

Introdução a Java

Parte II

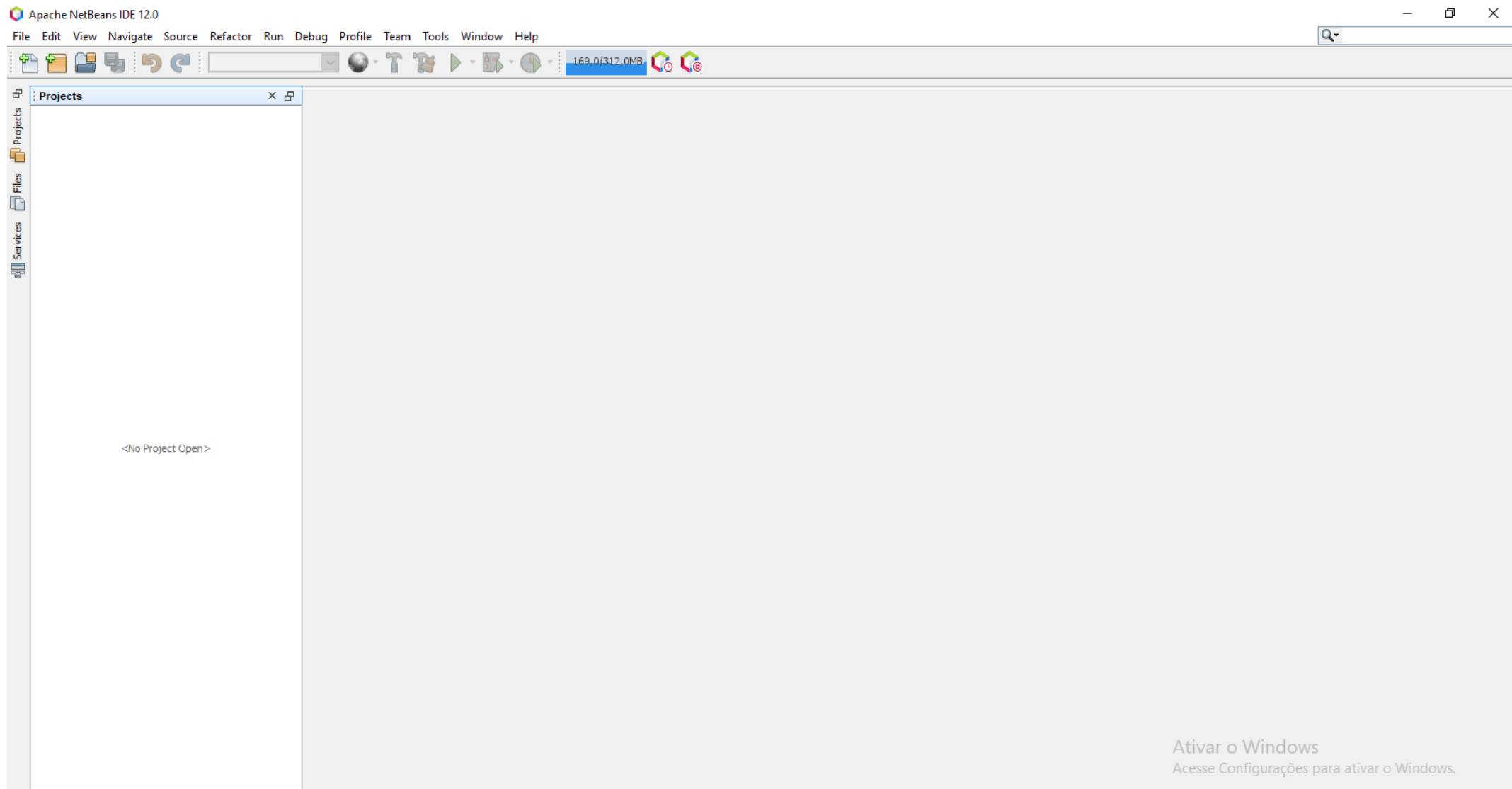
Conhecendo o Netbeans



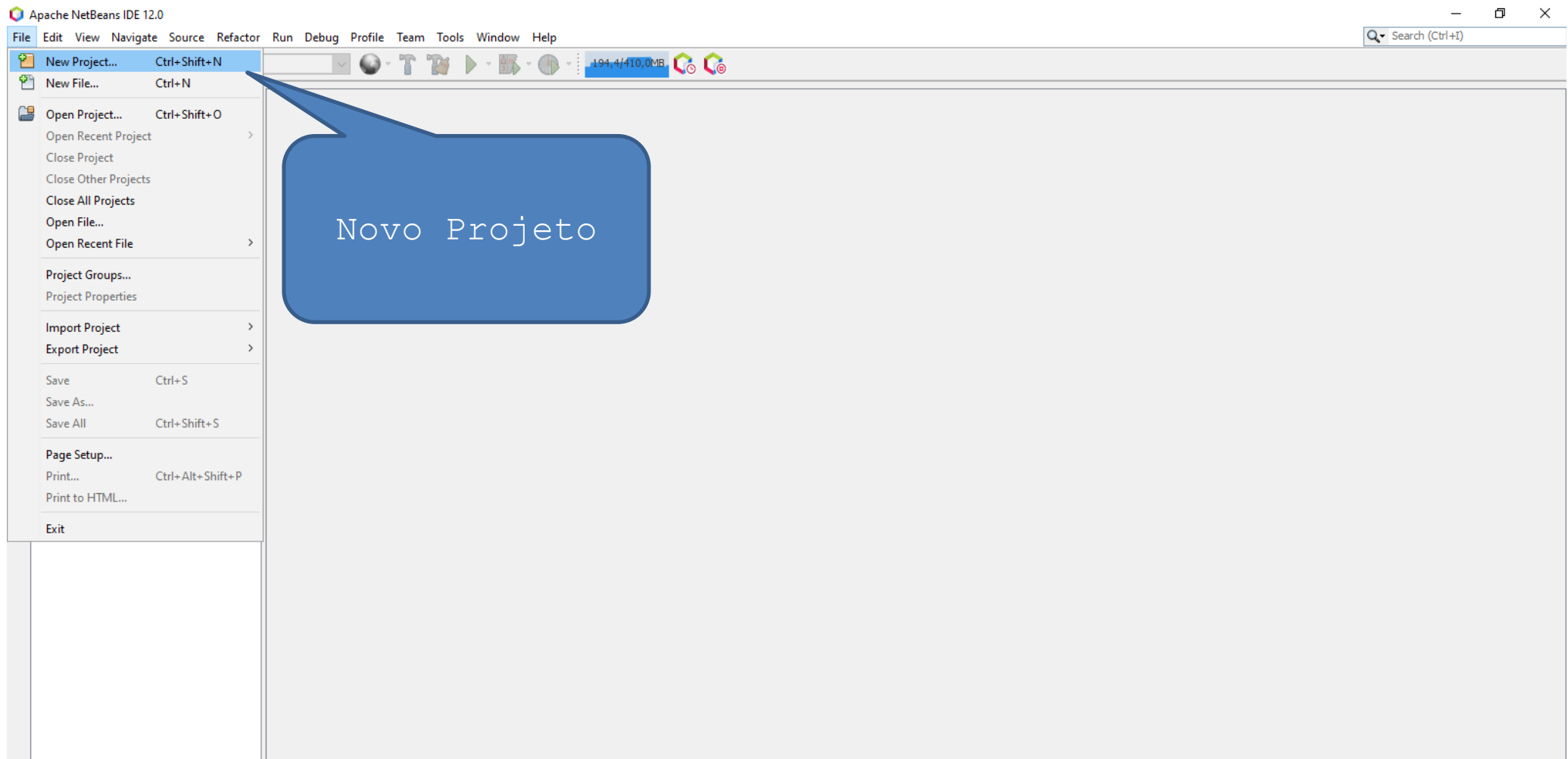
Apache

NetBeans IDE

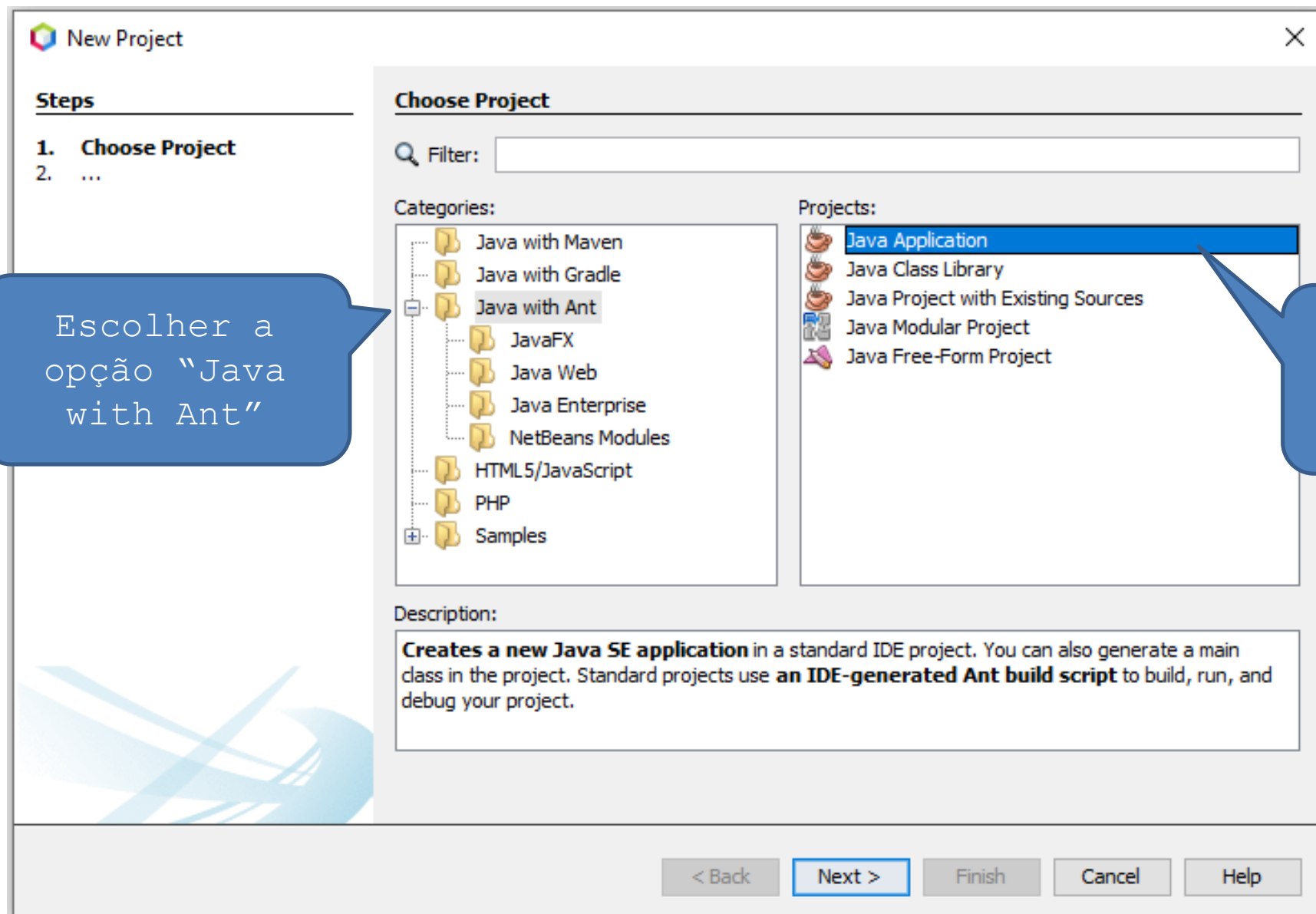
Conhecendo o Netbeans



Criando um novo projeto Java SE



Criando um novo projeto Java SE



Escolher a opção "Java with Ant"

"Java Application"

Criando um novo projeto Java

New Java Application

Steps

1. Choose Project
2. **Name and Location**

Name and Location

Project Name:

Project Location:

Project Folder:

☐ Use Dedicated Folder for Storing Libraries

Libraries Folder:

Different users and projects can share the same compilation libraries (see Help for details).

