DCC205 – PROGRAMAÇÃO ORIENTADA A OBJETOS

Aula 02

Carlos Bruno Oliveira Lopes
carlosbrunocb@gmail.com

Conceito de orientação a objetos

- O que enxergamos a nossa volta?
  - Como os identificamos e os agrupamos?

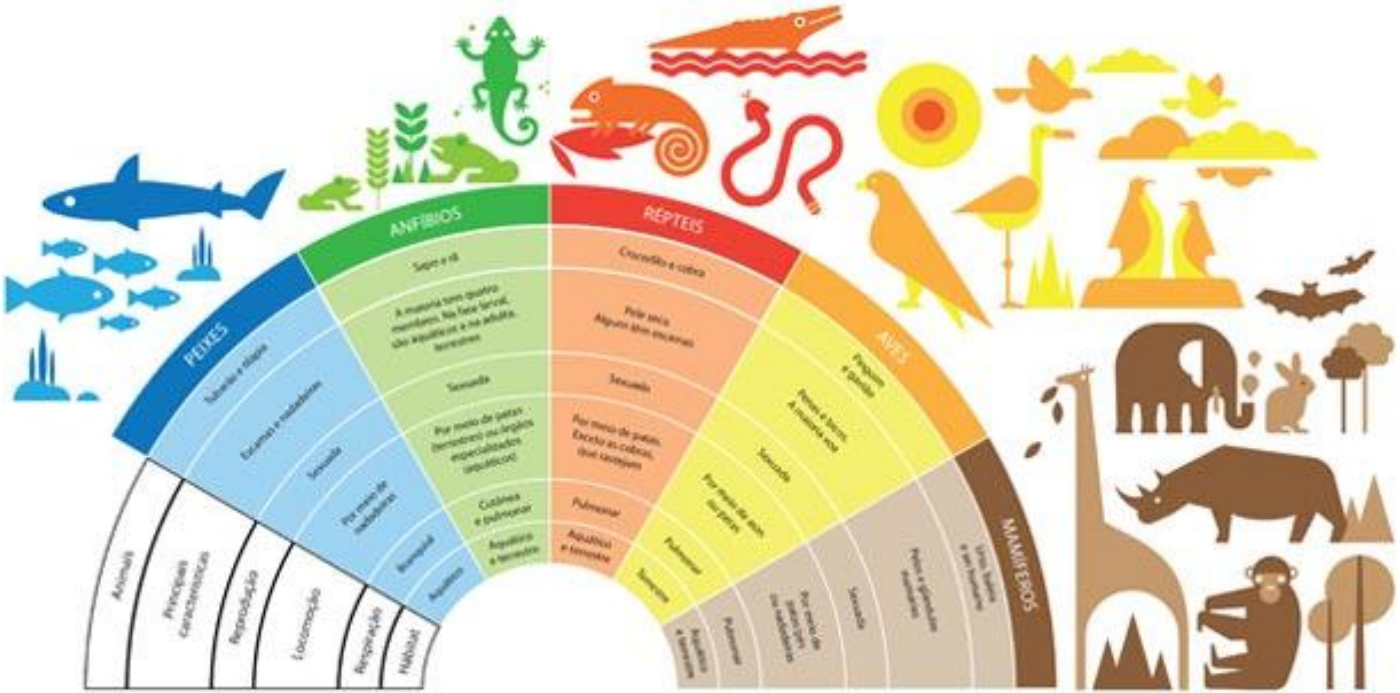
Abstrações *do todo (mundo)...*

Como um ***ser único*** *...*

Identificamos como ***OBJETOS*** *as coisas que nos cercam...*

Agrupamos os objetos com ***características comuns*** *em* ***CLASSES*** *...*

Conceito de orientação a objetos



Conceito de orientação a objetos

- O que torna um objeto um ser único?

Suas características...

Quais características?

