Programação em Java

16-08-2021

Tecnologia JAVA

Algoritmos

Celina Lopes

Sumário

- exceções
- Try…catch

Exceções em java

A linguagem Java tem um mecanismo especial para o tratamento de erros que possam ocorrer em tempo de execução do programa. Diferente de outras linguagens, o surgimento de um erro causa a interrupção imediata do programa, mas, em Java podemos tratar esta situação de erro de forma adequada evitando, assim, a interrupção do programa.

Uma exceção, é uma classe em Java representada na sua forma mais genérica pela classe java.lang.Exception, logo todas as exceções que ocorram ao longo da execução do programa podem ser tratadas como objetos do tipo Exception.

Uma característica importante sobre exceções é que, pelo facto delas poderem ocorrer em qualquer momento, elas são literalmente "lançadas" (de volta) para o fluxo de execução e chamada das classes.

Exceções em java

Bloco try … catch

Como a exceção é lançada na cadeia de classes do sistema, a qualquer momento é possível "apanhar" essa exceção e dar-lhe o tratamento adequado.

Para se fazer este tratamento, é necessário indicar um determinado bloco de código e que uma possível exceção será tratada de uma determinada forma,

exemplo deste novo bloco:

Exceções em java

Bloco try ... catch

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

      /* bloco de código no qual uma
       * exceção pode acontecer.
       */
    } catch (Exception ex) {

      /* bloco de código no qual uma
       * exceção do tipo "Exception" será tratada.
       */
    }
  }
}
```

Exceções em java

Bloco try ... catch

O bloco try, é a parte do código em que uma exceção é esperada e o bloco catch, correspondente ao bloco try, prepara-se para "apanhar" a exceção ocorrida e dar-lhe o tratamento necessário.
Uma vez declarado um bloco try, a declaração do bloco catch torna-se obrigatória.

Na linha 8 o bloco catch declara receber um objeto do tipo Exception, lembremo-nos do conceito de herança, todas as exceções do Java são classes filhas de Exception.

Importante: é que quando uma exceção ocorre, as outras linhas de código deixam de ser executadas até encontrar o bloco catch que irá tratar a exceção.

Exceções em java

Bloco try ... catch

```
/**
 * Classe que demonstra o uso do try / catch.
 */
public class ExemploTryCatch {
  public static void main(String[] args) {
    Scanner s = new Scanner(System.in);
    try {
      System.out.print("Digite um valor inteiro..:");
      int numero1 = s.nextInt();
      System.out.print("Digite um valor inteiro..:");
      int numero2 = s.nextInt();

      System.out.println(numero1+ " + " + numero2 + " = " + (numero1+numero2));
    } catch (Exception ex) {
      System.out.println("ERRO - Valor digitado nao e um numero inteiro!");
    }
  }
}
```

Exceções em java

Bloco try … catch

Neste código, caso aconteça um erro na linha 9, as linhas de 10 a 13 não seriam executadas, retornando então o código a ser executado apenas a partir da linha 15 (trecho correspondente ao bloco catch).

Uma vez que uma exceção foi tratada por um bloco catch, a execução do programa segue normalmente.

Exceções em java

Palavra-chave throw

Também é possível definirmos uma exceção nalguma situação especifica, como num login em que o user digita incorretamente a password.
Para realizarmos tal tarefa é necessária a utilização da palavra-chave throw da seguinte forma:

throw new << Exceção desejada >>();

Exceções em java

```
package material.excecao;

import java.util.Scanner;

/**
 * Classe utilizada para demonstrar o uso da palavra chave throw,
 * utilizada quando queremos lançar uma exceção.
 */
public class ExemploThrow {
  public static final String SENHASECRETA = "123456";

  public static void main(String[] args) {
    try {
      Scanner s = new Scanner(System.in);
      System.out.print("Digite a senha: ");
      String senha = s.nextLine();
      if(!senha.equals(SENHASECRETA)) {
        throw new Exception("Senha invalida!!!");
      }
      System.out.println("Senha correta!!!\nBem vindo(a)!!!");
    } catch (Exception ex) {
      System.out.println(ex.getMessage());
    }
  }
}
```

Exceções em java

Assim que a palavra-chave throw for utilizada, podemos tentar tratar a exceção no bloco try / catch ou lançar essa exceção nesse método.
Observemos também que a palavra reservada new foi utilizada, uma vez que a exceção é um novo objeto que deve ser criado na memória.
Isto é necessário para que a exceção possa ser lançada por toda a pilha de execução até que seja devidamente tratada.

Fazer o exemplo no netbeans.

Exceções em java

Alguns métodos importantes da classe Exception:

printStackTrace() - Imprime no ecrã a pilha de execução. Muito comum para auxiliar no diagnóstico de erros.

getMessage() - Retorna uma String com a mensagem contida na exceção.

getClass() - Retorna uma String com o nome da classe da exceção.

Exceções em java

Bloco finnaly

A palavra-chave finally representa um bloco de código que será sempre executado, independentemente da ocorrência da exceção.

