Módulo 1

Introdução à Programação I

Lição 12

Tratamento básico de exceções

Versão 1.0 - Jan/2007

Autor
Florence Tiu Balagtas

Equipe
Joyce Avestro
Florence Balagtas
Rommel Feria
Reginald Hutcherson
Rebecca Ong
John Paul Petines
Sang Shin
Raghavan Srinivas
Matthew Thompson

Necessidades para os Exercícios

Sistemas Operacionais Suportados

NetBeans IDE 5.5 para os seguintes sistemas operacionais:

- Microsoft Windows XP Profissional SP2 ou superior
- Mac OS X 10.4.5 ou superior
- Red Hat Fedora Core 3
- Solaris™ 10 Operating System (SPARC® e x86/x64 Platform Edition)

NetBeans Enterprise Pack, poderá ser executado nas seguintes plataformas:

- Microsoft Windows 2000 Profissional SP4
- Solaris™ 8 OS (SPARC e x86/x64 Platform Edition) e Solaris 9 OS (SPARC e x86/x64 Platform Edition)
- Várias outras distribuições Linux

Configuração Mínima de Hardware

Nota: IDE NetBeans com resolução de tela em 1024x768 pixel

Sistema Operacional	Processador	Memória	HD Livre
Microsoft Windows	500 MHz Intel Pentium III workstation ou equivalente	512 MB	850 MB
Linux	500 MHz Intel Pentium III workstation ou equivalente	512 MB	450 MB
Solaris OS (SPARC)	UltraSPARC II 450 MHz	512 MB	450 MB
Solaris OS (x86/x64 Platform Edition)	AMD Opteron 100 Série 1.8 GHz	512 MB	450 MB
Mac OS X	PowerPC G4	512 MB	450 MB

Configuração Recomendada de Hardware

Sistema Operacional	Processador	Memória	HD Livre
Microsoft Windows	1.4 GHz Intel Pentium III workstation ou equivalente	1 GB	1 GB
Linux	1.4 GHz Intel Pentium III workstation ou equivalente	1 GB	850 MB
Solaris OS (SPARC)	UltraSPARC IIIi 1 GHz	1 GB	850 MB
Solaris OS (x86/x64 Platform Edition)	AMD Opteron 100 Series 1.8 GHz	1 GB	850 MB
Mac OS X	PowerPC G5	1 GB	850 MB

Requerimentos de Software

NetBeans Enterprise Pack 5.5 executando sobre Java 2 Platform Standard Edition Development Kit 5.0 ou superior (JDK 5.0, versão 1.5.0_01 ou superior), contemplando a Java Runtime Environment, ferramentas de desenvolvimento para compilar, depurar, e executar aplicações escritas em linguagem Java. Sun Java System Application Server Platform Edition 9.

- Para **Solaris**, **Windows**, e **Linux**, os arquivos da JDK podem ser obtidos para sua plataforma em http://java.sun.com/j2se/1.5.0/download.html
- Para **Mac OS X**, Java 2 Plataform Standard Edition (J2SE) 5.0 Release 4, pode ser obtida diretamente da Apple's Developer Connection, no endereço: http://developer.apple.com/java (é necessário registrar o download da JDK).

Para mais informações:
http://www.netbeans.org/community/releases/55/relnotes.html

