

## EJERCICIO GUIADO. JAVA. INFORMES DE BASE DE DATOS

### Conexión con Informes desde Java

Una vez creados los informes con iReport, tendremos que acceder a ellos desde nuestra aplicación Java.

En esta hoja guiada veremos los pasos básicos necesarios para mostrar un informe desde una aplicación Java.

Nota: En esta hoja guiada se usará el informe creado en la hoja anterior, llamado *serviciosagrupados*.


## EJERCICIO GUIADO Nº 1. CREACIÓN DEL INFORME COMPILADO

1. Entra en iReport y abra el informe *serviciosagrupados* (se encontrará en la carpeta Mis Documentos)

Los ficheros que contienen un informe son ficheros del tipo .jrxml. Así pues, en la carpeta Mis Documentos encontrará el fichero *serviciosagrupados.jrxml*, que es el que tiene que abrir.

Listado de Servicios		
Numero	Fecha	Cantidad
Tipo	\$F{Tipo}	
\$F{Numero}	\$F{Fecha}	\$F{Cantidad}
new Date()		
"Page " + \$V{PAGE_NUMBER} + " " + \$V		

(Nota: Si le echa un vistazo a la carpeta Mis Documentos, verá también un fichero llamado *serviciosagrupados.bak*. Este fichero es simplemente una copia de seguridad que se crea automáticamente al guardar el informe.)

2. Vamos a ver el aspecto del informe. Pulse el botón: 

Ya sabe que este botón, llamado *Ejecutar informe (usando conexión activa)*, permite visualizar el informe relleno con los datos de la base de datos, según la consulta SQL que haya usado.

## Listado de Servicios

Numero	Fecha	Cantidad
<b>Tipo Electricidad</b>		
8	01/08/2006	160,00
3	21/12/2005	130,00
<b>Tipo Fontanería</b>		
10	08/08/2006	265,00
5	03/06/2006	214,00
4	10/11/2006	250,00
2	22/05/2005	238,00
<b>Tipo Limpieza</b>		
11	09/08/2006	139,00
9	05/08/2006	250,00
7	20/07/2006	170,00
6	12/06/2006	265,00

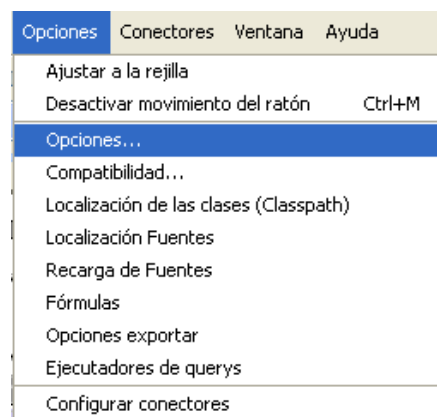
- Sin embargo, este botón no solo sirve para visualizar el informe final, sino que además actúa como compilador, generando un fichero del tipo .jasper.

Los ficheros .jasper son informes compilados, de forma que puedan ser usados en aplicaciones java.

