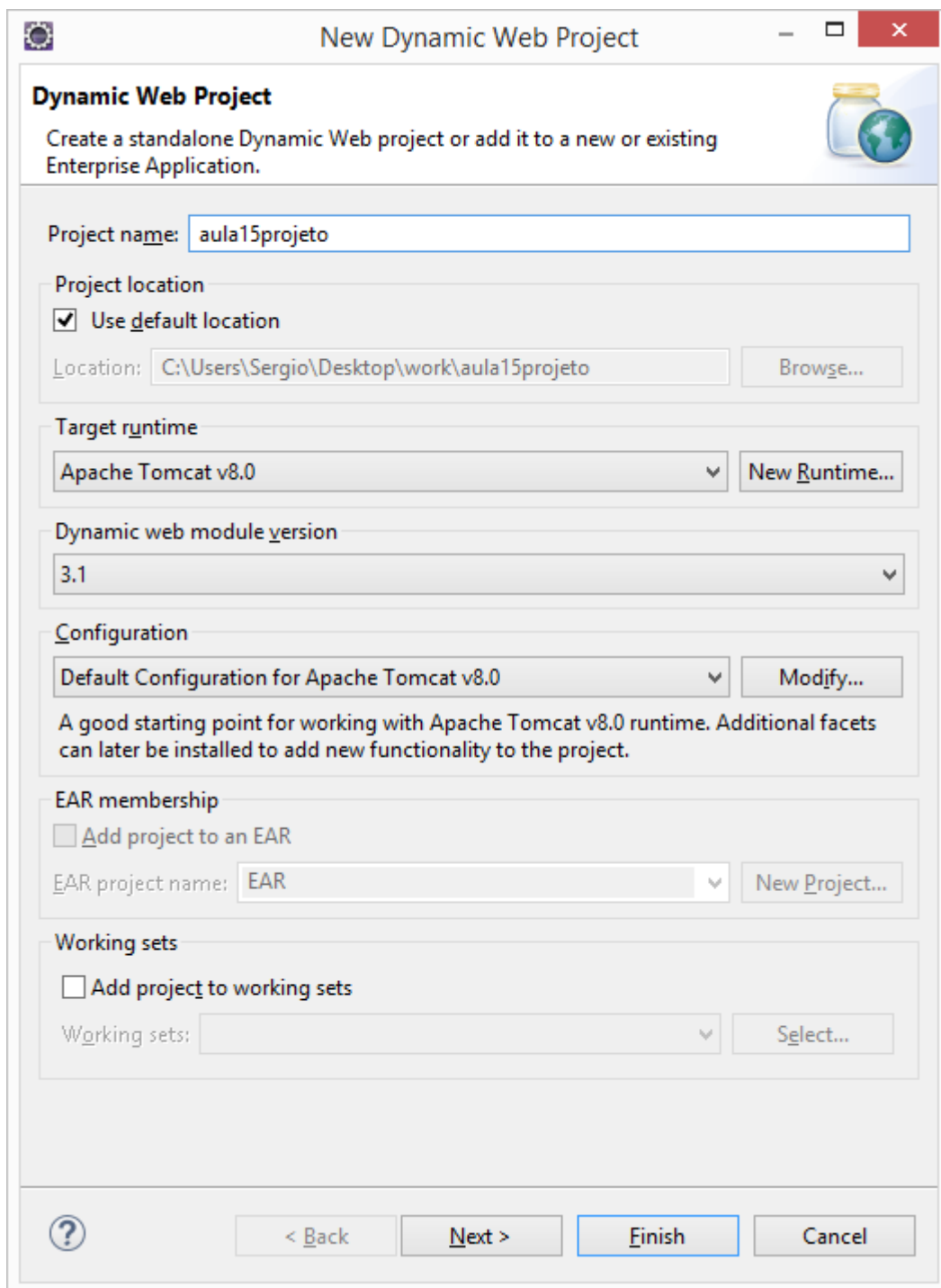


Criando o projeto

## File > New > Dynamic Web Project



The screenshot shows the 'New Dynamic Web Project' dialog box in the Eclipse IDE. The dialog is titled 'New Dynamic Web Project' and has a close button (X) in the top right corner. It contains several sections for configuring a new project:

- Dynamic Web Project**: A section with a description: 'Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.' and a small icon of a globe and a jar.
- Project name:** A text field containing 'aula15projeto'.
- Project location**: A section with a checkbox 'Use default location' which is checked. Below it is a 'Location:' text field containing 'C:\Users\Sergio\Desktop\work\aula15projeto' and a 'Browse...' button.
- Target runtime**: A dropdown menu showing 'Apache Tomcat v8.0' and a 'New Runtime...' button.
- Dynamic web module version**: A dropdown menu showing '3.1'.
- Configuration**: A dropdown menu showing 'Default Configuration for Apache Tomcat v8.0' and a 'Modify...' button. Below this is a text description: 'A good starting point for working with Apache Tomcat v8.0 runtime. Additional facets can later be installed to add new functionality to the project.'
- EAR membership**: A section with a checkbox 'Add project to an EAR' which is unchecked. Below it is an 'EAR project name:' text field containing 'EAR' and a 'New Project...' button.
- Working sets**: A section with a checkbox 'Add project to working sets' which is unchecked. Below it is a 'Working sets:' text field and a 'Select...' button.

At the bottom of the dialog, there is a question mark icon, a '< Back' button, a 'Next >' button, a 'Finish' button, and a 'Cancel' button.

