[空间](#)[博客](#)[好友](#)[相册](#)[留言](#)

用户操作

[\[留言\]](#) [\[发消息\]](#) [\[加为好友\]](#)**商学子** ID: lenolon[g](#)

共 89446 次访问, 排名 1786, 好友 199 人, 关注者 523 人。

努力做一个真实的我, 诚信的我, 有用的我! 要相信自己的实力, 一定能战胜自我!!!

商学子的文章

原创 169 篇

翻译 0 篇

转载 91 篇

评论 91 篇

订阅我的博客



lenolong的公告

文章分类

☐ .net☐ Ajax系列☐ Dreamweav

er

公告:

Rambook让人家CSDN提交作品的来登记哦!

POI导出EXCEL经典实现 [收藏](#)

在web开发中, 有一个经典的功能, 就是数据的导入导出。特别是数据的导出, 在生产管理或者财务系统中用的非常普遍, 因为这些系统经常要做一些报表打印的工作。而数据导出的格式一般是EXCEL或者PDF, 我这里就用两篇文章分别给大家介绍一下。(注意, 我们这里说的数据导出可不是数据库中的数据导出! 么误会啦^_^)

呵呵, 首先我们来导出EXCEL格式的文件吧。现在主流的操作Excel文件的开源工具有很多, 用得比较多的就是Apache的POI及JExcelAPI。这里我们用Apache POI! 我们先去Apache的大本营下载POI的jar包: <http://poi.apache.org/>, 我这里使用的是3.0.2版本。

将3个jar包导入到classpath下, 什么? 忘了怎么导包? 不会吧! 好, 我们来写一个导出Excel的实用类(所谓实用, 是指基本不用怎么修改就可以在实际项目中直接使用的!)。我一直强调做类也好, 做方法也好, 一定要通用性和灵活性强。下面这个类就算基本贯彻了我的这种思想。那么, 熟悉许老师风格的人应该知道, 这时候该要甩出一长串代码了。没错, 大伙请看:

```
package org.leno.export.util;
```

```
import java.util.Date;
```

```
public class Student {
```

```
    private long id;
```

```
    private String name;
```

```
    private int age;
```

```
    private boolean sex;
```

```
    private Date birthday;
```

```
    public Student() {
```

```
        super();
```

```
        // TODO Auto-generated constructor stub
```

```
    }
```

```
    public Student(long id, String name, int age, boolean sex, Date birthday) {
```

```
        super();
```

```
        this.id = id;
```

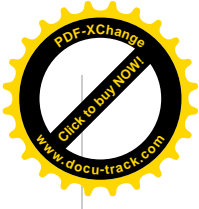
```
        this.name = name;
```

```
        this.age = age;
```

```
        this.sex = sex;
```

```
        this.birthday = birthday;
```

```
    }
```

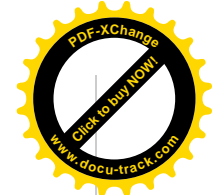


- Flex系列
- hibernate系
- 列
- javaEE
- java世界
- JS系列
- Linux/CVS/S
- VN/VSFTP...
- Spring系列
- Struts与Strut
- s2系列
- xml系列
- 工作流引擎b
- pm
- 面试经典
- 设计模式
- 数据库应用
- 搜索引擎
- 心情随笔
- 中间件/集群