Colaboradores que auxiliaram no processo de tradução e revisão

Alexandre Mori
Alexis da Rocha Silva
Aline Sabbatini da Silva Alves
Allan Wojcik da Silva
André Luiz Moreira
Andro Márcio Correa Louredo
Antoniele de Assis Lima
Antonio Jose R. Alves Ramos
Aurélio Soares Neto
Bruno da Silva Bonfim
Bruno dos Santos Miranda
Bruno Ferreira Rodrigues
Carlos Alberto Vitorino de Almeida
Carlos Alexandre de Sene
Carlos André Noronha de Sousa
Carlos Eduardo Veras Neves
Cleber Ferreira de Sousa
Cleyton Artur Soares Urani
Cristiano Borges Ferreira
Cristiano de Siqueira Pires
Derlon Vandri Aliendres
Fabiano Eduardo de Oliveira
Fábio Bombonato
Fernando Antonio Mota Trinta
Flávio Alves Gomes
Francisco das Chagas
Francisco Marcio da Silva
Gilson Moreno Costa
Givailson de Souza Neves
Gustavo Henrique Castellano
Hebert Julio Gonçalves de Paula
Heraldo Conceição Domingues
Hugo Leonardo Malheiros Ferreira
Ivan Nascimento Fonseca
Jacqueline Susann Barbosa
Jader de Carvalho Belarmino
João Aurélio Telles da Rocha
João Paulo Cirino Silva de Novais
João Vianney Barrozo Costa
José Augusto Martins Nieviadonski
José Leonardo Borges de Melo
José Ricardo Carneiro
Kleberth Bezerra G. dos Santos
Lafaiete de Sá Guimarães
Leandro Silva de Morais
Leonardo Leopoldo do Nascimento
Leonardo Pereira dos Santos
Leonardo Rangel de Melo Filardi
Lucas Mauricio Castro e Martins
Luciana Rocha de Oliveira
Luís Carlos André
Luís Octávio Jorge V. Lima
Luiz Fernandes de Oliveira Junior
Luiz Victor de Andrade Lima
Manoel Cotts de Queiroz
Marcello Sandi Pinheiro
Marcelo Ortolan Pazzetto
Marco Aurélio Martins Bessa
Marcos Vinicius de Toledo
Maria Carolina Ferreira da Silva
Massimiliano Giroldi
Mauricio Azevedo Gamarra
Mauricio da Silva Marinho
Mauro Cardoso Mortoni
Mauro Regis de Sousa Lima
Namor de Sá e Silva
Néres Chaves Rebouças
Nolyanne Peixoto Brasil Vieira
Paulo Afonso Corrêa
Paulo José Lemos Costa
Paulo Oliveira Sampaio Reis
Pedro Antonio Pereira Miranda
Pedro Henrique Pereira de Andrade
Renato Alves Félix
Renato Barbosa da Silva
Reyderson Magela dos Reis
Ricardo Ferreira Rodrigues
Ricardo Ulrich Bomfim
Robson de Oliveira Cunha
Rodrigo Pereira Machado
Rodrigo Rosa Miranda Corrêa
Rodrigo Vaez
Ronie Dotzlaw
Rosely Moreira de Jesus
Seire Pareja
Sergio Pomerancblum
Silvio Sznifer
Suzana da Costa Oliveira
Tásio Vasconcelos da Silveira
Thiago Magela Rodrigues Dias
Tiago Gimenez Ribeiro
Vanderlei Carvalho Rodrigues Pinto
Vanessa dos Santos Almeida
Vastí Mendes da Silva Rocha
Wagner Eliezer Roncoletta

Auxiliadores especiais

Revisão Geral do texto para os seguintes Países:

- **Brasil** – Tiago Flach
- **Guiné Bissau** – Alfredo Cá, Bunene Sisse e Buon Olossato Quebi – ONG Asas de Socorro

Coordenação do DFJUG

- **Daniel deOliveira** – JUGLeader responsável pelos acordos de parcerias
- **Luci Campos -** Idealizadora do DFJUG responsável pelo apoio social
- **Fernando Anselmo -** Coordenador responsável pelo processo de tradução e revisão, disponibilização dos materiais e inserção de novos módulos
- **Regina Mariani -** Coordenadora responsável pela parte jurídica
- **Rodrigo Nunes -** Coordenador responsável pela parte multimídia
- **Sérgio Gomes Veloso -** Coordenador responsável pelo ambiente JEDI™ (Moodle)

Agradecimento Especial

John Paul Petines – Criador da Iniciativa JEDI™
Rommel Feria – Criador da Iniciativa JEDI™

1. Objetivos

Nesta lição, iremos aprender uma técnica utilizada em Java para tratar condições incomuns que interrompem a operação normal da classe. Esta técnica é chamada de **tratamento de exceção**.

Ao final desta lição, o estudante será capaz de:

- Definir o que são exceções
- Tratar exceções utilizando **try-catch-finally**

2. O que são Exceções (Exception)?

Uma exceção é um evento que interrompe o fluxo normal de processamento de uma classe. Este evento é um erro de algum tipo. Isto causa o término anormal da classe.

