**Programação I**

# Aula 1: A tecnologia Java

## Apresentação

Nesta aula, compreenderemos como surgiu a linguagem de programação Java. Abordaremos como a linguagem Java foi importante no processo de portabilidade no desenvolvimento de sistemas. Além disso, destacaremos as principais características da linguagem, o ambiente de desenvolvimento e descreveremos o processo de instalação e criação de um projeto nas ferramentas IDEs NetBeans e Eclipse.

## Objetivos

- Conhecer como surgiu a linguagem Java.
- Identificar as principais características da linguagem Java;
- Descrever o ambiente de programação Java.

## Primeiras palavras

Hoje em dia, o desenvolvimento de sistemas se baseia em vários e diferentes paradigmas, tais como os listados a seguir:

IMPERATIVO

Segue sequências de comandos ordenados segundo uma lógica.

FUNCIONAL

Trabalha com a divisão de problemas através de funções, que resolvem separadamente problemas menores e que, ao serem organizados, resolvem o problema como um todo.

LÓGICO

Voltado ao desenvolvimento de problemas de lógica e usado em sistemas de inteligência computacional.

ORIENTADO A OBJETOS

Define um conjunto de classes para dividir o problema e realiza a interação entre as diferentes classes para também resolver o problema como um todo.

# Histórico da linguagem Java

Clique no botão acima.

## Histórico da linguagem Java

A tecnologia Java foi desenvolvida na década de 1990, a partir de um projeto pessoal de um funcionário da Sun Microsystems. A ideia inicial estava ligada à criação de uma linguagem de programação que pudesse ser utilizada em diferentes sistemas, alterando o paradigma de que uma aplicação só poderia ser desenvolvida para uso em um único ambiente de *hardware* e sistema operacional, como era bastante comum na época.

As grandes empresas desenvolviam suas aplicações voltadas para seu ambiente de *hardware* e *software* (sistema operacional - SO), e estas aplicações não eram capazes de serem executadas em diferentes plataformas, principalmente de outros fabricantes. Se analisarmos a linguagem C, criada junto com o sistema operacional UNIX, temos uma biblioteca muito vasta de funções, mas poucas são consideradas padrão para atender a diferentes sistemas; e, mesmo assim, uma aplicação compilada em um sistema operacional (ambiente) não pode ser executada em outro.

A linguagem Java rompeu este paradigma e passou a permitir que uma aplicação desenvolvida em um ambiente - *hardware* + *software* (SO) - possa ser executada em outro sem necessidade de qualquer outro procedimento. A Sun Microsystems, ao tomar conhecimento desta ideia, deu total apoio ao seu desenvolvimento e criou um grupo com 13 membros, liderado por James Gosling, que passaram a trabalhar exclusivamente neste projeto. A equipe foi batizada de “Green Team” e o grupo passou a trabalhar em um conjunto de escritórios fora das dependências físicas da Sun, e sem qualquer tipo de comunicação com a matriz, durante 18 meses para a concretização desta ideia.

Com a tecnologia Java, as aplicações passaram a ser portáveis de um sistema para o outro, sem nenhuma necessidade de alteração. Por isso, afirmamos que a portabilidade é uma das mais importantes características da linguagem Java.

Ainda naquela época, o grupo já havia antecipado uma *nova onda* na computação, na convergência entre dispositivos controlados digitalmente e computadores. Hoje em dia, percebemos bem isso quando analisamos um smartphone, um dispositivo digital que possui inúmeras funções de computadores; entre elas, podemos destacar a execução de aplicativos. Inicialmente, a linguagem foi batizada de **Oak**, pois o grupo tinha como vista da janela do escritório um carvalho. Posteriormente, a linguagem foi rebatizada como **Java**, em função do gosto do grupo pelo tipo de café. Por isso, temos como ícone da linguagem uma xícara de café com sua fumaça característica.

A linguagem é muito poderosa para o desenvolvimento de aplicações, seja para o desenvolvimento de aplicações menos sofisticadas ou para uso em dispositivos menos complexos que computadores, conhecidos como dispositivos inteligentes, tais como cafeteiras, micro-ondas, geladeiras e uma gama de outros dispositivos que possam ser controlados por software. A linguagem ainda é muito eficiente no desenvolvimento de sistemas de entretenimento doméstico, dando suporte a streaming de vídeo e televisão digital, que ainda não era tão desenvolvida na época.

A tecnologia Java permite ainda o desenvolvimento de todos os tipos de aplicações, indo do mais simples controle de um eletrodoméstico, passando por aplicações domésticas, comerciais, de automação, até o desenvolvimento de aplicações mais complexas, com comunicação de dados e aplicações para supercomputadores.

A linguagem Java teve início ao incorporar a tecnologia Java ao navegador de internet Netscape navigator, em sua versão de 1995. A tecnologia ganhou a aceitação do mercado e dos desenvolvedores, sendo uma das mais importantes linguagens de programação para o desenvolvimento de sistemas. São dezenas de milhões de desenvolvedores Java no mundo e, atualmente, esta tecnologia é encontrada em supercomputadores, servidores, desktops, notebooks, máquinas de cartões de crédito e débito, robôs, automóveis, jogos eletrônicos, bem como uma gama de dispositivos digitais, redes e demais tecnologias de programação. A linguagem Java ainda é a linguagem nativa para o desenvolvimento de aplicações para o Android (sistema operacional para smartphones).

A tecnologia Java foi totalmente gratuita por muito tempo, mas recentemente a Oracle, que passou a deter os direitos da linguagem após adquirir a Sun Microsystems, está licenciando o uso para empresas com custos. A empresa deve permitir o licenciamento gratuito somente para desenvolvedores avulsos que criam aplicações pessoais sem custo ou para simples aprendizado.

## Principais características e vantagens da tecnologia Java

Orientada a objetos, com uma grande diversidade de bibliotecas de classes disponível;
• Independe de plataforma: *write once, run everywhere* ;
• Segurança - Mecanismos para sistemas livres de vírus, pacotes para criptografia;
• Simplicidade;
• Sintaxe dos comandos básicos segue o padrão do C;
• Sintaxe da parte OO bem mais simples que o C++;
• Internacionalização;
• Unicode: padrão que permite manipular textos de qualquer sistema de escrita;
• Robustez;
• Tratamento de exceções;
• JVM impede que uma aplicação mal comportada paralise o sistema;
• Distribuída e multitarefa;
• Em inglês: escreva uma vez, rode em qualquer lugar.
• Os programas podem utilizar recursos da rede com a mesma facilidade que acessam arquivos locais;
• Trabalha com diversos protocolos (TCP/IP, HTTP, FTP);
• Execução simultânea de múltiplas *threads*;
• Gerenciamento de memória;
• Memória virtual gerenciada pela JVM;
• *Garbage collection* (limpeza de memória);
• Desempenho;
• Mais rápida que linguagens de *script*, porém mais lenta que as linguagens compiladas puras;
• Hoje, os problemas de desempenho são resolvidos com compilação *just-in-time*.

**Atenção!** Aqui existe uma videoaula, acesso pelo conteúdo online

## Plataforma Java

De forma geral, entendemos que plataforma (ambiente de execução) é composta por hardware + software básico (sistema operacional).

A plataforma Java é definida apenas em software e possui dois componentes:

1

Máquina Virtual Java (*JVM - Java Virtual Machine*)

2

Conjunto de bibliotecas que disponibilizam classes comuns

## API Java

"Diferentemente das linguagens convencionais, que são compiladas para código nativo, a linguagem Java é compilada para "*bytecode*" (gerando o .class ou .jar), que é executado por uma máquina virtual Java (JVM - *Java Virtual Machine*)."

O modelo inicial era interpretado. Já o atual trocou a etapa do interpretador por uma 2ª compilação (compilador JIT, isto é, *just-in-time*).

**A tecnologia Java é composta por três plataformas:**
**1. J2SE ou Java SE** *(Java Standard Edition)*: base da plataforma, inclui o ambiente de execução e as bibliotecas comuns;
**2. J2EE ou Java EE** *(Java Enterprise Edition)*: versão voltada para o desenvolvimento de aplicações corporativas e aplicações web;
**3. J2ME ou Java ME** *(Java Micro Edition)*: versão voltada para o desenvolvimento de aplicações móveis ou embarcadas.

## Ambiente de desenvolvimento

Clique nos botões para ver as informações.

*Java Development Kit (JDK)*

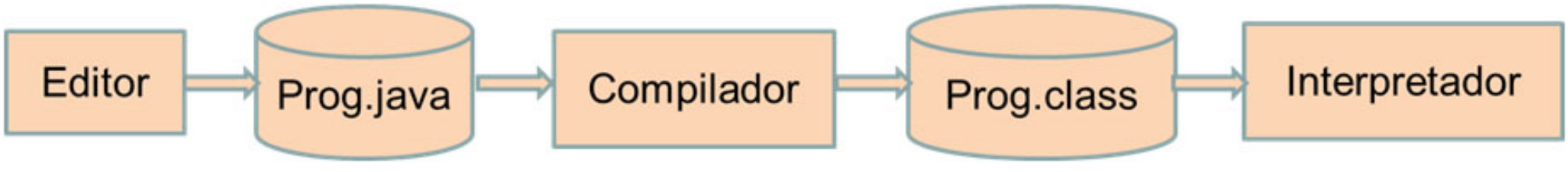
Coleção de programas para, dentre outras tarefas, compilar e executar aplicações Java. Este é o kit necessário para o desenvolvedor, pois contém todo o suporte para a criação de aplicações em Java.

Exemplo:
• Javac (compilador Java);
• Javadoc (utilitário para documentação);
• Java;
• Outros.

*Java Runtime Environment (JRE)*

Kit com todos os programas necessários para executar aplicações Java. Faz parte do JDK, mas pode ser instalado separadamente para execução em máquinas clientes, uma vez que o JDK é voltado para os desenvolvedores. O JRE pode ser instalado separadamente e dá suporte somente a execução de aplicações, por isso é a versão mais indicada para instalação nas máquinas clientes que apenas executarão aplicações, não sendo responsáveis pelo seu desenvolvimento.

## Fases de um programa em linguagem Java



O código de um programa Java é compilado apenas uma vez, gerando um código intermediário, o ***bytecode***, que pode ser executado quantas vezes forem necessárias em qualquer ambiente que possua uma máquina virtual Java (JVM) disponível.

Inicialmente a tecnologia Java realizava uma interpretação completa do *bytecode*, mas atualmente o interpretador realiza uma **compilação *just-in-time*** (compila o *bytecode* para o ambiente onde ocorrerá a execução), permitindo aumentar o desempenho da aplicação.

