到 128 之间的颜色值指定 **L\*a\*b\*** 色彩空间中的颜色。用于 **lab** 色彩空间的光源将为 **D50**（日光 5000K，2° 观察者）

**渲染方法** 虽然 PDFlib 客户端可以指定独立于设备的颜色值，但是特定的输出设备不一定能够准确地再现所需的颜色。在这种情况下，必须通过一个称作“色域压缩”的过程针对折衷方案做出一些妥协，即将颜色的范围减少到特定设备可以再现的一个较小的范围。渲染方法可以用于管理这个过程。通过将 *renderingintent* 参数或选项提供给 *load\_image()*，可以指定各个图像的渲染方法。另外，通过将 *renderingintent* 选项提供给 *create\_gstate()*，可以为文本和矢量图形指定渲染方法。表 3.7 列出了可用的渲染方法及其含义。

表 3.7 渲染方法

渲染方法	说明	典型使用
<i>Auto</i>	不在 PDF 文件中指定任何渲染方法，而改为使用设备的默认渲染方法。这是默认设置。	未知或非特定的使用
<i>AbsoluteColorimetric</i>	未对设备的白点（例如白纸）做出任何校正。色域外的颜色都将映射为设备的色域内最接近的值。	纯色的准确再现；不建议作其他使用。
<i>RelativeColorimetric</i>	颜色数据缩放到设备的色域内，当轻微地移位颜色时，映射到另外一个白点。	矢量图形
<i>Saturation</i>	当颜色值可能被移位时，将保存颜色的饱和度。	商业图形
<i>Perceptual</i>	通过修改色域内和色域外的颜色保留颜色关系，以提供愉悦的外观。	扫描的图像

**ICC 配置文件** International Color Consortium (ICC)<sup>1</sup> 定义了一种用来指定输入和输出设备的颜色特征的文件格式。这些 ICC 配置文件已被作为产业标准，受到所有主要的色彩管理系统和应用程序厂商的支持。在以下一些领域，PDFlib 支持带 ICC 配置文件的颜色管理：

- ▶ 为页面上文本和矢量图形定义基于 ICC 的色彩空间。
- ▶ 处理嵌入到导入的图像文件中的 ICC 配置文件。
- ▶ 将一 ICC 配置文件用于导入的图像（可能会覆盖已嵌入图像中的 ICC 配置文件）。
- ▶ 定义默认色彩空间以便将灰度、RGB 或 CMYK 数据映射到基于 ICC 的色彩空间。
- ▶ 通过外部 ICC 配置文件定义 PDF/X 输出方法。

色彩管理不会更改色彩规范中色彩成分的数目（例如从 RGB 到 CMYK）。

**搜索 ICC 配置文件** PDFlib 将使用提供给 *load\_iccprofile()* 的 *profilename* 参数，按照以下步骤搜索 ICC 配置文件：

- ▶ 若 *profilename* = *sRGB*，PDFlib 将使用其内部的 sRGB 描述文件（请参见以下内容）并终止搜索。
- ▶ 检查在 *ICCProfile* 资源类别中是否有一个名为 *profilename* 的资源。如果有，则在以下步骤中，将其值用作文件名称。如果没有此类资源，则直接使用 *profilename* 作为文件名称。
- ▶ 使用在前一步骤中确定的文件名称结合以下组合来查找磁盘文件：

```
<filename>
<filename>.icc
<filename>.icm
<colordir>/<filename>
<colordir>/<filename>.icc
<colordir>/<filename>.icm
```

在 Windows 2000/XP 上，*colordir* 指定操作系统用来存储特定于设备的 ICC 配置文件的目录（通常是 *C:\WINNT\system32\spool\drivers\color*）。在 Mac OS X 上，将对 *colordir* 尝试以下路径：

1. 请参见 [www.color.org](http://www.color.org)

```
/System/Library/ColorSync/Profiles  
/Library/ColorSync/Profiles  
/Network/Library/ColorSync/Profiles  
~/Library/ColorSync/Profiles
```

在其他系统上，将省略与 *colordir* 有关的步骤。

可接受的 ICC 配置文件 可接受的 ICC 配置文件的类型取决于提供给 *load\_iccprofile()* 的 *usage* 参数：

- ▶ 若 *usage = outputintent*，则仅接受输出设备（打印机）配置文件。
- ▶ 若 *usage = iccbased*，则接受输入设备、显示设备和输出设备（扫描仪、显示器和打印机）配置文件以及色彩空间转换配置文件。可以在灰色、RGB、CMYK 或 Lab 色彩空间中指定这些配置文件。

**sRGB 色彩空间和 sRGB ICC 配置文件** PDFlib 支持称作“sRGB”（正式名称为 IEC 61966-2-1）的产业标准 RGB 色彩空间。sRGB 受到各种软件和硬件厂商的支持，并被广泛用于消费型 RGB 设备的简化颜色管理。这些设备的示例有数码相机和办公设备（如彩色打印机和显示器）。PDFlib 支持 sRGB 色彩空间，并且在内部包含必需的 ICC 配置文件数据。因此，客户端在使用 sRGB 描述文件时无需且不可加入任何其他附加配置。该文件可通过 *load\_iccprofile()*，并设置 *filename = sRGB* 而调用。

在图像中使用嵌入的配置文件（ICC 标记的图像） 有些图像可能包含嵌入的 ICC 配置文件，用于说明图像的颜色值的本质。例如，嵌入的 ICC 配置文件可以说明用于产生图像数据的扫描仪的颜色特征。PDFlib 可以处理 PNG、JPEG 和 TIFF 图像文件格式中嵌入的 ICC 配置文件。若 *honoriccprofile* 选项或参数设置为 *true*（此为默认情况），则图像中嵌入的 ICC 配置文件将从该图像中被提取，并嵌入到 PDF 输出中，以便 Acrobat 将其应用到该图像。有时，此过程称为标记着 ICC 配置文件的图像”。PDFlib 将不会改变图像的像素值。

使用 *image:iccprofile* 参数可获得嵌于图像中的 ICC 配置文件句柄。这对于多个图像应用同一个描述文件的情况会有用。

为了检查未知 ICC 配置文件中色彩成分的数目，请使用 *iccomponents* 参数。

对图像应用外部 ICC 配置文件（标记） 作为使用图像中嵌入的 ICC 配置文件的一种替代方式，也可以通过提供一个配置文件句柄和 *load\_image()* 的 *iccprofile* 选项对单个图像使用外部配置文件。

用于页面说明的基于 ICC 的色彩空间 文本和矢量图像基于 ICC 的色彩空间的颜色值可直接通过配置文件指定。配置 ICC 配置文件句柄提供给 *setcolor: iccprofilegray*、*setcolor: iccprofilergb* 或 *setcolor: iccprofilecmyk* 参数来设置色彩空间。紧接着，可以将基于 ICC 的颜色值连同色彩空间关键字 *iccbasedgray*、*iccbasedrgb* 或 *iccbasedcmyk* 提供给 *setcolor()*：

```
icchandle = oPDF.load_iccprofile(...)  
If (icchandle = -1) Then  
    ...  
End If  
oPDF.set_value "setcolor:iccprofilecmyk", icchandle  
oPDF.setcolor "fill", "iccbasedcmyk", 0, 1, 0, 0
```

将设备颜色映射到基于 ICC 的默认色彩空间 PDF 提供了一个功能，可以将页面说明中的设备相关的灰色、RGB 或 CMYK 颜色映射到与设备无关的颜色。即对与设备相关的颜色附加精确的比色规范。这一颜色值映射可通过提供 *DefaultGray*、*DefaultRGB* 或 *DefaultCMYK* 色彩空间定义来实现。在 PDFlib 中，可以通过设置 *defaultgray*、*defaultrgb* 或 *defaultcmyk* 参数并

提供 ICC 配置文件句柄作为相应的值，完成此操作。以下示例将 sRGB 色彩空间设置为文本、图像和矢量图形的默认 RGB 色彩空间：

```
icchandle = oPDF.load_iccprofile("sRGB", "usage=iccbased")
If (icchandle = -1) Then
    return;
End If
oPDF.set_value "defaulttrgb", icchandle
```

定义 PDF/X 输出方法 输出设备（打印机）配置文件可以用于指定 PDF/X 的输出条件。这可以在对 load\_iccprofile() 的调用中提供 usage = outputintent 来实现。有关详细信息，请参见第 145 页上的第 7.4.2 节“生成符合 PDF/X 标准的输出”。

### 3.4 超文本元素

#### 3.4.1 创建超文本元素的示例

本节解释如何创建超文本元素（如书签、表单域和注释）。图 3.2 显示了包含我们将在本节中创建的所有超文本元素的结果文档。该文档包含以下超文本元素：

- ▶ 在页面的右上角，文本 [www.pdflib.com](http://www.pdflib.com) 处有一个不可见的 Web 链接。单击此区域将打开一个相关网页。
- ▶ Web 链接之下有一个文本类型的灰色表单域。通过使用 JavaScript 代码，将自动使用当前日期填充该表单域。
- ▶ 红色图钉包含带有一个附件的注释。单击此图钉将打开附加的文件。
- ▶ 在页面的左下角，有一个带有打印机符号的按钮类型的表单域。单击此按钮将执行 Acrobat 的菜单项“文件”->“打印”。
- ▶ 该导航页包含书签“[Our Paper Planes Catalog](#)”。单击此书签将打开另一页 PDF 文档。

在下一个段落中，我们将详细说明如何使用 PDFlib 创建这些超文本元素。

**Web 链接** 让我们从网站 [www.pdflib.com](http://www.pdflib.com) 的链接开始。此操作分为两步完成。第一步，我们创建 URI 类型的操作（在 Acrobat 中：打开一个 Web 链接）。这将为我们提供一个操作句柄，随后可将该句柄指定给一个或多个超文本元素：

```
web_action = oPDF.create_action("URI", "url http://www.pdflib.com")
```

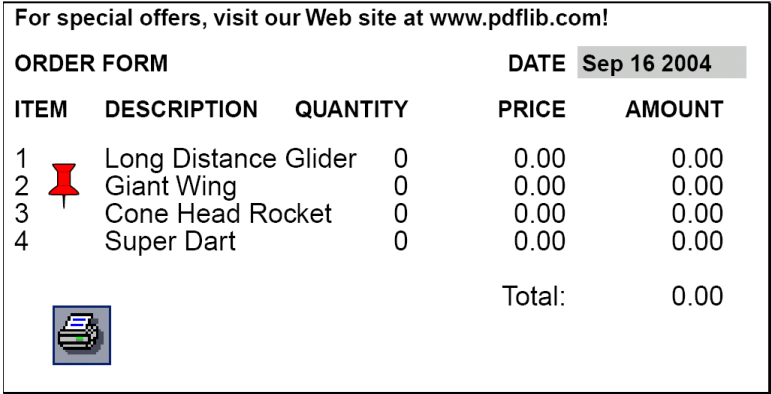


图 3.2  
包含超文本元素的文档



第二步，我们创建实际的链接。PDF 中的链接是一个 *Link* 类型的注释。该链接的 *action* 选项包含将触发操作的事件名称 *activate* 以及上面为操作自身创建的 *web\_action* 句柄：

```
oPDF.create_annotation left_x, left_y, right_x, right_y, "Link", & _  
    "linewidth=0 action {activate " & web_action & "}"
```

默认情况下，将使用一个细的黑色边框显示该链接。最初，准确定位非常方便，但是我们使用 *linewidth=0* 禁用边框。

作为一种选择方式，您可以按如下方式使用 *weblink* 选项调用 *fit\_textline()* 来创建文本输出和链接注释：

```
oPDF.fit_textline "http://www.pdfplib.com", x, y, "position=50 underline " & _  
    "weblink {linewidth=1 annotcolor={rgb 1 0 0}} fillcolor={rgb 0 0 1}"
```

用于跳转到另一文件的书签 现在我们来创建书签 “Our Paper Planes Catalog”，该书签可跳转到另一个名为 *paper\_planes\_catalog.pdf* 的 PDF 文件。首先我们创建一个 *GoToR* 类型的操作。在此操作的选项列表中，我们使用 *filename* 选项定义目标文档的名称；*destination* 选项指定将要扩大的页面的某个部分。更准确地说，文档将使用固定视图 (*type fixed*) 显示在第二页 (*page 2*) 上，在此视图中，页面的中间部分可见 (*left 50 top 200*) 并且缩放因子为 200% (*zoom 2*)：

```
optlist = "filename paper_planes_catalog.pdf " & _  
    "destination {page 2 type fixed left 50 top 200 zoom 2}"
```

```
goto_action = oPDF.create_action("GoToR", optlist)
```

在下一个步骤中，我们创建实际的书签。书签的 *action* 选项包含将触发操作的 *activate* 事件，以及上面为所需操作创建的 *goto\_action* 句柄。选项 *fontstyle bold* 指定粗体文本，*textcolor {rgb 0 0 1}* 使书签成为蓝色的。提供书签文本 “Our Paper Planes Catalog” 作为函数参数：

```
optlist="action {activate " & goto_action & " } fontstyle bold textcolor {rgb 0 0 1}"  
catalog_bookmark=oPDF.create_bookmark("Our Paper Planes Catalog", optlist)
```

单击书签将显示目标文档中的页面的指定部分。

带有文件附件的注释 在下一个示例中，我们创建一个文件附件。我们先创建一个 *FileAttachment* 类型的注释。*filename* 选项指定附件的名称，选项 *mimetype image/gif* 指定附件的类型（MIME 是一种用于给文件内容分类的常用约定）。注释将显示为一个红色 (*annotcolor {rgb 1 0 0}*) 的图钉 (*iconname pushpin*)，并带有一个工具提示 (*contents {Get the Kraxi Paper Plane!}*)。注释不进行打印 (*display noprint*)：

```
optlist = "filename kraxi_logo.gif mimetype image/gif iconname pushpin " & _  
    "annotcolor {rgb 1 0 0} contents {Get the Kraxi Paper Plane!} display noprint"
```

```
oPDF.create_annotation left_x, left_y, right_x, right_y, "FileAttachment", optlist
```

请注意，使用 *iconname* 定义的符号的大小不会变化；图标将按其标准尺寸显示在指定矩形的左上角。

用于打印的按钮表单域 下一个示例创建可用于打印文档的按钮表单域。在第一个版本中，我们将对按钮添加一个标题；在以后的版本中，我们将使用打印机符号，而不用标题。我们先创建一个 *Named* 类型的操作（在 Acrobat 中：执行菜单项）。同样，我们必须指定标题的字体：

```
print_action = oPDF.create_action("Named", "menuname Print")
button_font = oPDF.load_font("Helvetica-Bold", "winansi", "")
```

按钮表单域的 *action* 选项包含作为执行操作的触发器的 *up* 事件（在 Acrobat 中：鼠标松开），以及上面为操作本身创建的 *print\_action* 句柄。*backgroundcolor {rgb 11 0}* 选项指定黄色背景，*bordercolor {rgb 0 0 0}* 指定黑色边框。选项 *caption Print* 将文本 *Print* 添加到按钮上，*tooltip {Print the document}* 为用户创建附加说明。*font* 选项通过使用上面创建的 *button\_font* 句柄指定字体。默认情况下，将调整标题的尺寸以使其完全适合按钮的区域。最后，使用适当的坐标、名称 *print\_button*、类型 *pushbutton* 和合适的选项创建实际的按钮表单域：

```
optlist = "action {up " & print_action & "}" & _
          "backgroundcolor {rgb 11 0} bordercolor {rgb 0 0 0}" & _
          "caption Print tooltip {Print the form} font " & button_font
```

```
oPDF.create_field left_x, left_y, right_x, right_y, "print_button", "pushbutton", optlist
```

现在我们通过用一个小的打印机图标替换文本 *Print* 的方式扩展第一个版本的按钮。为了完成此操作，在创建页之前，我们加载相应的图像文件 *print\_icon.jpg* 作为模板。使用 *icon* 选项，我们可以对按钮域指定模板句柄 *print\_icon*，并创建类似于以上代码的表单域：

```
print_icon = oPDF.load_image("auto", "print_icon.jpg", "template")
If (print_icon = -1) Then
    ' Error handling
End
End If
oPDF.begin_page_ext pagewidth, pageheight, ""
...
optlist = "action {up " & print_action & "}" & _
          "icon " & print_icon & " tooltip {Print the document} font " & button_font
```

```
oPDF.create_field left_x, left_y, right_x, right_y, "print_button", "pushbutton", optlist
```

**简单文本域** 现在我们在页面的右上角附近创建一个文本域。用户将能够在此域中输入当前日期。我们获取一个字体句柄，并创建一个名为 *date* 且带有灰色背景的 *textfield* 类型的表单域：

```
textfield_font = oPDF.load_font("Helvetica-Bold", "winansi", "")
oPDF.create_field left_x, left_y, right_x, right_y, "date", "textfield", _
          "backgroundcolor {gray 0.8} font " & textfield_font
```

默认情况下，字体大小是 *auto*，这意味着域高度最初将用作字体大小。当输入文本达到域的末尾时，将减少字体大小以便使文本总是适合域大小。

带有 **JavaScript** 的文本域 为了改进上面创建的文本表单域，在页面打开的情况下，我们使用当前日期自动填充此文本表单域。首先，我们创建 *JavaScript* 类型的操作（在 Acrobat 中：运行 *JavaScript*）。此操作的选项列表中的 *script* 选项定义一个 JavaScript 片段，该片段在 *date* 文本域中按照“月-日-年”的格式显示当前日期：

```
optlist = "script {var d = util.printd('mmm dd yyyy', new Date()); " & _  
          "var date = this.getField('date'); date.value = d;}"
```

```
show_date = oPDF.create_action("JavaScript", optlist)
```

在第二个步骤中，我们创建页面。在选项列表中，我们提供 *action* 选项将上面创建的 *show\_date* 操作附加到触发器事件 *open*（在 Acrobat 中：进入页面）：

```
optlist = "action {open " & show_date & "}"  
oPDF.begin_page_ext pagewidth, pageheight, optlist
```

最后，我们按照上面的操作创建文本域。只要页面处于打开状态，就将用当前日期自动填充该页面：

```
textfield_font = oPDF.load_font("Helvetica-Bold", "winansi", "")  
oPDF.create_field left_x, left_y, right_x, right_y, "date", "textfield", _  
                  "backgroundcolor {gray 0.8} font " & textfield_font
```

### 3.4.2 文本域的格式化选项

在 Acrobat 中，可以指定用于格式化文本域内容（例如货币金额、日期或百分比）的各种选项。此操作可以通过由 Acrobat 使用的自定义 JavaScript 代码完成。PDFlib 不直接支持这些格式化功能，因为在 PDF 参考中未指定这些功能。不过，为了 PDFlib 用户的利益，我们在下面提供了一些信息，让您能够通过使用 *create\_field()* 的 *action* 选项提供简单 JavaScript 代码片段实现文本域的格式化选项。

为了对文本域应用格式化设置，JavaScript 片段将被附加到文本域作为 *keystroke* 和 *format* 操作。JavaScript 代码调用一些内部 Acrobat 函数，这些函数的参数可控制格式化设置的细节。

以下示例创建两个 *keystroke* 和 *format* 操作，并将它们附加到一个表单域，以便用两个小数位数和 EUR 货币标识符设置域内容的格式：

```
keystroke_action = oPDF.create_action("JavaScript", _  
                                     "script {AFNumber_Keystroke(2, 0, 3, 0, ""EUR "" , true); }")
```

```
format_action = oPDF.create_action("JavaScript", _  
                                   "script {AFNumber_Format(2, 0, 0, 0, ""EUR "" , true); }")
```

```
oPDF.create_field 50, 500, 250, 600, "price", "textfield", "font=" & font & _  
                  " action = {keystroke " & keystroke_action & " format " & format_action & "}"
```

为了指定 Acrobat 中支持的各种格式，您必须在 JavaScript 代码中使用适当的函数。表 3.8 列出了所有支持的格式的 *keystroke* 和 *format* 操作的 JavaScript 函数名称；相应的函数参数在表 3.9 中进行了说明。必须按照以上示例的类似方式使用这些函数。

表单域激活文档的脏标志 当在 Acrobat 中关闭包含表单域的 PDF 文档时，将会询问您是否需要保存该文件，即使您未动过任何域也是如此。从技术上讲，打开一个带有表单域的 PDFlib 生成的 PDF 将会导致设置文档的脏标志，即 Acrobat 将该文档视为已更改。尽管这通常是无关紧要的，因为用户反正需要填充表单域，但是，有一些用户可能会认为这一行为

表 3.8 文本域的 JavaScript 格式化函数

格式	要用于 <b>keystroke</b> 和 <b>format</b> 操作的 <b>JavaScript</b> 函数
数字	<b>AFNumber_Keystroke</b> (nDec, sepStyle, negStyle, currStyle, strCurrency, bCurrencyPrepend) <b>AFNumber_Format</b> (nDec, sepStyle, negStyle, currStyle, strCurrency, bCurrencyPrepend)
百分比	<b>AFPercent_Keystroke</b> (ndec, sepStyle) <b>AFPercent_Format</b> (ndec, sepStyle)
日期	<b>AFDate_KeystrokeEx</b> (cFormat) <b>AFDate_FormatEx</b> (cFormat)
时间	<b>AFTime_Keystroke</b> (tFormat) <b>AFTime_FormatEx</b> (cFormat)
特殊	<b>AFSpecial_Keystroke</b> (psf) <b>AFSpecial_Format</b> (psf)

表 3.9 JavaScript 格式化函数的参数

参数	说明和可能的值	
<b>nDec</b>	小数位数	
<b>sepStyle</b>	小数分隔符样式:	
	0	1,234.56
	1	1234.56
	2	1.234,56
	3	1234,56
<b>negStyle</b>	突出负数的方式:	
	0	正常
	1	使用红色文本
	2	显示括号
	3	显示括号并使用红色文本
<b>strCurrency</b>	要使用的货币字符串, 例如表示欧元符号的 “\u20AC”	
<b>bCurrency-Prepend</b>	<b>false</b>	不预置货币符号
	<b>true</b>	预置货币符号
<b>cFormat</b>	日期格式字符串。它可能包含以下格式占位符或下面为 <b>tFormat</b> 列出的时间格式:	
	<b>d</b>	日
	<b>dd</b>	带有前导零的日
	<b>ddd</b>	缩写的星期
	<b>m</b>	用数字表示的月份
	<b>mm</b>	用数字表示的带有前导零的月份
	<b>mmm</b>	缩写的月份名称
	<b>mmm</b>	完整的月份名称
	<b>yyyy</b>	用四位数字表示的年份
	<b>yy</b>	年份的后两位数字
<b>tFormat</b>	时间格式字符串。它可能包含以下格式占位符:	
	<b>h</b>	小时 (0-12)
	<b>hh</b>	带有前导零的小时 (0-12)
	<b>H</b>	小时 (0-24)
	<b>HH</b>	带有前导零的小时 (0-24)
	<b>M</b>	分钟
	<b>MM</b>	带有前导零的分钟
	<b>s</b>	秒
	<b>ss</b>	带有前导零的秒
	<b>t</b>	'a' 或 'p'
	<b>tt</b>	'am' 或 'pm'

表 3.9 JavaScript 格式化函数的参数（续）

参数	说明和可能的值
<i>psf</i>	描述几种附加格式：
0	邮政编码
1	邮政编码 + 4
2	电话号码
3	身份证号码

不正常，有点烦人。您可以使用一小段 JavaScript 代码在加载文件后重置文档的脏标记，从而解决这个问题。使用以下代码可以完成这一操作：

```
' ...create some form fields...
oPDF.create_field "100, 500, 300, 600, "field1", "textfield", "...

' Create a JavaScript action which will be hooked up in the document
action = oPDF.create_action("JavaScript", "script={this.dirty=false;}")
...
oPDF.end_document "action={open " & action & "}"
```



# 4 文本处理

## 4.1 字体和编码概述

字体处理是类似于 PDF 这样的页面描述和文档格式的一个最复杂的方面。在本节中，我们将总结 PDFlib 的有关字体和编码处理的主要特征（编码是指各个字节或字节组合与其实际表示的字符之间的映射）。除了另有说明之外，PDFlib 在所有平台上都支持相同的字体格式。

### 4.1.1 支持的字体格式

PDFlib 支持多种字体类型。本节总结 PDFlib 支持的字体类型，并对这些格式中的一些最重要的方面加以说明。

**PostScript Type 1 字体** PostScript 字体可以按照多种文件格式打包，并通常附有一个单独的文件，在其中包含字体规格和其他字体相关信息。PDFlib 支持 Mac 和 Windows PostScript 字体，以及用于 PostScript 字型 and 规格数据的所有常见文件格式。

**TrueType 字体** PDFlib 支持基于矢量的 TrueType 字体，但是不支持基于位图的 TrueType 字体。TrueType 字体文件必须用 Windows TTF 或 TTC 格式提供，或必须安装在 Mac 或 Windows 操作系统上。与 PostScript Type 1 字体不同的是，TrueType 和 OpenType 字体不需要任何附加的规格文件，这是因为相应的字体文件中自身已包含规格信息。

**OpenType 字体** OpenType 是一种现代字体格式，融合了 PostScript 和 TrueType 这两种技术，并使用独立于平台的文件格式。OpenType 字体有两种风格，且均受到 PDFlib 的支持：

- ▶ 具有 TrueType 字型的 OpenType 字体 (\*.ttf) 的外观与常见的 TrueType 字体一样。
- ▶ 具有 PostScript 字型的 OpenType 字体 (\*.otf) 包含类似于 TrueType 的文件格式的 PostScript 数据。此风格也称作 CFF (*Compact Font Format*)。

**中文、日文和韩文 (CJK) 字体** 除了 Acrobat 的标准 CJK 字体之外（请参见第 84 页上的第 4.7 节“中文、日文和韩文文本”），PDFlib 还支持 TrueType 和 OpenType 格式的自定义 CJK 字体。通常，这些字体的处理方式类似于西文字体。不过，存在某些限制。

**Type 3 字体** 除了 PostScript、TrueType 和 OpenType 字体之外，PDFlib 还支持用户定义的 (Type 3) PDF 字体概念。与常见字体格式不同，用户定义的字体不是从某个外部源（字体文件或操作系统服务）获取的，而是必须借助于 PDFlib 的本机文本、图形和图像函数由客户端完全定义。Type 3 字体可用于以下目的：

- ▶ 位图字体。
- ▶ 自定义图形，例如可以使用简单的文本运算符轻松打印徽标。
- ▶ 在任何预定义的字体或编码中都不包含日文 gaiji（用户定义的字符）。

### 4.1.2 编码

编码定义 PDFlib 和 Acrobat 将如何解释字符串中的实际字节，以及如何将这些字节转换成页面上的文本。PDFlib 支持多种编码方法。

所有受支持的编码都可以在一个文档中任意混合。您甚至可以对单个字体使用不同的编码，尽管这种情况很少出现。

注：并非所有编码都可以用于某一给定的字体。用户需要负责确保该字体包含某一特定编码所需的所有字符。这甚至对 Acrobat 的核心字体都是一个问题（请参见表 4.2）。

标识字形 有三种不同的基本方法用于标识某一字体中的各个字形（字符的表现方式）：

- ▶ **PostScript Type 1** 字体都基于字型名称的概念：每个字形都使用可用来标识字符的唯一名称进行标记，并构造适合某个环境的代码映射。尽管字形名称在相当一段时间内达到了其目的，但是由于其空间要求以及没有真正符合国际使用要求（在特殊的 CJK 字体中），它们给现代计算带来了很大的限制。
- ▶ **TrueType 和 OpenType** 字体标识基于其 **Unicode** 值的各个字形。这样就可以对文本字体中的所有字形轻松地添加清晰的语义。不过，没有用于 pi 或符号字体的标准 **Unicode** 分配。当在 **Unicode** 环境中使用符号字体时，这预示着会有一些困难。
- ▶ 中文、日文和韩文 **OpenType** 字体都基于字符 **ID (CID)** 的概念。这些是为相应的语言引用标准知识库（称作字符补集）的基本数字。

这些概念之间存在相当大的重叠。例如，由于一些兼容性原因，**TrueType** 字体可能包含 **PostScript** 字形名称的辅助表。另一方面，一些标准 **PostScript** 字形名称的 **Unicode** 语义可在 **Adobe Glyph List (AGL)** 中找到。**PDFlib** 支持所有三种方法（基于名称的、**Unicode**、**CID**）。

**8 位编码** 8 位编码（也称作单字节编码）将文本字符串中的每个字节映射到单个字符，因此每次最多 256 个不同字符。在 **PDFlib** 中使用的 8 位编码基于字形名称或 **Unicode** 值，并可以从各种源提取：

- ▶ 表 4.2 中列出的大量预定义的编码。这些编码涵盖了目前在多种系统上和多个区域中使用的最重要的编码。
- ▶ 可以在外部文件中提供或使用 `encoding_set_char()` 在运行时动态构造的用户定义的编码。这些编码可以基于字形名称或 **Unicode** 值。
- ▶ 源自操作系统的编码，也称作“系统编码”。此功能仅在 **Windows**、**IBM eServer iSeries** 和 **zSeries** 上可用。
- ▶ 简写的基于 **Unicode** 的编码，可以用于方便地使用 8 位值对任何 256 个连续字符的 **Unicode** 范围进行寻址。
- ▶ 与某一特殊字体相关的编码。这些编码也称作 *font-specific* 或 *builtin* 编码。

**宽字符寻址** 除了 8 位编码之外，也支持其他各种功能更为强大且不需遵从 256 个字符限制的寻址方案。

- ▶ 通过 *unicode* 编码关键字可以完全基于 **Unicode** 的寻址。在这种情况下，客户端直接为 **PDFlib** 提供 **Unicode** 字符串。可以根据一些标准方法（例如 **UTF-16**、**UTF-8**）和字节顺序（*little-endian* 或 *big-endian*）格式化 **Unicode** 字符串。
- ▶ 各种中文、日文和韩文标准的基于 **CMap** 的寻址。通过与标准 CJK 字体结合，**PDFlib** 支持 **Acrobat** 所支持的所有 **CMap**。这包括基于 **Unicode** 的 **CMap** 和其他 **CMap**（请参见第 84 页上的第 4.7 节“中文、日文和韩文文本”）。
- ▶ 通过 *glyphid* 编码关键字的 **TrueType** 和 **OpenType** 字体的字形 **ID** 寻址。这对高级文本处理应用程序非常有用，因为这类应用程序需要不通过参考任何特殊编码方案即可访问字体的各个字形，或必须对没有任何 **Unicode** 映射的字形进行寻址。可以使用 *fontmaxcode* 参数查询字体中的有效字形 **ID** 的数目。

### 4.1.3 支持 **Unicode** 标准

**Unicode** 是一个大的字符集，涵盖了世界上所有现行语言和脚本以及许多早期的语言和脚本，并对许多应用程序、操作系统和编程语言提供有效支持。**PDFlib** 在很大范围上支持 **Unicode** 标准。**PDFlib** 中的下面一些功能是支持 **Unicode** 的：

- ▶ 可以直接在页面说明中提供 **Unicode**。
- ▶ 可以为多种超文本元素提供 **Unicode**。



- ▶ 可以使用带任何字节顺序的 UTF-8 或 UTF-16 格式提供页面上的文本或超文本元素的 Unicode 字符串。
- ▶ PDFlib 将在 PDF 输出中包括其他信息 (*ToUnicode CMap*)，以便帮助 Acrobat 为导出文本（例如通过剪贴板）和搜索 Unicode 文本指定适当的 Unicode 值。

## 4.2 字体格式详细信息

### 4.2.1 Windows 和 Mac 上的宿主字体支持

在 Mac 和 Windows 系统上，PDFlib 可以访问已经安装在操作系统上的 TrueType、OpenType 和 PostScript 字体。我们将这样的字体称为“宿主字体”。只需在系统中安装字体（通常是将字体放入适当的目录中），而不用手动配置字体文件，PDFlib 就可使用相应的字体了。

当使用宿主字体时，使用准确的（区分大小写）字体名称很重要。由于字体名称非常重要，以下我们会提及一些用于不同平台的确定字体名称的特有方法。有关字体名称的更多信息可以在第 66 页上的第 4.2.2 节“PostScript 字体”和第 67 页上的第 4.2.3 节“TrueType 和 OpenType 字体”中找到。

**查找 Windows 上的宿主字体名称** 可以通过如下方法轻松找到已安装的字体的名称：双击字体文件，然后在显示的窗口的第一行中显示即可留意到字体全名。一些字体的名称的部分内容已根据所使用的相应的 Windows 版本进行了本地化。例如，常见字体名称部分 **Bold** 可能在德语系统上作为转换后的单词 **Fett** 出现。为了检索 Windows 系统中的宿主字体数据，必须使用 PDFlib 中字体名称的已转换的形式（如：**Arial Fett**），或使用字体样式名称（请参见第 67 页上的“TrueType 和 OpenType 字体的 Windows 字体样式名称”）。不过，为了直接从文件中检索字体数据，必须使用字体名称的一般形式（未本地化，如：**Arial Bold**）。

若需要更加详细地检查 TrueType 字体，可以参考 Microsoft 免费的“字体属性扩展”<sup>1</sup>，其中将按照用户可读的形式显示字体的 TrueType 表的许多项。

**查找 Mac 上的宿主字体名称** 使用 **Font Book** 实用工具（Mac OS X 的一部分），可以找到已安装的宿主字体的名称。不过，在某些情况下，Font Book 不会显示 PDFlib 所需的字体的正确的 QuickDraw 名称。

为此，我们推荐 Apple 的可免费获得的 Font Tools<sup>2</sup>。此命令行实用工具套件包含一个称作 `ftxinstalledfonts` 的程序，该程序用于确定所有已安装的字体的确切 QuickDraw 名称。为了确定 PDFlib 需要的字体名称，请安装 Font Tols 并在终端窗口中发出以下命令语句：

```
ftxinstalledfonts -q
```

**Windows 上有关宿主字体访问的潜在问题** 我们提请用户注意一个与在 Windows 上安装字体有关的潜在问题。若您通过“文件”->“安装新字体”菜单项（另一种对应的方法是将字体拖动到 **Fonts** 目录中）安装字体，将出现一个复选框“复制字体到 **Fonts** 文件夹”。若未选中此框，Windows 将在字体文件夹中仅放置一个指向原始字体文件的快捷方式（链接）。在这种情况下，原始字体文件必须位于一个应用程序可使用 PDFlib 访问的目录中。具体而言，位于 Windows 的 **Fonts** 目录之外的字体文件对于使用默认安全设置的 IIS 可能是不可访问的。解决方案：将字体文件复制到 **Fonts** 目录下，或将原始字体文件放置到 IIS 对其具有 **read** 权限的目录中。

在使用 Adobe Type Manager (ATM) 时，如果在安装字体时选中了“添加时不复制字体”选项，则会出现类似的问题。

1. 请参见 [www.microsoft.com/typography/TrueTypeProperty21.msp](http://www.microsoft.com/typography/TrueTypeProperty21.msp)

2. 请参见 [developer.apple.com/fonts/OSXTools.html](http://developer.apple.com/fonts/OSXTools.html)

Mac 上有关宿主字体访问的潜在问题 在测试中我们发现，对于无用户界面的应用程序（例如 PDFlib）而言，新安装的字体有时无法访问；用户需从控制台中注销然后重新登录方可正常使用。

## 4.2.2 PostScript 字体

**PostScript 字体文件格式** PostScript Type 1 字体总是分为两个部分：实际字型数据和规格信息。PDFlib 在所有平台上都支持以下用于 PostScript Type 1 字型和规格数据的文件格式：

- ▶ 用于规格信息的独立于平台的 AFM（Adobe 字体规格）和特定于 Windows 的 PFM（打印机字体规格）格式。虽然基于 AFM 的字体规格可以对字体支持的任何编码重新安排，但是一般的西文 PFM 字体规格文件（代码页 1252）仅可以与以下编码一起使用：*auto*、*winansi*、*iso8859-1*、*unicode* 和 *ebcdic*。符号字体的 PFM 文件可使用 *builtin* 编码。其他代码页的 PFM 文件可使用与 PFM 代码页（或其任何子集）相匹配的编码，也可选择 FM 的内部代码页或 *unicode* 作为 *builtin* 编码。例如，一个西里尔字体的 PFM 可使用 *cp1250*、*builtin* 或 *unicode* 编码。
- ▶ 用于 PostScript Type 1 格式（有时也称作“ATM 字体”）的字型信息的、独立于平台的 PFA（打印机字体 ASCII）和特定于 Windows 的 PFB（打印机字体二进制）格式。
- ▶ 在 Mac 上，也支持基于资源的 PostScript Type 1 字体，即 LWFN（LaserWriter 字体）字型字体。这些字体附有一个字体包（FOND 资源或 FFIL），其中包含了规格数据（以及 PDFlib 将忽略的屏幕字体）。PostScript 宿主字体可以与以下一些编码一起使用：*auto*、*macroman*、*macroman\_apple*、*unicode* 和 *builtin*。不过，对于一些需要字形补充的字体，可能不可以接受 *macroman* 和 *macroman\_apple*。  
当使用 PostScript 宿主字体时，LWFN 文件必须放置在与字体包相同的目录中，并且必须按照 5+3+3 的规则进行命名。请注意，在 PDFlib 的无 *CarbonLib* 支持的 OS 9 版本中不支持 PostScript 宿主字体。
- ▶ 使用 PostScript 字型 (\*.otf) 的 OpenType 字体。

**PostScript 字体名称** 若正在使用基于磁盘的字体文件，则可以使用任意别名（请参见第 69 页上的第 4.3.1 节“PDFlib 搜索字体的方法”）。可以使用以下几种方法查找 PostScript 字体的准确名称：

- ▶ 打开字型文件 (\*.pfa 或 \*.pfb)，并查找项 /FontName 后面的字符串。忽略此项的最前面的 / 字符，然后使用其余部分作为字体名称。
- ▶ 若正在使用 Windows 2000/XP 或 Mac OS X 10.4，可以双击字体文件，即可显示字体的 PostScript 名称以及相应的字体示例。
- ▶ 打开 AFM 规格文件并查找项 FontName 后面的字符串。

注：PostScript 字体名称可能与 Windows 字体菜单名称完全不同，例如“AvantGarde-Demi”（PostScript 名称）与“AvantGarde, Bold”（Windows 字体菜单名称）。另外，任何 Windows .inf 文件中给定的字体名称与使用 PDF 无关。

**PostScript 字形名称** 为了编写自定义编码文件或查找可以与某一提供的编码一同使用的字体，将必须查找有关编码定义的字符集的准确定义，以及字体文件中使用的准确的字形名称。还必须确保选择的字体提供了用于编码的所有必需的字符。例如，与 Acrobat 4/5 一起提供的核心字体不支持 ISO 8859-2 (Latin 2)，也不支持 Windows 代码页 1250。若您正巧有 FontLab<sup>1</sup> 字体编辑器（顺便说一下，此编辑器是用于处理所有种类的字体和编码问题的有效工具），则可以使用此编辑器查找有关某一给定字体所支持的编码的信息（在 FontLab 文档中查找代码页）。<sup>2</sup>

1. 请参见 [www.fontlab.com](http://www.fontlab.com)

为了方便 PDFlib 用户，分发文件集中的 PostScript 程序 *print\_glyphs.ps* 可以用于查找 PostScript 字体中包含的所有字符的名称。为了使用该程序，可在 PostScript 文件的末尾输入字体的名称并将其（连同字体一起）发送到 PostScript 打印机，然后使用 Acrobat Distiller 转换此名称，或者使用 PostScript 查看器查看此名称。该程序将打印字体中的所有字形，并按照字形名称的字母顺序排序。

若字体不包含自定义编码必需的字形，则相应的字符将从 PDF 文档缺失。

### 4.2.3 TrueType 和 OpenType 字体

**TrueType 和 OpenType 文件格式** TT 和 OT 字体文件都是自我包含的：它们在单个文件中包含所有必需的文件。PDFlib 支持 TrueType 和 OpenType 字体的以下文件格式：

- ▶ Windows TrueType 字体 (\*.ttf)，包括 CJK 字体。
- ▶ 使用 TrueType (\*.ttf) 或 PostScript 字型 (\*.otf) 的独立于平台的 OpenType 字体，包括 CJK 字体。
- ▶ 在单个文件中带有多个字体的 TrueType 集合 (\*.ttc)（主要用于 CJK 字体）。
- ▶ 终端用户定义的字符 (EUDC) 字体 (\*.tte)，由 Microsoft 的 *eudcedit.exe* 工具创建。
- ▶ 在 Mac 上，系统上安装的任何 TrueType 字体（包括 .dfont）也可在 PDFlib 中使用。

**TrueType 和 OpenType 字体名称** 若正在使用基于磁盘的字体文件，则可以使用任意别名（请参见第 69 页上的第 4.3.1 节“PDFlib 搜索字体的方法”）。在生成的 PDF 中，TrueType 字体的名称可能与在 PDFlib（或 Windows）中使用的名称不同。这种情况很正常，这是由于 PDF 使用 TrueType 字体的 PostScript 名称导致的，PostScript 名称与其原来的 TrueType 名称不同（例如 *TimesNewRomanPSMT* 与 *Times New Roman*）。

注：与 PostScript 字体相反，TrueType 和 OpenType 字体名称可能包含空字符。

**TrueType 和 OpenType 字体的 Windows 字体样式名称** 加载 Windows 操作系统的宿主字体时，PDFlib 用户可以访问 Windows 字体选择机制提供的功能：可以为 TrueType 或 OpenType 字体的粗细度或倾斜度提供样式名称，例如

Georgia,Bold

这将指示 Windows 搜索特殊的粗体、斜体或基本字体的其他变体。根据可用的字体，Windows 将选择与所请求的样式最为接近的字体（Windows 将不会创建新的字体变体）。Windows 找到的字体可能与所请求的字体不同，生成的 PDF 中的字体名称也可能与请求的名称不同；PDFlib 在 Windows 的字体选择上没有任何控制权。另外，字体样式名称仅可用于 TrueType 和 OpenType 宿主字体，而不可用于 PostScript 宿主字体或通过基于磁盘的字体文件配置的字体。

以下关键字（与字体名称之间用逗号分隔）可以附加到提供给 *load\_font()* 的基字体名称以指定字体粗细度：

none, thin, extralight, ultralight, light, normal, regular, medium, semibold, demibold, bold, extrabold, ultrabold, heavy, black

可以使用以下关键字替换上述关键字或附加到它们后面：

italic

这些关键字不区分大小写。若使用两种样式名称，则两者必须用逗号分开，例如：

2. 有关 PostScript 字体中使用的字形名称的信息可以在 [partners.adobe.com/asn/tech/type/unicodegn.jsp](http://partners.adobe.com/asn/tech/type/unicodegn.jsp) 网页上找到（并不要求字体供应商遵从这些字形命名建议）。

注：若必须处理经过本地化的字体名称，则字体的 **Windows** 样式名称可能会有用，因为这些 **Windows** 样式名称提供了访问字体变体的通用方法，而不需要考虑其本地化的名称。

#### 4.2.4 用户定义的 (Type 3) 字体

PDF 中的 Type 3 字体（与 PostScript Type 3 字体相对）实际上不是一种文件格式。Type 3 字体中的字形必须在运行时使用标准 PDFlib 图形函数进行定义。由于矢量图形、光栅图像甚至文本输出的所有 PDFlib 功能都可以在 Type 3 字体定义中使用，所以对于使用 Type 3 字体的字符内容没有任何限制。通过与 PDF 导入库 PDI 配合使用，甚至可以导入复杂的绘图作为 PDF 页，并使用这些绘图定义 Type 3 字体的字符。

注：PostScript Type 3 字体不支持。

必须完全在任何页面的外部定义 Type 3 字体（更确切地说，字体定义必须在“文档”范围中进行）。以下示例演示简单的 Type 3 字体的定义：

```
oPDF.begin_font "Fuzzyfont", 0.001, 0.0, 0.0, 0.001, 0.0, 0.0, ""

oPDF.begin_glyph "circle", 1000, 0, 0, 1000, 1000
oPDF.arc 500, 500, 500, 0, 360
oPDF.fill
oPDF.end_glyph

oPDF.begin_glyph "ring", 400, 0, 0, 400, 400
oPDF.arc 200, 200, 200, 0, 360
oPDF.stroke
oPDF.end_glyph

oPDF.end_font
```

字体将在 PDFlib 中注册，并且字体的名称可以连同同一个编码（包含 Type 3 字体中字形的名称）一起提供给 `load_font()`。在使用 Type 3 字体时，请注意以下内容：

- ▶ 与模式和模板类似，图像不能够在字形说明内打开。不过，在启动字形说明之前，可以打开图像并将其置于字形说明内。作为一种选择，内嵌图像可以用于小的位图以克服此限制。
- ▶ 由于 PDF 使用者方面的限制，所有在文本输出的字符必须确切地被定义在字体中：若字符代码 *x* 要使用 `show()` 或一个相似函数显示，并且编码包含位置 *x* 处的 *glyphname*，则必须通过 `begin_glyph()` 定义 *glyphname*。此限制仅影响 Type 3 字体；将完全忽略 PostScript Type 1、TrueType 或 OpenType 字体中的缺失字形。
- ▶ 如果要对字体中没有定义的相应字形名称使用代码，则一些 PDF 使用者（不包括 Acrobat）会要求一个名为 *.notdef* 的字形。*.notdef* 字形必须存在，但是它可以包含一个空的字形说明。
- ▶ 当用普通位图数据定义字符时，位图中未使用的像素将不管背景如何都会用白色输出。为了避免上述情况发生，显示原始的背景色，可使用 *mask* 参数构造位图图像。
- ▶ 图像的 *interpolate* 选项可能对增强 Type 3 位图字体的屏幕和打印外观很有用。

## 4.3 字体嵌入和子集化

### 4.3.1 PDFlib 搜索字体的方法

字体数据源 PDFlib 可以访问多种源中的字体数据：

- ▶ 基于磁盘的字体文件可通过 UPR 配置文件（请参见第 42 页上的第 3.1.5 节“资源配置和文件搜索”）静态配置的或通过 `set_parameter()` 和 `FontOutline` 资源类别动态配置。
- ▶ 尽管不需要任何配置就可以使用宿主字体（请参见第 65 页上的第 4.2.1 节“Windows 和 Mac 上的宿主字体支持”），但还是可以使用 `HostFont` UPR 资源类别明确配置宿主字体来管理搜索顺序。例如，此功能可以用于指定搜索宿主字体优先于内建核心字体。
- ▶ 客户端通过 PDFlib 虚拟文件 (PVF) 直接在内存中传递的字体数据。对于已在内存中加载字体数据并希望避免不必要的 PDFlib 磁盘访问（有关虚拟文件的详细信息，请参见第 42 页上的第 3.1.4 节“PDFlib 虚拟文件系统 (PVF)”）的高级应用程序，这一点很有用。

字体名称的别名使用 由于字体的准确内部名称难于查找，PDFlib 支持对 PostScript、TrueType 和 OpenType 字体的字体名称使用别名。通过使用字体名称的别名，可以为某个字体指定任意名称作为别名。在 UPR 文件中或在运行时，均可以为 `HostFont`、`FontOutline`、`FontAFM` 和 `FontPFM` 类型的资源指定别名。以下示例为基于磁盘的字体定义一个别名：

```
oPDF.set_parameter "FontOutline", "x=DFHSMincho-W3.ttf"
font = oPDF.load_font("x", "winansi", "")
```

搜索字体 提供给 `load_font()` 的字体名称是一个名称字符串（请参见第 77 页上的第 4.5.2 节“内容字符串、超文本字符串和名称字符串”）。然而，不是所有编码都支持任何字体源。根据以下方案搜索字体：

- ▶ 若名称是一个别名（通过 UPR 文件或调用 `set_parameter()` 进行配置），则可用 ASCII 或 UTF-8 编码。别名所引用的名称在后面的步骤中将用来查找字体文件（基于磁盘的字体）或宿主字体。
- ▶ 若名称指定一个宿主字体，则可以用 ASCII 对其编码。在 Windows 上，也可以使用 Unicode。
- ▶ 若发现字体不是（可能已本地化）宿主字体，并且未用 Unicode 编码，则将通过应用下面说明的基于扩展名的搜索方式搜索相应的字体文件。
- ▶ 对于 TTC（TrueType 集合）字体，可以使用 ASCII 或 Unicode 对名称进行编码，并将对照 TTC 文件中的所有字体的所有名称匹配这一名称。

基于磁盘的字体文件的基于扩展名的搜索 当 PDFlib 搜索磁盘上字体的字型或规格文件时（与从操作系统直接获取宿主字体不同），如果字体名称由纯 ASCII 字符组成，则 PDFlib 使用以下搜索算法：

- ▶ 当通过 UPR 文件或在运行时已经将字体配置为 `FontAFM`、`FontPFM` 或 `FontOutline` 资源时，将使用已配置的文件名称。
- ▶ 若无法找到文件，则将以下后缀添加到字体名称，然后尝试利用得到的文件名称一个一个地查找字体规格（对于 TrueType 和 OpenType 字体则查找字型）：

```
.ttf .otf .afm .pfm .ttc .tte
.TTF .OTF .AFM .PFM .TTC .TTE
```

- ▶ 若 PostScript 字体要求嵌入，则将以下后缀添加到字体名称，然后尝试利用得到的文件名称一个一个地查找字型文件：

```
.pfa .pfb
.PFA .PFB
```

- ▶ 以上所有的尝试文件名都会按原样搜索，并预先考虑 *SearchPath* 资源类别中配置的所有目录名称。

这意味着，只要相应的字体文件中包含字体名称和对应于字体类型的标准文件名后缀，并且该文件位于任一个 *SearchPath* 目录中，则不需要任何手动配置，PDFlib 就将查找相应的字体。

### 4.3.2 字体嵌入

**PDF 核心字体** PDF 浏览器支持含有 14 种字体的核心集（假定这些字体永远可用）。核心字体的完整规格信息已经构建到 PDFlib 二进制文件中，因此不再需要其他字体文件（除非要嵌入字体）。核心字体如下所示：

*Courier, Courier-Bold, Courier-Oblique, Courier-BoldOblique, Helvetica, Helvetica-Bold, Helvetica-Oblique, Helvetica-BoldOblique, Times-Roman, Times-Bold, Times-Italic, Times-BoldItalic, Symbol, ZapfDingbats*

注：PDFlib 当前只包含 Acrobat 4 和 5 中使用的传统核心字体的规格。Acrobat 6 以及更高版本在核心字体中支持更多字符（例如一些波兰语字符），而 PDFlib 不包含这些字符的正确字形宽度。

为了使用系统上已安装的宿主字体替换一种核心字体，必须在 *HostFont* 资源类别中配置相应的字体。例如，以下代码行确保将使用主机系统上的 *Symbol* 字体，而不使用内建核心字体数据：

```
oPDF.set_parameter "HostFont", "Symbol=Symbol"
```

PDF 通过一些方法支持除 14 种核心字体之外的字体。PDFlib 能够将字型嵌入到生成的 PDF 输出中。通过 *load\_font()* 的 *embedding* 选项可以控制字体嵌入（尽管在某些情况下，PDFlib 将强制实施字体嵌入）：

另外，也可以嵌入仅包含字符规格和有关字体的常规信息（不带实际字型）的字体说明符。若某种字体未嵌入到 PDF 文档中，Acrobat 将从目标系统获取相应的字体（如果存在），或者根据字体说明符构造替代字体。表 4.1 列出有关字体用法的不同情况，每种用法都对 PDFlib 必需的字体和规格文件提出了不同的要求。

*oPDF.set\_parameter "HostFont", "Symbol=Symbol"* 当字体带有局限于该字体的编码（符号字体）或包含 Adobe 的标准拉丁字符集之外的字形的字体时，但在 PDF 输出中没有嵌入这种字体，则只有当该字体已经安装在目标系统上产生的 PDF 才将可用（因为 Acrobat 只能模拟拉丁文字体）。这类 PDF 文件天生就不可移植，不过可以在受控环境下使用（例如，公司内部文档交换）。

表 4.1 不同的字体用法情况以及要求的规格和字型文件

字体用法	必须明确配置字体规格文件吗？	必须明确配置字型文件吗？
14 种核心字体之一	否	仅在要求嵌入的情况下需要
安装在 Mac 或 Windows 系统上的 TrueType、OpenType 或 PostScript Type 1 宿主字体。	否	否
非核心 PostScript 字体	是	仅在要求嵌入的情况下需要

表 4.1 不同的字体用法情况以及要求的规格和字型文件

字体用法	必须明确配置字体规格文件吗？	必须明确配置字型文件吗？
TrueType 字体	不适用	是
OpenType 字体，包括 CJK TrueType 和 OpenType 字体	不适用	是
标准 CJK 字体 <sup>1</sup>	否	否

1. 有关 CJK 字体的更多信息，请参见第 84 页上的第 4.7 节“中文、日文和韩文文本”。

强制的字体嵌入 PDF 需要为某些字体和编码的组合进行字体嵌入。因此，在以下情况下 PDFlib 将强制字体嵌入（与 *embedding* 选项无关）：

- ▶ 与具有 TT 字型的 TrueType 或 OpenType 字体一起使用 *glyphid* 或 *unicode* 编码。
- ▶ 与不同于 *winansi*、*macroman* 和 *ebcdic* 的编码一起使用具有 TrueType 字型的 TrueType 字体或 OpenType 字体。

请注意，对于具有 PostScript 字型的 OpenType 字体，将不会强制实施字体嵌入。字体嵌入的需求是由针对 CID 字体的内部转换引起的，可以通过将 *autocidfont* 参数设置为 *false* 来禁用此转换。这样做也将禁用强制嵌入。请注意，在这种情况下，并非所有拉丁字符都将可以访问，Adobe 字形列表 (AGL) 之外的字符根本不会起作用。

有关字体嵌入的法律事宜 请注意，仅拥有字体文件也许没有资格在 PDF 中嵌入字体，即使对于合法的字体许可的所有者也是一样。许多字体供应商都会限制他人嵌入其字体。一些字体厂商完全禁止 PDF 字体嵌入，其他厂商对于其字体提供特殊的在线或嵌入许可，同时仍有其他一些厂商在对字体应用于集化的前提下允许字体嵌入。在试图使用 PDFlib 嵌入字体之前，请检查字体嵌入是否合法。PDFlib 将遵从在 TrueType 或 OpenType 字体中可能指定的嵌入限制。若 TrueType 字体中的嵌入标记设置为 *no embedding*<sup>1</sup>，则 PDFlib 将尊重字体供应商的要求，从而拒绝有关嵌入相应字体的任何尝试。

### 4.3.3 字体子集化

为了减少 PDF 输出的大小，PDFlib 可以仅嵌入字体中的那些在文档中实际使用的字符。这个过程称作“字体子集化”。此过程将创建一个新的字体，其中包含的字形比原始字体包含的字形要少一些，并将省略 PDF 查看不需要的字体信息。不过，请注意，Acrobat 的 TouchUp 工具将拒绝处理使用子集字体的文本。字体子集化对于 CJK 字体特别重要。PDFlib 支持以下类型的字体的子集化：

- ▶ TrueType 字体。
- ▶ 具有 PostScript 或 TrueType 字型的 OpenType 字体。

当已请求子集化的字体在文档中使用时，PDFlib 将记录实际用于文本输出的字符。对于子集化行为有一些控制方式：

- ▶ 默认的子集化行为由 *autosubsetting* 参数控制。若该参数为 *true*，则将对所有可能发生产子集化的字体启用子集化。默认值为 *true*。
- ▶ 若 *autosubsetting* 参数为 *false*，而某一特殊字体需要子集化，则 *subsetting* 选项仍必须提供给 *load\_font()*：

```
font = oPDF.load_font("WarnockPro", "winansi", "subsetting")
```

- ▶ *subsetlimit* 参数包含一个百分比值。若文档使用的值大于字体中的此字形百分比值，则将对特殊字体禁用子集化，而改为嵌入完整字体。这将节省一些处理时间，不过输出文件会较大：

1. 更具体而言：若字体的 OS/2 表中的 *fsType* 标记的值为 2。

```
oPDF.set_value "subsetlimit", 75 ' set subset limit to 75%
```

*subsetlimit* 的默认值是 100%。换言之，将遵从 *load\_font()* 请求的 *subsetting* 选项，除非客户端显式请求一个低于 100% 的限制。

- ▶ *subsetminsize* 参数可以用于对小型字体完全禁用子集化。若原始字体文件小于 *subsetminsize* 的值（以 KB 为单位），则将禁用此字体的子集化。默认值是 100 KB。

**嵌入和子集化 TrueType 字体** 由于 PDF 中的某些要求，导致用于 TrueType 处理的依赖项有点混乱。以下是前面几个段落中的信息的小结。

若将 TrueType 字体与不同于 *winansi* 和 *macroman* 的编码一同使用，默认情况下，该字体将转换为用于 PDF 输出的 CID 字体。对于仅包含 Adobe 字形列表 (AGL) 中的字符的编码，可以通过将 *autocidfont* 参数设置为 *false* 阻止此转换。若字体转换为 CID 字体，它将总是嵌入的。默认情况下，将应用于子集化，除非 *autosubsetting* 参数设置为 *false*，或使用的字形的百分比高于 *subsetlimit* 参数，或字体文件大小（以 KB 为单位）小于 *subsetminsize* 参数的值。

## 4.4 编码详细信息

### 4.4.1 8 位编码

表 4.2 列出了 PDFlib 中预定义的编码，并详细介绍了如何将这些编码与一些重要类别的字体一起使用。某些脚本或语言有一些普通字体无法满足的要求，认识到这一点很重要。例如，Acrobat 的核心字体未包含 ISO 8859-2 必需的所有字符（例如波兰语），而 PostScript 3、OpenType Pro 和 TrueType 的大字体中包含了所有这些字符。

注：PDFlib 分发中包含的 *chartab* 示例可以用于轻松打印任意字体 / 编码组合的字符表。

**有关 macroman 编码的批注** 此编码反映 Mac OS 字符集，尽管在 219 = 0xDB 位置处带有旧的货币符号，而不是由 Apple 重新定义的欧元符号字形（PDF 规范指出此不兼容性）。除了以下一些差异之外，*macroman\_apple* 编码与 *macroman* 一致：

- ▶ *macroman\_apple* 中的 219 = 0xDB 位置处保留欧元符号字形，而不是货币符号。
- ▶ *macroman\_apple* 编码包括 Mac OS 字符集中定义的希腊 / 数学符号。虽然这些符号在 *macroman\_apple* 编码中可用，但只是在少数字体中包含所需的字形。

**宿主编码** 尽管此特殊的“宿主”编码没有任何固定的含意，但是将按如下方式根据当前平台映射为其他 8 位编码：

- ▶ 在 Mac OS Classic 上，它将映射为 *macroman*；
- ▶ 在带有 MVS 或 USS 的 IBM eServer zSeries 上，它将映射为 *ebcdic*；
- ▶ 在 IBM eServer iSeries 上，它将映射为 *ebcdic\_37*；
- ▶ 在 Windows 上，它将映射为 *winansi*；
- ▶ 在所有其他系统上（包括 Mac OS X），它将映射为 *iso8859-1*；

宿主编码主要用于编写独立于平台的测试程序（如 PDFlib 分发中包含的程序和其他简单应用程序）。建议不要在生产中使用宿主编码，而应使用任何适当的编码进行替代。

**自动编码** PDFlib 支持可用于为某些环境指定最自然的编码的机制，而不用再费周折。提供关键字 *auto* 作为编码名称，可以为以下文本字体指定特定于平台和环境的 8 位编码：

- ▶ 在 Windows 上：当前系统代码页（有关详细信息，请参见以下内容）
- ▶ 在 Unix 和 Mac OS X 上：*iso8859-1*（Mac 上的 LWFN PostScript 字体除外，此处 *auto* 将映射为 *macroman*）



表 4.2 用于预定义编码的字形在一些类别的字体中的可用性：一些语言无法用 Acrobat 的核心字体表示

代码页	受支持的语言	PS Level 1/2, Acrobat 4/5 <sup>1</sup>	Acrobat 6/7 <sup>2</sup> 核心字体	PostScript 3 字体 <sup>3</sup>	OpenType Pro 字体 <sup>4</sup>	TrueType 大字体 <sup>5</sup>
winansi	与 cp1252 相同（iso8859-1 的超集）	是	是	是	是	是
macroman	Mac Roman 编码，原始 Macintosh 字符集	是	是	是	是	是
macroman_apple	类似于 macroman，但替换了欧元货币符号，并包括其他数字 / 希腊符号	-	-	-	是	是
ebcdic	EBCDIC 代码页 1047	是	是	是	是	是
ebcdic_37	EBCDIC 代码页 037	是	是	是	是	是
pdfdoc	PDFDocEncoding	是	是	是	是	是
iso8859-1	(Latin-1) 西欧语言	是	是	是	是	是
iso8859-2	(Latin-2) 中欧斯拉夫语	-	-	是	是	是
iso8859-3	(Latin-3) 世界语、马耳他语	-	-	-	是	是
iso8859-4	(Latin-4) 爱沙尼亚语、波罗的语、格陵兰语	-	-	-	是	是
iso8859-5	保加利亚语、俄语、塞尔维亚语	-	-	-	是	是
iso8859-6	阿拉伯语	-	-	-	-	是
iso8859-7	现代希腊语	-	-	-	1 缺失。	是
iso8859-8	希伯来语和依地语	-	-	-	-	是
iso8859-9	(Latin-5) 西欧语言、土耳其语	5 缺失。	5 缺失。	是	是	是
iso8859-10	(Latin-6) 北欧语言	-	-	-	1 缺失。	是
iso8859-13	(Latin-7) 波罗的语	-	-	是	是	是
iso8859-14	(Latin-8) 凯尔特语	-	-	-	-	-
iso8859-15	(Latin-9) 将欧元符号、法语和芬兰语字符添加到 Latin-1	欧元符号缺失。	是	是	是	是
iso8859-16	(Latin-10) 匈牙利语、波兰语、罗马尼亚语、斯洛文尼亚语	-	-	是	是	是
cp1250	中欧语言	-	-	是	是	是
cp1251	古斯拉夫语	-	-	-	是	是
cp1252	西欧语言（与 winansi 相同）	是	是	是	是	是
cp1253	希腊语	-	-	-	1 缺失。	是
cp1254	土耳其语	5 缺失。	-	是	是	是
cp1255	希伯来语	-	-	-	-	是
cp1256	阿拉伯语	-	-	-	-	5 缺失。
cp1257	波罗的语	-	-	是	是	是
cp1258	越南语	-	-	-	-	是

1. 随 Acrobat 4/5 附带的核心字体（原始的 Adobe 拉丁字符集；自 1982 年起通常为 Type 1 字体）。  
2. Acrobat 6 和 7 依赖在系统上可用的字体以显示 Times 和 Helvetica。因此，结果会根据已安装的字体的数目和种类改变。例如，随 Windows XP 附带的系统字体包含的字形比早期版本的 Windows 中可用的字形多。  
3. 扩展的 Adobe 拉丁字符集 (CE-Fonts)，通常是随 PostScript 3 设备附带的 Type 1 字体。  
4. Adobe OpenType Pro 字体包含的字形比常规 OpenType 字体包含的字形多。  
5. Windows TrueType 字体包含大字形增补集，例如 Tahoma。

► 在 Mac OS Classic 上：macroman

- ▶ 在 IBM eServer iSeries 上: 当前的工作代码 (*IBMCCSID000000000000*)
- ▶ 在 IBM eServer zSeries 上: *ebcdic* (= 代码页 1047)

对于符号字体, 关键字 *auto* 将映射为 *builtin* 编码。虽然自动编码在许多环境下使用起来很方便, 但是使用这种方法将使您的 PDFlib 客户端程序天生不可移植。

用户定义的 8 位编码 除了预定义编码之外, PDFlib 支持用户定义的 8 位编码。若需要处理在 PDFlib 内部不可用的字符集 (例如, 与 PDFlib 内部支持的字符集不同的 EBCDIC 字符集), 则可以使用这些方法。PDFlib 既支持按照 PostScript 字形名称定义的编码表, 也支持按照 Unicode 值定义的表。

在用户定义的编码可以在 PDFlib 程序中使用之前, 必须完成以下任务 (或者, 也可以使用 *encoding\_set\_char()* 在运行时构造编码):

- ▶ 用简单的文本格式生成编码的说明。
- ▶ 在 PDFlib 资源文件中配置编码 (请参见第 42 页上的第 3.1.5 节 “资源配置和文件搜索”) 或通过 *set\_parameter()* 配置编码。
- ▶ 提供一种支持编码中使用的所有字符的字体 (规格和可能的字型文件)。

编码文件只是逐行列出字形名称和编号。以下节选显示了编码定义的开头部分:

```
% Encoding definition for PDFlib, based on glyph names
% name      code    Unicode (optional)
space       32      0x0020
exclam      33      0x0021
...
```

下面的示例显示了 Unicode 代码页的一个片段:

```
% Code page definition for PDFlib, based on Unicode values
% Unicode    code
0x0020       32
0x0021       33
...
```

更合乎规范的做法是按照以下规则管理编码或代码页文件的内容:

- ▶ 注释由百分比 “%” 字符引入, 并在行的末尾终止。
- ▶ 每一行的第一项是 PostScript 字形名称, 或是一个由 *ox* 前缀和四位数的十六进制数字 (大写或小写) 组成的十六进制 Unicode 值。其后紧跟空白和一个十六进制 (*ox00–oxFF*) 或十进制 (*0–255*) 字符代码。有时, 基于名称的编码文件可能包含带有相应 Unicode 值的第三列。
- ▶ 编码文件中未提及的字符代码假定为未定义的代码。或者, 可以将 Unicode 值 *oxoooo* 或字符名称 *.notdef* 提供给未使用的插槽。

按照命名约定, 我们将基于名称的表称为编码文件 (*\*.enc*), 而将基于 Unicode 的表称为代码页文件 (*\*.cpg*), 但 PDFlib 会以相同的方式处理这两种文件, 并不在意文件名称。实际上, PDFlib 将在基于名称的代码页文件和基于 Unicode 的代码页文件之间自动转换 (必要时)。此转换基于 Adobe 的 PostScript 字形名称标准列表 (Adobe 字形列表或 AGL<sup>1</sup>), 但是也可以使用非 AGL 名称。PDFlib 将为非 AGL 名称分配可用的 Unicode 值, 并在读取包含从字形名称到 Unicode 值的映射的 OpenType 字体文件时, 调整这些值。

AGL 已被构建到 PDFlib 中, 其中包含 1000 多种字形名称。带有非标准字形名称的 PostScript 字体需要编码文件, 而在处理基于 Unicode 的 TrueType 或 OpenType 字体时, 使用代码页更为方便。

1. AGL 可以在 [partners.adobe.com/public/developer/en/opentype/glyphlist.txt](http://partners.adobe.com/public/developer/en/opentype/glyphlist.txt) 上找到

## 4.4.2 符号字体和特定于字体的编码

由于符号或徽标字体（也称作 Pi 字体）通常未包含标准字符，因此它们必须使用与文本字体不同的编码方案。

**PostScript** 字体的 **builtin** 编码。编码名称 *builtin* 不是指特殊的字符顺序，而是指按原样使用此字体，而不要与字符集混在一起。这个概念有时也称作“特定于字体的编码”，这对于非文本字体（例如，徽标和符号字体）非常重要。它也广泛地用于（有点不适当地）非拉丁文本字体（例如希腊语和古斯拉夫语）。由于这些字符名称与这些编码中的字符名称不匹配，因此使用任一种标准编码都无法重新编码这些字体。因此，*builtin* 必须用于所有符号性或非文本 **PostScript** 字体。在 **AFM** 文件中，以下项可以识别非文本字体：

EncodingScheme FontSpecific

可以对文本字体重新编码（调整到某个代码页或字符集），但不可以对符号字体重新编码，而必须改用 *builtin* 编码。不过，广泛使用的 *Symbol* 和 *ZapfDingbats* 字体也可以与 *unicode* 编码一起使用。

*builtin* 编码无法用于用户定义的 (**Type 3**) 字体，因为这些字体未包含任何默认编码。

注：不幸的是，过去许多印刷商和字体供应商没有完全理解特定于字体的编码的概念（这可能是由于一些不甚完美的生产工具导致的）。为此，产生了许多标记为 *FontSpecific* 编码的拉丁文本字体以及许多错误地标记为文本字体的符号字体。

**TrueType** 字体的 **builtin** 编码 带有非文本字符的 **TrueType** 字体（例如 *Wingdings* 字体）必须与 *builtin* 编码一起使用。若字体要求 *builtin* 编码，但客户端请求不同的编码，**PDFlib** 仍将强制执行 *builtin* 编码。

具有 **PostScript** 字型 (\*.otf) 的 **OpenType** 字体的 **builtin** 编码 带有非文本字符的 **OTF** 字体必须与 *builtin* 编码一起使用。一些 **OTF** 字体包含一个内部默认编码。**PDFlib** 将检测这种情况，并动态构建适合此特殊的字体的编码。编码名称 *builtin* 将被修改为 *builtin\_<fontname>*。虽然此新的编码名称可以在将来对 *load\_font()* 的调用中使用，但只有与相同的字体使用才合理。

## 4.4.3 TrueType 和 OpenType 字体的字形 ID 寻址

除了 8 位编码、Unicode 和 CMap 之外，**PDFlib** 还支持一种在字体内针对各个字符进行寻址的方法，此方法称作“字形 ID 寻址”。为了使用此技术，必须满足以下所有要求：

- 字体在 **TrueType** 或 **OpenType** 格式中可用。
- 字体必须嵌入到 **PDF** 文档中（使用或不使用子集化）。
- 开发人员熟悉字体内的字形的内部编号。

字形 ID (*GID*) 在 **TrueType** 和 **OpenType** 字体中内部使用，并唯一地在字体中确定各个字形的地址。*GID* 寻址功能使开发人员免受给定的编码方案中的任何限制，并提供对已被字体设计人员放入到字体文件中的所有字形的访问权。不过，通常在 *GID* 和许多常见寻址方案之间根本没有任何关系，例如 **Windows** 编码或 **Unicode**。**PDFlib** 用户负责将特定于应用程序的代码转换为 *GID*。

通过提供关键字 *glyphid* 作为 *load\_font()* 的 *encoding* 参数，调用 *GID* 寻址。可以使用从 0 到最后的字形 ID 值之间的值为 *GID* 连续编号，最后的字形 ID 值可以使用 *fontmaxcode* 参数进行查询。

### 4.4.4 欧元符号字形

此符号表示欧洲货币，当需要正确显示和打印符号时，欧元符号会引发许多问题。在本节中，我们将提供一些提示以便您可以成功处理欧元字符。首先，您将必须选择包含欧元字符的编码并检查欧元字符的位置。请看一些示例：



- 在 *unicode* 编码中使用字符 **U+20AC**。
- 在 *winansi* 编码中，位置是 **0x80**（十六进制）或 **128**（十进制）。
- 普通 *iso8859-1* 编码不包含欧元字符。不过，*iso8859-15* 编码是 *iso8859-1* 的扩展，其中在 **0xA4**（十六进制）或 **164**（十进制）处添加了欧元字符。
- 原始的 *macroman* 编码不包含欧元字符。不过，Apple 修改了此编码并用位于 **0xDB**（十六进制）或 **219**（十进制）处的欧元字形替换旧的货币字形。为了使用此已修改的 Mac 编码，请使用 *macroman\_apple*，而不使用 *macroman*。

下一步，您必须选择包含欧元字形的字体。许多现代字体包含欧元字形，但是并不是所有字体都包含。同样，请看一些示例：

- **PostScript Level 1** 和 **Level 2** 设备中的内建字体不包含欧元字符，而 **PostScript 3** 设备中的内建字体通常包含欧元字符。
- 若字体未包含欧元字符，则可以使用 **Symbol** 核心字体的欧元符号，此欧元符号位于 **0xA0**（十六进制）或 **160**（十进制）。在 **Acrobat 4.0** 及更高版本的软件附带的 **Symbol** 字体的版本中带有该字符，并构建到 **PostScript 3** 设备内部。

## 4.5 Unicode 支持

PDFlib 支持 Unicode 标准<sup>1</sup>（几乎与 ISO 10646 相同）这包括对页面内容和超文本元素相关的多种功能的支持。



### 4.5.1 页面内容和超文本的 Unicode

可以在页面说明中直接提供 Unicode 字符串以便使用以下种类的字

- 带有 *unicode* 编码的 **PostScript** 字体。最多有 **255** 个不同的 **Unicode** 值可供使用。若请求更多的值，则它们将被空格字符替换。若字体带有 **PFM** 规格文件或编码 *unicode* 将总是映射为 *winansi*（若使用的字体）和 *macroman*（若使用 Mac 上的 **PostScript** 宿主字体）。
- 带有 *unicode* 编码的 **TrueType** 和 **OpenType** 字体。对于 **TrueType** 和 **OpenType** 字体，将强制执行字体嵌入。
- 带有基于 **Unicode** 的 **CMap** 的标准 **CJK** 字体。**Unicode** 兼容的 **CMap** 可以根据其名称中的 *Uni* 前缀轻松地进行识别（请参见表 4.5）。
- 带有 *unicode* 编码的自定义 **CJK** 字体。
- 在 **Windows** 系统上，可以使用 **Unicode** 文件名称。

除了 *unicode* 编码之外，PDFlib 支持用于选择 **Unicode** 字符的一些其他方法。

**PostScript** 字体和 **TrueType** 字体的 **Unicode** 代码页 PDFlib 支持对 Adobe 字形列表 (AGL) 内的字符进行 **Unicode** 寻址。此类 **Unicode** 支持可用于带有 AGL 中的字形名称的、基于 **Unicode** 的 **TrueType** 字体和 **PostScript** 字体。

可以通过使用任一 PDFlib 的内部代码页或通过提供合适的自定义编码或代码页文件激活此功能（请参见第 72 页上的第 4.4.1 节“8 位编码”）。

1. 请参见 [www.unicode.org](http://www.unicode.org)

用于为 **Unicode** 片段寻址的 **8 位字符串** PDFlib 支持一种缩写格式，该格式可用于为 256 个连续 **Unicode** 字符寻址（在 U+0000 和 U+FFFF 之间的任意偏移量处开始）。这可以用于轻松访问小范围的 **Unicode** 字符，而且仍使用 **8 位字符串**。

可以通过使用字符串 **U+XXXX** 作为 **load\_font()** 的 **encoding** 参数激活此功能，此处 **XXXX** 表示一个十六进制的偏移量。将 **8 位字符串** 添加到提供的偏移量。例如，使用编码

U+0400

将选择古斯拉夫语 **Unicode** 节，提供给文本函数的 **8 位字符串** 将选择 **Unicode** 字符 U+0400、U+0401 等。

用于剪贴和查找操作的 **Unicode** 值 PDFlib 将在 **PDF** 输出中包含额外信息 (**ToUnicode CMap**)，以便帮助 **Acrobat** 为导出文本（例如，通过剪贴板）和搜索文本指定合适的 **Unicode** 值。默认情况下，将为所有支持的字体类型生成 **ToUnicode CMap**，其前提是 **Unicode** 信息可用于给定的字体 / 编码组合。尽管对大多数字体 / 编码组合都是这样，但是有些字体（例如，用户定义的 **Type 3** 字体）可能会缺失 **Unicode** 信息。在这种情况下，PDFlib 将不能生成 **ToUnicode CMap**，文本导出或搜索将在 **Acrobat** 中不起作用。使用 **unicodemap** 参数可以全局禁用 **ToUnicode CMap** 的生成，或者在每个字体的基础上使用相同名称的 **load\_font()** 选项禁用 **ToUnicode CMap** 的生成。此参数 / 选项的默认值为 **true**。将此值设置为 **false** 将减少输出文件的大小，但在 **Acrobat** 中可能会禁用剪切 / 粘贴支持。

4.5.2 内容字符串、超文本字符串和名称字符串

根据字符串用法的不同，PDFlib API 中字符串分为不同的类型：

- ▶ 内容字符串：这些字符串将用于根据用户为特定的字体选定的编码创建真正的页内容（页面说明）。第 176 页上的第 8.3.4 节“简单文本输出”和第 181 页上的第 8.3.5 节“使用文本流的多行文本”中的页面内容函数的所有 **text** 参数都归为此类。
- ▶ 超文本字符串：这些字符串多数用于超文本函数，例如书签和注释，并在函数说明中明确标记为 **Hypertext string**。第 222 页上的第 8.9 节“超文本函数”中的许多函数的参数和选项与一些其他项都归为此类。

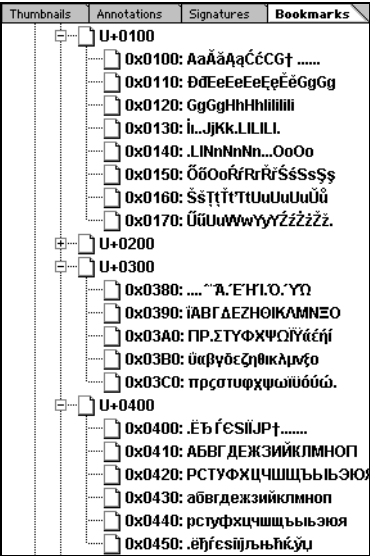


图 4.1  
Unicode 书签（左侧）和  
Unicode 文本批注（右侧）



- ▶ 名称字符串：这些字符串用于外部文件名称、字体名称、块名称等，并在函数说明中标记为 *name string*。

带有不可用字形的 **Unicode** 码位的替换机制 内容字符串将用特定的字体显示在页面上。不过，不存在包含所有最新 **Unicode** 标准中包含的字符的字体。尽管获取合适的字体显然是 **PDFlib** 用户的任务，但是，如果原始字形在字体中不可用，且 *glyphwarning* 选项设置为 *false*，则 **PDFlib** 会尝试通过使用可见的相似字形替换某些字符来解决一些常见问题。以下列表（不完整）包含一些字形映射。若列表中的第一个字符在字体中不可用，则使用第二个字符自动替换它：

U+00A0（非分行空格）	U+0020（空格）
U+00AD（自动连字符）	U+002D（减号连字符）
U+2010（连字符）	U+002D（减号连字符）
U+03BC（希腊小写字母 MU）	U+00C5（微符号）
U+212B（埃符号）	U+00B5（上部带有一个圆圈的拉丁大写字母 A Å）
U+220F（元积）	U+03A0（希腊大写字母 PI）
U+2126（欧姆符号）	U+03A9（希腊大写字母 OMEGA）

除了内建表之外，若全宽变体在字体中不可用，则使用相应的 **ISO 8859-1** 字符（即 U+0021 到 U+007E）替换从 U+FF01 到 U+FF5E 区间的全宽字符。

