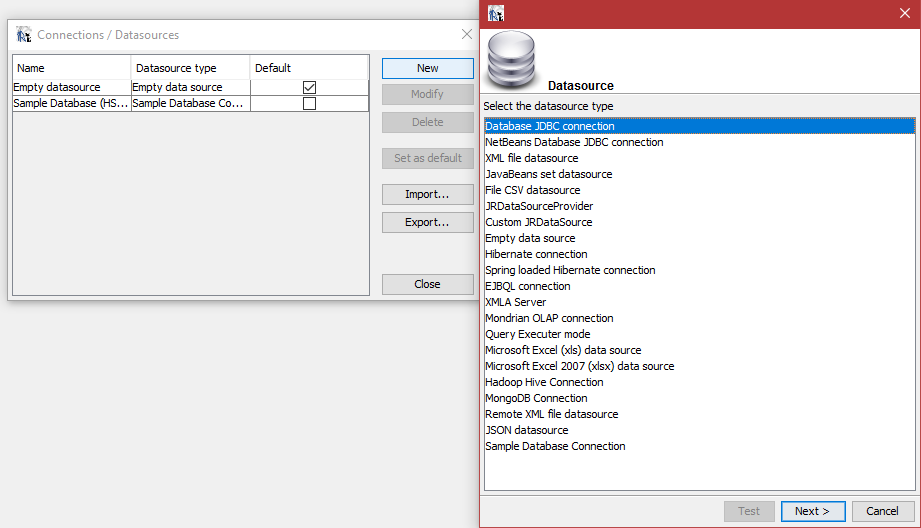
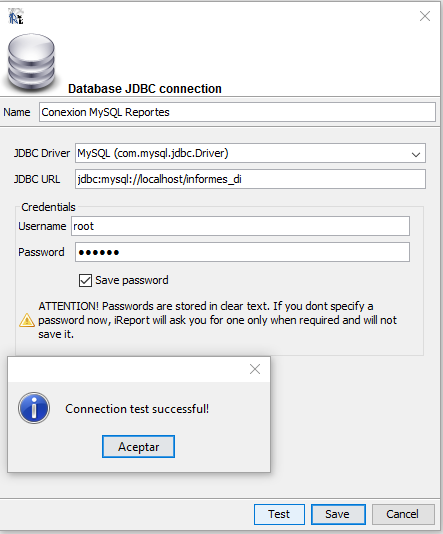
**iReport Reporte 1**

**1º Creamos una base de datos**

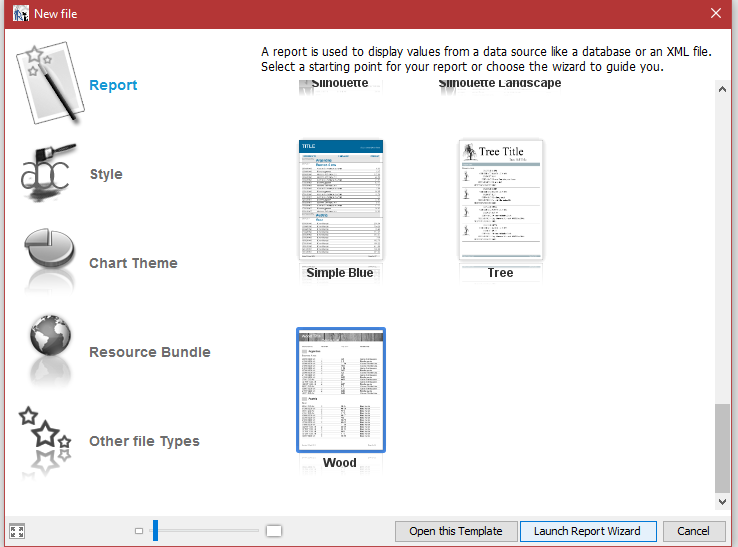
**2º Creamos una nueva conexión en el iReport**