## Criando as entidades:

```
package br.com.brq.entities.types;

public enum Categoria {

    CLIENTE_COMUM,
    CLIENTE_PREFERENCIAL,
    CLIENTE_VIP
}

package br.com.brq.entities;

import javax.persistence.Column;
import javax.persistence.MappedSuperclass;

@MappedSuperclass
public abstract class Pessoa {

    @Column(name = "nome", length = 50, nullable = false)
    private String nome;

    @Column(name = "sobrenome", length = 50, nullable = false)
    private String sobrenome;

    public Pessoa() {
        // TODO Auto-generated constructor stub
    }

    public Pessoa(String nome, String sobrenome) {
        super();
        this.nome = nome;
        this.sobrenome = sobrenome;
    }

    public String getNome() {
        return nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }

    public String getSobrenome() {
        return sobrenome;
    }
}
```

```
        public void setSobrenome(String sobrenome) {
            this.sobrenome = sobrenome;
        }

        @Override
        public String toString() {
            return "Pessoa [nome=" + nome + ", sobrenome=" + sobrenome + "];"
        }
    }

package br.com.brq.entities;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.EnumType;
import javax.persistence.Enumerated;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
import javax.persistence.Table;

import br.com.brq.entities.types.Categoria;

@Entity
@Table(name = "cliente")
@NamedQueries({
    {
        @NamedQuery(name = Cliente.FIND_ALL, //nome da consulta..
                    query = "select c from Cliente as c") //HQL..
    }
})
public class Cliente extends Pessoa{

    public static final String FIND_ALL = "cliente.findall";

    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    @Column(name = "idcliente")
    private Integer idCliente;
```

```
@Column(name = "cpf", length = 11, nullable = false, unique = true)
private String cpf;

@Enumerated(EnumType.STRING)
@Column(name = "categoria", nullable = false)
private Categoria categoria;

public Cliente() {
    // TODO Auto-generated constructor stub
}

public Cliente(String nome, String sobrenome, Integer idCliente, String cpf,
    Categoria categoria) {
    super(nome, sobrenome);
    this.idCliente = idCliente;
    this.cpf = cpf;
    this.categoria = categoria;
}

public Integer getIdCliente() {
    return idCliente;
}

public void setIdCliente(Integer idCliente) {
    this.idCliente = idCliente;
}

public String getCpf() {
    return cpf;
}

public void setCpf(String cpf) {
    this.cpf = cpf;
}

public Categoria getCategoria() {
    return categoria;
}

public void setCategoria(Categoria categoria) {
    this.categoria = categoria;
}

@Override
```

```
public String toString() {  
    return "Cliente [idCliente=" + idCliente + ", cpf=" + cpf  
        + ", categoria=" + categoria + "];"  
}  
}
```

---

## Configuração para acesso ao banco de dados:

```
<?xml version="1.0" encoding="UTF-8"?>  
<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate  
Configuration DTD 3.0//EN" "http://hibernate.sourceforge.net/hibernate-  
configuration-3.0.dtd">  
<hibernate-configuration>  
    <session-factory>  
        <property name="hibernate.dialect">  
            org.hibernate.dialect.MySQLDialect</property>  
        <property name="hibernate.connection.driver_class">  
            com.mysql.jdbc.Driver</property>  
        <property name="hibernate.connection.url">  
            jdbc:mysql://localhost:3306/aula15</property>  
        <property name="hibernate.connection.username">root</property>  
        <property name="hibernate.connection.password">brqbrq</property>  
  
        <property name="hibernate.show_sql">true</property>  
        <property name="hibernate.format_sql">true</property>  
  
        <mapping class="br.com.brq.entities.Cliente"/>  
  
    </session-factory>  
</hibernate-configuration>
```

---

## Classe de persistência para a entidade Cliente:

DAOCliente.java

```
package br.com.brq.persistence;  
  
import java.util.List;  
  
import org.hibernate.Query;  
import org.hibernate.Session;  
import org.hibernate.Transaction;  
  
import br.com.brq.entities.Cliente;  
  
public class DAOCliente {  
  
    private Session session;
```

```
private Transaction transaction;
private Query query;

public void saveOrUpdate(Cliente c) throws Exception{
    session = HibernateUtil.getSessionFactory().openSession();

    transaction = session.beginTransaction();
    session.saveOrUpdate(c);
    transaction.commit();

    session.close();
}

public void delete(Cliente c) throws Exception{
    session = HibernateUtil.getSessionFactory().openSession();

    transaction = session.beginTransaction();
    session.delete(c);
    transaction.commit();

    session.close();
}

public List<Cliente> findAll() throws Exception{
    session = HibernateUtil.getSessionFactory().openSession();

    query = session.getNamedQuery(Cliente.FIND_ALL);
    List<Cliente> lista = query.list();

    session.close();
    return lista;
}

public Cliente findById(Integer idCliente) throws Exception{
    session = HibernateUtil.getSessionFactory().openSession();

    Cliente c = (Cliente) session.get(Cliente.class, idCliente);
    session.close();

    return c;
}
}
```

**Criando um ManagedBean para exibir as opções do ENUM na página de clientes:**

```
package br.com.brq.managedbeans;

import java.util.ArrayList;
import java.util.List;
import br.com.brq.entities.types.Categoria;
```

```
public class ManagedBeanCategoria {  
  
    private List<String> listagemCategorias;  
  
    public List<String> getListagemCategorias() {  
  
        listagemCategorias = new ArrayList<String>();  
  
        //varrendo cada item (constante) do enum  
        for(Categoria c : Categoria.values()){  
            listagemCategorias.add(c.name());  
        }  
  
        return listagemCategorias;  
    }  
}
```

