

INSTITUTO FEDERAL
Goiás

Instituto Federal de Goiás
Câmpus Goiânia

Bacharelado em Sistemas de Informação
Disciplina: Programação Orientada a Objetos I

A Linguagem Java como Ferramenta de Experimentação

Prof. Ms. Renan Rodrigues de Oliveira
Goiânia - GO

Breve Histórico da Linguagem Java

A história de Java começou a ser escrita quando James Gosling e sua equipe iniciaram uma pesquisa acreditando que a próxima inovação seria a associação dos microprocessadores aos dispositivos eletrônicos.



James Gosling,
considerado o pai da
linguagem Java



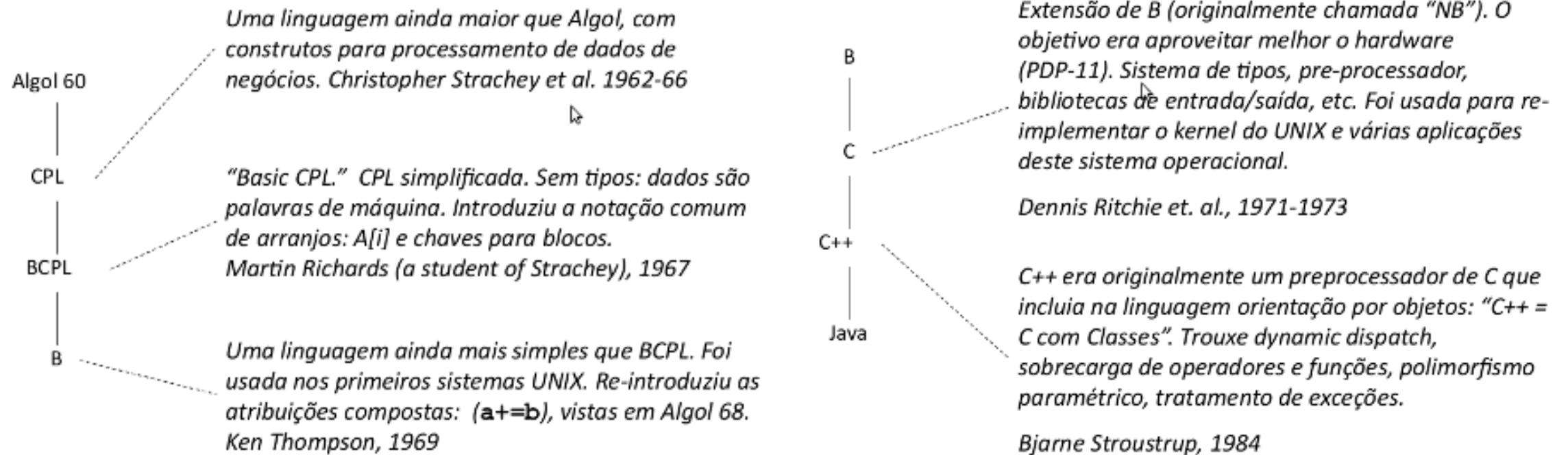
Reconhecendo essas ideias, a Sun Microsystems, lançou e financiou, em 1991, a pesquisa Green.

- ▶ Esta pesquisa resultou no desenvolvimento de uma linguagem baseada em C e C++, que seu criador, James Gosling, chamou de Oak.
- ▶ Entretanto, mais tarde, descobriu-se que já havia uma linguagem de programação com esse nome, foi então que sugeriram o nome Java.



Breve Histórico da Linguagem Java

A linguagem Java herdou várias funções de outras linguagens como C e C++.



Breve Histórico da Linguagem Java

Ainda em 1991, emergiu uma demonstração funcional da ideia inicial do projeto Green. Construíram um PDA (Personal Digital Assistance) batizado de Star7 (ou *7).



O *7 funcionava como uma espécie de controle remoto para vários dispositivos. O *7 era um dispositivo com:

- ▶ Monitor touch screen LCD colorido de 5 polegadas;
- Rede sem fio de 900 MHz; 4 GB de memória; Áudio multimídia; Entradas PCMCIA.



Existia um agente virtual chamado Duke.

- ▶ Logo depois, o pequeno agente se tornaria o conhecido mascote da tecnologia Java.



Breve Histórico da Linguagem Java



No entanto, o projeto Green passava por algumas dificuldades.

- ▶ O projeto era muito avançado para época.
- ▶ O Star7 não sobreviveu (apenas seis aparelhos foram construídos).

O mercado de dispositivos eletrônicos não estava se desenvolvendo como o esperado, e essa maravilha tecnológica não foi comercializada para a indústria de eletrônicos.

Breve Histórico da Linguagem Java

Todavia, em 1993, a World Wide Web ganhou grande popularidade e os idealizadores do projeto viram a oportunidade de utilizar Java para a criação de páginas da Web com conteúdo interativo e dinâmico.