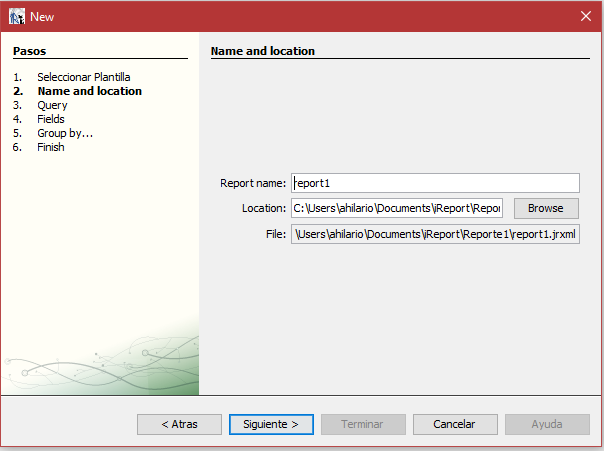


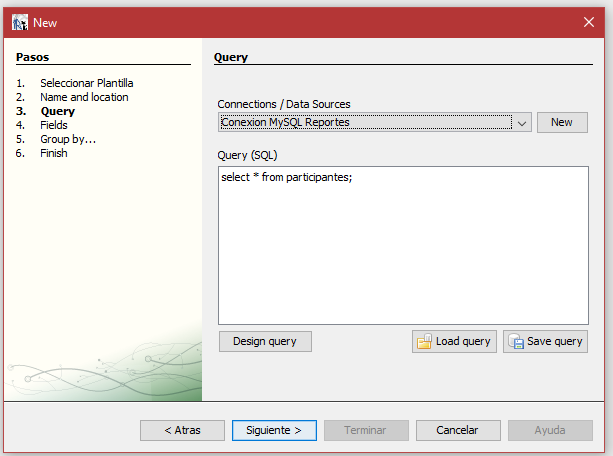
**Contraseña: campus**

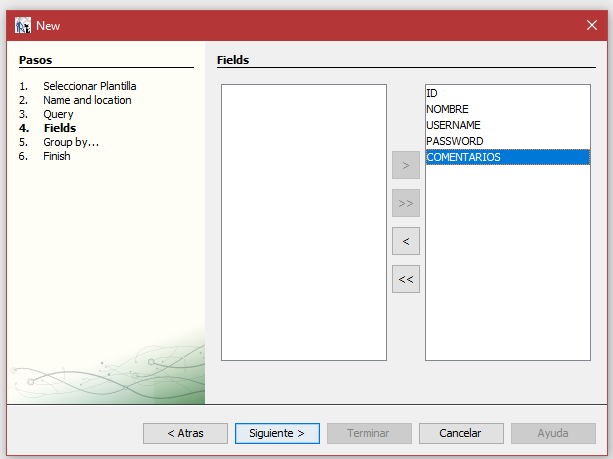
**3º Creamos un new File**

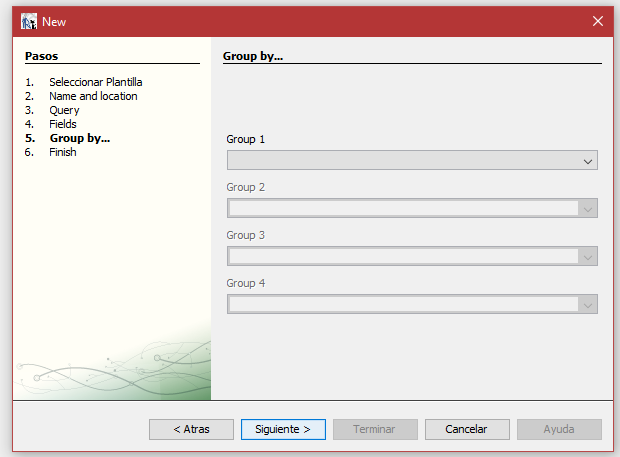


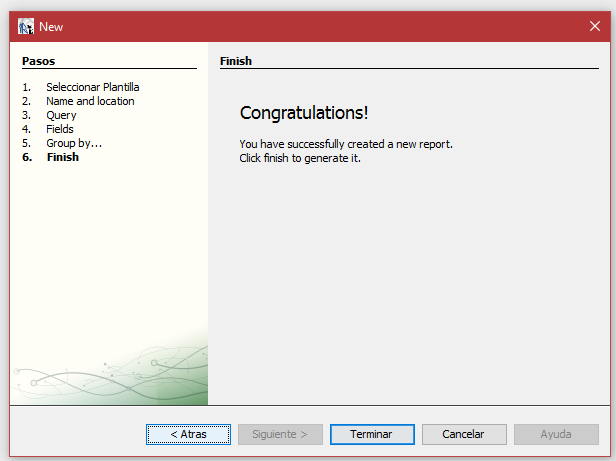
