Programação Java
Enumerações – Enum
Funções Lambdas

Profª. Heloisa Moura

Programação Java
Enumerações - Enum

Enum

São tipos de campos que consistem em um conjunto fixo de constantes (static final), sendo como uma lista de valores pré-definidos. No Java, pode ser definido um tipo de enumeração usando a palavra chave enum.

Todos os tipos enums implicitamente estendem a classe java.lang.Enum, sendo que o Java não suporta herança múltipla, não podendo estender nenhuma outra classe.
As constantes enum são implicitamente final e static.
.
Os tipos enumeradores enum foram incorporados no Java a partir da versão 5, tornando-se uma palavra reservada do Java. As definição de constantes em Java é bastante comum em diversas aplicações. O uso das constantes ajuda a tornar o nosso programa muito mais fácil de ler e de dar manutenção.

Programação Java
Enumerações - Enum

Características dos tipos enum

- As instâncias dos tipos enum são criadas e nomeadas junto com a declaração da classe, sendo fixas e imutáveis (o valor é fixo).;
- Não é permitido criar novas instâncias com a palavra chave new;
- O construtor é declarado private, embora não precise de modificador private explícito;
- As constantes enum podem ser utilizadas em qualquer lugar em que constantes podem ser utilizadas, como nos rótulos case de instruções switch e para controlar instruções for aprimorado.

Programação Java
Enumerações - Enum

Características dos tipos enum

- Seguindo a convenção, por serem objetos constantes e imutáveis (static final), os nomes declarados recebem todas as letras em MAIÚSCULAS;
- As instâncias dos tipos enum devem obrigatoriamente ter apenas um nome;
- Opcionalmente, a declaração da classe pode incluir variáveis de instância, construtor, métodos de instância, de classe, etc.

Programação Java
Enumerações - Enum

Declaração Enum

Na declaração é definida uma classe chamada de tipo enum. O corpo da classe enum pode incluir métodos e outros campos. O compilador automaticamente adiciona alguns métodos especiais quando se cria um enum.

Exemplo

```
public enum CartasEnum {
        A, J, Q, K;
}
```

Programação Java
Enumerações - Enum

Inicializando valores

Para iniciar os valores declarados dentro dos atributos Enum, é preciso declarar um construtor para iniciar os seus atributos que são declarados.

Exemplo

```
public enum CartasEnum {

    J(11),Q(12),K(13),A(14);
    public int valorCarta;
            CartasEnum(int valor) {
                valorCarta = valor;
            }
}
```

Programação Java
Enumerações - Enum

Percorrendo Valores

Os valores Enum tem um método estático chamado values() que retorna uma matriz contendo todos os valores do enum na ordem em que são declarados. Este método é normalmente usado em combinação com o for para construir cada repetição dos valores de um tipo de enumeração.

No exemplo seguinte é percorrido o tipo Enum OpcoesMenu

TesteOpcoesMenu

Programação Java
Enumerações - Enum

Comparando Valores Enum

Um Enum pode ser comparado com outro objeto através do método equals() ou utilizando ==

No exemplo seguinte declaramos o método comparaEnum do tipo static para ser acessível para toda a classe e fazer referência ao tipo Enum declarado.

EXEMPLO TesteOpcoesMenu

Programação Java
Enumerações - Enum

Imprimindo valor Enum

Nesse exemplo é mostrado como pode ser impresso um valor do tipo enum, através do método name().

Exemplo
MarcasEnum

Programação Java

Enumerações - Enum

Em geral as enums podem ser declaradas como classes separadas ou membros de classes. Normalmente, quando a enumeração se torna grande, com diversas constantes, propriedades e métodos deseja-se por declarar uma classe própria para essa enum, não poluindo assim a classe principal. No entanto, quando a enumeração é pequena possuindo poucas constantes ou métodos, geralmente declara-se a enumeração como um membro de uma classe. Lembrando que uma enumeração nunca pode ser declaração dentro de métodos

Pode ter métodos abstratos, que são implementados dentro do próprio tipo enum ou implementar uma interface.

