Lição 2

Exceções e Assertivas

Objetivos

Ao final desta lição, o estudante será capaz de:

- Tratar exceções pelo uso de *try*, *catch* e *finally*
- Diferenciar entre o uso de *throw* e *throws*
- Utilizar as classes de exceções existentes
- Diferenciar entre exceções verificadas e não verificadas
- Definir suas próprias classes de exceção
- Explicar os benefícios do uso de assertivas
- Utilizar declarações assertivas

O que são exceções?

- Eventos excepcionais
- Erros que ocorrem durante a execução do programa
- Causam distúrbios no fluxo normal do programa

Classe *Throwable*

- Classe de origem de todas as classes de exceção
- Sub-classe *Error*
- Sub-classe *Exception*

Classe *Exception*

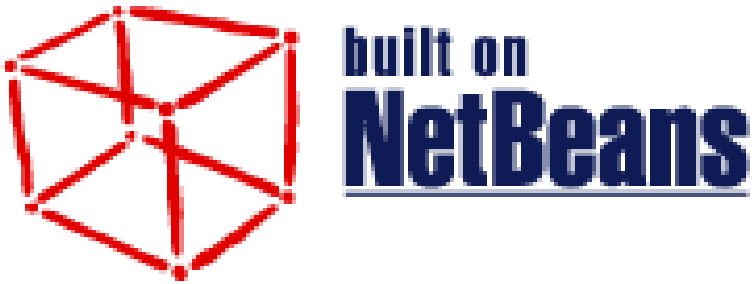
- Circunstâncias que os usuários podem tratar
- Resultam de falhas no código do programa
- Exemplos:
  - Erro em uma divisão por zero
  - Erro ao tentar acessar um índice inexistente um array

Classe *Error*

- Usada pela JVM para manipular erros ocorridos no ambiente de execução
- Geralmente, esses erros encontram-se além do controle dos programas
- Exemplos:
  - Error de falta de memória
  - Erro de acesso ao disco rígido

Exemplo de exceção

- Passaremos agora para o NetBeans



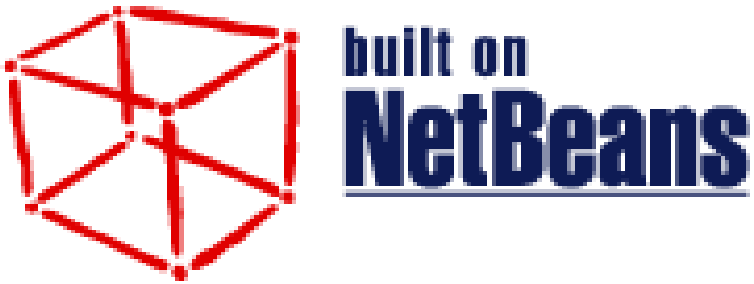
Capturando exceções: Instruções *try-catch*

- Sintaxe:

```
try {
   <código a ser monitorado para exceções>
} catch (<ClasseExceção1> <nomeObj>) {
   <tratar se ClasseExceção1 ocorrer>
}
...
} catch (<ClasseExceçãoN> <nomeObj>) {
   <tratar se ClasseExceçãoN ocorrer>
}
```

Capturando exceções: Instruções *try-catch*

- Passaremos agora para o NetBeans



Capturando exceções: Palavra-chave *finally*

- Sintaxe:

```
try {
    <código a ser monitorado para exceções>
} catch (<ClasseExceção1> <nomeObj>) {
    <tratar se ClasseExceção1 ocorrer>
} ...
} finally {
    <código a ser executado antes do bloco try
  ser finalizado>
}
```

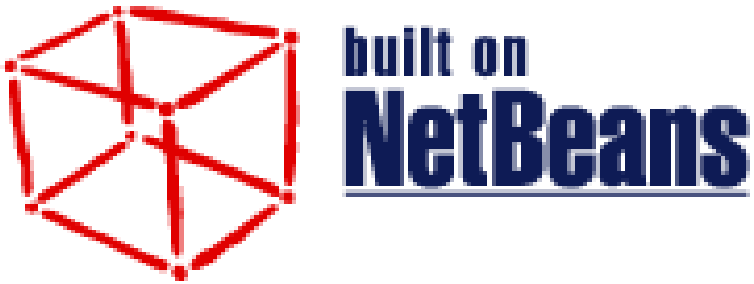
Capturando exceções: Palavra-chave *finally*