### 4.5.3 支持 **Unicode** 的语言中的字符串处理

以下 **PDFlib** 语言绑定支持 **Unicode**：

- ▶ **COM**
- ▶ **.NET**
- ▶ **Java**
- ▶ **REALbasic**
- ▶ **Tcl**

在这些环境中将直接处理字符串：所有字符串都将作为 **UTF-16** 格式的 **Unicode** 字符串自动提供给 **PDFlib** 内核。语言包装程序将正确处理由客户端提供的 **Unicode** 字符串，并自动设置某些 **PDFlib** 参数。这将产生以下结果：

- ▶ 由于语言包装程序自动设置 *textformat*、*hypertextformat* 和 *hypertextencoding* 参数，所以客户端无法访问这些包装程序，并禁止使用这些包装程序。**PDFlib** 语言包装程序使用所有必需的转换，以便客户端提供的超文本字符串将按照 *utf16* 格式和 *unicode* 编码传递给 **PDFlib**。
- ▶ 由于语言环境总是按照 **UTF-16** 格式将字符串传递给 **PDFlib**，所以 **UTF-8** 无法与支持 **Unicode** 的语言一起使用。在使用环境提供的本地方法之前，必须将其转换为 **UTF-16**。
- ▶ 处理识别 **Unicode** 的语言中的编码有一个最简单的方法，就是对页面内容使用 *unicode* 编码。
- ▶ 在页面说明中必须避免对标准 **CJK** 字体使用非 **Unicode CMap**，这是因为包装程序总是将 **Unicode** 提供给 **PDFlib** 核心；只可以使用 **Unicode CMap**。

总的效果是，客户端可以将纯 **Unicode** 字符串提供给 **PDFlib** 函数，而不需要任何其他配置或参数设置。

### 4.5.4 不支持 **Unicode** 的语言中的字符串处理

（只有通用版的 **PDFlib** 手册中包括此小节。）

- ▶
- ▶

## 4.5.5 字符引用

一些环境要求程序员使用 8 位编码编写源代码（例如 *winansi*、*macroman* 或 *ebcdic*）。在没有将文本中的所有字符更改为多字节编码的情况下，这使得很难在 8 位编码的文本中包含独立的 Unicode 字符。在这种情况下为了给开发人员提供帮助，PDFlib 支持字符引用，这是标记语言（例如，SGML 和 HTML）中已知的一种方法。

**HTML 样式字符引用** PDFlib 支持 HTML 4.0<sup>1</sup> 中定义的所有数字字符引用和字符实体引用。可以用十进制或十六进制表示法为字符的 Unicode 值提供数字字符引用。

注：码位 128-159（十进制）或 0x80-0x9F（十六进制）不引用 *winansi* 码位。在 *Unicode* 中，它们不引用可打印字符，而仅引用控制字符。

以下是有效的字符引用以及对得到的字符的说明的示例：

&#173;	自动连字符
&#xAD;	自动连字符
&shy;	自动连字符
&#229;	上方带有一个小圆圈的字母 a（十进制）
&#xE5;	上方带有一个小圆圈的字母 a（十六进制，小写 x）
&#Xe5;	上方带有一个小圆圈的字母 a（十六进制，大写 X）
&#x20AC;	欧元字形（十六进制）
&#8364;	欧元字形（十进制）
&euro;	欧元字形（实体名）
&lt;	小于号
&gt;	大于号
&amp;	& 符号
&Alpha;	希腊字母 Alpha

注：尽管可以使用字符引用来引用任何 *Unicode* 字符（例如希腊字符和数学字符），但是字体不会自动转换。若当前字体中未包含指定的字符，为了实际使用这些字符，必须明确选择合适的字体。

**Textflow 中用于控制字符的附加引用** 除了以上的 HTML 样式引用之外，PDFlib 还支持自定义字符实体引用，用以指定 **textflow** 的控制字符。表 4.3 列出了这些附加的字符引用。

**使用字符引用** 字符引用可以在所有的内容字符串、超文本字符串和名称字符串中使用，例如在使用 *show* 或 *textflow* 函数放置在页上的文本中使用，以及在提供给超文本函数的文本中使用。注意，在使用 *builtin* 编码的文本中将不会处理字符引用，这一点很重要。

默认情况下，将不会转换字符引用；如果需要在所有内容字符串中使用字符引用，则必须将 *charref* 参数明确设置为 *true*：

```
oPDF.set_parameter "charref", "true"
```

通过将 *charref* 选项提供给 *create\_textflow()*（直接提供或作为内嵌选项提供）、*fit\_textline()* 或 *fill\_textblock()*，也可以为 *textflow* 处理启用字符引用。

当启用字符引用时，可以在 8 位编码的文本中提供数字引用或实体引用：

```
oPDF.set_parameter "charref", "true"
font = oPDF.load_font("Helvetica", 0, "unicode", "")
oPDF.setfont font, 24
oPDF.show_xy "Price: 500&euro;", 50, 500
```

1. 请参见 [www.w3.org/TR/REC-html40/charset.html#h-5.3](http://www.w3.org/TR/REC-html40/charset.html#h-5.3)

表 4.3 *Textflow* 中的控制字符及其含意

Unicode 字符 (VB 等效项)	实体名称	相当于 <i>textflow</i> 选项	在使用 <i>Unicode</i> 兼容字体的 <i>textflow</i> 中的含意
U+0020	SP, space	space	对齐单词并分行
U+00A0	NBSP, nbsp	(none)	(非分行空格) 将不分行的空格字符
U+0009 (VbTab)	HT, hortab	(none)	水平制表位: 将根据 <i>ruler</i> 、 <i>tabalignchar</i> 和 <i>tabalignment</i> 选项进行处理
U+002D	HY, hyphen	(none)	用于断词的分隔符字符
U+00AD	SHY, shy	(none)	(自动连字符) 提供断词的机会, 仅在分行符处可见
U+000B U+2028	VT, verttab LS, linesep	nextline	(新行) 强制一个新行
U+000A (VbLf) U+000D (VbCr) U+000D and U+000A (VbCrLf) U+0085 U+2029	LF, linefeed CR, return CRLF NEL, newline PS, parasep	nextparagraph	(新段落) 与 “新行” 效果相同; 另外, <i>parindent</i> 选项将影响新行。
U+000C (VbFormFeed)	FF, formfeed	return	<i>fit_textflow()</i> 将停止, 然后返回字符串 <i>_nextpage</i> 。

在选项列表中将不会替换字符引用, 但是将在选项中使用 *Unichar* 数据类型识别字符引用 (请参见第 157 页上的第 8.1.2 节 “选项列表”)。此识别将总是活动的; 它不由 *charref* 参数控制。

当在未引入数字引用或字符引用的文本中找到 & 字符时, 若 *glyphwarning=true*, 将引发异常。换言之, 通过将 *glyphwarning* 设置为 *false*, 可以在同一文本中使用字符引用和独立的 & 字符。

4.5.6 Unicode 兼容的字体

精确的 Unicode 语义对于 PDFlib 的内部处理很重要, 而且对从 PDF 文档中正确提取文本或以其他方式重用文档 (例如将内容转换成其他格式) 至关重要。这在创建标签 PDF 时尤为重要, 因为标签 PDF 对于 Unicode 规范具有严格的要求 (请参见第 149 页上的第 7.5.1 节 “用 PDFlib 生成标签 PDF”)。除了标签 PDF 之外, Unicode 兼容性还与 *textflow* 功能有关。

**Unicode 兼容的字体** 对于使用 *load\_font()* 加载的字体 (更准确地说, 应为字体和编码的组合), 如果在加载字体时使用的编码符合以下条件, 则被认为是 Unicode 兼容的:

- ▶ 仅 *Symbol* 和 *ZapfDingbats* 字体以及基于 PostScript 的 OpenType 字体允许 *builtin* 编码。
- ▶ 编码不是 *glyphid*。

**Unicode 兼容的输出** 若想确保从生成的 PDF 中能够可靠地提取文本, 则对于生成标签 PDF, 输出必须是 Unicode 兼容的。如果以下条件都成立, 则使用 PDFlib 创建的 PDF 输出将是 Unicode 兼容的:

- ▶ 文档中使用的所有字体都必须是如上所定义的 Unicode 兼容的字体。
- ▶ 如果编码是使用 *encoding\_set\_char()* 和字形名称 (不带相应的 Unicode 值) 构造的, 或者是从一个编码文件加载的, 则所有字形名称必须包含在 Adobe 字形列表表中或 *Symbol* 字体中已知的字形名称的列表中。
- ▶ *unicodemap* 参数或选项为 *true*。
- ▶ 所有文本字符串都必须根据 Unicode 标准清楚地定义语义, 即不允许 Unicode 私人使用区域 (PUA) 的字符。



- ▶ 使用 PDI 导入的 PDF 页必须与 Unicode 兼容。PDI 不会更改导入的页的 Unicode 兼容性状态：它既不会移除也不会添加 Unicode 信息。

当创建标签 PDF 输出时，通过 `begin_item()` 中的 `ActualText` 选项提供合适的 Unicode 文本，违背这些规则的文本部分仍可以与 Unicode 兼容。

## 4.6 文本规格和文本变体

### 4.6.1 字体和字符规格

**文本位置** PDFlib 可以让文本位置不需要依赖当前点来绘制图形。文本位置可以通过 `textx/texty` 参数查询，而当前点可以通过 `currentx/currenty` 查询。

**字符规格** PDFlib 使用由 PostScript 和 PDF 使用的字符和字体规格系统，下面将对此进行简单的讨论。

字体大小（必须由 PDFlib 用户指定的）是相邻两文本行之间可避免字符重叠的最小距离。字体大小通常大于字体中的各个字符，因为它跨越了字母上伸部分（*ascender*）和字母下伸部分（*descender*），以及行与行之间可能的其他空间。

*leading*（行间距）指定文本的相邻行的基线之间的垂直距离。默认情况下，它设置为字体大小的值。*capheight* 在多数拉丁字体中是指大写字母如 *T* 或 *H* 的高度。*xheight* 在多数拉丁字体中是指小写字母如 *x* 高度。*ascender* 是指小写字母（例如多数拉丁字体中的 *f* 或 *d*）的高度。*descender* 是指从基线到小写字母（例如多数拉丁字体中的 *j* 或 *p*）的底部的距离。*descender* 通常是负值。*xheight*、*capheight*、*ascender* 和 *descender* 的值作为字体大小的一部分进行计算，在使用这些值之前，必须乘以所需的字体大小。

有些字体或规格文件并不包含这些参数，这时 PDFlib 不得不估算其相应值。若想查出所使用的是真实值还是估算值，可查询 *xheightfaked* 等参数值。查询特定字体的规格信息，如下所示：

```
font = oPDF.load_font("Times-Roman", "winansi", "")
oPDF.setfont font, fontsize

capheight = oPDF.get_value("capheight", font) * fontsize
ascender = oPDF.get_value("ascender", font) * fontsize
descender = oPDF.get_value("descender", font) * fontsize
```

注：无法从 PDFlib 查询上标和下标的位置和大小。

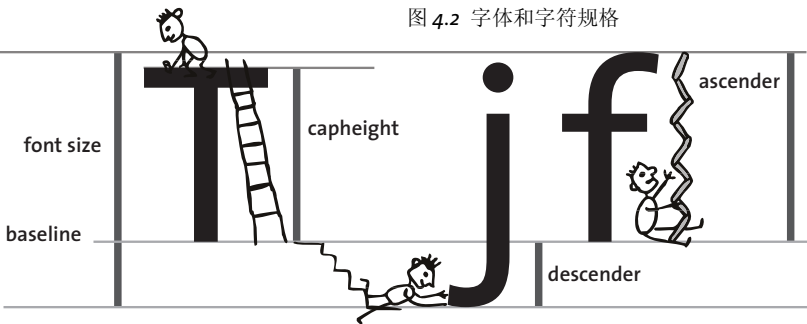


图 4.2 字体和字符规格

**CPI 计算** 虽然多数字体都具有变化的字符宽度，但是所谓的等宽字体对所有字符都使用相同的宽度。为了将 PDF 字体规格与高速度打印环境下经常使用的每英寸字符数 (CPI) 测量单位联系起来，等宽的 Courier 字体的一些计算示例可能会有帮助。在 Courier 中，所有字符的宽度均为 600 个单位，与此对应的是，整个字符单元为每点 1000 个单位（可以从相应的 AFM 规格文件检索此值）。例如，对于 12 点的文本，所有字符的绝对宽度应为

$12 \text{ 点} * 600 / 1000 = 7.2 \text{ 点}$

最佳行间距为 12 点。由于每英寸有 72 点，因此 1 英寸正好放下 10 个 Courier 12 点字符。换言之，12 点 Courier 就是 10 cpi 字体。对于 10 点文本，字符宽度是 6 点，经过计算应为 72/6 = 12 cpi 字体。同样，8 点 Courier 对应于 15 cpi。

4.6.2 字距调整

一些字符组合会导致难看的外观。例如，紧邻的两个 V 看起来就像一个 W，而 T 和 e 之间的距离必须减少以避免不协调的空白。这种调整通常称为“字距调整”。许多字体都包含有全面的字距调整表，提供了用于某些关键字母对的间距调整值。PDFlib 提供了两种控制字距调整行为的方式：

- ▶ 默认情况下，加载字体时不会读取字体中的字距调整信息。若要求进行字距调整，则在对 load\_font() 的相应的调用中必须设置 kerning 选项。这将指示 PDFlib 读取字体的字距调整数据（如果可用）。
- ▶ 当已读取了字距调整数据的字体与任何文本输出函数一起使用时，将应用字距调整数据提供的位置修正。不过，也可以通过将 kerning 参数设置为 false 禁用字距调整：

```
oPDF.set_parameter "kerning", "false" ' disable kerning
```

有时可能需要临时禁用字距调整，例如，由于经过字距调整的数字在表中无法对齐，因此对于在字距调整表中具有对应的数字对的表中数字，应临时禁用字距调整。除了可能激活的任何字符间距、单词间距和水平缩放之外，还应用字距调整。PDFlib 对字体中的字距调整对的数目没有任何限制。

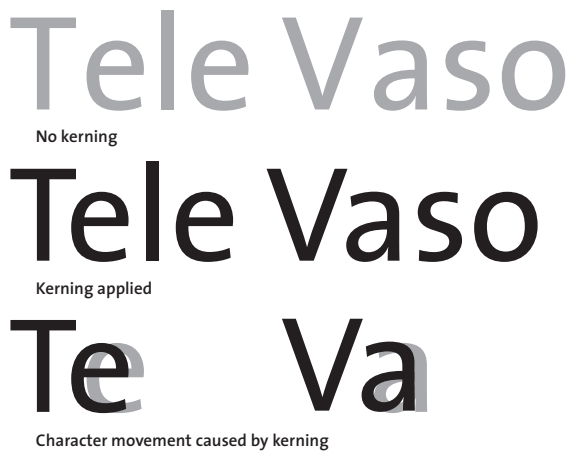


图 4.3 字距调整

### 4.6.3 文本变体

伪字体样式 字体的粗体和斜体变体通常应通过选择适合的字体来创建。此外，PDFlib 也支持伪字体样式：在常规字体的基础上，Acrobat 将通过增强或倾斜基本字体模拟粗体、斜体或粗斜体样式。伪字体样式的美感质量比不上由字体设计器精心调配出的实际粗体或斜体字体。不过，在不能直接使用特殊字体样式的情况下，可以使用伪样式作为一种变通方式。伪字体样式对于标准 CJK 字体尤其有用，因为该字体仅支持常规字体，而不支持任何粗体或斜体变体。

注：不建议对标准 CJK 字体以外的字体使用 `fontstyle` 功能。还请注意，`fontstyle` 功能在 Adobe Acrobat 之外的 PDF 浏览器中可能不起作用。

由于 Adobe Acrobat 中的限制，仅在以下所有条件都满足的情况下，伪字体样式才起作用：

- ▶ 基字体是 TrueType 或 OpenType 字体，包括标准和自定义 CJK 字体。基字体不得为 PDF 核心字体（请参见第 70 页上的第 4.3.2 节“字体嵌入”）。字体样式不能应用于 TrueType 集合 (TTC)。
- ▶ 编码是 `winansi`、`macroman` 或表 4.5 中列出的预定义 CJK CMaps 之一（因为如果不是这样的话，PDFlib 将强制执行字体嵌入）。
- ▶ `embedding` 选项必须设置为 `false`。
- ▶ 基字体必须安装在查看 PDF 所使用的目标系统上。

虽然 PDFlib 将检查最初三个条件，但是用户需要负责确保最后一个条件。

通过对 `load_font()` 的 `fontstyle` 选项使用 `normal`（对基字体不做任何更改）、`bold`、`italic` 或 `bolditalic` 关键字，可以请求伪字体样式：

```
oPDF.load_font "HeiseiKakuGo-W5", "UniJIS-UCS2-H", "fontstyle bold"
```

`fontstyle` 功能不应与 Windows 字体样式名称的类似概念混淆。`fontstyle` 仅在以上条件下才起作用，并需要 Acrobat 来模拟伪字体样式，而 Windows 样式名称完全基于 Windows 字体选择引擎，并且不能用于模拟不存在的样式。

模拟的斜体字体 作为 `fontstyle` 的一种替代方式，`italicangle` 参数或选项可用于在只提供有常规字体的情况下模拟斜体字体。此方法通过将常规字体倾斜用户提供的角度来创建伪斜体字体，并且不受以上提到的 `fontstyle` 限制。负值将顺时针倾斜文本。注意，使用实际斜体或倾斜的字体将会得到更加满意的输出。不过，若没有可用的斜体字体，则可以使用 `italicangle` 参数轻松地模拟一款斜体字体。此功能可能对 CJK 字体特别有用。`italicangle` 参数的典型值的范围是 -12 到 -15 度之间：

```
oPDF.set_value "italicangle", -12 ' create fake italic font
```

注：垂直书写模式不支持 `italicangle` 参数。

下划线、上划线和删除线文本 可以指示 PDFlib 将文本的下方、上方或中间放置直线。横线的粗细以及它与基线的距离将基于字体的规格信息进行计算。另外，当计算横线的宽度时，需要考虑水平缩放因素和文本字模的当前值。`set_parameter()` 可用于开启或关闭下划线、上划线和删除线功能，如下所示：

```
oPDF.set_parameter "underline", "true" ' enable underlines
```

使用当前的笔触颜色绘制横线。不过，将忽略当前的 `linecap` 和 `dash` 参数。美学上的提示：在大多数字体中，下划线将接触到下行字母，而上划线将接触到上行字母顶部的发音标记。

注：除非使用 `Unicode CMap`，否则标准 CJK 字体不支持下划线、上划线和删除线功能。

文本渲染模式 PDFlib 支持一些将影响文本外观的渲染模式。其中包括空心文本和将文本用作剪贴路径的能力。文本也可以通过不可见的方式渲染，这样可能有助于在扫描的图像上放置文本，从而可以搜索文本和构建文本索引，同时确保文本将不会直接可见。在表 8.18 中说明了渲染模式。可以使用 `set_value()` 和 `textrendering` 参数设置这些模式。

```
oPDF.set_value "textrendering", 1 ' set stroked text rendering (outline text)
```

当对文本描边时，将对字形轮廓使用图形状态参数，例如 `linewidth` 和 `color` 参数。渲染模式对使用 **Type 3** 字体显示的文本没有影响。

注：文本渲染模式 7 (用文本作为剪贴路径) 当使用 `fit_textline()` 或 `fit_textflow()` 创建文本输出时，将没有任何影响。

文本颜色 通常文本将使用当前填充色显示，填充色可以使用 `setcolor()` 进行设置。不过，若选定的是除 `o` 之外的渲染模式，则描边和填充色对文本的影响可能会因选定的渲染模式而异。

# 4.7 中文、日文和韩文文本

## 4.7.1 Acrobat 和 PDF 中支持的 CJK

Acrobat/PDF 支持一组没有字体嵌入的标准 CJK 字体，以及自定义的嵌入式 CJK 字体。尽管嵌入的 CJK 字体在所有版本的 Acrobat 中都可以起作用而无需再费周折，但是在 Acrobat 中使用任何标准 CJK 字体都要求用户执行以下操作之一<sup>1</sup>：

- ▶ 使用本地化的 CJK 版本的 Acrobat。
- ▶ 若使用任何非 CJK 版本的完全版 Acrobat 产品，请选择 Acrobat 安装程序中的选项“亚洲语言支持”(Windows) 或“语言工具包”(Mac)。将从 Acrobat 产品光盘中安装必需的支持文件（字体和 CMap）。
- ▶ 若使用 Acrobat Reader，则安装在 Web 上可用的亚洲字体包之一。<sup>2</sup>

打印带有 CJK 文本的 PDF 文档 打印 CJK 文档会引起许多超出本手册讨论范围的问题。不过，为了方便 PDFlib 用户，我们将提供一些有用的提示。若您在使用 Acrobat 打印 CJK 文档（特别是那些使用了标准字体的 CJK 文档）时遇到麻烦，则请考虑以下内容：

- ▶ 由于字符数量巨大，CJK 字体消耗大量打印机内存，除非已应用字体子集化功能。不是所有的打印机都具备打印此类字体所需的足够内存。
- ▶ 非日文 PostScript 打印机没有安装任何日文字体。为此，您必须在 Acrobat 的打印对话框中选中“下载亚洲字体”。
- ▶ 若您无法使用下载的字体成功打印，则请在 Acrobat 的打印对话框中选中“作为图像打印”。这样会指示 Acrobat 将页面的位图版本发送到打印机（但分辨率只有 300 dpi）。

## 4.7.2 标准 CJK 字体和 CMap

在历史上，不同的标准化团体、公司和其他组织已开发出大量 CJK 编码方案。幸运的是，默认情况下，Acrobat 和 PDF 支持所有现行编码。由于编码的概念对于 CJK 文本相对于拉丁文本来说复杂很多，所以仅使用简单的 8 位编码已不够。PostScript 和 PDF 使用字符集和字符映射 (CMap) 的概念来组织字体中的字符。

1. 这里需要感谢 Ken Lunde 的开创性著作《CJKV 信息处理－中文、日文、韩文和越南文计算》(CJKV information processing－Chinese, Japanese, Korean & Vietnamese Computing, O'Reilly 1999, ISBN 1-56592-224-7)，以及他本人在 Adobe 的工作。他与其他人一道推动了 PostScript 和 PDF 中的 CJK 支持工作。

2. 请参见 [www.adobe.com/products/acrobat/acrrasianfontpack.html](http://www.adobe.com/products/acrobat/acrrasianfontpack.html)

Acrobat 支持许多用于 CJK 文本的标准字体。这些字体通过 Acrobat 安装（或亚洲字体包）提供，因此不必在 PDF 文件中嵌入。这些字体包含常用编码所需的所有字符，并且支持水平书写和垂直书写两种模式。在表 4.4 中列出了这些标准字体和 CMaps。Acrobat 4 字体也可以与 Acrobat 5 一起使用，如果系统上未安装所需的字体，则使用相应的 Acrobat 5 字体进行显示和打印。

注：Acrobat 的标准 CJK 字体不支持粗体和斜体变体。不过，可以使用伪字体样式功能模拟这些字体（请参见第 83 页上的第 4.6.3 节“文本变体”）。

正如表中显示的一样，默认 CMap 支持在 Mac、Windows 和 Unix 系统上使用的大多数 CJK 编码，以及一些其他的特定于供应商的编码。具体而言，如支持主要的日文编码方案 Shift-JIS、EUC、ISO 2022 和 Unicode（UCS-2 和 UTF-16）。包含所有支持的字符的表可以从 Adobe<sup>1</sup> 获得；在表 4.5 中可以找到 CMap 说明。

注：支持 Unicode 的语言绑定必须仅使用 Unicode 兼容的 CMap（UCS2 或 UTF16）。不支持其他的 CMap。

表 4.4 Acrobat 的用于日文、中文和韩文文本的标准字体和 CMap（编码）

区域	字体名称	示例	支持的 CMap（编码）
简体中文	STSong-Light <sup>1</sup>	国际 国际 国际	UniGB-UCS2-H, UniGB-UCS2-V, UniGB-UTF16-H <sup>4</sup> , UniGB-UTF16-V <sup>4</sup>
	STSongStd-Light-Acro <sup>2</sup>		
	AdobeSongStd-Light-Acro <sup>3</sup>		
繁体中文	MHei-Medium <sup>1</sup>	中文 中文 中文	UniCNS-UCS2-H, UniCNS-UCS2-V, UniCNS-UTF16-H <sup>4</sup> , UniCNS-UTF16-V <sup>4</sup>
	MSung-Light <sup>1</sup>		
	MSungStd-Light-Acro <sup>2</sup> AdobeMingStd-Light-Acro <sup>3</sup>		
日文	HeiseiKakuGo-W5 <sup>1</sup>	日本語 日本語 日本語 日本語	UniJIS-UCS2-H, UniJIS-UCS2-V, UniJIS-UCS2-HW-H <sup>5</sup> , UniJIS-UCS2-HW-V <sup>5</sup> , UniJIS-UTF16-H <sup>4</sup> , UniJIS-UTF16-V <sup>4</sup>
	HeiseiMin-W3 <sup>1</sup>		
	KozMinPro-Regular-Acro <sup>2, 5</sup>		
	KozGoPro-Medium-Acro <sup>3, 5</sup>		
韩文	HYGoThic-Medium <sup>1</sup>	한국 한국 한국 한국	UniKS-UCS2-H, UniKS-UCS2-V, UniKS-UTF16-H <sup>4</sup> , UniKS- UTF16-V <sup>4</sup>
	HYSMyeongJo-Medium <sup>1</sup>		
	HYSMyeongJoStd-Medium-Acro <sup>2</sup>		
	AdobeMyungjoStd-Medium-Acro <sup>3</sup>		

1. 在 Acrobat 4 中可用；Acrobat 5 和 6 将使用不同的字体替换这些字体。  
2. 仅在 Acrobat 5 中可用。  
3. 仅在 Acrobat 6 中可用。  
4. 仅当生成 PDF 1.5 或更高版本时可用。  
5. KozMinPro-Regular-Acro 和 KozGoPro-Medium-Acro 字体不允许 HW CMap，因为这些字体仅包含比例 ASCII 字符，而不包含任何半宽字符。

1. 有关与 CID 字体相关的资源宝库，包括列出了所有支持的字形的表（搜索“character collection”），请参见 [partners.adobe.com/asn/tech/type/cidfonts.jsp](http://partners.adobe.com/asn/tech/type/cidfonts.jsp)。

水平书写和垂直书写模式 PDFlib 支持对标准 CJK 字体和 CMap 使用水平书写和垂直书写模式。通过选择适当的 CMap 名称，可连同编码一起选定模式。其名称结尾为 -H 的 CMap 选择水平书写模式，而 -V 后缀选择垂直书写模式。

表 4.5 用于日文、中文和韩文文本的预定义 CMap（摘自 PDF Reference）

区域	CMap 名称	字符集和文本格式
简体中文	UniGB-UCS2-H	用于 Adobe-GB1 字符集的 Unicode (UCS-2) 编码
	UniGB-UCS2-V	
	UniGB-UTF16-H	用于 Adobe-GB1 字符集的 Unicode (UTF-16BE) 编码。包含对 GB18030-2000 字符集中所有字符的映射。
	UniGB-UTF16-V	
繁体中文	UniCNS-UCS2-H	用于 Adobe-CNS1 字符集的 Unicode (UCS-2) 编码
	UniCNS-UCS2-V	
	UniCNS-UTF16-H	用于 Adobe-CNS1 字符集的 Unicode (UTF-16BE) 编码。包含对 HKSCS-2001 的所有字符（2 字节和 4 字节字符代码）的映射
	UniCNS-UTF16-V	
日文	UniJIS-UCS2-H, -V	用于 Adobe-Japan1 字符集的 Unicode (UCS-2) 编码
	UniJIS-UCS2-HW-H	与 UniJIS-UCS2-H 一样，但是用半宽窗体替换成比例的拉丁字符
	UniJIS-UCS2-HW-V	
	UniJIS-UTF16-H	用于 Adobe-Japan1 字符集的 Unicode (UTF-16BE) 编码。在 JIS X 0213:1000 字符集中包含所有字符的映射。
韩文	UniKS-UCS2-H, -V	用于 Adobe-Korea1 字符集的 Unicode (UCS-2) 编码
	UniKS-UTF16-H, -V	

注：一些 PDFlib 函数可根据书写模式更改它们的语义。例如，不应在垂直书写模式中使用 `continue_text()`，并且必须将字符间距设置为负值以便在垂直书写模式中将字符分开。

用于标准 CMap 的 CJK 文本编码 客户负责提供与请求的 CMap 匹配的编码文本。PDFlib 对提供的文本是否符合请求的 CMap 不作检查。

对于多字节编码，字符的高位字节必须首先出现。另外，可以使用 `textformat` 参数选定字节顺序和文本格式（请参见第 76 页上的第 4.5.1 节“页面内容和超文本的 Unicode”），前提是使用 Unicode CMap（UCS-2 或 UTF-16）。

标准 CJK 字体和 CMap 的限制 与非 Unicode CMap 组合的标准 CJK 字体不支持以下功能（Unicode CMap 在其名称中带有 UCS2 或 UTF16）：

- ▶ 使用 `stringwidth()` 计算文本的宽度（请参见第 87 页上的“强制等宽字体”）
- ▶ 使用 `fit_textline()`、`create_textflow()` 和相关的 `Textflow` 函数
- ▶ 激活下划线 / 上划线 / 删除线模式
- ▶ 检索 `textx/texty` 位置

这些限制适用于标准 CJK 字体。请注意，尽管在这些情况下无法查询 CJK 文本的宽度，但在 PDF 输出中将正确地生成此宽度。还请注意，自定义 CJK 字体完全支持以上所有功能。

标准 CJK 字体示例 通过使用 `load_font()` 接口并提供 CMap 名称作为 `encoding` 参数，可选择标准 CJK 字体。不过，必须考虑到一点：给定的 CJK 字体仅支持一组 CMap（请参见表 4.4），并且识别 Unicode 的语言绑定仅支持 UCS2 兼容的 CMap。表 4.4 中的 `KozMinPro-Regular-Acro` 示例可以使用以下代码生成：

```
font = oPDF.load_font("KozMinPro-Regular-Acro", "UniJIS-UCS2-H", "")
oPDF.setfont font, 24
oPDF.set_text_pos 50, 500
oPDF.show ChrW(&H65E5) & ChrW(&H672C) & ChrW(&H8A9E)
```

上述语句查找一种日文标准字体，并且选择 Shift-JIS 兼容的 CMap (*Ext-RKSJ*) 和水平书写模式 (*H*)。 *fontname* 参数必须是字体的准确名称，并且不带任何编码或书写模式后缀。 *encoding* 参数是一种受支持的 CMap 的名称（根据字体选择），并且还将指示书写模式（请参见前面的内容）。 PDFlib 支持 Acrobat 的所有默认 CMap，并且会在检测到请求的字体和 CMap 不匹配时做出反应。例如，PDFlib 将拒绝对韩文字体使用日文编码的请求。

**强制等宽字体** 一些应用程序没有能力处理比例 CJK 字体和根据字形宽度和字形数量来计算文本的宽度。可以指示 PDFlib 甚至对通常具有宽度不定的字形的字体强制使用等宽字形。使用 *load\_font()* 的 *monospace* 选项可以指定所有字形所需的宽度。对于标准 CJK 字体，值 1000 将产生满意的结果：

```
font = oPDF.load_font("KozMinPro-Regular-Acro", "UniJIS-UCS2-H", "monospace 1000")
```

仅对标准 CJK 字体推荐使用 *monospace* 选项。

## 4.7.3 自定义 CJK 字体

除了 Acrobat 的标准 CJK 字体之外，PDFlib 还支持 TrueType（包括 TrueType 集合 (TTC)）中的自定义 CJK 字体（表 4.4 中的列表之外的字体）和 OpenType 格式。将按如下方式处理自定义 CJK 字体：

- ▶ 不管客户端提供的嵌入设置如何，字体将转换为 CID 字体并嵌入到 PDF 输出中。因为 PDFlib 遵从可能在字体中定义的字体嵌入限制，所以不允许嵌入的字体不能用作自定义 CJK 字体。
- ▶ 默认情况下，将对嵌入的自定义 CJK 字体使用字体子集化，除非该字体与标准 CMap 一起使用。可以使用多个参数控制上述行为，请参见第 69 页上的第 4.3 节“字体嵌入和子集化”。
- ▶ 自定义 CJK 字体和一些编码支持比例拉丁字符和半宽字符（请参见表 8.15）。
- ▶ Windows 上的日文宿主字体名称可以提供给 *load\_font()* 作为 UCS-2，但在 Mac 上不支持非拉丁宿主字体名称。

注：虽然支持 *Windows EUDC* 字体（终端用户定义的字符），但是不支持将单个最终用户定义的字符链接到所有字体中（请参见以下内容）。

注： *TrueType CJK* 字体不支持垂直书写模式。

**自定义 CJK 字体支持的编码** 自定义 CJK 字体可以与以下编码一起使用（另请参见表 8.15）：

- ▶ 使用 PostScript 字型的 OpenType CJK 字体（CID 字体）可以与相应区域的所有 CMap 一起使用（例如，日文字体仅可以与日文 CMap 一起使用）。
- ▶ *unicode* 编码。
- ▶ *glyphid* 寻址（请参见第 75 页上的第 4.4.3 节“TrueType 和 OpenType 字体的字形 ID 寻址”）。

**用于中文 Unicode 文本的自定义 CJK 字体示例** 以下示例使用 *ArialUnicodeMS* 字体以显示一些中文文本。该字体必须安装在系统上或者必须根据第 69 页上的第 4.3.1 节“PDFlib 搜索字体的方法”进行配置：

```
' This is not required if the font is installed on the system
oPDF.set_parameter "FontOutline", "Arial Unicode MS=ARIALUNI.TTF"
font = oPDF.load_font("Arial Unicode MS", "unicode", "")

oPDF.setfont font, 24
oPDF.set_text_pos 50, 500
oPDF.show ChrW(&H4E00) & ChrW(&500B) & ChrW(&H4EBA)
```

在访问 **TrueType** 集合 (**TTC**) 中的各个字体。TTC 文件包含多个单独的字体。可以通过提供正确的名称来访问相应的字体。不过，若不知道 TTC 文件中包含了哪些字体，则可以通过在 TTC 文件内追加一个冒号字符和字体编号（以 o 开始）来以数字方式对每个字体寻址。若索引为 o，则它可以被省略。例如，TTC 文件 *msgothic.ttc* 包含多个字体，这些字体可以通过如下方式用 *load\_font()* 进行寻址（每一行包含相当的字体名称）：

```
msgothic:0      MS Gothic      msgothic:
msgothic:1      MS PGothic
msgothic:2      MS UI Gothic
```

请注意，*msgothic*（不带任何后缀）将不能用作字体名，因为它不能唯一地识别字体。字体名称别名（请参见第 69 页上的第 4.3.1 节“PDFlib 搜索字体的方法”）可以与 TTC 索引结合使用。若无法找到具有指定索引的字体，并且 *fontwarning=true*，将引发异常。

TTC 字体文件仅需要配置一次；TTC 文件中所有编入索引的字体都将自动找到。以下代码足以配置 *msgothic.ttc* 中的所有编入索引的字体：

```
oPDF.set_parameter "FontOutline", "msgothic=msgothic.ttc"
```

终端用户定义的字符 (**EUDC**) PDFlib 不支持将终端用户定义的字符链接到字体中，但是可以使用 Windows 中可用的 EUDC 编辑器创建自定义字符，以便与 PDFlib 一起使用。按如下所示操作：

- ▶ 使用 *eudcedit.exe* 在所需的 Unicode 位置创建一个或多个自定义字符。
- ▶ 在目录 `\Windows\fonts` 中查找 *EUDC.TTE* 文件，并将该文件复制到另外的目录。由于此文件在 Windows 资源管理器中不可见，因此需要在 DOS 对话框中使用 *dir* 和 *copy* 命令来查找这个文件。现在使用第 69 页上的第 4.3.1 节“PDFlib 搜索字体的方法”中讨论的方法之一配置字体以便用于 PDFlib：

```
oPDF.set_parameter "FontOutline", "EUDC=EUDC.TTE"
oPDF.set_parameter "SearchPath", "...directory name..."
```

或在当前目录下放置 *EUDC.TTE*。

- ▶ 作为前一步骤的替代方式，可以使用以下函数调用以从 Windows 目录直接配置字体文件。这样能总是访问在 Windows 中使用的当前 EUDC 字体：

```
oPDF.set_parameter "FontOutline", "EUDC=C:\Windows\fonts\EUDC.TTE"
```

- ▶ 使用以下调用来加载字体：

```
font = oPDF.load_font("EUDC", "unicode", "")
```

并提供在第一个步骤中选定的 Unicode 字符代码以输出字符。

## 4.8 放置并调整单行文本

用于在页上放置单行文本的函数 *fit\_textline()* 提供了许多格式化选项。在本节中，我们使用一些常见的应用程序示例来讨论最重要的选项。在表 8.19 中可以找到这些选项的完整说明。*fit\_textline()* 的大多数选项都与 *fit\_image()* 的选项相同。因此，在这里我们仅使用文本相关的示例；建议查看第 113 页上的第 5.3 节“放置图像和导入的 PDF 页”中的示例以获得初步的了解。

以下示例仅演示函数 *fit\_textline()* 的相关调用，并假定已加载必需的字体并设置了所需的字体大小。



`fit_textline()` 使用所谓的文本框来确定文本的位置：文本框宽度与文本的宽度一致，而文本框的高度与字体中大写字母的高度一致。使用 `margin` 选项可以左右扩展或上下扩展文本框。页边距将随同文本行一起缩放。

图 4.4  
将文本放置在  
底部中央位置

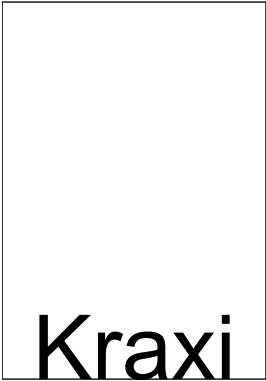
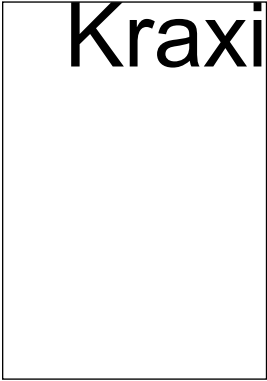


图 4.5  
将文本放置在右上角



### 4.8.1 简单的文本放置

将文本放置在底部中央位置。 我们将文本与参考点对齐，即将文本框底边的中心与参考点对齐（请参见图 4.4）：

```
oPDF.fit_textline text, 297, 0, "position {50 0}"
```

此代码片段将文本框的底边中心 (`position {50 0}`) 与参考点 (`297, 0`) 对齐。

将文本放置在右上角 现在我们将文本与参考点对齐，即将文本框的右上角与参考点对齐（请参见图 4.5）：

```
oPDF.fit_textline text, 595, 842, "position 100"
```

此代码片段将文本框的右上角 (`position 100`) 参考点 (`595, 842`) 对齐。

在页边距中放置文本 为了扩展上一示例，我们可以对文本添加水平页边距以向右扩展一段距离。这可能有助于在表格列中放置文本： `oPDF.fit_textline text, 595, 842, "position 100 margin {20 0}"`

### 4.8.2 在文本框中放置文本

在框中居中放置文本 我们定义一个框，并在该框内居中放置文本（请参见图 4.6）：  
`oPDF.fit_textline text, 10, 200, "boxsize {500 220} position 50"`

此代码片段将文本在一个文本框中居中放置，其中，文本框的中心位置为 (`position 50`)，左下角位置为 (`10, 200`)，文本框的宽度和长度分别为 500 个单位和 220 个单位 (`boxsize {500 220}`)。

按比例调整文本以使其适合文本框 我们扩展上一示例并使文本完全适合文本框（请参见图 4.7）：

图 4.6  
在文本框中居中放置文本



图 4.7  
按比例调整文本以使其适合  
文本框

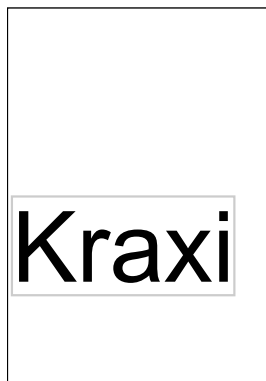
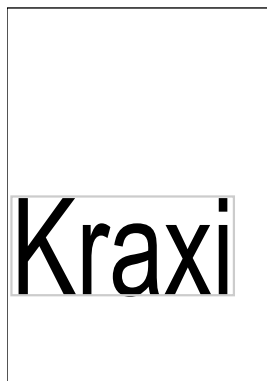


图 4.8  
完全调整文本以使其适合  
文本框



```
10, 200, "boxsize {500 220} position 50 fitmethod meet"oPDF.fit_textline text, 10, 200,  
"boxsize {500 220} position 50 fitmethod meet"
```

请注意，当使用 *fitmethod meet* 使文本适合文本框时，字体大小将发生更改。为了防止文本变大，请使用 *auto* 而不要使用 *meet*。

完全调整文本以使其适合文本框 我们可以进一步修改上一示例，使文本不能按比例适合文本框，而是完全覆盖文本框。不过，此组合将很少使用，因为文本可能会扭曲（请参见图 4.8）：

```
10, 200, "boxsize {500 220} position 50 fitmethod entire"oPDF.fit_textline text, 10, 200,  
"boxsize {500 220} position 50 fitmethod entire"
```

### 4.8.3 对齐文本

简单对齐 我们的下一个目标是旋转文本，使其原始左下角与给定的参考点对齐（请参见图 4.9）。这可能很有用，例如，在表头中放置经过旋转的列标题：

```
oPDF.fit_textline text, 5, 5, "orientate west"
```

图 4.9  
简单对齐

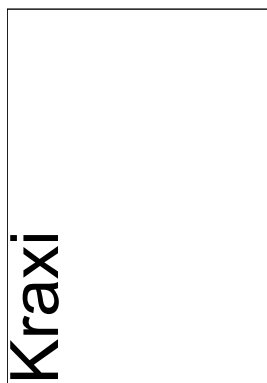


图 4.10  
沿垂直线对齐文本

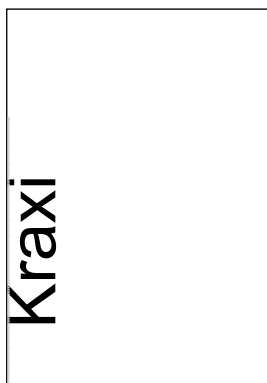
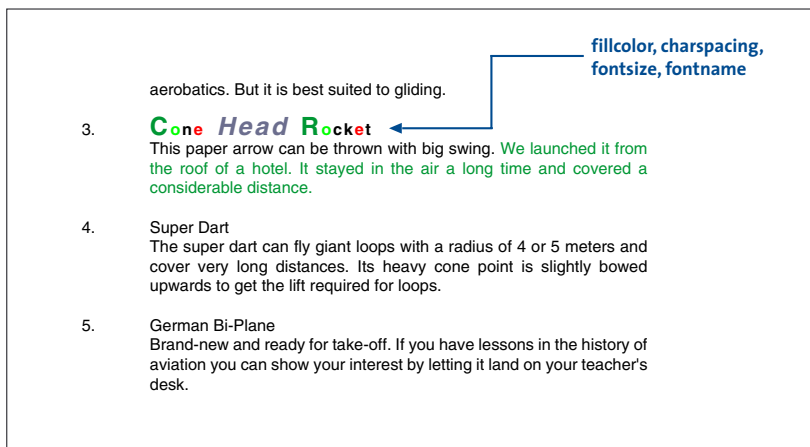


图 4.12  
格式化  
textflow



此代码片段将文本的方向调整为朝西（逆时针旋转 90 度），然后将旋转文本的左下角对准参考点 (5, 5)。

沿垂直线对齐文本 沿垂直线（即，宽度为零的文本框）放置文本是一种有点极端的情况，不过它可能很有用（请参见图 4.10）：

```
oPDF.fit_textline text, 0, 0, "boxsize {0 600} position {0 50} orientate west"
```

此代码片段旋转文本，并沿线的中心从 (0, 0) 到 (0, 600) 放置文本。

## 4.9 多行 textflow

除了在页上放置单行文本之外，PDFlib 支持一种称作“Textflow”的功能，可用于放置任意长度的文本部分。文本可能会延伸到任意数量的行或页，文本的外观可以使用多个选项进行控制。字符属性（例如字体、大小和颜色）可以适用于文本的任何部分。可以指定 textflow 属性（例如分散对齐的文本或锯齿形的文本、段落缩排和制表位等）；将考虑文本中由自动连字符指定的换行机会。图 4.11 和图 4.12 演示如何使用 textflow 功能在页面上放置发票的不同的部分。在以下的部分，我们将更加详细地讨论用于控制输出的选项。

多行 textflow 可以放置在一个或多个页上的一个或多个矩形（所谓的“限定框”）中。以下步骤是在页上放置 textflow 所必需的步骤：

- ▶ 函数 `create_textflow()` 分析文本，创建一个 textflow 对象并返回一个句柄。它不会在页上放置任何文本。
- ▶ 函数 `fit_textflow()` 在提供的限定框中放置部分 textflow 或全部 textflow。为了完全放置文本，可能必须重复执行几次此步骤，其中的每个函数调用都会提供一个新的限定框（可能位于同一页或其他页上）。
- ▶ `delete_textflow()` 函数可删除已放置在文档中的 textflow 对象。

用于创建 textflow 的函数 `create_textflow()` 支持多种用于控制格式化过程的选项。这些选项可以在函数的列表中提供，也可作为 `inline` 选项在文本中嵌入。我们将使用一些常见的应用程序示例来讨论 textflow 的放置。可以在表 8.24 中找到 textflow 选项的完整列表。

`create_textflow()` 中支持的许多选项都与 `fit_textline()` 的选项一致。因此建议您自己熟悉第 88 页上的第 4.8 节“放置并调整单行文本”中的示例。在下面的部分，我们将重点介绍与多行文本有关的选项。

4.9.1 在限定框中放置 textflow

在单个限定框中放置文本 让我们从简单的示例开始。以下代码片段使用默认的格式化选项在页上的单个限定框中放置 `textflow`。字体、字体大小和编码都已明确指定（可在图 4.13 中查看结果）：

```
textflow = oPDF.create_textflow(text, "fontname=Helvetica fontsize=9 encoding=winansi")
oPDF.fit_textflow(textflow, left_x, left_y, right_x, right_y, "")
oPDF.delete_textflow textflow
```

在两个限定框中放置文本 若使用 `fit_textflow()` 放置在页上的文本在限定框中放不下，则输出将中断，该函数将返回字符串 `_boxfull`。PDFlib 将会记住已经放置在页上的文本数量，

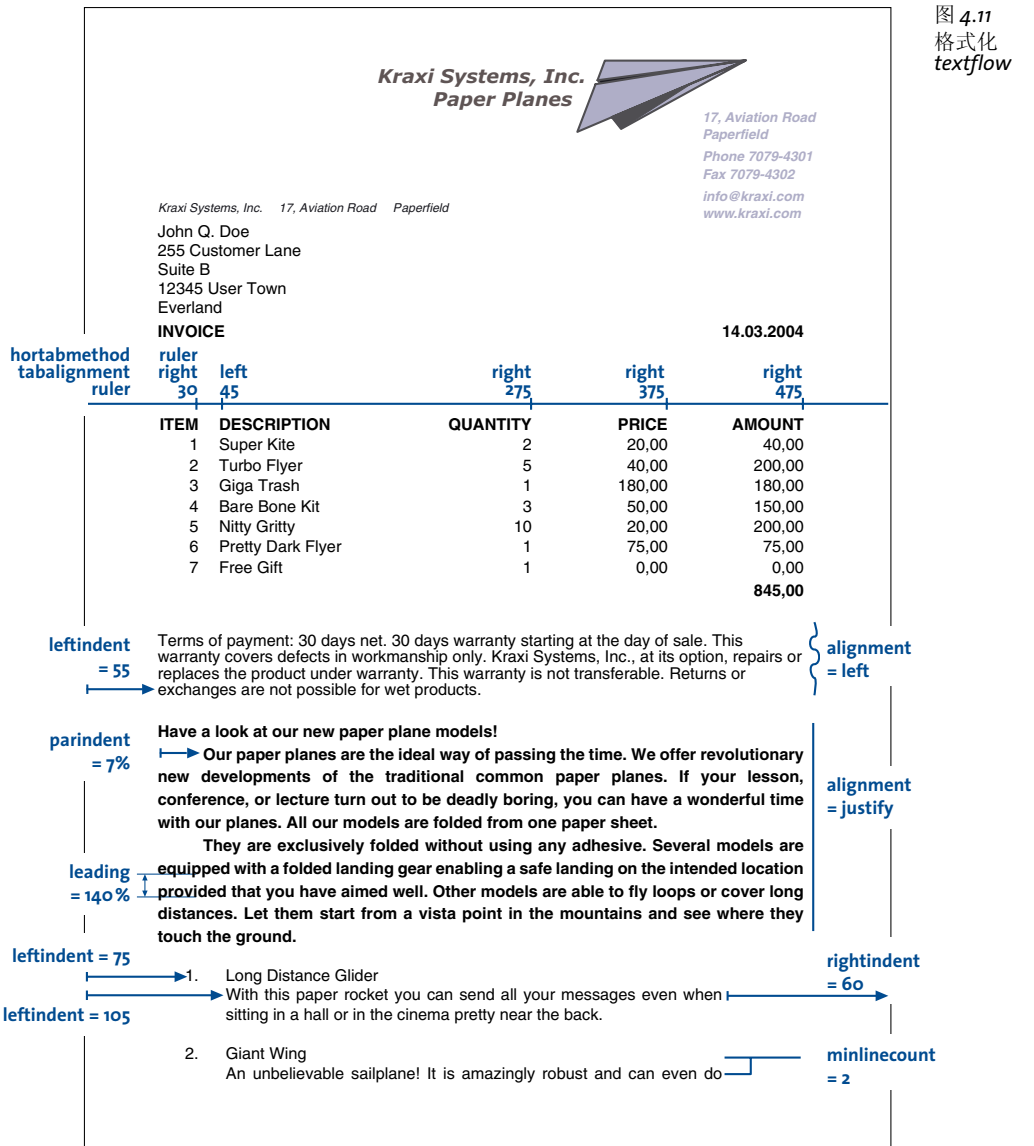


图 4.11  
格式化  
`textflow`

图 4.13  
简单 *textflow* 放置

Terms of payment: 30 days net. 30 days warranty starting at the day of sale. This warranty covers defects in workmanship only. Kraxi Systems, Inc., at its option, repairs or replaces the product under warranty. This warranty is not transferable. Returns or exchanges are not possible for wet products.

并在再次调用该函数时，继续文本的剩余部分。以下代码片段演示如何在两个限定框中放置 *textflow*（可在图 4.14 中查看结果）：

```
textflow = oPDF.create_textflow(text, "fontname=Helvetica fontsize=9 encoding=winansi")
result = oPDF.fit_textflow(textflow, left_x, left_y, right_x, right_y, "")
' Check whether the text could be fully placed in the fitbox
if not strcmp(result, "_boxfull") then
    oPDF.fit_textflow textflow, left_x + offset, left_y, right_x + offset, right_y, ""
end if
oPDF.delete_textflow textflow
```

在多个页上放置文本 若使用 *fit\_textflow()* 放置的文本在限定框中不能全部放下，则可能有必要创建一个新页。用于在多个页上放置 *textflow* 的基础代码如下所示：

```
textflow = oPDF.create_textflow(text, optlist)
Do
    oPDF.begin_page_ext pagewidth, pageheight, ""
    result = oPDF.fit_textflow (textflow, left_x, left_y, right_x, right_y, "")
    oPDF.end_page_ext ""
    if not strcmp(result, "_stop") then
        exit do
    end if
Loop
oPDF.delete_textflow textflow
```

4.9.2 段落格式设置选项

在上一个示例中，我们使用段落的默认设置。例如，默认对齐方式是左对齐，行距是 100%（与字体大小相同）。

为了调整段落的格式设置，我们可以对 *create\_textflow()* 提供更多选项。例如，我们可以将文本从左边距缩排 15 个单位，从右边距缩排 10 个单位。另外，每个段落的第一行应缩排 20 个单位。文本应对照左右边距对齐，并将行距增加到 140%。最终，我们将字体大小减少到 8 个单位。用于实现此设想的、使用了选项列表的扩展代码如下所示（可在图 4.15 中查看结果）：

```
optlist = "leftindent=15 rightindent=10 parindent=20 alignment=justify " & _
    "leading=140% fontname=Helvetica fontsize=8 encoding=winansi"

' place textflow in a fitbox using the options
```

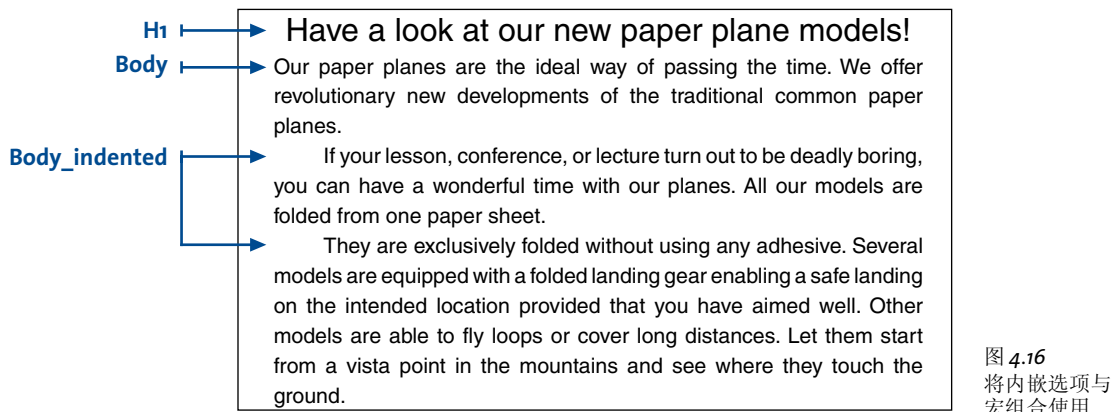
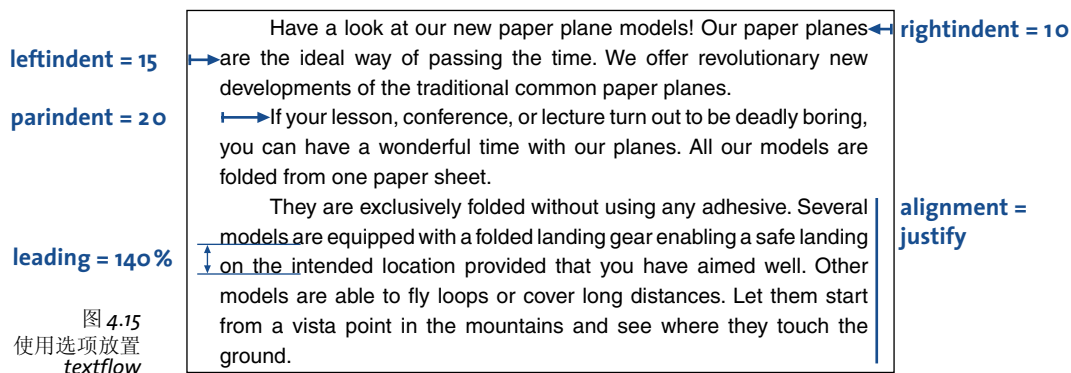
第一个限定框

Terms of payment: 30 days net. 30 days warranty starting at the day of sale. This warranty covers defects in workmanship only. Kraxi Systems, Inc., at its option, repairs or

第二个限定框

replaces the product under warranty. This warranty is not transferable. Returns or exchanges are not possible for wet products.

图 4.14  
在两个限定框中放置 *textflow*



```
textflow = oPDF.create_textflow(text, optlist)
oPDF.fit_textflow textflow, left_x, left_y, right_x, right_y, ""
oPDF.delete_textflow textflow
```

### 4.9.3 内嵌选项列表和宏

图 4.15 中的文本还不正确。标题“Have a look at our new paper plane models!”（看一看我们的新纸飞机模型！）应独自占用一行，并应使用较大的字号居中显示。有一些方法可以用来达到此目的。

到现在为止，我们是在直接提供给函数的选项列表中提供格式化选项。为了继续相同的方式，我们必须拆分文本，并将其放置到两个单独的调用中，一个调用针对标题，另一个调用针对剩余的文本。不过，这将比较麻烦。

为此，**textflow** 功能支持所谓的内嵌选项。简单来说就是将选项嵌入在文本中。内嵌选项列表作为正文的一部分提供。默认情况下，它们用“<”和“>”字符分隔。因此，我们将按如下方式将用于设置标题和其余段落格式的选项集成到正文中（在后面的所有示例中，内嵌选项列表都会用颜色标出；段落结束字符将使用箭头表示）：

```
<leftindent=15 rightindent=10 alignment=center fontname=Helvetica fontsize=12
encoding=winansi>Have a look at our new paper plane models! ◀
<alignment=justify fontname=Helvetica leading=140% fontsize=8 encoding=winansi>
Our paper planes are the ideal way of passing the time.We offer
revolutionary new developments of the traditional common paper planes. ◀
<parindent=20>If your lesson, conference, or lecture
```

turn out to be deadly boring, you can have a wonderful time with our planes.All our models are folded from one paper sheet. ↵  
They are exclusively folded without using any adhesive.Several models are equipped with a folded landing gear enabling a safe landing on the intended location provided that you have aimed well. Other models are able to fly loops or cover long distances.Let them start from a vista point in the mountains and see where they touch the ground.

可以使用 *begoptlistchar* 和 *endoptlistchar* 选项重新定义围绕选项列表的字符（请参见表 8.24）。将关键字 *none* 提供给 *begoptlistchar* 选项会完全禁用对选项列表的搜索。若文本未包含内嵌选项列表，且想确保“<”和“>”将作为常规字符进行处理，则可以使用这一方法。

宏 上面的文本基本上包含一些不同类型的段落，例如标题和带缩排或不带缩排正文。每一种段落类型都单独设置格式，并在较长的 *textflow* 中多次出现。为了避免每一段落都用相应的内嵌选项开始，我们可以将这些选项组合成宏的形式，并通过宏名称在文本中引用相应的宏。如图 4.16 中所示，我们可以定义三个宏，它们分别是用于标题的名为 *H1* 的宏，用于主要段落的名为 *Body* 的宏和用于缩排的段落的名为 *Body\_indented* 的宏。为了使用宏，我们将 *&* 字符放置在宏名称前面，并将其放入选项列表中。以下代码片段根据先前使用的内嵌选项定义三个宏并在文本中使用这些宏：

```
<macro {  
H1 {leftindent=15 rightindent=10 alignment=center  
fontname=Helvetica fontsize=12 encoding=winansi}  
  
Body {leftindent=15 rightindent=10 alignment=justify leading=140%  
fontname=Helvetica fontsize=8 encoding=winansi}  
  
Body_indented {parindent=20 leftindent=15 rightindent=10 alignment=justify  
leading=140% fontname=Helvetica fontsize=8 encoding=winansi}  
}>  
<&H1>Have a look at our new paper plane models! ↵  
<&Body>Our paper planes are the ideal way of passing the time.We offer  
revolutionary new developments of the traditional common paper planes. ↵  
<&Body_indented>If your lesson, conference, or lecture  
turn out to be deadly boring, you can have a wonderful time  
with our planes.All our models are folded from one paper sheet. ↵  
They are exclusively folded without using any adhesive.Several  
models are equipped with a folded landing gear enabling a safe  
landing on the intended location provided that you have aimed well.  
Other models are able to fly loops or cover long distances.Let them  
start from a vista point in the mountains and see  
where they touch the ground.
```

明确设置选项 请注意，所有未在宏中设置的选项将保留其先前的值。为了避免由选项的不必要“继承”产生的副作用，应明确指定特殊的宏所需的所有设置。这样，不管宏的顺序或与其他选项列表的组合如何，都可以确保宏将始终如一地工作。

另一方面，可以利用这种方式有意保留上下文的某些设置，而不用明确提供它们。例如，宏可能会指定字体名称，而不提供 *fontsize* 选项。结果，字体大小将总是匹配先前文本中的字体大小。

使用内嵌选项还是使用作为函数参数传递的选项？ 当使用 *textflow* 时，文本是全部包含在程序代码中还是来自于一些外部源，格式化说明是与文本中分离的还是作为文本的一部分

hortabmethod	ruler				
tabalignment	left		right	right	right
ruler	30		150	250	350

图 4.17  
放置文本  
作为表

ITEM	DESCRIPTION	QUANTITY	PRICE	AMOUNT
1	Super Kite	2	20.00	40.00
2	Turbo Flyer	5	40.00	200.00
3	Giga Trash	1	180.00	180.00
TOTAL				420.00

分，这些方面的差异会导致结果出现很大的不同。在大多数应用程序中，实际的文本将来自于一些外部源（例如数据库）。在实际中主要有两种情形：

- ▶ 文本内容来自外部源，格式化选项来自程序：外部源传递小段文本，然后在程序内部进行合成，并在运行时与格式化选项（在函数调用中）进行组合。
- ▶ 文本内容和格式化选项均来自外部源：包括格式化选项在内的大批文本都来自于外部源。格式设置通过文本中的内嵌选项（用简单选项或宏表示）提供。对于宏，必须对宏定义和宏调用进行区分。这样就允许存在一种有趣的中间形式：文本内容来自于外部源，并包含用于格式化的宏调用。不过，仅在运行时混合宏定义。这样做有个优势，即不需要修改外部文本就可以轻松更改格式。例如，当生成贺卡时，可以通过宏定义不同样式，以使贺卡呈现浪漫风格、技术风格或其他风格。

#### 4.9.4 制表位

在下一个示例中，我们将使用制表符制作一个带有左对齐列和右对齐列的表。此表包含以下文本行，其中的各个项用制表符分隔（用箭头指示）：

```
ITEM → DESCRIPTION → QUANTITY → PRICE → AMOUNT ←
1 → Super Kite → 2 → 20.00 → 40.00 ←
2 → Turbo Flyer → 5 → 40.00 → 200.00 ←
3 → Giga Trash → 1 → 180.00 → 180.00 ←
→ → → → TOTAL 420.00
```

以下代码片段制作一个表，其中，使用 *ruler* 选项定义制表符位置，使用 *tabalignment* 以指定制表位对齐方式，使用 *hortabmethod* 指定用于处理制表位的方法（可以在图 4.17 中查看结果）：

```
' assemble option list
optlist ="ruler      {30      150    250   350}" & _
      "tabalignment {left right right right}" & _
      "hortabmethod ruler leading=120% fontname=Helvetica fontsize=9 encoding=winansi"

' place textflow in fitbox
textflow = oPDF.create_textflow(table, optlist)
oPDF.fit_textflow textflow, left_x, left_y, right_x, right_y, ""
oPDF.delete_textflow textflow
```

#### 4.9.5 编号列表

以下示例演示如何使用内嵌选项 *leftindent* 设置编号列表的格式（可在图 4.18 中查看结果）：



leftindent = &indent  
parindent = - &indent

图 4.19  
使用宏的  
编号列表

- 
1. Long Distance Glider: With this paper rocket you can send all your messages even when sitting in a hall or in the cinema pretty near the back.
  2. Giant Wing: An unbelievable sailplane! It is amazingly robust and can even do aerobatics. But it is best suited to gliding.
  3. Cone Head Rocket: This paper arrow can be thrown with big swing. We launched it from the roof of a hotel. It stayed in the air a long time and covered a considerable distance.

```
1.<leftindent 10>Long Distance Glider: With this paper rocket you can send all
your messages even when sitting in a hall or in the cinema pretty near the back. ↵
<leftindent 0>2.<leftindent 10>Giant Wing: An unbelievable sailplane! It is amazingly
robust and can even do aerobatics. But it is best suited to gliding. ↵
<leftindent 0>3.<leftindent 10>Cone Head Rocket: This paper arrow can be thrown with big
swing. We launched it from the roof of a hotel. It stayed in the air a long time and
covered a considerable distance.
```

设置和重新设置缩排值比较麻烦，特别是每个段落都需要进行此操作。更加完美的解决方案是定义一个名为 *list* 的宏。为了方便起见，其中定义一个宏 *indent* 用作常量。宏定义如下所示：

```
<macro {
indent {25}

list {parindent=-&indent leftindent=&indent hortabsize=&indent
hortabmethod=ruler ruler={&indent}}
}>
<&list>1. → Long Distance Glider: With this paper rocket you can send all your messages
even when sitting in a hall or in the cinema pretty near the back. ↵
2. → Giant Wing: An unbelievable sailplane! It is amazingly robust and can even do
aerobatics. But it is best suited to gliding. ↵
3. → Cone Head Rocket: This paper arrow can be thrown with big swing. We launched
it from the roof of a hotel. It stayed in the air a long time and covered a
considerable distance.
```

*leftindent* 选项定义距离左页边距的距离。*parindent* 选项被设置为 *leftindent* 的负值，以便取消缩排每个段落的第一行。选项 *hortabsize*、*hortabmethod* 和 *ruler* 指定与 *leftindent* 对应的制表位。它使数字后面的文本缩排 *leftindent* 中指定的距离。图 4.19 显示了所使用的 *parindent* 和 *leftindent* 选项。

4.9.6 控制字符、字符映射和符号字体

**textflow** 中的控制字符 可对 **textflow** 中的不同字符进行特殊的处理。PDFlib 支持符号字符名称，此类名称可以用于替换 *charmapping* 选项（在处理文本之前替换其中的字符，请

- 
1. Long Distance Glider: With this paper rocket you can send all your messages even when sitting in a hall or in the cinema pretty near the back.
  2. Giant Wing: An unbelievable sailplane! It is amazingly robust and can even do aerobatics. But it is best suited to gliding.
  3. Cone Head Rocket: This paper arrow can be thrown with big swing. We launched it from the roof of a hotel. It stayed in the air a long time and covered a considerable distance.

图 4.18  
编号列表

参见以下内容) 中对应的字符代码。表 4.3 列出了 `textflow` 函数使用的所有控制字符以及相应的符号名称, 并解释各自的含意。虽然一个选项列表只能使用一个选项, 但是可以接连提供多个选项列表。例如, 以下序列将创建一个空行:

```
<nextline><nextline>
```

替换字符或字符序列 `charmapping` 选项可以用于使用其他字符替换文本中的一些字符。让我们从简单的示例开始, 在此示例中我们将使用空格字符替换文本中的所有制表符。可通过 `charmapping` 选项来达到此目的, 如下所示:

```
charmapping {horthab space}
```

此命令使用了符号字符名称 `horthab` 和 `space`。可以在表 4.3 中找到所有已知字符名称的列表。若要一次实现多个映射, 可以使用以下命令, 使用空格字符替换所有制表符和换行符组合:

```
charmapping {horthab space CRLF space LF space CR space}
```

以下命令移除所有自动连字符:

```
charmapping {shy {shy 0}}
```

每个制表符都将被替换为四个空格字符:

```
charmapping {horthab {space 4}}
```

每个任意长度的换行符序列都将减少为单个换行符:

```
charmapping {linefeed {linefeed -1}}
```

每个 CRLF 组合序列都将被替换为单个空格:

```
charmapping {CRLF {space -1}}
```

我们将更进一步地讨论最后一个示例。让我们假定, 您接收的文本中的行已由一些其他软件通过固定的换行符分离, 因此无法正确地设置这些行的格式。您需要用空格符替换这些换行符以便在限定框内实现正确的格式设置。若要达到此目的, 我们用单个空格符替换任意长度的换行符序列。最初的文本如下所示:

```
To fold the famous rocket looper proceed as follows:  ← ←
Take a sheet of paper.Fold it ←
lengthwise in the middle. ←
Then, fold down the upper corners.Fold the ←
long sides inwards ←
that the points A and B meet on the central fold.
```

以下代码片段演示如何替换多余的换行符以及如何对替换后的文本设置格式:

```
' assemble option list
optlist = "fontname=Helvetica fontsize=9 encoding=winansi alignment=justify " & _
        "charmapping {CRLF {space -1}}"
' place textflow in fitbox
textflow = oPDF.create_textflow(text, optlist)
oPDF.fit_textflow textflow, left_x, left_y, right_x, right_y, ""
oPDF.delete_textflow textflow
```

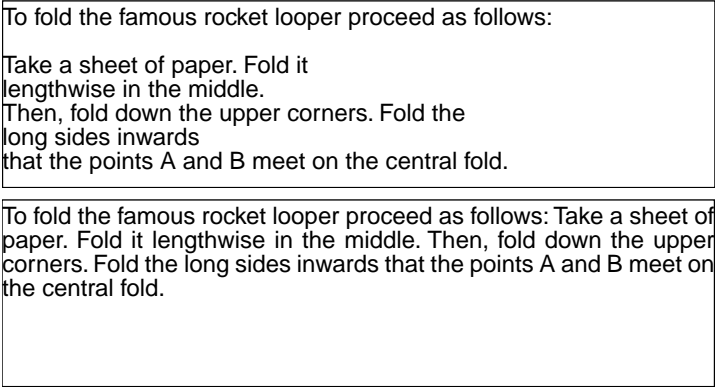


图 4.20  
顶部：带有多余的换行符的文本  
底部：使用 *charmapping* 选项替换换行符

图 4.20 显示了文本未经修改的 *textflow* 输出以及使用 *charmapping* 选项修复过的 *textflow* 输出。

**textflow** 中的符号字体 在 *textflow* 中使用符号字体（更准确地说，应是用第 8o 页上的第 4.5.6 节“Unicode 兼容的字体”表中列出的字体之外的非 Unicode 兼容的字体的文本）时，应特别注意：

- ▶ 表 4.3 中列出的控制字符将不会进行特殊处理，也就是说这些控制字符没有特殊含意。
- ▶ 一些 *textflow* 选项会因为对符号字体没有任何意义而被忽略，例如 *tabalignchar*。
- ▶ 表 8.24 列出了将对非 Unicode 兼容字体忽略的所有选项。
- ▶ 因为内嵌选项列表不能在带有符号字体的文本部分中使用（由于符号没有内在含意，因此不可能定位和解释选项列表），必须使用 *textlen* 选项显式指定由符号字符组成的文本片段的长度。
- ▶ 在 *textlen* 字符的后面，必须在文本中放置一个新的内嵌选项列表以便转换为其他字体 / 编码组合。

以下文本包含一个来自符号字体的、插入在拉丁字符之间的字形：

```
<fontname=Helvetica fontsize=12 encoding=winansi>The Greek letter  
<fontname=Symbol encoding=builtin textlen=1>Α  
<fontname=Helvetica encoding=winansi> symbolizes beginning.
```

若在 *Symbol* 片段中省略 *textlen* 选项，或未能在 *Symbol* 片段后立即提供其他内嵌选项，将产生一个异常。

4.9.7 断字

*PDFlib* 不会自动对文本进行断字，但是可以在出现由自动连字符明确标记的断字机会时实现断字。自动连字符在 Unicode 中的位置是 *U+00AD*，但是也可以使用其他一些方法在非 Unicode 环境中指定自动连字符：

- ▶ 在所有 *cp1250* – *cp1258*（包括 *winansi*）和 *iso8859-1* – *iso8859-16* 编码中，自动连字符对应于十进制 173、八进制 255 或十六进制 *0xAD*。
- ▶ 在 *ebcdic* 编码中，自动连字符对应于十进制 202、八进制 312 或十六进制 *0xCA*。
- ▶ 若编码未包含自动连字符，则可以使用字符实体引用，例如 *macroman*（请参见第 79 页上的第 4.5.5 节“字符引用”）。&shy；

除了自动连字符指定的中断机会以外，在极端情况下，当其他调整方法（例如更改字间距或挤压文本）不可用时，可以强行断字。

注： *PDFlib* 遵守 *CIK* 字符的排版规则。

Our paper planes are the ideal way of passing the time. We offer revolutionary brand new developments of the traditional common paper planes. If your lesson, conference, or lecture turn out to be deadly boring, you can have a wonderful time with our planes. All our models are folded from one paper sheet. They are exclusively folded without using any adhesive. Several models are equipped with a folded landing gear enabling a safe landing on the intended location provided that you have aimed well. Other models are able to fly loops or cover long distances. Let them start from a vista point in the mountains and see where they touch the ground.

图 4.21

带有自动连字符的分散对齐的文本，使用默认设置和宽限定框

Our paper planes are the ideal way of passing the time. We offer revolutionary brand new developments of the traditional common paper planes. If your lesson, conference, or lecture turn out to be deadly boring, you can have a wonderful time with our planes. All our models are folded from one paper sheet. They are exclusively folded without using any adhesive. Several models are equipped with a folded landing gear enabling a safe landing on the intended location provided that you have aimed well. Other models are able to fly loops or cover long distances. Let them start from a vista point in the mountains and see where they touch the ground.

图 4.22

不带自动连字符的分散对齐的文本，使用默认设置和宽限定框。

带有或不带自动连字符的分散对齐的文本 在以下的示例中，我们将使用分散对齐的方式打印下面的文本。文本中包含自动连字符（在此处以短线显示）：

Our paper planes are the ideal way of pas - sing the time. We offer revolu - tionary brand new dev - elop - ments of the tradi - tional common paper planes. If your lesson, confe - rence, or lecture turn out to be deadly boring, you can have a wonder - ful time with our planes. All our models are folded from one paper sheet. They are exclu - sively folded without using any adhe - sive. Several models are equip - ped with a folded landing gear enab - ling a safe landing on the intended loca - tion provided that you have aimed well. Other models are able to fly loops or cover long dist - ances. Let them start from a vista point in the mount - ains and see where they touch the ground.

图 4.21 显示了使用默认选项设置生成的分散对齐的文本输出。由于各项条件都很好，因此文本输出很美观：限定框足够宽，并且自动连字符指定了一些明显的中断机会。从图 4.22 中可以看到，即使没有明确的自动连字符，输出效果看上去也不错。两种情况下的选项列表如下所示：

```
fontname=Helvetica fontsize=9 encoding=winansi alignment=justify
```

## 4.9.8 控制换行算法

PDFlib 执行一种复杂的换行算法。<sup>1</sup>表 4.6 列出了可用来控制换行算法的 `textflow` 选项。

1. 对此感兴趣的用户请注意，PDFlib 遵从“Unicode 标准附录 #14：换行属性”中的建议（请参见 [www.unicode.org/reports/tr14](http://www.unicode.org/reports/tr14)）。不考虑组合标记。

表 4.6 用于控制换行算法的选项

选项	类型	说明
<i>adjustmethod</i>	关键字	如果在压缩或扩展单词之间的距离后文本部分无法在某个行中放下，此时用来调整行的方法需要考虑到由 <i>minspacing</i> 和 <i>maxspacing</i> 选项指定的限制。默认值: <i>auto</i> <i>auto</i> 按顺序应用以下方法: <i>shrink</i> 、 <i>spread</i> 、 <i>nofit</i> 、 <i>split</i> 。 <i>clip</i> 与 <i>nofit</i> 相同（请参见以下内容），但将剪切位于限定框右边缘的长文本部分（将考虑 <i>rightindent</i> 选项）。 <i>nofit</i> 最后一个单词将移动到下一行，前提是剩余（短）行的长度不会比 <i>nofitlimit</i> 选项中指定的百分比长度短。在这种情况下，甚至分散对齐的段落看上去也会有点错乱。 <i>shrink</i> 若一个单词在行中放不下，则会根据 <i>shrinklimit</i> 选项压缩文本直到可以将该单词完整放入。若它仍无法放入到行中，则将应用 <i>nofit</i> 方法。 <i>split</i> 最后的单词将不会移动到下一行，但会强制断字。对于文本字体，将插入一个连字符，但符号字体除外。 <i>spread</i> 最后的字将移动到下一行，并服从 <i>spreadlimit</i> 选项，通过增加字中的字符之间的距离对齐剩余的行。若仍无法对齐，将应用 <i>nofit</i> 方法。
<i>avoidbreak</i>	布尔值	若为 <i>true</i> ，则避免任何换行直到 <i>avoidbreak</i> 重置为 <i>false</i> 。默认值: <i>false</i>
<i>hyphenchar</i>	整数	用于在换行处替换自动连字符的字符的 <i>Unicode</i> 值。默认值: 若在字体中可用，则为 <i>U+00AD</i> （自动连字符），否则为 <i>U+002D</i> （减号连字符）
<i>maxspacing</i> <i>minspacing</i>	浮点数或百分比	指定单词之间的最大距离或最小距离（通过用户坐标表示，或作为空格字符宽度的百分比）。计算的字间距受提供的值的限制（但是仍将加上 <i>wordspacing</i> 选项的值）。默认值: <i>minspacing=50%</i> , <i>maxspacing=500%</i>
<i>nofitlimit</i>	浮点数或百分比	<i>nofit</i> 方法对行的长度的下限（通过用户坐标表示，或作为限定框宽度的百分比）。默认值: <i>75%</i> 。
<i>shrinklimit</i>	百分比	<i>shrink</i> 方法对压缩文本的下限: 计算的 <i>shrinking</i> 因素受提供的值的限制，但将用 <i>horizscaling</i> 选项的值相乘。默认值: <i>85%</i> 。
<i>spreadlimit</i>	浮点数或百分比	<i>spread</i> 方法对两个字符之间的距离的上限（通过用户坐标表示，或作为字体大小的百分比）: 计算的值将与 <i>charspacing</i> 选项的值相加。默认值: <i>0</i> 。

换行规则 当某个单词或由空格字符环绕的其他文本序列无法放入到行中时，必须将它移动到下一行。在这种情况下，换行算法决定可能在哪个字符后换行。

例如，决不会断开像 *-12+235/8\*45* 这样的公式，而字符串 *PDF-345+LIBRARY* 可能会在减号字符处断开换到下一行。若文本包含自动连字符，则也可以在该字符后断开。

对于圆括号和引号，它取决于我们是否有一个开始或结束字符: 开始圆括号和开始引号不会提供任何断开机会。为了查找引号是否开始或结束一个序列，需要检查引号对。

通常内嵌选项列表不会创建换行机会以允许字内的选项更改。不过，当选项列表由空格字符环绕时，在选项列表的开始处有换行机会。若换行出现在选项列表处且 *alignment=justify*，则将忽略选项列表前面的空格。选项列表后面的空格将保留，并将出现在下一行的开始处。

Our paper planes  
are the ideal way of  
passing the time. We  
offer revolutionary  
brand new develop-  
ments of the traditional  
common paper planes.  
If your lesson, conf-  
erence, or lecture  
turn out to be deadly  
boring, you can have  
a wonderful time  
with our planes. All  
our models are  
folded from one  
paper sheet. They  
are exclusively folded  
without using any

减少字间距（默认方法， **minspacing** 选项）

压缩行（**shrink** 方法， **shrinklimit** 选项）

强制使用连字符（**split** 方法）

增加字间距（默认方法， **maxspacing** 选项）

图 4.23  
在窄限定框中使用默认设置分散对齐的文本

窄限定框中分散对齐的文本 限定框越窄，就需要越多地处理用于控制分散对齐文本的选项。图 4.23 演示在窄限定框中用于调整文本的多种方法的结果。图 4.23 中的选项设置基本上合适，但 **maxspacing** 提供的字间距相当大。不过，建议为窄限定框保留此选项设置，因为若不这样，由 **split** 方法引起的难看的强制断字将会更多地出现。

若限定框太窄以至于出现很少使用的强制断字，则应考虑插入连字符，或修改控制分散对齐文本的选项。

用于分散对齐文本的选项 **shrinklimit** 视觉效果最好的解决方案是减少 **shrinklimit** 选项的值，此选项指定 **shrink** 方法应用的收缩因子的下限。图 4.24 显示如何通过将文本压缩 50% 来避免强制断字。选项列表如下所示：

fontname=Helvetica fontsize=9 encoding=winansi alignment=justify shrinklimit=50%

用于分散对齐文本的选项 **spreadlimit** 扩展文本是用来控制换行的另外一种方法，它由 **spread** 方法实现并由 **spreadlimit** 选项控制。不过，这种方法的效果并不令人满意，很少使用。图 4.25 演示了最大字符间距为 5 个单位的文本。选项列表如下所示：

Our paper planes  
are the ideal way of  
passing the time. We  
offer revolutionary  
brand new develop-  
ments of the traditional  
common paper planes.  
If your lesson, conference,  
or lecture turn out to be  
deadly boring, you can  
have a wonderful time  
with our planes. All our  
models are folded from  
one paper sheet. They  
are exclusively folded  
without using any  
adhesive.

