

¿Qué ofrece Autentia Real Business Solutions S.L?

Somos su empresa de **Soporte a Desarrollo Informático**.
 Ese apoyo que siempre quiso tener...

1. Desarrollo de componentes y proyectos a medida



2. Auditoría de código y recomendaciones de mejora

3. Arranque de proyectos basados en nuevas tecnologías

1. Definición de frameworks corporativos.
2. Transferencia de conocimiento de nuevas arquitecturas.
3. Soporte al arranque de proyectos.
4. Auditoría preventiva periódica de calidad.
5. Revisión previa a la certificación de proyectos.
6. Extensión de capacidad de equipos de calidad.
7. Identificación de problemas en producción.



4. Cursos de formación (impartidos por desarrolladores en activo)

Spring MVC, JSF-PrimeFaces /RichFaces,
 HTML5, CSS3, JavaScript-jQuery

Gestor portales (Liferay)
 Gestor de contenidos (Alfresco)
 Aplicaciones híbridas

Tareas programadas (Quartz)
 Gestor documental (Alfresco)
 Inversión de control (Spring)

Control de autenticación y
 acceso (Spring Security)
 UDDI
 Web Services
 Rest Services
 Social SSO
 SSO (Cas)

JPA-Hibernate, MyBatis
 Motor de búsqueda empresarial (Solr)
 ETL (Talend)

Dirección de Proyectos Informáticos.
 Metodologías ágiles
 Patrones de diseño
 TDD

BPM (jBPM o Bonita)
 Generación de informes (JasperReport)
 ESB (Open ESB)

E-mail: Contraseña:

[Deseo registrarme](#)
[He olvidado mis datos de acceso](#)

[Inicio](#) [Quiénes somos](#) [Tutoriales](#) [Formación](#) [Comparador de salarios](#) [Nuestro libro](#) [Charlas](#) [Más](#)

Estás en: [Inicio](#) [Tutoriales](#) Generar hojas de cálculo con fórmulas mediante Apache POI

	DESARROLLADO POR: Miguel Arlandy Rodríguez	Consultor tecnológico de desarrollo de proyectos informáticos.
		Puedes encontrarme en Autentia : Ofrecemos servicios de soporte a desarrollo, factoría y formación
		Somos expertos en Java/JEE
		Ver tutoriales de Miguel Arlandy Rodríguez



Fecha de publicación del tutorial: 2011-11-02



Share |

0

[Regístrate para votar](#)

Generar hojas de cálculo con fórmulas mediante Apache POI.

[Catálogo de servicios Autentia](#)

0. Índice de contenidos.

- 1. Introducción.
- 2. Entorno.
- 3. Diseñando la hoja de cálculo.
- 4. La clase Piloto.
- 5. Generando la hoja de cálculo.
- 6. Ejecutando el ejemplo.
- 7. Referencias.
- 8. Conclusiones.

1. Introducción

Las hojas de cálculo son una poderosa herramienta de la que nos proveen las suites de ofimática. Con ellas podemos realizar infinidad de tareas como llevar nuestras cuentas domésticas, gestionar tareas o explotar métricas. Un aspecto muy interesante de las hojas de cálculo son las fórmulas, que nos permiten tratar datos que tengamos almacenados en ellas.

En este tutorial vamos a ver cómo generar hojas de cálculo con fórmulas con ayuda de la librería Apache POI.

2. Entorno.

El tutorial está escrito usando el siguiente entorno:

- Hardware: Portátil MacBook Pro 15' (2.2 Ghz Intel Core I7, 4GB DDR3).
- Sistema Operativo: Mac OS Snow Leopard 10.6.7
- Entorno de desarrollo: Eclipse 3.7 Indigo.
- Apache POI 3.7.
- Libre Office 3.4.3

3. Diseñando la hoja de cálculo.

En nuestro ejemplo vamos a generar una hoja de cálculo que almacene los tiempos que tardan los pilotos de carreras en dar las vueltas de entrenamiento. Además la hoja de cálculo deberá ser capaz de tratar esos tiempos para generar información adicional. Los requisitos serán los siguientes:

- En la primera fila de la hoja de cálculo debe aparecer una descripción de las columnas.
- En la primera columna debe aparecer el nombre del piloto.
- Cada piloto dará 5 vueltas de entrenamiento por lo que las 5 columnas siguientes deben corresponder a los tiempos en segundos que el piloto ha tardado en dar cada una de las vueltas.
- La siguiente columna será el total en segundos que el piloto tardó en dar las 5 vueltas. Se calculará mediante una fórmula.
- La siguiente columna será el promedio de tiempo que tardó el piloto teniendo en cuenta las 5 vueltas. Se calculará mediante una fórmula.
- La siguiente columna será el mejor tiempo de las 5 vueltas. Se calculará mediante una fórmula.

4. La clase Piloto.

Para almacenar el nombre del piloto y el tiempo de sus vueltas crearemos una clase a la que llamaremos Piloto y de la que hará uso la clase que genere nuestra hoja de cálculo.

```
01 import java.util.ArrayList;
02 import java.util.List;
03
04 public class Piloto {
05
06     public static final int NUMERO_VUELTAS_ENTRENAMIENTO = 5;
07
08     private final String nombre;
09
10     private final List<Double> tiemposVueltas;
11
```



Últimas Noticias

- [Crónica del evento de Liferay en Madrid](#)
- [El primer capítulo de Terrakas ya está online](#)
- [Ya ha terminado la CAS 2011, ahora toca pensar cómo me gustaría que fuera la CAS 2012](#)
- [Restrospectiva, Carrera de las empresas 2011](#)
- [¿Qué ganan los demás con que tu vayas a una conferencia?](#)

[Histórico de NOTICIAS](#)

Últimos Tutoriales

- [Geoposicionamiento Web con HTML5 y Google Maps](#)
- [Primeros pasos con Spring Web Flow 2](#)
- [Karmacracry, diviertete compartiendo.](#)
- [Cómo centrar una página web en el navegador.](#)
- [Como hacer nuestros test más legibles con Hamcrest](#)

-  [El patrón de diseño Template Method](#)
-  [Clean Code: reglas y principios](#)
-  [Clean Code: Impresiones](#)
-  [Spring MVC: acceder a las propiedades de un fichero desde una JSP con Expression Language \(EL\)](#)
-  [Configurar Spring Security 3.1 para autenticarse contra un Active Directory](#)

Síguenos a través de:



Últimas ofertas de empleo

- 2011-09-08  [Comercial - Ventas - MADRID.](#)
- 2011-09-03  [Comercial - Ventas - VALENCIA.](#)
- 2011-08-19  [Comercial - Compras - ALICANTE.](#)
- 2011-07-12  [Otras Sin catalogar - MADRID.](#)
- 2011-07-06  [Otras Sin catalogar - LUGO.](#)

```

13 public Piloto(String nombre) {
14     this.tiemposVueltas = new ArrayList<Double>();
15 }
16
17 public String getNombre() {
18     return nombre;
19 }
20
21 public List<Double> getTiemposVueltas() {
22     return tiemposVueltas;
23 }
24 }

```

De momento fácil, ¿no?.

5. Generando la hoja de cálculo.

Una vez tenemos la clase que nos proporcionará el nombre del piloto y sus tiempos en dar las vueltas, vamos con la gracia de todo esto: la clase que nos creará la hoja de cálculo.

La clase generará una hoja de cálculo (recordemos que un documento puede tener más de una hoja de cálculo) llamada "Tiempos entrenamientos" y generará un fichero llamado tiempos-entrenamientos.xls

Para las fórmulas utilizaremos las siguientes funciones: SUM (sumará el total de tiempos de un piloto), AVERAGE (calculará la media de tiempos de un piloto) y MIN (calculará el mejor tiempo en dar una vuelta).