Para o desenvolvimento de aplicações em Java é comum o uso de ferramentas **IDEs** (*Integrated Development Environment*), que facilitam a codificação e a realização de testes, sendo as mais conhecidas:

- **Eclipse;**
- **NetBeans;**
- **IntelliJ;**
- **BlueJ.**

# Ambiente de Programação

Existem várias ferramentas para o desenvolvimento de sistemas utilizando a linguagem Java, mas os desenvolvedores têm preferência pelos IDEs Netbeans e Eclipse. Ambos são gratuitos e podem ser adquiridos pela internet através de download.

É importante que você já tenha instalado o JDK antes de instalar o seu IDE escolhido (Netbeans ou Eclipse).

**Atenção!** Aqui existe uma videoaula, acesso pelo conteúdo online

## Netbeans

Existem dois arquivos diferentes: o primeiro, com o ***source***, contém os códigos fonte no Netbeans e não é o ideal para trabalharmos o desenvolvimento. A versão adequada para nós é a versão bin, que contém todos os códigos já compilados e prontos para a execução e desenvolvimento de projetos e aplicações Java.

Em suas últimas versões, o Netbeans não usa instalador, mas você deve descompactar o arquivo do Netbeans (Bin - executável) em uma pasta em seu computador. Uma vez o Java JDK instalado, você poderá executar o Netbeans tranquilamente.

### Dica

Clique para baixar o **Netbeans**.

## Eclipse

O Eclipse tem como padrão a não necessidade de instalador, pois sua instalação é feita por meio da descompactação do pacote em uma pasta em seu computador.

É comum para usuários Eclipse instalar em um pendrive e, quando necessário, executá-lo diretamente do mesmo. Atualmente, procedimento idêntico pode ser feito com o Netbeans.

### Dica

Clique para baixar o **Eclipse**.

Você não precisa e não deve instalar as duas versões, pois elas são concorrentes e possuem o mesmo objetivo. Normalmente, cada desenvolvedor tem sua preferência por uma delas.

A linguagem Java possui uma base de construção semelhante à linguagem C e, por isso, boa parte de sua estrutura e sintaxe se assemelha a ela. Desta forma, programadores com conhecimento nesta linguagem tem grande facilidade com a sintaxe da linguagem Java. Outra importante semelhança está nas estruturas de controle de fluxo, que são construídas da mesma forma em ambas as linguagens.

### Atenção

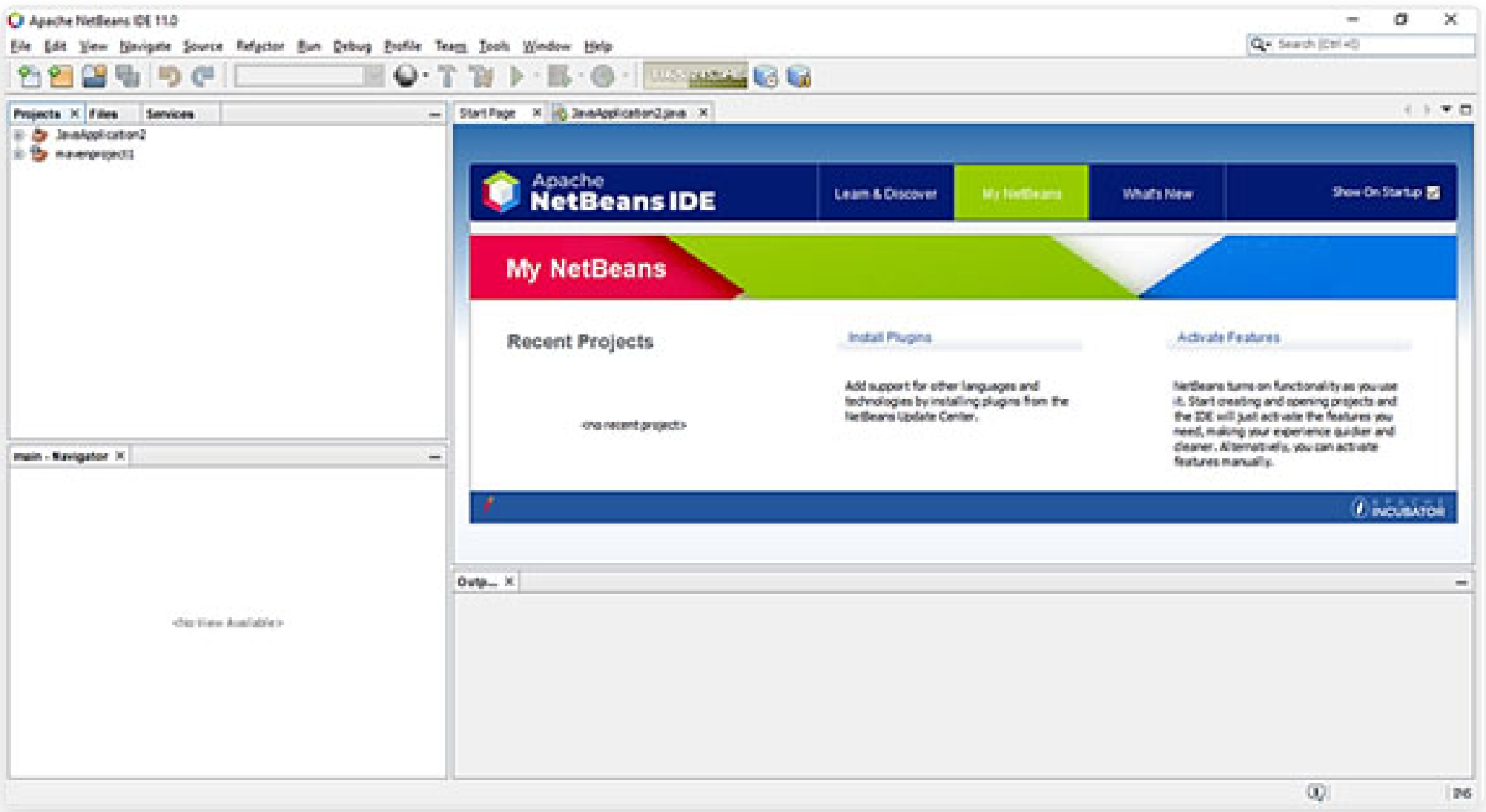
Cuidado com as diferenças de versões no sistema operacional: se instalar o Java para 64 bits, você deverá usar um IDE (Netbeans ou Eclipse) de 64 bits. O mesmo para a versão de 32 bits: tanto o Java quanto o IDE deverão ser para 32 bits.

A seguir, aprenderemos a criar uma aplicação Java básica em cada um deles para que você possa escolher o de sua preferência.

## Exemplos de aplicação utilizando o Netbeans

No Netbeans, a prática foi realizada na versão 11 (mais atual):


Tela padrão do Netbeans:

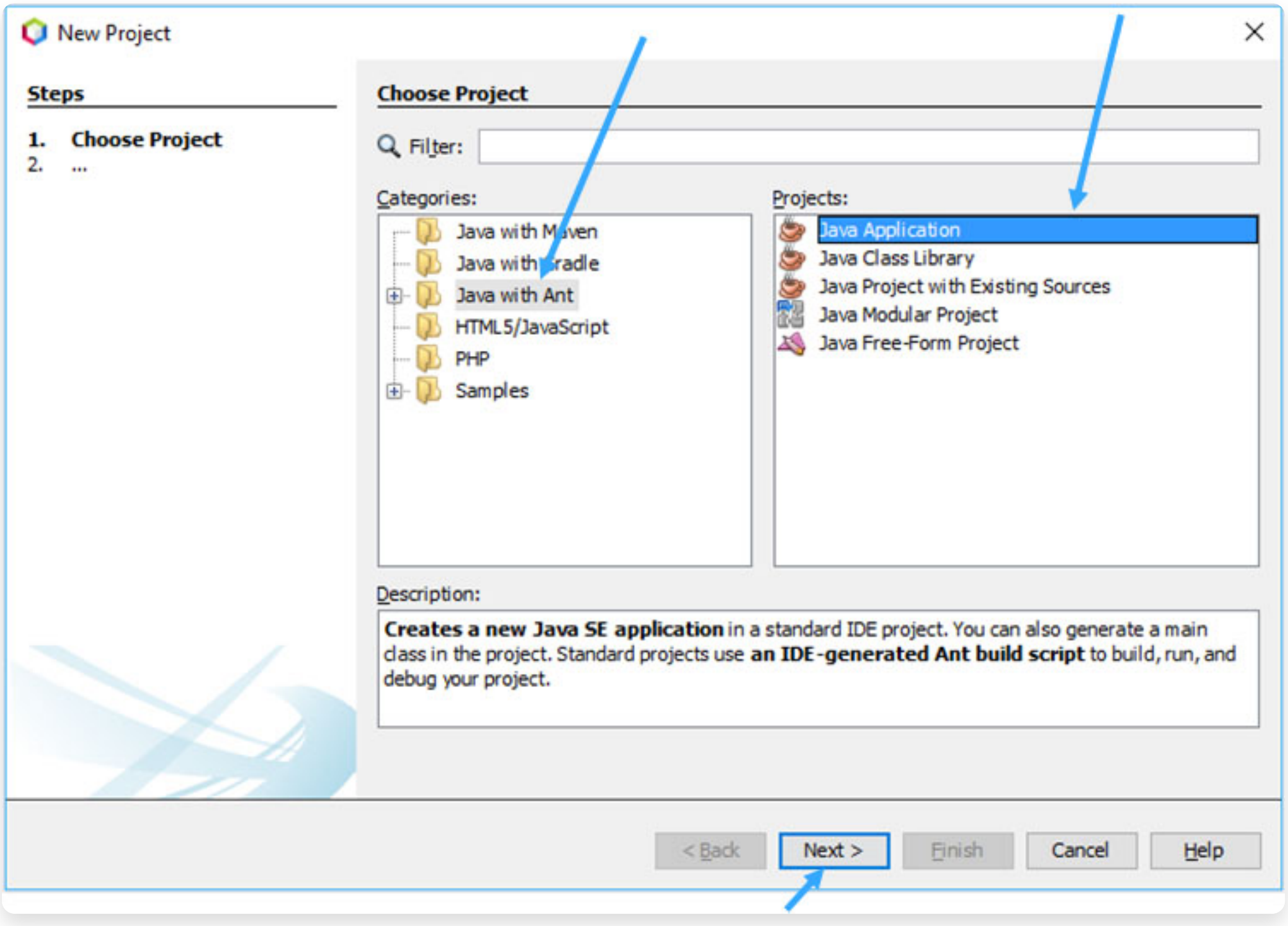



## Para criar um novo projeto:

No menu *File >New Project*.