存档

2010年04月(1)
2009年12月(2)
2009年11月(1)
2009年10月(2)
2009年03月(35)
2009年02月(7)
2009年01月(6)
2008年12月(206)

```
public long getId() {  
    return id;  
}  
  
public void setId(long id) {  
    this.id = id;  
}  
  
public String getName() {  
    return name;  
}  
  
public void setName(String name) {  
    this.name = name;  
}  
  
public int getAge() {  
    return age;  
}  
  
public void setAge(int age) {  
    this.age = age;  
}  
  
public boolean getSex() {  
    return sex;  
}  
  
public void setSex(boolean sex) {  
    this.sex = sex;  
}  
  
public Date getBirthday() {  
    return birthday;  
}  
  
public void setBirthday(Date birthday) {  
    this.birthday = birthday;  
}  
  
}  
  
package org.leno.export.util;  
  
public class Book {  
    private int bookId;  
    private String name;  
    private String author;
```



```
private float price;
private String isbn;
private String pubName;
private byte[] preface;

public Book() {
    super();
}

public Book(int bookId, String name, String author, float
price,
        String isbn, String pubName, byte[] preface) {
    super();
    this.bookId = bookId;
    this.name = name;
    this.author = author;
    this.price = price;
    this.isbn = isbn;
    this.pubName = pubName;
    this.preface = preface;
}

public int getBookId() {
    return bookId;
}

public void setBookId(int bookId) {
    this.bookId = bookId;
}

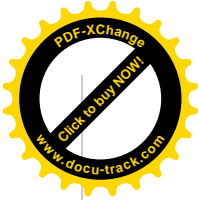
public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name;
}

public String getAuthor() {
    return author;
}

public void setAuthor(String author) {
    this.author = author;
}

public float getPrice() {
    return price;
}
```



```
public void setPrice(float price) {
    this.price = price;
}

public String getIsbn() {
    return isbn;
}

public void setIsbn(String isbn) {
    this.isbn = isbn;
}

public String getPubName() {
    return pubName;
}

public void setPubName(String pubName) {
    this.pubName = pubName;
}

public byte[] getPreface() {
    return preface;
}

public void setPreface(byte[] preface) {
    this.preface = preface;
}
}
```

上面这两个类一目了然，就是两个简单的javabeen风格的类。再看下面真正的重点类：

```
package org.leno.export.util;
```

```
import java.io.*;
import java.lang.reflect.*;
import java.util.*;
import java.util.regex.Matcher;
import java.util.regex.Pattern;
import java.text.SimpleDateFormat;
import javax.swing.JOptionPane;
import org.apache.poi.hssf.usermodel.*;
import org.apache.poi.hssf.util.HSSFColor;
```

```
/**
```

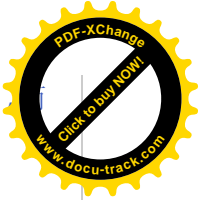
```
 * 利用开源组件POI3.0.2动态导出EXCEL文档
```

```
 * 转载时请保留以下信息，注明出处！
```

```
 * @author leno
```

```
 * @version v1.0
```

```
 * @param <T> 应用泛型，代表任意一个符合javabeen风格的类
```



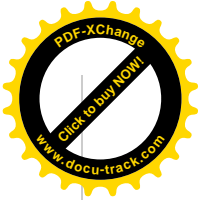
```
* 注意这里为了简单起见, boolean型的属性xxx的get器方式为getXxx()
不是isXxx()
* byte[]表jpg格式的图片数据
*/
public class ExportExcel<T> {

    public void exportExcel(Collection<T> dataset, OutputStream out) {
        exportExcel("测试POI导出EXCEL文档", null, dataset, out,
"yyyy-MM-dd");
    }

    public void exportExcel(String[] headers, Collection<T> dataset,
        OutputStream out) {
        exportExcel("测试POI导出EXCEL文档", headers, dataset, out,
"yyyy-MM-dd");
    }

    public void exportExcel(String[] headers, Collection<T> dataset,
        OutputStream out, String pattern) {
        exportExcel("测试POI导出EXCEL文档", headers, dataset, out, pattern);
    }

    /**
     * 这是一个通用的方法, 利用了JAVA的反射机制, 可以将放置在JAVA集合中
     * 并且符号一定条件的数据以EXCEL 的形式输出到指定IO设备上
     *
     * @param title
     *         表格标题名
     * @param headers
     *         表格属性列名数组
     * @param dataset
     *         需要显示的数据集合, 集合中一定要放置符合javabeen风格
     *         的类的对象。此方法支持的
     *         javabeen属性的数据类型有基本数据类型及String, Date, byte[] (图片数据)
     * @param out
     *         与输出设备关联的流对象, 可以将EXCEL文档导出到本地文件或者网络中
     * @param pattern
     *         如果有时间数据, 设定输出格式。默认为"yyy-MM-dd"
     */
    @SuppressWarnings("unchecked")
    public void exportExcel(String title, String[] headers,
        Collection<T> dataset, OutputStream out, String pattern) {
```



