Curso de Spring Boot

Instrutor: Bergson Barros

Apresentação do instrutor

- 41 anos, casado, pai do Davi e da Laura
- Bacharel em Ciência da Computação (UFAL)
- Pós-Graduado em Segurança de Redes e Criptografia (UFF)
- Analista de Sistemas do Serpro
- Trabalha profissionalmente com Java há 19 anos
- Certificações em Python (PCEP), Azure (DP-900), LGPD (LGPDF) e Scrum Foundations (SFPC)
- Trabalha com Spring Boot desde 2019

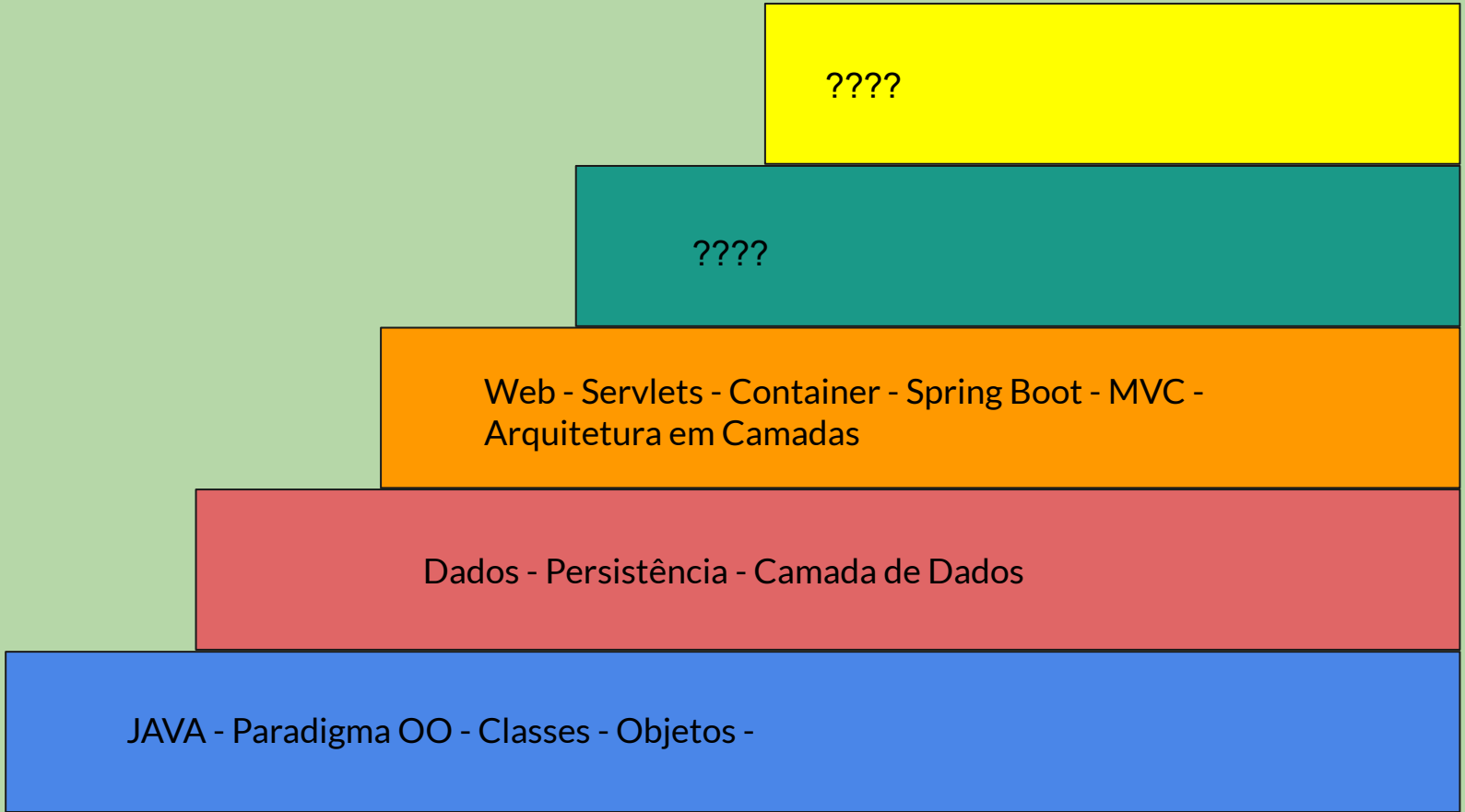
Motivação

Relaxe, pause para afiar o seu machado!

