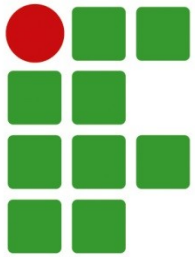


INSTITUTO FEDERAL
Goiás

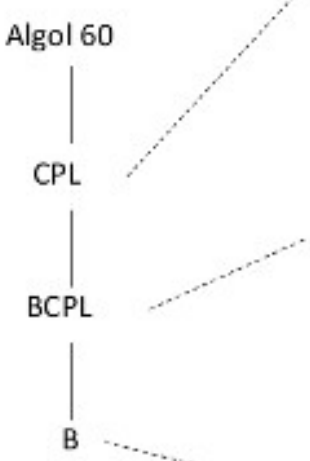
Instituto Federal de Goiás
Campus Goiânia
Bacharelado em Sistemas de Informação
Disciplina: Programação Orientada a Objetos I

Breve Histórico da Linguagem Java
e
Classes x Objetos

Prof. Ms. Dory Gonzaga Rodrigues
Goiânia - GO

Breve Histórico da Linguagem Java

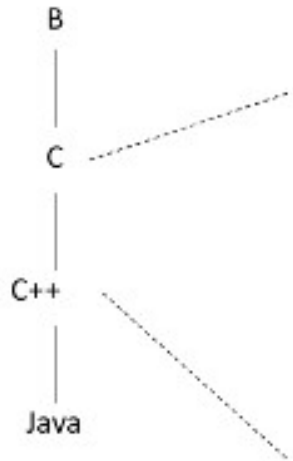
A linguagem Java herdou várias funções de outras linguagens como C e C++.



Uma linguagem ainda maior que Algol, com construtos para processamento de dados de negócios. Christopher Strachey et al. 1962-66

"Basic CPL." CPL simplificada. Sem tipos: dados são palavras de máquina. Introduziu a notação comum de arranjos: A[i] e chaves para blocos. Martin Richards (a student of Strachey), 1967

Uma linguagem ainda mais simples que BCPL. Foi usada nos primeiros sistemas UNIX. Re-introduziu as atribuições compostas: (`a+=b`), vistas em Algol 68. Ken Thompson, 1969



Extensão de B (originalmente chamada "NB"). O objetivo era aproveitar melhor o hardware (PDP-11). Sistema de tipos, pre-processador, bibliotecas de entrada/saída, etc. Foi usada para re-implementar o kernel do UNIX e várias aplicações deste sistema operacional.

Dennis Ritchie et. al., 1971-1973

C++ era originalmente um preprocessador de C que incluia na linguagem orientação por objetos: "C++ = C com Classes". Trouxe dynamic dispatch, sobrecarga de operadores e funções, polimorfismo paramétrico, tratamento de exceções.

Bjarne Stroustrup, 1984

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.

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

James Gosling, considerado o pai da linguagem Java

- 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

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

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

Em 1993, com a grande popularidade da INTERNET no mundo, os idealizadores do projeto Green viram a oportunidade de utilizar Java para a criação de páginas 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:

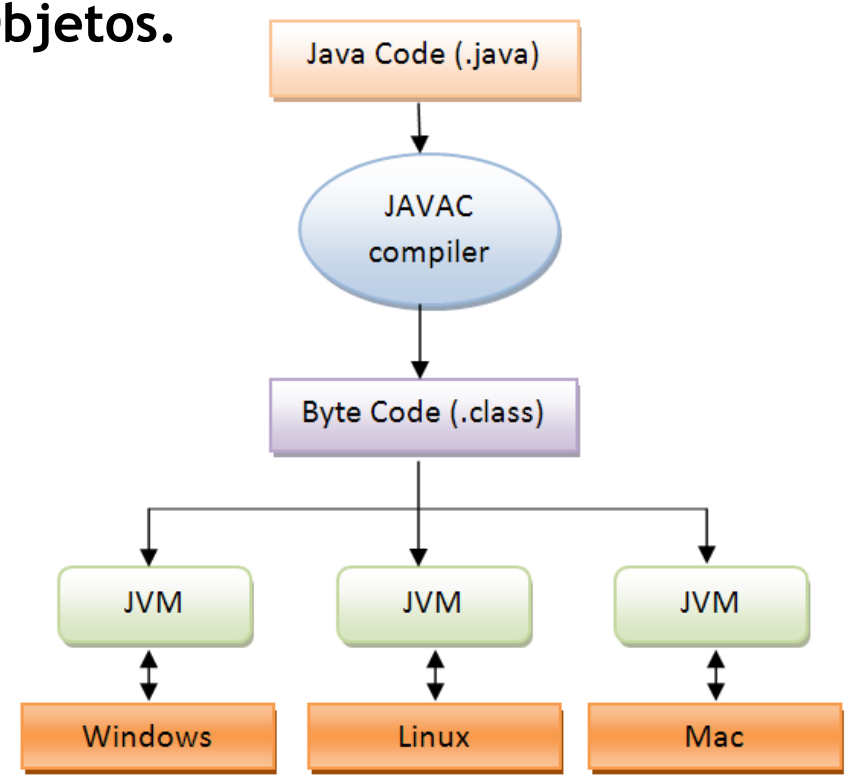
- Lógica de Programação.

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.

