Exceções

Objetivos

- Construir classes de exceções
- Criar e lançar exceções
- Captura e tratar erros com exceções

Motivação

- Orientação a objetos
  - Diferente da programação estruturada
    - Campos das entidades são encapsuladas
  - Estado
    - Valores dos campos
  - Forma de alterar o estado de um objeto
    - Executando métodos!
- O que é exceção?
  - Método
    - Desejo de realizar uma operação com um objeto
    - Não há garantia de realização, pois pode haver erros
  - Exceções
    - Mecanismo para indicar a ocorrência de situações fora do esperado
  - Vai além do retorno de um código de erro
    - Obriga o programador a estar ciente do acontecimento

Trabalhando com exceções

A classe Exception

- Contida no pacote java.lang
- Utilizada para indicar condições anormais de execução de um método
- Principal construtor
  - Exception(String mensagem)
    - Recebe como argumento uma mensagem que descreve o erro
- Principais métodos
  - public String getMessage()
    - Retorna mensagem que descreve o erro
  - public void printStackTrace()
    - Imprime o conteúdo da pilha de execução
    - Indica onde foi o ponto de lançamento da exceção
- Criando suas próprias exceções
  - Como criar nossas próprias classes de exceções?
    - Programação Orientada a Objetos
    - Basta estender a classe Exception e usufruir dos seus métodos
  - Exemplo

```
public class SaldoInsuficienteException extends Exceptio
{   public SaldoInsuficienteException ()
    {   super("O saldo insuficiente para realizar a opera
    }
}
```

- Recomendação de codificação Java
  - Classes de exceções devem terminar com a palavra Exception

Lançando um exceção

- Adicionar a indicação `throws <ListaDeExeceções>` no final do cabeçalho de um método
- Exemplo

```
public boolean sacar(double quantia) throws     SaldoInsufic
{   //...
}
```

- Passos
  - Criação da exceção
  - Lançamento da exceção (com throw)
    - Interrompe imediatamente o processamento
    - Semelhante ao return
    - Diferença é o tipo de retorno
- Exemplos

```
public void sacar(int conta, double valor) throws SaldoInsu
{   //...
    Exception e = new SaldoInsuficienteException();
    throw e;
}

public void sacar(int conta, double valor) throws SaldoInsuf
{   //...
    throw new SaldoInsuficienteException();
}
```

Capturando exceções

- Bloco try/catch
  - Utilizado para executar métodos que possam lançar exceções
  - Exemplo com único catch

```
try
{   conta.sacar(1000.00);
} catch (SaldoInsuficienteException e)
{   System.out.println(e.getMessage());
}
```

  - Exemplo com múltiplos catchs

```
try
{   Conta conta = getConta(); //acessa uma conta de algur
    conta.sacar(1000.00);
} catch (NullPointerException e)
{   System.out.println("Conta não encontrada");
} catch (SaldoInsuficienteException e)
{   System.out.println(e.getMessage());
}
```

- Exemplo com múltiplos catchs

```
try
{   Conta conta = getConta(); //acessa uma conta de algur
    conta.sacar(1000.00);
} catch (SaldoInsuficienteException e)
{   System.out.println(e.getMessage());
} finally
{   caixaEletronico.voltarMenuInicial();
}
```

- Bloco try/catch/finally
  - Sempre executado com ou sem ocorrência de exceção
  - Exemplo

```
try
{   conta.sacar(1000.00);
} catch (SaldoInsuficienteException e)
{   System.out.println(e.getMessage());
} finally
{   caixaEletronico.voltarMenuInicial();
}
```

Runtime Exception

- Exceção especial
  - Captura e tratamento não é obrigatório em tempo de compilação
- Pode ser capturada e tratada como uma exceção normal
- Normalmente indica falta de programação defensiva
  - Exemplo
    - Conversão de uma String para um valor numérico inteiro

Referências bibliográficas

- HORSTMANN, C. S.; CORNELL, G. **Core Java - Fundamentos** - Vol 1. 8 ed. cap. 11 (Exceções, registro em log, assertivas e depuração).
- DEITEL, H. M.; DEITEL, P. J. **Java, Como programar**. 8 ed. cap. 11 (Tratamento de exceções).
- FURGERI, S. **Java 7 - Ensino Didático**. 1 ed. cap. 3 (Estruturas condicionais e de repetição).