

Apache Poi

the Java API for ~~Microsoft~~ Documents



Brian S. Sheldon

Me

Married since 1982

Programmer since 1987

Two grown children and five soon to be adopted ages 2-16

Two dogs and a cat

I don't know what poi is made from

I have run over a dog and a skunk while riding my motorcycle

Kitchen





Goal

My goal for this talk is to provide all the bits and pieces you would need to create a spreadsheet with some charts. And some resources to find out more.

- Active sheet
- Cell styles
- Formulae
 - Evaluator
- Charts

** Ask questions at any time - remind me to repeat the question for those who are not here.

Speaking of Resources

09 April 2019 - POI 4.1.0 available

<https://github.com/brianssheldon/OkcJugPoi>

<https://poi.apache.org/>

<https://www.mkyong.com/java/apache-poi-reading-and-writing-excel-file-in-java/>

<https://mvnrepository.com/artifact/org.apache.poi>

<http://www.jfree.org/jfreechart/samples.html>

<http://zetcode.com/java/jfreechart/>

<https://www.mkyong.com/java/apache-poi-reading-and-writing-excel-file-in-java>

Things to know

Spreadsheets

- Workbook contains everything
- Worksheet is one page within the workbook. One or more worksheets may be in the workbook.
- Cell - one box within a sheet that contains a value or formula
- Row is a horizontal line of cells
- Column is a vertical line of cells
- Range - a grouping of cells by one or more rows by one or more columns
- Active sheet - the sheet that is displayed when you open a workbook

Class prefixes

<https://www.mkyong.com/java/apache-poi-reading-and-writing-excel-file-in-java/>

HSSF is prefixed before the class name to indicate operations related to a Microsoft Excel **2003** file.

XSSF is prefixed before the class name to indicate operations related to a Microsoft Excel **2007** file or later.

XSSFWorkbook and HSSFWorkbook are classes which act as an Excel Workbook

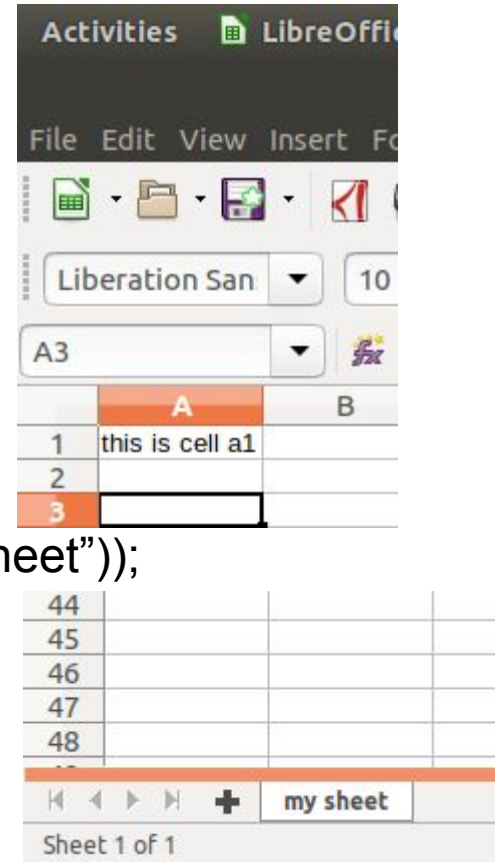
HSSFSheet and XSSFSheet are classes which act as an Excel Worksheet

Row defines an Excel row

Cell defines an Excel cell addressed in reference to a row.

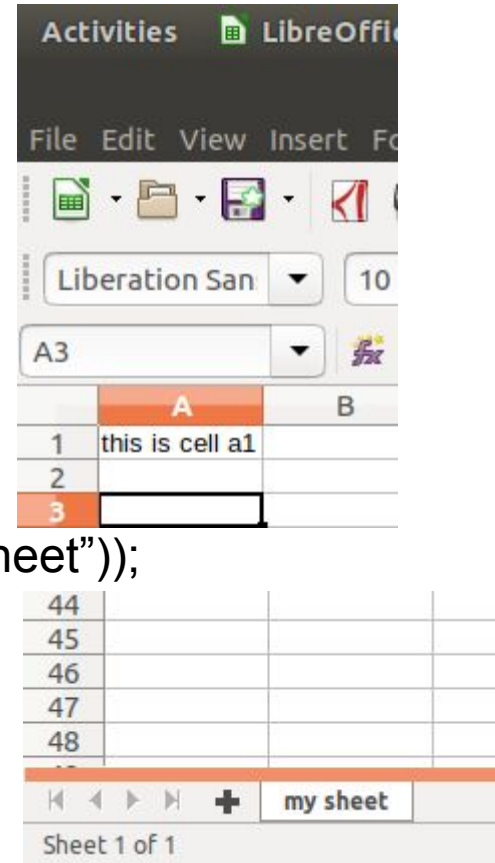
```
XSSFWorkbook workbook = new XSSFWorkbook();  
XSSFSheet sheet = workbook.createSheet("my sheet");  
Row row = sheet.createRow(0);  
Cell cell = row.createCell(0);  
cell.setCellValue((String) "this is cell a1");  
workbook.setActiveSheet(0);  
workbook.setActiveSheet(workbook.getSheetIndex("my sheet"));  
sheet.setColumnWidth(1, 5000); poiUno 73
```

```
FileOutputStream outputStream = new FileOutputStream("File1.xls");  
workbook.write(outputStream);  
workbook.close();
```



