GERMINARE
Escola de Negócios

Desenvolvimento II

Tratamento de Exceções

Tratamento de Exceções em Java

Germinare Tech

Marcelo Modolo

GERMINARE
Escola de Negócios

GerminaTECH

Tratamento de Exceções em Java

Marcelo Modolo

GERMINARE
Escola de Negócios

Erros e Exceções em Java

Erros

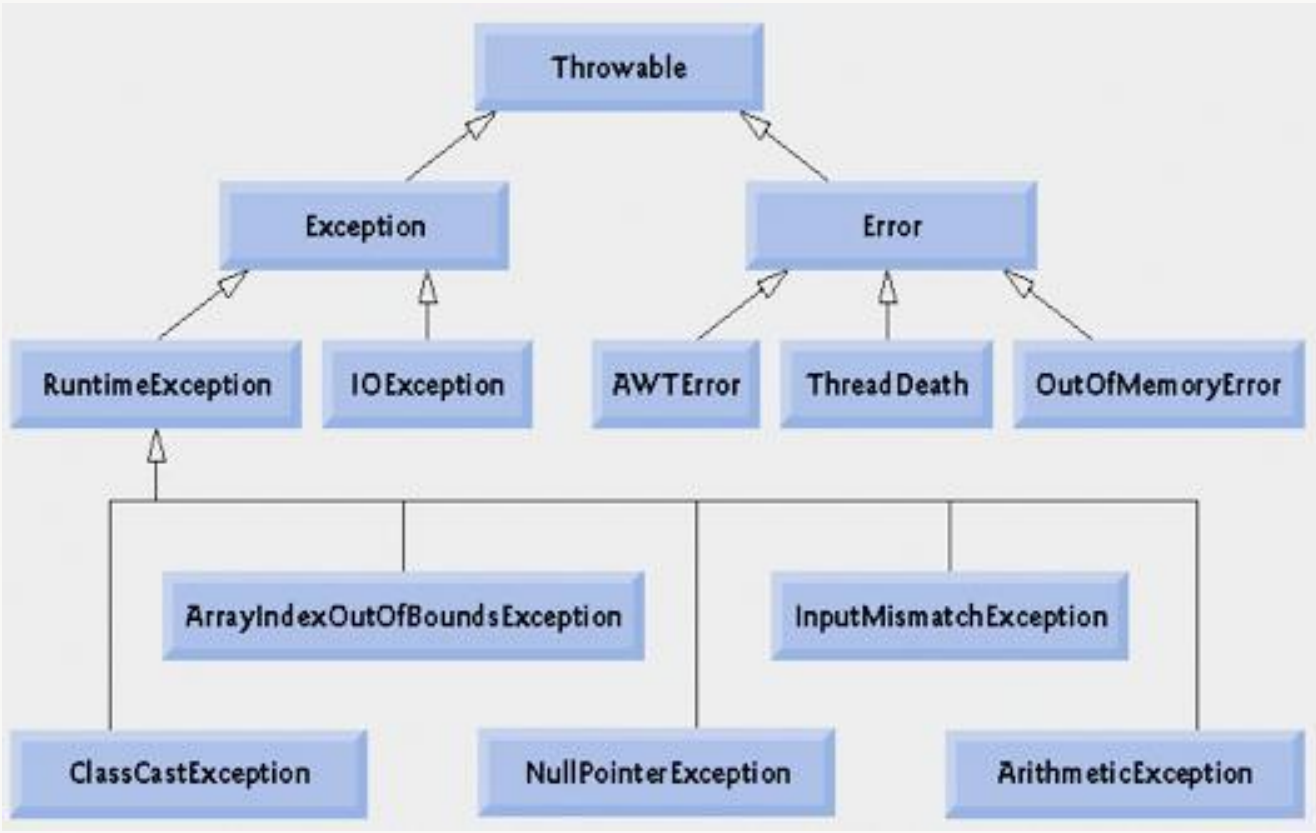
- Problemas do programa
- Não são tratados
- Programa é interrompido

