

# 原创经典，程序员典藏

资深Java程序员耗时一年时间写作，十年开发经验毫无保留  
全面、系统、深入介绍Java Web开发的核心技术与主流框架

赠送6小时Java Web开发教学视频  
赠送书中涉及的开源工具包、高质量源代码



## 作者介绍

刘京华 毕业于山东大学计算机科学与技术学院。先后任职于用友软件、IBM中国研究院、Symbian中国研究院等公司或机构。毕业后一直从事Java开发，有超过10年的Java开发经验。精通Java Web开发的各种技术与框架，对图形化开发、Eclipse插件开发等各种技术也有深入研究。曾经开发过大型的分布式Web系统。参与开发过中国移动、中国联通、中国电信的网关系统。参与开发过物流管理系统、供应链处理系统、联想公司的RMS系统、IBM的SmartSCOR和Nokia Carbide平台。

ISBN: 978-7-302-20976-8

定价: 99.80元 (附DVD视频光盘1张)

## 第 40 章 PDF 组件 iText

本章将学习 Java 的 PDF 开源框架 iText。通过实际的编程例子来学习如何生成 pdf 文档，包括设置 pdf 属性、在文档中显示格式化文本、显示图表、显示图像等。完成本章的学习后，可以运用这些基本知识将数据转化成 pdf 文件。

### 40.1 下载 iText 组件

iText 组件是著名的开放源码的站点 sourceforge 一个项目，是用于生成 PDF 文档的一个 Java 类库。通过 iText 不仅可以生成 pdf 或 rtf 的文档，而且可以将 XML、Html 文件转化为 pdf 文件。iText 的官方网址是 <http://www.lowagie.com/iText/>，如图 40.1 所示。

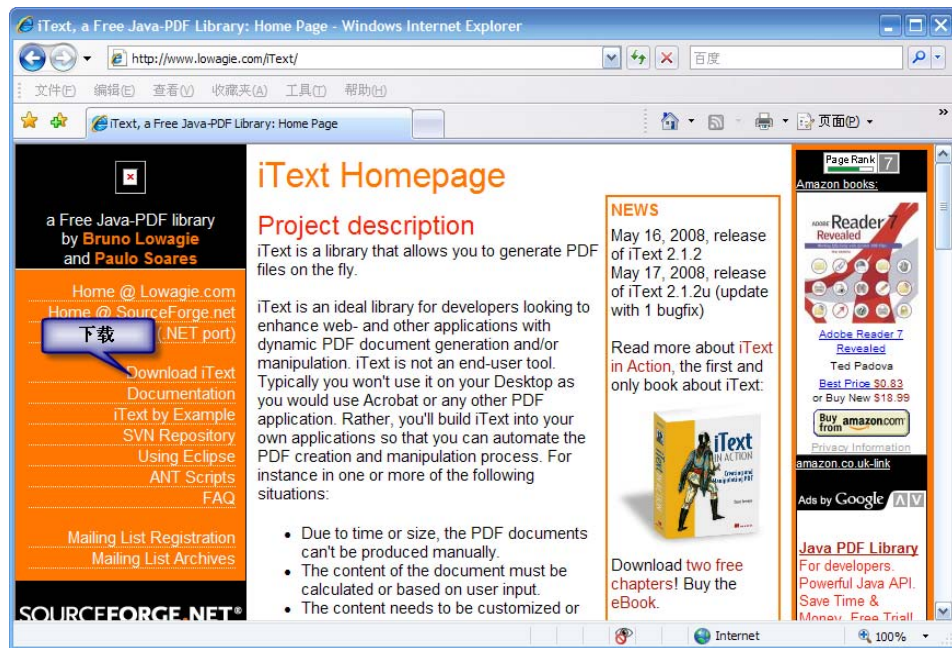


图 40.1 iText 官方主页

在项目主页 <http://sourceforge.net/projects/itext/> 中单击“Download”链接，下载最新版本的 iText 类库。下载完成之后会得到一个 iText-2.1.2u.jar 包，运行程序时把这个包加入类路径中即可。

