

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).

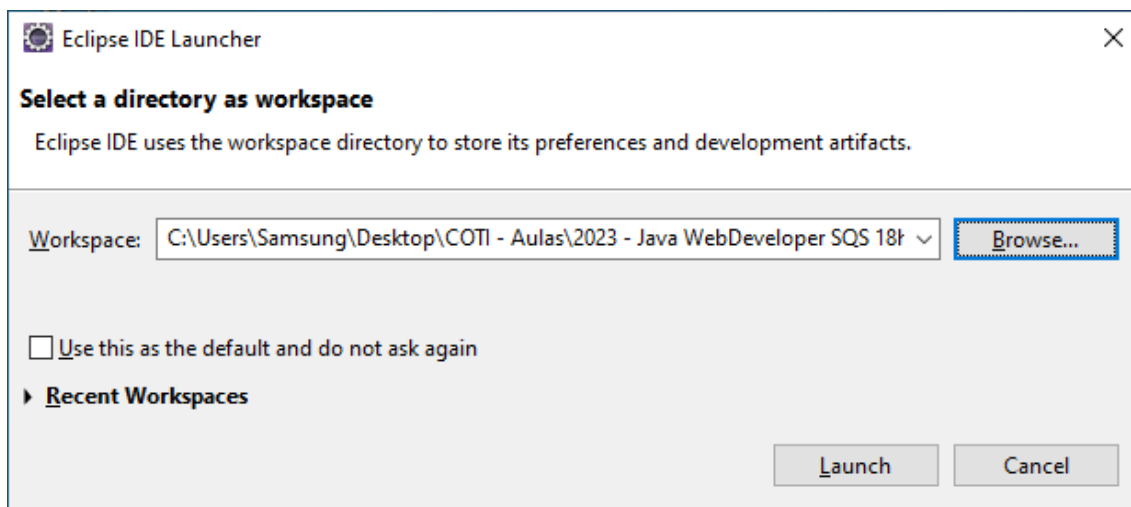
- **Herança**

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

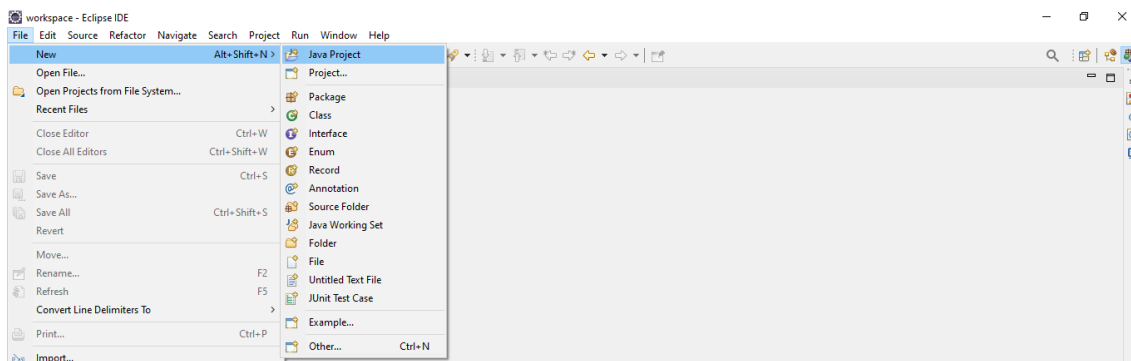
- **Polimorfismo**

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.

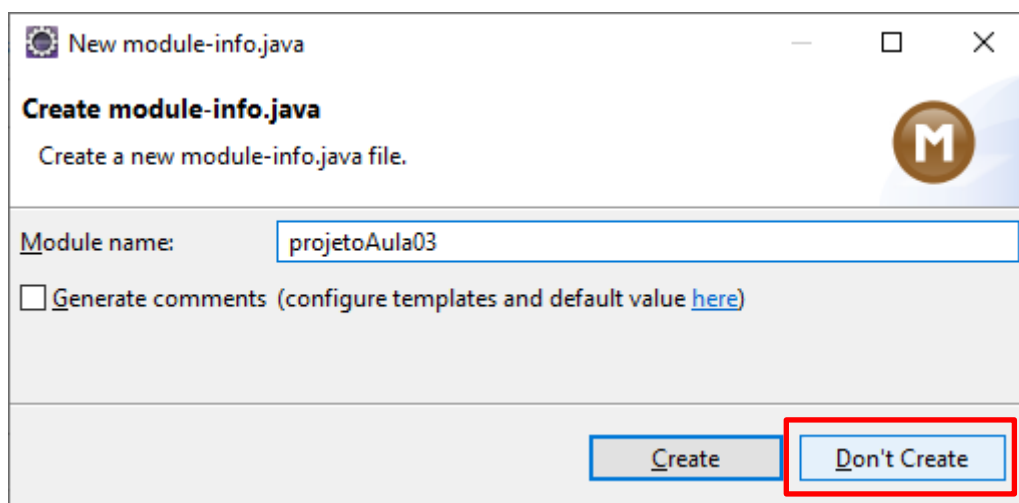
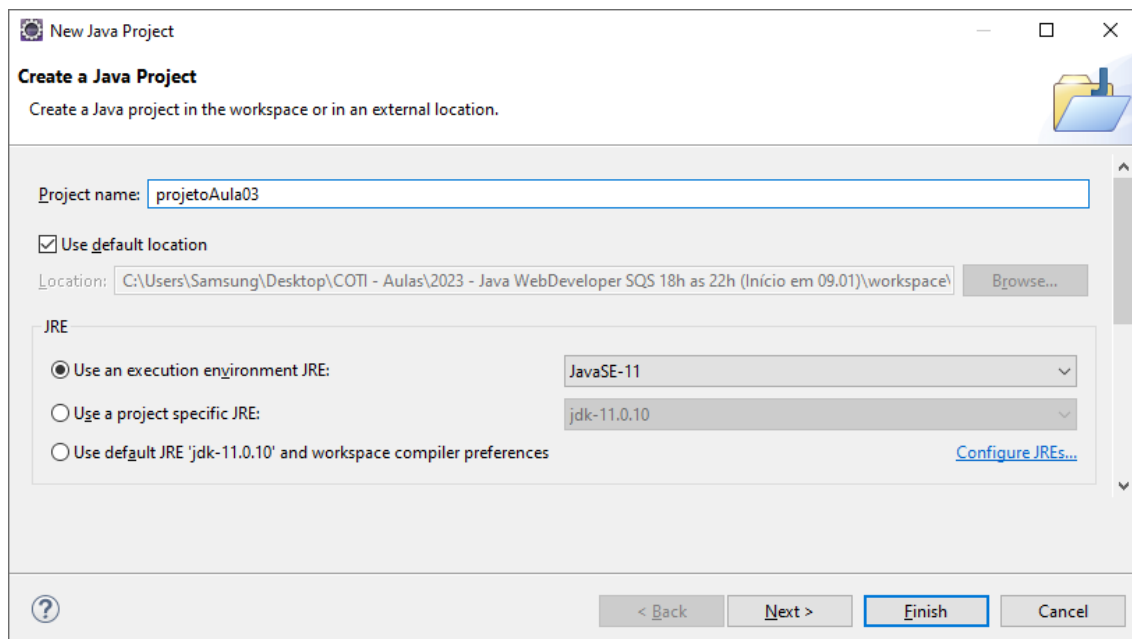
Abrindo o eclipse:



Novo projeto:

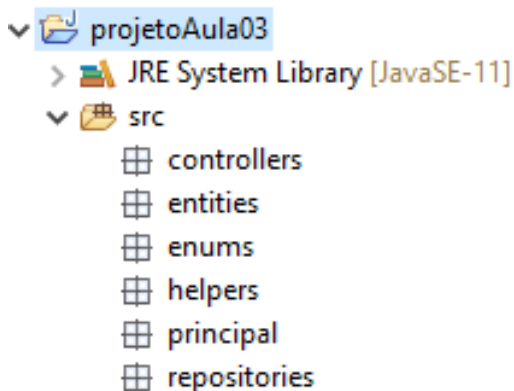


Nome do projeto: **projetoAula03**



/src/ (Source folder)

Pasta raiz para criação do projeto Java.



/entities

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

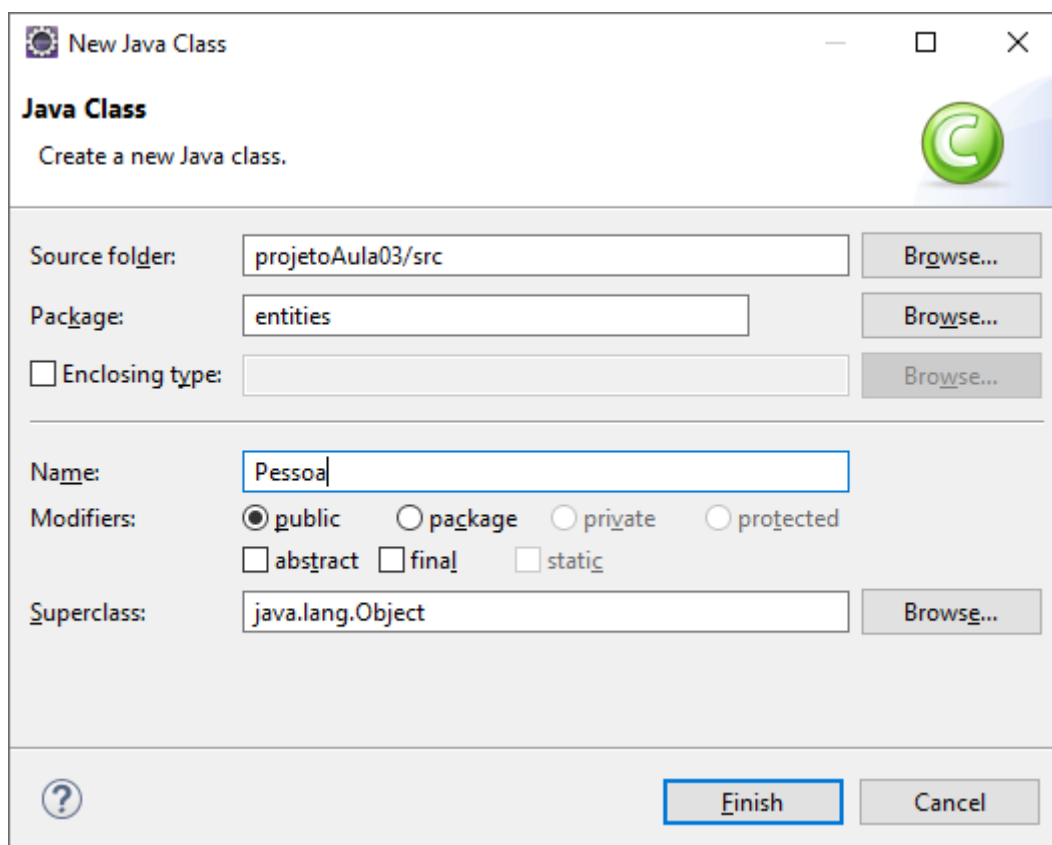
/entities

Pessoa
- id : Integer
- nome : String

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



New Java Class

Java Class
Create a new Java class.

Source folder: projetoAula03/src Browse...

Package: entities Browse...

☐ Enclosing type: Browse...

Name: Pessoa

Modifiers: ☒ public ☐ package ☐ private ☐ protected
☐ abstract ☐ final ☐ static

Superclass: java.lang.Object Browse...

Finish Cancel

Atributos privados

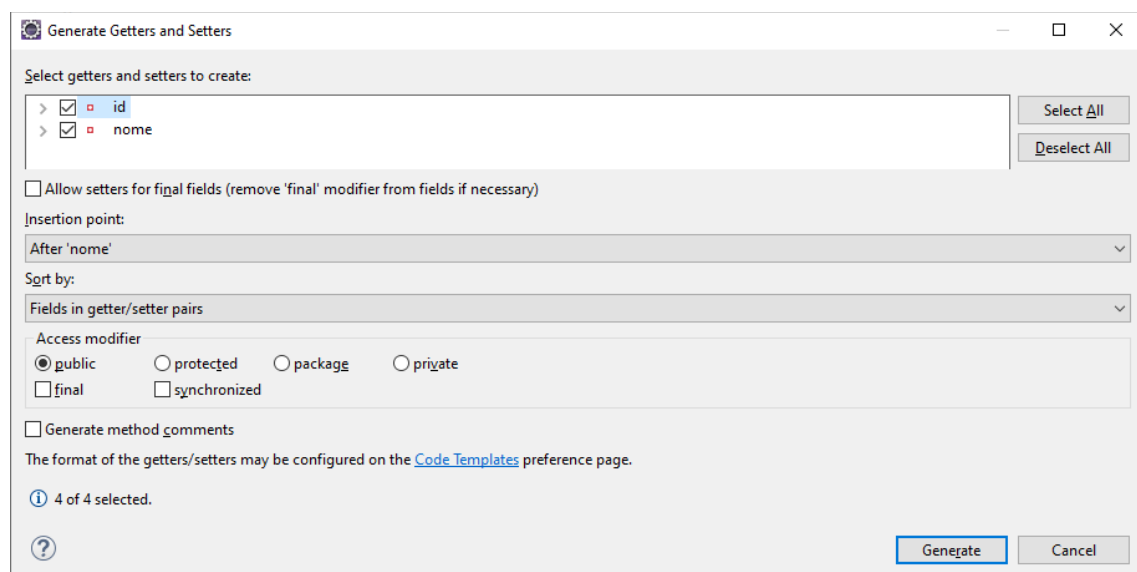
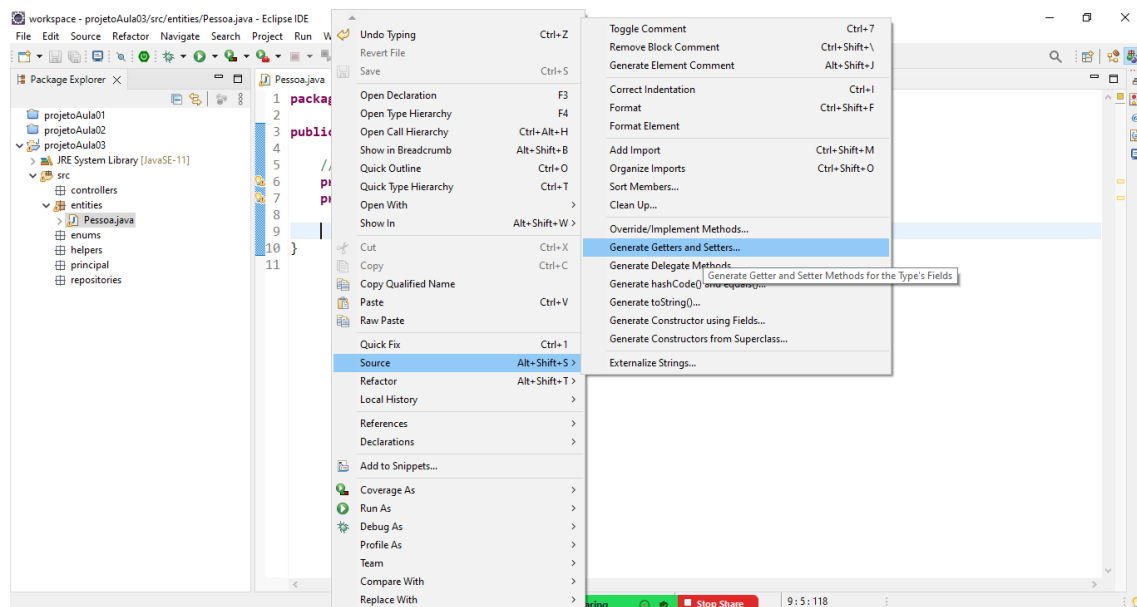
`package` entities;

```
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