图 4.24  
将行压缩 50%

压缩行（**shrink** 方法， **shrinklimit** 选项）

Our paper planes  
are the ideal way of  
passing the time. We  
offer revolutionary  
brand new develop-  
ments of the traditional  
common paper planes.  
If your lesson,  
conference, or lecture  
turn out to be deadly  
boring, you can have  
a wonderful time  
with our planes.

缩短行（**nofit** 方法，**nofitlimit** 选项）

图 4.26  
最小宽度为 50% 的分散对齐文本

```
fontname=Helvetica fontsize=9 encoding=winansi alignment=justify spreadlimit=5
```

用于分散对齐文本的选项 **nofitlimit** *nofitlimit* 选项在应用 *nofit* 方法时控制行缩小的程度。当限定框非常窄时，减少默认值 75% 相对于强制断字更为可行。图 4.26 显示了生成的最小文本宽度为 50% 的文本输出。选项列表如下所示：

```
fontname=Helvetica fontsize=9 encoding=winansi alignment=justify nofitlimit=50
```

4.9.9 使用 **textflow** 格式化 CJK 文本

**textflow** 引擎现在可以按照 Unicode 标准将 CJK 字符作为表意字形进行处理。因此，CJK 文本将决不会断字。为了改进格式设置，当使用带有 CJK 文本的 **textflow** 时，推荐使用以下选项；这些选项将对插入的拉丁文本禁用断字并创建间距均匀的文本输出：

```
hyphenchar=none  
alignment=justify  
shrinklimit=100%  
spreadlimit=100%
```

对 CJK 文本使用 **textflow** 时，请注意以下限制：

- ▶ 不支持垂直书写模式。
- ▶ 只能使用 Unicode 兼容的编码，即 *unicode* 或 Unicode 兼容的预定义 CMap 之

Our paper planes  
are the ideal way of  
passing the time. We  
offer revolutionary  
brand new develop-  
ments of the traditional  
common paper planes.  
If your lesson,  
conference, or lecture  
turn out to be deadly  
boring, you can have  
a wonderful time  
with our planes.

图 4.25  
最大字符间距为 5 个单位的分散对齐文本

扩展行（**spread** 方法，**spreadlimit** 选项）





## 5 导入和放置对象

PDFlib 提供多种功能以便从现有的 PDF 文档导入光栅图像和页，并将它们放置到页面上。本章介绍有关处理光栅图像和从现有 PDF 文档导入页的详细信息。它还显示一些示例，这些示例演示如何在输出页面上放置图像和 PDF 页。

### 5.1 导入光栅图像

#### 5.1.1 基本图像处理

在 PDFlib 中嵌入光栅图像可以轻松实现。首先，必须用一个 PDFlib 函数打开图像文件，对图像参数进行简短的分析。`load_image()` 函数返回一个句柄用作图像说明。此句柄可以随同 `positioning` 和 `scaling` 参数，在对 `fit_image()` 的调用中使用：

```
lImage = oPDF.load_image("auto", "image.jpg", "")
If (lImage = -1) Then
    MsgBox "Couldn't read image."
End
Else
    oPDF.fit_image lImage, 0, 0, ""
    oPDF.close_image lImage
End If
```

`fit_image()` 函数的最后的参数是一个选项列表，它支持用于定位、缩放和旋转图像的各种选项。有关这些选项的详细信息将在第 113 页上的第 5.3 节“放置图像和导入的 PDF 页”中讨论。

**重用图像数据** PDFlib 支持一项重要的 PDF 优化技术，以便使用重复的光栅图像。让我们以在多个页面上使用的不变的徽标或背景的布局为例。在这种情况下，可以只在 PDF 中包含一次实际图像，并在使用该图像的每个页面上仅生成一个引用。每次需要在特殊页面上放置徽标或背景时，只需加载一次图像文件，并调用 `fit_image()`。用户也可以在多个页面上放置图像，或对出现在不同位置的相同图像使用不同的缩放因素（只要图像没有被关闭）。根据图像的尺寸和出现次数，此技术会导致产生巨大的空间存储需要。

**内嵌图像** 与可重用的图像（作为图像 XObject 写入到 PDF 输出）相反，内嵌图像直接写入相应的内容流（页、模式、模板或字形说明）中。这将产生一些空间存储需要，但是根据 PDF Reference 中的建议，只应用于数量不大的图像数据（最多 4 KB）。内嵌图像主要是在 Type 3 字体的点阵字形说明中使用。

通过提供 `inline` 选项，可以使用 `load_image()` 接口生成内嵌图像。内嵌图像不可重用，也就是说，相应的句柄不得提供给接受图像句柄的任何调用。为此，若 `inline` 选项已提供，则 `load_image()` 在内部执行以下代码的等效代码：

```
oPDF.fit_image lImage, 0, 0, ""
oPDF.close_image lImage
```

**缩放和 dpi 计算** PDFlib 从不更改导入的图像中的像素的数量。缩放不是扩大图像像素就是缩小图像像素，但是不会执行任何缩减像素取样（图像中的像素数量将总是保持不变）。在用户坐标中，缩放因素为 1 导致像素尺寸为 1 单位。换言之，若尚未缩放用户坐标系统，则将使用本地分辨率导入图像（若未包含任何分辨率信息，则为 72 dpi，因为默认每英寸有 72 个单位）。

已导入图像的色彩空间 除了添加或移除 ICC 配置文件并根据 `load_image()` 中提供的选项使用专色之外, **PDFlib** 通常将保留已导入图像的本地色彩空间。不过, 这对一些不常见的组合不可行, 例如 **TIFF** 中的 **YCbCr** 将转换为 **RGB**。

**PDFlib** 不执行 **RGB** 和 **CMYK** 之间的任何转换。若这样的转换是必需的, 则在 **PDFlib** 中加载图像之前, 必须对图像数据应用此转换。

## 5.1.2 受支持的图像文件格式

**PDFlib** 处理以下描述的图像文件格式。默认情况下, 如果可能的话, **PDFlib** 将压缩的图像数据无改变地传递给 **PDF** 输出, 因为 **PDF** 内部支持在常见图像文件格式中使用的大多数压缩方案。此技术 (在以下的描述中称作 *pass-through* 模式) 导致快速图像导入, 因为此技术不再需要对图像数据进行解压缩以及随后的再压缩。不过, 在这种模式下, **PDFlib** 无法检查已压缩图像数据的完整性。当在 **Acrobat** 中使用 **PDF** 文档时, 不完整的图像数据或损坏的图像数据可能导致错误或警告消息 (例如读取比预期少的图像数据)。可以使用 `load_image()` 的 *passthrough* 选项控制 *Pass-through* 模式。

若无法成功导入图像文件, `load_image()` 将返回一个错误代码。若想知道有关图像失败的更多详细信息, 请调用 `get_errmsg()` 以检索详细的错误消息。

**PNG** 图像 **PDFlib** 支持所有风格的 **PNG** 图像 (**ISO 15948**)。多数情况下, 都是在 *pass-through* 模式中处理 **PNG** 图像。用了隔行扫描或包含一个 **alpha** 通道 (此通道总会丢失, 请参见以下内容) 的 **PNG** 图像必须是未压缩的, 这样将比 *pass-through* 模式花多很多时间。若 **PNG** 图像包含透明度信息, 则在生成的 **PDF** 中保留透明度 (请参见第 108 页上的第 5.1.3 节 “图像蒙版和透明度”)。不过, **alpha** 通道不受 **PDFlib** 支持。

**JPEG** 图像 从不对 **JPEG** 图像 (**ISO 10918-1**) 进行解压缩, 但是一些风格可能需要代码转换以便在 **Acrobat** 中正确显示。**PDFlib** 对一些关键类型的 **JPEG** 图像自动使用代码转换, 但是也可以通过 `load_image()` 的 *passthrough* 选项控制代码转换。**PDFlib** 支持以下 **JPEG** 图像风格:

- ▶ 灰度、**RGB** (通常编码为 **YCbCr**) 和 **CMYK** 颜色。
- ▶ 基线 **JPEG** 压缩, 被大量 **JPEG** 图像采用。
- ▶ 渐进 **JPEG** 压缩。

可以用一些不同的文件格式对 **JPEG** 图像进行打包。**PDFlib** 支持所有常见的 **JPEG** 文件格式, 并将读取以下风格的分辨率信息:

- ▶ **JFIF**, 由多种图像应用程序生成。
- ▶ **JPEG** 文件由 **Adobe Photoshop** 和其他 **Adobe** 应用程序编写。**PDFlib** 采用了一个必要的替代方法以正确处理 **Photoshop** 生成的 **CMYK JPEG** 文件。

注: **PDFlib** 不解释 *SPIFF* 文件格式的 **JPEG** 图像中的分辨率信息, 也不解释 *EXIF* 文件格式的 **JPEG** 图像中的色彩空间信息。

**JPEG2000** 图像 **JPEG2000** 图像 (**ISO 15444-2**) 需要 **PDF 1.5** 或更高版本, 并总是在 *pass-through* 模式中进行处理。**PDFlib** 支持 **JPEG2000** 图像, 如下所示:

- ▶ **JP2** 和 **JPX** 基线图像 (通常是 *\*.jp2* 或 *\*.jpf*) 受支持, 但具有以下色彩空间条件约束。支持所有有效的颜色深度值。
- ▶ 支持以下色彩空间: **sRGB**、**sRGB-grey**、**ROMM-RGB**、**sYCC**、**e-sRGB**、**e-sYCC**、**CIELab**、基于 **ICC** 的色彩空间 (受限制的或全部 **ICC** 配置文件) 和 **CMYK**。**PDFlib** 将不会改变 **JPEG2000** 图像文件中的原始色彩空间。
- ▶ 包含软蒙版的图像可以与 *mask* 选项一起使用以准备一个可以应用于其他图像的蒙版。

- ▶ 无法对 JPEG2000 图像应用外部 ICC 配置文件，也就是说，禁止使用 *iccprofile* 选项。并将一直保留 JPEG2000 图像文件中包含的 ICC 配置文件，也就是说，*honoriccprofile* 选项总是为 *true*。
- ▶ JPEG2000 图像不支持 *colorize* 选项。

注：根据 ISO 15444-6（通常为 \*.jpm），不带 JPX 包装的原始 JPEG2000 代码流（通常为 \*.j2k）和 JPM 复合图像文件将不被支持。

**GIF 图像** PDFlib 支持带有隔行扫描和未隔行扫描的像素数据的所有 GIF 风格（特别是 GIF 87a 和 89a）和所有调色板尺寸。并总是使用 Flate 压缩对 GIF 图像进行再压缩。

**TIFF 图像** PDFlib 将在 pass-through 模式中处理多数 TIFF 图像。PDFlib 支持以下风格的 TIFF 图像：

- ▶ 压缩方案：在 pass-through 模式中处理未压缩、CCITT（group 3、group 4 和 RLE）、ZIP (=Flate) 和 PackBits (=RunLength)；其他压缩方案如 LZW 和 JPEG 将通过解压缩处理。
- ▶ 颜色：黑白、灰度、RGB、CMYK、CIELab 和 YCbCr 图像；将忽略在文件中可能出现的任何 alpha 通道或蒙版。
- ▶ 包含多个的图像 TIFF 文件（请参见第 110 页上的第 5.1.5 节“多页图像文件”）
- ▶ 颜色深度必须是每个颜色采样 1、2、4、8 或 16 位。在 PDF 1.5 模式中，多数情况下，16 位颜色深度将与 pass-through 模式一起保留，但是对于某些图像文件会减少到 8 位（带有 little-endian/Intel 字节顺序和 16 位调色板图像的 ZIP 压缩）。

在 PDF 文件中，多带区 TIFF 图像会转换为多个图像，PDF 文件视觉上完全再现了原始图像，但是可以使用 Acrobat 的 TouchUp 对象工具选择个别图像。可以使用 TIFFlib 包中的 *tiffcp* 命令行工具将多带区 TIFF 图像转换为单带区图像。<sup>1</sup> ImageMagick<sup>2</sup> 工具总是写入单带区 TIFF 图像。

PDFlib 可完全处理方向标签，该标签在一些 TIFF 文件中指定所需的图像方向。通过将 *ignoreorientation* 选项设置为 *true*，可以指示 PDFlib 忽略方向标签（和许多应用程序的做法一样）。

一些 TIFF 功能（如专色）和某些功能组合（如带蒙版的 CMYK 图像）都不受支持。尽管带 JPEG 压缩的 TIFF 图像通常都受支持，但一些所谓旧式 TIFF-JPEG 风格的 TIFF 图像除外。

**BMP 图像** 在 pass-through 模式中无法处理 BMP 图像。PDFlib 支持以下风格的 BMP 图像：

- ▶ BMP 版本 2 和 3；
- ▶ 每组件的颜色深度为 1、4 和 8 位，包括 3 x 8 = 24 位 TrueColor。16 位图像将作为 5+5+5 以及 1 个未使用的位进行处理。32 位图像将作为 3 x 8 位图像处理（剩余的 8 位将被忽略）。
- ▶ 黑白或 RGB 颜色（索引的和直接的）；
- ▶ 未压缩以及 4 位和 8 位 RLE 压缩；
- ▶ 若像素是按自下而上的顺序存储，则 PDFlib 将不映像图像（这是 BMP 中很少使用的功能，在各应用程序中将进行不同的解释）。

**CCITT 图像** Group 3 或 Group 4 传真压缩的图像数据总是在 pass-through 模式中处理。请注意，这种格式实际意味着原始 CCITT 压缩的图像数据，而不是使用 CCITT 压缩的 TIFF 文件。原始 CCITT 压缩的图像文件通常在终端用户应用程序中不受支持，而只能使用与传真

1. 请参见 [www.libtiff.org](http://www.libtiff.org)

2. 请参见 [www.imagemagick.org](http://www.imagemagick.org)

有关的软件生成。因为 PDFlib 无法分析 CCITT 图像，所以客户端必须将所有相关的图像参数传递给 `load_image()`。

原始数据 未压缩的（原始）图像数据对一些特殊的应用程序可能很有用。可以从颜色分量的数量来推导图像的本质：1 个分量表示灰度图像，3 个分量表示 RGB 图像，4 个分量表示 CMYK 图像。

### 5.1.3 图像蒙版和透明度

**PDF 中的透明度** PDF 支持各种透明度功能，所有这些功能都在 PDFlib 中实现：

- ▶ 根据位置蒙版：图像可能携带内部信息“打印前景或背景”。这可以由 1 位蒙版图像实现，在目录图像中使用经常会使用到它。
- ▶ 根据颜色值蒙版：某种颜色的像素不会绘制，而页面上先前绘制的部分透射出来（即，忽略图像中所有蓝色像素）。在电视和视频技术中，这也称作蓝色屏蔽，经常用于将气象人员和气象图组合到一个图像中。
- ▶ PDF 1.4 介绍 alpha 通道或软蒙版。它们可以用于创建前景和背景之间的平滑转换，或创建半透明对象（混合图像和背景）。软蒙版由带 1-8 位 / 像素的 1 分量图像表示。

PDFlib 支持图像中的三种透明度信息：隐式透明度、显式透明度和图像蒙版。

注：蒙版必须与底层的图像具有相同的方向，否则将会被拒绝。因为方向取决于图像文件格式和其他因素，所以难以检测。为此，建议对蒙版和图像使用相同的文件格式和创建软件。

**隐式透明度** 在隐式情况下，将使用外部图像文件中的透明度信息，前提是图像文件格式支持透明度或 alpha 通道（并非所有图像文件格式都是如此）。在以下的图像文件格式中检测透明度信息：

- ▶ GIF 图像文件可能包含一个 PDFlib 遵从的透明度颜色值。
- ▶ PNG 图像文件可能包含一些风格的透明度信息或整个 alpha 通道。PDFlib 将保留单个透明度颜色值；若给定带附加 alpha 值的多个颜色值，则仅使用第一个带有 50% 以下的 alpha 值的颜色值。忽略整个 alpha 通道。

**显式透明度** 显式情况下要求两个涉及图像操作的步骤。首先，必须准备好稍后作为透明度蒙版使用的图像。通过使用 `mask` 选项打开图像来完成此操作。在 PDF 1.3（仅支持 1 位蒙版）中，此选项为必选；在 PDF 1.4 中，此选项为可选的。以下类型的图像可以用于构造蒙版：

- ▶ PNG 图像
- ▶ TIFF 图像（仅单带区）
- ▶ 原始图像数据

蒙版中的为 0（零）像素值将导致所蒙版图像的相应部分被绘制，而高像素值导致背景透射出来。若像素具有大于 1 位 / 像素，中间值将针对背景混合前景图像，提供透明效果。在第二步中，蒙版应用于其他通过以下图像函数之一获得的图像：

```
lMask = oPDF.load_image("png", MaskFileName, "mask")
lImage = oPDF.load_image(type, FileName, "masked " & lMask)
If (lImage <> -1 And lMask <> -1) Then
    oPDF.fit_image lImage, x, y, ""
Else
    ...
End If
```

请注意，`load_image()` 的选项列表的不同用法：`mask` 用于定义蒙版，`masked` 用于对其他图像应用蒙版。

图像和蒙版可能有不同的像素尺寸；蒙版将指定缩放到图像的尺寸。

注: **PDFlib** 将多带区 **TIFF** 图像转换为多个 **PDF** 图像, 转换后各个 **PDF** 图像将单独蒙版。因为通常不希望这样做, 所以会拒绝将此类图像用作蒙版和所蒙版的目标图像。避免混淆隐式情况和显式情况也很重要, 也就是说, 不要将带有透明颜色值的图像作为蒙版使用。

**图像蒙版** 图像蒙版是指位深度为 1 (位图) 的图像, 在此图像中零位将视为透明: 已在页面上存在的任何内容都将通过图像的透明部分透射出来。对 1 位像素使用当前填充色进行着色。以下类型的图像可以作为图像蒙版使用:

- ▶ **PNG** 图像
- ▶ **TIFF** 图像 (单带区或多带区)
- ▶ **JPEG** 图像 (仅作为软蒙版, 请参见以下内容)
- ▶ **BMP**: 请注意, **BMP** 图像的定向方式与其他图像类型的定向方式不同。为此, 在 **BMP** 图像作为蒙版使用之前, 必须沿 **x** 轴反射此图像。
- ▶ 原始图像数据

使用 **mask** 选项打开图像蒙版, 在完成对所需填充色的设置后, 将该蒙版放置在页面上:

```
lMask = oPDF.load_image("tiff", MaskFileName, "mask")
oPDF.setcolor "fill", "rgb", 1, 0, 0, 0
If (lMask <> -1) Then
    oPDF.fit_image lMask, 0#, 0#, ""
End If
```

若需要对图像 (没有表示透明的零位像素) 应用颜色, 必须使用 **colorize** 选项 (请参见第 109 页上的第 5.1.4 节 “图像着色”)。

**软蒙版** 软蒙版将图像蒙版的概念推广到具有 1 位以上深度的蒙版。PDF 1.4 中已引入了软蒙版, 用于将图像与一些现有背景混合。**PDFlib** 接受所有类型的单通道 (灰度) 图像作为软蒙版。它们的使用方法与图像蒙版相同, 只要 PDF 输出兼容性只要包括 PDF 1.4。

**忽略透明度** 有时, 需要忽略在图像文件中可能包含的任何透明度信息。例如, **Acrobat** 的消除锯齿功能 (也称作 “平滑”) 不能用于仅包含黑色和透明的 1 位图像。为此, 当透明度信息保留在生成的 PDF 中时, 细节丰富的导入图像 (例如, 光栅化文本) 可能看起来很不舒服。为了处理这种情况, 当打开文件时, 可以用 **ignoremask** 选项禁用 **PDFlib** 的自动透明度支持:

```
lImage = oPDF.load_image("gif", FileName, "ignoremask")
```

## 5.1.4 图像着色

与图像蒙版 (对图像的非透明部分应用颜色) 类似, **PDFlib** 支持用专色给图像着色。此功能可用于以下格式的黑白或灰度图像:

- ▶ **BMP**
- ▶ **PNG**
- ▶ **JPEG**
- ▶ **TIFF**
- ▶ **GIF**

对于带 **RGB** 调色板的图像, 仅当调色板只包含灰色值并且调色板索引与灰色值一致时, 着色才算合理。

为了用专色给图像着色, 您必须在加载图像时提供 **colorize** 选项并提供由 **makespotcolor()** 返回的相应的专色句柄:

```
oPDF.setcolor "both", "cmyk", 1, .79, 0, 0
spot = oPDF.makespotcolor("PANTONE Reflex Blue CV")
```

```
lImage = oPDF.load_image("tiff", "image.tif", "colorize " & spot)
If (lImage <> -1) Then
oPDF.fit_image lImage, 0#, 0#, ""
End If
```

### 5.1.5 多页图像文件

PDFlib 支持包含多个图像的 TIFF 文件，也称作“多页文件”。为了使用多页 TIFF，需在对 *load\_image()* 的调用中使用附加的字符串和数字参数：

```
lImage = oPDF.load_image("tiff", FileName, "page 2")
```

*page* 选项指示要使用多图像文件。最后一个参数指定要使用的图像的编号。第一个图像的编号为 1。此选项可以增加直到 *load\_image()* 返回 -1（假定 *imagewarning=false*），发出信号表示文件中再没有图像可用。

类似以下的代码片段可以用于将多图像 TIFF 文件中的所有图像转换为多页 PDF 文件：

```
Frame = 1
Do
    lImage = oPDF.load_image("tiff", FileName, "page " & Frame)
    If (lImage = -1) Then Exit Endif
    oPDF.begin_page lWidth, lHeight
    oPDF.fit_image lImage, 0, 0, ""
    oPDF.close_image lImage
    oPDF.end_page
    Frame = Frame + 1
Loop
```

### 5.1.6 OPI 支持

当加载图像时，根据 OPI（开放印前作业界面）版本 1.3 或 2.0，在对 *load\_image()* 的调用中将提供附加信息。PDFlib 接受所有标准 OPI 1.3 或 2.0 PostScript 注释作为选项（不是相应的 PDF 关键字！），并将提供的 OPI 信息传递到生成的 PDF 输出（不做任何修改）。以下示例将 OPI 信息附加到图像：

```
optlist13 =
"OPI-1.3 { ALDImageFilename bigfile.tif _
ALDImageDimensions {400 561} _
ALDImageCropRect {10 10 390 550} _
ALDImagePosition {10 10 10 540 390 540 390 10} } _

image = oPDF.load_image("tiff", filename, 0, optlist13)
```

注：一些 OPI 服务器（例如 *Helios EtherShare* 中包含的一种 OPI 服务器）不能正确实现对 PDFlib 在默认情况下生成的 *PDF ImageXObject* 的 OPI 处理。在这种情况下，通过对 *load\_image()* 提供模板选项可以强制生成 *Form XObject*。

## 5.2 使用 PDI（PDF 导入库）生成 PDF 页

注：本节中描述的所有函数都需要 *PDFlib+PDI*。PDF 导入库 (PDI) 未包含在 *PDFlib* 或 *PDFlib Lite* 中。尽管 *PDI* 已经在所有预编译版本的 *PDFlib* 中集成，但是仍需要 *PDI*（或包含 *PDI* 的 *PPS*）的许可密钥。

### 5.2.1 PDI 功能和应用程序

若可选 PDI（PDF 导入）库已附加到 PDFlib，则可以导入现有 PDF 文档中的页。PDI 包含用于 PDF 文件格式的分析器，并准备现有 PDF 文档中的页以便轻松使用 PDFlib。从概念上讲，导入的 PDF 页与导入的光栅图像（例如 TIFF 或 PNG）的处理方法类似。打开一个 PDF 文档，选择要导入的页，然后将该页放置在输出页面上，应用任何 PDFlib 的转换函数以转换，缩放、旋转或歪移导入的页。将导入的 PDF 页放置在输出页面上后，通过使用任何 PDFlib 的文本或图形函数，可以轻松将到导入的页与新内容组合（将导入的页看作新内容的背景）。使用 PDFlib 和 PDI，您可以轻松完成以下任务：

- ▶ 覆盖多 PDF 文档中的两个或多个页（例如对现有文档添加信纸格式以模拟预先印刷的信纸）；
- ▶ 在现有文档中放置 PDF 广告；
- ▶ 剪裁 PDF 页面的可见区域以便除去不需要的元素（例如，剪裁标记）或缩放页；
- ▶ 在一张纸上打印多页；
- ▶ 处理符合多 PDF/X 的文档以创建新的 PDF/X 文件；
- ▶ 添加一些文本（例如标题、页脚、戳记、页码）或图像（例如，公司徽标）到现有 PDF 页；
- ▶ 将输入文档中的所有页复制到输出文档，并在页面上放置条形码。

为了放置 PDF 背景页并用动态数据填充该页（例如，邮件合并、在 Web 上个性化 PDF 文档、表单填充），我们建议与 PDFlib 块一起使用 PDI（请参见第 6 章）。

### 5.2.2 与 PDFlib 一起使用 PDI 函数

常规注意事项 PDI 将仅导入实际页内容，而并不是在导入的 PDF 文档中可能将出现的任何超文本功能（例如声音、电影、嵌入的文件、超文本链接、表单域、JavaScript、书签、缩略图和注释），了解这一点很重要。可以使用相应的 PDFlib 函数生成这些超文本功能。当导入页面时，PDFlib 块也将被忽略。

不能通过其他 PDFlib 函数重用导入页的各个元素。例如，对一些其他内容不可能重用导入的文档的字体，所有必需的字体都必须在 PDFlib 中配置。若多个导入的文档都包含相同字体的嵌入的字体信息，则 PDI 将不会移除任何重复的字体数据。另一方面，若字体从一些导入的 PDF 中缺失，则这些字体也将从生成的 PDF 输出文件中缺失。作为最佳选择，您应尽可能长时间地保持导入的文档为打开状态，以避免在输出文档中多次嵌入相同的字体。

PDI 无论如何也不会更改导入的 PDF 文档的颜色。例如，若 PDF 包含 ICC 配置文件，则这些文件将在输出文档中保留。

PDFlib 使用模板功能以便在输出页面上放置导入的 PDF 页。因为一些第三方 PDF 软件不能正确支持模板，所以在 Acrobat 之外的某些环境中会存在一些限制（请参见第 49 页上的第 3.2.4 节“模板”）。

对于 PDFlib 生成的、包含从其他 PDF 文档导入的页的输出，可以再次使用 PDFlib+PDI 进行处理。不过，由于 PostScript 打印中的限制，嵌套级别不应超过 10。

用于导入 PDF 页的代码片段 可以使用非常简单的代码结构处理现有 PDF 文档中的页。以下代码片段从现有文档中打开某个页，并将页面内容复制到输出 PDF 文档中的新页上（此文档必须事先已打开）：

```
Dim doc, page, pageno;
Dim filename

...

pageno = 1

doc = oPDF.open_pdi(filename, "", 0)
```

```

If (doc = -1) Then
    MsgBox "Couldn't open input file!"
End
End If

page = oPDF.open_pdi_page(doc, pageno, "")
if (page = -1) Then
    MsgBox "Couldn't open page in input file!"
End
End If

' dummy page size, will be modified by the adjustpage option
oPDF.begin_page 20, 20
oPDF.fit_pdi_page page, 0, 0, "adjustpage"
oPDF.close_pdi_page page

...add more content to the page using PDFlib functions...

oPDF.end_page

```

*fit\_pdi\_page()* 的最后一个参数是一个选项列表，此列表支持用于定位、缩放和旋转导入的页的各种选项。有关这些选项的详细信息将在第 113 页上的第 5.3 节“放置图像和导入的 PDF 页”中讨论。

导入的 **PDF** 页的尺寸 导入的 **PDF** 页的处理方式类似于导入的光栅图像，并可以使用 *fit\_pdi\_page()* 将其放置在输出页面上。默认情况下，**PDI** 会按照页在 **Acrobat** 中的相同显示方式进行导入，具体如下：

- ▶ 裁剪将被保留（从技术上讲，若裁剪框存在，**PDI** 将相对于媒体框优先考虑裁剪框；请参见第 48 页上的第 3.2.2 节“页面大小和坐标限制”）；
- ▶ 已对页使用的旋转将保留。

另外，您可以使用 *pdiusebox* 选项来显式指示 **PDI** 使用页面的任何媒体框、裁剪框、出血框、裁切框或作品框项（若存在），以便确定导入页面的尺寸（有关详细信息，请参见表 8.46）。

许多重要属性（例如导入的 **PDF** 页的宽度和高度）、所有方框项以及文档中的页的数目都可以通过 **PDFlib** 的参数机制查询。相关参数已在表 8.45 和表 8.46 中列出。对导入的 **PDF** 页在输出页上的位置作出决定时，这些属性会有用。

带图层的导入 **PDF** 页 **Acrobat 6 (PDF 1.5)** 引入了图层功能（技术上称作“可选内容”）。**PDI** 将忽略可能在文件中存在的任何图层信息。在导入的页中的所有图层（包括不可见图层）都将在生成的输出中可见。

带 **OPI** 信息的导入的 **PDF** 在输入 **PDF** 中存在的 **OPI** 信息将在输出中毫无修改地保留。

## 5.2.3 可接受的 **PDF** 文档

通常，**PDI** 将乐意处理所有类型的、可以用 **Acrobat** 打开的 **PDF** 文档，而不管在文件内使用的 **PDF** 版本号或功能。为了导入已加密文档（即具有权限设置或口令的文件）中的页，必须提供相应的许可口令。

不过，在少数情况下，**PDF** 文档或文档的某个页会被 **PDI** 拒绝。

默认情况下，若 **PDF** 文档或页无法成功导入，*open\_pdi()* 和 *open\_pdi\_page()* 将返回一个错误代码。不过，若想知道有关失败的更多详细信息，可以使用 *get\_errmsg()* 查询原因。另外，可以将 *pdiwarning* 选项或参数设置为 **true**，这样在文档无法打开的情况下，将引发一个异常：



```
oPDF.set_parameter "pdwarning", "true" ' enable PDI warnings
```

这将导致 `open_pdi()` 和 `open_pdi_page()` 引发一个异常，并在相应的异常消息中提供有关失败的更多详细信息。使用 PDI 无法导入以下类型的 PDF 文档：

- ▶ 其使用的 PDF 版本号比当前要生成的 PDF 输出文档更高的 PDF 文档。因为在导入带有更高版本号的 PDF 之后，PDFlib 不再能够确保输出将实际符合所请求的 PDF 版本。解决方案：使用 `begin_document()` 中的 `compatibility` 选项，将输出 PDF 的版本设置为要求的级别。
- ▶ 其 PDF/X 规范等级与当前输出文档的 PDF/X 级别不兼容的 PDF 文档。
- ▶ 带有损坏的交叉引用表的 PDF 文档。通过 Acrobat 的警告消息“文件已损坏但正在修复”，可以识别这样的文件。解决方案：用 Acrobat 打开并重新保存文件。

另外，默认情况下，将拒绝以下类型的 PDF 文档；不过，通过将 `infomode` 选项设置为 `true`，可以打开这些文档以便查询信息（与导入页相反）：

- ▶ 没有相应的口令的已加密 PDF 文档（例外：使用 Distiller 设置“对象级别压缩：最大值”创建的 PDF 1.6 文档）；
- ▶ 在 `begin_document()` 中的 `tagged` 选项为 `true` 的情况下的标签 PDF。

### 5.3 放置图像和导入的 PDF 页

用于放置光栅图像和模板的 `fit_image()` 函数以及用于放置导入的 PDF 页的 `fit_pdi_page()` 都提供大量的选项以便在页面上控制放置位置。本节通过列举一些常见应用程序任务说明最重要的选项。完整的列表和所有选项的说明可以在表 8.41 中找到。

使用 PDFlib 可以轻松完成光栅图像嵌入。必须使用 `load_image()` 首先加载图像文件。此函数返回一个图像句柄，它可以与定位和缩放选项在 `fit_image()` 中使用。

嵌入导入的 PDF 页的方式相同。必须使用 `open_pdi_page()` 打开 PDF 页来获取页句柄以便在 `fit_pdi_page()` 中使用。可以对光栅图像使用相同的定位和缩放选项。

本节中的所有示例对光栅图像、模板和导入的 PDF 页都起相同的作用。尽管这里仅提供了用于光栅图像的代码示例，但我们谈论的一般意义的对象放置。请注意，在调用任何 `fit` 函数之前，必须发出对 `load_image()` 或 `open_pdi()` 和 `open_pdi_page()` 的调用。为了简单起见，在这里不重现这些调用。

#### 5.3.1 缩放、方向和旋转

**简单放置** 让我们从最简单的情况开始（请参见图 5.1）：按照对象的原始尺寸将其放置在某个位置：

图 5.1  
简单放置

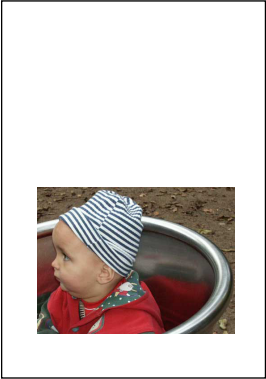


图 5.2  
带缩放比例的放置



```
oPDF.fit_image image, 80, 100, ""
```

在此代码片段中，对象的左下角放置在用户坐标系统的点 **(80,100)** 上。此点称作参考点。选项列表（函数的最后一个参数）为空。这表示按对象的原始尺寸将其放置在提供的参考点上。

在矩形中居中放置图像 为了使图像在预定义的矩形中居中放置，您自己不需要作任何计算，而只需使用如下所示的合适的选项就可以实现：

```
oPDF.fit_image image, 80, 100, "boxsize {200 100} position 50 fitmethod meet"
```

带缩放比例的放置 也可以轻松使用以下变体（请参见图 5.2）。我们按照上一示例中的操作放置对象，但是将修改对象的缩放比例：

```
oPDF.fit_image lImage, 80, 100, "scale 0.5"
```

此代码片段将对象的左下角放置在用户坐标系统的点 **(80,100)** 上。另外，按照缩放因子 **0.5**，沿 **x** 和 **y** 方向缩小对象，使该对象以其原始尺寸的 **50%** 出现。

带方向的放置 在下一个代码片段中，我们将使对象朝西放置（请参见图 5.3）：

```
oPDF.fit_image lImage, 80, 100, "scale 0.5 orientate west"
```

此代码片段将对象向西放置（逆时针方向 **90** 度），然后将对象的左下角（在应用 **orientate** 选项后）转换到参考点 **(x,y)**。该对象将自我旋转。

带旋转的放置 旋转对象（请参见图 5.4）的方式与定向类似。不过，旋转对象不仅影响放置的对象，而且还会影响整个坐标系统。在放置对象之前，坐标系统将在参考点 **(x,y)** 沿逆时针方向旋转 **90** 度。旋转的对象的右下角（也就是未旋转的对象的左下角）将最终落在在参考点上。实现此操作的函数调用如下所示：

```
oPDF.fit_image lImage, 80, 100, "scale 0.5 rotate 90"
```

在这种情况下，因为没有转换，所以对象将局部移出页面。

方向与旋转的比较 方向和旋转是非常类似的两个概念，但是二者之间还是有差异，用户应知道这些差异。图 5.5 和图 5.6 演示了 **orientate** 和 **rotate** 选项之间的主要差异：

图 5.3  
带方向的放置

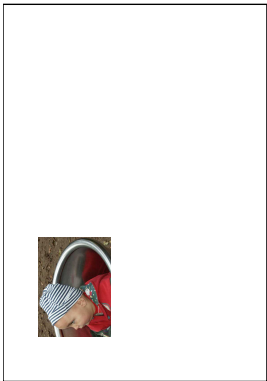
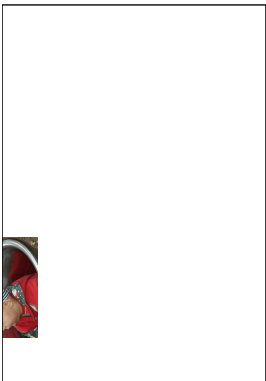


图 5.4  
带旋转的放置



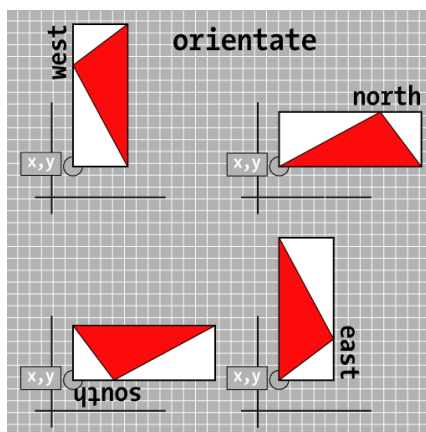


图 5.5  
orientate 选项

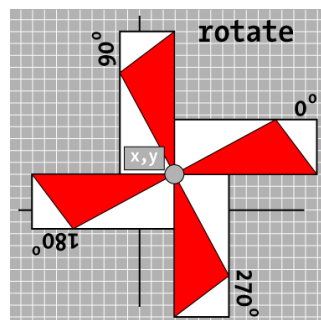


图 5.6  
rotate 选项

- **orientate** 选项在参考点  $(x,y)$  旋转对象，随后转换对象。此选项支持方向关键字为 **north**、**east**、**west** 和 **south**。
- **rotate** 选项在参考点  $(x,y)$  上旋转对象，但不作任何转换。此选项支持任意旋转角度。必须以度为单位用数字指定这些角度（整圆为 360 度）。

## 5.3.2 调整页面尺寸

在下一个示例中（请参见图 5.7），我们将页面的尺寸自动调整到对象的尺寸。例如，这对存档 PDF 格式的图像很有用。参考点  $(x,y)$  可以用于指定页面的尺寸是正好等于对象的尺寸，还是比对象的尺寸大一点或小一点。当扩大页面尺寸时（请参见图 5.7），会在图像周围保留边框；当页面尺寸小于图像尺寸时，将剪切图像的某些部分。让我们从页面尺寸与对象的尺寸完全匹配开始：

```
oPDF.fit_image lImage, 0, 0, "adjustpage"
```

下一个代码片段使页面尺寸在  $x$  和  $y$  方向上比对象尺寸大 40 个单位，并在对象周围产生边框：

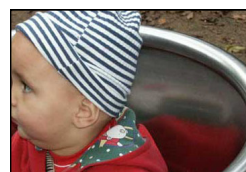
```
oPDF.fit_image lImage, 40, 40, "adjustpage"
```

下一个代码片段使页面尺寸在  $x$  和  $y$  方向上比对象尺寸小 40 个单位。将在页边框处剪裁对象，对象内的一些区域（宽度为 40 个单位）将不可见：

```
oPDF.fit_image lImage, -40, -40, "adjustpage"
```



图 5.7  
调整页面尺寸。从左向右：  
完全相同、扩大、缩小



除了按照 **x** 坐标和 **y** 坐标的方法放置之外（通常情况下，此方法从页边缘或坐标轴指定对象的距离），您还可以指定一个目标框。此目标框是一个矩形区域，放置在此区域中的对象将遵从各种格式规则。使用 *boxsize*、*fitmethod* 和 *position* 选项可控制以上操作。

使对象适合框 首先，我们在页面的右上区域放置一个公司徽标（请参见图 5.8）。徽标将在其中出现的目标矩形的尺寸已固定。不过，我们不知道如何缩放徽标以使其适合此框，而又要避免任何扭曲（禁止更改宽度和高度的比例）。以下语句可以执行这项工作：

```
oPDF.fit_image lImage, 350, 750, "boxsize {200 100} position 0 fitmethod meet"
```

此代码片段将宽为 200 个单位、高为 100 个单位 (*boxsize {200 100}*) 的框的左下角放置在点 (350, 750) 上。对象的左下角将放置在框的左下角 (*position 0*) 上。此对象将进行缩放以使其高度和 / 或宽度完全适合该框 (*fitmethod meet*)，但没有任何扭曲。

此概念提供了形形色色的变体。例如，*position* 选项可以用于指定对象中用作参考点的点（指定为宽度和高度的百分比）。*position* 选项还将指定目标框内的参考点。若宽度和高度位置的百分比一致，则指定其中一个值就可以了。例如 *position 50* 可以用于选定对象的中点和框的中点作为参考点以便放置对象。

将对象放入框中时进行剪裁 使用其他风格的 *fitmethod* 选项，我们可以剪裁对象以使其完全适合目标框（请参见图 5.9）。在这种情况下，将不会缩放对象：

```
oPDF.fit_image lImage, 50, 80, "boxsize {100 400} position 50 fitmethod clip"
```

此代码片段在坐标 (50, 80) 处放置一个宽度为 100、长度为 400 (*boxsize {100 400}*) 的框。将按照对象的原始尺寸将其放置在框中间 (*position 50*)，若对象尺寸超过框 (*fitmethod clip*)，则将剪裁此对象。

按照页面大小调整对象 通过选择页面作为用来放置对象的目标框，可以轻松将对象调整到给定页面的尺寸。以下语句使用尺寸为 595 x 842 的 A4 大小的页：

```
oPDF.fit_image lImage, 0, 0, "boxsize {595 842} position 0 fitmethod slice"
```

在此代码片段中，在页面的左下角放置一个框。该框的尺寸等于 A4 页的尺寸。将对象放置在该框的左下角，并按比例缩放此对象直到它完全覆盖框以及页面。若对象超过该框，则此对象将被剪切。请注意，*fitmethod slice* 导致对象被缩放（与此相反，*fitmethod clip* 不缩放对象）。当然，此示例中的 *position* 和 *fitmethod* 选项也可以变化。

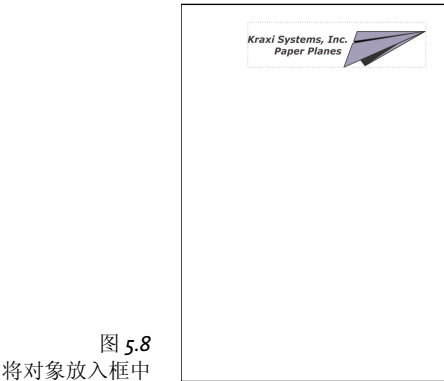


图 5.8  
将对象放入框中

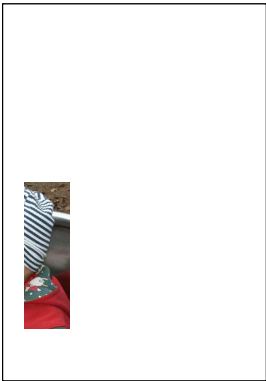


图 5.9  
将对象放入框中时  
进行剪裁

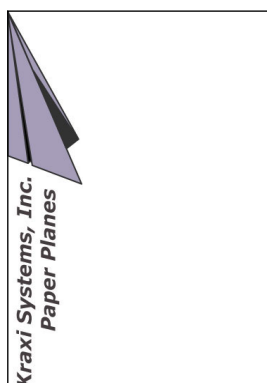


图 5.10  
使徽标适合页面

使徽标适合页面 我们如何实现图 5.10 中旋转的公司徽标？从左下角开始，将其逆时针方向旋转 90 度，并覆盖页面的整个高度：

```
oPDF.fit_image lImage, 0, 0, "boxsize {595 842} orientate west fitmethod meet"
```

参考点为  $(0, 0)$ ，方向指定为 *orientate west*。为了确保徽标覆盖整个页面高度，我们选择框的高度等于页面的高度 (842)，并为框的宽度选择一个足够大的值 (595)。徽标的比例不应更改，因此我们选择 *fitmethod meet*。



# 6 数据变量和块

PDFlib 的模板式 PDF 工作流支持数据变量的处理。通过使用块的概念，可以在导入的页中置入可变数量的单行或多行文本、图像或 PDF 图形，这些项都可以从外部源获得。以这种方式可以非常容易地完成需要生成客户化 PDF 文档的应用程序，如：

- ▶ 邮件合成
- ▶ 灵活的直投邮件
- ▶ 事务和报表处理
- ▶ 名片个性化

注：块处理需要 *PDFlib Personalization Server (PPS)*。尽管在所有商业 *PDFlib* 包中都包含 *PPS*，但是您必须购买 *PPS* 的许可证密钥；仅有 *PDFlib* 或 *PDFlib+PDI* 许可证密钥是不够的。在 *PDF* 模板中交互建立块需要 *Adobe Acrobat* 的 *PDFlib* 块增效工具。

## 6.1 安装 PDFlib 块增效工具

PDFlib 的块增效工具和它的姐妹，PDF 表单域转换增效工具均支持 Windows 和 Mac 平台上的 Acrobat 5，Acrobat 6/7 标准版及 Acrobat 6/7 专业版。但它们不支持 Acrobat 6/7 Elements 及任何版本的 Acrobat Reader/Adobe Reader。

注：增效工具含有多种语言版本的用户界面，并将自动使用与在 *Acrobat* 应用程序中使用的界面语言一样的界面语言（如果可能）。若增效工具不支持本地 *Acrobat* 语言，则将使用英语用户界面。

在 Windows 平台上为 Acrobat 5/6/7 安装 PDFlib 块增效工具 若要在 Acrobat 5、6 或 7 中安装 PDFlib 块增效工具和 PDF 表单域转换增效工具，则必须将增效工具文件拷贝到 Acrobat 增效工具文件夹的子目录中。虽然增效工具安装程序将自动完成此操作，但是也可以手动完成。增效工具文件称作 *Block.api* 和 *AcroFormConversion.api*。此增效工具文件夹的典型位置如下所示：

C:\Program Files\Adobe\Acrobat 7.0\Acrobat\plug\_ins\PDFlib Block Plugin

在 Mac 平台上，为 Acrobat 6/7 安装 PDFlib 块增效工具 使用 Acrobat 6/7，增效工具文件夹在 Finder 中将不可见。不是将增效工具文件拖到增效工具文件夹中，而是使用以下步骤（确保 Acrobat 没有运行）：

- ▶ 双击磁盘映像文件，提取增效工具文件并存入一个文件夹。
- ▶ 在 Finder 中找到 Acrobat 应用程序图标。通常，该图标位于一个具有以下名称的文件夹中：  
/Applications/Adobe Acrobat 7.0 Professional
- ▶ 单击 Acrobat 应用程序图标，并选择“文件”->“获取信息”。
- ▶ 在弹出的窗口中单击“增效工具”旁边的三角形。
- ▶ 单击 *Add...*，然后在第一步所建立的文件夹中选择 *PDFlib Block Plugin* 文件夹（如果是 Acrobat 7）或 *PDFlib Block Plugin Acro 5-6* 文件夹（如果是 Acrobat 6）。注意，在安装后，相应的文件夹不会马上显示在增效工具列表里，只有在重新打开窗口时，它才会显示在列表中。

在 Mac 平台上为 Acrobat 5 安装 PDFlib 块增效工具 若要安装 Acrobat 5 的增效工具，先双击磁盘映像文件。将 *PDFlib Block Plugin Acro 5-6* 文件夹拉入到 Acrobat 5 增效工具文件夹中。典型的增效工具文件夹名称如下所示：

故障处理 若 PDFlib 块增效工具看起来无法工作，则请确保在编辑、首选项、[ 常规 ...]、启动 (Acrobat 6/7) 或选项 (Acrobat 5) 中，框“仅使用认证的增效工具”未选中。若 Acrobat 正在认证模式下运行，则将不会加载增效工具。

## 6.2 PDFlib 块概念的概述

### 6.2.1 文档设计和程序代码的完整分离

使用 PDFlib 数据块，可以很方便地在导入页上放置可变文本、图像或图形。相对于普通的 PDF 页，内部包含数据块的页携带了稍后在服务器端将要处理的信息。PDFlib 块概念将下列的任务非常清晰的分开：

- ▶ 设计师可以创建页布局，并指定可变文本和图像元素的位置和一些相关的属性如字体大小、颜色或图像的比例等。将布局创建为 PDF 文档后，设计器使用 Acrobat 的 PDFlib 块增效工具以指定可变数据块及其相关属性。
- ▶ 程序员编写代码以连接包含在 PDFlib 块中的信息，该 PDFlib 块位于带有动态信息（例如，数据库域）的导入 PDF 页上。程序员不需要知道有关块的任何详细信息（它是否包含名称或 ZIP 代码、页上的准确位置、它的格式等等），因此该块独立于任何布局更改。PDFlib 将维护基于文件中找到的块属性的与所有块相关的详细信息。

换言之，程序员编写的代码是“数据遮蔽” – 它较为普通并不依赖这些块。例如，设计师可能决定在邮件中使用收件人的名，而不使用姓。普通块处理代码不需要改变，一旦设计器使用 Acrobat 增效工具更改块属性以使用名替换姓，则将生成正确的输出。

示例：在模板中加入文本变量 用户经常需要将动态文本加入到 PDF 模板。以下代码片断将打开输入 PDF 文件的其中一页（模板），将其放置在输出页面上，并将一些文本变量填入名为 *firstname* 的文本块中：

```
doc = oPDF.open_pdi(filename, "", 0)
if (doc = -1) Then
    MsgBox "Couldn't open PDF template!"
End
End If
page = oPDF.open_pdi_page(doc, pageno, "")
if (page = -1) Then
    MsgBox "Couldn't open page in PDF template"
End
End If

oPDF.begin_page_ext width, height, ""
oPDF.fit_pdi_page page, 0, 0, ""
oPDF.fill_textblock page, "firstname", "Serge", 0, "encoding winansi"
oPDF.close_pdi_page page
oPDF.end_page_ext ""
```



## 6.2.2 块属性

块的行为可以通过块属性来控制。使用 Acrobat 里的 PDFlib 块增效工具可以将属性指定给块。

**标准块属性** PDFlib 块在页上定义为矩形，并将对这些块指定名称、类型和稍后将在服务器端处理的一组公开属性。名称是用来标识块的任意字符串，例如 *firstname*、*lastname* 或 *zipcode*。PDFlib 支持以下几种类型的块：

- ▶ **Text**（文本）类型意味着块将保留一行或多行原文数据。将使用 **Textflow** 功能格式化多行文本。请注意，不能链接文本块以便文本将从一个块流到另一个块。
- ▶ **Image**（图像）类型意味着块将保留一个光栅图像。这相当于在 DTP 应用程序中导入一个 TIFF 或 JPEG 文件。
- ▶ **PDF** 类型意味着块将保留从其他 PDF 文档中的页上导入的任意 PDF 图形。这相当于在 DTP 应用程序中导入一个 EPS 图形。

不同的类型块会带有不同的标准属性。例如，文本块可能指定文本的字体和尺寸，图像块或 PDF 块可能指定缩放因子或旋转。对于每种块类型，PDFlib API 提供用于处理该块的专用函数。这些函数按照块的名称搜索该块的导入的 PDF 页，分析页的属性并根据相应的块属性将一些客户端提供的数据（文本、光栅图像或 PDF 页）放置在新页上。

**自定义块属性** 标准块属性可以快速执行可变数据处理应用程序，但是限于属性集，此属性集在内部为 PDFlib 所知并可以自动被处理。为了提供更多的灵活性，设计器也可以对块指定自定义属性。这些属性可以用于扩展块概念以符合大多数请求的可变数据处理程序的要求。

不存在自定义属性的规则，因为 PDFlib 除了使自定义属性对客户端可用之外，将不会以任何方式处理自定义属性。客户端代码可以检查自定义属性并用它认为是合适的任何方法使用该自定义属性。基于块的一些自定义属性，代码可以作出布局相关的或数据收集的决定。例如，科学应用程序的自定义属性可以指定用于数字输出的数字的数量，或者数据域名称可以定义为自定义块属性以便检索此块相应的数据。

**块属性重写** 在特定的情况下，程序员只需要使用块定义中提供的部分属性，而使用自定义值重写其他属性。在各种情况下，这一点很有用：

- ▶ 图像或 PDF 页的缩放因子将被计算，而不是从块定义中得到。
- ▶ 例如，当用可变数量的数据项生成发票时，通过编程方式更改块坐标。
- ▶ 应提供单独的专色名称以满足打印机应用程序中的各个用户的要求。

通过提供属性名称和所有 *fill\_\*block()* 函数的选项列表中的相应的值完成属性重写，如下所示：

```
oPDF.fill_textblock page, "firstname", "Serge", "fontsize 12"
```

这将用提供的值 12 重写块的内部 *fontsize* 属性。几乎所有常规属性的名称都可以用作选项，以及那些指定给特殊块类型的选项。例如，仅 *fill\_textblock()* 允许 *underline* 选项，而 *fill\_imageblock()* 和 *fill\_pdfblock()* 都允许 *scale* 选项，因为 *scale* 是图像和 PDF 块的有效属性。

属性重写仅可以应用到相应的函数调用；它们将不会存储在块定义中。

**坐标系统** 块坐标是以 PDF 的缺省坐标系统为依据。当带有块的页放置在输出页上时，几个与位置和比例相关的选项将提供给 *fit\_pdi\_page()*。处理块的时候，这些参数将被用到。这使得将同一模版在同一页重复出现多次成为可能。例如，一个名片模版可以在同一张纸上重复四次，这样一版可印 4 张名片。块函数会处理坐标系统的变换，并将块内容填到正确的位置。用户需做的是先将输入模版所有块填写后生成页。然后再将该页重复几次写入输出页不同的位置，并接着做一系列的块处理，依次类推。

注：块增效工具所显示的坐标值不同于保存在 *PDF* 文档内的。这是因为增效工具采用的是坐标原点位于页面左上角的 *Acrobat* 坐标系；而真正存于 *PDF* 文档内的坐标值（存于块的坐标值）是采用坐标原点位于页面左下角的 *PDF* 坐标系统。

### 6.2.3 为什么不使用 *PDF* 表单域？

有经验的 *Acrobat* 用户会问，为什么不使用已规范化的 *PDF* 表单域系统而使用新的 *PDFlib* 块概念。这两者的本质区别在于，*PDF* 表单域是为交互式填写而设计的，而 *PDFlib* 块是为自动化填写而设计的。对于既需要交互式填写又需要自动化填写的应用程序，可以使用将表单域转换为块的功能（见第 126 页上的第 6.3.4 节“将 *PDF* 表单域转换为 *PDFlib* 块”）。

虽然这两者在功能上有许多重合，*PDFlib* 块在一些方面优于 *PDF* 表单域。详见表 6.1。


表 6.1 *PDF* 表单域和 *PDFlib* 块的对比

功能	<i>PDF</i> 表单域	<i>PDFlib</i> 块
设计目标	便于交互式使用	便于自动填充
印刷上的功能 (除字体和字体大小的选择)	—	字距调整、单词和字符间的空格调整、上下划线及删除线
字体控制	字体嵌入	字体嵌入，子集和编码
文本格式的控制	支持左、中、右对齐方式	向左对齐、居中对齐、向右对齐、调整；多种格式算法和控制；可内嵌选项控制文本外观
更改文本内的字体或其他文本属性	—	是
合并的结果是 <i>PDF</i> 页面说明的整体部分	—	是
用户可以编辑合并的字段内容	是	否
可扩展的属性集	—	是（自定义块属性）
使用图像文件以填充	—	<i>BMP</i> 、 <i>CCITT</i> 、 <i>GIF</i> 、 <i>PNG</i> 、 <i>JPEG</i> 、 <i>JPEG2000</i> 、 <i>TIFF</i>
颜色支持	<i>RGB</i>	灰度、 <i>RGB</i> 、 <i>CMYK</i> 、专色、 <i>Lab</i>
<i>PDF/X</i> 兼容	—	是（带块的模板和合并的结果）
填充时，可以重写图形和文本属性	—	是

## 6.3 建立 *PDFlib* 块

### 6.3.1 利用 *PDFlib* 块增效工具交互式建立块

**激活 *PDFlib* 块工具** 用于建立 *PDFlib* 块的 *PDFlib* 块增效工具类似于 *Acrobat* 中的表单工具。当块工具被激活，所以在页上的块将被显示出来。当其他 *Acrobat* 工具被选择后，块依旧存在但会被隐藏。用户可以用以下几种方式激活块工具：

- ▶ 在 *Acrobat* 的“高级编辑”工具条中点击块图标 （在 *Acrobat* 5 中：“编辑”工具栏）；
- ▶ 通过菜单项“*PDFlib* 块”->“*PDFlib* 块工具”；
- ▶ 使用快捷键 *P*；但在使用之前要确信在“编辑”->“首选项”->“[常规...]”->“常规”窗口中选择了“使用加速键访问工具”，默认值是不选（*Acrobat* 5 不需此步）

**建立和修改块** 当您激活了块工具后，就可以移动十字光标以在页面上要求的位置建立所需尺寸的块。所有块都是矩形的，它们的四边总是与页边缘平行。当用户建立了新块后，将出现块属性对话框，您可以在对话框中编辑各种块属性（见第 125 页上的第 6.3.2 节“编辑块属性”）。块工具会自动建立一个块名称，该块名称可以在属性对话框中更改。页块名称在同一页中必须是唯一的。您可以将顶部区域中的块类型更改为某一文本、图像或 *PDF*。常

规和自定义标签将总是可用，而根据选择的块类型不同，一次只能激活一个文本、图像和 PDF 标签。*Textflow* 标签仅在 *textflow* 属性为 *true* 的情况下，用作类型文本块。另一个标记为 *Tabs*（跳格键位置）的标签仅在 *Textflow* 标签中的 *hortabmethod* 属性设置为 *ruler* 的情况下可用。

注：当用户增加块或对 PDF 中的现有块作出更改后，使用 *Acrobat* 的“另存为 ...”指令（相对于“保存”指令）可取得较小的文件大小。

注：当使用 *Acrobat* 的 *Enfocus PitStop* 增效工具去编辑带有 *PDFlib* 块的文档时，您可能会看见此消息：此文档包含 *PDFlib* 的 *PieceInfo*。单击“确定”以继续编辑或单击“取消”以中止编辑。可以忽略此消息；在这种情况下，安全操作是单击“确定”。

选择块 块拷贝，块移动等几个块操作，都需在选择块后方可进行。用户可以通过块工具去选择一个或多个块，如下所示：

- ▶ 若选择一个块，只需用鼠标单击相应块。
- ▶ 在按下 **Shift** 键的同时单击其他块以扩大选择。
- ▶ 若选择当前页的所与块，按 **Ctrl-A**（在 **Windows** 平台上）或 **Cmd-A**（在 **Mac** 平台上）或编辑，全部选定。

上下文菜单 若已选定一个或多个块，则您可以打开上下文菜单以快速访问块相关的函数（在 *PDFlib* 块菜单中同样可用）如要打开上下文菜单，在 **Windows** 上，在选定的块上单击鼠标右键，或者在 **Mac** 上，按住 **Ctrl** 键并单击块。

例如，想删除一个块，用块工具选择此块，然后按 **Delete** 键，或在上下文菜单中选择“编辑”->“删除”。

微调块大小和位置 使用块工具，您可以将一个或多个选定的块移动到不同的位置。在移动块时按住 **Shift** 键，可以保持块将沿着水平和垂直方向的移动。这对于块准确对齐很有帮助。当指针位于块的转角处时将变为箭头，并且您可以重新调整块的大小。要调整多个块的位置和大小，可以在选择两个或两个以上的块后，使用“*PDFlib* 块”菜单或上下文菜单中的“对齐”、“居中”“分布”或“大小”命令进行调整。单个或多个块的位置也可以用箭头键进行调整。

另外，您可以在属性对话框中输入用数字表示的块坐标。坐标系统的原点位于页的左上角。当前在 *Acrobat* 中选定的坐标将按照单位显示。更改显示单位的过程，如下所示：

- ▶ 在 *Acrobat 6/7* 中，转到“编辑”->“首选项”->“[常规...]”->“单位和网线”[或“页面单位”]，并选择点、英寸、毫米、派卡、厘米中的某一项。您还可以跳至“视图”->“导航标签”->“信息”，并从“选项”菜单中选择一个单位。
- ▶ 在 *Acrobat 5* 中，跳至“编辑”->“首选项”->“常规”->“显示”->“页面单位”，并选择点、英寸、毫米中的某一项。您还可以跳至“窗口”->“信息”，并从“信息”菜单选择一个单位。

请注意，选定的单位只会影响 *Rect* 属性，而不会影响任何其他数字属性。

通过选择图像或图形建立块 作为手动拖动块矩形的一个替换，您可以使用现有页内容定义块大小。首先，确保已启用菜单项“*PDFlib* 块”->“单击对象以定义块”。现在您可以使用块工具单击页上的图像以建立图像大小的块。您也可以单击其他图形对象，块工具将试图选择周围的图形（例如徽标）。单击对象功能是作为定义块的一个辅助功能。若您想重写调整块的位置和大小，以后您可以不受任何限制的这样做。块将不会对作为定位和调整大小的辅助工具使用的图像或图形对象锁定。

单击对象功能将试图识别形成页上的逻辑元素的矢量图形和图像。当单击一些页面内容时，将选定此页面内容的边框（周围的矩形），除非此对象是空白或太大。在下一个步骤中，

图 6.1  
编辑块属性：“文本流”窗格  
仅在 *textflow=true* 时可见；  
“选项卡”窗格仅在  
*hortabmethod=ruler* 时可见



其他部分包含在检测的矩形框中的对象将被添加导选定的区域，等等。最终区域将用作生成的块矩形的基础。最终的结果是单击对象功能将试图选择完整图形，而不是选择单个行。

单击对象功能并不完善：根据页面内容的性质的不同，它不会总是选择用户所需内容。请记住该功能仅可作为用于快速建立块矩形的定位辅助工具。

自动检测字体属性 PDFlib 块增效工具可以分析在块所在的位置出现的以下字体，并能够自动填写块的相应属性：

fontname, fontsize, fillcolor, charspacing, horizscaling, wordspacing, textrendering, textrise

因为在背景应被忽略时，对字体属性进行自动检测会引起不希望发生的行为，使用 “PDFlib 块” -> “检测基本字体和颜色” 可以启用或停用此自动检测。默认情况下，此功能被关闭。

锁定块 块可以被锁定，从而可以保护它们不会被意外的移动、改变大小或删除。当块工具活动时，选择块并从块的上下文菜单中选择锁定。当块被锁定后，此块不能被移动、改变大小或删除；也不可以显示块属性对话框。

在块中使用 PDF/X 与 PDF 表单域不同，PDFlib 块支持 PDF/X 印前标准。带有块的输入文档和生成的输出 PDF 都可以符合 PDF/X 印前标准。然而，在准备符合 PDF/X 印前标准 *textflow* 的块文件时，用户可能会遇到以下几个问题：

- ▶ PDF/X-1:2001、PDF/X-1a:2001 和 PDF/X-3:2002 都基于 Acrobat 4/PDF 1.3，并且不支持 Acrobat 5 文件；
- ▶ PDFlib 块增效工具要求 Acrobat 5 或更高版本。

为解决这一问题，用户可以将生成的带块信息的 PDF 文件通过 Acrobat 转换成所需的 PDF 版本。详情见第 142 页上的 “更改文档的 PDF 版本”。

### 6.3.2 编辑块属性

当您建立新块、双击现有块，或从块的上下文菜单选择“属性”时，属性对话框将弹出，您可以在其中编辑与选定的块相关的所有设置（请参见图 6.1）。详情见第 128 页上的第 6.4 节“用于自动化处理的标准属性”，块属性大致分以下几类：

- ▶ 块名称、块类型、说明和常规标签里的所有属性将适用于所有块类型。
- ▶ *Text*、*Image* 和 *PDF* 标签中的属性只分别适用于对应的块类型。仅与块的类型相应的标签将是活动状态，而其他标签是非活动状态。
- ▶ 若 *Text*（文本）类型块的 *textflow* 属性被设置为 *true*，另一个名为 *Textflow* 的带有 *textflow* 相关的设置的标签将出现。
- ▶ 若 *Text*（文本）类型块的 *textflow* 属性被设置为 *true*，同时 *Textflow* 标签中的 *hortabmethod* 属性设置为 *ruler*，仍将出现另一个名为 *Tabs* 的窗格，您可以在此窗格中编辑跳格键设置。
- ▶ “自定义”选项卡中的属性可以由用户定义，并应用于任何块类型。

若要更改一个属性的值，请在属性的输入区域（例如 *linewidth*）中输入需要的数字或字符串，从下拉列表选择一个值（例如 *fontname*、*fitmethod*）或通过单击对话框右侧的“...”按钮选择一个颜色值或文件名称（例如 *backgroundcolor*）。对于 *fontname* 属性，您可以从系统中已安装的字体的列表中选择，或键入自定义字体名称。不管用何方法输入字体名称，字体必须在系统上可用，在该系统上将使用 *PDFlib Personalization Server* 填充块。

完成对属性的编辑后，单击“确定”以关闭属性对话框。刚刚定义的属性将存储在 PDF 文件中作为块定义的一部分。

**交叠块** 交叠块很难进行选择，因为用鼠标点击一个区域将总是选定最上层的块。在这种情况下，可以使用上下文菜单中的“选择块”项以按照块名称选择块。当块被选择后，随后所有对于该区域的操作（如双击）都针对被选择块，其他在此区域的块不会受影响。这样，即使被局部或完全覆盖的块属性也可以很方便地编辑。

**使用和恢复属性默认值** 为了保存一些数量的键入和单击，块工具会记住在上一个块的属性对话框中输入的属性值。当您建立一个新的块时，将再次使用这些值。当然，您可以随时通过不同的值重写这些值。

按下属性对话框中的“全部重置”按钮将大多数块属性重置为其相应的默认值。不过，以下项将保持不变：

- ▶ *Name*、*Type*、*Rect* 和 *Description* 属性
- ▶ 所有自定义属性。

**共享块属性** 通过按住 *Shift* 键并使用块工具选择多个块的方式，您可以选择在页上选择任意数目的块。双击任意一个被选块或按 *Enter* 键，一个共享块属性对话框将弹出，其中的块属性值将适用于所有被选块。不过，由于不是所有属性都可以被多个块共享，只有一个属性子集可以被编辑。若想具体了解哪些属性可被多块共享，见第 128 页上的第 6.4 节“用于自动化处理的标准属性”。自定义属性不能被共享。

### 6.3.3 页和文件之间的块拷贝

块增效工具提供了几种在当前页，当前文档或不同文件间移动和拷贝块的方法：

- ▶ 拖动鼠标进行块移动或拷贝，或粘贴到别的页或打开的文件
- ▶ 在同一文件中，复制相同的块到一或多页里
- ▶ 导出块到新文件（空白页面）或已存在文件（对现有页使用块）
- ▶ 从其他文档导入块

为了在保留块定义的同时更新页面内容，您可以在保留块的同时替换以下页。使用“文档”->“替换页面”（Acrobat 5 和 7）或“文档”->“页面”->“替换”（Acrobat 6）。

**移动和复制块** 在按下 **Ctrl** 键（在 **Windows** 上）或 **Alt** 键（在 **Mac** 上）的同时，选择一个或多个块并将其拖动到新的位置，可以重新定位块或创建块的副本。当按下键的时候，鼠标指针将改变。复制的块将与原始的块有相同的属性，除了其名称和位置将自动更改以外。

您还可以使用复制 / 粘贴将块复制到同一页的其他位置、同一文档中的其他页或当前在 **Acrobat** 中打开的其他文档：

- ▶ 启用块工具并选择您想要复制的块。
- ▶ 使用 **Ctrl-C**（在 **Windows** 平台上）或 **Cmd-C**（在 **Mac** 平台上）或“编辑”->“复制”，将选定的块复制到剪贴板。
- ▶ 使用 **Ctrl-V**（在 **Windows** 平台上）或 **Cmd-V**（在 **Mac** 平台上）或“编辑”->“粘贴”，粘贴当前位于剪贴板中的块。

**复制块到其他页** 您可以同时创建当前文档中的任意数目页上的一个或多个块的副本：

- ▶ 启用块工具，并选择您想要复制的块。
- ▶ 从“**PDFlib** 块”菜单或上下文菜单选择“导入和导出”->“复制”。
- ▶ 选择要复制的块（选定的块或页上的所有块）以及您想要用来复制块的目标页的范围。

**导出和导入块** 通过使用块的导出 / 导入功能，可以共享单个页上的块定义或多个 **PDF** 文件之间的某一文档中的所有块。在保留现有块定义的同时更新页面内容，这一点很有用。将块定义导出到一个分离的文件的过程，如下所示：

- ▶ 启用块工具，并选择您想要导出的块。
- ▶ 从“**PDFlib** 块”菜单或上下文菜单中选择“导入和导出”->“导出”。为包含块定义的文件输入页范围和文件名称。

您可以通过“**PDFlib** 块”->“导入和导出”->“导入”导入块定义。导入块时，您可以选择是对文档中的所有的页还是仅对页范围使用导入的块。若已选定多个页，则块定义将不变地被复制到这些页。若目标范围内的页多于导入的块定义文件中的页，则您可以使用“重复模板”复选框。若此复选框被激活，则将在当前文档中重复导入的文件中的块的顺序，直至文档的结尾。

在导出时，将块复制到其他文档 当导出块时，您可以立即对其他文档中的页使用这些块，从而使这些块从一个文档传播到另一个文档。为此，选择一个现有文档，以将块导出到此文档。若您启用复选框“删除已存块”，则在将新的块复制到文档中之前，将删除目标文档中可能存在的所有块。

### 6.3.4 将 PDF 表单域转换为 PDFlib 块

作为手动创建 **PDFlib** 块的一个替换，您可以将 **PDF** 表单域自动转换为块。若您需要使用 **PDFlib Personalization Server** 自动填写复杂的 **PDF** 表单，或需要大量的现有 **PDF** 表单以便自动填充，这一点特别方便。为了将页上的所有表单域转换为 **PDFlib** 块，请选择“**PDFlib** 块”->“转换表单域”->“当前页”。若要转换文档中的所有表单域，则请选择“所有页”。最终，您只能使用“所选表单域”转换所选表单域（选择 **Acrobat** 的表单工具或选择对象工具来选定表单域）。

表单域转换详细信息 自动表单域转换将“PDFlib 块”->“转换表单域”->“转换选项”对话框中选定的类型的表单域转换为 **Text** 类型块。默认情况下，将转换所有表单域类型。根据表 6.2，已转换域的属性将转换为相应的块属性。

表 6.2 PDF 表单域到 PDFlib 块的转换

PDF 表单域属性 ...	... 将转换成对应的 PDFlib 属性
所有域	
位置	一般, <i>Rect</i>
名称	一般, <i>Name</i>
工具提示	一般, <i>Description</i>
外观, 文本, 字体	文本, <i>fontname</i>
外观, 文本, 字体大小	文本, <i>fontsize</i> ; “ <i>auto</i> ”值将会换算为相当于 2/3 块高度的固定字体大小, 并且 <i>fitmethod</i> 将被设为 “ <i>auto</i> ”。对于多行文本的表单域 / 块, 这种组合将会自动生成适合的字体大小, 其值可能比 2/3 块高度的初始值要小。
外观, 文本, 文本颜色	文本, <i>strokecolor</i> ; 文本, <i>fillcolor</i>
外观, 边框, 边框颜色	一般, <i>bordercolor</i>
外观, 边框, 填充颜色	一般, <i>backgroundcolor</i>
外观, 边框, 线宽	一般, <i>linewidth</i> : 细 =1, 中 =2, 厚 =3
常规, 一般属性, 表单域	一般, <i>Status</i> : 显示 = <i>active</i> , 隐藏 = <i>ignore</i> , 显示但不打印 = <i>ignore</i> , 隐藏但可打印 = <i>active</i>
常规, 一般属性, 方向	一般, <i>orientate</i> : 0= <i>north</i> , 90= <i>west</i> , 180= <i>south</i> , 270= <i>east</i>
文本域	
选项, 默认值	文本, <i>defaulttext</i>
选项, 对齐	一般, 位置: 左 = <i>{left center}</i> , 中 = <i>{center center}</i> , 右 = <i>{right, center}</i>
选项, 多行	文本, <i>textflow</i> : 选中 = <i>true</i> , 未选中 = <i>false</i>
单选按钮和复选框	
若选择了“默认选中复选框 / 按钮”: 选项, 复选框样式或选项, 按钮样式	文本, <i>defaulttext</i> : 方格 =4, 圆圈 =1, 十字 =8, 菱形 =u, 正方形 =n, 星形 =H (这些字符分别对应于 <i>ZapfDingbats</i> 字体的符号)
列表框和组合框	
选项, 所选 (默认) 项目	文本, <i>defaulttext</i>
按钮	
选项, 图标和标签, 标签	文本, <i>defaulttext</i>

具有相同名称的多个表单域 允许同一页面上的多个表单域具有相同的名称，而页上的块名称必须是唯一的。当见表单域转换为块时，将为生成的块的名称添加一个用数字表示的后缀以创建唯一的块名称 ( 参见第 127 页上的 “表单域与其相应块的关联” )。

请注意，由于 Acrobat 中的某个问题，导致无法准确报告具有相同名称的表单域的域属性。若多个域的名称相同而属性不同，则将不会在生成的块中反映这些差异。在这种情况下，转换过程将发出警告并提供受影响的表单域的名称。在这种情况下，您应仔细检查生成的块中的属性。

表单域与其相应块的关联 当转换多个具有相同名的表单域（如：单选按钮）为块时，表单域名将被修改。这给准确识别块与其特定的表单域的联系带来难度。尤其是对于使用 PDF 或 XPDF 文件作为填写块的源文件从而取得类似于表单域填写效果的情况。

为了解决这个问题，AcroFormConversion 增效工具在创建相应块时将原表单域作为自定义属性具体纪录下来。

表 6.3 识别原表域对应块的自定义属性

自定义属性	意义
PDFlib:field:name	完全合乎规范的表域名。
PDFlib:field:pagenumber	表域在原文档的页码（字符串形式）。
PDFlib:field:type	相应表域的类型，可为：pushbutton, checkbox, radiobutton, listbox, combobox, textfield 或 signature.
PDFlib:field:value	(仅对 type=checkbox) 表域的出口值。

将块绑定到相应的表单域 为了使 PDF 表单域和生成的 PDFlib 块保持同步，可以将生成的块绑定到相应的表单域。这意味着块工具将内部保持表单域和块之间的关系。当再次启用转换过程时，将更新绑定块以反映相应的 PDF 表单域的属性。绑定块对于避免重复工作很有用：当更新表单以便交互使用时，也会自动更新相应的块。

块生成以后，若您不想保留转换的表单域，则可以在“PDFlib 块”->“转换表单域”->“转换选项”对话框中选择“删除转换的表单域”。在转换过程完成后，此选项将永久性移除表单域。任何与受影响的域相关的操作（例如 JavaScript）也将从文档移除。

批量转换 若您有许多带表单域的 PDF 文档需要转换为 PDFlib 块，则您可以使用批转换功能自动处理任意数量的文档。可以通过“PDFlib 块”->“转换表单域”->“批转换”使用批处理对话框。

- ▶ 可以单个选定输入文件；另外，可以处理文件夹的全部内容。
- ▶ 输出文件可以编写到输入文件所在的同一文件夹，或编写到不同的文件夹。输出文件可以接受一个前缀以便将其与输入文件区分开来。
- ▶ 当对大量文档进行处理时，建议指定一个日志文件。在转换完成后，此日志文件将包含已处理文件的完全列表以及有关每个转换的结果的详细信息（该信息中包含可能存在的错误消息）。

在转换过程中，尽管转换的 PDF 文档将在 Acrobat 中可见，但是您无法使用任何 Acrobat 的菜单函数或工具。

## 6.4 用于自动化处理的标准属性

PDFlib 支持可以指定给各种块类型的一般属性。另外，还存在一些可以与块类型 Text、Image 和 PDF 相关的属性。一些属性是共享的，这意味着使用块增效工具可以立即将这些属性指定给多个块。

除了句柄和操作列表之外，属性支持与选项列表相同的数据类型（请参见第 157 页上的第 8.1.2 节“选项列表”）。

许多的块属性具有与 fit\_image() 等功能选项（如 fitmethod）或 PDFlib 参数（如 charspacing）相同的名称。在这种情况下，这些块属性的行为与文档描述的相同选项或参数的行为完全一致。

PDFlib 的属性处理 PDFlib 块功能 fill\_\*block() 将根据下列顺序处理块属性：

- ▶ 若 backgroundcolor 属性存在且具有非 None 的色彩空间关键字，则使用指定的颜色填充块矩形。
- ▶ 处理除了 bordercolor 和 linewidth 外的其他属性。



- ▶ 若 *bordercolor* 属性存在且具有非 *None* 的色彩空间关键字，则将使用指定的颜色和行宽绘制块边框。

若文本和缺省文本均未提供，则不会有包括背景颜色和块边框在内的任何输出。PDFlib 不提供剪切；若您想确保块内容未超过块矩形区域，应避免使用 *fitmethod nofit*。

若在块属性中使用分色，则指定的专色名称必须是 PDFlib 理解的（见第 51 页上的第 3.3.3 节“专色”），或者在较早已调用 *makespotcolor()* 专门指定的；否则，块功能将失败。

### 6.4.1 一般属性

一般属性适用于所有块类型 (*Text, Image, PDF*)。对于块管理、自身描述块矩形的外观和管理在块内部放置内容的方式，一般属性都是必需的。PDFlib 块增效工具将自动生成必需的属性。表 6.4 列出了一般属性。

