

## Exercício Dirigido

- 04 -

Procure ser um homem de valor, em vez de ser um homem de sucesso.  
Albert Einstein

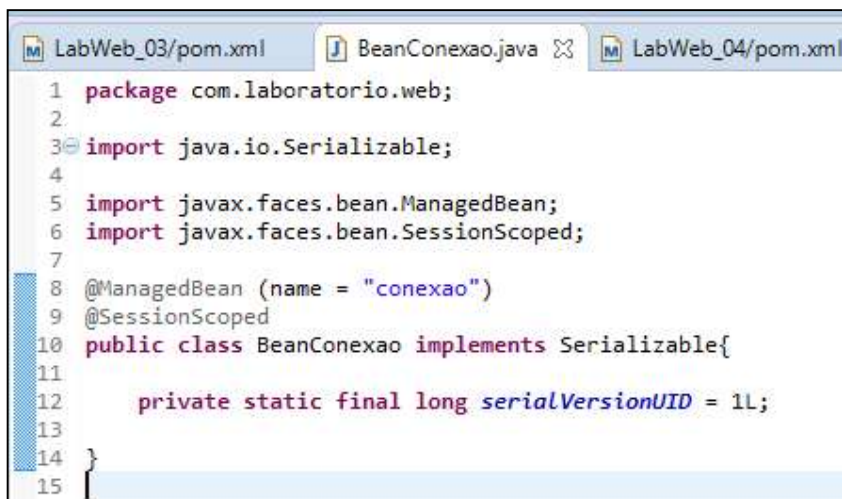
### Integração com Banco de Dados e aplicações Java Web com JavaServer Faces

**Observação:** Para esse exercício deve existir uma implementação de software de fonte aberta das tecnologias Java Servlet e JavaServer Pages Apache Tomcat <sup>TM</sup> funcional, caso não exista faça a configuração seguindo o Exercício Dirigido 01.

01) Usando o workspace com o nome de **Laboratórios**, crie um novo projeto do tipo **Maven Project**, esse projeto deve seguir os seguintes requisitos:

- **Tipo:** Simple Project (Projeto Simples)
- **Nome:** LabWeb\_04
- **Group Id:** br.com.laboratorio
- **Empacotamento:** war
- **Suporte a caracteres:** UTF-8
- **Versão do plugin maven:** 3.0
- **Versão do Java:** 1.8
- **Deve ser configurado o suporte:** JSF
- **O servlet JSF deve tratar as páginas:** xhtml
- **Descritor web:** Versão do descritor de 2.5 para 3.0
- **Configure como página principal da aplicação:** index.xhtml
- **Usando o Maven adicione as bibliotecas (jars):** JSF
- **Adicione os seguintes arquivos xhtml:** index.xhtml, sucesso.xhtml, erro.xhtml

02) Adicione uma classe Java com o nome de BeanConexao no pacote com.laboratorio.web, faça a serialização da classe, essa classe deve ser um bean gerenciado com o escopo de sessão, deixe o código da classe igual a figura abaixo.



```
1 package com.laboratorio.web;
2
3 import java.io.Serializable;
4
5 import javax.faces.bean.ManagedBean;
6 import javax.faces.bean.SessionScoped;
7
8 @ManagedBean (name = "conexao")
9 @SessionScoped
10 public class BeanConexao implements Serializable{
11
12     private static final long serialVersionUID = 1L;
13
14 }
15
```

03) Adicione uma classe Java com o nome de ConnectionFactory no pacote com.laboratorio.util, deixe o código da classe igual a figura abaixo.

```
ConnectionFactory.java
1 package com.laboratorio.util;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5
6 public class ConnectionFactory {
7
8     private static final String DRIVER = "";
9     private static final String URL = "";
10    private static final String USER = "";
11    private static final String PASSWORD = "";
12
13    public static Connection getConnection(){
14        try {
15            Class.forName(DRIVER);
16            return DriverManager.getConnection(URL, USER, PASSWORD);
17        } catch (Exception e){
18            System.err.println("Erro na conexão");
19        }
20        return null;
21    }
22
23
24 }
25
```

04) Adicione um método com o nome de actionConexao(), na classe BeanConexao esse método deve ser publico e retornar uma String, conforme a figura abaixo:

```
ConnectionFactory... BeanConexao... erro.xhtml index.xhtml
1 package com.laboratorio.web;
2
3 import java.io.Serializable;
4
5
6 @ManagedBean(name = "conexao")
7 @SessionScoped
8 public class BeanConexao implements Serializable {
9
10    private static final long serialVersionUID = 1L;
11
12    public String actionConexao(){
13        if (ConnectionFactory.getConnection() != null){
14            return "sucesso";
15        }
16        return "erro";
17    }
18 }
19
```