```
package entities;

public class Pessoa {

    // atributos (dados)
    private Integer id;
    private String nome;

    public Integer getId() {
        return id;
    }

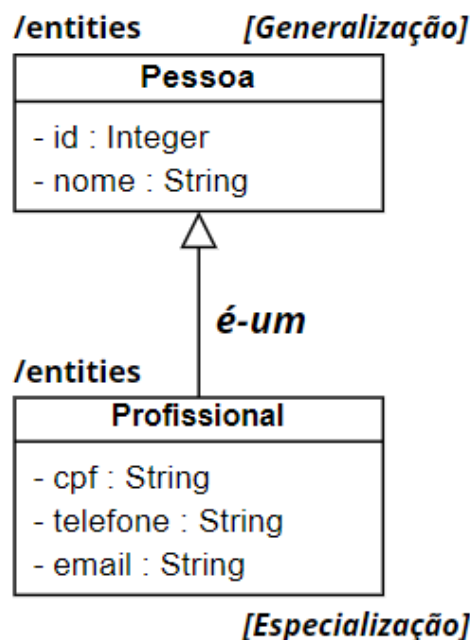
    public void setId(Integer id) {
        this.id = id;
    }

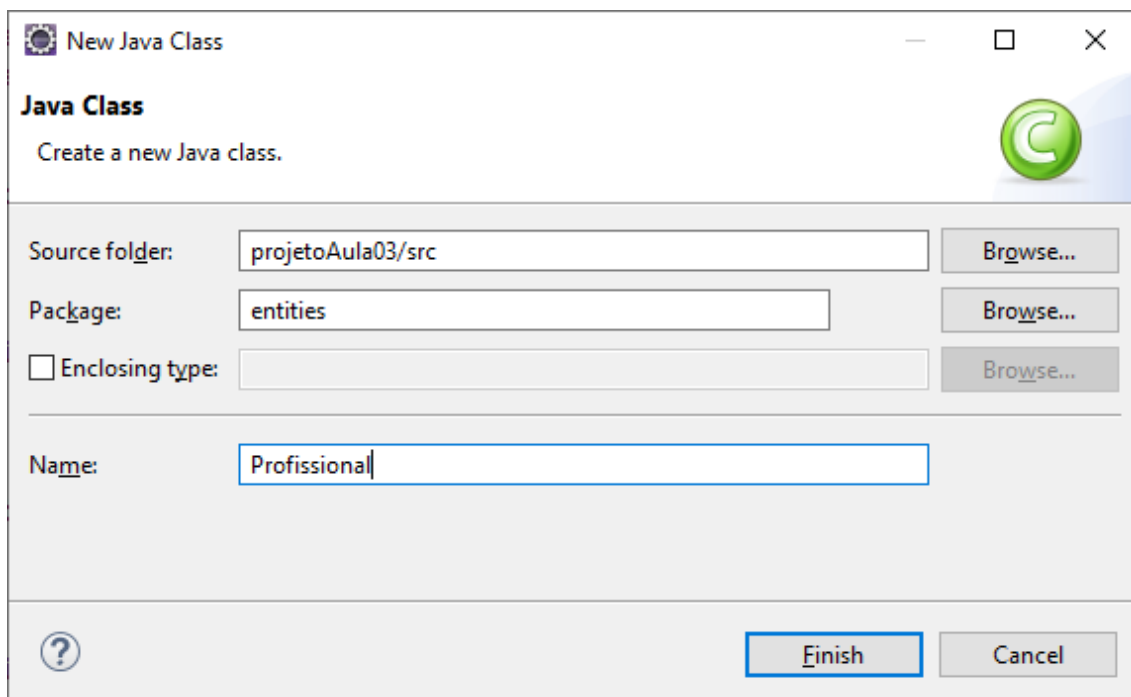
    public String getNome() {
        return nome;
    }

    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**.





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;  
}  
}
```

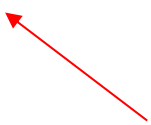
Regras sobre Herança no Java:

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

