

Técnico Subsequente em Desenvolvimento de Sistemas

Lógica de Programação II

Introdução a Java

Alex Helder Cordeiro do Rosário de Oliveira

Instituto Federal de Brasília - *Campus* Brasília

1º semestre de 2023

Objetivo da Aula

- Apresentar a Plataforma Java;
- Apresentar as principais características do Java;
- Mostrar como se faz para escrever, compilar e executar um programa em Java.

Sumário

- 1 Visão Geral da Plataforma Java
 - Características
 - Plataformas
 - Ambientes
- 2 Desenvolvimento Básico em Java
 - Escrita
 - Compilação
 - Execução
- 3 Apresentando e Obtendo Informações do Usuário
 - Apresentando dados na tela
 - Obtendo dados do teclado
 - Comentários
- 4 Dicas para o uso do NetBeans

Visão Geral da Plataforma Java

Histórico

- Projeto Green da SUN:
 - 4 integrantes;
 - Software para eletrodomésticos;
 - Baixo uso de memória;
 - Baixo custo.
- Em 1994, a linguagem é adaptada para construir aplicativos que rodam na Web;
- Em 1995: A SUN lança o Java 1.0. Possibilitando a execução de applets em páginas de Internet;
- Netscape Navigator inclui o Java em seu navegador;
- Java torna-se popular para aplicações Web;
- Em 1999: O Java é uma das linguagens mais populares entre os programadores.

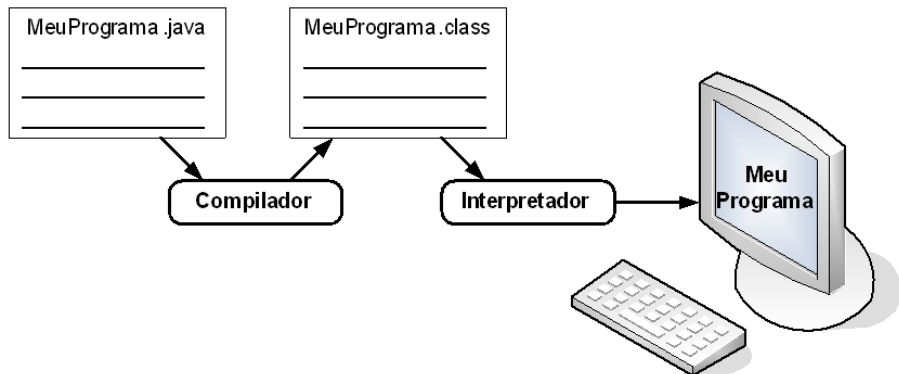
Características

- Totalmente aberta;
- Independente da plataforma ou sistema operacional - Portável;
- Orientada a Objetos;
- Linguagem interpretada;
- Multiprocessada;
- Robusta e Segura;
- Alto desempenho (para uma linguagem interpretada).

Portabilidade

Linguagem tanto compilada quando interpretada.

- 1 Código fonte.
- 2 Compilado para Java bytecode.
- 3 Interpretado pela plataforma da Java Virtual Machine (JVM).