POI导出EXCEL经典实现 - lenolong的...

```
// 声明一个工作簿
HSSFWorkbook workbook = new HSSFWorkbook();

// 生成一个表格
HSSFSheet sheet = workbook.createSheet(title);
// 设置表格默认列宽度为15个字节
sheet.setDefaultColumnWidth((short) 15);
// 生成一个样式
HSSFCellStyle style = workbook.createCellStyle();
// 设置这些样式
style.setFillForegroundColor(HSSFColor.SKY_BLUE.index);

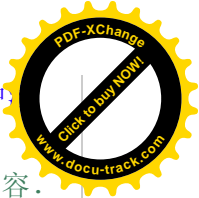
;

style.setFillPattern(HSSFCellStyle.SOLID_FOREGROUND);
style.setBorderBottom(HSSFCellStyle.BORDER_THIN);
style.setBorderLeft(HSSFCellStyle.BORDER_THIN);
style.setBorderRight(HSSFCellStyle.BORDER_THIN);
style.setBorderTop(HSSFCellStyle.BORDER_THIN);
style.setAlignment(HSSFCellStyle.ALIGN_CENTER);
// 生成一个字体
HSSFFont font = workbook.createFont();
font.setColor(HSSFColor.VIOLET.index);
font.setFontHeightInPoints((short) 12);
font.setBoldweight(HSSFFont.BOLDWEIGHT_BOLD);
// 把字体应用到当前的样式
style.setFont(font);
// 生成并设置另一个样式
HSSFCellStyle style2 = workbook.createCellStyle();
style2.setFillForegroundColor(HSSFColor.LIGHT_YELLOW.index);
style2.setFillPattern(HSSFCellStyle.SOLID_FOREGROUND);
style2.setBorderBottom(HSSFCellStyle.BORDER_THIN);
style2.setBorderLeft(HSSFCellStyle.BORDER_THIN);
style2.setBorderRight(HSSFCellStyle.BORDER_THIN);
style2.setBorderTop(HSSFCellStyle.BORDER_THIN);
style2.setAlignment(HSSFCellStyle.ALIGN_CENTER);
style2.setVerticalAlignment(HSSFCellStyle.VERTICAL_CENTER);

// 生成另一个字体
HSSFFont font2 = workbook.createFont();
font2.setBoldweight(HSSFFont.BOLDWEIGHT_NORMAL);
// 把字体应用到当前的样式
style2.setFont(font2);

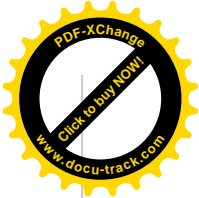
// 声明一个画图的顶级管理器
HSSFPatriarch patriarch = sheet.createDrawingPatriarch();

// 定义注释的大小和位置, 详见文档
HSSFComment comment = patriarch.createComment(new HSSFClientAnchor(0, 0, 0, 0, (short) 4, 2, (short) 6, 5));
// 设置注释内容
```



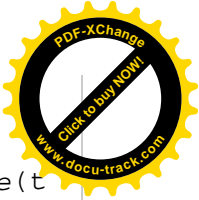
POI导出EXCEL经典实现 - lenolong的...

```
comment.setString(new HSSFRichTextString("可以在POI中  
加注释!"));  
// 设置注释作者, 当鼠标移动到单元格上是可以在状态栏中看到该内容.  
comment.setAuthor("leno");  
  
//产生表格标题行  
HSSFRow row = sheet.createRow(0);  
for (short i = 0; i < headers.length; i++) {  
    HSSFCell cell = row.createCell(i);  
    cell.setCellStyle(style);  
    HSSFRichTextString text = new HSSFRichTextString(headers[i]);  
    cell.setCellValue(text);  
}  
  
//遍历集合数据, 产生数据行  
Iterator<T> it = dataset.iterator();  
int index = 0;  
while (it.hasNext()) {  
    index++;  
    row = sheet.createRow(index);  
    T t = (T) it.next();  
    //利用反射, 根据javabean属性的先后顺序, 动态调用getXxx()方法得到属性值  
    Field[] fields = t.getClass().getDeclaredFields();  
    for (short i = 0; i < fields.length; i++) {  
        HSSFCell cell = row.createCell(i);  
        cell.setCellStyle(style2);  
        Field field = fields[i];  
        String fieldName = field.getName();  
        String getMethodName = "get"  
            + fieldName.substring(0, 1).toUpperCase()  
            + fieldName.substring(1);  
        try {  
            Class tCls = t.getClass();  
            Method getMethod = tCls.getMethod(getMethodName,  
ame, new Class[] {});  
            Object value = getMethod.invoke(t, new Object[] {});  
            //判断值的类型后进行强制类型转换  
            String textValue = null;  
            if (value instanceof Integer) {  
                int intValue = (Integer) value;  
                cell.setCellValue(intValue);  
            } else if (value instanceof Float) {  
                float fValue = (Float) value;  
                textValue = new HSSFRichTextString(String.valueOf(fValue));  
            }  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
    }  
}
```



