

Arquitetura MVC

Prof. MSc. Rodrigo Ayres
rmayres@gmail.com

Quem sou eu?

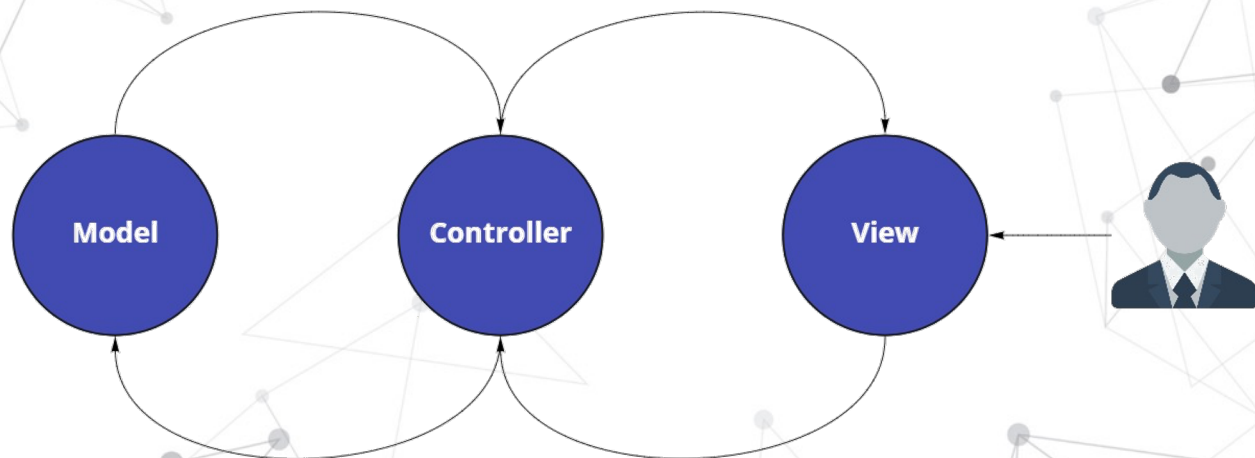


E você?

- Nome
- Idade
- Profissão

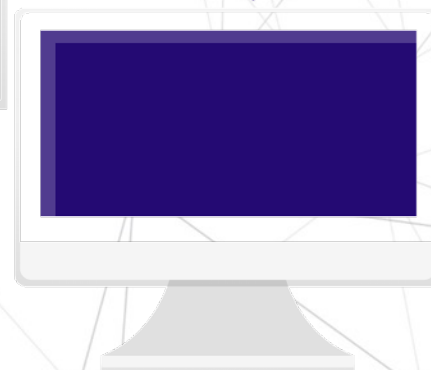
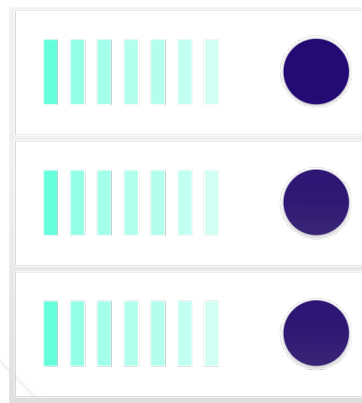


O que é MVC?



