Treinamento de Java

Spring

Abril de 2016

Fonte: PluralSight

Spring

O que é Spring?

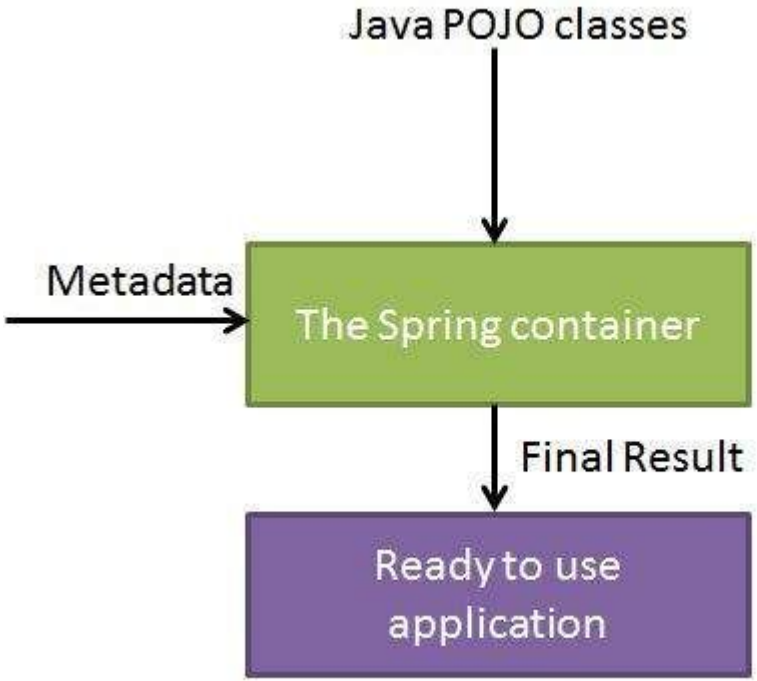
Spring é um **Framework Java** voltado para reduzir complexidade e deixar o desenvolvedor focado nas regras de negócio.

Construído **sobre as melhores práticas** de desenvolvimento e padrões de projeto.

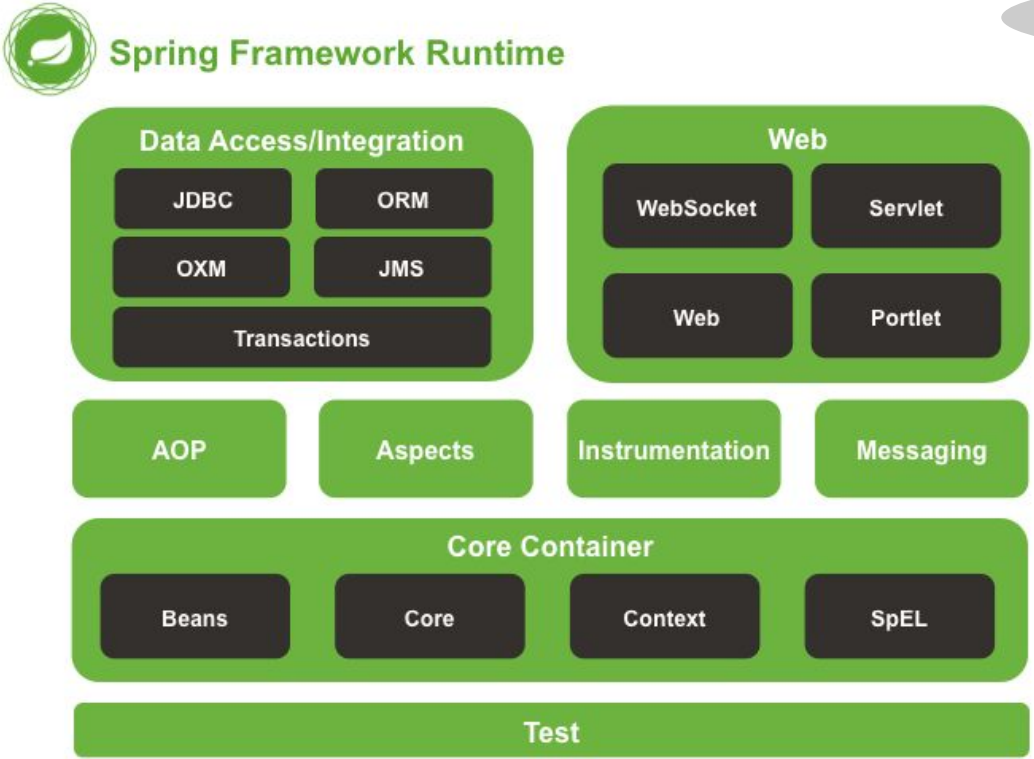
Leve e pouco intrusivo quando comparado ao J2EE.