```
class A {  
  
}  
  
class B {  
  
}  
  
class C extends A, B {  
  
}
```

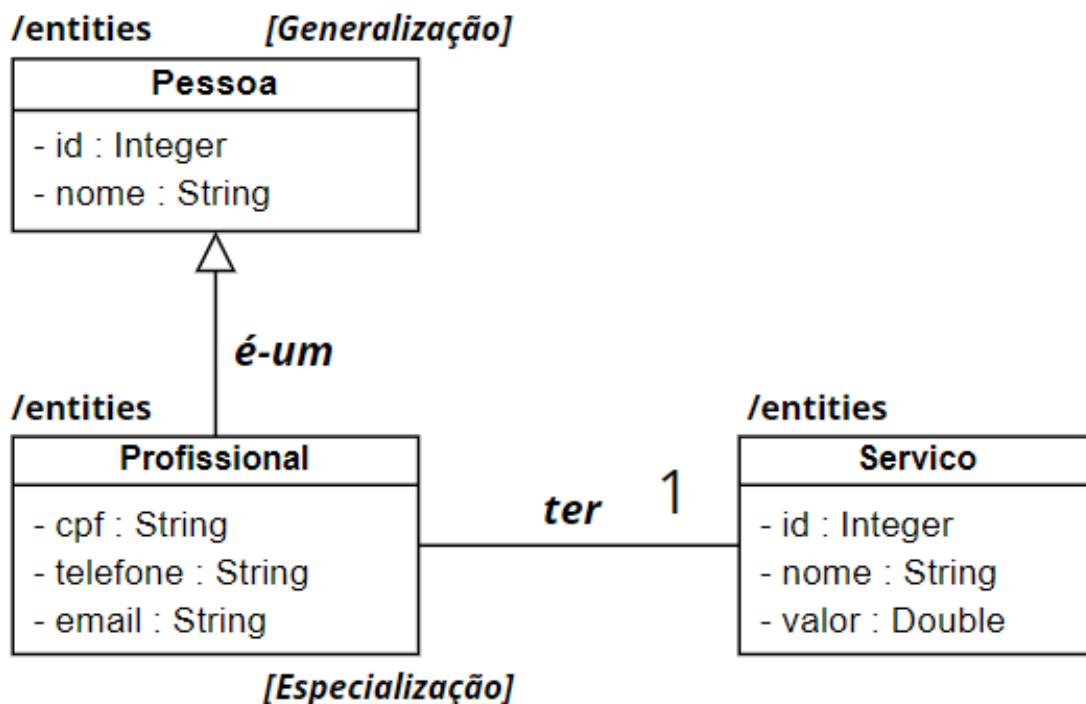
- Em Java, se uma classe for declarada como **final**, ela não pode ter filhos.

```
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.



New Java Class

Java Class
Create a new Java class.

Source folder: [Browse...](#)

Package: [Browse...](#)

☐ Enclosing type: [Browse...](#)

Name:

Modifiers: ☒ public ☐ package ☐ private ☐ protected
☐ abstract ☐ final ☐ static

[?](#) [Finish](#) [Cancel](#)

```
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;
}

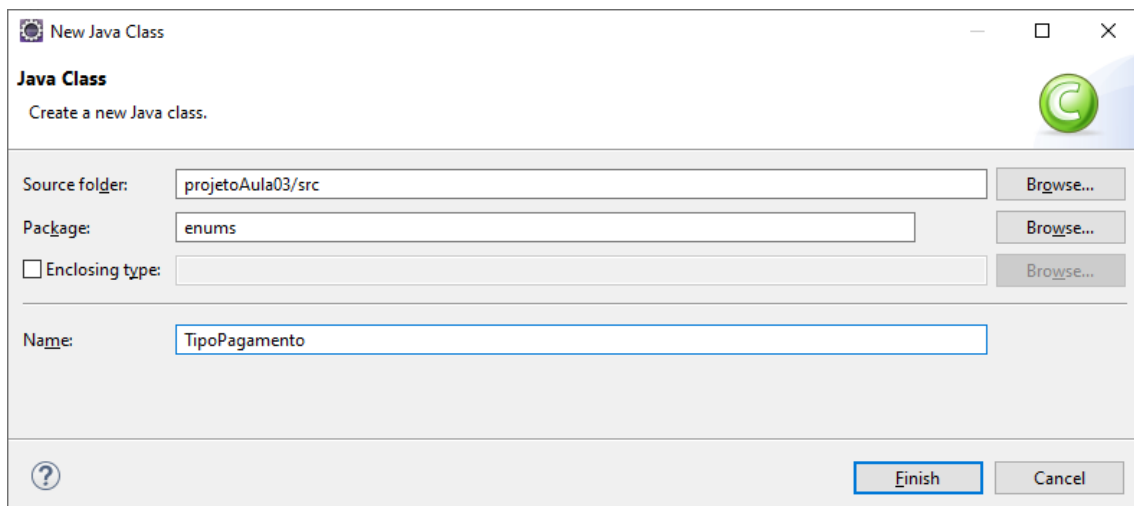
public void setEmail(String email) {
    this.email = email;
}

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.



```
package enums;
```

```
public enum TipoPagamento {
```

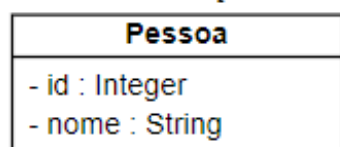
```
    DÉBITO,
```

```
    CRÉDITO,
```

```
    PIX
```

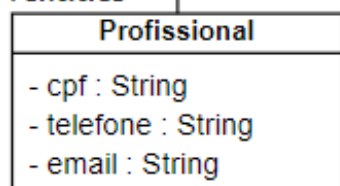
```
}
```

/entities [Generalização]



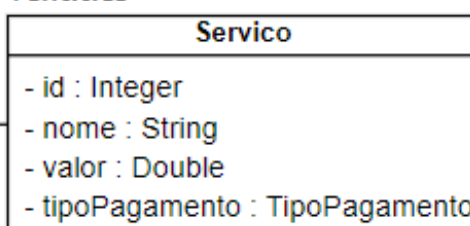
é-um

/entities



[Especialização]

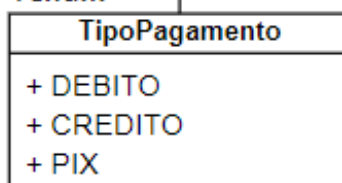
/entities



ter

1

/enum



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;
```