Exemplo TesteEnumMes

Programação Java
Enumerações - Enum

Nos tipos Enum também existem outros métodos descritos abaixo.

String toString() - retorna uma String com o nome da instância (em maiúsculas).

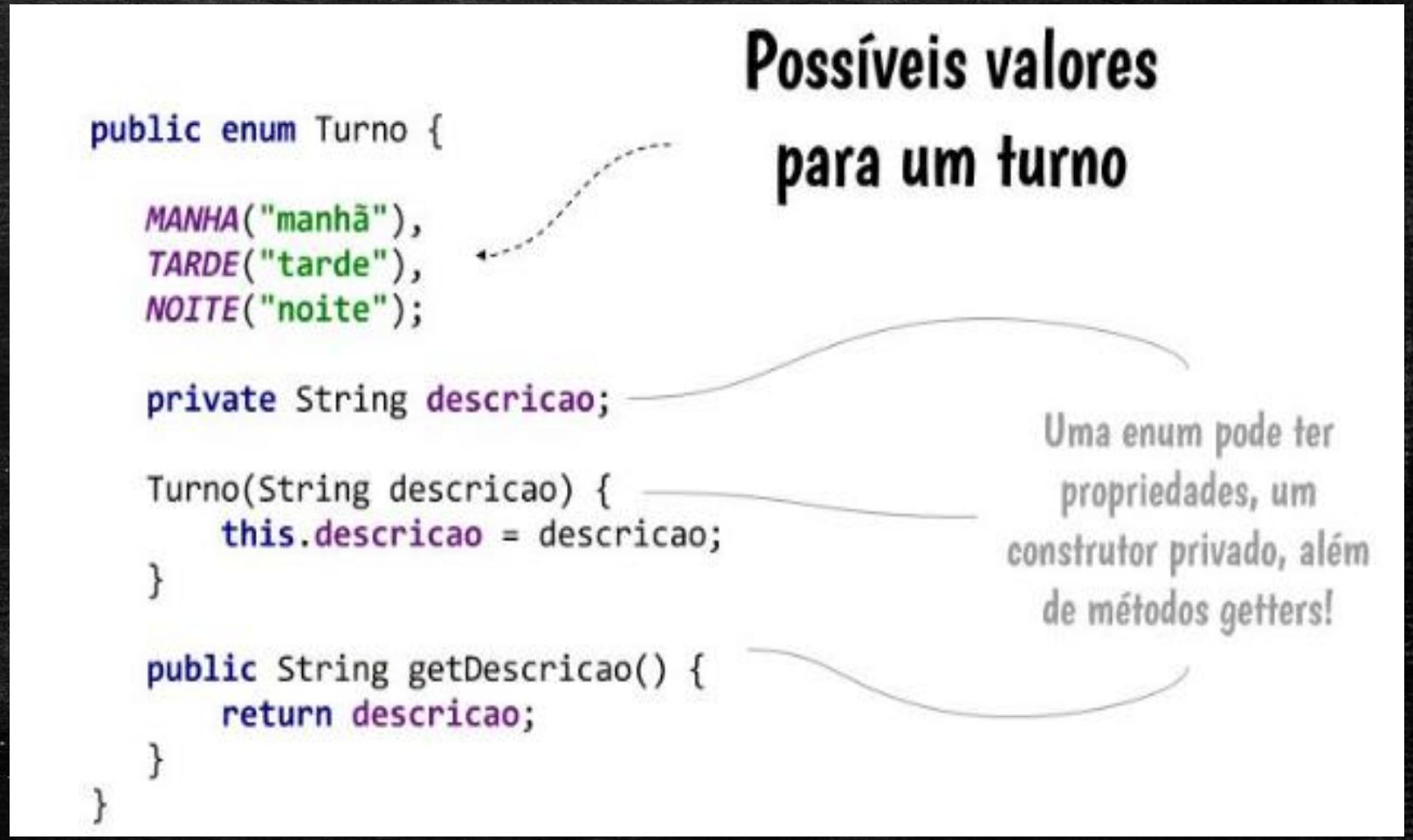
valueOf (String nome) - retorna o objeto da classe enum cujo nome é a string do argumento.

int ordinal() - retorna o número de ordem do objeto na enumeração.

