C206 - POO

Arthur Openheimer

Inatel

Informações Gerais

- Atendimento → Terça-feira 17:30-19:10, prédio 1, sala 19
- Email → arthur.openheimer@ges.inatel.br
- Github →https://github.com/ArthurOpenheimer/C206-Monitoria

Inatel

Exceptions

Em alguns casos, o Java pode nos apresentar erros que travam completamente a execução do nosso programa

Nessas situações, precisamos lidar com esses erros de alguma forma, por exemplo com os arrays, onde sempre verificamos se o objeto existe antes de fazer uma operação

```
for(Item lista : listaDeCompra) {
    if(lista != null) {
        System.out.println(lista.nome);
    }
}
```

Exceptions

O problema é que essa é uma solução para resolver um NullPointerException, um um cenário específico

Para lidarmos com erros de uma forma geral, utilizaremos Exceptions e suas técnicas

```
Exception in thread "main" java.lang.NullPointerException
        at Main.main(Main.java:10)
```

Inatel

Try Catch

Usaremos a estrutura do try-catch para isolar os erros e adicionar um tratamento para o erro

```
for(Item lista : listaDeCompra) {
    try{
        System.out.println("Nome: " + lista.nome + " Preço: " + lista.preco);
    }
    catch(NullPointerException e){
        System.out.println("Erro: " + e);
    }
}
```

```
Erro: java.lang.NullPointerException
Erro: java.lang.NullPointerException
Erro: java.lang.NullPointerException
```

Inatel

Criando exceções

Além das exceções já existentes do java, podemos criar as nossas próprias exceções

Vamos imaginar o cenário de um banco: Um cliente tenta sacar um valor maior do que o saldo existente em sua conta

Criando exceções

Para criar a exceção, criamos um arquivo com o nome da exceção e herdamos da classe Exception

Depois, criamos seu método construtor, que irá servir para mostrar a mensagem do erro quando ele acontecer

No exemplo abaixo, o arquivo SaldoInsuficienteException:

```
public class SaldoInsuficienteException extends Exception {

    public SaldoInsuficienteException(String message) {
        super(message);
    }
}
```

Inatel

Criando exceções e usando Throws

Depois que criamos a exceção, ela está pronta para ser usada!

Quando queremos que a exceção seja lançada (durante um método, por exemplo) utilizamos “throws” no método, e indicamos qual exceção será lançada

Depois disso basta utilizar “throw” para que a exceção seja mostrada no momento desejado

```
public void sacar(float quantia) throws SaldoInsuficienteException {
    if(quantia <= saldo) {
        this.saldo -= quantia;
    }else {
        throw new SaldoInsuficienteException(message:"Saldo Insuficiente");
    }
}
```

Inatel

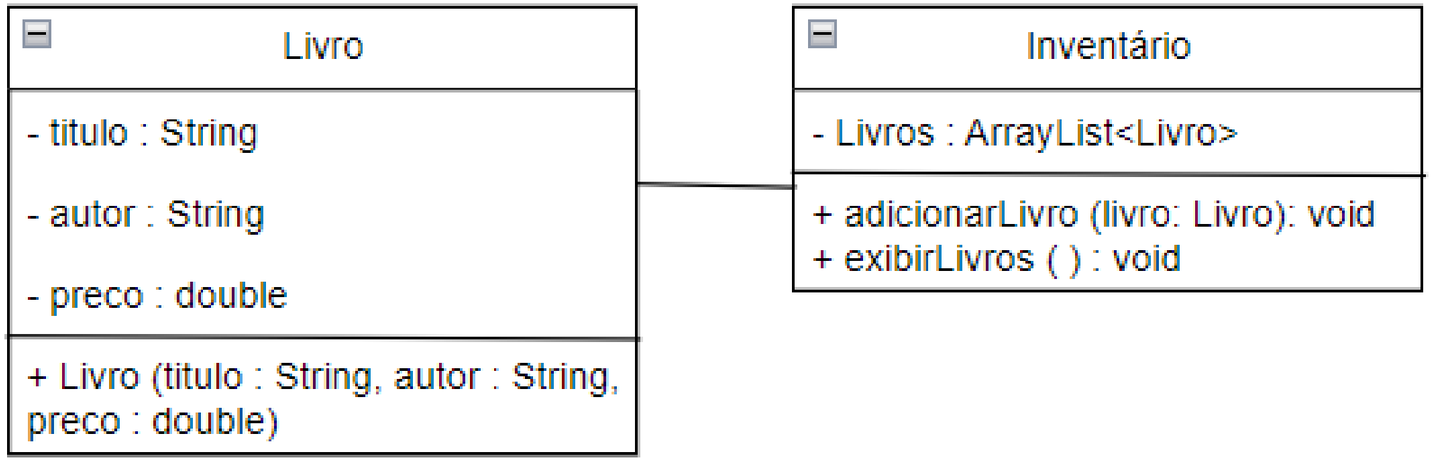
Criando exceções e usando Throws

Então, no momento em que iremos executar o método que pode nos trazer o erro, nós devemos fazer um try-catch, que tenta capturar esse erro

```
System.out.println("Sacando mais do que tem");
try {
    conta.sacar(quantia:600);
} catch (SaldoInsuficienteException e) {
    System.out.println(e.getMessage());
}
```

```
Sacando mais do que tem
Saldo Insuficiente
```

Exercício



- A exceção **ExcecaoLivroInvalido** deve ser lançada no método **adicionarLivro**, quando:
  - O título do livro for uma string vazia “ ” (utilize o método .isEmpty());
  - O preço do livro for menor que zero;
- A mensagem da exceção deve ser diferente em cada caso, indicando o erro;
- Na main, faça dois try-catches, cada um criando um livro corretamente e um livro com um erro e os adicionando no arraylist depois, chame o método **exibirLivros.**

Inatel