LINGUAGEM DE PROGRAMAÇÃO MULTIPLATAFORMA

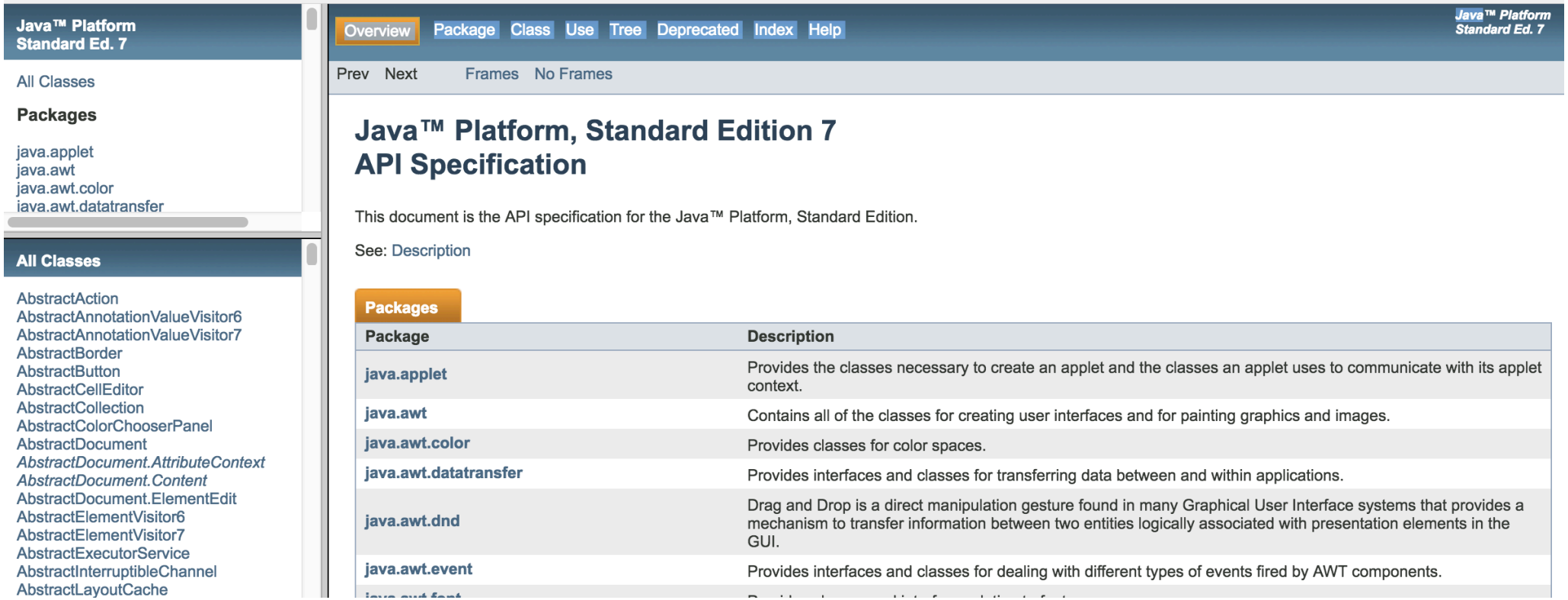
(LINGUAGEM JAVA)

Profª. Tania Basso

Colégio Técnico de Limeira

Application Programming Interface

JAVA API Specification → documentação que mostra pacotes, classes, interfaces Java



Pacote java.lang → classe String

String não é um tipo primitivo, é uma Classe, e como a maioria das classes, possui atributos, construtores e métodos.

```
public static void main(String[] args) {
    String nome = "Mariazinha";
    String nome2 = new String();
    String nome3 = new String("Java");
    int tamanho = nome.length(); //variável deve ser inicializada
    System.out.println("Tamanho da string: "+tamanho);
    System.out.println("Tamanho da string: "+nome3.length());
}
```

Construtores

Método length() não recebe parâmetros e retorna um valor inteiro

Pacote java.lang → classe String

Java não permite que a mudança de valor em uma string afete outras que façam referência → toda string é imutável.

```
public static void main(String[] args) {
    String str1 = new String ("Java");
    str1.replace("y", "c");
    System.out.println(str1);
}
```

Valor atual

Valor que vai substituir

```
run:
Java
CONSTRUÍDO COM SUCESSO (tempo total: 1 segundo)
```

Saída → não substituiu

Pacote java.lang → classe String

Para obter o resultado esperado, resgatar o retorno do método.

```
public static void main(String[] args) {
    String str1 = new String ("Java");
    String str2 = str1.replace("v", "c");
    System.out.println(str2);
}
```

Retorno do método

```
run:
Jaca
CONSTRUÍDO COM SUCESSO (tempo total: 1 segundo)
```

Pacote java.lang → classe String

Comparando strings: **sempre usar método equals()**

```
public static void main(String[] args) {
    String pessoa = new String ("Maria");
    String aluno = new String ("Maria");
    if (aluno==pessoa)
        System.out.println("objetos iguais");
    if (aluno.equals(pessoa))
        System.out.println("conteúdo dos objetos iguais");
}
```