POI导出EXCEL经典实现 - lenolong的...

```
//
//      cell.setCellValue(textValue);
//
//      } else if (value instanceof Double) {
//
//          double dValue = (Double) value;
//
//          textValue = new HSSFRichTextString(
//              String.valueOf(dValue));
//
//          cell.setCellValue(textValue);
//
//      } else if (value instanceof Long) {
//
//          long longValue = (Long) value;
//
//          cell.setCellValue(longValue);
//
//      }
//
//      if (value instanceof Boolean) {
//
//          boolean bValue = (Boolean) value;
//
//          textValue = "男";
//
//          if (!bValue) {
//
//              textValue = "女";
//
//          }
//
//      } else if (value instanceof Date) {
//
//          Date date = (Date) value;
//
//          SimpleDateFormat sdf = new SimpleDateFormat
at(pattern);
//
//          textValue = sdf.format(date);
//
//      } else if (value instanceof byte[]) {
//
//          // 有图片时, 设置行高为60px;
//
//          row.setHeightInPoints(60);
//
//          // 设置图片所在列宽度为80px,注意这里单位的一个换
算
//
//          sheet.setColumnWidth(i, (short) (35.7 * 8
0));
//
//          // sheet.autoSizeColumn(i);
//
//          byte[] bsValue = (byte[]) value;
//
//          HSSFClientAnchor anchor = new HSSFClientA
nchor(0, 0,
//
//              1023, 255, (short) 6, index, (short
) 6, index);
//
//          anchor.setAnchorType(2);
//
//          patriarch.createPicture(anchor, workbook.
addPicture(
//
//              bsValue, HSSFWorkbook.PICTURE_TYPE_
JPEG));
//
//      } else{
//
//          //其它数据类型都当作字符串简单处理
//
//          textValue = value.toString();
//
//      }
//
//      //如果不是图片数据, 就利用正则表达式判断textValue是
否全部由数字组成
//
//      if(textValue!=null){
//
//          Pattern p = Pattern.compile("^\\d+(\\.\\d
+)?$");
//
//          Matcher matcher = p.matcher(textValue);
```

```
        if(matcher.matches()){
            //是数字当作double处理
            cell.setCellValue(Double.parseDouble(t
extValue));
        }else{
            HSSFRichTextString richString = new HS
SFRichTextString(textValue);
            HSSFFont font3 = workbook.createFont()
;

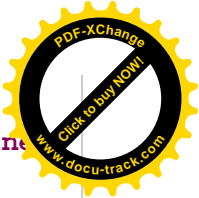
            font3.setColor(HSSFColor.BLUE.index);
            richString.applyFont(font3);
            cell.setCellValue(richString);
        }
    }
} catch (SecurityException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
} catch (NoSuchMethodException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
} catch (IllegalArgumentException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
} catch (IllegalAccessException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
} catch (InvocationTargetException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
} finally {
    //清理资源
}

}

try {
    workbook.write(out);
} catch (IOException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}

}

public static void main(String[] args) {
    // 测试学生
    ExportExcel<Student> ex = new ExportExcel<Student>();
    String[] headers = { "学号", "姓名", "年龄", "性别", "出生
日期" };
};
```