ATRIBUTOS

Material de fabricação
Ano de fabricação
Número de identificação
Peso
Diâmetro
Raio
Área

ATRIBUTOS

Conceito de orientação a objetos

- Tendo em vista a ideia de abstração do todo;

- A maneira que compreendemos mundo e a maneira como nos relacionamos com ele;

- Podemos identificar as seguintes características:
  - **Objeto**: *ser único*.
  - **Atributos**: *características do objeto que o tornam únicos.*
  - **Classe**: *Generalização do objeto, ou seja, os classificam ou agrupam através de suas características comuns que são compartilhadas pelos seus membros.*

Conceito de orientação a objetos

- Objetos são estáticos? Eles mudam? Ou eles podem exercer ações?

Cor: vermelho;
Para-choque: tipo, formato, cor;
Faróis: tipo, formato, cor;
Roda: tamanho, cor, estilo;

O carro esta andando?
O carro esta trancado?
O carro esta com defeitos?

Conceito de orientação a objetos

- Objetos são estáticos? Eles mudam? Ou eles podem exercer ações?

- **Cor:** vermelho;
- **Para-choque:** tipo, formato, cor;
- **Faróis:** tipo, formato, cor;
- **Roda:** tamanho, cor, estilo;
- **O carro esta andando?**
- **O carro esta trancado?**
- **O carro esta com defeitos?**

Os atributos e ações (ou operações) de um objetos podem sofrer modificações ao longo do tempo

Como realizar essas mudanças que ocorrem sobre o objeto?

Métodos
(“espécie de função”)

Conceito de orientação a objetos

Classe

Bola
N° de identificação Material de fabricação Ano de fabricação Tipo de bola ⋮ Peso;

setNoId();
getNoId();
setMatFab();
getMatFab();
setAnoFab();
getAnoFab();
⋮
getPeso();

Atributos

Métodos

- Resumo d do a objetos:
  - **Classe:** ou seja, defini suas características e ações comuns **e métodos** d bjeto);
    - **Atr** bjeto" (***variáv***
    - **Mét** u mudanças na ***(funções)*** ;
  - **Objetos** *ser criado cu* *scrita pela sua classe")*

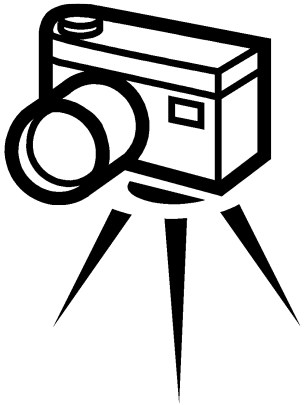
Bola de Futebol de Futsal

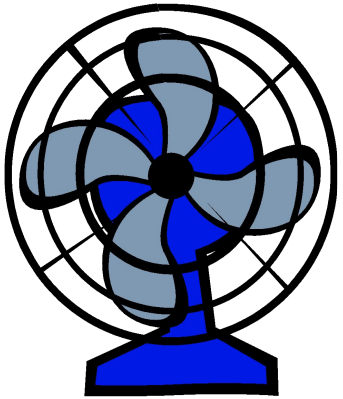
Objeto instanciado

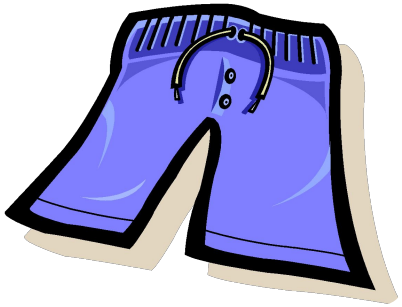
Conceito de orientação a objetos

Colocando em prática o que conceitos!

- Exercícios de abstração e modelagens de classes. Observe as figuras ao lado e gere a classe, isto é, Nome da classe, atributos da classe e métodos da classe.