05) Deixe o código do arquivo **erro.xhtml**, igual a figura abaixo:

```
BeanConexao.j... erro.xhtml index.xhtml sucesso.xhtml
1 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
2 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
3
4 <html xmlns="http://www.w3.org/1999/xhtml"
5       xmlns:h="http://java.sun.com/jsf/html">
6
7 <h:head>
8     <title>Laboratório Web - versão 4.0</title>
9 </h:head>
10 <h:body>
11     erro
12 </h:body>
13 </html>
```

06) Deixe o código do arquivo **sucesso.xhtml**, igual a figura abaixo:




```
1 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
2 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
3
4 <html xmlns="http://www.w3.org/1999/xhtml"
5       xmlns:h="http://java.sun.com/jsf/html">
6 <h:head>
7   <title>Laboratório Web - versão 4.0</title>
8 </h:head>
9 <h:body>
10  sucesso
11 </h:body>
12 </html>
```

07) Deixe o código do arquivo **index.xhtml**, igual a figura abaixo:



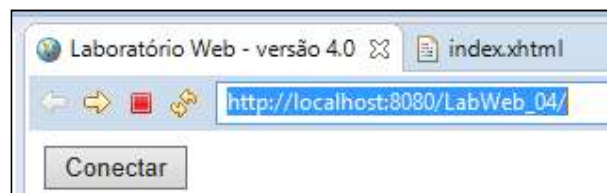
```
1 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
2 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
3
4 <html xmlns="http://www.w3.org/1999/xhtml"
5       xmlns:h="http://java.sun.com/jsf/html">
6
7 <h:head>
8   <title>Laboratório Web - versão 4.0</title>
9 </h:head>
10 <h:body>
11   <h:form>
12     <h:commandButton value="Conectar" action="#{conexao.actionConexao}" />
13   </h:form>
14 </h:body>
15 </html>
```

08) Usando o Maven adicione o Driver do SGBD PostgreSQL, use o trecho de código, destacado na figura abaixo no arquivo pom.xml.

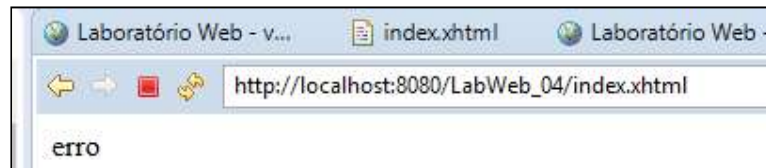


```
40
27 <dependencies>
28 <dependency>
29   <groupId>com.sun.faces</groupId>
30   <artifactId>jsf-api</artifactId>
31   <version>2.2.12</version>
32 </dependency>
33 <dependency>
34   <groupId>com.sun.faces</groupId>
35   <artifactId>jsf-impl</artifactId>
36   <version>2.2.12</version>
37 </dependency>
38 <dependency>
39   <groupId>org.postgresql</groupId>
40   <artifactId>postgresql</artifactId>
41   <version>9.4-1201-jdbc41</version>
42 </dependency>
43 </dependencies>
44
```

09) Execute o seu projeto, e clique no botão **Conectar**, figura abaixo:



10) Observe que a página erro.xhtml, figura abaixo, foi carregada.



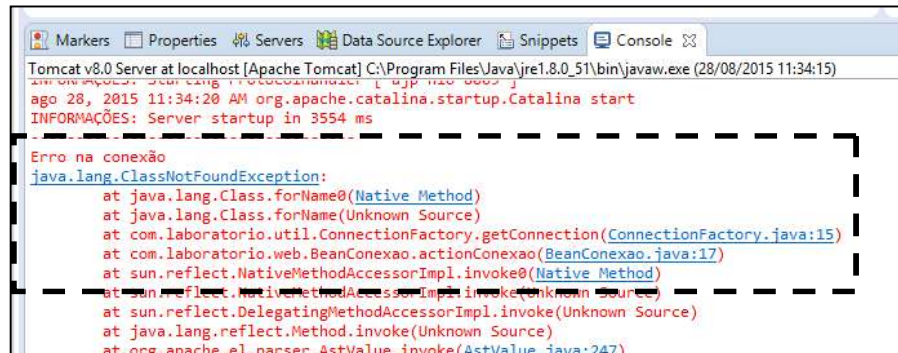
11) Observe a mensagem **Erro na conexão**, mostrada na guia console, figura abaixo:



12) Altere o código o método getConnection() da classe ConnectionFactory, para ficar igual a figura abaixo:

```
public static Connection getConnection(){
    try {
        Class.forName(DRIVER);
        return DriverManager.getConnection(URL, USER, PASSWORD);
    } catch (Exception e){
        System.err.println("-----");
        System.err.println("Erro na conexão");
        e.printStackTrace();
    }
    return null;
}
```