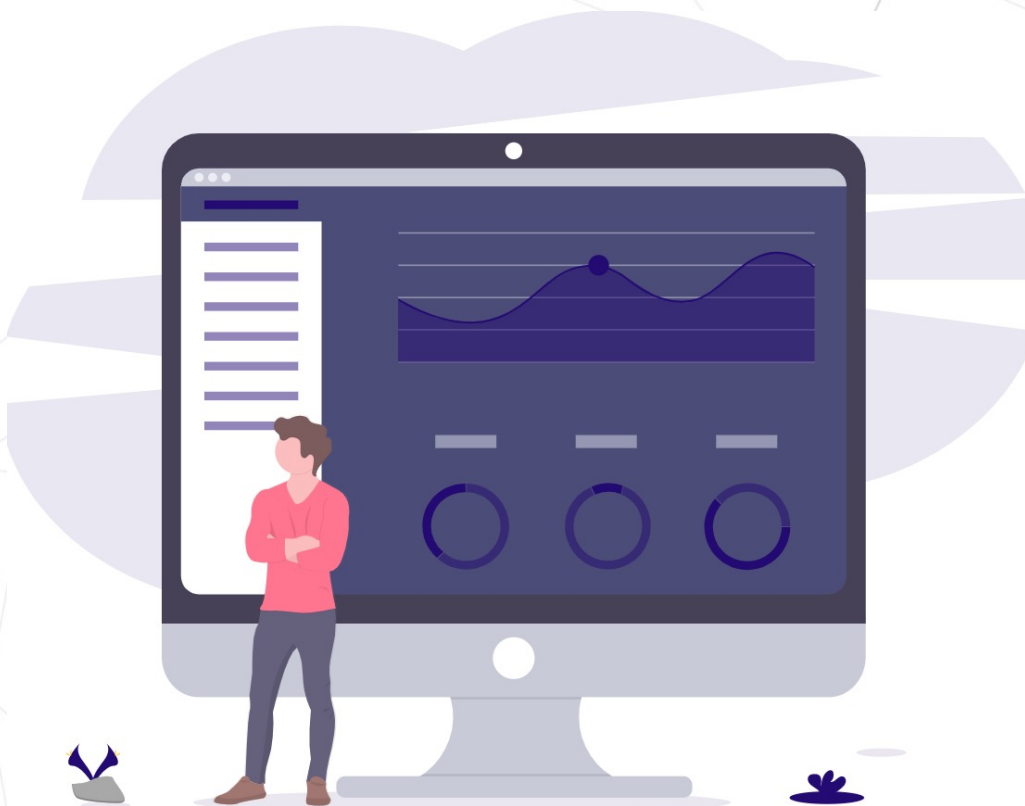
Model

- Persistência de dados
- Validações



View

- Exibição de dados
- Usuário interage



Controller

- Recebe o que o usuário solicitou
- Envia para a model
- Garçom



Há outras camadas?

- Repositório
- Testes
- Etc



Onde fica a regra de negócio?

Mas o que é afinal “lógica” ou “regra” de negócio

- Um fórum precisa mostrar antes as respostas com mais votos. Isso é uma regra de negócios?
- O fórum não pode admitir uma pergunta sem título.
- Essas regras precisam estar no Model?

Onde fica a regra de negócio?

- Usarei essa mesma lógica em dois ou mais lugares do meu código?
- A separação representa algum ganho real na minha aplicação que justifica o aumento da complexidade?
- Pode e não pode ficar no Controller

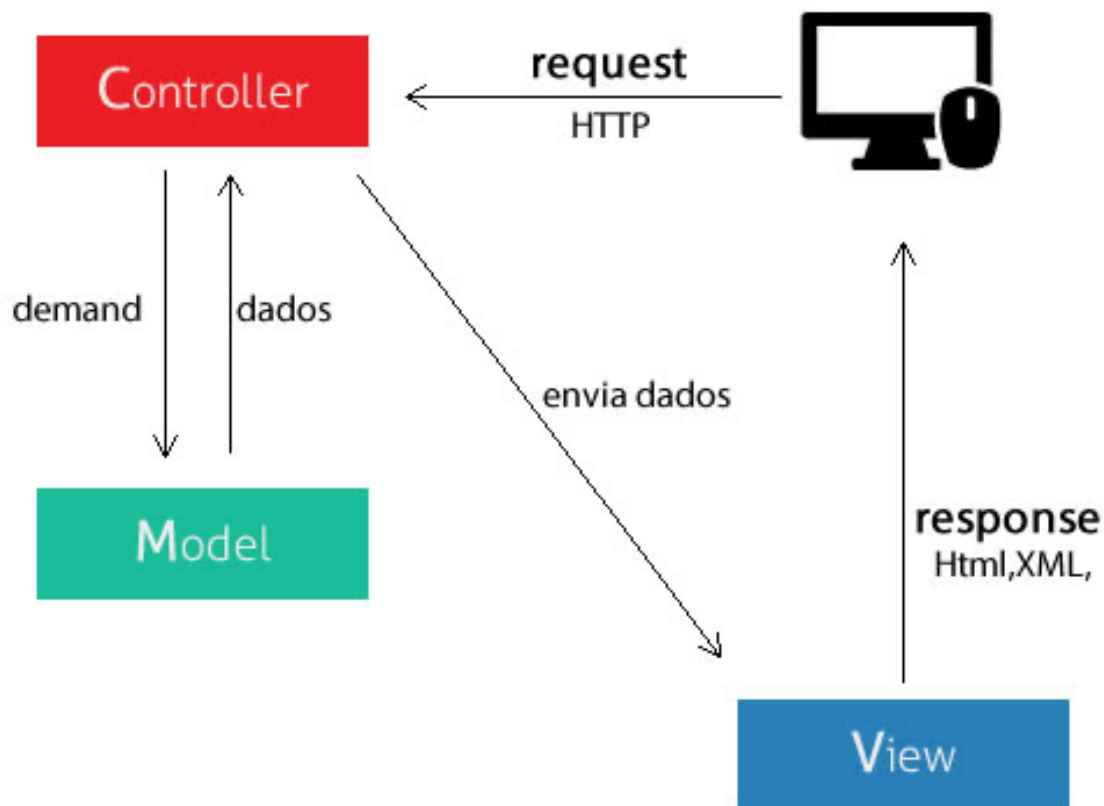
Se decida: as regras podem ou não ficar no Controller?

- Podem se o sistema tiver regras simples, regras basicamente de apresentação.
- Não deveriam se o sistema tiver regras complexas, um domínio complexo, e alta necessidade de reutilização.
- As regras precisam estar disponíveis para mais de um tipo de consumidor - como usuário final, fachada de serviços, outros serviços de negócio do próprio sistema, integrações, bases de código de outros sistemas, etc.