Baseado em **POJOs e Interfaces**

IoC

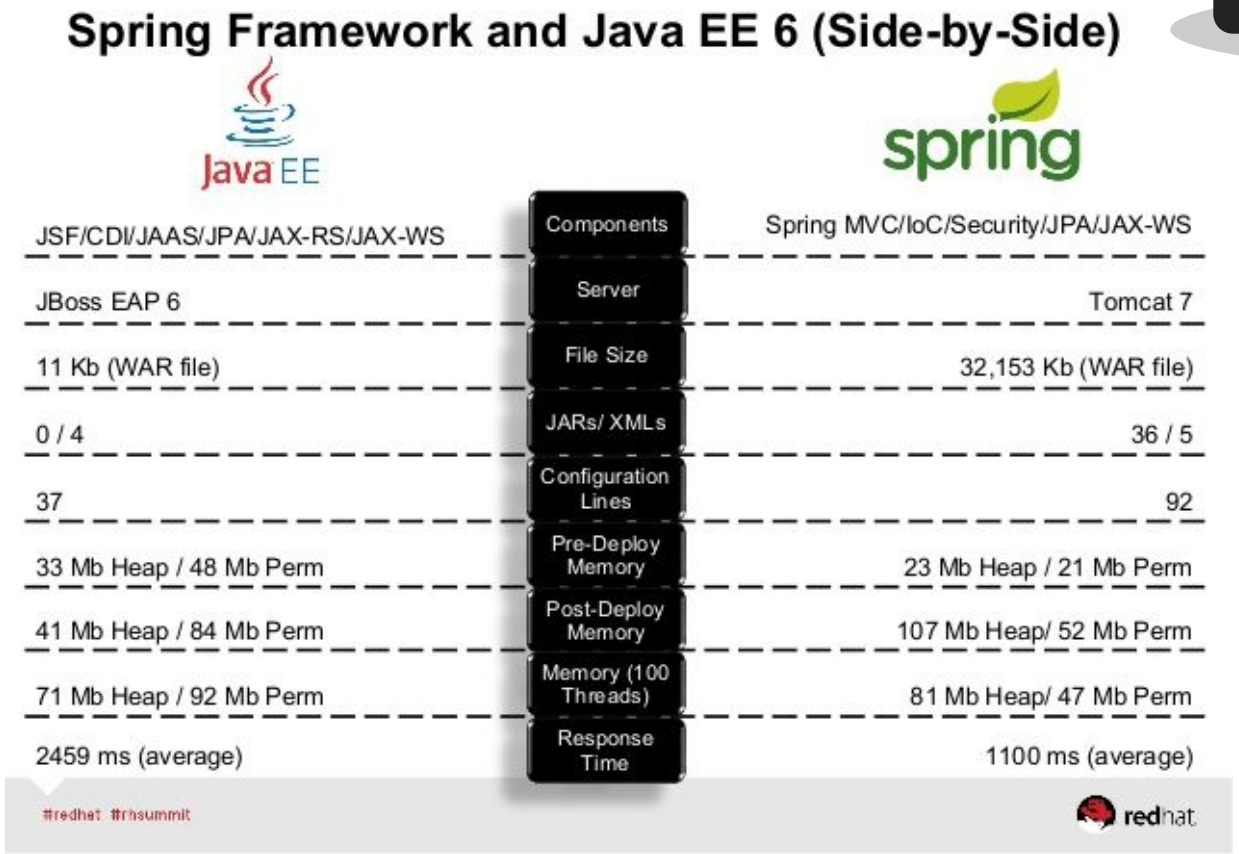

Java POJO classes
Metadata
The Spring container
Final Result
Ready to use application

Módulos



http://docs.spring.io/spring/docs/current/spring-framework-reference/htmlsingle/

Comparativo



Vamos a um exemplo?

applicationContext.xml

- Doesn't have to be named applicationContext.xml
  - More of a loose standard
- A simple view of Spring is that it is Hashmap of objects
  - Objects are name/value pairs
- Although not the intention of Spring, it can be used as a simple Registry
- XML configuration begins with a file named the applicationContext.xml
- There are namespaces that aid in configuration and validation

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
```

Beans

- **Id or Name**
  - Can be used interchangeably
  - Id has to be a valid XML identifier
    - Can't contain special characters, ie: "*", "/", "."
  - Name can contain special characters
    - Often doesn't matter with just Spring, but when building URLs with Spring MVC can be problematic
- **Default No-Args Constructor**
  - Setter Injection VS Constructor Injection
- **Class**

```
<bean name="customerRepository"
    class="com.pluralsight.repository.HibernateCustomerRepositoryImpl">
</bean>
```

Injection

- Setter injection is more common

