

Introdução a Orientação a Objetos

Reforço

Vinicius Takeo Friedrich Kuwaki

Universidade do Estado de Santa Catarina

Seções

Introdução

Projetos

Pacotes

Classes

Atributos

Encapsulamento

Métodos

Getters e Setters

This

Construtor

Estáticos

toString

equals

- Alan Kay;
- Linguagem Small Talk;
- Inspiração Biológica;

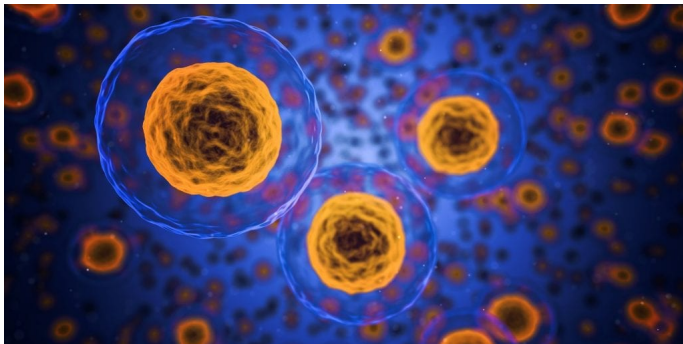


Figura 1: Aglutinado de células

- Objetivos:
 - Reutilização de código;
 - Produtividade;
 - ...

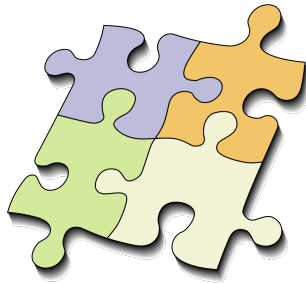


Figura 2: Quebra-cabeça.

Seções

Introdução

Projetos

Pacotes

Classes

Atributos

Encapsulamento

Métodos

Getters e Setters

This

Construtor

Estáticos

toString

equals

- Contém os códigos-fontes, bibliotecas utilizadas, etc;
- Comum o uso de IDE's:
 - VSCode;
 - Eclipse;
 - Netbeans;

Seções

Introdução

Projetos

Pacotes

Classes

Atributos

Encapsulamento

Métodos

Getters e Setters

This

Construtor

Estáticos

toString

equals

- Significado semântico;
- Organizar as classes de acordo com suas funcionalidades;
- Basicamente: pasta com arquivos;

- Objetivos:
 - Separar de acordo com o que a classe faz;
 - Padrão utilizado na disciplina:
 - Pseudo-MVC;

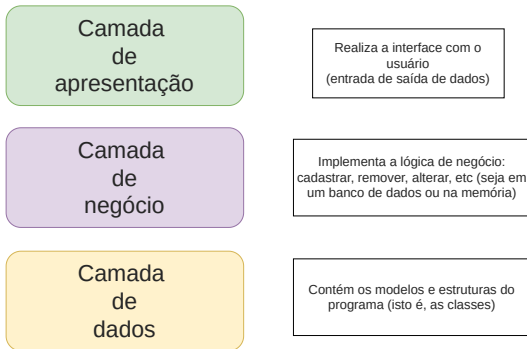


Figura 3: Estrutura das Camadas.

Seções

Introdução

Projetos

Pacotes

Classes

- Atributos

- Encapsulamento

- Métodos

- Getters e Setters

- This

- Construtor

- Estáticos

- toString

- equals

- Define um novo tipo de dado;
- Agrupamento de tipos primitivos (atributos) e métodos (funções);
- Classes x Tipos Primitivos:
 - int: primitivo;
 - float: primitivo;
 - String: classe;
 - Integer: classe;

- Classes: analogia a structs em C:

```
typedef struct pessoa{  
    char nome[20];  
    int idade;  
}Pessoa;
```

- Diferença: não é possível definir métodos!

```
void main(){  
    Pessoa p;  
    p.nome = "Julia";  
    p.idade = 20;  
}
```

Atributos

- Em Java:

```
class Pessoa{  
    String nome;  
    int idade;  
}
```

- Modelam o que um objeto (instancia da classe) irá ter;

```
public static void Main(String [] args){  
  
    Pessoa p = new Pessoa();  
    p.nome = "Julia"  
    p.idade = 20;  
  
}
```

Encapsulamento

- private: apenas pode ser acessado de dentro da classe;
- public: visível a todas as classes;
- protected: apenas filhos (veremos a frente);
- package: apenas ao pacote (default);

```
public class Pessoa{  
    private String nome;  
    private int idade;  
  
}
```

- Atributos sempre privados!