☒ Create Main Class

< Back Next > **Finish** Cancel Help

Criando um novo projeto Java SE

Seus projetoss

```
1  /*
2   * To change this license header, choose License Headers in Project Properties.
3   * To change this template file, choose Tools | Templates
4   * and open the template in the editor.
5   */
6  package primeiroprojeto;
7
8  /**
9   *
10   * @author joao_
11   */
12  public class PrimeiroProjeto {
13
14      /**
15       * @param args the command line arguments
16       */
17      public static void main(String[] args) {
18          // TODO code application logic here
19      }
20
21  }
22
```

Escreva o código aqui !!!

Comando de saída de dados em Java

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package primeiroprojeto;

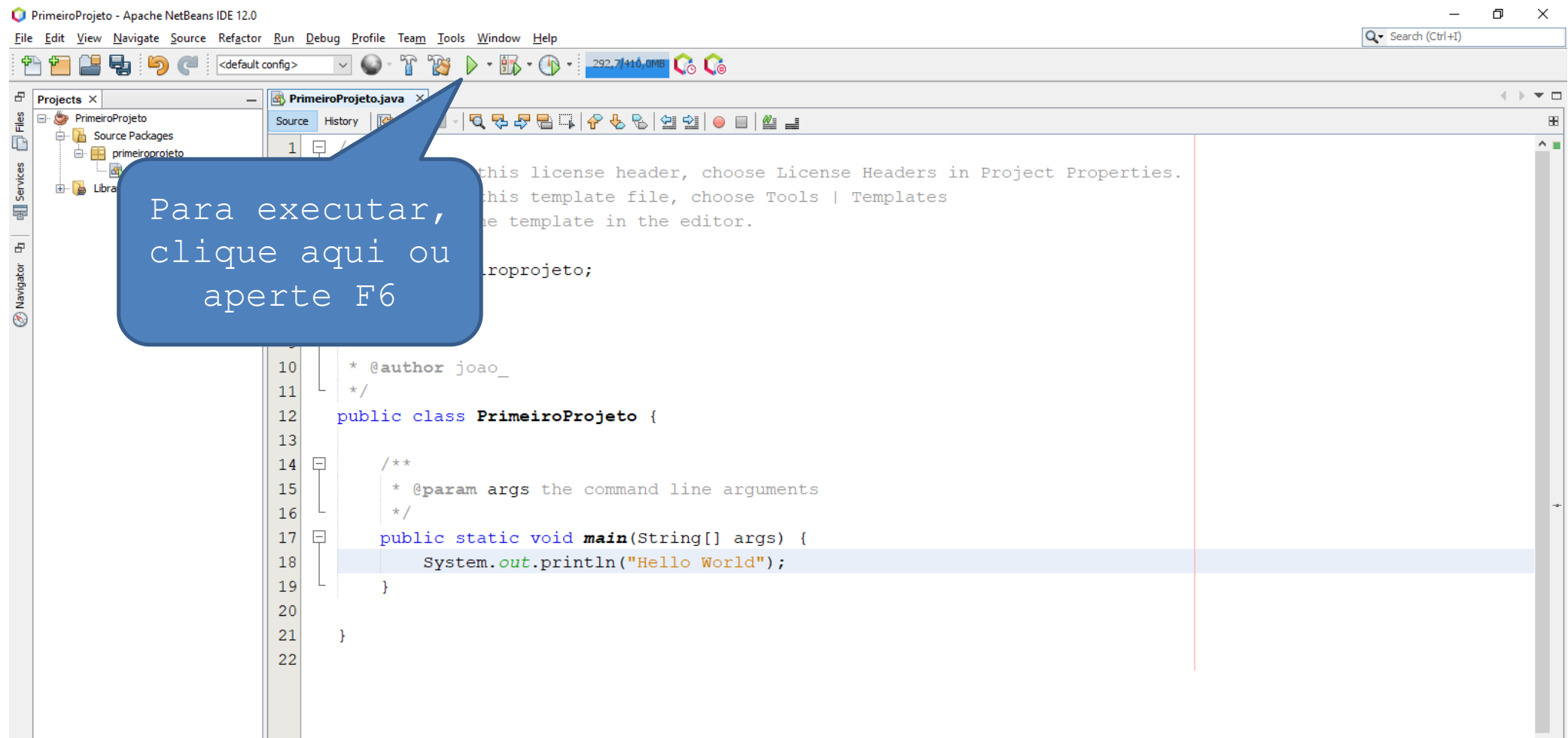
/**
 *
 * @author joao_
 */
public class PrimeiroProjeto {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}
```

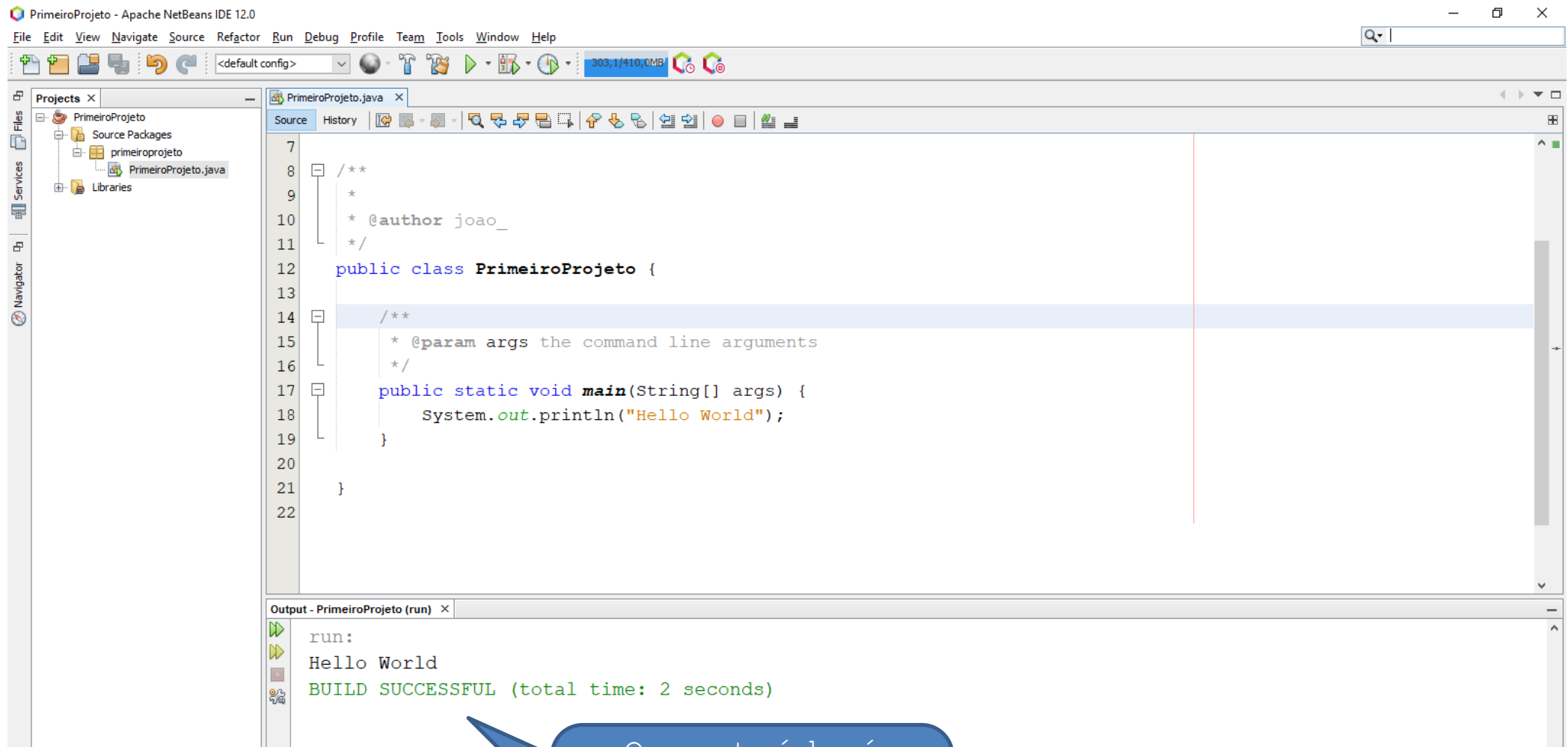
Imprime a mensagem
dentro do ().

Atalho: `sout+tab`

Compilando o projeto Java SE



Compilando o projeto Java SE



O conteúdo é exibido na janela output da IDE

Concatenação em Java

