As 3 principais características de uma

linguagem orientada a objetos são:

* **Encapsulamento**

Capacidade de uma classe proteger os atributos do acesso externo. Isso é feito por meio da declaração de atributos privados e criação de métodos públicos para acesso aos atributos (estes métodos são chamados de setters e getters).

* **Herança**

Capacidade de uma classe poder extender o seu comportamento, ou seja, o desenvolvedor pode criar superclasses e subclasses, fazendo uso de generalização e especialização.

* **Polimorfismo**

Capacidade de um objeto mudar o seu comportamento de acordo com a sua instância. Ou seja, podemos criar muitas formas de um objeto ser implementando, cada uma delas fornecendo um comportamento diferente para o objeto.

Abrindo o eclipse:

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Novo projeto:

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Nome do projeto: **projetoAula03**

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**/src/ (**Source folder**)**

Pasta raiz para criação do projeto Java.

Interface gráfica do usuário, Texto, Aplicativo, Word

Descrição gerada automaticamente

/**entities**

Neste pacote, iremos criar as classes do projeto que serão utilizadas como entidades do Java. Aqui vamos usar o padrão **JavaBeans**.

Interface gráfica do usuário, Aplicativo, Tabela, Excel

Descrição gerada automaticamente

JavaBeans

Padrão utilizado em Java para criação de classes que tem como objetivo modelagem de dados, tais como as entidades. São características de uma classe JavaBean:

* Atributos privados
* Métodos de encapsulamento para cada atributo, denominados de setters e getters
* Construtor sem entrada de argumentos
* Construtor com entrada de argumentos
* Sobrescrita dos métodos da classe Object

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

**Atributos privados**

**package** entities;

**public** **class** Pessoa {

//atributos (dados)

**private** Integer id;

**private** String nome;

}

**Métodos de encapsulamento para cada atributo, denominados de setters e getters**

SOURCE / GENERATE GETTERS AND SETTERS

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**package** entities;

**public** **class** Pessoa {

// atributos (dados)

**private** Integer id;

**private** String nome;

**public** Integer getId() {

**return** id;

}

**public** **void** setId(Integer id) {

**this**.id = id;

}

**public** String getNome() {

**return** nome;

}

**public** **void** setNome(String nome) {

**this**.nome = nome;

}

}

**Herança (é-um)**

É um tipo de relacionamento entre classes que utiliza o conceito de **generalização / especialização**, ou seja, **superclasses e subclasses**.

Interface gráfica do usuário

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

***extends***

Palavra reservada para criar um vínculo de herança em Java.

Exemplo:

**package** entities;

**public class Profissional extends Pessoa {**

**private** String cpf;

**private** String telefone;

**private** String email;

**public** String getCpf() {

**return** cpf;

}

**public** **void** setCpf(String cpf) {

**this**.cpf = cpf;

}

**public** String getTelefone() {

**return** telefone;

}

**public** **void** setTelefone(String telefone) {

**this**.telefone = telefone;

}

**public** String getEmail() {

**return** email;

}

**public** **void** setEmail(String email) {

**this**.email = email;

}

}

**Regras sobre Herança no Java:**

* Em Java, não podemos fazer herança múltipla entre classes:

**class** A {

}

**class** B {

}

**class C extends A**~~, B~~ {

}

* Em Java, se uma classe for declarada como **final**, ela não pode ter filhos.

**class** A {

}

**final** **class** B **extends** A{

}

**class** C ~~extends B~~ {

}

**Associação (**ter-1 ou ter-muitos**)**

Relacionamento de associação (utilização) entre classes,

podendo ser do tipo TER- 1 ou TER-MUITOS.

Interface gráfica do usuário, Diagrama, Excel

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**package** entities;

**public** **class** Servico {

}

**package** entities;

**public** **class** Servico {

**private** Integer id;

**private** String nome;

**private** Double valor;

**public** Integer getId() {

**return** id;

}

**public** **void** setId(Integer id) {

**this**.id = id;

}

**public** String getNome() {

**return** nome;

}

**public** **void** setNome(String nome) {

**this**.nome = nome;

}

**public** Double getValor() {

**return** valor;

}

**public** **void** setValor(Double valor) {

**this**.valor = valor;

}

}

**Fazendo o vínculo de Profissional com Serviço**

PROFISSIONAL TEM 1 SERVIÇO

**package** entities;

**public** **class** Profissional **extends** Pessoa {

**private** String cpf;

**private** String telefone;

**private** String email;

**private Servico servico;**

**public** String getCpf() {

**return** cpf;

}

**public** **void** setCpf(String cpf) {

**this**.cpf = cpf;

}

**public** String getTelefone() {

**return** telefone;

}

**public** **void** setTelefone(String telefone) {

**this**.telefone = telefone;

}

**public** String getEmail() {

**return** email;

}

**public** **void** setEmail(String email) {

**this**.email = email;

}

**public Servico getServico() {**

**return servico;**

**}**

**public void setServico(Servico servico) {**

**this.servico = servico;**

**}**

}

**Enums**

São classes simples para modelagem de tipos multivalorados no Java,

ou seja, tipos de dados que já possuam valores pré-definidos.