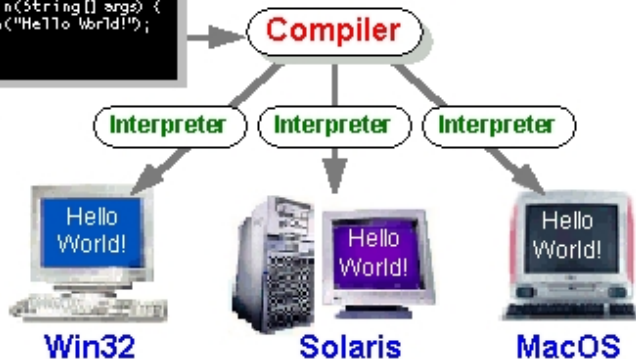


Portabilidade

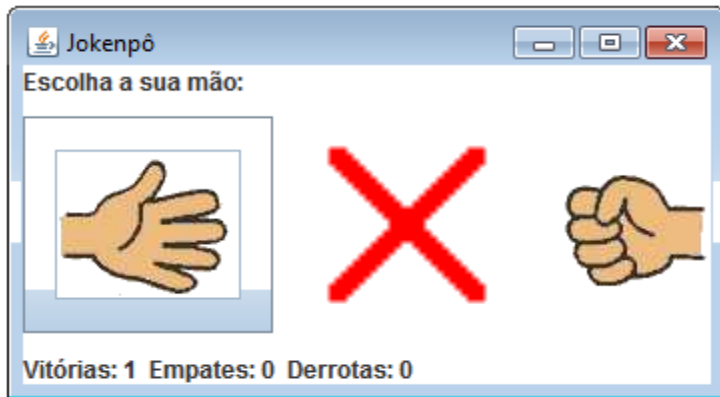
Java Program

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

HelloWorldApp.java



Aparência do Java em Windows



Aparência do Java em Linux



Plataformas Java

- JSE

Java Standard Edition: Plataforma desenvolvida para computadores pessoais.

- JEE

Java Enterprise Edition: Plataforma desenvolvida para aplicações empresariais e multi-usuários.

- JME

Java Micro Edition: Plataforma desenvolvida para dispositivos com poucos recursos de memória e/ou energia.

Ambientes Java

Ambiente de Desenvolvimento:

- Java Development Kit (JDK)
- Coleção de ferramentas para compilar, executar e depurar aplicações Java.

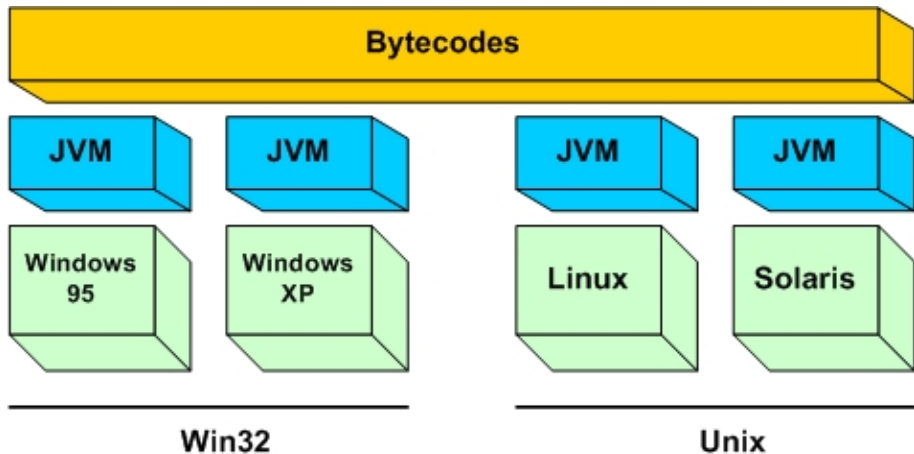
Ambiente de Execução:

- Java Runtime Environment (JRE)
- Ferramenta que traz a JVM (Java Virtual Machine), necessária para a execução de aplicações no dispositivo.