Estes são alguns dos exemplos de exceções que podem ter ocorridos em exercícios anteriores:

- **ArrayIndexOutOfBoundsException**, ocorre ao acessar um elemento inexistente de um array.
- **NumberFormatException**, ocorre ao enviar um parâmetro não-numérico para o método **Integer.parseInt()**.

3. Tratando Exceções

Para tratar exceções em Java utilizamos a declaração **try-catch-finally**. O que devemos fazer para proteger as instruções passíveis de gerar uma exceção, é inserí-las dentro deste bloco.

A forma geral de um **try-catch-finally** é:

```
try{
    // escreva as instruções passíveis de gerar uma exceção
    // neste bloco
} catch (<exceptionType1> <varName1>){
    // escreva a ação que o seu programa fará caso ocorra
    //  uma exceção de um determinado
} . . .
} catch (<exceptionTypen> <varNamen>){
    // escreva a ação que o seu programa fará caso ocorra
    //  uma exceção de um determinado tipo
} finally {
    // escreva a ação que o seu programa executará caso ocorra
    // ou não um erro ou exceção
}
```

Exceções geradas durante a execução do bloco **try** podem ser detectadas e tratadas num bloco **catch**. O código no bloco **finally** é sempre executado, ocorrendo ou não a exceção.

A seguir são mostrados os principais aspectos da sintaxe da construção de um **try-catch-finally**:

- A notação de bloco é obrigatória.
- Para cada bloco **try**, pode haver um ou mais blocos **catch**, mas somente um bloco **finally**.
- Um bloco **try** deve que ser seguido de PELO MENOS um bloco **catch** OU um bloco **finally**, ou ambos.
- Cada bloco **catch** define o tratamento de uma exceção.
- O cabeçalho do bloco **catch** recebe somente um argumento, que é a exceção (Exception) que este bloco pretende tratar.
- A exceção deve ser da classe **Throwable** ou de uma de suas subclasses.

Para um melhor entendimento, observe a figura 1 que demonstra o fluxo seguido pelo **try-catch-finally**:

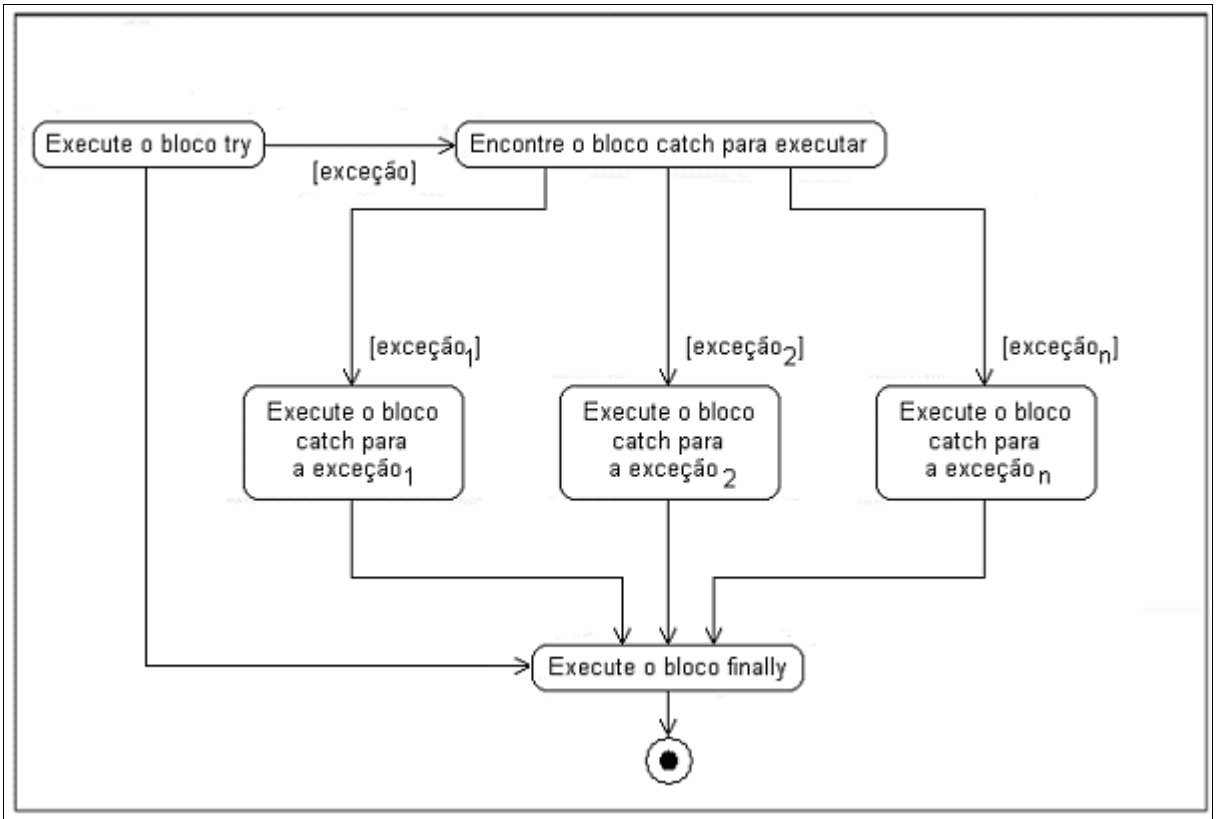

Figura 1: Fluxo em um **try-catch-finally**

Tomemos, por exemplo, uma classe que imprime o segundo argumento passado através da linha de comandos. Supondo que não há verificação no código para o número de argumentos.

```
public class ExceptionExample {
    public static void main( String[] args ) {
        System.out.println(args[1]);
        System.out.println("Finish");
    }
}
```

Ao executar esta classe sem informar nenhum argumento e, ao tentar acessar diretamente, conforme o exemplo descrito, o segundo argumento args[1], uma exceção é obtida que interromperá a execução normal do programa, e a seguinte mensagem será mostrada:

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 1
            at ExceptionExample.main(ExceptionExample.java:5)
```

Para prevenir que isto ocorra, podemos colocar o código dentro de um bloco **try-catch**. O bloco **finally** é opcional. Neste exemplo, não utilizaremos o bloco **finally**.

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

        try {
            System.out.println( args[1] );
        } catch (ArrayIndexOutOfBoundsException exp) {
            System.out.println("Exception caught!");
```

```
                }
                System.out.println("Finish");
            }
        }
```

Assim, quando tentarmos rodar o programa novamente sem a informação dos argumentos, a saída trataria a exceção e o fluxo do programa não seria interrompido, mostrando o resultado:

```
Exception caught!
Finish
```

4. Exercícios

4.1. Capturando Exceções 1

Dada a seguinte classe:

```
public class TestException {
    public static void main(String[] args) {
        for (int i=0; true; i++) {
            System.out.println("args["+i+"]="+ args[i]);
        }
        System.out.println("Quiting...");
    }
}
```

Compile e rode a classe **TestException**. E como saída será:

```
java TestExceptions  one  two  three
args[0]=one
args[1]=two
args[2]=three
Exception in thread "main"
      java.lang.ArrayIndexOutOfBoundsException: 3
            at TestExceptions.main(1.java:4)
```

Modifique a classe **TestException** para tratar esta exceção. A saída depois do tratamento da exceção deverá ser:

```
java TestExceptions  one  two  three
args[0]=one
args[1]=two
args[2]=three
Exception caught: java.lang.ArrayIndexOutOfBoundsException: 3
Quiting...
```

4.2. Capturando Exceções 2

Há uma boa chance de que algumas classes escritas anteriormentes tenham disparados exceções. Como as exceções não foram tratadas, simplesmente interromperam a execução. Retorne a estes programas e implemente o tratamento de exceções.

Parceiros que tornaram JEDI™ possível

Instituto CTS
Patrocinador do DFJUG.

Sun Microsystems
Fornecimento de servidor de dados para o armazenamento dos vídeo-aulas.

Java Research and Development Center da Universidade das Filipinas
Criador da Iniciativa JEDI™.

DFJUG
Detentor dos direitos do JEDI™ nos países de língua portuguesa.

Banco do Brasil
Disponibilização de seus *telecentros* para abrigar e difundir a Iniciativa JEDI™.

Politec
Suporte e apoio financeiro e logístico a todo o processo.

Borland
Apoio internacional para que possamos alcançar os outros países de língua portuguesa.

Instituto Gaudium/CNBB
Fornecimento da sua infra-estrutura de hardware de seus servidores para que os milhares de alunos possam acessar o material do curso simultaneamente.