表 6.4 一般块属性

关键字	类型	可能的值和说明
块管理		
<i>Name</i>	<i>string</i>	（必需的）块名。在一页里，块名必须是唯一的，但同一块名可出现在同一文档的不同页中。块名中不允许出现 [ ] / 这三个字符。块名最不能超过 125 个字符。
<i>Description</i>	<i>string</i>	块的函数的可供人读的说明，采用 <i>PDFDocEncoding</i> 或 <i>Unicode</i> 编码（在后者情况下，以 <i>BOM</i> 开头）。此属性仅用于用户信息，在进行块处理时该属性将被忽略。
<i>Locked</i>	<i>boolean</i>	（可共享的）若值为 <i>true</i> ，块增效工具将不能编辑该块及其属性。当进行块处理时该属性将被忽略。默认值： <i>false</i> 。
<i>Rect</i>	<i>rectangle</i>	（必需的）块坐标。坐标系统的原点位于页的左下角。不过，块增效工具将按照 <i>Acrobat</i> 的表示法显示坐标，也就是说，原点在页的左上角。坐标将用在 <i>Acrobat</i> 中当前选定的单位进行显示，但在 <i>PDF</i> 文档里将以点为单位进行存储。
<i>Status</i>	<i>keyword</i>	关键字描述块将如何被处理（默认值： <i>active</i> ）： <i>active</i> 该块将按照其属性设置被完全处理。 <i>ignore</i> 该块将被忽略。 <i>static</i> 不将放置任何可变内容；而将使用块的默认文本、图像或 <i>PDF</i> 内容（如果可用）。
<i>Subtype</i>	<i>keyword</i>	（必需的）块类型，其值可为 <i>Text</i> 、 <i>Image</i> 或 <i>PDF</i>
<i>Type</i>	<i>keyword</i>	（必需的）始终为 <i>Block</i>
块外观		
<i>backgroundcolor</i>	<i>color</i>	（可共享的）若此属性存在并具有非 <i>None</i> 的色彩空间关键字，则将绘制一个矩形并用提供的颜色填充该矩形。这对覆盖当前页已有内容很有效。默认值： <i>None</i>
<i>bordercolor</i>	<i>color</i>	（可共享的）若此属性存在并具有非 <i>None</i> 的色彩空间关键字，则将绘制一个矩形并用提供的颜色为该矩形描边。默认值： <i>None</i>
<i>linewidth</i>	<i>float</i>	（可共享的；必须大于 0）用于绘制块边框的线宽度；只有在 <i>bordercolor</i> 被设置时才有效。默认值： 1

表 6.4 一般块属性

关键字	类型	可能的值和说明
内容排版		
<i>fitmethod</i>	<i>keyword</i>	(可共享的) 当提供的內容不能完全排入块矩形框时的策略。其可选值为 <i>auto</i> 、 <i>nofit</i> 、 <i>clip</i> 、 <i>meet</i> <sup>1</sup> 、 <i>slice</i> <sup>1</sup> 和 <i>entire</i> <sup>1</sup> 。对于简单的文本块、图像和 PDF 块，将根据表 8.19 和表 8.41 说明此属性。默认值: <i>auto</i> 。对于 <i>textflow</i> 块（此处的块对于文本太小），说明如下所示： <i>auto</i> <i>fontsize</i> 和 <i>leading</i> 将被减小直至文本可完全排入矩形框 <i>nofit</i> 文本将超出块矩形框底线。 <i>clip</i> 文本将在块矩形框处剪切。
<i>orientate</i>	<i>keyword</i>	(可共享的) 当內容被放置时，指定內容的必需方向（请参见表 8.41）。其可选值为 <i>north</i> 、 <i>east</i> 、 <i>south</i> 和 <i>west</i> 。默认值: <i>north</i>
<i>position</i> <sup>1</sup>	<i>float</i> 或 <i>keyword list</i>	(可共享的) 用于指定內容内的参考点的位置的一个或两个值（对于文本请参见表 8.19，对于图像 /PDF 请参见表 8.41）。 <i>position</i> 被指定为块内的百分比。默认值: {0, 0}，即左下脚
<i>rotate</i>	<i>float</i>	(可共享的) 在处理开始之前，块将反时针方向旋转的旋转角（以度为单位）。参考点为旋转的中心。默认值: 0

1. *textflow* 块 (*text blocks with textflow=true*) 不支持此关键字或属性。

6.4.2 Text 属性

文本相关的属性适用于 *Text* 类型块（除了一般属性之外）。所有文本相关的属性都可以被共享。

所有文本块的属性 文本块取决于 *textflow* 属性，可以包含单行或多行文本。表 6.5 列出了适用于这两种类型的文本相关的属性。

表 6.5 *Text* 块属性

关键字	类型	可能的值和说明
<i>charspacing</i>	<i>float</i> 或 <i>percentage</i>	字符间距（请参见表 8.18）。其值为字体大小的百分比。默认值: 0
<i>defaulttext</i>	<i>string</i>	若客户端未提供替代文本，则将使用的文本。 <sup>1</sup>
<i>fillcolor</i>	<i>color</i>	填充文本的颜色。默认值: <i>gray 0 (=black)</i>
<i>fontname</i> <sup>2</sup>	<i>string</i>	<i>load_font()</i> 要求的字体的名称。PDFlib 块增效工具将显示一个系统已安装的字体的列表。不过，这些字体名称可能在 <i>Mac</i> 、 <i>Windows</i> 和 <i>Unix</i> 系统之间不可移植。  当对块进行填充时，除非已经提供字体选项，否则必须将文本的编码指定为 <i>fill_textblock()</i> 的选项。
<i>fontsize</i> <sup>2</sup>	<i>float</i>	字体的大小（以点为单位）
<i>fontstyle</i>	<i>keyword</i>	字体样式必须是 <i>normal</i> 、 <i>bold</i> 、 <i>italic</i> 或 <i>bolditalic</i> 中的一款（请参见表 8.16）
<i>horizscaling</i>	<i>float</i> 或 <i>percentage</i>	水平文本缩放（请参见表 8.18）。默认值: 100%
<i>italicangle</i>	<i>float</i>	文本的倾斜角度，（请参见表 8.18）。默认值: 0
<i>kerning</i>	<i>boolean</i>	字距调整行为（请参见表 8.18）。默认值: <i>false</i>
<i>margin</i>	<i>float list</i>	用来描述文本框的其他水平和垂直扩展的一个或两个浮点值（请参见表 8.19）。默认值: 0
<i>monospace</i>	<i>integer 1...2048</i>	强制字体中的所有字符使用相同宽度（请参见表 8.16）默认值: <i>absent</i> （使用字体的规格）
<i>overline</i>	<i>boolean</i>	上划线模式（请参见表 8.18）。默认值: <i>false</i>
<i>strikeout</i>	<i>boolean</i>	删除线模式（请参见表 8.18）。默认值: <i>false</i>

表 6.5 Text 块属性

关键字	类型	可能的值和说明
<i>strokecolor</i>	<i>color</i>	文本描边色。默认值: <i>gray 0 (=black)</i>
<i>textflow</i>	<i>boolean</i>	控制单行或多行文本处理 (默认值: <i>false</i> ): <i>false</i> 文本可以跨单行并将使用 <i>fit_text()</i> 处理此文本。 <i>true</i> 文本将跨多行并将使用 <i>fit_textflow()</i> 处理此文本。将忽略常规属性位置。除了标准文本属性之外, 还可以指定所有 <i>textflow</i> 相关的属性 (请参见表 6.6)。
<i>textrendering</i>	<i>integer</i>	文本渲染模式 (请参见表 8.18)。默认值: <i>0</i>
<i>textrise</i>	<i>float</i> 或 <i>percentage</i>	文本提升参数 (请参见表 8.18)。其值为字体大小的百分比。默认值: <i>0</i>
<i>underline</i>	<i>boolean</i>	下划线模式 (请参见表 8.18)。默认值: <i>false</i>
<i>underlineposition</i>	<i>float</i> 、 <i>percentage</i> 或 <i>keyword</i>	用于相对于基线的加下划线的文本的描边线的位置 (请参见表 8.20)。其值为字体大小的百分比。默认值: <i>auto</i>
<i>underlinewidth</i>	<i>float</i> 、 <i>percentage</i> 或 <i>keyword</i>	加下划线的文本的行宽 (请参见表 8.20)。其值为字体大小的百分比。默认值: <i>auto</i>
<i>wordspacing</i>	<i>float</i> 或 <i>percentage</i>	单词间距 (请参见表 8.18)。其值为字体大小的百分比。默认值: <i>0</i>

- 1. 将用 *winansi* 编码或 *Unicode* 说明此文本。
- 2. 文本块中需要此属性; *PDFlib* 块增效工具将自动强制使用此属性。

**textflow** 块属性 *textflow* 相关的属性适用于 *Text* 类型的块, 此处 *textflow* 属性为 *true*。文本相关的属性将用于构建初始选项列表以便处理 *textflow* (与 *create\_textflow()* 的 *optlist* 参数对应)。尽管使用增效工具无法指定选项列表, 但是当使用 *fill\_textblock()* 填充块时, 可以在服务器上提供这些选项列表作为文本内容的一部分。所有 *textflow* 相关的属性都可以共享。表 6.6 列出 *textflow* 相关的属性。

表 6.6 textflow 块属性

关键字	类型	可能的值和说明
文本语义属性:		
<i>tabalignchar</i>	<i>integer</i>	小数部分以之对齐的字符 <i>Unicode</i> 值。默认值: <i>'</i> 字符 ( <i>U+002E</i> )
控制文本排版的属性:		
<i>alignment</i>	<i>keyword</i>	指定文本行在段落中的对齐方式 (默认值: <i>left</i> ): <i>left</i> 起于 <i>leftindent</i> , 向左对齐 <i>center</i> 以 <i>leftindent</i> 和 <i>rightindent</i> 之间的中心对齐 <i>right</i> 向右对齐, 止于 <i>rightindent</i> <i>justify</i> 左右两边对齐
<i>firstlinedist</i>	<i>float</i> 、 <i>percentage</i> 、或 <i>keyword</i>	定界框顶部与第一行文本的基线之间的距离, 以用户坐标或相关字体大小的百分比为单位 (用该行的第一个字体大小为单位, 如果 <i>fixedleading=true</i> 并且该字体大小是该行中最大的), 或者用关键字 (默认值: <i>leading</i> ): <i>leading</i> 第一行的行距值; 常用可触及定界框顶部的典型字符如 <i>À</i> 为基准。 <i>ascender</i> 第一行的字体上伸部分; 常用可触及定界框顶部的带较大的上伸部分的典型字符如 <i>d</i> 和 <i>h</i> 为基准。 <i>capheight</i> 为首行确定的大写高度值; 常用可触及定界框顶部的典型的大写字母字符如 <i>H</i> 为基准。 <i>xheight</i> 第一行的 <i>xheight</i> 值; 常用可触及定界框顶部的典型的小写字母字符如 <i>x</i> 为基准。  若 <i>fixedleading=false</i> , 第一行里 <i>leading</i> 、 <i>ascender</i> 、 <i>xheight</i> 或 <i>capheight</i> 中的最大值将被使用。

表 6.6 *textflow* 块属性（续）

关键字	类型	可能的值和说明
<i>fixedleading</i>	<i>boolean</i>	如果值为 <i>true</i> ，每行的第一个行距将作为该行的 <i>leading</i> 值。否则，每行的最大行距将作为该行的 <i>leading</i> 值。默认值: <i>false</i>
<i>hortabsize</i>	<i>float</i> 或 <i>percentage</i>	水平制表键的宽度 <sup>1</sup> 。具体解释将取决于 <i>hortabmethod</i> 选项。默认值: <i>7.5%</i>
<i>hortabmethod</i>	<i>keyword</i>	文本中的水平制表键的处理。若计算的位置位于当前文本位置的左侧，则将忽略制表键（默认值: <i>relative</i> ）: <i>relative</i> 位置将按 <i>hortabsize</i> 中指定的数量前进。 <i>typewriter</i> 位置将移到下一个 <i>hortabsize</i> 的倍数的位置。 <i>ruler</i> 位置将前进到 <i>ruler</i> 选项中的第 <i>n</i> 个制表键值，此处 <i>n</i> 为目前在前一行中找到的选项卡数量。若 <i>n</i> 大于制表键的数量，则将使用相关方法。
<i>lastalignment</i>	<i>keyword</i>	排版段落中的最后一行。支持所有 <i>alignment</i> 选项的关键字，外加以下（默认值: <i>auto</i> ）: <i>auto</i> 若 <i>alignment</i> 选项值不是 <i>justify</i> ，使用该方式排版。在后者情况下，将使用 <i>left</i> 方式。
<i>lastlinedist</i>	<i>float</i> 、 <i>percentage</i> 或 <i>keyword</i>	（若 <i>fitmethod=nofit</i> ，将忽略）最后的文本行基线与定界框底部之间的最小距离，以用户坐标或相关字体大小的百分比为单位（用该行的第一个字体大小为单位，如果 <i>fixedleading=true</i> 并且该字体大小是该行中最大的），或者使用关键字（默认值: <i>o</i> ，也就是说，定界框的底部用作基线，且典型的下伸部分将超出定界框）: <i>descender</i> 最后一行的字体下伸值；常用可触及定界框底部的带下伸部分的典型字符如 <i>g</i> 和 <i>j</i> 为基准。  如果 <i>fixedleading=false</i> ，最后一行 <i>descender</i> 中的最大值将被使用。
<i>leading</i>	<i>float</i> 或 <i>percentage</i>	两个相连文本行基线之间的距离，以用户坐标或相关字体大小的百分比为单位。默认值: <i>100%</i>
<i>parindent</i>	<i>float</i> 或 <i>percentage</i>	段落第一行的左缩排 <sup>1</sup> 。该数值将追加到 <i>leftindent</i> 上。在一行中指定这一选项，其行为就如制表键。默认值: <i>o</i>
<i>rightindent</i> <i>leftindent</i>	<i>float</i> 或 <i>percentage</i>	所有文本行的左或右缩排 <sup>1</sup> 。若在前一行指定 <i>leftindent</i> ，并且确定的位置在当前文本位置的左侧，则将为当前行忽略此选项。默认值: <i>o</i>
<i>rotate</i>	<i>float</i>	旋转坐标系统，以定界框的左下角为中心并将指定的值作为旋转角（以度为单位）。这将导致框和文本被旋转。一旦文本已放置，旋转将重置。默认值: <i>o</i>
<i>ruler</i> <sup>2</sup>	<i>list of floats</i> 或 <i>percentages</i>	当 <i>hortabmethod=ruler</i> 时，制表键绝对值列表 <sup>1</sup> 。这个列表可包含多至 32 个非负数的值，以上升秩序排列。默认值: <i>hortabsize</i> 的整数倍数
<i>tabalignment</i>	<i>list of keywords</i>	制表键对齐方式列表中的每个项都定义 <i>ruler</i> 选项中的相应的项的对齐方式（默认值: <i>left</i> ） <i>center</i> 制表键内文本向中对齐 <i>decimal</i> 第一个 <i>tabalignchar</i> 示例将在制表键位置左对齐。如果找不到 <i>tabalignchar</i> ，则将采用向右对齐。 <i>left</i> 制表键内文本向左对齐。 <i>right</i> 制表键内文本向右对齐。
<i>verticalalign</i>	<i>keyword</i>	定界框内的文本垂直对齐（默认值: <i>top</i> ）: <i>top</i> 排版文本将从第一行开始，由上至下。若文本不能填满定界框，会在文本下端留出空白。 <i>center</i> 文本将以定界框的垂直中心对齐。若文本不能填满定界框，会在文本上、下端留出空白。 <i>bottom</i> 排版文本将从最后一行开始，由下至上。若文本不能填满定界框，会在文本上端留出空白。 <i>justify</i> 文本将对齐定界框的顶部及底部。为了获得此效果，行距可增加至 <i>linespreadlimit</i> 定义的最大限。当 <i>firstlinedist=leading</i> 时，第一行的高度将会被增加。

表 6.6 *textflow* 块属性（续）

关键字	类型	可能的值和说明
控制分行算法的属性：		
<i>adjustmethod</i>	<i>keyword</i>	当使用大于等于 <i>minspacing</i> 并小于等于 <i>maxspacing</i> 的单词距离选项进行增加或缩减单词间距离的调整后，仍有部分文本不能排入同一文本行时，所采用的方法。默认值： <i>auto</i> <i>auto</i> 按顺序使用以下方法： <i>shrink</i> 、 <i>spread</i> 、 <i>nofit</i> 、 <i>split</i> 。 <i>clip</i> 类似于 <i>nofit</i> ，除了将剪切定界框（考虑 <i>rightindent</i> 选项）右边缘的长的部分以外。 <i>nofit</i> 末尾单词将移动到下一行，以使剩余（短）行将不会少于 <i>nofitlimit</i> 选项中指定的百分比即使对齐的段落也可能看起来有点不规则。 <i>shrink</i> 若单词不能完全排入当前行，将缩减文本直至 <i>shrinklimit</i> 。若仍不能满足要求，则将使用 <i>nofit</i> 方法。 <i>split</i> 末尾单词将不会移动到下一行，但是将强制对其添加连字符。为文本字体，而非为符号字体插入连字符。 <i>spread</i> 末尾字将移动到下一行，并将根据 <i>spreadlimit</i> 通过增加词中字符之间的距离对齐剩余（短）行。若仍无法实现对齐，则将使用 <i>nofit</i> 方法。
<i>linespreadlimit</i>	<i>float</i> 或 <i>percentage</i>	（只有当 <i>verticalalign=justify</i> 时有效）垂直排版行距调整最大值，以用户坐标或相关行距的百分比为单位。默认值： <i>200%</i>
<i>maxlines</i>	<i>integer</i> 或 <i>keyword</i>	该属性值可以是整数代表定界框可容纳文本行的最大值；亦可是关键字 <i>auto</i> ，代表定界框所含文本行数没有限制。若已放置最大行数，则 <i>fit_textflow()</i> 将返回 <i>string_boxfull</i> 。
<i>maxspacing</i> <i>minspacing</i>	<i>float</i> 或 <i>percentage</i>	单词间最大或最小的距离（以用户坐标或间隔字符的宽度的百分比为单位）。计算的字间距受提供的值的限制（但是仍将添加 <i>wordspacing</i> 选项）。默认值： <i>minspacing=50%</i> 、 <i>maxspacing=500%</i>
<i>nofitlimit</i>	<i>float</i> 或 <i>percentage</i>	对于 <i>nofit</i> 方式，所需文本行长度的下限（以在用户坐标中或定界框的宽度的百分比为单位）。默认值： <i>75%</i>
<i>shrinklimit</i>	<i>percentage</i>	对于 <i>shrink</i> 方式，可缩减文本的下限；计算的收缩因素受限于提供的值，但是将乘以 <i>horizscaling</i> 选项提供的值。默认值： <i>85%</i>
<i>spreadlimit</i>	<i>float</i> 或 <i>percentage</i>	对于 <i>spread</i> 方式，可调整字间距的上限；（以用户坐标中或字体大小的百分比为单位）；计算的字符距离将被添加到 <i>charspacing</i> 选项的值。默认值： <i>0</i>

1. 在用户坐标中，或作为定界框的宽度的百分比
2. 可以在“块属性”对话框中的“制表位”区域编辑标尺。

6.4.3 Image 属性

图像相关的属性适用于 *Image* 类型块（除常规属性之外）。所有图像相关的属性都可以被共享。表 6.7 列出了图像相关的属性。

表 6.7 *Image* 块属性

关键字	类型	可能的值和说明
<i>defaultimage</i>	<i>string</i>	若客户端未提供替代图像，将使用的图像的路径名称。建议使用不带绝对路径的文件名，并在 <i>PPS</i> 客户应用程序中使用 <i>SearchPath</i> 功能。这将使块处理独立于平台及文件系统的细节。
<i>dpi</i>	<i>float list</i>	以像素 / 英寸为单位，用一或两个数值指定所需图像在水平和垂直方向的分辨率。如果此值为 <i>0</i> ，若图像内部的分辨率存在，将使用内部分辨率；否则，将使用 <i>72 dpi</i> 分辨率。若 <i>fitmethod</i> 属性与关键字 <i>auto</i> 、 <i>meet</i> 、 <i>slice</i> 或 <i>entire</i> 中的某一关键字一起被提供，此属性将被忽略。默认值： <i>0</i>
<i>scale</i>	<i>float list</i>	用一或两个数值指定在水平和垂直方向的缩放比例。若 <i>fitmethod</i> 属性与关键字 <i>auto</i> 、 <i>meet</i> 、 <i>slice</i> 或 <i>entire</i> 中的某一关键字一起被提供，此属性将被忽略。默认值： <i>1</i>

### 6.4.4 PDF 属性

PDF 相关的属性适用于 *PDF* 类型块（除常规属性之外）。所有 PDF 相关的属性都可以被共享。表 6.8 列出了 PDF 相关的属性。

表 6.8 PDF 块属性

关键字	类型	可能的值和说明
<i>defaultpdf</i>	<i>string</i>	若客户端未提供替代 <i>PDF</i> ，将使用的 <i>PDF</i> 文档的路径名称。建议使用不带绝对路径的文件名，并在 <i>PPS</i> 客户应用程序中使用 <i>SearchPath</i> 功能。这将使块处理独立于平台和文件系统的细节。
<i>defaultpdfpage</i>	<i>integer</i>	默认 <i>PDF</i> 文档的页数。默认值：1
<i>scale</i>	<i>float list</i>	用一或两个数值指定在水平和垂直方向必需的缩放比例因素。若 <i>fitmethod</i> 属性与关键字 <i>auto</i> 、 <i>meet</i> 、 <i>slice</i> 或 <i>entire</i> 中的某一关键字一起被提供，此选项将被忽略。默认值：1
<i>pdiusebox</i>	<i>keyword</i>	（其值可为： <i>media</i> 、 <i>crop</i> 、 <i>bleed</i> 、 <i>trim</i> 、 <i>art</i> ）使用已放置的页的媒体框、裁剪框、出血框、裁切框或作品框以确定页的大小（请参见表 8.46）。默认值： <i>crop</i>

### 6.4.5 自定义属性

自定义属性适用于各种块类型（除常规属性和特定块类型的属性之外）。自定义属性是可选的，并不能共享。表 6.9 列出了自定义属性的命名规则。

表 6.9 自定义块属性

关键字	类型	可能的值和说明
任何名称都不包含 [、]、/ 这三个字符。	<i>string</i> 、 <i>name</i> 、 <i>float</i> 、 <i>float list</i>	与自定义属性对应的值的说明完全适用于客户端应用程序。

## 6.5 询问块名及其属性

除了自动块处理外，PDFlib 支持一些可用于列举块名并询问标准属性或自定义属性的功能。

**寻找块的数目和名称** 客户端必须不知道导入的页上的块的名称和数量，因为这些也可以查询。以下语句返回当前页的块数：

```
blockcount = oPDF.get_pdi_value("vdp/blockcount", doc, page, 0)
```

以下语句返回页上的块号为 5 的块的名称（从 0 开始进行块计数），若此块不存在，则返回一个空字符串（不过，如果 *pdiwarning* 参数或选项设置为 *true*，则将引发异常）：

```
blockname = oPDF.get_pdi_parameter("vdp/Blocks[5]/Name", doc, page, 0)
```

返回的块名称可以在后面用于查询块的属性或用于使用文本、图像或 PDF 内容填充块。

在路径语法中，对于为块属性寻址，以下表达式都是同等的，假定带序列 *<number>* 的块 *Name* 属性已设置为 *<blockname>*：

```
Blocks[<number>]/  
Blocks/<blockname>/
```

**查找块坐标** 两个坐标对 (*llx, lly*) 和 (*urx, ury*) 描述名为 *foo* 的块的左下角和右上角可以按如下所示查询：

```
llx = oPDF.get_pdi_value("vdp/Blocks/foo/Rect[0]", doc, page, 0)  
lly = oPDF.get_pdi_value("vdp/Blocks/foo/Rect[1]", doc, page, 0)  
urx = oPDF.get_pdi_value("vdp/Blocks/foo/Rect[2]", doc, page, 0)  
ury = oPDF.get_pdi_value("vdp/Blocks/foo/Rect[3]", doc, page, 0)
```

请注意，在默认的用户坐标系（该系统原点位于左下角，也许被页的裁剪框修改）中提供这些坐标，而块增效工具根据 Acrobat 的用户界面坐标系（该系统原点位于页的左上角）显示坐标。因为用于重写块坐标的 *Rect* 选项不会考虑裁剪框使用的任何修改，所以若裁剪框存在，则从原始块查询的坐标不能直接作为新坐标使用。作为工作区，您可以使用 *refpoint* 和 *boxsize* 选项。

还请注意，当查询块坐标时，不会考虑 *topdown* 参数。

**不存在的块属性及其缺省值** 由于 API 的局限性，*get\_pdi\_value()* 不能被用于判断一个块属性是确实存在且其值为 0，还是在文档中缺失该属性从而使用缺省值。

如果是 *string* 属性缺失，*get\_pdi\_parameter()* 将会返回一个空的字符串。如果是 *numerical* 属性缺失（从而使用缺省值），*get\_pdi\_value()* 将会返回值 0（零）。因为大多数的 *numerical* 属性缺省值为 0，故这类情况无需特殊处理。然而需注意一些 *numerical* 属性如：*linewidth*、*horizscaling*、*leading*，和 *scale*，它们的缺省值非 0（并且 0 值不合逻辑）。在这种情况下，用户需做特殊处理以便在 *get\_pdi\_value()* 返回 0 值时，可正确匹配相应属性的缺省值。

PDFlib pCOS 可在 PDF 文件里查询包括块属性在内的各种信息。

**查询自定义属性** 可以按照以下示例中那样查询自定义属性，在此示例中，从名为 *b1* 的块中查询属性 *zipcode*：

```
zip = oPDF.get_pdi_parameter("vdp/Blocks/b1/Custom/zipcode", doc, page, 0)
```

若不知道块中包含哪些自定义块属性，可以在运行时获得。在此示例中，从名为 *b1* 的块中查询第一个自定义块属性的名称：

```
propname = oPDF.get_pdi_parameter("vdp/Blocks/b1/Custom[0].key", doc, page, 0)
```

用增加下标变量来替代 *o*，可获得所有自定义属性的名字。当一个空字符串返回时意味着已查询了该块所有自定义属性。

自定义属性的命名空间 为了在对不同的源的 PDF 文档进行交换时避免混淆，建议在所有自定义属性名称中使用 **Internet** 域名作为公司特定的前缀，紧跟冒号“:”和实际属性名称。例如，ACME 公司应使用以下属性名称：

```
acme.com:digits
acme.com:refnumber
```

由于标准属性和自定义属性是按照不同方式在块中存储，所以标准 PDFlib 属性名称（在第 128 页上的第 6.4 节“用于自动化处理的标准属性”中定义）将决不会与自定义属性名称起冲突。

## 6.6 PDFlib 块规格

PDFlib 块语法完全符合 PDF 引用，该引用指定一个扩展机制，此机制允许应用程序存储附加到包含一个 PDF 页的数据结构的私人数据。下面提供 PDFlib 块语法的具体描述，以方便一些用户想通过 PDFlib 块增效工具之外的方式建立 PDFlib 块。对于使用 PDFlib 增效工具的用户，可跳过此章节。

### 6.6.1 PDFlib 块的 PDF 对象结构

PDF 页字典包含了 */PieceInfo* 项，其值将指向另一个字典。此字典含有 */PDFlib* 项，其值指向一个应用程序数据字典。此应用程序数据字典包含表 6.10 中列出的两个标准项。

表 6.10 PDFlib 应用程序数据字典项

项	类型	值
<i>LastModified</i>	<i>date string</i>	（必需的）有关页上的块创建或最近被修改的日期和时间。
<i>Private</i>	<i>dictionary</i>	（必需的）块列表（请参见表 6.11）

块列表是一个包含与块处理相关的常规信息及当前页所有块的字典。表 6.11 列出了块列表字典中的项。

表 6.11 块列表字典中的项

项	类型	值
<i>Version</i>	<i>number</i>	（必需的）文件要遵从的块规格的版本号。此文档说明块规格的版本 6。
<i>Blocks</i>	<i>dictionary</i>	（必需的）每个项都是一个包含块的名称的名称对象；相应的值是用于此块的块字典（请参见表 6.13）。块字典中的 <i>/Name</i> 项必须与此字典中的块的名称一致。
<i>PluginVersion</i>	<i>string</i>	（必需的，除非 <i>pdfmark</i> 项存在 <sup>1</sup> ）包含 PDFlib 块增效工具的版本识别的字符串，该增效工具已用于创建块。
<i>pdfmark</i>	<i>boolean</i>	（必需的，除非 <i>PluginVersion</i> 项存在 <sup>1</sup> ）若块列表是用 <i>pdfmarks</i> 生成的，则此值必须为 <i>true</i> 。

1. *PluginVersion* 和 *pdfmark* 这两项中必须有一个项存在。



块属性的数据类型 属性支持相同的数据类型作为选项列表（请参见第 157 页上的第 8.1.2 节“选项列表”），句柄和操作列表除外。表 6.12 详细记录这些类型如何映射为 PDF 数据类型。

表 6.12 用于块属性的数据类型

块类型	PDF 类型	备注
<i>boolean</i>	<i>boolean</i>	
<i>string</i>	<i>string</i>	
<i>keyword</i>	<i>name</i>	提供特殊属性支持的关键字的列表之外的关键字是错误的。
<i>float, integer</i>	<i>number</i>	选项列表支持点和逗号作为十进制分隔符，而 <i>PDF</i> 数字仅支持小数点。
<i>percentage</i>	<i>array with two elements</i>	数组中的第一个元素是数字，第二个元素是含有百分比符号的字符串。
<i>color</i>	<i>array with two elements</i>	数组中的第一个元素指定一个色彩空间，第二个元素指定一个色彩值，如下所示。数组中的第一个元素支持以下项： <i>/DeviceGray</i> 第二个元素是单个灰度值。 <i>/DeviceRGB</i> 第二个元素是含有三个元素（ <i>RGB</i> 值）的数组。 <i>/DeviceCMYK</i> 第二个元素是含有四个元素（ <i>CMYK</i> 值）的数组。 <i>[/Separation/spotname]</i> 第一个元素是一个含有关键字 <i>/Separation</i> 和颜色名称的数组。第二个元素是一个淡色值。 <i>[/Lab]</i> 第一个元素是包含关键字 <i>/Lab</i> 的数组。第二个元素是一个含有三个元素（ <i>Lab</i> 值）的数组。 若要指定不使用某种颜色，必须省略相应的属性。

块字典项 块字典可能包含表 6.13 中的项。根据常规组中的 */Subtype* 项，仅可能存在某一文本、图像或 PDF 组中的项（请参见表 6.4）。

表 6.13 块字典中的项

项	类型	值
常规属性		（其中的一些项是必需的）根据表 6.4 的常规属性
<i>text</i> 属性		（可选的）根据表 6.5 和表 6.6 的 <i>Text</i> 和 <i>textflow</i> 属性
<i>image</i> 属性		（可选的）根据表 6.7 的 <i>Image</i> 属性
<i>PDF</i> 属性		（可选的）根据表 6.8 的 <i>PDF</i> 属性
<i>Custom</i>	<i>dict</i>	（可选的）一个含有自定义属性选项及其相应值的字典（根据表 6.9）。
<i>Internal</i>	<i>dict</i>	（可选的）保留此项以供私人使用，且应用程序不应依赖于此项的存在或特定的行为。目前，此项用于维护转换的表单域和相应的块之间的关系。

示例 以下片段显示用于两个块的 PDF 代码，这两个块分别为称作 *job\_title* 的文本块和称作 *logo* 的图像块。文本块包含称作 *format* 的自定义属性：

```
<<
    /Contents 12 0 R
    /Type /Page
    /Parent 1 0 R
    /MediaBox [ 0 0 595 842 ]
    /PieceInfo << /PDFlib 13 0 R >>
>>

13 0 obj
```

```

<<
  /Private <<
    /Blocks <<
      /job_title 14 0 R
      /logo 15 0 R
    >>
    /Version 6
    /PluginVersion (2.2.0)
  >>
  /LastModified (D:20050813200730)
>>
endobj

14 0 obj
<<
  /Type /Block
  /Rect [ 70,740,200,800 ]
  /Name /job_title
  /Subtype /Text
  /fitmethod /auto
  /fontname (Helvetica)
  /fontsize 12
  /Custom << /format 5 >>
>>
endobj

15 0 obj
<<
  /Type /Block
  /Rect [ 250 700 400 800 ]
  /Name /logo
  /Subtype /Image
  /fitmethod /auto
>>

```

## 6.6.2 利用 pdfmarks 生成 PDFlib 块

作为使用增效工具创建 PDFlib 块的一个替换，可以通过将适当的 pdfmark 命令插入 PostScript 流并将其制作为 PDF 来生成块。有关 pdfmark 运算符的详细信息将在 Acrobat 文件中讨论。以下片段显示前一节中可用于生成块定义的 pdfmark 运算符：

```

% ----- Setup for the blocks on a page -----
[/objdef {B1} /type /dict /OBJ pdfmark          % Blocks dict

[{ThisPage} <<
  /PieceInfo <<
    /PDFlib <<
      /LastModified (D:20050813200730)
      /Private <<
        /Version 6
        /pdfmark true
        /Blocks {B1}
      >>
    >>
  >>
>> /PUT pdfmark

% ----- text block -----

```

```
[{B1} <<
    /job_title <<
        /Type /Block
        /Name /job_title
        /Subtype /Text
        /Rect [ 70,740,200,800 ]
        /fitmethod /auto
        /fontsize 12
        /fontname (Helvetica)
        /Custom << /format 5 >>
    >>
>> /PUT pdfmark

% ----- image block -----
[{B1} <<
    /logo <<
        /Type /Block
        /Name /logo
        /Subtype /Image
        /Rect [ 250 700 400 800 ]
        /fitmethod /auto
    >>
>> /PUT pdfmar
```





# 7 生成各种风格的 PDF

## 7.1 Acrobat 和 PDF 版本

按照用户的选项，PDFlib 可生成 PDF 1.3 (Acrobat 4)、PDF 1.4 (Acrobat 5)、PDF 1.5 (Acrobat 6) 或 PDF 1.6 (Acrobat 7) 的文件。可使用 `begin_document()` 中的 `compatibility` 选项控制生成文件的版本。

**用于 PDF 1.4 或更高版本的 PDFlib 功能** 在 PDF 1.3 兼容模式下，将无法使用 PDFlib 针对 PDF 1.4（参见表 7.1）和 PDF 1.5（参见表 7.2）而设计的功能。在 PDF 1.3 模式下尝试使用这些功能会导致异常。

表 7.1 无法在 PDF 1.3 兼容模式下使用的、用于 PDF 1.4 的 PDFlib 功能

功能	PDFlib API 函数和参数
平滑着色（颜色混合）	<code>shading_pattern()</code> , <code>shfill()</code> , <code>shading()</code>
软蒙版	<code>load_image()</code> , 带有指向像素深度大于 1 位的图像的 <code>masked</code> 选项
128 位加密	<code>begin_document()</code> , 带有 <code>userpassword</code> 、 <code>masterpassword</code> 和 <code>permissions</code> 选项
扩展权限设置	<code>begin_document()</code> , 带有 <code>permissions</code> 选项, 参见表 7.3
CJK 字体的某些 CMap	<code>load_font()</code> , 参见表 4.5
透明度及其他图形状态选项	<code>create_gstate()</code> , 带有 <code>alphaishshape</code> 、 <code>blendmode</code> 、 <code>opacityfill</code> 、 <code>opacitystroke</code> 和 <code>textknockout</code> 选项
某些动作选项	<code>create_action()</code> , 参见表 8.49
某些注释选项	<code>create_annotation()</code> , 参见表 8.51
某些域选项	<code>create_field()</code> 和 <code>create_fieldgroup()</code> , 参见表 8.52
标签 PDF	<code>begin_document()</code> 中的 <code>tagged</code> 选项

**用于 PDF 1.5 或更高版本的 PDFlib 输出功能** 在 PDF 1.3 或 1.4 兼容模式下，将无法使用表 7.2 列出的、PDFlib 针对 PDF 1.5 而设计的功能。在 PDF 1.3 或 PDF 1.4 模式下尝试使用这些功能会导致异常。

表 7.2 无法在 PDF 1.3 或 1.4 兼容模式下使用的、用于 PDF 1.5 的 PDFlib 功能

功能	PDFlib API 函数和参数
某些域选项	<code>create_field()</code> 和 <code>create_fieldgroup()</code> , 参见表 8.52
某些注释选项	<code>create_annotation()</code> , 参见表 8.51
扩展权限设置	<code>begin_document()</code> 中的 <code>permissions=plainmetadata</code> , 参见表 7.3
CJK 字体的某些 CMap	<code>load_font()</code> , 参见表 4.5
标签 PDF	某些 <code>begin_item()</code> 选项, 参见表 8.57 和表 8.58
图层	<code>define_layer()</code> , <code>begin_layer()</code> , <code>end_layer()</code> 和 <code>layer_dependency()</code>
JPEG2000 图像	<code>load_image()</code> 中的 <code>imagetype=jpeg2000</code>

**PDI 导入文档的 PDF 版本** 在所有兼容模式下，只能用 PDI 导入兼容级别相同或较低的 PDF 文档。如果必须导入兼容级别较高的 PDF，必须相应地设置 *compatibility* 选项，参见第 112 页上的第 5.2.3 节“可接受的 PDF 文档”。

**更改文档的 PDF 版本** 如果必须根据特定 PDF 版本创建输出，但需要导入使用较高 PDF 版本的 PDF，您必须将文档转换为所需的较低 PDF 版本，然后才能用 PDI 导入这些文档。可使用 Acrobat 完成转换；具体取决于所使用的 Acrobat 版本：

- ▶ **Acrobat 7:** 可以使用高级、PDF 优化器、兼容版本将文件另存为 PDF 1.3 至 PDF 1.6 格式。
- ▶ **Acrobat 6:** 可以使用文件、减少文件大小 ...，将文件另存为 PDF 1.3 至 PDF 1.5 格式。
- ▶ **Acrobat 5:** 可使用 callas software 开发的附加增效工具（称为 *pdfSaveAs1.3*），将文档转换为 PDF 1.3。可从 callas 网站上下载功能完全的演示版本<sup>1</sup>。因为有些版本的 PDF/X 要求 PDF 1.3，所以这个转换增效工具对处理块和 PDF/X 特别有用（参见第 124 页上的“在块中使用 PDF/X”和第 145 页上的第 7.4.2 节“生成符合 PDF/X 标准的输出”）。

## 7.2 加密 PDF

### 7.2.1 PDF 安全性的优缺点

PDF 支持各种安全功能，可帮助保护文档的内容。这些功能基于使用对称加密的 Acrobat 标准加密处理程序。Acrobat Reader 和完全版的 Acrobat 产品都支持下列安全功能：

- ▶ 通过权限限制某些对 PDF 文档的操作，如打印或提取文本等。
- ▶ 打开文件时要求文档打开口令。
- ▶ 更改任何安全设置都要求许可口令，如权限、文档打开口令或许可口令等设置。如果具有文档打开口令和许可口令，可使用任一口令打开文件，以便阅读或打印。

如果文件具有文档打开口令、许可口令或任何权限限制设置，则表明文件是加密的。

**破解受保护的 PDF 文档** 保护文档使用的加密密钥长度取决于客户端选择的 PDF 兼容级别：

- ▶ 对于 1.3 及较低的 PDF 版本（即 Acrobat 4），密钥长度为 40 位。
- ▶ 对于 PDF 版本 1.4 及更高版本，密钥长度为 128 位。这要求 Acrobat 5 或更高版本。对于 PDF 1.5，密钥长度同样为 128 位，但采用的加密方法略有不同；这要求 Acrobat 6。

大家都知道，用于对称加密的 40 位密钥（如 PDF 中使用的密钥）不是很安全。实际上，使用公开出售的破解软件，可以通过强力攻击停用 40 位 PDF 安全性设置；根据口令的长度和质量，可能需要数天或数周时间。为了最大程度提高安全性，我们建议采取以下措施：

- ▶ 尽可能使用 128 位加密（即 PDF 1.4 兼容设置）。这要求该文档的所有用户都使用 Acrobat 5 或更高版本。
- ▶ 口令长度至少应包含六个字符，且应包含非字母字符。口令不应与配偶姓名、宠物昵称或生日等类似，以防止所谓的词典攻击或口令猜测。重要的一点是，即使采用 128 位加密，也可以在短短几分钟内破解较短的口令。

**访问权限** 设置禁止打印等访问限制将停用 Acrobat 中对应的功能。但是，这并不一定能停用第三方 PDF 浏览器或其他软件的相关功能。是否遵守访问权限具体取决于 PDF 工具的开发人员。实际上，我们已经知道，有几种 PDF 工具会完全忽略权限设置；而且，有些公开出售的 PDF 破解工具也可用来停用任何访问限制。这与破解加密无关；目前没有办法在保持 PDF 文件可以查看的同时，确保不被打印。Adobe 自己的《PDF Reference》中便有相关说明：

1. 参见 [www.callassoftware.com](http://www.callassoftware.com)

PDF 加密本身不能执行加密词典中规定的文档权限。PDF 浏览器的实施者可自行决定，是否尊重文档创建程序根据文件所含权限限制用户对加密 PDF 文件的访问的意图。

### 7.2.2 用 PDFlib 保护文档

口令 可使用 `begin_document()` 中的 `userpassword` 和 `masterpassword` 选项设置口令。PDFlib 与客户端提供的口令的交互方式如下：

- ▶ 如果提供了文档打开口令或权限（参见下文），但没有提供许可口令，则一般用户将能够更改安全性设置。因此，PDFlib 会认为这种情况是个错误。
- ▶ 如果文档打开口令和许可口令相同，将无法区分文件的用户和所有者；这同样不能有效地保护文档。PDFlib 会认为这种情况是个错误。
- ▶ 对于文档打开口令和许可口令，最多可使用 32 个字符。多余的字符将被忽略，但不影响加密效果。不允许空口令。

提供的口令将用于此后生成的所有文档。

口令中的非 ASCII 字符 在口令中使用 `0x20` 至 `0x7E` 范围以外的字符时，应特别注意；传统 ASCII 字符集不包括这些字符。例如，在口令中使用字符 `Ä`。在 Mac 上，该字符的代码是 `0x80`；而在 Windows 上，它的代码是 `0xC4`。因为用户希望在两个平台上都可以使用口令 `Ä` 打开文件，所以 Acrobat 会在应用口令之前将提供的口令转换为一种内部编码（称为 `PDFDocEncoding`）。此编码中不包括的字符将映射到空格字符。`PDFDocEncoding` 包含 Mac 和 Windows 平台的所有字符，但要求对几个字符进行转换。在以上示例中，用户在 Mac 上使用口令 `Ä` 加密文件时，如果直接使用 `Ä` 的代码，PDI 将不能解密文件。因此，PDI 会应用与 Acrobat 相同的口令转换，以确保成功解密用 Mac 或 Windows 版 Acrobat 加密的文件。解密过程中，PDI 将自动检测所需的转换：

- ▶ 如果文档使用 Windows 上的 Acrobat 或者 PDFlib PLOP 2.1 或更高版本加密，将进行 WinAnsi 到 `PDFDocEncoding` 的转换；
- ▶ 如果文档使用 Mac 上的 Acrobat 加密，则进行 MacRoman 到 `PDFDocEncoding` 的转换；
- ▶ 如果文档使用其他软件加密，则不进行转换；这包括 PDFlib PLOP 2.0（但不包括任何更新的版本）。

加密文件时，PDFlib 会像 Windows 上的 Acrobat 那样，用 WinAnsi 编码解释提供的口令，即：它将对提供的文档打开口令和许可口令应用 WinAnsi 到 `PDFDocEncoding` 的转换；在 EBCDIC 平台上，它将在这之前应用 EBCDIC 到 WinAnsi 的转换。

权限 可以使用 `begin_document()` 中的 `permissions` 选项设置访问限制。它包含一个或多个访问限制关键字。设置 `permissions` 选项时，必须同时设置 `masterpassword` 选项；这是因为，如不同时设置，Acrobat 用户很容易就能删除权限设置。默认情况下，允许所有动作。指定访问限制将停用 Acrobat 中相应的功能。无需任何文档打开口令便可应用访问限制。如下示例所示，可以指定多个限制关键字：

```
oPDF.begin_document filename, "permissions {noprint nocopy}"
```

表 7.3 列出了所有支持的访问限制关键字。如表中的详细信息所述，有些关键字要求兼容 PDF 1.4 或更高版本。如果 PDF 输出版本过低，它们将被拒绝。

表 7.3 `begin_document()` 中 `permissions` 选项的访问限制关键字

关键字	说明
<code>noprint</code>	Acrobat 将阻止打印文件。
<code>nomodify</code>	Acrobat 将阻止用户添加表单域或做任何其他更改。

表 7.3 `begin_document()` 中 `permissions` 选项的访问限制关键字

关键字	说明
<code>nocopy</code>	<i>Acrobat</i> 将阻止复制或提取文本或图形，并将停用辅助工具接口。
<code>noannots</code>	<i>Acrobat</i> 将阻止添加或更改注释或表单域。
<code>noforms</code>	(PDF 1.4) <i>Acrobat</i> 将阻止填充表单域；即使没有指定 <code>noannots</code> ，也是如此。
<code>noaccessible</code>	(PDF 1.4) <i>Acrobat</i> 将阻止提取文本或图形，以限制辅助工具（如读屏器程序）。
<code>noassemble</code>	(PDF 1.4) 即使没有指定 <code>nomodify</code> ， <i>Acrobat</i> 也将阻止插入、删除或旋转页面的操作，并将阻止创建书签和缩览图。
<code>nohiresprint</code>	(PDF 1.4) <i>Acrobat</i> 将阻止高分辨率打印。如果没有指定 <code>noprint</code> ，则只能使用“打印为图像”功能；这将打印以低分辨率呈现的页面。
<code>plainmetadata</code>	(PDF 1.5) 保持文档元数据不加密；即使加密文档，也是如此。在这里，是指 <i>XMP</i> 元数据相关，与文档信息域元数据无关。

## 7.3 网页优化的（线性化的）PDF

PDFlib 可以对 PDF 文档应用一个称为线性化处理的过程（在 *Acrobat 4* 中，线性化的 PDF 称为优化，在 *Acrobat 5* 和更高版本中称为快速 *Web* 查看）。线性化处理可重新组织 PDF 文件内的对象，并添加可用来提高访问速度的补充信息。

非线性化的 PDF 必须完全传送到客户端，而 Web 服务器可以使用一个称为 *byteserving* 的过程传送线性化的 PDF，一次只传送一个页面。这允许 *Acrobat*（作为浏览器增效工具运行）单独检索 PDF 文档的各个部分。因此，不必等待从服务器下载整个文档，即可向用户显示文档的第一个页面。这样可增强用户的体验。

请注意，Web 服务器向浏览器传送 PDF 数据，而非 PDFlib。而 PDFlib 负责准备 *byteserving* 的 PDF 文件。要利用 *byteserving* PDF，必须满足下列所有要求：

- ▶ PDF 文档必须进行线性化处理，这可通过 `begin_document()` 中的 `linearize` 选项实现。在 *Acrobat* 中，可以通过查看文档属性来检查文件是否已经过线性化处理；具体属性是快速 *Web* 查看：是。
- ▶ Web 服务器必须支持 *byteserving*。基础字节范围协议是 HTTP 1.1 的一部分，因此所有最新版 Web 服务器都实施了该协议。具体讲，下列 Web 服务器支持 *byteserving*：

Microsoft Internet Information Server (IIS) 3.0 及更高版本  
Apache 1.2.1 及更高版本；但是 Apache 1.3.14（不包括其他版本）存在一个错误，会阻止 *byteserving*

- ▶ 用户必须作为浏览器增效工具使用 *Acrobat*，并在 *Acrobat* 中启用“每次一页”下载方式（*Acrobat 6*：编辑、首选项、因特网、允许快速 *Web* 查看；*Acrobat 5*：编辑、首选项、常规 ...、选项、允许快速 *Web* 查看）。请注意，默认启用该选项。

通过 Web 发送时，PDF 文件越大（以页数或 MB 值来衡量），就越能发挥线性化的优势。

注：由于要添加额外的线性化信息，对 PDF 文档进行线性化处理一般会导致文件大小略有增大。

**线性化的临时存储空间要求** 在进行线性化处理之前，PDFlib 必须创建整个文档；文档创建完毕后，必须作为单独的步骤应用线性化过程。因此，PDFlib 对线性化有额外的存储要求。要求的临时存储空间大体相当于生成文档的大小（未进行线性化处理）。根据 `begin_document()` 中的 `inmemory` 选项，PDFlib 会将线性化数据置于内存或临时磁盘文件内。



# 7.4 PDF/X

## 7.4.1 PDF/X 系列标准

ISO 15930 标准系列规定的 PDF/X 格式旨在提供一种一致、可靠的 PDF，用来传送适用于商业印刷的数据<sup>1</sup>。PDFlib 能够根据以下 PDF/X 版本生成输出并处理输入。

**ISO 15930-1 规定的 PDF/X-1:2001 和 PDF/X-1a:2001** 这些用于“盲交换”的标准基于 PDF 1.3，并支持 CMYK 和专色数据；所谓盲交换，就是在交换印刷数据时不要求事先讨论任何技术问题。这些标准明确禁止使用 RGB 和设备无关颜色（基于 ICC，Lab）。PDF/X-1:2001 支持一种在 PDF 工作流程中集成旧文件（如 TIFF/IT）的机制；此标准已经过时了。PDF/X-1a:2001 不包含这种支持旧文件的功能，被广泛用来交换出版物广告及其他应用（尤其是在北美）。

**ISO 15930-4 中规定的 PDF/X-1a:2003** 此标准的前身是 PDF/X-1a:2001。它基于 PDF 1.4，禁止使用某些功能（例如，透明度）。PDF/X-1a:2003 完全包括在 PDF/X-3:2003 中，支持 CMYK 和专色，以及 CMYK 输出设备。

**ISO 15930-5 中规定的 PDF/X-2:2003** 此标准针对“部分交换”；这种交换方式要求文件的提供方和接收方事先讨论相关技术问题。根据此标准，PDF 文档可以引用外部实体（指向当前文档外部的 PDF 页面）。PDF/X-2:2003 基于 PDF 1.4。作为 PDF/X-3:2003 的扩展，它支持设备无关颜色。

**ISO 15930-3 中规定的 PDF/X-3:2002** 此标准基于 PDF 1.3；除灰度、CMYK 和专色之外，它还支持基于设备无关颜色的现代工作流程。此标准在欧洲国家 / 地区尤为盛行。输出设备可以采用单色、RGB 或 CMYK。

**ISO 15930-6 中规定的 PDF/X-3:2003** 此标准的前身是 PDF/X-3:2002。它基于 PDF 1.4，禁止使用某些功能（例如，透明度）。

如下文引用的 PDF/X 标准没有带任何标准颁布年份信息，则指该标准的所有版本。例如，PDF/X-3 指 PDF/X-3:2002 和 PDF/X-3:2003。

注：PDF/X-1:2001、PDF/X-1a:2001 和 PDF/X-1a:2003 模式不支持 PANTONE® 颜色。

## 7.4.2 生成符合 PDF/X 标准的输出

可通过以下方法，使用 PDFlib 创建符合 PDF/X 标准的输出：

- ▶ PDFlib 将自动完成几种正式的 PDF/X 设置，如 PDF 版本号 和 PDF/X 规范等级密钥等。
- ▶ PDFlib 客户端必须明确使用表 7.4 中列出的某些函数调用或参数设置。
- ▶ PDFlib 客户端不能使用表 7.5 中列出的某些函数调用和参数设置。
- ▶ 从现有符合 PDF/X 标准的文档中导出页面时，还应遵守其他规则（参见第 148 页上的第 7.4.3 节“用 PDI 导入 PDF/X 文档”）。

**要求的操作** 表 7.4 列出了生成 PDF/X 兼容输出要求的所有操作。除非另有说明，否则其中列出的各项操作均适用于所有 PDF/X 规范等级。处于 PDF/X 模式下时，如果不调用要求的函数，将触发异常。

**禁止的操作** 表 7.5 列出了在生成 PDF/X 兼容输出时所有禁止使用的操作。除非另有说明，否则其中列出的各项操作均适用于所有 PDF/X 规范等级。处于 PDF/X 模式下时，调用禁止

1. 请参见 [www.pdfx3.org](http://www.pdfx3.org) 和 [www.pdf-x.com](http://www.pdf-x.com)

表 7.4 实现 PDF/X 兼容性必须执行的操作

项目	实现 PDF/X 兼容性的 PDFlib 函数和参数要求
规范等级	<code>begin_document()</code> 中的 <code>pdfx</code> 选项必须设为要求的 PDF/X 规范等级。
输出条件 (输出方法)	在 <code>begin_document()</code> 之后, 必须立即调用 <code>usage=outputintent</code> 的 <code>load_iccprofile()</code> 或 <code>action=copyoutputintent</code> 的 <code>process_pdi()</code> 。如果使用 <i>HKS</i> 或 <i>Pantone</i> 专色、基于 ICC 的颜色或 <i>Lab</i> 颜色, 则必须嵌入输出设备的 ICC 配置文件; 这种情况下, 不允许使用标准的输出条件。PDF/X-1 和 PDF/X-1a: 输出设备必须为单色或 CMYK 设备; PDF/X-3: 输出设备必须为单色、RGB 或 CMYK 设备。
字体嵌入	将 <code>load_font()</code> 的 <code>embedding</code> 选项设为 <code>true</code> , 启用字体嵌入。
页面大小	页面框可通过裁剪框、出血框、裁切框和作品框参数设置, 必须满足以下要求: ▶ 必须设置裁切框或作品框, 但不能同时设置这两个框条目。如果既没有裁切框也没有作品框, 若裁剪框存在, PDFlib 将其作为裁切框, 否则将媒体框作为裁切框。 ▶ 出血框 (如果有) 必须完全包含作品框和裁切框。 ▶ 裁剪框 (如果有) 必须完全包含作品框和裁切框。
灰度颜色	PDF/X-3: 如果使用灰度图像, 或者 <code>setcolor()</code> 使用灰度色彩空间, 并且 PDF/X 输出条件不是 CMYK 或灰度设备, 则必须设置 <code>begin_page_ext()</code> 中的 <code>defaultgray</code> 选项。
RGB 颜色	PDF/X-3: 如果使用 RGB 图像, 或者 <code>setcolor()</code> 使用 RGB 色彩空间, 并且 PDF/X 输出条件不是 RGB 设备, 则必须设置 <code>begin_page_ext()</code> 中的 <code>defaultrgb</code> 选项。
CMYK 颜色	PDF/X-3: 如果使用 CMYK 图像, 或者 <code>setcolor()</code> 使用 CMYK 色彩空间, 并且 PDF/X 输出条件不是 CMYK 设备, 则必须设置 <code>begin_page_ext()</code> 中的 <code>defaultcmkyk</code> 选项。
文档信息关键字	必须用 <code>set_info()</code> 设置 <i>Creator</i> 和 <i>Title</i> 信息关键字。

的函数将触发异常。但是, 根据 `imagewarning` 参数的设置, 不可接受的图像可能不会导致异常。同样, 如果导入的 PDF 页面不匹配当前 PDF/X 规范等级, 对应的 PDI 调用将失败, 而不触发异常 (具体取决于 `pdiwarning` 参数)。

表 7.5 实现 PDF/X 兼容性必须避免的操作

项目	实现 PDF/X 兼容性应避免的 PDFlib 函数和参数
灰度颜色	PDF/X-1 和 PDF/X-1a: 必须避免使用 <code>begin_page_ext()</code> 中的 <code>defaultgray</code> 选项。
RGB 颜色	PDF/X-1 和 PDF/X-1a: 必须避免使用 RGB 图像和 <code>begin_page_ext()</code> 中的 <code>defaultrgb</code> 选项。
CMYK 颜色	PDF/X-1 和 PDF/X-1a: 必须避免使用 <code>begin_page_ext()</code> 中的 <code>defaultcmkyk</code> 选项。
基于 ICC 的颜色	PDF/X-1 和 PDF/X-1a: 必须避免使用 <code>setcolor()</code> 中的 <code>iccbasedgray/rgb/cmyk</code> 色彩空间以及 <code>setcolor:iccprofilegray/rgb/cmyk</code> 参数。
Lab 颜色	PDF/X-1 和 PDF/X-1a: 必须避免使用 <code>setcolor()</code> 中的 <i>Lab</i> 色彩空间。
注释和表单域	必须避免在出血框内使用注释 (如果没有出血框, 应避免在裁切框 / 作品框内使用注释): <code>create_annotation()</code> 、 <code>create_field()</code> 及相关的已淘汰的函数。
动作和 JavaScript	必须避免使用任何包括 JavaScript 的动作: <code>create_action()</code> 及相关的已淘汰的函数
图像	PDF/X-1 和 PDF/X-1a: 必须避免使用带 RGB、基于 ICC、YCbCr 或 Lab 颜色的图像。对于着色图像, 用来替代专色的颜色必须满足相同的条件。  必须避免 <code>load_image()</code> 中的 <code>OPI-1.3</code> 和 <code>OPI-2.0</code> 选项。
透明度	必须避免对图像使用软蒙版: 除非蒙版指向 1 位图像, 否则必须避免使用 <code>load_image()</code> 的 <code>mask</code> 选项。  除非设置的值为 1, 否则必须避免使用 <code>create_gstate()</code> 的 <code>opacityfill</code> 和 <code>opacitystroke</code> 选项。
查看程序首选项 / 浏览和打印区域	对 <code>set_parameter()</code> 使用 <code>viewarea</code> 、 <code>viewclip</code> 、 <code>printarea</code> 和 <code>printclip</code> 关键字时, 不允许使用除 <i>media</i> 或 <i>bleed</i> 之外的其他值。
文档信息关键字	必须避免对 <code>set_info()</code> 使用除 <i>True</i> 或 <i>False</i> 之外的 <i>Trapped</i> 信息关键字。

表 7.5 实现 PDF/X 兼容性必须避免的操作

项目	实现 PDF/X 兼容性应避免的 PDFlib 函数和参数
安全性	<p>PDF/X-1（而非 PDF/X-1a）：必须避免使用 <code>userpassword</code> 选项及 <code>begin_document()</code> 中 <code>permissions</code> 选项的值 <code>noprint</code>。</p> <p>PDF/X-1a 和 PDF/X-3：必须避免使用 <code>begin_document()</code> 中的 <code>userpassword</code>、<code>masterpassword</code> 和 <code>permissions</code> 选项。</p>
PDF 版本 / 兼容性	<p>在调用 <code>begin_document()</code> 函数时，由于 PDFlib 会自动完成 <code>compatibility</code> 选项的填写，所以应必须避免使用 <code>compatibility</code> 选项（有关不同 PDF 版本中各功能的详细信息，参见表 7.1 和表 7.2）。</p> <p>PDF/X-1:2001、PDF/X-1a:2001 和 PDF/X-3:2002 都基于 PDF 1.3。必须避免使用要求 PDF 1.4 或更高版本的操作，例如透明度或软蒙版。</p> <p>PDF/X-1a:2003、PDF/X-2:2003 和 PDF/X-3:2003 都基于 PDF 1.4。必须避免使用要求 PDF 1.5 的操作，例如图层。</p>
PDF 导入 (PDI)	导入的文档必须遵守与输出文档相同的 PDF/X 等级，且必须已根据相同的输出方法做了预先处理。

**标准输出条件** 输出条件可定义预期的目标设备，这主要用于实现可靠的校样。输出方法可由 ICC 配置文件指定，也可通过提供标准输出方法的名称指定。表 7.6 列出了 PDFlib 已知的标准输出方法名称。可使用 `StandardOutputIntent` 资源类别定义其他标准输出方法（参见第 42 页上的第 3.1.5 节“资源配置和文件搜索”）。用户应确保只添加 PDF/X 处理软件能够识别的标准输出方法名称。可通过以下方式引用标准输出方法：

```
oPDF.load_iccprofile("CGATS TR 001", "usage outputintent");
```

创建 PDF/X-3 输出或者使用任何 HKS、Pantone、Lab 颜色或基于 ICC 的颜色时，不允许使用标准输出方法，但必须嵌入输出设备的 ICC 配置文件。

表 7.6 PDF/X 的标准输出方法

输出方法	描述
CGATS TR 001	美国 SWOP（出版物）印刷
OF COM PO P1 F6o	ISO 12647-2、阳图版、1 型纸（光亮铜版纸）
OF COM PO P2 F6o	ISO 12647-2、阳图版、2 型纸（粗面铜版纸）
OF COM PO-P3 F6o <sup>1</sup>	ISO 12647-2、阳图版、3 型纸（低克重铜版卷筒纸）
OF COM PO P4 F6o	ISO 12647-2、阳图版、4 型纸（白色胶印胶版纸）
OF COM NE P1 F6o	ISO 12647-2、阴图版、1 型纸（光亮铜版纸）
OF COM NE P2 F6o	ISO 12647-2、阴图版、2 型纸（粗面铜版纸）
OF COM NE P3 F6o	ISO 12647-2、阴图版、3 型纸（低克重铜版卷筒纸）
OF COM NE P4 F6o	ISO 12647-2、阴图版、4 型纸（白色胶印胶版纸）
SC GC2 CO F3o	ISO 12647-5、2 类色域、传统紫外线或水性风干纸
Ifra NP_40lcm_neg+CTP_o5.00	冷凝胶版印刷（计算机直接制版）

1. 尽管 Po 与 P3 间的破折号看上去不太一样，这实际上是标准规定的要求。

### 7.4.3 用 PDI 导入 PDF/X 文档

将现有 PDF 文档中的页面导入 PDF/X 兼容输出文档时，有一些特殊规则（有关 PDF 导入库 PDI 的详细信息，参见第 110 页上的第 5.2 节“使用 PDI（PDF 导入库）生成 PDF 页”）。所有导入的文档都必须遵守可接受的 PDF/X 规范级别，具体参见表 7.7。作为一条通行规则，可以接受遵守与生成输出文档相同的 PDF/X 规范级别或同一级别的较低版本的输入文档。此外，也可接受某些其他组合。如果在 PDFlib 中配置了某个 PDF/X 规范级别，且导入文档遵守某个可接受的级别，则可以保证生成的输出文档遵守选定的 PDF/X 规范级别。不遵守某一可接受的 PDF/X 级别的导入文档将被拒绝。

表 7.7 各种 PDF/X 输出级别可以接受的 PDF/X 输入级别；必须避免采用其他组合。

PDF/X 输出级别	导入文档的 PDF/X 级别					
	PDF/X-1:2001	PDF/X-1a:2001	PDF/X-1a:2003	PDF/X-2:2003	PDF/X-3:2002	PDF/X-3:2003
PDF/X-1:2001	允许	允许				
PDF/X-1a:2001		允许				
PDF/X-1a:2003		允许	允许			
PDF/X-2:2003		允许	允许	允许	允许	允许
PDF/X-3:2002		允许			允许	
PDF/X-3:2003		允许	允许		允许	允许

如果要导入多个 PDF/X 文档，则必须根据相同的输出条件准备这些文档。尽管 PDFlib 能够纠正某些项目，但它并非为完全校验 PDF/X 而设计的，也不会强制导入文档完全的兼容 PDF/X。例如，PDFlib 不会嵌入导入 PDF 页面中缺失的字体，也不会对导入的页面进行任何颜色校正。

如果希望合并导入的页面，以使生成的 PDF 输出文档遵守与输入文档相同的 PDF/X 规范级别和输出条件，可查询导入 PDF 的 PDF/X 状态，具体如下：

```
pdfxlevel = oPDF.get_pdi_parameter("pdfx", doc, -1, 0)
```

如果遵守某个 ISO PDF/X 级别，此语句将检索指示导入文档 PDF/X 规范级别的字符串；否则，将返回 *none*。返回的字符串可用来适当地设置输出文档的 PDF/X 规范级别，具体使用 *begin\_document()* 中的 *pdfx* 选项。

除查询 PDF/X 规范级别之外，还可以复制导入文档中的 PDF/X 输出方法，具体如下：

```
ret = oPDF.process_pdi(doc, -1, "action copyoutputintent")
```

*load\_iccprofile()* 设置输出方法的一种替代方式，并可以将导入文档的输出方法复制到生成的输入文档中，而不考虑输出方法是用标准名称还是 ICC 配置文件定义的。生成输出文档的输出方法必须一次设置完成，可以将复制导入文档的输出方法，也可以使用 *usage* 选项设置为 *outputintent* 的 *load\_iccprofile()* 来明确设置输出方法。

## 7.5 标签 PDF

标签 PDF 是一种增强型 PDF，可支持 PDF 浏览器的其他工具，例如：辅助功能的支持、文本重排、可靠的文本提取以及将文档转换为 RTF 或 XML 等其他格式。

PDFlib 支持标签 PDF 的生成。但是，只有客户端在生成 PDF 输出时提供文档内部结构的相关信息并遵守某些规则的情况下，才能创建标签 PDF。

注：PDFlib 目前不支持自定义结构元素类型（即，只能使用 PDF 定义的标准结构类型）、角色映射以及结构元素属性。

7.5.1 用 PDFlib 生成标签 PDF

要求的操作 表 7.8 列出了生成标签 PDF 输出要求的所有操作。处于 PDF 模式下时，如果不调用要求的函数，将触发异常。

表 7.8 生成标签 PDF 必须应用的操作

项目	实现 PDF 兼容性的 PDFlib 函数和参数要求
标签 PDF 输出	<code>begin_document()</code> 中的 <code>tagged</code> 选项必须设为 <code>true</code> 。
文档语言	必须设置 <code>begin_document()</code> 中的 <code>lang</code> 选项，以指定文档的自然语言。最初，必须设置整个文档的自然语言，但此后可在任意结构级别上覆盖具体项目的设置。
结构信息	必须标识结构信息和伪像等。所有内容生成 API 函数都必须用 <code>begin_item()/end_item()</code> 对话框起来。

兼容 Unicode 的文本输出 生成标签 PDF 时，所有文本输出都必须使用兼容 Unicode 的字体；详细信息，参见第 8o 页上的第 4.5.6 节“Unicode 兼容的字体”。这表示，使用的所有字体都必须提供到 Unicode 的映射。只有通过 `begin_item()` 中的 `ActualText` 或 `Alt` 选项为内容提供替代文本的情况下，才允许使用不兼容 Unicode 的字体。如果生成标签 PDF 时使用了没有正确 Unicode 映射的文本，PDFlib 将引发异常。

注：有些情况下，PDFlib 无法检测到字体编码错误的问题，例如编码为文本字体的符号字体等。而且，由于历史问题，具有某些排版变化的 PostScript 字体（例如专业字体）也可能导致输出无法访问。

页面内容顺序 定义页面内容的文本、图形和页面运算符的顺序称为内容流顺序；由逻辑结构树定义的内容顺序称为逻辑顺序。生成标签 PDF 要求客户端遵守某些内容顺序方面的规则。

推荐的、也是很自然的方法是，依次生成结构元素的所有构成部分，然后移至下一个元素。从技术角度讲，应以深度优先遍历创建结构树。

以下方法应避免：输出第一个元素的一部分、切换到下一元素的一部分，然后在返回第一个元素，等等。如果采用这种方法，结构树是在多次遍历中建立的，且每次遍历只生成元素的一部分。

用 PDI 导入页面 由于导入文档的结构会干扰生成的结构，所以不能在标签 PDF 模式下导入标签 PDF 文档或其他包含结构信息的 PDF 文档页面。

但是，可导入非结构化文档中的页面。请注意，除非使用适当的 `ActualText` 添加了标签，否则 Acrobat 的辅助功能会“按原样”处理这些页面。

伪像 不属于作者原始内容的一部分的图形或文本对象称为伪像。伪像应使用 `Artifact` 伪标签等加以标识，可分为以下几个类别：

- ▶ 分页：自动标题和页码等功能
- ▶ 版面：标尺和表格阴影等排版或设计元素
- ▶ 页面：裁切标记和颜色条等印刷辅助功能

尽管没有严格规定，但我们强烈建议标识伪像，以帮助实现文本重排和辅助功能。

内嵌项目 PDF 定义了块级结构元素 (BLSE) 和内嵌级结构元素 (ILSE)；有关确切定义，参见表 8.57。BLSE 可包含其他 BLSE 或实际内容，而 ILSE 始终只包含内容。此外，PDFlib 还可区分如下项目：

`ASpan` 项目属于常规项目还是内嵌项目的决定由客户端通过 `begin_item()` 的 `inline` 选项控制。建议将辅助功能范围强制设为常规项目 (`inline=false`)，例如：包含多种语言且被多个页

表 7.9 常规项目和内嵌项目

	常规项目	内嵌项目
受影响的项目	所有分组元素和 <i>BLSE</i>	所有 <i>ILSE</i> 和非结构标签（伪标签）
可更改常规 / 内嵌状态	否	仅用于 <i>ASpan</i> 项目
文档结构树的一部分	是	否
可跨越页面边界	是	否
可被其他项目中断	是	否
可暂停和启动	是	否
可以任意深度嵌套	是	只能与其他内嵌项目一起使用

面分隔的段落。此外，也可以关闭项目，在下一页上开始一个新项目。必须在打开项目的同一页面上关闭内嵌项目。

**可选操作** 表 7.10 列出了生成标签 PDF 输出时的所有可选操作。尽管没有严格规定，但这些功能将提升生成标签 PDF 输出的质量，因此建议使用。

表 7.10 生成标签 PDF 的可选操作

项目	实现标签 PDF 兼容性的可选 <i>PDFlib</i> 函数和参数
连字	断字符（在行尾将单字分为两部分）应使用自动连字符 ( <i>U+ooAo</i> ) 表示，而非强制连字符 ( <i>U+oo2D</i> )
字边界	即使定位方面没有严格规定，但还是应该使用空格 ( <i>U+oo20</i> ) 分隔单字。可使用 <i>autospace</i> 参数，在每次调用 <i>show</i> 函数后自动生成空格字符。
伪像	为了区分实际内容和页面伪像，应使用 <i>tag=Artifact</i> 的 <i>begin_item()</i> 等标识伪像。

**禁止的操作** 表 7.11 列出了在生成标签 PDF 输出时禁止的所有操作。处于标签 PDF 模式下时，调用禁止的函数将触发异常。

表 7.11 生成标签 PDF 时必须避免的操作

项目	实现标签 PDF 兼容性应避免的 <i>PDFlib</i> 函数和参数
不兼容 <i>Unicode</i> 的字体	必须避免使用不兼容 <i>Unicode</i> 的字体，具体参见第 8o 页上的第 4.5.6 节“ <i>Unicode</i> 兼容的字体”。
PDF 导入	不能导入包含结构信息的 <i>PDF</i> 文档中的页面（尤其是：标签 <i>PDF</i> 文档）。

7.5.2 用直接 Text Output 和 Textflows 创建标签 PDF

**最低标签 PDF 示例** 以下示例代码可创建一个非常简单的标签 PDF 文档。它的结构树只包含一个 *P* 元素。以下代码使用 *autospace* 功能，在文本片段之间自动生成空格字符：

```
ret = oPDF.begin_document("hello-tagged.pdf", "tagged=true lang=en")
If (ret = -1) Then
    MsgBox "Error:" & oPDF.get_errmsg
End
End If

' 在文本块之间自动创建空格
oPDF.set_parameter "autospace", "true"

' 作为文档结构根级的子级打开第一个结构元素 (=0)
```

```

id = oPDF.begin_item("P", "Title = {Simple Paragraph}")

oPDF.begin_page_ext 0, 0, "width=a4.width height=a4.height"
font = oPDF.load_font("Helvetica-Bold", "unicode", "")

oPDF.setfont font, 24
oPDF.show_xy "Hello, Tagged PDF!", 50, 700
oPDF.continue_text "This PDF has a very simple"
oPDF.continue_text "document structure."

oPDF.end_page_ext ""
oPDF.end_item id
oPDF.end_document ""

```

**用文本流生成标签 PDF** 文本流功能提供了强大的文本格式化功能（参见第 91 页上的第 4.9 节“多行 `textflow`”）。由于具体文本片段不再受客户端的直接控制，而将由 PDFlib 自动完成格式化，因此用文本流生成标签 PDF 时，必须格外小心：

- ▶ 文本流不能包含具体的结构元素，但文本流可以包含在结构元素之中。
- ▶ 文本流的所有部分（用特定文本流句柄执行的所有 `fit_textflow()` 调用）应包含在一个结构元素中。
- ▶ 由于文本流的各个部分可能分布在多个页面中，而这些页面可能包含其他结构项目，所以选择适当的父级项目时应特别注意（不要使用 `parent` 参数 -1，因为这可能指向错误的父级元素）。

### 7.5.3 启动复杂布局的项目

为了帮助创建具有复杂的非线性页面布局的结构信息，PDFlib 支持一种称为项目启动的功能。在开发人员必须跟踪多个结构分支，且每个分支都可能跨一个或多个页面的情况下，可使用这种功能启动以前创建的结构元素。这种技术通常可用于以下情况：

- ▶ 一个页面上的多个栏
- ▶ 小结或插入等中断主文本的插入操作
- ▶ 放置在两栏之间的表格和插图

通过启动功能，可在两个逻辑分支之间切换以及其他情况下使用经过改进的页面内容生成方法。这种方法比依次完成每个分支更加高效。我们以图 7.1 中显示的页面布局为例，说明启动功能的优点。该布局包含两个主文本栏，文本栏被黑色背景显示的框中的一个表格和插入注释以及页眉和页脚中断。

**按逻辑顺序生成页面内容** 从逻辑结构的角度讲，应按照以下顺序创建页面内容：左栏、右栏（页面右下角部分）、表格、插入、页眉和页脚。以下伪代码便采用了这一顺序：

' 按逻辑结构顺序创建页面布局

```

id_art = oPDF.begin_item("Art", "Title = Article")

id_sect1 = oPDF.begin_item("Sect", "Title = {First Section}")
' 1 创建左栏的上半部分
oPDF.set_text_pos x1_left, y1_left_top
...
' 2 创建左栏的下半部分
oPDF.set_text_pos x1_left, y1_left_bottom
...
' 3 创建右栏的上半部分

```

```

        oPDF.set_text_pos x1_right, y1_right_top
        ...
oPDF.end_item id_sect1

id_sect2 = oPDF.begin_item("Sect", "Title = {Second Section}")
' 4 创建右栏的下半部分
oPDF.set_text_pos x2_right, y2_right
...
' 可在下一页继续第二部分
oPDF.end_item id_sect2

id_table = oPDF.begin_item("Table", "Title = Table parent=" & id_art)
' 5 创建表格结构和内容
oPDF.set_text_pos x_start_table, y_start_table
...
oPDF.end_item id_table

id_insert = oPDF.begin_item("P", "Title = Insert parent=" & id_art)
' 6 创建插入结构和内容
oPDF.set_text_pos x_start_table, y_start_table
...
oPDF.end_item id_insert

id_artifact = oPDF.begin_item("Artifact", "")
' 7+8 创建页眉和页脚
oPDF.set_text_pos x_header, y_header
...
oPDF.set_text_pos x_footer, y_footer
...
oPDF.end_item id_artifact

' 文章可在下一页中继续
...
oPDF.end_item id_art

```



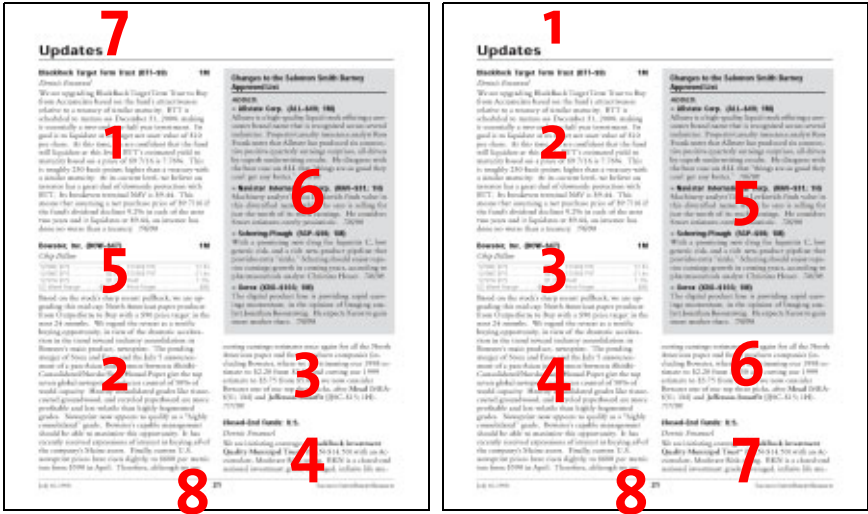


图 7.1  
按照逻辑结构顺序（左侧）和显示顺序（右侧）创建复杂页面布局。在继续创建片段 4 和 6 之前，右侧示例使用项目启动功能创建第一部分。

按照显示顺序创建页面内容 采用“逻辑顺序”的方法会强制创建程序按逻辑顺序构建页面内容；即使按显示顺序创建会更加容易的情况下，也是如此：页眉、左栏上半部分、左栏下半部分、插入、右栏、页脚。使用 `activate_item()` 时，可按如下所示实施此顺序：

' 按照显示顺序创建页面布局

```
id_header = oPDF.begin_item("Artifact", "")
    ' 1 创建页眉
    oPDF.set_text_pos x_header, y_header
    ...
oPDF.end_item id_header

id_art = oPDF.begin_item("Art", "Title = Article")

id_sect1 = oPDF.begin_item("Sect", "Title = {First Section}")
    ' 2 创建左栏的上半部分
    oPDF.set_text_pos x1_left, y1_left_top
    ...

id_table = oPDF.begin_item("Table", "Title = Table parent=" & id_art)
    ' 3 创建表格结构和内容
    oPDF.set_text_pos x_start_table, y_start_table
    ...
oPDF.end_item id_table

' 继续第一部分
oPDF.activate_item id_sect1
    ' 4 创建左栏的下半部分
    oPDF.set_text_pos x1_left, y1_left_bottom
    ...

id_insert = oPDF.begin_item("P", "Title = Insert parent=" & id_art)
    ' 5 创建插入结构和内容
    oPDF.set_text_pos x_start_table, y_start_table
```

```

    ...
oPDF.end_item id_insert

' 继续创建第一部分的其他内容
oPDF.activate_item id_sect1
    ' 6 创建右栏的上半部分
    oPDF.set_text_pos x1_right, y1_right_top
    ...
oPDF.end_item id_sect1

id_sect2 = oPDF.begin_item("Sect", "Title = {Second Section}")
    ' 7 创建右栏的下半部分
    oPDF.set_text_pos x2_right, y2_right
    ...
    ' 可在下一页继续第二部分
oPDF.end_item id_sect2

id_footer = oPDF.begin_item("Artifact", "")
    ' 8 创建页脚
    oPDF.set_text_pos x_footer, y_footer
    ...
oPDF.end_item id_footer

' 文章可在下一页中继续
...
oPDF.end_item id_art

```

采用这种结构元素的顺序时，主文本（跨一个半栏）会被表格和插入内容中断两次。因此，必须使用 *activate\_item()* 启动两次。

如果内容跨多个页面，可以采用相同的技术。例如，可以首先创建页眉或其他插入内容，然后再次启动主页面内容元素。

## 7.5.4 在 Acrobat 中使用标签 PDF

本节介绍我们在 Adobe Acrobat 6.0 中对标签 PDF 输出进行测试期间的发现，主要与 Acrobat 中的错误和异常行为有关。同时，还提出了我们发现的变通方法。

