**NOTAS**

**Desafio: Aprenda na Prática Programação Orientada a Objetos**

Resumo de modificações feitas em um fork do projeto disponibilizado pela professora Camila Cavalcante em: https://github.com/cami-la/desafio-poo-dio

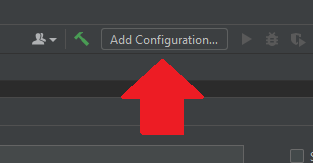
**2022/01**

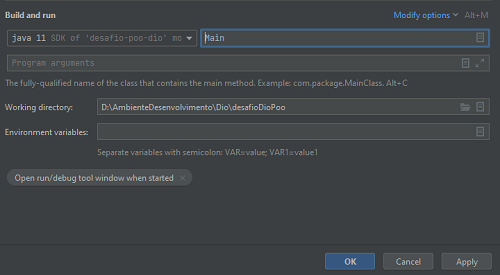
# Criar projeto base

***Passo 1***: Acesse o link: <https://github.com/cami-la/desafio-poo-dio>, crie um fork, depois faça um clone do seu for usando o comando

git clone https://github.com/biagolini/DesafioDioPoo

***Passo 2***: Defina a configuração para rodar seu projeto na IDE IntelliJ.





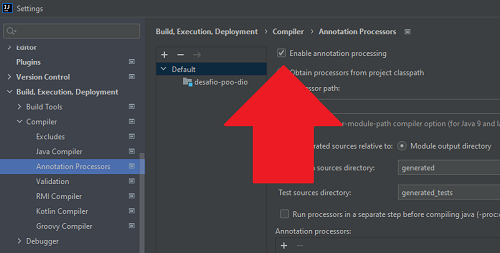
***Passo 3***: Implementar o Lombok.

Faça download do arquivo .jar do Lombok, disponível no link <https://projectlombok.org/download>. Salve no local de sua preferência. Em seguida configure sua IDE para habilitar o uso das anotações do Lombok, bem como o

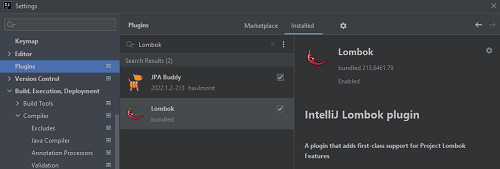
***Passo 3.1***: No Eclipse, veja referências: <https://youtu.be/OhD5xvj4kQw> + <https://youtu.be/745W-dng3wk>

***Passo 3.2***: No IntelliJ.

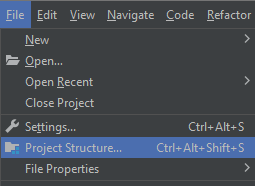
Certifique-se que o processamento de anotações está habilitado: Veja: <https://stackoverflow.com/a/27430992/4678899>



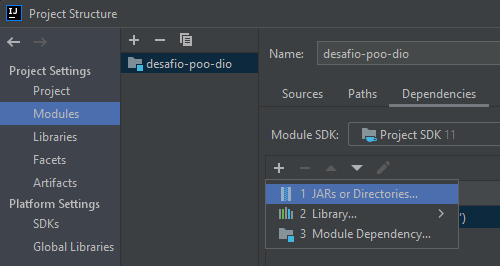
Em settings, busque pelo Plugin do Lombok e ative o mesmo.

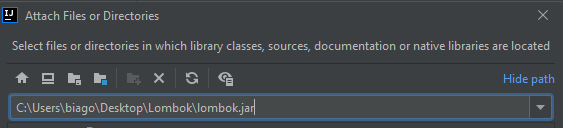


File → Project Structure ...



Modules → + JARs or Directories.. → Selecione o Arquivo Jar que você fez download no inicio desse passo do tutorial → Ok → Ok.