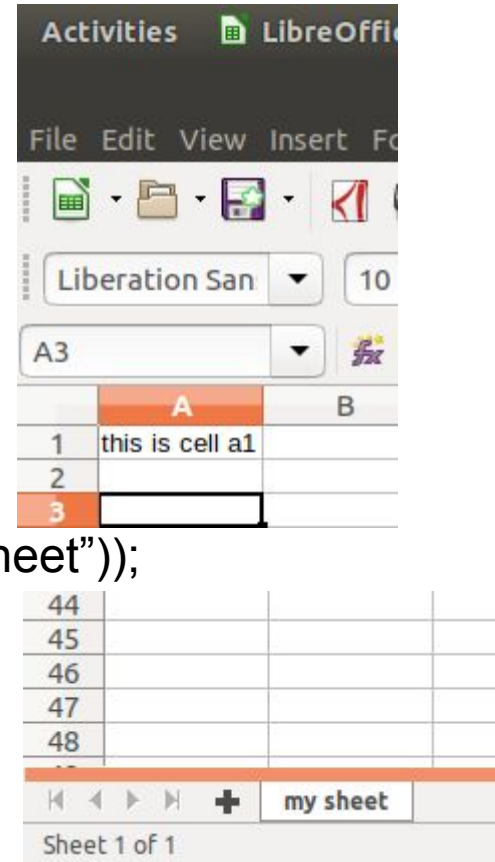
```
XSSFWorkbook workbook = new XSSFWorkbook();  
XSSFSheet sheet = workbook.createSheet("my sheet");  
Row row = sheet.createRow(0);  
Cell cell = row.createCell(0);  
cell.setCellValue((String) "this is cell a1");  
workbook.setActiveSheet(0);  
workbook.setActiveSheet(workbook.getSheetIndex("my sheet"));  
sheet.setColumnWidth(1, 5000); poiUno 73
```

```
FileOutputStream outputStream = new FileOutputStream("File1.xls");  
workbook.write(outputStream);  
workbook.close();
```



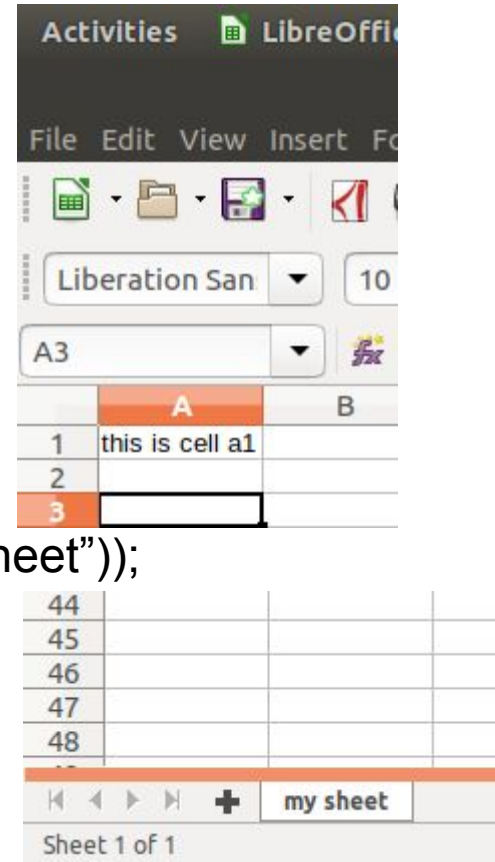
```
XSSFWorkbook workbook = new XSSFWorkbook();  
XSSFSheet sheet = workbook.createSheet("my sheet");  
Row row = sheet.createRow(0);  
Cell cell = row.createCell(0);  
cell.setCellValue((String) "this is cell a1");  
workbook.setActiveSheet(0);  
workbook.setActiveSheet(workbook.getSheetIndex("my sheet"));  
sheet.setColumnWidth(1, 5000); poiUno 73
```

```
FileOutputStream outputStream = new FileOutputStream("File1.xls");  
workbook.write(outputStream);  
workbook.close();
```



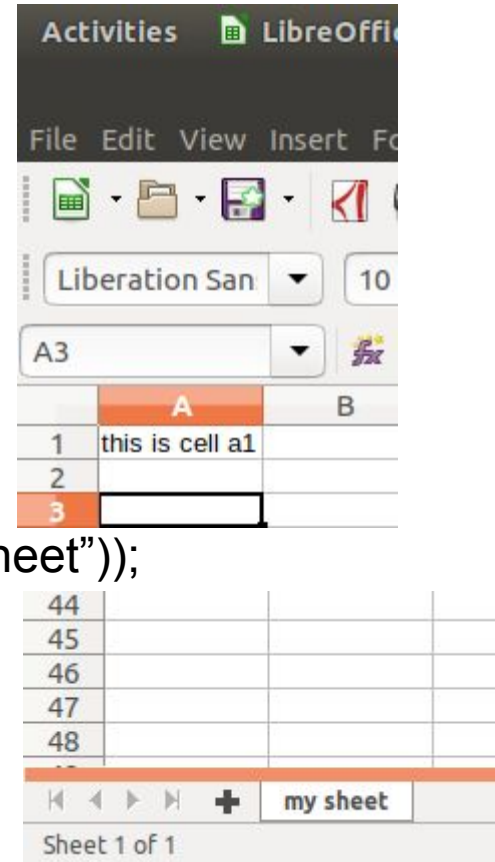
```
XSSFWorkbook workbook = new XSSFWorkbook();  
XSSFSheet sheet = workbook.createSheet("my sheet");  
Row row = sheet.createRow(0);  
Cell cell = row.createCell(0);  
cell.setCellValue((String) "this is cell a1");  
workbook.setActiveSheet(0);  
workbook.setActiveSheet(workbook.getSheetIndex("my sheet"));  
sheet.setColumnWidth(1, 5000); poiUno 73
```

```
FileOutputStream outputStream = new FileOutputStream("File1.xls");  
workbook.write(outputStream);  
workbook.close();
```