**Acrobat 的重排功能** Acrobat 允许标签 PDF 文档进行重排，即：调整页面内容以适应当前窗口大小。在测试标签 PDF 的过程中，我们在重排功能方面有如下发现：

- ▶ 页面内容的顺序应遵守所需的重排顺序。
- ▶ 符号（非 Unicode 字体）可能导致 Acrobat 的重排功能崩溃。因此，我们建议将文本置于插图元素中。
- ▶ BLSE 可能同时包含子级结构元素和直接内容元素。为使重排功能（以及辅助功能检查器和朗读功能）正常运行，我们建议将直接元素置于第一个子级元素之前。
- ▶ 应为表格和插图提供 BBox 选项。BBox 应准确无误；但是，对于表格，只需准确设置左下角即可。除提供 BBox 条目之外，也可在 BLSE 标签内创建图形，例如 *P*、*H* 等。但是，启动重排时，将不显示矢量图形。如果客户端未提供 BBox 选项（而依靠自动 BBox 生成），应在表格元素之外绘制单元格边框等所有表格图形。
- ▶ 表格元素的子级元素只应包含与表格相关的元素（*TR*、*TD*、*TH*、*THHead*、*TBody* 等），不能包含任何其他元素。例如，尽管能生成正确的标签 PDF，但在表格内使用标题元素可能会导致重排问题。
- ▶ 私有标签包含的内容不能导出为其他格式。但是，它们要受重排和朗读功能的影响，因此私有标签内的插图必须具有替代文本。

- ▶ 导入图像应包含在插图元素内，导入的 PDF 页面应包含在表单项目内。为了防止重排问题，应避免使用公式项目类型。
- ▶ 重排功能似乎无法正确处理用 *topdown* 选项生成的 PDF 文档。
- ▶ 应避免使用具有混合子级类型的结构项目（例如，同时具有页面内容序列和非内嵌结构元素），否则可能导致重排失败。
- ▶ 如果启动项目只包含内容，不包含结构子级，重排可能会失败；尤其是在另一页面上启动项目时。可在非内嵌范围标签内包装启动项目，以避免此问题。

**Acrobat 的辅助功能检查器** Acrobat 的辅助功能检查器可用来确定标签 PDF 文档是否适用于读屏器等辅助技术。

- ▶ 包含导入图像的元素应使用 *Alt* 属性。*ActualText* 属性可能会导致辅助功能检查器崩溃。使用 *Alt* 而不使用 *ActualText* 的另一个原因是，朗读功能会捕捉真实文本。
- ▶ 如果导入 PDF 页面的第一个项目是表单标签，则可导致辅助功能检查器出现问题。
- ▶ 如果在 *TOCI* 标签内设置 *Lbl* 标签（如《PDF Reference》中所述），辅助功能检查器将发出警告，指出未在 *LI* 标签内设置 *Lbl* 标签。

**用 Acrobat 导出为其他格式** 标签 PDF 可显著提高 Acrobat 将 PDF 文档导出为其他格式的效果。

- ▶ 如果导入 PDF 页面具有表单标签，在 Acrobat 中，*ActualText* 选项内提供的文本将导出为其他格式，而 *Alt* 标签中提供的文本将被忽略。但是，朗读功能可用于以上两种文本。
- ▶ 包含导入图像的元素应使用 *Alt* 属性，而非 *ActualText* 属性，以使导出功能可以捕捉真实文本。
- ▶ *NonStruct* 标签的内容不会导出到 HTML 4.01 CSS 1.0 格式（但是可用于 HTML 3.2 导出）。
- ▶ 应为 ILSE 提供替代文本（例如 *Code*、*Quote* 或 *Reference*）。如果使用了 *Alt* 选项，朗读功能将会读出提供的文本，但导出为其他格式的将是实际的内容。如果使用 *ActualText* 选项，读出和导出的都是提供的文本。

**Acrobat 的朗读功能** 标签 PDF 可增强 Acrobat 的文本朗读功能。

- ▶ 提供 *Alt* 或 *ActualText* 时，最好在开头包括一个空格字符。这样，朗读功能便可将文本与前一句区分开来。由于同样的原因，最好同时在结尾处包括一个句点“.”符号。否则，朗读功能将尝试组合前一句的最后一个单字和替代文本的第一个单字



# 8 用于 PDFlib、PDI 和 PPS 的 API 参考

API 参考记录了所有支持的 PDFlib、PDI (PDF Import) 和 PPS (PDFlib Personalization Server) 函数。

## 8.1 数据类型

### 8.1.1 PDFlib 数据类型

通常，所有支持 COM- 或 .NET 的开发环境都能以相同的方式访问 PDFlib 例程。有些情况下，您应注意 PDFlib 函数参数使用的不同数据类型。尤其是，字符串和二进制数据使用的数据类型可能会略有区别。表 8.1 列出了有关这些数据类型的有用信息。

表 8.1 语言绑定中的数据类型

开发环境	字符串数据类型	二进制数据类型
通用 COM	BSTR （字符串）	VT_ARRAY   VT_Uh 类型的变体 <sup>1</sup>
具有 COM 的 Delphi	字符串 （用于 8 位编码）或 WideString （用于 Unicode）	OleVariant
.NET	字符串	byte[ ]
REALbasic	字符串	MemoryBlock

1. 换言之，指由未签名字节的组成的变体数组。

**Unicode 字符串** PDFlib 可在所有相关区域接受 Unicode 字符串，并支持各种与 Unicode 相关的格式和设置。有关详细信息，请查阅第 77 页上的第 4.5.2 节“内容字符串、超文本字符串和名称字符串”；同时，请注意本章将使用的下列字符串类型：

- ▶ 内容字符串
- ▶ 超文本字符串
- ▶ 名称字符串

### 8.1.2 选项列表

选项列表是种强大且简便的 PDFlib 操作控制方法。许多 PDFlib API 方法都支持选项列表（简称 **optlists**），而不要求大量的函数参数。这些选项列表是字符串，可以包含任意数量的选项。选项列表支持各种数据类型以及数组等复合数据。在大多数语言中，可通过连接所需关键字和值来简便地构建 **optlists**。一个 **optlist** 是一个字符串，包含一对或多对表单

key value

可用任意空白字符（例如空格、制表符、回车符、换行符等）分隔名称和值，以及多个名称 / 值对。值可以是多个值组成的列表。也可以在名称和值之间使用等号 '='：

key=value

如果值是个包含空白字符或等号的字符串，则必须用括号将该字符串括起来：

key={multiple words}

key={value=containing=equal=signs}

若选项列表带有子选项，则需另加一对括号将子选项括起，即使子选项只有一个元素：

```
key={{value}}
```

由于选项列表将由左至右计算，可以在同一列表内多次使用同一选项。这时，最后一次出现的选项将覆盖前面的选项。在下面的示例中，第一个选项赋值将被第二个覆盖；该选项列表处理完后，键值将为 2：

```
key=1 key=2
```

选项列表支持表 8.2 所列的数据类型。本章下文将更加详细地讨论这些数据类型。

表 8.2 选项列表中的数据类型

种类	数据类型
简单	布尔型、字符串、内容 / 超文本 / 名称字符串、单字符、关键字、浮点型、整型、句柄
复杂	列表、矩形、动作列表、颜色

简单值 简单值可以使用下列任一数据类型：

- ▶ 布尔型： *true* 或 *false* ；如果省略布尔选项的值，则默认值为 *true*。作为简略表示法，可使用 *noname* 替代 *name=false*。
- ▶ 字符串：这些字符串是 ASCII 纯文本字符串，通常用于不可本地化的关键字。包含空白字符或 = 的字符串必须用 { 和 } 括起来。可以使用 {} 构建空字符串。如果字符 { 和 } 是字符串的一部分，则必须在前面添加 \ 字符。
- ▶ 内容字符串、超文本字符串和名称字符串：这些字符串可容纳各种格式的 Unicode 内容；有关详细信息，请参见第 77 页上的第 4.5.2 节“内容字符串、超文本字符串和名称字符串”。
- ▶ 单字符：这些字符串是单个的 Unicode 字符，可支持多种语法变体：10 进制数值（例如 173），带 x、X、0x、0X 或 U+（xAD、0xAD、U+00AD）前缀的 16 进制数值，或者第 79 页上的第 4.5.5 节“字符引用”介绍的不带 ‘&’ 和 ‘;’ 修饰符（shy、#xAD、#173）的数字或字符引用。单字符必须处于 0-65535（0-xFFFF）的范围内。
- ▶ 关键字：预定义固定关键字列表中的一项。
- ▶ 浮点型和整型：10 进制浮点小数或整数；可使用小数点和逗号作为浮点值的小数分隔符。整型值可以 x、X、0x 或 0X 开头，指定 16 进制值。有些选项可直接在值后添加 % 字符，以支持百分比；相关文档中有具体说明。
- ▶ 句柄：包括几种 PDFlib 内部句柄，例如：字体、图像或动作句柄。从技术角度讲，这些值是整型值。

根据选项的类型和解释，可能还有其他限制。例如，整型或浮点型选项可能会限制在一定的值范围内；句柄必须对相应的对象类型有效，等等。第 8 章中相关函数的说明中具体规定了各个选项的条件。以下是一些简单值的示例（第一行显示包含空白字符的口令字符串）：

```
open_pdi(): password {secret string}
create_gstate(): linewidth 0.5 blendmode overlay opacityfill 0.75
load_font(): embedding=true subsetting=true subsetlimit=50 kerning=false
load_font(): embedding subsetting subsetlimit=50 nokerning
create_textflow(): leading=150%
create_textflow(): charmapping={ 0x0A 0x20 }
```

列表值 列表值由多个值组成，可以是简单值或依次排列的列表值。列表要用 { 和 } 括起来。以下是一些列表值的示例：

```
fit_image():           boxsize={500 600} position={50 0}
create_gstate():       dasharray={11 22 33}
create_annotation():   polylinelist={{10 20 30 40}}
```

矩形 矩形是个包含四个浮点值的列表，指定矩形左下角和右上角的坐标。根据不同的选项，解释矩形坐标的坐标系也不尽相同（如标准或用户坐标系），将单独论述。示例：

```
begin_document():      cropbox {0 0 500 600}
```

动作列表 动作列表可指定一个或多个动作。列表中的每一项都包括一个事件关键字（触发器）和一个动作句柄列表；该动作句柄列表必须用 `create_action()` 创建。各个动作将按照列表顺序执行。对于各个选项，将单独介绍允许的事件集（例如 `docopen`）和动作类型（例如 `JavaScript`）。示例（假定先前的 `create_action()` 调用已返回值 0、1 和 2）：

```
begin_document():      action {open 0}
create_bookmark():     action {activate {0 1 2}}
create_field():        action {keystroke 0 format 1 validate 2}
```

颜色 颜色选项是个列表，包含色彩空间关键字和具有不同数量的浮点值的列表（具体取决于特定色彩空间）。色彩空间关键字等同于 `setcolor()`（参见第 202 页上的第 8.5.1 节“设置颜色和色彩空间”）；其所有可能的值在第 50 页上的第 3.3.1 节“颜色和色彩空间”中给出说明：

- ▶ 色彩空间关键字 `gray`、`rgb`、和 `cmymk` 可能附带一个、三个或四个浮点值。
- ▶ 色彩空间关键字 `lab` 可能附带三个浮点值。
- ▶ 色彩空间关键字 `spot` 可能附带一个专色句柄和色调值。另外，色彩空间关键字 `spotname` 可附带一个专色名称和一个包含色调的浮点值。
- ▶ 色彩空间关键字 `iccbasedgray`、`iccbasedrgb` 和 `iccbasedcmymk` 可能附带一个、三个或四个浮点值。
- ▶ 色彩空间关键字 `pattern` 可通过一个 `pattern` 句柄来补充。
- ▶ 色彩空间关键字 `none` 可能用来指定色彩缺失的情况。

根据具体函数的详细说明，特定选项列表可能只提供上述关键字的一部分。以下是一些颜色值的示例：

```
fill_textblock():      strokecolor={ rgb 1 0 0 }
fill_textblock():      bordercolor=none
fill_textblock():      fillcolor={ spotname {PANTONE 281 U} 0.5 }
fill_textblock():      fillcolor={ spotname 1 0.8 }
```

## 8.2 常规函数

### 8.2.1 设置

表 8.3 列出了本节的相关参数及相应的值。

表 8.3 设置函数的参数和值

函数	关键字	说明
<code>set_parameter</code>	<code>compatibility</code>	不常用，使用 <code>begin_document()</code> 的 <code>compatibility</code> 选项。
<code>set_parameter</code>	<code>pdfx</code>	不常用，使用 <code>begin_document()</code> 的 <code>pdfx</code> 选项。
<code>set_parameter</code>	<code>SearchPath</code>	包含要读取文件的目录的相对或绝对路径名。可设置多个 <code>SearchPath</code> ；各项将累积，并首先使用最近设置的一项（参见第 42 页上的第 3.1.5 节“资源配置和文件搜索”）。范围：任意

表 8.3 设置函数的参数和值

函数	关键字	说明
<code>set_parameter</code>	<code>resourcefile</code>	<i>PDFlib UPR</i> 资源文件的相对或绝对文件名。将立即载入资源文件。将保留现有资源；如果再次设置，他们的值将由新值覆盖。范围：任意
<code>set_parameter</code>	<code>license</code>	设置 <i>PDFlib</i> 、 <i>PDFlib+PDI</i> 或 <i>PPS</i> 的许可密钥。可在第一次调用 <i>begin_document()</i> 之前设置密钥（甚至可以多次设置，以进行累积密钥）。范围：对象
<code>set_parameter</code>	<code>licensefile</code>	设置包含许可密钥的文件的文件名。该文件在第一次调用 <i>begin_document()</i> 之前被设置一次。范围：对象。
<code>set_value</code>	<code>compress</code>	设置压缩级别。该参数不会影响以通过 ( <i>pass-through</i> ) 模式处理的图像数据。默认值：6。范围：页面、文档 0 无压缩 1 最快速度 9 最佳压缩
<code>get_value</code>	<code>major, minor revision</code>	分别返回 <i>PDFlib</i> 的主要、次要或修订版本号。范围：任意、空。
<code>get_parameter</code>	<code>version</code>	返回完整的 <i>PDFlib</i> 版本字符串，格式为 <major>.<minor>.<revision>，可能附带 <i>beta</i> 、 <i>rc</i> 等附加限定符后缀。范围：任意、空。
<code>get_parameter</code>	<code>scope</code>	返回当前范围的名称（参见表 3.1）。范围：任意。
<code>set_parameter</code>	<code>trace</code>	（不支持）如果为 <i>true</i> ，将记录所有 <i>API</i> 函数调用。日志文件的内容可能对调试很有用， <i>PDFlib</i> 支持人员可能也需要这些内容。范围：任意。默认值： <i>false</i> 。
<code>set_parameter</code>	<code>tracefile</code>	（不支持）设置跟踪文件名。范围：任意，但必须在启用跟踪之前设置。默认值： <i>PDFlib.trace</i> 。
<code>set_parameter</code>	<code>logmsg</code>	（不支持）如果启用了日志，将提供的文本写入日志文件。这可能对调试客户端代码很有用。范围：任意。

8.2.2 文档和页面

表 8.4 列出了本节的相关参数及相应的值。

表 8.4 文档和页面函数的参数和值

函数	关键字	说明
<code>set_parameter</code>	<code>openwarning</code>	不常用，打开文档时使用 <i>get_errmsg()</i> 查找尝试失败的原因。
<code>set_value</code>	<code>pagewidth pageheight</code>	不常用，使用 <i>begin_page_ext()</i> 的 <i>width</i> 和 <i>height</i> 参数，或者使用 <i>begin_page_ext()</i> 或 <i>end_page_ext()</i> 的 <i>mediabox</i> 选项。
<code>set_parameter</code>	<code>topdown</code>	如果为 <i>true</i> ，将假定页面、图案或模板开头坐标系统的原点位于页面左上角， <i>y</i> 坐标值向下增加；否则，将使用默认坐标系统（参见第 46 页上的第 3.2.1 节“坐标系统”）。范围：文档。默认值： <i>false</i>
<code>get_value</code>	<code>pagewidth pageheight</code>	获取当前页面的页面大小（ <i>MediaBox</i> 的尺寸）。范围：页面、路径
<code>set_value</code>	作品框 <i>BleedBox</i> <i>CropBox</i> <i>TrimBox</i>	不常用，使用 <i>begin_page_ext()</i> 或 <i>end_page_ext()</i> 的 <i>artbox</i> 、 <i>bleedbox</i> 、 <i>cropbox</i> 和 <i>trimbox</i> 选项。
<code>set_parameter</code>	<code>userpassword masterpassword permissions</code>	不常用，使用 <i>begin_document()</i> 的 <i>userpassword/masterpassword/permissions</i> 选项。



VB RB

Function begin\_document(filename As String, optlist As String) As Long

C#

int begin\_document(String filename, String optlist)

根据各种选项创建新的 PDF 文件。

**filename** （名称字符串）将生成 PDF 输出文件的绝对或相对文件名。如果 *filename* 为空，则将在内存中生成 PDF 文档，而不创建文件，因而只能从客户端通过 *get\_buffer()* 函数来获取生成的 PDF 数据。在 Windows 上，可以使用 UNC 路径或映射网络驱动器。

**optlist** 选项列表，根据表 8.5 或表 8.6 指定文档选项。*end\_document()* 中指定的选项优先于 *begin\_document()* 中指定的相同选项。

返回值

-1 未产生错误时，返回 -1，否则为 1。如果 *filename* 为空，此函数始终都能成功运行，则不会返回错误值 -1。

详细说明

此函数使用提供的 *filename* 创建新 PDF 文件。PDFlib 将尝试使用给定文件名打开文件，并在 PDF 文档完成时关闭文件。

范围

对象；如果文件可以成功打开，此函数将为文档范围的起点；必须始终与对应的 *end\_document()* 调用成对使用。

绑定

ASP：应使用 *MapPath* 工具创建传递给此函数的完全路径名。

表 8.5 *begin\_document()* 的文档选项

选项	类型	描述
<i>compatibility</i>	关键字	将文档的 PDF 版本设为 Acrobat 4、5、6 或 7 的 »1.3«、»1.4«、»1.5« 或 »1.6« 字符串。有关详细信息，请参见第 141 页上的第 7.1 节 “Acrobat 和 PDF 版本”。如果使用 <i>pdfx</i> 参数，则忽略此选项。默认值：»1.5«
<i>groups</i>	字符串列表	定义文档中使用的页面组的名称和顺序
<i>inmemory</i>	布尔型	如果为 <i>true</i> ，并且 <i>linearize</i> 选项也为 <i>true</i> ，PDFlib 将不为线性化处理创建任何临时文件，而在内存中处理文件。这可能会大大提高某些系统上的性能（尤其是 MVS），但要求两倍于文档大小的内存。如果为 <i>false</i> ，将为线性化处理创建临时文件。默认值： <i>false</i>
<i>lang</i>	字符串	（如果 <i>tagged=true</i> ，则要求此选项）将文档的自然语言设为由两个字符组成的 ISO 639 语言代码（例如：DE、EN、FR、JA），后面可以跟一个连字符和由两个字符组成的 ISO 3166 国家 / 地区代码（例如：EN-US、EN-GB、ES-MX）。不区分大小写。  在初始化时，需为文档的语言规范统一赋值，随后可在结构树的各个级别上用新值覆盖。
<i>linearize</i>	布尔型	如果为 <i>true</i> ，将对输出文档进行线性化处理（参见第 144 页上的第 7.3 节 “网页优化的（线性化的）PDF”）。在 MVS 系统上，此选项不能用于将生成的 PDF 输出到内存（即，空 <i>filename</i> ）。默认值： <i>false</i>
<i>masterpassword</i>	字符串	文档的许可口令。如果此选项为空，将不应用许可口令。默认值：empty
<i>permissions</i>	关键字列表	输出文档的访问权限列表。此选项可包含任意数量的 <i>noprint</i> 、 <i>nomodify</i> 、 <i>nocopy</i> 、 <i>noannots</i> 、 <i>noassemble</i> 、 <i>noforms</i> 、 <i>noaccessible</i> 、 <i>nohiresprint</i> 和 <i>plainmetadata</i> 关键字（参见表 7.3）。默认值：empty
<i>pdfx</i>	关键字	设置 PDF/X 规范等级，例如 »PDF/X-1:2001«、»PDF/X-1a:2001«、»PDF/X-1a:2003«、»PDF/X-2:2003«、»PDF/X-3:2002«、»PDF/X-3:2003«，或 »none« 或（参见第 145 页上的第 7.4 节 “PDF/X”）。默认值：none
<i>tagged</i>	布尔型	（PDF 1.4 或更高版本）如果为 <i>true</i> ，则生成加标签的 PDF 输出。必须由客户端在加标签的 PDF 模式下，提供正确的结构信息（参见第 236 页上的第 8.10 节 “标签 PDF 的结构函数”）。默认值： <i>false</i>

表 8.5 *begin\_document()* 的文档选项

选项	类型	描述
<i>tempdirname</i>	字符串	在其中创建线性化处理所需临时文件的目录的名称。如果为空， <i>PDFlib</i> 将在当前目录内生成临时文件。如果提供了 <i>tempfilenames</i> 选项，则将忽略此选项。默认值: <i>empty</i>
<i>userpassword</i>	字符串	文档的文档打开口令。如果为空，则不应用文档打开口令。默认值: <i>empty</i>

**VB RB** **Sub end\_document(optlist As String)**

**C#** **void end\_document(String optlist)**

关闭生成的 PDF 文件并应用各种选项。

**optlist** 选项列表，根据表 8.6 指定文档选项。*end\_document()* 中指定的选项优先于 *begin\_document()* 中指定的相同选项。

**详细说明** 此函数完成 PDF 文档生成，释放所有文档相关的资源；如果已经用 *begin\_document()* 打开了文档，此函数还可关闭输出文件。无论使用何种方法打开 PDF 文档，客户端完成页面生成操作时都必须调用此函数。

在内存中生成文档时（而不在文件中），调用此函数后仍将保留文档缓冲区（因此，可使用 *get\_buffer()* 提取数据），并将在下次调用 *begin\_document()* 或者 *PDFlib* 对象在 中的超出范围时释放文档缓冲区。

**范围** 文档；此函数终结文档范围，应始终与 *PDF\_begin\_document()* 函数调用成对使用。

表 8.6 *begin\_document()* 和 *end\_document()* 的文档选项

选项	类型	描述
<i>action</i>	动作列表	（打开动作适用于 <i>PDF 1.3</i> 或更高版本，而其余动作只适用于 <i>PDF 1.4</i> 或更高版本。由于只能在调用 <i>begin_document()</i> 之后创建动作，所以此选项只适用于 <i>end_document()</i> ）下列一项或多项事件的文档动作列表（默认值：空列表）： <i>open</i> 打开文档时执行的动作。请注意，由于在 <i>Acrobat</i> 中的执行顺序，文档级 <i>JavaScript</i> 不能用于 <i>open</i> 动作。 <i>didprint</i> 打印文档后执行的 <i>JavaScript</i> 动作。 <i>didsave</i> 保存文档后执行的 <i>JavaScript</i> 动作。 <i>willclose</i> 关闭文档前执行的 <i>JavaScript</i> 动作。 <i>willprint</i> 打印文档前执行的 <i>JavaScript</i> 动作。 <i>willsave</i> 保存文档前执行的 <i>JavaScript</i> 动作。
<i>destination</i>	选项列表	选项列表，根据表 8.50 指定文档打开口令。打开口令将取代此选项，并将用作默认动作。
<i>labels</i>	选项列表的列表	包含根据表 8.8 提供的一个或多个选项列表的列表，指定符号页面名称。页面名称将在 <i>Acrobat</i> 的状态行中显示为页面标签（而非页码）。在一个文档中， <i>style/prefix/start</i> 值的组合必须是唯一的。默认值: <i>none</i>
<i>moddate</i>	布尔型	如果为 <i>true</i> ，将创建 <i>ModDate</i> （修改日期）文档信息关键字，以符合某些 <i>preflight</i> 工具的要求。默认值: <i>false</i>
<i>openmode</i>	关键字	设置文档打开时的外观。默认值: 如果文档包含任何书签，则为 <i>bookmarks</i> ；否则为 <i>none</i> ： <i>none</i> 打开时不显示任何其他面板。 <i>bookmarks</i> 打开时显示书签面板。 <i>thumbnails</i> 打开时显示缩览图面板 <i>fullscreen</i> 以全屏模式打开（不适用于浏览器）。 <i>layers</i> ( <i>PDF 1.5</i> ) 打开时显示图层面板。

表 8.6 `begin_document()` 和 `end_document()` 的文档选项（续）

选项	类型	描述
<code>pagelayout</code>	关键字	打开文档时使用的页面版面（默认值: <code>singlepage</code> ）： <code>singlepage</code> 一次显示一页。 <code>onecolumn</code> 在一列中连续显示多页。 <code>twocolumnleft</code> 在两列中显示多页，奇数页在左侧。 <code>twocolumnright</code> 在两列中显示多页，奇数页在右侧。
<code>uri</code>	字符串	设置文档的基本 URL。将具有指向其他文档的相对 Web 链接的文档移至不同位置时，此选项非常有用。设置到 »old« 位置的基本 URL 可确保相对链接仍然可以使用。默认值: <code>none</code>
<code>viewerpreferences</code>	选项列表	选项列表，根据表 VB RB 指定各种查看器首选项。默认值: <code>empty</code>
<code>metadata</code>	选项列表	(PDF 1.4) 提供文档的元数据。PDFlib 不同步元数据和文档信息域。此选项列表可以包含下列选项： <code>inputencoding (keyword)</code> 用来解释提供的数据的编码方式。默认值: <code>unicode</code> <code>inputformat (keyword)</code> 提供的数据的格式。默认值: <code>utf8</code> ；但是，如果 <code>inputencoding</code> 为 8 位编码方式，则默认为 <code>bytes</code> 。 <code>filename (name string, required)</code> 包含元数据的基于磁盘的文件或虚拟文件的文件名。此文件必须包含格式正确的 XMP 元数据，无需压缩即可将其复制到输出中。PDFlib 将自动生成 XDP 包头和包尾。 <code>outputformat (keyword)</code> 数据写入 PDF 输出时采用的格式（输出编码始终为 <code>unicode</code> ）。可能的值包括 <code>utf8</code> 、 <code>utf16be</code> 和 <code>utf16le</code> 。默认值: 如果 <code>inputformat=bytes</code> ，为 <code>utf8</code> ；否则为 <code>inputformat</code>

表 8.7 `begin_document()` 和 `end_document()` 中 `viewerpreference` 选项的子选项

选项	类型	描述
<code>centerwindow</code>	布尔型	指定是否将文档窗口置于屏幕中央。默认值: <code>false</code>
<code>direction</code>	关键字	阅读文档的顺序，将影响双页面视图中的滚动顺序。（默认为 <code>l2r</code> ）： <code>l2r</code> 从左至右 <code>r2l</code> 从右至左（包括直排书写系统）
<code>displaydoctitle</code>	布尔型	指定是否在 Acrobat 的标题栏 ( <code>true</code> ) 或是文件名 ( <code>false</code> ) 中显示 Title 文档信息域。默认值: <code>false</code>
<code>fitwindow</code>	布尔型	指定是否将文档窗口的大小调整为第一页的大小。默认值: <code>false</code>
<code>hidemenubar</code>	布尔型	指定是否隐藏 Acrobat 菜单栏。默认值: <code>false</code>
<code>hidetoolbar</code>	布尔型	指定是否隐藏 Acrobat 工具栏。在浏览器中查看 PDF 时，Acrobat 会忽略此设置。默认值: <code>false</code>
<code>hidewindowui</code>	布尔型	指定是否隐藏 Acrobat 的窗口控件。默认值: <code>false</code>
<code>nonfullscreenpagemode</code>	关键字	（仅用于 <code>openmode</code> 选项设为 <code>fullscreen</code> 的情况）指定如何在现有全屏模式下显示文档（默认值: <code>none</code> ）： <code>bookmarks</code> 显示页面和书签窗格 <code>thumbnails</code> 显示页面和缩览图窗格 <code>layers</code> 显示页面和图层窗格 <code>none</code> 只显示页面

表 8.7 `begin_document()` 和 `end_document()` 中 `viewerpreference` 选项的子选项（续）

选项	类型	描述
<code>viewarea</code> <code>viewclip</code> <code>printarea</code> <code>printclip</code>	关键字	在屏幕上查看或打印文档时，代表要显示或剪贴的页面区域的页面边界框类型。 <i>Acrobat</i> 会忽略此设置，但可能对其他应用程序很有用（默认值： <code>crop</code> ）： <code>art</code> 使用作品框 <code>bleed</code> 使用出血框 <code>crop</code> 使用裁剪框 <code>media</code> 使用媒体框 <code>trim</code> 使用裁切框  <i>PDF/X</i> ：不允许除 <code>media</code> 或 <code>bleed</code> 之外的值。

表 8.8 `begin/end_document()` 中 `labels` 选项和 `begin/end_page_ext()` 中 `label` 选项的子选项

选项	类型	描述
<code>group</code>	字符串	（仅用于 <code>begin_document()</code> ；如果文档使用页面组，则要求此选项；否则，不允许使用此选项。）标签将应用于指定组以及应用新标签之前的所有后续组中的所有页面。
<code>pagenumber</code>	整型	（仅用于 <code>end_document()</code> ；如果文档不使用页面组，则要求此选项，否则不允许使用此选项）标签将应用于指定页面以及应用新标签之前的所有页面。
<code>prefix</code>	超文本字符串	处于该范围内的所有标签的标签前缀。默认值： <code>none</code>
<code>start</code>	整型 $\geq 1$	该范围内第一个标签的数值。该范围内的后续页面将从该值开始，依次编号。默认值： <code>1</code>
<code>style</code>	关键字	要使用的编号样式（默认值： <code>none</code> ）： <code>none</code> 无页码；标签将只包含前缀。 <code>D</code> 十进制阿拉伯数字（ <code>1</code> 、 <code>2</code> 、 <code>3</code> 、...） <code>R</code> 大写罗马数字（ <code>I</code> 、 <code>II</code> 、 <code>III</code> 、...） <code>r</code> 小写罗马数字（ <code>i</code> 、 <code>ii</code> 、 <code>iii</code> 、...） <code>A</code> 大写字母（ <code>A</code> 、 <code>B</code> 、 <code>C</code> 、...、 <code>AA</code> 、 <code>BB</code> 、 <code>CC</code> 、...） <code>a</code> 小写字母（ <code>a</code> 、 <code>b</code> 、 <code>c</code> 、...、 <code>aa</code> 、 <code>bb</code> 、 <code>cc</code> 、...）

**VB RB** Function `get_buffer()`

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

获取 PDF 输出缓存区的内容。

**返回值** 充满二进制 PDF 数据的缓存区，供客户端使用。大多数 COM 客户端都使用 `Variant` 类型存放缓存内容。调用任何其他 `PDFlib` 函数之前，返回的缓存区必须供客户端使用。如果希望在调用其他 `PDFlib` 函数时使用，切记要复制数据（尤其是在调用 `create_pvf()` 创建包含该数据的 PVF 文件之前）。

**详细说明** 提取全部或部分包含生成 PDF 数据的缓存区。如果此函数在页面描述之间调用，它将返回迄今为止生成的 PDF 数据。如果在内存中生成 PDF，此函数至少须在 `end_document()` 之后调用，并将返回 PDF 文档的剩余部分。可以在此之前调用此函数，以提取部分文档数据。如果此函数只在 `end_document()` 之后调用一次，则返回的缓存区肯定会在连续缓存区内包含整个 PDF 文档。

由于 PDF 输出包含二进制字符，客户端软件必须准备接受包括空值在内的不可打印字符。

范围 对象、文档（换言之：在 `end_page_ext()` 之后和 `begin_page_ext()` 之前，或者在 `end_document()` 之后和 `delete()` 之前。此函数只能在已确保向 `begin_document()` 提供了空的 `filename` 的情况下，才只能使用。

如果 `begin_document()` 中的 `linearize` 选项已设为 `true`，则范围将限制在对象内；即：只 能在 `end_document()` 之后调用此函数。

VB RB

Sub begin\_page\_ext(width As Double, height As Double, optlist As String)

C#

void begin\_page\_ext(double width, double height, String optlist)

向文档添加一个新页面并指定各种选项。

**width, height** `width` 和 `height` 参数以点数为单位指定新页面的尺寸。可用具有相同名称的选项覆盖这两个参数（这种情况下，可为参数使用虚值 `o`）。表 3.4 列出了常用的页面格式。有关更多详细信息（`width` 和 `height` 选项），另请参见表 8.10。

**optlist** 根据表 8.9 和表 8.10 提供的选项列表。这些选项的优先级低于 `end_page_ext()` 中指定的相同选项。

详细说明 此函数将新页面的所有文本、图形和颜色状态参数重设为他们的默认值。

范围 文档；此参数是页面范围的开头，必须与对应的 `end_page_ext()` 调用成对使用。

参数 使用此函数时，将忽略下列不常用的参数：`pagewidth`、`pageheight`、`ArtBox`、`BleedBox`、`CropBox`、`TrimBox`。

表 8.9 `begin_page_ext()` 的选项

选项	类型	描述
<code>group</code>	字符串	（如果文档使用页面组，则要求此选项；否则，不允许使用此选项。）页面所属页面组的名称。此名称可使页面保持在一个页面组内，并可用 <code>resume_page()</code> 确定页面地址。
<code>pagenumber</code>	整型	如果用值 <code>n</code> 指定此选项，则将在 <code>group</code> 选项指定的页面组内现有第 <code>n</code> 页前插入页面（如果文档不使用页面组，则在文档中插入页面）。如果未指定此选项，将在页面组末尾插入页面。
<code>separationinfo</code>	选项列表	包含当前页面的详细分色信息的选项列表。 <code>Acrobat</code> 中将忽略此选项，但第三方软件可能会用它在预分色工作流程中确定并正确预览分色页： <code>pages</code> （整型；一组分色页的第一个页面要求此选项，但同一组的后续页面不允许使用此选项）属于同一组分色页的页面数量；分色页包含单个复合页的颜色数据。同一组内的所有页面必须在文件中依次显示。 <code>spotname</code> （字符串；除非提供了 <code>spotcolor</code> ，否则要求使用此选项）当前页面的色料名称。 <code>spotcolor</code> （专色句柄）描述当前页面色料的颜色句柄。
<code>topdown</code>	布尔型	如果为 <code>true</code> ，将假定页面开头坐标系统的原点位于页面左上角， <code>y</code> 坐标值向下增加；否则，将使用默认坐标系统（参见第 46 页上的第 3.2.1 节“坐标系统”）。默认值： <code>false</code>

**VB RB** Sub end\_page\_ext(optlist As String)  
**C#** void end\_page\_ext(String optlist)

完成页面并应用各种选项。

**optlist** 根据表 8.10 提供的选项列表。*end\_page\_ext()* 中指定的选项优先于 *begin\_page\_ext()* 中指定的相同选项。

范围 页面；此函数终结页面范围，必须与对应的 *begin\_page\_ext()* 调用成对使用。

表 8.10 *begin\_page\_ext()* 和 *end\_page\_ext()* 的选项

选项	类型	描述
<i>action</i>	动作列表	下列一个或多个事件的页面动作列表。默认值： <i>empty list</i> ： <i>open</i> 打开页面时执行的动作。 <i>close</i> 关闭页面时执行的动作。
<i>artbox</i> <i>bleedbox</i> <i>cropbox</i> <i>mediabox</i> <i>trimbox</i>	矩形	（如果 <i>topdown</i> 选项或参数为 <i>true</i> ，不允许使用 <i>mediabox</i> 选项）更改当前页面的页面框参数。各个框的坐标在默认坐标系统中指定（有关详细信息，请参见第 48 页上的第 3.2.2 节“页面大小和坐标限制”）。默认情况下，将只使用 <i>width</i> 和 <i>height</i> 参数创建媒体框 ( <i>MediaBox</i> )。 <i>mediabox</i> 选项将覆盖 <i>width</i> 和 <i>height</i> 选项和参数。
<i>defaultgray</i> <i>defaultrgb</i> <i>defaultcmymk</i>	icc 句柄	根据提供的配置文件句柄，设置页面的默认灰色、 <i>RGB</i> 或 <i>CMYK</i> 色彩空间。
<i>duration</i>	浮点型	如果 <i>openmode=fullscreen</i> ，则以秒为单位，设置当前页面的显示时间（参见表 8.6）。默认值： <i>1</i>
<i>label</i>	选项列表	根据表 8.8 提供的选项列表，指定符号页面名称。页面名称将在 <i>Acrobat</i> 的状态行中显示为页面标签（而非页码）。指定的编号方案将用于当前和后续页面，直至再次更改为止。在一个文档中， <i>style/prefix/start</i> 值的组合必须是唯一的。
<i>metadata</i>	选项列表	( <i>PDF 1.4</i> ) 提供页面的元数据。有关详细信息，请参阅表 8.6 中对 <i>metadata</i> 选项的描述。
<i>rotate</i>	整型	页面旋转值。旋转将影响页面显示，但不会修改坐标系统。可能的值包括 <i>0</i> 、 <i>90</i> 、 <i>180</i> 、 <i>270</i> 。默认值： <i>0</i>
<i>transition</i>	关键字	如果 <i>openmode=fullscreen</i> ，设置当前页面的页面转换，以实现特殊效果；在 <i>Acrobat</i> 的全屏模式下，以演示文稿的形式显示 <i>PDF</i> 时，这个选项会很有用（参见表 8.6）。默认值： <i>replace</i> <i>split</i> 两行在屏幕上掠过，显示整个页面 <i>blinds</i> 多行在屏幕上掠过，显示整个页面 <i>box</i> 在框中显示页面 <i>wipe</i> 单行掠过屏幕，显示整个页面 <i>dissolve</i> 旧页面分解，逐渐显示页面 <i>glitter</i> 分解效果从屏幕的一个边缘移动到另一边缘 <i>replace</i> 简单地用新页面替代旧页面 <i>fly</i> ( <i>PDF 1.5</i> ) 新页面飞入旧页面 <i>push</i> ( <i>PDF 1.5</i> ) 新页面将旧页面推出屏幕 <i>cover</i> ( <i>PDF 1.5</i> ) 新页面滑入屏幕，覆盖旧页面 <i>uncover</i> ( <i>PDF 1.5</i> ) 旧页面滑出屏幕，显示新页面 <i>fade</i> ( <i>PDF 1.5</i> ) 在旧页面中逐渐显示新页面

表 8.10 *begin\_page\_ext()* 和 *end\_page\_ext()* 的选项（续）

选项	类型	描述
<i>width</i> <i>height</i>	浮点型或关键字	（如果 <i>topdown</i> 选项或参数为 <i>true</i> ，则不允许此选项）新页面的尺寸，以点数为单位。第 48 页上的第 3.2.2 节“页面大小和坐标限制”介绍了 <i>Acrobat</i> 的页面大小限制。要生成横向页面，应使用 <i>width &gt; height</i> 或 <i>rotate</i> 选项。 <i>PDFlib</i> 使用 <i>width</i> 和 <i>height</i> 选项构建页面的媒体框，但也可以使用 <i>mediabox</i> 选项显式设置媒体框。 <i>width</i> 和 <i>height</i> 选项将覆盖名称相同的参数。  可在下列符号页面大小名称后附加 <i>.width</i> 或 <i>.height</i> ，将其用作关键字，例如： <i>a4.width</i> 、 <i>a4.height</i> 。有关下列数字值的信息，请参见表 3.4： <i>a0</i> 、 <i>a1</i> 、 <i>a2</i> 、 <i>a3</i> 、 <i>a4</i> 、 <i>a5</i> 、 <i>a6</i> 、 <i>b5</i> 、 <i>letter</i> 、 <i>legal</i> 、 <i>ledger</i> 或 <i>11x17</i>

**VB RB** Sub suspend\_page(optlist As String)

**C#** void suspend\_page(String optlist)

暂停当前页面，以便可以在稍后恢复。

**optlist** 供将来使用的选项列表。

将在内部保存当前页面的完全状态（图形、颜色、文本等）。稍后可使用 *resume\_page()* 恢复页面，以添加更多内容。必须先恢复暂停的页面，然后才能将其关闭。

范围 页面；此参数是文档范围的开头，必须与对应的 *resume\_page()* 调用成对使用。不能在标签 PDF 模式下使用此函数。

**VB RB** Sub resume\_page(optlist As String)

**C#** void resume\_page(String optlist)

恢复页面，以向其添加更多内容。

**optlist** 根据表 8.11 提供的选项列表。

必须已经使用 *suspend\_page()* 暂停页面。页面将再次打开；这样，便能够添加更多内容。关闭之前，必须先恢复所有已暂停的页面；即使未添加更多内容，也是如此。

范围 文档；此参数是页面范围的开头，必须与对应的 *suspend\_page()* 调用成对使用。

表 8.11 *resume\_page()* 的选项

选项	类型	描述
<i>group</i>	字符串	（如果文档使用页面组，则要求此选项；否则，不允许使用此选项。）恢复的页面所属页面组的名称。必须已经用 <i>begin_document()</i> 中的 <i>groups</i> 选项定义了组名。
<i>pagenumber</i>	整型	如果提供了此选项，将恢复 <i>group</i> 选项中选择的页面组内、具有指定页码的页面（如果文档不使用页面组，则在文档中恢复）。如果缺少此选项，将恢复该组中的最后一页。

**Function open\_file(filename As String) As Long**

**Sub close()**

不常用，使用 *begin\_document()* 和 *end\_document()*。

```
Sub begin_page(width As Double, height As Double)
Sub end_page()
```

不常用，使用 *begin\_page\_ext()* 和 *end\_page\_ext()*。

8.2.3 参数处理

PDFlib 可维护众多内部参数，用来控制 PDFlib 的操作和 PDF 输出的外观。有四个函数用来设置和返回数字及字符串参数。所有参数（包括关键字和值）都区分大小写。本章以下各个小节将分别介绍提供的参数。

```
VB RB Function get_value(key As String, modifier As Double) As Double
C# double get_value(String key, double modifier)
```

获取某些数字型 PDFlib 参数的值。

**key** 要查询的参数名。

**modifier** 应用到参数的可选修饰符。各参数表中将说明是否要求修饰符，以及与之对应的内容。如果不使用，修饰符必须为 o。

返回值 参数的数值。

范围 取决于 *key*。

另请参见 *get\_pdi\_value()*

```
VB RB Sub set_value(key As String, value As Double)
C# void set_value(String key, double value)
```

设置某些数字型 PDFlib 参数的值。

**key** 要设置的参数名。

**value** 要设置的新参数值。

范围 取决于 *key*。

```
VB RB Function get_parameter(key As String, modifier As Double) As String
C# String get_parameter(String key, double modifier)
```

获取某些字符串型 PDFlib 参数的内容。

**key** 要查询的参数名。

**modifier** 应用到参数的可选修饰符。各参数表中将说明是否要求修饰符，以及与之对应的内容。如果不使用，修饰符必须为 o。

返回值 作为超文本字符串的参数字符串值。可以在所处文档范围结束之前使用返回的字符串。如果没有信息，将返回空字符串。

范围 取决于 *key*。

另请参见 *get\_pdi\_parameter()*



VB RB

Sub set\_parameter(key As String, value As String)

C#

void set\_parameter(String key, String value)

设置某些字符串型 PDFlib 参数。

**key** 要设置的参数名。

**value** （名称字符串）要设置的新参数值。

范围 取决于 *key*。

8.2.4 PDFlib 虚拟文件系统 (PVF) 函数

VB RB

Sub create\_pvf(filename As String, data, optlist As String)

C#

void create\_pvf(String filename, byte[] data, String optlist)

用内存中提供的数据创建命名的只读虚拟文件。

**filename** （名称字符串）虚拟文件的文件名。这是个任意字符串，稍后可用来引用其他 PDFlib 调用中的虚拟文件。

**data** 一个变体 （在 REALbasic 中：MemoryBlock），包含构成虚拟文件的数据。

**optlist** 根据表 8.12 提供的选项列表。

详细说明 可以为任何使用输入文件的 API 函数提供虚拟文件名（虚拟文件不能用于生成的 PDF 输出，可在 *begin\_document()* 中使用空文件名，以实现这一目的）。其中一些函数可能会在虚拟文件上设置锁定，直至数据不再需要为止。虚拟文件将一直保存在内存中，直至使用 *delete\_pvf()* 显式删除或在 *delete()* 中自动删除为止。

每个 PDFlib 对象都将维护自己的 PVF 文件组。虚拟文件不能在不同 PDFlib 对象之间共享，但可以通过虚拟文件，使用相同的 PDFlib 对象创建多个文档。处理不同 PDFlib 对象的多个线程无需同步 PVF 的使用。如果 *filename* 引用现有虚拟文件，将引发异常。此函数不检查一般磁盘文件是否正在使用 *filename*。

除非已经提供了 *copy* 选项，否则在相应地成功调用 *delete\_pvf()* 之前，调用方不能修改或释放（删除）提供的数据。如不遵守此规则，很可能会导致崩溃。

范围 任意

表 8.12 *create\_pvf()* 的选项

选项	类型	描述
<i>copy</i>	布尔型	PDFlib 会立即创建所提供数据的内部副本。在这种情况下，调用方可以在此调用后立即删除提供的数据。在 <i>COM</i> 、 <i>.NET</i> 和 <i>Java</i> 绑定中， <i>copy</i> 选项将自动设为 <i>true</i> （其他绑定的默认设置： <i>false</i> ）。在其他语言绑定中，除非提供 <i>copy</i> 选项，否则不会复制数据。

VB RB

Function delete\_pvf(filename As String) As Long

C#

int delete\_pvf(String filename)

删除命名的虚拟文件，并释放它的数据结构（但不释放内容）。

**filename** （名称字符串）提供给 *create\_pvf()* 的虚拟文件文件名。

返回值 -1 其条件是，对应的虚拟文件存在，但已被锁定；否则，为 1。

详细说明 如果文件没有锁定，PDFlib 将立即删除与 *filename* 关联的数据结构。如果 *filename* 未引用有效的虚拟文件，则此函数不执行任何操作，也不发出任何提示。成功调用此函数后，可以重复使用 *filename*。所有虚拟文件都将在 *delete()* 自动中删除。

具体语义取决于是否为对应的 *create\_pvf()* 调用提供了 *copy* 选项：如果提供了 *copy* 选项，将同时释放文件的管理数据结构和实际的文件内容；否则，由于假定会由客户端完成此动作，所以不会释放内容。

范围 任意

8.2.5 异常处理

表 8.13 列出了本节的相关参数及相应的值。

表 8.13 用于异常处理的参数和值

函数	关键字	说明
<i>set_parameter</i>	<i>warning</i>	启用或禁用警告（非致命异常）。可能的值包括 <i>true</i> 和 <i>false</i> 。 范围：任意。默认值： <i>true</i>

**VB RB** Function *get\_errnum()* As Long  
**C#** int *get\_errnum()*

获取上次引发的异常的编号或者函数调用失败的原因。

返回值 异常的编号，或者最近调用失败并返回错误代码的原因代码。

范围 PDFlib 异常引发和 PDFlib 对象消亡之间。或者，也可以在函数返回 -1 错误代码之后调用此函数，但必须在调用任何其他函数之前调用；本节列出的函数除外。

绑定 在 .NET 和 REALbasic 中，还可以在 *PDFlibException* 对象中将此函数用作 *get\_errnum()*。

**VB RB** Function *get\_errmsg()* As String  
**C#** String *get\_errmsg()*

获取上次引发的异常的文本或者函数调用失败的原因。

返回值 包含上次引发的异常描述或者最近一次函数调用失败并返回错误代码的原因的文本。

范围 PDFlib 异常引发和 PDFlib 对象消亡之间。或者，也可以在函数返回 -1 错误代码之后调用此函数，但必须在调用任何其他函数之前调用；本节列出的函数除外。

绑定 在 .NET 和 REALbasic 中，还可以在 *PDFlibException* 对象中将此函数用作 *get\_errmsg()*。

**VB RB** Function *get\_apiname()* As String  
**C#** String *get\_apiname()*

获取引发上次异常或失败的 API 函数名。

返回值 引发异常的函数名，或最近一次函数调用失败并返回错误代码的函数名。

范围 PDFlib 异常引发和 PDFlib 对象消亡之间。或者，也可以在函数返回 -1 错误代码之后调用此函数，但必须在调用任何其他函数之前调用；本节列出的函数除外。

绑定 在 .NET 和 REALbasic 中，还可以在 *PDFlibException* 对象中将此函数用作 *get\_apiname()*。

### 8.2.6 实用程序函数

本节不适用于 COM、.NET 和 REALbasic。

## 8.3 文本函数

### 8.3.1 字体处理

表 8.14 列出了本节的相关参数及相应的值。

表 8.14 字体函数的参数和值（参见第 168 页上的第 8.2.3 节“参数处理”）

函数	关键字	说明
<i>set_parameter</i>	<i>FontAFM</i> <i>FontPFM</i> <i>FontOutline</i> <i>Encoding</i> <i>HostFont</i> <i>ICCProfile</i> <i>SearchPath</i> 标准 - <i>OutputIntent</i>	如 <i>UPR</i> 文件中各个类别所示的对应资源文件行（参见第 42 页上的第 3.1.5 节“资源配置和文件搜索”）。可由多个调用向内部列表添加新条目。（另请参见表 8.3 中的 <i>resourcefile</i> ）。范围：任意
<i>get_value</i>	<i>font</i>	返回已经用 <i>setfont()</i> 设置的当前字体的标识符；如果没有设置字体，则返回 -1 范围：页面、图案、模板、字形
<i>get_value</i>	<i>fontmaxcode</i>	返回标识符中的字体的有效字形 ID 值。范围：任意
<i>get_parameter</i>	<i>fontname</i>	当前字体的名称，必须事先用 <i>setfont()</i> 设置。范围：页面、图案、模板、字形
<i>get_parameter</i>	<i>fontencoding</i>	当前字体使用的编码名称或 <i>CMap</i> 。字体必须事先用 <i>setfont()</i> 设置。范围：页面、图案、模板、字形
<i>get_value</i>	<i>fontsize</i>	当前字体的字体大小，必须事先用 <i>setfont()</i> 设置。范围：页面、图案、模板、字形
<i>get_parameter</i>	<i>fontstyle</i>	当前字体的样式，类似 <i>fontstyle</i> 选项（ <i>normal</i> 、 <i>bold</i> 、 <i>italic</i> 或 <i>bolditalic</i> ）。范围：页面、图案、模板、字形
<i>get_value</i>	<i>capheight</i> <i>ascender</i> <i>descender</i>	返回修饰符标示的字体的字体规格信息。有关更多详细信息，参见第 81 页上的第 4.6 节“文本规格和文本变体”。这些值以字体大小的分数表示，因此必须乘以所需的字体大小。范围：任意
<i>set_parameter</i>	<i>fontwarning</i>	如果为 <i>false</i> ，且不能载入字体 / 编码组合，将 <i>load_font()</i> 将返回 -1。默认值： <i>true</i> 。范围：任意
<i>get_value</i>	<i>monospace</i>	如果已经设置了字体，将返回当前字体的 <i>monospace</i> 选项值；否则，返回 0。范围：页面、图案、模板、字形
<i>set_value</i>	<i>subsetlimit</i>	如果文档使用字体子集大于字体中的字符百分比，将停用字体子集。默认值： <i>100 percent</i> 。范围：任意
<i>set_value</i>	<i>subsetminsize</i>	子集化将只应用于大于此 KB 大小的字体（参见第 69 页上的第 4.3 节“字体嵌入和子集化”）。默认值： <i>100 KB</i> 。范围：任意
<i>set_parameter</i>	<i>autosubsetting</i>	控制 <i>TrueType</i> 和 <i>OpenType</i> 字体的子集化自动启动。默认值： <i>true</i> 。范围：任意
<i>set_parameter</i>	<i>autocidfont</i>	控制是否自动将使用除 <i>macroman</i> 和 <i>winansi</i> 之外的编码的 <i>TrueType</i> 字体转换为 <i>CID</i> 字体（参见第 69 页上的第 4.3 节“字体嵌入和子集化”）。默认值： <i>true</i> 。范围：任意

表 8.14 字体函数的参数和值（参见第 168 页上的第 8.2.3 节“参数处理”）

函数	关键字	说明
<i>set_parameter</i>	<i>unicodemap</i>	控制 <i>ToUnicode CMap</i> 的生成（参见第 76 页上的第 4.5.1 节“页面内容和超文本的 <i>Unicode</i> ”）。标签 <i>PDF</i> 模式将忽略此参数。默认值： <i>true</i> 。范围：任意

**VB RB** **Function** *load\_font*(*fontname* As String, *encoding* As String, *optlist* As String) As Long  
**C#** **int** *load\_font*(String *fontname*, String *encoding*, String *optlist*)

搜索字体，并准备稍后使用。

**fontname**（名称字符串）字体的真实名称或别名。它将用来根据第 69 页上的第 4.3.1 节“PDFlib 搜索字体的方法”中的描述查找字体数据。区分大小写。

**encoding** 字体使用的编码（区分大小写），必须为下列编码之一（有关更多信息，参见第 72 页上的第 4.4 节“编码详细信息”）：

- ▶ 一种预定义 8 位编码，*winansi*、*macroman*、*macroman\_apple*、*ebcdic*、*ebcdic\_37*、*pdfdoc*、*iso8859-X*、*cpXXXX* 或 *U+XXXX*；
- ▶ 用于自动选择编码的 *host* 或 *auto*；
- ▶ 从文件载入的用户定义编码或通过 *encoding\_set\_char()* 定义的编码的名称；
- ▶ 用于基于 *Unicode* 的寻址的 *unicode*；
- ▶ 用于字形 ID 寻址的 *glyphid*；
- ▶ 用于选择字体内部编码的 *builtin*；
- ▶ 标准 *CMap* 的名称（参见第 84 页上的第 4.7 节“中文、日文和韩文文本”）；
- ▶ 操作系统已知的编码名称（并非所有平台上都可以使用）。

编码必须兼容选择的字体。表 8.15 列出了所有允许的编码和字体类型组合。

**optlist** 根据表 8.16 提供的选项列表。

返回值 字体句柄，供稍后与 *setfont()* 一起使用。*fontwarning* 参数或选项设为 *false* 时，此函数的行为会发生改变。在这种情况下，如果不能载入请求的字体 / 编码组合，此函数将返回错误代码 -1，而不引发异常。但是，提供错误参数时，仍将引发异常。

除用作 *setfont()* 及相关函数的自变量外，返回的数值字体句柄对用户没有任何意义。尤其是，在不同文档中请求相同的字体 / 编码组合可能会导致不同的字体句柄。

除非提供了不同的 *encoding* 参数或 *fontstyle* 选项，否则再次使用相同字体名称调用此函数时，将返回与第一次调用相同的字体句柄。

冲突的选项：通过 *load\_font()* 载入或者通过 *fill\_textblock()* 请求字体，而不使用 *embedding*、*subsetting* 或 *subsetting* 时，如果稍后再次载入相同字体，将忽略这些选项。

表 8.15 编码与字体类型的关系

字体格式	8 位 编码	内建	字形	<i>Unicode</i> 、 <i>cp936</i> <sup>1</sup> 等。	<i>Unicode CMap</i>	其他 <i>CMap</i>
<i>PostScript Type 1</i>	是	是 <sup>2</sup>	–	是 <sup>3</sup>	–	–
<i>Type 3</i>	是	–	–	–	–	–
带有 <i>TrueType</i> 轮廓的 <i>TrueType</i> 和 <i>OpenType</i> <sup>4</sup>	是	仅限 <i>Symbol</i> 字体	是 <sup>5</sup>	是 <sup>5</sup>	–	–
具有 <i>PostScript</i> 轮廓的 <i>Western OpenType (SID)</i> <sup>4</sup>	是	是	是 <sup>5</sup>	是 <sup>5</sup>	–	–

表 8.15 编码与字体类型的关系

字体格式	8 位 编码	内建	字形	Unicode、 cp936 <sup>1</sup> 等。	Unicode CMap	其他 CMap
具有 <i>PostScript</i> 轮廓的 <i>CJK</i> <i>OpenType</i> (CID)	—	—	是	是	是	是 <sup>6,7</sup>
标准 <i>CJK</i> 字体（不能嵌入）	—	—	—	—	是	是 <sup>7</sup>

1. 目前只支持 *Windows* 系统。
2. 不支持 *PDF* 核心字体，*Symbol* 和 *ZapfDingbats* 除外。
3. 最多可寻址 256 个字形。
4. 除非使用内建编码或 8 位编码，否则将作为 *CID* 字体嵌入；8 位编码只包含 *Adobe* 标准拉丁字符集提供的字符。对于 8 位编码，可将 *autocidfont* 选项设为 *false*，来禁用 *CID* 字体生成。
5. 字体必须允许嵌入。
6. 不支持子集化。
7. 不支持字体规格，即：不支持 *stringwidth()*、 *Kerning*、*textflow*、*overline/underline/strikeout* 和 *textx/texty*。

详细说明 此函数准备字体，以供稍后用于 *setfont()*。将从内存或者（虚拟或基于磁盘）的规格文件载入字体规格。如果由于配置问题而不能使用请求的字体 / 编码组合（例如，无法找到字体、规格和编码文件，或检测到匹配错误），除非将 *fontwarning* 参数设为 *false*，否则将引发异常。如能使用，此参数返回的值可以用作其他字体相关函数的 *font* 自变量。

范围 文档、页面、图案、模板、字形

参数 参见表 8.14。

PDF/X *embedding* 选项必须为 *true*。

表 8.16 *load\_font()* 的选项

选项	类型	描述
<i>autosubsetting</i>	布尔型	动态地决定是否对字体进行子集化处理，具体取决于 <i>subsetlimit</i> 和 <i>subsetminsize</i> 参数以及实际使用的字形。如果已经提供了 <i>subsetting</i> 选项，将忽略此选项。默认值：全局 <i>autosubsetting</i> 参数的值。
<i>autocidfont</i>	布尔型	如果为 <i>true</i> ，具有除 <i>winansi</i> 、 <i>macroman</i> 、 <i>builtin</i> 之外的 8 位编码的 <i>TrueType</i> 字体以及没有字形名称的 <i>OpenType</i> 字体将自动存储为 <i>CID</i> 字体。这样可防止某些 <i>winansi</i> 编码之外的字形无法访问的问题。默认值：全局 <i>autocidfont</i> 参数的值。
<i>embedding</i>	布尔型	控制是否嵌入字体。此选项对 <i>Type 3</i> 字体没有任何影响。如果要嵌入字体，除规格信息外，还必须提供字体轮廓文件（ <i>TrueType</i> 和 <i>OpenType</i> 字体则无此要求），并且， <i>PDF</i> 输出内必须包括实际的字体轮廓定义。如果不嵌入字体， <i>PDF</i> 输出中只包括字体的常规信息。默认值： <i>false</i>
<i>fontstyle</i>	关键字	控制人工字形的生成。这些伪字形仅用于没有嵌入的 <i>TrueType</i> （非 <i>TTC</i> ）和 <i>OpenType</i> 字体（参见第 83 页上的第 4.6.3 节“文本变体”）。可能的关键字包括 <i>normal</i> 、 <i>bold</i> 、 <i>italic</i> 和 <i>bolditalic</i> 。默认值： <i>normal</i> 。
<i>fontwarning</i>	布尔型	如果为 <i>true</i> ，无法载入请求的字体 / 编码组合时，将引发异常；如果为 <i>false</i> ，将返回错误代码。（编码搜索受 <i>fontwarning</i> 参数的控制，而不受 <i>fontwarning</i> 选项的控制。）默认值：全局 <i>fontwarning</i> 参数的值。
<i>kerning</i>	布尔型	控制是否从字体中读取字距调整值（参见第 81 页上的第 4.6 节“文本规格和文本变体”）。默认值： <i>false</i>
<i>monospace</i>	整型 1...2048	强制字体中的所有字形使用指定的宽度（在字体坐标系中：1000 单位等于字体大小）。对于 <i>Type 3</i> 字体，将修改所有不同于 <i>o</i> 的字形宽度。只建议为标准 <i>CJK</i> 字体使用此选项，不支持核心字体；如果是嵌入字体，则忽略此选项。默认值： <i>absent</i> （将使用字体提供的规格）
<i>subsetlimit</i>	浮点型或 百分比	如果文档中使用的字形与字体中字形总数的百分比超出提供的百分比，将停用字体子集化。默认值：全局 <i>subsetlimit</i> 参数的值。

表 8.16 `load_font()` 的选项（续）

选项	类型	描述
<code>subsetminsize</code>	浮点型	如果原始字体文件的大小小于提供的 <i>KB</i> 值，将停用字体子集化。默认值：全局 <code>subsetminsize</code> 参数的值。
<code>subsetting</code>	布尔型	控制是否对字体进行子集化处理，具体取决于文档中使用的字形总数以及 <code>subsetlimit</code> 和 <code>subsetminsize</code> 选项（参见第 69 页上的第 4.3 节“字体嵌入和子集化”）。默认值： <i>false</i>
<code>unicodemap</code>	布尔型	控制 <i>ToUnicode CMap</i> 的生成（参见第 76 页上的第 4.5.1 节“页面内容和超文本的 <i>Unicode</i> ”）。标签 <i>PDF</i> 模式将忽略此选项。默认值： <i>true</i>

**VB RB** Sub setfont(font As Long, fontsize As Double)

**C#** void PDF\_setfont(int font, double fontsize)

以指定的大小设置当前字体。

**font** `load_font()` 返回的字体句柄。

**fontsize** 字体的大小，以当前用户坐标系统的单位表示。字体大小不能为 0；如果为负值，将产生与当前转换矩阵相对的镜像文本。

**详细说明** 在绘制任何文本之前，必须在每个页面上设置字体。不能在多个页面间保留字体设置。在每一页上，更改当前字体的次数不受任何限制。

**范围** 页面、图案、模板、字形

**参数** 参见表 8.14。此函数自动将 *leading* 参数设为 *fontsize*。

### 8.3.2 用户定义 (Type 3) 字体

**VB RB** Sub begin\_font(fontname As String, a As Double, b As Double, c As Double, d As Double, e As Double, f As Double, optlist As String)

**C#** void begin\_font(String fontname, double a, double b, double c, double d, double e, double f, String optlist)

开始 Type 3 字体定义。

**fontname**（名称字符串）用来注册字体的名称，稍后可用于 `load_font()`。

**a, b, c, d, e, f** 字体矩阵的元素。此矩阵定义将在其中绘制字形的坐标系统。这六个值在 PostScript 和 PDF 中，以相同的方式构成坐标系统（参见相关参考）。为了避免转换退化， $a*d$  不能等于  $b*c$ 。1000 x 1000 坐标系统典型的字体矩阵是  $[0.001, 0, 0, 0.001, 0, 0]$ 。

**optlist** 根据表 8.17 提供的选项列表。

**详细说明** 此函数将所有文本、图形和颜色状态参数重设为他们的默认值。字体可以包含任意数量的字形，但只能通过编码访问 256 个字形。可以在当前文档范围结束之前使用定义的字形。

**范围** 文档、页面；此参数是字体范围的开头，必须与对应的 `end_font()` 调用成对使用。

表 8.17 `begin_font()` 的选项

选项	类型	描述
<code>colorized</code>	布尔型	如果为 <i>true</i> ，字体可以显式指定具体字符的颜色。如果为 <i>false</i> ，将使用（使用字体时，而非定义字体时的）当前颜色绘制所有字符，字形定义不能包含除蒙版外的任何其他色彩运算符或图像。默认值： <i>false</i>

VB RB

Sub end\_font()

C#

void end\_font()

终结 Type 3 字体定义。

范围 字体；此函数终结字体范围，必须与对应的 *begin\_font()* 调用成对使用。

VB RB

Sub begin\_glyph(glyphname As String,  
wx As Double, llx As Double, lly As Double, urx As Double, ury As Double)

C#

void begin\_glyph(String glyphname,  
double wx, double llx, double lly, double urx, double ury)

开始 Type 3 字体的字形定义。

**glyphname** 字形的名称。此名称必须用于将与该字体一起使用的任何编码。字体内的字形名称必须是唯一的。

**wx** 字形坐标系统内字形的宽度，具体由字体的矩阵指定。

**llx, lly, urx, ury** 如果字体的 *colorized* 选项为 *false*（默认值），这些值可指定字形定界框左下角和右上角的坐标。为避免出现 PostScript 打印问题，定界框值必须是正确的。如果字体的 *colorized* 选项为 *true*，所有四个值都必须为 0。

详细说明 字体中的字形可使用文本、图形和图像函数定义。但是，如果字体的 *colorized* 选项为 *true*，或者已使用 *mask* 选项打开了图像，则只能使用图像。强烈建议使用内嵌图像功能定义 Type 3 中的位图（参见第 105 页上的第 5.1.1 节“基本图像处理”）。

因为 *colorized* 选项为 *true* 时，字形定义将继承所处页面的整个图形状态，所以字形定义应显式设置图像状态与字形定义相关的各个方面（例如 *linewidth*）。

范围 页面，字体；此参数是字形范围的开头，必须与对应的 *end\_glyph()* 调用成对使用。

VB RB

Sub end\_glyph()

C#

void end\_glyph()

终结 Type 3 字体的字形定义。

范围 字形；此函数终结字形范围，必须与对应的 *begin\_glyph()* 调用成对使用。

### 8.3.3 编码定义

VB RB

Sub encoding\_set\_char(encoding As String, slot As Long, glyphname As String, uv As Long)

C#

void encoding\_set\_char(String encoding, int slot, String glyphname, int uv)

向自定义编码添加字形名称和 / 或 Unicode 值。

**encoding** 编码的名称。此名称必须用于 *load\_font()*。编码名称必须不同于任何内置编码以及所有以前使用过的编码。

**slot** 要定义的字符位置， $0 \leq \text{slot} \leq 255$ 。在给定的编码中，特定插槽只能填充一次。

**glyphname** 字符的名称。

**uv** 字符的 Unicode 值。

详细说明 此函数可多次调用，最多可在编码内定义 256 个字符插槽。在第一次使用之前，可向特定编码添加更多的字符；否则，将引发异常。并非所有代码点都必须指定，未定义的插槽将用 `.notdef` 填充。

字形名称和 Unicode 值可能有三种组合：

- ▶ 提供 `glyphname`，`uv = 0`：这相当于没有 Unicode 值的编码文件；
- ▶ 提供 `uv`，但没有提供 `glyphname`：这相当于代码页文件；
- ▶ 提供 `glyphname` 和 `uv`：这相当于有 Unicode 值的编码文件；

可以在当前对象范围结束之前使用定义的编码。

范围 对象、文档、页面、图案、模板、路径、字体、字形

### 8.3.4 简单文本输出

注：在本节中，所有提供给函数的文本都必须匹配用 `load_font()` 选择的编码。这适用于 8 位文本以及 *Unicode* 或其他通过 *CMap* 选择的编码。由于 *Acrobat* 中的限制，文本字符串的长度不能超过 32 KB。

表 8.18 列出了本节的相关参数及相应的值。

表 8.18 文本函数的参数和值

函数	关键字	说明
<code>set_parameter</code>	<code>autospace</code>	如果为 <code>true</code> ，且当前字体兼容 <i>Unicode</i> ， <i>PDFlib</i> 将在每个用 <code>show</code> 操作生成的文本输出后添加一个空格字符 ( <code>&amp;H2o</code> )。这可能对生成标签 <i>PDF</i> 很有用（参见第 148 页上的第 7.5 节“标签 <i>PDF</i> ”）。请注意，在 <code>show</code> 操作后添加空格会改变当前文本位置。默认值： <code>false</code> 。范围：任意
<code>set_parameter</code>	<code>charref</code>	如果为 <code>true</code> ，将启用数字和字符实体引用的替换（参见第 79 页上的第 4.5.5 节“字符引用”）。默认值： <code>false</code>
<code>set_value</code> <code>get_value</code>	<code>charspacing</code>	设置或获取字符间距，即：在字符串中放置具体字符之后，当前点的位移。它以用户坐标系统的单位指定，并将在每页的开头和结尾重设为默认值 0。要扩大字符间距，可在横排书写模式中使用正值，在直排书写模式中使用负值。范围：页面、图案、模板、字形、文档。
<code>set_parameter</code>	<code>glyphwarning</code>	如果为 <code>true</code> ，由于字体不包含相应的字形描述而无法显示字形时，将引发异常。如果为 <code>false</code> ，将用空格字符或字形 <i>ID 0</i> 替换缺失的字形。默认值： <code>false</code> 。范围：任意
<code>set_value</code> <code>get_value</code>	<code>horizscaling</code>	将水平文本缩放比例设定为给定的百分比，或获取相关设置。文本缩放比例可按给定的比例收缩或扩展文本。在每页的开头和结尾，此参数将设为默认值 100。文本缩放比例始终与水平坐标相对。范围：页面、图案、模板、字形、文档。
<code>set_value</code> <code>get_value</code>	<code>italicangle</code>	指定文本的斜体（倾斜）角度的度数，范围在 -90° 和 90° 之间。如果只有常规字体，可使用负值模拟斜体文本，尤其是 <i>CJK</i> 字体（参见第 83 页上的第 4.6.3 节“文本变体”）。默认值：0（将在每页的开头和结尾重设此参数）。范围：页面、图案、模板、字形、文档
<code>set_parameter</code> <code>get_parameter</code>	<code>kerning</code>	如果为 <code>true</code> ，将为已用 <code>kerning</code> 选项打开的字体启用字距微调；如果为 <code>false</code> ，则停用字距微调。（参见第 81 页上的第 4.6 节“文本规格和文本变体”）。默认值： <code>true</code> 。范围：任意
<code>set_value</code> <code>get_value</code>	<code>leading</code>	设置或获取行距，即相邻文本行的基线之间的距离。 <code>leading</code> 用于 <code>continue_text()</code> 。使用 <code>setfont()</code> 选择新字体时，它设为字体大小的值。设置 <code>leading</code> 等于字体大小将导致密集行距（ <code>leading = 0</code> 将导致叠印行）。但是，相邻行的字母上缘和字母下缘通常不会重叠。范围：页面、图案、模板、字形。



表 8.18 文本函数的参数和值（续）

函数	关键字	说明
<i>set_parameter</i> <i>get_parameter</i>	<i>textformat</i>	指定文本输出函数预计客户端提供字符串时使用的格式。可能的值包括 <i>bytes</i> 、 <i>utf8</i> 、 <i>utf16</i> 、 <i>utf16le</i> 、 <i>utf16be</i> 和 <i>auto</i> （参见第 77 页上的第 4.5.2 节“内容字符串、超文本字符串和名称字符串”）。默认值： <i>auto</i> 。范围：任意
<i>set_value</i> <i>get_value</i>	<i>textrendering</i>	设置或获取当前文字渲染模式。在每页的开头，它设为默认值 <i>o</i> 。 范围：页面、图案、模板、字形。 <i>0</i> 填充文本 <i>1</i> 描边文本（轮廓） <i>2</i> 填充和描边文本 <i>3</i> 不可见文本 <i>4</i> 填充文本并将其添加到剪贴路径 <i>5</i> 描边文本并将其添加到剪贴路径 <i>6</i> 填充并描边文本并将其添加到剪贴路径 <i>7</i> 将文本添加到剪贴路径
<i>set_value</i> <i>get_value</i>	<i>textrise</i>	设置或获取文本抬升参数，该参数指定所需文本位置与默认基线之间的距离。如文本抬升参数为正值，将向上移动文本。文本抬升始终与垂直坐标相对。这可能对上标和下标很有用。在每页开头，文本抬升设为默认值 <i>o</i> 。范围：页面、图案、模板、字形。
<i>get_value</i>	<i>textx</i> <i>texty</i>	分别获取当前文本位置的 <i>x</i> 或 <i>y</i> 坐标。范围：页面、图案、模板、字形。
<i>set_parameter</i> <i>get_parameter</i>	<i>underline</i> <i>overline</i> <i>strikeout</i>	设置或获取当前下划线、上划线和删除线模式；显式更改和开始新页面之前，将一直保留这些模式。这些模式可以彼此独立设置，并在每页开头重设为 <i>false</i> （参见第 81 页上的第 4.6 节“文本规格和文本变体”）。 范围：页面、图案、模板、字形。 <i>true</i> 下划线 / 上划线 / 删除线文本 <i>false</i> 不使用下划线 / 上划线 / 删除线文本
<i>set_value</i> <i>get_value</i>	<i>underlineposition</i>	下划线文本的描边线相对于基线的位置（以字体大小的分数表示）。可提供值 <i>10000000</i> ，以设定将从字体规格或轮廓文件返回的字体特定值。默认值： <i>10000000</i>
<i>set_value</i> <i>get_value</i>	<i>underlinewidth</i>	下划线文本的绝对线宽。可提供值 <i>o</i> ，以设定将从字体规格或轮廓文件返回的字体特定值。默认值： <i>o</i>
<i>set_value</i> <i>get_value</i>	<i>wordspacing</i>	设置或获取字间距，即：在字符串中放置具体单字之后，当前点的位置。换言之，当前点将水平移至每个空格字符 ( <i>&amp;H2o</i> ) 之后。间距值以文本间隔单位指定，并将在每页的开头和结尾重设为默认值 <i>o</i> 。范围：页面、图案、模板、字形、文档。

VB RB

Sub set\_text\_pos(x As Double, y As Double)

C#

void set\_text\_pos(double x, double y)

设置文本输出在页面上的位置。

**x, y** 要设置的当前文本位置。

详细说明 在每页开头，文本位置设为默认值 (*o*, *o*)。图形输出的当前点和当前文本位置将单独维护。

范围 页面、图案、模板、字形

参数 参见表 8.18。

**VB RB** Sub show(text As String)

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

在当前文本位置，使用当前字体和大小打印文本。

**text** （内容字符串）要打印的文本。

详细说明 必须已经使用 *setFont()* 设置了字体。当前文本位置移至打印文本的结尾。

范围 页面、图案、模板、字形

参数 参见表 8.18。

**VB RB** Sub show\_xy(text As String, x As Double, y As Double)

**C#** void show\_xy(String text, double x, double y)

以当前字体打印文本。

**text** （内容字符串）要打印的文本。

**x, y** 用户坐标系统中的位置，将在其上打印文本。

详细说明 必须已经使用 *setFont()* 设置了字体。当前文本位置移至打印文本的结尾。

范围 页面、图案、模板、字形

参数 参见表 8.18。

**VB RB** Sub continue\_text(text As String)

**C#** void continue\_text(String text)

在下一行打印文本。

**text** （内容字符串）要打印的文本。如果是空字符串，文本位置仍将移至下一行。

详细说明 文本的位置（*x* 和 *y* 位置）以及各行间的间隔由 *leading* 参数及最近一次 *fit\_textline()*, *show\_xy()* 或 *set\_text\_pos()* 调用决定。当前点将移至打印文本的结尾处；此函数后续调用的 *x* 位置将不会更改。

范围 页面、图案、模板、字形；不应在直排书写模式下使用此函数。

参数 参见表 8.18。

**VB RB** Sub fit\_textline(text As String, x As Double, y As Double, optlist As String)

**C#** void fit\_textline(String text, double x, double y, String optlist)

根据各种选项，在位置 (*x, y*) 上放置单行文本。

**text** （内容字符串）要打印的文本。

**x, y** 用户坐标系统中的参考点坐标；将根据各种选项，在此坐标上放置文本。

**optlist** 选项列表，根据表 8.19 指定格式化选项，并根据表 8.20 指定外观选项。

详细说明 除非用选项显式覆盖，否则将使用当前文本和图形状态参数控制文本输出的外观。另一方面，此函数不会修改当前图形状态（尤其是，不会影响当前字体）。但是，当前文本位置将调整到指向生成文本输出的结尾处。

此函数不会为具有非 **Unicode CMap** 的标准 **CJK** 字体创建任何输出（参见第 86 页上的“标准 **CJK** 字体和 **CMap** 的限制”）。

范围 页面、图案、模板、字形；不应在直排书写模式下使用此函数。

参数 参见表 8.18。

表 8.19 *fit\_textline()* 的格式化选项

关键字	类型	说明
<i>boxsize</i>	浮点值列表	指定框宽度和高度的两个值，将相对于这两个值放置并可能缩放文本。框的左下角即为参考点 <i>(x, y)</i> 。放置文本并使其填充此框的操作由 <i>position</i> 和 <i>fitmethod</i> 选项控制。如果 <i>width = 0</i> ，只考虑高度；如果 <i>height = 0</i> ，则只考虑宽度。在这些情况下，将分别相对于 <i>(x, y)</i> 到 <i>(x, y+height)</i> 的垂直直线或 <i>(x, y)</i> 到 <i>(x+width, y)</i> 的水平直线放置文本。默认值：{0 0}
<i>fitmethod</i>	关键字	指定使文本填充指定框的方法。如果尚未指定框，将忽略此选项。 默认值： <i>nofit</i> 。 <i>nofit</i> 只确定文本位置，不进行任何缩放或剪裁处理。 <i>clip</i> 确定文本位置，并在框的边缘剪裁文本。 <i>meet</i> 根据 <i>position</i> 选项确定文本位置，并在保持长宽比的同时进行缩放以使其完全装入框内。通常，至少有文本的两个边缘与框的对应边缘对齐。 <i>auto</i> 这种方法会自动尝试将文本完全置于框内。具体讲：如果文本可以填充于框内，则相当于 <i>nofit</i> 。否则，将计算缩放因子，以使文本填入框内。如果此因子大于 <i>shrinklimit</i> 选项，将扭曲文本以填充框；否则，将应用 <i>meet</i> 方法。 <i>slice</i> 根据 <i>position</i> 选项确定文本位置，并进行缩放以使其覆盖整个框，可在保持长宽比的同时确保至少文本的一个尺寸可完全包含在框内。通常，文本的某些其他尺寸会延伸到框外，因此，将被剪裁。 <i>entire</i> 根据 <i>position</i> 选项确定文本的位置，并进行缩放以使其覆盖整个框。通常，这种方法会扭曲文本。 <i>scale</i> 选项将被忽略。
<i>locallink</i>	选项列表	如果提供了此选项，将通过文本创建可以使用的本地链接；即：将 <i>destname</i> 选项设为 <i>text</i> 参数，并创建 <i>type=Link</i> 的批注。此文本必须匹配已用 <i>add_nameddest()</i> 创建的命名目标的名称。可提供下列选项（有关详细信息，参见表 8.51）： <i>annotcolor</i> 、 <i>borderstyle</i> 、 <i>dasharray</i> 和 <i>highlight</i> （将自动设置 <i>action</i> 和 <i>usercoordinates</i> 选项）。
<i>margin</i>	浮点值列表	一个或两个浮点值，描述文本框的额外水平和垂直扩展。默认值：0
<i>orientate</i>	关键字	指定放置文本时所需的文本方向。默认值： <i>north</i> <i>north</i> 向上 <i>east</i> 指向右侧 <i>south</i> 向下 <i>west</i> 指向左侧
<i>position</i>	浮点值或关键字列表	（对齐方式控制）一个或两个值，指定文本定界框内的参考点位置 <i>(x, y)</i> ；{0 0} 表示位于文本框左下角，{100 100} 表示位于右上角。如果已经指定了 <i>boxsize</i> 选项，则 <i>position</i> 选项还可指定目标框的位置。这些值表示为文本宽度和高度的百分比。如果两个百分比相等，指定一个浮点值即可。  关键字 <i>left</i> 、 <i>center</i> 、 <i>right</i> （ <i>x</i> 方向）或者 <i>bottom</i> 、 <i>center</i> 、 <i>top</i> （ <i>y</i> 方向）可用来替代值 0、50 和 100。如果只指定一个关键字，将添加另一方向对应的关键字。以下是一些示例： {0 50} 或 {left center} 生成左对齐文本； {50 50} 或 {center} 生成居中文本； {100 50} 或 {right center} 生成右对齐文本。  默认值：0（左下角）