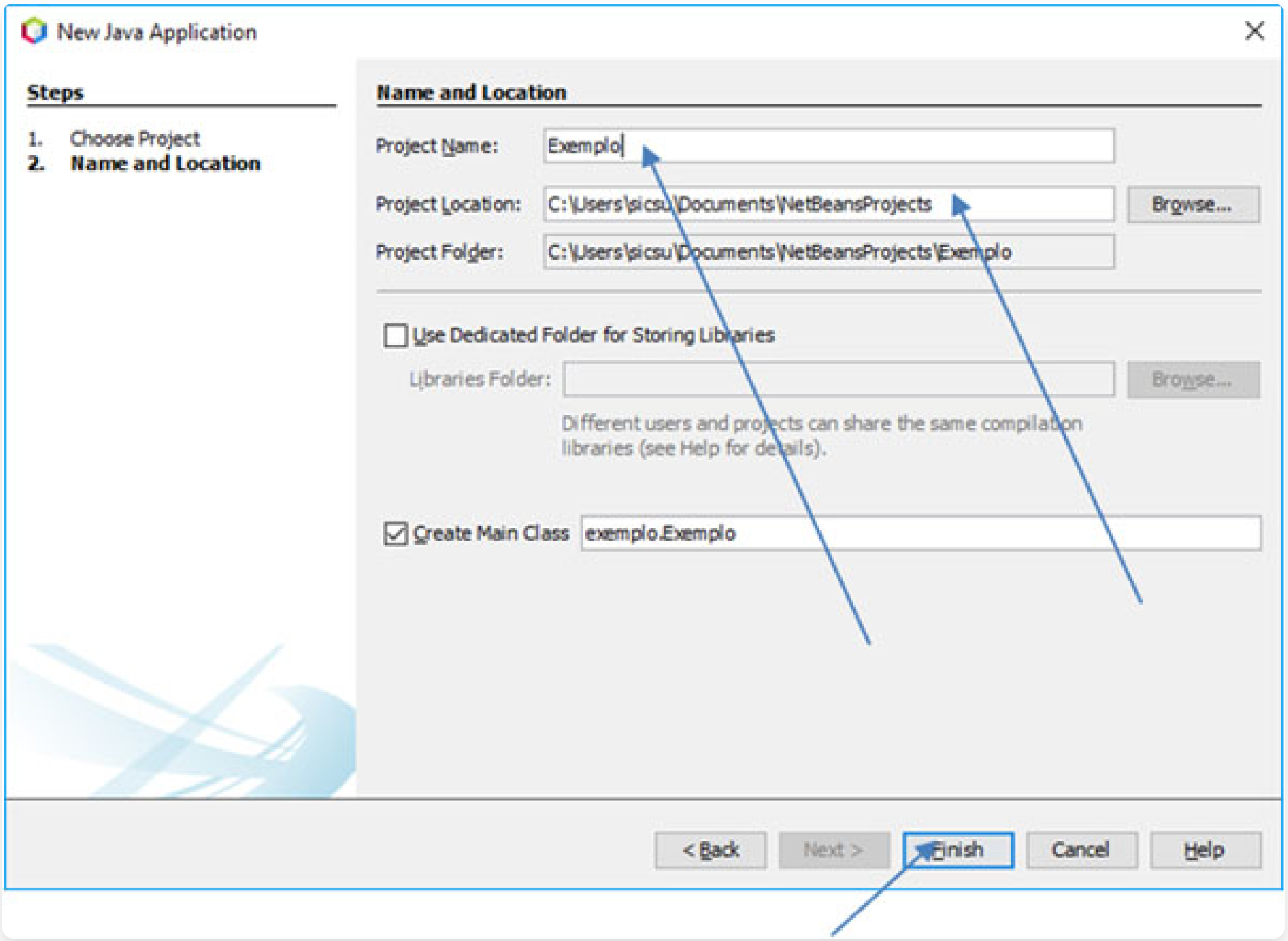
## Tela de criação do projeto:

Para os nossos projetos usaremos sempre Java with Ant e Java Application.



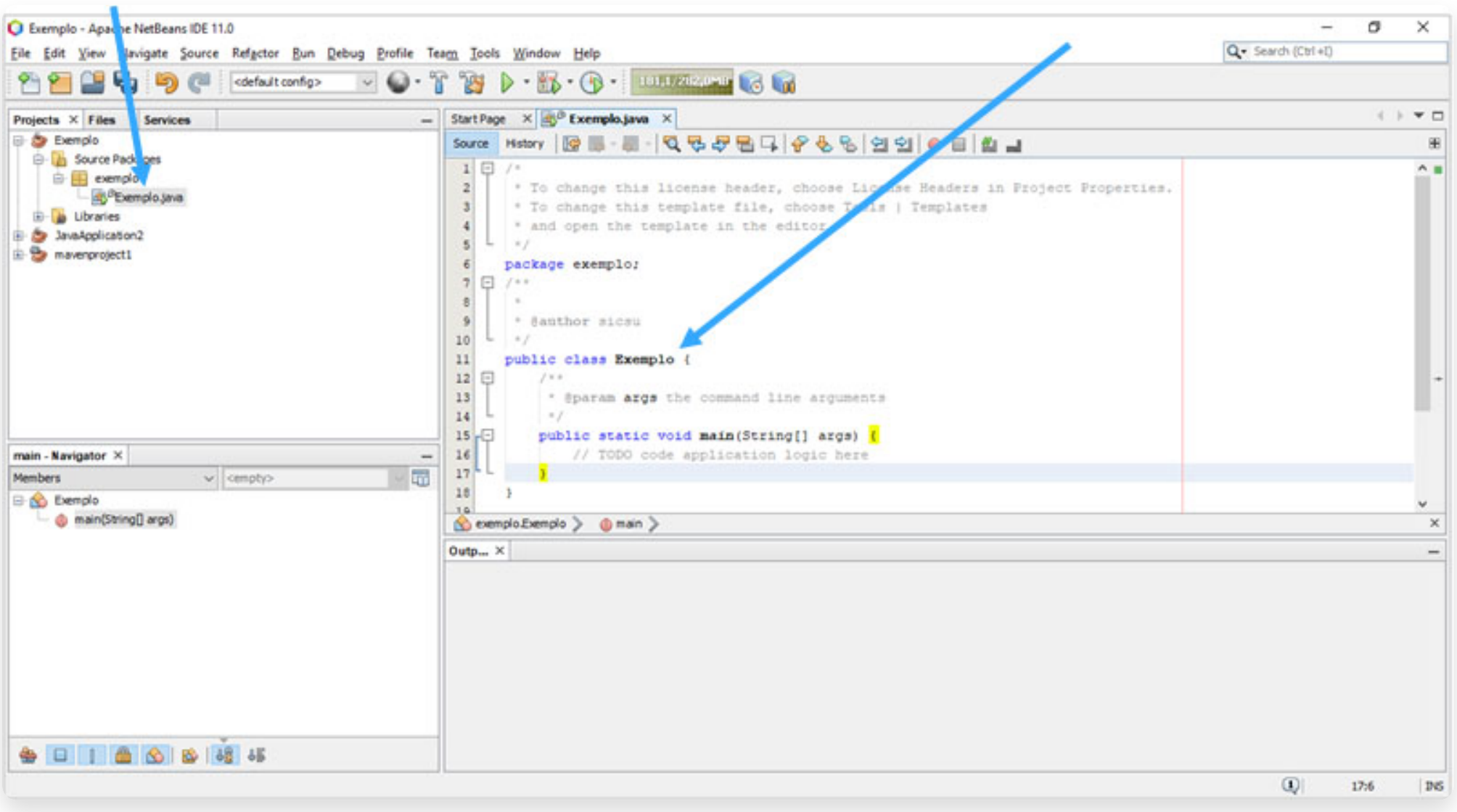

Clique no botão *Next* para prosseguiremos com a criação do projeto.

Defina o nome do Projeto e o local onde o mesmo será criado.

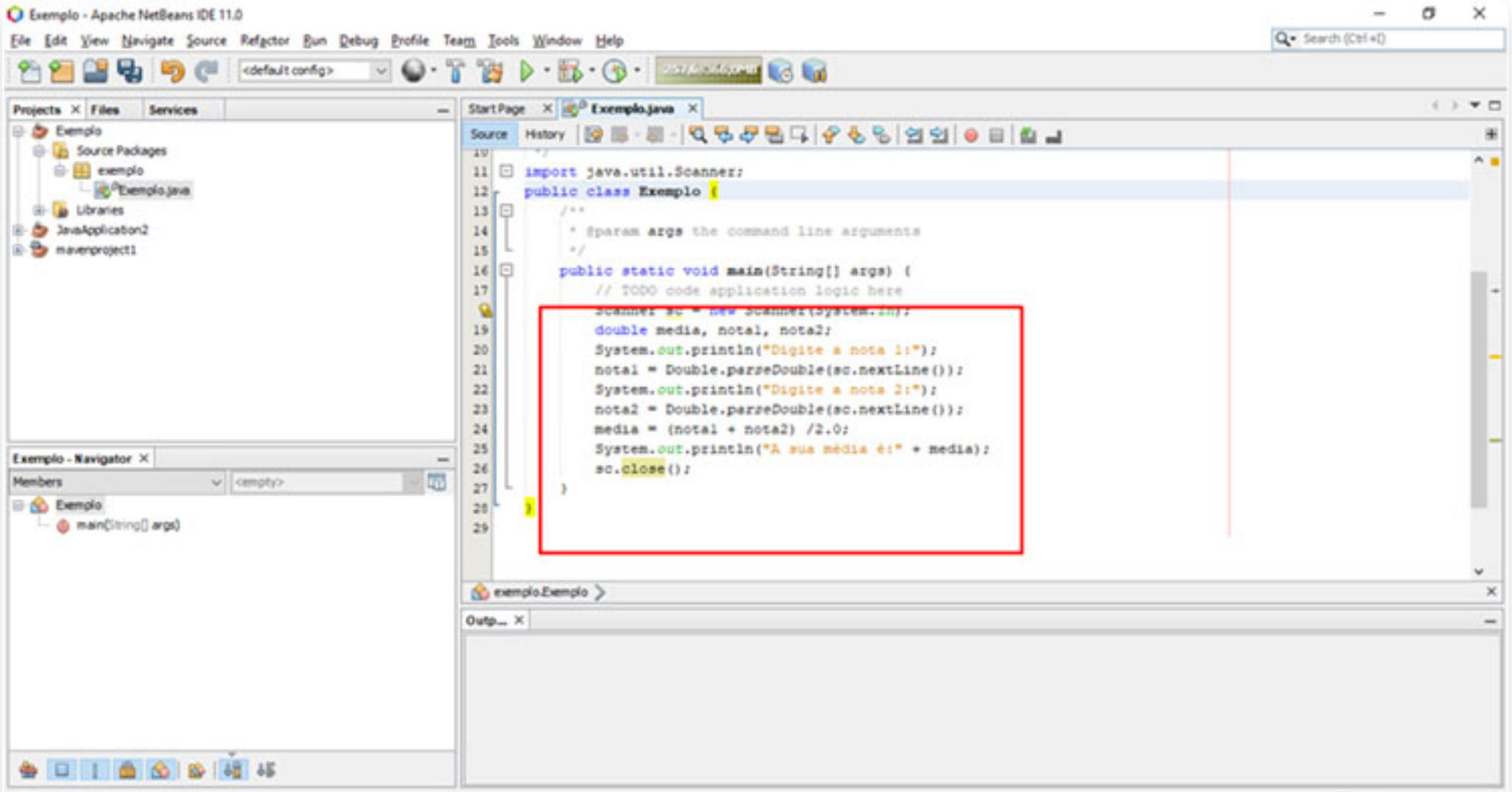



Clique em ***Finish*** para concluir a criação do projeto.