“Se eu tivesse apenas uma hora para cortar uma árvore, eu usaria os primeiros quarenta e cinco minutos afiando meu machado.”

“Tempo de treinamento não é tempo perdido.”

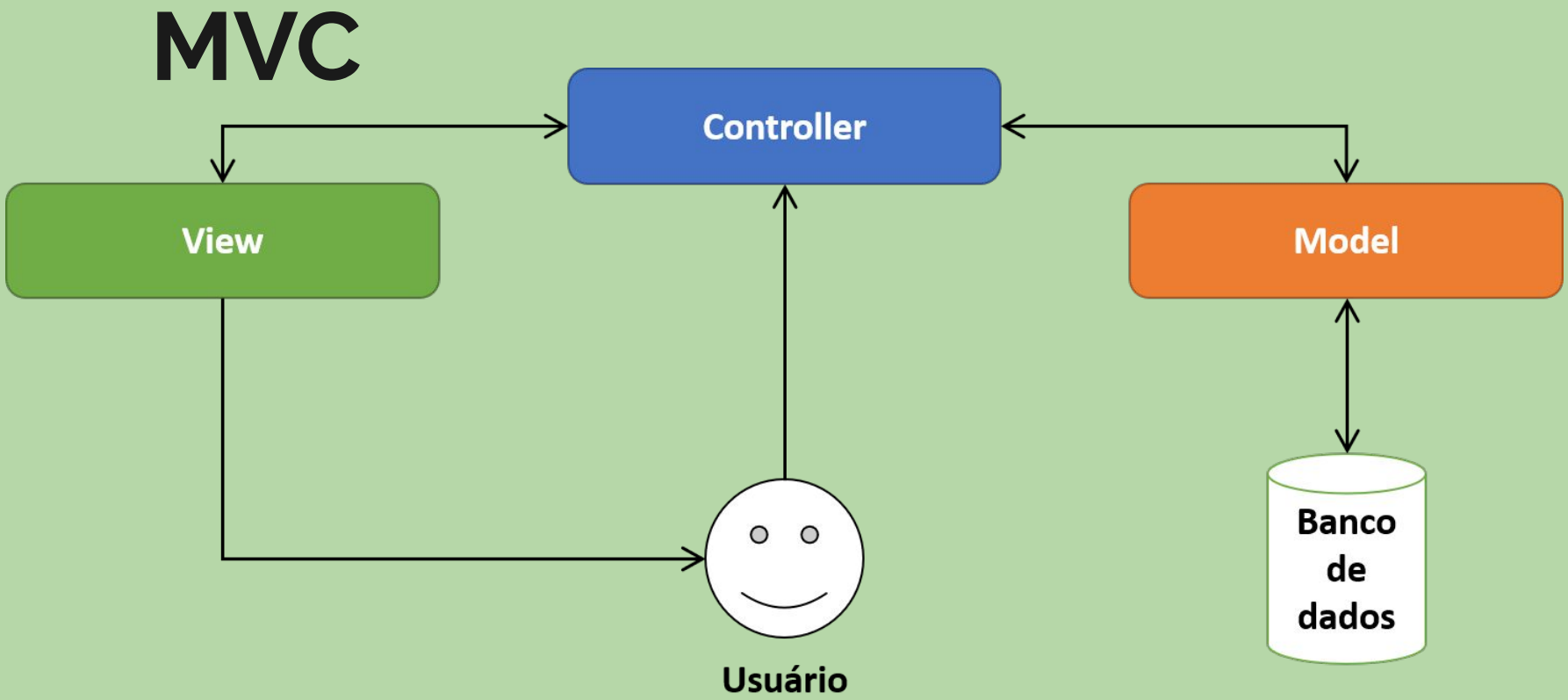
Avançando no Aprendizado



MVC

- Acrônimo para **Model-View-Controller**

- É um padrão de projeto de software focado no **reúso de código** e na separação de conceitos em **três camadas interconectadas**, onde a apresentação dos dados e interação dos usuários são separados dos métodos que interagem com o banco de dados.



Fonte: https://www.treinaweb.com.br/blog/o-que-e-mvc

Por que estudar Spring?

"Spring torna a programação em Java mais **rápida**, mais **simples** e mais **segura** para todos. Spring foca na **velocidade**, **simplicidade** e a **alta produtividade** tornou o Spring o framework Java mais popular do mundo." (fonte: site Spring)

- Relatório dos frameworks mais utilizados no mundo: https://snyk.io/jvm-ecosystem-report-2021/

Benefícios de usar Spring?