如果生成的 PDF 文件中需要出现中文、日文、韩文字符，则还需要下载 iTextAsian.jar 包，运行时也将其导入到类路径中。

## 40.2 输出 pdf 的基本步骤

pdf 文档输出有一个基本的框架。首先看下面这个程序，它用 iText 生成一个简单的 pdf 文档，代码如下：

代码 40.1 FirstPDF.java

```
package com.helloweenvsfei.itext;
import java.io.FileOutputStream;
import com.lowagie.text.Document;
import com.lowagie.text.PageSize;
import com.lowagie.text.Paragraph;
import com.lowagie.text.pdf.PdfWriter;
public class FirstPDF {

    public static void main(String[] args) {
        // 创建文档对象，A4 纸大小
        Document document = new Document(PageSize.A4);
        try {
            // 输出为 E:\itext.pdf 文件
            PdfWriter writer = PdfWriter.getInstance(document,
                new FileOutputStream("E:\\itext.pdf "));
            // 打开文档
            document.open();
            // 在 pdf 文件中写入文字
            document.add(new Paragraph("Hello World, Hello iText"));
            // 关闭文档
            document.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

该程序运行后，将在 E 盘的根目录下生成一个 itext.pdf 文件，该 pdf 文件打开后，效果如图 40.2 所示。

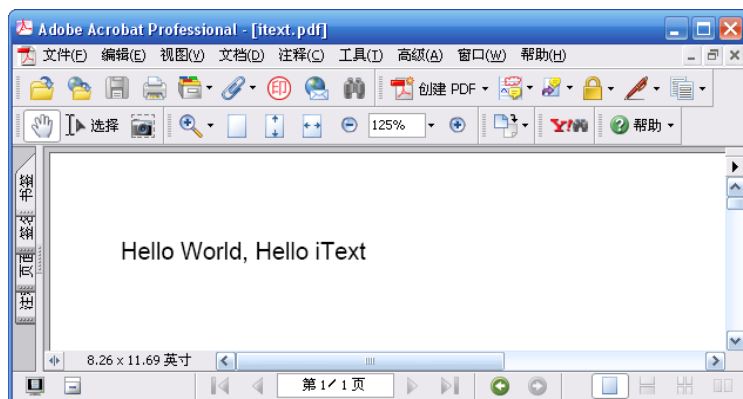


图 40.2 iText 生成的 pdf 文件

通过以上的程序，总结出使用 iText 生成一个 pdf 的步骤如下：

(1) 创建 Document 对象。

```
Document document = new Document();
```

其中，Document 有 3 个构造方法，如下：

- ❑ public Document();
- ❑ public Document(Rectangle pageSize);
- ❑ public Document(Rectangle pageSize,int marginLeft,int marginRight,int marginTop,int marginBottom)。

pageSize 是纸张类型的大小，通常可以使用 PageSize 中的常量来表示，例如 PageSize.A4 表示 A4 纸张。marginLeft、marginRight、marginTop 和 marginBottom 分别是正文距离页边的左、右、上、下的补白大小。

(2) 创建书写器 (Writer) 与 document 对象关联，通过书写器可以将文档写入磁盘中。

```
PdfWriter writer = PdfWriter.getInstance(document,new FileOutputStream("E:\\itext.pdf "));
```

(3) 打开文档。

```
document.open();
```

(4) 写入文档内容。

```
document.add(new Paragraph("Hello iText"));
```

写入的文档内容可以是多种类型，这里是带格式的文本 Paragraph，还可以是 Phrase、Paragraph、Table、Graphic 对象等。

(5) 关闭文档。

```
document.close();
```

通过以上 5 个步骤，就可以生成 pdf 文档了。

## 40.3 设置 pdf 文件属性

在打开 Document 对象写入内容之前，可以设置文档的属性，包括文档的标题、主题、作者、关键字、装订方式、创建者、生产者、创建日期等，方法如以下：

- ❑ public boolean addTitle(String title): 设置标题；
- ❑ public boolean addSubject(String subject): 设置主题；
- ❑ public boolean addKeywords(String keywords): 设置关键字；
- ❑ public boolean addAuthor(String author): 设置作者；
- ❑ public boolean addCreator(String creator): 设置创建者；
- ❑ public boolean addProducer(): 设置生产者；
- ❑ public boolean addCreationDate(): 设置创建日期；
- ❑ public boolean addHeader(String name, String content): 设置文件夹信息。

其中方法 addHeader() 对于 pdf 文档无效，addHeader() 仅对 html 文档有效，用于添加文

档的头信息。例如以下是设置了文档属性的代码：

