DR1

Disciplina Regular 1 Fundamentos do Desenvolvimento Java

Graduação em Engenharia de Software - 2020

DR1

Etapa 4 Aula 1

Tratamento de Erros e Processamento de Arquivos

DR1

Competências Trabalhadas Nesta Etapa

- Implementar o tratamento de erros em programas Java.
  - Escrever o tratamento de exceções com try/catch/finally.
  - Escrever o lançamento de exceções com throws/throw.
  - Compreender e utilizar a hierarquia de exceções do Java.

No Moodle esse conteúdo se refere à etapa 5

Erros e Exceções

- Exceção é um evento que indica uma condição anormal que ocorreu dentro de um método.
- Características do tratamento de exceções em Java:
  - Não é possível ignorar a exceção: o tratamento é obrigatório em alguns casos.
  - Separa o código de tratamento de erro da lógica do programa.

Erros e Exceções

DR1

- Quando acontece um evento anormal, é criado um objeto representando a exceção.
- Métodos que podem lançar exceções são declarados com a palavra-chave **throws**.
- Um método lança (**throw**) uma exceção conforme uma condição testada.
- Qualquer código que possa dar erro é colocado dentro de um bloco **try**.
- Um manipulador captura (**catch**) a exceção e lhe dá o tratamento necessário.

Exemplo - Classe String

DR1

charAt

```
public char charAt(int index)
```

Returns the `char` value at the specified index. An index ranges from `0` to `length() - 1`. The first `char` value of the sequence is at index `0`, the next at index `1`, and so on, as for array indexing.

If the `char` value specified by the index is a surrogate, the surrogate value is returned.

Specified by:

`charAt` in interface `CharSequence`

Parameters:

`index - the index of the char value.`

Returns:

`the char value at the specified index of this string. The first char value is at index 0.`

Throws:

`IndexOutOfBoundsException - if the index argument is negative or not less than the length of this string.`