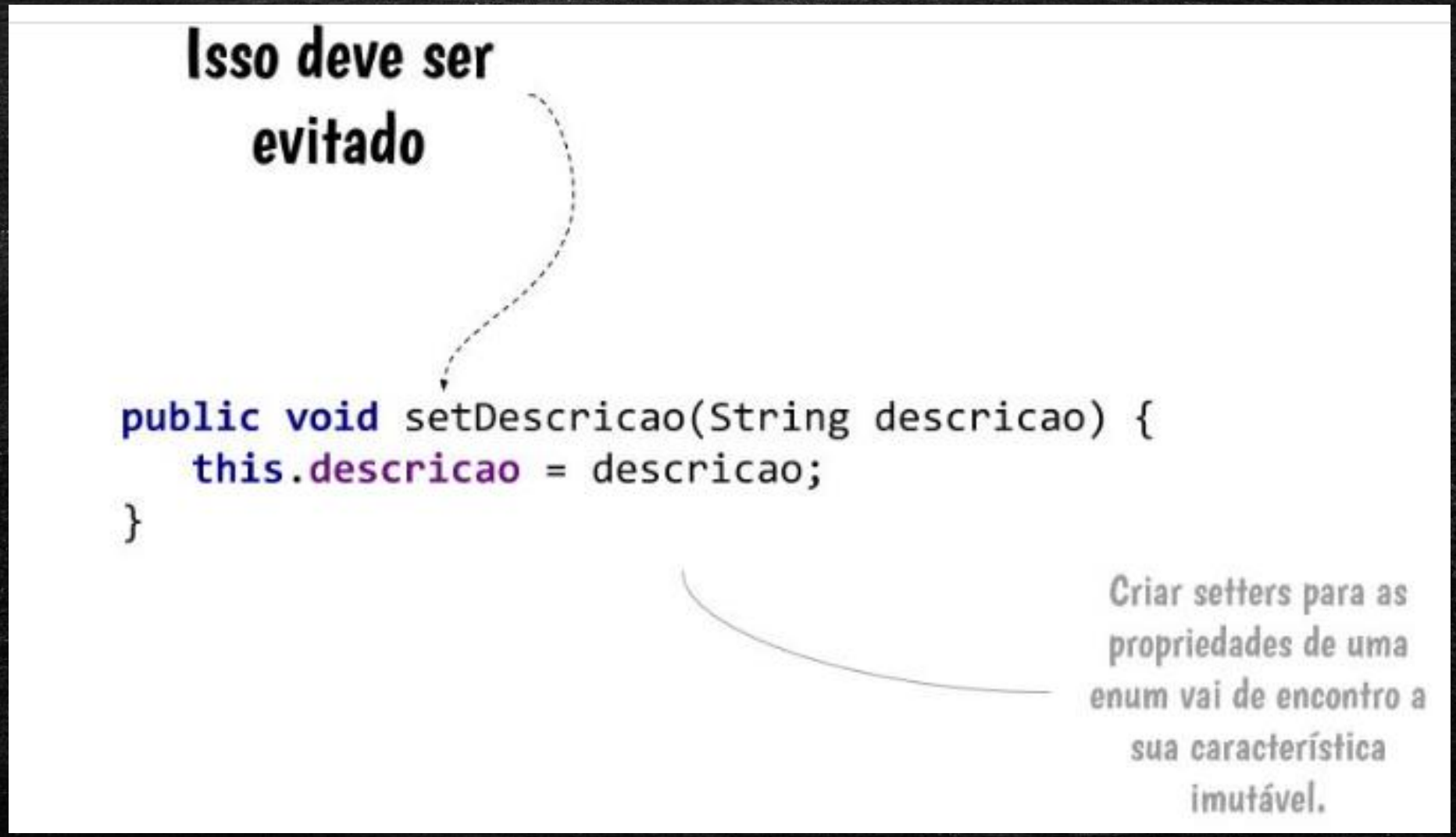
Programação Java
Enumerações - Enum

Enum

Exemplo
usando o método
valueOf()