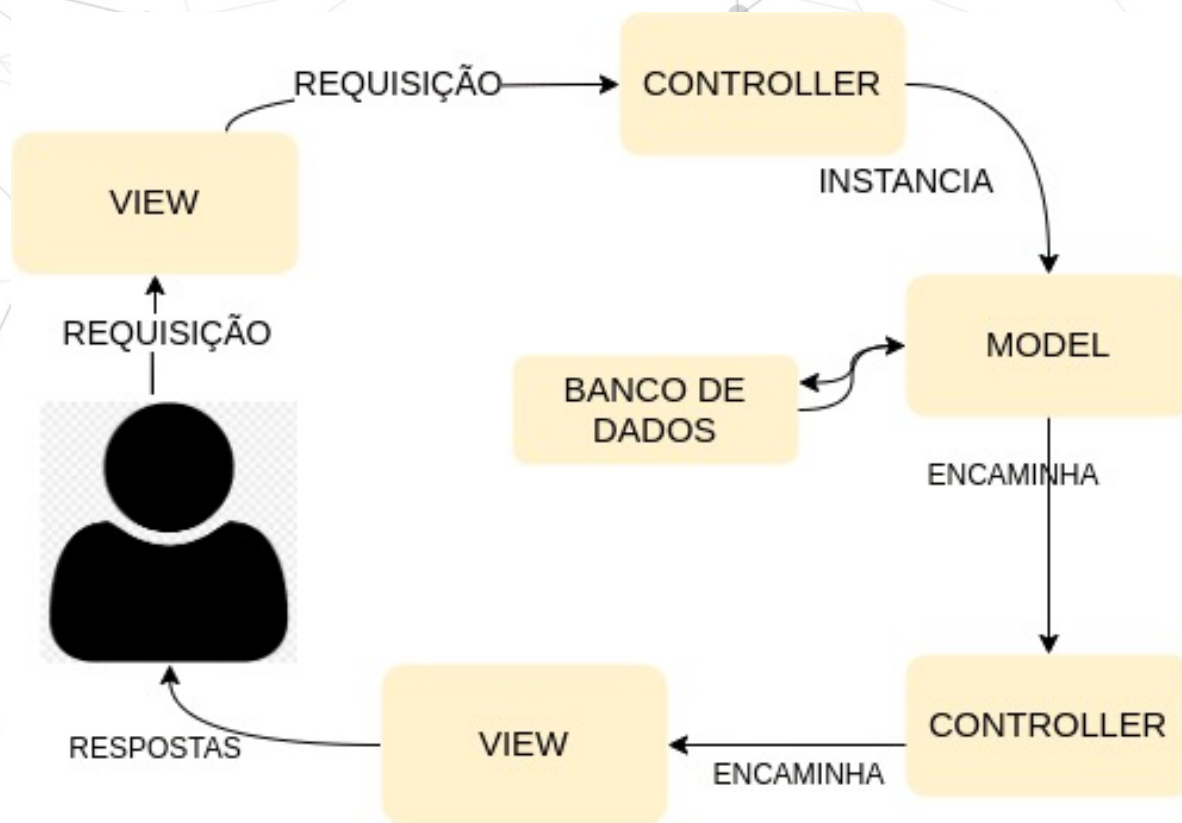
Vantagens do Padrão MVC

- Essa divisão em camadas torna mais fácil e define melhor as responsabilidades e proporciona uma independência.
- É altamente testável, pois quando se tem as camadas divididas o desenvolvedor consegue testar cada parte de forma separada, favorecendo o Desenvolvimento Dirigido por Testes (TDD).

Arquitetura MVC



Funcionamento



Funcionamento

- O usuário em seu browser realiza uma requisição HTTP. Ao chegar no controller, o mesmo verifica e comunica ao model.
- O model realiza uma consulta ao banco de dados, retornando os dados requisitados, vale lembrar que essa parte do model é invisível ao usuário.

Funcionamento

- O model retorna ao controller que tem o trabalho de renderizar a informação na view.
- A view retornará um evento para o controller que devolverá a resposta HTTP para o browser onde o usuário poderá visualizar o que foi requerido.

Atividade 1

- Toda segunda-feira as 08:30 a empresa X realizava uma reunião destinada a troca de conhecimentos entre os desenvolvedores.
- Em todas reuniões podcasts eram mencionados, portanto, o gestor do setor de desenvolvimento designou a um de seus desenvolvedores o desenvolvimento de um sistema MVC.
- Como primeira atividade, foi solicitado que o desenvolvedor desenhasse o funcionamento de um sistema MVC para a realização de busca do podcast e envio da resposta.
- Enviar resposta para:
- https://drive.google.com/drive/folders/1nCMK53_3StJiNwJR7oFvzVmfpXsXISff?usp=sharing

FRAMEWORKS

O que é um framework?