```
public class Exemplo1 {  
  
    public static void main(String[] args) {  
        System.out.println("Hello World" + "Gabriel");  
    }  
  
}
```

Utilizamos o
símbolo de "+"
para
concatenar.

Mas e se quisermos concatenar um texto seguido de uma soma?

Exemplo:

```
public class Exemplo01 {  
  
    /**  
     * @param args the command line arguments  
     */  
    public static void main(String[] args) {  
  
        System.out.println("Matrícula de número: " + 1 + 3);  
  
    }  
  
}
```

Como Resolver?

run:

Matrícula de número: 13

BUILD SUCCESSFUL (total time: 0 seconds)

Solução:

```
public class Exemplo01 {  
  
    /**  
     * @param args the command line arguments  
     */  
    public static void main(String[] args) {  
  
        System.out.println("Matrícula de número: " + (1 + 3));  
  
    }  
  
}
```

run:

Matrícula de número: 4

BUILD SUCCESSFUL (total time: 1 second)

Outro comando para saída de dados (print):

```
public class Exemplo1 {  
  
    public static void main(String[] args) {  
        System.out.print("Hello World");  
        System.out.println("Gabriel");  
    }  
}
```

Não houve uma quebra de linha da primeira saída para segunda.

run:

Hello WorldGabriel

BUILD SUCCESSFUL (total time: 0 seconds)

Quebra de linha com Java