**Wood y “Launch Report Wizard”**



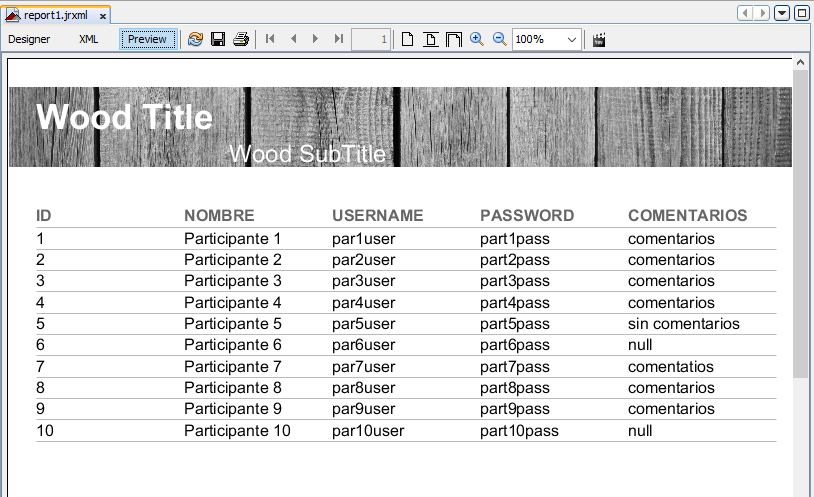








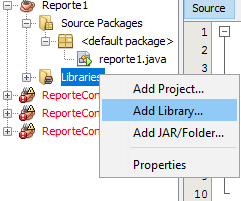
**Terminar y dar a Preview:**

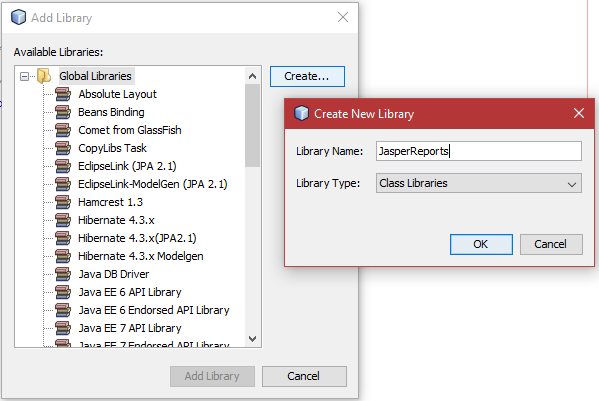


**¡Ahora vamos a crear Reportes en una aplicación Java!!**

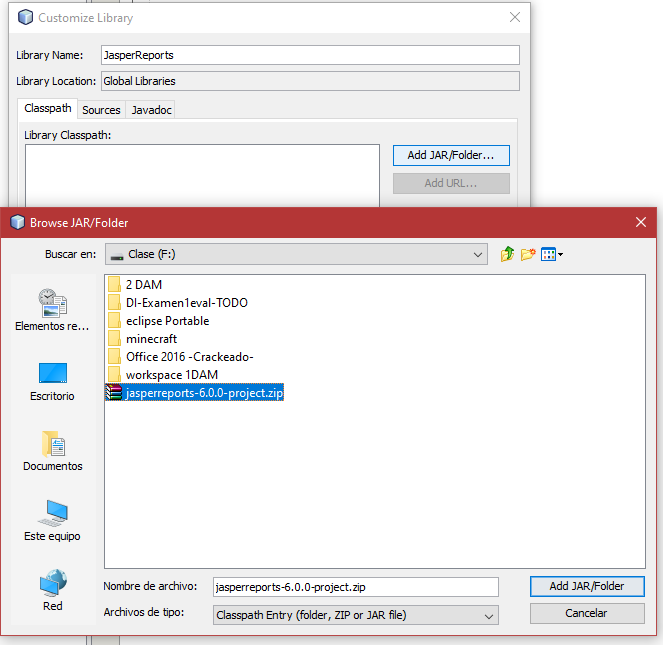
**File 🡪 new Project 🡪 Java Application**