```
// 创建文档对象，A4 纸大小
Document document = new Document(PageSize.A4);
try {
    // 输出为 E:\itext.pdf 文件
    PdfWriter writer = PdfWriter.getInstance(document,
        new FileOutputStream("E:\\itext.pdf "));
    // 设置文档作者
    document.addAuthor("Helloween");
    // 设置文档标题
    document.addTitle("This is itext pdf file");
    // 设置主题
    document.addSubject("First pdf");
    // 设置关键字
    document.addKeywords("iText");

    // 打开文档
    document.open();
    // 在 pdf 文件中写入文字
    document.add(new Paragraph("Hello World, Hello iText"));
    // 关闭文档
    document.close();
} catch (Exception e) {
    e.printStackTrace();
}
```

设置属性后，在生成的 pdf 文件上右击，在弹出的快捷菜单中选择“属性”选项，将弹出文档属性对话框，如图 40.3 所示，在这里可以查看程序中设置的文档属性。



图 40.3 查看 pdf 文档属性

## 40.4 插入文本

iText 中用文本块（Chunk）、短语（Phrase）和段落（Paragraph）处理文本：

- ❑ 文本块是处理文本的最小单位，由一串带格式（包括字体、颜色、大小）的字符串组成。
- ❑ 短语由一个或多个文本块组成，短语也可以设定字体，但对于其中已设定过字体的文本块无效。
- ❑ 段落由一个或多个文本块或短语组成，相当于 Word 文档中的段落概念，同样可以设定段落的字体大小、颜色等属性。另外也可以设定段落的首行缩进、对齐方式（左对齐、右对齐、居中对齐）。

这里最常用的是 Paragraph 对象，所以着重讲述该对象的使用方法。首先看下面一段程序，将一段文本设置了字体、大小和颜色等属性，代码如下：

```
document.open();
//在 pdf 文件中写入文字
document.add(new Paragraph("Hello iText"));
//设置字体的文字
document.add(new Paragraph("Hello World, Hello iText",FontFactory.getFont(
(FontFactory.COURIER, 12, Font.BOLDITALIC)))); //关闭文档
document.close();
```

运行程序后，生成的 pdf 显示的效果如图 40.4 所示，可以看到第二行文字设置了字体并增加了加粗倾斜的格式。

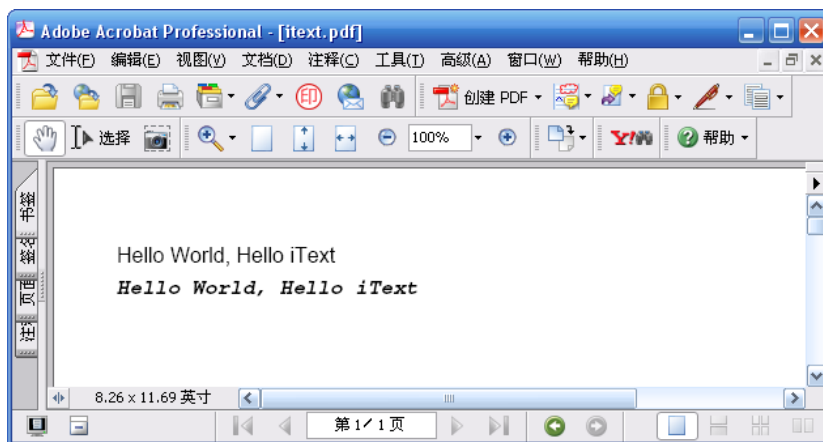


图 40.4 带格式的文本

除了字体外，还可以设置段落的缩进等属性，这些方法如下：

- ❑ `public void setAlignment(int alignment)`: 设置对齐方式，1 为居中对齐，2 为右对齐，3 为左对齐，默认为左对齐。
- ❑ `public void setFirstLineIndent(float firstLineIndent)`: 设置首行缩进。
- ❑ `public void setSpacingBefore(float spacing)`: 设置段前距。

❑ `public void setSpacingAfter(float spacing)`: 设置段后距。

## 40.5 文本的中文处理

默认的 iText 字体设置不支持中文字体, 如果在 pdf 中使用了中文字符, 需要下载亚洲语言字体包 `iTextAsian.jar`。通过下面的代码就可以在文档中显示中文了。