- *“Será que não existe uma forma mais fácil de fazer isso no meu projeto?”*
- enquanto estava escrevendo linhas e linhas de código para obter algum comportamento
- Ou então repetindo códigos nas estruturas
- Adicionando dependências para desenvolver alguma funcionalidade.

O que é um framework?

- O framework nada mais é do que uma ferramenta que vai te ajudar a ter como único objetivo focar em desenvolver o projeto, não em detalhes de configurações.
- Se precisarmos criar um formulário de cadastro de usuário, ele sempre vai requerer algum tipo de validação como e-mail e senha.
- O framework já terá essa validação pronta para ser utilizada.

O que é um framework?

- Rege o desenvolvimento da aplicação
- Aumento na produtividade
- Documentação



Vantagens de utilizar frameworks

- Nossos esforços se voltam para o desenvolvimento, em vez de nos preocuparmos tanto com detalhes de configurações e padrões de projeto.
- Devido a ser um trabalho colaborativo, cada vez mais soluções são implementadas à ferramenta.
- temos por padrão um código mais limpo, garantindo maior clareza de entendimento em tudo que é implementado pela ferramenta

Existem Contrás?

Existem Contrastes?

- Problemas de configurações, o que demanda tempo para a manutenção.
- Aumento de Complexidade

JavaServer Faces

JavaServer Faces

- Framework MVC para desenvolvimento web em Java.
- Tecnologia definida pelo JCP (Java Community Process)
- Desenvolvedores podem criar componentes adicionais.
- Componentes visuais se conectam com os dados do servidor.
- Eventos de componentes visuais chamam funções no servidor.
- Validadores e conversores de dados.

Implementações

- Myfaces – Implementação da Apache Software Foundation.
- Mojarra – Implementação da Sun Microsystem (RI).
- ADF – Implementação da Oracle.

Gerenciador de dependências (Maven)

*Maven*TM



Apache Maven é uma ferramenta de gerenciamento de projetos de software.

Com base no conceito de um modelo de objeto de projeto (POM), o Maven pode gerenciar a construção de um projeto a partir de uma informação central.

Objetivos do Maven

- O objetivo principal do Maven é permitir que um desenvolvedor tenha um desenvolvimento mais eficiente:
- Facilitando o processo de construção
- Fornecendo um sistema de construção uniforme
- Fornecendo informações de projeto de qualidade
- Incentivando melhores práticas de desenvolvimento

Atividade 2

- Para que serve o gerenciador de dependências?
- Qual a diferença de biblioteca e framework?
- Responder em:
- <https://drive.google.com/drive/folders/1nCMK533StJiNWJR7oFvzVmfpXsXISff?usp=sharing>



Apache Tomcat

- O Tomcat é um servidor web Java, mais especificamente, um container web.
- Ele tem a capacidade de atuar também como servidor web.

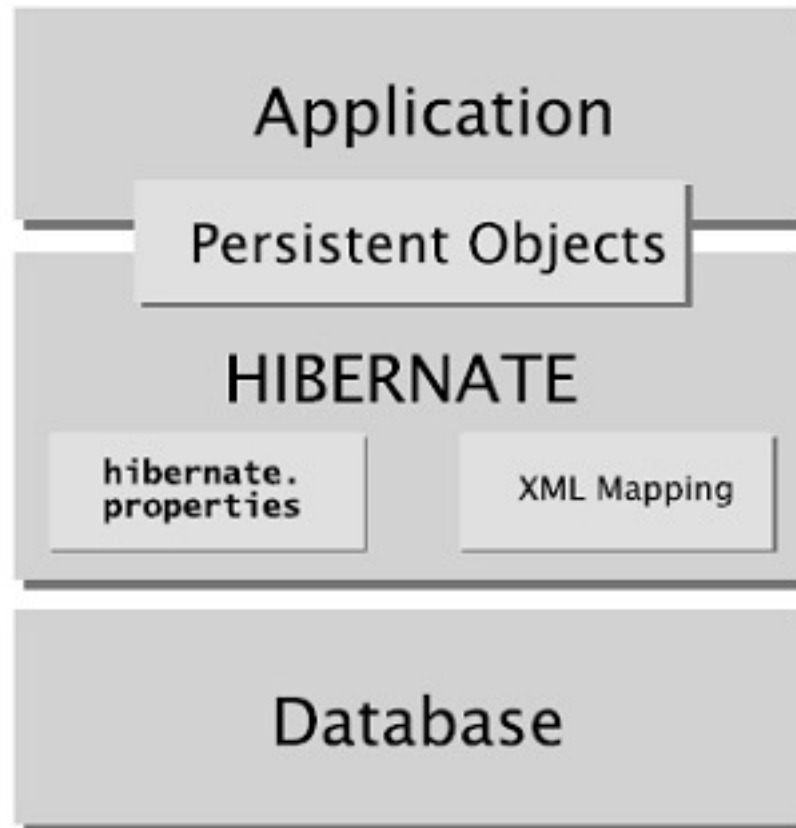
ORM – O que é?