O projeto **Exemplo** foi criado e automaticamente teremos uma **classe inicial** para execução da aplicação.




O ambiente está pronto para digitarmos o código da aplicação: preencha o código conforme o exemplo a seguir.

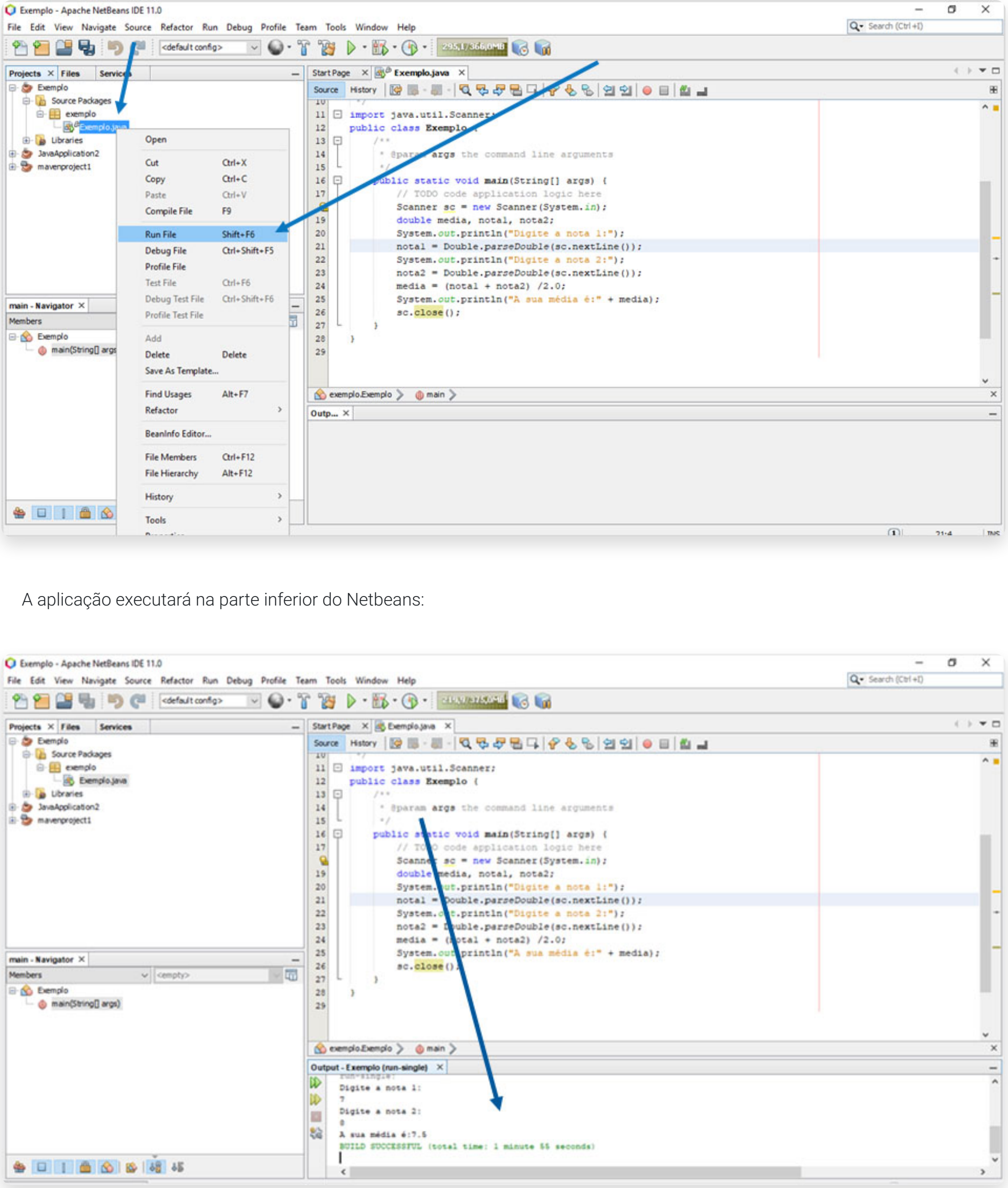



**Código completo da nossa primeira aplicação:**

```
import java.util.Scanner;

public class Exemplo {

                public static void main(String[] args) {
                                // TODO Auto-generated method stub
                                Scanner sc = new Scanner(System.in);
                                double media, nota1, nota2;
                                System.out.println("Digite a nota 1:");
                                nota1 = Double.parseDouble(sc.nextLine());
                                System.out.println("Digite a nota 2:");
                                nota2 = Double.parseDouble(sc.nextLine());
                                media = (nota1 + nota2) /2.0;
                                System.out.println("A sua média é:" + media);
                                sc.close();
                }
}
```

---

Após o código estar pronto e sem erros, podemos executar a aplicação clicando sobre o “arquivo da classe” com o botão direito, e em seguida clicar sobre a opção ***Run file***.




A aplicação executará na parte inferior do Netbeans:

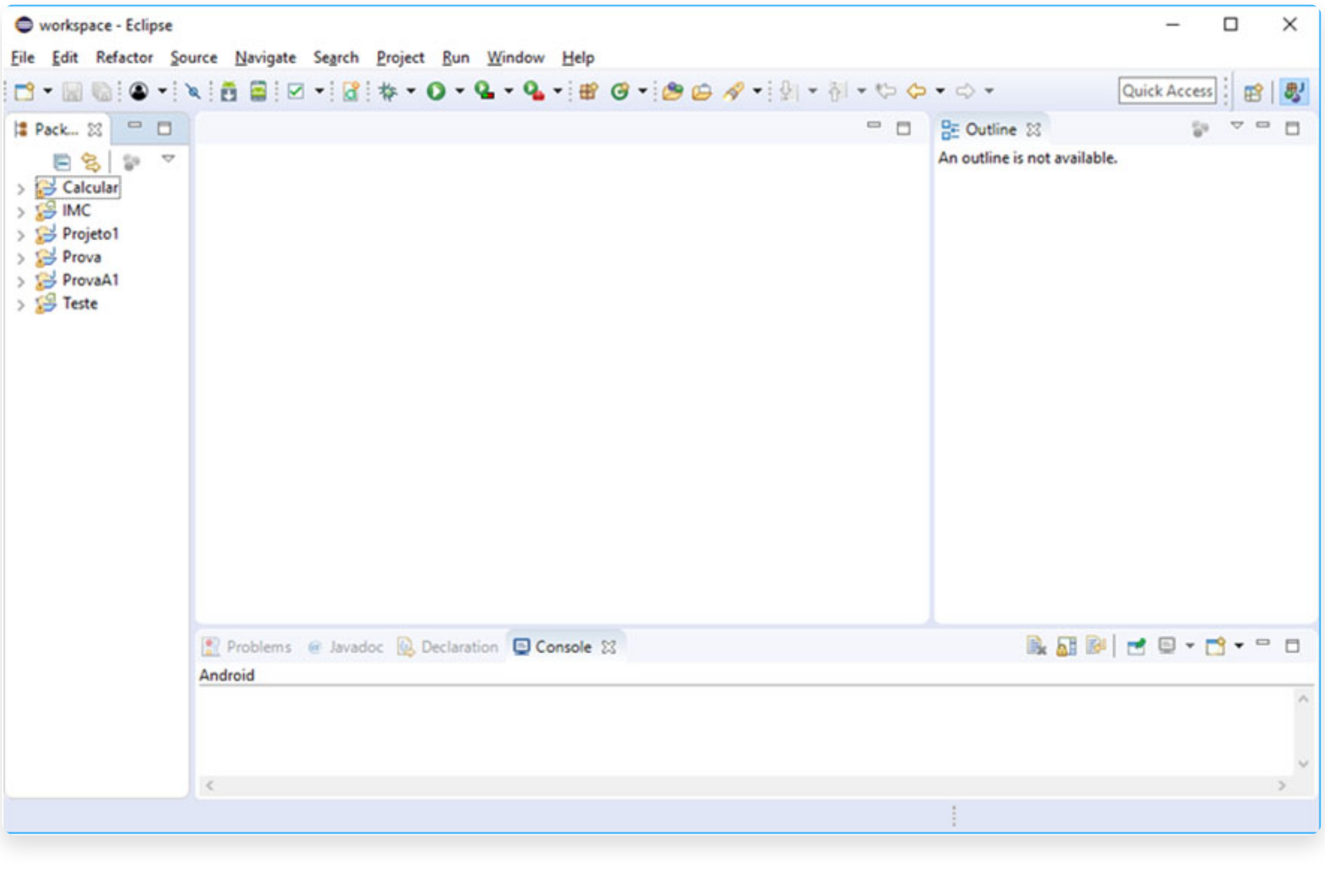



Você poderá fechar outros códigos clicando sobre o X ao lado de cada janela de código para facilitar a criação de novos programas.