Java Virtual Machine - JVM

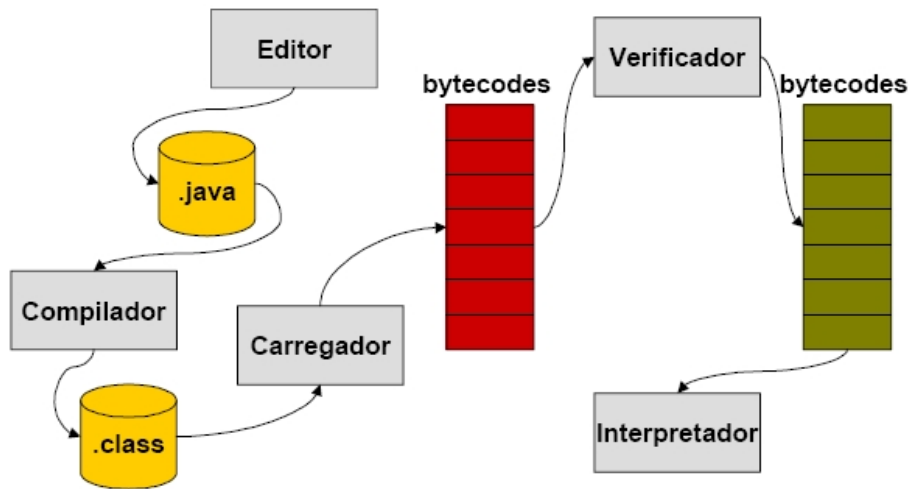
- Máquina imaginária implementada como uma aplicação de software em uma máquina real.
- Interpreta o bytecode gerado na compilação de um programa Java.

Java Virtual Machine - JVM



Desenvolvimento Básico em Java

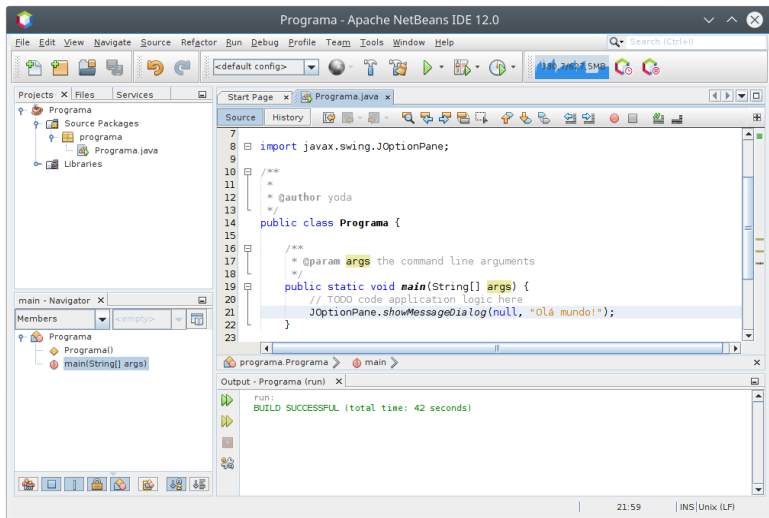
Processo de Desenvolvimento em Java



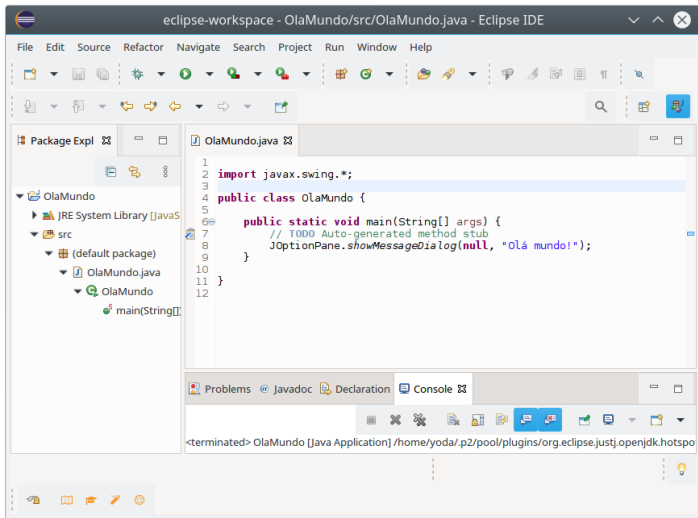
Processo de Escrita de um Aplicativo em Java

- Pode ser feito em qualquer aplicativo que edite arquivos de texto simples:
 - NotePad (Windows);
 - WordPad (Windows);
 - VI (linux);
 - Kate (linux com KDE); ...
- Pode ser utilizado alguma IDE de java:
 - NetBeans;
 - Eclipse; ...