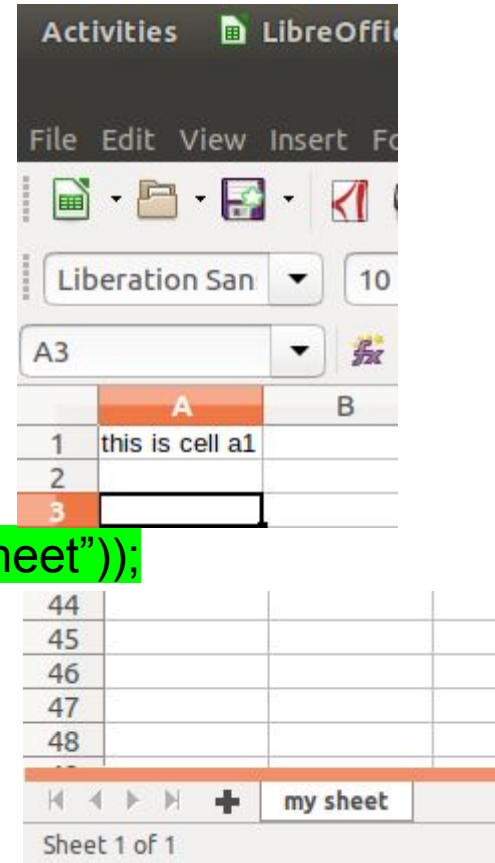


```
XSSFWorkbook workbook = new XSSFWorkbook();  
XSSFSheet sheet = workbook.createSheet("my sheet");  
Row row = sheet.createRow(0);  
Cell cell = row.createCell(0);  
cell.setCellValue((String) "this is cell a1");  
workbook.setActiveSheet(0);  
workbook.setActiveSheet(workbook.getSheetIndex("my sheet"));  
sheet.setColumnWidth(1, 5000); poiUno 73
```

```
FileOutputStream outputStream = new FileOutputStream("File1.xls");  
workbook.write(outputStream);  
workbook.close();
```



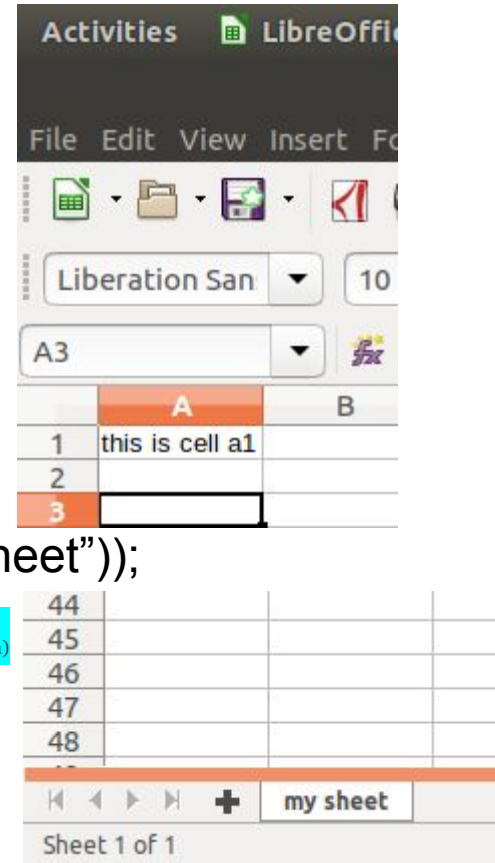
```
XSSFWorkbook workbook = new XSSFWorkbook();  
XSSFSheet sheet = workbook.createSheet("my sheet");  
Row row = sheet.createRow(0);  
Cell cell = row.createCell(0);  
cell.setCellValue((String) "this is cell a1");  
workbook.setActiveSheet(0);  
workbook.setActiveSheet(workbook.getSheetIndex("my sheet"));  
Int activeSheetIndex = workbook.getActiveSheetIndex();  
sheet.setColumnWidth(1, 5000); poiUno 73  
FileOutputStream outputStream = new FileOutputStream("File1.xls");  
workbook.write(outputStream);  
workbook.close();
```




```
XSSFWorkbook workbook = new XSSFWorkbook();  
XSSFSheet sheet = workbook.createSheet("my sheet");  
Row row = sheet.createRow(0);  
Cell cell = row.createCell(0);  
cell.setCellValue((String) "this is cell a1");  
workbook.setActiveSheet(0);  
workbook.setActiveSheet(workbook.getSheetIndex("my sheet"));  
sheet.setColumnWidth(1, 5000);
```

