EJERCICIO GUIADO. JAVA. INFORMES CON PARÁMETROS

Conexión con Informes desde Java: Parámetros

En la hoja guiada anterior se conectó una aplicación Java con un informe realizado con iReport. La aplicación contaba con un botón que al ser pulsado mostraba el informe.

Hay que tener en cuenta que los datos que se muestran en el informe dependen del contenido de la base de datos. Es decir, si se añaden nuevos registros a las tablas, estos registros aparecerán en el informe (no será necesario modificarlo de alguna forma)

Sin embargo, esta forma de trabajar puede ser poco interesante, ya que el usuario no tiene ningún control sobre el informe que se muestra. En la hoja guiada anterior el informe mostraba todos los servicios agrupados por tipo, pero, y si el usuario quisiera en un determinado momento que solo se mostraran los servicios anteriores a una fecha, o que se mostraran los servicios que costaran más de una cantidad, etc...

El problema está en que la consulta SQL del informe es una consulta fija, y siempre sacará los mismos datos. El usuario de nuestro programa no tiene forma de modificar esa consulta para cambiar los datos deben aparecer en el informe.

Para poder hacer esto, es necesario usar *parámetros* en la consulta SQL del informe. Un parámetro se podría definir como dato de la consulta que no está definido aún. Es necesario definir ese dato antes de ejecutar la consulta y extraer los datos que se mostrarán.

Puede ser el usuario de nuestro programa el que aporte ese dato que le falta a la consulta, de manera que así tenga cierto control sobre la información que aparece en el listado.

En esta hoja se verá un ejercicio paso a paso de uso de parámetros en la consulta de informe, y de esta manera entender la utilidad de esta característica.

EJERCICIO GUIADO Nº 1. CREACIÓN DE UN INFORME CON PARÁMETROS

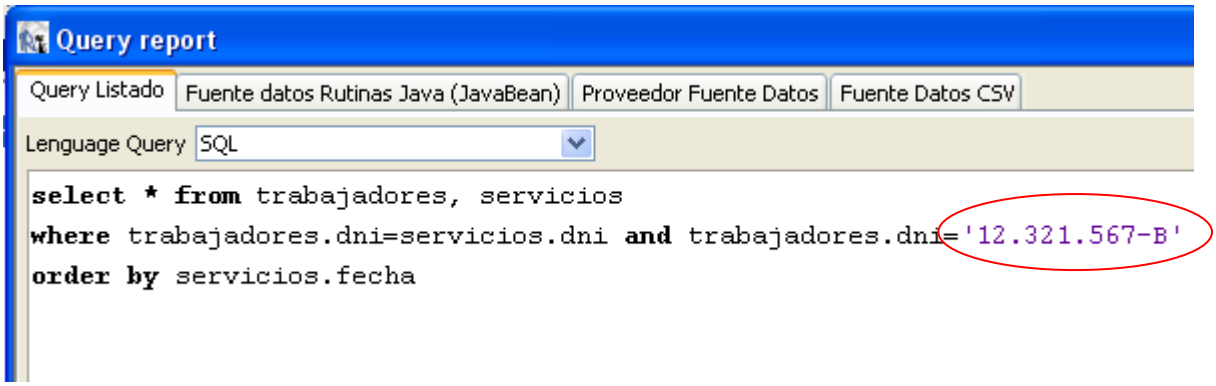
1. Entra en iReport y abre el informe llamado *serviciosjuan*, realizado en una de las últimas hojas guiadas. Este informe, concretamente, fue construido desde cero y mostraba los servicios del trabajador *Juan Pérez*:

2. Visualízalo para recordar su aspecto:

3. Este informe tiene un gran diseño pero está muy limitado, ya que solo es capaz de mostrar el listado de servicios realizados por el trabajador *Juan Pérez* (DNI 12.321.567-B) Si quisiéramos sacar el listado de servicios de otro trabajador, este informe no nos serviría.