Interface do Netbeans



Interface do Eclipse



Processo de Escrita de um Aplicativo em Java

- Criação de um arquivo com extensão `.java`.
- O nome do arquivo deve ser o mesmo nome da classe presente no arquivo.
- O Java faz diferenciação entre maiúsculo e minúsculo.
- A classe que deve ser aplicativo precisa do método:

```
public static void main (String[] args)
```

Processo de Escrita de um Aplicativo em Java

- Arquivo FazNada.java.

```
public class FazNada {  
    public static void main (String[] args) {  
    }  
}
```

Compilação

- É necessário a presença do ambiente de desenvolvimento Java (JDK).
- Utilizando o comando `javac`.

```
javac FazNada.java
```

Execução

- Utilizando o comando `java`.

```
java FazNada
```

- Não se utiliza a extensão `.class` no comando.

Demonstrações

- Demonstração com o Notepad e linha de comando.
- Demonstração com o Netbeans.

*Exemplo: FazNada.java

Apresentando e Obtendo Informações do Usuário

Apresentando dados na tela

- Via console:

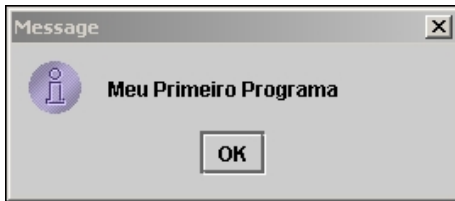
```
System.out.println("Informação");
```

*Exemplo: FazQualquerCoisa.java

Apresentando dados na tela

- Via ferramenta gráfica JOptionPane:

```
JOptionPane.showMessageDialog(null, "Meu primeiro programa");
```



- Necessário importar a classe `javax.swing.JOptionPane`*

Pode ser importado o pacote que contém a classe: `javax.swing.`

†Exemplo: `FazQualquerOutraCoisa.java`

Imports

- Necessário para encontrar classes que não fazem parte da linguagem.
- Devem ser colocados antes da definição de classe.

Comandos

- Todo comando deve ser encerrado com ';' ;
- A mesma ordem que usamos para escrever os comandos é usada para a execução;
- Enquanto um comando não for encerrado, o comando seguinte não irá entrar em execução.

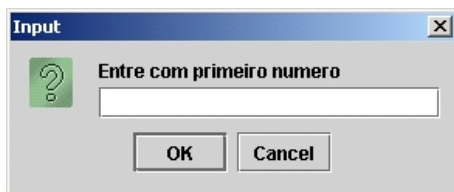
Sua vez:

- Faça um programa que apresente alguma informação (pode ser seu nome) na tela.

Obtendo dados do teclado

- Via ferramenta gráfica JOptionPane:

```
String variavel =  
    JOptionPane.showInputDialog("Entre com o primeiro numero");
```



- Também é necessário importar a classe `javax.swing.JOptionPane`.

*Exemplo: LeQualquerCoisa.java

Sua vez:

- Escreva um programa que receba um argumento e apresente-o na tela do computador.

Obtendo números inteiros

- O `JOptionPane` só obtêm valores textuais.
- Podemos extrair valores inteiros de um texto* através do comando:

```
int numero = Integer.parseInt(variavel);
```

*Se ele for a representação de um número.

†Exemplo: `LeNumero.java`

Sua vez:

- Escreva um programa que receba dois números e apresente na tela a soma destes números.

Comentários

- Trechos do código fonte que são ignorados pelo compilador.
- Usados para explicar parte do código.
- Fazem parte da documentação de código.
- Auxilia na programação em equipe.
- Permite que outro programador compreenda o seu código.