```
// 创建中文字体
BaseFont bfChinese = BaseFont.createFont("STSong-Light", "UniGB-UCS2-H",
BaseFont.NOT_EMBEDDED);
Font FontChinese = new Font(bfChinese, 12, Font.NORMAL);
// 设置为中文
Paragraph paragraph=new Paragraph("你好,这是中文", FontChinese);
document.add(paragraph);
```

程序运行后, 效果如图 40.5 所示。

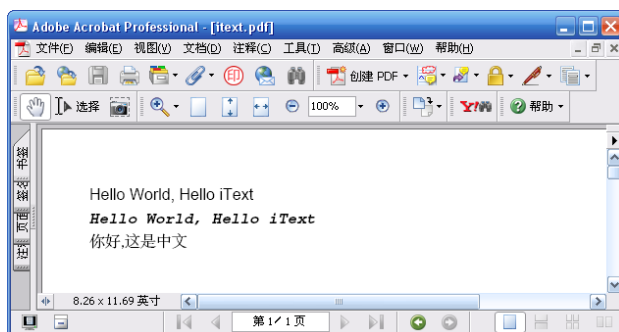


图 40.5 显示中文的 pdf 文件

## 40.6 插入表格

iText 中处理表格的类为 `com.lowagie.text.Table` 和 `com.lowagie.text.PDF.PDFPTable` 两个。对于比较简单的表格处理可以用 `Table` 类; 但是如果要处理复杂的表格, 就需要使用 `PDFPTable` 进行处理。这里就详细讲解如何使用 `Table`。

首先看以下一段小程序, 显示的为一个 2 行 3 列的表格, 代码如下:

```
// 创建 2 行 3 列的表
Table table = new Table(3, 2);
// 设置表格边框颜色
table.setBorderColor(new Color(220, 100, 100));
// 设置表格边距
table.setPadding(5);
// 设置表格间距
table.setSpacing(3);
// 设置表格线条宽度
table.setBorderWidth(3);
```



```
// 创建单元格对象
Cell cell = new Cell(" Header 1 ");
// 将单元格添加到表格中
table.addCell(cell);
cell = new Cell(" Header 2 ");
// 设置单元格占 2 列
cell.setColspan(2);
table.addCell(cell);
// 将普通文本添加到表格中
table.addCell(" Cell 1 ");
table.addCell(" Cell 2 ");
table.addCell(" Cell 3 ");
// 将表格添加到文档中
document.add(table);
```

代码运行后，生成的 pdf 表格的效果如图 40.6 所示。

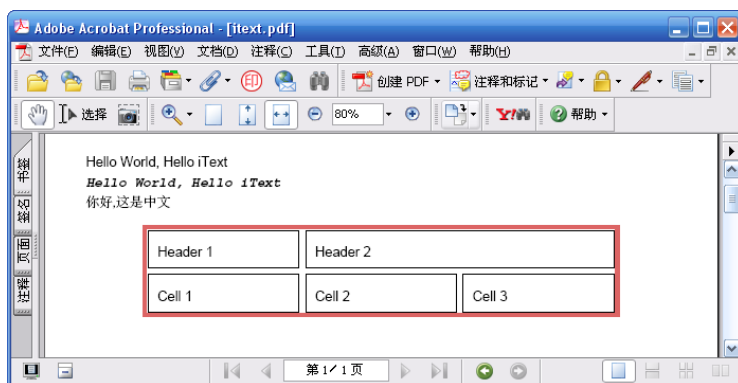


图 40.6 pdf 表格

在使用 `com.lowagie.text.Table` 类时需要注意以下几个问题：

(1) `Table` 的构造函数有 3 个：

- ☐ `Table (int columns);`
- ☐ `Table(int columns, int rows);`
- ☐ `Table(Properties attributes);`

参数 `columns`、`rows`、`attributes` 分别为表格的列数、行数、表格属性。创建表格时必须指定表格的列数，而对于行数可以不用指定，默认行数为 1。

(2) 往表格中添加单元格 (`cell`) 时，按自左向右、从上而下的次序添加。使用 `addCell()` 方法添加单元格。

(3) 对于复杂的表格，可以设置单元格的合并列 `cell.setColspan(2)` 和合并行 `cell.Rowspan(2)`。类似 `html` 中 `<table>` 的使用。

## 40.7 插入图像

iText 中处理表格的类为 `com.lowagie.text.Image`，目前 iText 支持的图像格式有 GIF、Jpeg、PNG、wmf 等格式。对于不同的图像格式，iText 用同样的构造函数自动识别图像格



式。以下代码为在 pdf 中生成图片的代码：