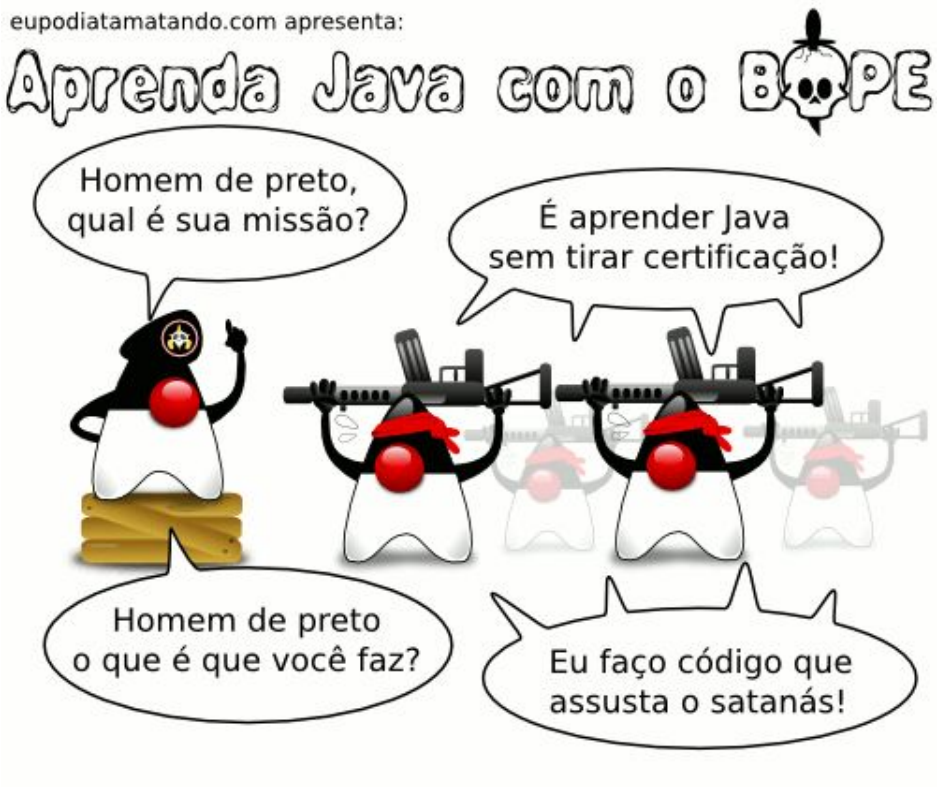
Java

Linguagem de programação

Sun Microsystems

Orientada a objetos

Java

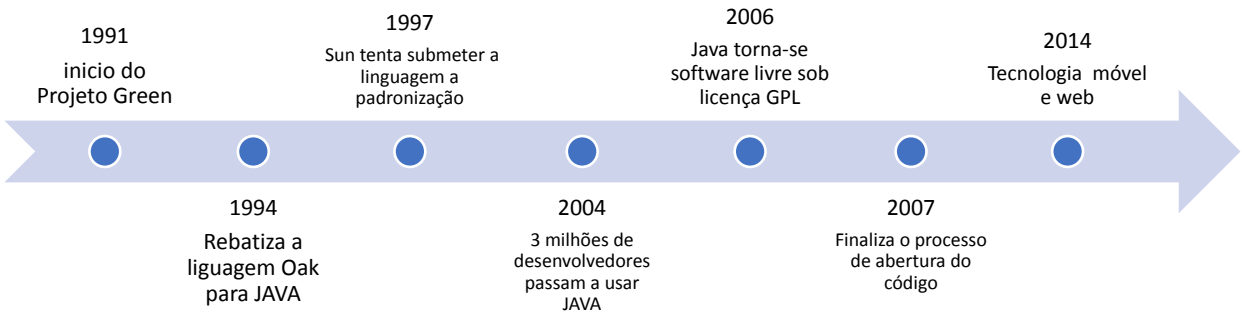

eupodiatamatando.com apresenta:
Aprenda Java com o BOPE
Homem de preto, qual é sua missão?
É aprender Java sem tirar certificação!
Homem de preto o que é que você faz?
Eu faço código que assusta o satanás!