Programação Java
Enumerações - Enum



Programação Java

Enumerações - Enum

Conclusão

Utilizar enumerações por si só ajudam na redução do número crescente de bugs na aplicação. Isso ocorre pois as enumerações reduzem o número infinito de possibilidades de valores que podem ser atribuídos.

Documentação:

https://docs.oracle.com/javase/6/docs/api/java/lang/Enum.html

.

Programação Java
Funções Lambda

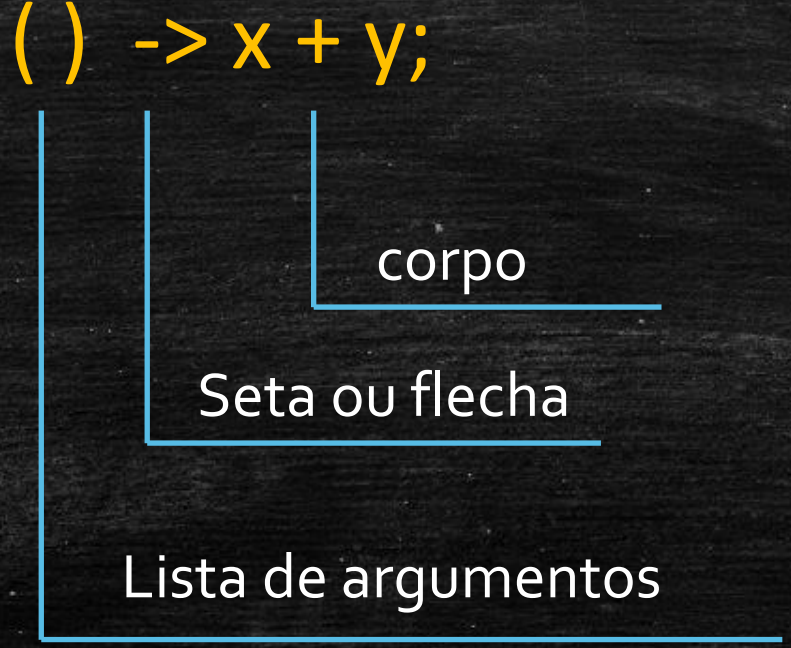
Profª. Heloisa Moura

Programação Java
Funções Lambda

As funções Lambda foram adicionadas ao Java na versão 8.
Uma função lambda é uma função sem declaração, isto é, não é necessário colocar um nome, um tipo de retorno e o modificador de acesso. A ideia é que o método seja declarado no mesmo lugar em que será usado. Pode ser chamada também de métodos anônimos.
Podemos dividir o entendimento das funções lambda em duas partes: As funções lambda e as interfaces funcionais.
Esse entendimento se torna mais lógico quando percebemos que as funções lambda implementam os métodos que estão definidos em uma interface funcional. Então toda vez que quisermos trabalhar com funções lambda precisamos fazer uso de uma interface funcional.

