简介：

Apache POI是Apache软件基金会的开放源码函式库，POI提供API给Java程序对Microsoft Office格式档案读和写的功能。

Hssf：通过该可以对文档进行读写修改EXsl。HSSF 为读取操作提供了两类API：

usermodel和eventusermodel，即“用户模型”和“事件-用户模型”。

步骤：

## 1得到Excel常用对象

[c-sharp] [view plaincopyprint?](http://blog.csdn.net/xjun15/article/details/5805429)

1. POIFSFileSystem fs=newPOIFSFileSystem(new FileInputStream("d:/test.xls"));
2. //得到Excel工作簿对象
3. HSSFWorkbook wb = new HSSFWorkbook(fs);
4. //得到Excel工作表对象
5. HSSFSheet sheet = wb.getSheetAt(0);
6. //得到Excel工作表的行
7. HSSFRow row = sheet.getRow(i);
8. //得到Excel工作表指定行的单元格
9. HSSFCell cell = row.getCell((short) j);
10. cellStyle = cell.getCellStyle();//得到单元格样式

## 2、建立****Excel常用对象****

**[c-sharp]** [**view plaincopyprint?**](http://blog.csdn.net/xjun15/article/details/5805429)

1. HSSFWorkbook wb = new HSSFWorkbook();//创建Excel工作簿对象
2. HSSFSheet sheet = wb.createSheet("new sheet");//创建Excel工作表对象
3. HSSFRow row = sheet.createRow((short)0); //创建Excel工作表的行
4. cellStyle = wb.createCellStyle();//创建单元格样式
5. row.createCell((short)0).setCellStyle(cellStyle); //创建Excel工作表指定行的单元格
6. row.createCell((short)0).setCellValue(1); //设置Excel工作表的值

## 3、设置sheet名称和单元格内容

**[c-sharp]** [**view plaincopyprint?**](http://blog.csdn.net/xjun15/article/details/5805429)

1. wb.setSheetName(1, "第一张工作表",HSSFCell.ENCODING\_UTF\_16);
2. cell.setEncoding((short) 1);
3. cell.setCellValue("单元格内容");

## 4、取得****sheet的数目****

**[c-sharp]** [**view plaincopyprint?**](http://blog.csdn.net/xjun15/article/details/5805429)

1. wb.getNumberOfSheets()

## ****5、  根据index取得sheet对象****

**[c-sharp]** [**view plaincopyprint?**](http://blog.csdn.net/xjun15/article/details/5805429)

1. HSSFSheet sheet = wb.getSheetAt(0);

## ****6、取得有效的行数****

**[c-sharp]** [**view plaincopyprint?**](http://blog.csdn.net/xjun15/article/details/5805429)

1. int rowcount = sheet.getLastRowNum();

## 7、取得一行的有效单元格个数

**[c-sharp]** [**view plaincopyprint?**](http://blog.csdn.net/xjun15/article/details/5805429)

1. row.getLastCellNum();

## **8、单元格值类型读写**

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/xjun15/article/details/5805429)

1. cell.setCellType(HSSFCell.CELL\_TYPE\_STRING); //设置单元格为STRING类型
2. cell.getNumericCellValue();//读取为数值类型的单元格内容

## **9、设置列宽、行高**

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/xjun15/article/details/5805429)

1. sheet.setColumnWidth((short)column,(short)width);
2. row.setHeight((short)height);

## **10、添加区域，合并单元格**

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/xjun15/article/details/5805429)

1. Region region = new Region((short)rowFrom,(short)columnFrom,(short)rowTo
2. ,(short)columnTo);//合并从第rowFrom行columnFrom列
3. sheet.addMergedRegion(region);// 到rowTo行columnTo的区域
4. //得到所有区域
5. sheet.getNumMergedRegions()

## **11、保存Excel文件**

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/xjun15/article/details/5805429)

1. FileOutputStream fileOut = new FileOutputStream(path);
2. wb.write(fileOut);

## **12、根据单元格不同属性返回字符串数值**

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/xjun15/article/details/5805429)

1. public String getCellStringValue(HSSFCell cell) {
2. String cellValue = "";
3. switch (cell.getCellType()) {
4. case HSSFCell.CELL\_TYPE\_STRING://字符串类型
5. cellValue = cell.getStringCellValue();
6. if(cellValue.trim().equals("")||cellValue.trim().length()<=0)
7. cellValue=" ";
8. break;
9. case HSSFCell.CELL\_TYPE\_NUMERIC: //数值类型
10. cellValue = String.valueOf(cell.getNumericCellValue());
11. break;
12. case HSSFCell.CELL\_TYPE\_FORMULA: //公式
13. cell.setCellType(HSSFCell.CELL\_TYPE\_NUMERIC);
14. cellValue = String.valueOf(cell.getNumericCellValue());
15. break;
16. case HSSFCell.CELL\_TYPE\_BLANK:
17. cellValue=" ";
18. break;
19. case HSSFCell.CELL\_TYPE\_BOOLEAN:
20. break;
21. case HSSFCell.CELL\_TYPE\_ERROR:
22. break;
23. default:
24. break;
25. }
26. return cellValue;
27. }