Hibernate

- Um grande problema enfrentado pelos desenvolvedores que trabalham com linguagem orientada a objetos é o mapeamento desses objetos em banco de dados.
- Na versão 3.x o Hibernate implementa a especificação JPA (Java Persistence API) através do conceito de anotações (implementada a partir do JDK5), o que facilita ainda mais o mapeamento objeto-relacional, que pode agora ser feito diretamente na classe.

Hibernate



Configuração do Ambiente

Configurando ambiente: eclipse Java EE

- Acesse:

<https://www.eclipse.org/downloads/packages/release/2022-03/r/eclipse-ide-enterprise-java-and-web-developers>

- Realize o download de acordo com seu sistema operacional.

Instalando Eclipse

- Eclipse não possui um instalador.
- Apenas descompacte o arquivo baixado na raiz de seu sistema operacional.

Instalando Tomcat

- Acesse:
- <https://archive.apache.org/dist/tomcat/tomcat-9/v9.0.54/src/>
- Realize o download do seguinte arquivo:
- [apache-tomcat-9.0.54-src.zip](#)

Configurando o JAVA

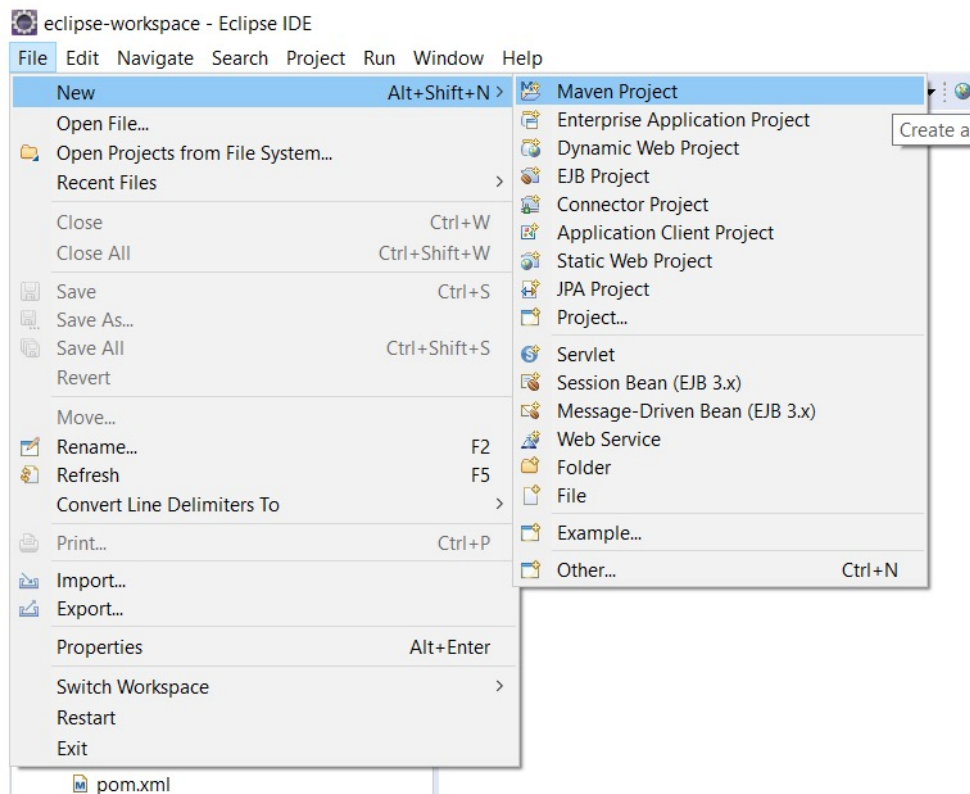
- Acesse:
- <https://www.oracle.com/br/java/technologies/javase/javase8u211-later-archive-downloads.html>
- Realize o download da seguinte versão:
- [jdk-8u301-windows-x64.exe](#)
- Após instalar abra o terminal e execute o seguinte comando:
 - Java -version
 - Verifique se aparece a versão instalada.

Configurando MySql

- Acesse:
- <https://dev.mysql.com/downloads/mysql/>
- Realize o download do MySQL Installer versão 8.028.
- A partir do instalador baixe o MySQL versão 8.0.27.
 - Obs. A Configuração banco de dados faremos juntos