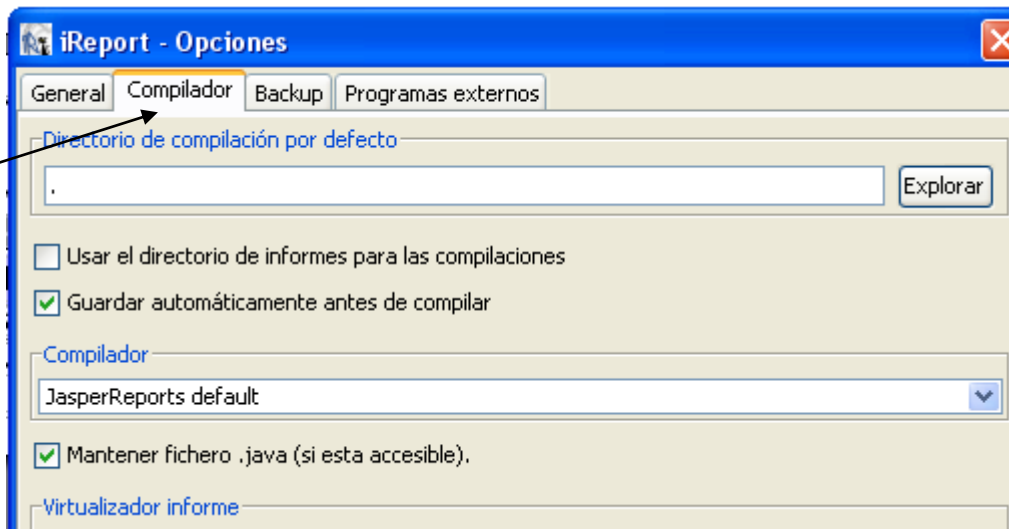
Cada vez que ha visualizado un informe, ha generado sin darse cuenta un fichero .jasper, con el nombre del informe que estaba visualizando. En nuestro ejemplo, acabamos de generar un fichero *serviciosagrupados.jasper*

- Los ficheros .jasper generados se suelen almacenar dentro de la carpeta del programa iReport, sin embargo, nosotros configuraremos el programa iReport para que los ficheros compilados se almacenen directamente en la carpeta Mis Documentos, de forma que estén más accesibles.

Para ello, active la opción *Opciones – Opciones*:

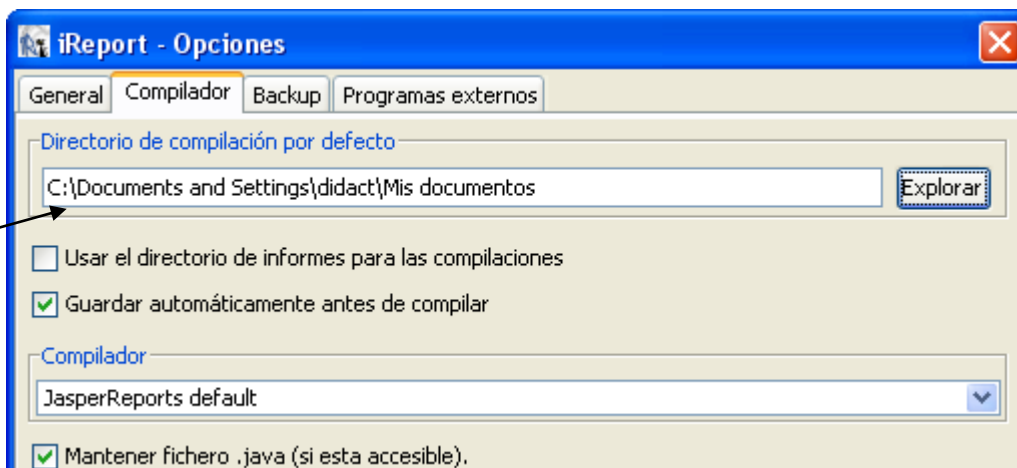


5. En el cuadro de opciones que aparece active la pestaña *Compilador*.



6. En el primer recuadro que aparece se tiene que indicar el directorio donde aparecerán los ficheros compilados. Para ello, pulse el botón *Explorar* y seleccione la carpeta *Mis Documentos*.

Si todo va bien, el resultado será parecido al siguiente:



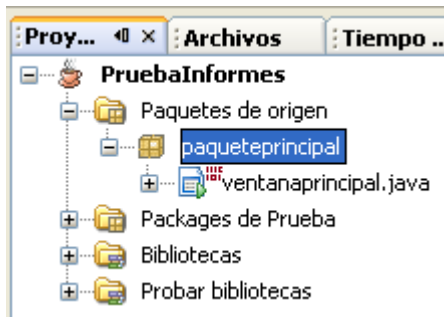
7. Pulse el botón *Grabar*. A partir de ahora cada vez que visualice un informe el fichero compilado se guardará en la carpeta *Mis Documentos*. Lo vamos a comprobar.