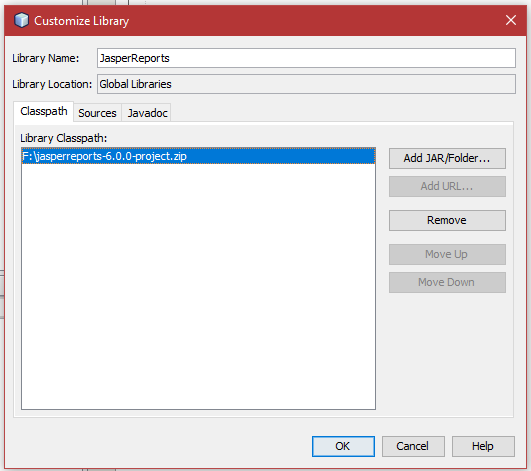
**Y añadimos las librerías necesarias:**

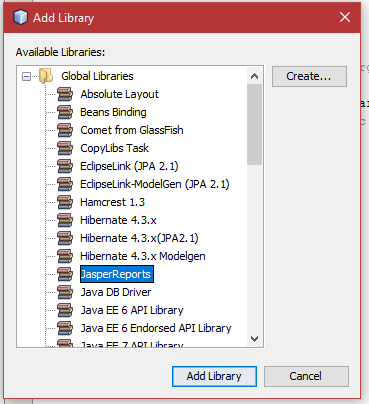


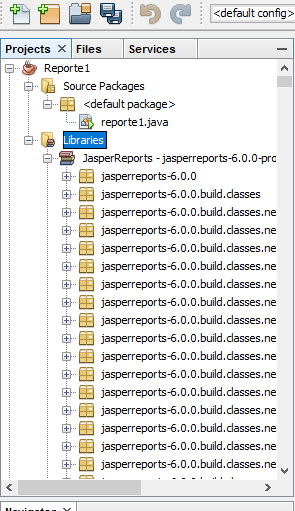


**Clicamos en “Create” y ponemos nombre**



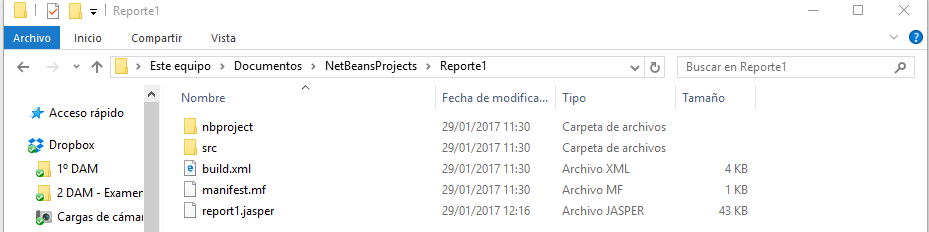






**(Si no va asi, descomprimir .rar y compiar librerías de las carpetas lib y dist)**

**El archivo que se genero en el iReport “.jasper” cuando lo dimos a preview, lo copiamos y lo pegamos en nuestro proyecto de NetBeans.**



**\* También copiar el .wood**

**Reporte 1 Codigo:**

public static void main(String[] args) {

try {

Class.forName("com.mysql.jdbc.Driver");

Connection conexion =

(Connection) DriverManager.getConnection("jdbc:mysql://localhost:3306/informes\_di",

"root", "campus");

JasperReport reporte = (JasperReport) JRLoader.loadObjectFromFile("report1.jasper");

JasperPrint jasperPrint = JasperFillManager.fillReport(reporte, null, conexion);

//Para PDF

Exporter exporter;

exporter = new JRPdfExporter();

// new JRPdfExporter(); // o También así: Exporter exporter = new JRPdfExporter();

exporter.setExporterInput(new SimpleExporterInput(jasperPrint));

exporter.setExporterOutput(new SimpleOutputStreamExporterOutput(new

java.io.File("reporteSesion1PDF.pdf")));

SimplePdfExporterConfiguration configuration = new

SimplePdfExporterConfiguration();

exporter.setConfiguration(configuration);

exporter.exportReport();