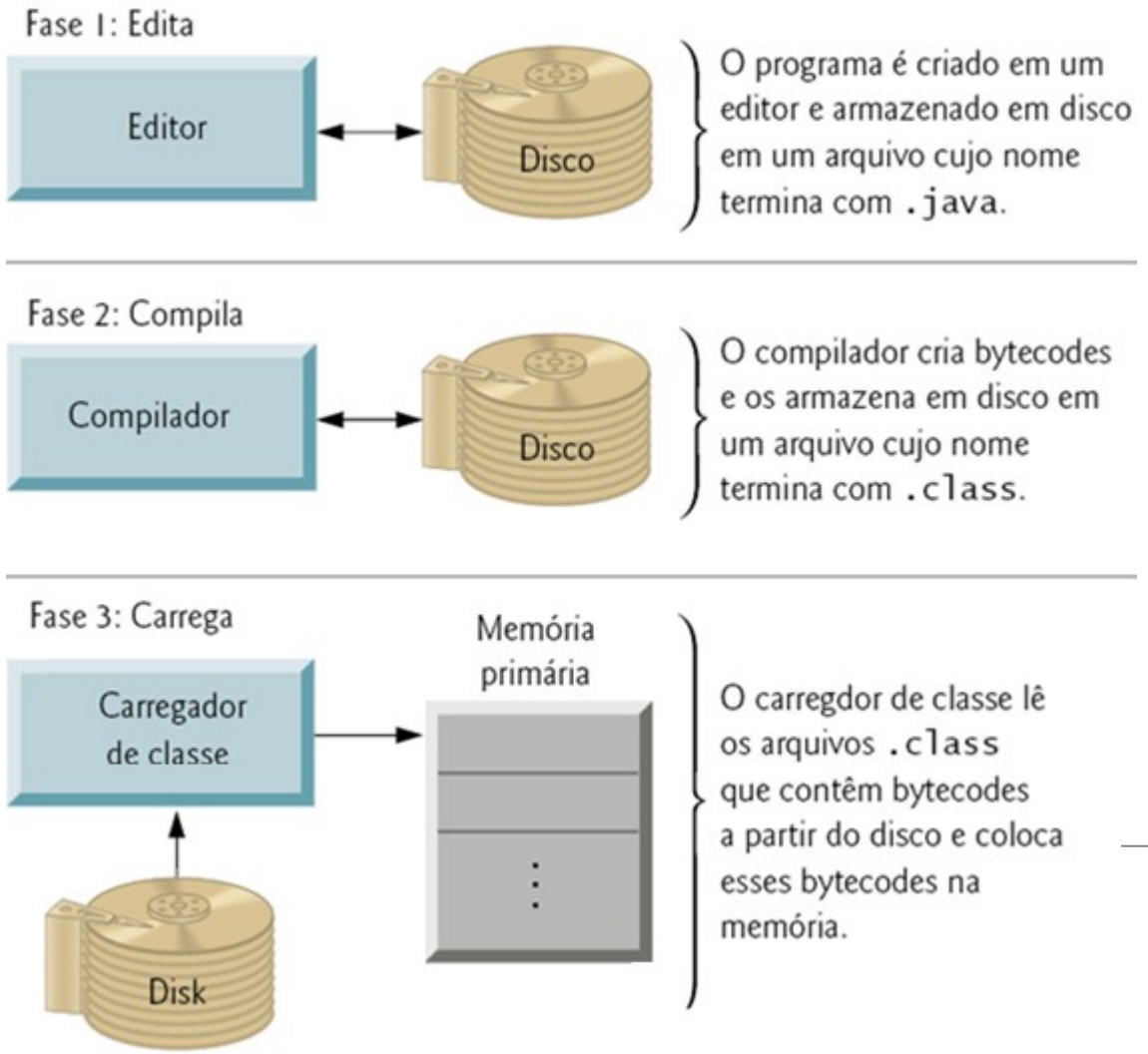
Interpretada, Neutra, Portável, Simples e Orientada a Objetos.

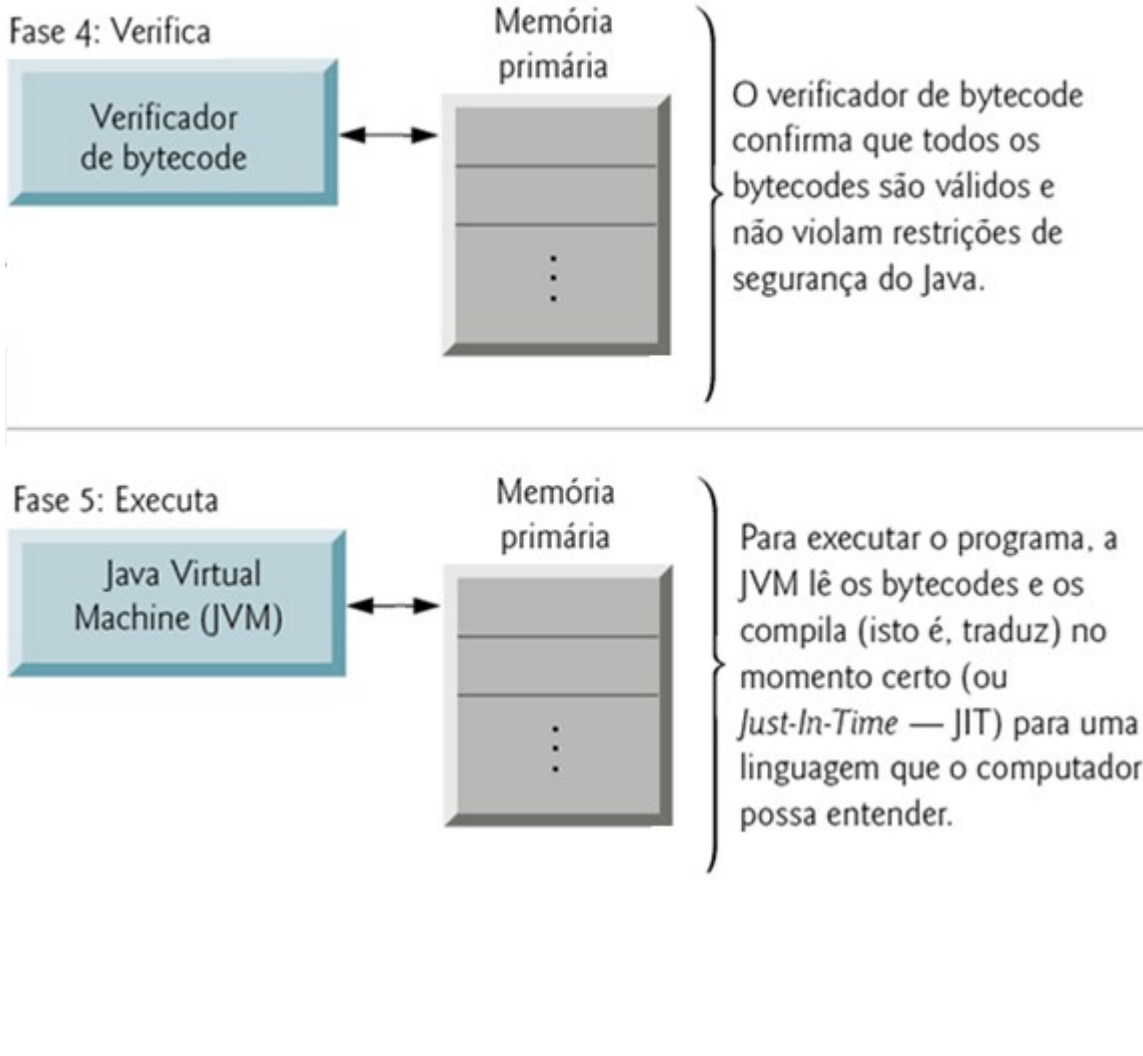
- Permite que o código em Java possa ser escrito independente da plataforma;
- Bytecodes executam em qualquer dispositivo que possua uma Java Virtual Machine;
- Neutra em relação à arquitetura, uma característica que permite uma grande portabilidade.
- É de fácil aprendizado e orientada a objetos.



Ciclo de Execução em Java

Programas Java normalmente passam por cinco fases:

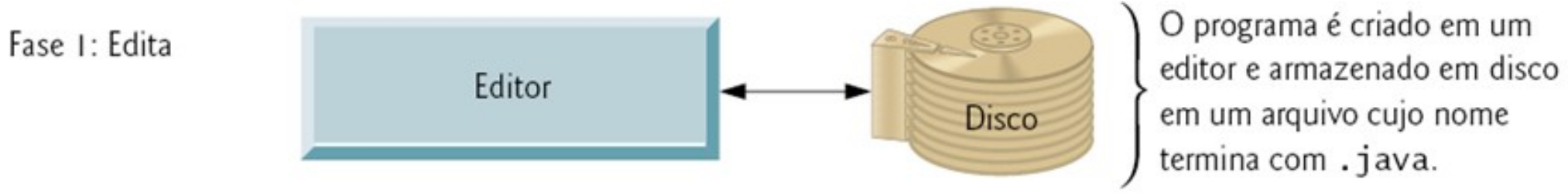




Fase 1: Edição

Consiste em editar um arquivo com um programa editor.

