



# Módulo 19





# BackEnd Java

---

Rodrigo Pires



A decorative graphic on the left side of the slide. It features a large central hexagon with a blue-to-teal gradient, containing the number '1'. Surrounding this central hexagon are several smaller hexagons of varying shades of blue and teal. Some of these smaller hexagons contain white icons: a lightbulb, a thumbs-up, a smartphone, a magnifying glass, a gear, and a speech bubble. There is also a small network-like icon with a central node and five connecting lines.

1

# Reflection / Reflexão



# O que são?

Reflexão é um recurso da API Java que possibilita aos aplicativos o acesso e a modificação do comportamento de aplicações que estão rodando na Java Virtual Machine. Uma classe pode acessar outras classes em tempo de execução, sem conhecer sua definição no momento da compilação. Informações relativas à esta definição, como seus construtores, métodos e atributos, podem ser facilmente acessados através de métodos de reflexão da API Java.






# Classes

```
Class c1 = boolean.class;  
System.out.println(c1);
```

```
Class c2 = java.io.PrintStream.class;  
System.out.println(c2);
```

```
Class c = Class.forName("br.com.rpires.reflextions.ReflectionsClasses");  
System.out.println(c);
```


```
System.out.println(ReflectionsClasses.class);|
```





# Classes

```
boolean  
class java.io.PrintStream  
class br.com.rpires.reflextions.ReflectionsClasses  
class br.com.rpires.reflextions.ReflectionsClasses  
ReflectionsClasses  
br.com.rpires.reflextions.ReflectionsClasses
```



# Acesso a partir das classes

Membro	Class	Lista dos Membros?	Membros herdados?	Membros privados?
Field	<code>getDeclaredField()</code>	Não	Não	Sim
	<code>getField()</code>	Não	Sim	Não
	<code>getDeclaredFields()</code>	Sim	Não	Sim
	<code>getFields()</code>	Sim	Sim	Não
Method	<code>getDeclaredMethod()</code>	Não	Não	Sim
	<code>getMethod()</code>	Não	Sim	Não
	<code>getDeclaredMethods()</code>	Sim	Não	Sim
	<code>getMethods()</code>	Sim	Sim	Não
Constructor	<code>getDeclaredConstructor()</code>	Não	N/A <sup>1</sup>	Sim
	<code>getConstructor()</code>	Não	N/A <sup>1</sup>	Não
	<code>getDeclaredConstructors()</code>	Sim	N/A <sup>1</sup>	Sim
	<code>getConstructors()</code>	Sim	N/A <sup>1</sup>	Não



# Construtores


```
System.out.println("**** Construtores ****");
Class prodC = ProdutoReflexion.class;
System.out.println(prodC);

Constructor con = prodC.getConstructor();
System.out.println(con);

ProdutoReflexion prod = (ProdutoReflexion) con.newInstance();
System.out.println(prod);

Constructor con1 = prodC.getConstructor(Long.class);
System.out.println(con1);
ProdutoReflexion prod1 = (ProdutoReflexion) con1.newInstance(...initargs: 10L);
System.out.println(prod1 + " tem o valor: " + prod1.getCodigo());

Constructor[] constructos = prodC.getDeclaredConstructors();
System.out.println("Construtores declarados");
for (Constructor cons : constructos) {
    System.out.println(cons);
}
```








# Propriedades

```
System.out.println("**** Propriedades ****");
ProdutoReflexion prod = new ProdutoReflexion();
Field[] fields = prod.getClass().getDeclaredFields();
for (Field field : fields) {
    System.out.println("Nome completo: " + field);
    System.out.println("Nome simples: " + field.getName());
    System.out.println("Tipo da propriedade: " + field.getType());
    System.out.println();
}
```






# Métodos

```
System.out.println("**** Métodos ****");
ProdutoReflexion prod = new ProdutoReflexion();
Method[] methods = prod.getClass().getDeclaredMethods();
for (Method m : methods) {
    System.out.println("Nome completo: " + m);
    System.out.println("Nome simples: " + m.getName());
    System.out.println("Tipo de retorno: " + m.getReturnType());
    System.out.println();
}

Method method = prod.getClass().getMethod(name: "getNome");
System.out.println("Pegando método pelo nome: " + method.getName());

Method method1 = prod.getClass().getMethod(name: "setNome", String.class);
System.out.println("Pegando método pelo nome: " + method1.getName());

method1.invoke(prod, ...args: "Rodrigo");
System.out.println("Pegando valor do Nome: " + method.invoke(prod));
```





# Referências

[Exemplos disponíveis no meu github:](#)

<https://github.com/digaomilleniun/backend-java-ebac>