```
<bean name="customerService" class="com.pluralsight.service.CustomerServiceImpl">
    <property name="customerRepository" ref="customerRepository"></property>
</bean>
```

- Constructor injection guarantees the contract

```
<bean name="customerService" class="com.pluralsight.service.CustomerServiceImpl">
    <constructor-arg index="0" ref="customerRepository" />
</bean>
```

Configurar tudo por XML?
Annotations!

Stereotype Annotations

- **@Component, @Service, @Repository**
- **Semantically the same**
  - @Service and @Repository both extend @Component, but don't add any features
- **@Component**
  - Regular components/beans, any POJO
- **@Service**
  - Service tier where business logic is contained
- **@Repository**
  - Data Access tier / database interaction layer

Autowiring

- Spring can automatically wire beans together for you
- byType
  - Allows a property to be autowired if exactly one bean of the property type exists in the container. If more than one exists, a fatal exception is thrown, which indicates that you may not use byType autowiring for that bean. If there are no matching beans, nothing happens; the property is not set.
- byName
  - Autowiring by property name. Spring looks for a bean with the same name as the property that needs to be autowired.
- constructor
  - Analogous to byType, but applies to constructor arguments. If there is not exactly one bean of the constructor argument type in the container, a fatal error is raised.
- no
  - No Autowiring.

Autowired

- Using Annotations is more straightforward
- Autowiring method is somewhat hidden because it is tied to where you place the annotation
- Member variables

```
@Autowired
private CustomerRepository customerRepository;
```

- Constructor

```
@Autowired
public CustomerServiceImpl(CustomerRepository customerRepository) {
    this.customerRepository = customerRepository;
}
```

- Setter

```
@Autowired
public void setCustomerRepository(CustomerRepository customerRepository) {
    this.customerRepository = customerRepository;
}
```

@Configuration

- The Java file(s) that have the @Configuration annotation replace the applicationContext files
- Methods used in conjunction with the @Bean annotation are used to get instances of Spring Beans
- @Configuration is a class level annotation:

```
@Configuration
public class AppConfig {
```

- @Bean is a method level annotation:

```
@Bean(name="customerService")
public CustomerService getCustomerService() {
```

- Class and method names can be anything, Spring doesn't care

Autowired

- We add a @ComponentScan annotation:

```
@ComponentScan({"com.pluralsight"})
```

- Just like XML configuration, mark whatever you want as @Autowired
  - byName uses the @Bean name
  - byType uses the Instance Type

Dojo!

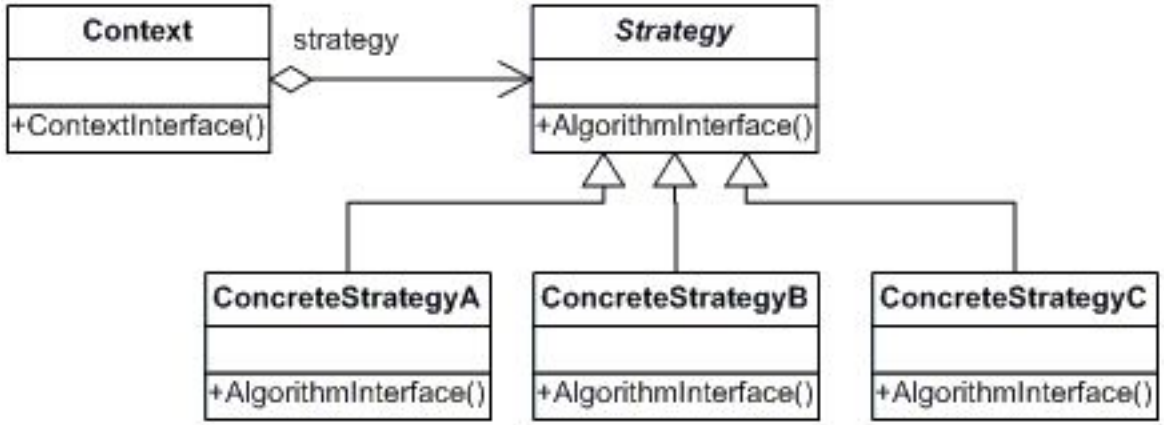
Exercício 01

Imagine uma calculadora de impostos que execute o cálculo de imposto sobre o valor de um Produto.

Exemplo: TV Samsung R$ 1.500,00

- ISS 5%
- ICMS 18%

Design Patterns - Strategy



Exercício 02

Imagine uma calculadora de Descontos que calcule descontos sobre um carrinho de compras de uma loja online.

Exemplo:

- Carrinho com mais de 2 itens de 5%
- Desconto para carrinho com maior que 1000 reais de 10%
- Os descontos não são acumulativos

Design Patterns - Chain of Responsibility