- 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 .java indica que o arquivo contém o código-fonte Java.



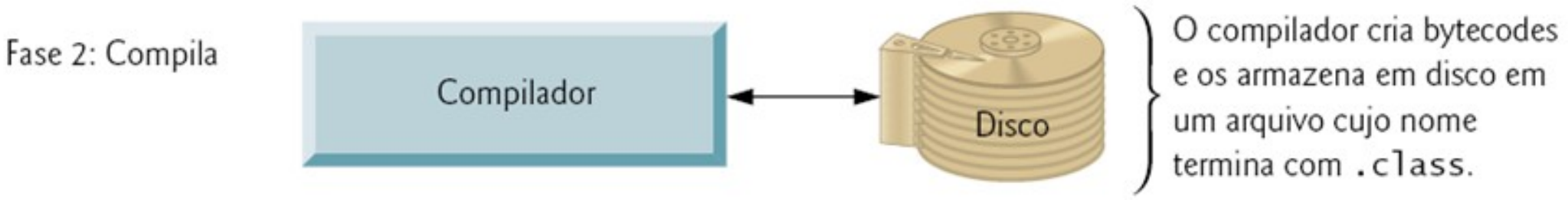
Fase 2: Compilação

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

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

  javac Sistema.java

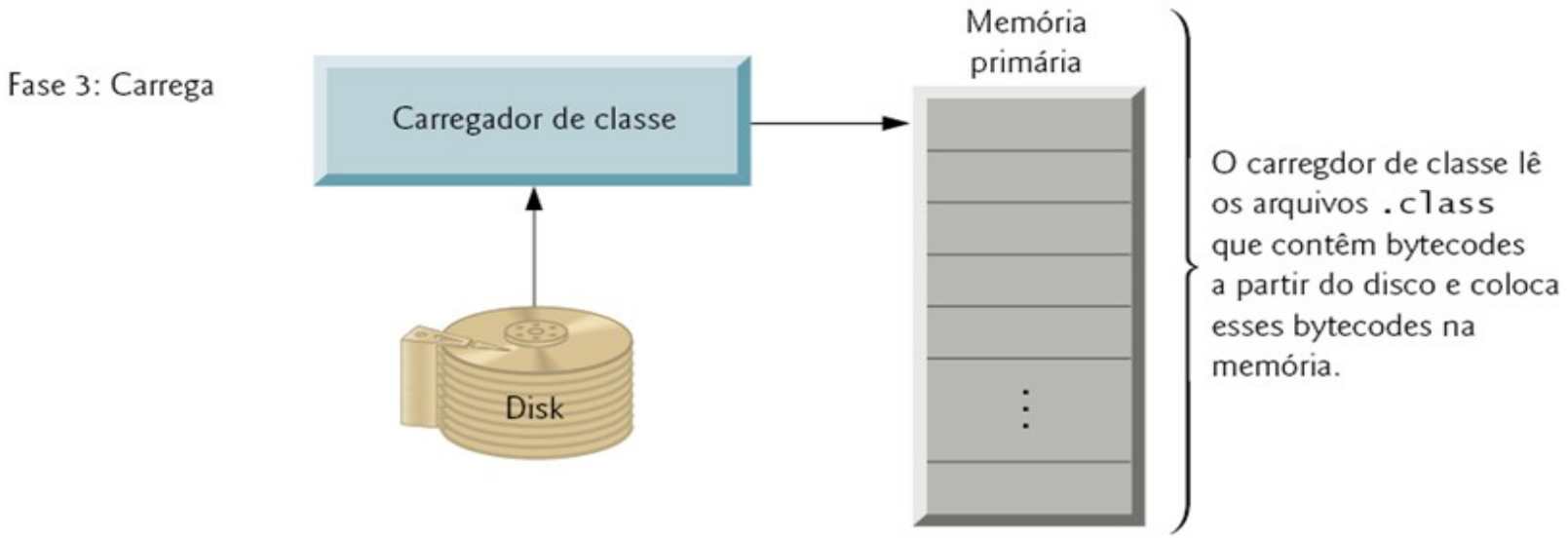
- Se o programa compilar, o compilador produz um arquivo .class chamado Sistema.class que contém a versão compilada do programa.



Fase 3: Carregamento

A Java Virtual Machine (JVM) armazena o programa na memória para executá-lo.

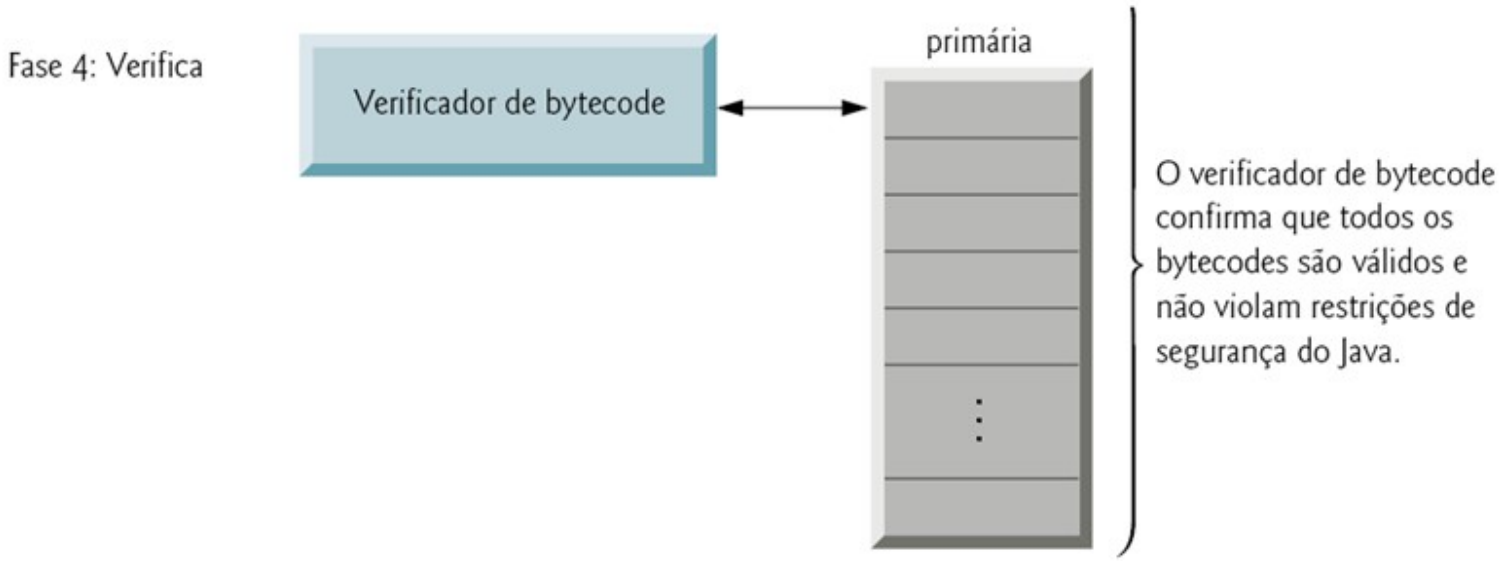
- O carregador de classe pega os arquivos .class que contêm os bytecodes do programa e transfere-os para a memória primária.
- Também carrega qualquer arquivo .class fornecido pelo Java que seu programa utiliza.