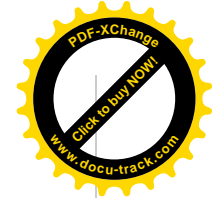


POI导出EXCEL经典实现 - lenolong的...

```
List<Student> dataset = new ArrayList<Student>();
dataset.add(new Student(10000001, "张三", 20, true, new
Date()));
dataset.add(new Student(20000002, "李四", 24, false, ne
w Date()));
dataset.add(new Student(30000003, "王五", 22, true, new
Date()));
// 测试图书
ExportExcel<Book> ex2 = new ExportExcel<Book>();
String[] headers2 = { "图书编号", "图书名称", "图书作者",
"图书价格", "图书ISBN",
"图书出版社", "封面图片" };
List<Book> dataset2 = new ArrayList<Book>();
try {
    BufferedInputStream bis = new BufferedInputStream(
        new FileInputStream("book.jpg"));
    byte[] buf = new byte[bis.available()];
    while ((bis.read(buf)) != -1) {
        //
    }
    dataset2.add(new Book(1, "jsp", "leno", 300.33f, "1
234567",
        "清华大学出版社", buf));
    dataset2.add(new Book(2, "java编程思想", "bruce", 300
.33f, "1234567",
        "阳光出版社", buf));
    dataset2.add(new Book(3, "DOM艺术", "lenotang", 300.
33f, "1234567",
        "清华大学出版社", buf));
    dataset2.add(new Book(4, "c++经典", "leno", 400.33f,
"1234567",
        "清华大学出版社", buf));
    dataset2.add(new Book(5, "c#入门", "leno", 300.33f,
"1234567",
        "汤春秀出版社", buf));

    OutputStream out = new FileOutputStream("E:\\a.xls"
);
    OutputStream out2 = new FileOutputStream("E:\\b.xls
");

    ex.exportExcel(headers, dataset, out);
    ex2.exportExcel(headers2, dataset2, out2);
    out.close();
    JOptionPane.showMessageDialog(null, "导出成功!");
    System.out.println("excel导出成功!");
} catch (FileNotFoundException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}
```

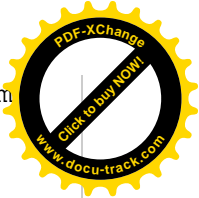


POI导出EXCEL经典实现 - lenolong的...

```
} catch (IOException e) {  
    // TODO Auto-generated catch block  
    e.printStackTrace();  
}  
}  
}
```

不行，头有点晕^_^。呵呵，又是泛型，又是反射，又是正则表达式，又是重载，还有多参数列表和POI API。一下子蹦出来，实在让人吃不消。不管了，顶住看效果先。在本地运行后，我们发现在E:\下生成了两份excel文件：学生记录和图书记录，并且中文，数字，颜色，日期，图片等等一旦正常。恩，太棒了。有人看到这里开始苦脸了：喂，我怎么一运行就报错啊！呵呵，看看什么错吧！哦，找不到文件，也就是说你没有book.jpg嘛。好，**拷贝一张小巧的图书图片命名为book.jpg放置到当前工程下吧**。注意，您千万别把张桌面大小的图片丢进去了^_^！看到效果了吧。现在我们来简单梳理一下代码，实际上上面就做了一个导出excel的方法和一个本地测试main()方法。并且代码的结构也很清晰，只是涉及的知识点稍微多一点。大家细心看看注释，结合要完成的功能，应该没有太大问题的。好啦，吃杯茶，擦把汗，总算把这个类消化掉，你又进步了。咦，你不是说是在WEB环境下导出的吗？别急，因为导出就是一个下载的过程。我们只需要在服务器端写一个Jsp或者Servlet组件完成输出excel到浏览器客户端的工作就好了。我们以Servlet为例，还是看代码吧：

```
package org.leno.export.util;  
  
import java.io.*;  
import java.util.ArrayList;  
import java.util.List;  
  
import javax.servlet.ServletException;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
  
/**  
 * @author leno  
 * 使用servlet导出动态生成的excel文件，数据可以来源于数据库  
 * 这样，浏览器客户端就可以访问该servlet得到一份用java代码动态生成的excel文件  
 */  
public class Export extends javax.servlet.http.HttpServlet {  
    static final long serialVersionUID = 1L;  
  
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
        File file = new File(getServletContext().getRealPath("WEB-INF/book.jpg"));  
        response.setContentType("octets/stream");  
        response.addHeader("Content-Disposition", "attachment; filename=test.xls");  
        //测试图书  
        ExportExcel<Book> ex = new ExportExcel<Book>();  
        String[] headers = { "图书编号", "图书名称", "图书作者", "图书价格", "图书ISBN",  
            "图书出版社", "封面图片" };  
        List<Book> dataset = new ArrayList<Book>();  
        try {
```



