05-TratamentoDeErros-Exceções-Parte-2

Conteúdo: Tratamento de Erros. Exceções.

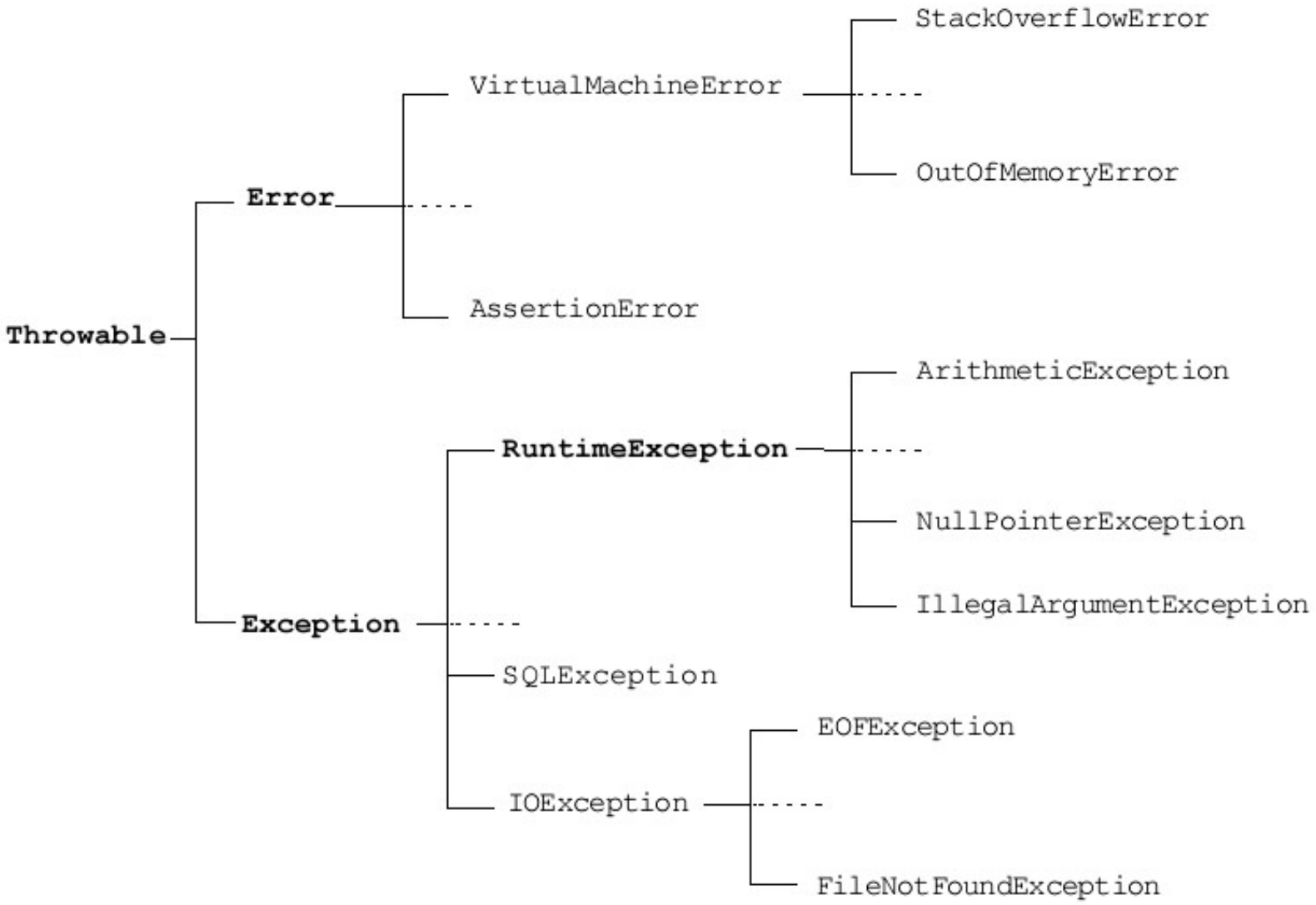
Introdução (1)

“O bom programador é aquele que olha para os dois lados antes de atravessar uma via de mão única.”. Doug Linder.

Tipos de Exceções (1)

- Verificadas: exceções que devem obrigatoriamente ser tratadas (que herdam diretamente de Exception)
- Não verificadas: não é obrigatorio tratar. Para alguns tipos de erros, não existe tratamento típico, podendo causar a parada brusca da execução do código (Error e RuntimeException)

Tipos de Exceções (2)



Tipos de Exceções (3)

- Exceções mais comuns
  - NullPointerException
  - ArrayIndexOutOfBoundsException
  - ArithmeticException
  - FileNotFoundException
  - NumberFormatException
  - NoSuchElementException

Exceções Verificadas

- Mecanismo trata ou repassa
  - Compilador força tratamento de exceções verificadas
  - Claúsula **throws**: especifica as exceções que um método pode lançar e força que o código chamador contenha tratamento para cada exceção declarada
- `void metodoA() throws IOException, MinhaExcecao {...}`
  - O código que chama metodoA deve estar num bloco try..catch ou então repassar (throws) a exceção

Prática Orientada

- Abrir um IDE e compilar código Integer.parseInt
- Abrir a IDE e tentar compilar o trecho a seguir:

```
import java.io.FileWriter;
public class Main {
 public static void main(String args[]) {
    new Main().gravaArquivo();
 }
 private void gravaArquivo() {
    FileWriter fw=new FileWriter("ok.txt");
    fw.write(new char[]{'O','l','a'});
    fw.close();
 }
}
```

- Formas: tratar ou repassar a exceção

Exceções Personalizadas (1)

- Criando uma exceção personalizada

```
public class MinhaException extends Exception {
   int meuCodigo;
   public MinhaException(String mensagem,int codigo) {
      super(mensagem);
      meuCodigo=codigo;
   }
   public int getCodigo() {
      return meuCodigo;
   }
}
```

Chama o construtor da superclasse (Exception). O construtor da Exception recebe uma string como argumento. Posso obter a mensagem através do método herdado de Exception, o getMessage()

Exceções Personalizadas (2)

- Lançando uma exceção personalizada

```
public static void conectar(String servidor)
   throws MinhaException  {
   boolean result;
   result=abrir(servidor);
   if(!result) {
     throw new MinhaException(“Não pude abrir”,171);
   }
}
```

O uso do método conectar implica a obrigação do tratamento (try..catch) pelo código usuário. Ver exemplo completo no código em TrataExcPersonalizado.java

Exceções Personalizadas (3)

- Usando um método que dispara exceção

```
public void BotaoClicado() {
  // Não trata nem repassa
  Conexao.conectar(tfServidor.getText());
}

public void BotaoClicado() {
  try {
    Conexao.conectar(tfServidor.getText());
  } catch(MinhaException me) {
    JOptionPane.showMessage(this,
      "Ocorreu o erro "+me.getCodigo()+
       ". Detalhe: " + me.getMessage());
  }
}
```

Supondo que conectar é um método estático de Conexao!
Código não compila.

De qual outra forma posso tratar a exceção verificada?

Tarefas Extraclasse

- Faça o questionário 05-TratamentoDeErros-Exceções-Parte-2
- T4 (sobre Exceções)