Fase 4: Verificação

À medida que as classes são carregadas, o verificador de bytecode examina seus bytecodes.

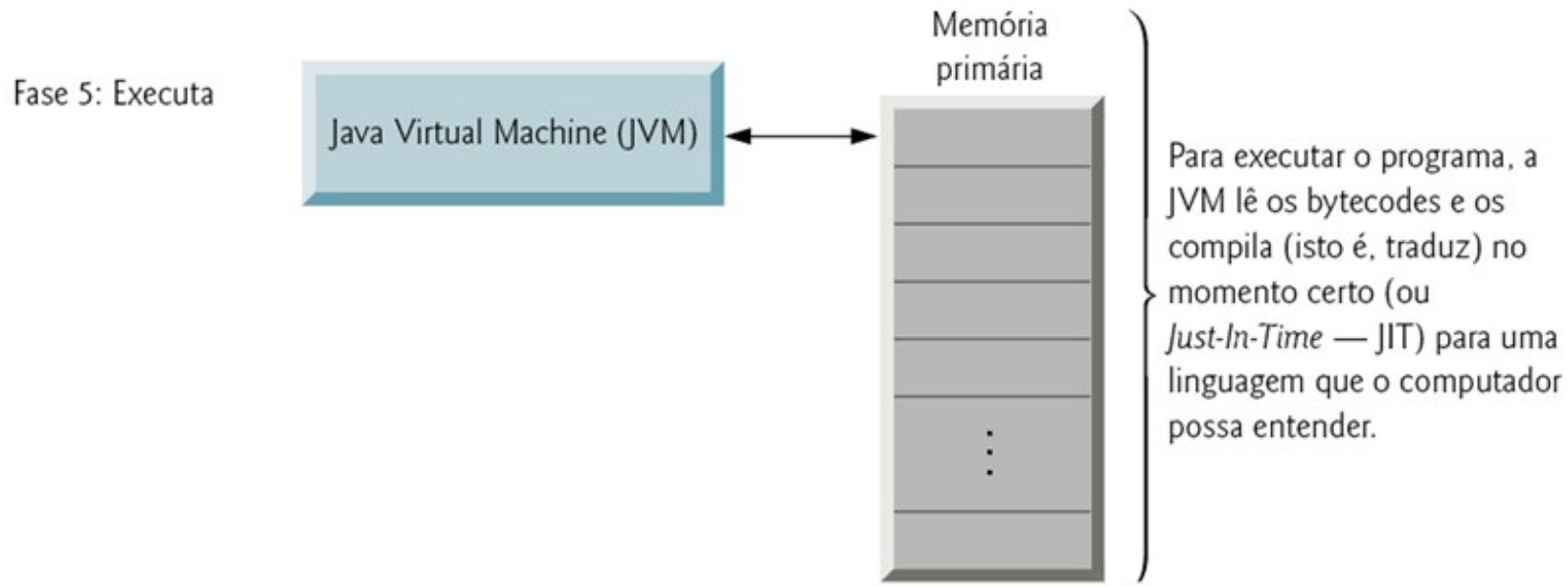
- O Java impõe uma forte segurança para certificar-se de que os programas Java não danificam os arquivos ou o sistema (como vírus e vermes de computador).
- O carregamento assegura que eles são válidos e não violam as restrições de segurança do Java.



Fase 5: Execução

A JVM executa os bytecodes do programa.