Vuelva a visualizar su informe: 

8. Acuda a la carpeta *Mis Documentos* y compruebe que en ella hay ahora un fichero *serviciosagrupados.jasper*. Este es el fichero del informe compilado. Este fichero será el que usará para visualizar informes desde java.

## EJERCICIO GUIADO Nº 2. CONEXIÓN CON UN INFORME DESDE JAVA

1. Entra en NetBeans y crea un nuevo proyecto Java. El proyecto se llamará *PruebaInformes* y contendrá un *paqueteprincipal*. Este paquete a su vez contendrá una *ventanaprincipal*.

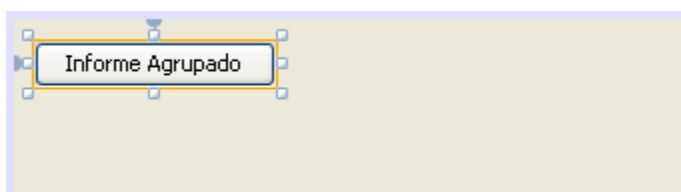


2. Crea la carpeta Base dentro de la carpeta del proyecto y copia en ella la base de datos MANEMPSA.
3. Prepara tu proyecto para que pueda acceder a la base de datos MANEMPSA (ya sabes, crear el objeto conexión, sentencia, etc...)
4. Para que nuestro programa sea capaz de presentar informes, crearemos una carpeta llamada *Informes* dentro de la carpeta del proyecto. Y dentro de dicha carpeta copiaremos los ficheros .jasper de los informes que queremos utilizar. En nuestro caso, copiaremos el fichero *serviciosagrupados.jasper* que creamos en el apartado anterior.

Como ve, es algo parecido a lo que hacemos con la base de datos.

Así pues crea la carpeta *Informes* dentro de la carpeta de proyecto y copia el fichero *serviciosagrupados.jasper* dentro de ella.

5. Crea un botón en la ventana de tu proyecto. El botón contendrá el texto "Informe Agrupado" y su nombre será *btnInformeAgrupado*.



6. El objetivo es hacer que aparezca el informe *serviciosagrupados* cuando se pulse este botón. Para ello, programe lo siguiente en el *actionPerformed* del botón:

```
private void btnInformeAgrupadoActionPerformed(java.awt.event.ActionEvent evt) {  
    TODO: Agregue su código aquí:  
    try {  
        String rutaInforme = "informes\\serviciosagrupados.jasper";  
        JasperPrint informe = JasperFillManager.fillReport(rutaInforme,null,conexion);  
        JasperViewer ventanavisor = new JasperViewer(informe,false);  
        ventanavisor.setTitle("INFORME DE SERVICIOS");  
        ventanavisor.setVisible(true);  
    } catch (Exception e) {  
        JOptionPane.showMessageDialog(null,"Error al mostrar el informe"+e);  
    }  
}
```

Este código le muestra muchos errores, no se preocupe ahora de ellos. En los siguientes pasos se corregirán. De momento lo que se va a hacer es explicar el código de este evento:

- Lo primero que observará es como se usa una variable *rutaInforme* para almacenar el camino del informe que queremos mostrar. Observa que dicho informe se especifica con el fichero *.jasper* generado con iReport, y que este fichero se encuentra dentro de la subcarpeta *informes*.
- A continuación viene una línea donde se crea un objeto informe del tipo *JasperPrint*:

```
JasperPrint informe = JasperFillManager.fillReport(rutainforme,null,conexión);
```

- Esta línea es la que genera el informe que se va a mostrar. El informe en java es un objeto del tipo *JasperPrint* y es creado a través de otro objeto del tipo *JasperFillManager*.
- Es interesante que observes el uso de un método llamado *fillReport* (rellenar informe) que se encarga de rellenar el informe con datos.
- Este método necesita tres parámetros:
  - El sitio donde se encuentra el fichero del informe.
  - Los parámetros (Null en nuestro ejemplo. Los parámetros se estudiarán en próximas hojas guiadas)
  - Y el objeto *conexión*, que le permite a java acceder a la base de datos para recoger los datos que se presentarán en el informe.
- Una vez creado el objeto informe, solo tenemos que presentarlo en pantalla. Los informes se presentan a través de un objeto del tipo *JasperViewer*. Los objetos *JasperViewer* son ventanas donde se muestran los informes.

```
JasperViewer ventanavisor = new JasperViewer(informe,false);
```