Mesmo ”conteúdo”, objetos diferentes (diferentes referências)

Compara **referências** ao objeto

Método equals() compara ”**conteúdo**” de objetos

```
run:
conteúdo dos objetos iguais
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

Pacote java.lang → classe String

Comparando strings: **sempre usar método equals()**

```
public static void main(String[] args) {
    String pessoa = new String ("Maria");
    String aluno = pessoa;
    if (aluno==pessoa)
        System.out.println("objetos iguais");
    if (aluno.equals(pessoa))
        System.out.println("conteúdo dos objetos iguais");
}
```

```
run:
objetos iguais
conteúdo dos objetos iguais
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

Faz referência ao mesmo objeto

Compara **referências** ao objeto

Método equals() compara "**conteúdo**" de objetos

Orientação a Objeto

Pacote java.lang → classe String

Outros métodos:

```
run:
JAVA
java
J
true
false
true
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

```
public static void main(String[] args) {
    String str1 = new String ("Java");

    String maiusculas = str1.toUpperCase();
    System.out.println(maiusculas);

    String minusculas = str1.toLowerCase();
    System.out.println(minusculas);

    char carac = str1.charAt(0);
    System.out.println(carac);

    boolean terminaCom = str1.endsWith("a");
    System.out.println(terminaCom);

    boolean comecaCom = str1.startsWith("x");
    System.out.println(comecaCom);

    boolean semCase =str1.equalsIgnoreCase("JAVA");
    System.out.println(semCase);
}
```

Atributos e métodos static

Instâncias de classes (objetos) possuem valores diferentes para seus atributos

Classes precisam de atributos/métodos acessíveis a todas instâncias

Modificador static → atributo é compartilhado por todas instâncias → ”atributo da classe”

Orientação a Objeto

```java
public class Jogador {

    private int id;
    private String nome;
    private static int totalJogadores=0;

    public Jogador(int id, String nome)
    {
        this.setId(id);
        this.setNome(nome);
        totalJogadores= totalJogadores + 1;
        //setTotalJogadores(totalJogadores+1);
    }

    public static int getTotalJogadores() {
        return totalJogadores;
    }

    public static void setTotalJogadores(int aTotalJogadores) {
        totalJogadores = aTotalJogadores;
    }
```

Mesmo valor compartilhado para todas instâncias

```java
public class TesteStatic {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        Jogador jog1 = new Jogador (123, "Pelé");
        Jogador jog2 = new Jogador (456, "Maradona");
        Jogador jog3 = new Jogador (789, "Neymar");
        System.out.println(Jogador.getTotalJogadores());
    }

}
```

```
run:
3
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

No construtor, incrementa atributo ou chama método

Não usa **this**

Pacote java.lang → Classe Math

Atributos e métodos estáticos

max

```
public static int max(int a,
      int b)
```

Returns the greater of two `int` values. That is, the result is the argument closer to the value of `Integer.MAX_VALUE`. If the arguments have the same value, the result is that same value.

Parameters:

`a` - an argument.

`b` - another argument.

Returns:

the larger of `a` and `b`.

Pacote java.lang → Classe Math

```
public class TesteMath {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        int a=10, b=15, result;

        result = Math.max(a,b);
        System.out.println(result);

    }
```

Método static

Orientação a Objeto

```
public static void main(String[] args) {
    int a=10, b=15, result;
    double result2;

    result = Math.max(a,b);
    System.out.println(result);

    result2 = Math.cos(60);
    System.out.println(Math.round(result2));

    result2 = Math.pow(2, 4);
    System.out.println(result2);

    result2  = Math.random();
    System.out.println(result2);

    result  = (int) (1+ Math.random()*(10-1));
    System.out.println(result);
}
```

```
run:
15
-1
16.0
0.3987845954949173
4
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

Gera número aleatório de 0.0 a 1.0

Gera número aleatório de 1 a 10

Type cast

Exercício

1 - Crie uma classe chamada **Poupança**. Utilize um atributo static **taxaJurosAnual** para armazenar a taxa de juros anual para cada um dos correntistas. Crie um atributo privado **saldo** para indicar a quantia que os correntistas têm atualmente em depósito. Forneça um método **calcularJurosMensais**, que calcula os juros mensais multiplicando o **saldo** pelo **taxaJurosAnual** dividido por 12; esses juros devem ser adicionados a **saldo**. Forneça um método static **modificaTaxaJuro** que configura o static **taxaJurosAnual** com um novo valor. Escreva um programa para testar a classe **Poupança**. Instancie dois objetos diferentes: **poupança1** e **poupança2**, com saldos de R$2.000,00 e R$3.000,00, respectivamente. Configure o **taxaJurosAnual** como 3%. Em seguida, calcule os juros mensais e imprima os novos saldos de cada uma das poupanças. Então configure o **taxaJurosAnual** como 4%, calcule os juros do próximo mês e imprima os novos saldos para cada uma das poupanças.

Dúvidas?

tbasso@unicamp.br