Injeção de dependências com Spring

Daniela Morais

$ whoami

Entusiasta de Java e de códigos “limpos”
Ativista de software livre

danielammorais.com / @danielammorais

beans

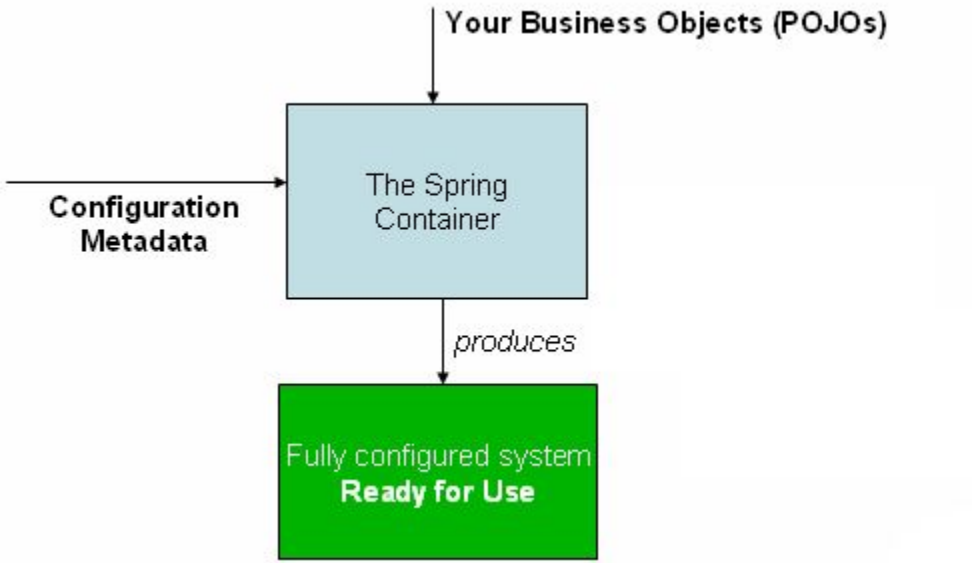
Java Beans: classes semelhantes a entidades JPA e POJOS, possuem getters/setters e nenhuma lógica de negócio

Spring Beans: Classes gerenciadas pelo container Spring através das anotações *@Bean, @Controller, @Service, @Repository, @Component etc*

beans

Exemplo de Java Beans vs Spring Beans

container spring



injeção de dependências

Singleton: Uma única instância compartilhada entre todos que necessitam

Stateless

injeção de dependência: declaração

```
@Autowired

private MyExample myExample;


public String toString() {

        return myExample.toString();

}
```

injeção de dependência: declaração

Gerenciada automaticamente pelo Spring

Forma mais conhecida e rápida

Riscos alto de *nullPointer*: Erro humano e utilizar a mesma classe fora desse container

injeção de dependência: setter

```
private MyExample myExample;

    @Autowired
    public void setMyExample(MyExample myExample) {
       this.myExample = myExample;
    }

    public String toString() {

      return myExample.toString();

      }
```

injeção de dependência: setter

MyExample myExample = new MyExample();

Permissão de criação de uma instância com new()

Riscos alto de *nullPointer*: Erro humano e utilizar a mesma classe fora desse container

injeção de dependência: construtor

```
private final MyExample myExample;

    @Autowired
    public MyService(MyExample myExample) {
        this.myExample = myExample;
    }
```

injeção de dependência: construtor

Forma mais verbosa

Uso do do “final” otimiza o código

Obriga a inicialização de todos os atributos realmente necessários

Facilidade em criar testes unitários/integração

lombok

```
@RequiredArgsConstructor(onConstructor = @__(@Autowired))
 public class MyService {

    private final @lombok.NonNull MyExample myExample;

 }
```

annotation mania

Thanks to @Annotations, @Progress is @Unstoppable!

http://www.annotatiomania.com/

```
@SuppressWarnings({"unchecked", "rawtypes"})
@Deprecated
@OneToMany(@HowManyDBADoYouNeedToChangeALightBulb)
@OneToManyMore @AnyOne @AnyBody
@YouDoNotTalkAboutOneToMany // Fightclub, LOL
@TweakThisWithThat(
    tweak = {
        @TweakID(name = "id", preferredValue = 1839),
        @TweakID(name = "test", preferredValue = 839),
        @TweakID(name = "test.old", preferredValue = 34),
    },
    inCaseOf = {
        @ConditionalXMLFiltering(run = 5),
    }
)
@ManyToMany @Many @AnnotationsTotallyRock @DeclarativeProgrammingRules @NoMore
@Fetch @FetchMany @FetchWithDiscriminator(name = "no_name")
@SeveralAndThenNothing @MaybeThisDoesSomething
@JoinTable(joinColumns = {
    @JoinColumn(name = "customer_id", referencedColumnName = "id")
})
@DoesThisEvenMeanAnything @DoesAnyoneEvenReadThis
@PrefetchJoinWithDiscriminator @JustTrollingYouKnow @LOL
@IfJoiningAvoidHashJoins @ButUseHashJoinsWhenMoreThan(records = 1000)
@XmlDataTransformable @SpringPrefechAdapter
private Collection employees;
```

that’s all

“A economia do século 21 não é sustentada por aço. A economia do século 21 é sustentada por software. Software é elemento tão fundamental para o desenvolvimento econômico no século 21 quanto o foi a produção de aço no século 20. A estrutura societária neste país mudou, no resto do mundo desenvolvido está mudando, e continuará mudando nos países em desenvolvimento, rumo a economias cuja commodity primária fundamental à produção é software. E a boa notícia é que ninguém o possui.”

Software e Comunidade no começo do Século 21