DR1

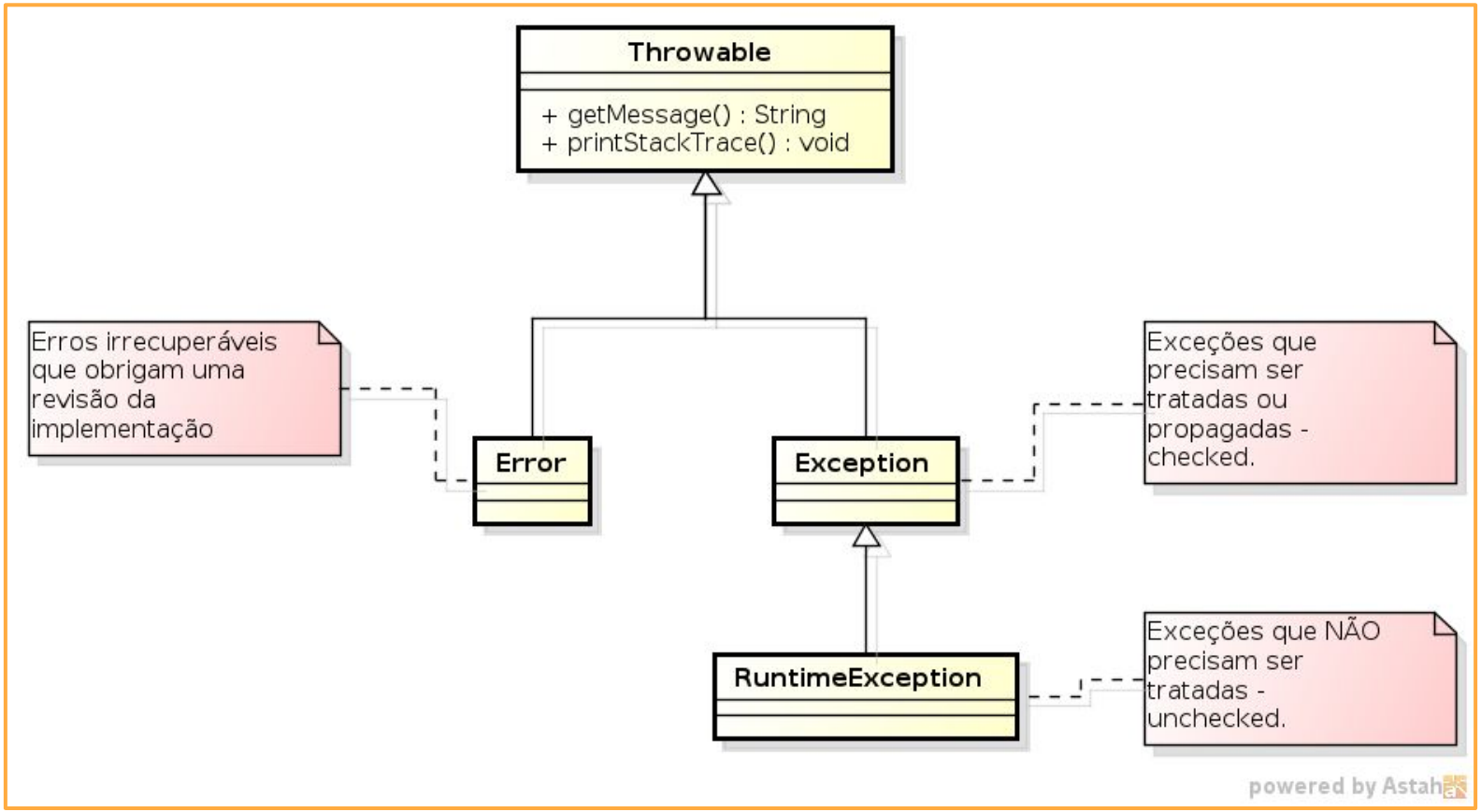
Exemplo - Classe String

```
1  package br.edu.infnet.laboratorio1;
2
3  import java.util.Scanner;
4
5  public class TesteExcecoes {
6
7      public static void main(String[] args) {
8
9
10         System.out.println("Passo 1 - fora do try - inicio");
11         try {
12
13             Scanner scanner = new Scanner(System.in);
14             System.out.print("Digite uma frase > ");
15             String frase = scanner.nextLine();
16             System.out.println("Setimo caractere = " + frase.charAt(6));
17             System.out.println("Passo 2 - fim do try");
18         } catch(Exception e) {
19
20             System.out.println("Passo 3 - dentro do catch");
21         } finally {
22
23             System.out.println("Passo 4 - dentro do finally");
24         }
25         System.out.println("Passo 5 - fora do try - fim");
26     }
27 }
```

```
Passo 1 - fora do try - inicio
Digite uma frase > laboratorio
Setimo caractere = t
Passo 2 - fim do try
Passo 4 - dentro do finally
Passo 5 - fora do try - fim
```

```
Passo 1 - fora do try - inicio
Digite uma frase > abc
Passo 3 - dentro do catch
Passo 4 - dentro do finally
Passo 5 - fora do try - fim
```

Erros e Exceções



Tratar ou Propagar Exceções

DR1

- Para propagar, coloque **throws** na declaração do método.
- A exceção passa para o método que a chamou.
- Podemos listar quantas exceções quisermos na cláusula **throws** bastando, para isso, separar os nomes das classes por vírgula.

```
public boolean abreArquivoConfiguracao ()  throws IOException, ExcecaoDesejada {
    // código que pode lançar IOException
    throw new IOException();
    ...
    throw new ExcecaoDesejada();
}
```

Exemplo - Disparando Exceções

```java
package br.edu.infnet.laboratorio1;

import java.util.Scanner;

public class TesteNaoGostoPares {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);
        System.out.print("Digite um numero > ");
        int numero = scanner.nextInt();
        try {

            verificar(numero);
            System.out.println("OK, tudo bem...");
        } catch (Exception e) {

            System.out.println(e.getMessage());
        }
    }

    private static void verificar(int numero) throws Exception {

        if(numero % 2 == 0) {

            throw new Exception("Não gosto de números pares");
        }
    }
}
```

Exercício

DR1

- Criar as classes, conforme o diagrama.
- Criar o método toString nas duas classes, retornando os conteúdos das propriedades.
- Criar a implementação dos métodos depositar e sacar considerando as seguintes regras:
  - Não é possível sacar mais do que o saldo permita - ContaCorrente.
  - Não é possível sacar mais do que o saldo + limite permita - ContaEspecial.
  - Não é possível depositar e sacar valores negativos.
- Disparar exceções do tipo Exception nos casos das regras serem violadas.
- Criar uma classe TestaConta para testar o funcionamento de uma ContaCorrente e uma ContaEspecial.