Comentários

Comentários de linha:

- Formados a partir de duas barras inclinadas consecutivas ('//').
- Tudo o que estiver a direita das barras na linha será desconsiderado.
- Não se propaga para outras linhas.
- Usado para explicar a finalidade de um comando específico.

```
System.out.println("Informação");//Escreve Informação na tela
```

*Exemplo: Comentarios.java

Comentários

Comentários de blocos:

- Começa com `/*` e termina com `*/`.
- Tudo o que estiver entre estes dois marcadores será desconsiderado.
- Se propaga por diversas linhas.
- Usado para explicar a função de um determinado bloco de código, método ou classe.

```
/* O método main abaixo serve para  
   escrever alguma coisa na tela. */  
public static void main (String[] args) {  
    System.out.println("Informação");  
}
```

*Exemplo: Comentarios.java

Documentação de Código

- Se faz através de comentários*;
 - Útil para que o programador se lembre qual o funcionamento do bloco de código implementado;
 - Muito útil para que outros programadores compreendam o que estava sendo feito no programa;
 - Extremamente importante para programação em equipes.
-
- Em Java, podemos gerar o JavaDoc de nossas classes: A documentação em formato HTML, descrevendo as classes, seus atributos e métodos, permitindo assim que outros programadores entendam o que nossos códigos fazem sem precisar ver o código-fonte.

*A Convenção de Código determina o comentário de bloco, iniciando com `'/**'`.

Indentação

- É a utilização de diferentes recuos à esquerda, de acordo com os laços onde as linhas se encontram.
- É importante para facilitar na identificação de início ou fim de laços ou funções, correção de *bugs*, e compreensão do código por parte de outros programadores.
- A regra básica de indentação é a seguinte:
 - a cada vez que se abre chaves, as linhas seguintes passam a ser alinhadas 4 espaços mais à direita;
 - a cada vez que se fecha chaves, as linhas seguintes voltam 4 espaços à esquerda.

*Regras definidas na Convenção de Código.

Dicas para o uso do NetBeans

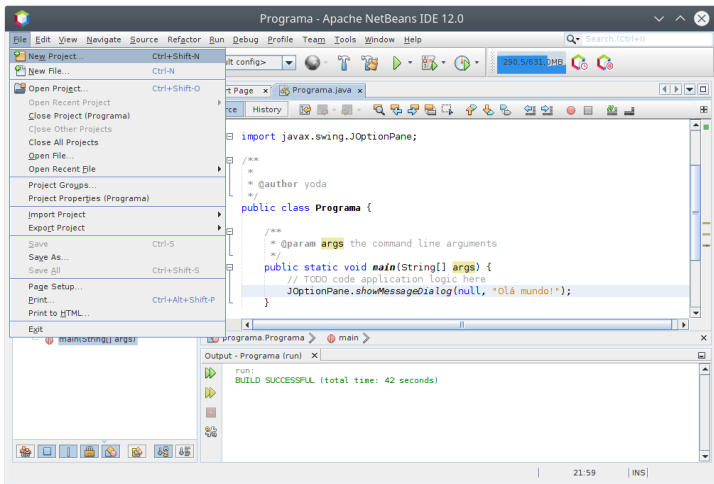
Instalação

- É necessário antes instalar o JDK, que pode ser obtido no site:
<https://www.oracle.com/technetwork/pt/java/javase/downloads/index.html>
- O NetBeans pode ser baixado no próprio site:
<https://netbeans.apache.org/download/index.html>;
- Outra opção é usar a versão *bundle*, que instala o JSDK e o Netbeans juntos, também encontrada no site:
<https://www.oracle.com/technetwork/pt/java/javase/downloads/index.html>

