Tratamento de Exceções em Java

Profa Andréa Schwertner Charão

DLSC/CT/UFSM

Tratamento de exceções

- **Exceção**: evento "excepcional" que interrompe o fluxo normal da execução de um programa
- Exemplos:
  - Uso indevido de referência null
  - Divisão por zero, raiz quadrada de número negativo, etc.
  - Uso de índice inválido em arrays e coleções
  - Problemas na manipulação de arquivos
  - Problemas na comunicação em rede
  - Etc.
- Exceções tratadas deixam o programa mais confiável

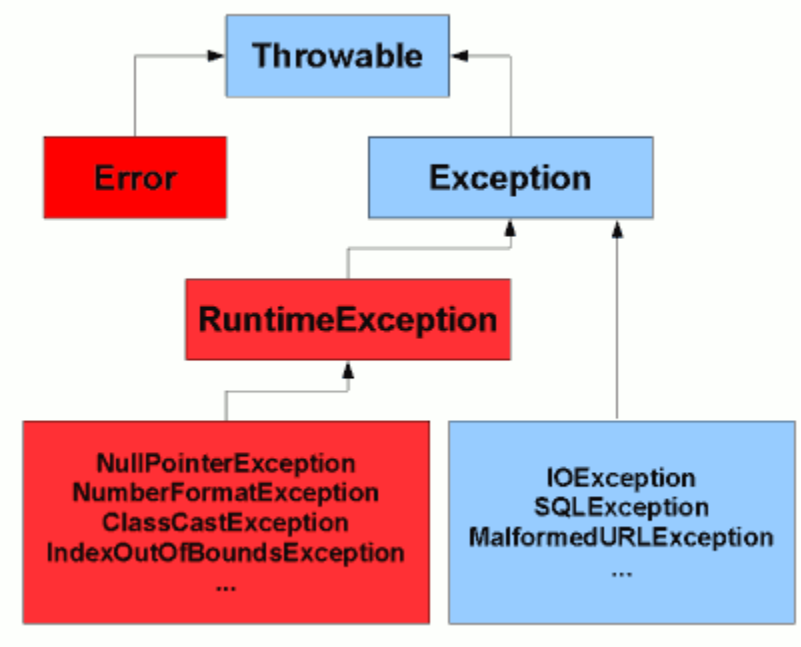
Tratamento de exceções em C

- Em C, geralmente testamos o valor de retorno das funções para verificar se ocorreu erro/exceção
- Programador deve se lembrar de testar cada valor retornado
- Código pode ficar pouco legível

```
FILE *fpt = fopen(file, "rb");
if (!fpt) {
  // some error occurred -- find out what
} else {
  // read from file
}
```

Exceções em Java

- Java oferece vários recursos (classes e sintaxe de linguagem) para lidar com exceções
- Classes
  - **Exception**: hierarquia de classes que representam exceções tratáveis
  - **Error**: erro grave que normalmente não pode ser tratado



Fonte:
http://www.javamex.com/tutorials/exceptions/exceptions_hierarchy.shtml

Exceções em Java

- Recursos da linguagem
  - Bloco try/catch:
    - ✓ try: declara bloco de código que pode produzir exceções
    - ✓ catch: trata exceções que ocorrerem no bloco try
    - ✓ finally: declara bloco de código que sempre executa, com ou sem exceção ocorrida (geralmente para código de "limpeza")
  - throw: lança exceção explicitamente
  - throws: indica exceções que um método/classe podem produzir

Exceções em Java

```
class ExemploExcecao {
   public static void main (String[] args) {
      int[] numbers = new int[10];
      int outofbounds = numbers[20];
      System.out.println("Vou terminar a execucao");
   }
}
```

```
Exception in thread "main"
java.lang.ArrayIndexOutOfBoundsException: 20
        at ExemploExcecao.main(ExemploExcecao.java:4)
```

Exceções em Java: try-catch

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

      try {
         int[] numbers = new int[10];
         int outofbounds = numbers[20];
      }
      catch (ArrayIndexOutOfBoundsException e)
      {
         System.out.println("Indice fora dos limites");
      }
      System.out.println("Vou terminar a execucao");
   }
}
```

Saída:
Indice fora dos limites
Vou terminar a execucao

Exceções em Java: try-catch-finally