Métodos

- Funções internas a classe;
- Manipulam os atributos;

```
public class Pessoa{  
  
    private String nome;  
    private int idade;  
  
    public boolean ehDeMaior(){  
        if(idade>18){  
            return true;  
        }else{  
            return false;  
        }  
    }  
}
```

Getters e Setters

- **get:** acessar o valor de um atributo;
- **set:** alterar o valor de um atributo;

```
public class Pessoa{  
  
    private String nome;  
    private int idade;  
  
    public String getNome(){  
        return nome;  
    }  
  
    public void setNome(String novoNome){  
        nome = novoNome;  
    }  
  
}
```


- Aponta para o objeto em questão:

```
public class Pessoa{  
    private String nome;  
    private int idade;  
  
    public void setNome(String nome){  
        this.nome = nome;  
    }  
}
```

Construtor

- Define a função a ser chamada quando o operador **new** for invocado;
- Aloca memória;
- Função com o mesmo nome da classe;
- Convenção: primeiro método declarado;

```
public class Pessoa{  
  
    private String nome;  
    private int idade;  
  
    public Pessoa(){ // Construtor default  
    }  
  
}
```

- Construtor com parâmetro:

```
public class Pessoa{  
    private String nome;  
    private int idade;  
  
    public Pessoa(){ // Construtor default  
    }  
  
    public Pessoa(String nome, int idade){  
        this.nome = nome;  
        this.idade = idade;  
    }  
}
```

static

- Propriedades da classe;
- Todos os objetos vão compartilhar!

```
public class Pessoa{  
    private String nome;  
    private int idade;  
  
    public static int ESPECTATIVA_VIDA = 70;  
  
}
```

```
public static void main(String [] args){  
    System.out.println(Pessoa.ESPECTATIVA_VIDA);  
}
```

toString

- Representação de um objeto na forma de String;
- Método invocado pelo System.out.println()!

```
public class Pessoa{  
    private String nome;  
    private int idade;  
  
    public String toString(){  
        return "Nome: " + this.nome;  
    }  
}
```

toString

```
public static void main(String [] args){  
    Pessoa p = new Pessoa();  
    System.out.println(pessoa);  
}
```

equals

- Define uma forma de comparação entre objetos;
- Assinatura: **public boolean equals(Object objeto);**

```
public class Pessoa{  
  
    private String nome;  
    private int idade;  
  
    public boolean equals(Object outro){  
        if(outro instanceof Pessoa){  
            Pessoa pessoa2 = (Pessoa)(outro);  
            if(pessoa2.nome == this.nome){  
                return true;  
            }  
        }  
        return false;  
    }  
}
```


- Utilização:

```
public static void main(String [] args){  
  
    Pessoa p1 = new Pessoa();  
    p1.setNome("Julia");  
  
    Pessoa p2 = new Pessoa();  
    p2.setNome("Joao");  
  
    if ( p1.equals(p2) ){  
        System.out.println("Possuem o mesmo nome");  
    } else {  
        System.out.println("Nao possuem o mesmo nome");  
    }  
  
}
```


 FIGURA Células. In: . [S.l.]: Disponível em:
<<https://conhecimentocientifico.r7.com/wp-content/uploads/2020/01/celula-o-que-e-definicao-teorias-e-principais-estruturas-das-celulas.jpg>>. Acesso em: 12 nov. 2020.

 FIGURA Puzzle. In: . [S.l.]: Disponível em:
<<https://upload.wikimedia.org/wikipedia/commons/thumb/7/75/Jigsaw.svg/150px-Jigsaw.svg.png>>. Acesso em: 12 nov. 2020.



Attribution 4.0 International (CC BY 4.0)

Duvidas:
Vinicius Takeo Friedrich Kuwaki
vtkwki@gmail.com
github.com/takeofriedrich