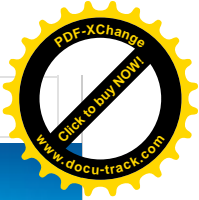
POI导出EXCEL经典实现 - lenolong的...

```
BufferedInputStream bis = new BufferedInputStream(
    new FileInputStream(file));
byte[] buf = new byte[bis.available()];
while ((bis.read(buf)) != -1) {
    //将图片数据存放到缓冲数组中
}
dataset.add(new Book(1, "jsp", "leno", 300.33f, "12
34567",
    "清华大学出版社", buf));
dataset.add(new Book(2, "java编程思想", "bruc1", 300.
33f, "1234567",
    "阳光出版社", buf));
dataset.add(new Book(3, "DOM艺术", "lenotang", 300.3
3f, "1234567",
    "清华大学出版社", buf));
dataset.add(new Book(4, "c++经典", "leno", 400.33f,
"1234567",
    "清华大学出版社", buf));
dataset.add(new Book(5, "c#入门", "leno", 300.33f, "
1234567",
    "汤春秀出版社", buf));
OutputStream out = response.getOutputStream();
ex.exportExcel(headers, dataset, out);
out.close();
System.out.println("excel导出成功!");
} catch (FileNotFoundException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
} catch (IOException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}
}

protected void doPost(HttpServletRequest request, HttpSer
vletResponse response) throws ServletException, IOException
{
    doGet(request, response);
}
}
```

写完之后,如果您不是用eclipse工具生成的Servlet,千万别忘了在web.xml上注册这个Servlet。而且同样的,拷贝一张小巧的图书图片命名为book.jpg放置到当前WEB根目录的/WEB-INF/下。部署好web工程,用浏览器访问Servlet看下效果吧!是不是下载成功了。呵呵,您可以将下载到本地的excel报表用打印机打印出来,这样您就大功告成了。完事了我们就思考:我们发现,我们做的方法,不管是本地调用,还是在WEB服务器端用Servlet调用;不管是输出学生列表,还是图书列表信息,代码都几乎一样,而且这些数据我们很容易结合后台的DAO操作数据库动态获取。恩,类和方法的通用性和灵活性开始有点感觉了。好啦,祝您学习愉快!

发表于 @ 2009年03月04日 21:22:00 | [评论\(14\)](#) | [举报](#) | [收藏](#)



旧一篇:25个优秀的Ajax技术和实例 | 新一篇:Struts 2导出EXCEL



相关文章

[查看最新精华文章](#) [请访问博客首页](#)

[翅膀教会我在快乐中寻找幸福](#)

[Windows 经典命令行 - 管理员手册](#)

[真有点道理\(推荐版\)](#)

[让DataGrid拥有单击回传事件并带回指定字段一份还可以的面试题](#)

[解析C++中的内部连接与外部连接](#)

[广州seo:怎么处理被百度K的站](#)

[深入.NET DataTable \(转载\)](#)

[cfq8585](#) 发表于Thu Nov 26 2009 10:53:32 GMT+0800 (China Standard Time) [举报](#) [回复](#)



非常感谢!很不错, 希望写篇POI操作Word的经典实现

[匿名用户](#) 发表于Mon Jun 07 2010 19:44:01 GMT+0800 (China Standard Time) [举报](#) [回复](#)



顶!

[匿名用户](#) 发表于Mon Jun 07 2010 19:44:49 GMT+0800 (China Standard Time) [举报](#) [回复](#)



下下来看能不能运行ok

[songminghong](#) 发表于Mon Jun 21 2010 16:31:23 GMT+0800 (China Standard Time) [举报](#) [回复](#)



我想弱弱的问一声该把哪些jar文件导入?

[songminghong](#) 发表于Thu Jun 24 2010 14:25:26 GMT+0800 (China Standard Time) [举报](#) [回复](#)



不错

[songminghong](#) 发表于Thu Jun 24 2010 14:25:59 GMT+0800 (China Standard Time) [举报](#) [回复](#)



我想到处一个有多个工作薄的exel, 怎样做?

