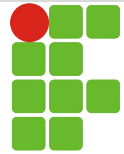
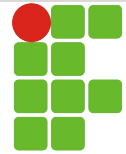


Programação orientada a objetos

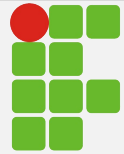
Maximilian Jaderson de Melo

Aula 7

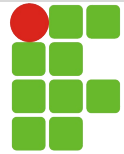


Introdução a Orientação a objetos

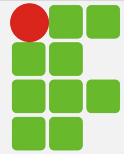
Conteúdo de hoje



- Encapsulamento.
  - Qualificadores de acesso.
- Pacotes.

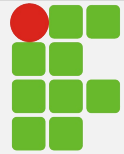


Relembrando



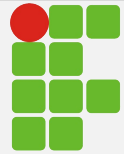
- Pilares.
- Abstração.
- Classe vs Objeto.
- Construtores.

Se existem construtores...



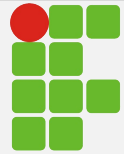
- Java não trabalha com **destrutores**.
  - *Garbage collector* cuida das desalocações.
- Método *finalize()*
  - Não há garantias de que será executado.
  - Ou que executará quando o *GC* for invocado.
  - *System.gc();*

Encapsulamento



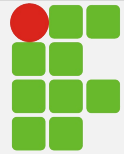
- O ponto central do encapsulamento é oferecer o que o objeto faz e não como faz.
  - Não confunda com abstração.
  - Abstração é conseguir entender o todo e não as partes.
  - Abstração é como representar aquilo que consegue-se pensar sobre.

Encapsulamento



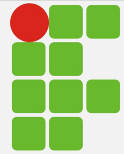
- Impedir que os atributos de uma classe não sejam mais acessados diretamente.
- Ao invés disso, disponibilizar comportamento responsável pelo acesso.

Encapsulamento



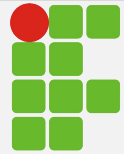
- Imagine uma TV.
  - O acesso a mudar de canal é feito diretamente?
  - O acesso a aumentar/diminuir o volume é feito diretamente?
- Considere que volume e canal sejam os atributos de uma TV.
  - Faz sentido seus atributos serem acessados diretamente?

Encapsulamento



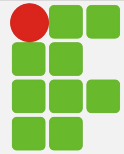
- Qualificadores de acesso
  - *public*
  - *private*
  - *protected*

Encapsulamento



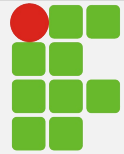
- Modifique os qualificadores de acesso dos atributos dos exercícios da aula passada.

Getters e Setters



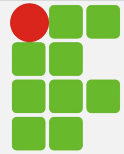
- Para restringir o acesso dos atributos, criamos métodos acessores.

Encapsulamento



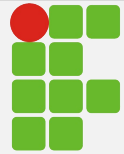
```
class Pessoa{
      String nome;

      public String getNome(){
          return nome;
      }

      public void setNome(String nome){
          this.nome = nome;
      }
}
```

Encapsulamento



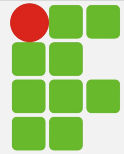
- Percebeu um termo novo no slide anterior?
- O que o *this* faz?

Problema de cobertura



- Os símbolos { } são operadores de escopo.
- Veja o próximo código.

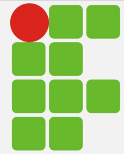
Problema de cobertura



- O que será impresso?

```
class Pessoa{
        String nome;
        public Pessoa(){
                nome = "max";
        }
        public void falar(){
                String nome = "Joaozinho";
                System.out.println(nome);
        }
//continua
```

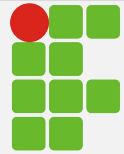
Problema de cobertura



- O que será impresso?

```
//continuação
        public static void main(String args[]){
            Pessoa p = new Pessoa();
            p.falar();
        }
 }
```

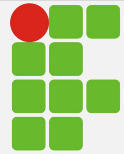
Problema de cobertura



- Há uma ambiguidade nas referências de *nome*.
  - Neste caso o atributo foi encoberto.
- O mesmo problema ocorre abaixo?

