

# Orientação a objetos

- **Orientação a objetos**
  - Modelar objetos do mundo real.
  - Comportamento dos objetos do mundo real.
- **Objetos possuem as seguintes características:**
  - Identidade
  - Estado
  - Comportamento

- **Orientação a objetos**
  - Alta coesão
  - Baixo acoplamento
  - Abstração
  - Encapsulamento
  - Herança
  - Polimorfismo
  - Sobrecarga (Overloading)
  - Sobreposição (Overriding)

- Abstração

- O que um objeto é e faz antes da implementação.
- Entendimento do problema a ser resolvido.

- Encapsulamento

- Aspecto externo e interno de um objeto.
- Quais serão acessíveis por outros objetos e quais ficarão escondidos.

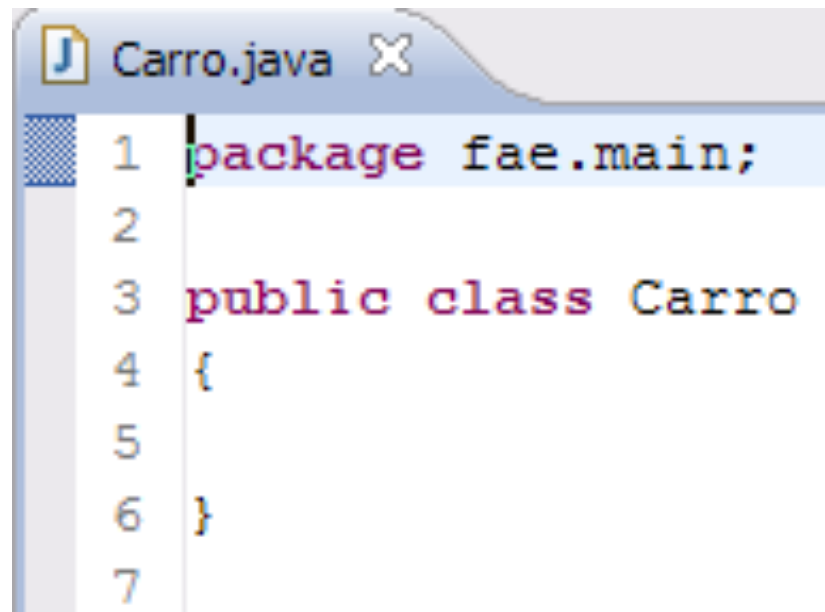
Exemplo:  
Controle da TV



- **Sistemas orientados a objetos**
  - Modela objetos (classes) do mundo real.
  - Baixo acoplamento.
  - Alto nível de coesão dos objetos(classes).
  - Testável
  - Fácil manutenção

- Sistemas orientado a objetos
  - Qual é a base?

Classe



```
Carro.java X
1 package fae.main;
2
3 public class Carro
4 {
5
6 }
7
```

- Para criar um objeto é necessário uma classe?

Não

Linguagens baseadas em protótipos:

- Tcl/tk
- JavaScript
- ActionScript/Flex
- Perl
- Lua
- OpenLaszlo

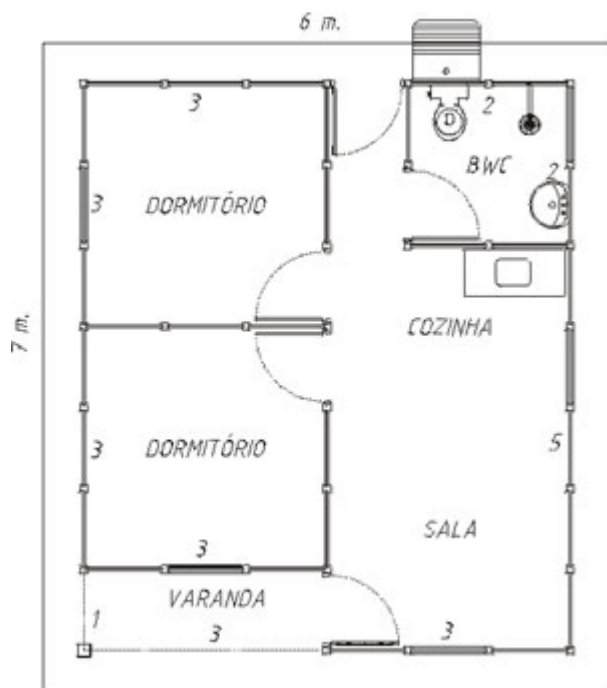




- O que é uma Classe e o que é um Objeto???