Además, vamos a añadir un estilo propio a las celdas de la primera fila (la que contiene los nombres de las columnas) y otro estilo a las celdas con fórmula para que resalten un poco más. Finalmente auto-ajustaremos el ancho de las columnas para que se adapten al contenido.

```

001 import java.io.FileOutputStream;
002 import java.io.IOException;
003 import java.io.OutputStream;
004
005 import org.apache.poi.hssf.usermodel.HSSFWorkbook;
006 import org.apache.poi.ss.usermodel.Cell;
007 import org.apache.poi.ss.usermodel.CellStyle;
008 import org.apache.poi.ss.usermodel.Font;
009 import org.apache.poi.ss.usermodel.IndexedColors;
010 import org.apache.poi.ss.usermodel.Row;
011 import org.apache.poi.ss.usermodel.Sheet;
012 import org.apache.poi.ss.usermodel.Workbook;
013
014 public class GeneradorHojaCalculo {
015
016     // documento con las hojas de calculo
017     private final Workbook libro;
018
019     // la hoja de calculo
020     private final Sheet hojaTiemposPilotos;
021
022     // estilo de las celdas del encabezado (con el nombre de las columnas)
023     private final CellStyle estiloTitulo;
024
025     // estilo de las celdas con fórmula
026     private final CellStyle estiloCeldaConFormula;
027
028     public GeneradorHojaCalculo() {
029         this.libro = new HSSFWorkbook();
030         this.hojaTiemposPilotos = this.libro.createSheet("Tiempos entrenamientos");
031         this.estiloTitulo = getEstiloTitulo();
032         this.estiloCeldaConFormula = getEstiloCeldaConFormula();
033         anadeFilaEncabezado();
034     }
035
036     // crea una fila con los datos del piloto: nombre, tiempos, total, media y mejor tiempo
037     public void anadeTiemposPiloto(Piloto piloto) {
038         final Row filaPiloto = getNuevaFila();
039         filaPiloto.createCell(0).setCellValue(piloto.getNombre());
040         for (int i = 1; i <= Piloto.NUMERO_VUELTAS_ENTRENAMIENTO; i++) {
041             final Cell celda = filaPiloto.createCell(i);
042             celda.setCellValue(piloto.getTiemposVueltas().get(i - 1));
043             celda.setCellType(Cell.CELL_TYPE_NUMERIC);
044         }
045         generaFormulasSumaTiempos(filaPiloto);
046         generaFormulasMediaTiempos(filaPiloto);
047         generaFormulasMejorTiempo(filaPiloto);
048     }
049
050     // crea la celda con la fórmula de suma de tiempos correspondiente a una fila
051     private void generaFormulasSumaTiempos(Row filaPiloto) {
052         final int numeroFila = filaPiloto.getRowNum() + 1;
053         final String formula = "SUM" + generaRangoFormulaEnFila(numeroFila);
054         anadeFormulasEstiloACelda(filaPiloto.createCell(Piloto.NUMERO_VUELTAS_ENTRENAMIENTO + 1),
055 formula);
056     }
057
058     // crea la celda con la fórmula de media de tiempos correspondiente a una fila
059     private void generaFormulasMediaTiempos(Row filaPiloto) {
060         final int numeroFila = filaPiloto.getRowNum() + 1;
061         final String formula = "AVERAGE" + generaRangoFormulaEnFila(numeroFila);
062         anadeFormulasEstiloACelda(filaPiloto.createCell(Piloto.NUMERO_VUELTAS_ENTRENAMIENTO + 2),
063 formula);
064     }
065
066     // crea la celda con la fórmula de que calcula el mejor tiempo a una fila
067     private void generaFormulasMejorTiempo(Row filaPiloto) {
068         final int numeroFila = filaPiloto.getRowNum() + 1;
069         final String formula = "MIN" + generaRangoFormulaEnFila(numeroFila);
070         anadeFormulasEstiloACelda(filaPiloto.createCell(Piloto.NUMERO_VUELTAS_ENTRENAMIENTO + 3),
071 formula);
072     }
073
074     // devuelve el rango de columnas sobre las que actuará la formula. Ej: (B2:F2)
075     // la columna donde se situa el primer tiempo será la B (codigo ASCII 66) ya que en la A
076     // está el nombre del piloto)
077     private static String generaRangoFormulaEnFila(int numeroFila) {
078         final byte columnaB = 66;
079         final char primeraColumna = (char)columnaB;
080         final char ultimaColumna = (char)columnaB + Piloto.NUMERO_VUELTAS_ENTRENAMIENTO - 1;
081         return "(" + primeraColumna + numeroFila + ":" + ultimaColumna + numeroFila + ")";
082     }
083
084     // añade la fórmula a una celda y añade el estilo de las celdas con fórmula
085     private void anadeFormulasEstiloACelda(Cell celda, String formula) {
086         celda.setCellFormula(formula);
087         celda.setCellStyle(estiloCeldaConFormula);
088     }
089 }