- A JVM lê os bytecodes e os compila para uma linguagem o computador entende

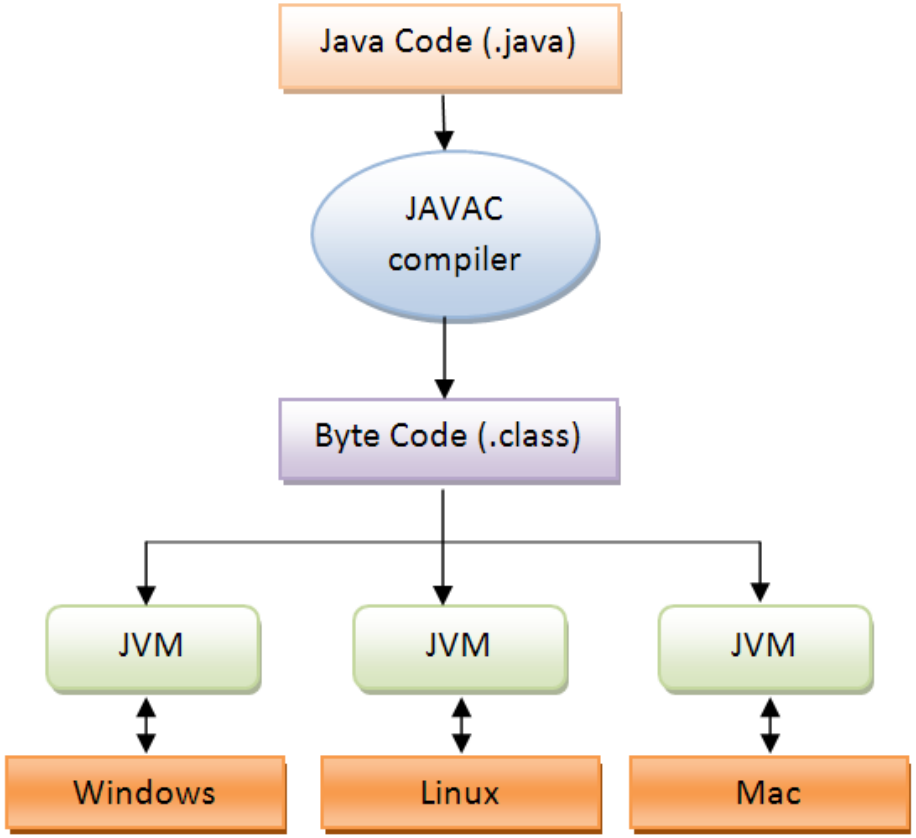


Ciclo de Execução em Java

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

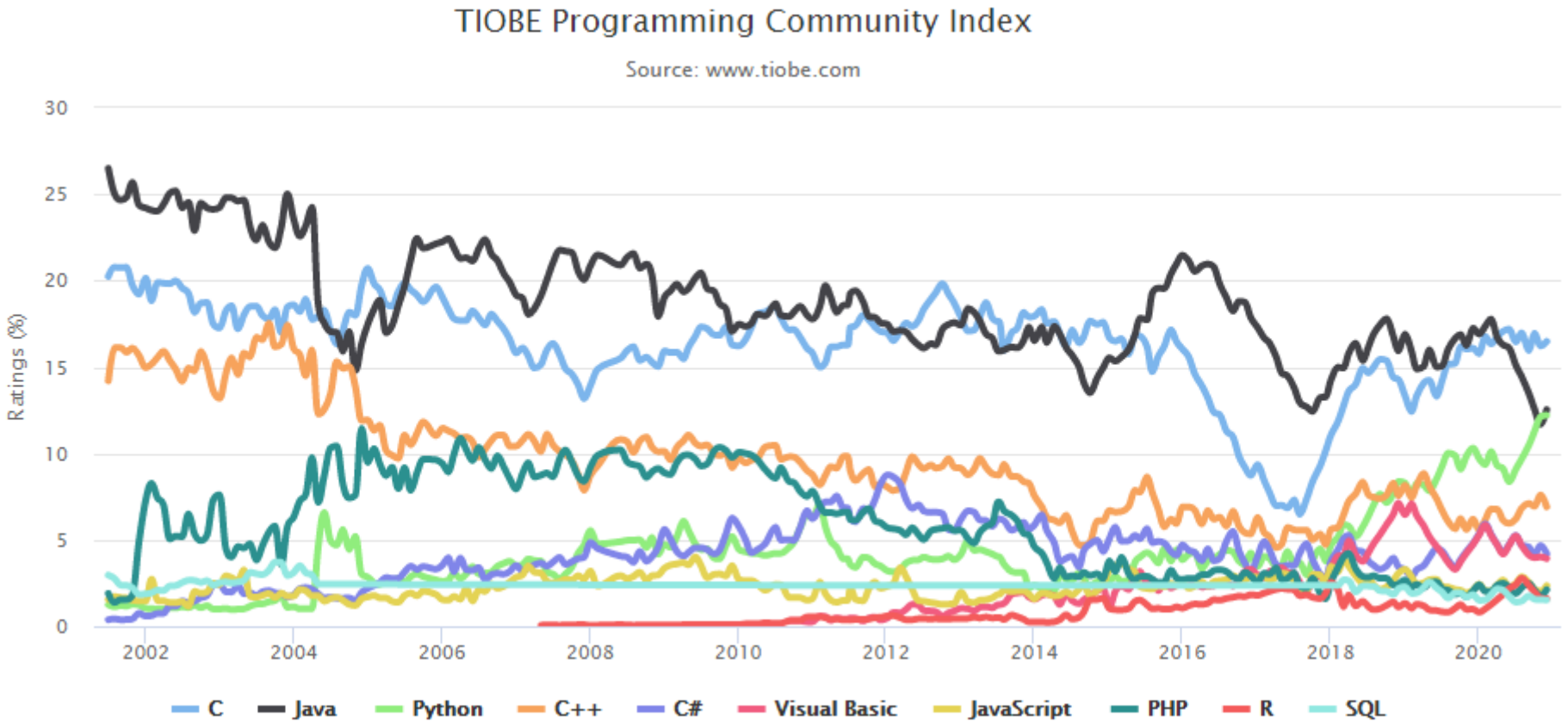
- Uma fase em que código-fonte é traduzido em bytecodes (para a portabilidade entre JVMs de diferentes plataformas de computador)

- 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

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.

Classe

A unidade fundamental de programação em orientação a objetos é a classe.

Classes contém:

Atributos

- São os dados (simples ou compostos) que caracterizam objetos daquela classe;
- São armazenadas em variáveis;
- Constituem o estado do objeto.

Operações

- São os métodos (procedimentos ou funções) que manipulam os dados.

Abstração

Construção de um modelo para representação de uma realidade, com foco nos aspectos essenciais.

Na POO, uma CLASSE é a abstração que representa a entidade existente no mundo real.

Como modelar a classe Conta

Realidade

Modelando a Classe Conta

Atributos:

Numero

Titular

Saldo

Métodos:

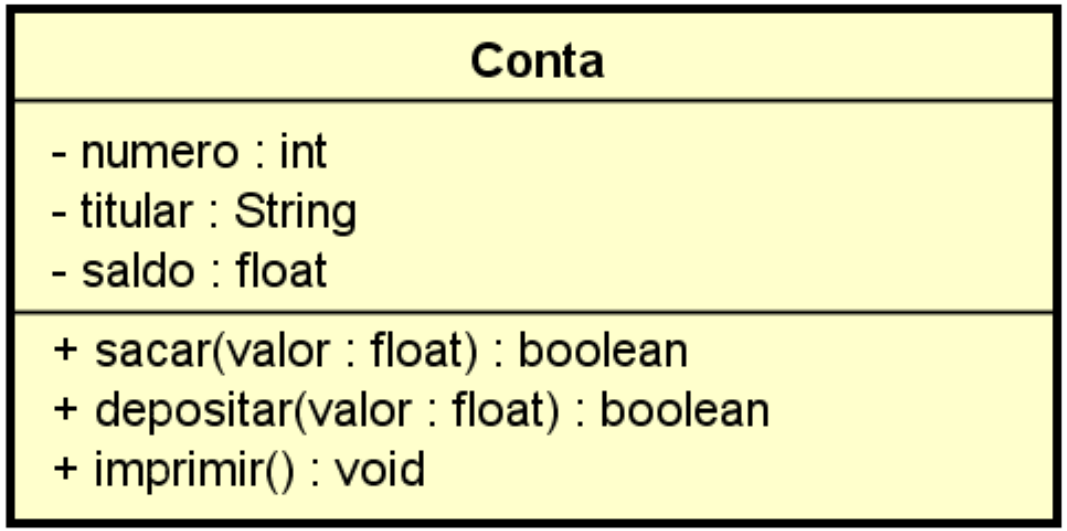
sacar()	Recebe o valor do saque e devolver um valor booleano indicando se existia saldo suficiente e a operação foi bem sucedida.
depositar()	Recebe o valor do depósito e incrementa o valor do saldo. O método deve devolver um valor booleano indicando se a operação foi bem sucedida.
imprimir()	Este método não retorna valor e deverá mostrar na tela todos os atributos da classe Conta. Para imprimir em Java, utilize o comando System.out.println().

Diagrama da Classe Conta

Diagrama de Classe

Estrutura

Comportamentos



Objetos

Um programa orientado a objetos é composto por um conjunto de objetos que interagem entre si.