/\*

// Para HTML

Exporter exporterHTML = new HtmlExporter();

exporterHTML.setExporterInput(new SimpleExporterInput(jasperPrint));

exporterHTML.setExporterOutput(new SimpleHtmlExporterOutput(new

java.io.File("reporteSesion1HTML.html")));

SimpleHtmlExporterConfiguration configuration = new

SimpleHtmlExporterConfiguration();

exporter.setConfiguration(configuration);

exporter.exportReport();

\*/

// JFrame con una vista del pdf generado

JFrame frame = new JFrame("Reporte");

frame.getContentPane().add(new JRViewer(jasperPrint));

frame.pack();

frame.setSize(800, 600);

frame.setLocationRelativeTo(null);

frame.setVisible(true);

} catch(Exception e) {

e.printStackTrace();

}

}

**iReport 2**

**Creamos otro Proyecto y una clase participante (Alt + insert (para generar constructores getters and setters))**

**Clase Participante**

public class Participante {

private int id;

private String nombre;

private String username;

private String password;

private String comentarios;

public Participante(int id, String nombre, String username, String password, String comentarios) {

this.id = id;

this.nombre = nombre;

this.username = username;

this.password = password;

this.comentarios = comentarios;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getNombre() {

return nombre;

}

public void setNombre(String nombre) {

this.nombre = nombre;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getComentarios() {

return comentarios;

}

public void setComentarios(String comentarios) {

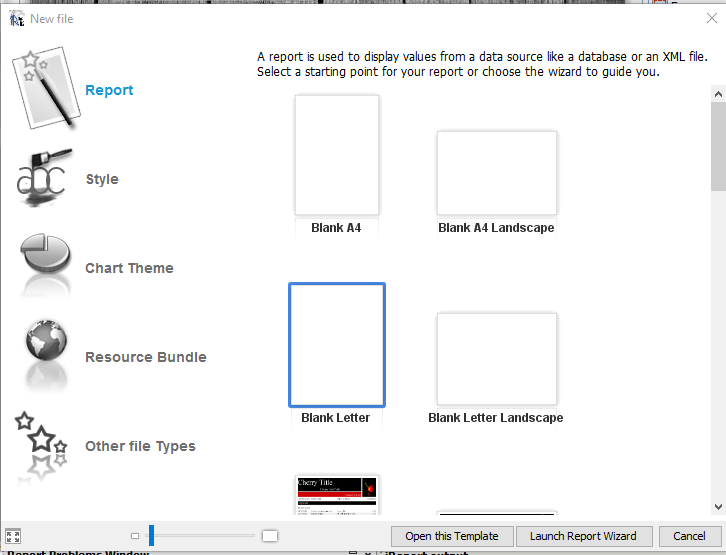
this.comentarios = comentarios;

}

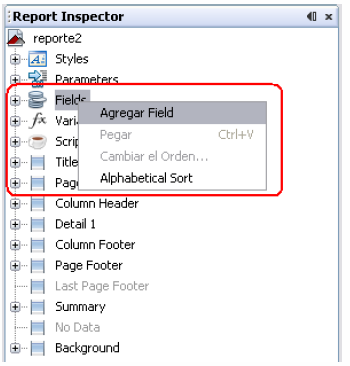
}

**Ya tenemos nuestra clase y pasaremos al iReport**

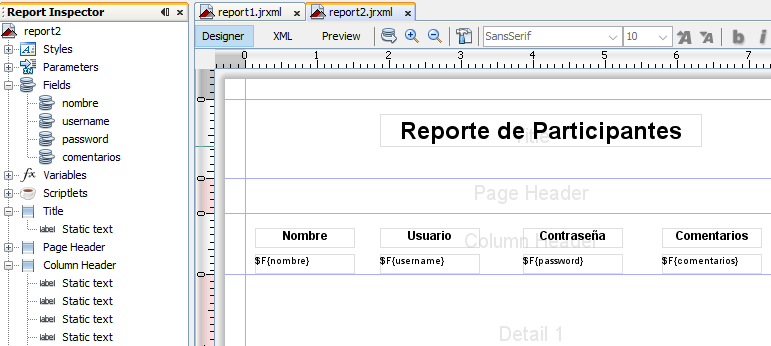
**Archivo 🡪 Nuevo 🡪 Blank letter (Open this Template).**

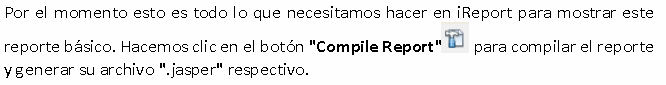


Con Statics fields creamos esto: \*creando el id 



Agregar los fields, con el mismo nombre que hemos creado en la clase Participante, y los arrastramos debajo de los Static Fields: (field id (propiedades: java.lang.Integer)).

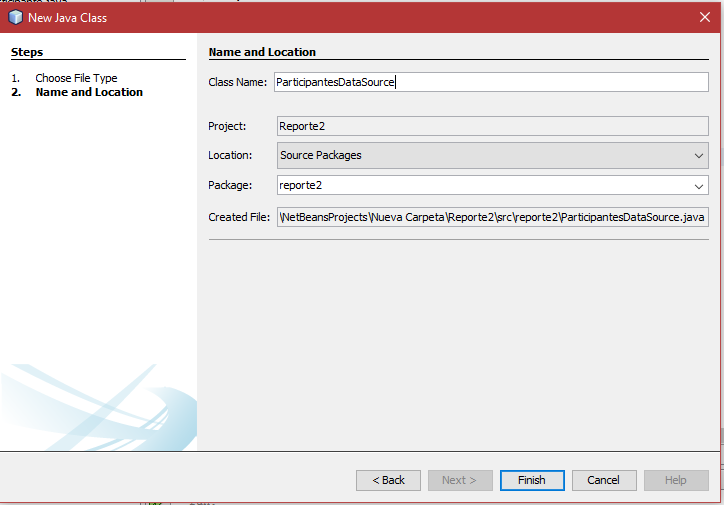




En el report2.jrxml, clicamos en Preview, pero ponemos “Empty datasource”:



Ahora ya solo nos falta crear la clase “ParticipantesDataSource”



Agregamos packages, e implementamos los métodos

Alt + insert.

**Clase ParticipantesDataSource**

public class ParticipantesDataSource implements JRDataSource {

private List<Participante> listaParticipantes = new ArrayList<>();

private int indiceParticipanteActual = -1;

@Override

public boolean next() throws JRException {

return ++indiceParticipanteActual<listaParticipantes.size();

}

@Override

public Object getFieldValue(JRField jrField) throws JRException {

Object valor = null;

if("id".equals(jrField.getName())) {

valor = listaParticipantes.get(indiceParticipanteActual).getId();

} else if("nombre".equals(jrField.getName())) {

valor = listaParticipantes.get(indiceParticipanteActual).getNombre();

} else if("username".equals(jrField.getName())) {

valor = listaParticipantes.get(indiceParticipanteActual).getUsername();

} else if("password".equals(jrField.getName())) {

valor = listaParticipantes.get(indiceParticipanteActual).getPassword();

} else if("comentarios".equals(jrField.getName())) {

valor = listaParticipantes.get(indiceParticipanteActual).getComentarios();

}

return valor;

}

public void addParticipante(Participante participante){

this.listaParticipantes.add(participante);

}

public static void main(String[] args) {

ParticipantesDataSource datasource = new ParticipantesDataSource();

for (int i= 1; i<11; i++) {

Participante p = new Participante(i, "Participante " + i, "Usuario " + i, "Pass " + i, "Comentarios para " + i);

datasource.addParticipante(p);

}

try{

JasperReport reporte = (JasperReport)

JRLoader.loadObjectFromFile("report2.jasper");

JasperPrint jasperPrint = JasperFillManager.fillReport(reporte, null, datasource);

Exporter exporter = new JRPdfExporter();

exporter.setExporterInput(new

SimpleExporterInput(jasperPrint));

exporter.setExporterOutput(new

SimpleOutputStreamExporterOutput(new

java.io.File("report2PDF.pdf")));

SimplePdfExporterConfiguration configuration = new

SimplePdfExporterConfiguration();

exporter.setConfiguration(configuration);

exporter.exportReport();

} catch (JRException ex) {

Logger.getLogger(ParticipantesDataSource.class.getName()).log(Level.SEVERE,

null, ex);

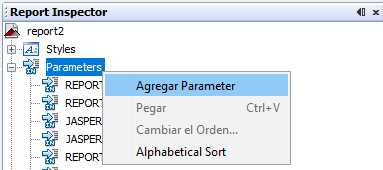
}

}

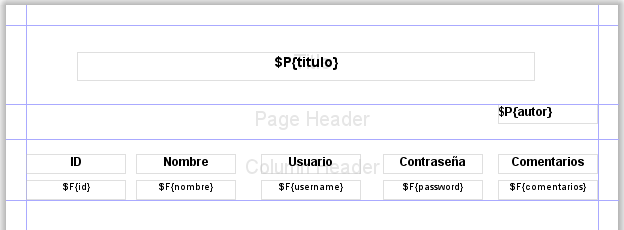
}

**iReport 3**

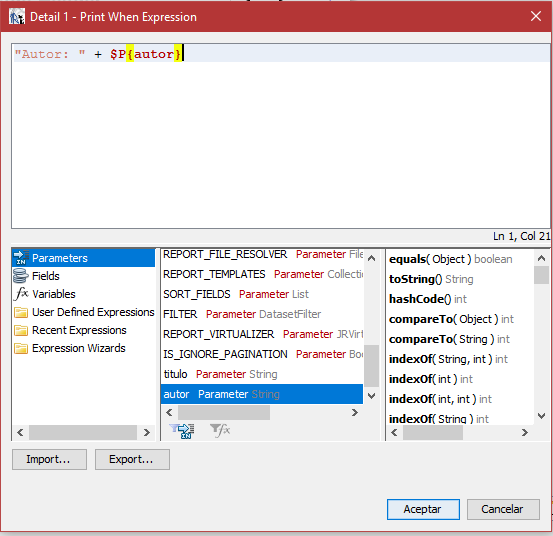
**Copiamos** el **report2** y lo **llamamos** **report3** y lo abrimos en el iReport:





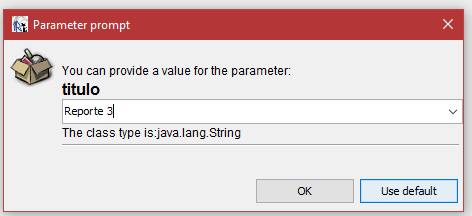


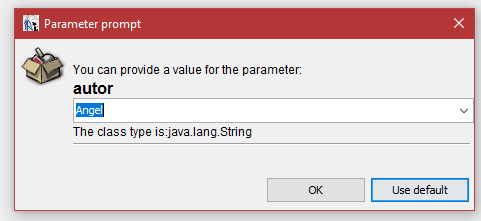
 (clicas)

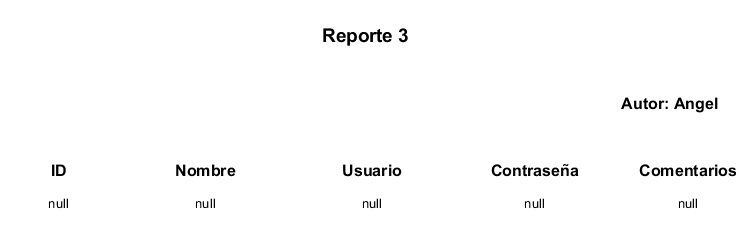


Compilas y le das a Preview: (Rellenas título y autor)







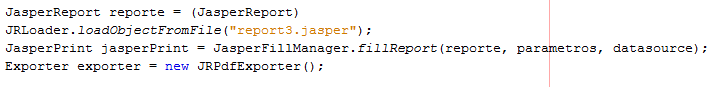


**Código de Reporte 3: (Copias y pegas el de Reporte2)**

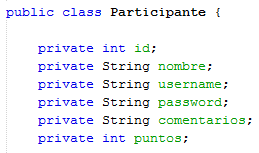
**En participantesDataSource pones estas líneas en el try:**

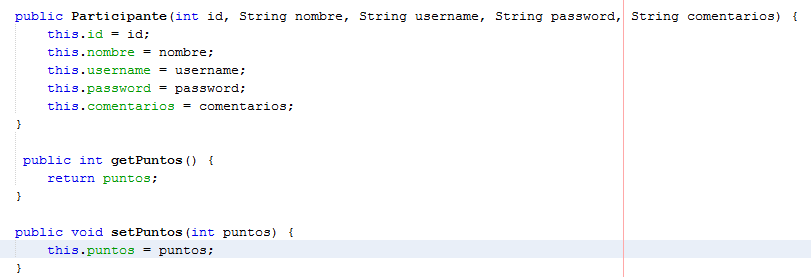


En este fragmento de código, en el JasperPrint, pasamos el parametro



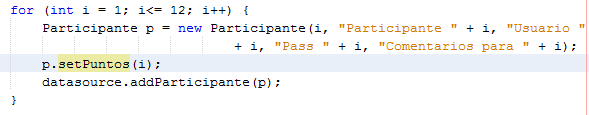
**En Participante creas un private int puntos, con sus getters y setters:**





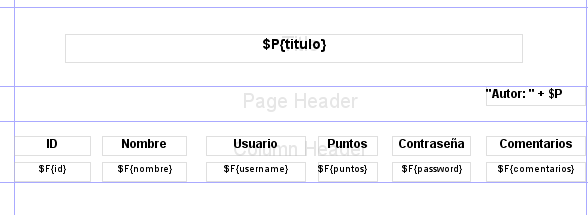
**En participantesDataSource pones esta línea en el MAIN:**

p.setPutos(i)



**AHORA REGRESAMOS AL iReport**

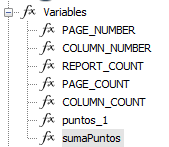
Creas un Fields puntos con un static Text



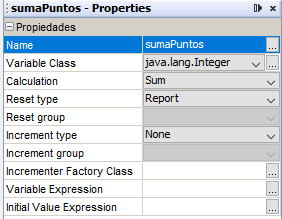
El Field puntos, lo pones Integer



Agregamos la variable sumaPuntos

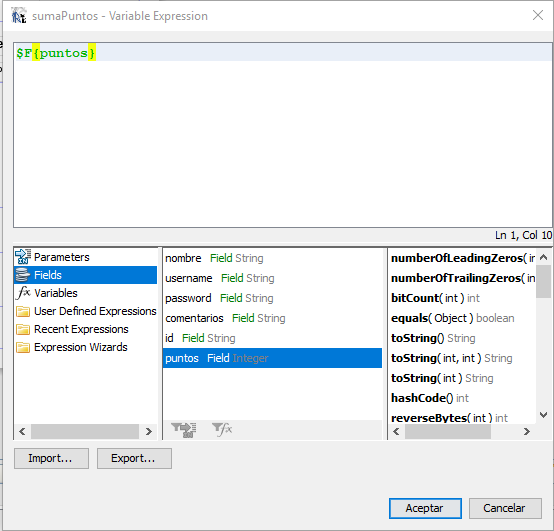


Con estas propiedades:



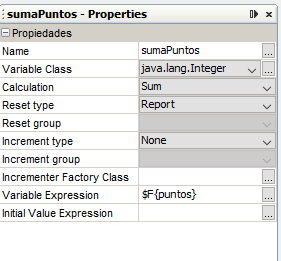
Clicamos en los “…” de Variable Expression:

Clicas dos veces en puntos Field Integer



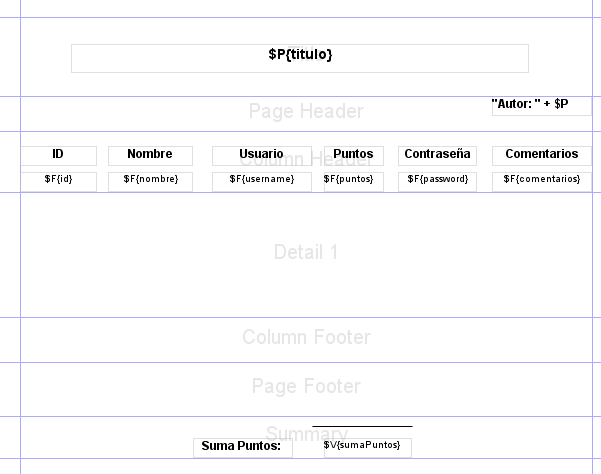
Aceptar

**Y se queda tal cual:**



Arrastramos sumaPuntos a Summary

Creamos un Static Text y una línea:



Arrastramos la variable PAGE\_NUMBER a “Page Footer”