表 8.19 `fit_textline()` 的格式化选项

关键字	类型	说明
<code>rotate</code>	浮点型	旋转坐标系，将中心用作参考点并将指定的值用作以度数表示的旋转角度。这将旋转框和文本。放置文本时，会重设旋转选项。默认值： <code>0</code>
<code>weblink</code>	选项列表	如果提供了此选项，将通过文本创建可以使用的 <code>weblink</code> ；即：将使用默认选项或选项列表中提供的选项创建 <code>type=Link</code> 的批注。可提供下列选项（有关详细信息，参见表 8.51）： <code>annotcolor</code> 、 <code>borderstyle</code> 、 <code>dasharray</code> 、 <code>highlight</code> 和 <code>linewidth</code> （将自动设置 <code>action</code> 和 <code>usercoordinates</code> 选项）。提供的文本必须包含有效的 URL。
<code>xadvancelist</code>	浮点值列表	使用用户坐标，指定文本中所有字形的预置宽度。列表的长度必须小于或等于文本中的字形数。如果长度小于字形数，且 <code>glyphwarning</code> 设为 <code>true</code> ，将引发警告。将使用 <code>xadvance</code> 值替代标准字形宽度。字距微调和字符间距等其他效果不受影响。

表 8.20 `fit_textline()` 的外观选项及 `create_textflow()` 的直接或内嵌选项

关键字	类型	说明
<code>charref</code>	布尔型	如果为 <code>true</code> ，将启用数字和字符实体引用的替换（参见第 79 页上的第 4.5.5 节“字符引用”）。默认值： <code>false</code>
<code>charspacing</code>	浮点型或百分比	字符间距（参见表 8.18）。百分比基于 <code>fontsize</code> 。默认值：全局 <code>charspacing</code> 参数。
<code>fillcolor</code>	颜色	文本的填充色。默认值：当前填充色
字体	字体句柄	<code>load_font()</code> 返回的字体句柄。默认值：当前字体
<code>fontsize</code>	浮点型	（如提供了 <code>font</code> 选项，则要求此选项）字体的大小，以当前用户坐标系统的单位表示。默认值：当前字体大小
<code>glyphwarning</code>	布尔型	如果为 <code>true</code> ，由于字体不包含相应的字形描述而无法显示字形时，将引发异常。如果为 <code>false</code> ，将用空格字符或字形 ID <code>o</code> 替换字体中缺失的字形。默认值：全局 <code>glyphwarning</code> 参数。
<code>horizscaling</code>	浮点型或百分比	水平文本缩放（参见表 8.18）。默认值：全局 <code>horizscaling</code> 参数。
<code>italicangle</code>	浮点型	指定文本的斜体（倾斜）角度的度数（参见第 83 页上的第 4.6.3 节“文本变体”）。默认值：全局 <code>italicangle</code> 参数。
<code>kerning</code>	布尔型	字距微调行为（参见表 8.18）。默认值：全局 <code>kerning</code> 参数。
<code>overline</code>	布尔型	上划线模式（参见表 8.18）。默认值：全局 <code>overline</code> 参数。
<code>strikeout</code>	布尔型	删除线模式（参见表 8.18）。默认值：全局 <code>strikeout</code> 参数。
<code>strokecolor</code>	颜色	文本的描边色。默认值：当前描边色
<code>shrinklimit</code>	浮点型或百分比	收缩因子的下限，将用来调整文本。默认值： <code>0.75</code>
<code>textformat</code>	关键字	用来解释提供的文本的格式（参见第 77 页上的第 4.5.2 节“内容字符串、超文本字符串和名称字符串”）。默认值：全局 <code>textformat</code> 参数。
<code>textrendering</code>	整型	文本渲染模式（参见表 8.18）。默认值：全局 <code>textrendering</code> 参数
<code>textrise</code>	浮点型或百分比	文本抬升模式（参见表 8.18）。百分比基于 <code>fontsize</code> 。默认值：全局 <code>text rise</code> 参数
<code>underline</code>	布尔型	下划线模式（参见表 8.18）。默认值：全局 <code>underline</code> 参数。
<code>underlineposition</code>	浮点型、百分比或关键字	下划线文本的描边线相对于基线的位置（绝对值或相对于 <code>fontsize</code> 的值；典型值为 <code>-10%</code> ）。可使用关键字 <code>auto</code> ，设定将从字体规格或轮廓文件返回的字体特定值。默认值： <code>auto</code>
<code>underlinewidth</code>	浮点型、百分比或关键字	下划线文本的线宽（绝对值或相对于 <code>fontsize</code> 的值；典型值为 <code>5%</code> ）。可使用关键字 <code>auto</code> 或值 <code>o</code> 设定，将从字体规格或轮廓文件返回的字体特定值。默认值： <code>auto</code>

表 8.20 `fit_textline()` 的外观选项及 `create_textflow()` 的直接或内嵌选项

关键字	类型	说明
<code>wordspacing</code>	浮点型或百分比	字间距（参见表 8.18）。百分比基于 <code>fontsize</code> 。默认值：全局 <code>wordspacing</code> 参数

VB RB

Function stringwidth(text As String, font As Long, fontsize As Double) As Double

C#

double stringwidth(String text, int font, double fontsize)

以任意字体返回 `text` 的宽度。

**text** （内容字符串）请求其宽度的文本。

**font** `load_font()` 返回的字体句柄。具有非 Unicode CMap 的标准 CJK 字体不能使用字体规格。如果 `font` 引用这种字体，此函数将返回 0，而不考虑 `text` 和 `fontsize` 参数（除非已经在载入字体时提供了 `monospace` 选项）。

**fontsize** 字体的大小，以用户坐标系统的单位表示（参见 `setfont()`）。

返回值 使用任意字体的 `text` 的宽度；该字体用 `load_font()` 和提供的 `fontsize` 选择。返回的宽度值可以是负值（例如，已设置负值的水平缩放比例）。

详细说明 宽度的计算会考虑下列文本参数的当前值：水平缩放比例、字距微调、字符间距和字间距。

范围 页面、图案、模板、路径、字形、文档

参数 参见表 8.18。

Function show\_boxed(text As String, x As Double, y As Double, width As Double, height As Double, mode As String, feature As String) As Long

不常用，对于单行文本，可使用 `fit_textline()`，多行格式化可使用 `*_textflow()`。如果是多行格式化，使用 `minspacing=100%`、`maxspacing=10000%`、`nofitlimit=100%` 和 `shrinklimit=100%` 将产生与使用 `show_boxed()` 相同的结果。格式化后剩余的字符数（即，`show_boxed()` 返回的值）可使用 `info_textflow()` 中的 `remainchars` 选项返回。

### 8.3.5 使用文本流的多行文本

VB RB

Function create\_textflow(text As String, optlist As Long) As Long

C#

int create\_textflow(String text, String optlist)

预处理文本以便稍后进行格式化并创建文本流对象。

**text** （内容字符串）文本流的内容。它可能包含表 8.21 和表 8.24 中列出的各种编码、内嵌选项列表（另请参见第 185 页上的“文本流的内嵌选项列表”）以及各种宏（参见第 185 页上的“文本流的宏选项”）所产生的文本。不过，如果文本为空字符串，将返回有效的文本流句柄。

**optlist** 选项列表，根据表 8.21 或表 8.24 指定文本流选项。将在 `text` 中包含的内嵌选项列表之前计算 `optlist` 中指定的选项，以使内嵌选项优先于 `optlist` 参数中提供的选项。

返回值 文本流句柄，可在 `fit_textflow()`、`info_textflow()` 和 `delete_textflow()` 调用中使用。在关闭文档范围结束和用该句柄调用 `delete_textflow()` 之前，句柄将一直有效。如果出现错误，如

果 *textwarning* 参数或选项为 *false*，此函数将返回错误代码 -1。如果设为 *true*，此参数将在出现错误时引发异常。

**详细说明** 此函数处理提供的文本并据此创建内部数据结构。它决定稍后将由文本格式器使用的文本部分（例如单词），处理内嵌选项列表，在可能的情况下将文本转换为 Unicode 字符串，确定潜在的换行符并基于字体和文本选项计算文本部分的宽度。可应用 *optlist* 参数中的 *textlen* 选项，对部分或全部文本停用内嵌选项列表搜索（第 185 页上的“文本流的内嵌选项列表”）。

此函数不会在生成的 PDF 文档中创建任何输出，只准备文本。使用 *fit\_textflow()*，用预处理过的文本流句柄创建输出。

默认情况下，将用 VT、LS、LF、CR、CRLF、NEL、PS 和 FF 字符强制生成新行（有关这些字符的描述，参见表 4.3）。除 VT 和 LS 之外，所有这些字符都可强制生成新段落（这表示，*parindent* 选项将有效）。FF 立即停止使文本填充 *fitbox* 的操作（将退出 *fit\_textflow()* 函数，并返回字符串 *\_nextpage*）。

水平制表符 (HT) 设置后续文本的新起点。具体由 *hortabmethod* 和 *hortabsize* 选项控制。

如果自动连字符后是一个换行符，将用 *hyphenchar* 选项中指定的字符替换自动连字符 (SHY)。有关更多详细信息，参见第 100 页上的第 4.9.8 节“控制换行算法”。

不支持直排书写模式。

**范围** 任意，对象除外

表 8.21 *create\_textflow()* 的选项

选项	类型	说明
<i>textwarning</i>	布尔型	如果为 <i>true</i> ，选项列表或 <i>load_font()</i> (当 <i>fontname</i> 选项被指定，内部会自动调用此函数) 中发现错误时，将引发异常。 <i>textwarning</i> 选项设 <i>fontwarning</i> 和 <i>glyphwarning</i> 选项为缺省。如果是 <i>false</i> ，此函数将返回错误代码 -1。默认值： <i>true</i>

---

**VB RB** Function *fit\_textflow*(*textflow* As Long,  
    *llx* As Double, *lly* As Double, *urx* As Double, *ury* As Double, *optlist* As String) As String

**C#** String *fit\_textflow*(int *textflow*,  
    double *llx*, double *lly*, double *urx*, double *ury*, String *optlist*)

---

将文本流的下一部分格式化成一个矩形区域。

**textflow** *create\_textflow()* 调用返回的文本流句柄。

**llx, lly, urx, ury** 用户坐标系统中，目标矩形 (*fitbox*) 左下角和右上角的 *x* 和 *y* 坐标。也可以按相反的顺序指定左下角和右上角坐标。

**optlist** 选项列表，根据表 8.22 指定处理选项。

**返回值** 指定函数返回的原因的字符串。

- ▶ *\_stop*: 文本流内的所有文本均已处理完毕。
- ▶ *\_nextpage*: 等候下一页（由于换页符 U+000C 导致）。处理剩余文本要求再次调用 *fit\_textflow()*。
- ▶ *\_boxfull*: *fitbox* 中的空间已全部用完，或框中已设置了最大行数（如通过 *maxlines* 指定的行数）。处理剩余文本要求再次调用 *fit\_textflow()*。
- ▶ *\_boxempty*: 在处理后的 *fitbox* 不包含任何文本，这发生在 *fitbox* 太小而不能装任何文本时。为了避免无穷循环，不对相同的 *fitbox* 再次调用 *fit\_textflow()*。
- ▶ 任何其他字符串：在内嵌选项列表中，提供给 *return* 命令的字符串。

如果同时由多种原因导致返回，将报告列表中的第一项（由上至下）。在下次调用此函数前，返回的字符串将一直有效。

当前文本和图形状态不影响用此函数创建的文本输出（这不同于 `fit_textline()`）。可在 `create_textflow()` 中使用 `fillcolor`、`strokecolor` 和其他外观选项（参见表 8.20）控制文本的外观。从此函数返回后，文本状态不会发生更改。但是，当前文本位置将调整到生成的文本输出的末尾（除非 `blind` 选项已设为 `true`）。

页面、图案、模板、字形

表 8.22 `fit_textflow()` 的选项

选项	类型	说明
<i>blind</i>	布尔型	不生成输出，但将执行所有计算，并可使用 <i>info_textflow()</i> 检查格式化结果。 默认值: <i>false</i>
<i>firstlinedist</i>	浮点型、百分比或关键字	由用户坐标指定的 <i>fitbox</i> 顶部与第一行文本基线之间的距离，可表示为相关字体大小的百分比（如果 <i>fixedleading=true</i> ，将使用行中的第一个字体大小；否则，使用行中最大的字体大小）或关键字（默认值: <i>leading</i> ）: <i>leading</i> 为第一行确定的行距值； <i>Å</i> 等典型的变音字符将与 <i>fitbox</i> 顶部相接。 <i>ascender</i> 为第一行确定的字母上缘值； <i>d</i> 和 <i>h</i> 等字母上缘较高的典型字符将与 <i>fitbox</i> 顶部相接。 <i>capheight</i> 为第一行确定的大写高度值； <i>H</i> 等典型的大写字符将与 <i>fitbox</i> 顶部相接。 <i>xheight</i> 为第一行确定的 <i>xheight</i> 值； <i>x</i> 等典型的小写字符将与 <i>fitbox</i> 顶部相接。  如果 <i>fixedleading=false</i> ，将在第一行的所有行距值、字母上缘值和大写高度值中，选取最大值。
<i>fitmethod</i>	关键字	指定使文本填充 <i>fitbox</i> 的方法。默认值: <i>clip</i> <i>auto</i> 将在隐蔽模式下，使用缩小的字体大小及其他字体相关选项（参见 <i>fontscale</i> ）重复调用 <i>fit_textflow()</i> ，直至全部文本可填充于 <i>fitbox</i> 为止。 <i>clip</i> 文本将在 <i>fitbox</i> 底部截断。 <i>nofit</i> 文本可以延伸到 <i>fitbox</i> 底部之外（但只能延伸到页面边缘）。
<i>fontscale</i>	浮点型	将用提供的缩放因子乘以 <i>fontsize</i> 的值以及 <i>leading</i> 、 <i>minspacing</i> 、 <i>maxspacing</i> 、 <i>spreadlimit</i> 和 <i>space</i> 的绝对值（而非百分比）。默认值: 如果 <i>rewind=0</i> ，默认值为 1；否则，将为与相应 <i>fit_textflow()</i> 调用提供的值。
<i>lastlinedist</i>	浮点型、百分比或关键字	（如果 <i>fitmethod=nofit</i> ，将忽略此选项）由用户坐标指定的，文本最后一行的基线与 <i>fitbox</i> 底部之间的最小距离，可表示为字体大小的百分比（如果 <i>fixedleading= true</i> ，则使用行中的第一个字体大小；否则，使用行中最大的字体大小）或关键字（默认值: 0，即: <i>fitbox</i> 的底部将用作基线，典型的字母下缘将延伸到 <i>fitbox</i> 下方）: <i>descender</i> 为第一行确定的字母下缘值； <i>g</i> 和 <i>j</i> 等具有字母下缘的典型字符将与 <i>fitbox</i> 底部相接。  如果 <i>fixedleading=false</i> ，将于最后一行所有的字母下缘值中，选取最大值。
<i>linespreadlimit</i>	浮点型或百分比	（仅用于 <i>verticalalign=justify</i> 的情况）用户坐标中的最大量，或用以增加垂直对齐行距的行距百分比。默认值: 200%
<i>maxlines</i>	整型或关键字	<i>fitbox</i> 中的最大行数，或关键字 <i>auto</i> ； <i>auto</i> 表示，将在 <i>fitbox</i> 内放置尽可能多的行。达到最大行数限制时， <i>fit_textflow()</i> 将返回字符串 <i>_boxfull</i> 。默认值: <i>auto</i>
<i>orientate</i>	关键字	指定放置文本时所需的文本方向。默认值: <i>north</i> <i>north</i> 向上 <i>east</i> 指向右侧 <i>south</i> 向下 <i>west</i> 指向左侧

表 8.22 `fit_textflow()` 的选项（续）

选项	类型	说明
<i>rewind</i>	整型: -2、-1、 0 或 1	提供文本流的状态重设为另一 <i>fit_textflow()</i> 调用之前的状态。目前支持下列值（默认值: 0）:
		1 退回第一次调用 <i>fit_textflow()</i> 之前的状态。
		0 不重设文本流。
		-1 退回上一次调用 <i>fit_textflow()</i> 之前的状态。
		-2 退回倒数第二次调用 <i>fit_textflow()</i> 之前的状态。
<i>rotate</i>	浮点型	旋转坐标系统, 将 <i>fitbox</i> 的左下角用作中心并将指定的值用作以旋转角度的度数。这将旋转框和文本。放置文本时, 会重设旋转选项。默认值: 0
<i>showborder</i>	布尔型	如果为 <i>true</i> , 将对 <i>fitbox</i> 的边框做描边处理（使用当前图形状态）。这可能会对开发和调试很有用。默认值: <i>false</i>
<i>verticalalign</i>	关键字	<i>fitbox</i> 中文本使用的垂直对齐方式; 将根据需要, 考虑 <i>firstlinedist</i> 和 <i>lastlinedist</i> 选项（默认值: <i>top</i> ）:
		<i>top</i> 格式化将从第一行开始, 向下继续。如果文本不能填满 <i>fitbox</i> , 文本下方将出现空白。
		<i>center</i> 文本在 <i>fitbox</i> 中垂直居中。如果文本不能填满 <i>fitbox</i> , 文本上方和下方都将出现空白。
		<i>bottom</i> 格式化将从最后一行开始, 向上继续。如果文本不能填满 <i>fitbox</i> , 文本上方将出现空白。
		<i>justify</i> 文本与 <i>fitbox</i> 的顶部和底部对齐。为了实现这种效果, 将增加行间距, 直至达到 <i>linespreadlimit</i> 指定的限制为止。如果 <i>firstlinedist=leading</i> , 将只增加第一行的高度。

**VB RB** Function `info_textflow(textflow As Long, keyword As String) As Double`  
**C#** double `info_textflow(int textflow, String keyword)`

查询文本流的当前状态。

**textflow** `create_textflow()` 调用返回的文本流句柄。

**keyword** 根据表 8.23 指定请求的信息的关键字。

返回值 *keyword* 请求的、某些文本流参数的值。即使在隐蔽模式下, 此参数也可返回正确的几何信息（不同于 *textx/texty* 参数）。

范围 文档、页面、图案、模板、字形

参数 *textx*、*texty*

表 8.23 `info_textflow()` 的关键字

关键字	说明
<i>boxlinecount</i>	最后一个 <i>fitbox</i> 中的行数。
<i>firstparalinecount</i>	<i>fitbox</i> 中第一个段落的行数。
<i>lastmark</i>	最后一个 <i>fitbox</i> 内已处理文本流部分中找到的最后一个标记的编号（可用 <i>mark</i> 选项设置标记）。
<i>leading</i>	<i>leading</i> 选项的当前值, 由文本流中的文本和选项确定。
<i>lastparalinecount</i>	<i>fitbox</i> 中最后一个段落的行数。
<i>leftlinex, leftliney, rightlinex, rightliney</i>	分别表示最近填充的 <i>fitbox</i> 内、从最左边开始的行和在最右边结束的行的 <i>x</i> 和 <i>y</i> 坐标。
<i>minlinelength, maxlinelength</i>	分别表示最近填充的 <i>fitbox</i> 中、最短和最长文本行的长度。



表 8.23 `info_textflow()` 的关键字（续）

关键字	说明
<i>minliney</i> <i>maxliney</i>	分别表示最近填充的 <i>fitbox</i> 中、最短和最长文本行的基线 <i>y</i> 坐标。
<i>remainchars</i>	（不常用）尚未处理的字符数。此计数不包括内嵌选项列表和字符引用中的字符数。由于各种文本替换过程（例如 <i>CR/NL</i> 组合），此值可能不可靠。
<i>textendx</i> , <i>textendy</i>	放置文本后，当前点的 <i>x</i> 或 <i>y</i> 坐标。
<i>used</i>	到目前为止，已放置的文本的百分比 ( <i>0...100</i> )。

**VB RB** `Sub delete_textflow(textflow As Long)`

**C#** `void delete_textflow(int textflow)`

删除文本流及其相关数据结构。

**textflow** `create_textflow()` 调用返回的文本流句柄。

**详细说明** 尚未使用此函数删除的文本流将在关闭文档范围结束之后自动删除。

**范围** 任意

文本流的内嵌选项列表 `create_textflow()` 的 *text* 参数中提供的内容可以包括任意数量的选项列表（内嵌选项），根据表 8.24 指定文本流选项。另外，也可以在 `create_textflow()` 的 *optlist* 参数中提供所有这些选项。在一个选项列表中，同一选项可指定多次；这时，将只考虑最后一次指定的选项。

内嵌选项列表必须使用 `winansi` 编码或被选择的 *Unicode* 格式（若 *fixedtextformat=true*），而且必须用 *begoptlistchar* 和 *endoptlistchar* 选项中指定的字符括起来（默认值：< 和 >）。很显然，如果实际文本中必须使用用来开始内嵌选项列表的字符，可能会引发冲突。有几种方法可以解决这个冲突；具体取决于文本是否包含任何内嵌选项列表。

如果文本不包含任何内嵌选项列表，可以通过下列一种方法完全停用搜索内嵌选项列表的操作：

- ▶ 在 `create_textflow()` 的 *optlist* 参数中，设置 *begoptlistchar=none*。
- ▶ 将 `create_textflow()` 的 *optlist* 参数中的 *textlen* 选项设为文本的长度。

如果文本确实包含内嵌选项列表，可通过下列一种方法避免文本内容与开始内嵌选项列表的 *begoptlistchar* 之间的冲突：

- ▶ 用对应的数字或字符实体引用（`&#x3C;` 或 `&lt;`）替换出现的所有 < 字符，或用文字 < 字符开始内嵌选项列表。
- ▶ 将 `create_textflow()` 的 *optlist* 中的 *begoptlistchar* 选项设为文本中没有使用的字符，并使用该字符开始内嵌选项列表。
- ▶ 在每个内嵌选项列表中，使用 *textlen* 选项指定下一文本片段的长度（直至下一内嵌选项列表开始为止）。有些情况下，实际上要求这种设置；例如：如果不能用 *Unicode* 解释内嵌选项列表间的某些文本片段等（有关详细信息，参见表 8.24 和第 97 页上的第 4.9.6 节“控制字符、字符映射和符号字体”）。

**注：** 如果直接在前一个选项列表之后提供了一个内嵌选项列表，将假定他们之间存在长度为 0 的文本片段。在第一个选项列表中提供 *textlen* 选项时，这一点很重要。

**文本流的宏选项** 文本流的选项列表（无论是文本中内嵌选项列表，或是在 `create_textflow()` 调用中直接提供选项列表）可以包含宏定义和宏调用；具体规则，参见表 8.25。宏可用来集中定义多次使用的选项值，如字体名称、缩进量等。解析选项列表之前，将用宏

表 8.24 `create_textflow()` 的 `optlist` 参数及文本中的内嵌选项列表的选项

选项	类型	说明
<b><code>load_font()</code></b> 的所有选项和参数（参见表 8.16）：		
<code>encoding</code>	字符串	（必须与 <code>fontname</code> 选项一起使用）编码的名称。
<code>fontname</code>	名称字符串	（必须与 <code>encoding</code> 选项一起使用）字体的名称；切记用 <code>{</code> 和 <code>}</code> 将包含空格字符的名称括起来。
<code>autocidfont</code> <code>autosubsetting</code> <code>embedding</code> <code>fontstyle</code> <code>fontwarning</code> <code>monospace</code> <code>subsetlimit</code> <code>subsetminsize</code> <code>subsetting</code> <code>unicodemap</code>		这些选项只在同时提供 <code>fontname</code> 和 <code>encoding</code> 选项的情况下使用。注意： <code>font-</code> 将用 <code>textwarning</code> 选项的值进行初始化。
<b><code>fit_textline()</code></b> 的所有外观选项（参见表 8.20）：		
<code>charref</code>	<code>font</code>	如果同时指定 <code>fontname</code> 和 <code>encoding</code> ，将忽略此选项。
<code>charspacing</code>	<code>fontsize</code>	要求使用此选项。
<code>dasharray</code> <code>fillcolor</code> <code>font</code> <code>fontsize</code> <code>glyphwarning</code> <code>horizscaling</code> <code>italicangle</code> <code>kerning</code> <code>overline</code> <code>strikeout</code> <code>strokecolor</code> <code>strokewidth</code> <code>textrendering</code> <code>textrise</code> <code>underline</code> <code>underlineposition</code> <code>underlinewidth</code> <code>wordspacing</code>		
控制嵌入选项列表的处理的选项：		
<code>begoptlistchar</code>	单字符或关键字	内嵌选项列表开头使用的字符。如果文本本身出现了默认字符，则替换该字符会很有用（参见第 185 页上的“文本流的内嵌选项列表”）。可使用关键字 <code>none</code> 完全停用搜索选项列表的操作。默认值： <code>U+003C (&lt;)</code>
<code>endoptlistchar</code>	单字符	内嵌选项列表结尾使用的字符。不允许使用字符 <code>U+007D (})</code> 。默认值： <code>U+003F (&gt;)</code>
<code>textlen</code>	整型或关键字	（使用不兼容 <code>Unicode</code> 的字体的文本要求此选项）数下一内嵌选项列表之前的字符数（另请参见第 185 页上的“文本流的内嵌选项列表”）。字符数在解析字符引用之前计算。例如： <code>&lt;textlen=8&gt;&amp;#x2460;&lt;...&gt;</code> 。默认值： <code>0</code>
文本语义的选项：		
<code>charmapping</code> <sup>1</sup>	列出两个单字符对或一个字符和一个整数	用一个或多个另一字符的实例替代具体字符。选项列表包含一对或多对单字符。每对中的第一个字符将由第二个字符替代。每对中的第二个元素可以是包含一个单字符和一个计数的选项列表，而非一一对应： <code>count &gt; 0</code> 此计数指示字符替代将重复的次数。 <code>count &lt; 0</code> 多个字符示例的序列将缩减到指定数字的绝对值。 <code>count = 0</code> 将删除字符。

表 8.24 `create_textflow()` 的 `optlist` 参数及文本中的内嵌选项列表的选项（续）

选项	类型	说明
<i>hyphenchar</i> <sup>1</sup>	单字符或关键字	在换行符处替代自动连字符的字符。值 <code>o</code> 和关键字 <code>none</code> 将完全禁用连字符。默认值：如果自动连字符 ( <code>U+00AD</code> ) 存在于字体中，默认值为 <code>U+00AD</code> ；否则，为 <code>U+002D</code> （连字符减号）。
<i>tabalignchar</i> <sup>1</sup>	单字符	将在此对齐小数制表符的字符。默认值： <code>U+002E</code> (.)
控制文本版面的选项：		
<i>alignment</i>	关键字	指定段落中各行的格式（默认值： <code>left</code> ）： <i>left</i> 左对齐，自 <code>leftindent</code> 处开始 <i>center</i> 在 <code>leftindent</code> 和 <code>rightindent</code> 之间居中 <i>right</i> 右对齐，至 <code>rightindent</code> 处止 <i>justify</i> 左右对齐
<i>avoidemptybegin</i>	布尔型	如果为 <code>true</code> ，将删除 <code>fitbox</code> 开头的空行。默认值： <code>false</code>
<i>fixedleading</i>	布尔型	如果为 <code>true</code> ，将使用每行中发现的第一个行距值。否则，将使用该行中最大的行距值。默认值： <code>false</code>
<i>horthabsize</i> <sup>1</sup>	浮点型或百分比	水平制表符的宽度 <sup>2</sup> 。具体解释取决于 <code>horthabmethod</code> 选项。默认值： <code>75%</code>
<i>horthabmethod</i> <sup>1</sup>	关键字	文本中水平制表符的处理方式。如果计算的位置在当前文本位置的左侧，将忽略制表符（默认值： <code>relative</code> ）： <i>relative</i> 位置将向前移动 <code>horthabsize</code> 中指定的量。 <i>typewriter</i> 位置将向前移动 <code>horthabsize</code> 的下一个倍数值。 <i>ruler</i> 位置将向前移动至 <code>ruler</code> 选项中的第 <code>n</code> 个制表符值；其中， <code>n</code> 等于迄今为止、行内发现的制表符数量。如果 <code>n</code> 大于制表符位置的数量，将应用 <code>relative</code> 方法。
<i>lastalignment</i>	关键字	段落中最后一行的格式化处理。支持 <code>alignment</code> 选项的所有关键字，以及下列关键字（默认值： <code>auto</code> ）： <i>auto</i> 除非为 <code>justify</code> ，否则使用 <code>alignment</code> 选项的值。如果为 <code>justify</code> ，将使用 <code>left</code> 。
<i>leading</i>	浮点型或百分比	相邻文本基线间的距离 <sup>3</sup> 。实际值将按以下方式确定：如果行的开头有选项列表，将根据最后一个相关选项（ <code>font</code> 、 <code>fontsize</code> 、 <code>leading</code> 等）确定行距。如果同一行内还有其他选项列表，只有当 <code>fixedleading=false</code> 时，才考虑其任何与行距相关的选项。如果行内没有选项列表，将考虑前一个行距值。默认值： <code>100%</code>
<i>minlinecount</i>	整型	<code>fitbox</code> 中最后一个段落的最小行数。如果行数小于该值，将置于下一 <code>fitbox</code> 。值 <code>2</code> 可用来防止在 <code>fitbox</code> 结尾处出现单行（» 孤行 «）。默认值： <code>1</code>
<i>parindent</i>	浮点型或百分比	段落第一行的左缩进 <sup>2</sup> 。将向 <code>leftindent</code> 添加此值。在行中指定此选项就相当于使用制表符。默认值： <code>o</code>
<i>rightindent</i> <i>leftindent</i>	浮点型或百分比	所有文本行的右缩进或左缩进 <sup>2</sup> 。如果在行内指定了 <code>leftindent</code> ，且确定的位置在当前文本位置左侧，将对当前行忽略此选项。默认值： <code>o</code>
<i>ruler</i> <sup>1</sup>	浮点值或百分比列表	<code>horthabmethod=ruler</code> 时的绝对制表符位置列表 <sup>2</sup> 。此列表最多可包含 <code>32</code> 个按升序排序的非负值条目。默认值： <code>horthabsize</code> 的整数倍数值
<i>tabalignment</i> <sup>1</sup>	关键字列表	制表位的对齐方式。列表中的每个条目均可定义 <code>ruler</code> 选项中对条目的对齐方式（默认值： <code>left</code> ） <i>center</i> 文本将在制表符位置居中。 <i>decimal</i> 将在制表符位置左对齐第一个 <code>tabalignchar</code> 的实例。如果没有找到 <code>tabalignchar</code> ，将使用右对齐。 <i>left</i> 文本将在制表符位置左对齐。 <i>right</i> 文本将在制表符位置右对齐。

表 8.24 `create_textflow()` 的 `optlist` 参数及文本中的内嵌选项列表的选项（续）

选项	类型	说明
控制换行算法的选项：		
<code>adjustmethod</code>	关键字	如果根据 <code>minspacing</code> 和 <code>maxspacing</code> 选项指定的限制压缩或扩展两个字之间的距离后，文本部分不能置于一行中，则使用此选项调整该行。 默认值： <code>auto</code> <code>auto</code> 依次应用下列方法： <code>shrink</code> 、 <code>spread</code> 、 <code>nofit</code> 和 <code>split</code> 。 <code>clip</code> 与 <code>nofit</code> 相同；但是，会剪裁 <code>fitbox</code> 右边缘处较长的部分（考虑 <code>rightindent</code> 选项）。 <code>nofit</code> 最后一个词将移至下一行；条件是，剩余（较短）行不短于 <code>nofitlimit</code> 选项中指定的百分比。即使经过对齐，段落仍可能略显不齐。 <code>shrink</code> 如果某个字不能置于行中，将根据 <code>shrinklimit</code> 压缩文本。如果仍无法置于行中，将应用 <code>nofit</code> 方法。 <code>split</code> 最后一个字不移至下一行，但将强制使用连字符连接。对于文本字体，将插入连字符；而符号字体，则不插入。 <code>spread</code> 最后一个字将移至下一行，剩余（较短）行将进行对齐，根据 <code>spreadlimit</code> 增加该字内各字符间的距离。如果仍无法实现对齐，将应用 <code>nofit</code> 方法。
<code>avoidbreak</code> <sup>1</sup>	布尔型	如果为 <code>true</code> ，将尝试避免任何换行符，直至 <code>avoidbreak</code> 重设为 <code>false</code> 为止。 默认值： <code>false</code>
<code>maxspacing</code> <sup>1</sup> <code>minspacing</code> <sup>1</sup>	浮点型或百分比	两个字之间的最大或最小距离（可使用用户坐标，或空格字符宽度的百分比）。计算的间距将受提供的值限制（但仍将添加 <code>wordspacing</code> 选项）。 默认值： <code>minspacing=50%</code> ， <code>maxspacing=500%</code>
<code>nofitlimit</code>	浮点型或百分比	使用 <code>nofit</code> 方法的行的长度下限 <sup>2</sup> 。默认值： <code>75%</code>
<code>shrinklimit</code>	百分比	使用 <code>shrink</code> 方法时的文本压缩下限；计算的收缩因子将受提供的值限制，但将乘以 <code>horizscaling</code> 选项的值。默认值： <code>85%</code>
<code>spreadlimit</code> <sup>1</sup>	浮点型或百分比	<code>spread</code> 方法中，两个字符之间的距离上限 <sup>3</sup> ；计算的值将与 <code>charspacing</code> 选项的值相加。默认值： <code>0</code>
用作命令的选项：		
<code>comment</code>	字符串	将被忽略的任意文本，对注释选项列表或宏很有用
<code>mark</code>	整型	作为标记，在内部存储提供的值。稍后，可使用 <code>info_textflow()</code> 调用最近一次存储的标记。这可能对确定文本的哪些部分已置于页面很有用。
<code>nextline</code> <code>nextparagraph</code>	布尔型	强制生成新行或新段落；即使使用不兼容 <code>Unicode</code> 的字体，也是如此。
<code>resetfont</code>	布尔型	将 <code>font</code> 和 <code>fontsize</code> 选项重设为它们以前的值。这可能对插入操作之后重设字体很有用，例如斜体文本。 <code>font</code> 选项优先于此选项。只有在第二次设置任何字体相关参数后，才能使用此命令；否则，此命令将被忽略。
<code>return</code>	字符串	退出 <code>create_textflow()</code> ，并返回提供的字符串。字符串不能用下划线 <code>_</code> 字符开头。
<code>space</code>	浮点型或百分比	文本位置将向前移动提供的值 <sup>3</sup> 。这同样适用于不兼容 <code>Unicode</code> 的字体。

1. 此选项不影响使用根据第 80 页上的第 4.5.6 节“`Unicode` 兼容的字体”、不兼容 `Unicode` 的字体的文本。

2. 使用用户坐标，或 `fitbox` 宽度的百分比。

3. 使用用户坐标，或字体大小的百分比。

定义中提供的对应选项列表的内容替代其中包含的每个宏。然后，会解析由此生成的选项列表。以下是两个宏的宏定义示例：

```
<macro {
    comment { The following macros are used as paragraph styles }
    H1 {fontname=Helvetica-Bold encoding=winansi fontsize=14 }
    body {fontname=Helvetica encoding=winansi fontsize=12 }
}>
```

这些宏在选项列表中的使用方式如下：

```
<&H1>Chapter 1
<&body>This chapter talks about...
```

宏的定义和使用应遵守下列规则：

- ▶ 宏的嵌套可达到任意深度（宏定义可以包含对其他宏的调用）
- ▶ 宏不能直接在定义他们的选项列表中使用。但是，可以在定义宏的选项列表结束之后，立即开始使用该宏的新选项列表。
- ▶ 宏名称不区分大小写。
- ▶ 未定义的宏将导致异常。
- ▶ 可随时重新定义宏。

表 8.25 *fit\_textflow()* 的选项列表宏定义和宏调用

选项	类型	说明
<i>macro</i>	成对的列表	每一对都分别描述一个宏的名称和定义；具体如下： <i>name</i> （字符串）稍后可用于宏调用的宏名称。稍后可重新定义已经定义的宏。将忽略特殊名称注释。 <i>suboptlist</i> 调用宏时，将实际替代宏名称的选项列表。将忽略前导空白和尾随空白。
<i>&amp;name</i>	-	将扩展具有指定名称的宏，并用宏的内容替代宏名称（包括 & 字符），即：已经为宏定义的子选项列表（两端没有大括号）。宏名称以空白、{、}、= 或 & 终结。因此，不能将这些字符用作宏名称的一部分。  将扩展嵌套的宏，且没有任何嵌套限制。同时，还将扩展字符串选项中的宏。宏替代必须生成有效的选项列表。
<i>comment</i>	字符串	将被忽略的任意文本，对宏注释很有用

## 8.4 图形函数

### 8.4.1 图形状态函数

在页面开头，所有图形状态参数都将还原为默认值。相关函数说明中记录了这些默认值。第 171 页上的第 8.3 节“文本函数”中列出了与文本状态相关的函数。

注：路径范围内不能使用任何图形状态函数（参见第 46 页上的第 3.2 节“页面说明”）。

```
VB RB Sub setdash(b As Double, w As Double)
C# void PDF_setdash(double b, double w)
```

设置当前虚线样式。

**b, w** 交替显示的黑白单位的数量。**b** 和 **w** 必须为正值数值。

详细说明 要生成实线，应设置 **b = w = 0**。在每页的开头，**dash** 参数将设为实线。

范围 页面、图案、模板、字形

**VB RB** Sub setdashpattern(optlist As String)  
**C#** void setdashpattern(String optlist)

设置选项列表定义的虚线样式。  
**optlist** 根据表 8.26 提供的选项列表。空列表将生成实线。

表 8.26 setdashpattern() 的选项

选项	类型	描述
<i>dasharray</i>	浮点值列表	一个由 2 至 8 个交替值组成的列表，指示虚线的长度和描边路径的间隔（使用用户坐标系表示）。数组值必须大于 0。这些值将循环使用，直至整个描边路径完成为止。
<i>dashphase</i>	浮点型	虚线样式之间的距离。默认值：0

详细说明 在每页的开头，dash 参数将设为实线。  
范围 页面、图案、模板、字形

**VB RB** Sub setflat(flatness As Double)  
**C#** void setflat(double flatness)

设置 flatness 参数。  
**flatness** 一个正值，描述路径之间的最大距离（以设备像素为单位）和根据直线段计算的近似值。




详细说明 在每页开头，flatness 参数设为默认值 1。  
范围 页面、图案、模板、字形

**VB RB** Sub setlinejoin(linejoin As Long)  
**C#** void setlinejoin(int linejoin)

设置 linejoin 参数。  
**linejoin** 指定描边路径拐角的形状，参见表 8.27。

详细说明 在每页开头，linejoin 参数设为默认值 0。  
范围 页面、图案、模板、字形

表 8.27 linejoin 参数的值


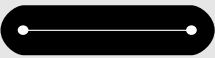

value	描述（摘自《PDF Reference》）	示例
0	斜角连接：两个线段的描边外缘继续延伸，直至相接为止。当斜角外延超出斜角极限时，将由斜削连接取代。	
1	圆角连接：以在线段相接的点为圆心，绘制直径等于线宽的圆弧，并填充圆弧，生成圆角。	
2	斜面连接：绘制两个具有平头端点的路径线段（参见有关 linecap 参数的讨论），用三角形填充生成的线段端点以外的切口。	

VB RB Sub setlinecap(linecap As Long)  
C# void setlinecap(int linecap)

设置 linecap 参数。  
linecap 对于描边，控制路径端点的形状，参见表 8.28。

详细说明 在每页开头，linecap 参数设为默认值 0。  
范围 页面、图案、模板、字形

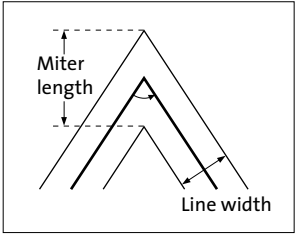
表 8.28 linecap 参数的值

value	描述（摘自《PDF Reference》）	示例
0	平头端点：描边在路径端点处切成方形。	
1	圆头端点：以端点为圆心，绘制直径等于线宽的半圆弧，并填充圆弧。	
2	方形突出终端点：描边延伸出线的端点（距离为线宽的一半），并以方形结束。	

VB RB Sub setmiterlimit(miter As Double)  
C# void setmiterlimit(double miter)

设置斜接限制。  
miter 一个大于或等于 1 的值，控制斜接连接生成的尖端。

详细说明 如果 linejoin 参数设为 0（斜角连接），以小角度连接的两个线段将生成很尖锐的尖端。斜接长度和线宽的比例超出斜接限制时，将用垂直端点替代尖端（即，斜接连接将改为斜面连接）。在每页开头，斜接限制设为默认值 10。这大约相当于 11.5 度角。



范围 页面、图案、模板、字形

VB RB Sub setlinewidth(width As Double)  
C# void setlinewidth(double width)

设置当前线宽。  
width 以当前用户坐标系统的单位设置的线宽。

详细说明 在每页开头，width 参数设为默认值 1。  
范围 页面、图案、模板、字形

<b>VB RB</b>	<b>Sub initgraphics</b>
<b>C#</b>	<b>void initgraphics()</b>
将所有颜色和图形状态参数重设为他们的默认值。	
详细说明	<b>color</b> 、 <b>linewidth</b> 、 <b>linecap</b> 、 <b>linejoin</b> 、 <b>miterlimit</b> 、 <b>dash</b> 参数及当前转换矩阵（但不包括文本状态参数）都重设为他们各自的默认值。不影响当前剪切路径。 在程序流不允许方便地使用 <i>save()/restore()</i> 时，此函数可能会很有用。
范围	页面、图案、模板、字形

8.4.2 保存和恢复图形状态

<b>VB RB</b>	<b>Sub save()</b>
<b>C#</b>	<b>void save()</b>
将当前图形状态保存到一个堆栈。	
详细说明	图形状态包含控制各种类型的图形对象的参数。PDF 不要求保存图形状态；只有应用程序希望稍后返回某些特定图形状态（例如，自定义坐标系），而又不想再次显式设置所有相关参数时，才需要保存图形状态。下列项目受 <b>save/restore</b> 控制： <ul style="list-style-type: none"> <li>已经使用相应函数设置的图形参数：<b>剪切路径</b>、<b>坐标系</b>、<b>当前点</b>、<b>平滑度 (flatness)</b>、<b>线条端点样式</b>、<b>虚线样式</b>、<b>线条连接样式 (line join style)</b>、<b>线宽</b>、<b>斜接限制</b>；</li> <li><b>颜色参数</b>：<b>填充色</b>和<b>描边色</b>；</li> <li>已经使用 <i>set_gstate()</i> 和显式图形状态设置的图形参数（参见第 194 页上的第 8.4.4 节“显式图形状态”）；</li> <li><b>文本位置</b>及大多数与文本相关的参数；参见以下列表；</li> <li>某些 <b>PDFlib</b> 参数；参见以下列表。</li> </ul> <p><i>save()</i> 和 <i>restore()</i> 可以嵌套成对。尽管 PDF 规范没有限制 <b>save/restore</b> 对的嵌套深度，但应用程序必须保持嵌套深度小于 26，以避免由 PDF 浏览器所生成的带有限制的 PostScript 输出导致的打印问题，并允许 <b>PDFlib</b> 内部要求的额外 <b>save</b> 级别。</p>
范围	页面、图案、模板、字形；必须与对应的 <i>restore()</i> 调用成对使用。在每个页面、图案、模板和字形描述中， <i>save()</i> 和 <i>restore()</i> 调用必须成对出现。
参数	下列参数受 <b>save/restore</b> 控制： <i>charspacing</i> 、 <i>wordspacing</i> 、 <i>horizscaling</i> 、 <i>italicangle</i> 、 <i>leading</i> 、 <i>font</i> 、 <i>fontsize</i> 、 <i>textrendering</i> 、 <i>textrise</i> ； 下列参数不受 <b>save/restore</b> 控制： <i>fillrule</i> 、 <i>kerning</i> 、 <i>underline</i> 、 <i>overline</i> 、 <i>strikeout</i> 。

<b>VB RB</b>	<b>Sub restore()</b>
<b>C#</b>	<b>void restore()</b>
从堆栈中恢复最近一次保存的图形状态。	
详细说明	必须是已经在同一页面、图案或模板上保存了的相应的图形状态。
范围	页面、图案、模板、字形；必须与对应的 <i>save()</i> 调用成对使用。在每个页面、图案、模板和字形描述中， <i>save()</i> 和 <i>restore()</i> 调用必须成对出现。



8.4.3 坐标系统转换函数

所有转换函数（*translate()*、*scale()*、*rotate()*、*skew()*、*concat()*、*setmatrix()* 和 *initgraphics()*）都可更改绘制后续对象使用的坐标系统。他们不会对页面上的现有对象产生任何影响。

VB RB

Sub translate(tx As Double, ty As Double)

C#

void translate(double tx, double ty)

转换坐标系的原点。  
**tx, ty** 坐标系统的新原点为旧坐标系统中的 (tx, ty) 点。

范围 页面、图案、模板、字形

VB RB

Sub scale(sx As Double, sy As Double)

C#

void scale(double sx, double sy)

缩放坐标系统。  
**sx, sy** x 和 y 方向上的缩放因子。

详细说明 此函数可按 **sx** 和 **sy** 值缩放坐标系统。通过使用负值缩放因子，还可使用此函数实现对称（镜像）效果。在新坐标系统中，x 方向上的一个单位等于旧坐标系统中 x 方向上的 **sx** 个单位，y 坐标同样如此。

范围 页面、图案、模板、字形  
绑定 COM: 此函数也可通过名称 *pscale* 使用，以绕过 VB 中的错误。

VB RB

Sub rotate(phi As Double)

C#

void rotate(double phi)

旋转坐标系统。  
**phi** 旋转角度的度数。

详细说明 自当前坐标系统的正 x 轴起，按逆时针方向测量角度。将旧坐标轴旋转 *phi* 指示的度数，计算新坐标轴。  
范围 页面、图案、模板、字形

VB RB

Sub skew(alpha As Double, beta As Double)

C#

void skew(double alpha, double beta)

歪斜坐标系统。  
**alpha, beta** x 和 y 方向上的歪斜角度数。

详细说明 歪斜（或剪切）会在 x 和 y 方向上按指定的角度扭曲坐标系统。自当前坐标系统的正 x 轴起，按逆时针方向测量 *alpha*；自正 y 坐标起，按顺时针方向测量 *beta*。两个角都必须在 -360° < *alpha, beta* < 360° 范围内，且必须不同于 -270°、-90°、90° 和 270°。  
范围 页面、图案、模板、字形

**VB RB** Sub concat(a As Double, b As Double, c As Double, d As Double, e As Double, f As Double)  
**C#** void concat(double a, double b, double c, double d, double e, double f)

将一个矩阵连接到当前转换矩阵。  
**a, b, c, d, e, f** 转换矩阵的元素。这六个值以 PostScript 和 PDF 中相同的方式构成坐标系统（参见相关参考）。为了避免转换退化， $a*d$  不能等于  $b*c$ 。

**详细说明** 此函数可将矩阵连接到文本和图形的当前转换矩阵 (CTM)。它允许大多数常规的转换形式。使用此函数需对转换矩阵很熟悉，否则建议另用 `translate()`、`scale()`、`rotate()` 和 `skew()` 函数。在每页开头，CTM 重设为单位矩阵  $[1, 0, 0, 1, 0, 0]$ 。

**范围** 页面、图案、模板、字形

**VB RB** Sub setmatrix(a As Double, b As Double, c As Double, d As Double, e As Double, f As Double)  
**C#** void setmatrix(double a, double b, double c, double d, double e, double f)

明确地设置当前转换矩阵。  
**a, b, c, d, e, f** 参见 `concat()`。

**详细说明** 此函数类似于 `concat()`。但是，它会废弃当前转换矩阵，并用新的矩阵完全替代旧矩阵。

**范围** 页面、图案、模板、字形

8.4.4 显式图形状态

**VB RB** Function create\_gstate(optlist As String) As Long  
**C#** int create\_gstate(String optlist)

根据各种选项创建图形状态对象。  
**optlist** 选项列表，根据表 8.29 包含图形状态的选项。

**返回值** gstate 句柄；在关闭的文档范围中，可在后续 `set_gstate()` 调用中使用此句柄。

**详细说明** 此选项列表可以包含任何数量的图形状态参数。并非所有参数都可用于所有 PDF 版本。表中列出了要求的最低 PDF 版本。

**范围** 文档、页面、图案、模板、字形

**PDF/X** 除非具有值 1，否则必须避免使用 `opacityfill` 和 `opacitystroke` 选项。

表 8.29 `create_gstate()` 的选项

关键字	类型	说明和可能的值
<code>alphaishshape</code>	布尔型	(PDF 1.4) 此值为真 ( <i>true</i> ) 则 <i>Alpha</i> 的数据描述形状，若为假 ( <i>false</i> ) 则描述不透明度。默认值: <i>false</i>
<code>blendmode</code>	关键字列表	(PDF 1.4) 混合模式的名称。可指定多个混合模式。可能的值: <i>None</i> 、 <i>Normal</i> 、 <i>Multiply</i> 、 <i>Screen</i> 、 <i>Overlay</i> 、 <i>Darken</i> 、 <i>Lighten</i> 、 <i>ColorDodge</i> 、 <i>ColorBurn</i> 、 <i>HardLight</i> 、 <i>SoftLight</i> 、 <i>Difference</i> 、 <i>Exclusion</i> 、 <i>Hue</i> 、 <i>Saturation</i> 、 <i>Color</i> 、 <i>Luminosity</i> 。默认值: <i>None</i>
<code>flatness</code>	浮点型	路径与其近似值之间的最大距离（参见 <code>setflat()</code> ），必须 > 0
<code>linecap</code>	整型或关键字	路径结尾处的形状（参见 <code>setlinecap()</code> ），必须为 0、1 或 2

表 8.29 create\_gstate() 的选项

关键字	类型	说明和可能的值
linejoin	整型或关键字	路径拐角处的形状（参见 setlinejoin()），必须为 0、1 或 2
linewidth	浮点型	线宽（参见 setlinewidth()），必须 > 0
miterlimit	浮点型	控制斜接连接生成的尖角，必须 >= 1（参见 setmiterlimit()）
opacityfill	浮点型	(PDF 1.4) 用于填充操作的 alpha 常数，必须 >= 0 并且 <= 1
opacitystroke	浮点型	(PDF 1.4) 用于描边操作的 alpha 常数，必须 >= 0 并且 <= 1
overprintfill	布尔型	压印除描边之外的其他操作。默认值: false
overprintmode	整型	压印模式。0（默认值）表示，每个颜色组件均替代以前放置的标记；模式 1（在 Acrobat 中，称 » 压印缺省值是非零压印 «）表示，颜色组件值为 0 时将不对相应的组件做任何更改。
overprintstroke	布尔型	描边操作的压印。默认值: false
renderingintent	关键字	色域压缩使用的颜色渲染方法；可能的值: Auto、AbsoluteColorimetric、RelativeColorimetric、Saturation、Perceptual
smoothness	浮点型	着色线性插值的最大误差，必须 >= 0 并且 <= 1
strokeadjust	布尔型	是否应用自动描边调整。默认值: false
textknockout	布尔型	(PDF 1.4) 对于复合操作，文本对象中的字形将视为单独对象 (false) 或单一对象 (true)。默认值: true

VB RB Sub set\_gstate(gstate As Long)

C# void set\_gstate(int gstate)

启动图形状态对象。

gstate 由 create\_gstate() 生成的的图形状态对象的句柄。

详细说明 将设置图形状态对象中包含的所有选项。多次调用此函数时，会累积图形状态选项。未在图形状态对象中明确设置的选项将保留各自的当前值。在页面开头，所有图形状态选项都将重设为默认值。

范围 页面、图案、模板、字形

8.4.5 路径构建

表 8.30 列出了本节的相关参数及其相应值。

表 8.30 路径段函数的参数和值（参见第 168 页上的第 8.2.3 节“参数处理”）

函数	关键字	说明
get_value	currentx currenty	分别指示当前点的 x 或 y 坐标（以当前坐标系统的单位表示）。范围：页面、图案、模板、路径
get_value	ctm_a ctm_b ctm_c ctm_d ctm_e ctm_f	矢量图形的当前转换矩阵 (CTM) 的要素。范围：页面、图案、模板、路径

注：使用本节介绍的函数之后，一定要调用第 198 页上的第 8.4.6 节“路径绘制和剪切”中的一个函数；否则，构建的路径不会有任何效果，而且后续操作也可能会引发 PDFlib 异常。

---

**VB RB** Sub moveto(*x* As Double, *y* As Double)  
**C#** void moveto(double *x*, double *y*)

---

设置图形输出的当前点。  
**x, y** 新当前点的坐标。

详细说明 在每页开头，当前点设为 *undefined* 的默认值。图形的当前点和当前文本位置将分别维护。  
范围 页面、图案、模板、路径、字形；此函数是路径范围的开头。  
参数 *currentx*, *currenty*

---

**VB RB** Sub lineto(*x* As Double, *y* As Double)  
**C#** void lineto(double *x*, double *y*)

---

从当前点至另一点，绘制一条直线。  
**x, y** 直线第二个端点的坐标。

详细说明 此函数将在当前路径上添加一条自当前点至 (*x, y*) 的直线。使用此函数之前，必须先设置当前点。点 (*x, y*) 将成为新的当前点。  
绘制的直线将围绕 » 理想 « 直线居中放置，即：在连接两个端点的直线两侧分别绘制线宽的一半（根据 *linewidth* 参数的值确定）。端点处的行为由 *linecap* 参数的值设置。  
范围 路径  
参数 *currentx*, *currenty*

---

**VB RB** Sub curveto(*x1* As Double, *y1* As Double, *x2* As Double, *y2* As Double, *x3* As Double, *y3* As Double)  
**C#** void curveto(double *x1*, double *y1*, double *x2*, double *y2*, double *x3*, double *y3*)

---

使用另外三个控制点，自当前点起，绘制 Bézier 曲线。  
**x1, y1, x2, y2, x3, y3** 三个控制点的坐标。

详细说明 将 (*x1, y1*) 和 (*x2, y2*) 用作控制点，在当前路径上添加一条从当前点至 (*x3, y3*) 的 Bézier 曲线。使用此函数之前，必须先设置当前点。该曲线的端点 (*x3, y3*) 将成为新的当前点。  
范围 路径  
参数 *currentx*, *currenty*

---

**VB RB** Sub circle(*x* As Double, *y* As Double, *r* As Double)  
**C#** void circle(double *x*, double *y*, double *r*)

---

绘制一个圆形。  
**x, y** 圆形原点的坐标。  
**r** 圆形的半径。

详细说明 此函数作为完全子路径，在当前路径上添加一个圆形。点 (*x + r, y*) 将成为新的当前点。生成的形状将为使用用户坐标表示的圆形。如果坐标系统在 *x* 和 *y* 方向上的缩放比例不同，生成的曲线将为椭圆形。

范围	页面、图案、模板、路径、字形；此函数是路径范围的开头。
参数	<i>currentx, currenty</i>
绑定	COM: 此函数也可通过名称 <i>pcircle</i> 使用，以绕过 VB 中的错误。
<b>VB RB</b>	<b>Sub arc(x As Double, y As Double, r As Double, alpha As Double, beta As Double)</b>
<b>C#</b>	<b>void arc(double x, double y, double r, double alpha, double beta)</b>
	绘制一个逆时针圆弧线段。
	<b>x, y</b> 圆弧线段中心点的坐标。
	<b>r</b> 圆弧线段的半径。 <i>r</i> 必须为非负值。
	<b>alpha, beta</b> 圆弧线段开头和结尾角度的度数。
详细说明	<p>此函数将在当前路径上，按逆时针方向添加一条圆弧线段，自 <i>alpha</i> 角度延伸至 <i>beta</i> 角度。对于 <i>arc()</i> 和 <i>arcn()</i>，都自当前坐标系统的正 <i>x</i> 轴起，按逆时针方向测量角度。如果存在当前点，将自当前点起至圆弧的起始点止，绘制另一条直线。该圆弧的端点将成为新的当前点。</p> <p>圆弧线段将为使用用户坐标表示的圆形。如果坐标系统在 <i>x</i> 和 <i>y</i> 方向上的缩放比例不同，生成的曲线将为椭圆形。</p>
范围	页面、图案、模板、路径、字形；此函数是路径范围的开头。
参数	<i>currentx, currenty</i>
<b>VB RB</b>	<b>Sub arcn(x As Double, y As Double, r As Double, alpha As Double, beta As Double)</b>
<b>C#</b>	<b>void arcn(double x, double y, double r, double alpha, double beta)</b>
	绘制一个顺时针圆弧线段。
详细说明	除绘制方向外，此函数的行为与 <i>arc()</i> 完全相同。尤其是，仍将自正 <i>x</i> 轴起，按逆时针方向测量角度。
<b>VB RB</b>	<b>Sub rect(x As Double, y As Double, width As Double, height As Double)</b>
<b>C#</b>	<b>void rect(double x, double y, double width, double height)</b>
	绘制一个矩形。
	<b>x, y</b> 矩形左下角的坐标。
	<b>width, height</b> 矩形的大小。
详细说明	<p>此函数作为完全子路径，在当前路径上添加一个矩形。使用此函数前，不要求设置当前点。点 (<i>x, y</i>) 将成为新的当前点。绘制的直线段将围绕 » 理想 « 直线居中放置，即：在连接各个端点的直线两侧分别绘制线宽的一半（根据 <i>linewidth</i> 参数的值确定）。</p>
范围	页面、图案、模板、路径、字形；此函数是路径范围的开头。
参数	<i>currentx, currenty</i>

**VB RB** Sub closepath()  
**C#** void closepath()

关闭当前路径。

详细说明 此函数将关闭当前子路径，即：添加一条自当前点起、至子路径起始点止的直线。

范围 路径

参数 *currentx, currenty*

8.4.6 路径绘制和剪切

表 8.31 列出了本节的相关参数及相应的值。

表 8.31 路径绘制和剪切函数的参数和值

函数	关键字	说明
<i>set_parameter</i>	<i>fillrule</i>	将当前填充规则设为 <i>winding</i> 或 <i>evenodd</i> 。PDF 查看程序将使用此填充规则确定形状的内部，以进行填充或剪切。因为对于简单形状而言，这两种算法将产生相同的结果，所以大多数应用程序都无需更改填充规则。在每页开头，填充规则将重设为默认的 <i>winding</i> 。范围：页面、图案、模板、字形。

注：本节内的大多数函数都将清除路径，而不定义当前点。因此，调用其中一个函数之后，后续的绘制操作都必须明确设置当前点（例如，使用 *moveto()*）。

**VB RB** Sub stroke()  
**C#** void stroke()

使用当前线宽和当前描边色描边路径，并清除路径。

范围 路径；此函数终结路径范围。

**VB RB** Sub closepath\_stroke()  
**C#** void closepath\_stroke()

关闭路径，并描边。

详细说明 此函数关闭当前子路径（添加一条自当前点起、至路径起点止的直线段）并使用当前线宽和当前描边色描边整个当前路径。

范围 路径；此函数终结路径范围。

**VB RB** Sub fill()  
**C#** void fill()

使用当前填充色填充路径的内部。

详细说明 此函数使用当前填充色填充当前路径的内部。路径的内部可由其中一种算法确定（参见 *fillrule* 参数）。填充之前，会自动关闭仍开放的路径。

范围 路径；此函数终结路径范围。

参数 *fillrule*

<b>VB RB</b>	<b>Sub fill_stroke()</b>
<b>C#</b>	<b>void fill_stroke()</b>

使用当前填充色和描边色，填充并描边路径。

范围 路径；此函数终结路径范围。

参数 *fillrule*

<b>VB RB</b>	<b>Sub closepath_fill_stroke()</b>
<b>C#</b>	<b>void closepath_fill_stroke()</b>

关闭路径，填充并描边路径。

详细说明 此函数关闭当前子路径（添加一条自当前点起、至路径起点止的直线段），然后填充并描边整个当前路径。

范围 路径；此函数终结路径范围。

参数 *fillrule*

<b>VB RB</b>	<b>Sub clip()</b>
<b>C#</b>	<b>void clip()</b>

将当前路径用作剪切路径并终结路径。

详细说明 此函数将当前路径与当前剪贴路径的交叉部分用作后续操作的剪切路径。在每页开头，剪切路径设为页面大小的默认值。剪切路径受 *save()/restore()* 影响。只能通过 *save()/restore()* 的方式放大剪切路径。

范围 路径；此函数终结路径范围。

<b>VB RB</b>	<b>Sub endpath()</b>
<b>C#</b>	<b>void endpath()</b>

结束当前路径，既不填充也不描边。

详细说明 此函数不会在页面上显式任何效果。它会在页面上生成不可见的路径。

范围 路径；此函数终结路径范围。

8.4.7 图层参数

<b>VB RB</b>	<b>Function define_layer(name As String, optlist As String) As Long</b>
<b>C#</b>	<b>int define_layer(String name, String optlist)</b>

创建新的图层定义（要求 PDF 1.5）。  
**name** （超文本字符串）图层的名称。

**optlist** 选项列表，根据表 8.32 指定图层设置。

返回值 图层句柄，可用于 *begin\_layer()* 和 *set\_layer\_dependency()* 调用，直至关闭文档范围结束为止。

详细说明 如果定义了图层，但未在文档中使用，PDFlib 将发出警告。

范围 文档、页面

表 8.32 *define\_layer()* 的选项

选项	类型	说明
<i>creatorinfo</i>	选项列表	描述内容和创建应用程序的选项列表。如果使用此选项，要求同时提供以下两个条目： <i>creator</i> (超文本字符串) 创建图层的应用程序名称 <i>subtype</i> (字符串) 内容的类型。建议使用 <i>Artwork</i> 和 <i>Technical</i> 值。
<i>defaultstate</i>	布尔型	默认值: <i>true</i>
<i>initialexportstate</i>	布尔型	指定图层的建议导出状态。如果为 <i>true</i> ， <i>Acrobat</i> 将在转换 / 导出到旧 PDF 版本或其他文档格式时包括图层。默认值: <i>true</i>
<i>initialprintstate</i>	布尔型	图层的建议打印状态。如果为 <i>true</i> ， <i>Acrobat</i> 将在打印文档时包括图层。默认值: <i>true</i>
<i>initialviewstate</i>	布尔型	图层的建议查看状态。如为 <i>true</i> ， <i>Acrobat</i> 将在打开文档时显示图层。默认值: <i>true</i>
<i>intent</i>	关键字	查看或设计
<i>language</i>	选项列表	指定图层的语言： <i>lang</i> (字符串，必需的) 在表 8.5 中描述的 <i>lang</i> 选项的语言和可能的区域设置。 <i>preferred</i> (布尔型) 如果为 <i>true</i> ，且图层语言只部分匹配系统语言，将使用此图层。默认值: <i>false</i>
<i>onpanel</i>	布尔型	如果为 <i>false</i> ，将不在 <i>Acrobat</i> 的图层面板显示图层名称，因此用户不能操作图层。默认值: <i>true</i>
<i>pageelement</i>	关键字	指定图层包含分页伪像: <i>HF</i> (页眉 / 页脚)、 <i>FG</i> (前景图像或图形)、 <i>BG</i> (背景图像或图形) 或 <i>L</i> (徽标) 之一。
<i>printssubtype</i>	选项列表	指定图层是否用于打印： <i>subtype</i> (关键字) <i>Trapping</i> 、 <i>PrintersMarks</i> 或 <i>Watermark</i> 之一，指定图层内容的类型。 <i>printstate</i> (布尔型) 如果为 <i>true</i> ， <i>Acrobat</i> 将在打印时启动图层内容。
<i>zoom</i>	浮点值或百分比列表	一个或两个值，根据缩放因子指定图层的可见性 ( <i>1.0</i> 表示缩放因子为 <i>100%</i> )。如果只提供一个值，将用作图层显示时使用的最大缩放因子；如果提供两个值，则指定最大和最小缩放因子。可使用关键字 <i>maxzoom</i> ，指定可能的最大缩放因子。

**VB RB** Sub *set\_layer\_dependency*(type As String, optlist As String)

**C#** void *set\_layer\_dependency*(String type, String optlist)

定义图层间的层次和分组关系 (要求 PDF 1.5)。

**type** 依存关系的类型，必须为下列值之一：

- ▶ *GroupAllOn*: 如果 *group* 选项中指定的所有图层都可以显示，将显示 *depend* 选项中指定的图层。
- ▶ *GroupAnyOn*: 如果 *group* 选项中指定的任一图层可以显示，将显示 *depend* 选项中指定的图层。



- ▶ **GroupAllOff**: 如果 *group* 选项中指定的所有图层都不显示, 将显示 *depend* 选项中指定的图层。
- ▶ **GroupAnyOff**: 如果 *group* 选项中指定的任一图层不显示, 将显示 *depend* 选项中指定的图层。
- ▶ **Parent**: 指定 *parent* 选项中指定的图层与 *children* 选项中指定的图层之间的层次关系。一个图层只能属于一个父级图层。
- ▶ **Radiobtn**: 指定 *group* 选项中指定的各个图层间的单选按钮关系。这表示, 一次最多只能显示该组中的一个图层; 这对多语言图层很有用。
- ▶ **Title**: *parent* 选项中指定的图层句柄不能直接控制任何页面内容, 但可以用作 *children* 选项中指定的图层的父级图层节点。

**optlist** 选项列表, 根据表 8.33 指定图层依存关系。

范围 文档、页面

表 8.33 *set\_layer\_dependency()* 的选项

选项	类型	说明
<i>parent</i>	图层句柄	(仅用于 <i>type=Parent</i> 和 <i>Title</i> 的情况) <i>children</i> 选项中指定的图层的父级图层。
<i>children</i>	图层句柄列表	(仅用于 <i>type=Parent</i> 和 <i>Title</i> 的情况) 一个或多个图层句柄, 指定从属于提供的父级图层的图层。
<i>depend</i>	图层句柄	(仅用于 <i>type=GroupAllOn</i> 、 <i>GroupAnyOn</i> 、 <i>GroupAllOff</i> 和 <i>GroupAnyOff</i> 的情况) 由 <i>group</i> 选项中指定的图层控制的图层。
<i>group</i>	图层句柄列表	(仅用于 <i>type=GroupAllOn</i> 、 <i>GroupAnyOn</i> 、 <i>GroupAllOff</i> 、 <i>GroupAnyOff</i> 和 <i>Radiobtn</i> 的情况) 构成图层组的一个或多个图层。

**VB RB** Sub begin\_layer(layer As Long)

**C#** void begin\_layer(int layer)

开始一个图层, 用于后续的页面输出 (要求 PDF 1.5)。

**layer** 图层的句柄, 必须用 *define\_layer()* 返回。

在此调用之后、但在任何后续 *begin\_layer()* 或 *end\_layer()* 调用之前置于页面上的所有内容都将构成指定图层的一部分。内容的可见性取决于图层的设置。

此函数将启动指定图层, 并停用任何当前现用图层。

范围 页面

**VB RB** Sub end\_layer()

**C#** void end\_layer()

停用所有活动图层 (要求 PDF 1.5)。

此调用之后置于页面上的内容不属于任何图层。所有图层都必须在页面结尾处结束。若想调换层 A 与层 B 的位置, 只需调用 *begin\_layer()* 即可。并不需要专门调用 *end\_layer()* 来关闭层 A。 *end\_layer()* 只在创建无条件内容时 (总是可见的), 和在页尾关闭所有层时才使用。

范围 页面

# 8.5 颜色函数

## 8.5.1 设置颜色和色彩空间

表 8.34 列出了本节的相关参数及相应的值。

表 8.34 颜色函数的参数

函数	关键字	说明
<code>set_parameter</code>	<code>preserveoldpantonenames</code>	如果为 <i>false</i> ，旧版 <i>Pantone</i> 专色名称将转换为对应的新颜色名称；否则，将保留旧名称。默认值: <i>false</i> 。范围: 任意
<code>set_parameter</code>	<code>spotcolorlookup</code>	如果为 <i>false</i> ， <i>PDFlib</i> 将不使用它的内部专色名称数据库。可以使用此参数提供已知专色的自定义，这将为匹配其他应用程序提供的定义提供了变通方式。使用此功能时要小心，建议不使用此功能。默认值: <i>true</i> 。范围: 任意

VB RB

Sub setcolor(fstype As String, colorspace as String,  
                  c1 As Double, c2 As Double, c3 As Double, c4 As Double)

C#

void setcolor(String fstype, String colorspace, double c1, double c2, double c3, double c4)

设置当前色彩空间和颜色。

**fstype** *fill*、*stroke* 或 *fillstroke* 之一，指定是否将颜色设为填充、描边，或同时设为填充和描边。

**colorspace** *gray*、*rgb*、*cmymk*、*spot*、*pattern*、*iccbasedgray*、*iccbasedrgb*、*iccbasedcmymk* 或 *lab* 之一，指定色彩空间。

- c1, c2, c3, c4** 所选色彩空间的颜色组件（参见第 50 页上的第 3.3 节“使用颜色”）：
- ▶ 如果 *colorspace* 为 *gray*，*c1* 指定灰色值；
  - ▶ 如果 *colorspace* 为 *rgb*，*c1*、*c2* 和 *c3* 分别指定红色、绿色和蓝色值；
  - ▶ 如果 *colorspace* 为 *cmymk*，*c1*、*c2*、*c3* 和 *c4* 分别指定青色、洋红、黄色和黑色值；
  - ▶ 如果 *colorspace* 为 *spot*，*c1* 指定 *makespotcolor()* 返回的专色句柄，*c2* 指定介于 0 和 1 之间的色调值。
  - ▶ 如果 *colorspace* 为 *pattern*，*c1* 指定 *begin\_pattern()* 或 *shading\_pattern()* 返回的图案句柄。
  - ▶ 如果 *colorspace* 为 *iccbasedgray*，*c1* 指定灰色值；
  - ▶ 如果 *colorspace* 为 *iccbasedrgb*，*c1*、*c2* 和 *c3* 分别指定红色、绿色和蓝色值；
  - ▶ 如果 *colorspace* 为 *iccbasedcmymk*，*c1*、*c2*、*c3* 和 *c4* 分别指定青色、洋红、黄色和黑色值；
  - ▶ 如果 *colorspace* 为 *lab*，*c1*、*c2* 和 *c3* 分别指定 CIE L\*a\*b\* 色彩空间中的颜色值，用 D50 标准光源解释。*c1* 指定介于 0 到 100 之间的 L\*（亮度），*c2* 和 *c3* 指定介于 -127 到 128 之间的 a\*、b\*（色度）值。

详细说明 *gray*、*rgb* 和 *cmymk* 色彩空间的所有颜色值以及 *spot* 色彩空间的 *tint* 值都必须为 0 到 1 之间的数值，包括 0 和 1。不使用的参数应设为 0。

灰度、RGB 值和专色色调根据加色配色法解释，即：0 表示没有颜色，1 表示完全强度。因此，灰色值 0 和 (r, g, b) = (0, 0, 0) 的 RGB 值代表黑色，灰色值 1 和 (r, g, b) = (1, 1, 1) 的 RGB 值代表白色。但是，CMYK 值会根据减色配色法解释，即：(c, m, y, k) = (0, 0, 0, 0) 代表白色，(c, m, y, k) = (0, 0, 0, 1) 代表黑色。介于 0 到 255 之间的颜色值必须除以 255，计算出介于 0 到 1 之间的值。

在每页开头，*gray*、*rgb* 和 *cmymk* 色彩空间的填充和描边颜色值设为黑色的默认值。专色和图案颜色没有默认值。

如果使用 *iccbasedgray/rgb/cmyk* 色彩空间，使用 *setcolor:iccprofilegray/rgb/cmyk* 参数之前，必须先设置对应的 ICC 配置文件（参见表 8.36）。

范围 页面、图案（仅限于图案的绘画类型为 1 的情况）、模板、字形（仅限于 **Type 3** 字体的 *colored* 选项为 **true** 的情况）和文档；图案颜色不能在自己的定义中使用。在文档范围中设置颜色可能对用 *makespotcolor()* 定义专色很有用。

PDF/X PDF/X-1 和 PDF/X-1a：不允许 *colorspace = rgb*、*iccbasedgray/rgb/cmyk* 和 *lab*。  
PDF/X-3：除非 PDF/X 输出方法为灰度或 CMYK 设备，否则 *colorspace = gray* 要求事先设置 *begin\_page\_ext()* 中的 *defaultgray* 选项。除非 PDF/X 输出方法为 RGB 设备，否则 *colorspace = rgb* 要求事先设置 *begin\_page\_ext()* 中的 *defaultrgb* 选项。除非 PDF/X 输出方法为 CMYK 设备，否则 *colorspace = cmyk* 要求事先设置 *begin\_page\_ext()* 中的 *defaultcmyk* 选项。使用 *iccbasedgray/rgb/cmyk* 和 *lab* 颜色要求在输出方法中的提供 ICC 配置文件（在这种情况下，标准名称是不够的）。

参数 *setcolor:iccprofilegray/rgb/cmyk*

**VB RB Function makespotcolor(spotname As String) As Long**

**C# int makespotcolor(String spotname)**

查找内建专色名称，或通过当前填充色创建命名专色。

**spotname** 内建专色的名称，或要定义的专色的任意名称。此名称最长不得超过 126 字节。专色名称只支持单字节（8 位）字符，不支持 Unicode 或嵌入的 null 字符。

返回值 颜色句柄，可用于整个文档中的后续 *setcolor()* 调用。可在所有页面之间重复使用专色句柄，但不能跨越不同文档。在一个文档中，专色的数量不受任何限制。

详细说明 如果 *spotname* 在内部颜色表中是已知的，并且 *spotcolorlookup* 参数为 **true**（这是默认值），将使用提供的专色名称。否则，将使用当前填充色的（CMYK 或其他）颜色值定义新专色的外观。这些替代值只用于屏幕预览和低端打印。提供的专色名称将用来生成分色，而不使用替代值。

如果已在前一个 *makespotcolor()* 调用中使用了 *spotname*，返回值将与前一次调用中的值相同，不会影响当前颜色。

可使用特殊专色名称 *All* 将颜色应用到所有分色，这对绘制注册商标很有用。专色名称 *None* 将在任何分色上生成不可见的输出。

范围 页面、图案、模板、字形（仅限于 **Type 3** 字体的 *colored* 选项为 **true** 的情况）、文档；如果定义自定义颜色，当前填充色不能是专色或图案。

PDF/X PDF/X-1 和 PDF/X-1a 模式不支持 **PANTONE®** 专色。

参数 *spotcolorlookup, preserveoldpantonenames*

**VB RB Function load\_iccprofile(profilename As String, optlist As String) As Long**

**C# int load\_iccprofile(String profilename, String optlist)**

搜索 ICC 配置文件 (ICC Profile)，并准备稍后使用。

**profilename** （名称字符串）*ICCProfile* 资源的名称，基于磁盘的文件或虚拟文件的文件名，或 PDF/X 的标准输出条件名称。只有 *usage* 选项设为 *outputintent* 时，才允许 PDF/X 的标准输出条件名称。

**optlist** 选项列表，根据表 8.35 描述配置文件处理的各个方面。

表 8.35 `load_iccprofile()` 的选项

关键字	类型	说明和可能的值
<code>usage</code>	关键字	描述 ICC 配置文件的预期用途（默认值： <code>iccbased</code> ）： <code>iccbased</code> ICC 配置文件将用来定义基于 ICC 的色彩空间，或应用到图像。 <code>outputintent</code> ICC 配置文件将用来定义 PDF/X 输出方法。
描述	字符串	只有 <code>usage = outputintent</code> 时，才使用此选项。它包含对 CC 配置文件的可读描述；此配置文件将与 PDF/X 输出方法一起使用。默认值：如果 <code>profilename</code> 指标准输出方法，将从内部列表中提取此描述；否则，将不做任何描述。
<code>embedprofile</code> <code>e</code>	布尔型	只有 <code>usage = outputintent</code> 时，才使用此选项。强制使用嵌入的 ICC 配置文件；即使作为 <code>profilename</code> 提供了标准输出方法，也是如此。默认值： <code>false</code>

**返回值** 配置文件句柄，可在后续 `load_image()` 调用中使用，或用来设置配置文件相关的参数。必须检查返回值是否为 `-1`；这表示出现错误。为了获得有关配置文件相关问题的性质的更多信息（未找到文件、不支持的格式等），可将 `iccwarning` 参数设为 `true`。不能在多个 PDF 文档之间使用返回的配置文件句柄。而且，如果 `usage` 选项为 `outputintent`，则不能将返回句柄应用到图像。在一个文档中，ICC 配置文件的数量不受任何限制。

**详细说明** 如果 `usage` 选项为 `iccbased`，将根据第 53 页上的第 3.3.4 节“色彩管理和 ICC 配置文件”中讨论的搜索策略搜索命名的配置文件。如果找到配置文件，将检查它是否适用（例如，颜色组件的数量）。`sRGB` 配置文件始终都可在内部使用，但不能进行配置。

如果 `usage` 为 `outputintent`，将首先在内部标准输出方法列表中搜索命名的配置文件。如果搜索不成功，将在用户配置的输出方法列表中搜索该名称。如果已找到提供的名称，并根据内建或用户配置列表确认为标准输出方法，将不再搜索 ICC 配置文件；而且，`description` 选项提供的名称将嵌入到 PDF 输出中，用作 PDF/X 输出方法。如果发现此名称不是标准输出方法标识符，则将其用作配置文件名称，并将对应的 ICC 配置文件嵌入 PDF，用作 PDF/X 输出方法。

