

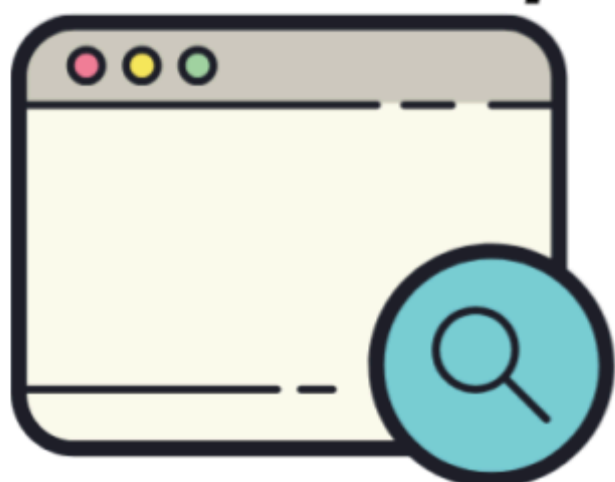


---

# Desarrollo Interfaces

---

***Tarea: Crear informe con  
JasperReport***



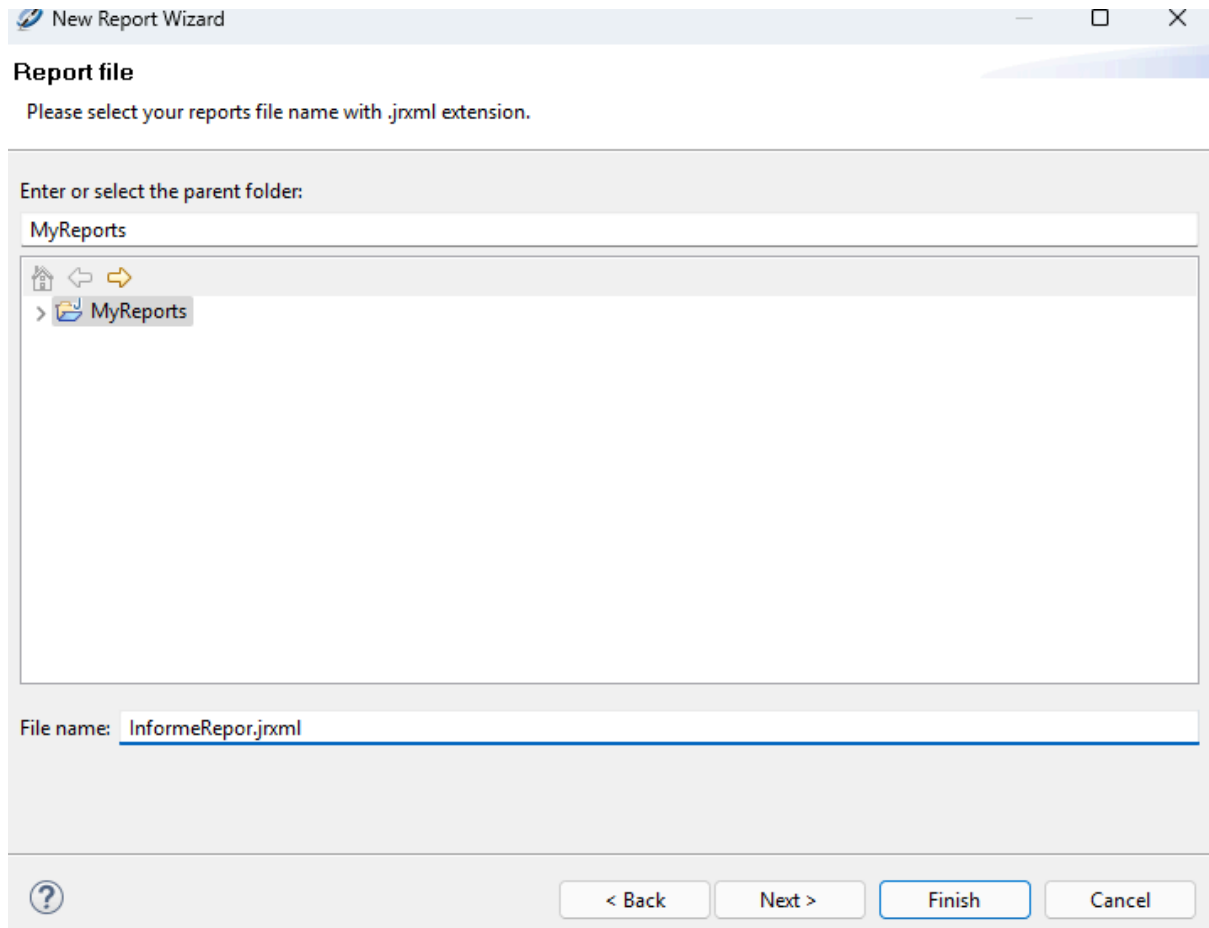
**Juan Carlos  
Filter Martin**

---

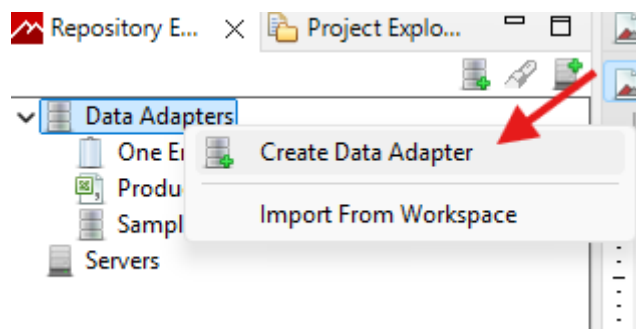
**2º DAM**

# JasperSoft Studio

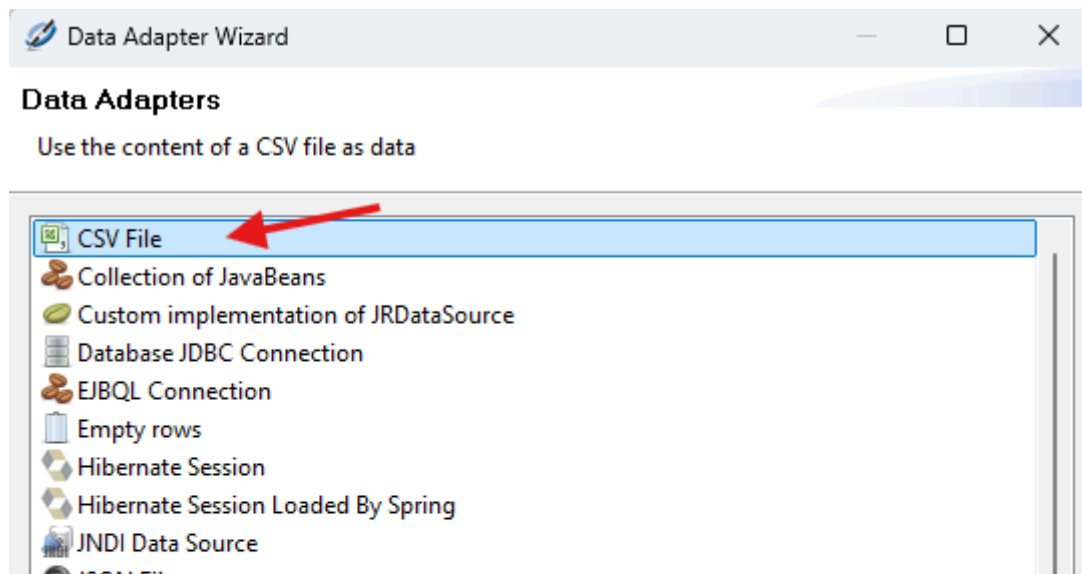
Para crear el DataSource, botón derecho sobre nuestro proyecto de Jasper Studio para sacar el menú y "New" -> "Data Adapter". Nos aparecerá una ventana para elegir en qué carpeta queremos crear el Data Adapter, ponemos un nombre y pulsamos finish