[wmzjutbaobao2](#) 发表于Mon Aug 02 2010 00:38:36 GMT+0800 (China Standard Time) [举报](#) [回复](#)



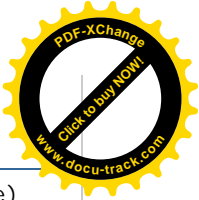
非常感谢你

[kobeehui](#) 发表于Thu Sep 23 2010 11:27:51 GMT+0800 (China Standard Time) [举报](#) [回复](#)



代码段的切分显示出设计时思路不清晰, 没有用面向对象的思想。利用反射读取bean内属性值那一段有点啰嗦, 而且你没有考虑到如果bean中的属性是private的, 你怎么处理。

[znihaonihao](#) 发表于Mon Sep 27 2010 22:10:05 GMT+0800 (China Standard Time) [举报](#) [回复](#)



真的 很不错写的 对与现在的我来说 很需要



[nysnthpjel](#) 发表于Mon Sep 27 2010 22:44:00 GMT+0800 (China Standard Time)

[举报](#) [回复](#)



[zeng7960983](#) 发表于Tue Sep 28 2010 10:50:08 GMT+0800 (China Standard Time)

[举报](#) [回复](#)



[f891379133](#) 发表于Thu Oct 07 2010 17:16:05 GMT+0800 (China Standard Time)

[举报](#) [回复](#)



这不错啊.典型的过程式编程. 我刚刚实现的.客户端的所有Ext grid一键导出excel. window.suform.document.write("<form name='formExcel' action='"+root+"/yunhao/report.do' method='post' target='suform' style='display:none'>"); window.suform.document.write("<input type='text' name='action' value='exportToExcel'>"); window.suform.document.write("<input type='text' name='jsonStore' value='"+jsonStore+"'>"); window.suform.document.write("<input type='text' name='context' value='"+context+"'>"); window.suform.document.write("</form>"); window.suform.document.formExcel.submit(); contexnt 就是从grid中的数据每个页面加个这个.就可以导了

[shun_zi](#) 发表于Thu Nov 11 2010 15:51:53 GMT+0800 (China Standard Time) [举报](#)

[回复](#)



对基本的操作有很大帮助

[davidbjh](#) 发表于Tue Jan 04 2011 10:41:00 GMT+0800 (China Standard Time) [举报](#)

[回复](#)



强烈要求，发送完整的信息。book.jpg是什么样子的，也一块发下呗

发表评论

表情:



评论内容:

【UC优视】UC浏览器，聘技术经理、高级工程师、产品经理

【DeNA China】10K~30K月薪诚聘Android开发工程师

【成都奥毕】诚聘软件开发工程师，高薪+奖金+优厚福利邀您共同发展

【Red/SAFI】北京公司诚聘开发人员.net iPhone等

用户名: 匿名用户

[登录](#) [注册](#)

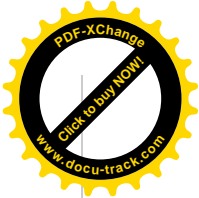
【Google背景公司】年薪10万到30万急聘linux、产品经理、java、UI设计、有期权

【上海博大】热聘JAVA网站客户端研发人员

【中国领先手机游戏索乐公司】诚聘英才

【上海杰图】诚聘中高级C++人才，待遇优、环境好、邀你共创

【北京东方国信】年薪4到25万



招聘JAVA程序员 / 知识 中国

【诺姆四达EHR】高薪诚聘软件
系统工程师、JAVA工程师、数据

【E人E本】热招各类开发人才（
Android、java、PHP),更多职

【中国知网】高薪诚聘精英技术
人才

[公司简介](#) | [招贤纳士](#) | [广告服务](#) | [银行汇款帐号](#) | [联系方式](#) | [版权声明](#) | [法律顾问](#) | [问题报告](#)

北京创新乐知信息技术有限公司 版权所有, 京 ICP 证 070598 号

世纪乐知(北京)网络技术有限公司 提供技术支持

江苏乐知网络技术有限公司 提供商务支持

 Email: webmaster@csdn.net

Copyright © 1999-2010, CSDN.NET, All Rights Reserved