## **13、常用单元格边框格式**

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/xjun15/article/details/5805429)

1. HSSFCellStyle style = wb.createCellStyle();
2. style.setBorderBottom(HSSFCellStyle.BORDER\_DOTTED);//下边框
3. style.setBorderLeft(HSSFCellStyle.BORDER\_DOTTED);//左边框
4. style.setBorderRight(HSSFCellStyle.BORDER\_THIN);//右边框
5. style.setBorderTop(HSSFCellStyle.BORDER\_THIN);//上边框

## **14、设置字体和内容位置**

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/xjun15/article/details/5805429)

1. HSSFFont f  = wb.createFont();
2. f.setFontHeightInPoints((short) 11);//字号
3. f.setBoldweight(HSSFFont.BOLDWEIGHT\_NORMAL);//加粗
4. style.setFont(f);
5. style.setAlignment(HSSFCellStyle.ALIGN\_CENTER);//左右居中
6. style.setVerticalAlignment(HSSFCellStyle.VERTICAL\_CENTER);//上下居中
7. style.setRotation(short rotation);//单元格内容的旋转的角度
8. HSSFDataFormat df = wb.createDataFormat();
9. style1.setDataFormat(df.getFormat("0.00%"));//设置单元格数据格式
10. cell.setCellFormula(string);//给单元格设公式
11. style.setRotation(short rotation);//单元格内容的旋转的角度

## **15、插入图片**

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/xjun15/article/details/5805429)

1. //先把读进来的图片放到一个ByteArrayOutputStream中，以便产生ByteArray
2. ByteArrayOutputStream byteArrayOut = new ByteArrayOutputStream();
3. BufferedImage bufferImg = ImageIO.read(new File("ok.jpg"));
4. ImageIO.write(bufferImg,"jpg",byteArrayOut);
5. //读进一个excel模版
6. FileInputStream fos = new FileInputStream(filePathName+"/stencil.xlt");
7. fs = new POIFSFileSystem(fos);
8. //创建一个工作薄
9. HSSFWorkbook wb = new HSSFWorkbook(fs);
10. HSSFSheet sheet = wb.getSheetAt(0);
11. HSSFPatriarch patriarch = sheet.createDrawingPatriarch();
12. HSSFClientAnchor anchor = new HSSFClientAnchor(0,0,1023,255,(short) 0,0,(short)10,10);
13. patriarch.createPicture(anchor , wb.addPicture(byteArrayOut.toByteArray(),HSSFWorkbook.PICTURE\_TYPE\_JPEG));

## **16、调整工作表位置**

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/xjun15/article/details/5805429)

1. HSSFWorkbook wb = new HSSFWorkbook();
2. HSSFSheet sheet = wb.createSheet("format sheet");
3. HSSFPrintSetup ps = sheet.getPrintSetup();
4. sheet.setAutobreaks(true);
5. ps.setFitHeight((short)1);
6. ps.setFitWidth((short)1);

## **17、设置打印区域**

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/xjun15/article/details/5805429)

1. HSSFSheet sheet = wb.createSheet("Sheet1");
2. wb.setPrintArea(0, "$A$1:$C$2");

## **18、标注脚注**

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/xjun15/article/details/5805429)

1. HSSFSheet sheet = wb.createSheet("format sheet");
2. HSSFFooter footer = sheet.getFooter()
3. footer.setRight( "Page " + HSSFFooter.page() + " of " + HSSFFooter.numPages() );

## **19、在工作单中清空行数据，调整行位置**

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/xjun15/article/details/5805429)

1. HSSFWorkbook wb = new HSSFWorkbook();
2. HSSFSheet sheet = wb.createSheet("row sheet");
3. // Create various cells and rows for spreadsheet.
4. // Shift rows 6 - 11 on the spreadsheet to the top (rows 0 - 5)
5. sheet.shiftRows(5, 10, -5);

## **20、选中指定的工作表**

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/xjun15/article/details/5805429)

1. HSSFSheet sheet = wb.createSheet("row sheet");
2. heet.setSelected(true);

## **21、工作表的放大缩小**

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/xjun15/article/details/5805429)

1. HSSFSheet sheet1 = wb.createSheet("new sheet");
2. sheet1.setZoom(1,2);   // 50 percent magnification

## **22、头注和脚注**

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/xjun15/article/details/5805429)

1. HSSFSheet sheet = wb.createSheet("new sheet");
2. HSSFHeader header = sheet.getHeader();
3. header.setCenter("Center Header");
4. header.setLeft("Left Header");
5. header.setRight(HSSFHeader.font("Stencil-Normal", "Italic") +
6. HSSFHeader.fontSize((short) 16) + "Right w/ Stencil-Normal Italic font and size 16");

## **23、自定义颜色**

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/xjun15/article/details/5805429)