O projeto ganhou vida nova:

- ▶ A web prometia oferecer o nível de interatividade que se esperava anteriormente para a TV Interativa.
- ▶ Em maio de 1995, a Sun anunciou o Java formalmente em uma conferência, o que despertou interesse na comunidade empresarial.



Desde então Java tem sido utilizada para:

- ▶ Criar páginas na Web com o conteúdo interativo e dinâmico, desenvolvimento de aplicativos corporativos, sistemas de TV, criação de aplicativos para dispositivos móveis, entre outros.

Breve Histórico da Linguagem Java

Em 2009 a Oracle Corporation adquire a empresa responsável pela linguagem Java, a Sun Microsystems, por US\$ 7,4 bilhões, com o objetivo de levar o Java e outros produtos da Sun a seu portfólio.



A Linguagem de Programação Java



Algo que você precisa saber:

- Lógica de Programação.



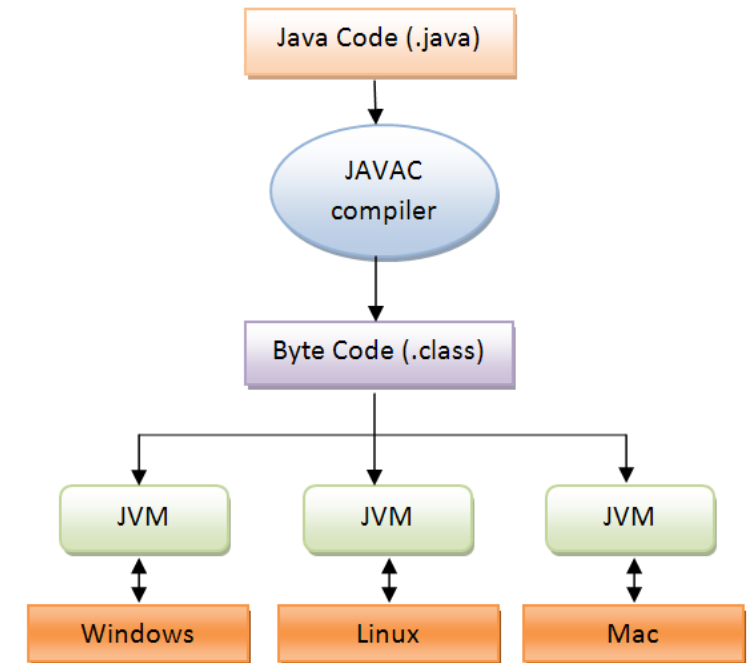
A Linguagem de Programação Java

A linguagem de programação Java é orientada a objetos, compilada em bytecodes e executada através de uma Máquina Virtual Java.



Interpretada, Neutra, Portável, Simples e Orientada a Objetos.

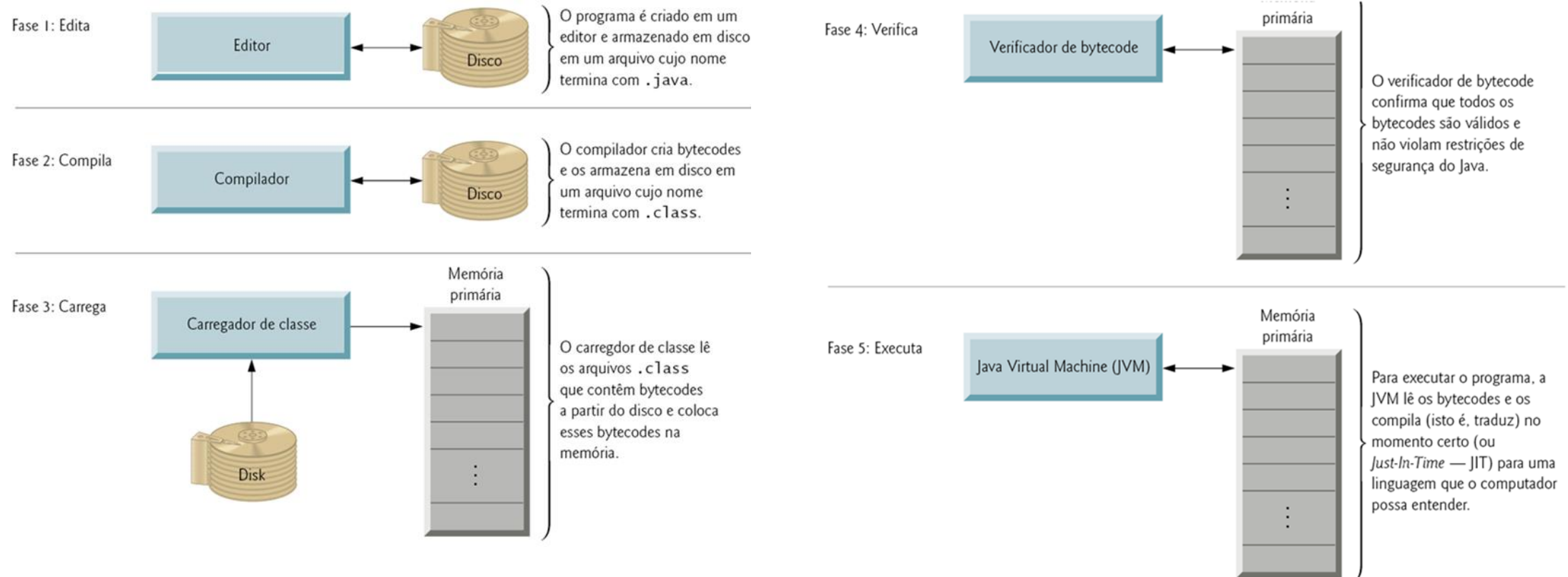
- ▶ Permite que o código em Java possa ser escrito independente da plataforma;
- ▶ Bytecodes executam em qualquer dispositivo que possua uma Java Virtual Machine;
- ▶ Neutra em relação à arquitetura, uma característica que permite uma grande portabilidade.
- ▶ É de fácil aprendizado e orientada a objetos.