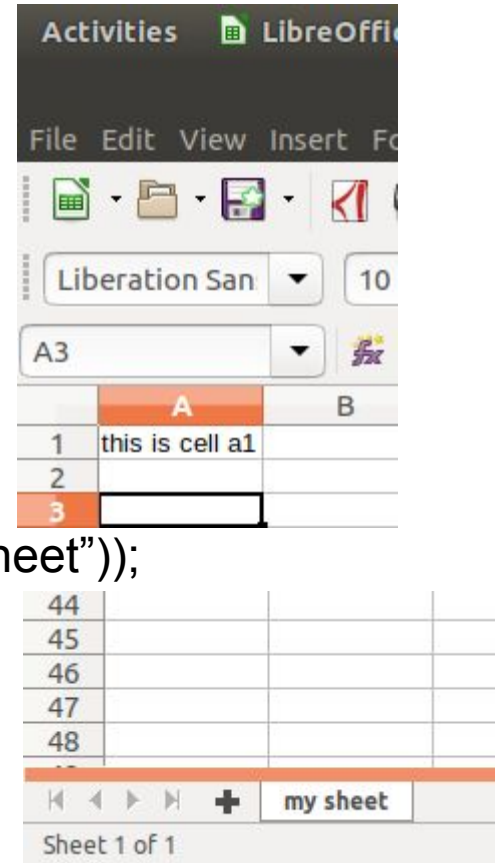
poiUno 73 Set the width (in units of 1/256th of a character width)

```
FileOutputStream outputStream = new FileOutputStream("File1.xls");  
workbook.write(outputStream);  
workbook.close();
```



```
XSSFWorkbook workbook = new XSSFWorkbook();  
XSSFSheet sheet = workbook.createSheet("my sheet");  
Row row = sheet.createRow(0);  
Cell cell = row.createCell(0);  
cell.setCellValue((String) "this is cell a1");  
workbook.setActiveSheet(0);  
workbook.setActiveSheet(workbook.getSheetIndex("my sheet"));  
sheet.setColumnWidth(1, 5000); poiUno 73
```

```
FileOutputStream outputStream = new FileOutputStream("File1.xls");  
workbook.write(outputStream);  
workbook.close();
```



Cell Style

Borders, fill, fill pattern, rotation, indent...

<https://poi.apache.org/apidocs/dev/org/apache/poi/ss/usermodel/CellStyle.html>

Center the contents of a cell

```
XSSFCellStyle style =  
    (XSSFCellStyle) my_workbook.createCellStyle();  
style = my_workbook.getCellStyleAt(0);
```

XSSFCellStyle implements CellStyle

```
CellStyle cellStyle = cell.getCellStyle();
```

XSSFCellStyle implements CellStyle

```
XSSFCellStyle ss = my_workbook.createCellStyle();
```

```
ss = my_workbook.getCellStyleAt(0);
```

```
XSSFFont font = ss.getFont();
```

```
XSSFFont font = my_workbook.getFontAt(fontIndex);
```

```
cellStyle.setAlignment(HorizontalAlignment.CENTER);  
cell.setCellStyle(cellStyle);
```

```
CellStyle cellStyle = cell.getCellStyle();
```

```
cellStyle.setAlignment(HorizontalAlignment.CENTER);
```

```
cellStyle.setBorderLeft(BorderStyle.DASHED);
```

```
cellStyle.setFillForegroundColor(HSSFColorPredefined.LIME.getIndex());
```

```
cellStyle.setFillPattern(FillPatternType.SOLID_FOREGROUND);
```

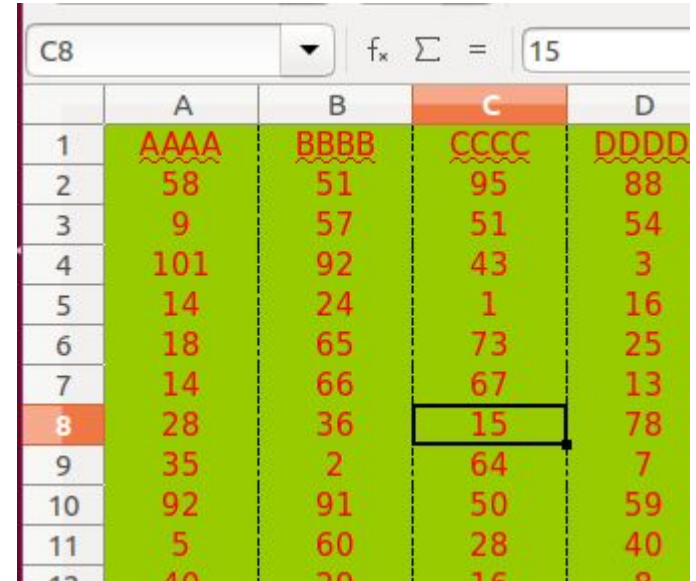
```
XSSFFont font = my_workbook.createFont();
```

```
font.setColor(HSSFColorPredefined.RED.getIndex());
```

```
cellStyle.setFont(font);
```

```
cellStyle = cell.getCellStyle();
```

```
cell.setCellStyle(cellStyle);
```