1. HSSFCellStyle style = wb.createCellStyle();
2. style.setFillForegroundColor(HSSFColor.LIME.index);
3. style.setFillPattern(HSSFCellStyle.SOLID\_FOREGROUND);
4. HSSFFont font = wb.createFont();
5. font.setColor(HSSFColor.RED.index);
6. style.setFont(font);
7. cell.setCellStyle(style);

## **24、填充和颜色设置**

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/xjun15/article/details/5805429)

1. HSSFCellStyle style = wb.createCellStyle();
2. style.setFillBackgroundColor(HSSFColor.AQUA.index);
3. style.setFillPattern(HSSFCellStyle.BIG\_SPOTS);
4. HSSFCell cell = row.createCell((short) 1);
5. cell.setCellValue("X");
6. style = wb.createCellStyle();
7. style.setFillForegroundColor(HSSFColor.ORANGE.index);
8. style.setFillPattern(HSSFCellStyle.SOLID\_FOREGROUND);
9. cell.setCellStyle(style);

## **25、强行刷新单元格公式**

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/xjun15/article/details/5805429)

1. HSSFFormulaEvaluator eval=new HSSFFormulaEvaluator((HSSFWorkbook) wb);
2. private static void updateFormula(Workbook wb,Sheet s,int row){
3. Row r=s.getRow(row);
4. Cell c=null;
5. FormulaEcaluator eval=null;
6. if(wb instanceof HSSFWorkbook)
7. eval=new HSSFFormulaEvaluator((HSSFWorkbook) wb);
8. else if(wb instanceof XSSFWorkbook)
9. eval=new XSSFFormulaEvaluator((XSSFWorkbook) wb);
10. for(int i=r.getFirstCellNum();i
11. c=r.getCell(i);
12. if(c.getCellType()==Cell.CELL\_TYPE\_FORMULA)
13. eval.evaluateFormulaCell(c);
14. }
15. }

说明：FormulaEvaluator提供了evaluateFormulaCell(Cell cell)方法，计算公式保存结果，但不改变公式。而evaluateInCell(Cell cell) 方法是计算公式，并将原公式替换为计算结果，也就是说该单元格的类型不在是Cell.CELL\_TYPE\_FORMULA而是Cell.CELL\_TYPE\_NUMBERIC。HSSFFormulaEvaluator提供了静态方法evaluateAllFormu

laCells(HSSFWorkbook wb) ，计算一个Excel文件的所有公式，用起来很方便。

## -------------------------------------------poi 方法总结-------------------------------------------------------------------

-.设置不显示excel网格线   
  sheet.setDisplayGridlines(false);其中sheet是Sheet对象   
2.设置excel单元格中的内容换行   
  cellStyle.setWrapText(true);其中cellStyle是WorkBook创建的CellStyle对象,然后将cellStyle设置到要换行的Cell对象,最后在要换行的对象(一般为字符串)加入"/r/n"。如   
topTile.append("/r/n" +"cellContent");   
  
3.单元格的合并   
  sheet.addMergedRegion(new CellRangeAddress(0, 4, 0, 2));本示例为合并4行2列   
  
4.设置页眉和页脚的页数   
    HSSFHeader header = sheet.getHeader();   
    header.setCenter("Center Header");   
    header.setLeft("Left Header");   
    header.setRight(HSSFHeader.font("Stencil-Normal", "Italic") +   
    HSSFHeader.fontSize((short) 16) + "Right w/ Stencil-Normal Italic font and size 16");   
  
  HSSFFooter footer = (HSSFFooter )sheet.getFooter()   
  footer.setRight( "Page " + HSSFFooter.page() + " of " + HSSFFooter.numPages() );   
  
5.使得一个Sheet适合一页   
  sheet.setAutobreaks(true);   
6.设置放大属性(Zoom被明确为一个分数，例如下面的75%使用3作为分子，4作为分母)   
  sheet.setZoom(3,4);     
  
7.设置打印   
  HSSFPrintSetup print = (HSSFPrintSetup) sheet.getPrintSetup();   
  print.setLandscape(true);//设置横向打印   
  print.setScale((short) 70);//设置打印缩放70%   
  print.setPaperSize(HSSFPrintSetup.A4\_PAPERSIZE);//设置为A4纸张   
  print.setLeftToRight(true);//設置打印顺序先行后列,默认为先列行              
  print.setFitHeight((short) 10);设置缩放调整为10页高   
  print.setFitWidth((short) 10);设置缩放调整为宽高   
  
  sheet.setAutobreaks(false);   
  if (i != 0 && i % 30 == 0)   
      sheet.setRowBreak(i);//設置每30行分頁打印   
  
8.反复的行和列（设置打印标题）   
  HSSFWorkbook wb = new HSSFWorkbook();   
  wb.setRepeatingRowsAndColumns(0, 0, 12, 1, 6);//设置1到12列,行1到6每一页重复打印   
  
9.调整单元格宽度   
　　sheet.setAutobreaks(true);   
　　sheet.setColumnWidth((short)i,colsWidth[i]); //设定单元格长度   
    sheet.autoSizeColumn((short) i);//自动根据长度调整单元格长度