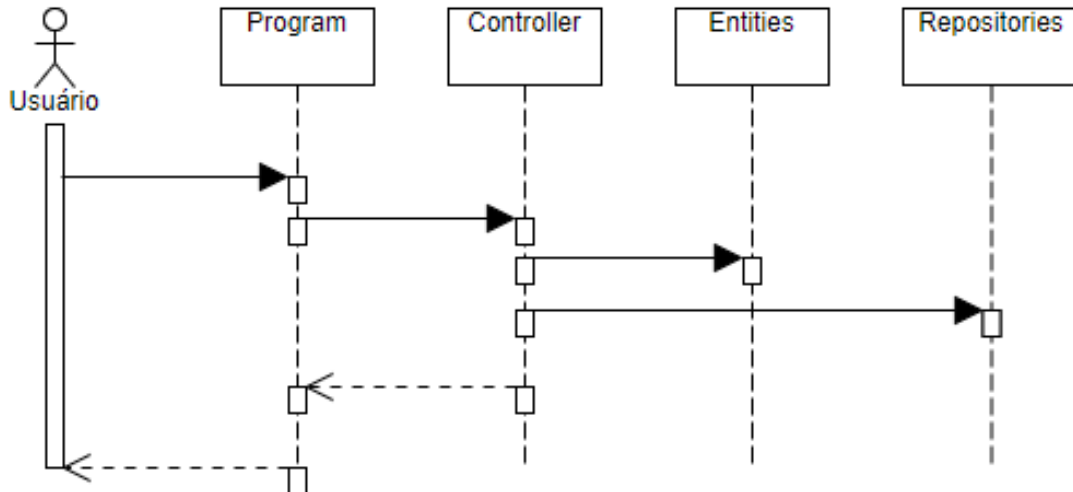
```
public Integer getId() {  
    return id;  
}  
  
public void setId(Integer id) {  
    this.id = id;  
}  
  
public String getNome() {  
    return nome;  
}  
  
public void setNome(String nome) {  
    this.nome = nome;  
}  
  
public Double getValor() {  
    return valor;  
}  
  
public 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.

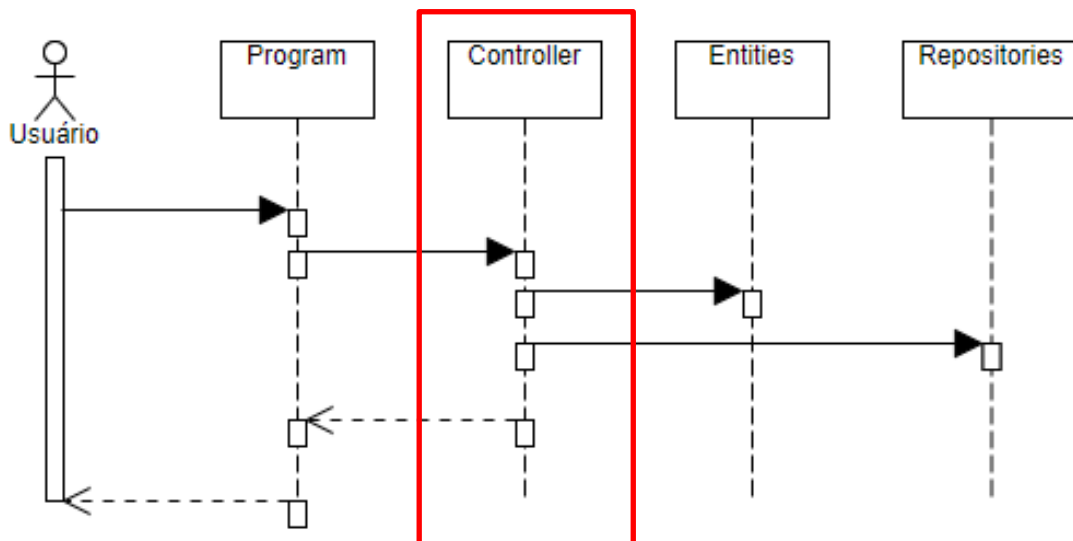
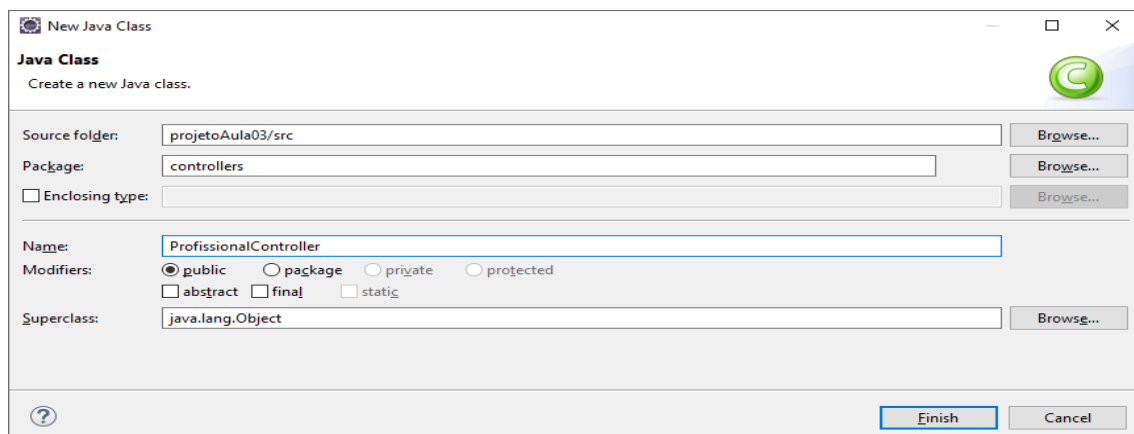
```
package principal;  
  
public class Program {  
  
    public static void main(String[] args) {  
  
        // TODO Auto-generated method stub  
  
    }  
}
```

Desenvolvimento baseado em camadas:



/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.

New Java Class

Java Class
Create a new Java class.

Source folder: projetoAula03/src Browse...

Package: controllers Browse...

☐ Enclosing type: Browse...

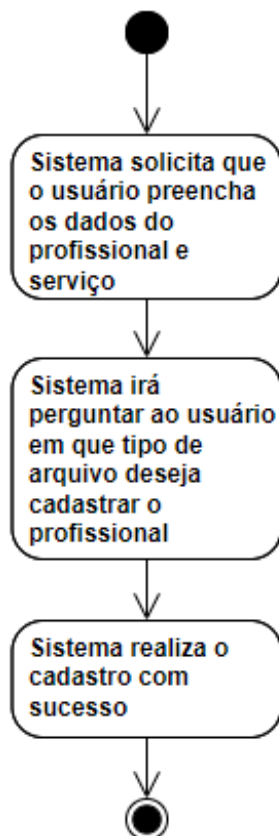
Name: ProfissionalController

Modifiers: ☒ public ☐ package ☐ private ☐ protected
☐ abstract ☐ final ☐ static

Superclass: java.lang.Object Browse...

Finish Cancel

Fluxo para cadastrar um profissional:



```
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");

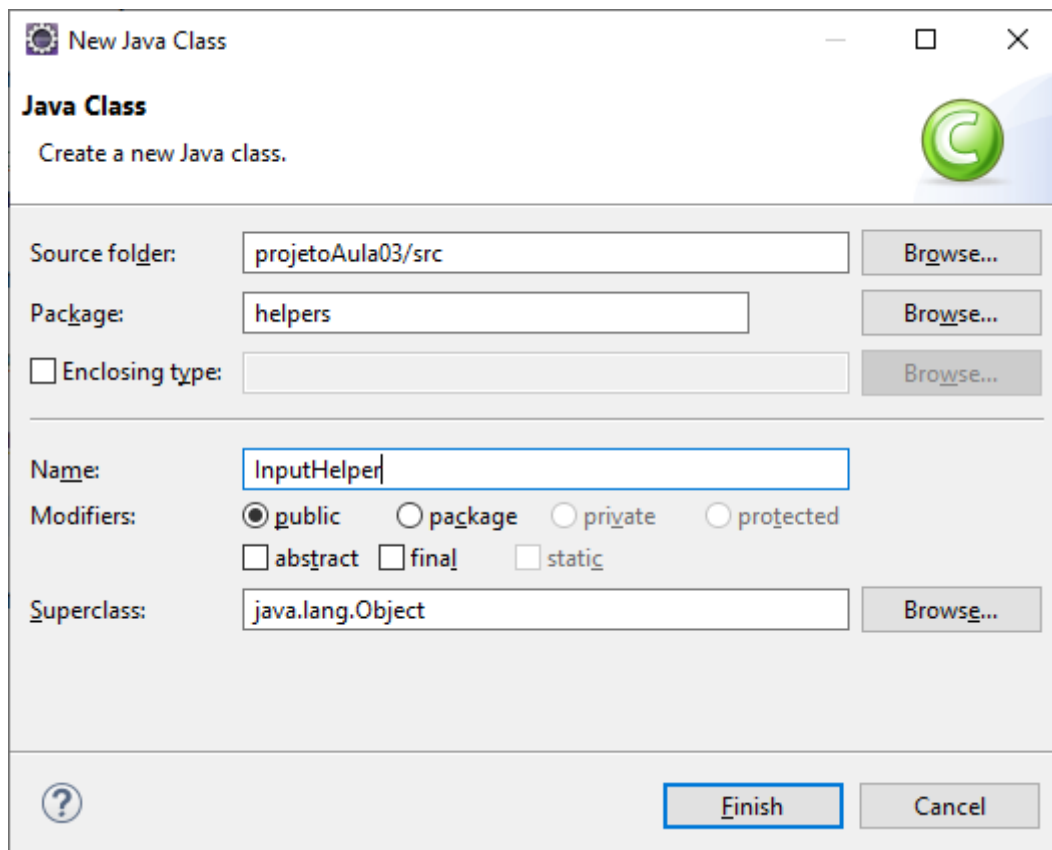
        //Sistema solicita que o usuário entre
        //com os dados do Profissional
        Profissional profissional = new Profissional();

        //TODO

        System.out.println("\nFIM!");
    }
}
```

/helpers/**InputHelper.java**

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



```
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: "));
```

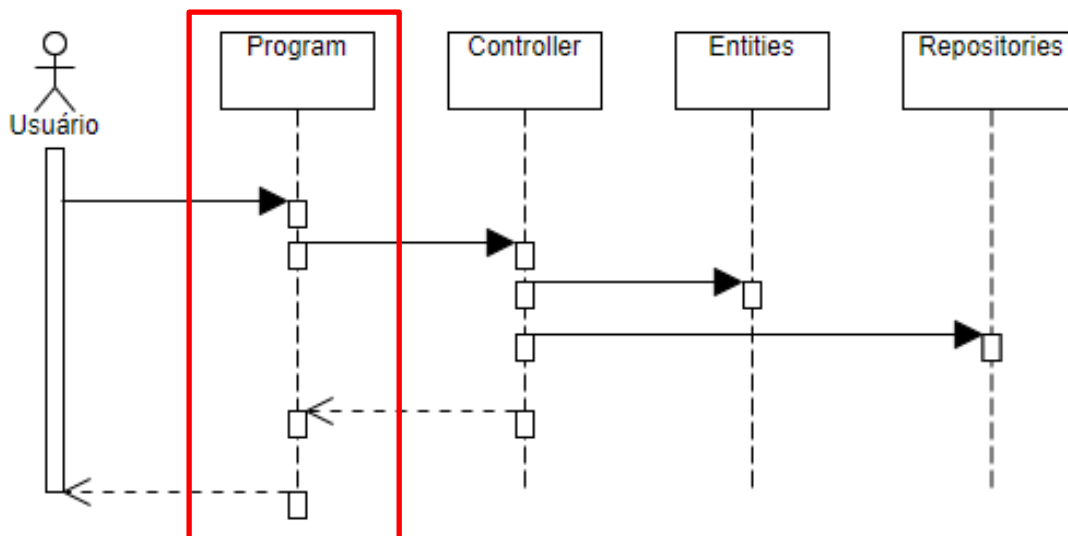
```
        profissional.setTelefone(inputHelper.inputData  
            ("TELEFONE DO PROFISSIONAL: "));
```

```
        //TODO
```

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

```
    }  
}
```

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



```
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 {
```

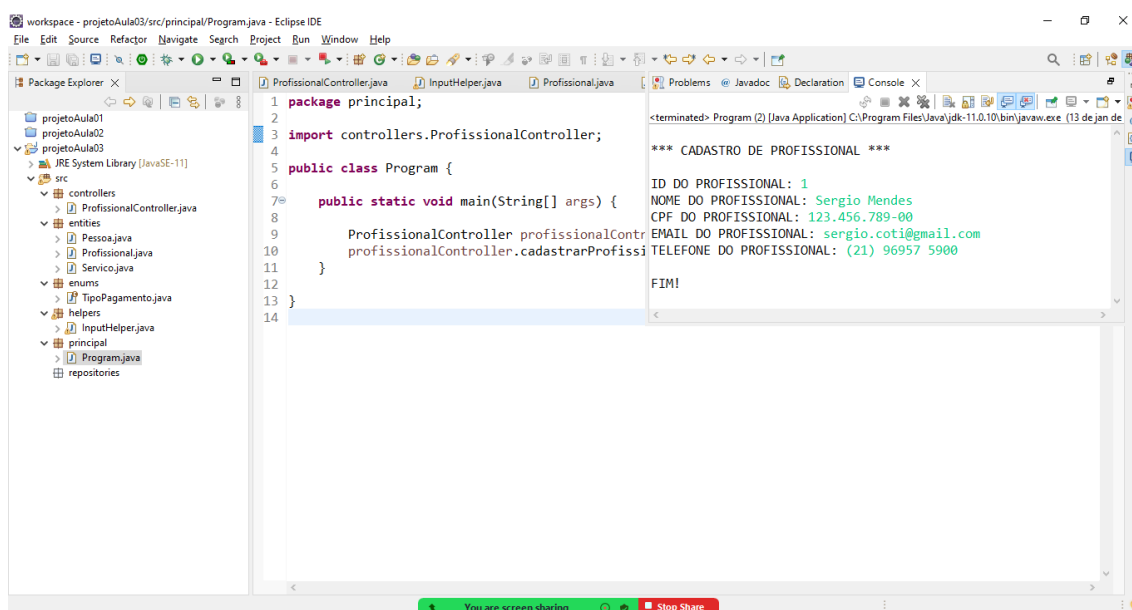
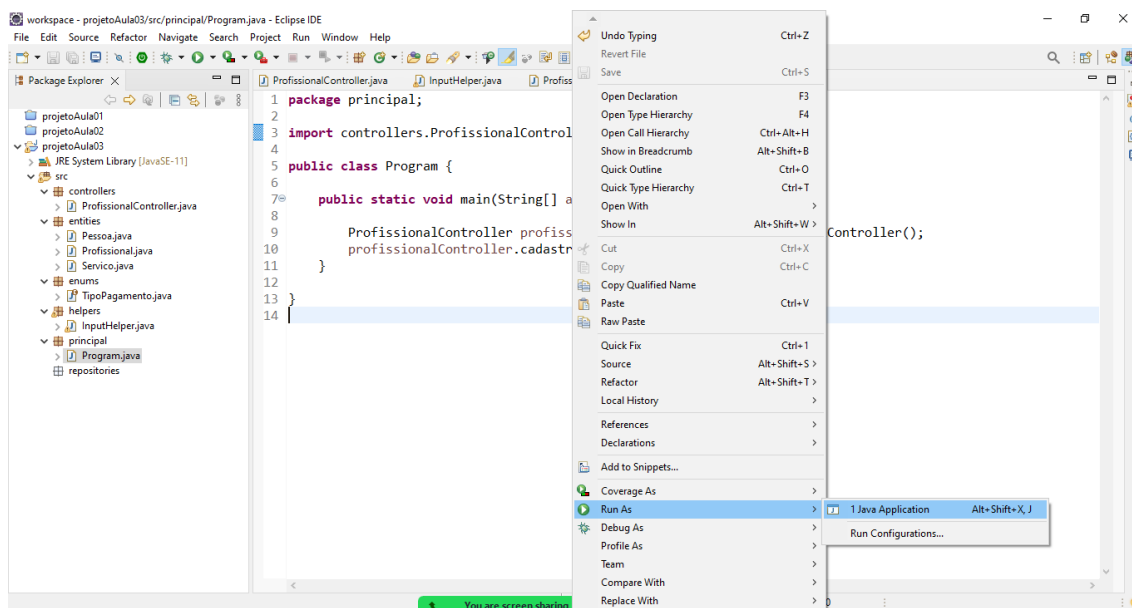
```
    public static void main(String[] args) {
```