Ciclo de Execução em Java



Programas Java normalmente passam por cinco fases:



Fase 1: Edição



Consiste em editar um arquivo com um programa editor de texto plano (em geral, conhecido simplesmente como editor).

- ▶ Escrever um programa Java (código-fonte), usando o editor;
- ▶ Fazer quaisquer correções necessárias;
- ▶ Salvar o programa;
- ▶ Um nome de arquivo que termina com a extensão extensão `.java` indica que o arquivo contém o código-fonte Java.

Fase 1: Edita



Fase 2: Compilação



Use o comando javac (o compilador Java) para compilar um programa.

- ▶ Por exemplo, para compilar um programa chamado Welcome.java, você digitaria:

`javac Welcome.java`

- ▶ Se o programa compilar, o compilador produz um arquivo .class chamado Welcome.class que contém a versão compilada do programa.

Fase 2: Compila



O compilador cria bytecodes e os armazena em disco em um arquivo cujo nome termina com .class.

Fase 2: Compilação

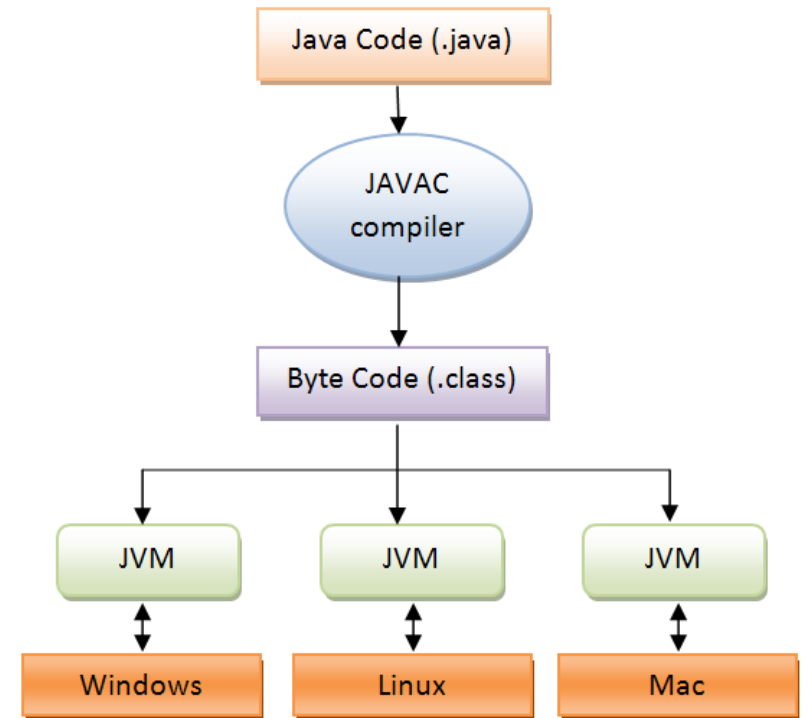
O compilador Java converte o código-fonte Java em bytecodes que representam as tarefas a ser executadas.

Os bytecodes são executados pela Java Virtual Machine (JVM) - uma parte do JDK e a base da plataforma Java.

Máquina Virtual (VM) - um aplicativo de software que simula um computador.

Oculta o sistema operacional e o hardware subjacentes dos programas que interagem com ela.

