

The background is a dark navy blue. On the left side, there are several parallel white lines that form a corner-like shape, extending from the top left towards the center. In the bottom right corner, there are several parallel teal lines that form a diagonal shape, extending from the bottom right towards the center. The word 'JAVA' is written in a large, white, sans-serif font, positioned in the upper left quadrant. Below it, the words 'LINGUAGEM DE PROGRAMAÇÃO' are written in a smaller, teal, sans-serif font, centered horizontally.

# JAVA

LINGUAGEM DE PROGRAMAÇÃO

# Classe

LINGUAGEM DE PROGRAMAÇÃO  
JAVA

# Classe

Classe é a definição de um conjunto de objetos, seres ou coisas do mundo real que possuem características em comum.

Todo programa em Java está dentro de uma estrutura lógica de código chamada classe.

# Classe

Cada declaração de classe contém a palavra-chave class seguida imediatamente pelo nome da classe


```
public class Student
```

# Classe

```
public class Hello {  
}
```

Toda classe é um arquivo Java, e este possui a extensão **.java**; O arquivo deve possuir o mesmo nome da classe, respeitando letras maiúsculas ou minúsculas.

Assim, nossa classe **Hello** está armazenada fisicamente num arquivo **Hello.java**.



As classes são  
compostas de  
**atributos e métodos.**

# Classe

Classes possuem atributos que são **variáveis** criadas dentro da classe, mas fora do método, é conhecida como variável de instância.

```
public class Student {  
    // variáveis de instância  
    String nome;  
    Int idade;  
}
```

# Modificadores de atributos

**public** - o campo é acessível de todas as classes.

**private** - o campo é acessível somente dentro de sua própria classe.

**protected** - o campo é acessível às classes do mesmo pacote ou através de herança, seus membros herdados não são acessíveis a outras classes fora do pacote em que foram declarados.



# Modificadores de atributos

**final** - *campo não pode ser alterado depois que já tenha sido atribuído um valor.*

**static** - *campo será a mesma em todas as instâncias e quando seu conteúdo é modificado numa das instâncias, a modificação ocorre em todas as demais.*

Modificador	Em uma classe	Em um método	Em um atributo
private	Não aplicável	Acesso pela classe	Acesso pela classe
protected	Não aplicável	Acesso pelo pacote e subclasses	Acesso pelo pacote e subclasses
default	Somente pacote	Acesso pelo pacote	Acesso pelo pacote
public	Acesso total	Acesso total	Acesso total
abstract	Não instância	Deve ser sobrescrito	Não aplicável
final	Sem herança	Não pode ser sobrescrito	Constante
static	Não aplicável	Acesso pela classe	Acesso pela classe
native	Não aplicável	Indica código nativo	Não aplicável
transient	Não aplicável	Não aplicável	Não serializavel
synchronized	Não aplicável	Sem acesso simultâneo	Não aplicável

# Classe

Algumas observações de boas práticas de atributos:

- As declarações dos atributos são feitas sempre na primeira parte do código da classe.
- Declare um atributo por linha, mesmo que ele seja do mesmo tipo.
- Declare atributo como `private`; assim, somente a própria classe pode manipulá-lo; esse é o conceito é muito importante na Orientação a Objeto.

# Classe

Cada classe que você declara tem como fornecer um **construtor** com parâmetros que podem ser utilizados para inicializar um objeto.

# Classe

**ATENÇÃO**: O compilador fornece automaticamente um construtor padrão sem argumentos para qualquer classe sem construtores.

# Classe

```
public class Student {
```

```
    //construtor
```

```
    public Student() {  
    }  
}
```

# Classe

Através dos construtores, é possível inicializar os atributos da classe com parâmetros do construtor.

```
public Student(String new_name)
{
    this.name = new_name;
}
```

# Classe

```
public class Student {  
    // variável de instância  
    private String name;  
    // construtor com parâmetro.  
    public Student(String new_name)  
    {  
        this.name = new_name;  
    }  
}
```



# Classe

## Importante:

- Um construtor deve ter o **mesmo nome** que a classe;
- Os parâmetros de método são **variáveis locais**;
- A palavra-chave **new** solicita memória do sistema para armazenar o novo objeto;
- A chamada é indicada pelos parênteses após o nome da classe;
- **Construtores não podem retornar valores!!!**

# Atividade:

1. Crie uma classe **Televisor**.

Essa classe deve possuir os três atributos e um construtor:

*canal // inicia em 1 e vai até 16*

*volume // inicia em 0 e vai até 10*

*ligado // inicia em desligado ou false*

# Métodos de Classe

LINGUAGEM DE PROGRAMAÇÃO  
JAVA

# Métodos

Métodos são os comportamentos ou funções de um objeto do mundo real na forma como ele é tratado no mundo computacional.

# Métodos

## Nomeando um método

Embora um nome de método possa ser qualquer identificador legal, as convenções de código restringem os nomes de método.