- Esta línea de código crea una ventana llamada *ventanavisor* donde se muestra el informe *informe*. El parámetro *false* indica que al cerrarse la ventana del informe no debe acabar el programa. Si este parámetro fuera *true*, al cerrar la ventana del informe se cerraría el programa entero (esto no interesa).

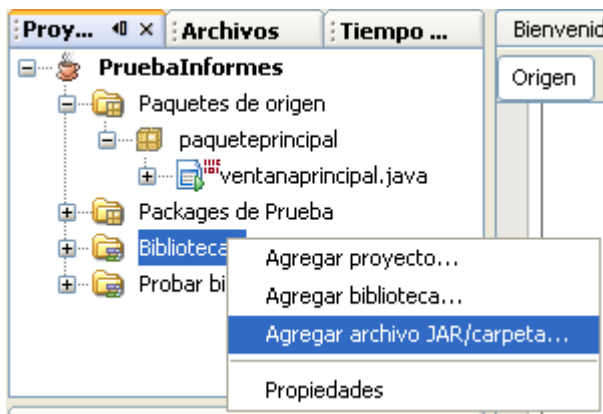
- El objeto *ventanavisor* es una ventana normal y corriente y puede ser visualizada con *setVisible*. Por otro lado, se le asigna el título “Informe de servicios” con el método *setTitle*.
  - Todo este código debe estar rodeado de un *try...catch* para evitar errores inesperados.