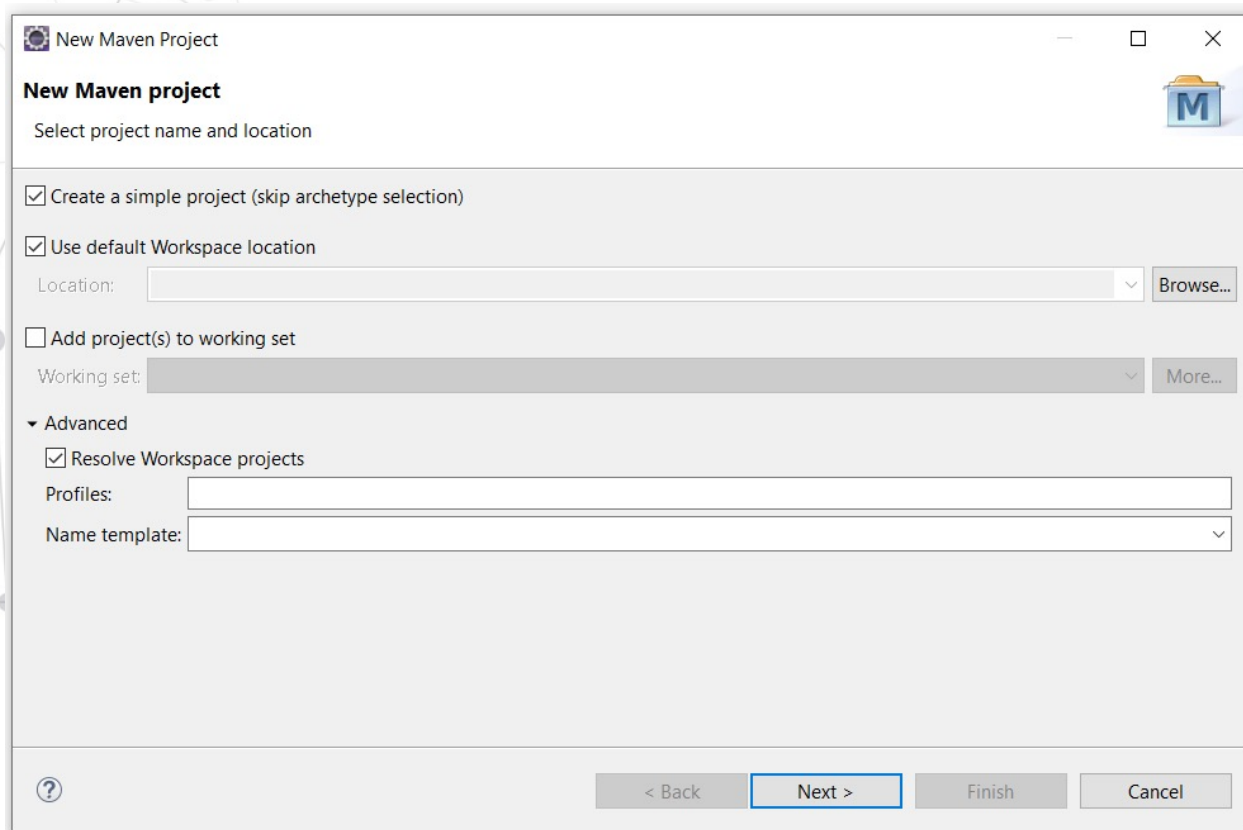
Criando o Projeto

- File -> new -> Maven Project



Criando o Projeto

- Marcar a seguinte opção:
- Create a simple project



New Maven Project

Select project name and location

☒ Create a simple project (skip archetype selection)

☒ Use default Workspace location

Location: Browse...

☐ Add project(s) to working set

Working set: More...


▼ **Advanced**

☒ Resolve Workspace projects


Profiles:

Name template:

? < Back **Next >** Finish Cancel

 New Maven Project

New Maven project
Configure project



Artifact

Group Id:

Artifact Id:

Version:

Packaging:

Name:

Description:


Parent Project

Group Id:

Artifact Id:

Version:

Advanced



Criando o Projeto

- Group Id: é um identificador da organização que criou o projeto. Como padrão usamos o formato “br.com.Nome_Projeto”.
- Artifact Id: Inserir o nome do Projeto.
- Version: Indica a versão do artefato gerado pelo projeto.
- Packaging: Indica o tipo de pacote a ser usado por este artefato.

- ProjetoWeb
 - Deployment Descriptor: ProjetoWeb
 - JAX-WS Web Services
 - Java Resources
 - src/main/java
 - br.com.projetoWeb.domain
 - br.com.projetoWeb.util
 - HibernateUtil.java
 - src/main/resources
 - hibernate.cfg.xml
 - src/test/java
 - (default package)
 - TesteConectar.java
 - src/test/resources
 - Libraries
 - JavaScript Resources
 - Deployed Resources
 - src
 - main
 - java
 - resources
 - webapp
 - WEB-INF
 - web.xml
 - test
 - target
 - pom.xml

Pacote responsável por manter as classes Java - POJOS

Classe java para instanciar uma Session

Arquivo de configuração do hibernate

A pasta WEB-INF deve ser criada manualmente

Arquivo de configuração do JSF

Arquivo em que se mantém as dependências do projeto (bibliotecas)

Hibernatecfg.xml

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

```
<!DOCTYPE hibernate-configuration PUBLIC  
    "-//Hibernate/Hibernate Configuration DTD 3.0//EN"  
    "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
```

```
<hibernate-configuration>
```

```
<session-factory>
```

```
<!-- Configurações de Conexão com o Banco de Dados -->
```

```
<property name="connection.driver_class">com.mysql.jdbc.Driver</property>
```

```
<property name="hibernate.connection.url">jdbc:mysql://localhost:3306/javaWeb</property>
```

```
<property name="connection.username">root</property>
```

```
<property name="connection.password">admin</property>
```