```
// 打开文档
document.open();
// 创建图片对象，参数为图片的文件名
Image bmp = Image.getInstance("E:\\ Java_logo.jpg");
// 缩小到原来的 25%
bmp.scalePercent(25f);
document.add(bmp);
// 关闭文档
document.close();
```

代码运行后，效果如图 40.7 所示。



图 40.7 pdf 图像

创建 Image 对象后，还可以设置一些常用的属性，这些属性有：

(1) 设置图像的位置：public void setAlignment(int alignment)，其中 alignment 可使用 Image 类中的常量。

- ☐ Image.RIGHT、Image.MIDDLE、Image.LEFT 分别指右对齐、居中、左对齐。
- ☐ Image.TEXTWRAP、Image.UNDERLYING 分别指文字绕图形显示、图形作为文字的背景显示。
- ☐ 这两种参数可以结合以达到预期的效果，如 setAlignment(Image.RIGHT|Image.TEXTWRAP)显示的效果为图像右对齐，文字围绕图像显示。

(2) 设置图像的大小，方法分别有：

- ☐ public void scaleAbsolute(int newWidth, int newHeight)：设置图片的宽和高。
- ☐ public void scalePercent(int percent)：设置缩放比例，如 scalePercent(50)表示显示的大小为原尺寸的 50%。
- ☐ public void scalePercent(int percentX, int percentY)：设置长和宽的缩放比例。

(3) 设置图像旋转：public void setRotation(double r)，其中 r 为弧度，如果旋转角度为 30 度，则参数  $r = \text{Math.PI} / 6$ 。

## 40.8 在 Servlet 中输出 pdf

在 Java Web 应用中,通常可以访问某个 Servlet 来查看 pdf 文件,通过 Servlet 读取 pdf 文件的步骤如下:

(1) 在 web.xml 中配置 Servlet,代码如下:

代码 40.2 web.xml

```
<servlet>
  <servlet-name>PDFServlet</servlet-name>
  <servlet-class>com.helloeenvsfei.itext.PDFServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>PDFServlet</servlet-name>
  <url-pattern>/pdf</url-pattern>
</servlet-mapping>
```

(2) 对应的 Servlet 代码如下:

代码 40.3 PDFServlet.java

```
package com.helloeenvsfei.itext;
import java.io.ByteArrayOutputStream;
import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.ServletOutputStream;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import com.lowagie.text.Document;
import com.lowagie.text.PageSize;
import com.lowagie.text.Paragraph;
import com.lowagie.text.pdf.PdfWriter;
public class PDFServlet extends HttpServlet {
    private static final long serialVersionUID = -1770041214313726403L;
    public void doGet(HttpServletRequest request, HttpServletResponse
        response)
        throws ServletException, IOException {
        this.doPost(request, response);
    }

    public void doPost(HttpServletRequest request, HttpServletResponse
        response)
        throws ServletException, IOException {

        // 创建文档对象, A4 纸大小
        Document document = new Document(PageSize.A4);
        ByteArrayOutputStream stream = new ByteArrayOutputStream();
        try {
            PdfWriter writer = PdfWriter.getInstance(document, stream);
            // 打开文档
            document.open();
            // 在 pdf 文件中写入文字
            document.add(new Paragraph("Hello World, Hello iText"));
            // 关闭文档
```

```
        document.close();
    } catch (Exception e) {
        e.printStackTrace();
    }
    // 设置响应文档类型为 pdf
    response.setContentType("application/pdf");
    // 设置响应数据大小
    response.setContentLength(stream.size());
    // 获得响应数据流
    ServletOutputStream out = response.getOutputStream();
    // 将 pdf 数据流写入响应数据流中
    stream.writeTo(out);
    out.flush();
    out.close();
}
}
```

通过以上代码，总结出访问 Servlet 读取 pdf 需要注意以下几点：

❑ 需要设置响应类型为 application/pdf，代码如下：

```
response.setContentType("application/pdf");
```

❑ 需要将生成的 pdf 数据流写入响应客户端数据流中。

## 40.9 本章小结

本章首先讲述了 iText 基础知识，包括 iText 的下载。接着又详细介绍了如何利用 iText 生成 pdf 文件等，包括在 pdf 文件中插入文本、插入表格、插入图像以及中文字符的处理等。

除了本章介绍的这些知识外，iText 还可以输出 XML 文档、复杂 rtf 文档等，由于这些类型的文件通常使用的不多，所以本章不做详细介绍，读者若感兴趣，可参阅其参考文档。