Objetos:

- São instâncias da classe;
- Objetos de software são conceitualmente

similares a objetos do mundo real: eles consistem do estado e o comportamento;

- Um objeto armazena seu estado em campos (variáveis) e expõe seu comportamento através de métodos (funções);

Classe

Conta
- numero : int - titular : String - saldo : float
+ sacar(valor : float) : boolean + depositar(valor : float) : boolean + imprimir() : void

Objetos (Instância da Classe)

Conta A	Conta B
123 Dory R$ 2.500,00	001 Caroline R$ 1.000,00

Modificadores de Acesso

Definem o escopo/visibilidade de um atributo ou método.

Modificadores de acesso

public

- Simbolizado por **+**
- Indica que o atributo ou método pode ser acessado por objetos de outras classes.

private

- Simbolizado por **-**
- Não permite o acesso externo (encapsulamento), ou seja, é permitindo o acessado somente por métodos da própria classe e pelo objeto instanciado desta classe;

protected

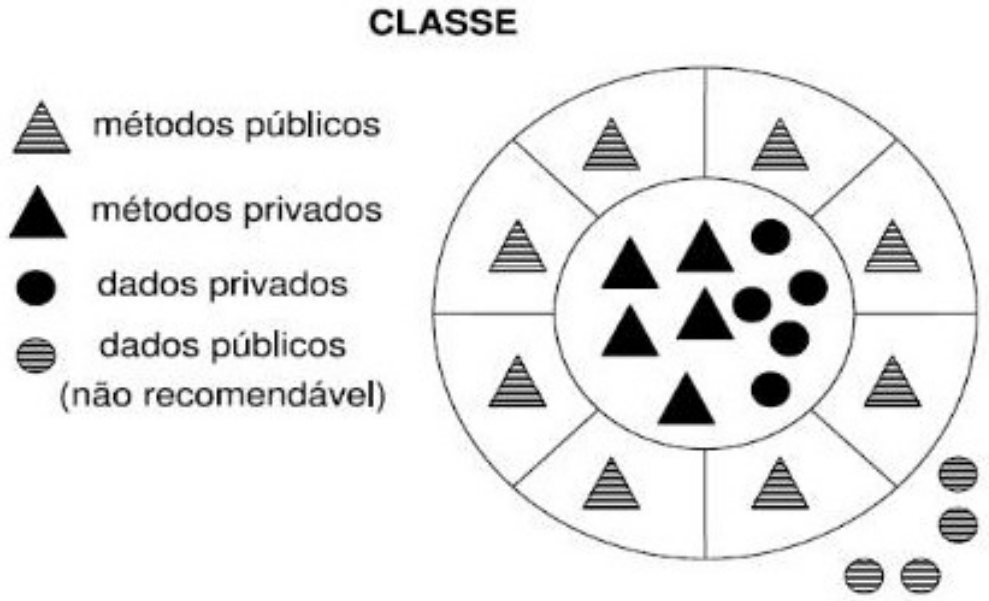
- Simbolizado por **#**
- Permite o acesso externo da classes filhas, mas impede o acesso de objetos que não fazem parte da hierarquia de classes.

A Ideia de Encapsulamento ...

Esconder todos os membros de uma classe, além de esconder como funcionam as rotinas do nosso sistema.

Encapsular é fundamental para que seu sistema seja suscetível a mudanças:

- Não precisaremos mudar uma regra de negócio em vários lugares, mas sim em apenas um único lugar, já que essa regra está encapsulada
- O conjunto de métodos públicos de uma classe, é a única maneira de se comunicar com os objetos dessa classe.



Reforçando a ideia de Encapsulamento

Muitas vezes, é necessário consultar e alterar o valor de um atributo a partir de qualquer lugar do sistema.

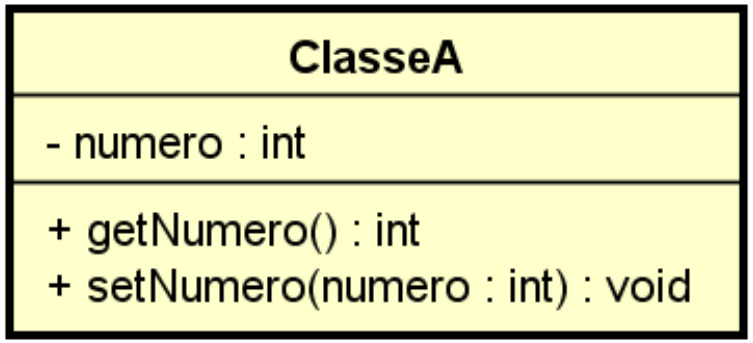
O que é melhor?
Criar métodos de leitura e outro de escrita ou deixar o atributo com acesso público ?

- Quando queremos consultar o saldo da conta corrente, utilizamos o terminal de autoatendimento ou olhamos dentro do cofre do banco?
- Quando queremos consultar a quantidade de combustível de um automóvel, olhamos o painel ou abrimos o tanque de combustível?
- Quando queremos alterar o toque da campainha de um celular, utilizamos os menus do celular ou desmontamos o aparelho?

Métodos Getters e Setters

Para permitir o acesso aos atributos, já que eles encapsulados na classe (private) de uma maneira controlada, criamos dois métodos:

- Um que retorna o valor;
- E outro que muda o valor.



O nome padrão para esses métodos:
a palavra **get** ou **set** + nome do atributo

O padrão do método get para variáveis do tipo boolean

- Esses atributos são acessados via **is** e **set**
- Exemplo: para um objeto lâmpada, seriam criados os métodos:

isLigado() e setLigado()

Comentários 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 do Javadoc, que permite incorporar a documentação do programa diretamente nos programas.

Comentário de fim de linha, que termina no fim da linha em que aparece.

Comentário tradicional, que pode se distribuir por várias linhas

Vamos a prática !

Implementando a Classe Conta

Conta.java

```
1  public class Conta {
2
3      private int numero;
4      private String titular;
5      private float saldo;
6
7      public boolean sacar(float valor){...}
8      public boolean depositar(float valor){...}
9      public void imprimir(){}
10 }
```

Conta
- numero : int - titular : String - saldo : float
+ sacar(valor : float) : boolean + depositar(valor : float) : boolean + imprimir() : void

- Classes iniciam com a palavra reservada class;
- Convenciona-se utilizar o nome da classe com a 1ª letra em maiúscula;
- Uma "{" (abre chave) marca o início e "}" (fecha chave) marca o fim da classe;
- Normalmente cada classe é salva como um arquivo de mesmo nome da classe com extensão java

Exemplo: Conta.java

Implementando a Classe Conta

Conta.java

```
1  public class Conta {
2
3      private int numero;
4      private String titular;
5      private float saldo;
6
7      public boolean sacar(float valor){...}
8      public boolean depositar(float valor){...}
9      public void imprimir(){}
10 }
```

Tipo do Atributo

Atributos
(Variáveis de Instância)

Operações
(Métodos de Instância)

Tipo de Retorno
Modificador de Acesso (Visibilidade)

Conta
- numero : int - titular : String - saldo : float
+ sacar(valor : float) : boolean + depositar(valor : float) : boolean + imprimir() : void

- Atributos e métodos iniciam pelo modificador de acesso;
- Atributos obrigatoriamente possuem um tipo;
- Métodos devem indicar o tipo de retorno ou void quando não retornam nada;
- Métodos podem receber argumentos (parâmetros).