	A	B	C	D
1	AAAA	BBBB	CCCC	DDDD
2	58	51	95	88
3	9	57	51	54
4	101	92	43	3
5	14	24	1	16
6	18	65	73	25
7	14	66	67	13
8	28	36	15	78
9	35	2	64	7
10	92	91	50	59
11	5	60	28	40
12	40	20	16	8

```
CellStyle cellStyle = cell.getCellStyle();
```

```
cellStyle.setAlignment(HorizontalAlignment.CENTER);
```

```
cellStyle.setBorderLeft(BorderStyle.DASHED);
```

```
cellStyle.setFillForegroundColor(HSSFColorPredefined.LIME.getIndex());
```

```
cellStyle.setFillPattern(FillPatternType.SOLID_FOREGROUND);
```

```
XSSFFont font = my_workbook.createFont();
```

```
font.setColor(HSSFColorPredefined.RED.getIndex());
```

```
cellStyle.setFont(font);
```

```
cellStyle = cell.getCellStyle();
```

```
cell.setCellStyle(cellStyle);
```

	A	B	C	D
1	AAAA	BBBB	CCCC	DDDD
2	58	51	95	88
3	9	57	51	54
4	101	92	43	3
5	14	24	1	16
6	18	65	73	25
7	14	66	67	13
8	28	36	15	78
9	35	2	64	7
10	92	91	50	59
11	5	60	28	40
12	40	20	16	8


```
CellStyle cellStyle = cell.getCellStyle();
```

```
cellStyle.setAlignment(HorizontalAlignment.CENTER);
```

```
cellStyle.setBorderLeft(BorderStyle.DASHED);
```

```
cellStyle.setFillForegroundColor(HSSFColorPredefined.LIME.getIndex());
```

```
cellStyle.setFillPattern(FillPatternType.SOLID_FOREGROUND);
```

```
XSSFFont font = my_workbook.createFont();
```

```
font.setColor(HSSFColorPredefined.RED.getIndex());
```

```
cellStyle.setFont(font);
```

```
cellStyle = cell.getCellStyle();
```

```
cell.setCellStyle(cellStyle);
```

	A	B	C	D
1	AAAA	BBBB	CCCC	DDDD
2	58	51	95	88
3	9	57	51	54
4	101	92	43	3
5	14	24	1	16
6	18	65	73	25
7	14	66	67	13
8	28	36	15	78
9	35	2	64	7
10	92	91	50	59
11	5	60	28	40
12	40	20	16	8

```
CellStyle cellStyle = cell.getCellStyle();
```

```
cellStyle.setAlignment(HorizontalAlignment.CENTER);
```

```
cellStyle.setBorderLeft(BorderStyle.DASHED);
```

```
cellStyle.setFillForegroundColor(HSSFColorPredefined.LIME.getIndex());
```

```
cellStyle.setFillPattern(FillPatternType.SOLID_FOREGROUND);
```

```
XSSFFont font = my_workbook.createFont();
```

```
font.setColor(HSSFColorPredefined.RED.getIndex());
```

```
cellStyle.setFont(font);
```

```
cellStyle = cell.getCellStyle();
```

```
cell.setCellStyle(cellStyle);
```

	A	B	C	D
1	AAAA	BBBB	CCCC	DDDD
2	58	51	95	88
3	9	57	51	54
4	101	92	43	3
5	14	24	1	16
6	18	65	73	25
7	14	66	67	13
8	28	36	15	78
9	35	2	64	7
10	92	91	50	59
11	5	60	28	40
12	40	20	16	8

```
CellStyle cellStyle = cell.getCellStyle();
```

```
cellStyle.setAlignment(HorizontalAlignment.CENTER);
```

```
cellStyle.setBorderLeft(BorderStyle.DASHED);
```

```
cellStyle.setFillForegroundColor(HSSFColorPredefined.LIME.getIndex());
```

```
cellStyle.setFillPattern(FillPatternType.SOLID_FOREGROUND);
```

```
cellStyle.setFillPattern(FillPatternType.DIAMONDS);
```

```
XSSFFont font = my_workbook.createFont();
```

```
font.setColor(HSSFColorPredefined.RED.getIndex());
```

```
cellStyle.setFont(font);
```

```
cellStyle = cell.getCellStyle();
```

```
cell.setCellStyle(cellStyle);
```

	A	B	C	D
1	AAAA	BBBB	CCCC	DDDD
2	58	51	95	88
3	9	57	51	54
4	101	92	43	3
5	14	24	1	16
6	18	65	73	25
7	14	66	67	13
8	28	36	15	78
9	35	2	64	7
10	92	91	50	59
11	5	60	28	40
12	40	20	16	8

```
CellStyle cellStyle = cell.getCellStyle();
```

```
cellStyle.setAlignment(HorizontalAlignment.CENTER);
```

```
cellStyle.setBorderLeft(BorderStyle.DASHED);
```

```
cellStyle.setFillForegroundColor(HSSFColorPredefined.LIME.getIndex());
```

```
cellStyle.setFillPattern(FillPatternType.SOLID_FOREGROUND);
```

```
XSSFFont font = my_workbook.createFont();
```

```
font.setColor(HSSFColorPredefined.RED.getIndex());
```

```
cellStyle.setFont(font);
```

```
cellStyle = cell.getCellStyle();
```

```
cell.setCellStyle(cellStyle);
```