Se a mesma VM for implementada nas várias plataformas de computador, os aplicativos que ela executa podem ser utilizados em todas essas

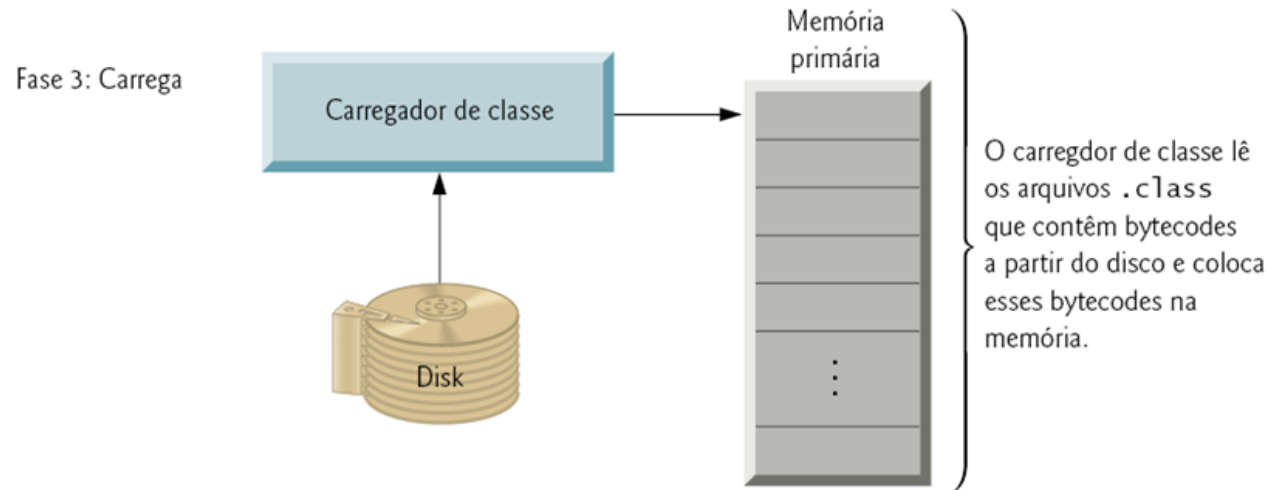


Fase 3: Carregamento



A JVM armazena o programa na memória para executá-lo. Isso é conhecido como carregamento.

- ▶ O carregador de classe pega os arquivos .class que contêm os bytecodes do programa e transfere-os para a memória primária.
- ▶ Também carrega qualquer arquivo .class fornecido pelo Java que seu programa utiliza.



Fase 4: Verificação



À medida que as classes são carregadas, o verificador de bytecode examina seus bytecodes. O carregamento assegura que eles são válidos e não violam as restrições de segurança do Java.

- ▶ O Java impõe uma forte segurança para certificar-se de que os programas Java que chegam pela rede não danificam os arquivos ou o sistema (como vírus e vermes de computador).

Fase 4: Verifica



primária



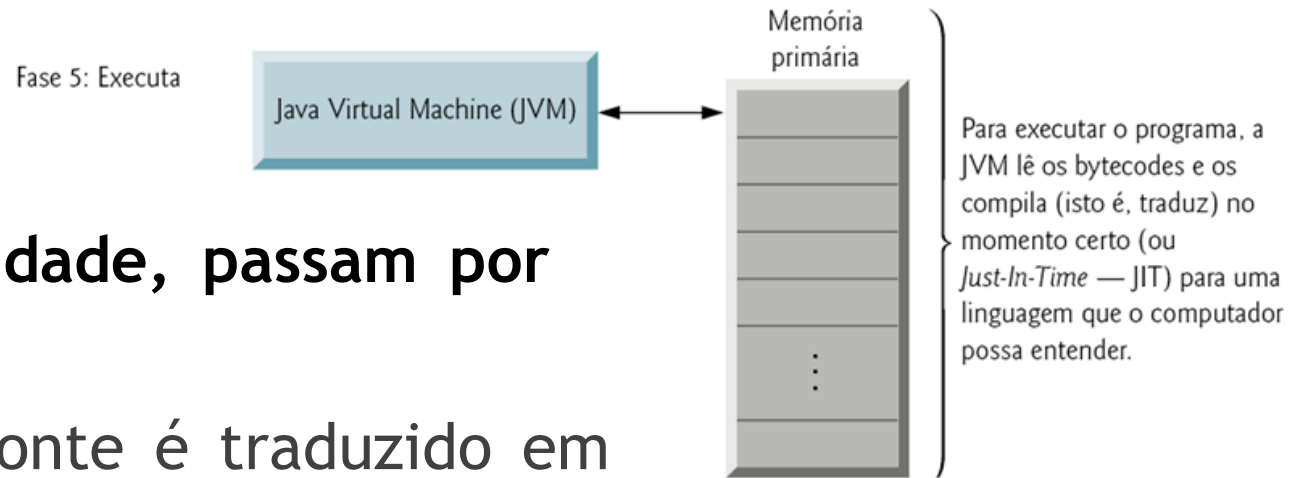
O verificador de bytecode confirma que todos os bytecodes são válidos e não violam restrições de segurança do Java.

Fase 5: Execução



A JVM executa os bytecodes do programa.

- ▶ A JVM lê os bytecodes e os compila para uma linguagem que o computador possa entender



Os programas Java, na realidade, passam por duas fases de compilação:

- ▶ Uma fase em que código-fonte é traduzido em bytecodes (para a portabilidade entre JVMs de diferentes plataformas de computador).
- ▶ Uma segunda em que, durante a execução, os bytecodes são traduzidos em linguagem de máquina para o computador real em que o programa é executado.

A Linguagem de Programação Java

Atualmente a linguagem Java é uma das mais utilizadas pelos programadores, até mesmo nas mais recentes inovações, como por exemplo, a tecnologia Android.

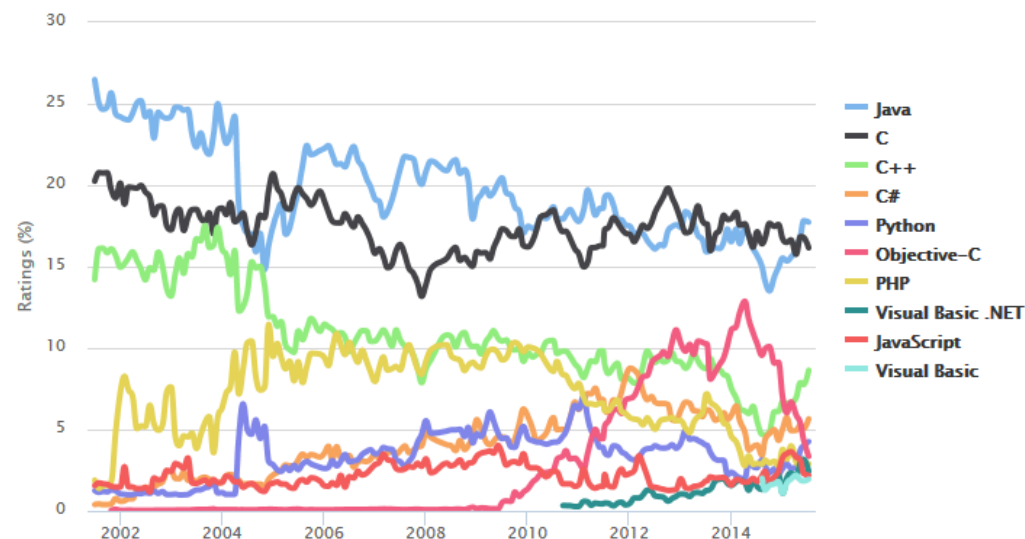
TIOBE Index for July 2015

Jul 2015	Jul 2014	Change	Programming Language	Ratings	Change
1	2	▲	Java	17.728%	+2.04%
2	1	▼	C	16.147%	-1.00%
3	4	▲	C++	8.641%	+3.12%
4	6	▲	C#	5.652%	+1.60%
5	8	▲	Python	4.257%	+1.60%
6	3	▼	Objective-C	3.344%	-6.95%
7	7		PHP	2.893%	-0.02%
8	12	▲▲	Visual Basic .NET	2.423%	+0.93%
9	9		JavaScript	2.194%	+0.39%
10	-	▲▲	Visual Basic	1.946%	+1.95%



TIOBE (TIOBE Programming Community Index)

- ▶ É uma lista ordenada de linguagens de programação, classificada pela frequência de pesquisa na web usando o nome da linguagem como a palavra-chave.
- ▶ De acordo com o site, o índice TIOBE não é sobre a melhor linguagem de programação, ou em qual se tem escrito a maior quantidade de linhas de código.



Exemplo de Código em Java

```
Welcome.java
1  /**
2   * Programa Welcome.java
3   * Meu primeiro programa em Java
4   */
5
6   //Programa de Impressão de Texto
7  public class Welcome {
8
9      //Método principal que inicia a execução do programa Java
10     public static void main(String[] args) {
11
12         /**
13          * Imprime a string de caracteres contida entre as aspas duplas
14          * na anela de comandoa partir da qual o aplicativo Java executa.
15          */
16         System.out.println("Olá Mundo!");
17
18     } //Fim do método Main
19
20 } //Fim da classe Welcome
21
```

Comentários em Java

The image shows a code editor window titled 'Welcome.java' with the following code and annotations:

```
1  /**
2   * Programa Welcome.java
3   * Meu primeiro programa em Java
4   */
5
6  //Programa de Impressão de Texto
7  public class Welcome {
8
9      //Método principal que inicia a execução do programa Java
10     public static void main(String[] args) {
11
12         /**
13          * Imprime a string de caracteres contida entre as aspas duplas
14          * na anela de comandoa partir da qual o aplicativo Java executa.
15          */
16         System.out.println("Olá Mundo!");
17
18     } //Fim do método Main
19
20 } //Fim da classe Welcome
21
```

Callout 1 (lines 1-4): Comentários do Javadoc, que permite incorporar a documentação do programa diretamente nos programas.

Callout 2 (line 6): Comentário de fim de linha, que termina no fim da linha em que aparece.

Callout 3 (lines 12-15): Comentário tradicional, que pode se distribuir por várias linhas

Comentários em Java

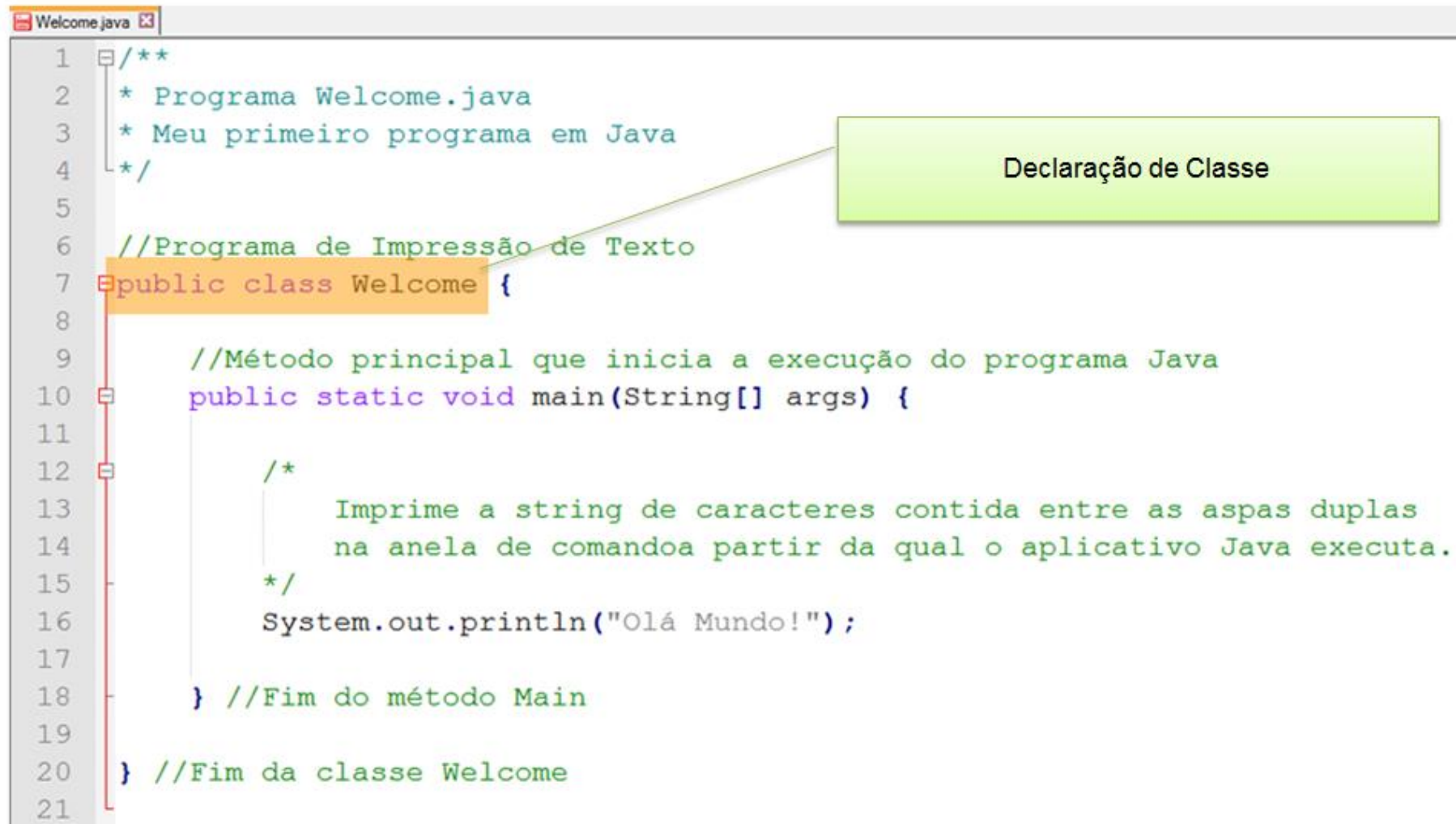
Utilizados para documentar programas e aprimorar sua legibilidade.