Implementando a Classe Conta

```java
public boolean sacar(float valor) {
    boolean sucesso = false;

    if (saldo >= valor) {
        saldo -= valor;
        sucesso = true;
    }

    return sucesso;
}
```

```java
public boolean depositar(float valor) {
    boolean sucesso = false;

    if (valor >= 0) {
        saldo += valor;
        sucesso = true;
    }

    return sucesso;
}
```

```java
public void imprimir(){
    System.out.println("------");
    System.out.println("Número da Conta: " + numero);
    System.out.println("Nome do Titular: " + titular);
    System.out.println("Saldo Atual: " + saldo);
}
```

Regras e convenções de nomenclatura

- Nome de Classes
  - Toda classe deve começar com uma letra maiúscula

    Ex: Pessoa, Conta

  - Não é possível declarar uma classe com caractere especial

    Ex: @, #, $, %, &, *, _, etc ou número.

  - Classe com nome composto, a primeira letra de cada palavra deve ser em maiúscula.

    Ex: ImpostoDeRenda, AgenciaDeEmprego

  - O nome da classe deve ser exatamente o mesmo nome de seu arquivo fonte.

Regras e convenções de nomenclatura

- Nome de Pacotes
  - Devem começar com uma letra minúscula.

    Ex: usuarios

  - Não poderemos iniciar o nome de um pacote com caracteres especiais

    Ex: @, #, $, %, &, *, _, etc ou número.

  - Pacote com nome composto, a primeira letra de cada palavra deve ser em maiúscula.

    Ex: conexõesDeBancoDeDados

  - O nome do pacote deve ser o mesmo nome da pasta a qual ele se refere.

Regras e convenções de nomenclatura

- Nome de atributos ou variáveis
  - O atributos (variáveis) podem começar com qualquer letra e os caracteres **$** ou **_**

    Ex: x, y, resultado, $salario, _status
  - Não podem começar com números.
  - Atributo com nome composto, a primeira letra de cada palavra deve ser em maiúscula.

    Ex: valorDeX, valorDeY.

Regras e convenções de nomenclatura

- Nome de atributos finais ou constantes
  - Os atributos finais (constantes) devem ser escritos em letras maiúsculas.

    Ex: TAMANHO
  - Usamos underline (_) para separar nomes compostos de atributos finais (constantes).

    Ex: PARAR_DE_EXECUTAR

Métodos Construtores

Método especial chamado automaticamente pelo ambiente de execução quando um objeto é criado.

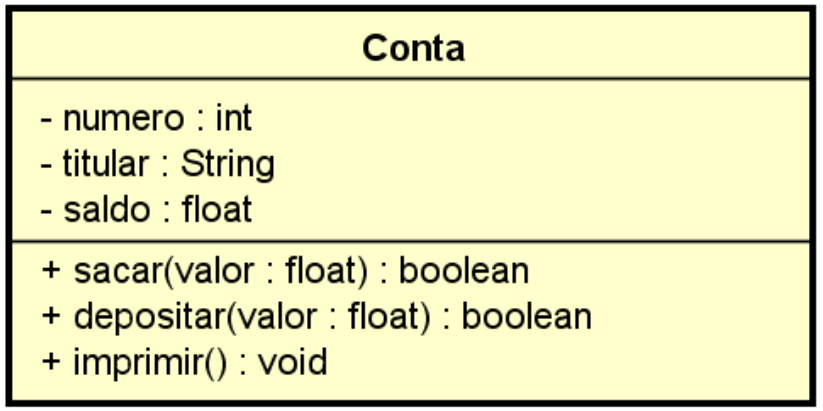
Por padrão o Java já cria esse construtor sem parâmetros (construtor default) para todas as Classes.

Conta
- numero : int - titular : String - saldo : float
+ sacar(valor : float) : boolean + depositar(valor : float) : boolean + imprimir() : void

- Na declaração do Objeto, o ***comando new*** é o responsável pela chamado do construtor;
- Pode ser utilizado para receber argumentos, podendo inicializar o estado do objeto durante a sua construção;
- Um construtor tem sempre o mesmo nome da classe a qual pertence;
- Uma classe pode possuir mais de um construtor.

Métodos Construtores com Assinaturas Diferentes

```java
public Conta(int numero, String titular, float saldo) {
    this.numero = numero;
    this.titular = titular;
    this.saldo = saldo;
}

public Conta(int numero, String titular) {
    this.numero = numero;
    this.titular = titular;
}
```



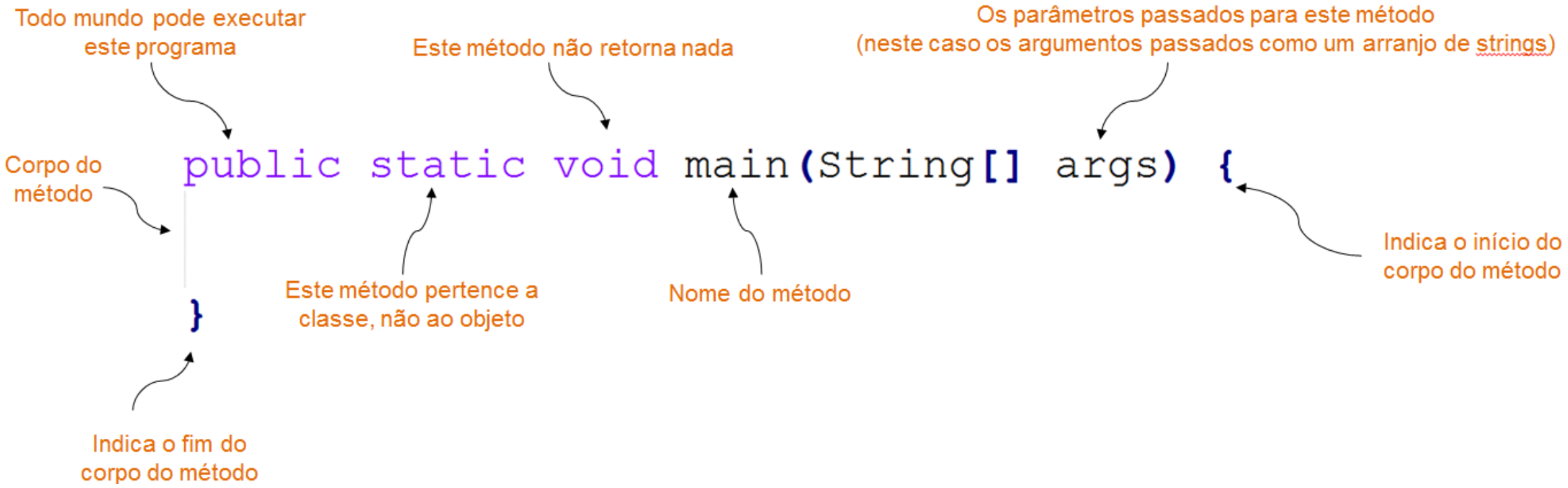
- As variáveis de instância são inicializadas automaticamente com o valor padrão do seu tipo.
- Neste caso, ***this*** resolve a ambiguidade de nomes (parâmetros x atributos)
- É usual criar construtores que recebem diversos argumentos para não obrigar o usuário de uma classe chamar diversos métodos do tipo "set".