## Criando a página:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"  
    pageEncoding="ISO-8859-1"%>  
  
<jsp:useBean class="br.com.brq.managedbeans.ManagedBeanCategoria" id="mb"/>  
  
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>  
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fnc" %>  
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt" %>  
  
<html>  
<head>  
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">  
<title>Projeto</title>  
  
    <link rel="stylesheet" type="text/css"  
        href="css/bootstrap.min.css"/>  
    <link rel="stylesheet" type="text/css"  
        href="css/bootstrap-theme.min.css"/>  
  
</head>  
<body class="container">  
  
    <div class="well well-sm">  
        <h2>Projeto Controle de Clientes</h2>  
    </div>  
  
    <div class="col-md-12">  
  
        <h3>Cadastro de Clientes</h3>  
        <hr/>
```

```
<div class="col-md-4">

    <form name="formulario" method="post"
        action="ControleCliente?action=cadastrar">

        <label>Nome do Cliente:</label>
        <input type="text" name="nome" class="form-control"
            placeholder="Digite aqui"/>
        <br/>

        <label>Sobrenome:</label>
        <input type="text" name="sobrenome"
            class="form-control"
            placeholder="Digite aqui"/>
        <br/>

        <label>Cpf:</label>
        <input type="text" name="cpf" class="form-control"
            placeholder="Digite aqui"/>
        <br/>

        <label>Selecione a Categoria:</label>
        <br/>

        <c:forEach items="${mb.listagemCategorias}"
            var="cat">

            <input type="radio" name="categoria"
                value="${cat}"/> ${cat}
            <br/>

        </c:forEach>

        <br/>

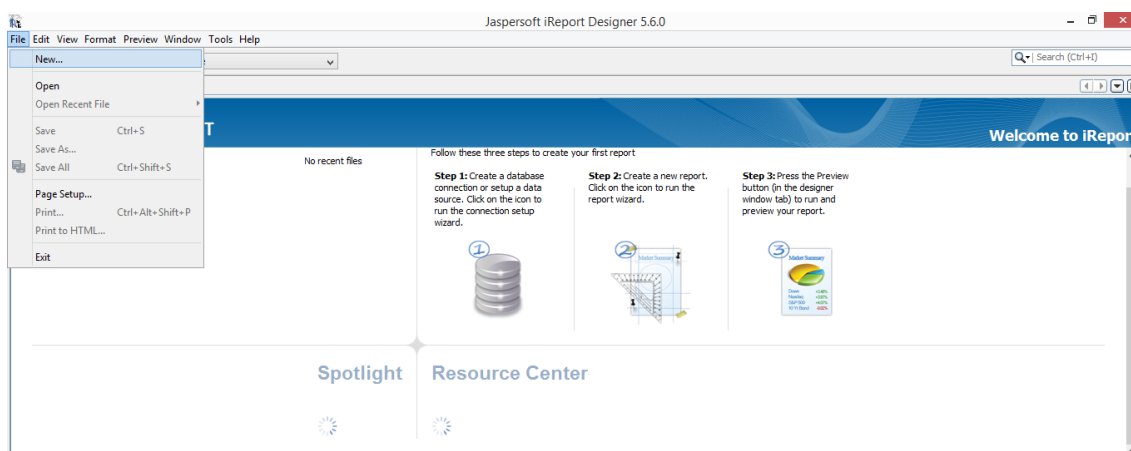
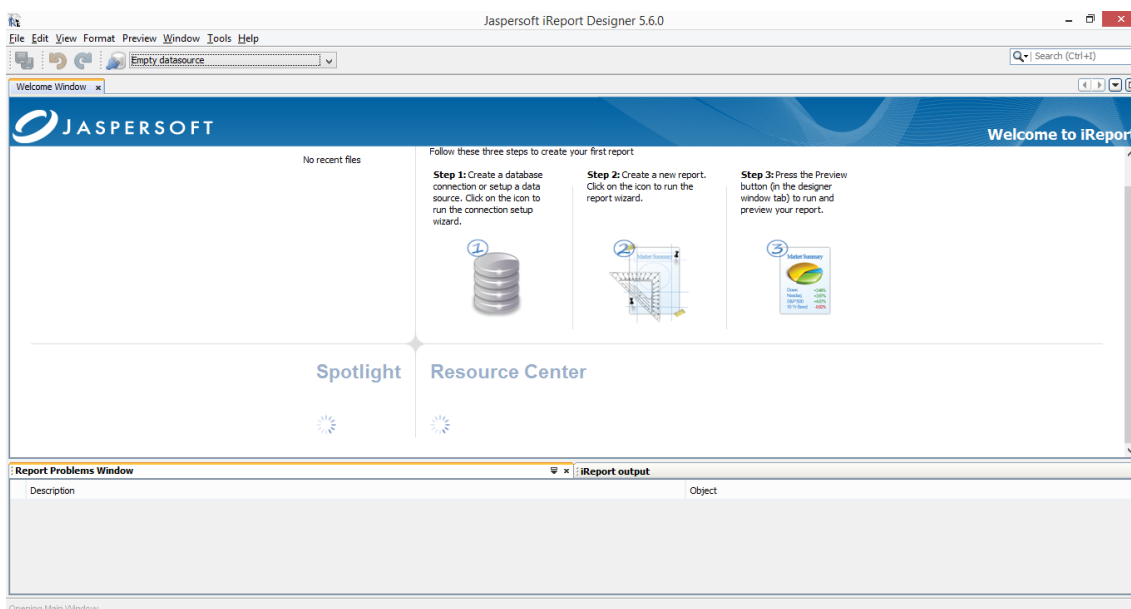
        <input type="submit" value="Cadastrar Cliente"
            class="btn btn-success"/>
        <br/><br/>

