Tratamento de Exceções

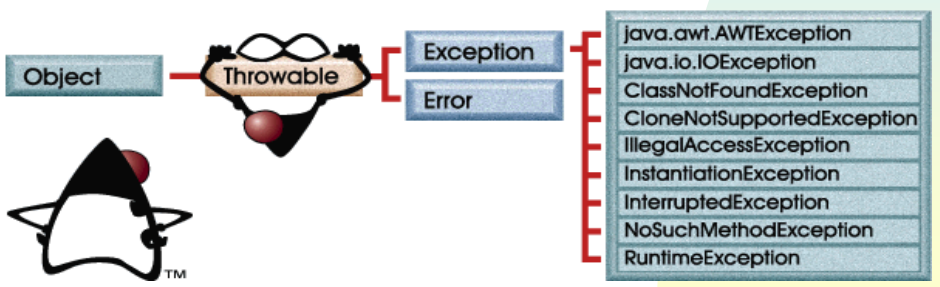
Tipos
Manipulação
Exemplos
Próprias exceções

1

–1

Tipos de Exceção

- **Error** = condições muito graves, sem recuperação
- **Exception** = superclasse das classes usadas para representar dois tipos de condições excepcionais:
  - classe RuntimeException = geradas pelo sistema de tempo de execução Java (falhas de projeto ou programação)
  - Todas as outras exceções



2

–2

Exceções

- Exceções ocorrem quando é tentada uma ação inválida
  - ex.: divisão por zero, acessar uma posição que não existe num array, abrir um arquivo que não existe, tentar conectar um site não habilitado, ...
- Manipulação de exceções
  - Pode-se definir que o método não irá tratar as exceções (**throws**), ou
  - pode-se tratá-las individualmente (**try / catch**).

```
int i =0;
try {
    i = i/i;
} catch (Exception e) {
    System.out.println(
    "Erro " + e)
}
```

```
try {
    LigaTorneira();
    MolhaGramado();
} finally {
    DesligaTorneira();
}
```

3

–3

Exemplo: tratando a exceção

```
class Arquivo {
public static void main(String[] args) {
 try {
   FileReader arqin = new FileReader(args[0]);
   int lido;
   while ((lido = arqin.read()) != -1) {
     System.out.print((char)lido);
   }
   arqin.close();
 } catch (ArrayIndexOutOfBoundsException e){
   System.out.println("Deve ser informado o arquivo como
  parâmetro");
 } catch (FileNotFoundException e) {
   System.out.println("Erro na abertura do arquivo " + args[0]);
 } catch (IOException e) {
   System.out.println("Erro de leitura");
 }
```

4

–4

Exemplo: propagando a exceção

```
class Arquivo {

public static void main(String[] args)
   throws FileNotFoundException, IOException {

try {
   FileReader arqin = new FileReader(args[0]);
   int lido;
   while ((lido = arqin.read()) != -1) {
     System.out.print((char)lido);
   }
   arqin.close();
} catch (ArrayIndexOutOfBoundsException e){
   System.out.println("Deve ser informado o arquivo como
                       parâmetro");
}
```

5

–5

Exceções comuns (Runtime)

- **ArithmeticException**: normalmente, o resultado de uma operação de divisão por zero realizada com números inteiros:

  int i = 12 / 0;

- **NullPointerException**: acessar um objeto ou método antes de ele ter sofrido uma instanciação:

  Image im;

  System.out.println(im.toString());

- **NegativeArraySizeException**: criar uma matriz comum tamanho de dimensão negativo.
- **ArrayIndexOutOfBoundsException**: acessar um elemento de matriz fora da definição original de tamanho da matriz.
- **IllegalArgumentException**: indica que um método passou um parâmetro ilegal ou inapropriado.

6

–6

Próprias exceções

1. **Criar a classe de exceção**
2. **Definir o ponto de lançamento da exceção (situação excepcional)**
3. **Tratar o lançamento da exceção**

```
class MinhaConsistenciaException extends Exception {
    private String razao, solucao;

    public MinhaConsistenciaException(String r,String s) {
        razao = r;
        solucao = s;
    }
    public String getRazao()       { return razao;}
    public String getSolucao()     { return solucao;}

}
```

7

–7

Próprias exceções

```
public void setIdade(int valor) throws MinhaConsistenciaException{
       if (valor < 0 || valor > 150)  // teste para consistência
              throw new MinhaConsistenciaException("Valor
              incorreto em Idade","Valores entre 0 e 150");
       idade = valor;
   }
```

No uso:

```
   try {
       pessoaAtual.setIdade(Integer.parseInt(aux2));
   } catch (MinhaConsistenciaException excecao) {
       System.out.println(excecao.getRazao()+
       "\n"+excecao.getSolucao());
   }
```

8

–8

Resumo

- Classe **Throwable** com subclasses **Error** e **Exception**
- A exceção é tratada (**try / catch**) ou é lançada/propagada para tratador em outro nível (**throws**)
- Pode se personalizar as exceções, criando as próprias

```
public class Integers {
  public static void main (String args[]) {
     int i = 0;
     int numbers[] = {1,2,3,4};
     while (i < 5) {
        try {
                System.out.print (numbers[i]);
        } catch (ArrayIndexOutOfBoundsException e) {
                System.out.println
                ("Caught Exception");
        } finally {
                System.out.println
                (" is an integer.");
        }
        i++;
}  }  }
```

https://programming.guide/java/list-of-java-exceptions.html

9

–9