```
public class Exemplo1 {  
  
    public static void main(String[] args) {  
        System.out.print("Hello World\n");  
        System.out.println("Gabriel");  
    }  
}
```

Para quebrar linha no Java, utilizamos do "\n"

run:

Hello World

Gabriel

BUILD SUCCESSFUL (total time: 0 seconds)

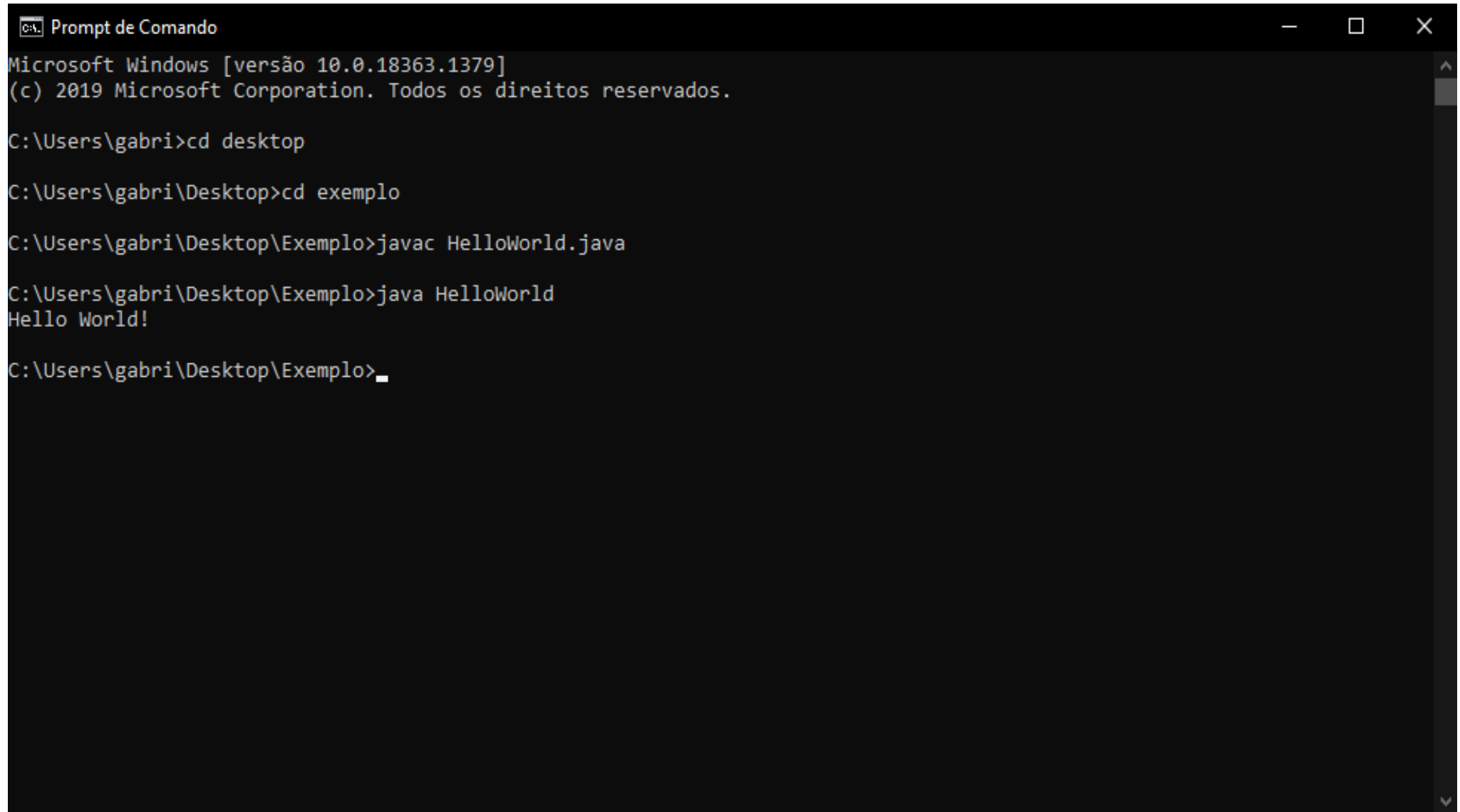
Fazendo a mão...

- Crie uma pasta na Área de Trabalho;
- Abra o Notepad (Bloco de Notas);
- Escreva o seguinte programa;

```
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello, World");  
    }  
}
```

- Salve o arquivo dentro da pasta criada como HelloWorld.java

Comandos javac e java (sem pacote)



```

C:\> Prompt de Comando
Microsoft Windows [versão 10.0.18363.1379]
(c) 2019 Microsoft Corporation. Todos os direitos reservados.

C:\Users\gabri>cd desktop

C:\Users\gabri\Desktop>cd exemplo

C:\Users\gabri\Desktop\Exemplo>javac HelloWorld.java

C:\Users\gabri\Desktop\Exemplo>java HelloWorld
Hello World!

C:\Users\gabri\Desktop\Exemplo>_

```

Curiosidade

O que significa `public static void main(String[] args)`?

`public`

É o modificador de acesso do método. Usando este modificador o método pode ser acessado por qualquer classe dentro (e fora) do projeto.

Outros modificadores são `protected`, `private` ou **sem modificador***. [Aqui](#) pode-se ler mais sobre os modificadores de acesso do Java.

`static`

Define o método como **estático**, isso quer dizer que a classe não precisa ser instanciada para chamar este método.

No exemplo, tenho a classe `Cliente` com os métodos (estático) `FazerAlgo()` e (não-estático) `FazerAlgoDois()`, o uso seria assim:

```
Cliente cliente = new Cliente();  
cliente.FazerAlgoDois(); // Este é o método não-estático  
  