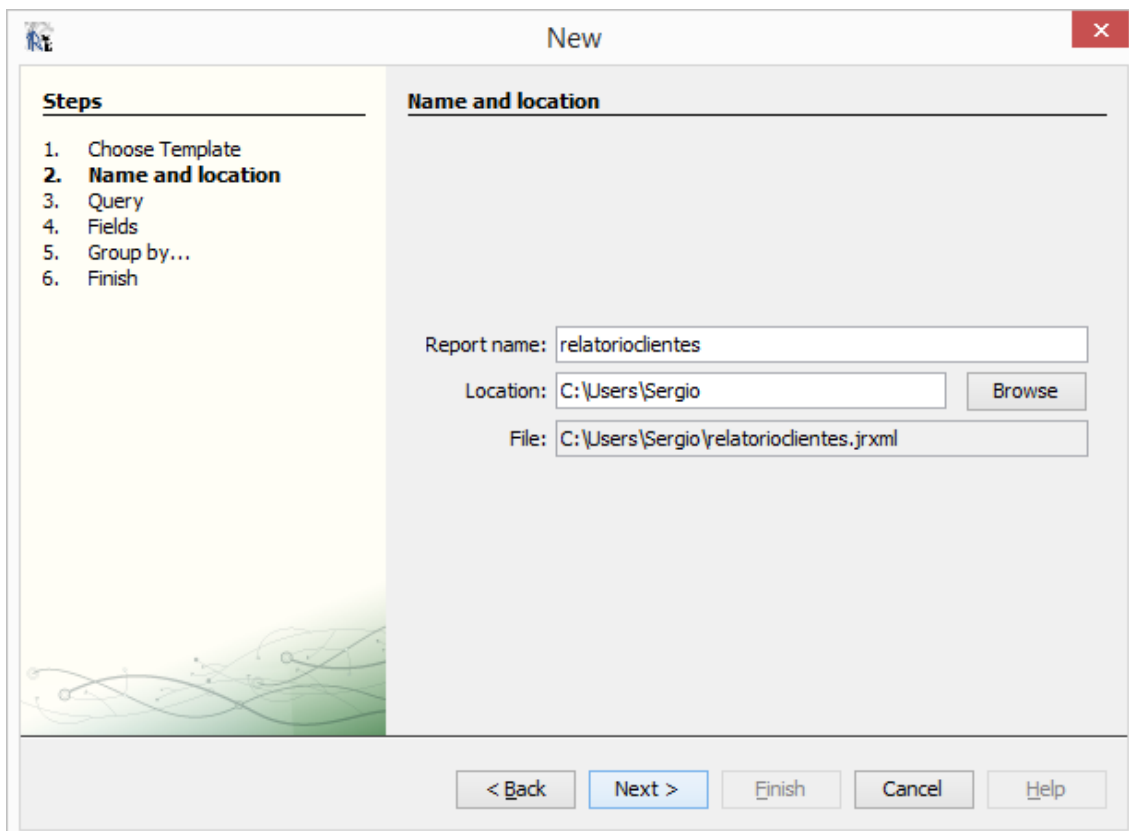
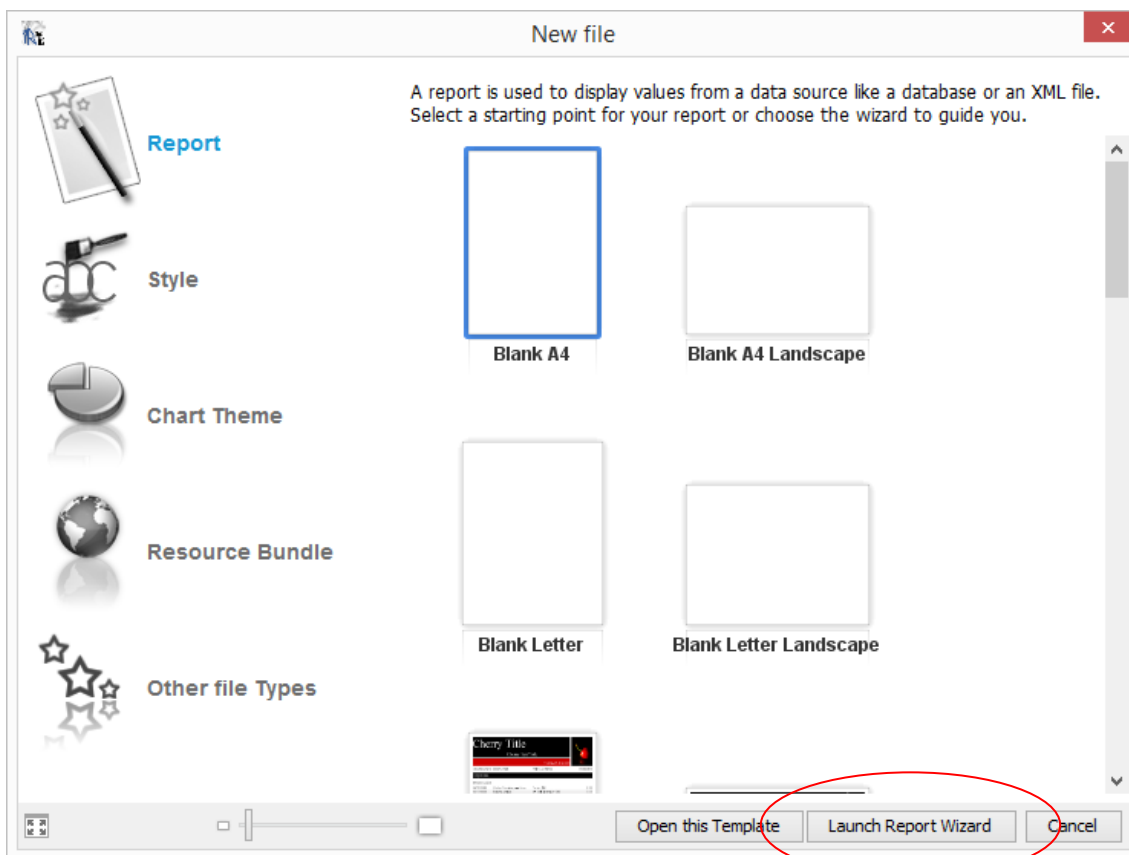
        ${mensagem}
    </form>