O que é o java?

- Uma plataforma integral, com uma extensa biblioteca contendo uma grande quantidade de códigos reutilizáveis e um ambiente de execução que fornece serviços como:
  - Segurança;
  - Portabilidade para diferentes SO’s;
  - Coleta de lixo automático.
- Linguagem com sintaxe amigável e uma semântica compreensível e com uma vasta documentação.

História do Java

- Um pouco sobre Java:



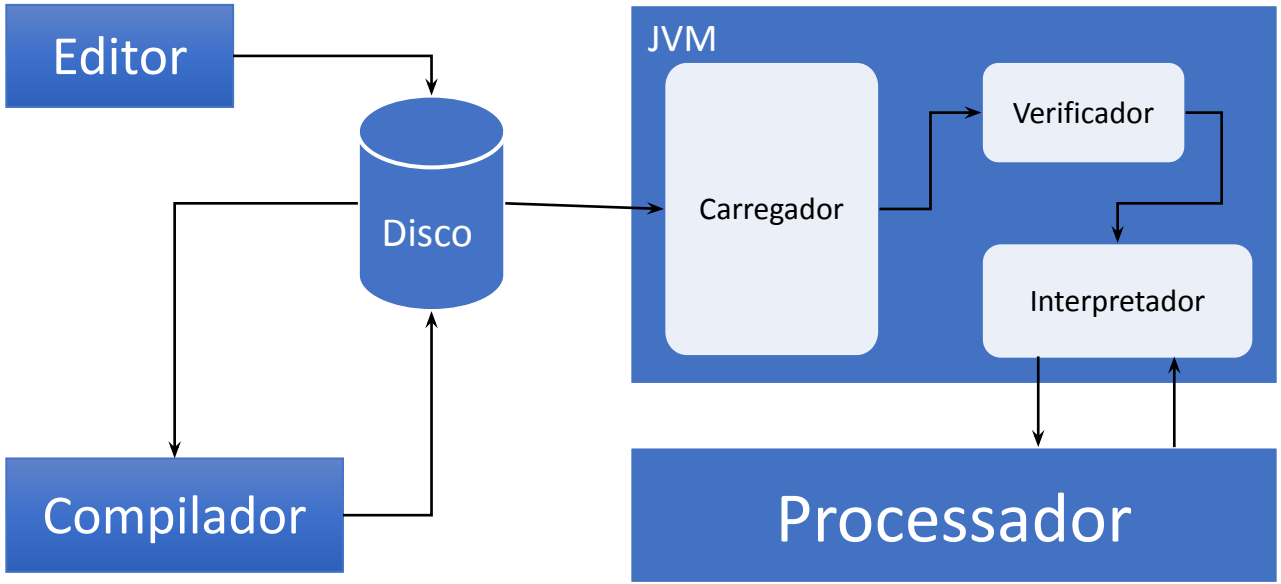
Comunicação de computadores com os equipamentos eletrodomésticos.

Portabilidade de software para múltiplas plataformas de dispositivos heterogêneos.

Características Importantes

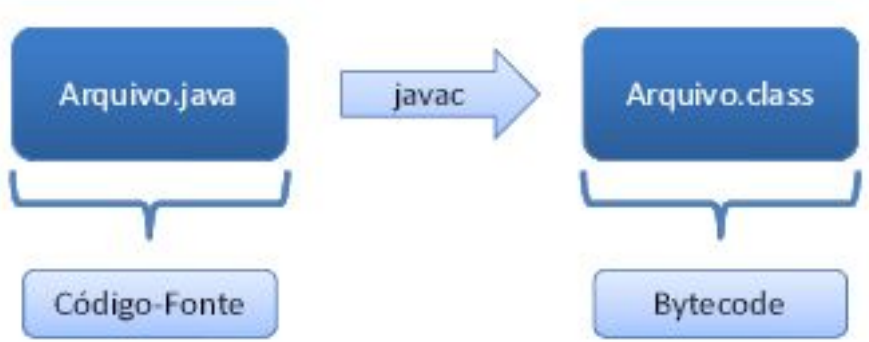
- A linguagem Java:
  - Orientada a Objetos;
  - Independência quanto a plataformas;
    - Bytecode;
    - Máquina Virtual Java (JVM – Java Virtual Machine);
  - Ausência de ponteiros;
    - Garbage collector;
  - Alto desempenho;
  - Segurança;
  - Multithreading;
    - Executa inúmeras threads concorrentes;