Url de conexão com bd, o ultimo parâmetro é o nome do banco de dados

Usuário do banco de dados

Senha do banco de dados

Generics em Java

- Generics é uma funcionalidade incorporada ao Java a partir da versão 5.0
- Permite aos programadores escreverem métodos genéricos
 - Os parâmetros dos métodos, variáveis locais e o tipo de retorno podem ser definidos na chamada do método
 - Permite ao mesmo método ser invocado usando-se tipos distintos (sem precisar sobrescrevê-lo)

Generics em Java

- Permite também a definição de classes genéricas
 - Os atributos da classe podem ser definidos no momento da instanciação do objeto
- Uma ressalva: métodos genéricos (e classes genéricas) podem ser definidos apenas para tipos referenciáveis:
- Logo, não podem ser definidos para tipos primitivos {byte, short, int, long, float, double, boolean, char}

Mapeamento Objeto Relacional

- A principal tarefa do **MOR** envolve a identificação das construções da orientação a objetos que se deseja extrair do esquema relacional, entre elas a identificação das classes e dos relacionamentos.

Mapeamento Objeto Relacional

- As principais técnicas de mapeamento de objetos em **SGBDR** podem ser descritas como:
 - **1.Mapeamento Classe – Tabela**
 - Mapeamento de uma classe em uma ou mais tabelas, ou de uma tabela para uma ou mais classes, e mapeamento de herança.

Mapeamento Objeto Relacional

- **2. Mapeamento Atributo – Coluna**
 - Mapeamento de tipos em atributos.
- **2. Mapeamento Relacionamento – Chave estrangeira**
 - Mapeamento dos relacionamentos OO em relacionamentos entre tabelas.

Mapeamento Objeto Relacional

- **Categorias da estratégia de mapeamento classe-tabela**
 - **Mapeamento de Subset**
 - Onde os atributos da classe persistente representam algumas ou todas colunas de uma tabela.
- **Atributos de uma classe**
 - **Atributos Primitivos:** Atributo de uma classe que é mapeado a uma coluna de uma tabela. Valor de um tipo de dados específico (*int*, *float*, *double*, dentre outros).

Mapeamento Objeto Relacional

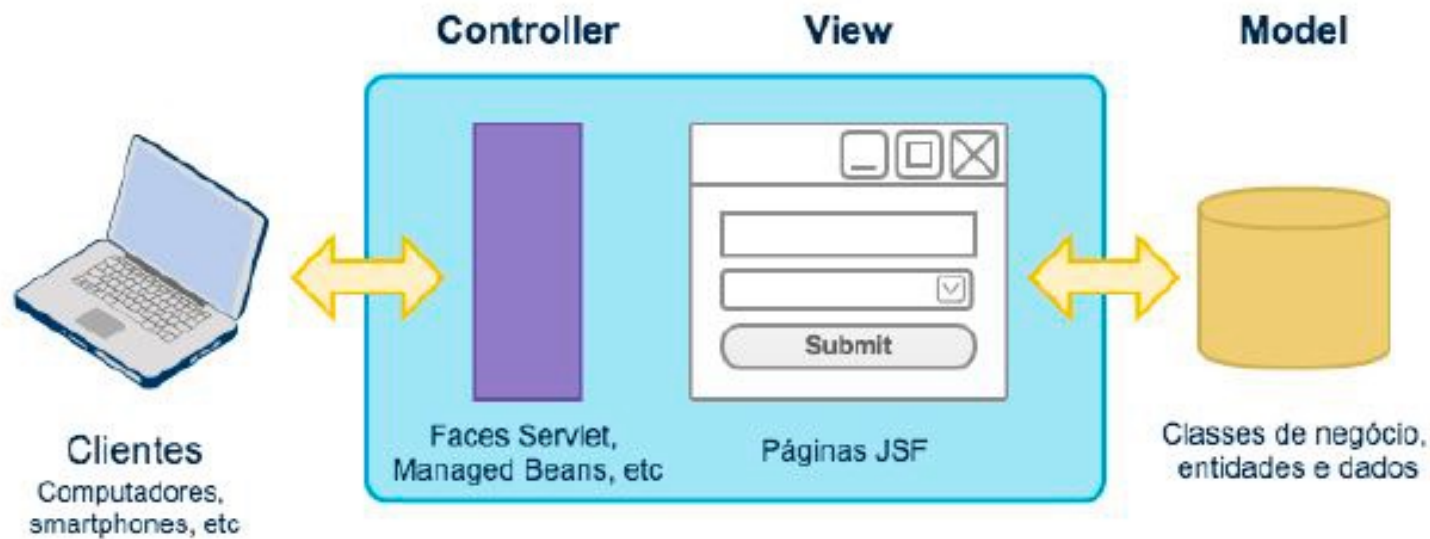
- **Atributos de Referência**

- Atributos que representam relacionamentos com outras classes.
- Atributos cujo tipo é uma referência a outro objeto ou conjunto de objetos (composição).