```
        ProfissionalController profissionalController
            = new ProfissionalController();
```

```
        profissionalController.cadastrarProfissional();
```

```
    }
```

```
}
```



*** 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;

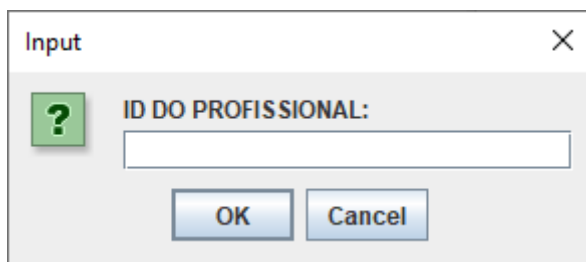
import javax.swing.JOptionPane;

public class InputHelper {

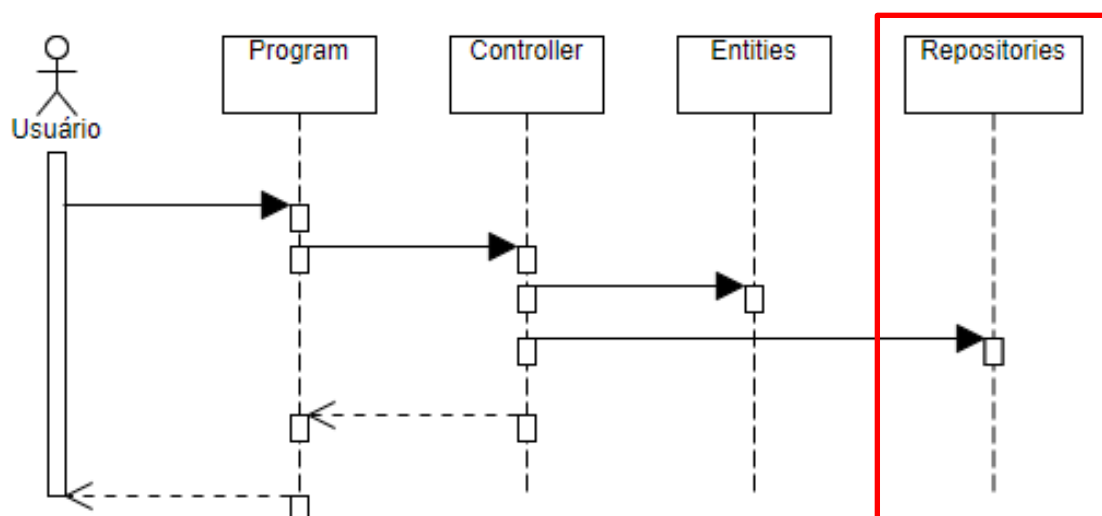
    //método para fazer a captura de
    //um dado informado pelo usuário
    public String inputData(String message) {

        String valor = JOptionPane.showInputDialog(message);
        return valor;
    }
}
```

Testando:

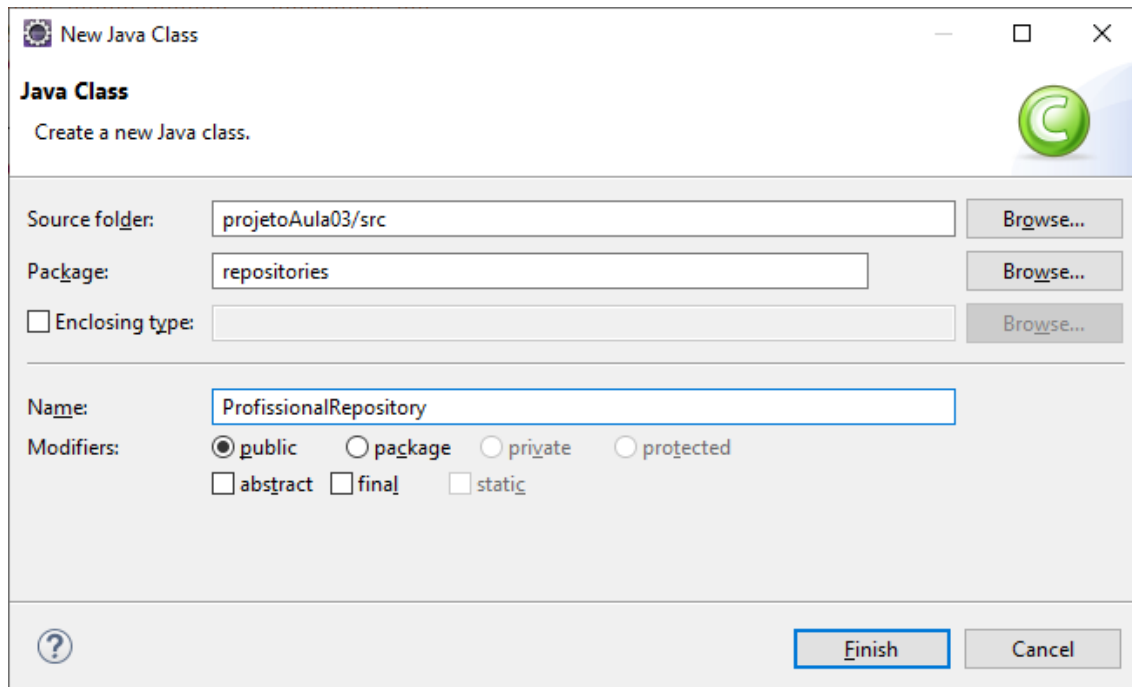


Construindo a camada de repositório do sistema:



/Repositories/ProfissionalRepository.java

Classe para persistência de dados do profissional.



```
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.....: "
```