Cliente.FazerAlgo(); // Este é o método estático
```

Curiosidade

`void`

É tipo de retorno do método. Este tipo de retorno significa vazio/nada, o método não dá retorno nenhum. Os métodos podem retornar qualquer tipo do seu projeto, até mesmo os criados por você.

`main`

É o nome do método. Todo e qualquer método precisa ter um nome. Os nomes são definidos pelo programador e geralmente seguem algum padrão convencional definido previamente pela linguagem ou pela comunidade, embora isso seja opcional. [No caso do Java, a própria Oracle define estas convenções](#). A convenção referente a nomeação de métodos diz:

Methods should be verbs, in mixed case with the first letter lowercase, with the first letter of each internal word capitalized.

Em tradução livre:

Métodos devem ser verbos, em "mixed case" com a primeira letra minúscula e a primeira letra das palavras internas em letra maiúscula.

No Java (e em outras linguagens também) o `main` é o ponto de entrada da aplicação. É o método que a JRE procura para executar a aplicação. Por isso, em alguns tipos de aplicação (como Swing ou console) é obrigatório tê-lo implementado. É possível ver mais detalhes sobre isso em [Por que é obrigatório implementar "public static void main \(String \[\] args\)"?](#)

Curiosidade

`(String[] args)`

Define que o método deve receber como parâmetro um array de `String` (nomeado `args`). Nesse caso específico: este parâmetro serve para caso seu programa precise receber algum valor como argumento, isso é muito comum quando o programa é iniciado por outro programa ou pelo terminal (CMD, Shell, Bash, etc.).

Um exemplo muito comum é o `Git`. Quando você digita `git commit` no seu terminal está chamando o `Git` com o parâmetro `commit`. Todas as "strings" que vierem depois do nome do programa serão recebidas pelo mesmo dentro do `array (args)`. Geralmente a primeira posição do array é o caminho que a aplicação se encontra.

Resposta retirada do StackOverflow

<https://pt.stackoverflow.com/questions/93048/o-que-significa-public-static-void-mainstring-args>

Conhecendo JavaDoc para esclarecer duvidas

The screenshot shows the Oracle Java Platform Standard Edition 8 API Specification website. The browser address bar displays the URL <https://docs.oracle.com/javase/8/docs/api/>. The page features a navigation sidebar on the left with links for 'All Classes', 'All Profiles', and a list of 'Packages' including `java.applet`, `java.awt`, `java.awt.color`, `java.awt.datatransfer`, and `java.awt.dnd`. Below the sidebar, the main content area is titled 'Java™ Platform, Standard Edition 8 API Specification'. It includes a description: 'This document is the API specification for the Java™ Platform, Standard Edition.' and a link to 'See: Description'. A 'Profiles' section lists `compact1`, `compact2`, and `compact3`. A 'Packages' table is also present, detailing the purpose of each package.

Package	Description
<code>java.applet</code>	Provides the classes necessary to create an applet and the classes an applet uses to communicate with its applet context.
<code>java.awt</code>	Contains all of the classes for creating user interfaces and for painting graphics and images.
<code>java.awt.color</code>	Provides classes for color spaces.
<code>java.awt.datatransfer</code>	Provides interfaces and classes for transferring data between and within applications.
<code>java.awt.dnd</code>	Drag and Drop is a direct manipulation gesture found in many Graphical User Interface systems that provides a mechanism to transfer information between two entities logically associated with presentation elements in the GUI.

<https://docs.oracle.com/javase/8/docs/api/>

Exercícios

1. Instale o Netbeans
2. Crie um projeto chamado "PrimeiraAula"
3. Faça um programa que imprima a seguinte mensagem:
"Minha primeira aula de POO com Java"
4. Agora altere sua aplicação para que ela imprima a seguinte mensagem utilizando o comando print e a quebra de linha:
"Minha primeira aula de POO com Java.
Estamos usando o Netbeans!"
5. **Desafio:** imprima a mesma mensagem do exercício 4 utilizando três comandos de impressão.

Obrigado !