Chave Primária (Generated Value)

Estratégia	Descrição
GenerationType.AUTO	Valor padrão, deixa com o provedor de persistência a escolha da estratégia mais adequada de acordo com o banco de dados.
GenerationType.IDENTITY	Informamos ao provedor de persistência que os valores a serem atribuídos ao identificador único serão gerados pela coluna de auto incremento do banco de dados. Assim, um valor para o identificador é gerado para cada registro inserido no banco. Alguns bancos de dados podem não suportar essa opção.
GenerationType.SEQUENCE	Informamos ao provedor de persistência que os valores serão gerados a partir de uma sequence. Caso não seja especificado um nome para a sequence, será utilizada uma sequence padrão, a qual será global, para todas as entidades. Caso uma sequence seja especificada, o provedor passará a adotar essa sequence para criação das chaves primárias. Alguns bancos de dados podem não suportar essa opção.
GenerationType.TABLE	Com a opção TABLE é necessário criar uma tabela para gerenciar as chaves primárias. Por causa da sobrecarga de consultas necessárias para manter a tabela atualizada, essa opção é pouco recomendada.

MVC - JSF



O que é ManagedBean

- Um sistema de cadastro por exemplo, assim que o usuário terminar de digitar seus dados e clicar em concluir o managedBeans irá receber estas informações e verificar se tem algum erro e retornar uma página dizendo que o cadastro foi feito ou irá retornar uma página informando os erros.
- O managedBean é o Controller neste caso.

O que é ManagedBean

- A principal responsabilidade de um managedBean é intermediar a comunicação entre as páginas (componentes do JSF) e nosso modelo. Escutar eventos, processá-los e delegar para a camada de negócios são apenas algumas de suas responsabilidades.

Escopos do ManagedBean

- Anotações de escopo definem o escopo no qual o bean gerenciado será colocado. Se o escopo não for especificado, o bean será padronizado escopo de `@Request`.
- Quando referenciamos um `managedBean` via EL, o framework do JSF instanciará um objeto da classe do `managedBean`, ou recuperará uma instância existente.
- Todas as instâncias possuem um tempo de vida, que é definido dependendo do escopo usado no `managedBean`. Os escopos de `managedBeans` JSF podem ser definidos através de anotações do pacote `javax.faces.bean`.

Escopos do ManagedBean

- `@NoneScoped`: o bean será instanciado a cada vez que for referenciado.
- `@RequestScoped` (padrão): tem vida curta, começando quando é referenciado em uma única requisição HTTP e terminando quando a resposta é enviada de volta ao cliente.
- `@ViewScoped`: a instância permanece ativa até que o usuário navegue para uma próxima página.

Escopos do ManagedBean

- `@SessionScoped`: mantém a instância durante diversas requisições e até mesmo navegações entre páginas, até que a sessão do usuário seja invalidada ou o tempo limite é atingido. Cada usuário possui sua sessão de navegação, portanto, os objetos não são compartilhados entre os usuários.
- `@ApplicationScoped`: mantém a instância durante todo o tempo de execução da aplicação. É um escopo que compartilha os objetos para todos os usuários do sistema.

Páginas XHTML

- As páginas em JSF são estruturas em XML, no início sempre ficam declarados as bibliotecas e a referência deles para as tags JSF.
- Logo após termos sempre o HEAD que é o nosso cabeçalho, onde são definidos CSS, JavaScript, título e outros recursos.

Páginas XHTML

- E a coisa mais importante que você deve saber é que para o JSF funcionar todos os elementos devem estar dentro do FORM e o este dentro do BODY.
- Tendo está estrutura inicial dentro do FORM será onde criaremos toda a estrutura de uma página, por exemplo um cadastro de pessoa.

Páginas XHTML

```
1 <?xml version="1.0" encoding="ISO-8859-1" ?>
2
3 <html xmlns="http://www.w3.org/1999/xhtml"
4       xmlns:f="http://java.sun.com/jsf/core"
5       xmlns:h="http://java.sun.com/jsf/html"
6       xmlns:p="http://primefaces.org/ui">
7
8 <h:head>
9 </h:head>
10
11 <h:body>
12     <h:form>
13
14     </h:form>
15 </h:body>
16
17 </html>
18
```

Expression Language

Front-End XHTML

```
<h:form id="formPessoa">

  <h1>Olá JSF</h1>

  <h:inputText id="campoNome" value="#{pessoaBean.nome}">

  <br/>
```

Expression Language (EL) JSF
#{managedBean.atributo}
#{managedBean.acao}

setName(String nome)
getNome()

Back-End ManagedBean Servidor

```
@ManagedBean(name = "pessoaBean")
public class PessoaBean {

  private String nome;
  private String sobrenome;
```

Atividade Prática

FIM

Prof. MSc. Rodrigo Ayres
rmayres@gmail.com