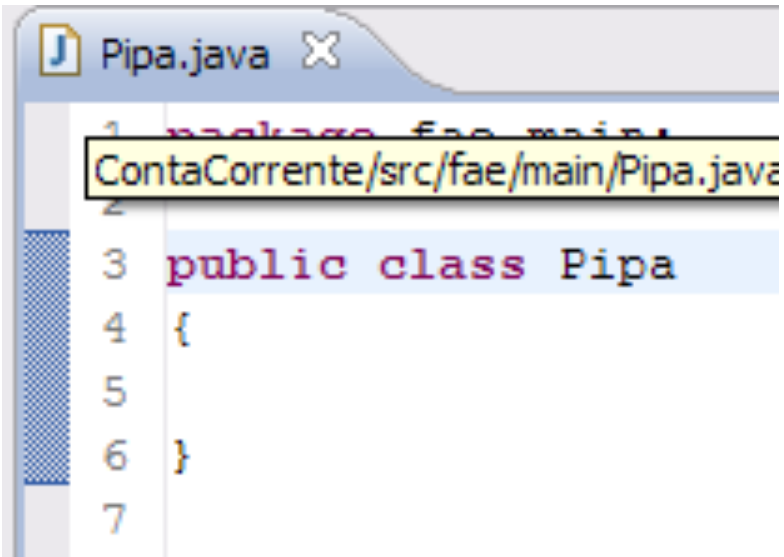
Classe é o gabarito, é como a planta de uma casa.

Objeto é a realização deste gabarito, casas feitas a partir desta planta.



- Classe - Uma **classe** é uma estrutura que abstrai um conjunto de objetos com características similares.
- Um **objeto**, em programação orientada a objetos, é uma instância de uma classe.

- Como fazer para criar um Objeto?



```
Pipa.java X
1 package fae.main;
2
3 public class Pipa
4 {
5
6 }
7
```

NEW

Pipa pipa = new Pipa();



Uma instância é uma referência (similar a ponteiros).  
`Pipa pipa = new Pipa();`



or do objeto.

`Pipa pipa = new Pipa();`

`pipa = null;`

- Objetos possuem características e comportamentos.
- Características – Atributos
- Comportamento - Métodos



```
Pessoa.java X
1 package fae.edu.pessoa;
2
3 import java.io.Serializable;
4
5 public class Pessoa implements Serializable
6 {
7     private static final long serialVersionUID = 1L;
8     private Long id;
9     private String nome;
10    private String login;
11    private String senha;
12
13    public Pessoa()
14    {
15    }
16
17    public Pessoa(String nome)
18    {
19        this.nome = nome;
20    }
21
22    public Long getId() {
23        return id;
24    }
25
26    public void setId(Long id) {
27        this.id = id;
28    }
```

Public  
Private  
Protected  
Static

# Construtores

- Métodos construtores são executados na criação do objeto.
- Servem para inicializar atributos da classe com algum valor.



- Como se faz para chamar um método?

```
Pessoa.java X
1 package fae.edu.pessoa;
2
3 import java.io.Serializable;
4
5 public class Pessoa implements Serializable
6 {
7     private static final long serialVersionUID = 1L;
8     private Long id;
9     private String nome;
10    private String login;
11    private String senha;
12
13    public Pessoa()
14    {
15    }
16
17    public Pessoa(String nome)
18    {
19        this.nome = nome;
20    }
21
22    public Long getId() {
23        return id;
24    }
25
26    public void setId(Long id) {
27        this.id = id;
28    }
```

Pessoa pessoa = new Pessoa();

pessoa.setId(20);





```
Carro.java X
1 package fae.carro;
2
3 public class Carro
4 {
5
6 }
7
```

- Atributos:
  - Devem ser definidos no corpo da classe;
  - Devem ter o método de acesso;
  - Nome do atributo;



```
Carro.java X
1 package fae.carro;
2
3 public class Carro
4 {
5     private String nome;
6     private String marca;
7     private String modelo;
8     private Float potencia;
9 }
10
```

- Métodos
  - Definidos no corpo da classe
  - Modo de acesso (visibilidade)
  - Nome
  - Retorno
  - Parâmetros



```
Carro.java X
1 package fae.carro;
2
3 public class Carro
4 {
5     private String nome;
6     private String marca;
7     private String modelo;
8     private Float potencia;
9
10    public void ligar()
11    {
12        System.out.println("Ligando carro");
13    }
14
15    public void desligar()
16    {
17        System.out.println("Desligando carro");
18    }
19 }
20
```