Exceções

- Exceções em tempo de execução
- São tratadas
- Programa não é interrompido

GERMINARE
Escola de Negócios

Hierarquia dos Erros e Exceções



Fonte: http://www.devmedia.com.br/trabalhando-com-excecoes-em-java/27601

GERMINARE
Escola de Negócios

Classe Error

- Usada pela Máquina Virtual Java (Java Virtual Machine – JVM) para indicar se existe problemas no programa
- Não permite tratamento e a execução do programa é interrompida
- Exemplos:
  - StackOverflowError
  - OutOfMemoryError

GERMINARE
Escola de Negócios

Exceções

- Na linguagem Java existe um mecanismo que permite o tratamento de exceções em tempo de execução do programa
- O Java possibilita tratar e as exceções para evitar a interrupção do programa
- As exceções são “lançadas” de volta para o fluxo de execução do programa

GERMINARE
Escola de Negócios

Classe Exception

- A classe Exception é a forma mais genérica de representar uma exceção em Java
- Todas as exceções em Java pode ser tratadas por objetos do tipo Exception
- Existem outras classes que tratam exceções específicas, mas todas são classes filhas da Exception, como por exemplo, IOException, SQLException, etc.

GERMINARE
Escola de Negócios

Subclasses de Exception

Deve-se usar a classe mais específica possível para tratar uma exceção:

- Porque assim é possível tratar essa exceção de forma mais precisa e eficaz
- Porque permite ter uma lógica de tratamento diferente para cada tipo de exceção
- Porque se a exceção capturada for muito geral, pode ser difícil determinar qual parte do código está lançando a exceção
- Porque permite que o código seja mais claro e legível, tornando-o mais fácil de manter e modificar no futuro

GERMINARE
Escola de Negócios

Exceções Previstas e Não Previstas

- Checked:
  - Exceções previstas pelo compilador
  - Obrigatoriamente os comandos devem estar dentro do bloco try/catch ou devem ser lançadas pelo comando throw
  - Exemplos: ClassNotFound Exception, SQLException, IOEXception
- Unchecked:
  - Exceções NÃO previstas pelo compilador por acontecer durante a execução do programa
  - Exemplos: ArithmeticException, IndexOutOfBoundsException

Exceções Externas e Internas

Externas

- Motivos externos ao programa
- Exemplos:
  - Tentar abrir um arquivo que não existe
  - Tentar fazer consulta a um banco de dados que não está disponível
  - Tentar escrever algo em um arquivo sobre o qual não se tem permissão de escrita
  - Tentar conectar em servidor inexistente

Internas

- Erro de lógica durante a execução do programa
- Exemplos:
  - Tentar manipular um objeto que está com o valor nulo
  - Dividir um número por zero
  - Tentar manipular um tipo de dado como se fosse outro
  - Tentar utilizar um método ou classe não existentes

GERMINARE
Escola de Negócios

Bloco try/catch

- As exceções e os erros “lançados” de volta para o fluxo de execução do programa pode ser tratados pelo bloco try/catch
- O try delimita o trecho do código que pode ocorrer a exceção e o catch “pega” a exceção “lançada”
- Sintaxe:

```
try {
    // trecho onde a exceção pode ocorrer
} catch (Exception exc) {
    // comandos a executar se a exceção ocorrer
}
```

GERMINARE
Escola de Negócios

Qual exceção pode ocorrer aqui?

```
public static void main( String args[] ) {
        Scanner teclado = new Scanner( System.in );
        System.out.print("Digite a quantidade de pessoas: ");
        int quantPessoas = teclado.nextInt();
        System.out.print("Digite a quantidade de objetos por pessoa: ");
        int quantObjetos = teclado.nextInt();
        int total = quantPessoas * quantObjetos;
        System.out.print("Total = " + total);
}
```

Tratamento da Exceção InputMismatchException

```
public static void main( String args[] ) {
    Scanner teclado = new Scanner( System.in );
    try {
        System.out.print("Digite a quantidade de pessoas: ");
        int quantPessoas = teclado.nextInt();
        System.out.print("Digite a quantidade de objetos por pessoa: ");
        int quantObjetos = teclado.nextInt();
        int total = quantPessoas * quantObjetos;
        System.out.print("Total = " + total);
    } catch (InputMismatchException ime){
        System.out.println("Digite apenas números inteiros:  " + ime );
    }
}
```