7. Todos los errores del código se producen debido a que es necesario indicar los *imports* correspondientes a las clases *JasperPrint*, *JasperFillManager*, *JasperViewer* que participan en el código.

Un *import* es una línea de código que le dice a java en qué librería debe encontrar la clase correspondiente.

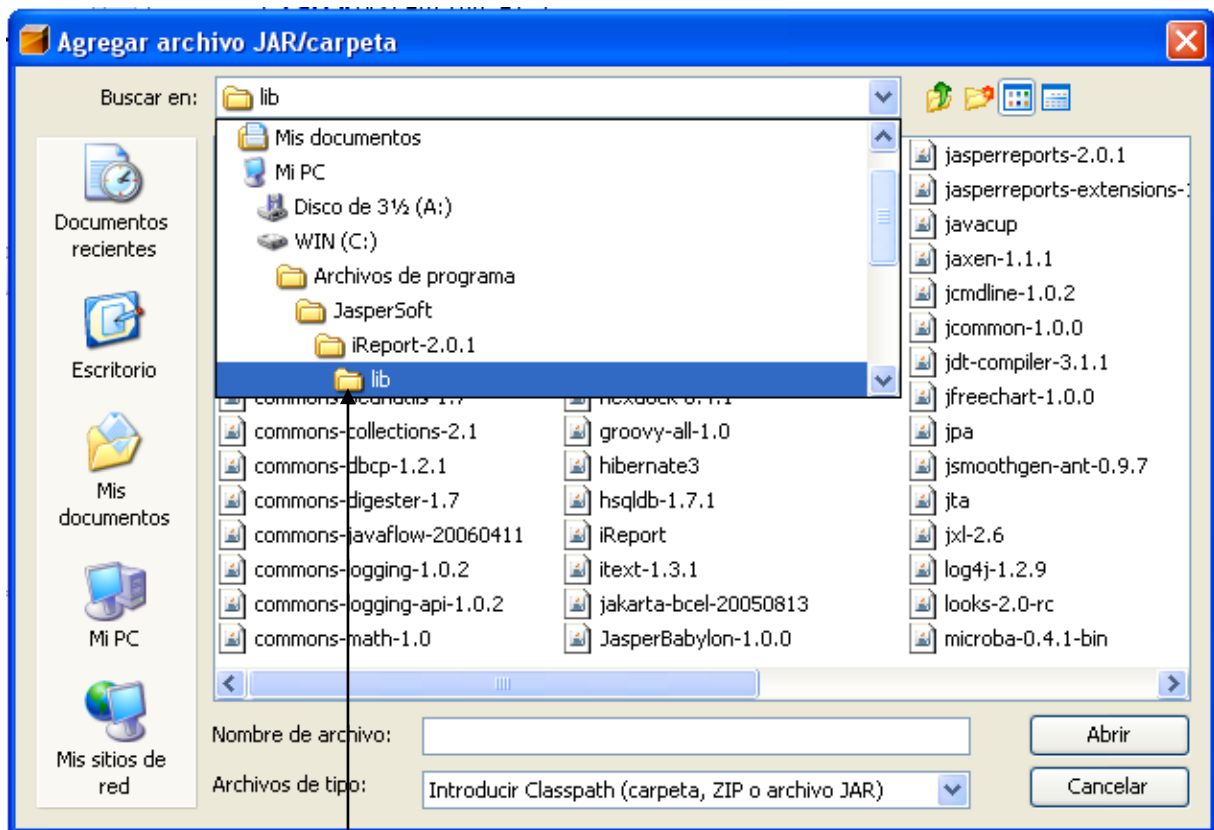
Normalmente, al hacer clic sobre la línea que da el error, aparece el icono de una bombilla pequeña en la parte izquierda de la ventana. Al hacer clic sobre esta bombilla se nos informa de la librería que hace falta *importar*. Ya sabe que basta hacer clic sobre esta librería para que se añada la instrucción *import* correspondiente de forma automática.