# Deixar o código mais “limpo” com o Lombok

***Passo 1***: Para facilitar a leitura do código, vamos utilizar as anotações abaixo para deixar nosso código mais consigo, ou seja, reduz a verborragia.

import lombok.Getter;  
import lombok.NoArgsConstructor;  
import lombok.Setter;  
  
@Getter  
@Setter  
@NoArgsConstructor

***Passo 2***: Postar parcial no git

git add .

git commit -m "Parte 01 Lombok"

git push origin master

# Implementar seleção de comandos

***Passo 1***: Aqui, criamos um método dentro de uma nova classe, de forma que por dentro da função Main, vamos chamar apenas esse método.

src/br/com/dio/desafio/dominio/Comandos.java

package br.com.dio.desafio.dominio;  
  
import lombok.NoArgsConstructor;  
  
import java.time.LocalDate;  
import java.util.ArrayList;  
import java.util.List;  
import java.util.Scanner;  
  
@NoArgsConstructor  
public class Comandos {  
 Scanner scanner = new Scanner(System.*in*);  
  
 public void corrida(){  
 Integer opcao = 0;  
 do{  
 System.*out*.printf("Escolha uma opção:\n" +  
 "\t1) Instanciar um Curso;\n"+  
 "\t2) Instanciar uma Mentoria;\n"+  
 "\t3) Instanciar um Bootcamp;\n"+  
 "\t4) Instanciar Devs;\n" +  
 "\t5) Simular Bootcamp com inscritos e calculos de Xp;\n" +  
 "\t0) Finalizar/Sair;\n" );  
 opcao = scanner.nextInt();  
 scanner.nextLine();  
  
 switch (opcao) {  
 case 1:  
 Curso cuso = operacao1();  
 System.*out*.println("Os dados do seu curso são:");  
 System.*out*.println(cuso.toString());  
 break;  
 case 2:  
 Mentoria mentoria = operacao2();  
 System.*out*.println("Os dados da mentoria são:");  
 System.*out*.println(mentoria.toString());  
 break;  
 case 3:  
 Bootcamp bootcamp = operacao3();  
 System.*out*.println("Os dados do Bootcamp são:");  
 System.*out*.println(bootcamp.toString());  
 break;  
 case 4:  
 List<Dev> devs = operacao4();  
 System.*out*.println("A lista de devs são:");  
 for (int i = 0; i < devs.size(); i++) {  
 System.*out*.println(devs.get(i).getNome());  
 }  
 break;  
 case 5: this.operacao5(); break;  
 case 0: System.*out*.printf("Obrigado por utilizar nosso programa, até logo;\n" ); break;  
 default:System.*out*.printf("Opção inválida;\n");  
 }  
  
 System.*out*.printf("\n\nDigite enter para proseguir;\n");  
 scanner.nextLine();  
 String str= scanner.nextLine();  
 } while (opcao!=0);  
 }  
  
 private Curso operacao1() {  
 Curso curso = new Curso();  
 String titulo, descricao;  
 Integer cargaHoraria;  
  
 System.*out*.println("Informe o titulo do curso:");  
 titulo= scanner.nextLine();  
 curso.setTitulo(titulo);  
  
 System.*out*.println("Informe a descrição do curso:");  
 descricao= scanner.nextLine();  
 curso.setDescricao(descricao);  
  
 System.*out*.println("Informe a carga horaria do curso:");  
 cargaHoraria = scanner.nextInt();  
 scanner.nextLine();  
 curso.setCargaHoraria(cargaHoraria);  
  
 return curso;  
 }  
  
  
 private Mentoria operacao2() {  
 Mentoria mentoria = new Mentoria();  
 String titulo, descricao;  
  
 System.*out*.println("Informe o titulo da mentoria:");  
 titulo= scanner.nextLine();  
 mentoria.setTitulo(titulo);  
  
 System.*out*.println("Informe a descrição da mentoria:");  
 descricao= scanner.nextLine();  
 mentoria.setDescricao(descricao);  
  
 mentoria.setData(LocalDate.*now*());  
  
 return mentoria;  
  