12) Reexecute o seu aplicativo, clique no botão Conectar da página index.xhtml, e observe na guia console a mensagem destacada, na figura abaixo:

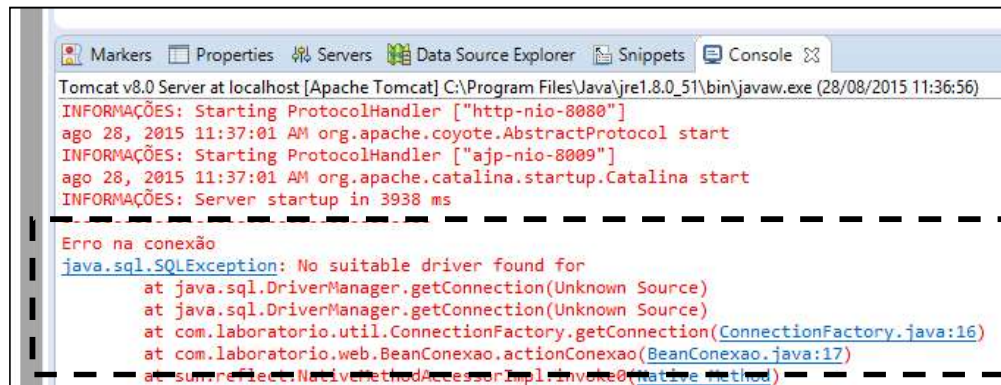


13) Altere o código da constante **DRIVER** da classe ConnectionFactory, para ficar igual a figura abaixo:

```
5
6 public class ConnectionFactory {
7
8     private static final String DRIVER = "org.postgresql.Driver";
9     private static final String URL = "jdbc:postgresql://localhost:5432/db";
```

14) Reexecute o seu aplicativo, clique no botão Conectar da página index.xhtml, e observe na guia console a mensagem destacada, na figura abaixo:

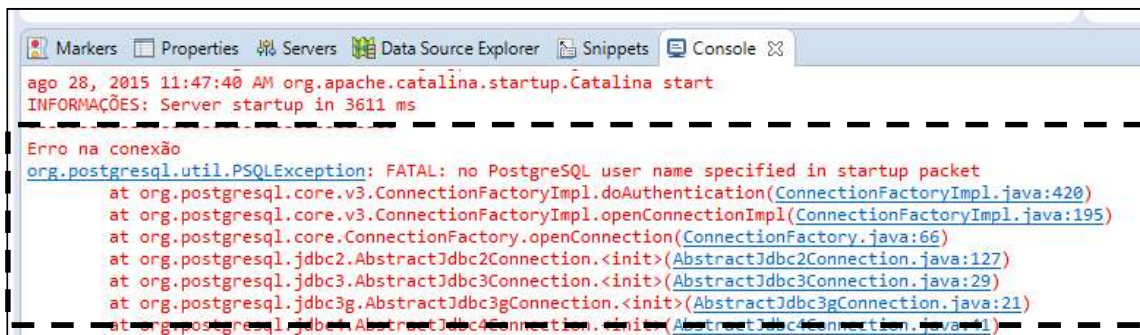




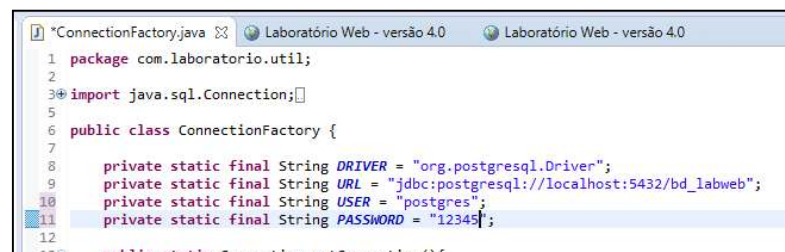
15) Altere o código da constante **DRIVER** da classe **ConnectionFactory**, para ficar igual a figura abaixo:



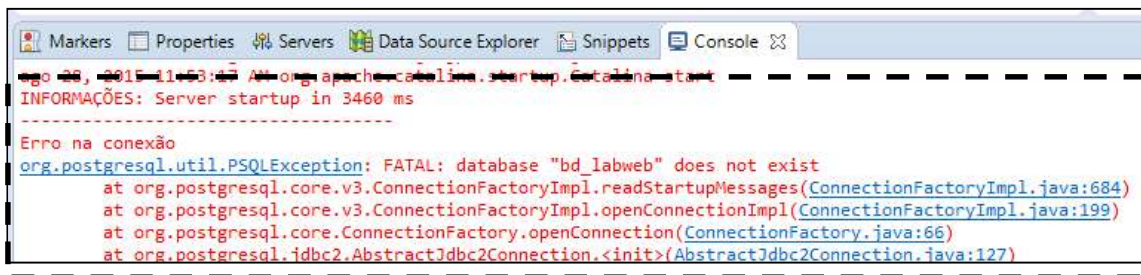
16) Reexecute o seu aplicativo, clique no botão Conectar da página index.xhtml, e observe na guia console a mensagem destacada, na figura abaixo:



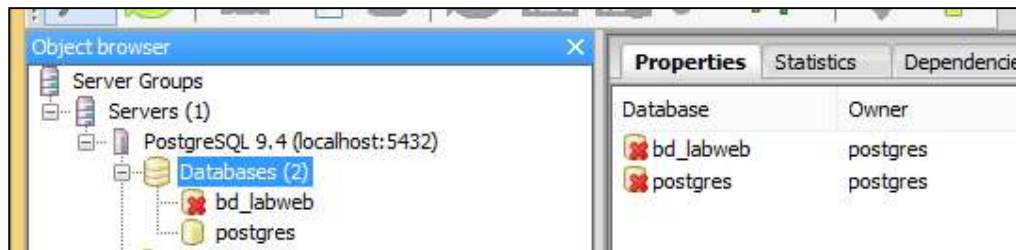
17) Altere o código da constante **USER** e **PASSWORD** da classe **ConnectionFactory**, coloque o nome do usuário e a senha de seu SGBD PostgreSQL. Em uma instalação padrão o usuário e **postgres** e a senha foi definida por **você**, para ficar igual a figura abaixo:



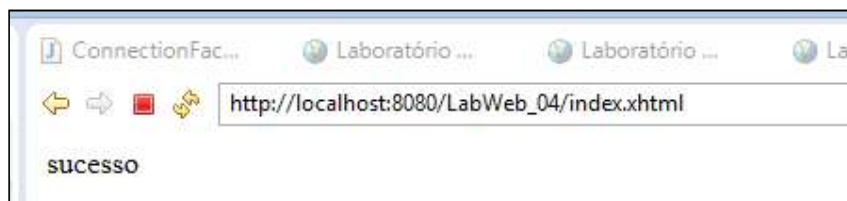
18) Reexecute o seu aplicativo, clique no botão Conectar da página index.xhtml, e observe na guia console a mensagem destacada, na figura abaixo:



19) Abra o pgAdmin III, e crie um banco de dados com o nome bd\_labweb, figura abaixo.



20) Se todos os exercícios acima estiverem corretos, reexecute o seu aplicativo, clique no botão Conectar da página index.xhtml, e observe na guia console que nenhuma mensagem de erro será mostrada e a página sucesso.xhtml será carregada em seu navegador, figura abaixo:



21) Caso não tenha sucesso, refaça o exercício prestando atenção nos seguintes pontos.

- Carregamento dos jars pelo Maven, o arquivo pom.xml encontra-se nos anexos.
- Digitação dos códigos.
- Digitação dos valores das constantes da classe ConnectionFactory.

**Anexo**

Arquivo pom.xml do projeto.

```
1. <project xmlns="http://maven.apache.org/POM/4.0.0"
2.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3.     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
4.       http://maven.apache.org/xsd/maven-4.0.0.xsd">
5.   <modelVersion>4.0.0</modelVersion>
6.   <groupId>br.com.laboratorio</groupId>
7.   <artifactId>LabWeb_04</artifactId>
8.   <version>0.0.1-SNAPSHOT</version>
9.   <packaging>war</packaging>
10.  <properties>
11.    <project.build.sourceEncoding>
12.      UTF-8
13.    </project.build.sourceEncoding>
14.  </properties>
15.
16.  <build>
17.    <plugins>
18.      <plugin>
19.        <artifactId>maven-compiler-plugin</artifactId>
20.        <version>3.0</version>
21.        <configuration>
22.          <source>1.8</source>
23.          <target>1.8</target>
24.        </configuration>
25.      </plugin>
26.    </plugins>
27.  </build>
28.
29.  <dependencies>
30.    <dependency>
31.      <groupId>com.sun.faces</groupId>
32.      <artifactId>jsf-api</artifactId>
33.      <version>2.2.12</version>
34.    </dependency>
35.    <dependency>
36.      <groupId>com.sun.faces</groupId>
37.      <artifactId>jsf-impl</artifactId>
38.      <version>2.2.12</version>
39.    </dependency>
40.    <dependency>
41.      <groupId>org.postgresql</groupId>
42.      <artifactId>postgresql</artifactId>
43.      <version>9.4-1201-jdbc41</version>
44.    </dependency>
45.
46.  </dependencies>
47. </project>
```