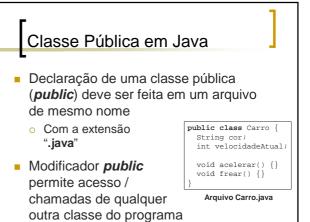
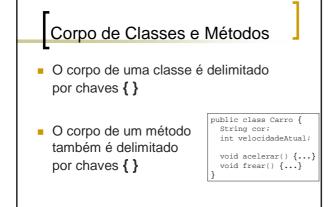


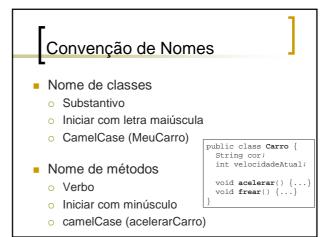
Implementando um Projeto

- Assim como em outras engenharias, o projeto não é suficiente
 - Ninguém pode dirigir o desenho de um carro
 - Ninguém pode executar o projeto de um sistema
 - o Por enquanto...









Assinatura de Métodos

- Modificador + Tipo de Retorno +
 Nome do Método + Lista de Parâmetros
 - o **Modificador**: public | private | protected | ...
 - o Tipo de Retorno: void | int | String | ...
 - o Nome: acelerar | frear | ...
 - Parâmetros: delimitado por parênteses

```
public class Carro {
  private String cor;
  private int velocidadeAtual;

public void acelerar() {}
  public void frear() {}
```

Implementando Métodos

```
public class Carro {
   String cor;
   int velocidadeAtual = 0;

public void acelerar() {
    velocidadeAtual = velocidadeAtual + 1;
   }

public void frear() {
    velocidadeAtual = velocidadeAtual - 1;
   }
}
```

O Método Main

```
public class TesteCarro {
   public static void main(String[] args) {
     ...
   }
}
```

É uma boa prática colocar o método main em uma classe separada



O Método Main

```
public class TesteCarro {
  public static void main(String[] args) {
    ...
}
```

O método **main** é automaticamente chamado pela máquina virtual Java

Implementando Métodos

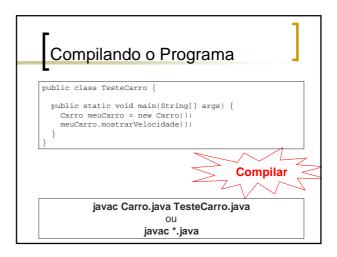
```
public class Carro {
   String cor;
   int velocidadeAtual = 0;

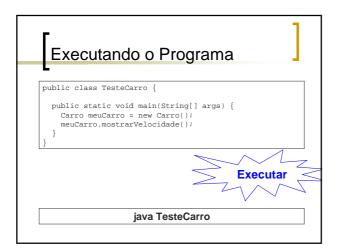
public void acelerar() {
    velocidadeAtual = velocidadeAtual + 1;
   }

public void frear() {
    velocidadeAtual = velocidadeAtual - 1;
   }

public void mostrarVelocidade() {
    System.out.println("Minha velocidade & " +
        velocidadeAtual);
   }
}
```

