



# INGENIERÍA DE SOFTWARE III

**Eric Gustavo Coronel Castillo**

[gcoronelc.blogspot.com](http://gcoronelc.blogspot.com)

[ecoronel@uch.edu.pe](mailto:ecoronel@uch.edu.pe)

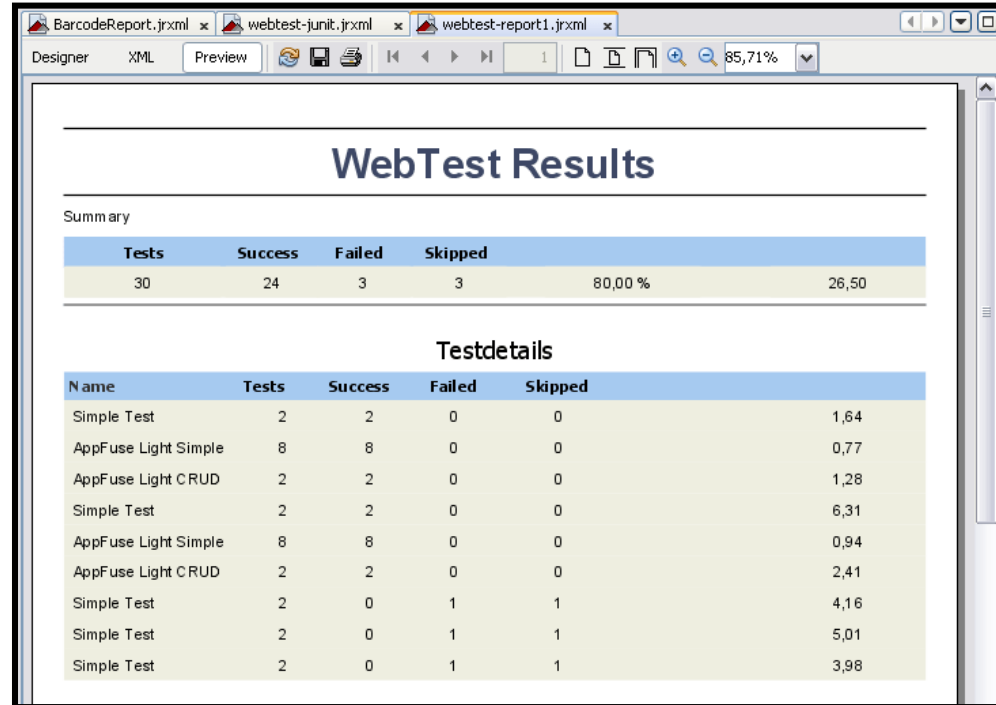


## TEMA: REPORTES



- Objetivo
- JasperReport
- iReport
- JAR dependientes
- JRXML
- Compilando y Visualizando el JRXML
- Visualizando Archivo Compilado
- Trabajando con BEANS
- Reporte en Web

- Obtener reportes impresos.



The screenshot shows a web browser window with three tabs: BarcodeReport.jrxml, webtest-junit.jrxml, and webtest-report1.jrxml. The active tab displays a preview of a report titled "WebTest Results". The report is rendered in a clean, professional style with a light blue header and a white body. It includes a summary table and a detailed test results table.

### WebTest Results

Summary

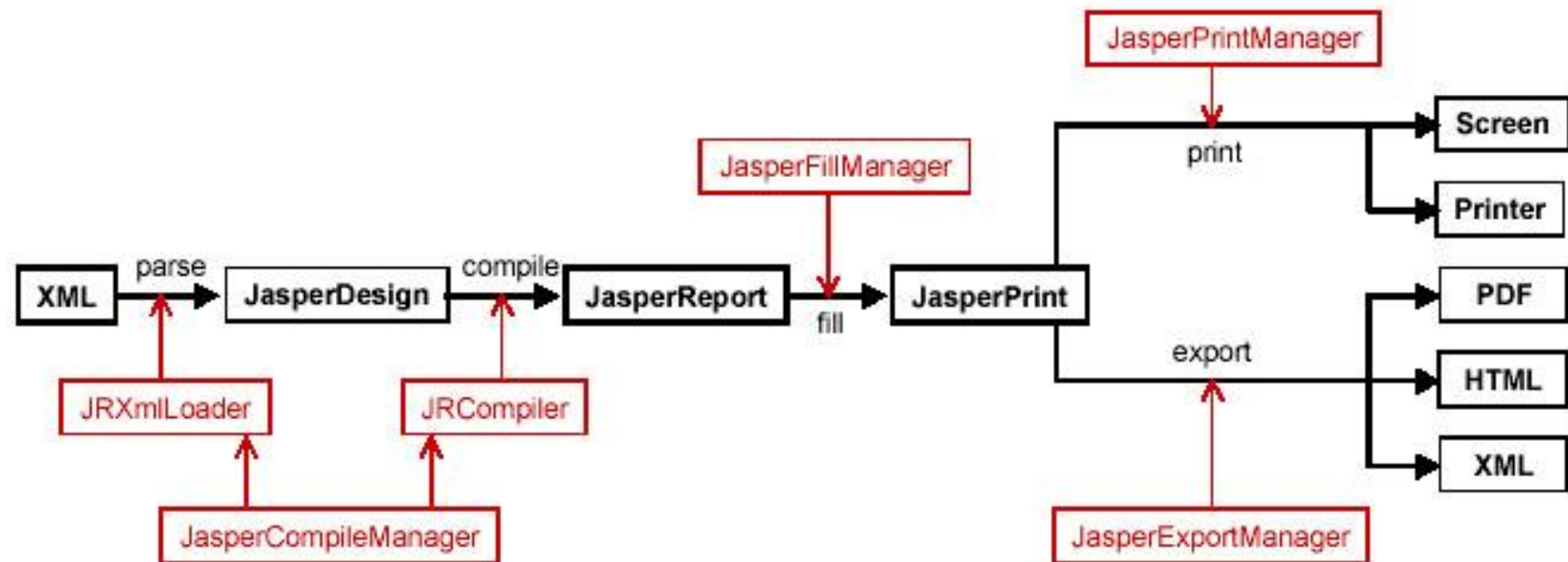
Tests	Success	Failed	Skipped		
30	24	3	3	80,00 %	26,50

Testdetails

Name	Tests	Success	Failed	Skipped	
Simple Test	2	2	0	0	1,64
AppFuse Light Simple	8	8	0	0	0,77
AppFuse Light CRUD	2	2	0	0	1,28
Simple Test	2	2	0	0	6,31
AppFuse Light Simple	8	8	0	0	0,94
AppFuse Light CRUD	2	2	0	0	2,41
Simple Test	2	0	1	1	4,16
Simple Test	2	0	1	1	5,01
Simple Test	2	0	1	1	3,98

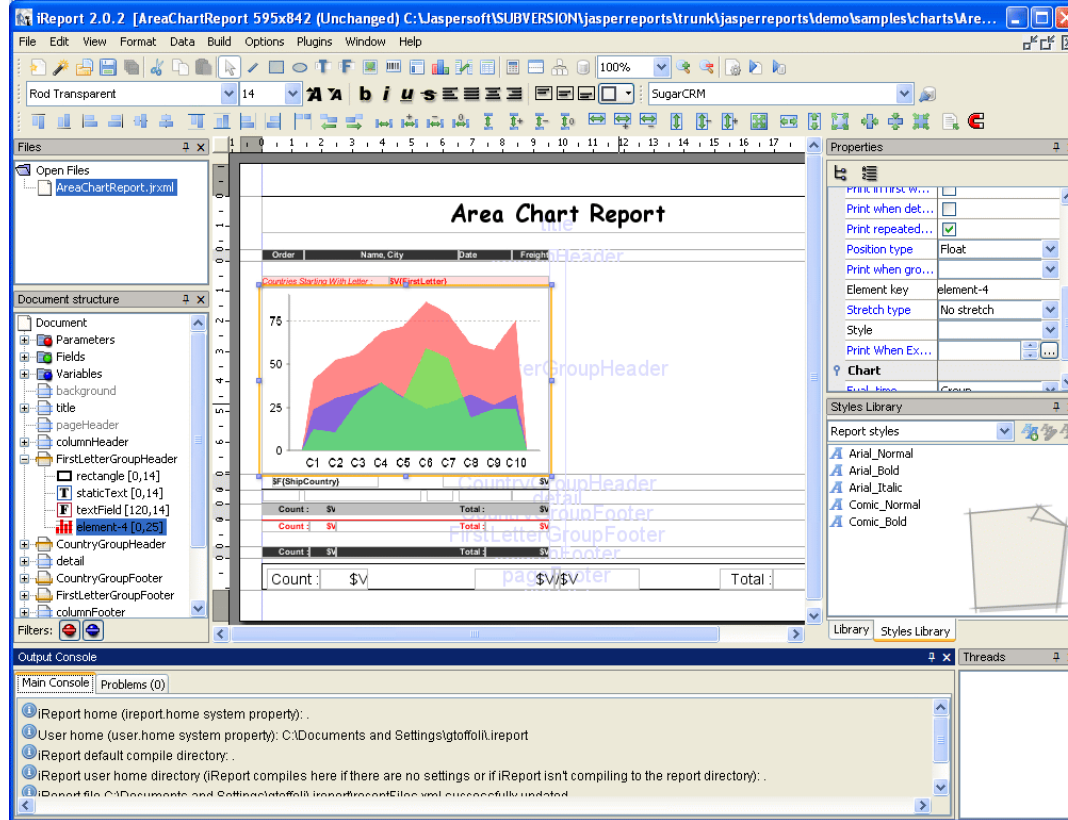


- Es una librería Open Source en Java para generar informes desde plantillas XML y diferentes fuentes de datos.
- La salida puede ser mostrada en una ventana, impresa o escrita en archivos XML, PDF, archivos de Excel y HTML.
- Pasos para trabajar con JasperReport
  - Crear el archivo de diseño del reporte (\*.jrxml)
  - Cargar el archivo de diseño del reporte (\*.jrxml)
  - Compilar el archivo de diseño del reporte (\*.jasper)
  - Llenar los datos al reporte.
  - Enviar el reporte a (Pantalla, PDF, XLS, etc)
- Home Page: <http://www.jasperforge.org/>





- Herramienta gráfica para diseñar informes, usa la librería JasperReport y genera los archivos \*.jrxml (XML) y \*.jasper (COMPILADO)
- Home Page: <http://www.jasperforge.org/>







- El JAR JasperReport depende de otras librerías para realizar diversas tareas por ejemplo.
  - SAX 2.0 XML Parser
  - Jakarta Commons Digester Component
  - Jakarta Commons BeanUtils Component
  - Jakarta Commons Collections Component
  - Jakarta Commons Logging Component
  - iText
  - JFreeChart
  - Jakarta POI (v1.5.1+)
  - etc.

```
<?xml version="1.0" encoding="UTF-8"?>

<jasperReport name="report1"
  pageWidth="595" pageHeight="842" columnWidth="535"
  leftMargin="20" rightMargin="20" topMargin="20" bottomMargin="20">

  <queryString>
    <![CDATA[select * from cliente]]>
  </queryString>

  <field name="nombre" class="java.lang.String">
    <fieldDescription><![CDATA[]]></fieldDescription>
  </field>

  <detail>
    <textFieldExpression class="java.lang.String">
      <![CDATA[${nombre}]]>
    </textFieldExpression>
  </detail>

</jasperReport>
```

```
// Objeto Connection
Connection cn = AccesoDB.getConnection();
// Parámetros
Map pars = new HashMap();
// Usando el archivo JRXML
String fileRepo = "/pe/egcc/eurekaappcs/report/repoClientes.jrxml";
InputStream isRepo = Class.class.getResourceAsStream(fileRepo);
JasperReport jrRepo = JasperCompileManager.compileReport(isRepo);
JasperPrint jpRepo= JasperFillManager.fillReport(jrRepo, pars, cn);
// Mostrando el reporte
JasperViewer viewer = new JasperViewer(jpRepo,false);
viewer.setTitle("Mi Reporte");
viewer.setVisible(true);
```

```
// El objeto Connection
cn = AccesoDB.getConnection();
// Parámetros
String fileLogo = "/pe/egcc/eurekaappcs/img/logo.gif";
InputStream isLogo = Class.class.getResourceAsStream(fileLogo);
Map<String,Object> pars = new HashMap<>();
pars.put("LOGO", fileLogo);
// Cargando el archivo compilado
String fileRepo = "/pe/egcc/eurekaappcs/report/repoClientes.jasper";
InputStream isRepo = Class.class.getResourceAsStream(fileRepo);
JasperReport jrRepo = (JasperReport) JRLoader.loadObject(isRepo);
JasperPrint jpRepo = JasperFillManager.fillReport(jrRepo, pars, cn);
// Visualizando el Reporte
JasperViewer viewer = new JasperViewer(jpRepo, false);
viewer.setTitle("Mi Reporte");
viewer.setVisible(true);
```

```
// Obteniendo Colección de datos
Collection lista = service.TraerListaBeans();
// Transformando Colección de datos a format Jasper
JRBeanCollectionDataSource data;
data = new JRBeanCollectionDataSource(lista);
// Parámetros
Map pars = new HashMap();
pars.put("USUARIO", "Gustavo Coronel");
pars.put("LOGO", "logo.gif");
// Usando el archivo JASPER (compilado)
String fileRepo = "/pe/egcc/eurekaappcs/report/repoClientes.jasper";
InputStream isRepo = Class.class.getResourceAsStream(fileRepo);
JasperReport jrRepo = (JasperReport) JRLoader.loadObject(isRepo);
JasperPrint jpRepo = JasperFillManager.fillReport(jrRepo, pars, data);
// Creando y Visualizando PDF
JasperExportManager.exportReportToPdf(jpRepo);
JasperViewer.viewReport(jpRepo, false);
```

```
// Conexión a la base de datos
cn = AccesoDB.getConnection();
// Parámetro
String cuenta = request.getParameter("cuenta");
Map<String, Object> parms = new HashMap<>();
parms.put("CUENTA", cuenta);
// El archivo
String fileRepo = "/pe/egcc/eurekaweb/reports/repoMovimientos.jasper";
InputStream isRepo = getClass().getResourceAsStream(fileRepo);
byte[] bytes = JasperRunManager.runReportToPdf(isRepo, parms, cn);
// Enviar el arreglo al browser
response.setContentType("Application/pdf");
response.setContentLength(bytes.length);
ServletOutputStream out = response.getOutputStream();
out.write(bytes, 0, bytes.length);
out.flush();
out.close();
```