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**package** enums;

**public** **enum** TipoPagamento {

***DÉBITO***,

***CRÉDITO***,

***PIX***

}

Interface gráfica do usuário, Diagrama

Descrição gerada automaticamente

**Relacionando Serviço com TipoPagamento**

SERVIÇO TEM 1 TIPO DE PAGAMENTO

**package** entities;

**import** enums.TipoPagamento;

**public** **class** Servico {

**private** Integer id;

**private** String nome;

**private** Double valor;

**private TipoPagamento tipoPagamento;**

**public** Integer getId() {

**return** id;

}

**public** **void** setId(Integer id) {

**this**.id = id;

}

**public** String getNome() {

**return** nome;

}

**public** **void** setNome(String nome) {

**this**.nome = nome;

}

**public** Double getValor() {

**return** valor;

}

**public** **void** setValor(Double valor) {

**this**.valor = valor;

}

**public TipoPagamento getTipoPagamento() {**

**return tipoPagamento;**

**}**

**public void setTipoPagamento(TipoPagamento tipoPagamento) {**

**this.tipoPagamento = tipoPagamento;**

**}**

}

/principal/Program.java

Classe principal do projeto, utilizada para executar o sistema.

**package** principal;

**public** **class** Program {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

}

}

**Desenvolvimento baseado em camadas:**

Interface gráfica do usuário

Descrição gerada automaticamente

/controllers/ProfissionalController.java

Classe para executar o fluxo de cadastro de um profissional, pedindo que o usuário do sistema preencha os dados necessários para incluir um profissional.

Interface gráfica do usuário

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**Fluxo para cadastrar um profissional:**

Interface gráfica do usuário, Aplicativo, Tabela, Excel

Descrição gerada automaticamente

**package** controllers;

**import** entities.Profissional;

**public** **class** ProfissionalController {

//método para realizar o fluxo de cadastro de um profissional..

**public** **void** cadastrarProfissional() {

System.***out***.println

("\n\*\*\* CADASTRO DE PROFISSIONAL \*\*\*\n");

//Sistema solicita que o usuário entre

//com os dados do Profissional

Profissional profissional = **new** Profissional();

//**TODO**

System.***out***.println("\nFIM!");

}

}

/helpers/InputHelper.java

Classe para fornecer métodos que facilitem a captura de dados fornecidos pelo usuário.

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**package** helpers;

**import** java.util.Scanner;

**public** **class** InputHelper {

//método para fazer a captura de

//um dado informado pelo usuário

**public** String inputData(String message) {

Scanner scanner = **new** Scanner(System.***in***);

System.***out***.print(message);

**return** scanner.nextLine();

}

}

Voltando para o controlador:

/controllers/ProfissionalController.java

**package** controllers;

**import** entities.Profissional;

**import** helpers.InputHelper;

**public** **class** ProfissionalController {

//método para realizar o fluxo de cadastro de um profissional..

**public** **void** cadastrarProfissional() {

System.***out***.println

("\n\*\*\* CADASTRO DE PROFISSIONAL \*\*\*\n");

//Sistema solicita que o usuário entre

//com os dados do Profissional

Profissional profissional = **new** Profissional();

InputHelper inputHelper = **new** InputHelper();

profissional.setId(Integer.*parseInt*(inputHelper.inputData

("ID DO PROFISSIONAL: ")));

profissional.setNome(inputHelper.inputData

("NOME DO PROFISSIONAL: "));

profissional.setCpf(inputHelper.inputData

("CPF DO PROFISSIONAL: "));

profissional.setEmail(inputHelper.inputData

("EMAIL DO PROFISSIONAL: "));

profissional.setTelefone(inputHelper.inputData

("TELEFONE DO PROFISSIONAL: "));

//**TODO**

System.***out***.println("\nFIM!");

}

}

Voltando na classe Program.java para então executarmos o método “cadastrarProfissional” da classe ProfissionalController:

Interface gráfica do usuário

Descrição gerada automaticamente

**package** principal;

**import** controllers.ProfissionalController;

**public** **class** Program {

**public** **static** **void** main(String[] args) {

ProfissionalController profissionalController

= **new** ProfissionalController();

profissionalController.cadastrarProfissional();

}

}

**Executando:**

**package** principal;

**import** controllers.ProfissionalController;

**public** **class** Program {

**public** **static** **void** main(String[] args) {

ProfissionalController profissionalController

= **new** ProfissionalController();

profissionalController.cadastrarProfissional();