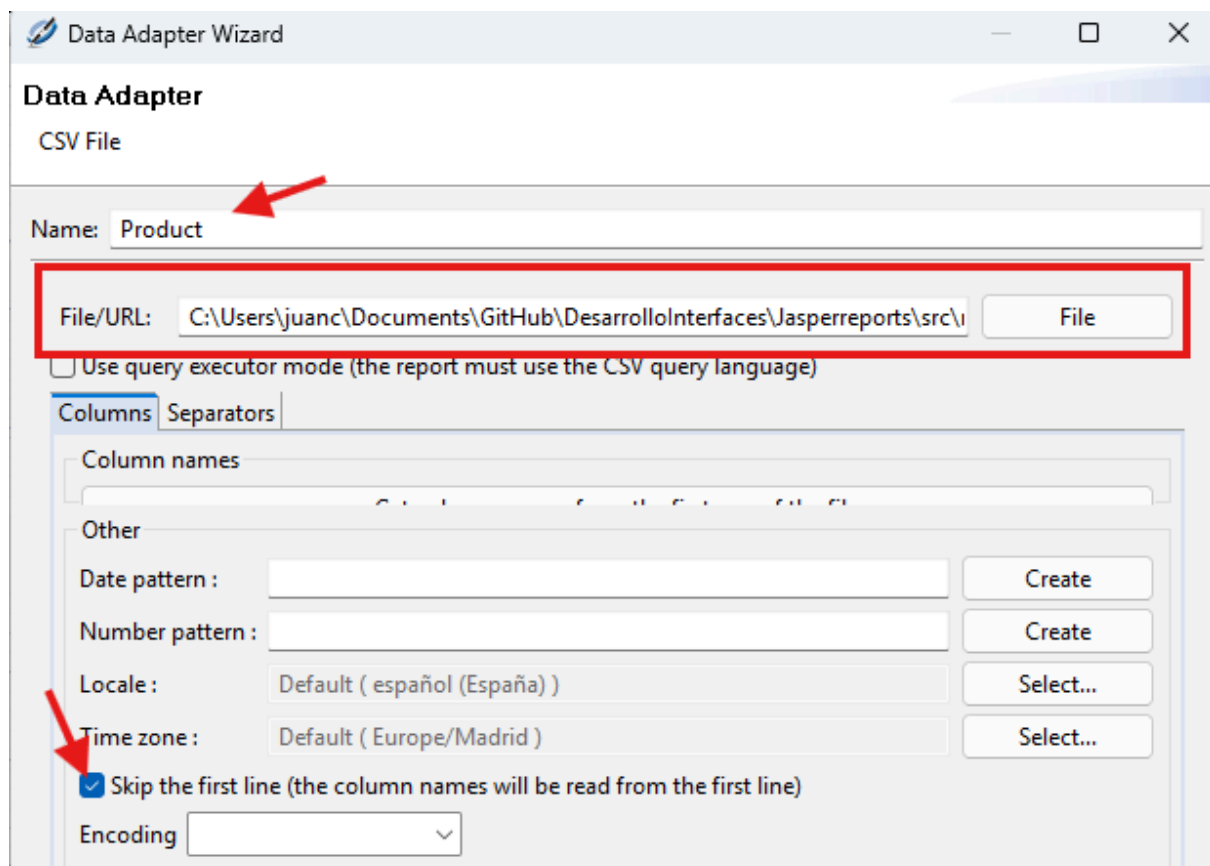
Una vez creado con botón derecho sobre Data Adapters > Create Data Adapter



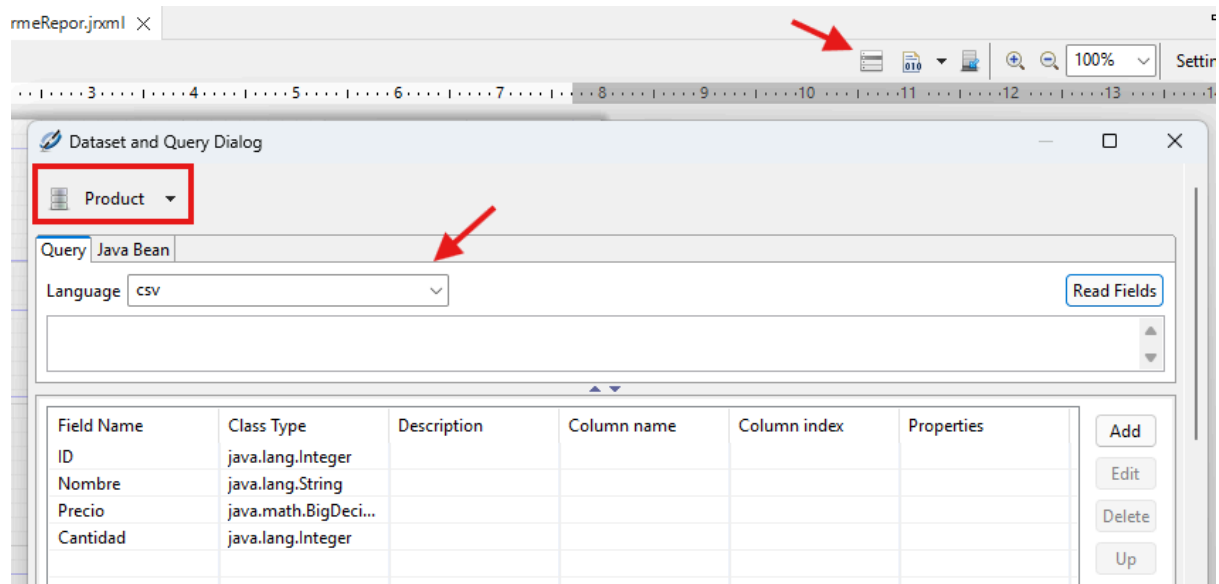
Elegimos CSV File



Ponemos un nombre al csv, indicamos la ruta donde se encuentra y marcamos la casilla de abajo para decir que ignore la primera fila siendo esta la columna del csv y en la segunda fila es donde inicia los datos.

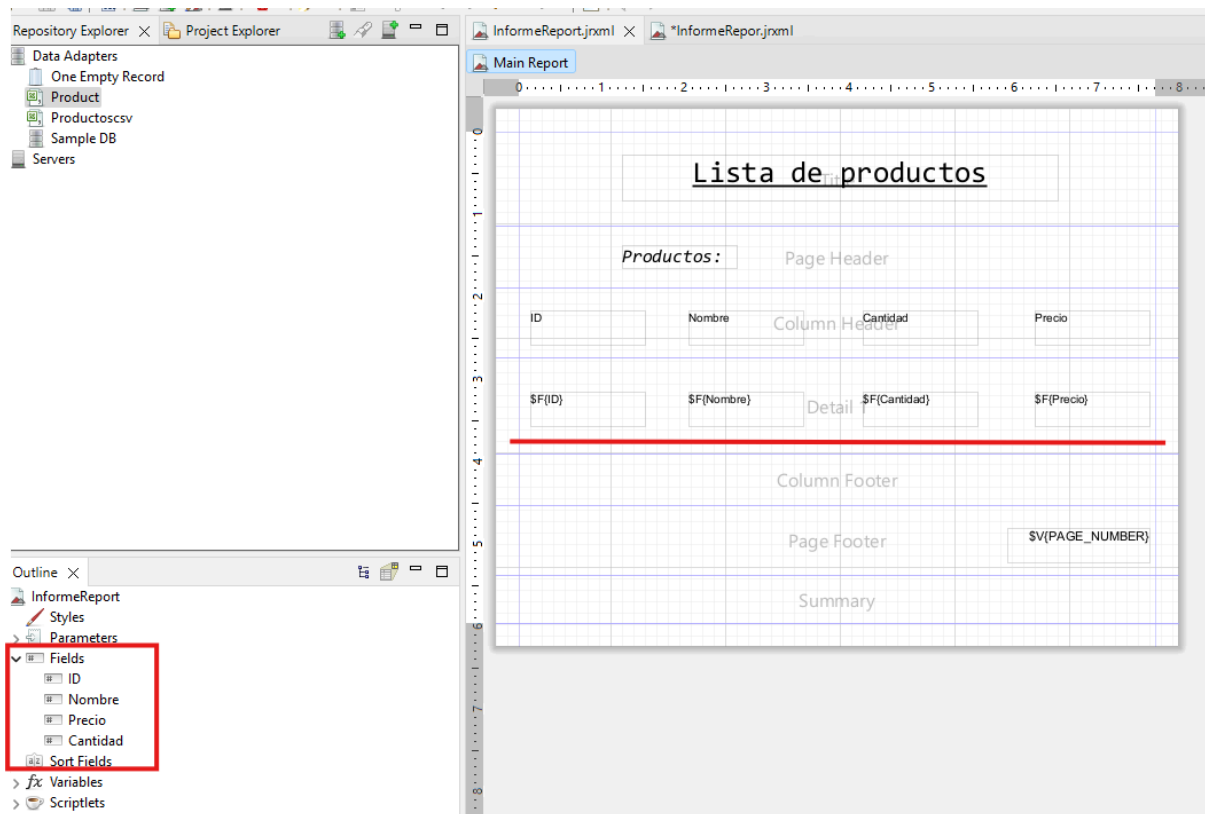


Con el csv añadido a la Data Adapters vamos a Dataset and Query dialog seleccionamos el csv Product e indicamos el lenguaje csv

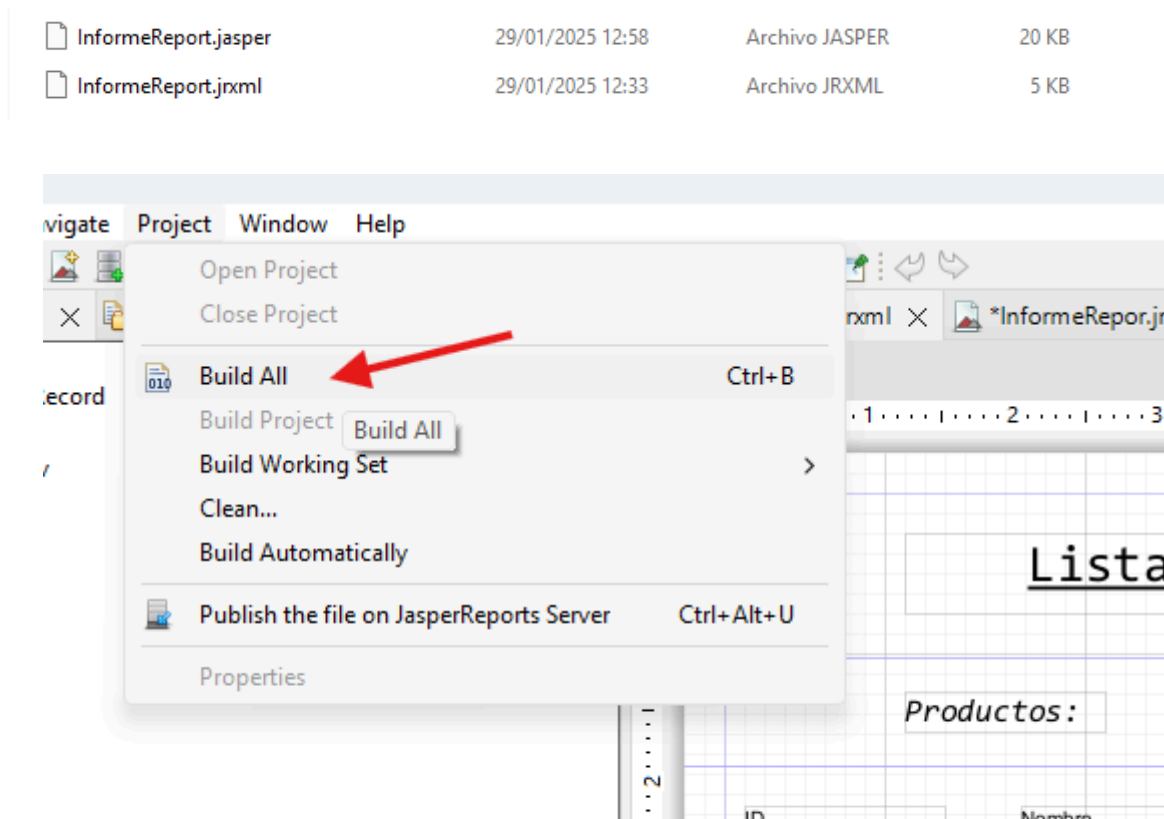


Ahora aparecerán los campos en Field y arrastrando en Detail podremos añadir esos campos al documento.

Aparte también podemos crear un título, pié de página, etc



Una vez creado el informe guardamos para generar el .jrxml y para el .jasper vamos a Project > build All.



Ahora quedará generar ese informe en java a partir del archivo .csv que se ha utilizado.

## Proyecto Java

Para generar el informe lo primero tenemos que añadir las dependencias en el pom

```
16
17     <!-- Añadimos las dependencias para jasperreports-->
18     <dependencies>
19         <dependency>
20             <groupId>net.sf.jasperreports</groupId>
21             <artifactId>jasperreports</artifactId>
22             <version>6.21.3</version>
23         </dependency>
24         <dependency>
25             <groupId>org.apache.commons</groupId>
26             <artifactId>commons-csv</artifactId>
27             <version>1.9.0</version>
28         </dependency>
29     </dependencies>
```

Una vez añadida y actualizado el pom creamos nuestra clase para generar informe

### En esta clase tenemos lo siguiente:

```
public class GenerarInforme { 1 usage  ⤴ Juan Carlos
    public static void generarInforme() { 1 usage  ⤴ Juan Carlos
        try {
            String csvFile = "src/main/jasperreports/productos.csv";
            String jrxmlFile = "C:\\Users\\juanc\\JaspersoftWorkspace\\MyReports\\InformeReport.jrxml";
            String jasperFile = "C:\\Users\\juanc\\JaspersoftWorkspace\\MyReports\\InformeReport.jasper";
            String pdfFile = "src/main/jasperreports/reporte.pdf";
            String htmlFile = "src/main/jasperreports/reporte.html";
```

Variables donde se van a indicar las rutas necesarias como son las siguientes:

**csvFile** = ruta del csv

**jrxmlFile** = ruta del jrxml generado al guardar el informe

**jasperFile** = ruta del jasper generado al crear el build

**pdfFile** = ruta del pdf que va a ser generado con esta aplicación

**htmlFile** = ruta del html que va a ser generado con esta aplicación

```
// Compilar el archivo JRXML
JasperCompileManager.compileReportToFile(jrxmlFile, jasperFile);

// Cargar datos desde el archivo CSV
JRCsvDataSource dataSource = new JRCsvDataSource(new File(csvFile));
dataSource.setUseFirstRowAsHeader(true); // Usa la primera fila como encabezado

// Llenar el informe con datos
JasperPrint jasperPrint = JasperFillManager.fillReport(jasperFile, params: null, dataSource);
```

Con el método `compileReportToFile` le pasamos tanto el jrxml como el jasper por parámetros para compilar el archivo .jrxml

Luego se carga los datos del csv creando un nuevo `JRCsvDataSource` con el nombre `dataSource` que a su vez indicamos que la primera fila es el encabezado y no contiene datos.

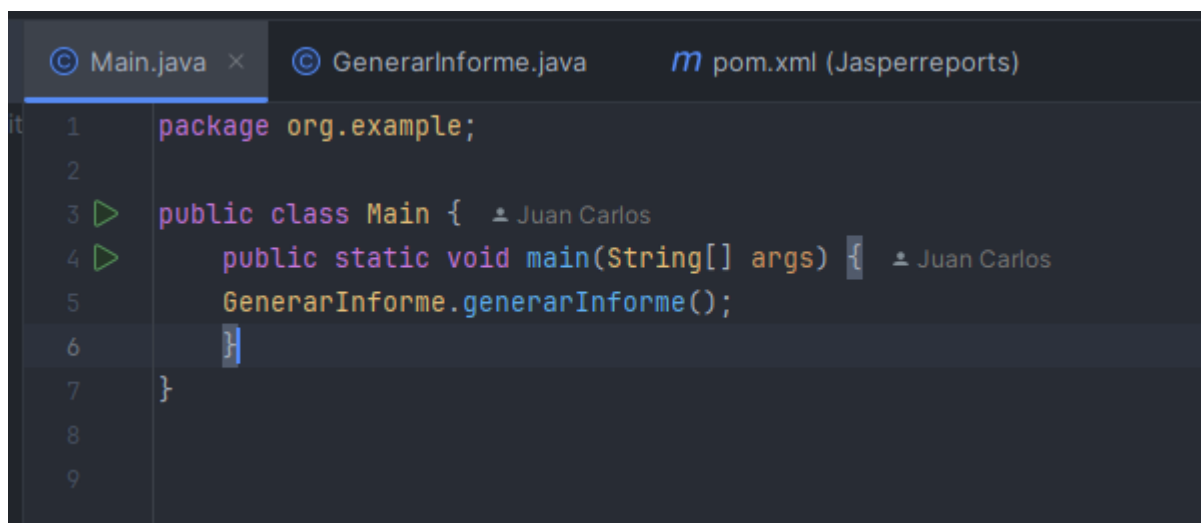
Con el método `fillReport` de la clase `JasperFillManager` pasamos por parámetro tanto el jasper como la variable donde está almacenado el csv para llenar el informe con los datos.

```
// Exportar a PDF
JasperExportManager.exportReportToPdfFile(jasperPrint, pdfFile);
System.out.println("Informe generado correctamente en PDF: " + pdfFile);

// Exportar a HTML
JasperExportManager.exportReportToHtmlFile(jasperPrint, htmlFile);
System.out.println("Informe generado correctamente en HTML: " + htmlFile);
```

Por último exportamos tanto a PDF como a HTML.