	A	B	C	D
1	AAAA	BBBB	CCCC	DDDD
2	58	51	95	88
3	9	57	51	54
4	101	92	43	3
5	14	24	1	16
6	18	65	73	25
7	14	66	67	13
8	28	36	15	78
9	35	2	64	7
10	92	91	50	59
11	5	60	28	40
12	40	20	16	8

```
CellStyle cellStyle = cell.getCellStyle();
```

```
cellStyle.setAlignment(HorizontalAlignment.CENTER);
```

```
cellStyle.setBorderLeft(BorderStyle.DASHED);
```

```
cellStyle.setFillForegroundColor(HSSFColorPredefined.LIME.getIndex());
```

```
cellStyle.setFillPattern(FillPatternType.SOLID_FOREGROUND);
```

```
XSSFFont font = my_workbook.createFont();
```

```
font.setColor(HSSFColorPredefined.RED.getIndex());
```

```
cellStyle.setFont(font);
```

```
cellStyle = cell.getCellStyle();
```

```
cell.setCellStyle(cellStyle);
```

	A	B	C	D
1	AAAA	BBBB	CCCC	DDDD
2	58	51	95	88
3	9	57	51	54
4	101	92	43	3
5	14	24	1	16
6	18	65	73	25
7	14	66	67	13
8	28	36	15	78
9	35	2	64	7
10	92	91	50	59
11	5	60	28	40
12	40	20	16	8

Formulas

aka

Formulae

<https://poi.apache.org/components/spreadsheet/eval.html>

```
Int i = 8;
```

```
dataSheet.getRow(i).createCell(15)  
    .setCellFormula("SUM(A" + (i + 1) + ":O" + (i + 1) + ")");
```

```
// in row 9, cell 16 should be the formula "=sum(A9:O9)"
```

```
dataSheet.getRow(51).createCell(i).setCellFormula(  
    "AVERAGE(" + x + "2:" + x + "50)");  
    =AVERAGE(B2:B50)
```

```
XSSFCellStyle style = (XSSFCellStyle) my_workbook.createCellStyle();
```

```
style.setDataFormat(my_workbook.createDataFormat().getFormat("0.00"));
```


Evaluate

4 flavors

<https://poi.apache.org/components/spreadsheet/eval.html>

```
FormulaEvaluator evaluator = wb.getCreationHelper().createFormulaEvaluator();  
  
if (c.getCellType() == Cell.CELL_TYPE_FORMULA) {  
    CellValue cellValue = evaluator.evaluate(cell); // returns the calc'd value  
}
```

<https://poi.apache.org/components/spreadsheet/eval.html>

```
FormulaEvaluator evaluator = wb.getCreationHelper().createFormulaEvaluator();  
  
if (c.getCellType() == Cell.CELL_TYPE_FORMULA) {  
    evaluator.evaluateFormulaCell(c); // recalculates the formula  
}
```

```
FormulaEvaluator evaluator = wb.getCreationHelper().createFormulaEvaluator();
```

```
if (c.getCellType() == Cell.CELL_TYPE_FORMULA) {
```

```
    evaluator.evaluateInCell(cell); // replaces the formula with the calculated value
```

```
}
```

<https://poi.apache.org/components/spreadsheet/eval.html>

```
XSSFFormulaEvaluator.evaluateAllFormulaCells(workbook);
```

Evaluator examples

```
cell.setCellFormula("5*10*2");
```

```
CellValue cellValue = evaluator.evaluate(cell); // returns the calc'd value
```

```
System.err.println("1 " + cellValue.getNumberValue()); //1 100.0
```

Evaluator examples

```
cell.setCellFormula("5*10*2");  
evaluator.evaluateFormulaCell(cell); // recalculates the formula  
System.out.println("2 " + cell.getCellType()); //2 FORMULA  
System.out.println("2 " + cell.getCellFormula()); //2 5*10*2
```

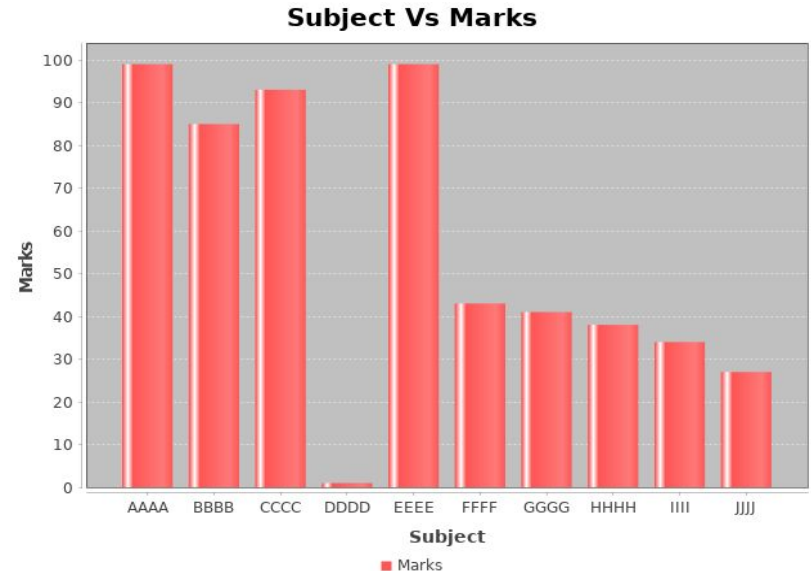
Evaluator examples

```
cell.setCellFormula("5*10*2");  
evaluator.evaluateInCell(cell); // replaces the formula with the calculated value  
System.out.println("3 " + cell.getCellType());      //3 NUMERIC  
System.out.println("3 " + cell.getNumericCellValue()); //3 100.0
```