}

}

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

**\*\*\* CADASTRO DE PROFISSIONAL \*\*\***

**ID DO PROFISSIONAL: 1**

**NOME DO PROFISSIONAL: Sergio Mendes**

**CPF DO PROFISSIONAL: 123.456.789-00**

**EMAIL DO PROFISSIONAL: sergio.coti@gmail.com**

**TELEFONE DO PROFISSIONAL: (21) 96957 5900**

**FIM!**

Modificando a classe **InputHelper.java**

**package** helpers;

**import** javax.swing.JOptionPane;

**public** **class** InputHelper {

//método para fazer a captura de

//um dado informado pelo usuário

**public** String inputData(String message) {

String valor = JOptionPane.*showInputDialog*(message);

**return** valor;

}

}

**Testando:**

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

**Construindo a camada de**

**repositório do sistema:**

Interface gráfica do usuário

Descrição gerada automaticamente

/Repositories/ProfissionalRepository.java

Classe para persistência de dados do profissional.

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**package** repositories;

**import** java.io.File;

**import** java.io.FileOutputStream;

**import** java.io.PrintWriter;

**import** entities.Profissional;

**public** **class** ProfissionalRepository {

//método para gravar os dados do profissional

//em um arquivo na máquina do usuário

**public** **void** exportarDados(Profissional profissional)

**throws** Exception {

//abrindo um arquivo para gravação

PrintWriter printWriter = **new** PrintWriter

(**new** FileOutputStream

(**new** File("c:\\temp\\profissionais.txt"), **true**));

//escrevendo os dados

printWriter.write("\nID DO PROFISSIONAL....: "

+ profissional.getId());

printWriter.write("\nNOME DO PROFISSIONAL..: "

+ profissional.getNome());

printWriter.write("\nCPF...................: "

+ profissional.getCpf());

printWriter.write("\nTELEFONE..............: "

+ profissional.getTelefone());

printWriter.write("\nEMAIL.................: "

+ profissional.getEmail());

printWriter.write("\nNOME DO SERVIÇO.......: "

+ profissional.getServico().getNome());

printWriter.write("\nVALOR DO SERVIÇO......: "

+ profissional.getServico().getValor());

printWriter.write("\nTIPO DE PAGAMENTO.....: "

+ profissional.getServico().getTipoPagamento());

printWriter.write("\n...");

//salvando e fechando o arquivo

printWriter.flush();

printWriter.close();

}

}

Voltando no controlador:

Interface gráfica do usuário

Descrição gerada automaticamente

**package** controllers;

**import** entities.Profissional;

**import** entities.Servico;

**import** enums.TipoPagamento;

**import** helpers.InputHelper;

**import** repositories.ProfissionalRepository;

**public** **class** ProfissionalController {

//método para realizar o fluxo de cadastro de um profissional..

**public** **void** cadastrarProfissional() {

System.***out***.println

("\n\*\*\* CADASTRO DE PROFISSIONAL \*\*\*\n");

Profissional profissional = **new** Profissional();

Servico servico = **new** Servico();

InputHelper inputHelper = **new** InputHelper();

//Sistema solicita que o usuário entre

//com os dados do Profissional

profissional.setId(Integer.*parseInt*

(inputHelper.inputData("ID DO PROFISSIONAL: ")));

profissional.setNome(inputHelper.inputData

("NOME DO PROFISSIONAL: "));

profissional.setCpf(inputHelper.inputData

("CPF DO PROFISSIONAL: "));

profissional.setEmail(inputHelper.inputData

("EMAIL DO PROFISSIONAL: "));

profissional.setTelefone(inputHelper.inputData

("TELEFONE DO PROFISSIONAL: "));

//Sistema solicita que o usuário entre

//com os dados do Serviço

servico.setNome(inputHelper.inputData

("NOME DO SERVIÇO: "));

servico.setValor(Double.*parseDouble*

(inputHelper.inputData("VALOR DO SERVIÇO: ")));

String opcao = inputHelper.inputData

("TIPO DE PAGAMENTO (C)Crédito, (D)Débito ou (P)Pix: ");

**switch**(opcao) {

**case** "C":

servico.setTipoPagamento(TipoPagamento.***CRÉDITO***);

**break**;

**case** "D":

servico.setTipoPagamento(TipoPagamento.***DÉBITO***);

**break**;

**case** "P":

servico.setTipoPagamento(TipoPagamento.***PIX***);

**break**;

}

//associar o profissional ao serviço

profissional.setServico(servico);

**try** {

ProfissionalRepository profissionalRepository

= **new** ProfissionalRepository();

profissionalRepository.exportarDados(profissional);

System.***out***.println

("\nPROFISSIONAL CADASTRADO COM SUCESSO!");

}

**catch**(Exception e) {

System.***out***.println("\nERRO: " + e.getMessage());

}

System.***out***.println("\nFIM!");

}

}

Executando:

Interface gráfica do usuário, Texto, Aplicativo, Word

Descrição gerada automaticamente

**Arquivo gerado:**

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente com confiança média

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente