

# EducaCiência FastCode

Fala Galera,

- Artigo: 27/2020 Data: Outubro/2020
- Público Alvo: Desenvolvedores – Iniciantes
- Tecnologia: Java
- Tema: JSP – criando Método Select
- Link: <https://github.com/perucello/DevFP>

Criaremos uma série de artigos para explanarmos o CRUD em um projeto Java Server Page ou JSP.

Neste artigo 27/2020 traremos uma maneira simples de criar o Método Select em um projeto JSP, e daremos continuidade no CRUD nos demais artigos ( 28,29,30).

Para este ambiente , criaremos um Banco de Dados onde chamaremos de EducaJSP.

Nosso ambiente consiste em:

⇒ Banco de Dados MySql

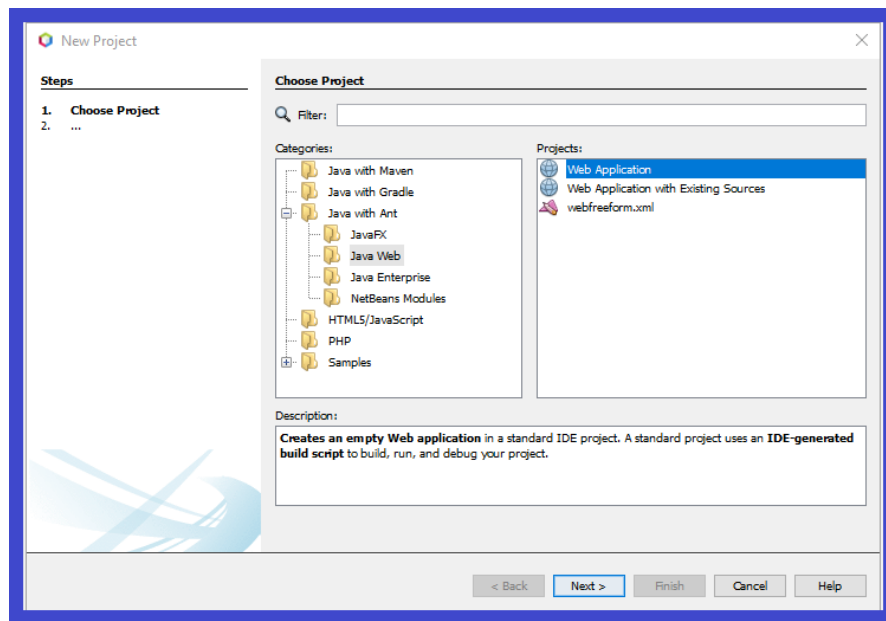
```
1
2 • create database EducaJSP;
3 • use EducaJSP;
4 • CREATE TABLE pessoa (
5     id INT auto_increment PRIMARY KEY not null,
6     nome VARCHAR(50),
7     endereco VARCHAR(50),
8     cidade VARCHAR(50),
9     estado char(2),
10    email VARCHAR(50));
11
12 • insert into pessoa value(
13     '1',
14     'Fulano de Tal',
15     'Rua das Hortas',
16     'Sao Paulo',
17     'SP',
18     'fulano@detal.com.br'
19 );
20
21 • select * from pessoa;
```

Result Grid

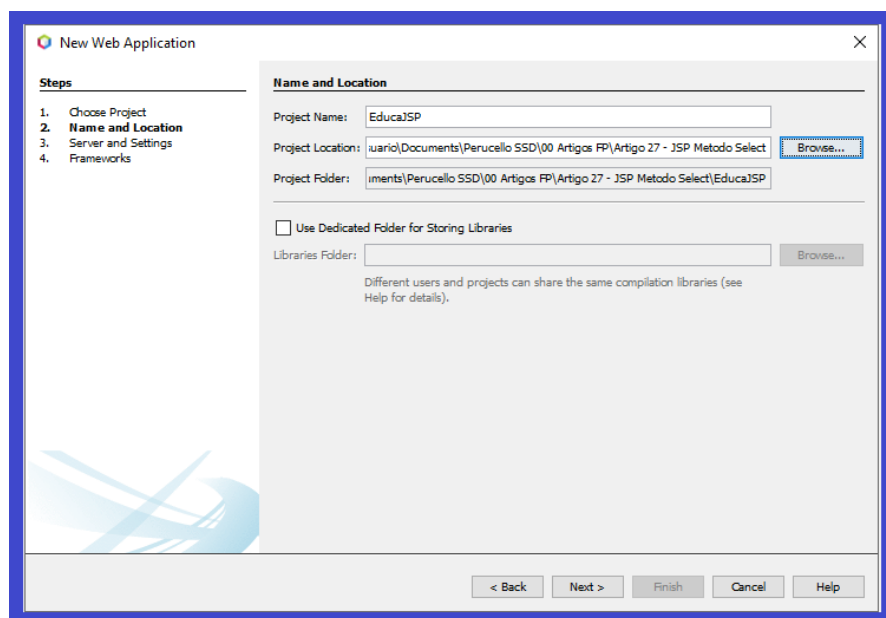
id	nome	endereco	cidade	estado	email
1	Fulano de Tal	Rua das Hortas	Sao Paulo	SP	fulano@detal.com.br
*	NULL	NULL	NULL	NULL	NULL

Utilizaremos da IDE Netbeans 11 para elaborarmos nosso propósito.  
Para isso, criaremos nosso projeto JSP que se chamará EducaJSP, vamos seguir os seguintes passos:

⇒ Novo Projeto



⇒ Usaremos Servidor Glassfish 4.1.1



New Web Application

**Steps**

1. Choose Project
2. Name and Location
3. **Server and Settings**
4. Frameworks

**Server and Settings**

Add to Enterprise Application: <None>

Server: GlassFish Server Add...

Java EE Version: Java EE 7 W... Note: Source Level 7 will be set for Java EE 7 project.

Context Path: /EducaJSP

< Back Next > Finish Cancel Help

New Web Application

**Steps**

1. Choose Project
2. Name and Location
3. Server and Settings
4. **Frameworks**

**Frameworks**

Select the frameworks you want to use in your web application.

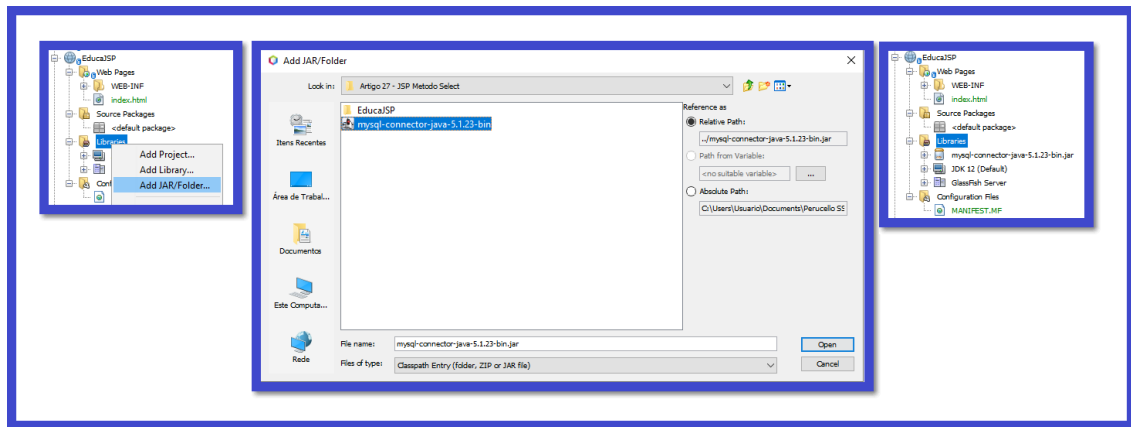
☐ Spring Web MVC

☐ JavaServer Faces

☐ Struts 1.3.10

< Back Next > Finish Cancel Help

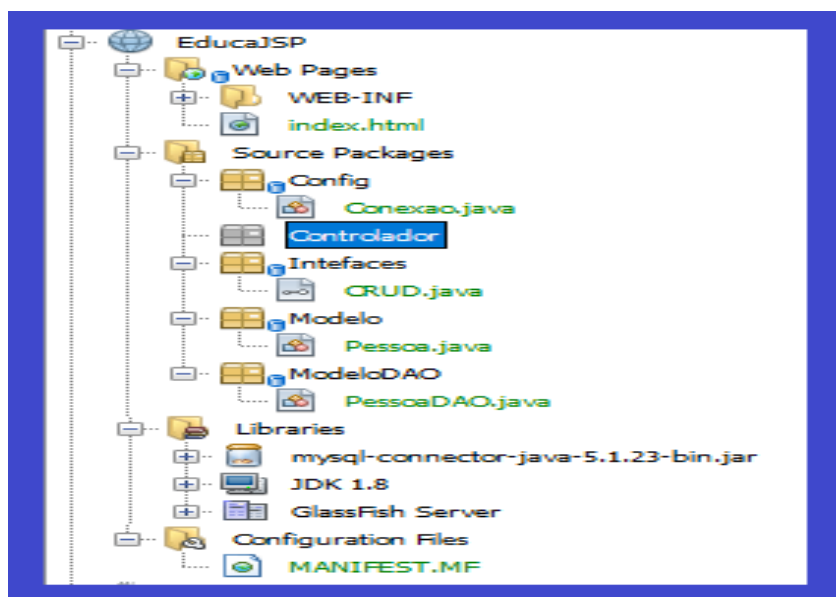
⇒ Para conexão com Banco de Dados , utilizaremos do driver jdbc e já adicionaremos ao nosso Projeto como ilustração abaixo:



⇒ **Como meu projeto foi criado em Java12 , nesse momento alterei para Java 8**

⇒ Agora, iremos criar nossos pacotes sendo:

- Config** => teremos nesse pacote nossa classe chamada Conexao.java que será responsável pela conexão com Banco de Dados
- Controlador** => teremos nosso Controlador.java que será nosso Servlet  
**Obs.: O Servlet iremos criar depois, pois iremos explanar duas maneiras de criarmos.**
- Interface** => teremos nossa interface CRUD dos Métodos
- Modelo** => teremos na classe Pessoa.java nossos Construtores e nossos Métodos Getters/Setters
- ModeloDAO** => teremos nossa classe PessoaDAO.java que receberá nossos métodos que atuarão junto ao Banco de Dados



Na nossa classe de Conexão com o Banco de Dados, abordaremos duas maneiras, sendo uma mais detalhada para facilitar o entendimento e outra mais objetiva.

Vamos lá então:

### Conexao.java

```
1 package Config;
2
3 import java.sql.*;
4
5 public class Conexao {
6
7     Connection con;
8
9     private static final String DATABASE = "EducaJSP";
10    private static final String PORT = "3306";
11    private static final String HOST = "jdbc:mysql://localhost:" + "/" + DATABASE;
12    private static final String DRIVER = "com.mysql.jdbc.Driver";
13    private static final String SSL = "?useTimezone=true&serverTimezone=UTC&useSSL=false";
14    private static final String URL = "jdbc:mysql://localhost:3306" + "/" + DATABASE + SSL;
15    private static final String USR = "root";
16    private static final String PWD = "";
17
18    public void Conexao() {
19        try {
20            Class.forName(DRIVER);
21            con = DriverManager.getConnection(URL, USR, PWD);
22            System.out.println("Banco de Dados => " + DATABASE);
23            System.out.println("Host => " + HOST);
24            System.out.println("Porta => " + PORT);
25            System.out.println("Driver => " + DRIVER);
26            System.out.println("URL => " + URL);
27            System.out.println("SSL => " + SSL);
28            System.out.println("Usuario => " + USR);
29            System.out.println("Senha => " + PWD);
30        } catch (ClassNotFoundException | SQLException e) {
31            System.err.println("Error = " + e);
32            System.out.println("Erro de conexao com Banco de Dados - verificar se está ativo - Mysql ! " + e);
33        }
34    }
35
36    public static void Desconectar(Connection con) {
37        try {
38            if (con != null) {
39                con.close();
40            }
41        } catch (SQLException e) {
42            System.out.println("ERRO : " + e.getMessage());
43        }
44    }
45
46    public Connection getConnection() {
47        return con;
48    }
49 }
```

## Pessoa.java

```

1 package Modelo;
2
3 public class Pessoa {
4     int id;
5     String email;
6     String nome;
7     String endereco;
8     String cidade;
9
10    public Pessoa() {
11    }
12
13    public Pessoa(int id, String email, String nome, String endereco, String cidade) {
14        this.id = id;
15        this.email = email;
16        this.nome = nome;
17        this.endereco = endereco;
18        this.cidade = cidade;
19    }
20
21    public int getId() {
22        return id;
23    }
24
25    public void setId(int id) {
26        this.id = id;
27    }
28
29    public String getEmail() {
30        return email;
31    }
32
33    public void setEmail(String email) {
34        this.email = email;
35    }
36
37    public String getNome() {
38        return nome;
39    }
40
41    public void setNome(String nome) {
42        this.nome = nome;
43    }
44
45    public String getEndereco() {
46        return endereco;
47    }
48
49    public void setEndereco(String endereco) {
50        this.endereco = endereco;
51    }
52
53    public String getCidade() {
54        return cidade;
55    }
56
57    public void setCidade(String cidade) {
58        this.cidade = cidade;
59    }
60
61 }
62
63

```

## PessoaDAO.java

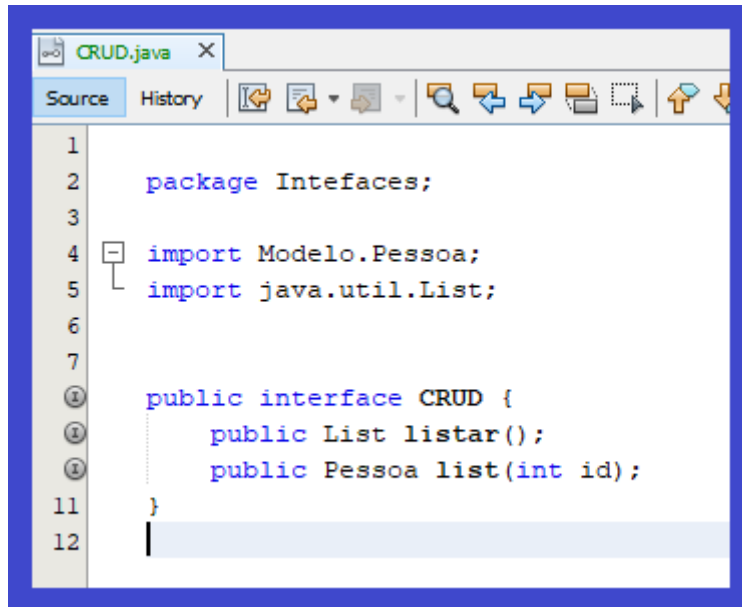
```

1 package ModeloDAO;
2
3 import ...9 lines
4
5 public class PessoaDAO implements CRUD {
6     Conexao cn = new Conexao();
7     Connection con;
8     PreparedStatement ps;
9     ResultSet rs;
10    Pessoa a = new Pessoa();
11
12    public PessoaDAO() {
13        super();
14        con = this.cn.getConnection();
15    }
16
17    @Override
18    public List listar() {
19        cn = new Conexao();
20        ArrayList<Pessoa> list = new ArrayList<>();
21        String sql = "select * from pessoa";
22        try {
23            con = cn.getConnection();
24            ps = con.prepareStatement(sql);
25            rs = ps.executeQuery();
26            while (rs.next()) {
27                Pessoa a = new Pessoa();
28                a.setId(rs.getInt("Id"));
29                a.setNome(rs.getString("Nome"));
30                a.setEndereco(rs.getString("Endereco"));
31                a.setCidade(rs.getString("Cidade"));
32                a.setEmail(rs.getString("Email"));
33                list.add(a);
34            }
35        } catch (SQLException e) {
36        } finally {
37            cn.Desconectar(con);
38        }
39
40        return list;
41    }
42
43    @Override
44    public Pessoa list(int id) {
45        cn = new Conexao();
46        String sql = "select * from pessoa where Id = " + id;
47        try {
48            con = cn.getConnection();
49            ps = con.prepareStatement(sql);
50            rs = ps.executeQuery();
51            while (rs.next()) {
52                Pessoa a = new Pessoa();
53                a.setId(rs.getInt("Id"));
54                a.setNome(rs.getString("Nome"));
55                a.setEndereco(rs.getString("Endereco"));
56                a.setCidade(rs.getString("Cidade"));
57                a.setEmail(rs.getString("Email"));
58            }
59        } catch (SQLException e) {
60        } finally {
61            cn.Desconectar(con);
62        }
63
64        return a;
65    }
66
67 }
68
69
70
71
72
73
74
75
76

```

## CRUD.java

⇒ Interface



```
1 package Intefaces;
2
3
4 import Modelo.Pessoa;
5 import java.util.List;
6
7
8 public interface CRUD {
9     public List listar();
10    public Pessoa list(int id);
11 }
12
```

## Chegamos ao nosso Servlet !

Antes de criarmos nossa classe Servlet , vamos dar um overview sobre o que é Servlet.

### Servlet

**Servlet** => Podemos simplificar o Servlet como uma classe escrita em Java que contém código HTML para geração de páginas web, é uma resposta à antiga técnica de desenvolvimento de aplicações web denominada **Common Gateway Interface** ou CGI.

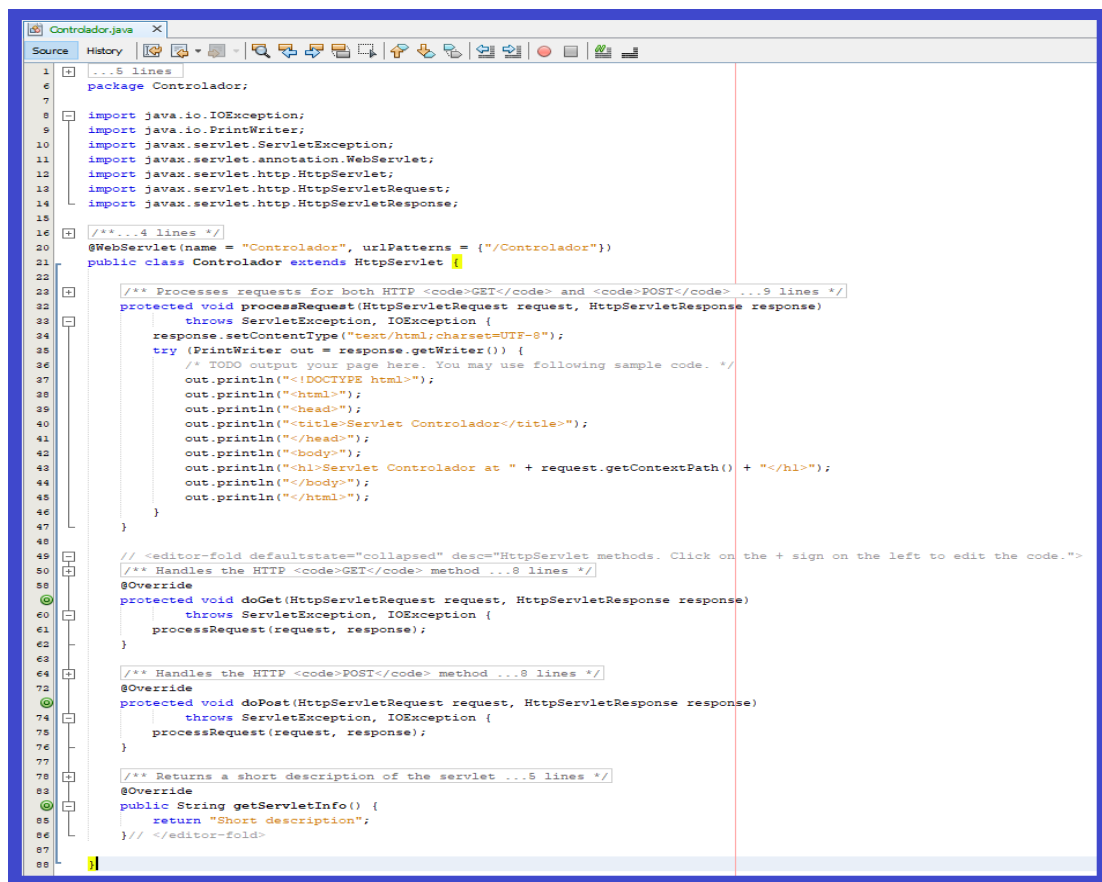
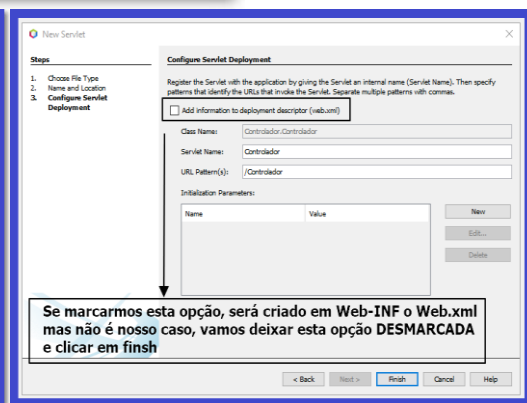
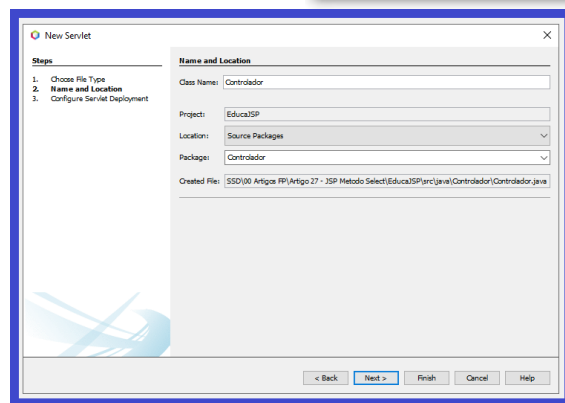
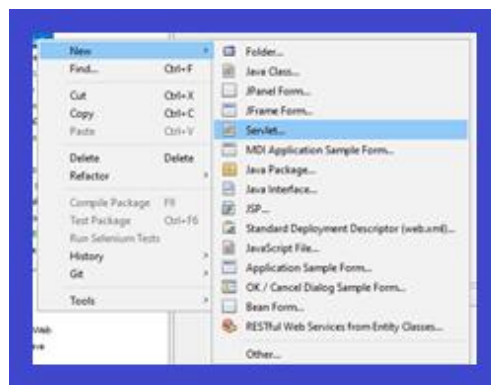
Java

HTML

Neste cenário, ao criarmos nosso Servlet, optamos em duas maneiras sendo:

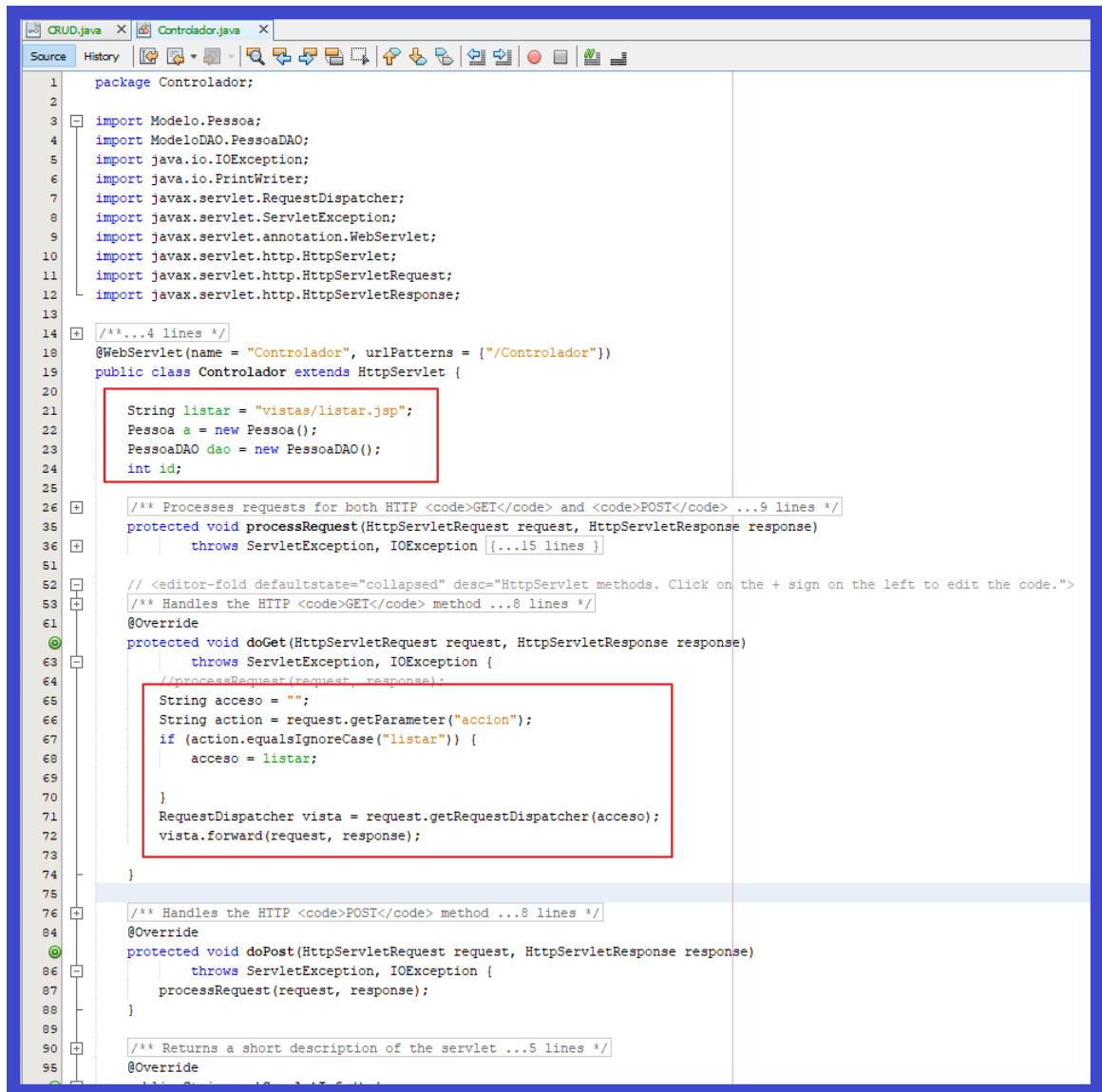
- a) Criando com web.xml no pacote Web-INF ou
- b) Se anotamos no Controlador.

Iremos optar por anotar , e vamos seguir os seguintes passos:





O único método que manipularemos será o doGet



```
1 package Controlador;
2
3 import Modelo.Pessoa;
4 import ModeloDAO.PessoaDAO;
5 import java.io.IOException;
6 import java.io.PrintWriter;
7 import javax.servlet.RequestDispatcher;
8 import javax.servlet.ServletException;
9 import javax.servlet.annotation.WebServlet;
10 import javax.servlet.http.HttpServlet;
11 import javax.servlet.http.HttpServletRequest;
12 import javax.servlet.http.HttpServletResponse;
13
14 /**...4 lines */
15 @WebServlet(name = "Controlador", urlPatterns = {"/Controlador"})
16 public class Controlador extends HttpServlet {
17
18     String listar = "vistas/listar.jsp";
19     Pessoa a = new Pessoa();
20     PessoaDAO dao = new PessoaDAO();
21     int id;
22
23     /** Processes requests for both HTTP <code>GET</code> and <code>POST</code> ...9 lines */
24     protected void processRequest(HttpServletRequest request, HttpServletResponse response)
25         throws ServletException, IOException {
26
27     }
28
29     /** <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">
30     /** Handles the HTTP <code>GET</code> method ...8 lines */
31     @Override
32     protected void doGet(HttpServletRequest request, HttpServletResponse response)
33         throws ServletException, IOException {
34         //processRequest(request, response);
35
36         String acesso = "";
37         String action = request.getParameter("accion");
38         if (action.equalsIgnoreCase("listar")) {
39             acesso = listar;
40
41         }
42         RequestDispatcher vista = request.getRequestDispatcher(acesso);
43         vista.forward(request, response);
44     }
45
46     /** Handles the HTTP <code>POST</code> method ...8 lines */
47     @Override
48     protected void doPost(HttpServletRequest request, HttpServletResponse response)
49         throws ServletException, IOException {
50         processRequest(request, response);
51     }
52
53     /** Returns a short description of the servlet ...5 lines */
54     @Override
55     public String getServletInfo() {
56         return null;
57     }
58 }
```

Feito !

Agora vamos manipular o JSP.

Iniciaremos pela nossa Index.jsp

```

1  index.jsp
2  Source History
3  <%@page contentType="text/html" pageEncoding="UTF-8"%>
4  <!DOCTYPE html>
5  <html>
6  <head>
7  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
8  <title>JSP EducaCiencia FastCode</title>
9  </head>
10 <body>
11 <br>
12 <br>
13 <div class="container">
14 <h1>EducaCiencia FastCode</h1>
15 <h1></h1>
16 <a class="btn btn-success btn-lg" href="Controlador?accion=listar">EducaCiencia FastCode - CRUD</a>
17 </div>
18 </body>
19 </html>

```

Agora Listar.jsp

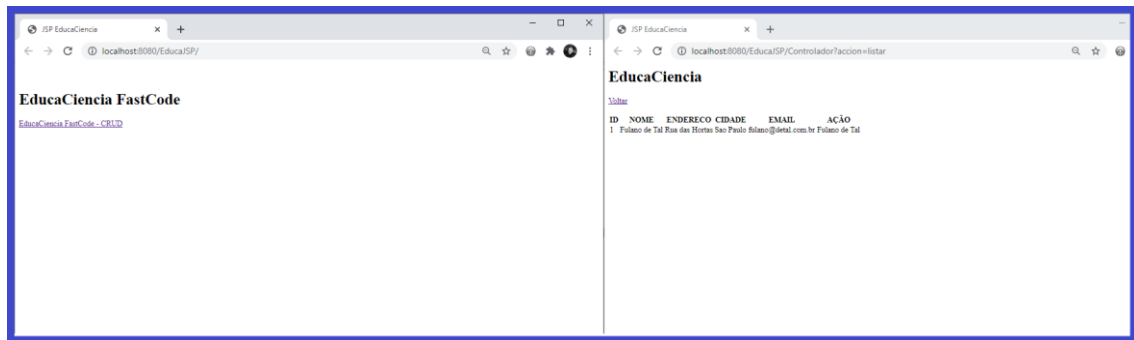
```

1  listar.jsp
2  Source History
3  <%@page import="java.util.Iterator"%>
4  <%@page import="Modelo.Pessoa"%>
5  <%@page import="java.util.List"%>
6  <%@page import="ModeloDAO.PessoaDAO"%>
7  <%@page contentType="text/html" pageEncoding="UTF-8"%>
8  <!DOCTYPE html>
9  <html>
10 <head>
11 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
12 <title>JSP EducaCiencia</title>
13 </head>
14 <body>
15 <div class="container">
16 <h1>EducaCiencia</h1>
17 <a class="btn btn-success" href="Controlador?accion=add">Adicionar</a>
18 <a class="btn btn-success" href="index.jsp">Voltar</a>
19 <br>
20 <br>
21 <table class="table table-bordered">
22 <thead>
23 <tr>
24 <th class="text-center">ID</th>
25 <th class="text-center">EMAIL</th>
26 <th class="text-center">NOME</th>
27 <th class="text-center">ENDERECO</th>
28 <th class="text-center">CIDADE</th>
29 <th class="text-center">EMAIL</th>
30 <th class="text-center">AÇÃO</th>
31 </tr>
32 </thead>
33 <tbody>
34 <tr>
35 <td class="text-center"><%= a.getId() %></td>
36 <td class="text-center"><%= a.getNome() %></td>
37 <td class="text-center"><%= a.getEndereco() %></td>
38 <td class="text-center"><%= a.getCidade() %></td>
39 <td class="text-center"><%= a.getEmail() %></td>
40 <td class="text-center"><%= a.getNome() %></td>
41 <td class="text-center"><%= a.getNome() %></td>
42 </tr>
43 </tbody>
44 </table>
45 </div>
46 </body>