Comentários

- ▶ O compilador ignora comentários;
- ▶ Comentar programas é uma boa prática de programação.

Declaração de Classe



```
1  /**
2   * Programa Welcome.java
3   * Meu primeiro programa em Java
4   */
5
6  //Programa de Impressão de Texto
7  public class Welcome {
8
9      //Método principal que inicia a execução do programa Java
10     public static void main(String[] args) {
11
12         /**
13          * Imprime a string de caracteres contida entre as aspas duplas
14          * na anela de comandoa partir da qual o aplicativo Java executa.
15          */
16         System.out.println("Olá Mundo!");
17
18     } //Fim do método Main
19
20 } //Fim da classe Welcome
21
```

Declaração de Classe

Classe em Java

A palavra-chave class introduz uma definição de classe em Java.



Classes em Java

- ▶ Todo programa Java consiste em pelo menos uma classe que você define;
- ▶ Por convenção, iniciam com uma letra maiúscula e apresentam a letra inicial de cada palavra que eles incluem em maiúscula;
- ▶ O nome de uma classe Java é um identificador;
- ▶ O Java faz distinção entre maiúsculas e minúsculas.

Chaves

```
1  /**
2   * Programa Welcome.java
3   * Meu primeiro programa em Java
4   */
5
6   //Programa de Impressão de Texto
7  public class Welcome {
8
9      //Método principal que inicia a execução do programa Java
10     public static void main(String[] args) {
11
12         /**
13          * Imprime a string de caracteres contida entre as aspas duplas
14          * na anela de comandoa partir da qual o aplicativo Java executa.
15          */
16         System.out.println("Olá Mundo!");
17
18     } //Fim do método Main
19
20 } //Fim da classe Welcome
21
```

Uma **chave esquerda**, {, inicia o **corpo** de toda definição de classe.

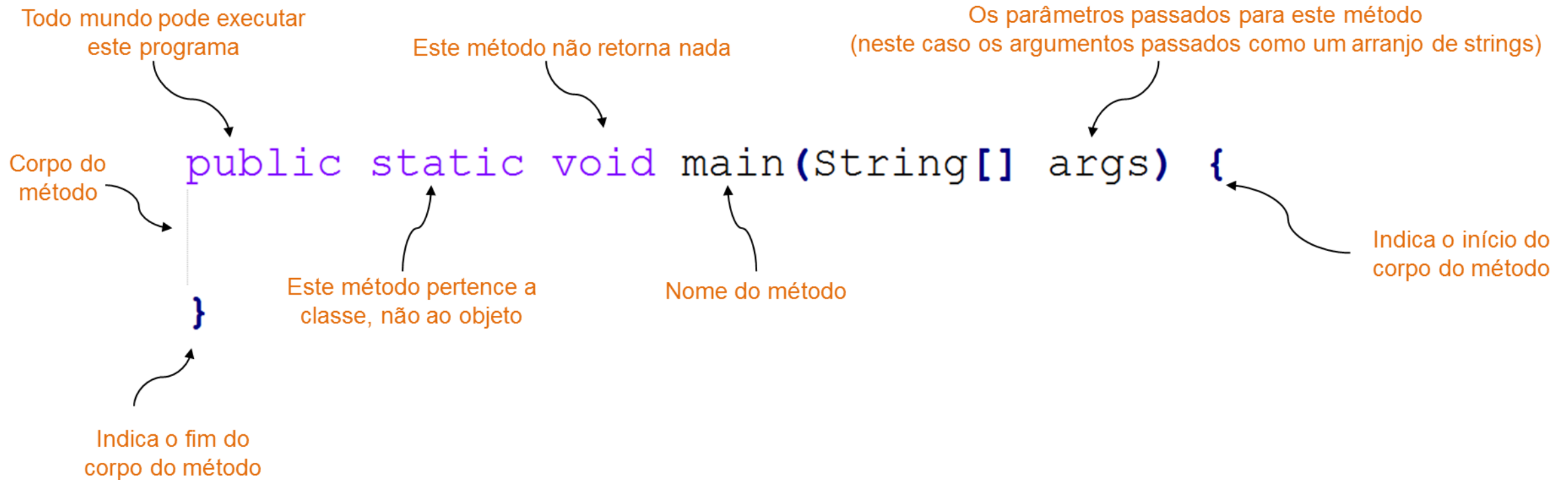
Uma **chave direita** correspondente, }, deve terminar cada definição de classe.

Método Main

```
Welcome.java
1  /**
2   * Programa Welcome.java
3   * Meu primeiro programa em Java
4   */
5
6  //Programa de Impressão de Texto
7  public class Welcome {
8
9      //Método principal que inicia a execução do programa Java
10     public static void main(String[] args) {
11
12         /*
13          * Imprime a string de caracteres contida entre as aspas duplas
14          * na anela de comandoa partir da qual o aplicativo Java executa.
15          */
16         System.out.println("Olá Mundo!");
17
18     } //Fim do método Main
19
20 } //Fim da classe Welcome
21
```

O ponto de partida de todo aplicativo Java.