Sin embargo, si prueba esto mismo con las líneas erróneas del código anterior, verá que no aparece ninguna bombilla. Esto es debido a que las librerías donde se encuentran las clases *JasperPrint*, *JasperFillManager* y *JasperViewer* no vienen incluidas en NetBeans, ya que son clases pertenecientes al programa iReport. Así pues, NetBeans no sabe donde encontrar estas clases.

8. Para evitar este error, es necesario incluir en el proyecto una serie de librerías propias del programa iReport. Esto se hace de la siguiente forma:
9. Haga clic con el botón derecho del ratón sobre el apartado *Bibliotecas* de su proyecto y luego active la opción *Agregar archivo JAR/carpeta*.



10. Debe acudir a la carpeta del programa iReport para buscar en ella las bibliotecas necesarias. Concretamente, la carpeta donde se encuentran estas bibliotecas es la siguiente: *Archivos de programa – JasperSoft – iReport-2.0.1 – lib*

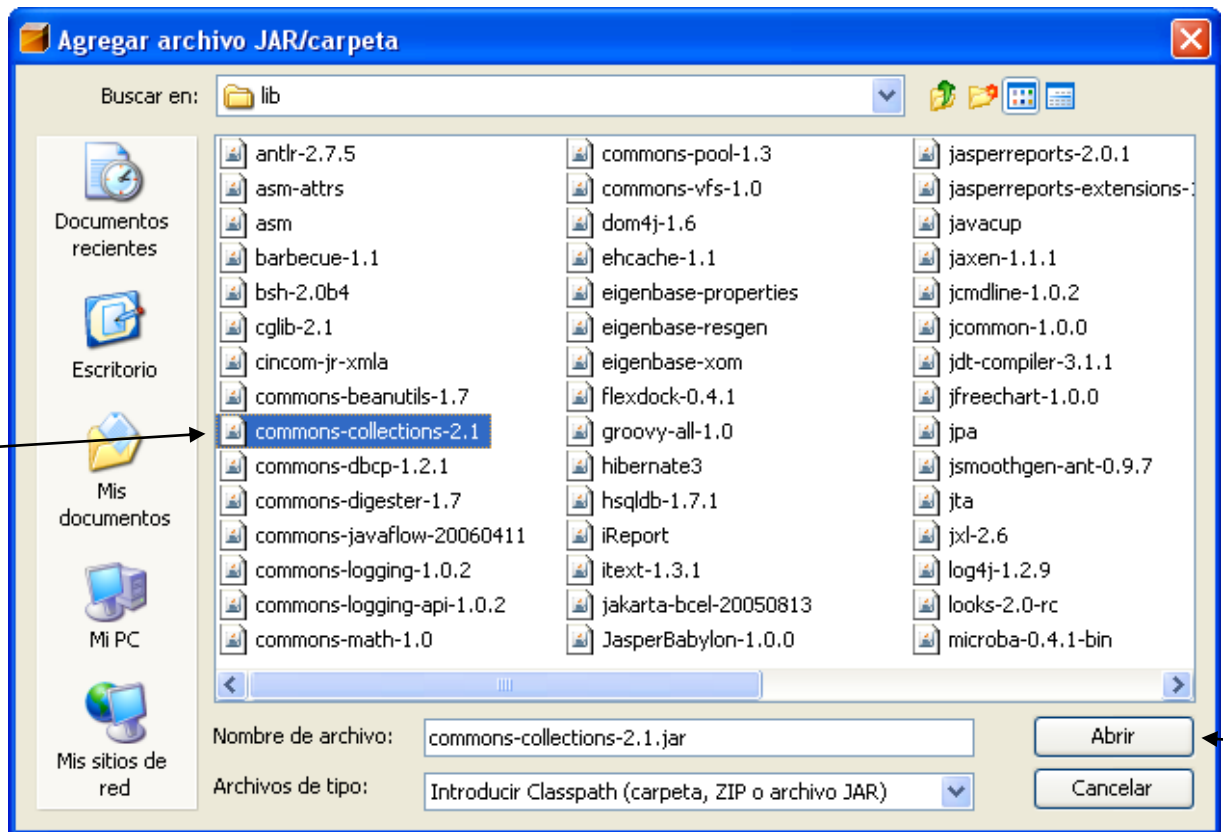


Archivos de programa – JasperSoft – iReport-2.0.1 – lib

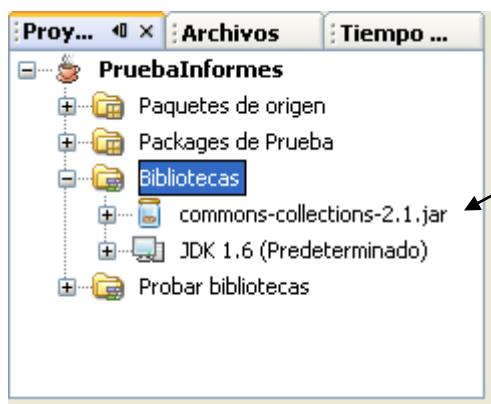
9. Los ficheros que contienen las librerías necesarias para este programa son lo siguientes: *commons-collections-2.1*, *commons-logging-1.0.2* y *jasperreports-2.0.1*

Empezaremos añadiendo el fichero *commons-collections-2.1*. Selecciónelo y active Abrir:

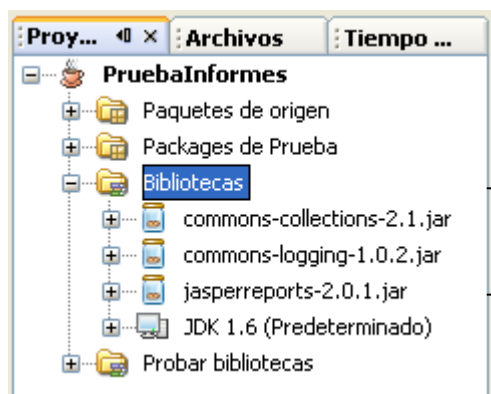




10. Observará que se ha agregado el fichero a la zona de bibliotecas:

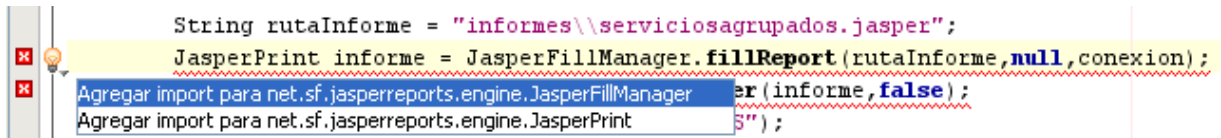


11. Repita el proceso para agregar los otros dos ficheros necesarios. Al final, su zona de Bibliotecas debe quedar así (observe los tres ficheros añadidos):



12. Gracias a la inclusión de estos ficheros, nuestro proyecto ya dispone de las librerías de clases que contienen a las clases que necesitamos, que recuerda que son: *JasperPrint*, *JasperFillManager*, *JasperViewer*

Si ahora haces clic sobre la primera línea que da error, verá como ahora NetBeans ya sabe el import que debe añadirse, y por tanto aparecerá la bombilla. Pulse sobre la bombilla:



```
String rutaInforme = "informes\\serviciosagrupados.jasper";
JasperPrint informe = JasperFillManager.fillReport(rutaInforme, null, conexion);
Agrega import para net.sf.jasperreports.engine.JasperFillManager er(informe, false);
Agrega import para net.sf.jasperreports.engine.JasperPrint S");
```

13. Tiene que agregar el import para la clase *JasperFillManager*:

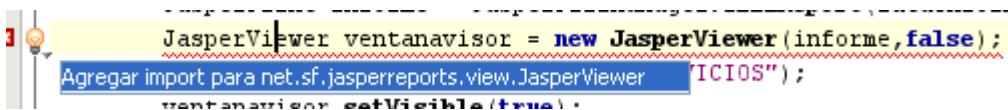
```
net.sf.jasperreports.engine.JasperFillManager
```

14. Ahora agregue el import para la clase *JasperPrint*:

```
net.sf.jasperreports.engine.JasperPrint
```

15. Y finalmente agregue el import para la clase *JasperViewer*:

```
net.sf.jasperreports.view.JasperViewer
```



```

JasperViewer ventanavisor = new JasperViewer(informe, false);
Agrega import para net.sf.jasperreports.view.JasperViewer ICIOS");
ventanavisor.setVisible(true);
```

16. Los errores han desaparecido. Ya puede ejecutar el programa.

17. Si pulsa sobre el botón *Informe Agrupado* aparecerá el informe.

El informe aparece en una ventana (el visor de informes – objeto *JasperViewer*) que tiene una pequeña barra de herramientas. En esta barra podrá por ejemplo imprimir el informe, o cambiar su zoom, entre otras cosas:

INFORME DE SERVICIOS

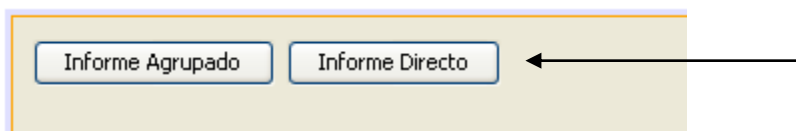
## Listado de Servicios

Numero	Fecha	Cantidad
<b>Tipo Electricidad</b>		
8	01/08/2006	160,00
3	21/12/2005	130,00
<b>Tipo Fontanería</b>		
10	08/08/2006	265,00
5	03/06/2006	214,00
4	10/11/2006	250,00
2	22/05/2005	238,00
<b>Tipo Limpieza</b>		
11	09/08/2006	139,00

Ventana visor de informes (JasperViewer)  
Herramientas: guardar, imprimir, zoom, etc...

18. Tal como se ha programado este botón, al pulsarse se visualiza el informe antes de que el usuario decida imprimirlo o no. En algunos casos, puede ser interesante que el informe se imprima directamente sin que se visualice antes.