**范围** 文档；如果 `usage` 选项为 `iccbased`，将允许下列范围：页面、图案、模板、字形。

**参数** 参见表 8.36。

**PDF/X** 生成文档的输出方法必须使用此函数设置，或者使用 `process_pdi()` 复制导入文档的输出方法。

表 8.36 ICC 配置文件的参数和值

函数	关键字	说明
<code>set_parameter</code>	<code>ICCProfile</code> <code>StandardOutputIntent</code>	如 UPR 文件中各个类别所示的对应资源文件行（参见第 42 页上的第 3.1.5 节“资源配置和文件搜索”）。可由多个调用向内部列表添加新条目。（另请参见表 8.3 中的 <code>resourcefile</code> ）。范围：任意
<code>set_parameter</code>	<code>iccwarning</code>	启用或禁用与 ICC 配置文件相关的警告（非致命异常）。可能的值为 <code>true</code> 和 <code>false</code> ，默认值为 <code>false</code> 。范围：任意
<code>get_value</code>	<code>icccomponents</code>	通过 <code>modifier</code> 中提供的句柄返回 ICC 配置文件内的颜色组件数量。
<code>set_value</code>	<code>setcolor:iccprofilegray</code>	设置 ICC 配置文件；该配置文件指定用于 <code>setcolor()</code> 的灰色色彩空间。范围：文档、页面、图案、模板、字形
<code>set_value</code>	<code>setcolor:iccprofilergb</code>	设置 ICC 配置文件；该配置文件指定用于 <code>setcolor()</code> 的 RGB 色彩空间。范围：文档、页面、图案、模板、字形
<code>set_value</code>	<code>setcolor:iccprofilecmky</code>	设置 ICC 配置文件；该配置文件指定用于 <code>setcolor()</code> 的 CMYK 色彩空间。范围：文档、页面、图案、模板、字形
<code>set_value</code>	<code>defaultgray</code> <code>defaultrgb</code> <code>defaultcmky</code>	不常用，在 <code>begin/end_page_ext()</code> 中使用 <code>defaultgray</code> 、 <code>defaultrgb</code> 和 <code>defaultcmky</code> 选项。

8.5.2 图案和着色

VB RB

Function begin\_pattern(width As Double, height As Double,  
xstep As Double, ystep As Double, painttype As Long) As Long

C#

int begin\_pattern(double width, double height, double xstep, double ystep, int painttype)

开始图案定义。

**width, height** 图案定界框的尺寸（以点数为单位）。

**xstep, ystep** 重复放置图案以描边或填充某个对象时的偏移量。大多数应用程序会将他们分别设为图案的 *width* 和 *height* 值。

**painttype** 如果 *painttype* 为 1，图案必须包含自己的颜色规范；将在使用图案时应用此颜色图案；如果 *painttype* 为 2，图案不能包含任何颜色规范，使用该图案进行填充或描边时将应用当前填充色或描边色。

返回值 图案句柄；在关闭的文档范围内，可在后续 *setcolor()* 调用中使用此句柄。

详细说明 此函数将所有文本、图形和颜色状态参数重设为他们的默认值。图案定义过程中，不能使用超文本函数和打开图像的函数，但可以使用所有文本、图形和颜色函数（正在定义过程中的图案除外）。

范围 文档、页面；此函数是图案范围的开头，必须与对应的 *end\_pattern()* 调用成对使用。

VB RB

Sub end\_pattern()

C#

void end\_pattern()

完成图案定义。

范围 图案；此函数结束图案范围，必须与对应的 *begin\_pattern()* 调用成对使用。

VB RB

Function shading\_pattern(shading As Long, optlist As String) As Long

C#

int shading\_pattern(int shading, String optlist)

使用着色对象定义着色图案（要求 PDF 1.4 或更高版本）。

**shading** *shading()* 返回的着色句柄。

**optlist** 选项列表，根据表 8.37 描述着色图案的各个方面。

返回值 图案句柄；在关闭的文档范围内，可在后续 *setcolor()* 调用中使用此句柄。

详细说明 此函数可用来使用着色填充任意对象。为此，必须使用 *shading()* 获取一个着色句柄，然后使用 *shading\_pattern()*，基于此着色定义图案。最后，可将图案句柄应用到 *setcolor()*，将当前颜色设为着色图案。

范围 文档、页面、字体

表 8.37 *shading\_pattern()* 的选项

关键字	类型	说明和可能的值
<i>gstate</i>	句柄	图形状态句柄

**VB RB** Sub shfill(shading As Long)  
**C#** void shfill(int shading)

基于着色对象，使用着色填充某个区域（要求 PDF 1.4 或更高版本）。

**shading** *shading()* 返回的着色句柄。

**详细说明** 通过此函数，可在不涉及 *shading\_pattern()* 和 *setcolor()* 的情况下使用着色。但是，这只适用于简单的形状；其中，待填充对象的几何形状与着色本身完全相同。由于当前剪切区域将被着色（根据着色的 *extend0* 和 *extend1* 选项），此函数通常会与 *clip()* 结合使用。

**范围** 页面、图案（仅限于图案的绘画类型为 1 的情况）、模板、字形（仅限于 Type 3 字体的 *colorized* 选项为 true 的情况）、文档

**VB RB** Function shading(shtype As String, xo As Double, yo As Double, x1 As Double, y1 As Double, c1 As Double, c2 As Double, c3 As Double, c4 As Double, optlist As String) As Long  
**C#** int shading(String shtype, double xo, double yo, double x1, double y1, double c1, double c2, double c3, double c4, String optlist)

通过当前填充色定义与另一颜色的混合效果（要求 PDF 1.4 或更高版本）。

**shtype** 着色的类型；对于线性着色，必须是 *axial*，或对于圆形着色必须是 *radial*。

**xo, yo, x1, y1** 对于轴向着色，(xo, yo) 和 (x1, y1) 是着色起点和端点的坐标。对于辐射着色，这些点可指定开始和结束圆形的圆心。

**c1, c2, c3, c4** 着色端点的颜色值；在当前填充色色彩空间中的解释方法与 *setcolor()* 中的颜色参数相同。如果当前填充色色彩空间为专色色彩空间，将忽略 *c1*，且 *c2* 将包含色调值。

**optlist** 选项列表，根据表 8.38 描述着色的各个方面。

**返回值** 着色句柄；在关闭的文档范围内，可在后续 *shading\_pattern()* 和 *shfill()* 调用中使用此句柄。

**详细说明** 当前填充色将用作起始颜色，它不能基于图案。

**范围** 文档、页面、字体

表 8.38 *shading()* 的选项

关键字	类型	说明和可能的值
<i>N</i>	浮点型	颜色过渡函数的指数，必须 > 0。默认值：1
<i>ro</i>	浮点型	（仅用于辐射着色，这里要求此选项）起始圆形的半径。
<i>r1</i>	浮点型	（仅用于辐射着色选项，这里要求此选项）结束圆形的半径。
<i>extend0</i>	布尔型	指定是否将着色延伸出起点。默认值：false
<i>extend1</i>	布尔型	指定是否将着色延伸出结点。默认值：false
<i>antialias</i>	布尔型	指定是否为着色启动消除锯齿功能。默认值：false

## 8.6 图像和模板函数

表 8.39 列出了本节的相关参数及相应的值。

表 8.39 图像函数的参数和值（参见第 168 页上的第 8.2.3 节“参数处理”）

函数	关键字	说明
<i>get_value</i>	<i>imagewidth</i> <i>imageheight</i>	分别获取图像的宽度或高度，以像素为单位。 <i>modifier</i> 是所选图像的整型句柄。范围：页面、图案、模板、字形、文档、路径。
<i>get_value</i>	<i>orientation</i>	获取图像的方向值。 <i>modifier</i> 是所选图像的整型句柄。对于包含方向标记的 <i>TIFF</i> 图像，将返回此标记的值；在所有其他情况下，将返回 <i>1</i> ；同时 <i>PDFlib</i> 也将相应调整图像数值以保持原貌。范围：页面、图案、模板、字形、文档、路径。
<i>get_value</i>	<i>resx</i> <i>resy</i>	分别获取图像的水平或垂直分辨率。 <i>modifier</i> 是所选图像的整型句柄。范围：页面、图案、模板、字形、文档、路径。  如果值为正值，返回值为图像分辨率，单位是像素 / 英寸 ( <i>dpi</i> )。如果返回值为负值，可使用 <i>resx</i> 和 <i>resy</i> 查找非方形像素的长宽比，但不具有任何绝对意义。如果返回值为零，则表示图像的分辨率未知。
<i>set_parameter</i>	<i>honoriccprofile</i>	读取图像内嵌入的 <i>ICC</i> 配置文件，并将其应用到图像数据。默认值： <i>true</i>
<i>set_parameter</i>	<i>imagewarning</i>	此参数可用于获得无法用 <i>load_image()</i> 成功打开图像的原因的相关详细信息。默认值： <i>false</i> ）。范围：任意 <i>true</i> 图像函数失败时，引发异常并返回 <i>-1</i> 异常提供的消息可能对调试很有用。 <i>false</i> 图像函数失败时，不引发异常。相反，此函数只是简单地返回 <i>-1</i>
<i>set_value</i>	<i>image:iccprofile</i>	<i>ICC</i> 配置文件的句柄；除非提供了 <i>iccprofile</i> ，否则该配置文件将应用到所有相关图像。
<i>get_value</i>	<i>image:iccprofile</i>	返回 <i>modifier</i> 中提供的图像句柄引用的图像内所嵌入的 <i>ICC</i> 配置文件句柄。
<i>set_value</i>	<i>renderingintent</i>	图像的渲染方法。有关可能的关键字列表及其含义，请参见表 3.7。默认值： <i>Auto</i> 。

### 8.6.1 图像

VB RB

Function load\_image(imagetype As String, filename As String, optlist As String) As Long

C#

int load\_image(String imagetype, String filename, String optlist)

根据各种选项打开基于磁盘的或虚拟的图像文件。

**imagetype** 字符串 *auto* 指示 *PDFlib* 自动检测图像文件类型（不能用于 *CCITT* 和原始图像）。用 *bmp*、*ccitt*、*gif*、*jpeg*、*jpeg2000*、*png*、*raw* 或 *tiff* 字符串之一显式指定图像格式可使性能略有提高（有关详细信息，参见第 106 页上的第 5.1.2 节“受支持的图像文件格式”）。类型 *jpeg2000* 要求 *PDF 1.5* 或更高版本。类型 *ccitt* 不同于 *TIFF*，此类型包含 *CCITT* 压缩图像数据。

**filename** （名称字符串）通常指定要打开的图像文件的文件名。此名称必须为基于磁盘的文件或虚拟文件的文件名；*PDFlib* 不会从 *URL* 提取图像数据。

**optlist** 选项列表，根据表 8.40 指定图像相关属性。

**返回值** 图像句柄，可用于后续的图像相关调用。必须检查返回值是否为 *-1*；这表示出现错误。要获得有关图像相关问题的更多详细信息（图像文件名错误、不支持格式、图像数据错误等

等），应将 *imagewarning* 参数或选项设为 *true*（参见表 8.39 和表 8.40）。不能在多个 PDF 文档之间使用返回的图像句柄。

详细说明 此函数将打开并分析使用 *imagetype* 参数中确定的一种支持格式的光栅图形，并将相关图像数据复制至输出文档。此函数不会在输出上显示任何效果。要将导入图像实际置于生成的输出文档内，必须使用 *fit\_image()*。不建议为每个生成的文档多次打开同一图像；因为这样，实际的图像数据将向输出文档复制多次。

PDFlib 将根据 *filename* 打开图像文件，处理内容，并在返回之前关闭文件。尽管可以在文档中多次放置图像（参见 *fit\_image()*），但实际的图像文件不会在此调用之后继续保持打开状态。

如果 *imagetype* = *raw* 或 *ccitt*，PDFlib 不能根据图像数据推算出 *width*、*height*、*components* 和 *bpc* 选项的值，故必须提供其相应值用户应负责提供与图像实际匹配的选项值。否则，可能会生成损坏的 PDF 输出，且 Acrobat 可能返回图像数据不足的错误信息。

如果 *imagetype* = *raw*，提供的图像数据的长度必须等于  $[width \times components \times bpc / 8] \times height$ （以字节为单位），括号内的项目将使用最接近的较大整数。图像样本将按照标准 PostScript/PDF 的顺序，即：由上而下、从左到右（假定没有坐标转换）。必须提供 16 位样本，首先提供最重要的字节（*big-endian* 或 «Mac» 字节顺序）。像素值的极性将在第 50 页上的第 3.3.1 节“颜色和色彩空间”讨论。如果 *bpc* 小于 8，则每个像素行都以字节边界开头，且颜色值必须从左到右封装在一个字节内。图像样本始终都采用交替顺序，即：首先提供第一个像素的所有颜色值，然后提供第二个像素的所有颜色值，依此类推。

范围 如果未提供 *inline* 选项，范围为文档、页面、字体；而且，此函数必须与对应的 *close\_image()* 调用成对使用。在文档或字体范围载入图像，而不在页面范围载入，可在某种程度上优化输出大小。

如果提供了 *inline* 选项，则范围为页面、图案、模板、字形，但不能调用 *close\_image()*。

PDF/X 所有 PDF/X 规范等级：

- ▶ 不允许 *OPI-1.3* 和 *OPI-2.0* 选项
- ▶ 如果蒙版指向 1 位图像，则只允许 *masked* 选项。
- ▶ 由于 JPEG2000 图像要求 PDF 1.5 或更高版本，所以不允许这些选项。

PDF/X-1 和 PDF/X-1a：不允许 RGB 图像。

PDF/X-3：除非 PDF/X 输出方法为灰度或 CMYK 设备，否则灰度图像要求事先设置 *begin\_page\_ext()* 中的 *defaultgray* 选项。除非 PDF/X 输出方法为 RGB 设备，否则 RGB 图像要求事先设置 *begin\_page\_ext()* 中的 *defaultrgb* 选项。除非 PDF/X 输出方法为 CMYK 设备，否则 CMYK 图像要求事先设置 *begin\_page\_ext()* 中的 *defaultcmky* 选项。

参数 *imagewidth*、*imageheight*、*resx*、*resy*、*imagewarning*

表 8.40 *load\_image()* 的选项

关键字	类型	说明
<i>bitreverse</i>	布尔型	（仅用于 <i>imagetype=ccitt</i> 的情况）如果为 <i>true</i> ，则对压缩数据中的所有字节执行位反转。默认值： <i>false</i>
<i>bpc</i>	整型	（仅用于 <i>imagetype=raw</i> 的情况且是必需的）每个组件的位数，必须为 1、2、4 或 8。在 PDF 1.5 中，也允许 <i>bpc=16</i> 。
<i>colorize</i>	专色句柄	（如果提供了 <i>iccprofile</i> 选项，将忽略此选项；不能用于 <i>imagetype=jpeg2000</i> 的情况）用专色句柄着色图像，必须事先使用 <i>makespotcolor()</i> 获取句柄。图像必须为黑白图像或灰度图像。
<i>components</i>	整型	（仅用于 <i>imagetype=raw</i> 的情况且是必需的）图像组件（通道）的数量，必须为 1、3 或 4。
<i>height</i>	整型	（仅用于 <i>imagetype = raw</i> 和 <i>ccitt</i> 的情况，且是必需的）以像素为单位的图像高度。



表 8.40 load\_image() 的选项（续）

关键字	类型	说明
honoriccprofile	布尔型	（仅用于 <i>imagetype=jpeg、png 和 tiff</i> 的情况，如果提供了 <i>colorize</i> 选项，将设为 <i>false</i> ）读取嵌入的 <i>ICC</i> 配置文件（如果有）并将其应用到图像。 默认值： <i>honoriccprofile</i> 参数的值。
iccprofile	icc 句柄	（仅用于 <i>imagetype=jpeg、png 和 tiff</i> 的情况）应用到图像的 <i>ICC</i> 配置文件的句柄。 <i>Default: an embedded profile if one is present in the image and the honoriccprofile option is true</i> 默认值：若在图像中带有 <i>ICC</i> 配置文件且 <i>honoriccprofile</i> 选项为 <i>true</i> ，则内嵌配置文件
iconname	超文本字符串	（如果 <i>inline=true</i> ，将忽略此选项；如果 <i>template=true</i> ，则是必需的）向图像附加一个名称，以便能够通过 <i>JavaScript</i> 引用，例如：将图像用作表单域的图标。
ignoremask	布尔型	忽略图像中的透明度信息。默认值： <i>false</i>
ignoreorientation	布尔型	（仅用于 <i>imagetype=tiff</i> 的情况）忽略图像中的所有方向标记。这可能会对弥补错误的方向信息很有用。默认值： <i>false</i>
imagewarning	布尔型	无法打开图像时，启用异常。默认值：全局 <i>imagewarning</i> 参数的值。
inline	布尔型	（仅用于 <i>imagetype=ccitt、jpeg 和 raw</i> 的情况）如果为 <i>true</i> ，图像将直接写入页面、图案、模板或字形描述的内容流中（参见第 105 页上的第 5.1.1 节“基本图像处理”）。
interpolate	布尔型	启用图像插值处理，以改善图像在屏幕和纸张上的显示效果。这对使用 <i>Type 3</i> 字体的字形描述的位图图像很有用。默认值： <i>false</i>
invert	布尔型	（除非 <i>mask=true</i> ，否则不能用于 <i>imagetype=jpeg2000</i> 的情况）对图像进行反相处理（亮色和暗色互换）。这可用作一种变通方法，处理应用程序会有不同解释的图像。默认值： <i>false</i>
K	整型	（仅用于 <i>imagetype=ccitt</i> 的情况）用于编码方案选择的 <i>CCITT</i> 压缩参数。默认值： <i>0</i> <i>-1</i> <i>G4</i> 压缩 <i>0</i> 一维 <i>G3</i> 压缩 ( <i>G3-1D</i> ) <i>1</i> 混合 <i>1</i> 维和 <i>2</i> 维压缩 ( <i>G3, 2-D</i> )
mask	布尔型	（仅用于具有一个颜色组件的图像，包括索引颜色）。图像将用作蒙版（参见第 108 页上的第 5.1.3 节“图像蒙版和透明度”）。 <i>1</i> 位蒙版此选项是必需的，但对于每像素位数大于 <i>1</i> 的蒙版此选项是可选的。但是，位数大于 <i>1</i> 的蒙版要求 <i>PDF 1.4</i> 。默认值： <i>false</i> 。蒙版有两种用途： ► 对另一图像蒙版：返回的图像句柄可用于后续调用，以打开另一图像，也可以提供给 » <i>masked</i> « 选项。 ► 放置一个着色透明图像：将图像中的 <i>0</i> 位像素视为透明，并使用当前填充色对 <i>1</i> 位像素着色。
masked	图像句柄	作为蒙版应用到当前图像的图像句柄。此图像句柄是由前一个 <i>load_image()</i> 调用返回的（如果是 <i>1</i> 位蒙版，且由 <i>PDF 1.3</i> 生成，将使用 » <i>masked</i> « 选项），此句柄尚未关闭。在 <i>PDF 1.3</i> 兼容模式下，图像句柄必须指向一个 <i>1</i> 位图像；这是由于，只有 <i>PDF 1.4</i> 及更高版本支持软蒙版。

表 8.4o load\_image() 的选项（续）

关键字	类型	说明
OPI-1.3	选项列表	<p>选项列表，包含用作选项名称的 <i>OPI 1.3 PostScript</i> 注释；下列条目是必需的：</p> <p><i>ALDImageFilename</i>（字符串）、<i>ALDImageDimensions</i>（整数值列表）、<i>ALDImageCropRect</i>（具有整数值矩形）、<i>ALDImagePosition</i>（浮点值列表）</p> <p>下列条目是可选的：</p> <p><i>ALDImageID</i>（字符串）、<i>ALDObjectComments</i>（字符串）、<i>ALDImageCropFixed</i>（矩形）、<i>ALDImageResolution</i>（浮点值列表）、<i>ALDImageColorType</i>（关键字，<i>Process</i>、<i>Spot</i>、<i>Separation</i>；默认值：<i>Spot</i>）、<i>ALDImageColorType</i>（介于 0 到 1 之间的 4 个颜色值列表和一个颜色名称）、<i>ALDImageTint</i>（浮点型）、<i>ALDImageOverprint</i>（布尔型）、<i>ALDImageType</i>（整数值列表）、<i>ALDImageGrayMap</i>（整数值列表）、<i>ALDImageTransparency</i>（布尔型）、<i>ALDImageAsciiTag</i>（整数 / 字符串对列表）</p> <p>子选项 <i>normalizefilename</i> 控制文件名的处理：如果为 <i>true</i>，将按照 <i>PDF</i> 参考的规定进行规范化处理。如果为 <i>false</i>，将复制至输出，而不做任何修改。后者可能对使用某些无法正确处理规范化文件名的 <i>OPI</i> 服务器很有用。默认值：<i>false</i></p>
OPI-2.0	选项列表	<p>选项列表，包含用作选项名称的 <i>OPI 2.0 PostScript</i> 注释；要求以下项目：</p> <p><i>ImageFilename</i>（字符串）</p> <p>应同时提供或同时省略下列条目：</p> <p><i>ImageCropRect</i>（矩形）、<i>ImageDimensions</i>（浮点值列表）</p> <p>下列条目是可选的：</p> <p><i>MainImage</i>（字符串）、<i>TIFFASCIITag</i>（整数 / 字符串对列表）、<i>ImageOverprint</i>（布尔型）、<i>ImageInks</i>（<i>full_color</i> 字符串、<i>registration</i> 字符串、或包含每个色料名称和色调的 <i>monochrome</i> 字符串和字符串 / 浮点值对的列表）、<i>IncludedImageDimensions</i>（整数值列表）、<i>IncludedImageQuality</i>（具有 1、2 或 3 值的整数值）</p> <p>同时，还支持 <i>normalizefilename</i> 选项（参见 <i>OPI-1.3</i>）。</p>
页面	整型	<p>（仅用于 <i>imagetype=gif</i> 或 <i>tiff</i> 的情况；如果使用其他格式，则必须为 1）根据给定数值，从多页图像文件中提取图像。第一个图像的编码为 1。默认值：1</p>
passthrough	布尔型	<p>（仅用于 <i>imagetype=tiff</i> 或 <i>jpeg</i> 的情况）控制图像数据的处理（默认值：<i>true</i>）：</p> <p><i>tiff</i>      如果为 <i>true</i>，压缩的 <i>TIFF</i> 图像数据将在可能的情况下直接传送到 <i>PDF</i> 输出。如果 <i>TIFF</i> 图像包含的数据受到破坏或者不完整，将此选项设为 <i>false</i> 会有所帮助。</p> <p><i>jpeg</i>      如果为 <i>true</i>，将 <i>JPEG</i> 图像数据直接复制至 <i>PDF</i> 输出。如果为 <i>false</i>，<i>PDFlib</i> 将转换 <i>JPEG</i> 图像数据的代码，以清理数据，使其兼容 <i>Acrobat</i>。将此参数设为 <i>false</i> 可帮助处理不兼容 <i>Acrobat</i> 的图像数据。对于 <i>multiscan</i> 和某些 <i>CMYK JPEG</i> 图像，此选项将强制设为 <i>false</i>。如果有 <i>JPEG</i> 图像要求 <i>passthrough=false</i>，请向我们的支持部门报告，以便我们能够自动检测这种方式。</p>
renderingintent	关键字	<p>图像的渲染方法。有关可能的关键字列表及其含义，请参见表 3.7。默认值：全局 <i>renderingintent</i> 参数的值。</p>
template	布尔型	<p>如果为 <i>true</i>，生成嵌入在表单 <i>XObject</i> 中的 <i>PDF</i> 图像 <i>XObject</i>（在 <i>PDFlib</i> 中，称为模板），而不生成单纯的图像 <i>XObject</i>。这可用来创建只包含一个图像的表单域图标模板。使用某个 <i>OPI-1.3</i> 或 <i>OPI-2.0</i> 选项时，同样要求此选项，以便兼容某些 <i>OPI</i> 服务器。默认值：<i>false</i>。范围：文档</p>
width	整型	<p>（仅用于 <i>imagetype = raw</i> 和 <i>ccitt</i> 的情况，且是必需的）以像素为单位的图像宽度。</p>

VB RB

Sub close\_image(image As Long)

C#

void close\_image(int image)

关闭图像。

**image** 由 `load_image()` 返回的有效图像句柄。

详细说明 此函数只会影响 PDFlib 的相关内部图像结构。如果已从文件中打开了图像，此调用不会影响实际图像文件；这是因为，已经在对应的 `load_image()` 调用结束时关闭了图像。由于此函数会断开 PDFlib 与图像的内部关联，所以用此函数关闭图像句柄后，将无法继续使用。

范围 文档、页面、字体；除非已经使用了 `inline` 选项，否则必须与对应的 `load_image()` 调用成对使用。

VB RB

Sub fit\_image(image As Long, x As Double, y As Double, optlist As String)

C#

void fit\_image(int image, double x, double y, String optlist)

根据各种选项，将图像或模板置于页面上的位置  $(x, y)$ 。

**image** 由 `load_image()` 或 `begin_template()` 函数返回的有效图像或模板句柄。

**x, y** 用户坐标系统中的参考点坐标；将根据各种选项，在此坐标上放置图像或模板。

**optlist** 选项列表，根据表 8.41 指定详细的放置信息。

详细说明 将相对于参考点  $(x, y)$  放置图像或模板（下文统称为对象）。默认情况下，对象的左下角将置于参考点上。但是，`orientate`、`boxsize`、`position` 和 `fitmethod` 选项可修改此行为。默认情况下，将根据其分辨率值缩放图像。可使用 `dpi`、`scale` 和 `fitmethod` 选项修改此行为。

范围 页面、图案、模板、字形（仅用于 Type 3 字体的 `colorized` 选项为 `true` 或图像为蒙版的情况）；只要尚未使用 `close_image()` 关闭图像句柄，便可在任意页面上多次调用此函数，次数不受任何限制。

表 8.41 `fit_image()` 和 `fit_pdi_page()` 的选项

关键字	类型	说明
<code>adjustpage</code>	布尔型	(若 <code>blind=true</code> 忽略) 将当前页面的尺寸调整为对象的尺寸，以使页面的右上角与图像的右上角加 $(x, y)$ 的点重叠。 <code>position</code> 选项值为 <code>o</code> 时，应注意下列有用的情况： $x \geq 0$ 且 $y \geq 0$ 对象被白色边距环绕。边距的水平宽度为 $y$ ，垂直宽度为 $x$ 。 $x < 0$ 且 $y < 0$ 将从图像中裁切水平和垂直的条带。 此选项只在页面范围内有效，且不能用于 <code>topdown</code> 参数设为 <code>true</code> 的情况。 默认值: <code>false</code>
<code>blind</code>	布尔型	如果为 <code>true</code> ，将完成所有定位和缩放计算，但不会将对象置于输出页面之上。这可用于处理页面上的块，而不实际使用页面的内容。默认值: <code>false</code>
<code>boxsize</code>	浮点值列表	指定框宽度和高度的两个值，将相对于这两个值放置并可能缩放对象。框的左下角即为参考点 $(x, y)$ 。放置图像并使其填充此框的操作由 <code>position</code> 和 <code>fitmethod</code> 选项控制。如果 <code>width = 0</code> ，只考虑高度；如果 <code>height = 0</code> ，则只考虑宽度。在这些情况下，将分别相对于 $(x, y)$ 到 $(x, y+height)$ 的垂直直线或 $(x, y)$ 到 $(x+width, y)$ 的水平直线放置对象。默认值: <code>{0 0}</code>

8.6 图像和模板函数 211

表 8.41 `fit_image()` 和 `fit_pdi_page()` 的选项（续）

关键字	类型	说明
<i>dpi</i>	浮点值 列表	一个或两个值，指定水平和垂直方向上所需的图像分辨率，以像素 / 英寸为单位。如果只提供一个值，将同时用于两个方向。如果值为 <code>0</code> ，将使用图像的内部分辨率（如果有），否则使用 <code>72 dpi</code> 。除值 <code>0</code> 外，也可提供关键字 <i>internal</i> 。由此选项得到的缩放结果相对于当前用户坐标；如果已经过缩放，则生成的物理分辨率将不同于提供的值。  模板或 PDF 页面将忽略此选项；如果 <i>fitmethod</i> 选项已随 <i>auto</i> 、 <i>meet</i> 、 <i>slice</i> 或 <i>entire</i> 关键字一起提供，同样会忽略此选项。默认值： <i>internal</i>
<i>fitmethod</i>	关键字	给出使对象填充指定框的方法。如果尚未有指定框，将忽略此选项。默认值： <i>nofit</i>  <i>nofit</i> 只确定对象位置，不进行任何缩放或剪裁处理。 <i>clip</i> 确定对象位置，并在框的边缘剪裁对象。 <i>meet</i> 根据 <i>position</i> 选项确定对象位置，并在保持长宽比的同时进行缩放以使其完全装入框内。通常，至少有对象的两个边缘与框的对应边缘对齐。将忽略 <i>dpi</i> 和 <i>scale</i> 选项。  <i>auto</i> 相当于 <i>meet</i> 。 <i>slice</i> 根据 <i>position</i> 选项确定对象位置，并进行缩放以使其覆盖整个框，可在保持长宽比的同时确保至少对象的一个尺寸可完全包含在框内。通常，对象的某些其他尺寸会延伸到框外，因此，将被剪裁。将忽略 <i>dpi</i> 和 <i>scale</i> 选项。  <i>entire</i> 根据 <i>position</i> 选项确定图像的位置，并进行缩放以使其覆盖整个框。通常，这种方法会扭曲对象。将忽略 <i>dpi</i> 和 <i>scale</i> 选项。
<i>ignoreorientation</i>	布尔型	（仅用于 <i>TIFF</i> 图像）忽略图像中的所有方向标记。这可能会对弥补错误的方向信息很有用。默认值： <i>load_image()</i> 中 <i>ignoreorientation</i> 选项的值
<i>orientate</i>	关键字	指定放置对象时的所需对象方向。默认值： <i>north</i>  <i>north</i> 向上 <i>east</i> 指向右侧 <i>south</i> 向下 <i>west</i> 指向左侧
<i>position</i>	浮点值 或关键字 列表	一个或两个值，指定对象内的参考点位置 ( <i>x,y</i> )； <code>{0 0}</code> 表示位于文本框左下角， <code>{100 100}</code> 表示位于右上角。如果已经指定了 <i>boxsize</i> 选项，则 <i>position</i> 选项还可指定框的位置。这些值表示为对象宽度和高度的百分比。如果两个百分比相等，只指定一个浮点值即可。示例：  <code>0</code> 或 <code>{0 0}</code> 左下角 <code>{50 100}</code> 顶部边缘中间 <code>50</code> or <code>{50 50}</code> 对象中央  关键字 <i>left</i> 、 <i>center</i> 、 <i>right</i> ( <i>x</i> 方向) 或者 <i>bottom</i> 、 <i>center</i> 、 <i>top</i> ( <i>y</i> 方向) 可用来替代值 <code>0</code> 、 <code>50</code> 和 <code>100</code> 。如果只指定一个关键字，将添加另一方向对应的关键字。  默认值： <code>0</code> （左下角）
<i>rotate</i>	浮点型	旋转坐标系，将中心用作参考点并将指定的值用作以度数表示的旋转角度。这将旋转框和对象。放置对象时，会重设旋转选项。默认值： <code>0</code>
<i>scale</i>	浮点值 列表	按照指定的缩放因子（而非百分比），在水平和垂直方向上缩放对象。如果两个因子相等，指定一个浮点值即可。如果 <i>fitmethod</i> 选项已随 <i>auto</i> 、 <i>meet</i> 、 <i>slice</i> 或 <i>entire</i> 关键字一起提供，将忽略此选项。默认值： <code>1</code>

### 8.6.2 模板

注：本节描述的模板函数与 *PDFlib* 块的各种数据处理无关。可使用 *fill\_textblock()*、*fill\_imageblock()* 和 *fill\_pdfblock()* 填充用 *PDFlib* 块增效工具准备的块（参见第 219 页上的第 8.8 节“块填充函数 (PPS)”）。

<b>VB RB</b>	<b>Function begin_template(width As Double, height As Double) As Long</b>
<b>C#</b>	<b>int begin_template(double width, double height)</b>

开始模板定义。

**width, height** 模板定界框的尺寸（以点数为单位）。

**返回值** 模板句柄，可用于后续的图像相关调用，尤其是 *fit\_image()*。不返回错误。

**详细说明** 此函数将所有文本、图形和颜色状态参数重设为他们的默认值。定义模板期间，不能使用超文本函数或打开图像的参数，但可以使用所有文本、图形和颜色函数。

**范围** 文档、页面；此参数是模板范围的开头，必须与对应的 *end\_template()* 调用成对使用。

<b>VB RB</b>	<b>Sub end_template()</b>
<b>C#</b>	<b>void end_template()</b>

完成模板定义。

**范围** 页面；此函数终结模板范围，必须与对应的 *begin\_template()* 调用成对使用。

8.6.3 缩览图

<b>VB RB</b>	<b>Sub add_thumbnail(image As Long)</b>
<b>C#</b>	<b>void add_thumbnail(int image)</b>

作为缩览图，在当前页面上添加现有图像。

**image** 由 *load\_image()* 获取的有效图像句柄。

**详细说明** 此函数可作为缩览图，在当前页面上添加提供的图像。缩览图必须遵守以下限制：

- ▶ 图像不能大于 106 x 106 像素。
- ▶ 图像必须使用灰度、RGB 或索引 RGB 色彩空间。
- ▶ 由于缩览图必须由单一的 PDF 图像对象构建，multi-strip TIFF 不能构建缩览图（参见第 106 页上的第 5.1.2 节“受支持的图像文件格式”）。

此函数不会生成用于页面的缩览图图像，只提供一个挂钩，将现有图像加入缩览图中。实际的缩览图图像必须由客户端生成。客户端必须确保缩览图的颜色、高度 / 宽度比例和缩览图的实际内容与页面内容相匹配。

由于 Acrobat 5 及更高版本可即时生成缩览图（尽管 Acrobat 5 或 Adobe Reader 6 不能在浏览器中生成缩览图），且缩览图会增加生成的 PDF 的总文件大小，所以建议不添加缩览图，而由客户端生成缩览图。

**范围** 页面；每页只能调用一次。并非所有页面都需要附加缩览图。

8.7 PDF 导入函数 (PDI)

注： 本节描述的所有函数都要求额外的 PDF 导入库 (PDI) ； PDI 要求 PDFlib+PDI 或 PDFlib Personalization Server (PPS)，但 PDFlib 中并不包括这些组件。有关如何获得 PDI 的更多信息，请访问我们的网站。

8.7.1 文档和页面

VB RB

Function open\_pdi(filename As String, optlist As String, len As Long) As Long

C#

int open\_pdi(String filename, String optlist, int len)

打开基于磁盘的或虚拟的 PDF 文档，并准备稍后使用。

**filename** （名称字符串） PDF 文件的文件名。

**optlist** 选项列表，根据表 8.42 指定 PDF 打开选项。

**len** （仅用于 C 语言绑定，在其他语言中必须为 0）

**返回值** 文档句柄，可用于处理文档的具体页面，或用于查询文档属性。返回值 -1，表示无法打开页面。可同时打开任意数量的 PDF 文档。可以在关闭文档范围结束之前使用返回值。

**详细说明** 默认情况下，如符合下列任一条件或多个条件，则会拒绝文档（有关详细信息，参见第 112 页上的第 5.2.3 节“可接受的 PDF 文档”）：

- 文档已损坏。
- 文档使用的 PDF 版本比当前 PDF 文档的版本高。
- 文档加密，但未在 *password* 选项中提供相应的口令。
- 文档不符合当前 PDF/X 输出规范等级。
- 文档为加标签的 PDF，且 *begin\_document()* 中的 *tagged* 选项为 *true*。

但除第一个原因外，均可使用 *infomode* 选项打开文档。这可能对查询 PDF 的相关信息很有用，例如加密或 PDF/X 状态、文档信息域等。

为了获得 PDF 导入相关问题的性质的更多详细信息（PDF 文件名错误、不支持格式、PDF 数据错误，等），可使用 *get\_errmsg()* 接收更为详细的错误消息。

**范围** 对象、文档、页面；在对象范围内，只能使用 PDI 文档句柄，从 PDF 文档中查询信息。

**PDF/X** 除非 *infomode* 选项为 *true*，否则导入的文档必须符合当前 PDF/X 输出规范等级。

**参数** 参见表 8.45 和表 8.46。

表 8.42 *open\_pdi()* 的选项

关键字	类型	说明
<i>infomode</i>	布尔型	如果为 <i>true</i> ，将可以在打开文档时查询常规信息和块属性，但不能将页面导入当前输出文档。尤其是， <i>infomode=true</i> 时，可打开下列文档（默认值： <i>false</i> ）：  不兼容当前 PDF/X 模式的 PDF。  PDF 版本高于当前文档的版本的 PDF。  不知道口令的加密 PDF（例外：创建 PDF 1.6 文档时使用 <i>Distiller</i> 设置 » 对象级压缩：最大 »）  <i>begin_document()</i> 为 <i>true</i> 的标签 PDF。
<i>inmemory</i>	布尔型	如果为 <i>true</i> ，PDI 将整个文件载入内存，并在内存中处理文件。在有些系统中（尤其是 <i>MVS</i> ），这么做能够极大地提高性能，但会加大内存消耗。如果为 <i>false</i> ，将根据需要从磁盘中读取文档的各个部分。默认值： <i>false</i>
<i>password</i>	字符串	（最大字符串长度：32 个字符）打开待导入的受保护文档所要求的许可口令。如果 <i>infomode=true</i> ，只使用文档打开口令（甚至可能为空）即可查询文档信息。如果根本没有为加密文档提供口令，则只能用文档句柄查询文档的加密状态。
<i>pdiwarning</i>	布尔型	指定此函数是否在出现错误时引发异常。默认为 <i>pdiwarning</i> 参数的值（参见表 8.46）。

<b>VB RB</b>	<b>Sub close_pdi(doc As Long)</b>
<b>C#</b>	<b>void close_pdi(int doc)</b>
	关闭所有打开的 PDI 页面句柄，并关闭输入 PDF 文件。
	<b>doc</b> 由 <i>open_pdi*()</i> 获取的有效 PDF 文档句柄。
详细说明	此函数关闭 PDF 导入文档并释放所有与文档相关的资源。所有可能打开的文档页面都将隐式关闭。此调用后，便不能使用文档句柄。如果还要导入其他页面，则不应关闭 PDF 文档。尽管可以多次打开和关闭 PDF 导入文档，且次数不受任何限制，但这么做会导致 PDF 输出文件不必要的增大。
范围	对象、文档、页面
参数	参见表 8.45 和表 8.46。

<b>VB RB</b>	<b>Function open_pdi_page(doc As Long, pagenumber As Long, optlist As String) As Long</b>
<b>C#</b>	<b>int open_pdi_page(int doc, int pagenumber, String optlist)</b>
	使用 <i>fit_pdi_page()</i> 准备页面，以供稍后使用。
	<b>doc</b> 由 <i>open_pdi*()</i> 返回的有效 PDF 文档句柄。
	<b>pagenumber</b> 要打开的页面的页码。第一页的页码为 1。
	<b>optlist</b> 选项列表，根据表 8.43 指定页面选项。
返回值	页面句柄，可与 <i>fit_pdi_page()</i> 结合使用，以放置页面。返回值 -1，指示无法打开页面。可以在关闭文档范围结束之前使用返回值。如果 <i>infomode</i> 选项为 <i>true</i> ，或者已在使用 <i>open_pdi()</i> 打开文档时设为 <i>true</i> ，则此句柄只能与 <i>get_pdi_value()</i> 和 <i>get_pdi_parameter()</i> 结合使用，以检索页面的相关信息，但此句柄不能用于 <i>fit_pdi_page()</i> 。
详细说明	此函数会把构成导入页面的所有数据复制到输出文档，但不会在输出中显示任何效果。要将导入页面实际置于生成的输出文档内，必须使用 <i>fit_pdi_page()</i> 。为了获得 PDF 导入相关问题的更多详细信息（不支持格式、PDF 数据错误，等），应将 <i>pdiwarning</i> 参数或选项设为 <i>true</i> 。如果已在 <i>infomode</i> 选项设为 <i>true</i> 的情况下打开页面，则不会向输出文件复制任何数据。 如果导入文档的 PDF 版本号高于生成的 PDF 输出文档的 PDF 版本号，此函数将会失败。 可同时打开任意数量的页面。如果多次打开同一页面，将返回不同的句柄，但一个句柄只能关闭一次。
范围	文档、页面
参数	参见表 8.45 和表 8.46。

<b>VB RB</b>	<b>Sub close_pdi_page(page As Long)</b>
<b>C#</b>	<b>void close_pdi_page(int page)</b>
	关闭页面句柄，并释放所有页面相关的资源。
	<b>page</b> 由 <i>open_pdi_page()</i> 返回的有效 PDF 页面句柄（而非页码！）。
详细说明	此函数关闭与 <i>page</i> 确定的页面句柄相关联的页面，并释放所有相关资源。不能在此调用后使用 <i>page</i> 。

表 8.43 `open_pdi_page()` 的选项

关键字	类型	说明
<i>iconname</i>	超文本字符串	向导入页面附加一个名称，以便能够通过 <i>JavaScript</i> 引用页面，例如，将页面用作表单域的图标。
<i>infomode</i>	布尔型	如果为 <i>true</i> ，将可以在打开页面时查询常规信息，但不能将页面导入当前输出文档。默认值：提供给对应 <i>open_pdi()</i> 调用的 <i>infomode</i> 选项的值（默认为 <i>false</i> ）。对于使用 <i>infomode=true</i> 打开的文档，将忽略此选项。
<i>pdiusebox</i>	关键字	指定将用在确定导入页面大小的框尺寸。有关详细信息，参见第 111 页上的第 5.2.2 节“与 <i>PDFlib</i> 一起使用 <i>PDI</i> 函数”（默认值： <i>crop</i> ）： <i>media</i> 使用媒体框（始终提供此框） <i>crop</i> 如果有，则使用裁剪框，否则使用媒体框。 <i>bleed</i> 如果有，则使用出血框，否则使用裁剪框。 <i>trim</i> 如果有，则使用裁切框，否则使用裁剪框。 <i>art</i> 如果有，则使用作品框，否则使用裁剪框。
<i>pdiwarning</i>	布尔型	指定此函数是否在出现错误时引发异常。默认为 <i>pdiwarning</i> 参数的值（参见表 8.46）。

范围 文档、页面

参数 参见表 8.45 和表 8.46。

VB RB

C#

Sub fit\_pdi\_page(page As Long, x As Double, y As Double, optlist As String)

void fit\_pdi\_page(int page, double x, double y, String optlist)

根据各种选项，在页面上放置导入的 PDF 页面。

**page** 由 `open_pdi_page()` 返回的有效 PDF 页面句柄（而非页码！）。打开文档和页面时，`infomode` 选项必须已设为 `false`。不能事先关闭页面句柄。

**x, y** 用户坐标系统中的参考点坐标；将根据各种选项，在此坐标上放置页面。

**optlist** 选项列表，根据表 8.44 指定详细的放置信息。

详细说明 此函数类似于 `fit_image()`，但在导入的 PDF 页面上操作。PDF 页面同样支持表 8.44 中讨论的大多数缩放和放置选项。

范围 页面、图案、模板、字形

参数 参见表 8.45 和表 8.46。

PDF/X 被导入页面的文档必须遵守与最终输出的 PDF/X 等级相兼容的 PDF/X 等级（参见表 7.7），且必须使用与生成文档相同的输出方法。

8.7.2 其他 PDI 处理

VB RB

C#

Function process\_pdi(doc As Long, page As Long, optlist As String) As Long

int process\_pdi(int doc, int page, String optlist)

处理导入 PDF 文档的某些元素。

**doc** 由 `open_pdi*`() 返回的有效 PDF 文档句柄。



**page** 如果 *optlist* 要求页面句柄（参见表 8.44），*page* 必须为通过 *open\_pdi\_page()* 返回的有效 PDF 页面句柄（而非页码！）。不能事先关闭页面句柄。如果 *optlist* 不要求任何页面句柄，*page* 必须为 -1。

**optlist** 选项列表，根据表 8.44 指定处理选项。

**返回值** 如果此函数成功，返回值 1；如果失败，返回错误代码 -1。

**范围** 文档

**参数** 参见表 8.46。

**PDF/X** 生成文档的输出方法必须设为使用此函数和 *copyoutputintent* 选项，或者调用 *load\_profile()*。

表 8.44 *process\_pdi()* 的选项

关键字	类型	说明
<i>action</i> <sup>1</sup>	关键字	（必需的，尽管目前只定义了一个动作）指定 PDF 处理的方式： <i>copyoutputintent</i> 将导入文档的 PDF/X 输出方法复制到输出文档。将忽略第二个及后续的复制输出方法的尝试。
<i>pdiwarning</i> <sup>1</sup>	布尔型	指定此函数是否在出现错误时引发异常。默认值为 <i>pdiwarning</i> 参数的值（参见表 8.46）。

1. 不要求页面句柄

### 8.7.3 PDI 参数处理

<b>VB RB</b>	<b>Function get_pdi_value(key As String, doc As Long, page As Long, reserved As Long) As Double</b>
<b>C#</b>	<b>double get_pdi_value(String key, int doc, int page, int reserved)</b>

获取某些数值型 PDI 文档参数。

**key** 指定要获取的参数的名称，参见表 8.45 和表 8.46。

**doc** 由 *open\_pdi()* 返回的有效 PDF 文档句柄。

**page** 由 *open\_pdi\_page()* 返回的有效 PDF 页面句柄（而非页码！）。对于与页面无关的关键字，*page* 必须为 -1。

**reserved** 目前尚未使用，必须为 0。

**返回值** 从文档中检索的数值。

**范围** 任意

<b>VB RB</b>	<b>Function get_pdi_parameter(key As String, doc As Long, page As Long, reserved As Long) As String</b>
<b>C#</b>	<b>String get_pdi_parameter(String key, int doc, int page, int reserved)</b>

获取某些字符串型 PDI 文档参数。

**key** （名称字符串）指定要获取的参数名称，参见表 8.45 和表 8.46。

**doc** 由 *open\_pdi()* 返回的有效 PDF 文档句柄。

表 8.45 PDF 导入的页面相关参数和值

函数	关键字	说明
<i>get_pdi_value</i>	<i>width</i> <i>height</i>	分别获取导入页面的宽度和高度（使用默认单位）。同时，还会考虑裁切和旋转问题。
<i>get_pdi_value</i>	<i>/ 旋转</i>	以度数表示页面旋转（0、90、180 或 270）
<i>get_pdi_value</i>	<i>/CropBox</i> , <i>/BleedBox</i> , <i>/ArtBox</i> , <i>/TrimBox</i> , <i>/MediaBox</i>	查询页面的某个框参数。必须在参数名称后附加斜线 '/' 字符并附加 <i>llx</i> 、 <i>lly</i> 、 <i>urx</i> 或 <i>ury</i> 之一，例如： <i>/CropBox/llx</i> （有关详细信息，参见第 48 页上的第 3.2.2 节“页面大小和坐标限制”）。请注意，与页面的宽度和高度值不同，它们已体现了页面旋转的因素，这些值并未加入 <i>/Rotate</i> 值的影响。
<i>get_pdi_parameter</i>	<i>isempty</i>	如果页面是空的，将返回字符串 <i>true</i> ；如果页面不是空的，将返回 <i>false</i> ；如果页面包含不支持的滤镜，将返回 <i>unknown</i> 。

**page** 由 *open\_pdi\_page()* 返回的有效 PDF 页面句柄（而非页码！）。对于与页面无关的关键字，*page* 必须为 -1。

**reserved** 目前尚未使用，必须为 0。

- 返回值
- 作为超文本字符串，从文档中检索的字符串参数。如果没有信息，将返回空字符串。  
在下次调用此参数之前，或所属对象范围结束之前，此字符串的内容将一直有效（以先出现情况的为准）。
- 详细说明
- 此函数可获取某些与导入 PDF 文档相关的字符串参数；有些情况下，可使用 *page* 和索引进一步定义。表 8.46 列出了相关参数组合。
- 范围
- 任意

表 8.46 PDF 导入的文档相关参数和值

函数	关键字	说明
<i>get_parameter</i>	<i>pdi</i> <sup>1</sup>	如果附加了 <i>PDI</i> 库，将返回字符串 <i>true</i> （ <i>PDFlib Lite</i> 不附加 <i>PDI</i> 库）；否则，返回 <i>false</i> 。范围：任意、空。
<i>get_pdi_value</i>	<i>/Root/Pages/Count</i> <sup>1</sup>	导入文档的总页数
<i>get_pdi_parameter</i>	<i>filename</i> <sup>1</sup>	导入 PDF 文件的文件名
<i>get_pdi_parameter</i>	<i>/Info/&lt;key&gt;</i> <sup>1</sup>	抽取文档信息字典 ( <i>document info dictionary</i> ) 中某个关键字的字符串值（例如 <i>/Info/Title</i> ）作为超文本字符串。也可查询文档信息字典中的自定义关键字。如果无法在文档中找到关键字，将返回空字符串。但是，如果 <i>pdiwarning</i> 设为 <i>true</i> ，将对无法找到的关键字引发异常。
<i>get_pdi_parameter</i>	<i>tagged</i>	如果文档为标签 PDF，将返回 <i>true</i> （因此无法在标签 PDF 输出模式下导入）。
<i>get_pdi_parameter</i>	<i>pdfx</i> <sup>1</sup>	返回导入文档的 <i>PDF/X</i> 规范等级。其结果可为 » <i>PDF/X-1:2001</i> «、» <i>PDF/X-1a:2001</i> «、» <i>PDF/X-3:2002</i> «、» <i>none</i> « 或指定较高 <i>PDF/X</i> 规范等级的字符串（参见第 145 页上的第 7.4 节“ <i>PDF/X</i> ”）。
<i>get_pdi_value</i>	<i>version</i> <sup>1</sup>	<i>PDF</i> 版本号乘以 10，例如：15 指 <i>PDF 1.5</i>
<i>set_parameter</i>	<i>pdiwarning</i> <sup>1</sup>	此参数可用来获得有关无法打开 <i>PDF</i> 或页面的原因的更多详细信息。默认值： <i>false</i> <i>true</i> <i>PDI</i> 函数失败时，引发非致命异常。异常提供的信息字符串可能对调试导入相关的问题很有用。 <i>false</i> <i>PDI</i> 函数失败时，不引发异常。相反，出现错误时，此函数只是返回 -1。
<i>set_parameter</i>	<i>pdiusebox</i> <sup>1</sup>	不常用，使用 <i>open_pdi_page()</i> 中的 <i>pdiusebox</i> 选项。

表 8.46 PDF 导入的文档相关参数和值

函数	关键字	说明
<i>get_pdi_parameter</i> <i>get_pdi_value</i>	<i>vdp/Blocks/</i> <block>/ <property> 或 <i>vdp/Blocks/</i> <block>/ <i>Custom/</i> <property>	查询标准和自定义块属性（参见第 135 页上的第 6.5 节“询问块名及其属性”）。只能用于 <i>PDFlib Personalization Server (PPS)</i> 。
<i>get_pdi_value</i>	<i>vdp/blockcount</i>	查询页面中块的总数。

1. *page* 参数必须为 -1。

## 8.8 块填充函数 (PPS)

PDFlib Personalization Server (PPS) 提供了专用函数，来处理文本、图像和 *PDF* 型变量数据块。这些块必须包含在导入 *PDF* 页面内，但不会在生成输出中保留。使用任何块填充函数之前，导入页面必须已经放置在输出页面中。计算页面上的块位置时，块函数将使用相应的 *PDF* 页面句柄提供给最近一次 *fit\_pdi\_page()* 调用的缩放选项。

如果只需要处理块，而不用将页面的内容实际放置在输出中（即，导出页面只用作块容器），则可以使用 *fit\_pdi\_page()* 的 *blind* 选项。如果希望将块放置在原始页面的内容的下方，这可能很有用。要实现这一点，应在 *fit\_pdi\_page()* 中使用 *blind* 选项，根据需要填充块，然后再次调用 *fit\_pdi\_page()*，且此次调用不使用 *blind* 选项。

注：本节讨论的块处理函数要求 *PDFlib Personalization Server(PPS)*。在 *DPF* 模板中创建块要求用于 *Adobe Acrobat* 的 *PDFlib* 块增效工具。有关 *PDFlib* 块增效工具的更多信息，参见第 6 章。

VB RB

Function fill\_textblock(  
    page As Long, blockname As String, text As String, optlist As String) As Long

C#

int fill\_textblock(int page, String blockname, String text, String optlist)

根据其属性，用变量数据填充文本块。

**page** 用于包含块的页面的有效 *PDF* 页面句柄。

**blockname** （名称字符串）块的名称。

**text** （内容字符串）要填充到块中的文本；如果使用默认文本（如块属性的定义），则为空字符串。

**optlist** 选项列表，根据表 8.47 指定详细的填充信息。

返回值 如果页面中不存在命名的块，不能填充块（例如，由于字体问题）或者块的处理要求更高版本的 *PDFlib*，则返回 -1；如果可以成功处理块，则返回 1。使用 *pdiwarning* 选项可获得有关问题性质的更多信息。

详细说明 提供的文本将根据块的属性进行格式化，以填充至块中。如果 **text** 为空，此函数将使用块的默认文本（如果存在）；否则，不返回任何信息。这可能对利用填充色或描边色等其他块属性很有用。

如果发现 *PDF* 文档受到破坏，将根据 *pdiwarning* 参数或选项引发异常，或者返回 -1。

范围 页面、模板

注：此函数只能用于 *PDFlib Personalization Server (PPS)*。

---

**VB RB** **Function fill\_imageblock(**  
    **page As Long, blockname As String, image As Long, optlist As String) As Long**  
**C#** **int fill\_imageblock(int page, String blockname, int image, String optlist)**

---

根据其属性，用变量数据填充图像块。

**page** 用于包含块的页面的有效 PDF 页面句柄。

**blockname** （名称字符串）块的名称。

**image** 要填充到块中的图像的有效图像句柄；如果要使用默认图像（如块属性的定义），将为 -1。

**optlist** 选项列表，根据表 8.47 指定详细的填充信息。

**返回值** 如果页面中不存在命名的块，不能填充块或者块的处理要求更高版本的 PDFlib，则返回 -1；如果可以成功处理块，则返回 1。使用 *pdiwarning* 选项可获得有关问题性质的更多信息。

**详细说明** 根据块的属性，将提供的图像句柄指向的图像置于块中。如果 *image* 为 -1，此函数将使用块的默认图像（如果存在）；否则不返回任何信息。

如果发现 PDF 文档受到破坏，将根据 *pdiwarning* 参数的设置或选项引发异常，或者返回 -1。

**范围** 页面、模板

**注：** 此函数只能用于 *PDFlib Personalization Server (PPS)*。

---

**VB RB** **Function fill\_pdfblock(**  
    **page As Long, blockname As String, contents As Long, optlist As String) As Long**  
**C#** **int fill\_pdfblock(int page, String blockname, int contents, String optlist)**

---

根据其属性，用变量数据填充 PDF 块。

**page** 用于包含块的页面的有效 PDF 页面句柄。

**blockname** （名称字符串）块的名称。

**contents** 要填充到块中的 PDF 页面的有效 PDF 页面句柄；如果要使用默认 PDF 页面（如块属性的定义），将为 -1。

**optlist** 选项列表，根据表 8.47 指定详细的填充信息。

**返回值** 如果页面中不存在命名的块，不能填充块或者块的处理要求更高版本的 PDFlib，则返回 -1；如果可以成功处理块，则返回 1。使用 *pdiwarning* 选项可获得有关问题性质的更多信息。

**详细说明** 根据块的属性，将提供的页面句柄 *contents* 指向的 PDF 页面置于块中。如果 *contents* 为 -1，此函数将使用块的默认 PDF 页面；否则不返回任何信息。

如果发现 PDF 文档受到破坏，将根据 *pdiwarning* 参数或选项引发异常，或者返回 -1。

**范围** 页面、模板

**注：** 此函数只能用于 *PDFlib Personalization Server (PPS)*。

表 8.47 fill\_\*block() 函数的选项

关键字	类型	说明
boxsize	浮点值列表	将块的宽度和高度更改为指定的值（用当前用户坐标系统中的坐标表示）。 默认值：如块 Rect 属性中所指定的值。
charref	布尔型	（仅用于 fill_textblock()）参见表 8.20
encoding	字符串	load_font() 要求的字体编码方式。除非符合下列条件之一，否则 fill_textblock() 要求使用此选项： text 参数中的字符串为空，且使用 defaulttext 属性。 已经提供 font 选项。
glyphwarning	布尔型	（仅用于 fill_textblock()）参见表 8.19
字体	字体句柄	（仅用于 fill_textblock()）load_font() 返回的字体句柄。无默认值，必须提供 font 或 fontname。
fontwarning	布尔型	（仅用于 fill_textblock()）指定出现字体相关问题时，此函数是否会引发异常。默认值为 pdiwarning 选项的值。
ignoreorientation	布尔型	（仅用于 fill_imageblock()）如果为 true，将忽略 TIFF 图像中的方向标记。 默认值：false
imagewarning	布尔型	（仅用于 fill_imageblock()）指定出现图像相关问题时，此函数是否会引发异常。默认为 pdiwarning 选项的值。
pdiwarning	布尔型	指定包含块的 PDF 页面或用作块内容的页面中出现错误时，此函数是否会引发异常。默认值为 pdiwarning 参数的值（参见表 8.46）。
refpoint	浮点值列表	将块的左下角移至使用用户坐标指定的点上。默认值：如块 Rect 属性中所指定的值。
shrinklimit	浮点型或百分比	（仅用于 fill_textblock()）参见表 8.19
textformat	字符串	（除非使用 defaulttext 属性，否则仅用于 fill_textblock()）用来解释提供的文本的格式（参见第 77 页上的第 4.5.2 节“内容字符串、超文本字符串和名称字符串”和表 8.19）。默认值：auto
load_font() 的所有选项		（仅用于 fill_textblock()，但仅限于没有提供 font 选项的情况）参见表 8.16
几乎任何属性名称		将用来覆盖块定义内对应名称和值的属性名称和值（参见第 128 页上的第 6.4 节“用于自动化处理的标准属性”）。有关详细信息，参见第 121 页上的第 6.2.2 节“块属性”。下列块属性不能覆盖： Name、Description、Locked、Subtype、Type defaulttext、defaultimage、defaultpdf、defaultpdfpage 作为提供 fontname 属性的变通，还可以使用 font 选项提供字体句柄（此时，将忽略 fontname）。 颜色属性支持下列色彩空间关键字：none、gray、rgb、cmyk、spot、spotname。

## 8.9 超文本函数

表 8.48 列出了本节的相关参数及相应的值。

表 8.48 超文本函数的参数（参见第 168 页上的第 8.2.3 节“参数处理”）

函数	关键字	说明
<code>set_parameter</code>	<code>usercoordinates</code>	如果为 <i>false</i> ，将预计超文本矩形的坐标使用默认坐标系（参见第 46 页上的第 3.2.1 节“坐标系”）；否则，将使用当前用户坐标系。默认值： <i>false</i> ）。范围：任意

### 8.9.1 动作

VB RB

Function create\_action(type As String, optlist As String) As Long

C#

int create\_action(String type, String optlist)

创建一个动作，可将其提供给各种对象和事件。

**type** 动作的类型，用下列一个关键字指定：

- ▶ *GoTo*：转到当前文档中的某个目标。
- ▶ *GoToR*：转到另一（远程）文档中的某个目标。
- ▶ *Launch*：启动应用程序或文档。
- ▶ *URI*：解析统一资源标识符，即：跳转到某个 Internet 地址。
- ▶ *Hide*：隐藏或显示批注或表单域。
- ▶ *Named*：执行由名称标识的 Acrobat 菜单项。
- ▶ *SubmitForm*：将数据发送到统一资源定位符，即 Internet 地址（请注意，不能在 Acrobat 中使用要求基本身份验证的提交操作）。
- ▶ *ResetForm*：将文档中的某些域或全部域设置为他们的默认值。
- ▶ *ImportData*：从文件中导入域值。
- ▶ *JavaScript*：执行用 JavaScript 代码编写的脚本。
- ▶ *SetOCGState*：(PDF 1.5) 隐藏或显示图层。

**optlist** 选项列表，根据表 8.49 指定动作的属性。

- 返回值

动作句柄，可用来将动作附加到文档内的对象。可以在关闭的文档范围结束之前使用动作句柄。
- 详细说明

此函数将创建单一动作。可为一个或多个动作提供各种对象（例如，页面、表单域事件、书签等），但是必须使用单独的 `create_action()` 调用生成每个动作。允许为不同的对象多次使用一个动作。
- 范围

页面、文档。在下一个 `end_document()` 调用之前，可以一直使用返回的句柄。
- PDF/X

所有 PDF/X 模式都禁止使用动作。

表 8.49 `create_action()` 的动作属性的选项

选项	类型	说明
<code>actionwarning</code>	布尔型	如果为 <i>true</i> ，没有任何效果的动作选项将引发非致命异常。如果为 <i>false</i> ，将忽略这些选项，而不返回任何信息。默认值： <i>true</i>
<code>canonicaldate</code>	布尔型	( <i>SubmitForm</i> ) 如果为 <i>true</i> ，可将任何表示日期的提交域值转换为标准格式。域本身不会专门指定将域解释为日期，只会在处理域的 JavaScript 代码中指定。默认值： <i>false</i>

表 8.49 create\_action() 的动作属性的选项（续）

选项	类型	说明
defaultdir	字符串	(Launch) 设置启动程序的默认目录。只有在 Windows 上运行的 Acrobat 支持此选项。默认值: none
destination	选项列表	(GoTo、GoToR；除非提供了 destname，否则是必需的) 选项列表，根据表 8.50 定义要跳转到的目标。
destname	超文本字符串	(GoTo、GoToR；除非提供了 destination，否则是必需的) 用 add_nameddest() 指定的目标名称（用于 GoTo）或者远程文档中的目标的名称（用于 GoToR）。
exclude	布尔型	(SubmitForm) 如果为 true，namelist 选项指定要排除的域；除 namelist 数组中列出的域以及 exportable 选项为 false 的域之外，将提交文档中的所有域。如果为 false，namelist 选项指定提交操作中包括的域。将同时提交指定域组中的所有成员。默认值: false  (ResetForm) 如果为 true，namelist 选项指定要排除的域；除 namelist 数组中列出的域之外，将重设文档中的所有域。如果为 false，namelist 选项指定重设操作中包括的域。将同时重设指定域组中的所有成员。默认值: false
exportmethod	关键字列表	(SubmitForm) 提交域名称和域值时使用的格式，加上对应的选项（默认值: fdf）： html 使用 HTML 格式 fdf 使用 FDF 格式 xfdf 使用 XFDF 格式 pdf 使用 MIME 内容类型应用程序 /pdf 的 PDF 格式 getrequest （仅用于 html 和 pdf）使用 HTTP GET 提交；否则使用 HTTP POST updates （仅用于 fdf）包括基础 PDF 文档中所有增量更新的内容。 exclurl （仅用于 fdf）提交的 FDF 将排除 url 字符串。 annotfields （仅用于 fdf）包括所有批注和域。 onlyuser （仅用于 fdf 和 annotfields）提交操作将只包括其名称匹配当前用户名称的批注；具体由要将域提交到的远程服务器确定。 coordinate （仅用于 html）将作为表单数据的一部分传导致提交动作的鼠标点击的坐标。坐标值相对于域的矩形的左上角。 选项组合的示例: exportmethod {fdf updates onlyuser}
filename	字符串	(GoToR、Launch；是必需的) 触发动作时将打开的外部（PDF 或其他）文件或应用程序的名称。  (ImportData；必需的)：包含表单数据的外部文件的文件名。
hide	布尔型	(Hide) 指示是否隐藏 (true) 或显示 (false) 批注。默认值: true
ismap	布尔型	(URI) 如果为 true，解析 url 时将向目标 URI 添加鼠标位置的坐标。默认值: false
layerstate	选项列表	(SetOCGState；是必需的) 成对列表；其中，每对都由一个关键字和一个图层句柄组成。支持下列关键字： on 启动图层。 off 停用图层 toggle 变换图层的状态。如果使用此关键字，preserveradio 选项应设为 false。
menuname	字符串	(Named；是必需的) 要执行的菜单项的名称。常用的名称包括 nextpage、prevpage、firstpage、lastpage，但也可接受其他名称。要查找其他菜单项的名称，可在 Acrobat 的 JavaScript 控制台或调试程序中执行以下代码： <pre>function MenuList(m, level) {     console.println(m.cName);     if (m.oChildren != null)         for (var i = 0; i &lt; m.oChildren.length; i++)             MenuList(m.oChildren[i], level + 1); } var m = app.listMenuItems(); for (var i=0; i &lt; m.length; i++)     MenuList(m[i], 0);</pre>

表 8.49 create\_action() 的动作属性的选项（续）

选项	类型	说明
<i>namelist</i>	字符串 列表	<b>(Hide ; 必需的)</b> 要隐藏或显示的批注或域的名称（包括组名）。  <b>(SubmitForm)</b> 提交操作中包括的或者要排除的表单域的名称（包括组名），具体取决于 <b>exclude</b> 选项的设置。默认值：除 <b>exportable</b> 选项为 <b>false</b> 的域之外，将提交所有域。  <b>(ResetForm)</b> 重设操作中包括的或者要排除的表单域的名称（包括组名），具体取决于 <b>exclude</b> 选项的设置。默认值：将重设所有域。
<i>newwindow</i>	布尔型	<b>(GoToR、Launch)</b> 一个标志，指定是否在新窗口中打开目标文档。如果此标志为 <b>false</b> ，目标文档将在同一窗口内替换当前文档。 <b>Launch</b> ：如果文件不是 PDF 文档，将忽略此项。默认值： <b>Acrobat</b> 的行为取决于当前用户首选项。
<i>operation</i>	关键字	<b>(Launch)</b> 一个关键字，指定应用到 <b>filename</b> 选项中指定文档的操作。只有在 <b>Windows</b> 上运行的 <b>Acrobat</b> 支持此选项。如果 <b>filename</b> 选项的目标是应用程序，而非文档，将忽略此选项，并启动应用程序（默认值： <b>open</b> ）： <b>open</b> 打开文档 <b>print</b> 打印文档
<i>parameters</i>	字符串	<b>(Launch)</b> 要传递给 <b>filename</b> 选项指定的应用程序的参数字符串。只有在 <b>Windows</b> 上运行的 <b>Acrobat</b> 支持此选项。可用空格字符分隔多个参数，但每个参数不能包含任何空格字符。如果 <b>filename</b> 的目标是文档，应省略此选项。默认值： <b>none</b>
<i>preserveradio</i>	布尔型	<b>(SetOCGState)</b> 如果为 <b>true</b> ，将保留图层间的单选按钮状态关系。默认值： <b>true</b>
<i>script</i>	超文本 字符串	<b>(JavaScript ; 必需的)</b> 包含待执行 <b>JavaScript</b> 代码的字符串。
<i>scriptname</i>	超文本 字符串	<b>(JavaScript)</b> 如果存在，将使用提供的名称，作为文档级 <b>JavaScript</b> 插入 <b>script</b> 选项中提供的 <b>JavaScript</b> 。如果文档中多次提供了相同的 <b>scriptname</b> ，将使用最后一个脚本，所有其他脚本都将忽略。文档级 <b>JavaScript</b> 将在文档载入 <b>Acrobat</b> 之后执行。这可能对用于表单域脚本很有用。
<i>submitemptyfields</i>	布尔型	<b>(SubmitForm ; PDF 1.4)</b> 如果为 <b>true</b> ，将提交 <b>namelist</b> 和 <b>exclude</b> 选项确定的所有域，而不考虑他们是否具有任何值。对于没有值的域，只传送域名称。如果为 <b>false</b> ，将不提交没有值的域。默认值： <b>false</b>
<i>url</i>	字符串	<b>(URI ; 必需的)</b> 用 7 位 <b>ASCII</b> 或 <b>EBCDIC</b> （但只能包含 <b>ASCII</b> 字符）编码的统一资源定位器，指定链接目标。它可指向任意（ <b>Web</b> 或本地）资源，必须以协议标识符开头（例如 <b>http://</b> ）。 <b>textx/texty、currentx/currenty</b> 和 <b>imagewidth/imageheight</b> 参数可用来检索定位信息，以计算链接矩形尺寸。  <b>(SubmitForm ; 必需的)</b> 用 <b>URL</b> 格式给出将，将用于在 <b>Web</b> 服务器端处理提交操作的脚本的统一资源定位器（地址）。

8.9.2 命名目标

VB RB

Sub add\_nameddest(name As String, optlist As String)

C#

void add\_nameddest(String name, String optlist)

在当前文档的任意页面上创建命名目标。

**name**    （超文本字符串）目标的名称，可用作链接、书签或其他触发器的目标。在一个文档内，目标名称必须是唯一的。如果文档中多次提供了同一名称，将只使用最后一个定义，所有其他定义都将忽略，而不返回任何信息。

**optlist** 选项列表，根据表 8.50 指定目标。空列表相当于 {type fitwindow page o}。

详细说明

目标详细信息必须在 **optlist** 中指定，且目标可位于当前文档中的任一页面之上。提供的 **name** 可与 **destname** 一起用于 **create\_action()**、**create\_annotation()**、**create\_bookmark()**、和 **begin/end\_document()** 函数。这种方式可将定义与使用目标分成两步。



若在使用时已知目标，可将定义与使用合并，直接在以上函数中使用已命名的 *destination* 选项。这种方式无需调用 *add\_nameddest()* 函数。

范围 文档、页面

表 8.50 *add\_nameddest()* 的目标选项以及 *create\_action()*、*create\_annotation()*、*create\_bookmark()* 和 *begin/end\_document()* 中的目标选项

选项	类型	说明
<i>group</i>	字符串	（如果已指定了 <i>page</i> 选项，且文档使用页面组，则是必需的；否则，不允许使用此选项。）目标页面所属页面组的名称。
类型	关键字	指定目标页面上的窗口位置（默认值： <i>fitwindow</i> ）： <i>fixed</i> 使用 <i>left</i> 、 <i>top</i> 和 <i>zoom</i> 选项指定的固定目标视图。如果缺少任一选项，将保留其当前值。 <i>fitwindow</i> 将整个页面填充至窗口内。 <i>fitwidth</i> 根据窗口调整页面宽度， <i>y</i> 坐标顶部与窗口的上边缘对齐。 <i>fitheight</i> 根据窗口调整页面高度， <i>x</i> 坐标左边与窗口的左边缘对齐。 <i>fitrect</i> 将 <i>left</i> 、 <i>bottom</i> 、 <i>right</i> 和 <i>top</i> 指定的矩形填充至窗口内。 <i>fitvisible</i> 将页面的可见内容（作品框）填充至窗口内。 <i>fitvisiblewidth</i> 将页面的可见内容填充至窗口内， <i>y</i> 坐标顶部与窗口的上边缘对齐。 <i>fitvisibleheight</i> 将页面的可见内容填充至窗口内， <i>x</i> 坐标左边与窗口的左边缘对齐。 <i>nameddest</i> 已淘汰
<i>name</i>	超文本字符串	已淘汰
页面	整型	目标页面的页码（第一页为 1）。目标 <i>PDF</i> 中必须存在此页面。在页面范围中， <i>Page 0</i> 表示当前页面；在文档范围中，表示第 1 页。请注意，由于存在错误， <i>Acrobat 6.0</i> 将忽略页码，始终都会跳转到第 1 页。 <i>Acrobat 6.0.1</i> 中已修复了这个错误，旧版本中亦不存在此错误。默认值： <i>0</i>
<i>zoom</i>	浮点型或百分比	（仅用于 <i>type = fixed</i> 的情况）用来显示页面内容的缩放因子（1 表示 100%）。如果此选项缺失，或者为 <i>0</i> ，将保留启用路径时生效的缩放因子。
<i>left</i>	浮点型	（仅用于 <i>type = fixed</i> 、 <i>fitheight</i> 、 <i>fitrect</i> 或 <i>fitvisibleheight</i> 的情况）页面的 <i>x</i> 坐标，位于窗口左边缘。默认值： <i>0</i>
<i>right</i>	浮点型	（仅用于 <i>type = fitrect</i> 的情况）页面的 <i>x</i> 坐标，位于窗口右边缘。默认值： <i>1000</i>
<i>bottom</i>	浮点型	（仅用于 <i>type = fitrect</i> 的情况）页面的 <i>y</i> 坐标，位于窗口的底边。默认值： <i>0</i>
<i>top</i>	浮点型	（仅用于 <i>type = fixed</i> 、 <i>fitwidth</i> 、 <i>fitrect</i> 或 <i>fitvisiblewidth</i> 的情况）页面的 <i>y</i> 坐标，位于窗口上边缘。默认值： <i>1000</i>

8.9.3 注释

VB RB

Sub create\_annotation(*llx* As Double, *lly* As Double, *urx* As Double, *ury* As Double, *type* As String, *optlist* As String)

C#

void create\_annotation(double *llx*, double *lly*, double *urx*, double *ury*, String *type*, String *optlist*)

在当前页面上创建一个矩形注释。

*llx*, *lly*, *urx*, *ury* 用默认坐标（如果 *usercoordinates* 参数或选项为 *false*）或用户坐标（如果为 *true*）确定的注释矩形左下角和右上角的 *x* 和 *y* 坐标。*Acrobat* 会将注释的左上角与指定矩形的左上角对齐。

请注意，注释坐标不同于 `rect()` 函数的参数。 `create_annotation()` 直接设置两个角的参数，而 `rect()` 只设置一个角的坐标，及宽度和高度值。

**type** 注释的类型，用下列一个关键字指定：

- **Circle**: 圆形注释
- **FileAttachment**: 文件附件注释。 Acrobat Reader 5 不能处理文件附件，将只显示一个问号。文件附件只适用于完全版的 Acrobat 软件。
- **FreeText**: 自由文本注释
- **Highlight**: 高亮注释
- **Ink**: 油墨注释
- **Line**: 线条注释
- **Link**: 链接注释
- **Polygon**: (PDF 1.5) 多边形注释（用直线连接顶点）
- **PolyLine**: (PDF 1.5) 折线 (polyline) 注释；类似于多边形，但第一个和最后一个顶点没有连接在一起。
- **Popup**: 弹出式注释
- **Square**: 正方形注释
- **Squiggly**: (PDF 1.4) 曲线下划线注释
- **Stamp**: 橡皮图章注释
- **StrikeOut**: 删除线注释
- **Text**: 文本注释。在 Acrobat 中，此类型称为附注注释。
- **Underline**: 下划线注释

**optlist** 选项列表，根据表 8.51 指定注释属性。

范围 页面

**PDF/X** 在所有 PDF/X 模式下，只有注释完全位于出血框之外（如果没有出血框，则应位于裁切框 / 作品框之外），才允许使用注释。

表 8.51 `create_annotation()` 的注释选项

选项	类型	说明
<b>action</b>	动作列表	下列事件的注释动作列表（默认值：空列表）： <b>activate</b> 启动注释时执行的动作。允许所有动作类型。
<b>alignment</b>	关键字	（仅用于 <b>type=FreeText</b> ）注释中的文本对齐方式： <b>left</b> 、 <b>center</b> 、 <b>right</b> 。此选项不能用于 <b>Acrobat 6</b> ，它一直使用 <b>left</b> 。默认值： <b>left</b>
<b>annotcolor</b>	颜色	关闭时的注释图标背景颜色，注释弹出窗口的标题栏颜色以及链接注释的边框颜色。支持的色彩空间： <b>none</b> 、 <b>gray</b> 、 <b>rgb</b> 。默认值： <b>none</b>
<b>annotwarning</b>	布尔型	如果为 <b>true</b> ，没有任何效果的注释选项将引发非致命异常。如果为 <b>false</b> ，将忽略这些选项，而不返回任何信息。默认值： <b>true</b>
<b>borderstyle</b>	关键字	注释边框样式或多边形、多边线条、线条、正方形、圆形和油墨型注释的线条样式。 <b>solid</b> 、 <b>beveled</b> 、 <b>dashed</b> 、 <b>inset</b> 、 <b>underline</b> 。请注意， <b>beveled</b> 、 <b>inset</b> 和 <b>underline</b> 样式无法在 <b>Acrobat</b> 中可靠运行。默认值： <b>solid</b>
<b>cloudy</b>	浮点型	（仅用于 <b>type=Polygon</b> 的情况； <b>PDF 1.5</b> ）指定用来渲染多边形的 辉撒诗 效果的强度。可能的值包括 <b>0</b> （无效果）、 <b>1</b> 和 <b>2</b> 。如果使用此选项，将忽略 <b>borderstyle</b> 选项。默认值： <b>0</b>
<b>contents</b>	超文本字符串	（当 <b>type=Text</b> 、 <b>FreeText</b> 、 <b>Line</b> 、 <b>Square</b> 、 <b>Circle</b> 、 <b>Highlight</b> 、 <b>Underline</b> 、 <b>PolyLine</b> 、 <b>Polygon</b> 、 <b>Squiggly</b> 、 <b>Strikeout</b> 、 <b>Stamp</b> 、 <b>Ink</b> 、 <b>FileAttachment</b> 时，是必需的；当 <b>type=Link</b> 、 <b>PopUp</b> 时，是可选的；如果 <b>type=FreeText</b> ，此选项必须为字符串型）注释显示的文本或者（如果注释不显示文本）注释内容的替代描述（必须使用用户可读的形式）。内容的最大长度为 <b>65535</b> 字节。可使用回车符或换行符强制生成新段落。

