



Domain Driven Design - Java

Prof. Gilberto Alexandre das Neves
profgilberto.neves@fiap.com.br

Java Exceptions

comando:
try - catch

Comando try-catch

Quando executamos códigos em Java, diversos erros podem acontecer: erros de codificação por parte do programador, erros devidos a entrada de valores incorretos entre outras possibilidades.

Quando erros ocorrem o Java normalmente irá interromper a execução do programa e exibir alguma mensagem de erro. O termo técnico para isso é: o Java irá “jogar uma exceção” (*throw an exception*).

Java try-catch

O comando `try` permite você definir um bloco de comandos que será testado em busca de erros enquanto é executado.

O comando `catch` permite definir um bloco de comandos a ser executado quando um erro é encontrado.

O comando `try` e `catch` “andam sempre juntos”.

Entrada de dados pelo teclado



A classe Scanner

Crie uma classe com o nome **EntradaDeDado** (no pacote **br.com.fiap**), digite o código abaixo e veja o comportamento da classe **Scanner** para leitura de dados via o teclado.

Dica: ao digitar o nome da classe **Scanner**, utilize a ferramenta de **autopreenchimento** da **IDE**, desta forma essa ferramenta também irá **importar** a biblioteca necessária.

```
5 ▶ public class EntradaDeDado {
6 ▶     public static void main(String[] args) {
7         int num1 = 0, num2 = 0, resultado = 0;
8         Scanner scan; // declaração do objeto scan
9         try {
10             scan = new Scanner(System.in); // instânciação do objeto scan
11             System.out.println("Digite dois números inteiros");
12             num1 = scan.nextInt();
13             num2 = scan.nextInt();
14             resultado = num1 + num2;
15             System.out.println("Valor 1: " + num1 + "\nValor2: " + num2 +
16                 "\nSoma dos valores: " + resultado);
17         } catch (Exception e) {
18             System.out.println("Formato de número incorreto");
19         }
20     }
```

I A classe Scanner

Scanner scan;

declara um objeto da classe Scanner.

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

cria uma instância da classe Scanner que permite entrada (*in*) de valores.

try { } catch { }

comando que permite realizar uma tentativa dos comandos (*try*) e caso ocorra algum erro ele é capturado (*catch*) e podemos configurar uma mensagem de erro para o usuário.

A classe JOptionPane

Crie uma classe com o nome **EntradaComJanela** (no pacote **br.com.fiap**), digite o código abaixo e veja o comportamento da classe **JOptionPane** para leitura de dados.

```
5 ▶ public class EntradaComJanela {
6 ▶     public static void main(String[] args) {
7         int num1 = 0, num2 = 0, resultado = 0;
8         String auxiliar;
9         try {
10             auxiliar = JOptionPane.showInputDialog("Digite um número");
11             num1 = Integer.parseInt(auxiliar);
12             auxiliar = JOptionPane.showInputDialog("Digite outro número");
13             num2 = Integer.parseInt(auxiliar);
14             resultado = num1 + num2;
15             JOptionPane.showMessageDialog(null, "Valor 1: " + num1 + "\nValor2: " + num2 + "\nSoma dos valores: " + resultado);
16         } catch (Exception e) {
17             JOptionPane.showMessageDialog(null, "Formato de número incorreto");
18         }
19     }
20 }
```

A classe JOptionPane

JOptionPane.showInputDialog("mensagem")

permite a leitura de valores via caixa de texto (**detalhe**: toda informação digitada aqui é do tipo **String**).

Integer.parseInt()

permite a conversão da **String** indicada em um valor numérico inteiro (temos também: **Float.parseFloat()** e **Double.parseDouble()** para conversão de números reais; **Boolean.parseBoolean()** para conversão de valores lógicos, entre outros).

JOptionPane.showMessageDialog(null**, "mensagem")**

permite exibir resultados (texto, valores, etc) via caixa de texto.

Praticando...

Desenvolva os exercícios propostos abaixo, cada um em uma classe diferente em duas versões (uma usando a classe **Scanner** e outra usando a classe **JOptionPane**).

1. Monte um programa que peça para o usuário digitar as notas das 4 provas (prova1, prova2, prova3 e prova4) e exiba a média aritmética simples.
2. Monte um programa que peça para o usuário digitar o ano atual e o ano de seu nascimento exiba ao final a idade deste usuário.
3. Monte um programa que peça para o usuário digitar o valor do raio de um círculo e exiba a área deste círculo (**lembrete**: $\text{área do círculo} = \text{PI} * \text{raio}^2$)

1. “Permite a leitura de valores via caixa de texto” de qual classe e método estamos falando?
 - a) JOptionPane.showMessageDialog(null, “mensagem”)
 - b) JOptionPane.showMessageDialog(“mensagem”)
 - c) JOptionPane.showInputDialog(null, “mensagem”)
 - d) JOptionPane.showInputDialog(“mensagem”)

2. Analise o trecho de código abaixo e escolha a alternativa que melhor representa a próxima linha código.

```
String aux;  
double estacao;  
aux = JOptionPane.showInputDialog("Escolha a estação:");
```

- a) **estacao** = aux;
- b) **estacao** = Integer.parseInt(aux);
- c) **estacao** = Boolean.parseBoolean(aux);
- d) **estacao** = Double.parseDouble(aux);



Java como programar. Paul Deitel e Harvey Deitel. Pearson, 2011.

Java 8 – Ensino Didático : Desenvolvimento e Implementação de Aplicações. Sérgio Furgeri. Editora Érica, 2015.

Até breve!

1. “Permite a leitura de valores via caixa de texto” de qual classe e método estamos falando?
 - a) JOptionPane.showMessageDialog(null, “mensagem”)
 - b) JOptionPane.showMessageDialog(“mensagem”)
 - c) JOptionPane.showInputDialog(null, “mensagem”)
 - ☒ d) JOptionPane.showInputDialog(“mensagem”)
2. Analise o trecho de código abaixo e escolha a alternativa que melhor representa a próxima linha código.

```
String aux;  
double estacao;  
aux = JOptionPane.showInputDialog("Escolha a estação:");
```

- a) `estacao = aux;`
- b) `estacao = Integer.parseInt(aux);`
- c) `estacao = Boolean.parseBoolean(aux);`
- ☒ d) `estacao = Double.parseDouble(aux);`