```

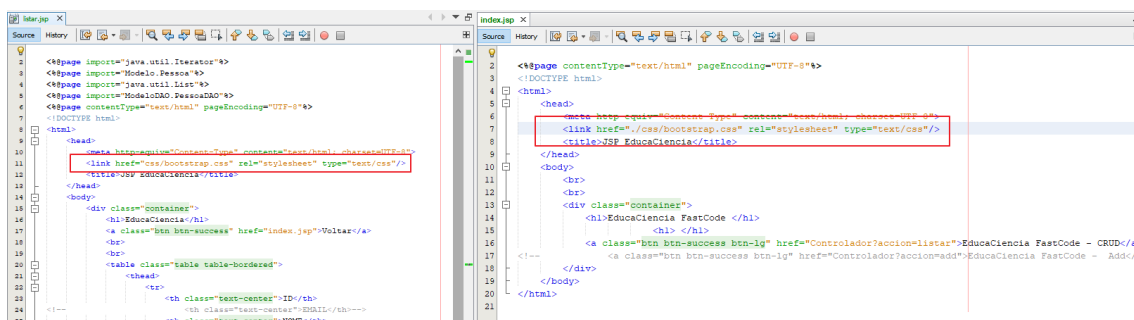
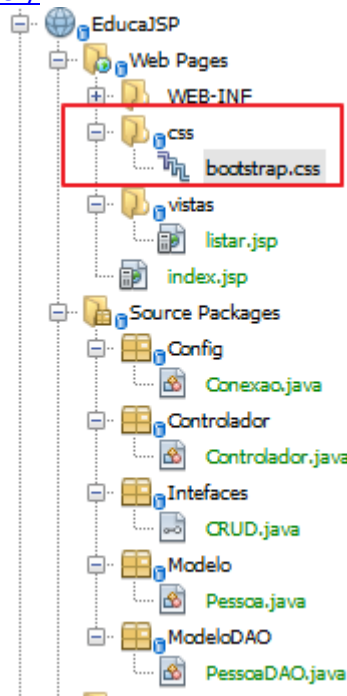


Feito isso, se tudo estiver OK, basta iniciarmos a aplicação em RUN.



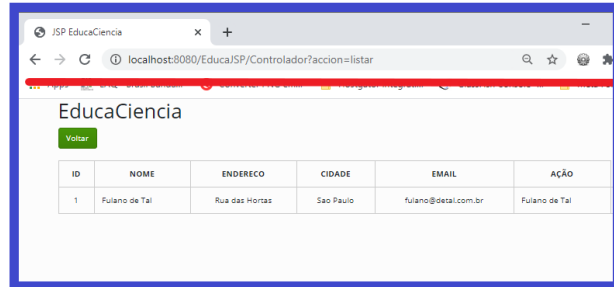
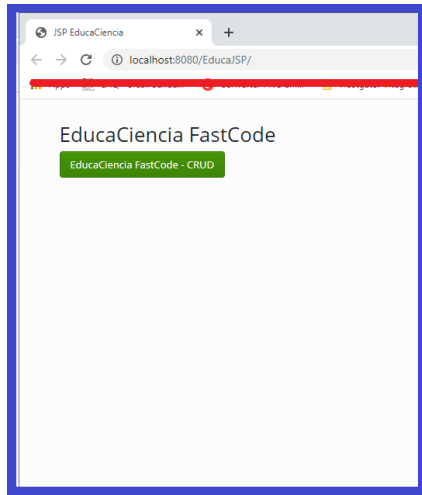
Vemos que funcionou, porém, vamos add um css para melhorar nosso layout.

<https://bootswatch.com/simplex/>





Agora vamos testar novamente !



Nota-se que os códigos funcionaram corretamente, no entanto, saliento que este artigo 27/2020 são baseados no curso de Java que ministro na Escola Evolua – Ensino Profissionalizante e como proposito de ajuda à comunidade, estamos trazendo parte da didática em forma de artigo comunitário e assim podemos contribuir com a comunidade Tecnológica como um todo.

Agradeço imensamente a Diretoria da Escola Evolua de Sumaré.

Os códigos estarão disponíveis no Git.

Até mais !

Espero ter ajudado !