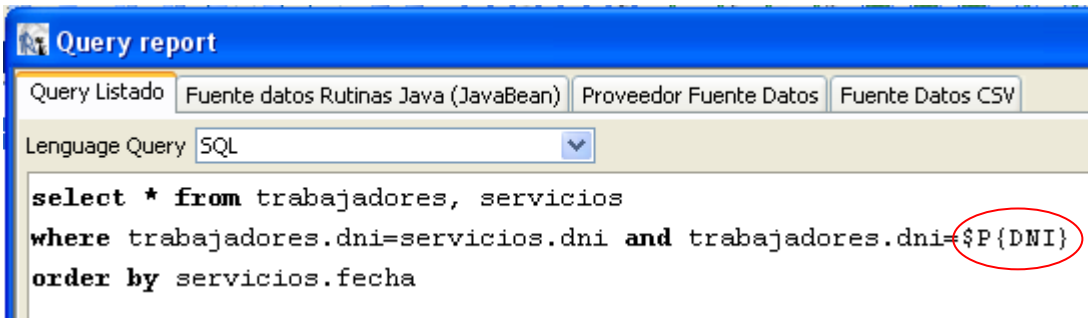
   Es necesario hacer que este informe sea más versátil, que sea capaz de mostrar el listado de servicios de cualquier trabajador. Para ello, será necesario crear un parámetro en la consulta SQL.

4. Acceda a la consulta del informe a través de la opción *Datos – Consulta de informe* y obsérvela:



   La consulta SQL extrae información de aquellos servicios que hayan sido realizados por el trabajador con DNI 12.321.567-B.

5. Cambie la consulta para que quede así:



   Lo que acaba de añadir es un parámetro llamado DNI. Observe su estructura:

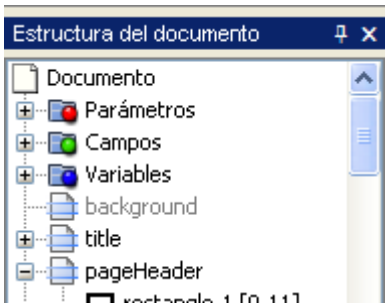
   $P{nombredelparámetro}

   En nuestro caso el nombre que le hemos dado al parámetro es “DNI”, por lo tanto el parámetro es:

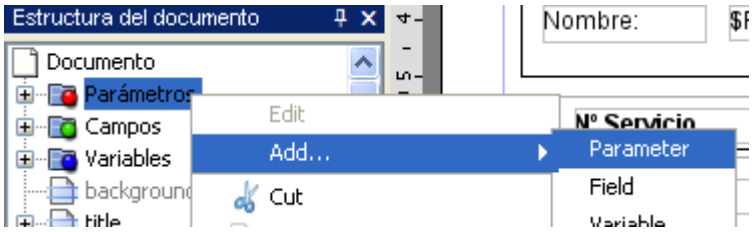
   $P{DNI}

   Este parámetro indica que el DNI no está definido aún. Será el usuario el que introduzca el DNI según el cual habrá que consultar en la base de datos.

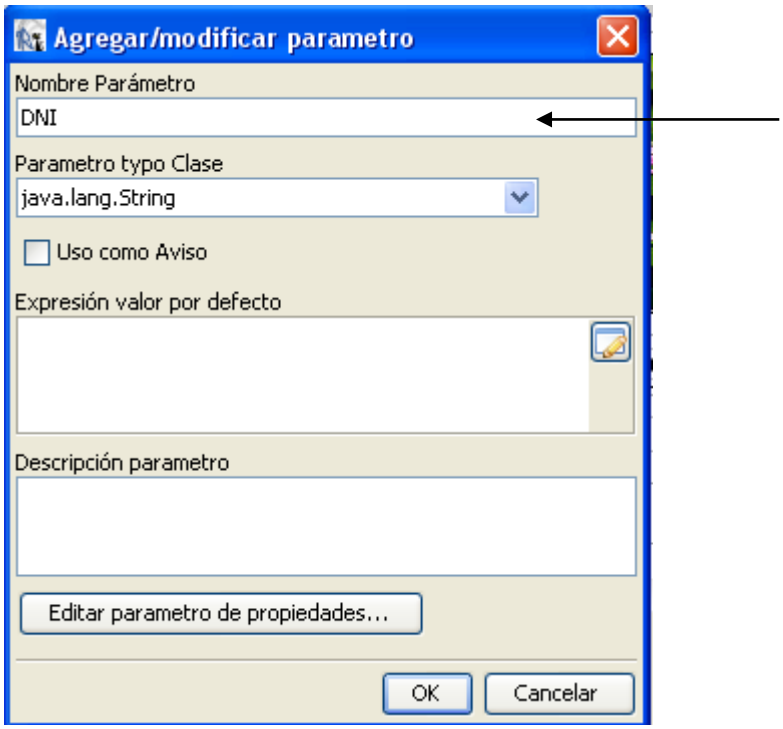
6. Pulse OK. Cada vez que use un parámetro hay que definirlo en el programa iReport. Esto se hace en la zona de campos. Observa que tenemos también una zona de parámetros:



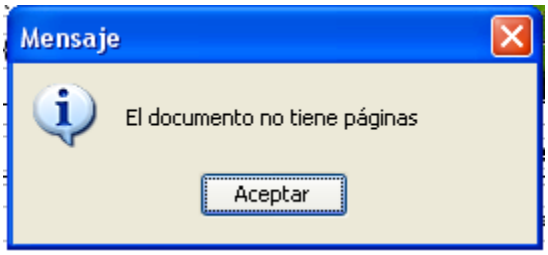
7. En la carpeta de parámetros haz clic con el botón derecho del ratón e indica la opción *Add – Parameter* para añadir un nuevo parámetro:



8. En la ventana que aparece debes indicar el nombre que le has dado al parámetro. En nuestro caso, el nombre asignado es “DNI”:



9. Ahora ya tiene definido un parámetro que se corresponde con el DNI del trabajador. El usuario introducirá ese DNI y será sustituido en la consulta SQL. Si intenta visualizar el informe, no podrá ver nada, ya que el parámetro no tiene ahora ningún valor. Al visualizar el informe aparecerá el siguiente mensaje:



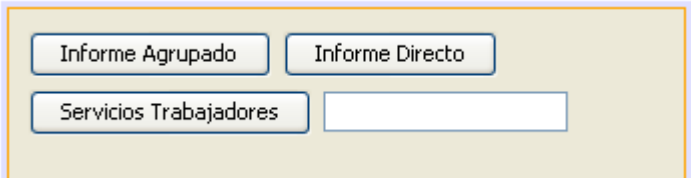
10. En cualquier caso, se tiene que haber generado el fichero .jasper correspondiente. Si el iReport sigue configurado tal como se hizo en la hoja guiada anterior, dicho fichero se tiene que haber guardado en Mis Documentos con el nombre *serviciosjuan.jasper*. Este fichero lo usará en el siguiente ejercicio.

11. Cierre iReport, guarde los cambios y pase al siguiente ejercicio.

EJERCICIO GUIADO Nº 2. USO DE PARÁMETROS DE UN INFORME DESDE JAVA

1. Abra la aplicación Java que realizó en la hoja guiada anterior. Añada en su ventana un nuevo botón y un cuadro de texto.

   El botón se llamará *btnServiciosTrabajador* y el cuadro de texto se llamará *txtDNI*.



2. El objetivo es hacer que el usuario introduzca un DNI en el cuadro de texto DNI y que al pulsar el botón *Servicios del Trabajador* aparezca un informe con el listado de servicios del trabajador con el DNI introducido.

   Para hacer esto es necesario que el DNI introducido en el cuadro de texto *txtDNI* se traslade directamente al informe creado anteriormente y se coloque en la posición del parámetro. Entonces la consulta SQL del informe se completará y se podrá rellenar el informe con los datos correctos.

3. De momento copie el fichero *serviciosjuan.jasper* dentro de la carpeta *informes* de su carpeta de proyecto.

4. Nota Importante: Recuerde que el informe contiene una imagen proporcionada por el fichero *nenúfares.jpg*. Para que el informe funcione correctamente es necesario que copie este fichero directamente en la carpeta de su proyecto (no dentro de la carpeta *informes*)

5. Ahora programe el siguiente código en el actionPerformed del botón *Servicios Trabajadores*.

```
private void btnServiciosTrabajadoresActionPerformed(java.awt.event.ActionEvent evt) {
TODO: Agrege su codigo aqui:
    try {
    String rutaInforme = "informes\\serviciosjuan.jasper";
    Map parametros = new HashMap();
    parametros.put("DNI",txtDNI.getText());
    JasperPrint informe = JasperFillManager.fillReport(rutaInforme,
                                                       parametros,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");
   }
}
```

Estudiemos con detenimiento este código.

En él verá dos líneas nuevas en las que se definen los valores de los parámetros:

```
Map parametros = new HashMap();
parametros.put("DNI",txtDNI.getText());
```

En la primera de estas dos líneas se define un objeto de tipo *Map*. Este objeto contendrá el nombre de cada parámetro que usará y el valor de dicho parámetro.

Observe la segunda línea, en ella se define el parámetro llamado "DNI" y se le asigna a él el valor del cuadro de texto del DNI, es decir, el DNI que haya escrito el usuario.