- Spring está em todos os lugares (big techs)
- Flexível (Spring Core e bibliotecas de terceiros)
- Produtivo (web serve embarcado)
- Rápido (iniciar, parar, execução otimizada)
- Seguro (cuidado dos desenvolvedores em gerenciar as vulnerabilidades das bibliotecas)
- Solidário (grande comunidade mundial para todas as diversidades, idades...)

Fonte: https://spring.io/why-spring

Ecossistema Spring

- Spring Boot
- Spring Framework
- Spring Data
- Spring Cloud
- Spring Security
- Spring Session
- Spring Batch

Todos os projetos do ecossistema Spring estão em https://spring.io/projects

Conhecendo o Spring Boot

- Documentação https://spring.io/projects/spring-boot

- Spring Boot facilita a criação de aplicações independentes (stand-alone), baseado em Spring, que você pode simplesmente executar

Funcionalidades do Spring Boot

- Criação de aplicações stand-alone (independentes)
- Tomcat embarcado, Jetty ou Undertown
- Não há necessidade de arquivos WAR
- Simplificação na configuração da build através do ‘starter’
- Configuração das bibliotecas Spring e de terceiros (3rd party)
- Provê ferramentas de apoio e monitoração da produção (metrics, health checks, etc)

Criando projetos com Spring Initializr

https://start.spring.io/

Apache Maven

- Apache Maven é uma ferramenta responsável pelo gerenciamento das builds do projeto, suas configurações e suas dependências
- Baseado em um arquivo **POM**
- Gerenciamento de dependências do projeto

Prática 1

- Criar um projeto inicial no Spring Initializr (https://start.spring.io/)
- Verificar no pom todas as informações passadas no Spring Initializr
- Criar uma classe controladora e anotar com @Controller
- Na classe criada, criar método hello que retorna uma mensagem de boas vindas. Exemplo: Olá aluno, seja bem-vindo!!!

Prática 2

- Criar uma classe controladora com seu nome e anotar com @Controller
- Na classe criada, criar método hello que retorna uma mensagem de boas vindas
- Após isto, decore a mensagem de boas vindas com tags HTML (h1, h2, p, strong, ect)

Projeto do Curso - Scholl Control API

- Criação de uma API REST para praticar os conceitos aprendidos nas aulas voltando para o negócio de uma escola.

Projetos:

- Spring MVC
- Spring Data JPA
- Spring Validation

Spring Core

- Atualmente na versão **5.3.22**
- **Inversão de Controle** (IoC) é um padrão de projeto (Abstrato) no qual, os objetos apenas declaram suas dependências, sem criá-las, delegando essa tarefa da criação de dependências a um Container IoC (Core Container).
- **Injeção de Dependência** é a implementação (Concreta) utilizada pelo Spring Framework para a aplicação da Inversão de Controle (quando necessário).

Beans e Estereótipos

- **Bean** é um objeto que é instanciado, montado e gerenciado por um container do Spring através da **Inversão de Controle** (IoC) e da **Injeção de Controle**.

- **Estereótipos**: Categorias de Beans específicos do Spring Framework: @Component, @Service, @Controller, @Repository

API

- **API**: Acrônimo para **Application Programming Interface**, ou Interface de Programação de/para Aplicações
- Define uma forma de como dois (ou mais) componentes de software possam se comunicar, através de um conjunto de definições e protocolos.
- Exemplo: API de uso do Java, API do Python, etc

REST - Representative State Transfer

- É um modelo de arquitetura que fornece diretrizes para que os **sistemas distribuídos** se comuniquem diretamente usando os princípios e protocolos existentes da Web sem a necessidade de SOAP ou outro protocolo sofisticado.
- Não é uma linguagem nem tecnologia/framework

