
ORACLE
Academy

Java Fundamentals

6-2

Processamento de erros

ORACLE
Academy

Objetivos

- Esta aula abrange os seguintes tópicos:
  - Descrever os diferentes tipos de erros que podem ocorrer e como eles são tratados no Java
  - Descrever quais exceções são usadas no Java
  - Determinar quais exceções são lançadas em qualquer classe de base
  - Escrever o código para tratar uma exceção lançada pelo método de um classe de base

Tipos de Erros

- Um erro indica que há um problema com a interpretação do programa
- Há três tipos de erros:
  - Erros de sintaxe
  - Erros lógicos
  - Exceções (erros de run-time)

Erros de sintaxe não serão compilados em Java. Erros lógicos serão compilados, mas o resultado poderá ou não ser o esperado. Os programas que lançam exceções serão compilados e começarão a execução, mas travarão ou, se a exceção for tratada, serão executados até o término.

Erros de Sintaxe

- Pode ocorrer um erro quando um arquivo é compilado
- Esses erros são erros de codificação e de sintaxe, como:
  - Pontos-e-vírgulas ausentes
  - Erros de ortografia
  - Atribuir um valor a uma variável que não é do tipo correto
- É comum passar por algumas rodadas de erros de sintaxe de fixação antes do sucesso na compilação do arquivo

A maioria dos IDEs modernos realçará erros de sintaxe de modo que você não precise tentar compilar seu código para encontrá-los.

Erros de Sintaxe: Pontos-e-vírgulas ausentes e Símbolos Incorretos

- Esquecer um ponto-e-vírgula no final de uma instrução Java é um erro de sintaxe

```
Exception in thread "main" java.lang.Error: Unresolved compilation
problem:
    Syntax error, insert ";" to complete Statement
    at ErrorIndicators.main(ErrorIndicators.java:8)
```

- O uso de =, em vez de ==, para comparar valores em uma condição if é um erro de sintaxe

```
if(x=y){
   ...
}//fim if
```

Incorreto

```
if(x==y){
    ...
}//fim if
```

Correto

ORACLE Academy

Erros de Sintaxe: Erros de Ortografia em Métodos ou Variáveis

- Errar a ortografia de uma variável ou do nome de um método é um erro de sintaxe
- Verifique se você escreveu o nome da variável ou do método da mesma forma que o declarou
- Para corrigir esse erro, verifique a ortografia de:
  - A variável ou o método onde ele está declarado
  - Onde você chama a variável ou o método

```
Exception in thread "main" java.lang.Error:
        Unresolved compilation problem:
       variableName cannot be resolved to a variable
      at MisspellingVariables.main(MisspellingVariables.java:7)
```

Erros Lógicos

- Erros lógicos podem ocorrer em decorrência de uma lógica incorreta do programador
- Esses erros não produzem um erro de compilação ou de tempo de execução
- Por exemplo, um loop é executado muitas vezes, ou o programa produz um resultado incorreto

Erros Lógicos

- Colocar um ponto-e-vírgula após uma condição if ou inicializar um loop:
  - Os mecanismos de interpretação leem o ponto-e-vírgula como o final do loop, o que significa que tudo que vier após o ponto-e-vírgula será tratado como fora do loop

```
for(int i = 0; i < 5; i++);
    System.out.println(i);
```

Esta instrução será executada somente uma vez. Por quê?

Exemplos de Erros Lógicos

- O uso de == para comparar Strings ou objetos é um erro lógico, a menos que você esteja verificando se eles se referem ao mesmo objeto

```
String s1 = "Hello";
String s2 = "Goodbye";
if(s1==s2)
   System.out.println("They are equal");
```

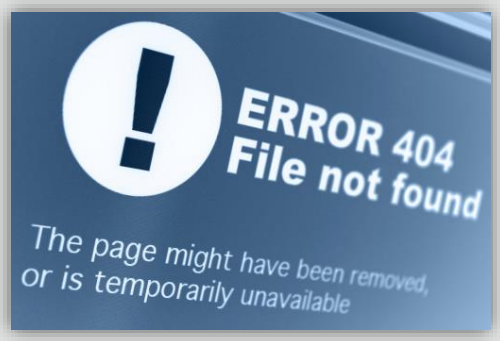
O == verificará se essas Strings são o mesmo objeto. Para verificar se os conteúdos das Strings são iguais, use o método equals() da classe String.

- Por causa deste ponto-e-vírgula, "Hello" será sempre exibido, mesmo se x for menor que 10

```
if(x > 10);
   System.out.println("Hello");
```

Exceções

- Quando um arquivo é compilado com sucesso, pode ocorrer um erro no momento de testá-lo no runtime
- Esses erros de runtime são chamados de exceções e devem ser tratados pelo programador usando código no programa, situação conhecida como “lançar” a exceção



Categorias de Exceções Java

- As exceções Java se enquadram em duas categorias:
  - Exceções não verificadas (erros no código)
  - Exceções verificadas (erros que ocorrem do lado de fora do código); a maior parte se origina de programas com Entrada/Saída (ES)

O programador tem a opção de verificar a existência de exceções não verificadas. Por exemplo, se uma ArrayOutOfBoundsException for lançada e não tratada, o programa travará. Se ela for tratada, o programa poderá ser concluído. As exceções verificados devem ser tratadas com um bloco try-catch, mesmo que o programador não queira tratá-las, ou lançadas para o método de invocação (pai). O programa não será compilado sem tratar todas as exceções verificadas.