- Bloco de código é executado em um desses quatro cenários:

1. Saída forçada usando instruções *return*, *continue* ou *break*
2. Execução normal sem exceções
3. Um bloco *catch* captura e trata uma exceção lançada
4. Exceção lançada não é capturada por nenhum bloco *catch*

Capturando exceções: Palavra-chave *finally*

- Passaremos agora para o NetBeans



Lançando de exceções: Palavra-chave *throw*

- Java nos permite lançar exceções:

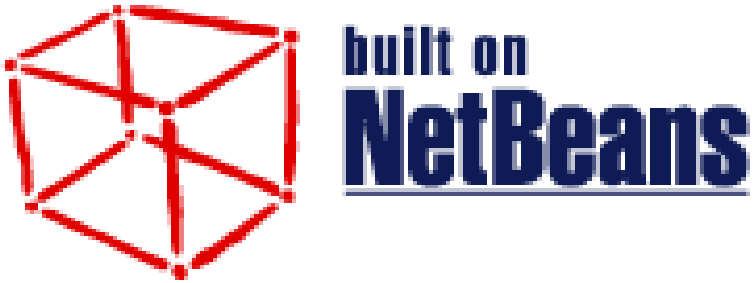
```
throw <objetoExceção>;
```

- Exemplo:

```
throw new ArithmeticException("testing...");
```

Lançando de exceções: Palavra-chave *throw*

- Passaremos agora para o NetBeans



Lançando de exceções: Palavra-chave *throws*

- É necessário um método para cada *catch* ou lista de exceções que podem ser lançadas
  - Exceto para as Classes *Error, RuntimeException* e suas sub-classes

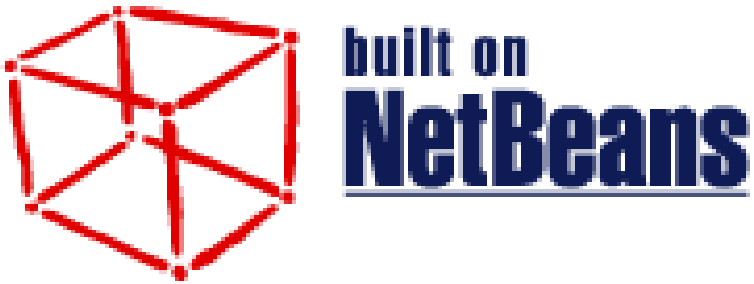
- Se um método causar uma exceção mas não capturá-la, então deve-se utilizar a palavra-chave *throws*

- Sintaxe:

```
<tipo> <nomeMetodo> (<argumento>*) throws
  <listaExceção> {
    <methodBody>
}
```

Lançando de exceções: Palavra-chave *throws*

- Passaremos agora para o NetBeans



Hierarquia classes de exceção

Hierarquia das Classes de Exceções			
Throwable	Error	LinkageError, ...	
		VirtualMachineError, ...	
	Exception	ClassNotFoundException,	
		CloneNotSupportedException,	
		IllegalAccessException,	
		InstantiationException,	
		InterruptedException,	
		IOException,	EOFException,
			FileNotFoundException,
			...
		RuntimeException,	ArithmeticException,
			ArrayStoreException,
			ClassCastException,
			IllegalArgumentException, (IllegalThreadStateException e NumberFormatException como sub-classes)
			IllegalMonitorStateException,
			IndexOutOfBoundsException,
			NegativeArraySizeException,
			NullPointerException,
			SecurityException
		...	

Hierarquia classes de exceção

- Múltiplos blocos *catch* devem ser ordenados da sub-classe para a super-classe.

```
class MultipleCatchError {
   public static void main(String args[]){
      try {
         int a = Integer.parseInt(args [0]);
         int b = Integer.parseInt(args [1]);
         System.out.println(a/b);
      } catch (ArrayIndexOutOfBoundsException e) {
      } catch (Exception ex) {
      }
   }
}
```

Exceções verificadas e não-verificadas

- Exceções Verificadas
  - O compilador Java verifica se cada *catch* ou lista de exceções encontram-se dentro da cláusula *throws*
  - Se não, ocorrerá um erro de compilação

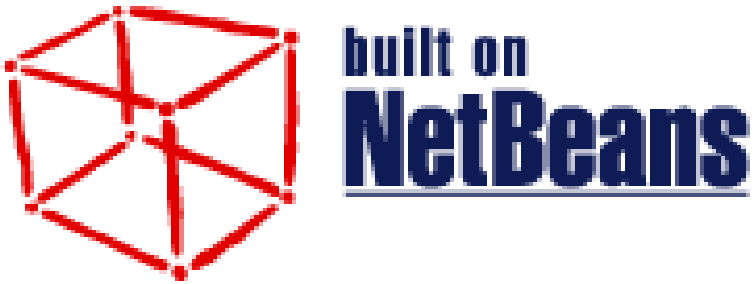
- Exceções Não-verificadas
  - Não são verificadas no momento da compilação
  - Classes de exceção não-verificadas
    - *Error*
    - *RuntimeException*
    - E suas sub-classes
  - Tratamento de todas as exceções pode bagunçar o código do programa causando transtornos

Exceções definidas pelo usuário

- Crie uma classe que estenda a classe *RuntimeException* ou *Exception*
- Customize a classe
  - Atributos de objetos e construtores podem ser adicionados à classe

Exceções definidas pelo usuário

- Passaremos agora para o NetBeans



O que são assertivas?

- Permitem ao programador descobrir se uma suposição foi encontrada
- Extensão de comentários em que a assertiva informa à pessoa que lê o código que uma condição específica deve ser sempre satisfeita
- Usuário pode optar por habilitar ou não na execução

Habilitando ou desabilitando assertivas

- Deve-se estar atento ao uso de assertivas no código, sob o risco do programa funcionar de forma incorreta

- Compilando
  - Com o recurso de assertivas:

```
javac –source 1.4 MyProgram.java
```

  - Sem o recurso de assertivas:

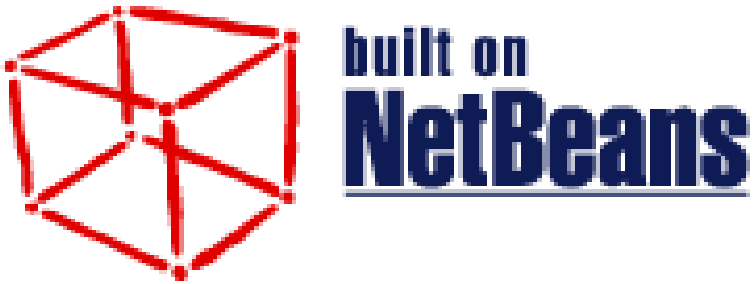
```
javac MyProgram.java
```

- Habilitando Assertivas:
  - Use os parâmetros –enableassertions ou –ea.

```
java –enableassertions MyProgram
```

Habilitando ou desabilitando assertivas: Exemplo

- Passaremos agora para o NetBeans



Sintaxe das assertivas

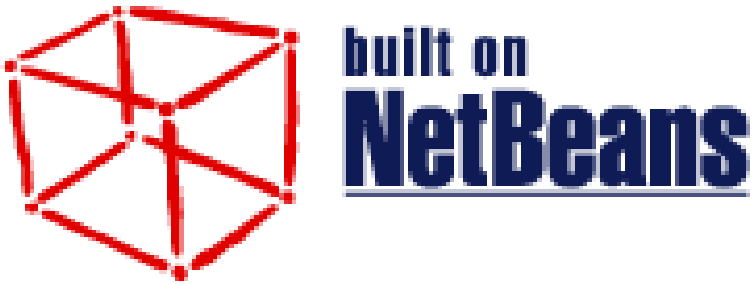
- Sintaxe:

```
assert <expressãoLógica>;

assert <expressãoLógica> : <Mensagem>;
```

Assertivas: Exemplo

- Passaremos agora para o NetBeans



Sumário

- Exceções
  - Definição
  - *try*, *catch* e *finally*
  - *throw* e *throws*
  - Classes *Throwable*, *Exception*, *Error*
  - Exceções Verificadas e Não-verificadas
  - Exceções Definidas pelo Usuário
- Assertivas
  - Definição
  - Habilitando e Desabilitando Assertivas
  - Sintaxe das Assertivas

Parceiros

- **Os seguintes parceiros tornaram JEDI™ possível em Língua Portuguesa:**