表 8.51 create\_annotation() 的注释选项（续）

选项	类型	说明
createdate	布尔型	（PDF 1.5 或更高版本）如果为 <b>true</b> ，将为注释创建日期和时间条目。默认值： <b>false</b>
custom	选项列表的列表	（仅面向高级用户）可使用此选项，在注释词典内插入任意数量的专用条目，这可能对特殊的应用程序（如为数字打印机插入处理指令）。使用此选项要求用户熟悉 <b>PDF</b> 文件格式及目标应用程序。如果提供的值不合适，可能会生成损坏的 <b>PDF</b> 输出。每个列表都必须包含 <b>3</b> 个选项： <b>key</b> （字符串）词典关键字的名称（不包括 <b>/</b> 字符）。可指定任何非标准的 <b>PDF</b> 关键字，以及下列标准关键字： <b>Contents</b> 、 <b>Name</b> （选项 <b>iconname</b> ）、 <b>NM</b> （选项 <b>name</b> ）和 <b>Open</b> 。此时，将忽略对应的选项。 <b>type</b> （关键字）对应值的类型，必须为布尔型、名称或字符串。 <b>value</b> （如果 <b>type=string</b> ，为超文本字符串；否则，为字符串） <b>PDF</b> 输出中显示的值； <b>PDFLib</b> 将自动提供字符串和名称要求的任何修饰效果。
dasharray	浮点值列表	（仅用于 <b>borderstyle=dashed</b> 的情况）。虚线边框的虚线和间隙长度（以默认单位表示）（参见 <b>setdash()</b> ）。默认值： <b>3 3</b>
destination	选项列表	（仅用于 <b>type=Link</b> 的情况；若有活跃的动作被指明将忽略此项）定义要跳转到目标。
destname	超文本字符串	（仅用于 <b>type=Link</b> 的情况；若 <b>destination</b> 选项被指明将忽略此项）用 <b>add_nameddest()</b> 定义的目标名称。 <b>Destination</b> 或 <b>destname</b> 动作将优先于此选项。
display	关键字	在屏幕和纸张上的可见性： <b>visible</b> 、 <b>hidden</b> 、 <b>noview</b> 、 <b>noprint</b> 。默认值： <b>visible</b>
endingstyles	关键字列表	（仅用于 <b>type=Line</b> 、 <b>PolyLine</b> 的情况）由两个关键字组成的列表，指定直线终点样式： <b>none</b> 、 <b>square</b> 、 <b>circle</b> 、 <b>diamond</b> 、 <b>openarrow</b> 、 <b>closedarrow</b> 。默认值： <b>none none</b>
filename	字符串	（仅用于 <b>type=FileAttachment</b> 的情况，要求此选项）与注释关联的文件。建议只在 <b>filename</b> 中使用 <b>ASCII</b> 字符。
fillcolor	颜色	（仅用于 <b>type=FreeText</b> 的情况）文本的填充色。支持的色彩空间： <b>gray</b> 、 <b>rgb</b> 、 <b>cmymk</b> 。默认值： <b>gray 0 (=black)</b>
字体	字体句柄	（仅用于 <b>type=FreeText</b> 的情况；是必需的）指定注释使用的字体。只允许使用 <b>PDF</b> 核心字体以及下列编码方式：任何 <b>8 位</b> 编码、 <b>UCS-2 CMaps</b> 、内置编码。
fontsize	浮点型	（仅用于 <b>type=FreeText</b> 的情况；必需的）根据 <b>usercoordinates</b> 选项或参数，确定字体大小的单位为默认坐标系或用户坐标系。
highlight	关键字	（仅用于 <b>type=Link</b> 的情况）用户点击注释时显示的高亮模式： <b>none</b> 、 <b>invert</b> 、 <b>outline</b> 、 <b>push</b> 。默认值： <b>invert</b>
iconname	字符串	（仅用于 <b>type=Text</b> 、 <b>Stamp</b> 、 <b>FileAttachment</b> 的情况）显示注释时使用的图标名称： <b>type=Text</b> （默认值： <b>note</b> ）： <b>comment</b>  、 <b>key</b>  、 <b>note</b>  、 <b>help</b>  、 <b>newparagraph</b>  、 <b>paragraph</b>  、 <b>insert</b>  <b>type=Stamp</b> （默认值： <b>draft</b> ）： <b>approved</b> 、 <b>experimental</b> 、 <b>notapproved</b> 、 <b>asis</b> 、 <b>expired</b> 、 <b>notforpublicrelease</b> 、 <b>confidential</b> 、 <b>final</b> 、 <b>sold</b> 、 <b>departmental</b> 、 <b>forcomment</b> 、 <b>topsecret</b> 、 <b>draft</b> 、 <b>forpublicrelease</b> 。 <b>type=FileAttachment</b> （默认值： <b>pushpin</b> ）： <b>graph</b>  、 <b>pushpin</b>  、 <b>paperclip</b>  、 <b>tag</b> 
interiorcolor	颜色	（仅用于 <b>type=Line</b> 、 <b>PolyLine</b> 、 <b>Square</b> 、 <b>Circle</b> 的情况）分别指定注释直线终点、矩形或椭圆形的颜色。支持的色彩空间： <b>none</b> 、 <b>gray</b> 、 <b>rgb</b> 。默认值： <b>none</b>
line	由 <b>4</b> 个浮点值组成的列表	仅用于 <b>type=Line</b> 的情况；要求此选项）由 <b>4</b> 个数值 <b>x1</b> 、 <b>y1</b> 、 <b>x2</b> 和 <b>y2</b> 组成的列表，指定线条的起点和终点坐标；如果 <b>usercoordinates</b> 参数为 <b>false</b> ，使用默认坐标系；如果此参数为 <b>true</b> ，使用用户坐标系。

表 8.51 create\_annotation() 的注释选项（续）

选项	类型	说明
<i>linewidth</i>	整型	注释边框宽度或线条、折线、多边形、正方形、圆形和油墨型注释的线条宽度，使用默认单位（= <i>pointpoints</i> ）。如果 <i>linewidth</i> = 0，将不显示边框。默认值：1
<i>locked</i>	布尔型	如果为 <i>true</i> ，无法在 <i>Acrobat</i> 中编辑注释属性。默认值： <i>false</i>
<i>mimetype</i>	超文本字符串	（仅用于 <i>type=FileAttachment</i> 的情况）文件的 <i>MIME</i> 类型。启动注释时， <i>Acrobat</i> 将用它启动适当的应用程序。
<i>name</i>	字符串	唯一标识注释的名称。有些动作必须具有名称，且在页面上必须是唯一的。默认值： <i>none</i>
<i>opacity</i>	浮点型或百分比	（ <i>PDF 1.4</i> 或更高版本）不透明度的常数值（0-1 或 0%-100%），绘制注释时会使用此常数值。默认值：1
<i>open</i>	布尔型	（仅用于 <i>type=Text</i> 、 <i>Popup</i> ）如果为 <i>true</i> ，注释的初始状态显示为打开。默认值： <i>false</i>
<i>parentname</i>	字符串	（仅用于 <i>type=PopUp</i> 的情况）注释的父级注释名称。
<i>polylinelist</i>	由 1 个或多个浮点列表组成的列表	（仅用于 <i>type=Polygon</i> 、 <i>PolyLine</i> 、 <i>Ink</i> 、 <i>Highlight</i> 、 <i>Underline</i> 、 <i>Squiggly</i> 、 <i>Strikeout</i> 的情况；要求此选项）折线是一个指定成对坐标的浮点值列表。如果 <i>usercoordinates</i> 选项为 <i>false</i> ，将使用默认坐标系做参照；如果此选项为 <i>true</i> ，则使用用户坐标系做参照。 <i>type=Polygon</i> 、 <i>PolyLine</i> 、 <i>Ink</i> 单一列表包含一条带有 <i>n</i> 个段的 <i>PolyLine</i> （最少： <i>n=2</i> ）。一条 <i>PolyLine</i> 是一个包含 <i>2 x n</i> 个浮点值（多对坐标值）的列表。这些点将用直线连接。 例如： <i>n=3: {{10 20 30 40 50 60}}</i> <i>others</i> 列表包含 <i>n</i> 个子列表，每个子列表含有 8 个浮点值。该列表用于指定 <i>n</i> 个四边形（最少： <i>n=1</i> ）。每个四边形都包含注释下方文本中的一个单词或一组连续单词。每个四边形的坐标都以 <i>x4 y4 x3 y3 x1 y1 x2 y2</i> 的形式给出，按逆时针方向指定四边形的顶点（ <i>x4 y4</i> 为左上角）。文本的方向取决于边缘连接（ <i>x1, y1</i> ）和（ <i>x2, y2</i> ）。 例如： <i>n=2: {{1 2 3 4 5 6 7 8} {10 20 30 40 50 60 70 80}}</i>
<i>popup</i>	字符串	弹出式注释的名称，用于输入或编辑此注释关联的文本。默认值： <i>none</i>
<i>readonly</i>	布尔型	如果为 <i>true</i> ，将不允许注释与用户交互。可以显示或打印注释，但不能响应鼠标点击操作或者根据鼠标动作更改注释外观。默认值： <i>false</i>
<i>rotate</i>	布尔型	如果为 <i>true</i> ，将旋转注释，以配合页面的旋转。否则，注释旋转将保持固定不变。文本注释的图标会忽略此选项。默认值： <i>true</i>
<i>subject</i>	超文本字符串	（ <i>PDF 1.5</i> 或更高版本）简要描述注释说明的主题的文本。默认值： <i>none</i>
<i>title</i>	超文本字符串	打开注释或处于活动状态时，注释弹出窗口标题栏显示的文本标签（在 <i>Acrobat</i> 中，此标签将显示 蚬髯擻）。标题的最大长度为 255 个单字节字符或 126 个 <i>Unicode</i> 字符。但是，建议将标题的实际限制设置为 32 个字符。此字符串与 <i>Acrobat</i> 中的 蚬髯擻 域相对应。默认值： <i>none</i>
<i>usercoordinates</i>	布尔型	如果为 <i>false</i> ，将预计注释坐标和字体大小将使用默认坐标系统（参见第 46 页上的第 3.2.1 节“坐标系统”）；否则，将使用当前用户坐标系统。默认值：全局 <i>usercoordinates</i> 参数的值
<i>zoom</i>	布尔型	如果为 <i>true</i> ，将缩放注释，以配合页面的放大。否则，注释大小将保持固定不变。文本注释的图标会忽略此选项。默认值： <i>true</i>

## 8.9.4 表单域

VB RB

Sub create\_field(llx As Double, lly As Double, urx As Double, ury As Double, name As String, type As String, optlist As String)

C#

void create\_field(double llx, double lly, double urx, double ury, String name, String type, String optlist)








根据各种选项，在当前页面上创建表单域。

**llx, lly, urx, ury** 确定的表单矩形左下角和右上角的 *x* 和 *y* 坐标；如果 *usercoordinates* 参数或选项为 *false*，用默认坐标系；如果为 *true*），用户坐标系。

请注意，表单域坐标不同于 *rect()* 函数的参数。*create\_field()* 直接设置两个角的参数，而 *rect()* 只设置一个角的坐标，及宽度和高度值。

**name** （超文本字符串）表单域名称，可能会附加用 *create\_fieldgroup()* 创建的一个或多个组的名称前缀。组名称必须用句点 «.» 字符相互隔开，并同域名称隔开。在一个页面上，域名称必须是唯一的，且不能以句点 «.» 字符结尾。

**type** 域的类型，必须为下列值之一：

- ▶  *pushbutton*
- ▶  *checkbox*
- ▶  *radiobutton*。由于单选按钮必须始终属于同一组，所以 *name* 必须附加组名称前缀。对于所有其他域类型，组关系是可选的。
- ▶  *listbox*
- ▶  *combobox*
- ▶  *textfield*
- ▶  *signature*

**optlist** 一个选项列表，根据表 8.52 指定域的属性。如此表所示，字符串选项将解释为超文本字符串或文本字符串。

**详细说明** 默认情况下，页面上各域的跳位顺序（按 Tab 键时，各个域接收焦点的顺序）取决于 *create\_field()* 调用的顺序，但可以使用 *taborder* 选项指定不同的顺序。创建域之后，便不能修改跳位顺序。

在 Acrobat 中，可以为文本域指定一个格式（数值、百分比等）。但 PDF Reference 中没有相关规定，而是使用自定义 JavaScript 实施的。将 JavaScript 动作附加到指向 Acrobat 中预定义的（而非标准化的）JavaScript 函数的表单域，可异曲同工。有关示例及更多信息，参见第 59 页上的第 3.4.2 节“文本域的格式化选项”。

**范围** 页面

**PDF/X** 在所有 PDF/X 模式下，只有表单域完全位于出血框之外（如果没有出血框，则应位于裁切框 / 作品框之外）时，才允许使用表单域。

表 8.52 `create_field()` 和 `create_fieldgroup()` 的表单属性选项

选项	类型	说明
<b>action</b>	动作列表	下列一个或多个事件的域动作列表。所有域类型都允许启动事件； <b>type=pushbutton</b> 、 <b>checkbox</b> 和 <b>radiobutton</b> 时，不允许使用其他事件（默认值：空列表）： <b>activate</b> 启动域时执行的动作。 <b>keystroke</b> 用户在文本域或组合框内键入内容或修改滚动列表框内的选定时，执行的 <b>JavaScript</b> 动作。 <b>format</b> 对域进行格式化以显示其当前值之前执行的 <b>JavaScript</b> 动作。这样便可以在格式化之前修改域的值。 <b>validate</b> 更改域的值时执行的 <b>JavaScript</b> 动作。这样便可以检查新值的有效性。 <b>calculate</b> 另一个域的值发生更改时执行的 <b>JavaScript</b> 动作；这样可以重新计算此域的值。 <b>enter</b> 鼠标进入域的区域时执行的动作。 <b>exit</b> 鼠标退出域的区域时执行的动作。 <b>down</b> 在域的区域按下鼠标按钮时执行的动作。 <b>up</b> 在域的区域释放鼠标按钮时执行的动作（这通常用来启动域）。 <b>focus</b> 域收到输入焦点时执行的动作。 <b>blur</b> 域失去输入焦点时执行的动作。
<b>alignment</b>	关键字	域内文本的对齐方式： <b>left</b> 、 <b>center</b> 、 <b>right</b> 。默认值： <b>left</b>
<b>backgroundcolor</b> <b>bordercolor</b>	颜色	域背景或边框的颜色。支持的色彩空间： <b>none</b> 、 <b>gray</b> 、 <b>rgb</b> 、 <b>cmymk</b> 。默认值： <b>none</b>
<b>borderstyle</b>	关键字	域边框的样式，可以是 <b>solid</b> 、 <b>beveled</b> 、 <b>dashed</b> 、 <b>inset</b> 或 <b>underline</b> 。默认值： <b>solid</b>
<b>buttonlayout</b>	关键字	（仅用于 <b>type=pushbutton</b> 的情况）按钮标题相对于按钮图标的位置；条件是，同时指定了按钮标题和按钮图标： <b>below</b> 、 <b>above</b> 、 <b>right</b> 、 <b>left</b> 或 <b>overlaid</b> 。默认值： <b>right</b>
<b>buttonstyle</b>	关键字	（仅用于 <b>type=radiobutton</b> 和 <b>checkbox</b> 的情况）指定域使用的符号： <b>check</b> 、 <b>cross</b> 、 <b>diamond</b> 、 <b>circle</b> 、 <b>star</b> 、 <b>square</b> 。默认值： <b>check</b>
<b>calcorder</b>	整型	（仅用于域具有计算事件的 <b>JavaScript</b> 动作的情况）指定此域相对于其他域的计算顺序。将先计算较小编码的域，然后计算较大编码的域。默认值： <b>10</b> 加上当前页面上使用的最大 <b>calcorder</b> （最初为 <b>10</b> ）
<b>caption</b>	内容字符串	（仅用于 <b>type=pushbutton</b> 的情况，对于下压按钮，必须提供一个标题或图标选项）标题文本，按钮没有输入焦点时将显示此文字。若不想有标题或图标，可用空字符串作为参数（即： <b>caption {}</b> ）。默认值： <b>none</b>
<b>captiondown</b>	内容字符串	（仅用于 <b>type=pushbutton</b> 的情况）启动按钮时显示的标题文字。默认值： <b>none</b>
<b>captionrollover</b>	内容字符串	（仅用于 <b>type=pushbutton</b> 的情况）按钮具有输入焦点时显示的标题文字。默认值： <b>none</b>
<b>charspacing<sup>1</sup></b>	浮点型	（不能用于 <b>type=radiobutton</b> 和 <b>checkbox</b> 的情况； <b>Acrobat 7</b> 忽略此项）域内文本的字符间距，用当前用户坐标系统的单位表示。默认值： <b>0</b>
<b>comb</b>	布尔型	（仅用于 <b>type=textfield</b> 的情况； <b>PDF 1.5</b> ）如果为 <b>true</b> ， <b>multiline</b> 、 <b>fileselect</b> 和 <b>password</b> 选项为 <b>false</b> ，且为 <b>maxchar</b> 选项提供了一个整数值，则针对具体的字符将域分割为几个等距离的子域（根据 <b>maxchar</b> 值）。默认值： <b>false</b>
<b>commitonselect</b>	布尔型	（仅用于 <b>type=listbox</b> 和 <b>combobox</b> 的情况； <b>PDF 1.5</b> ）如果为 <b>true</b> ，选中后将立即提交在列表中选中的项目。如果为 <b>false</b> ，将在退出域时提交所选项目。默认值： <b>false</b>

表 8.52 create\_field() 和 create\_fieldgroup() 的表单属性选项（续）

选项	类型	说明
currentvalue	（各种类型）	（不能用于 <b>type=pushbutton</b> 和 <b>signature</b> 的情况）域的初始值。类型和默认值取决于域类型。 <b>checkbox, radiobutton</b> （字符串）除 <b>Off</b> 之外的任何字符串均表示按钮已启动；如果指定了 <b>itemname</b> 并且 <b>/</b> 或者 <b>unisonselect</b> 为 <b>true</b> ，则 <b>Acrobat 6</b> 将显示不确定的行为。字符串 <b>Off</b> 表示按钮已停用。应为第一个按钮设置此选项。默认值： <b>Off</b> <b>textfield, combobox</b> （内容字符串）域的内容。默认值： <b>empty</b> <b>listbox</b> （整数列表） <b>itemtextlist</b> 内所选项目的索引。默认值： <b>none</b>
dasharray	浮点值列表	（仅用于 <b>borderstyle=dashed</b> 的情况）。虚线边框的虚线和间隙长度（以默认单位表示）（参见 <b>setdash()</b> ）。默认值： <b>3 3</b>
defaultvalue	（各种类型）	（仅用于 <b>type=textfield</b> 或 <b>combobox</b> 的情况；此选项的类型为内容字符串） <b>reset</b> 动作之后的域值。类型和默认值与 <b>currentvalue</b> 选项相同。例外：对于 <b>listbox</b> ，只允许一个整数值。
display	关键字	在屏幕和纸张上的可见性： <b>visible</b> 、 <b>hidden</b> 、 <b>noview</b> 、 <b>noprint</b> 。默认值： <b>visible</b>
editable	布尔型	（仅用于 <b>type=combobox</b> 的情况）如果为 <b>true</b> ，可以编辑框中当前选中的文本。默认值： <b>false</b>
exportable	布尔型	发生 <b>SubmitForm</b> 动作时，将导出域。默认值： <b>true</b>
fieldwarning	布尔型	如果为 <b>true</b> ，没有任何效果的域选项将引发非致命异常。如果为 <b>false</b> ，将忽略这些选项，而不返回任何信息。默认值： <b>true</b>
fileselect	布尔型	（仅用于 <b>type=textfield</b> 的情况）如果为 <b>true</b> ，域中的文本将视为一个文件名。默认值： <b>false</b>
fillcolor	颜色	文本的填充色。支持的色彩空间： <b>gray</b> 、 <b>rgb</b> 、 <b>cmymk</b> 。默认值： <b>gray 0 (=black)</b>
fitmethod	关键字	（仅用于 <b>type=pushbutton</b> 的情况）用按钮内的 <b>icon</b> 、 <b>icondown</b> 和 <b>iconrollover</b> 选项提供的模板的放置方法（默认值： <b>meet</b> ）： <b>auto</b> 如果模板可放入按钮内，则相当于 <b>meet</b> ；否则，相当于 <b>clip</b> 。 <b>nofit</b> 相当于 <b>clip</b> 。 <b>clip</b> 不缩放模板，而在域边框处剪切模板。 <b>meet</b> 将按比例缩放模板，使其可以置于按钮内。 <b>slice</b> 相当于 <b>meet</b> 。 <b>entire</b> 将缩放模板，使其可以完全置于按钮内。
字体	字体句柄	（必需的，除 <b>type=radiobutton</b> 和 <b>checkbox</b> 这两种总是使用 <b>ZapfDingbats</b> 的情况外）。指定域使用的字体。必须已在相应的 <b>load_font()</b> 调用中设置了下列选项： <b>embedding</b> （不需要嵌入的核心字体除外）、 <b>nosubsetting</b> 、 <b>noautocidfont</b> 。只允许下列编码方式： <b>8 位编码</b> 、 <b>Unicode CMaps</b> 、 <b>内置编码</b>
fontsize	浮点型或关键字	字体大小以用户坐标系衡量。如果提供了关键字 <b>auto</b> ，而非浮点值， <b>Acrobat</b> 将自动确定字体大小。默认值： <b>auto</b>
highlight	关键字	用户点击域时显示的高亮模式： <b>none</b> 、 <b>invert</b> 、 <b>outline</b> 、 <b>push</b> 。默认值： <b>invert</b>
icon	模板句柄 <sup>2</sup>	（仅用于 <b>type=pushbutton</b> 的情况，且必须提供一个标题或图标选项）模板的句柄，按钮没有输入焦点时将显示此句柄。默认值： <b>none</b>
icondown	模板句柄 <sup>2</sup>	（仅用于 <b>type=pushbutton</b> 的情况）模板的句柄，启动按钮时显示此句柄。默认值： <b>none</b>
iconrollover	模板句柄 <sup>2</sup>	（仅用于 <b>type=pushbutton</b> 的情况）模板的句柄，按钮具有输入焦点时显示此句柄。默认值： <b>none</b>

表 8.52 `create_field()` 和 `create_fieldgroup()` 的表单属性选项（续）

选项	类型	说明
<i>itemname</i>	超文本字符串	（仅用于 <b>type=radiobutton</b> 和 <b>checkbox</b> 的情况；如果导出值不是 <i>Latin 1</i> 字符串，是必需的）域的导出值。在一组中，多个单选按钮可具有相同的项目名称。 <b>Acrobat 6</b> ：即使位于不同的页面之上，一组中具有相同项目名称的复选框也会同时打开或关闭。默认值： <i>field name</i>
<i>itemnamelist</i>	超文本字符串	（仅用于 <b>type=listbox</b> 和 <b>combobox</b> 的情况，且是必需的）列表项目的导出值。多个项目可以具有相同的导出值。默认值： <i>none</i>
<i>itemtextlist</i>	内容字符串列表	（仅用于 <b>type=listbox</b> 和 <b>combobox</b> 的情况；这些情况下，要求此选项）列表内所有项目的文本内容。如果同时指定了 <i>itemnamelist</i> 和 <i>itemtextlist</i> ，则它们必须包含相同数量的字符串。
<i>linewidth</i>	整型	域边框的线宽，用默认单位表示（= <i>points</i> ）。默认值： <i>1</i>
<i>locked</i>	布尔型	如果为 <i>true</i> ，无法在 <b>Acrobat</b> 中编辑域属性。默认值： <i>false</i>
<i>maxchar</i>	整型或关键字	（仅用于 <b>type=textfield</b> 的情况）域内文本字符数量的上限；如果没有限制，则指定关键字 <i>unlimited</i> 。默认值： <i>unlimited</i>
<i>multiline</i>	布尔型	（仅用于 <b>type=textfield</b> 的情况）如果为 <i>true</i> ，文本将根据需要自动换行，生成多行。默认值： <i>false</i>
<i>multiselect</i>	布尔型	（仅用于 <b>type=listbox</b> 的情况）如果为 <i>true</i> ，可选择列表内的多个项目。默认值： <i>false</i>
<i>orientate</i>	关键字	域矩形内的内容的方向。 <i>north</i> 、 <i>west</i> 、 <i>south</i> 、 <i>east</i> 。默认值： <i>north</i>
<i>password</i>	布尔型	（仅用于 <b>type=textfield</b> 的情况）如果为 <i>true</i> ，将在输出时使用项目符号或星号模拟文本。默认值： <i>false</i>
<i>position</i>	浮点值或关键字列表	（仅用于 <b>type=pushbutton</b> 的情况）在按钮中，用 <i>icon...</i> 选项提供的模板的相对位置。默认值： <i>50 50</i>
<i>readonly</i> <sup>3</sup>	布尔型	如果为 <i>true</i> ，域不允许任何输入。默认值： <i>false</i>
<i>required</i>	布尔型	如果为 <i>true</i> ，提交表单时，域必须包含一个值。默认值： <i>false</i>
<i>richtext</i>	布尔型	（仅用于 <b>type=textfield</b> 的情况； <b>PDF 1.5</b> ）允许丰富的文本格式。如果为 <i>true</i> ， <i>fontsize</i> 不能为 <i>0</i> ，且 <i>fillcolor</i> 不能使用 <i>cmymk</i> 色彩空间。默认值： <i>false</i>
<i>scrollable</i>	布尔型	（仅用于 <b>type=textfield</b> 的情况）如果为 <i>true</i> ，且文本不能完全置于域内，文本将移至域外的隐藏区域。如果为 <i>false</i> ，文本填满整个域时，将不能接受任何更多的输入。默认值： <i>true</i>
<i>sorted</i>	布尔型	（仅用于 <b>type=listbox</b> 和 <b>combobox</b> 的情况）如果为 <i>true</i> ，将对列表的内容进行排序。默认值： <i>false</i>
<i>spellcheck</i>	布尔型	（仅用于 <b>type=textfield</b> 和 <b>combobox</b> 的情况）如果为 <i>true</i> ，将在域中启动拼写检查器。默认值： <i>true</i>
<i>strokecolor</i>	颜色	文本的描边色。支持的色彩空间： <i>gray</i> 、 <i>rgb</i> 、 <i>cmymk</i> 。默认值： <i>gray 0 (=black)</i> 。
<i>submitname</i>	超文本字符串	（建议只用于 <b>type=pushbutton</b> 的情况）表单将提交到的 <i>Internet</i> 地址的 <i>URL</i> 编码字符串。默认值： <i>None</i>
<i>taborder</i>	整型	指定此域相对于其他域的跳位顺序。将先移到较小编码的域，然后到达较大编码的域。默认值： <i>10</i> 加上当前页面上使用的最大 <i>taborder</i> （页面上的第一个域为 <i>10</i> ）；使用此默认值的结果是，创建顺序将指定跳位顺序。
<i>tooltip</i> <sup>3</sup>	超文本字符串	在域的工具提示中显示的文本。对于单选按钮和按钮组， <b>Acrobat</b> 将一直使用组中第一个按钮的工具提示；其余的将被忽略。默认值： <i>none</i>
<i>topindex</i>	整型	（仅用于 <b>type=listbox</b> 的情况）第一个显示的条目的索引。第一个项目的索引为 <i>0</i> 。默认值： <i>0</i>
<i>usercoordinates</i>	布尔型	如果为 <i>false</i> ，域坐标将使用默认坐标系（参见第 46 页上的第 3.2.1 节“坐标系”）；否则，将使用当前用户坐标系。默认值：全局 <i>usercoordinates</i> 参数的值



- 1. Acrobat 7 中会忽略此选项。
- 2. 可用 `begin_template()` 函数创建图标的模板：如果图标只包含一个图像，可向 `load_image()` 提供此选项，以创建模板。
- 3. 对于 `type=radiobutton` 的情况，此选项不能与 `create_field()` 一起使用，只能与 `create_fieldgroup()` 一起使用。如果 `fieldwarning=true`，且此选项与 `create_field()` 一起使用，将发出警告。

VB RB

Sub create\_fieldgroup(name As String, optlist As String)

C#

void create\_fieldgroup(String name, String optlist)

根据各种选项创建表单域组。

**name** （超文本字符串）表单域组的名称，也可作为另一组的前缀。可以任意级别嵌套域组。组名称必须用句点 «.» 字符分隔。在一个文档中，组名称必须是唯一的，且不能以句点 «.» 字符结尾。

**optlist** 选项列表，根据表 8.52 和表 8.53 指定域属性。

**详细说明** 域组对镜像一个或多个其他域中的域内容很有用。如果域组的名字将作为用 `create_field()` 创建的域名称的前缀提供一个域的名称，则新域将成为此组的一部分。域组 `optlist` 中提供的所有域属性选项都将被其属下所有的域继承。

**范围** 页面、文档

表 8.53 `create_fieldgroup()` 的域属性的其他选项

选项	类型	说明
<code>fieldtype</code>	关键字	组中包含的域的类型： <code>mixed</code> 、 <code>pushbutton</code> 、 <code>checkbox</code> 、 <code>radiobutton</code> 、 <code>listbox</code> 、 <code>combobox</code> 、 <code>textfield</code> 或 <code>signature</code> 。除非 <code>fieldtype=mixed</code> ，否则组中只能包含指定类型的域。如果已经为组指定了特定 <code>fieldtype</code> ，将在所有包含的域中同时显示当前值；即使这些域位于不同的页面上，也是如此。如果 <code>fieldtype=radiobutton</code> ，必须提供选项 <code>unisonselect</code> 。必须在域组选项中指定 <code>itemtextlist</code> 、 <code>itemnamelist</code> 、 <code>currentvalue</code> 和 <code>defaultvalue</code> 选项，而非在单独域的选项中指定。默认值： <code>mixed</code>
<code>toggle</code>	布尔型	（仅用于 <code>type=radiobutton</code> 的情况）如果为 <code>true</code> ，可通过单击启动或停用组内的单选按钮。如果为 <code>false</code> ，只能通过点击启动，通过点击另一按钮停用。默认值： <code>false</code>
<code>unisonselect</code>	布尔型	（仅用于 <code>type=radiobutton</code> 的情况；PDF 1.5）如果为 <code>true</code> ，将同时选中具有相同域名称或项目名称的单选按钮。默认值： <code>false</code>

8.9.5 书签

VB RB

Function create\_bookmark(text As String, optlist As String) As Long

C#

int create\_bookmark(String text, String optlist)

根据各种选项创建书签主题。

**text** （超文本字符串）包含书签的文本。`text` 的最大长度为 255 个单字节字符（8 位编码）或 126 个 Unicode 字符。但是，建议将 `text` 的实际限制设置为 32 个字符。

**optlist** 一个选项列表，根据表 8.54 指定书签的属性。

**返回** 生成书签的句柄，可在后续调用中与 `parent` 选项一起使用。

**详细说明** 此函数添加带有 `text` 的 PDF 书签。除非已指定了 `destination` 选项，否则书签将指向当前页（如果用于文档范围，将指向最后一页；如在第一页之前使用，将指向第一页）。

除非已经专门设置了另一种模式，否则创建书签的操作可将 `begin/end_document()` 的 `openmode` 选项设为 `bookmarks`。

范围 文档、页面

表 8.54 `create_bookmark()` 的选项

选项	类型	说明
<i>action</i>	动作列表	下列事件的书签动作列表（默认值： <code>GoTo</code> 动作且在 <i>destination</i> 选项中指定目标）： <i>activate</i> 启动书签时执行的动作。允许所有动作类型。
<i>destination</i>	选项列表	（若一活跃的动作被指定，将忽略此项）选项列表，根据表 8.50 指定书签目标。默认值：如果缺少 <i>destination</i> 、 <i>destname</i> 和 <i>action</i> ，默认值： <code>{type fitwindow page o}</code> 。
<i>destname</i>	超文本字符串	（可以为空；若 <i>destination</i> 选项被指定将忽略此项）用 <code>add_nameddest()</code> 定义的目标名称。 <i>Destination</i> 或 <i>destname</i> 动作将优先于此选项。如果 <i>destname</i> 为空字符串（即 <code>{}</code> ），且未指定 <i>destination</i> 或 <i>action</i> 选项，书签将不执行任何动作；如果将书签用作分隔符，这可能很有用。
<i>fontstyle</i>	关键字	指定书签文本的字体样式： <code>normal</code> 、 <code>bold</code> 、 <code>italic</code> 、 <code>bolditalic</code> 。默认值： <code>normal</code>
<i>index</i>	整型	在父级书签内插入书签的位置所处的索引。可使用处于 <code>o</code> 到同一级别的书签数量之间的值，在父级书签内的指定位置上插入书签。可使用值 <code>-1</code> ，作为当前级别的最后一个插入书签。默认值： <code>-1</code> 。但是，对于插入和恢复的页面，放置书签时会假定已经按照其物理顺序生成了所有页面（书签将反映页面顺序）。
<i>open</i>	布尔型	如果为 <code>false</code> ，将不显示从属书签。如果为 <code>true</code> ，将展开所有子级书签。默认值： <code>false</code>
<i>parent</i>	书签句柄	新书签将为句柄中指定书签的从属书签。如果 <i>parent</i> = <code>o</code> ，将创建顶级书签。默认值： <code>o</code>
<i>textcolor</i>	颜色	指定书签文本的颜色。支持的色彩空间： <code>none</code> 、 <code>gray</code> 、 <code>rgb</code> 。默认值： <code>rgb {o o o} (=black)</code>

### 8.9.6 文档信息域

**VB RB** Sub set\_info(key As String, value As String)

**C#** void set\_info(String key, String value)

用 *value* 填充信息域 *key*。

**key** （名称字符串）文档信息域的名称，可使用任何标准名称或任意的自定义名称（参见表 8.55）。自定义域的数量没有任何限制。对于自定义文档信息域的使用和语义，建议 PDFlib 用户参阅 Dublin Core Metadata 元素集。<sup>1</sup>

**value** （超文本字符串）*key* 将设为的字符串。Acrobat 会将 *value* 的最大长度强制设为 255 字节。请注意，由于 Windows 版 Adobe Reader 6 中的错误，无法在某些信息字符串中正确显示 `&` 字符。

详细说明 提供的信息值将只用于当前文档，不能用于在同一对象范围内生成的所有文档。

范围 对象、文档、页面。如果在对象范围内使用，提供的值将只用于下一文档。

1. 参见 [dublincore.org](http://dublincore.org)

表 8.55 文档信息域关键字的值

关键字	说明
<i>Subject</i>	文档的主题
<i>Title</i>	文档的标题
<i>Creator</i>	创建文档使用的软件（而非 <i>PDF</i> 的制作程序，它始终为 <i>PDFlib</i> ）。 <i>Acrobat 6</i> 会将此条目显示为 » <i>Application</i> «。
<i>Author</i>	文档的作者
<i>Keywords</i>	描述文档内容的关键字
<i>Trapped</i>	指示是否已对文档应用陷印。允许的值包括 <i>True</i> 、 <i>False</i> 和 <i>Unknown</i> 。
除 <i>CreationDate</i> 、 <i>Producer</i> 和 <i>ModDate</i> 之外的任何名称	用户定义域。 <i>PDFlib</i> 可支持任意数量的域。自定义域名称只能应用一次。如果同一域名称出现多次，将只使用最后一次出现的域。另请参见 <i>begin/end_document()</i> 的 <i>moddate</i> 选项。

8.9.7 不常用的超文本参数和函数

表 8.56 列出了不常用的超文本功能参数和值。

表 8.56 不常用的文档和超文本参数

函数	关键字	说明
<i>set_parameter</i>	<i>openaction</i>	使用 <i>begin/end_document()</i> 中的 <i>action</i> 选项的关键字 <i>open</i>
<i>set_parameter</i>	<i>openmode</i>	使用 <i>begin/end_document()</i> 中的 <i>openmode</i> 选项
<i>set_parameter</i>	<i>hidetoolbar</i> <i>hidemenubar</i> <i>hidewindowui</i> <i>fitwindow</i> <i>centerwindow</i> <i>displaydoctitle</i> <i>nonfullscreenpagemode</i> <i>direction</i> <i>viewarea</i> , <i>viewclip</i> <i>printarea</i> , <i>printclip</i>	使用 <i>begin/end_document()</i> 中的 <i>viewerpreferences</i> 选项
<i>set_parameter</i>	<i>bookmarkdest</i>	使用 <i>create_bookmark()</i> 中的 <i>action</i> 、 <i>destination</i> 、 <i>fontstyle</i> 和 <i>textcolor</i> 选项。
<i>set_parameter</i>	<i>transition</i>	使用 <i>begin/end_page_ext()</i> 中的 <i>transition</i> 选项
<i>set_value</i>	<i>duration</i>	使用 <i>begin/end_page_ext()</i> 中的 <i>duration</i> 选项
<i>set_parameter</i>	<i>base</i>	使用 <i>begin/end_document()</i> 中的 <i>uri</i> 选项
<i>set_parameter</i>	<i>launchlink:parameters</i> <i>launchlink:operation</i> <i>launchlink:defaultdir</i>	使用 <i>create_action()</i> 中的 <i>parameters</i> 、 <i>operation</i> 和 <i>defaultdir</i> 选项。

**Function add\_bookmark(text As String, parent As Long, open As Long) As Long**  
不常用，使用 *create\_bookmark()*。

**Sub add\_note(llx As Double, lly As Double, urx As Double, ury As Double, contents As String, title As String, icon As String, open As Long)**  
不常用，使用 *create\_annotation()*，且 *type=Text*。

---

**Sub attach\_file(lfx As Double, lfy As Double, urx As Double, ury As Double, filename As String, description As String, author As String, mimetype As String, icon As String)**

---

不常用，使用 `create_annotation()`，且 `type=FileAttachment`。`description` 参数对应于 `contents` 选项，`author` 参数对应 `title` 选项，`icon` 参数对应 `iconname` 选项。

---

**Sub add\_pdflink(lfx As Double, lfy As Double, urx As Double, ury As Double, filename As String, page As Long, optlist As String)**

---

不常用，使用 `create_action()` (`type=GoToR`) 和 `create_annotation()` (`type=Link`)。

---

**Sub add\_loccallink(lfx As Double, lfy As Double, urx As Double, ury As Double, page As Long, optlist As String)**

---

不常用，使用 `create_action()` (`type=GoTo`) 和 `create_annotation()` (`type=Link`)。

---

**Sub add\_launchlink(lfx As Double, lfy As Double, urx As Double, ury As Double, filename As String)**

---

不常用，使用 `create_action()` (`type=Launch`) 和 `create_annotation()` (`type=Link`)。

---

**Sub add\_weblink(lfx As Double, lfy As Double, urx As Double, ury As Double, url As String)**

---

不常用，使用 `create_action()` (`type=URI`) 和 `create_annotation()` (`type=Link`)。

---

**Sub set\_border\_style(style As String, width As Double)**

---

不常用，使用 `create_annotation()` 中的 `borderstyle` 和 `linewidth` 选项。

---

**Sub set\_border\_color(red As Double, green As Double, blue As Double)**

---

不常用，使用 `create_annotation()` 中的 `annotcolor` 选项。

---

**Sub set\_border\_dash(b As Double, w As Double)**

---

不常用，使用 `create_annotation()` 中的 `dasharray` 选项。

## 8.10 标签 PDF 的结构函数

`begin_document()` 中的 `tagged` 选项必须已设为 `true`，方可生成标签 PDF；同时，还必须提供 `lang` 选项。

VB RB

Function begin\_item(tag As String, optlist As String) As Long

C#

int begin\_item(String tag, String optlist)

用作选项提供的属性打开结构元素或其他内容项目。

**tag** 根据表 8.57 指定项目的元素类型。必须为当前 PDF 兼容级别所允许的一种标准结构类型或者是伪标签。

表 8.57 标准项目标签

种类	标签
分组	<i>Document</i> 、 <i>Part</i> 、 <i>Art</i> 、 <i>Sect</i> 、 <i>Div</i> 、 <i>BlockQuote</i> 、 <i>Caption</i> 、 <i>TOC</i> 、 <i>TOCI</i> 、 <i>Index</i> 、 <i>NonStruct</i> 、 <i>Private</i>
段落类型	<i>P</i> 、 <i>H</i> 、 <i>H1-H6 (BLSE)</i>
列表	<i>L</i> 、 <i>LI</i> 、 <i>Lbl</i> 、 <i>LBody (BLSE)</i>
表	<i>Table (BLSE)</i> 、 <i>TR</i> 、 <i>TH</i> 、 <i>TD</i> 、 <i>THead<sup>1</sup></i> 、 <i>TBody<sup>1</sup></i> 、 <i>TFoot<sup>1</sup></i>
内嵌级别	<i>Span</i> 、 <i>Quote</i> 、 <i>Note</i> 、 <i>Reference</i> 、 <i>BibEntry</i> 、 <i>Code</i> 、 <i>(ILSE)</i>
插图	<i>Figure</i> 、 <i>Formula</i> 、 <i>Form</i>
日语	<i>Ruby<sup>1</sup></i> （分组）、 <i>RB<sup>1</sup></i> 、 <i>RT<sup>1</sup></i> 、 <i>RP<sup>1</sup></i> 、 <i>Warichu<sup>1</sup></i> （分组）、 <i>WT<sup>1</sup></i> 、 <i>WP<sup>1</sup></i>
伪标签	下列标签可创建不属于结构元素的项目： <i>Artifact</i> 指定伪像，以区分实际的页面内容。 <i>ASpan</i> （可访问性范围，将作为 <i>Span</i> 写入 PDF，但必须区别于内嵌级别项目 <i>Span</i> ）可用于向不属于结构元素或只与结构元素部分相似的内容附加可访问性属性。 <i>ReversedChars</i> 指定从右到左书写的语言的文本使用相反的字符顺序。这样便可在 <i>Acrobat</i> 中搜索希伯来语或阿拉伯语文本。 <i>Clip</i> 指定标记的剪裁顺序。这只是个序列，只包含剪切路径或以渲染模式 7 显示的文本，而不包含显示的图形或 <i>save()</i> / <i>restore()</i> 。

1. 要求 PDF 1.5 或以上

**optlist** 选项列表，根据表 8.58 指定项目的详细信息。子级元素将继承所有可继承的设置，因此无需重复设置。因为此后便无法更改，所以必须在此处设置项目的所有属性。

返回值 项目句柄，可用于后续的项目相关调用。

详细说明 此函数可生成文档的结构树，这是标签 PDF 的根本。新元素在结构树中的位置可用 *parent* 和 *index* 选项控制。可以任意级别嵌套结构元素。常规项目不仅限于已在其上打开的页面，可在任意数量的页面上继续。

范围 对于内嵌项目，范围仅限于页面；常规项目还包括文档范围；必须始终与对应的 *end\_item()* 调用成对使用。只允许在标签 PDF 模式下使用此函数。

表 8.58 begin\_item() 的结构和伪标签属性选项

选项	类型	说明
Alt	超文本字符串	（除具有 <i>ASpan</i> 的 PDF 1.5 外，不能用于伪标签）内容项目的替代描述。应为插图和图像等提供替代描述，但不会识别为文本。为了实现可访问性，必须为图像提供替代文本。如果在 PDF 1.4 模式下使用此选项， <i>inline</i> 选项必须设为 <i>false</i> 。

表 8.58 `begin_item()` 的结构和伪标签属性选项

选项	类型	说明
<i>ActualText</i>	超文本字符串	（除具有 <i>ASpan</i> 的 <i>PDF 1.5</i> 外，不能用于伪标签；对于不兼容 <i>Unicode</i> 的字体的文本，这是必需的）与内容项目相当的替换文本。对于以某些非标准的方式显示的文本内容，应提供此选项，例如连字、插图中的花饰字符和首字下沉等。如果在 <i>PDF 1.4</i> 模式下使用此选项， <i>inline</i> 选项必须设为 <i>false</i> 。
<i>artifacttype</i>	关键字	（仅用于 <i>tag=Artifact</i> 的情况）标识内容项目的伪像类型： <i>Pagination</i> 、 <i>Layout</i> 或 <i>Page</i>
<i>Attached</i>	关键字列表	（仅用于 <i>tag=Artifact</i> 和 <i>artifacttype=Pagination</i> 的情况）一个列表，包含 1 到 4 个关键字 <i>Top</i> 、 <i>Bottom</i> 、 <i>Left</i> 和 <i>Right</i> 。
<i>BBox</i>	矩形	（仅用于 <i>tag=Artifact</i> 的情况以及所有表和插图标签；可选，但建议进行重排时使用）伪像的定界框；如果 <i>usercoordinates</i> 为 <i>false</i> ，用默认坐标系做参照；如果 <i>usercoordinates</i> 为 <i>true</i> ，用用户坐标系做参照。如果未提供此选项， <i>PDFlib</i> 将自动为导入的图像和 <i>PDF</i> 页面创建 <i>BBox</i> 条目。
<i>ColSpan</i>	整型	一个单元格所跨的表列数。
<i>E</i>	超文本字符串	（不能用于除 <i>ASpan</i> 的伪标签；结构标签要求 <i>PDF 1.5</i> ）内容项目的全名缩写的扩展。应提供此选项，以说明缩写。 <i>Acrobat</i> 的 <i>Read Aloud</i> 功能会把扩展文本视为单独的词；即使缺少显式断字符，也是如此。
<i>index</i>	整型	（不能用于伪标签）在父级元素内插入元素的位置所处的索引。可使用处于 0 到当前子级元素数量之间的值，在父级元素内的指定位置上插入项目。可使用值 -1，作为最后一个项目插入此元素。默认值：-1
<i>inline</i>	布尔型	（仅用于 <i>tag=ASpan</i> 的情况以及所有内嵌级别标签）如果为 <i>true</i> ，将以内嵌方式写入内容项目，而不创建结构元素。默认值： <i>true</i>
<i>Lang</i>	字符串	（不能用于除 <i>ASpan</i> 的伪标签）内容项目的语言标识符，使用有关 <i>lang</i> 选项的表 8.5 中描述的格式。此选项可用来覆盖具体内容项目的文档主要语言。
<i>parent</i>	项目句柄	（不能用于伪标签）另一 <i>begin_item()</i> 调用返回的元素父级的项目句柄。值 0 指结构树的根部。-1 指当前页面上、距上次打开时间最久的元素的父级。换言之， <i>parent=-1</i> 可打开当前元素的同级元素。默认值：-1
<i>RowSpan</i>	整型	一个单元格所跨的表行数。
<i>Title</i>	超文本字符串	（不能用于内嵌标签和伪标签）结构元素的名称

**VB RB** `Sub end_item(id As Long)`

**C#** `void end_item(int id)`

关闭结构元素或其他内容项目。

**id** 项目的句柄，必须由 *begin\_item()* 返回。

**详细说明** 所有内嵌项目都必须在页面结束之前关闭。所有常规项目都必须在文档结束之前关闭。但是，强烈建议完成操作后尽快关闭所有常规项目，以减少内存消耗。只有先关闭所有子级项目，才能关闭当前项目。如果项目关闭，它的父级将成为现用项目。

**范围** 对于内嵌项目，范围仅限于页面；常规项目还包括文档范围；必须始终与对应的 *begin\_item()* 调用成对使用。只允许在标签 *PDF* 模式下使用此函数。

VB RB

C#

Sub activate\_item(id As Long)

void activate\_item(int id)

启动以前创建的结构元素或其他内容项目。

**id** 项目的句柄，必须由 *begin\_item()* 返回，且尚未关闭。不能启动伪项目和内嵌级别项目。

详细说明 存储一个结构元素并在以后启动可实现更大的灵活性，以便高效地创建标签 PDF ；即使页面上由多个平行的结构分支。例如：中断主文本的多列版面或文本插入等。有关更多详细信息，参见第 151 页上的第 7.5.3 节 “启动复杂布局的项目”。

范围 文档、页面；只允许在标签 PDF 模式下使用此选项。





# A 相关文献

[1] Adobe Systems Incorporated: 《PDF Reference, Fifth Edition: Version 1.6》。可从 [partners.adobe.com/public/developer/pdf/index\\_reference.html](http://partners.adobe.com/public/developer/pdf/index_reference.html) 下载 PDF 版本

[2] 以下图书由 PDFlib 的主要作者编撰，只提供德语版本。本书讨论了各种 PostScript、PDF 和字体相关主题：

Thomas Merz, Olaf Drümmer: 《Die PostScript- & PDF-Bibel》。

Zweite Auflage。ISBN 3-935320-01-9, PDFlib Edition 2002

PDFlib GmbH, 80331 München, Tal 40, 传真: +49 • 89 • 29 16 46 86

可免费提供 PDF 版本，下载网址: [www.pdflib.com](http://www.pdflib.com)

如需订购印刷版本，请发邮件至 [books@pdflib.com](mailto:books@pdflib.com)





# B PDFlib 快速参考

## 常规参数

函数范例	页面
<i>Function begin_document(filename As String, optlist As String) As Long</i>	161
<i>Sub end_document(optlist As String)</i>	162
<i>Function get_buffer()</i>	164
<i>Sub begin_page_ext(width As Double, height As Double, optlist As String)</i>	165
<i>Sub end_page_ext(optlist As String)</i>	166
<i>Sub suspend_page(optlist As String)</i>	167
<i>begin_page_ext() 和 end_page_ext() 的选项</i>	166
<i>resume_page() 的选项</i>	167
<i>Sub set_value(key As String, value As Double)</i>	168
<i>Function get_parameter(key As String, modifier As Double) As String</i>	168
<i>Sub set_parameter(key As String, value As String)</i>	169
<i>Sub create_pvf(filename As String, data, optlist As String)</i>	169
<i>Function delete_pvf(filename As String) As Long</i>	169
<i>Function get_errnum() As Long</i>	170
<i>Function get_errmsg() As String</i>	170
<i>Function get_apiname() As String</i>	170

## 字体函数

函数范例	页面
<i>Function load_font(fontname As String, encoding As String, optlist As String) As Long</i>	172
<i>Sub setfont(font As Long, fontsize As Double)</i>	174
<i>Sub begin_font(fontname As String, a As Double, b As Double, c As Double, d As Double, e As Double, f As Double, optlist As String)</i>	174
<i>Sub end_font()</i>	175
<i>Sub begin_glyph(glyphname As String, wx As Double, llx As Double, lly As Double, urx As Double, ury As Double)</i>	175
<i>Sub end_glyph()</i>	175
<i>Sub encoding_set_char(encoding As String, slot As Long, glyphname As String, uv As Long)</i>	175

## 文本输出函数

函数范例	页面
<i>Sub set_text_pos(x As Double, y As Double)</i>	177
<i>Sub show(text As String)</i>	178
<i>Sub show_xy(text As String, x As Double, y As Double)</i>	178
<i>Sub continue_text(text As String)</i>	178
<i>Sub fit_textline(text As String, x As Double, y As Double, optlist As String)</i>	178
<i>Function stringwidth(text As String, font As Long, fontsize As Double) As Double</i>	181

函数范例	页面
<i>Function create_textflow(text As String, optlist As Long) As Long</i>	181
<i>Function fit_textflow(textflow As Long, llx As Double, lly As Double, urx As Double, ury As Double, optlist As String) As String</i>	182
<i>Function info_textflow(textflow As Long, keyword As String) As Double</i>	184
<i>fit_textline()</i> 的外观选项及 <i>create_textflow()</i> 的直接或内嵌选项	180

## 图形函数

函数范例	页面
<i>create_textflow()</i> 的选项	182
<i>Sub setdashpattern(optlist As String)</i>	190
<i>Sub setflat(flatness As Double)</i>	190
<i>Sub setlinejoin(linejoin As Long)</i>	190
<i>Sub setlinecap(linecap As Long)</i>	191
<i>Sub setmiterlimit(miter As Double)</i>	191
<i>Sub setlinewidth(width As Double)</i>	191
<i>Sub initgraphics</i>	192
<i>setdashpattern()</i> 的选项	190
<i>Sub restore()</i>	192
<i>Sub translate(tx As Double, ty As Double)</i>	193
<i>Sub scale(sx As Double, sy As Double)</i>	193
<i>Sub rotate(phi As Double)</i>	193
<i>Sub skew(alpha As Double, beta As Double)</i>	193
<i>Sub concat(a As Double, b As Double, c As Double, d As Double, e As Double, f As Double)</i>	194
<i>Sub setmatrix(a As Double, b As Double, c As Double, d As Double, e As Double, f As Double)</i>	194
<i>Function create_gstate(optlist As String) As Long</i>	194
<i>Sub set_gstate(gstate As Long)</i>	195
<i>Sub moveto(x As Double, y As Double)</i>	196
<i>Sub lineto(x As Double, y As Double)</i>	196
<i>Sub curveto(x1 As Double, y1 As Double, x2 As Double, y2 As Double, x3 As Double, y3 As Double)</i>	196
<i>Sub circle(x As Double, y As Double, r As Double)</i>	196
<i>Sub arc(x As Double, y As Double, r As Double, alpha As Double, beta As Double)</i>	197
<i>Sub arcn(x As Double, y As Double, r As Double, alpha As Double, beta As Double)</i>	197
<i>create_gstate()</i> 的选项	194
<i>Sub closepath()</i>	198
<i>Sub stroke()</i>	198
<i>Sub closepath_stroke()</i>	198
<i>Sub fill()</i>	198
<i>Sub fill_stroke()</i>	199
<i>Sub closepath_fill_stroke()</i>	199
<i>Sub clip()</i>	199
<i>Sub endpath()</i>	199

函数范例	页面
<i>Function define_layer(name As String, optlist As String) As Long</i>	199
<i>Sub set_layer_dependency(type As String, optlist As String)</i>	200
<i>Sub begin_layer(layer As Long)</i>	201
<i>Sub end_layer()</i>	201

## 颜色函数

函数范例	页面
<i>Sub setcolor(fstype As String, colorspace as String, c1 As Double, c2 As Double, c3 As Double, c4 As Double)</i>	202
<i>Function makespotcolor(spotname As String) As Long</i>	203
<i>Function load_iccprofile(profilename As String, optlist As String) As Long</i>	203
<i>Function begin_pattern(width As Double, height As Double, xstep As Double, ystep As Double, painttype As Long) As Long</i>	205
<i>Sub end_pattern()</i>	205
<i>Function shading_pattern(shading As Long, optlist As String) As Long</i>	205
<i>Sub shfill(shading As Long)</i>	206
<i>Function shading(shtype As String, xo As Double, yo As Double, x1 As Double, y1 As Double, c1 As Double, c2 As Double, c3 As Double, c4 As Double, optlist As String) As Long</i>	206

## 图像函数

函数范例	页面
<i>Function load_image(imagetype As String, filename As String, optlist As String) As Long</i>	207
<i>Sub close_image(image As Long)</i>	211
<i>Sub fit_image(image As Long, x As Double, y As Double, optlist As String)</i>	211
<i>Function begin_template(width As Double, height As Double) As Long</i>	213
<i>Sub end_template()</i>	213
<i>Sub add_thumbnail(image As Long)</i>	213

## PDF 导入函数 (PDI)

函数范例	页面
<i>Function open_pdi(filename As String, optlist As String, len As Long) As Long</i>	214
<i>Sub close_pdi(doc As Long)</i>	215
<i>Function open_pdi_page(doc As Long, pagenumber As Long, optlist As String) As Long</i>	215
<i>Sub close_pdi_page(page As Long)</i>	215
<i>Sub fit_pdi_page(page As Long, x As Double, y As Double, optlist As String)</i>	216
<i>Function process_pdi(doc As Long, page As Long, optlist As String) As Long</i>	216
<i>Function get_pdi_value(key As String, doc As Long, page As Long, reserved As Long) As Double</i>	217
<i>Function get_pdi_parameter( key As String, doc As Long, page As Long, reserved As Long) As String</i>	217

## 块填充函数 (PPS)

函数范例	页面
<i>Function fill_textblock( page As Long, blockname As String, text As String, optlist As String) As Long</i>	219
<i>open_pdi_page()</i> 的选项	216
<i>Function fill_pdfblock( page As Long, blockname As String, contents As Long, optlist As String) As Long</i>	220

## 超文本函数

函数范例	页面
<i>process_pdi()</i> 的选项	217
<i>Sub add_nameddest(name As String, optlist As String)</i>	224
<i>Sub create_annotation(llx As Double, lly As Double, urx As Double, ury As Double, type As String, optlist As String)</i>	225
<i>Sub create_field(llx As Double, lly As Double, urx As Double, ury As Double, name As String, type As String, optlist As String)</i>	229
<i>Sub create_fieldgroup(name As String, optlist As String)</i>	233
<i>Function create_bookmark(text As String, optlist As String) As Long</i>	233
<i>create_action()</i> 的动作属性的选项	222

## 加标签 PDF 和结构函数

函数范例	页面
<i>Function begin_item(tag As String, optlist As String) As Long</i>	237
<i>Sub end_item(id As Long)</i>	238
<i>create_fieldgroup()</i> 的域属性的其他选项	233

## 参数和值

种类	函数	关键字
设置	<i>set_parameter</i>	<i>resourcefile</i> 、 <i>SearchPath</i> 、 <i>license</i> 、 <i>licensefile</i> 、 <i>warning</i> 、 <i>trace</i> 、 <i>tracefile</i> 、 <i>logmsg</i>
	<i>set_value</i>	<i>compress</i>
版本	<i>get_value</i>	<i>major</i> 、 <i>minor</i> 、 <i>revision</i>
	<i>get_parameter</i>	<i>version</i>
页面	<i>get_value</i>	<i>pagewidth</i> 、 <i>pageheight</i>
字体	<i>set_parameter</i>	<i>FontAFM</i> 、 <i>FontPFM</i> 、 <i>FontOutline</i> 、 <i>Encoding</i> 、 <i>fontwarning</i> 、 <i>kerning</i> 、 <i>autosubsetting</i> 、 <i>autocidfont</i> 、 <i>textformat</i> 、 <i>unicodemap</i>
	<i>get_parameter</i>	<i>fontname</i> 、 <i>fontencoding</i> 、 <i>fontstyle</i> 、 <i>textformat</i>
	<i>set_value</i>	<i>subsetlimit</i> 、 <i>subsetminsize</i>
	<i>get_value</i>	<i>ascender</i> 、 <i>capheight</i> 、 <i>descender</i> 、 <i>font</i> 、 <i>fontsize</i> 、 <i>fontmaxcode</i> 、 <i>monospace</i>
文本	<i>set_value</i>	<i>leading</i> 、 <i>textrise</i> 、 <i>horizscaling</i> 、 <i>textrendering</i> 、 <i>charspacing</i> 、 <i>wordspacing</i> 、 <i>italicangle</i> 、 <i>underlinewidth</i> 、 <i>underlineposition</i>
	<i>get_value</i>	<i>leading</i> 、 <i>textrise</i> 、 <i>horizscaling</i> 、 <i>textrendering</i> 、 <i>charspacing</i> 、 <i>wordspacing</i> 、 <i>textx</i> 、 <i>texty</i> 、 <i>italicangle</i> 、 <i>underlinewidth</i> 、 <i>underlineposition</i>
	<i>set_parameter</i>	<i>autospace</i> 、 <i>underline</i> 、 <i>overline</i> 、 <i>strikeout</i> 、 <i>kerning</i> 、 <i>glyphwarning</i>
	<i>get_parameter</i>	<i>underline</i> 、 <i>overline</i> 、 <i>strikeout</i> 、 <i>fontstyle</i>

种类	函数	关键字
图形	set_ parameter	fillrule、topdown
	get_parameter	scope
	get_value	currentx、currenty、ctm_a、ctm_b、ctm_c、ctm_d、ctm_e、ctm_f
颜色	set_ parameter	iccwarning、honoriccprofile、ICCProfile、StandardOutputIntent、renderingintent、preserveoldpantonenames、spotcolorlookup
	set_value	defaultgray、defaultrgb、defaultcmymk、setcolor:iccprofilegray、setcolor:iccprofilergb、setcolor:iccprofilecmyk
	get_value	image:iccprofile、icccomponents
图形	get_value	imagewidth、imageheight、orientation、resx、resy
	set_ parameter	imagewarning
PDI	get_parameter	pdi
	set_ parameter	pdiwarning
	get_pdi_value	/Root/Pages/Count、/Rotate、version、width、height CropBox、BleedBox、ArtBox、TrimBox: 这些关键字后必须跟随斜线 '/' 字符并附加 lly、lly、urx 或 ury 之一，例如: CropBox/lly
	get_pdi_ parameter	filename、/Info/<key>、vdp/Blocks/<blockname>/<propertyname>、vdp/Blocks/<blockname>/Custom/<propertyname>
	set_ parameter	usercoordinates
超文本	set_ parameter	

# C 修订历史记录

日期	更改
2006 年 2 月 23 日	▶ 有关 <i>PDFlib 6.0.3</i> 的各种更新和修正，并增加 <i>RUBY</i> 绑定的扩展
2005 年 8 月 9 日	▶ 有关 <i>PDFlib 6.0.2</i> 的各种更新和修正
2004 年 11 月 17 日	▶ 有关 <i>PDFlib 6.0.1</i> 的小范围更新和修正 ▶ 第 8 章中介绍了语言特定函数范例的新格式 ▶ 第 3 章增加了超文本示例
2004 年 6 月 18 日	▶ 重要的 <i>PDFlib 6</i> 更改
2004 年 1 月 21 日	▶ 有关 <i>PDFlib 5.0.3</i> 的小范围内容增加和修正
2003 年 9 月 15 日	▶ 有关 <i>PDFlib 5.0.2</i> 的小范围内容增加和修正，并增加了块规范
2003 年 5 月 26 日	▶ 有关 <i>PDFlib 5.0.1</i> 的小范围更新和修正
2003 年 3 月 26 日	▶ 有关 <i>for PDFlib 5.0.0</i> 的小范围更改和内容改写
2002 年 6 月 14 日	▶ 有关 <i>PDFlib 4.0.3</i> 的小范围更改以及有关 <i>.NET</i> 绑定的扩展
2002 年 1 月 26 日	▶ 有关 <i>PDFlib 4.0.2</i> 的小范围更改以及有关 <i>IBM eServer</i> 版本的扩展
2001 年 5 月 17 日	▶ 有关 <i>PDFlib 4.0.1</i> 的小范围更改
2001 年 4 月 1 日	▶ 增添 <i>PDFlib 4.0.0</i> 的 <i>PDI</i> 及其他功能
2001 年 2 月 5 日	▶ 增添 <i>PDFlib 3.5.0</i> 中的模板和 <i>CMYK</i> 功能
2000 年 12 月 22 日	▶ <i>ColdFusion</i> 文档以及 <i>PDFlib 3.03</i> 相关增加内容；编制专门的 <i>COM</i> 版手册
2000 年 8 月 8 日	▶ <i>Delphi</i> 文档及有关 <i>PDFlib 3.02</i> 的小范围内容增加
2000 年 7 月 1 日	▶ 有关 <i>PDFlib 3.01</i> 的增加内容和相关澄清
2000 年 2 月 20 日	▶ 有关 <i>PDFlib 3.0</i> 的更改
1999 年 8 月 2 日	▶ 有关 <i>PDFlib 2.01</i> 的小范围更改及内容增加
1999 年 6 月 29 日	▶ 增添特定语言绑定的专门章节 ▶ 有关 <i>PDFlib 2.0</i> 的扩展
1999 年 2 月 1 日	▶ 有关 <i>PDFlib 1.0</i> 的小范围更改（未公开发布）
1998 年 8 月 10 日	▶ 有关 <i>PDFlib 0.7</i> 的扩展（仅面向一位客户）
1998 年 7 月 8 日	▶ 第一次尝试描述 <i>PDFlib 0.6</i> 中的 <i>PDFlib</i> 脚本支持
1998 年 2 月 25 日	▶ 小幅扩充手册以涵盖 <i>PDFlib 0.5</i>
1997 年 9 月 22 日	▶ 第一次公开发布 <i>PDFlib 0.4</i> 及本手册



# 索引

## 数字

16 位编码 76

8 位编码 72

## A

*activate\_item()* 239

*Active Server Pages* 22

    特别注意事项 22

*add\_nameddest()* 224

*add\_thumbnail()* 213

Adobe 字体规格 (AFM) 66

AFM (Adobe 字体规格) 66

All 专色名称 203

alpha 通道 108

*alphaissshape gstate* 选项 194

*antialias* 选项 206

API (应用程序编程接口)

    参考 157

*arc()* 197

*arcn()* 197

*ArtBox* 48

*ascender* 81

*ascender* 参数 171

ASP.NET 29, 35

    特别注意事项 35

*Author* 域 235

*autocidfont* 参数 71, 72, 171

*autospace* 函数 176

*autosubsetting* 参数 71, 72, 171

安全性 142

安装, 无提示 21

## B

*begin\_document()* 161

*begin\_font()* 174

*begin\_glyph()* 175

*begin\_item()* 237

*begin\_layer()* 201

*begin\_page\_ext()* 165, 166

*begin\_pattern()* 205

*begin\_template()* 213

Bezier 曲线 196

*BleedBox* 48

*blendmode gstate* 选项 194

BMP 107

Borland Delphi

    请参见 Delphi

*builtin* 编码 75

*byteserving* 144

绑定 19

编码 72

    CJK 86

    自定义 74

编码参数 171

表单域

    转换为块 126

标准页面大小 48

不可见文本 177

## C

C 绑定 30

C++ 绑定 31

*capheight* 81

    参数 171

CCITT 107

CFF (Compact Font Format) 63

*charref* 参数 176

*charspacing* 参数 176

CIE L\*a\*b\* 色彩空间 53

CJK (中文、日文、韩文)

    标准字体 84

    自定义字体 87

*circle()* 196

    与 VB 有关的问题 24

*clip()* 199

*close\_image()* 211

*closepath()* 198

*closepath\_fill\_stroke()* 199

*closepath\_stroke()* 198

*close\_pdi()* 215

*close\_pdi\_page()* 215

CMap 84, 86

CMYK 颜色 50

Cobol 绑定 19

*compress* 参数 160

COM (组件对象模型) 绑定 20

*concat()* 194

*continue\_text()* 178

CPI (字符/英寸) 82

*create\_action()* 222

*create\_annotation()* 225

*create\_bookmark()* 233

*create\_field()* 229

*create\_fieldgroup()* 233

*create\_gstate()* 194

*create\_pvf()* 169

*create\_textflow()* 181

Creator 域 235

*CropBox* 48

*currentx* 和 *currenty* 参数 81, 195  
*curveto()* 196  
裁剪框 218  
裁切框 218  
参数处理函数 168  
超文本字符串 77  
程序结构 39  
出血框 218  
垂直书写模式 85, 86  
错误处理 40  
    利用 .NET 32

## D

*defaultgray/rgb/cmyk* 参数 55  
*define\_layer()* 199  
*delete\_pvf()* 169  
*delete\_textflow()* 185  
*Delphi* 27  
    特别注意事项 27  
*descender* 81  
*descender* 参数 171  
DLL (动态链接库) 21  
*dpi* 计算 105  
*Dublin Core* 234  
打印机字体 *ASCII (PFA)* 66  
打印机字体二进制 (*PFB*) 66  
打印机字体规格 (*PFM*) 66  
代码页  
    基于 *Unicode* 的 76  
    *Microsoft Windows 1250-1258* 73  
单位 46  
当前点 49  
导入 *PDF* 文档中的信息关键字 218  
等宽字体 82  
对称 193  
对齐方式 (*position* 选项) 179  
多页图像文件 110

## E

*ebcdic* 编码 73  
*encoding\_set\_char()* 175  
*end\_document()* 162  
*end\_font()* 175  
*end\_glyph()* 175  
*end\_item()* 238  
*end\_layer()* 201  
*endpath()* 199  
*end\_pattern()* 205  
*end\_template()* 213  
*EUDC* 字体 67  
*EUDC* (终端用户定义的字符) 88  
*extendo* 和 *extendi* 选项 206

## F

*fill()* 198  
*fill\_imageblock()* 220

*fill\_pdfblock()* 220  
*fillrule* 参数 198  
*fill\_stroke()* 199  
*fill\_textblock()* 219  
*fit\_image()* 211  
*fit\_pdi\_page()* 216  
*fit\_textflow()* 182  
*fit\_textline()* 178  
*flatness gstate* 选项 194  
*FontAFM* 参数 171  
*fontencoding* 参数 171  
*fontmaxcode* 参数 75, 171  
*fontname* 参数 171  
*FontOutline* 参数 171  
*FontPFM* 参数 171  
*fontsize* 函数 171  
*FontSpecific* 编码 75  
*fontstyle* 参数 171  
*fontwarning* 参数 41, 171  
*form XObjects* 49  
范围 39  
分色色彩空间 50

## G

*gaiji* 字符 63  
*get\_apiname()* 170  
*get\_buffer()* 45, 164  
*get\_errmsg()* 170  
*get\_errnum()* 170  
*get\_parameter()* 168  
*get\_pdi\_parameter()* 217  
*get\_value()* 168  
*GIF* 107  
*global.asa* 22  
*glyphwarning* 参数 176  
*grid.pdf* 46  
*gstate* 205  
公制坐标 46  
光栅图像  
    函数 207  
规格 81

## H

*HKS* 专色 53  
*honoriccprofile* 参数 207  
*horizscaling* 参数 176  
*HostFont* 参数 171  
*HTML* 字符引用 79  
函数范围 39  
韩文 84, 86  
行  
    虚线和样式 189  
行的虚线样式 189  
行间距 81  
毫米 46  
核心字体 70  
横向模式 167

## J

*Java* 绑定 31  
*icccomponents* 参数 204  
*ICCPProfile* 参数 204  
*iccwarning* 参数 204  
*JFIF* 106  
*ignoremask* 109  
*image:iccpfile* 参数 55, 207  
*imagewarning* 参数 207  
*imagewidth* 和 *imageheight* 参数 207  
*info\_textflow()* 184  
*initgraphics()* 192  
*Internet* 服务提供商 21  
*JPEG* 106  
*JPEG2000* 106  
*JScript* 22  
*ISO 10646* 76  
*ISO 15930\**n* 145  
*ISO 8859-2* 到 *ISO 8859-15* 73  
*italicangle* 参数 176  
基线压缩 106  
基于 *ICC* 的颜色 50  
加密 142  
渐变 51  
剪切 49  
镜像 193

## K

*kerning* 参数 82, 176  
*Keywords* 域 235  
可移植文档格式 (*PDF*) 参考手册 241  
口令 142  
块 119  
    属性 121  
    增效工具 119  
块的变量数据处理 219  
快速 *Web* 查看 161

## L

*leading* 81  
*leading* 参数 176  
*license* 参数 160  
*licensefile* 参数 160  
*linecap* *gstate* 选项 194  
*linejoin* *gstate* 选项 195  
*lineto()* 196  
*linewidth* *gstate* 选项 195  
*load\_font()* 172  
*load\_iccpfile()* 203  
*load\_image()* 207  
*LWFN* (*LaserWriter* 字体) 66  
利用块进行数据变量处理 119  
临时磁盘空间要求 144  
路径 49

绘制和剪切 198  
轮廓文本 177

## M

*Mac OS*  
    *UPR* 配置 43  
*macroman* 编码 72, 73  
*macroman\_apple* 编码 76  
*major* 参数 160  
*makepsres* 实用工具 43  
*makespotcolor()* 203  
*masterpassword* 143  
*MediaBox* 48  
*metadata* 163  
*Microsoft Transaction Server* 20  
*minor* 参数 160  
*miterlimit* *gstate* 选项 195  
*monospace* 参数 171  
*moveto()* 196  
*MSI* 21  
*MTS* 20  
“每次一页” 下载 144  
蒙版 108  
蒙版的 108  
蒙版图像 108  
描边 49  
名称字符串 78  
模板 49  
默认坐标系统 46

## N

*N* 选项 206  
*.NET* 绑定 31  
    错误处理 32  
*None* 专色名称 203  
内嵌图像 105  
内容字符串 77

## O

*opacityfill* *gstate* 选项 195  
*opacitystroke* *gstate* 选项 195  
*open\_pdi()* 214  
*open\_pdi\_page()* 215  
*OpenType* 字体 63  
*orientation* 参数 207  
*overline* 参数 83, 177  
*overprintfill* *gstate* 选项 195  
*overprintmode* *gstate* 选项 195  
*overprintstroke* *gstate* 选项 195  
欧元字符 76

## P

*PANTONE* 专色 52  
*pcircle()* 196  
*PDF* 导入函数 213  
*PDF* 导入库 (*PDI*) 110, 213

PDF 的导入函数 213  
PDF/X 145  
    导入 PDF 文档 148  
    输出方法 217  
PDF/X 的标准输出条件 147  
PDF/X 的输出方法 146  
PDF/X 的输出条件 146  
PDFlib  
    程序结构 39  
    功能 13  
PDFlib 程序结构 39  
PDFlib 的功能 13  
PDFlib 的可用性 19  
PDFlib Personalization Server 119, 219  
pdflib.upr 45  
PDFLIBRESOURCE 环境变量 44  
PDI 110, 213  
PDI 的 filename 参数 218  
PDI 的 pdfx 参数 218  
PDI 的 vdp/Block 参数 219  
PDI 的 vdp/blockcount 参数 219  
PDI 的 version 参数 218  
PDI 的宽度和高度参数 218  
pdi parameter 218  
pdiwarning 参数 112, 218  
pdiusebox 112  
pdiusebox 参数 218  
Perl 绑定 36  
PFA (打印机字体 ASCII) 66  
PFB (打印机字体二进制) 66  
PFM (打印机字体规格) 66  
PHP 绑定 36  
PNG 106, 108  
PostScript 字体 63, 66  
PPS (PDFlib Personalization Server) 119  
PPS (PDFlib Personalization Server) 219  
preserveoldpantoneparameter 202  
print\_glyphs.ps 67  
process\_pdi() 216  
pscale() 193  
Python 绑定 36  
评估图章 9  
平滑混合 51  
平台 19

## Q

嵌入系统 19  
嵌入字体 70  
权限 142, 143

## R

ro 和 ri 选项 206  
REALbasic 绑定 36  
rect() 197  
regsvr32 21  
renderingintent 参数 207  
renderingintent gstate 选项 195

renderingintent 选项 54  
resourcefile 参数 45, 160  
restore() 192  
resume\_page() 167  
resx 和 resy 参数 207  
RGB 颜色 50  
rotate() 193  
RPG 绑定 38  
Ruby binding 38  
日文 84, 86  
软蒙版 108

## S

save() 192  
scale() 193  
    与 VB 有关的问题 24  
scope 参数 160  
SearchPath 参数 44, 159, 171  
setcolor  
    iccprofilegray/rgb/cmyk parameters 55  
setcolor() 202  
setcolor:iccprofilegray/rgb/cmyk 参数 204  
setdash() 189  
setdashpattern() 190  
setflat() 190  
setfont() 174  
set\_gstate() 195  
set\_info() 234  
set\_layer\_dependency() 200  
setlinecap() 191  
setlinejoin() 190  
setlinewidth() 191  
setmatrix() 194  
setmiterlimit() 191  
set\_parameter() 169  
set\_text\_pos() 177  
set\_value() 168  
shading() 206  
shading\_pattern() 205  
shfill() 206  
show() 178  
show\_xy() 178  
skew() 193  
smoothness gstate 选项 195  
SPIFF 106  
spotcolorlookup 参数 202  
sRGB 色彩空间 55  
StandardOutputIntent 参数 204  
strikeout 参数 83, 177  
stringwidth() 181  
stroke() 198  
strokeadjust gstate 选项 195  
Subject 域 235  
subsetlimit 参数 72, 171  
subsetminsize 参数 72, 171  
suspend\_page() 167  
Symbol 字体 75

上标 81, 177  
商业许可证 10  
设置函数 159  
输出准确性 49  
书写模式 85, 86  
水平书写模式 85, 86  
说明符 70  
宿主编码 72  
宿主字体 65  
缩放图像 105  
缩览图 213  
索引色彩空间 51

## T

Tcl 绑定 38  
textformat 参数 177  
textknockout gstate 选项 195  
textrendering 参数 84, 177  
textrise 参数 177  
textx 和 texty 参数 81, 86, 177  
TIFF 107  
    多页 110  
Title 域 235  
topdown 参数 47, 160  
ToUnicode CMap 65, 77  
trace、tracefile、logmsg 参数 160  
translate() 193  
Trapped 域 235  
TrimBox 48  
TrueType 字体 63  
TTC (TrueType 集合) 67, 87, 88  
TTF (TrueType 字体) 63  
Type 1 字体 66  
Type 3 (用户定义的) 字体 68  
提示标记 9  
填充 49  
透明度 108  
    问题 109  
图案 51  
图层和 PDI 112  
图像函数 207  
图像蒙版 108, 109  
图像数据, 重用 105  
图像缩放 105  
图像文件格式 106  
图形函数 189  
图形状态  
    显式 194  
图形状态函数 189

## W

U+XXXX 编码 77  
warning 参数 41, 170  
VB.NET 33  
VBA 20  
VBScript 26  
version 参数 160

winansi 编码 73  
Windows 安装程序 21  
Windows 的样式名称 67  
Windows 的字体样式名称 67  
Windows Script Host (WSH) 25  
Visual Basic 24  
Visual Basic for Applications 20  
UNC 22  
underline 参数 83, 177  
underlineposition 参数 177  
underlinewidth 参数 177  
Unicode 76  
unicodemap 参数 77, 172  
wordspacing 参数 177  
UPR (Unix PostScript 资源) 42  
    文件格式 43  
    文件搜索 44  
usercoordinates 参数 46, 222  
userpassword 143  
网页优化 PDF 161  
网页优化的 PDF 144  
伪字体样式 83  
文本变体 81  
文本规格 81  
文本函数 171  
文本流  
    内嵌选项列表 185  
文本流的内嵌选项列表 185  
文本位置 81, 177  
文档和页面函数 160  
文档信息域 234  
无提示安装 21

## X

xheight 81  
XMP 元数据 163  
XObject 49  
下标 81, 177  
线程模型 20  
显式透明度 108  
显式图形状态 194  
线性化 PDF 161  
线性化的 PDF 144  
像素取样 105  
信息域 234  
许可证 PDFlib 和 PDI 9  
渲染方法 54  
选项列表 157  
选项列表中的布尔值 158  
选项列表中的单字符值 158  
选项列表中的动作列表 159  
选项列表中的浮点型和整型值 158  
选项列表中的关键字 158  
选项列表中的句柄 158  
选项列表中的矩形 159  
选项列表中的列表值 158  
选项列表中的颜色值 159  
选项列表中的字符串值 158

旋转导入 *PDF* 页面中的项目 218

旋转对象 47

## Y

亚洲字体包 84

颜色 50

颜色函数 202

演示图章 9

页 110

页格式 48

页面大小格式 48

*Acrobat* 中的限制 48

页面说明 46

异常 40

隐式透明度 108

英寸 46

用户定义的 (*Type 3*) 字体 68

用户空间 46

用于建立块的 *Acrobat* 增效工具 119

用于建立块的增效工具 119

优化的 *PDF* 144

语言绑定

    请参见绑定

原始图像数据 108

## Z

*ZapfDingbats* 字体 75

着色 51

值

    参见参数

中文 84, 86

专色 (分色色彩空间) 50, 51

自定义编码 74

自动化 20

字符 / 英寸 82

字符规格 81

字符集 72

字符名称 67

字符引用 79

字距调整 82

子路径 49

自上而下的坐标 47

字体

*AFM* 文件 66

    等宽 82

*OpenType* 63

*PDF* 核心集 70

*PFA* 文件 66

*PFB* 文件 66

*PFM* 文件 66

*PostScript* 63, 66

    嵌入 70

    说明符 70

*TrueType* 63

*Type 1* 66

*Type 3* 68

*Type 3* (用户定义的) 字体 68

*Unicode* 支持 76

亚洲字体包 84

用户定义的 (*Type 3*) 68

有关嵌入的法律事宜 71

字形名称 67

资源配置 42

字体参数 171

字体规格 81

字体样式 83

字体子集化 71

字形 *ID* 寻址 75

资源的类别 43

资源类别 43

坐标范围 49

坐标系统 46

    公制 46

    自上而下 47

作品框 218