GERMINARE
Escola de Negócios

Qual exceção vai ocorrer aqui?

```
public static void main(String args[]) {
        String frase = null;
        String novaFrase = frase.toUpperCase();
        System.out.println("Frase antiga: “ + frase);
        System.out.println("Frase nova: “ + novaFrase);
}
```

GERMINARE
Escola de Negócios

Tratamento da exceção NullPointerException

```
public static void main(String args[]) {
    String frase = null;
    String novaFrase;
    try {
        novaFrase = frase.toUpperCase();
    } catch(NullPointerException npe) {
        System.out.println(“A frase inicial está nula, não lhe foi atribuído um valor.");
        System.out.println("Exceção: " + npe.getMessage());
        frase = "Frase vazia";
        novaFrase = frase.toUpperCase();
    }
    System.out.println("Frase antiga: "+frase);
    System.out.println("Frase nova: "+novaFrase);
}
```

GERMINARE
Escola de Negócios

Qual exceção pode ocorrer aqui?

```
public static void main(String args[]) {
        Scanner teclado = new Scanner(System.in);
        System.out.print("Digite o CPF com somente números: ");
        String cpf = teclado.nextLine();
        long numCPF = Long.parseLong(cpf);
        System.out.println("CPF digitado corretamente");
}
```

GERMINARE
Escola de Negócios

Tratamento da exceção NumberFormatException

```
public static void main(String args[]) {
    Scanner teclado = new Scanner(System.in);
    try {
        System.out.print("Digite o CPF com somente números: ");
        String cpf = teclado.nextLine();
        long numCPF = Long.parseLong(cpf);
        System.out.println("CPF digitado corretamente");
    } catch (NumberFormatException nfe) {
        System.out.println("CPF deve ter somente números: " + nfe);
    }
}
```

Tratamento da exceção SQLException

```
// método para buscar registros na tabela
   public ResultSet buscar() {
      try {
         // Abrindo a conexão com o banco
         conectar();
         // Instanciando o objeto preparedStatement (pstmt)
         pstmt = conn.prepareStatement("SELECT * FROM DEPT ORDER BY DEPTNO");
         // Executando o comando sql do objeto preparedStatement e armazenando no ResultSet
         rs = pstmt.executeQuery();
         // Retornando o ResultSet
         return rs;
         } catch (SQLException e) {
            System.out.println("Exceção ao Buscar: " + e.getMessage());
         return null;
      }
   }
```

Tratamento de múltipla exceções

- O Java permite o tratamento de mais de uma exceção que tenha ocorrido no try
- Para cada exceção deve ser escrito um catch
- Exemplo:

```
try {
        // trecho onde a exceção pode ocorrer
} catch (SQLException exc1) {
        // comandos a executar se a exceção ocorrer
} catch (ArithmeticException exc2) {
        // comandos a executar se a exceção ocorrer
}
```

GERMINARE
Escola de Negócios

Quais exceções podem ocorrer aqui?

```
public static void main(String args[]) {
        System.out.print( “Digite o numerador: " );
        int numerador = teclado.nextInt();
        System.out.print( “Digite o denominador: " );
        int denominador = teclado.nextInt();
        int result = numerador/denominador;
        System.out.print( “Resultado = “ + result);
}
```

Exemplo de tratamento de duas exceções

```
public static void main( String args[] ) {
        Scanner teclado = new Scanner( System.in );
        try {
                System.out.print( “Digite o numerador: " );
                int numerador = teclado.nextInt();
                System.out.print( “Digite o denominador: " );
                int denominador = teclado.nextInt();
                int result = numerador/denominador;
                System.out.print( “Resultado = “ + result);
        } catch ( InputMismatchException ime ) {
                System.out.println(“Digite apenas números: “ + ime);
        } catch ( ArithmeticException ae ) {
                System.out.println(“O denominador não pode ser zero: “ + ae);
        }
}
```

Exemplo de Exceção com dois catch

```
// método para fazer a conexão com o BD
public boolean conectar() {
    try {
        // Informando qual driver de conexão será utilizado pelo DriveManager
        Class.forName("oracle.jdbc.driver.OracleDriver");
        // Criando a conexão com o BD
        conn = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "MARCELO", "MARCELO");
        return true;
    } catch (ClassNotFoundException e) {
        System.out.println("Exceção Classe não Existe: " + e.getMessage());
        return false;
    } catch (SQLException e) {
        System.out.println("Exceção Comando SQL: " + e.getMessage());
        return false;
    }
}
```

GERMINARE
Escola de Negócios

Bloco finally

- Pode ser usado com o try/catch, mas é opcional
- O trecho de código delimitado pelo finally será sempre executado:
  - Quando ocorrer ou não uma exceção
  - Quando existir o comando *return* no *try* e/ou no catch
- Sintaxe:

```
try {
    // trecho onde a exceção pode ocorrer
} catch (Exception exc) {
    // comandos a executar se a exceção ocorrer
} finally {
    // comandos sempre executados, com ou sem exceção
}
```

GERMINARE
Escola de Negócios

Exemplo de Exceção com Finally

```
public static String transformaFraseParaMaiuscula (String frase) {
    String novaFrase;
    try {
        novaFrase = frase.toUpperCase();
        return novaFrase;
    } catch(NullPointerException npe) {
        return "Frase vazia.";
    } finally {
        System.out.println( "O método transformaFraseParaMaiuscula foi executado." );
    }
}
```

GERMINARE
Escola de Negócios

Resumo do Bloco try/catch/finally

- Caso ocorra exceção no trecho de código do try:
  - O código depois da linha com erro não será executado
  - Os comandos do catch serão executados
- Caso NÃO ocorra exceção no trecho de código do try:
  - Todos os comandos do try serão executados
  - Os comandos do catch NÃO serão executados
- Nos dois casos os comandos do finally serão executados

Palavra chave throw

- Usada quando se quer “lançar” uma exceção em alguma situação específica
- A exceção deve ser lançada dentro de um bloco try e será pega no bloco catch correspondente
- Sintaxe: throw new <exceção>(<mensagem>);
- Exemplo:

```
try {
        throw new Exception(“Senha inválida!”);
} catch (Exception e) {
        System.out.println(e.getMessage());
}
```

GERMINARE
Escola de Negócios

Exemplo da palavra chave throw

```
public static void main(String[] args) {
        Scanner teclado = new Scanner(System.in);
        try {
            System.out.print("Digite a senha: ");
            int senha = teclado.nextInt();
            if (senha != 123456){
                throw new Exception("Senha inválida!");
            }
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
}
```

Palavra chave throws

- Caso um método não queira tratar uma exceção, pode “lançar” essa exceção para o objeto do método que o chamou
- A exceção pode ser tratada no método que a recebeu usando o bloco try/catch
- Exemplos:

```
public static double dividir (double num1, double num2) throws Exception {
        // código do método
}
public static boolean inserir( ) throws SQLException {
        // código do método
}
```

GERMINARE
Escola de Negócios

Exemplo de throws

```
public static void main(String[] args) {
        try{
            System.out.println(transformaFraseParaMaiuscula(null));
        } catch(NullPointerException npe) {
            System.out.println("Frase vazia");
'       }
}

public static String transformaFraseParaMaiuscula (String frase) throws NullPointerException {
        String novaFrase;
        novaFrase = frase.toUpperCase();
        return novaFrase;
}
```

Exemplo de throws e throw

```
public class Conexao {
    private Connection conn;
    public boolean conectar() throws Exception {
        // Informando qual driver de conexão será utilizado pelo DriveManager
        Class.forName("oracle.jdbc.driver.OracleDriver");
        // Criando a conexão com o BD
        conn = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "MARCELO", MM");
        if (conn == null || conn.isClosed()) {
            throw new Exception(“Erro ao fazer a conexão com o BD”);
            return false;
        } else {
            return true;
        }
    }
} // fim da classe Conexao
```

GERMINARE
Escola de Negócios

Continuação do exemplo de throws e throw

```
public class Principal{
    public static void main(String[] args) {
        try {
            Conexao conexao = new Conexao();
            conexao.conectar();
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }
}
```

GERMINARE
Escola de Negócios

Perguntas???