Charts

```
JFreeChart barChartObject = ChartFactory.createBarChart(  
    "Subject Vs Marks",           // title  
    "Subject",                    // category axis label  
    "Marks",                      // category axis label  
    my_bar_chart_dataset,         // dataset  
    PlotOrientation.VERTICAL,     // PlotOrientation  
    true,                         // legend  
    true,                         // tooltips  
    false);                       //urls
```

<http://www.jfree.org/jfreechart/>



Create a byte array output stream from the chart

```
ByteArrayOutputStream chart_out = new ByteArrayOutputStream();
byte[] imageInByte = null;
try {
    BufferedImage bi = barChartObject.createBufferedImage(640, 480);
    ByteArrayOutputStream baos = new ByteArrayOutputStream();
    ImageIO.write(bi, "png", baos);
    baos.flush();
    imageInByte = baos.toByteArray();
    baos.close();
} catch (IOException ex) {
    Logger.getLogger(MakeBarChart.class.getName()).log(Level.SEVERE, null,
ex);
}
```

```
int my_picture_id = my_workbook.addPicture(imageInByte,  
    Workbook.PICTURE_TYPE_PNG);  
  
try {  
    chart_out.close();  
  
} catch (IOException ex) {  
    Logger.getLogger(MakeBarChart.class.getName()).log(Level.SEVERE, null, ex);  
}
```

```
/* Create the drawing container */
```

```
XSSFDrawing drawing = my_sheet.createDrawingPatriarch();
```

```
/* Create an anchor point */
```

```
ClientAnchor my_anchor = new XSSFClientAnchor();
```

```
/* Define top left corner, and we can resize picture suitable from there */
```

```
my_anchor.setCol1(4);
```

```
my_anchor.setRow1(5);
```

```
/* Invoke createPicture and pass the anchor point and ID */
```

```
XSSFPicture my_picture = drawing.createPicture(my_anchor, my_picture_id);
```

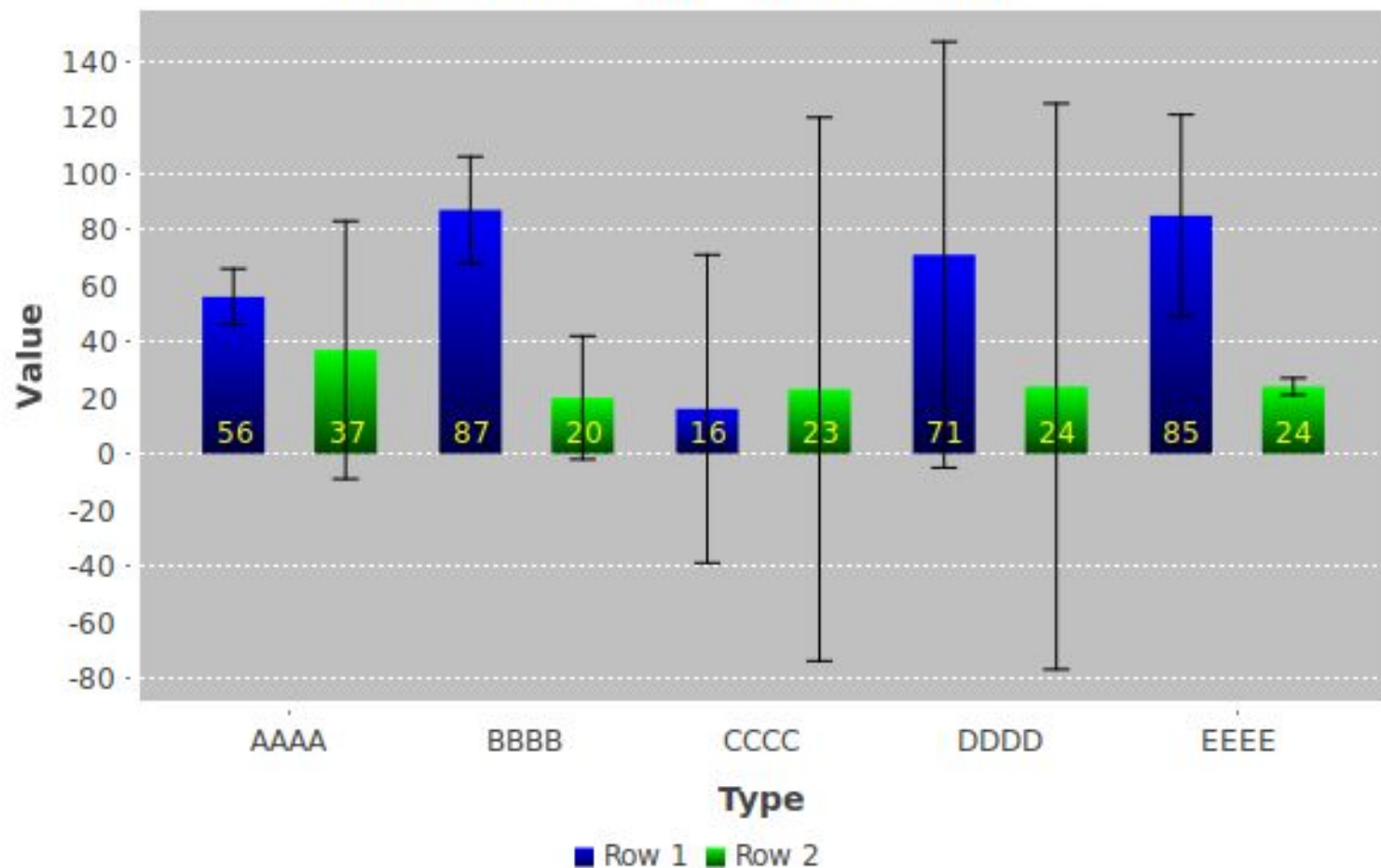
```
/* Call resize method, which resizes the image to the size of the image */
```

```
my_picture.resize(); // required or the image will not show up
```

