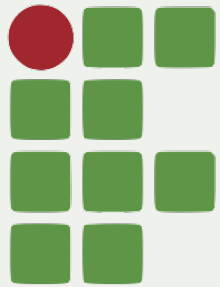
Palavras reservadas

POO29004 – Engenharia de Telecomunicações

Prof. Emerson Ribeiro de Mello

http://docente.ifsc.edu.br/mello/poo

13 de março de 2020



INSTITUTO FEDERAL
Santa Catarina
Câmpus São José

Palavra reservada: `this`

- `this` é uma referência para o próprio objeto e pode ser usado para garantir que estará referenciando os membros de uma classe e não uma variável local

Palavra reservada: `this`

- `this` é uma referência para o próprio objeto e pode ser usado para garantir que estará referenciando os membros de uma classe e não uma variável local

```
1  public class Pessoa{
2      private String nome;
3      private int idade;
4
5      public void setNome(String nome){
6          this.nome = nome; // o uso do this é obrigatório
7      }
8
9      public void setIdade(int i){
10         // o uso do this não é obrigatório para o entendimento
11         idade = i;
12     }
13 }
```


Palavra reservada: `this`: Invocar outros métodos da classe

```
14 public class Retangulo{
15     private int x, y;
16     private int largura, altura;
17
18     public Retangulo(){
19         this(0,0,10,10);
20     }
21     public Retangulo(int x, int y, int l, int a){
22         this.x = x; this.y = y; this.largura = l; this.altura = a;
23     }
24
25     public void setX(int x){
26        if ( x >= 0)
27           this.x = x;
28     }
29
30     public void setXY(int x, int y){
31         this.setX(x); this.setY(y);
32     }
33 }
```


Palavra reservada: `this` – argumento passado para outro objeto

```
34 public class AreaDesenho{
35     private Janela janela;
36
37     public AreaDesenho(Janela j){
38         this.janela = j;
39     }
40     public void desenhar(String t){
41         ...
42     }
43 }
```

```
44 public class Janela{
45     public void desenharTexto(){
46         AreaDesenho area = new AreaDesenho(this);
47         area.desenhar("texto");
48     }
49 }
```

Membros estáticos

Atributos não estáticos

Cada instância da classe terá uma cópia distinta desse atributo

```
62 public class Celular{
63     private int total;
64
65     public Celular(){
66         this.total++;
67     }
68
69     public int getTotal(){
70         return this.total;
71     }
72 }
```

```
73 Celular b = new Celular();
74 Celular c = new Celular();
75 int r = b.getTotal();
76 int t = c.getTotal();
```

- Qual será o valor de `r` e `t`?

Atributos não estáticos

Cada instância da classe terá uma cópia distinta desse atributo

```
62 public class Celular{
63     private int total;
64
65     public Celular(){
66         this.total++;
67     }
68
69     public int getTotal(){
70         return this.total;
71     }
72 }
```

```
73 Celular b = new Celular();
74 Celular c = new Celular();
75 int r = b.getTotal();
76 int t = c.getTotal();
```

- Qual será o valor de `r` e `t`?
  - r = 1
  - t = 1

Membros de classe estáticos

Membro de classe que pertença a classe e não a uma instância específica da classe

- Método estático não pode acessar membros não estáticos

```
77 public class Celular{
78   private static int total;
79
80   public Celular(){
81     total++;
82   }
83
84   public static int getTotal(){
85     return total;
86     }
87 }
```

```
88 Celular b = new Celular();
89 Celular c = new Celular();
90 int r = b.getTotal();
91 int t = c.getTotal();
92 int h = Celular.getTotal();
93 System.out.print(r+","+t+","+h);
```

- Qual será o valor de `r`, `t` e `h`?

Membro de classe que pertença a classe e não a uma instância específica da classe

- Método estático não pode acessar membros não estáticos

```
77 public class Celular{
78   private static int total;
79
80   public Celular(){
81     total++;
82   }
83
84   public static int getTotal(){
85     return total;
86     }
87 }
```

```
88 Celular b = new Celular();
89 Celular c = new Celular();
90 int r = b.getTotal();
91 int t = c.getTotal();
92 int h = Celular.getTotal();
93 System.out.print(r+","+t+","+h);
```

- Qual será o valor de `r`, `t` e `h`?
  - r = 2
  - t = 2
  - h = 2

Membros de classe estáticos

```
94  public class Celular {
95      private static int total = 0;
96      private int creditos = 0;
97
98      public Celular(int creditos){
99          total++; this.creditos = creditos;
100     }
101     // OK
102     public int getCreditos() { // não estático
103         return creditos; // não estático
104     }
105     // OK
106     public int getTotal(){ //não estático
107         return total; // estático
108     }
109     // ERRO DE SINTAXE
110     public static int retornaCreditos(){ //estático
111         return creditos; // não estático
112     }
113 }
```


Modificador **final** – para definir constantes

Pode ser usado em atributos, métodos ou em variáveis locais

- Uma variável após ter recebido um valor, esse não poderá ser alterado
- Métodos não poderão ser sobrescritos (conceito de herança)

```
114 public class Celular{
115     private final String FREQUENCIA = "1800";
116     private final int SERIAL;
117 
118     public Celular(int s){
119         this.SERIAL = s;
120     }
121 
122     public final void iniciarChamada(){
123     // ....
124     }
125 }
```

Exercício: classe utilitária `java.lang.Math`

http://docs.oracle.com/javase/8/docs/api/java/lang/Math.html

1. Para obter a raiz quadrada do número 4, basta:
   ```
   double d = Math.sqrt(4);
   ```
2. `Math.PI` é uma constante que contém o valor aproximado de PI. Para imprimir este valor, basta:
   ```
   System.out.println(Math.PI);
   ```

Responda

Dos conceitos apresentados nesta aula, quais deles a classe `java.lang.Math` faz uso?

Material de referência

Material de referência: Classe `StringBuilder`

```
126
127 String s = "Engenharia";
128
129 // String são imutáveis
130 s += " de Telecomunicações"; // JVM cria novos objetos na memória
131
132 // Classe mais adequada quando se deseja concatenar strings
133
134 StringBuilder sb = new StringBuilder("Engenharia");
135 sb.append(" de Telecomunicações");
136 String res = sb.toString();
```