Programação Java
Funções Lambda

As funções lambda em Java tem a sintaxe definida como (lista de argumento) -> (corpo), é constituída por três partes



A seta que indica que estamos trabalhando com uma instrução lambda

Programação Java
Funções Lambda

Sintaxes lambda válidas

```
(int a, int b) -> {  return a + b; };

( a, b ) ->  a + b;

() -> System.out.println("Hello World");

(String s) -> { System.out.println(s); };

() -> 42;

() -> { return 3.1415;};

a -> a > 10;
```

Programação Java
Funções Lambda

Função lambda

Uma Função lambda pode ter nenhum ou vários parâmetros e seus tipos podem ser colocados ou podem ser omitidos, dessa forma, eles são entendidos pelo Java. A expressão lambda pode ter nenhum ou vários comandos:

Se a mesma tiver apenas um comando as chaves, não são obrigatórias e a função retorna o valor calculado na expressão;

Se a função tiver vários comandos, é necessário colocar as chaves e também o comando return - se nada for retornado, a função tem um retorno void.

Uma expressão lambda não possui nome.

Programação Java
Funções Lambda

Interfaces Funcionais

Para entendermos melhor como funcionam as funções lambda precisamos saber o que são as Interfaces Funcionais.

Interface funcional é aquela que contém somente um método que é implicitamente abstrato e esse método especifica a finalidade desta interface.

As interfaces funcionais representaram uma única ação. Acabam fornecendo um tipo alvo para as funções lambda e a referencia de métodos. Os parâmetros das funções lambda e os tipos de retorno serão combinados e adaptados. Geralmente as interfaces funcionais apresentam a notação @FunctionalInterface. Mas não é obrigatório a utilização para que o compilador conheça uma Interface funcional como uma IF.

Programação Java
Funções Lambda

Interfaces Funcionais

Algumas interfaces funcionais no Java 8

Function: uma função que recebe um único argumento.

Predicate: Função que recebe um único argumento e retorna um booleano.

Consumer: Função que recebe um único argumento e não devolve nada.

Programação Java
Funções Lambda

Interfaces Funcionais no Java 8

Nome	Parâmetros	Retorno	Exemplo
BinaryOperator<T>	(T, T)	T	Fazer qualquer tipo de operação em dois objetos do mesmo tipo.
Consumer<T>	T	void	Imprimir um valor.
Function<T, R>	T	R	Receber um objeto do tipo Double e retorna sua representação em String.
Predicate<T>	T	boolean	Fazer um teste qualquer no objeto recebido como parâmetro: umaString.endsWith("sufixo")
Supplier<T>	–	T	Operação que não receba nenhum parâmetro mas tenha um retorno.

Programação Java
Funções Lambda

Interfaces Funcionais

Então podemos dizer que uma função lambda é qualquer objeto que implemente alguma dessas interfaces ou uma outra interface funcional desenvolvida por nós.

Vamos nos Exemplos de utilização das funções lambda

TesteLambda
TesteLambda2

Programação Java
Funções Lambda

API Collection com lambda

As funções lambdas podem ser bastante utilizadas com as classes de coleções do Java, pois nessas fazemos diversos tipos de funções que consistem basicamente em percorrer a coleção e fazer uma determinada ação, como por exemplo, imprimir todos os elementos da coleção, filtrar elementos da lista e buscar um determinado valor na lista.

Programação Java
Funções Lambda

Exemplos de utilização com as classes de coleção do Java, para aumentar a flexibilidade de diversas funções.

TesteLambda3Collection
TesteLambdaIPredicate
TesteLambdaIConsumer

Programação Java
Funções Lambda

Conclusão

As funções lambda são mecanismos bastante poderosos, que facilitam muito a escrita de código conciso e evitam que o programador seja obrigado a escrever um monte de código “inútil”, principalmente em operações simples, além de flexibilizar o mesmo.

Apesar de serem bastante úteis, as funções lambda nem sempre são a melhor opção, caso seja necessário reutilizar diversas vezes uma função, talvez seja melhor criar uma classe ou interface com apenas um método e não uma expressão lambda.
https://docs.oracle.com/javase/tutorial/java/javaOO/lambdaexpressions.html

https://www.oracle.com/webfolder/technetwork/tutorials/obe/java/Lambda-QuickStart/index.html

Programação Java
Funções Lambda

EXERCÍCIO