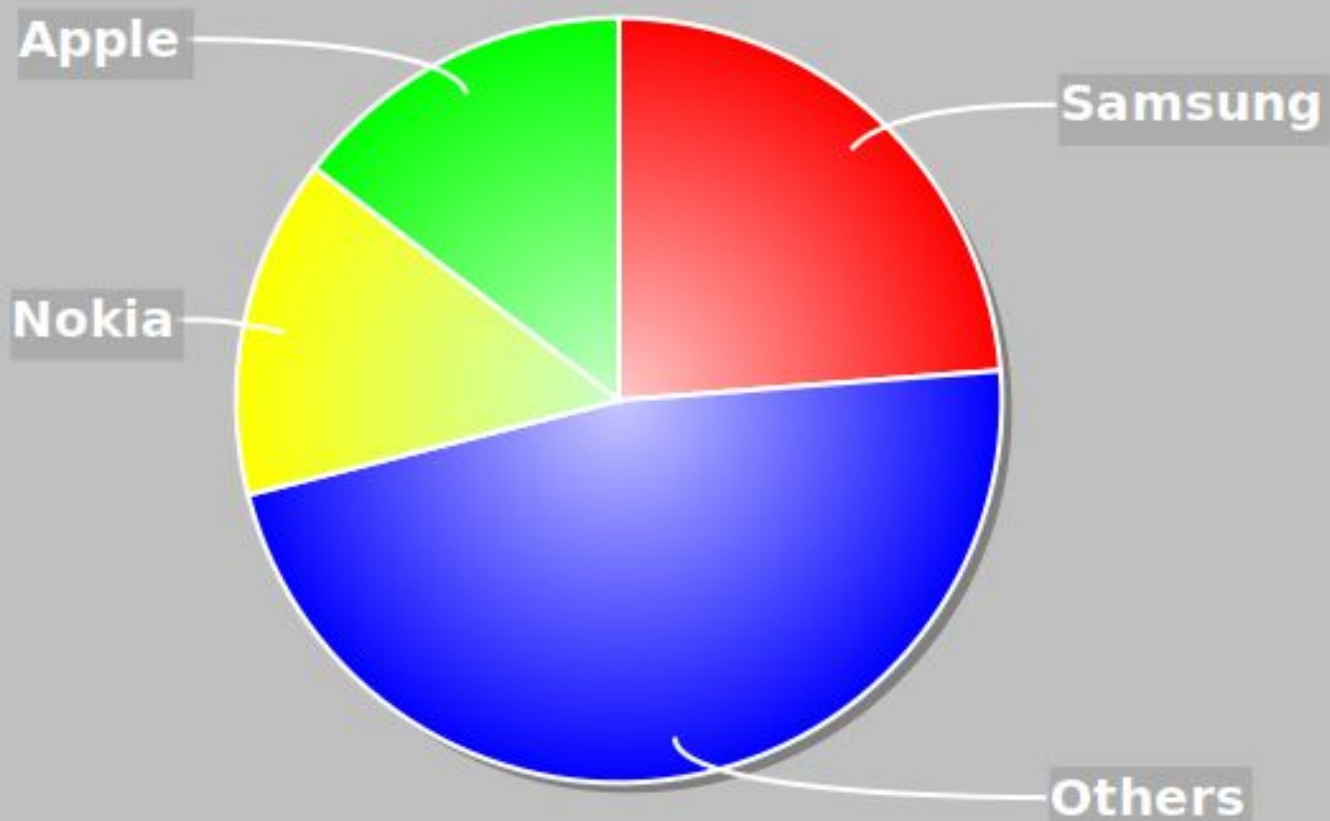
FreeChart

<http://www.java2s.com/Code/Java/Chart/JFreeChartDualAxisDemo.htm>

Statistical Bar Chart Demo 1



Smart Phones Manufactured / Q3 2011



Sounds useful & interesting <https://tika.apache.org/>

a content analysis toolkit

The Apache Tika™ toolkit detects and extracts metadata and text from over a thousand different file types (such as PPT, XLS, and PDF). All of these file types can be parsed through a single interface, making Tika useful for search engine indexing, content analysis, translation, and much more