No Eclipse, a prática foi realizada na versão Oxygen (atualizada para março de 2018), mas você poderá utilizar a versão 2019-03 da mesma forma:

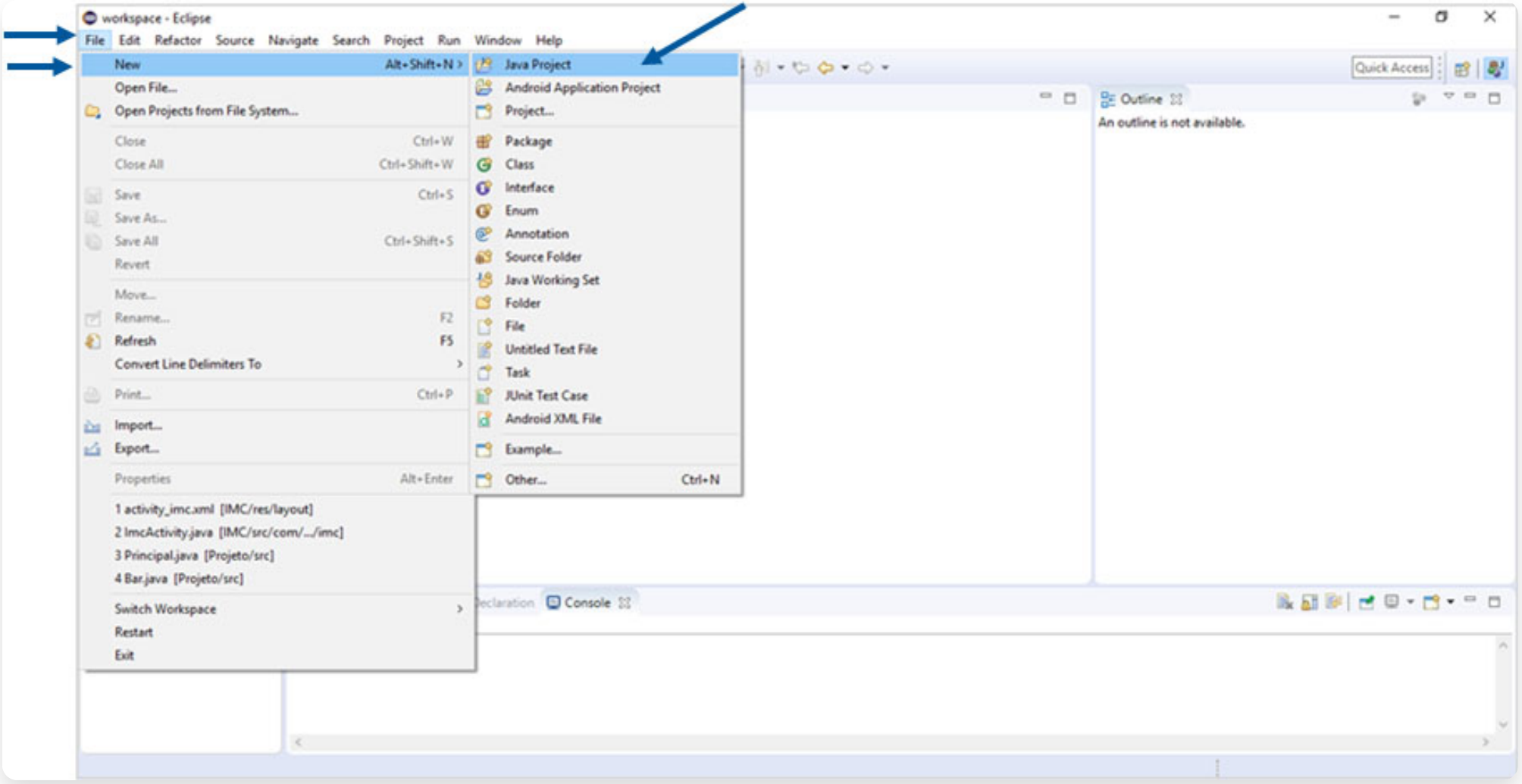


## Tela padrão do Eclipse

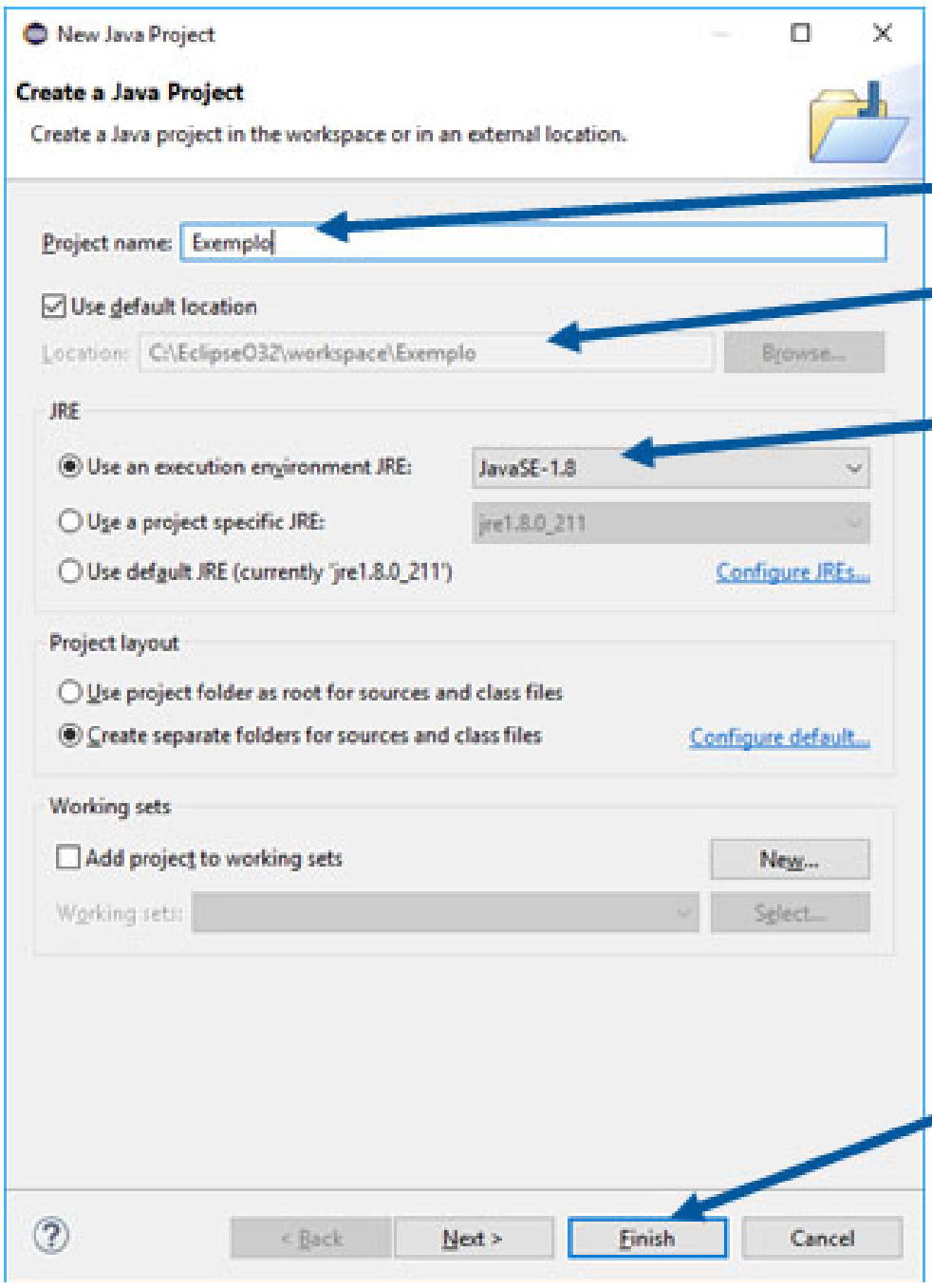



Para criar um **novo projeto:**

No menu, clique em *File > New, e escolha* ***Java Project***.




2) Defina o nome do projeto e o local onde o mesmo será criado.




Definir o nome do projeto.

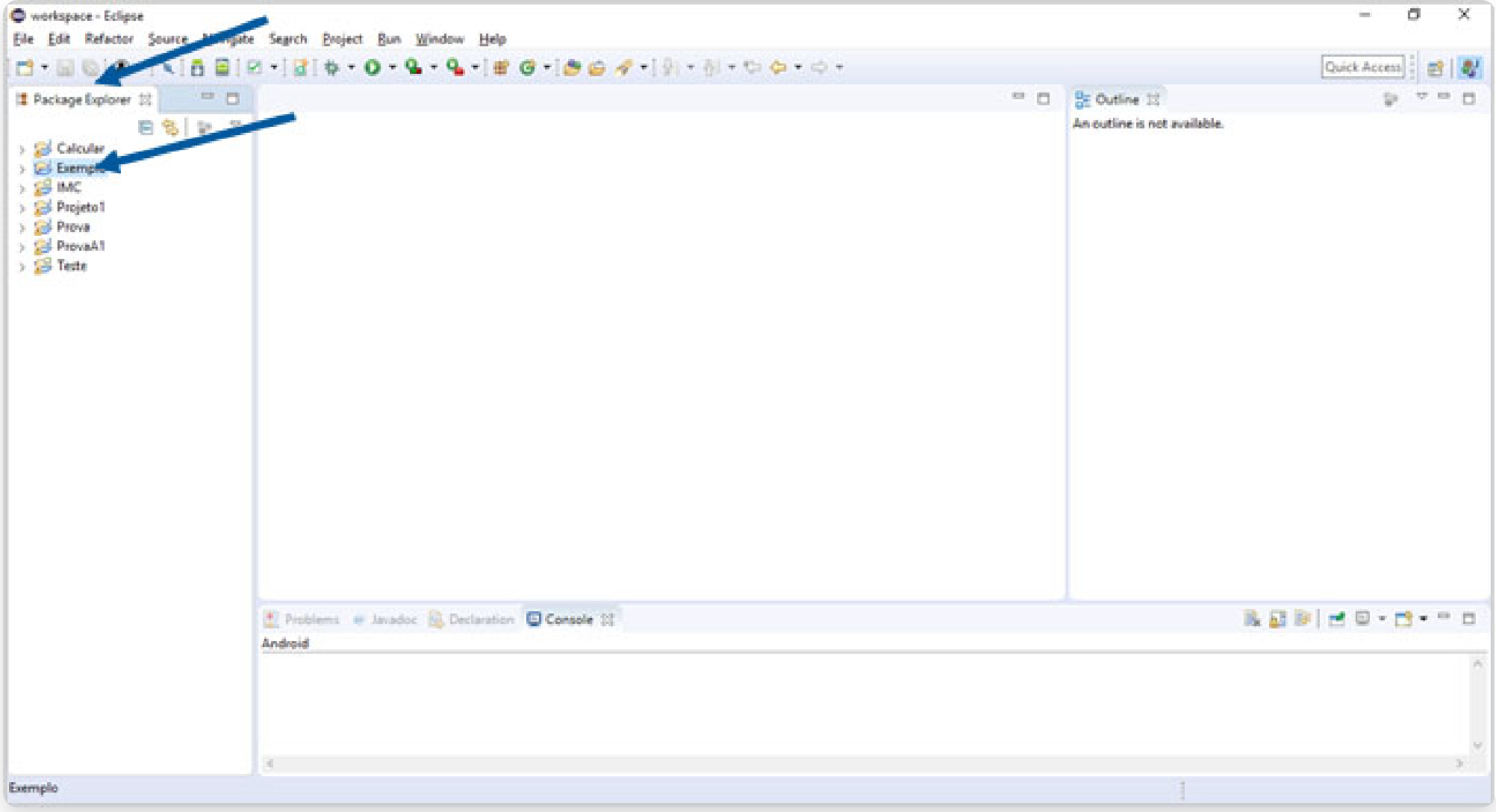
Local onde o projeto será criado.

Nossos projetos serão padronizados na versão 8 do Java.

Para criar o projeto [*Finish*].

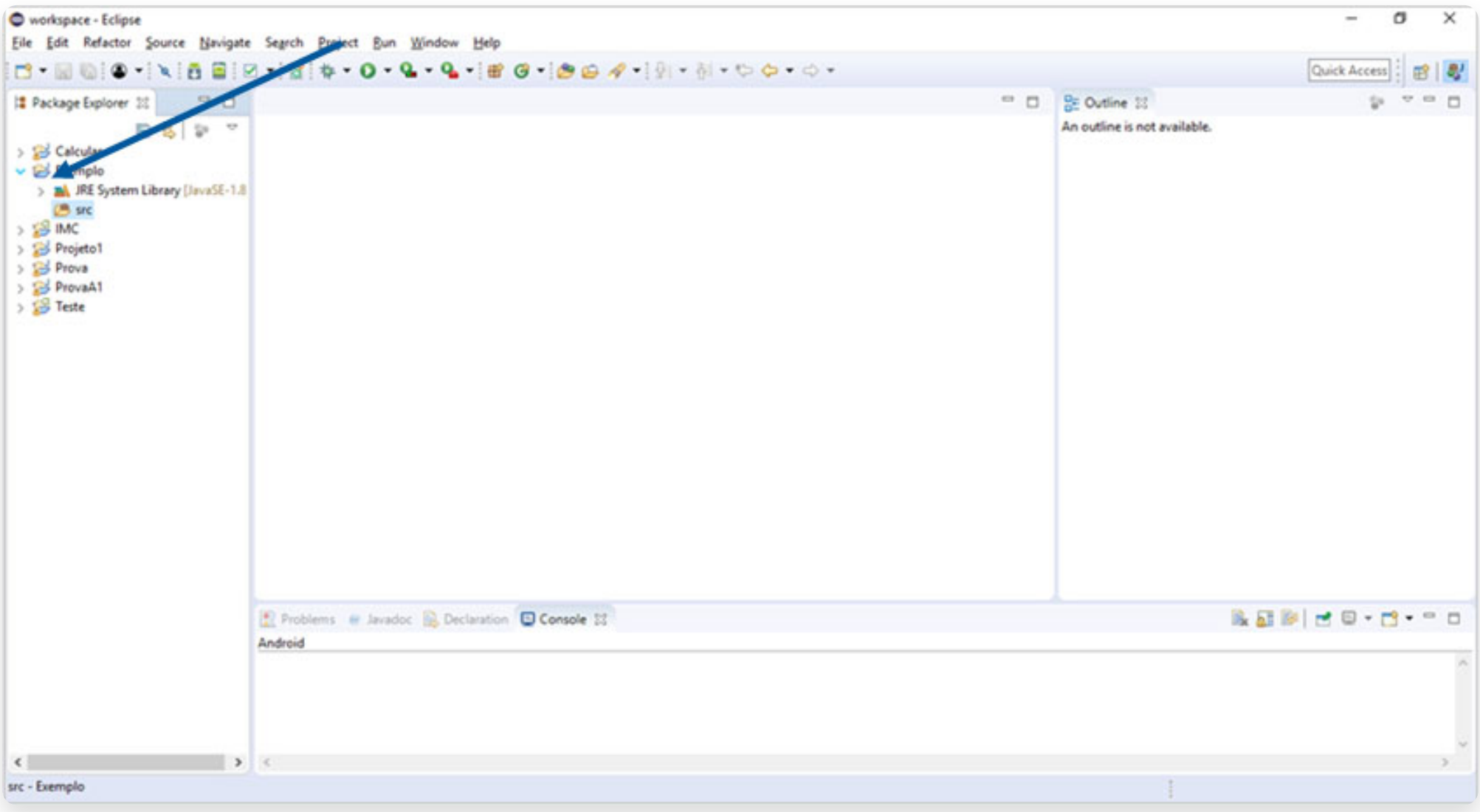
Vamos conferir se o projeto foi criado:

Podemos conferir no *Package Explorer* se o projeto **Exemplo** foi criado.

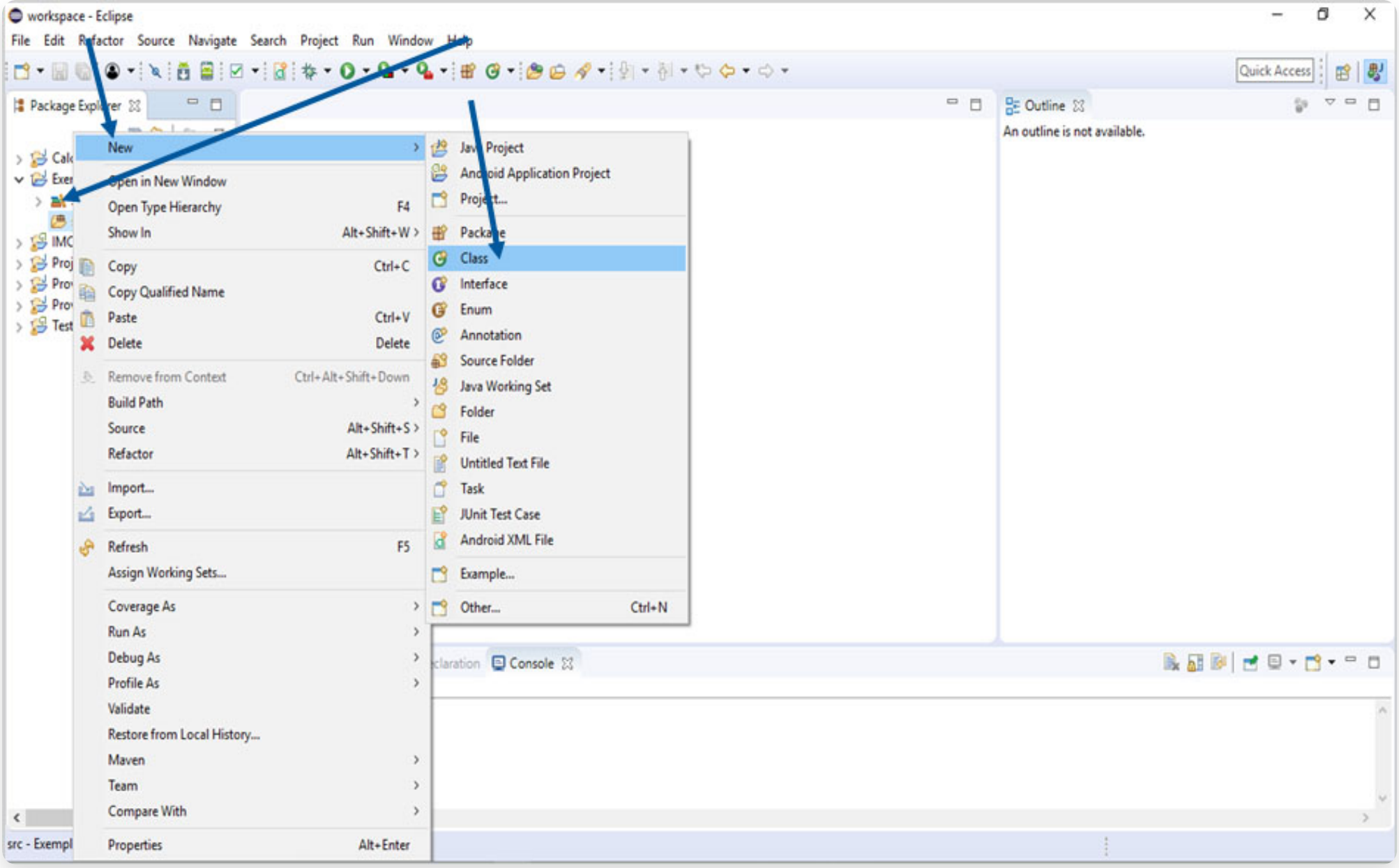


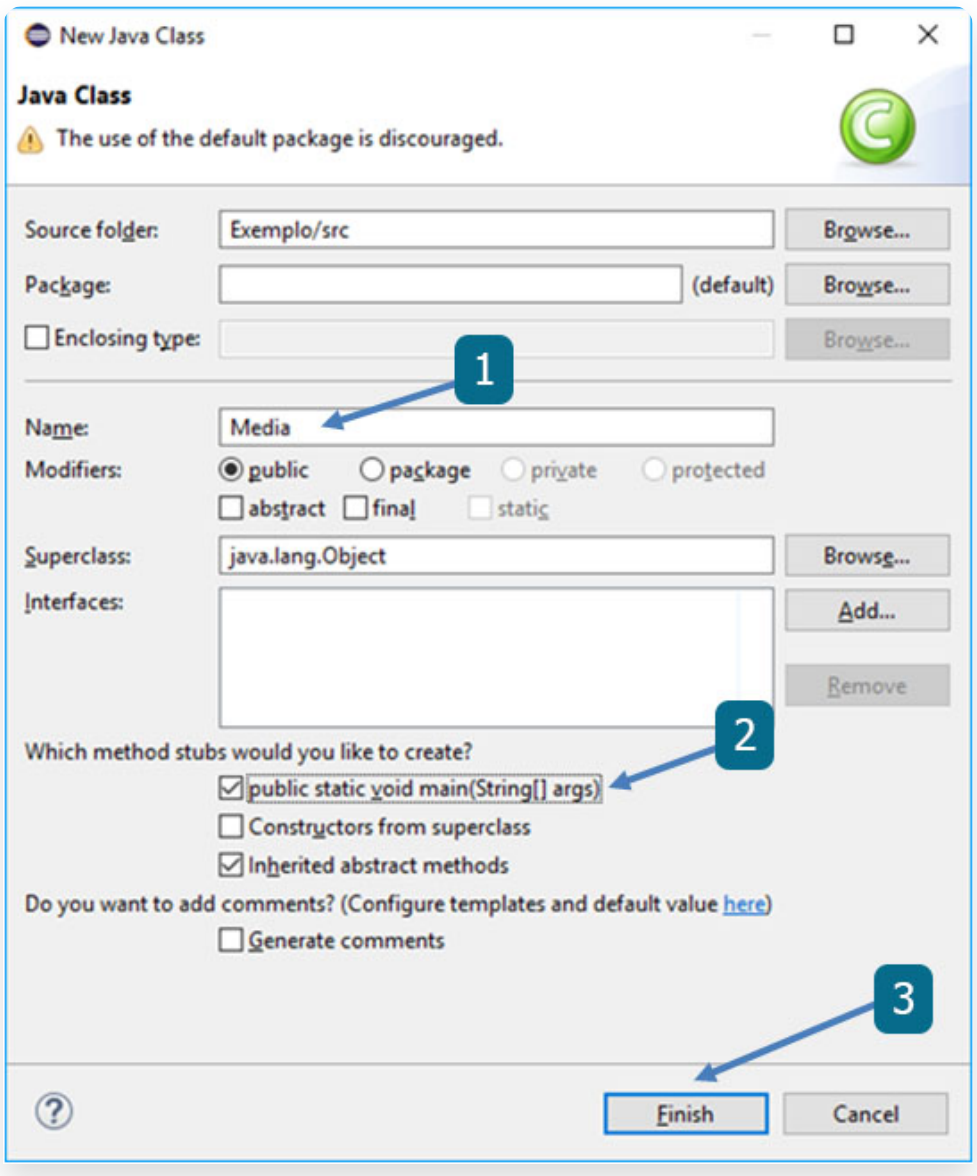

Uma vez que o projeto foi criado, vamos abri-lo e criar a nossa primeira classe Java, uma aplicação simples apenas para conhecermos a ferramenta. Posteriormente discutiremos todos os comandos utilizados.

Clique sobre o sinal de maior "**>**" que aparece ao lado do projeto **Exemplo**.




Pressione o botão direito do *mouse* sobre o **src** (*source*, onde ficam os arquivos do projeto), *New* e escolha *Class* (classe Java).



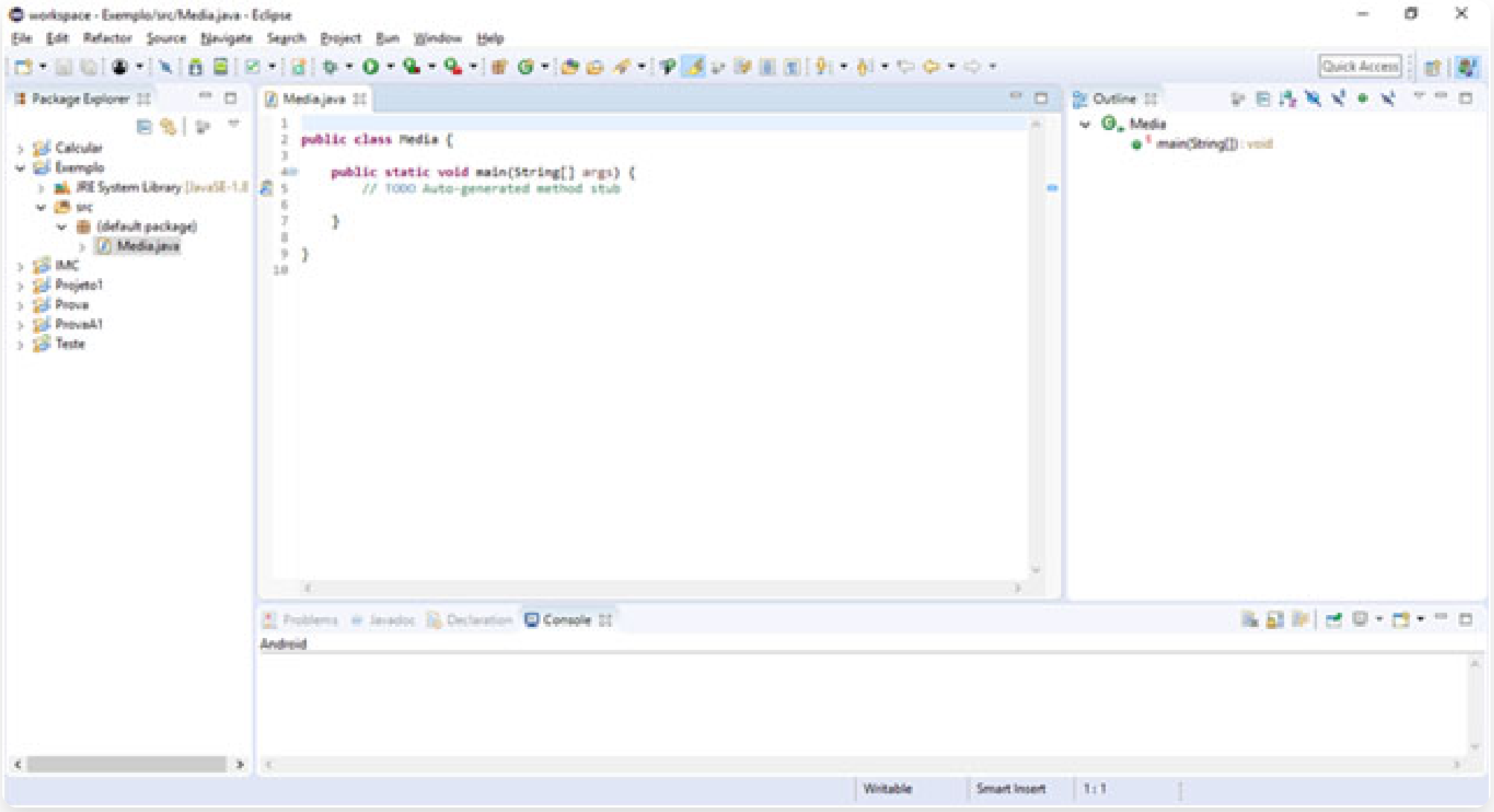




1. O nome da classe deve começar por letra maiúscula, sem espaços ou acentuação. O nome da classe deverá ser o mesmo da classe no código. No caso o exemplo é do cálculo da média de duas notas, por isso o nome será Media, começando por maiúsculas e sem acentuação.

2. Como esta classe será uma aplicação, devemos marcar a opção de aplicação.

3. Depois basta confirmar com [*Finish*].

O ambiente está pronto para digitarmos o código da aplicação.

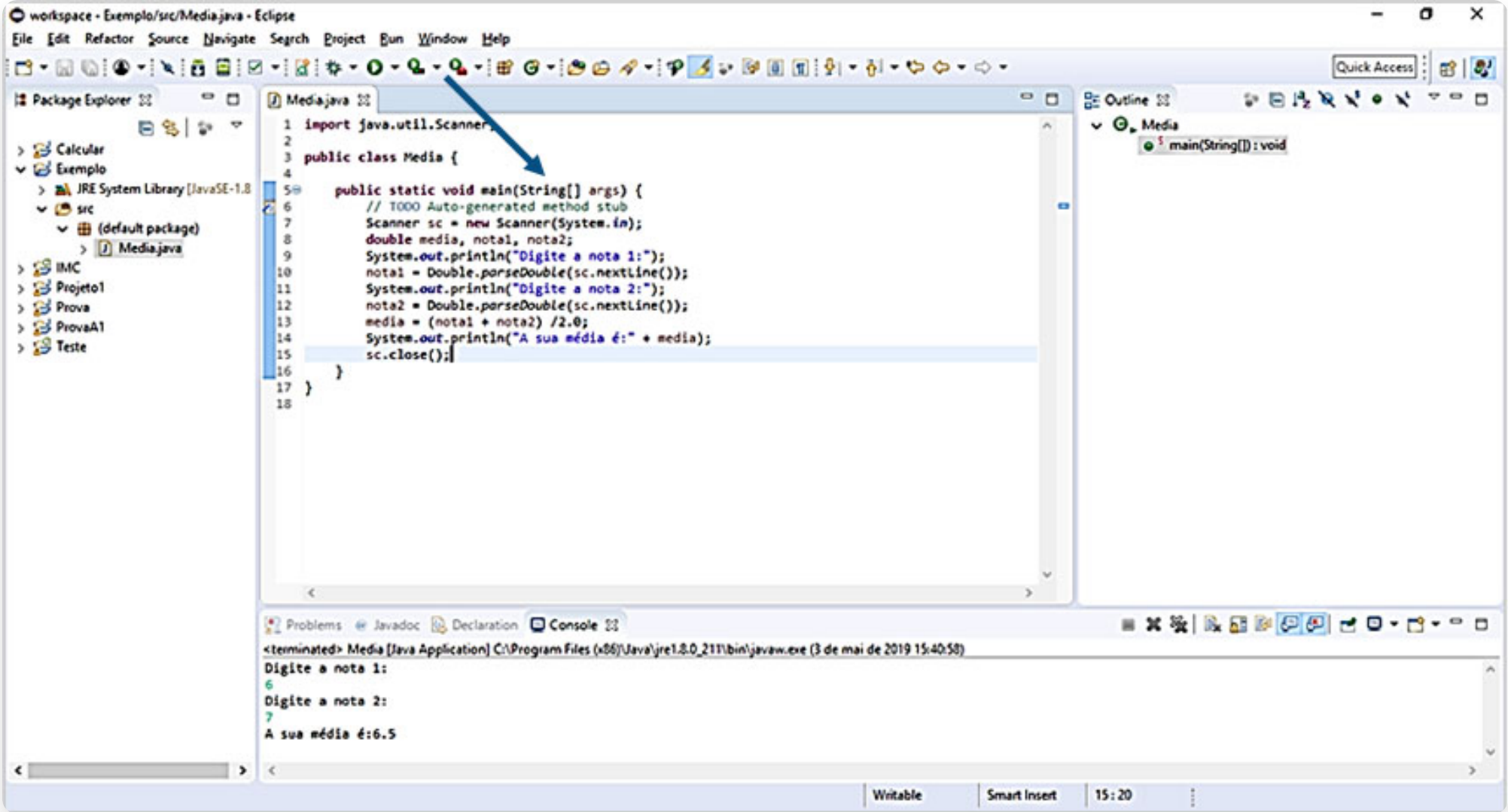


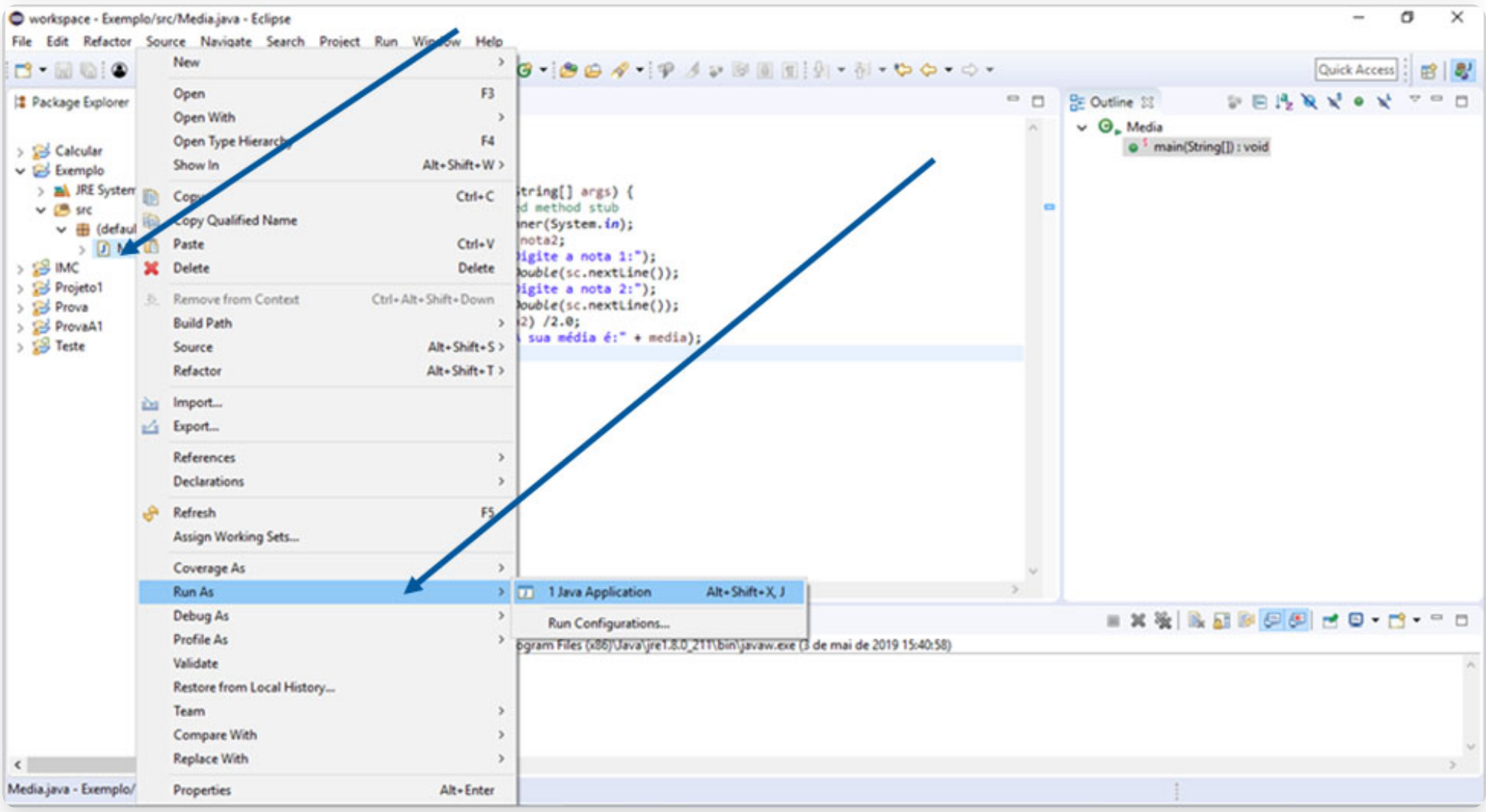

O código completo da nossa primeira aplicação será:

```
import java.util.Scanner;

public class Media {

		public static void main(String[] args) {
				// TODO Auto-generated method stub
				Scanner sc = new Scanner(System.in);
				double media, nota1, nota2;
				System.out.println("Digite a nota 1:");
				nota1 = Double.parseDouble(sc.nextLine());
				System.out.println("Digite a nota 2:");
				nota2 = Double.parseDouble(sc.nextLine());
				media = (nota1 + nota2) /2.0;
				System.out.println("A sua média é:" + media);
				sc.close();
		}
}
```

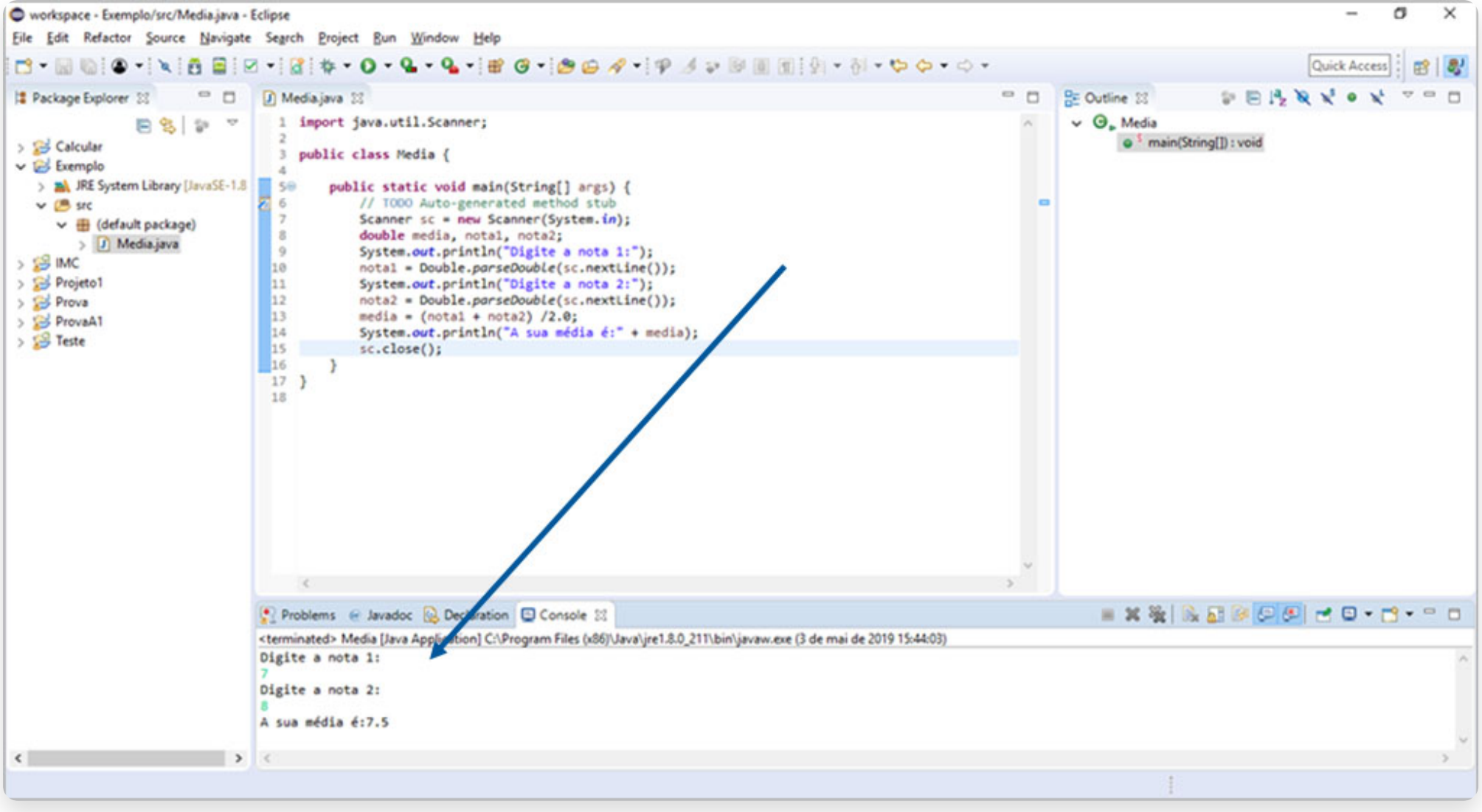
No Eclipse teremos:

Preencha o código todo.




A aplicação será executada na parte inferior do Eclipse:




• Para os testes, é necessário que você responda com os valores das entradas de dados e pressione a tecla [*enter*].

• As aplicações serão testadas com aplicações em formato texto e não gráfico. A criação de interfaces será outro assunto.

• Você poderá fechar outros códigos clicando sobre o X ao lado de cada janela para facilitar a criação de novos programas, evitando que alterações ocorram indevidamente em outros projetos.

Agora você já entendeu as características da tecnologia Java e já teve contato com as duas principais ferramentas de desenvolvimento mais utilizadas pelos desenvolvedores Java. Escolha aquela mais confortável para você e pratique bastante para aumentar seus conhecimentos e se preparar melhor para o mercado de trabalho.

Profissionalmente, é importante que você conheça mais de uma ferramenta, pois cada empresa possui um conjunto próprio e nem sempre usam ao mesmo tempo mais de uma ferramenta para o mesmo propósito. Como não sabemos em qual empresa iremos trabalhar, é importante conhecermos mais de uma ferramenta, ainda que tenhamos nossa preferência.

**Atenção!** Aqui existe uma videoaula, acesso pelo conteúdo online

## Atividade

1) Podemos utilizar a tecnologia Java no desenvolvimento de diferentes tipos de aplicações, e dentre elas podemos destacar o desenvolvimento de aplicações para dispositivos móveis. Em qual sistema operacional para smartphones a linguagem Java é uma linguagem “nativa”?

2) Avalie cada assertiva no que se refere às características da linguagem Java.

I. A linguagem Java é mais rápida do que as linguagens de scripts e as compiladas;

II. A linguagem Java é independente de plataforma;

III. Problemas de desempenhos são resolvidos com a compilação just-in-time.

Com base em sua análise, marque a opção que apresenta apenas as assertivas corretas.

a) I
b) II
c) III
d) I e II
e) II e III

---

3) Assinale a opção com a sequência correta das fases de um programa em linguagem Java.

a) Editor – Programa.java – Interpretador – Programa.class - Compilador
b) Editor – Programa.class – Compilador – Interpretador
c) Editor – Programa.class – Interpretador - Compilador
d) Editor – Programa.java – Compilador – Programa.class – Interpretador
e) Editor – Programa.java – Interpretador

---

## Referências

---

## Próxima aula

---

- Tipos de dados 1;
- Constantes e variáveis;
- Operadores e expressões;
- Comando de controle de fluxo;
- Estrutura de repetição.

## Explore mais

---

Visite o site oficial do ***Java*** (português).

Conheça os sites para baixar a ferramenta IDE:

- *Netbeans*;
- *Eclipse*;