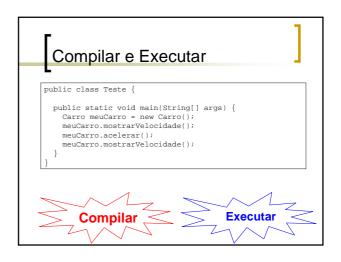
public class TesteCarro { public static void main(String[] args) { Carro meuCarro = new Carro(); meuCarro.mostrarVelocidade(); } }





```
public class TesteCarro {
  public static void main(String[] args) {
    Carro meuCarro = new Carro();
    meuCarro.mostrarVelocidade();
  }
}
Minha velocidade é 0
```

```
public class TesteCarro {
  public static void main(String[] args) {
    Carro meuCarro = new Carro();
    meuCarro.mostrarVelocidade();
    meuCarro.acelerar();
    meuCarro.mostrarVelocidade();
}
```







```
public class Carro {
    String cor;
    int velocidadeAtual = 0;
    public void acelerar() {
        velocidadeAtual = velocidadeAtual + 1;
    }
    public void frear() {
        velocidadeAtual = velocidadeAtual - 1;
    }
    public void mostrarVelocidade() {
        System.out.println("Minha velocidade é " + velocidadeAtual);
    }
}
```

```
public class Carro {
    String cor;
    int velocidadeAtual = 0;
    public void acelerar() {...}
    public void frear() {...}
    public void mostrarVelocidade() {...}

public void definirCor(string novaCor) {
    cor = novaCor;
    }
    public string retornarCor() {
        return cor;
    }
}
```

```
Métodos para Cor

public class Carro {
    String cor;
    int velocidadeAtual = 0;
    public void acelerar() {...}
    public void frear() {...}
    public void mostrarVelocidade() {...}

public void definirCor(String novaCor) {
    cor = novaCor;
    }

public String retornarCor() {
    return cor;
    }
}
Método
"set"

Método
"get"
```

```
public class Carro {
   String cor;
   int velocidadeAtual = 0;
   public void acelerar() {...}
   public void frear() {...}
   public void definirCor(String novaCor) {
      cor = novaCor;
   }
   public String retornarCor() {
      return cor;
   }
}

Método com um
   parâmetro
   public void definirCor(String novaCor) {
      cor = novaCor;
   }
```

```
O método definirCor

| public void definirCor(String novaCor) {
| cor = novaCor;
| }

Não retorna nada quando é chamado
| Tipo de retorno: void

| Recebe um parâmetro
| Parâmetro: String novaCor
```

```
O Método retornarCor

| public String retornarCor() {
| return cor; | }

Retorna um objeto
| Tipo de retorno: String

Não recebe nenhum parâmetro
| Parâmetro: vazio
```

```
public class TesteCarro {
  public static void main(String[] args) {
    Carro meuCarro = new Carro();
    meuCarro.definirCor("Vermelho");
  }
}
```

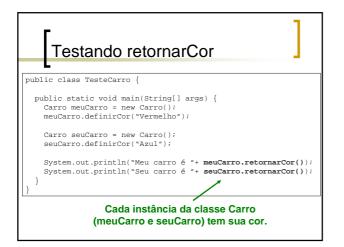
```
Passagem de Parâmetros

public class TesteCarro {
  public static void main(String[] args) {
    Carro meuCarro = new Carro();
    meuCarro.definirCor("Vermelho");
  }
}

Passando um
  parâmetro (String) para
  o método definirCor
```

```
public class TesteCarro {
  public static void main(String[] args) {
    Carro meuCarro = new Carro();
    meuCarro.definirCor("Vermelho");
    Carro seuCarro = new Carro();
    seuCarro.definirCor("Azul");
  }
  Instância
  "seuCarro" tem
    cor Azul
Instância

"cor Vermelho
```



Compilando e Executando public class TesteCarro { public static void main(String[] args) { Carro meuCarro = new Carro(); meuCarro.definirCor("Vermelho"); Carro seuCarro = new Carro(); seuCarro.definirCor("Azul"); System.out.println("Meu carro é "+ meuCarro.retornarCor()); System.out.println("Seu carro é "+ seuCarro.retornarCor()); } Compilar Executar

public class TesteCarro { public static void main(String[] args) { Carro meuCarro = new Carro(); meuCarro.definirCor("Vermelho"); Carro seuCarro = new Carro(); seuCarro.definirCor("Azul"); System.out.println("Meu carro é "+ meuCarro.retornarCor()); System.out.println("Seu carro é "+ seuCarro.retornarCor()); } Meu carro é Vermelho Seu carro é Azul

Bibliografia

- DEITEL, H. M.; DEITEL P. J. Java: Como Programar, 8a. Edição. Pearson, 2010.
 - o Seções 1.5 a 1.10
 - o Capítulo 3
- BOOCH, G., RUMBAUGH, J., JACOBSON, I. UML, Guia do Usuário. Rio de Janeiro: Campus, 2000.