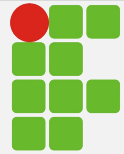
```
        ...

        public void setNome(String nome){
            this.nome = nome;
        }
}
```

Problema de cobertura



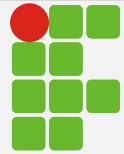
- O operador *this* resolve o problema.
  - É um ponteiro para a definição da classe.
  - Ele sempre aponta para os atributos **da classe!!!!**.
  - Não confundir com **variável de classe**.

Curiosidade



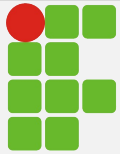
- No *Netbeans* o atalho *Alt+Insert* abre um menu disponibilizando:
  - *Getters* e *Setters*.
  - Construtores padrão e com atributos.
  - Etc.

Exercícios



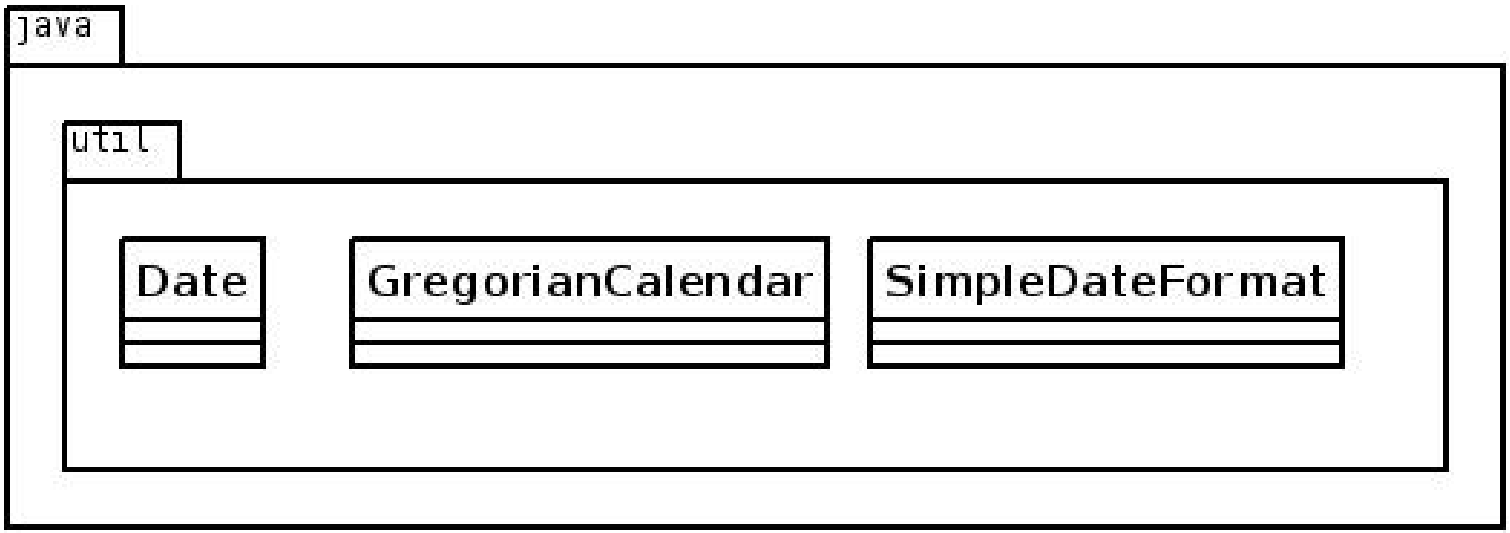
1. Crie *getters* e *setters* para cada atributo das classes Professor, Aluno, Turma e Escola.

Pacotes

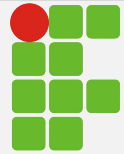


- Repare nas importações em java.
  - *import java.util.Date;*
- Pasta chamada *java*.
- Dentro dela há outra chamada *util*.
- Dentro dela há a classe Date.