```
public class ExemploFinally {
  public static void main(String[] args) {
    Scanner s = new Scanner(System.in);
    try {
      int dividendo, divisor;
      System.out.print("Digite o valor do
dividendo: ");
      dividendo = s.nextInt();
      System.out.print("Digite o valor do
divisor: ");
      divisor = s.nextInt();
```

Exceções em java

Bloco finnaly

```
 if(divisor == 0) {
        throw new Exception("Não é permitido fazer uma divisao por zero!");
      }

      System.out.println(dividendo+" / "+divisor+" = "+(dividendo / divisor));
    } catch (Exception ex) {
      System.out.println("Erro: " + ex.getMessage());
    } finally {
      System.out.println("Bloco Finally.");
    }
  }
}
```

Exceções em java

Bloco finnaly

```
 if(divisor == 0) {
        throw new Exception("Não é permitido fazer uma divisao por zero!");
      }

      System.out.println(dividendo+" / "+divisor+" = "+(dividendo / divisor));
    } catch (Exception ex) {
      System.out.println("Erro: " + ex.getMessage());
    } finally {
      System.out.println("Bloco Finally.");
    }
  }
}
```

Exceções em java

Bloco finnaly

No exemplo anterior, a mensagem “Bloco Final”, sempre será exibida, ocorrendo ou não um Exception

Exceções em java

Bloco finnaly

No exemplo anterior, a mensagem “Bloco Final!”, é sempre exibida, ocorrendo ou não um Exception

Se digitarmos zero no valor do divisor será lançada uma exceção, notemos que o bloco Finally será executado mesmo que ocorra alguma exceção

Exceções em java

Bloco finnaly

No exemplo anterior, a mensagem “Bloco Final!”, é sempre exibida, ocorrendo ou não um Exception

Se digitarmos zero no valor do divisor será lançada uma exceção, notemos que o bloco Finally será executado mesmo que ocorra alguma exceção

Este bloco é muito utilizado com as ligações à base de dados, ficheiros…

Exceções em java

Palavra-chave throws

Caso algum método precise lançar uma exceção, mas nós não queremos tratá-la, e retorna-la para o objeto que fez a chamada ao método que lançou a exceção, basta utilizar a palavra chave throws no final da assinatura do método.

Quando utilizamos o throws precisamos também informar qual ou quais as exceções a ser lançadas.

Exceções em java

```
package material.excecao;

import java.util.Scanner;

/**
 * Classe utilizada para demonstrar o uso da palavra chave throws,
 * deixando para quem chamar o método tratar alguma possivel exceção.
 */
public class ExemploThrows {
  public static void main(String[] args) {
    Scanner s = new Scanner(System.in);
    try {
      ExemploThrows et = new ExemploThrows();

      System.out.print("Digite o valor do dividendo: ");
      double dividendo = s.nextDouble();

      System.out.print("Digite o valor do divisor: ");
      double divisor = s.nextDouble();
```

Exceções em java

```
double resultado = et.dividir(dividendo, divisor);

      System.out.println("O resultado da divisao é: " + resultado);
    } catch (Exception ex) {
      System.out.println(ex.getMessage());
    }
  }

29:  public double dividir(double dividendo, double divisor) throws
Exception {
    if(divisor == 0) {
      throw new Exception("Nao e permitido fazer uma divisao por
zero!");
    }

    return dividendo / divisor;
  }
}
```

Exceções em java

Notar que na linha 29 declaramos na assinatura do método que ele pode ou não lançar uma exceção do tipo *java.lang.Exception*

E o método *public static void main(String[] args)*

será responsável por tratar alguma exceção que o método

dividir(double dividendo, double divisor)

possa lançar.

Exceções em java

Criando a nossa exceção

Na linguagem Java podemos criar as nossas próprias exceções, normalmente fazemos isso para garantir que os nossos métodos funcionem corretamente, desta forma podemos lançar exceções com mensagens de fácil entendimento para os users, ou para facilitar o entendimento do problema.

Vamos criar uma exceção chamada *ErroDivisao* que será lançada quando ocorrer uma divisão incorreta, para isso precisamos criar a classe filha de Exception.

Exceções em java

Criar a nossa exceção

Na linguagem Java podemos criar as nossas próprias exceções, normalmente fazemos isso para garantir que os nossos métodos funcionem corretamente, desta forma podemos lançar exceções com mensagens de fácil entendimento para os users, ou para facilitar o entendimento do problema.

Vamos criar uma exceção chamada *ErroDivisao* que será lançada quando ocorrer uma divisão incorreta, para isso precisamos criar a classe filha de Exception.

```
/**
 * Exceção que deve ser lançada quando uma divisão é invalida.
 */
public class ErroDivisao extends Exception {
  public ErroDivisao() {
    super("Divisao invalida!!!");
  }
}
```