```

```

085
086 // genera el documento
087 public OutputStream generaDocumento() throws IOException {
088     ajustaColumnas();
089     final OutputStream outputStream = new FileOutputStream("tiempos-entrenamientos.xls");
090     libro.write(outputStream);
091     outputStream.close();
092     return outputStream;
093 }
094
095 // crea la fila y celdas del encabezado con el nombre de las columnas
096 private void anadeFilaEncabezado() {
097     final Row filaEncabezado = getNuevaFila();
098     int numeroCelda = 0;
099     creaCeldaEncabezado(filaEncabezado, numeroCelda++, "Piloto");
100     for (int i = 1; i <= Piloto.NUMERO_VUELTAS_ENTRENAMIENTO; i++) {
101         creaCeldaEncabezado(filaEncabezado, numeroCelda++, "Vuelta " + i);
102     }
103     creaCeldaEncabezado(filaEncabezado, numeroCelda++, "Tiempo total");
104     creaCeldaEncabezado(filaEncabezado, numeroCelda++, "Promedio");
105     creaCeldaEncabezado(filaEncabezado, numeroCelda++, "Mejor tiempo");
106 }
107
108 // crea una celda de encabezado (las del título) y añade el estilo
109 private void creaCeldaEncabezado(Row filaEncabezado, int numeroCelda, String valor) {
110     final Cell celdaEncabezado = filaEncabezado.createCell(numeroCelda);
111     celdaEncabezado.setCellValue(valor);
112     celdaEncabezado.setCellStyle(estiloTitulo);
113 }
114
115 // ajusta el ancho de las columnas en función de su contenido
116 private void ajustaColumnas() {
117     final short numeroColumnas = hojaTiemposPilotos.getRow(0).getLastCellNum();
118     for (int i = 0; i < numeroColumnas; i++) {
119         hojaTiemposPilotos.autoSizeColumn(i);
120     }
121 }
122
123 // devuelve el estilo que tendrán las celdas del título (negrita y color de fondo azul)
124 private CellStyle getEstiloTitulo() {
125     final CellStyle cellStyle = libro.createCellStyle();
126     final Font cellFont = libro.createFont();
127     cellFont.setBoldweight(Font.BOLDWEIGHT_BOLD);
128     cellStyle.setFont(cellFont);
129     cellStyle.setFillForegroundColor(IndexedColors.LIGHT_BLUE.getIndex());
130     cellStyle.setFillPattern(CellStyle.SOLID_FOREGROUND);
131     return cellStyle;
132 }
133
134 // devuelve el estilo que tendrán las celdas con fórmula (color de fondo gris claro)
135 private CellStyle getEstiloCeldaConFormula() {
136     final CellStyle cellStyle = libro.createCellStyle();
137     cellStyle.setFillForegroundColor(IndexedColors.GREY_25_PERCENT.getIndex());
138     cellStyle.setFillPattern(CellStyle.SOLID_FOREGROUND);
139     return cellStyle;
140 }
141
142 // crea una nueva fila a continuación de la anterior
143 private Row getNuevaFila() {
144     return hojaTiemposPilotos.createRow(hojaTiemposPilotos.getPhysicalNumberOfRows());
145 }
146
147 }

```

Obsérvese que los métodos `getEstiloTitulo` y `getEstiloCeldaConFormula` crean los estilos de las celdas con los nombres de las columnas y las que contienen una fórmula.

El método encargado de añadir una fórmula a una celda es `setCellFormula` de la clase `Cell`, al que habrá que invocar por cada celda a la que se quiera adjuntar una fórmula. La fórmula encargada de sumar los tiempos del primer piloto será `SUM(B2:F2)` que indica que se va a sumar el contenido del rango de las columnas B a F de la fila 2 (fila donde están los tiempos del primer piloto). Para el segundo piloto, que está en la fila 3, la fórmula sería `SUM(B3:F3)`. Lo mismo se hace con las fórmulas de media (`AVERAGE`) y mejor tiempo (`MIN`).