Si existieran más parámetros habría que definirlos de la misma forma, por ejemplo:

```
parametros.put("sueldo",txtSueldo.getText());
parametros.put("codigo","A-54");
```

En este código ficticio se crea un parámetro *sueldo* al que se le asigna el valor de un cuadro de texto *txtSueldo*. Y se crea un parámetro *codigo* al que se le asigna el valor "A-54".

En nuestro caso solo necesitamos usar un parámetro llamado DNI. Hay que tener en cuenta que el nombre del parámetro debe ser el mismo que el nombre usado en iReport.

La siguiente línea es la que crea el objeto *informe* (JasperPrint) Esta línea crea el informe a partir del fichero de informe compilado *serviciosjuan.jasper*, a partir de la base de datos (representada por el objeto *conexión*) y lo más interesante ahora es que se indica el conjunto de parámetros a través del objeto *parametro* creado antes.

```
JasperPrint informe = JasperFillManager.fillReport(rutaInforme,parametros,conexion);
```

Cuando el informe que se va a mostrar no tiene parámetros, entonces se usa el valor null, en caso contrario, se usa el objeto *parámetro (Map)*

El resto del código es igual que los anteriores, se crea el visor de informe conteniendo el informe y se muestra en pantalla.

6. Ejecute el programa y pruebe a introducir un DNI existente en el cuadro de texto del DNI. Si escribió correctamente el DNI, podrá ver el listado de servicios realizados por el trabajador con dicho DNI.

   Como puede observar, ahora ya puede visualizar los datos de los servicios de cualquier trabajador y no solo del trabajador Juan Pérez.

   El uso de parámetros en los informes da potencia y versatilidad a las aplicaciones, y es algo muy usado.

Trabajos realizados por Ana Ruiz

NOTA

En los ejemplos guiados que se acaban de proponer, se usa una consulta SQL de informe que contiene un parámetro equivalente a un dato. Concretamente, a un DNI. Este dato forma parte de una condición. Observa la consulta SELECT del informe que hemos usado:

parámetro

```
select * from trabajadores, servicios
where trabajadores.dni=servicios.dni and trabajadores.dni=$P{DNI}
order by servicios.fecha
```

Condición

En este caso, el DNI será proporcionado por la aplicación java y la condición se completará.

Sin embargo, hay que indicar que también se pueden crear parámetros que sustituyan a una condición completa, en vez de a un solo dato. Observa el siguiente ejemplo:

```
select * from trabajadores, servicios
where trabajadores.dni=servicios.dni and $P!{COND}
order by servicios.fecha
```

En este caso, el parámetro se llama *COND* y no sustituye a un simple dato, sino que sustituye a toda la condición. En este caso, la aplicación java mandará una cadena que tenga forma de condición SQL, como por ejemplo: "`trabajadores.sueldo > 200`", que será reemplazada en el lugar del parámetro.

Para que lo vea claro, el código a usar en la aplicación java para dar valor al parámetro podría ser como sigue:

```
Map parametros = new HashMap();
parametros.put("COND", "trabajadores.sueldo>200");
```

Observa que el valor del parámetro es una condición entera.

Cuando se quiera usar un parámetro que represente a un dato, se usará la siguiente sintaxis:

$P{nombreparámetro}

Cuando se quiera usar un parámetro que represente a una condición, se usará la siguiente sintaxis:

$P!{nombreparámetro}

(Observa el uso de la admiración !)

CONCLUSIÓN

Cuando un informe tiene una instrucción SQL invariable, siempre mostrará los mismos datos.

Puede resultar interesante que parte de la instrucción SQL del informe pueda variar, de forma que se pueda extraer distinta información según el dato que varía. Estos datos variantes se denominan *parámetros*.

Un parámetro puede referirse a un dato concreto, o a una condición.

Si el parámetro se refiere a un dato concreto, entonces tiene la siguiente forma:

$P{nombre}

Si el parámetro se refiere a una condición, entonces tiene la siguiente forma:

$P!{nombre}

El parámetro debe ser sustituido por un valor que es proporcionado desde la aplicación java, y es entonces cuando la instrucción SQL se completa y se ejecuta.

En la aplicación java es necesario crear un objeto de tipo Map que contenga los distintos parámetros con los valores de cada uno para luego enviar este objeto al informe que se quiere mostrar.

El uso de parámetros en informes hace que estos sean mucho más versátiles y que el programa tenga más posibilidades.