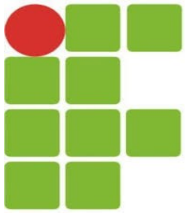


Instituto Federal

Campus Goiânia

Bacharelado em Sistemas de Informação

POO I

Prof. Dory Gonzaga Rodrigues

Ementa

- Classe System

A Classe java.lang.System

O pacote **java.lang** contém todas as classes da base da linguagem do Java, ou seja, as classes que pertencem ao pacote java.lang representam a linguagem Java na sua essência.

Vamos conhecer as funcionalidades de uma das classes mais utilizadas e mais antigas do Java, a classe **System** que está presente desde o jdk1.0.

Esta classe é conhecida pelos seus famosos métodos de entrada e saída de dados.

A Classe System

Esta classe é **FINAL**, ou seja, não pode ser estendida !!!

Todos os métodos são **ESTÁTICOS**, ou seja, são executados quando invocados sem a necessidade de uma instância do objeto.

Esta classe tem 3 variáveis/propriedades que são:

in: do tipo **InputStream**, armazena a entrada padrão do sistema, que pode ser, por exemplo, um teclado ou um arquivo de texto.

out: do tipo **PrintStream**, armazena a saída padrão de mensagens da aplicação.

err: do tipo **PrintStream**, armazena a saída padrão para mensagens de erro do sistema.

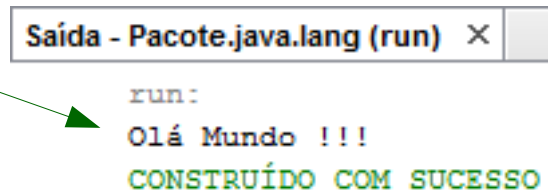
Ementa | Classe System

A variável OUT

A variável “out” armazena a saída de dados, que geralmente é feito no próprio console.

Para escrever no console podemos usar o método **print** ou **println**.

```
1  public class ClasseSystemOut {
2
3      public static void main(String[] args) {
4          System.out.println("Olá Mundo !!!");
5      }
6  }
```



A variável OUT

Por padrão, as saídas serão no console, mas podemos optar por mudar a saída padrão para um arquivo texto.

No exemplo a seguir, observe que no início do nosso método main criamos um objeto do tipo **PrintStream.**

Através do objeto **PrintStream** definimos que a saída deverá ocorrer em "/arq/teste.txt". Neste caso, informamos que o arquivo "teste.txt"será criado/acessado na pasta "arq" na raiz do nosso projeto.

E por fim, através do método **setOut()** dizemos qual será a nova saída do nosso programa para a **classe System**

Ementa | Classe System

A variável OUT

Neste exemplo teremos todas as saídas em um arquivo texto e nada no console.

```
1  import java.io.File;
2  import java.io.FileNotFoundException;
3  import java.io.FileOutputStream;
4  import java.io.PrintStream;
5
6  public class ClasseSystemOutArq {
7
8      public static void main(String[] args) throws FileNotFoundException {
9
10         File arquivo = new File("teste.txt");
11         FileOutputStream streamDados = new FileOutputStream(arquivo);
12         PrintStream escreveArq = new PrintStream(streamDados);
13
14         System.setOut(escreveArq);
15         System.out.println("Dory");
16         System.out.println("Caroline");
17     }
18 }
```

A variável IN

Sendo a variável “in”, uma variável que recebe a entrada padrão de dados da aplicação, conseguimos utilizar esta variável em diversas situações.

Vamos trabalhar a **variável IN** através da **classe Scanner**, que nos auxilia no tratamento da entrada de dados do usuário.

A classe **Scanner** irá capturar toda a entrada de dados definida pela classe **System**.

O **System.in** padrão é o próprio console, ou seja, a entrada de dados ocorre via teclado (digitação)

Utilizamos os métodos de Scanner para retornar o conteúdo digitado e armazenamos na variável desejada. No exemplo a seguir iremos utilizar o método **nextLine()** que retorna o conteúdo da linha onde o cursor está posicionado.

A variável IN

Neste exemplo utilizamos a System.in com a classe Scanner.

```
1  import java.util.Scanner;
2
3  public class ClasseSystemOutIn {
4
5      public static void main(String[] args) {
6
7          Scanner scan = new Scanner(System.in);
8          System.out.println("Digite: ");
9          String entrada = scan.nextLine();
10         System.out.println("Você digitou: " + entrada);
11
12     }
13 }
```

A variável IN

Por padrão, as entradas de dados ocorrem no console via teclado, mas podemos optar por mudar a saída padrão para um arquivo texto.

Neste caso o System.in dita ao Scanner qual a entrada ele deve escanear.

No exemplo a seguir, observe que no início do nosso método main criamos um objeto do tipo **ImputStream.**

Este objeto **ImpuStream** aponta para o arquivo “teste.txt” localizado dentro da pasta “arq” na raiz do nosso projeto.

E por fim, através do método **setIn()** dizemos que o arquivo apontado pelo objeto **InputStream** será a nova entrada de dados do nosso programa para a **classe System.**

A variável IN

Neste exemplo teremos todas as entradas de dados lidas do arquivo texto e nada no console.

```
1  import java.io.FileInputStream;..
5
6  public class ClasseSystemOutInArq {
7
8      public static void main(String[] args) throws FileNotFoundException {
9
10         InputStream arquivo = new FileInputStream("teste.txt");
11         System.out.println("Aberto arquivo...");
12         System.setIn(arquivo);
13         Scanner scan = new Scanner(System.in);
14         System.out.println("Conteúdo do arquivo: " + scan.nextLine());
15     }
16 }
```

Ementa | Classe System

A variável ERR

Este segue a mesma estrutura do “System.out”, sendo que é utilizado para impressão de textos de erro.

```
1  import java.io.FileInputStream;
6
7  public class ClasseSystemOutInErrArq {
8
9      public static void main(String[] args) throws FileNotFoundException {
10         try {
11                 InputStream arquivo = new FileInputStream("teste.txt");
12                 System.out.println("Aberto arquivo...");
13                 System.setIn(arquivo);
14                 Scanner scan = new Scanner(System.in);
15                 System.out.println("Conteúdo do arquivo: " + scan.nextLine());
16
17         }
18         catch (IOException e) {
19                 System.err.println("Falha ao tentar abrir o arquivo:");
20         }
21     }
22 }
```