Tratamento de Exceção

Programação Orientada a Objetos

Java

(Rone Ilídio)

Tratamento de exceção

- Exceção é uma contração de “Evento de Exceção”
  - Evento de exceção impede a execução normal de um programa
- Quando ocorre uma exceção, a JVM cria um objeto Exception, que contem informações sobre a exceção.
- O termo técnico é lança uma exceção.
- Exemplo:
  - Acesso a posição de vetor não existente
  - Utilização de objeto que ainda não foi criado
  - Acesso a arquivo inexistente

Tratando Exceção

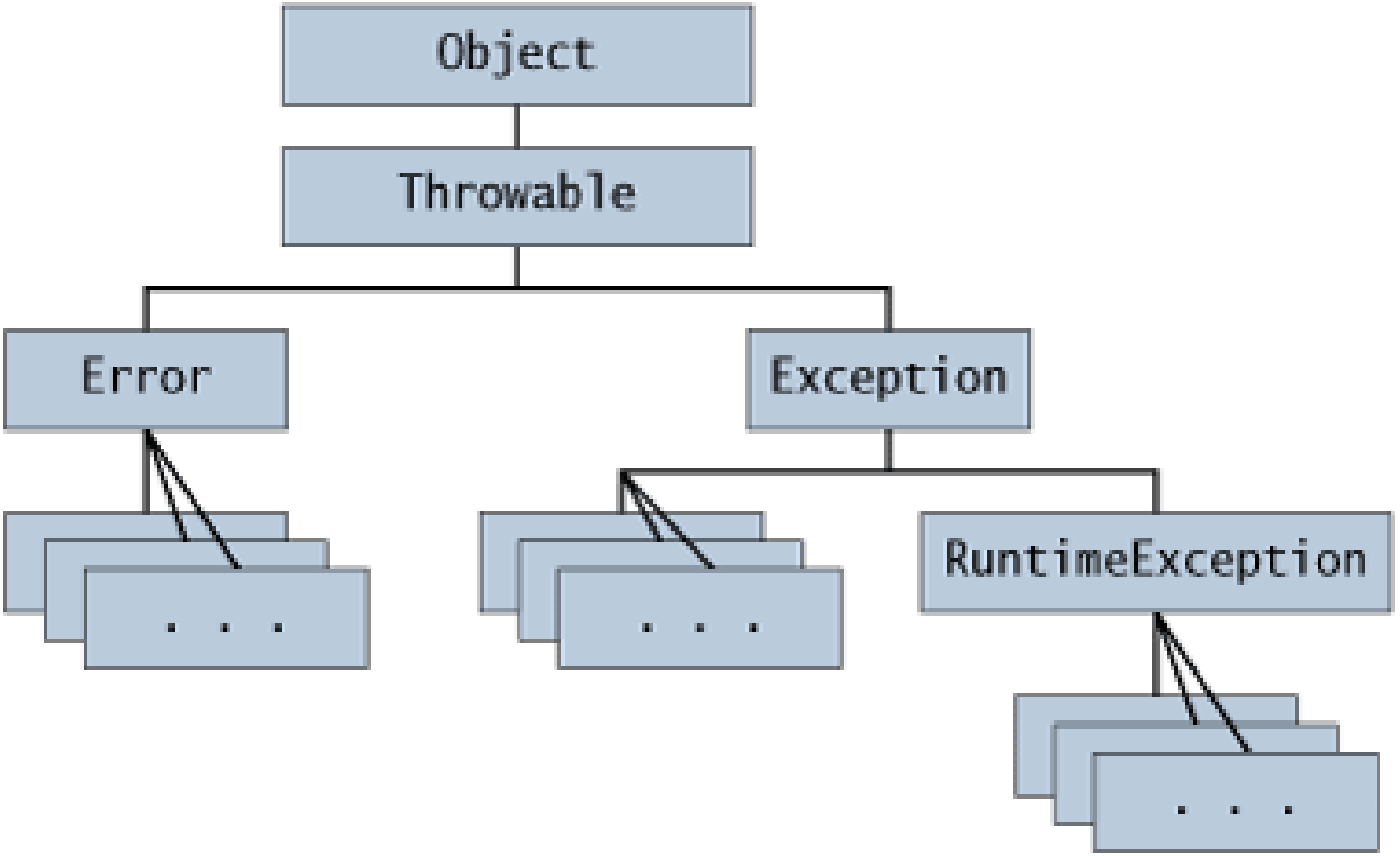
- Bloco: try-catch-finally
- Try: quanto ocorre uma exceção, a JVM faz seu lançamento e para a execução da sequência de comandos do bloco
- Catch: sequencia de comandos que é executada após uma exceção; faz o tratamento da exceção
- Finally: finaliza a exceção (bloco sempre executado)

Tratando Exceção

```
try{
  <comandos que podem ou nao lançar exceções>
}
catch(ExcecaoTipo1 parametro){
  <tratamento da instância da classe ExcecaoTipo1>
}
 catch(ExcecaoTipo2 parametro){
  <tratamento da instância da classe ExcecaoTipo2>
}
 ...
} finally{
  <estes comandos serão executados com ou sem lançamento de exceção>
}
```

```
class TrataExcecaoComTry{
  public static void main(String[] arg){
  int v[] = new int[3];
  try{
          for(int i=0; i<=3; i++)
          v[i] = i*i;
  }
  catch(Exception ex){
                  System.out.println("Acesso a posição inválida");
  }
  finally{
                   System.out.println("Executa de qualquer forma");
  }
  }
}
```

Classe Exception - Hierarquia



Classe Exception - Hierarquia

- A JVM lança (throw) exceções criando objetos do tipo Throwable (ou seus descendentes)
- Classe error trata exceções mais críticas (manipuladas pela JVM)
- A classe Exception possui um conjunto de filhas já criadas na API Java
- Exemplo de exceção é RuntimeException: indica uso incorreto da API → já possui várias funções para tratamento de exceções

Criando sua própria exceção

- O programador pode criar sua própria exceção
- Para isso, cria-se uma classe filha de Throwable
- Normalmente cria-se filhas de Exception → mais métodos implementados
- Exemplo: Criar uma classe para tratar divisão por 0:
  - Classe DivByZero → filha de Exception

Criando sua própria exceção

```
import javax.swing.JOptionPane;
public class DivByZero extends Exception {
  public DivByZero(int a){
  JoptionPane.showMessageDialog(null, a
    " não pode ser dividido!\n(Construtor de DivByZero)");
  }
  public void showError(){
  JoptionPane.showMessageDialog(null,
              "Dividiu por 0, não pode!\n(método showError())");
  }
}
```

Throws e Throw

- Throw: lança uma exceção
  - Na prática, cria um objeto que define uma exceção (filha de Throwable)
- Throws: define características de métodos
  - O método declarado com Throws Exception só poderá ser chamado dentro de um bloco try com catch Exception
  - Esse método poderá lançar uma exceção do tipo Exception
- Exemplo

```
import java.lang.*;
import javax.swing.*;
public class Divisao extends JApplet{
  public int divide(int a, int b) throws DivByZero {
        if (b == 0) throw new DivByZero(a);
        return a/b;
    }
    public void init(){
       try{
          JOptionPane.showMessageDialog(null,
                                              "Resultado:" + divide(10,0));
       }
       catch(DivByZero dex){
           System.out.println("Divisão por zero!!!!!");
           dex.showError();
        }
   }
}
```

Throws e Throw

- O método divide throws DivByZero, então
  - Ele pode lança uma exceção DivByZero
  - Ele tem que ser chamado dentro de um bloco try-catch
- O objeto dex é criado pela JVM
- Observação: nesse exemplo, são criados propositalmente dois objetos DivByZero, o primeiro pela JVM quando ocorre uma exceção. O segundo dentro do método divide, com o lançamento da execeção em throw DivByZero