    <hr/>
    <a href="ControleCliente?action=relatorio"
        class="btn btn-primary">
        Gerar Relatório de Clientes
    </a>
</div>
</div>
</body>
</html>
```

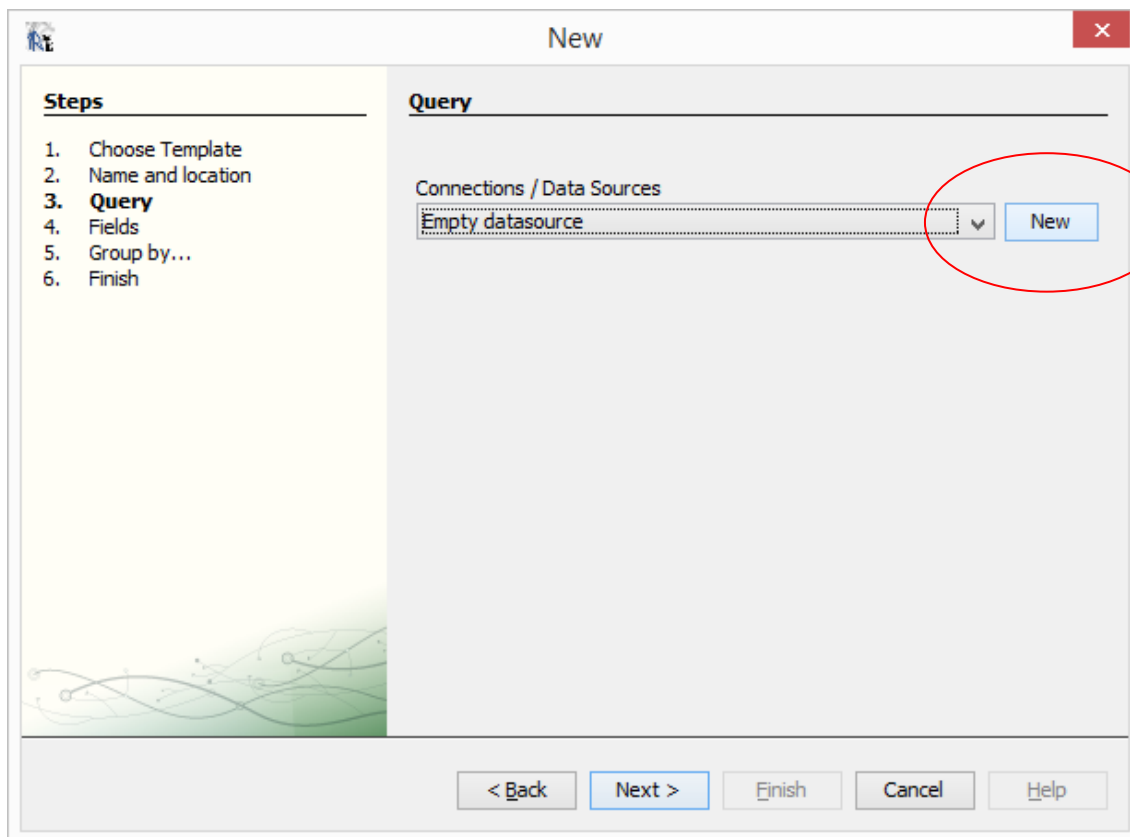
## Criando o Relatório no iReport



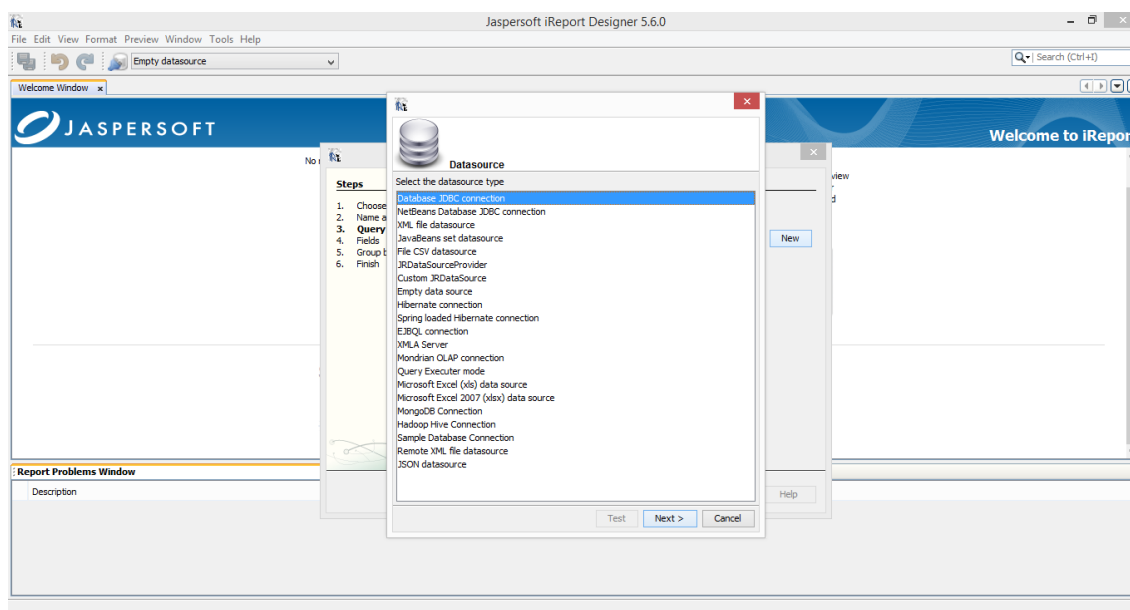


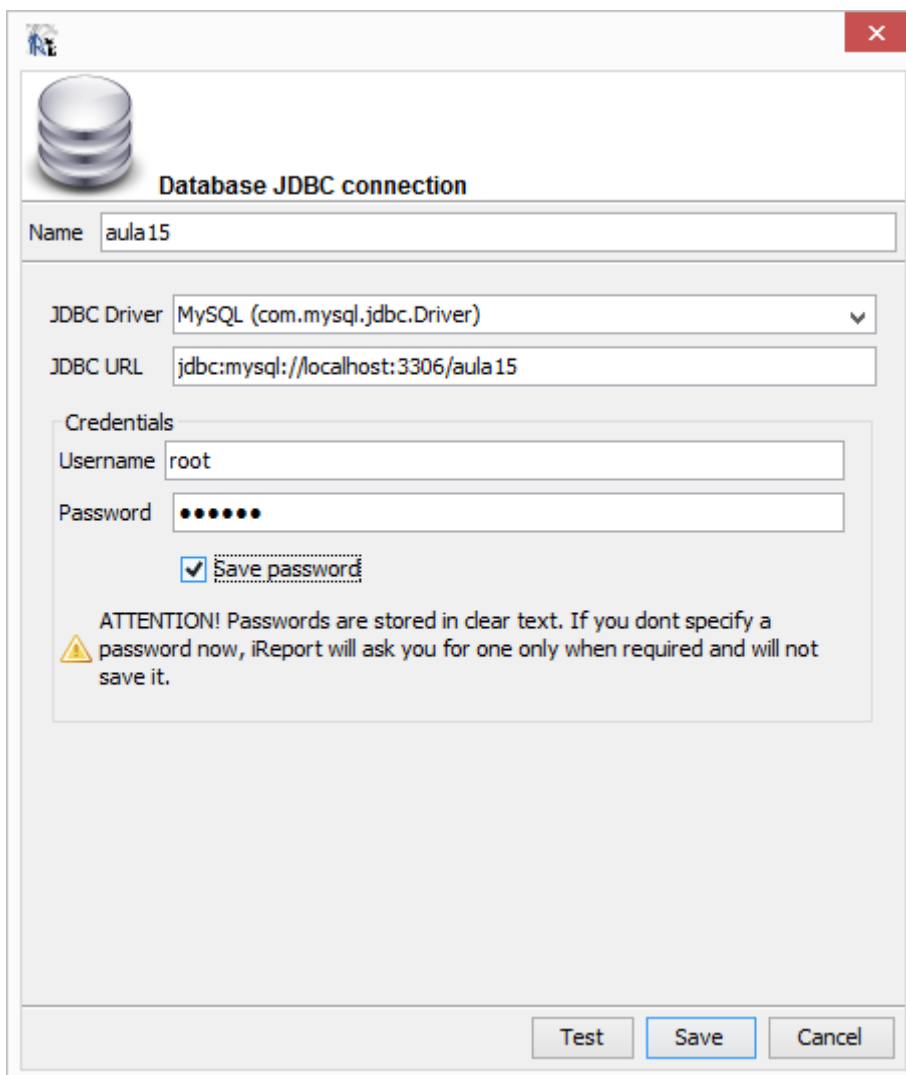


## Criando a conexão com o banco de dados:



## Selecione JDBC





**Database JDBC connection**

Name:

JDBC Driver:

JDBC URL:

Credentials

Username:

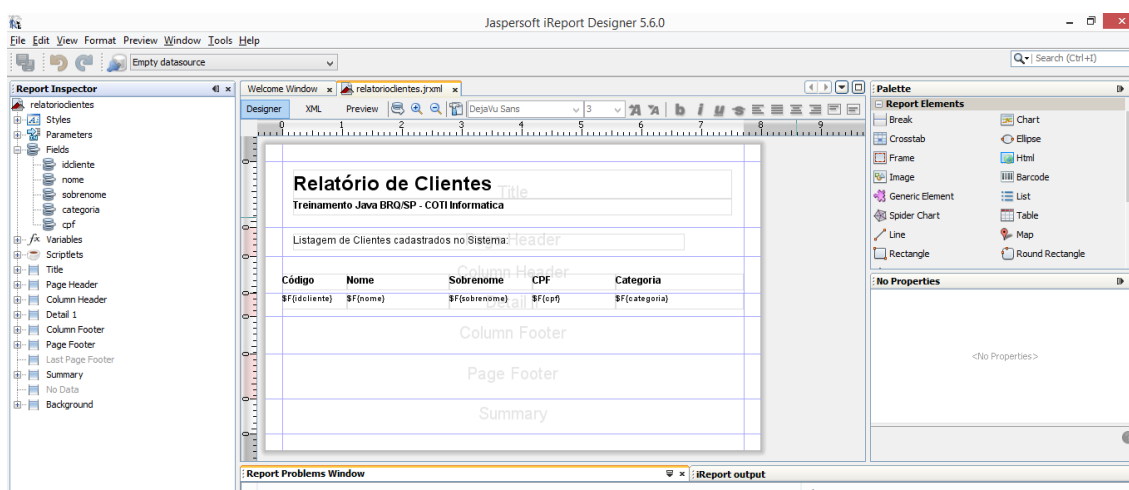
Password:

☒ Save password

**ATTENTION!** Passwords are stored in clear text. If you dont specify a password now, iReport will ask you for one only when required and will not save it.

Test Save Cancel

## Desenhando o relatório:



Jaspersoft iReport Designer 5.6.0

File Edit View Format Preview Window Tools Help

Empty datasource

Search (Ctrl+F)

**Report Inspector**

- relatorioclientes
  - Styles
  - Parameters
  - Fields
    - cliente
    - nome
    - sobrenome
    - categoria
    - cpf
  - Variables
  - Scriptlets
  - Title
  - Page Header
  - Column Header
  - Detail 1
  - Column Footer
  - Page Footer
  - Last Page Footer
  - Summary
  - No Data
  - Background

**Designer**

XML Preview

DejaVu Sans

**Relatório de Clientes**

Treinamento Java BRQ/SP - COTI Informática

Listagem de Clientes cadastrados no Sistema

Código	Nome	Sobrenome	CPF	Categoria
\$(cliente)	\$(nome)	\$(sobrenome)	\$(cpf)	\$(categoria)

**Palette**

**Report Elements**

- Break
- Crosstab
- Frame
- Image
- Generic Element
- Spider Chart
- Line
- Rectangle
- Chart
- Ellipse
- Html
- Barcode
- List
- Table
- Map
- Round Rectangle

**No Properties**

<No Properties>

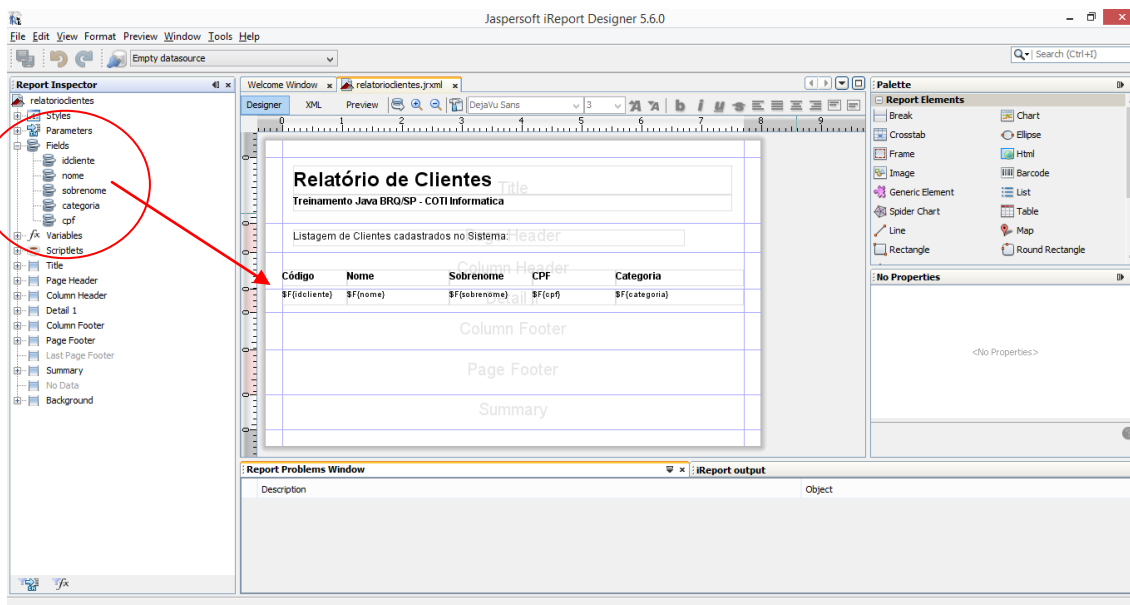
**Report Problems Window**

Description

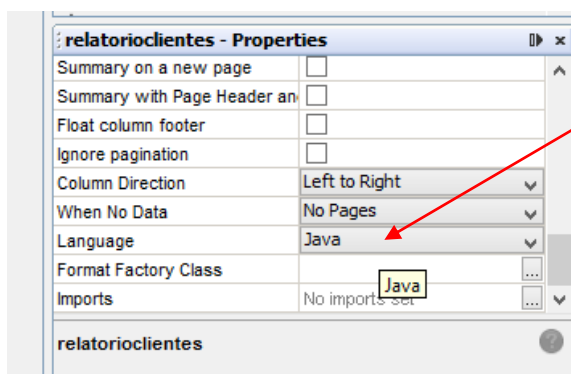
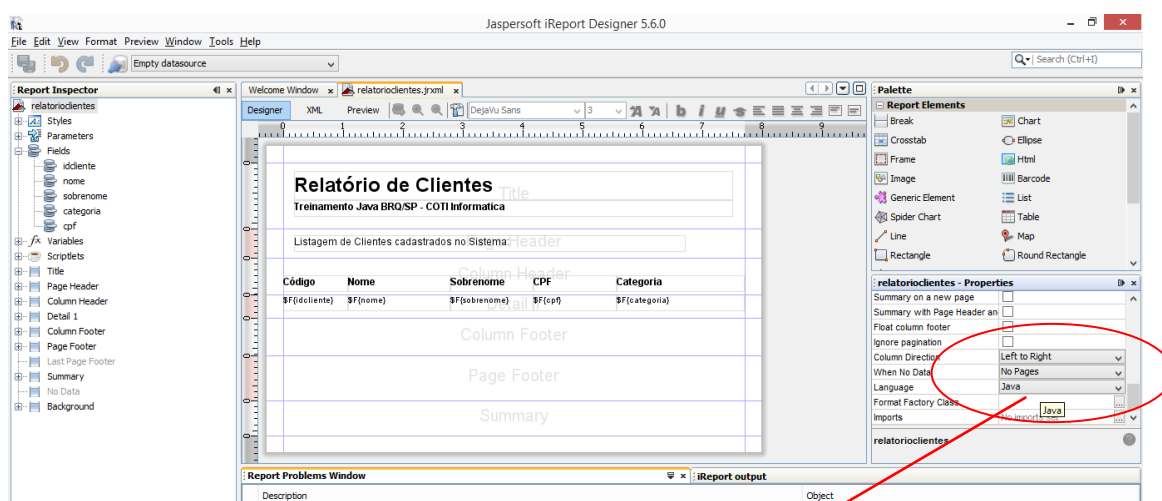
**iReport output**

Object

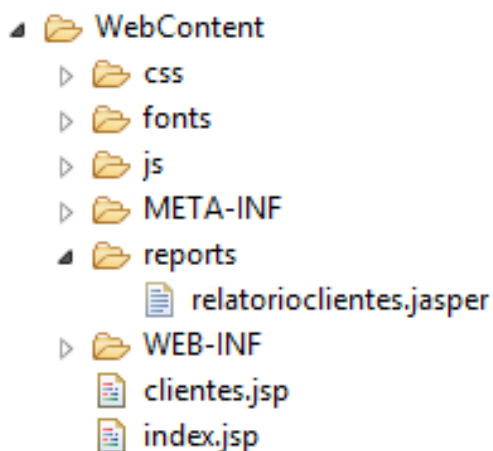
## Incluindo os campos:



## Marque no relatório a opção Java:



## Copie para dentro do projeto somente o arquivo JASPER



## Página:



## Servlet para cadastro do cliente e geração do relatório:

```
package br.com.brq.control;
```

```
import java.io.IOException;  
import java.io.InputStream;  
import java.io.OutputStream;
```

```
import javax.servlet.ServletContext;  
import javax.servlet.ServletException;
```

```
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import br.com.brq.entities.Cliente;
import br.com.brq.entities.types.Categoria;
import br.com.brq.persistence.DAOCliente;
import br.com.brq.persistence.HibernateUtil;
import net.sf.jasperreports.engine.JasperPrint;
import net.sf.jasperreports.engine.JasperRunManager;

@WebServlet("/ControleCliente")
public class ControleCliente extends HttpServlet {
    private static final long serialVersionUID = 1L;

    public ControleCliente() {
        super();
    }

    protected void execute(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

        String action = request.getParameter("action");

        if("cadastrar".equalsIgnoreCase(action)){
            try{

                Cliente c = new Cliente(); //entidade..

                c.setNome(request.getParameter("nome"));
                c.setSobrenome(request.getParameter("sobrenome"));
                c.setCpf(request.getParameter("cpf"));
                c.setCategoria(Categoria.valueOf(request.getParameter("categoria")));

                DAOCliente d = new DAOCliente();
                d.saveOrUpdate(c); //gravando..

                request.setAttribute("mensagem", "Cliente " + c.getNome()
                    + ", cadastrado com sucesso.");
            }
            catch(Exception e){
                request.setAttribute("mensagem", e.getMessage());
            }
            finally{
                request.getRequestDispatcher("clientes.jsp")
                    .forward(request, response);
            }
        }
    }
}
```

```
}
else if("relatorio".equalsIgnoreCase(action)){

    try{
        //pegando o caminho onde esta o relatorio..
        ServletContext context = getServletContext();
        InputStream input = (InputStream)
            context.getResourceAsStream
                ("/reports/relatorioclientes.jasper");

        //fornecer uma conexão de banco de dados para o relatorio..
        byte[] pdf = JasperRunManager.runReportToPdf(input, null,
            HibernateUtil.getSessionFactory()
                .openSession().connection());

        //download..
        response.setHeader("Content-Disposition",
            "filename=relatorio.pdf");
        response.setContentLength(pdf.length);

        OutputStream out = response.getOutputStream();
        out.write(pdf); //download..
        out.flush();
        out.close();
    }
    catch(Exception e){
        e.printStackTrace();
    }
}

protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    execute(request, response);
}

protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    execute(request, response);
}
}
```