```
class ExemploTrataE
   public static vo
      try {
         int[] numb
         int outofbound                    eis[20];
      }
      catch (ArrayIndexOutOfBoundsException e) {
         System.out.println("Indice fora dos limites");
      }
      catch (RuntimeException e) {
         e.getMessage();
      }
      finally {
         System.out.println("Codigo sempre executado");
      }
      System.out.println("Vou terminar a execucao");
   }
}
```

Ordem:
classes específicas antes de classes gerais

Saída:
Indice fora dos limites
Codigo sempre executado
Vou terminar a execucao

Exceções em Java: try-catch-finally

java.io

Class FileNotFoundException

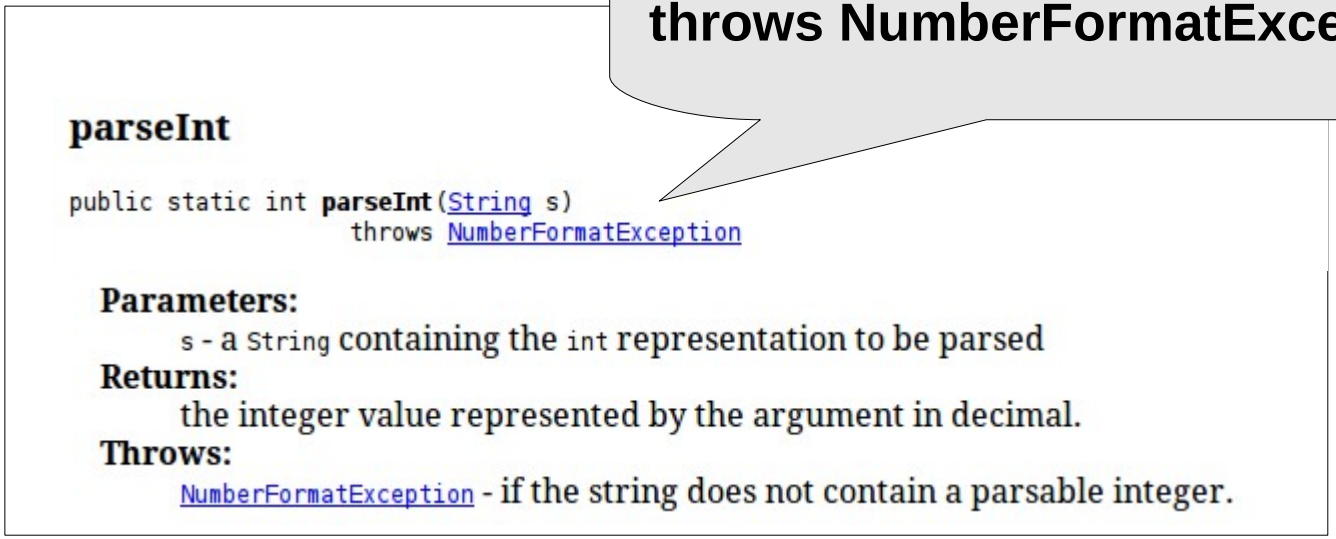
java.lang.Object
  └ java.lang.Throwable
    └ java.lang.Exception
      └ java.io.IOException
        └ **java.io.FileNotFoundException**

```
public void save() {
    PrintWriter out = null;
    try {
        out = new PrintWriter(new FileWriter("OutFile.txt"));
        for (Integer i : list) {
            out.println(i);
        }
    } catch (FileNotFoundException e) {
        System.err.println("FileNotFoundException: "
                + e.getMessage());
    } catch (IOException e) {
        System.err.println("IOException: "
                + e.getMessage());
    } finally {
        if (out != null) {
            System.out.println("Closing PrintWriter");
            out.close();
        } else {
            System.out.println("PrintWriter not open");
        }
    }
}
```

Exceções em Java: throws

- Uso de throws: indica exceções que um método/classe podem produzir
- Plataforma Java faz uso de throws em vários métodos

Integer.parseInt:
throws NumberFormatException



parseInt

```
public static int parseInt(String s)
                    throws NumberFormatException
```

Parameters:
s - a String containing the int representation to be parsed
Returns:
the integer value represented by the argument in decimal.
Throws:
NumberFormatException - if the string does not contain a parsable integer.

Exceções em Java

- Para usar um método que pode causar exceção:
  - Capturar a exceção em um bloco try-catch
  - **OU** delegar o tratamento para quem invocou o método (usando throws)
- Se nada disso for feito, compilação vai gerar erro

```
private Disciplina newFromView() {
    try {
        Disciplina discip = new Disciplina();
        discip.setAno(Integer.parseInt(view.getTextAno().getText()));
        discip.setSemestre(view.getComboSemestre().getSelectedIndex() + 1);
        discip.setNome(view.getTextNome().getText());
        discip.setNota(Float.parseFloat(view.getTextNota().getText()));
        return discip;
    } catch (NumberFormatException e) {
        view.showError("Dado(s) de entrada invalido(s)!");
        return null;
    }
}
```

Exceções em Java

- Para usar um método que pode causar exceção:
  - Capturar a exceção em um bloco try-catch
  - **OU** delegar o tratamento para quem invocou o método (usando throws)
- Se nada disso for feito, compilação vai gerar erro

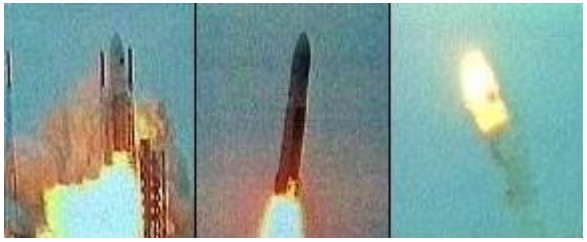
```
private Disciplina newFromView2() throws NumberFormatException {
    Disciplina discip = new Disciplina();
    discip.setAno(Integer.parseInt(view.getTextAno().getText()));
    discip.setSemestre(view.getComboSemestre().getSelectedIndex() + 1);
    discip.setNome(view.getTextNome().getText());
    discip.setNota(Float.parseFloat(view.getTextNota().getText()));
    return discip;
}
```

Finalizando...

- Objetivos/vantagens:
  - Separar código "normal" do código que trata situações "anormais"
  - Facilitar propagação de erros na pilha de chamadas
  - Agrupar e diferenciar tipos de erros
- Boas práticas:
  - Nunca ignorar exceções (bloco catch vazio)
  - Evitar fazer catch em exceções de nível mais alto (ex.: Exception)
  - Etc.
- Mais sobre exceções em:

  http://download.oracle.com/javase/tutorial/essential/exceptions/advantages.html

Finalizando: o caso Ariane 5



- Em 1996, foguete Ariane 5, da Agência Espacial Européia, explode durante lançamento
- Bug causado pelo reuso de software de Ariane 4
- **Exceção** ocorreu na conversão de um float (64 bits) para um inteiro (16 bits) - linguagem Ada
- Sistema primário não tratou exceção e falhou, sistema secundário assumiu, mas rodava o mesmo software e falhou também
- Mais sobre isso em: http://www.ima.umn.edu/~arnold/disasters/ariane5rep.html