Instruções: *Todos os programas devem ser resolvidos utilizando os conceitos de Programação Orientada a Objetos, a linguagem Java™ e os conceitos de Java Persistence API.*

09 - ORGANIZAÇÃO DE VERSÕES

O POM, cuja sigla significa **Project Object Model**, é o principal ponto de partida de um projeto que será gerenciado pelo Maven, pois é o arquivo (pom.xml) que descreve e disponibiliza as informações e configurações do projeto.

1. Configuração de Propriedades

1.1. Retorne ao *Eclipse* e abra o arquivo pom.xml.

1.2. Observe que temos a versão **2.2.2** do JSF em comum para *jsf-api* e *jsf-impl*:

```
<dependencies>
	<dependency>
		<groupId>com.sun.faces</groupId>
		<artifactId>jsf-api</artifactId>
		<version>2.2.2</version>
	</dependency>
	<dependency>
		<groupId>com.sun.faces</groupId>
		<artifactId>jsf-impl</artifactId>
		<version>2.2.2</version>
	</dependency>
</dependencies>
```

1.3. Para melhorar a manutenção, crie a propriedade abaixo:

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
	<modelVersion>4.0.0</modelVersion>
	<groupId>br.com.etechoracio</groupId>
	<artifactId>etectour</artifactId>
	<version>0.0.1</version>
	<packaging>war</packaging>

	<properties>
		<jsf.version>2.2.2</jsf.version>
	</properties>
```

1.4. Atualize a versão do JSF utilizando a propriedade:

```
<dependencies>
	<dependency>
		<groupId>com.sun.faces</groupId>
		<artifactId>jsf-api</artifactId>
		<version>${jsf.version}</version>
	</dependency>
	<dependency>
		<groupId>com.sun.faces</groupId>
		<artifactId>jsf-impl</artifactId>
		<version>${jsf.version}</version>
	</dependency>
</dependencies>
```

Instruções: *Todos os programas devem ser resolvidos utilizando os conceitos de Programação Orientada a Objetos, a linguagem Java™ e os conceitos de Java Persistence API.*

1.5. Pressione `Ctrl` + `Shift` + `F` para identar o arquivo e salve-o.

1.6. Atualize o projeto, clique com o botão direito no projeto e acesse ***Maven*** ➔ ***Update Project***.

1.7. Acesse a pasta **C:/Java-Projeto/etectour** (via **Prompt de Comando**).

1.8. Verifique a situação dos arquivos no repositório Git.

```
git status
```

1.9. Faça com que os arquivos sejam rastreados pelo Git.

```
git add .
```

1.10. Verifique a situação dos arquivos no repositório Git novamente.

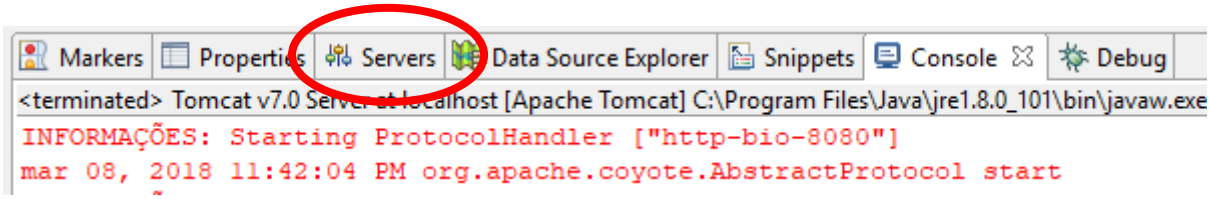
1.11. Execute o **commit** para gravar as mudanças no repositório com a mensagem **"Configuração de propriedades"**.

1.12. Efetue um **push** para enviar os arquivos ao repositório remoto.

1.13. Abra a página web do ***Github*** e verifique se o arquivo `pom.xml` foi atualizado (seção **commits**).

2. Testando o projeto

2.1. Na parte inferior do Eclipse, abra a *aba Servers*.



2.2. Clique na opção No servers are available. Click this link to create a new server...

2.3. Selecione o **Apache Tomcat7.0** e clique em **Next**.

2.4. Na próxima tela, selecione o diretório de instalação do Tomcat. A localização do diretório será: **C:\Arquivos de Programas\Apache Software Foundation\Tomcat 7.0**.

2.5. Clique em **Finish.**

2.6. Abra a pasta **\\Rede\Professor\PCII\exemplos** e copie o arquivo `index.xhtml`.

Instruções: *Todos os programas devem ser resolvidos utilizando os conceitos de Programação Orientada a Objetos, a linguagem Java™ e os conceitos de Java Persistence API.*

2.7. Acesse o **Eclipse** e navegue até a pasta **src/main/webapp** e cole o arquivo copiado.

2.8. Acesse o menu **Window** ➔ **Web Browser** ➔ **Default system web browser** ou selecione o navegador desejado para executar a aplicação.

2.9. Execute a página `index.xhtml`.