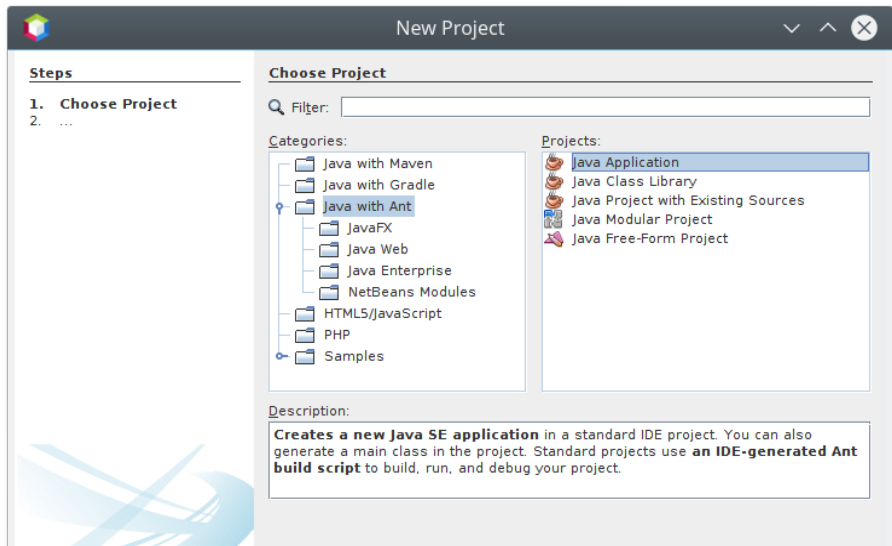
Criando um projeto no NetBeans

- Em IDEs avançadas como o NetBeans, é comum ser necessário a criação do projeto para podermos compilar um código fonte.
- Para criar o projeto em Java no NetBeans:
 - ❶ Clique no menu Arquivo → Novo Projeto;
 - ❷ Na categoria, escolha Java with Ant; em Projetos, escolha Java Application; então clique em Next;
 - ❸ Dê um nome ao projeto; determine a localização do projeto; dê um nome à classe principal (que terá o método `main()`). Clique em Finalizar.

Criando um projeto no NetBeans



Criando um projeto no NetBeans



Criando um projeto no NetBeans

Steps

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

Name and Location

Project Name:

Project Location:

Project Folder:

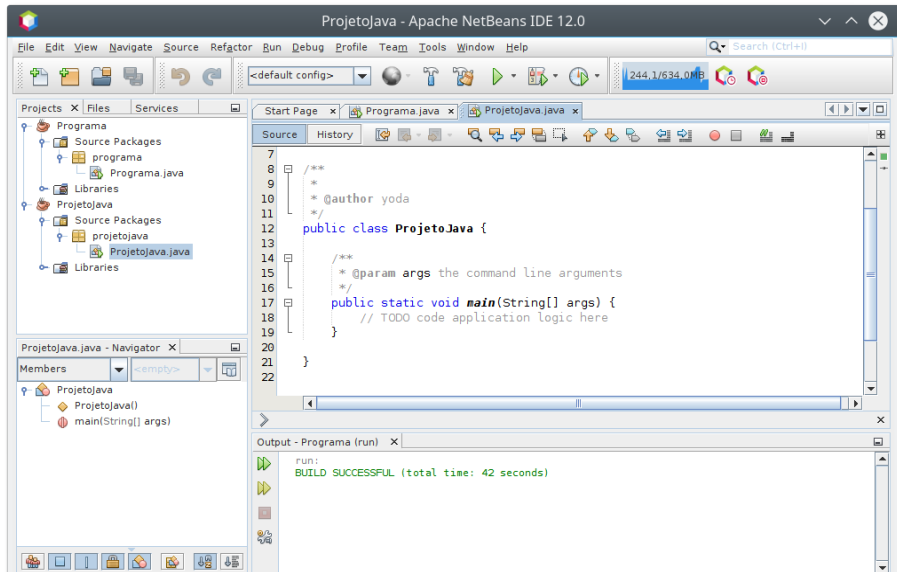
☐ Use Dedicated Folder for Storing Libraries

Libraries Folder:

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

☒ Create Main Class

Criando um projeto no NetBeans



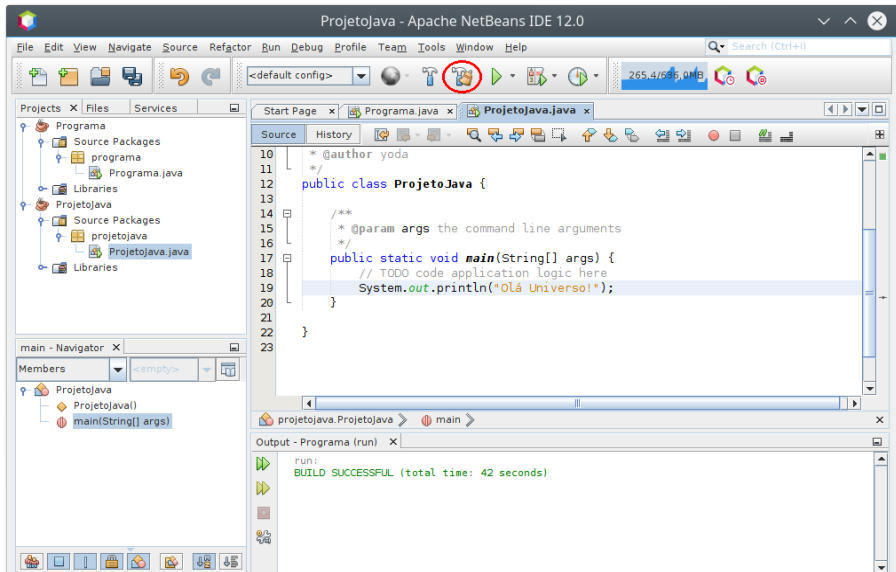
Compilando e executando

- O NetBeans preenche o arquivo principal com um “esqueleto” de código*.
- O arquivo criado pode ser totalmente sobrescrito[†] pelo código que deseje colocar no arquivo.
- Para compilar o arquivo, clique no botão cujo ícone é um martelo com uma vassoura.
- Para executar, clique no botão cujo ícone é uma seta verde para a direita.
- A saída do programa aparecerá no Netbeans, na caixa de Saída, abaixo do código fonte.

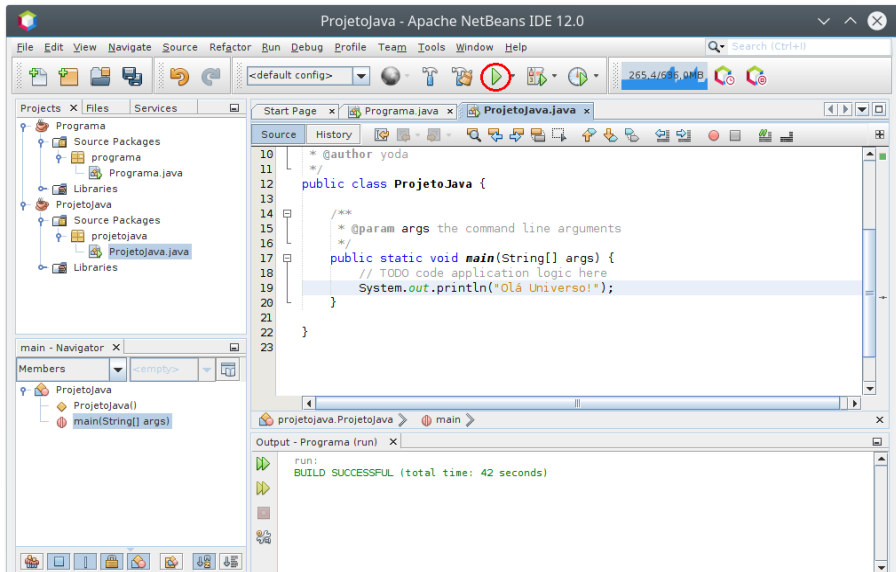
*composto pelas declarações de pacote, de classe e o método `main()` sem nenhum comando e algumas linhas de comentários.