Código Completo da Classe Conta

```java
public class Conta {

    private int numero;
    private String titular;
    private float saldo;

    public Conta(int numero, String titular, float saldo) {
        this.numero = numero;
        this.titular = titular;
        this.saldo = saldo;
    }

    public Conta(int numero, String titular) {
        this.numero = numero;
        this.titular = titular;
    }

    public int getNumero() {
        return numero;
    }
    public String getTitular() {
        return titular;
    }
    public float getSaldo() {
        return saldo;
    }

    public boolean sacar(float valor) {
        boolean sucesso = false;

        if (saldo >= valor) {
            saldo -= valor;
            sucesso = true;
        }

        return sucesso;
    }
    public boolean depositar(float valor) {
        boolean sucesso = false;

        if (valor >= 0) {
            saldo += valor;
            sucesso = true;
        }

        return sucesso;
    }
    public void imprimir(){
        System.out.println("------");
        System.out.println("Número da Conta: " + numero);
        System.out.println("Nome do Titular: " + titular);
        System.out.println("Saldo Atual: " + saldo);
    }
```

Implementando o Método Main

O ponto de partida de todo aplicativo Java.



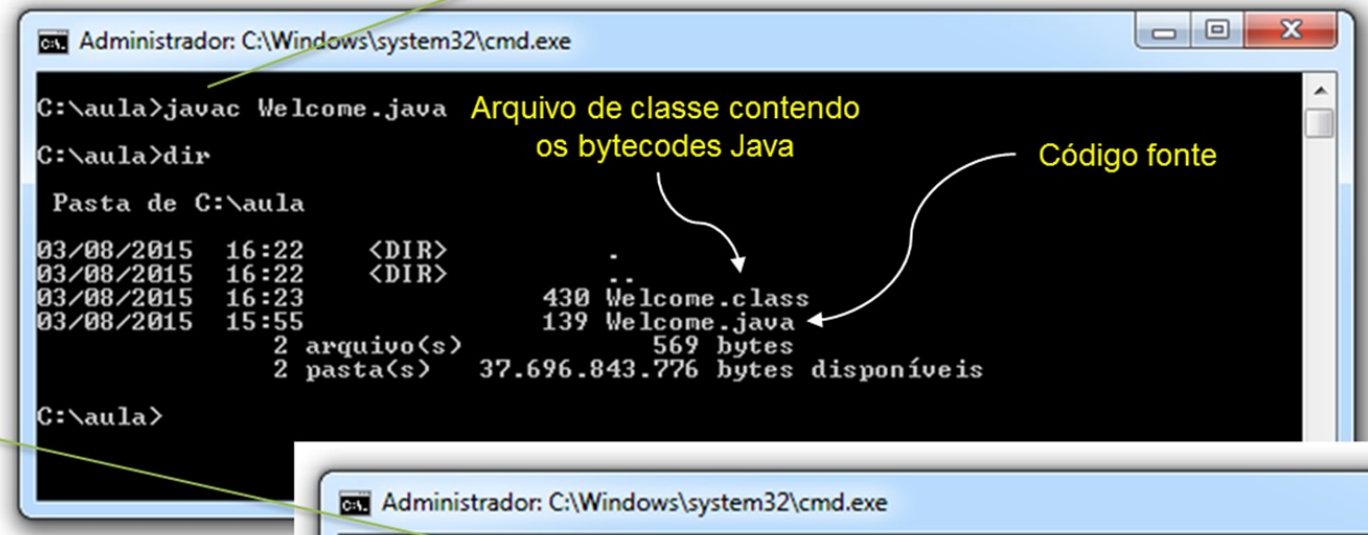
Implementando o Método Main

O ponto de partida de todo aplicativo Java.

Welcome.java

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

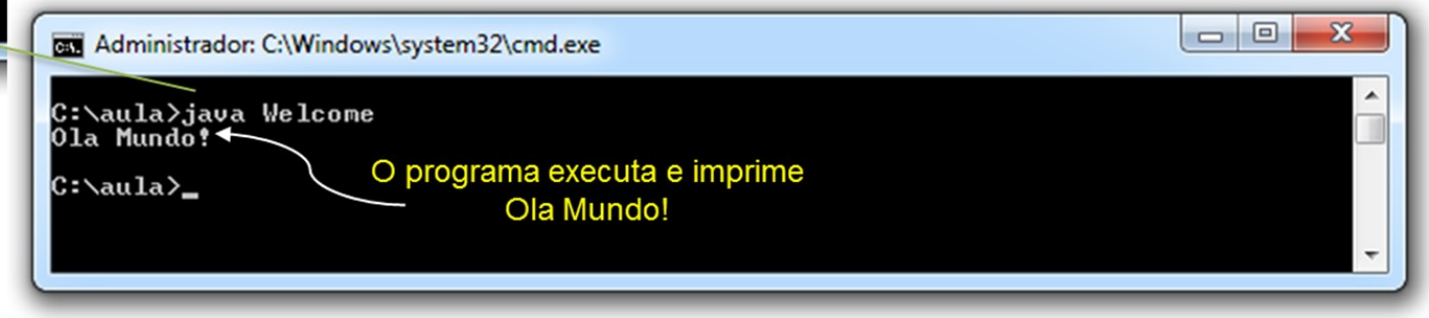
Para compilar o programa, digite:
`javac Welcome.java`



Arquivo de classe contendo os bytecodes Java

Código fonte

Para executar o programa, digite:
`java Welcome`



O programa executa e imprime Ola Mundo!

Implementando o Método Main

O ponto de partida de todo aplicativo Java.

Conta.java

```java
package br.edu.ifgoias.aula1;

public class Application {

    public static void main(String[] args) {

        Conta c = new Conta(123, "Dory G. Rodrigues", 2500);

        c.imprimir();

        c.depositar(100);
        c.sacar(800);
        c.depositar(5000);
        c.sacar(2000);

        c.imprimir();
    }
}
```

Saída do Programa

```
--- CONTA ---
Número: 123
Titular: Dory G. Rodrigues
Saldo: R$ 2500.0
--- CONTA ---
Número: 123
Titular: Dory G. Rodrigues
Saldo: R$ 4800.0
```

O operador new:

- Aloca memória para o novo objeto (a quantidade necessária é obtida a partir da definição da classe);
- Chama o construtor da classe para inicializar o estado do novo objeto;
- Retorna uma referência (um endereço de memória para o objeto recém criado);

Métodos e atributos:

- são acessados através do operador "." (ponto), precedido pelo nome do objeto (variável) e seguido do nome do método;