6. Ejecutando el ejemplo.

Pues bien, vamos a ver si esto funciona o no... Para ello vamos a dar de alta los datos de dos pilotos y ver si nos genera correctamente el documento.

```

01 public static void main (String args[]) {
02     final GeneradorHojaCalculo generadorHojaCalculo = new GeneradorHojaCalculo();
03
04     final Piloto piloto1 = new Piloto("Fernando Alonso");
05     piloto1.getTiemposVueltas().add(131.78);
06     piloto1.getTiemposVueltas().add(129.95);
07     piloto1.getTiemposVueltas().add(128.16);
08     piloto1.getTiemposVueltas().add(125.91);
09     piloto1.getTiemposVueltas().add(130.44);
10
11     final Piloto piloto2 = new Piloto("Jaime Alguersuari");
12     piloto2.getTiemposVueltas().add(133.16);
13     piloto2.getTiemposVueltas().add(132.32);
14     piloto2.getTiemposVueltas().add(129.86);
15     piloto2.getTiemposVueltas().add(128.02);
16     piloto2.getTiemposVueltas().add(132.45);
17
18     generadorHojaCalculo.anadeTiemposPiloto(piloto1);
19     generadorHojaCalculo.anadeTiemposPiloto(piloto2);
20
21     try {
22         generadorHojaCalculo.generaDocumento();
23     } catch (IOException e) {
24         e.printStackTrace();
25     }
26
27 }

```

El proceso no da ningún error y nos genera un documento como se muestra a continuación.

The screenshot shows the LibreOffice Calc interface with the following data in the spreadsheet:

	A	B	C	D	E	F	G	H	I	J
1	Piloto	Vuelta 1	Vuelta 2	Vuelta 3	Vuelta 4	Vuelta 5	Tiempo total	Promedio	Mejor tiempo	
2	Fernando Alonso	131,78	129,95	128,16	125,91	130,44	646,24	129,248	125,91	
3	Jaime Alguersuari	133,16	132,32	129,86	128,02	132,45	655,81	131,162	128,02	
4										
5										
6										
7										
8										

The formula bar shows: `H2 = PROMEDIO(B2:F2)`

Si nos posicionamos sobre alguna de las celdas con fórmula podremos ver en la línea de entrada que se han generado correctamente. Si, en la propia hoja de cálculo, cambiamos alguno de los tiempos de los pilotos observaremos que se recalcula el tiempo total, la media y el mejor tiempo.

7. Referencias.

- <http://poi.apache.org/spreadsheet/quick-guide.html>

8. Conclusiones.

En este tutorial hemos visto que no tiene mucho misterio generar hojas de cálculo con fórmulas gracias a la ayuda de Apache POI. Además, por hacer un poco más completo el tutorial hemos añadido estilos a las celdas y ajustado en función de su contenido.

En nuestras aplicaciones puede ser muy interesante generar hojas de cálculo a modo de informes o para exportar algunos datos.

Espero que este tutorial os haya sido de ayuda. Un saludo.

Miguel Arlandy

marlandy@autentia.com

Anímate y coméntanos lo que pienses sobre este **TUTORIAL**:

Puedes opinar o comentar cualquier sugerencia que quieras comunicarnos sobre este tutorial; con tu ayuda, podemos ofrecerte un mejor servicio.

Enviar comentario

(Sólo para usuarios registrados)

» **Regístrate** y accede a esta y otras ventajas «

COMENTARIOS



Esta obra está licenciada bajo licencia [Creative Commons de Reconocimiento-No comercial-Sin obras derivadas 2.5](#)

IMPULSA

Impulsores

Comunidad

¿Ayuda?

sin clicks

0 personas han traído clicks a esta página

+ + + + + + + +

powered by [kamacracy](#)

Copyright 2003-2011 © All Rights Reserved | Texto legal y condiciones de uso | Banners | Powered by Autentia | Contacto

[W3C XHTML 1.0](#)
[W3C CSS](#)
[XML RSS](#)
[XML RTDM](#)