Criando um Bloco Try/Catch

- Um bloco try/catch pode tratar de exceções verificadas e não verificadas
- Veja abaixo um exemplo desse bloco

```
try{
        ...code that might cause an exception
}

catch (exception e){
        ...code to handle the exception
}
```


O objeto Exception pode ser usado para invocar métodos, como getMessage(), para acessar informações sobre a exceção.

Exceções Não Verificadas

- O tratamento de exceções não verificadas em Java é opcional
- No entanto, se a exceção não verificada não for tratada e ocorrer um erro, o programa será interrompido
- Exceções não verificadas comuns:
  - Índice de exceção fora dos limites
  - Exceção de ponteiro nulo

Índice de Exceção Fora dos Limites

- Chame example[3] no seguinte array inicializado

```
int[] example = {1, 2, 3};
```

- A mensagem de exceção a seguir será exibida porque example[3] não existe no array

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 3
   at NullPointerTest.main(NullPointerTest.java:6)
```

- O array começa no índice 0 e termina no índice 2
- Chamar o índice 3 significa que você está fora dos limites do array

```
example[0] = 1;
example[1] = 2;
example[2] = 3;
```

Exceção de Ponteiro Nulo

- O código a seguir gerará uma exceção de ponteiro nulo porque os objetos da String no array não foram inicializados

```
String[] s1 = new String[5];
System.out.println(s1[0]);
```

- Para corrigir o problema, atribua valores às 5 Strings do array antes de tentar imprimir os valores

Exceção de Arquivo Não Encontrado

- Uma exceção de Arquivo Não Encontrado é uma exceção de E/S (Entrada/Saída)
- As exceções de E/S são exceções verificadas
- A maioria das exceções verificadas vem do uso de classes de E/S

Tratando de Exceções Verificadas

- As exceções verificadas devem ser tratadas
- Há duas formas de tratar uma exceção verificada:
  - Usar um bloco try/catch, que tratará da exceção “graciosamente”
  - Usar uma instrução “throws” na declaração do método, o que é um risco
  - Uma instrução throws é quando um programador diz que correrá o risco de uma exceção não ser gerada

Exceção de E/S

- Este código usa um bloco try/catch para tratar de uma exceção de E/S

```
try{
   FileReader reader = new FileReader("test.txt");
}//fim try
catch(IOException e){
   System.out.println("File not found");
}//fim catch
```

Exceção de Arquivo Não Encontrado

- Usa uma instrução “throws” para tratar uma Exceção de E/S
- A instrução “throws” é usada para alertar que pode haver uma exceção gerada No entanto, o programa ainda será interrompido caso ocorra um erro que gere uma exceção

```
public static void main(String[] args) throws IOException{

   FileReader reader = new FileReader("test.txt");
}//fim do método main
```

Lançando Exceções

- Até agora, você viu exceções tratadas com o uso de um bloco try/catch
- Você também pode tratar de exceções lançando-as
- Se você lançar uma exceção, o mecanismo de interpretação interromperá a execução do programa nesse ponto, o que indica ao usuário que ele atingiu a exceção
- No código, uma exceção é gerada da seguinte forma:

```
throw new Exception("Array index" + i + " is out of bounds!");
```

Capturando Exceções

- Capturar uma exceção significa tratá-la
- Você pode lançar uma exceção para alguns casos, como sair dos limites de um array, e capturar a exceção para continuar o programa da forma que escreveu para tratar da exceção
- Um bloco try/catch permite que você faça isso

Exemplo de Try/Catch

- Este é um exemplo do tratamento de uma exceção Índice Fora dos Limites com um bloco try/catch

```
try{
    //i é o índice de um array com tamanho 10
    if(i > 9 || i < 0)
         throw new Exception("Index " + i + " is out of bounds!");
}//fim try
catch(Exception e){
//Este código será executado somente se a exceção tiver sido gerada
    if(i > 9)
       i-=9;
    else
       i+=9;
}//fim catch
//Você pode ter códigos adicionais aqui que serão executados
//somente se a exceção não tiver sido gerada
```

Situações de Geração e Captura de Exceções

- Você está escrevendo um programa que move uma tartaruga para uma parte do oceano que o usuário especifica
- Como o programa seria executado se o usuário inserisse coordenadas que não estivessem no mapa do oceano onde a tartaruga pode se mover?
- Essa exceção estaria fora do controle do programador
- Pense em gerar e capturar uma exceção para resolver esse problema

Terminologia

- Os principais termos usados nesta aula foram:
  - Catch
  - Exceções verificadas
  - Erro
  - Exceção
  - Erro lógico
  - Erro de runtime
  - Erro de sintaxe
  - Throw
  - Bloco try/catch
  - Exceções não verificadas

Resumo

- Nesta aula, você deverá ter aprendido a:
  - Descrever os diferentes tipos de erros que podem ocorrer e como eles são tratados no Java
  - Descrever quais exceções são usadas no Java
  - Determinar quais exceções são lançadas em qualquer classe de base
  - Escrever o código para tratar uma exceção lançada pelo método de um classe de base

ORACLE Academy

As exceções são abordadas com mais detalhes em Programação Java.


ORACLE
Academy