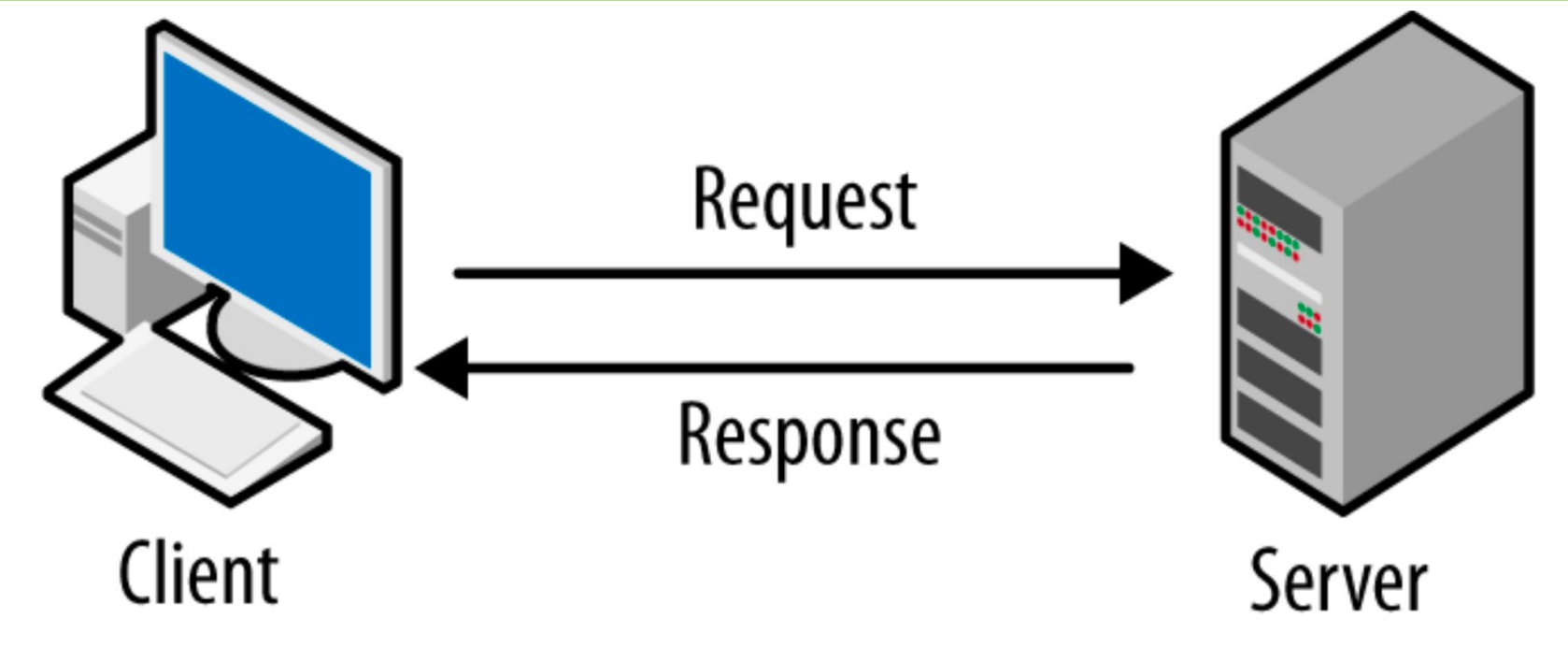
Responsabilidades no REST

- Cliente
- Servidor
- **STATELESSNESS** - Sem estado (não guarda estado)

“Servidor não precisa saber o estado do cliente e vice-versa”

- Request / Response

Responsabilidades no REST



Fonte: https://darvishdarab.github.io/cs421_f20/docs/readings/restful/api/

REST - Requisições e comunicações

- O REST precisa que um cliente faça uma requisição (request) para o servidor para enviar ou modificar dados (response).
- Uma requisição consiste em:

❏ Um método HTTP
❏ Um cabeçalho (header)
❏ Um caminho ou rota (path)
❏ Uma informação no corpo da requisição (opcional)

REST - Métodos HTTP

- Em aplicações REST, os métodos mais utilizados são:

❏ método GET
❏ método POST
❏ método PUT
❏ método DELETE

REST - Códigos de Resposta

- Para cada resposta de requisição, existe um código de status associado:
- 200 (**OK**), requisição atendida com sucesso;
- 201 (**CREATED**), objeto ou recurso criado com sucesso;
- 204 (**NO CONTENT**), objeto ou recurso deletado com sucesso;
- 400 (**BAD REQUEST**), ocorreu algum erro na requisição (podem existir inúmeras causas);
- 404 (**NOT FOUND**), rota ou coleção não encontrada;
- 500 (**INTERNAL SERVER ERROR**), ocorreu algum erro no servidor

Prática

- Classes de modelo do projeto

Links Úteis

- Site do Spring https://spring.io/
- Documentação do Spring Boot https://spring.io/projects/spring-boot
- Linguagens, frameworks e tecnologias mais usadas no mundo https://snyk.io/jvm-ecosystem-report-2021/
- Maven https://maven.apache.org/
- Eclipse IDE https://www.eclipse.org/
- Ecossistema Spring https://spring.io/projects
- REST - conceitos e fundamentos https://www.alura.com.br/artigos/rest-conceito-e-fundamentos
- API REST https://darvishdarab.github.io/cs421_f20/docs/readings/restful/api/