**Por convenção**, os nomes dos métodos devem ser um verbo em letras minúsculas ou um nome de várias palavras que começa com um verbo em letras minúsculas, seguido por adjetivos, substantivos, etc.

# Métodos – Declaração

**Modificadores** — *como public, private, protected, etc;*

**Tipo de retorno** — *o tipo de dados do valor retornado pelo método ou void (Se o método não retornar um valor).*

**Nome do método** — *as regras para nomes de campo também se aplicam a nomes de método;*

**Parâmetros** — *uma lista delimitada por vírgulas de parâmetros de entrada, precedida por seus tipos de dados. Se não houver parâmetros, você deve usar parênteses vazios*

# Métodos – Sem retorno

Esse tipo de método executa apenas o código que tem dentro dele, não retornando nenhum resultado, sendo identificados com a palavra-chave **void**.

```
public void escrever() {  
    System.out.println("Método sem Retorno");  
}
```

# Métodos – Com retorno

Esses métodos que não possuem a palavra-chave **void** incorporada na declaração, mas sim um tipo de dados, apresentam em seu corpo a palavra reservada **return**, que informa que o método terá que retornar o mesmo tipo de dados com o qual foi declarado.

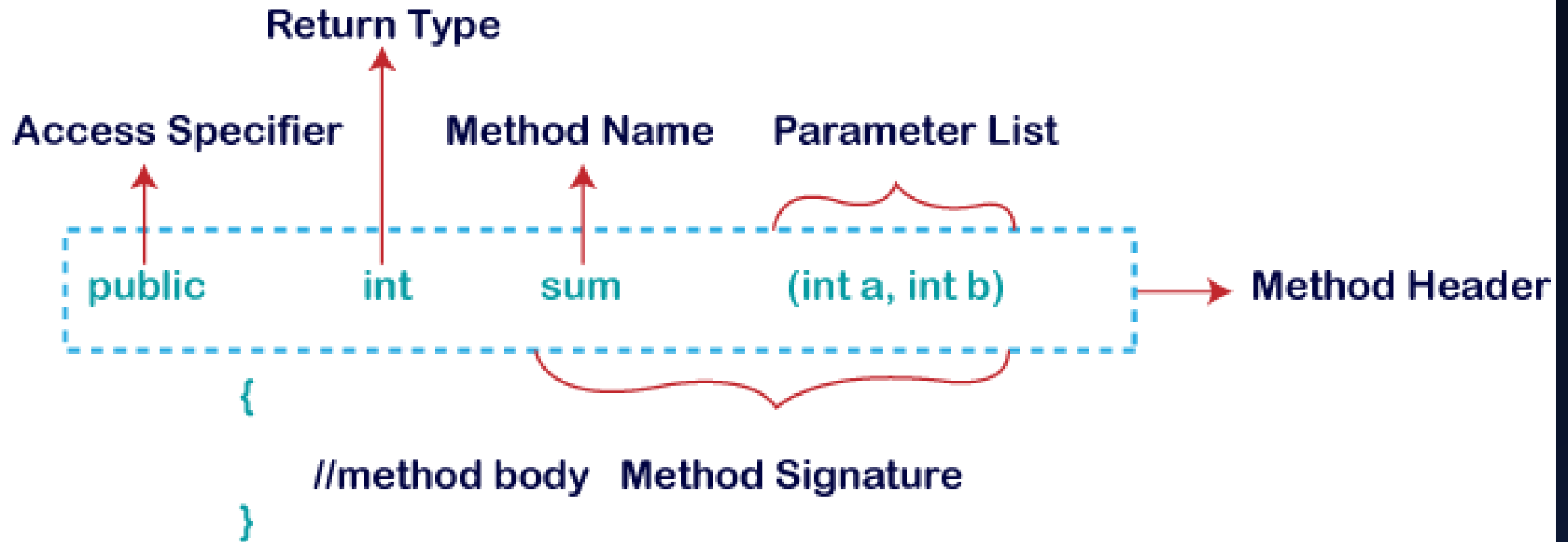


# Métodos – Com retorno

```
String nome = "Maicon Santos";
```

```
public String retornaNome(){  
    return nome;  
}
```

# Métodos – Declaração



# Atividade:

2.Dentro da classe **Televisor**. Crie os seguintes métodos:

<code>aumentarVolume()</code>	<code>// Aumenta em 1 o volume</code>
<code>reduzirVolume()</code>	<code>// Diminui em 1 o volume</code>
<code>subirCanal()</code>	<code>// Aumenta em 1 o canal</code>
<code>descerCanal()</code>	<code>// Diminui em 1 o canal</code>
<code>ligarTelevisor()</code>	<code>// Liga a televisão</code>
<code>desligarTelevisor()</code>	<code>// Desliga a televisão</code>
<code>mostrarStatus()</code>	<code>// Exibe o canal, Exibe o volume, Status tv;</code>



**PRATICANDO JAVA....**



# Obrigado!

INTRODUÇÃO A LINGUAGEM DE  
PROGRAMAÇÃO JAVA