Pacotes

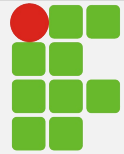


Pacotes



- Essas pastas são chamadas de pacotes. - Indicam a hierarquia.
- O programador também pode especificar hierarquia nas classes.
- No *Netbeans* basta ir em:
  - Novo -> pacote.

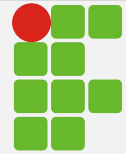
Pacotes



- Sempre que colocar uma classe em um pacote (pasta), a seguinte linha aparecerá:

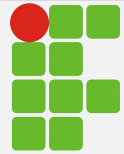
```
package nomePacote;

public class Pessoa{

}
```

Convenções



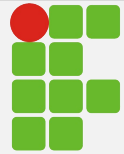
- Pense numa URL de site:
  - *www.ifms.edu.br*.
- Os pacotes combinam com a *url* de um site, de maneira reversa, sem o *www*:
  - *br.edu.ifms*.
- Quantas pastas teríamos no exemplo acima?
- É possível criar classes em cada uma dessas pastas.
- Considere *br.edu.ifms.pacote1* e *br.edu.ifms.pacote2*.
  - Em qual pasta pacote1 e pacote2 são criados?

Exercícios

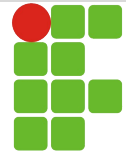


1. Nos exercícios da aula passada:
   - Crie um pacote chamado “entidades”.
   - Crie um pacote chamado “controlador”
   - Mova as classes Escola, Professor, Aluno e Turma para dentro de “entidades”.
   - Crie uma Classe PessoaControlador com um método *main* e instancie 3 alunos.
   - Atribua valores para os objetos usando um *JFrame*.
   - Imprima-os usando *JOptionPane*.

Exercícios

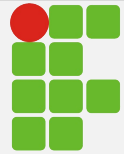


2. Modifique o qualificador de acesso dos atributos de *Aluno* para *protected* e diga se é possível acessá-los na *PessoaControlador*.



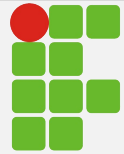
EM PHP

Encapsulamento



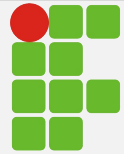
- Todas as regras vistas até aqui continuam válidas.

Pacotes e MVC



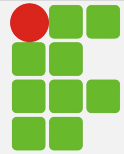
- PHP não implementa a ideia de pacotes.
- Não há a necessidade de criar a estrutura hierárquica conforme um site:
  - *www.ifms.edu.br* = *br.edu.ifms.nomePacote*.
- Cria-se apenas uma pasta com o nome do pacote.

Pacotes e MVC



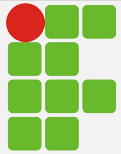
- O objetivo do MVC é separar as responsabilidades de cada classe.
  - Especializar cada classe no que faz.
  - Classes são separadas em camadas.
- Três pastas (ou pacotes quando suportado) principais.
  - Controlador.
  - Visão.
  - Modelo (Entidades).

Implementação em MVC



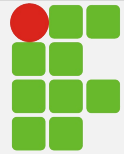
- Uma interpretação é criar um arquivo que centraliza as chamadas dos controladores.
- O papel dos controladores é mediar os dados das interfaces gráficas (visão) e modelo (persistir entidades no banco).
  - É no controlador que ficam as regras de negócio.

Exercícios



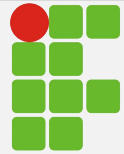
1. Nos exercícios da aula passada (PHP):
   - Crie um pacote chamado “entidades”.
   - Crie um pacote chamado “controlador”
   - Mova as classes Escola, Professor, Aluno e Turma para dentro de “entidades”.
   - Crie uma Classe PessoaControlador/AlunoControlador e realize o cadastro de um aluno.

Próxima aula



- Herança.

Dúvidas, críticas ou sugestões?



maximilian.melo@ifms.edu.br
max.mjm.melo@gmail.com