Veamos como hacer esto. Primero añada otro botón *btnImprimirDirectamente* a su ventana:



19. Ahora programe dentro de él lo siguiente:

```
private void btnImprimirDirectamenteActionPerformed(java.awt.event.ActionEvent evt) {
    TODO: Agregue su código aquí:
    try {
        String rutaInforme = "informes\\serviciosagrupados.jasper";
        JasperPrint informe = JasperFillManager.fillReport(rutaInforme, null, conexion);
        JasperPrintManager.printReport(informe, true);
    } catch (Exception e) {
        JOptionPane.showMessageDialog(null, "Error al mostrar el informe"+e);
    }
}
```

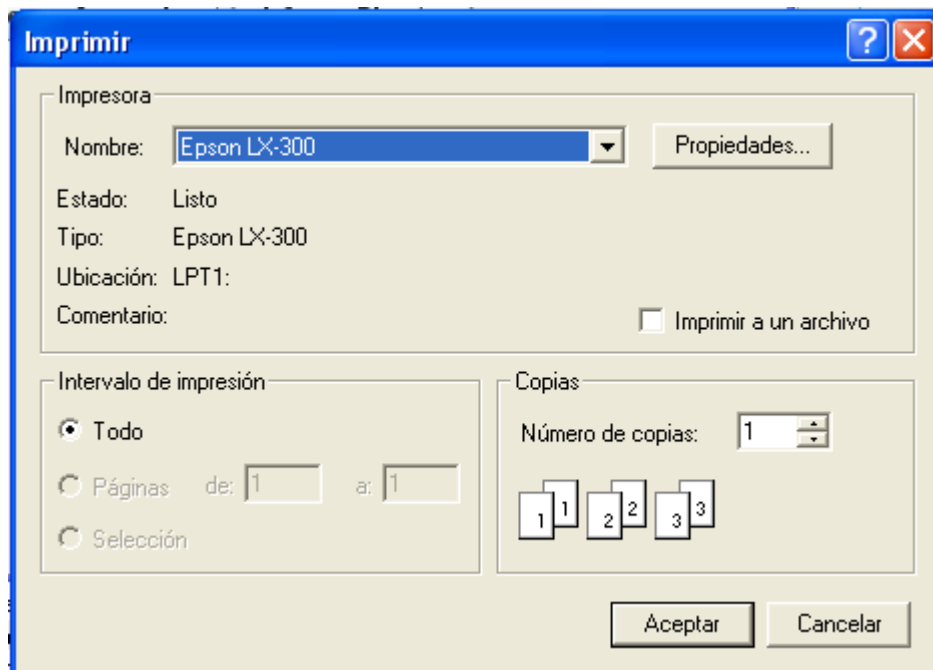
El código es similar al anterior, solo que se elimina todo uso del objeto *JasperViewer* (ventana visualizadora) ya que el informe ya no debe salir en pantalla.

Además se añade una nueva línea:

```
JasperPrintManager.printReport(informe,true);
```

Esta línea usa un objeto llamado *JasperPrintManager* que se encarga de imprimir el informe *informe* usando el método *printReport*.

Si el segundo parámetro tiene el valor *true*, entonces significa que aparecerá el cuadro de diálogo de impresión:



Si estuviera a *false*, este cuadro de diálogo no aparecería y el informe se enviaría directamente a la impresora configurada por defecto en el ordenador.

En nuestro caso, al tener el valor *true*, este cuadro sí aparecerá.

20. Ejecute el programa y pruebe el nuevo botón. No hace falta que imprima el informe.

## **CONCLUSIÓN**

**Para poder imprimir un informe creado con iReport desde una aplicación Java es necesario agregar ciertas librerías pertenecientes al programa iReport:**

***commons-collections-2.1, commons-logging-1.0.2 y JasperReports-2.0.1***

**Una vez agregadas estas librerías, el proceso para imprimir un informe es el siguiente:**

- **Crear un objeto del tipo JasperPrint (el informe)**
- **Crear un objeto visor (JasperViewer) que permita visualizar el informe.**
- **O bien imprimir directamente el informe con un objeto JasperPrintManager.**