Método Main



Método System.out.println

```
Welcome.java
1  /**
2   * Programa Welcome.java
3   * Meu primeiro programa em Java
4   */
5
6   //Programa de Impressão de Texto
7  public class Welcome {
8
9      //Método principal que inicia a execução do programa Java
10     public static void main(String[] args) {
11
12         /**
13          * Imprime a string de caracteres contida entre as aspas duplas
14          * na anela de comandoa partir da qual o aplicativo Java executa.
15          */
16         System.out.println("Olá Mundo!");
17     } //Fim do método Main
18 } //Fim da classe Welcome
19
20
21
```

Exibe (ou imprime) uma linha de texto na janela de comando.
A *string* entre parênteses é o **argumento** para o método.

Compilando e Executando um Programa em Java

```
Welcome.java
1 public class Welcome {
2
3     public static void main(String[] args) {
4         System.out.println("Ola Mundo!");
5     }
6
7 }
```

Para compilar o programa, digite:
javac Welcome.java

```
Administrador: C:\Windows\system32\cmd.exe

C:\aula>javac Welcome.java
C:\aula>dir

Pasta de C:\aula
03/08/2015  16:22    <DIR>          .
03/08/2015  16:22    <DIR>          ..
03/08/2015  16:23             430 Welcome.class
03/08/2015  15:55             139 Welcome.java
                2 arquivo(s)          569 bytes
                2 pasta(s) 37.696.843.776 bytes disponíveis

C:\aula>
```

Arquivo de classe contendo
os bytecodes Java

Código fonte

Para executar o programa, digite:
java Welcome

```
Administrador: C:\Windows\system32\cmd.exe

C:\aula>java Welcome
Ola Mundo!
C:\aula>
```

O programa executa e imprime
Ola Mundo!

Entrada/Saída simples com a Classe Scanner

```
1 // Figura 2.7: Addition.java
2 // Programa de adição que exibe a soma de dois números.
3 import java.util.Scanner; // programa utiliza a classe Scanner
4
5 public class Addition
6 {
7     // método principal inicia a execução do aplicativo Java
8     public static void main( String[] args )
9     {
10         // cria um Scanner para obter entrada da janela de comando
11         Scanner input = new Scanner( System.in );
12
13         int number1; // primeiro número a adicionar
14         int number2; // segundo número a adicionar
15         int sum; // soma de number1 e number2
16
17         System.out.print( "Enter first integer: " ); // prompt
18         number1 = input.nextInt(); // lê primeiro o número fornecido pelo usuário
19
20         System.out.print( "Enter second integer: " ); // prompt
21         number2 = input.nextInt(); // lê o segundo número fornecido pelo usuário
22
23         sum = number1 + number2; // soma os números, depois armazena o total em sum
24
25         System.out.printf( "Sum is %d\n", sum ); // exibe a soma
26     } // fim do método main
27 } // fim da classe Addition
```

Importa a classe Scanner para uso neste programa

Cria Scanner para ler dados fornecidos pelo usuário

Variáveis que são declaradas mas não inicializadas

Lê um valor int fornecido pelo usuário

Lê outro valor int fornecido pelo usuário

Soma os valores number1 e number2

```
Enter first integer: 45
Enter second integer: 72
Sum is 117
```

Entrada/Saída baseada em GUI simples com JOptionPane

```
1 // Figura 14.2: Addition.java
2 // Programa de adição que utiliza JOptionPane para entrada e saída.
3 import javax.swing.JOptionPane; // programa utiliza JOptionPane
4
5 public class Addition
6 {
7     public static void main( String[] args )
8     {
9         // obtém a entrada de usuário a partir dos diálogos de entrada JOptionPane
10        String firstNumber =
11            JOptionPane.showInputDialog( "Enter first integer" );
12        String secondNumber =
13            JOptionPane.showInputDialog( "Enter second integer" );
14
15        // converte String em valores int para utilização em um cálculo
16        int number1 = Integer.parseInt( firstNumber );
17        int number2 = Integer.parseInt( secondNumber );
18
19        int sum = number1 + number2; // soma os números
20
21        // exibe o resultado em um diálogo de mensagem JOptionPane
22        JOptionPane.showMessageDialog( null, "The sum is " + sum,
23            "Sum of Two Integers", JOptionPane.PLAIN_MESSAGE );
24    } // fim do método main
25 } // fim da classe Addition
```

Exibe um diálogo de entrada e retorna um digitado pelo usuário

Exibe um diálogo de mensagem centralizado na tela sem nenhum ícone.