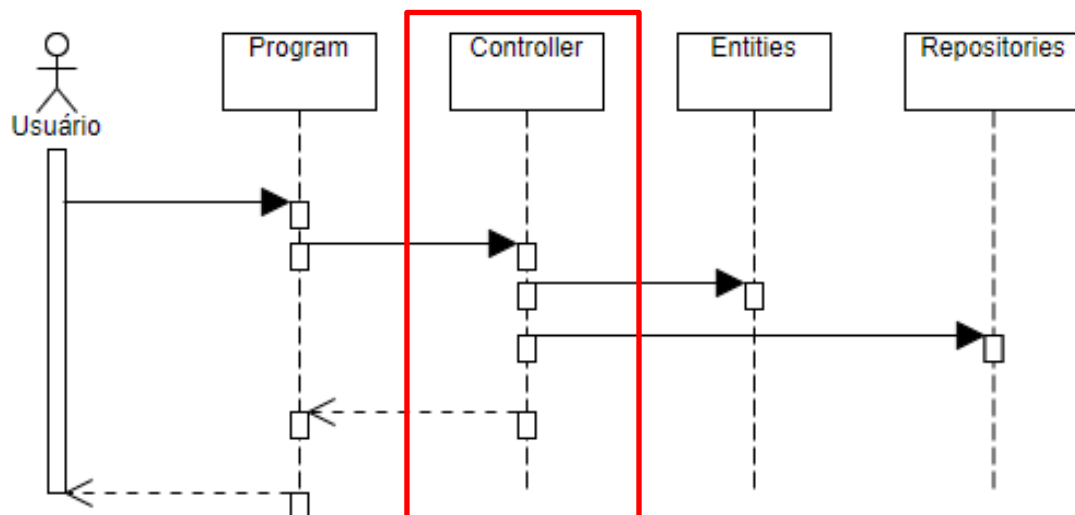
```

        + 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:



```

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 {

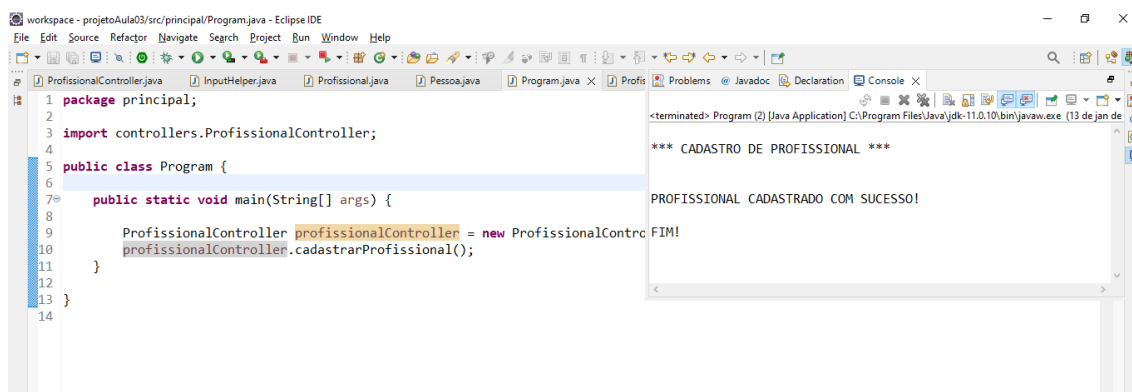
    ProfissionalRepository profissionalRepository
        = new ProfissionalRepository();

    profissionalRepository.exportarDados(profissional);

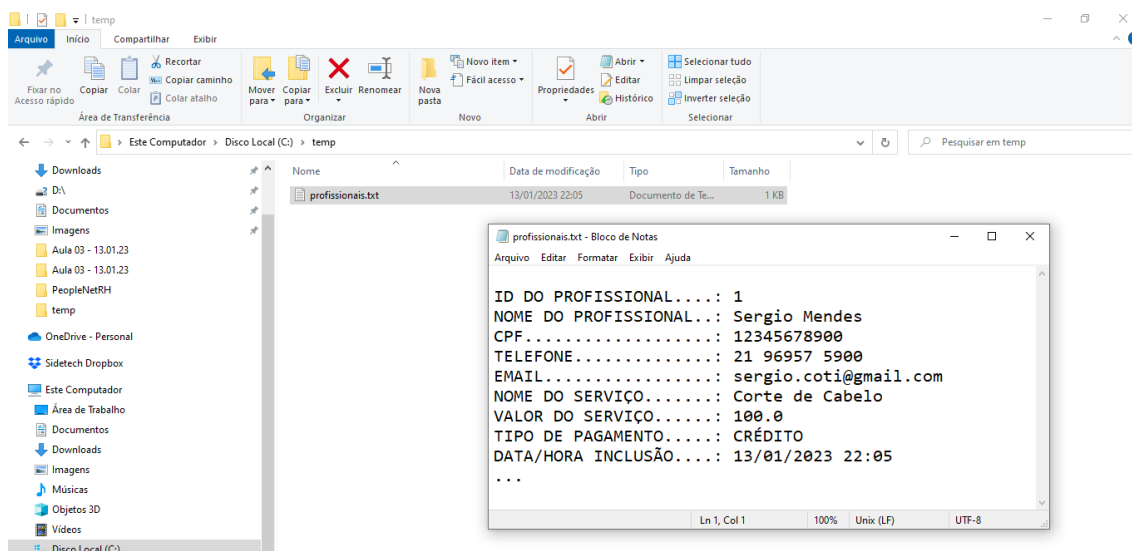
    System.out.println
        ("\nPROFISSIONAL CADASTRADO COM SUCESSO!");
}
catch(Exception e) {
    System.out.println("\nERRO: " + e.getMessage());
}

System.out.println("\nFIM!");
}
}
```

Executando:



Arquivo gerado:



```
profissionais.txt - Bloco de Notas
Arquivo  Editar  Formatar  Exibir  Ajuda

ID DO PROFISSIONAL.....: 1
NOME DO PROFISSIONAL...: Sergio Mendes
CPF.....: 12345678900
TELEFONE.....: 21 96957 5900
EMAIL.....: sergio.coti@gmail.com
NOME DO SERVIÇO.....: Corte de Cabelo
VALOR DO SERVIÇO.....: 100.0
TIPO DE PAGAMENTO.....: CRÉDITO
DATA/HORA INCLUSÃO.....: 13/01/2023 22:05
...
ID DO PROFISSIONAL.....: 2
NOME DO PROFISSIONAL...: Ana Paula
CPF.....: 123.456.321-00
TELEFONE.....: 21 98765 1234
EMAIL.....: anapaula@gmail.com
NOME DO SERVIÇO.....: Manicure
VALOR DO SERVIÇO.....: 80.0
TIPO DE PAGAMENTO.....: PIX
DATA/HORA INCLUSÃO.....: 13/01/2023 22:07
...

Ln 1, Col 1    100%    Unix (LF)    UTF-8
```

- ▼ projetoAula03
 - > JRE System Library [JavaSE-11]
 - ▼ src
 - ▼ controllers
 - > ProfissionalController.java
 - ▼ entities
 - > Pessoa.java
 - > Profissional.java
 - > Servico.java
 - ▼ enums
 - > TipoPagamento.java
 - ▼ helpers
 - > InputHelper.java
 - ▼ principal
 - > Program.java
 - ▼ repositories
 - > ProfissionalRepository.java