Ambiente Java

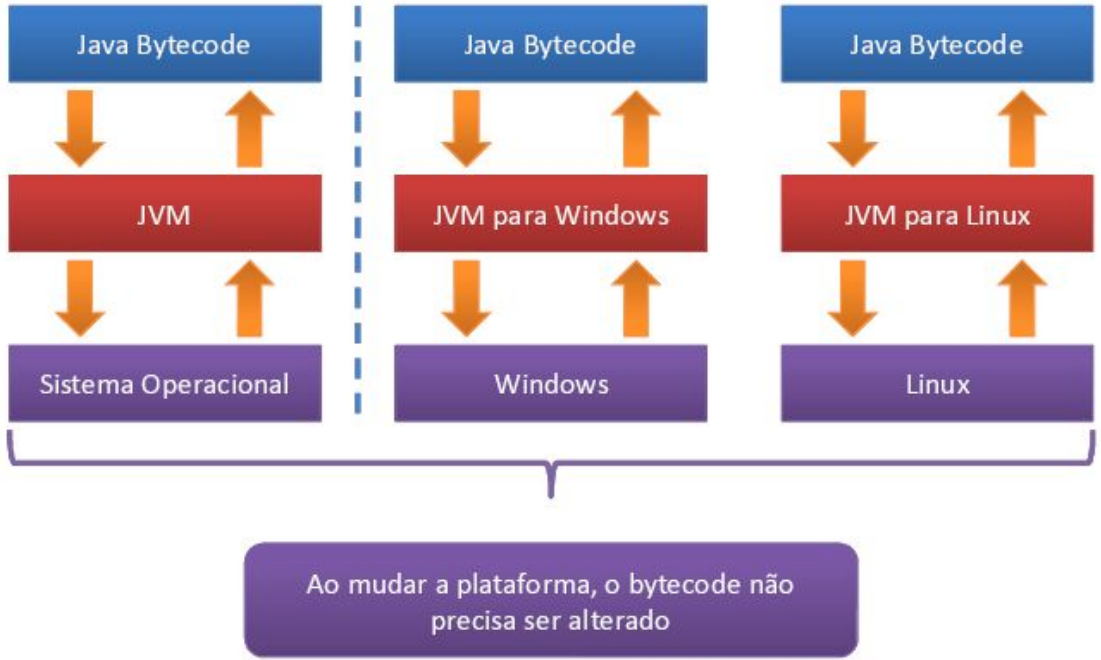


Bytecode

- O bytecode é uma linguagem entendida pela JVM
- A geração do bytecode é feita através da compilação do código Java



Máquina Virtual



Máquina Virtual Java (JVM)

- Independência de Plataforma (Portabilidade)
  - A aplicação roda sem nenhum envolvimento com o SO. Sempre conversando apenas com a JVM
- Garbage collection
- O JVM não entende código Java, mas sim um código de máquina específico. Esse código de máquina é gerado por um compilador Java, como o javac, e é conhecido por “bytecode”.

Princípio WORA
Write once, run anywhere

A JVM é uma Especificação

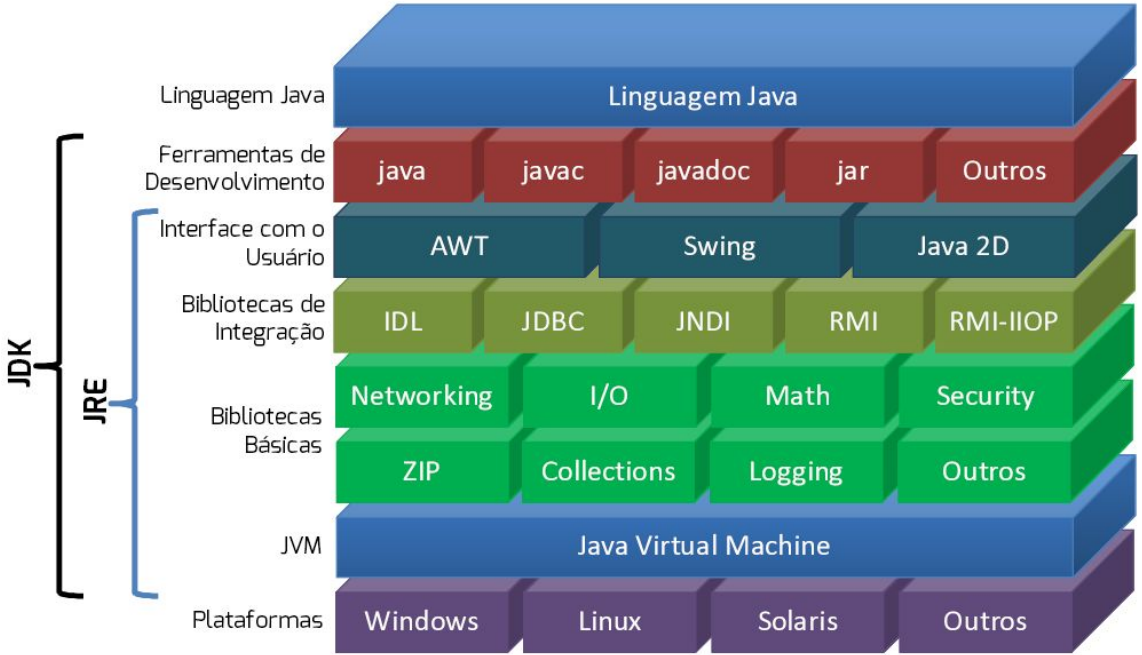
- Diversas empresas implementam a JVM
  - Oracle, IBM, etc.
- Como a JVM é uma especificação que deve ser seguida por quem a implementa, é possível trocar de JVM sem a necessidade de recompilar os códigos das aplicações

JVM? JRE? JDK? O que devo baixar?

- **JVM** = Java Virtual Machine.
- **JRE** = Java Runtime Environment, ambiente de execução Java, formado pela JVM e bibliotecas, tudo que você precisa para executar uma aplicação Java.
- **JDK** = Java Development Kit. Desenvolvedores necessitam para programar e precisam fazer o download do JDK do Java SE (Standard Edition). Ele é formado pela JRE somado a ferramentas, como o compilador.

Tanto o JRE e o JDK podem ser baixados do site http://java.sun.com, hoje gerenciado pela Oracle.

Elementos da JRE e JDK



Fonte: Oracle

Classes em Java

- A classes em java são salvas em arquivos .java;
  - Pode se definir várias classes em um arquivo, mas apenas um delas pode ser pública;
- Estrutura básica de uma classe:

```
qualificador class Nome_Da_Classe {
// atributos da classe Nome_Da_Classe
// métodos da classe
}
```

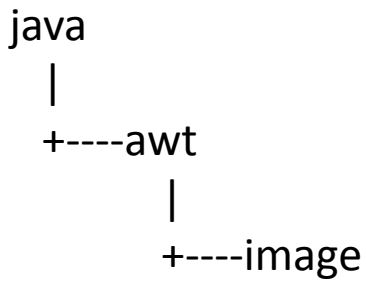
  - **Qualificadores**: são estruturas de controle;

Estruturas de controle

Qualificador	Significado
public	Indica que o conteúdo da classe pode ser utilizado tanto pela classe com por outra que faça referência ao objeto.
protected	Indica que a classe somente pode ser referenciada por métodos que estejam dentro do mesmo pacote, ou em membros da própria classe.
private	Indica acesso restrito a membros da mesma classe.
abstract	Indica que a classe possui métodos abstratos, ou que ela não implementou todos os métodos de uma interface que faça referencia a ela.
final	Esse qualificador faz com que a classe não possa servir de base para herança a outra classe.
strictfp	Indica que todos os valores utilizados nos métodos da classe serão transformados em valores de ponto flutuante, com valores temporários. Ou seja, internamente serão convertidos ou para float ou para Double, em operações com expressões.
static	Permite a utilização de campos estáticos mesmo que a classe seja declarada dentro de outra classe. Permite também que uma classe externa crie um objeto do tipo dessa classe.

Características da linguagem Java

- **Pacotes** (*Packages*): não são mais do que diretórios em que residem uma ou mais classes.



java.awt.Image

Características da linguagem Java

- **Pacotes** (*Packages*): não são mais do que diretórios em que residem uma ou mais classes.

java.awt.Image

```
java
  |
  +----awt
        |
        +----image
```

```
package
br.exemplo.toolbox
public class ImageButton {
```

Características da linguagem Java

- **import**: comando usado para importa as classes dentro de um pacote ou diretório.
  - Exemplos:

```
import java.awt.Button // importando a classe
Button
Import java.io.* // importando qualquer classe no
                 //pacote java.io
import aula3.exemplo.Aula1
```

Linguagem Java

- Exemplo de código:

```
package aula1;

public class Aula1 {
    public static void main(String[] args) {
        System.out.println("\nPrimeiro Programa");
    }
}
```

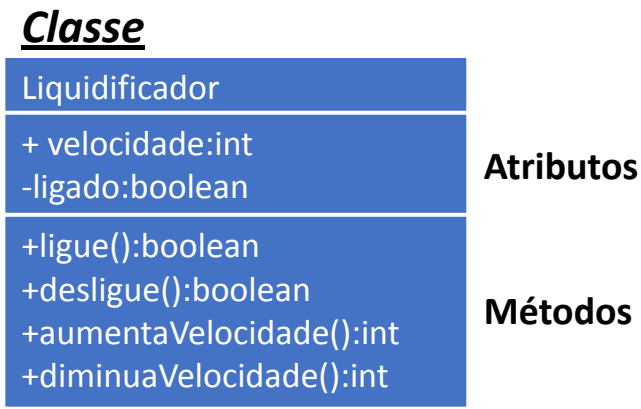
Linguagem Java

- Exemplo de código:

```
//Bola.java
public class Bola {
        // Atributos
        private float raio;
        public boolean furado;
        private nId;
}
```

Notação UML para classes

- UML (*Unified Modeling Language*): é um diagrama visual para modelagem de objetos;



Notação UML para classes

- Significados dos símbolos qualificadores de acesso na notação UML

Símbolo UML	Significado
+	O campo indicado é publico (public).
-	O campo indicado é privado (private).
#	O campo indicado é protegido (protected).
(em branco)	O campo indicado é padrão (default).

Boas práticas de programação

Convenção de código no Orientado a Objetos

- **Classes**: por convenção os nomes das classes começam com letra maiúscula e não se usa *underscores* para nomes compostos;
  - Ex.: MinhaClasse;
- **Métodos**: o nome de métodos podem ser verbos, ou misturas de maiúsculas e minúsculas. Porém, por convenção inicia-se por letra minúscula;
  - Ex.: getId();
- **Atributos**: por convenção inicia-se o nome de atributos por letra minúsculas. Os nomes pode conter misturar letras maiúsculas e minúsculas e usar *underscores* (no entanto, prática evitar o uso *underscores*);
  - Ex.: nomeCompleto;

Boas práticas de programação

- **Variáveis**: os nomes podem ser maiúsculos, minúsculos ou mistos e podem conter *underscores*.
  - Ex.: acc;
- **Comentários**: Use-se comentários para explicar segmentos de código que não são óbvios.
  - Ex.:

```
\\ calcula a média tomando-se apenas as 3
\\maiores notas das 4
```

- **Endentação**: usa-se para alinhar os códigos em suas respectivos blocos de estrutura de controle;
  - Ex.:

```
if (...) {
    x = b + c;
    a =+ x;
}
```

Compilando o Primeiro Código

- Vamos para o nosso primeiro código! O programa que imprime uma linha simples.

- Para mostrar uma linha, podemos fazer:

```
System.out.println("Minha primeira aplicação Java!");
```

O mínimo que precisaríamos escrever seria:

```
1 class MeuPrograma {
2     public static void main(String[] args) {
3
4         // miolo do programa começa aqui!
5         System.out.println("Minha primeira aplicação Java!!");
6         // fim do miolo do programa
7
8     }
9 }
```

Para saber mais: Como é o Bytecode?

```
javap -c MeuPrograma
```

E a saída:

```
MeuPrograma();
  Code:
   0:   aload_0
   1:   invokespecial   #1; //Method java/lang/Object."<init>":()V
   4:   return

public static void main(java.lang.String[]);
    Code:
    0:    getstatic    #2; //Field java/lang/System.out:Ljava/io/PrintStream;
    3:   ldc     #3; //String Minha primeira aplicação Java!!
    5:   invokevirtual   #4; //Method java/io/PrintStream.println:
                                (Ljava/lang/String;)V
    8:   return
```

Possibilidade de erros

- **Erros de Compilação**: Erros de digitação e de uso da sintaxe da linguagem.
- **Erros de Link-Edição**: Erro no uso de bibliotecas de subprogramas necessárias ao programa principal.
- **Erros de Execução**: Erro na lógica do programa (algoritmo).

O que pode dar Errado?

Código:

```
1 class X {
2     public static void main (String[] args) {
3         System.out.println("Falta ponto e vírgula")
4     }
5 }
```

Erro:

```
X.java:4: ';' expected
        }
        ^
1 error
```

O que pode dar Errado?

Exception in thread "main" java.lang.NoSuchMethodError: main

```
class X {
    public void main (String[] args) {
        System.out.println("Faltou o static, tente executar!");
    }
}
```

Main method not public.

```
class X {
    static void main (String[] args) {
        System.out.println("Faltou o public");
    }
}
```

Comando para ler uma variável:

```
1   package testarhabilidades;
2
3   import java.util.Scanner;
4
5   public class Main {
6
7       public static void main(String[] args) {
8           Scanner input = new Scanner(System.in);
9           int n1,n2, soma;
10          System.out.println("Entre com o primeiro valor: ");
11          n1 = input.nextInt();
12          System.out.println("Entre com o segundo valor: ");
13          n2 = input.nextInt();
14          soma = n1 + n2;
15          System.out.printf("A soma dos números é: "+soma);
16      }
17
18  }
```

Exercício em Sala

1. Crie um aplicativo em java para calcular a média aritmética de duas notas.
2. Crie um aplicativo em java para calcular a área de uma circunferência.
3. Crie um aplicativo em java para calcular o Índice de Massa Corpórea (IMC) de um indivíduo. **IMC = Peso/(Altura)^2**

Links interessantes:

- Tutorial de introdução à linguagem:
  - *http://docs.oracle.com/javase/tutorial/getStarted/index.html*
- Link do Javadoc oficial do Java 8. Ele detalha todas as classes, interfaces, métodos, etc. da API do Java:
  - *http://docs.oracle.com/javase/8/docs/api/*
- Preparação do ambiente para desenvolvimento em Java.
  - *http://www.devmedia.com.br/preparacao-do-ambiente-para-desenvolvimento-em-java/25188*
- Download Eclipse IDE
  - *https://eclipse.org/downloads/*
- Download NetBeans IDE
  - *https://netbeans.org/downloads/*