 }  
  
 private Bootcamp operacao3() {  
 Integer nCursos, nMentorias, i;  
  
 System.*out*.printf("Quantos Cursos você irá inscrever nesse Bootcamp?\n");  
 nCursos = scanner.nextInt();  
 scanner.nextLine();  
  
 System.*out*.printf("Quantos Mentorias você irá inscrever nesse Bootcamp?\n");  
 nMentorias = scanner.nextInt();  
 scanner.nextLine();  
  
 System.*out*.println("# 01: INFORME DADOS DE CURSO:");  
 List<Curso> cursos = new ArrayList<>();  
  
 for (i=0; i<nCursos; i++ ){  
 System.*out*.printf("\t --- %dº curso ---\n",i+1);  
 Curso atual = this.operacao1();  
 cursos.add(atual);  
 }  
  
 System.*out*.println("# 02: INFORME DADOS DE MENTORIA:");  
 List<Mentoria> mentorias = new ArrayList<>();  
 for (i=0; i<nMentorias; i++ ){  
 System.*out*.printf("\t --- %dº mentoria ---\n",i+1);  
 Mentoria atual = this.operacao2();  
 mentorias.add(atual);  
 }  
  
 System.*out*.println("# 03: INFORME DADOS DO BOOTCAMP:");  
 Bootcamp bootcamp = new Bootcamp();  
 String titulo, descricao;  
  
 System.*out*.println("Informe o titulo do bootcamp:");  
 titulo= scanner.nextLine();  
 bootcamp.setNome(titulo);  
  
 System.*out*.println("Informe a descrição do bootcamp:");  
 descricao= scanner.nextLine();  
 bootcamp.setDescricao(descricao);  
  
 for (i=0; i<nCursos; i++ ){  
 bootcamp.getConteudos().add(cursos.get(i));  
 }  
  
 for (i=0; i<nMentorias; i++ ){  
 bootcamp.getConteudos().add(mentorias.get(i));  
 }  
  
 return bootcamp;  
 }  
  
 private List<Dev> operacao4() {  
 String nome;  
 Integer nDevs, i;  
  
 System.*out*.printf("Quantos Devs você irá inscrever?\n");  
 nDevs = scanner.nextInt();  
 scanner.nextLine();  
  
 List<Dev> devs = new ArrayList<>();  
 for (i=0; i<nDevs; i++ ){  
 System.*out*.printf("Informe um nome:\n");  
 nome= scanner.nextLine();  
 devs.add(new Dev());  
 devs.get(i).setNome(nome);  
 }  
 return devs;  
 }  
  
 private void operacao5() {  
 System.*out*.println("# 03: INFORME DADOS DO BOOTCAMP:");  
 Bootcamp bootcamp = this.operacao3();  
 String titulo, descricao;  
  
 System.*out*.println("# 04: INFORME DADOS DO DEV:");  
 List<Dev> devs = this.operacao4();  
  
 System.*out*.println("# 05: PROCESSAR XP:");  
  
 Integer nDevs, i;  
  
 nDevs = devs.size();  
  
 for (i=0; i<nDevs; i++ ){  
 System.*out*.printf("Para inscrever o Dev:\n" + devs.get(i).getNome() + "no Bootcamp, digite 'S':\n");  
 String opcao = scanner.nextLine().toUpperCase();  
  
 if(opcao.equals("S")){  
 devs.get(i).inscreverBootcamp(bootcamp);  
 devs.get(i).progredir();  
 System.*out*.println("XP = " + devs.get(i).calcularTotalXp());  
 }  
 }  
 }  
}

***Passo 2***: Desenvolver função main.

src/Main.java

import br.com.dio.desafio.dominio.Comandos;  
  
public class Main {  
  
 public static void main(String[] args) {  
 Comandos comandos = new Comandos();  
 comandos.corrida();  
 }  
  
}

***Passo 3***: Postar parcial no git

git add .

git commit -m "Parte 02 Implementação de menu de opções"

git push origin master