[†]Você pode apagar e escrever um novo do “zero”.

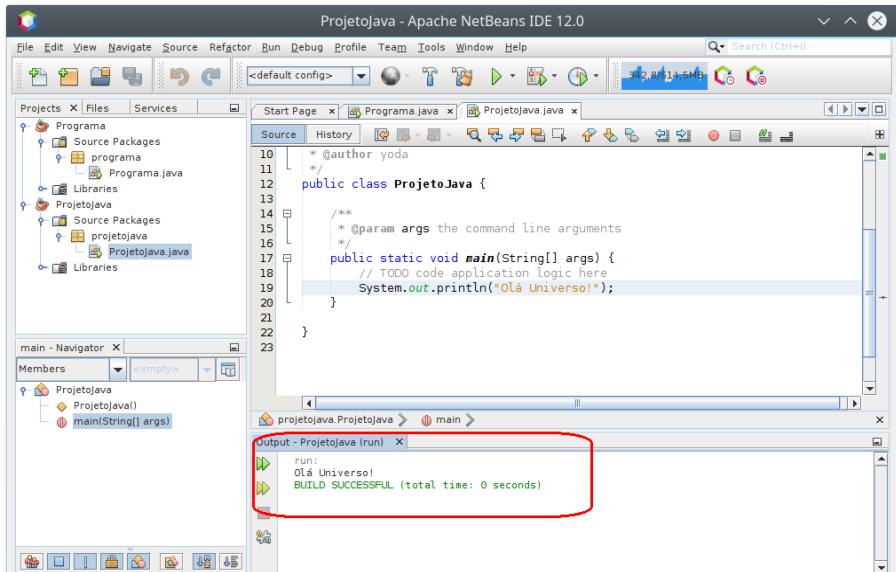
Compilando o projeto no NetBeans



Executando o projeto no NetBeans

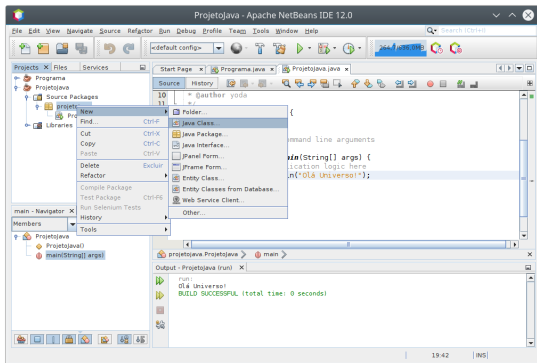


Executando o projeto no NetBeans



Acrescentando arquivos ao projeto

- 1 Clique com o botão direito do mouse sobre o pacote onde se deseja acrescentar o novo arquivo;
- 2 Clique em New;
- 3 Clique em Java Class ;
- 4 Dê um nome à classe e clique em Finish.



Definindo a classe principal

- Caso hajam mais de uma classe com método `main()`, deve-se determinar qual delas deve ser executada pelo clique do botão:
- 1 Clique com botão direito no nome do projeto;
 - 2 Clique em `Properties`;
 - 3 Clique em `Run`;
 - 4 Escreva (ou use a ferramenta de procura – `Browse...`) o nome da classe desejada;
 - 5 Clique em `OK`.

Renomeando ou movendo classes

- Em Java, o nome dos arquivos deve ter o mesmo nome da classe;
 - O pacote onde está a classe é declarado no código;
 - Para realizar uma alteração de nome (renomear o arquivo) ou de pacote (mover o arquivo), é recomendado o processo chamado *Refatorar*;
- 1 Clique com botão direito no nome da classe;
 - 2 Clique em Refactor;
 - 3 Escolher a opção desejada (Rename . . . , Move . . . , . . .);
 - 4 Indicar os parâmetros desejados.