Esta clase de generar informe con el método static es llamada en el main de la aplicación

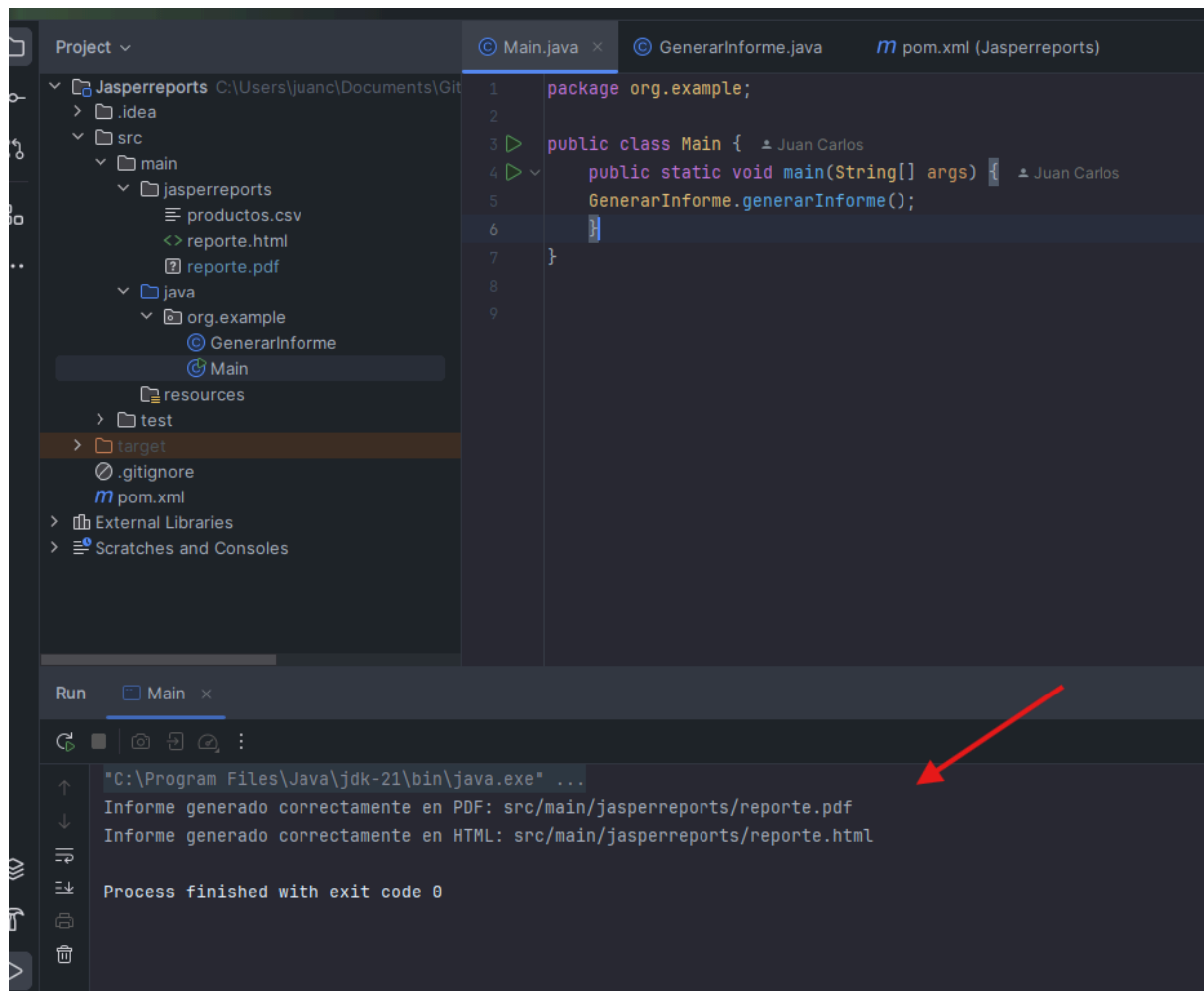


```
© Main.java ×  © GenerarInforme.java  m pom.xml (Jasperreports)

1 package org.example;
2
3 public class Main {  Juan Carlos
4     public static void main(String[] args) {  Juan Carlos
5         GenerarInforme.generarInforme();
6     }
7 }
8
9
```

## Prueba de la aplicación

Al ejecutarlo aparece que se ha generado tanto el pdf como el html



Si vamos a la ruta podemos ver como tenemos ambos archivos:

