Tratamento de Exceções

1

Exceções

- Uma exceção representa uma situação que normalmente não ocorre e representa algo de estranho ou errado no sistema.
- Exceções são construções usadas para indicar condições anormais dentro de um programa.
- Em Java, exceções são classes derivadas da classe Exception.
- Java provê diversos tipos de exceções, mas, caso necessário, outras podem ser criadas pelo programador.

2

2

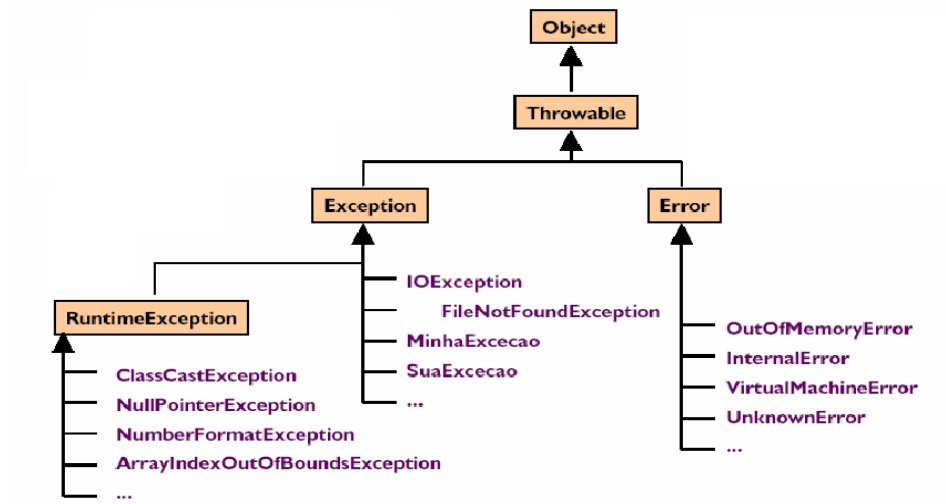
Classificação das Exceções

- Um objeto de exceção é sempre uma instância de uma classe derivada de Throwable.
- A hierarquia de classes de Throwable se divide em dois ramos: Error e Exception.
- Error descreve erros internos de execução da máquina virtual Java.
- Exception também se divide em dois ramos: exceções de RuntimeException ou não.

3

3

Hierarquia de Exceções



4

4

Verificação de Exceções

- Em Java qualquer exceção derivada das classes Error ou RuntimeException são chamadas de exceções não verificadas (*unchecked exception*).
- Todas as outras exceções são chamadas de exceções verificadas (*checked exceptions*).
- Um método deve lançar ou tratar todas as exceções verificadas.
- Exceções não verificadas estão além do seu controle, por isso não precisam ser tratadas.

5

5

Exceções

- Condições anormais são indicadas *lançando-se* exceções por meio da palavra-chave **`throw`**.

```
if (temperatura > 500)
    throw new SuperAquecimento();
```

- Métodos que podem lançar exceções devem indicar os tipos de exceção com a palavra-chave **`throws`** no final de suas assinaturas.

```
void aumentaTemperatura(int x) throws SuperAquecimento
{
   temperatura += x;

   if (temperatura > 500)
     throw new SuperAquecimento("Temperatura maior que 500 graus");
}
```

6

6

Situações de Lançamento de Exceções

- Uma exceção pode ser lançada nas seguintes situações:
  1. Chamando um método que lança uma exceção verificada, ex. O método readLine da classe BufferedReader.
  2. Detectando um erro e lançando uma exceção verificada com a declaração do comando **`throw`**.
  3. Cometendo um erro de programação, tal como a[-1] = 0 que cria uma exceção não verificada ArrayIndexOutOfBoundsException.
  4. Quando um erro interno acontece na máquina virtual ou na biblioteca de runtime.

7

7

Exceções

- Ao se chamar um método que pode gerar uma exceção, existem duas alternativas:
  - Tratar a possível exceção;
  - Passar o tratamento adiante.
- Para postergar o tratamento, basta indicar novamente que o método atual lança esta exceção.

```
void executaComando() throws SuperAquecimento {
    int temp = lerValorDoUsuario();
    aumentaTemperatura(temp);
}
```

8

8

Exceções

- Para postergar o tratamento de mais de uma exceção, basta indicar quais exceções o método lança, separadas por vírgula “,”.

```
void executaComando() throws SuperAquecimento, AltaPressao {
    int temp = lerValorDoUsuario();
    aumentaTemperatura(temp);
}
```

9

9

Exceções - Tratamento

- Em algum momento a exceção precisa ser tratada.
- O tratamento é feito com o bloco **`try`…`catch`**

```
void executaComando()
{
    int temp = lerValorDoUsuario();

    try {
        aumentaTemperatura(temp);
    }
    catch (SuperAquecimento sa) {
        desligar();
        alarme();
    }
}
```

10

10

Exceções - Tratamento

- Se necessário, mais de uma exceção pode ser tratada no mesmo bloco **try…catch**
- Para isso, é necessário apenas incluir mais um bloco **catch** após o **try**.

```
void executaComando() {
    int temp = lerValorDoUsuario();
    try {
       aumentaTemperatura(temp);
    }
    catch (AltaPressao ap) {
       abrir();
    }
    catch (SuperAquecimento sa) {
       desligar();
       alarme();
    }
}
```

11

11

Exceções - Tratamento

- O bloco **try…catch** pode ter opcionalmente uma cláususa **finally**, contendo um trecho de código que executará independentemente de ocorrer ou não a exceção.

```
void executaComando() {
    int temp = lerValorDoUsuario();
    try{
        aumentaTemperatura(temp);
    }
    catch (SuperAquecimento sa) {
        desligar();
        alarme();
    }
    finally {
        mostraTemperatura();
    }
}
```

12

12

Criando uma Exceção

- Se necessário você pode criar suas próprias exceções;
- Toda classe de exceção deve ser derivada de Exception ou de uma de suas classes filhas;
- Por conveniência, deve-se criar um construtor padrão e um com uma mensagem de parâmetro.

```
class SuperAquecimento extends Exception {
    SuperAquecimento(){
    }
    SuperAquecimento(String mensagem){
   super(mesagem);
    }
}
```

13

13