

Lập trình Java

Tuần 8: cách bắt lỗi try catch, throw, throws.

Nội dung

- Tìm hiểu về cách bắt lỗi try catch
- Throw và throws

Try catch

Exception là gì?

**Vd: chia cho 0, index lớn hơn, connect database,...
để tránh break ra khỏi chương trình một cách
không mong muốn.**

Try catch

Cú pháp:

```
try {  
    // câu lệnh  
}  
catch(Exception e) {  
    // câu lệnh  
}
```

Try catch

```
public class MyClass {  
    public static void main(String[ ] args) {  
        int[] myNumbers = {1, 2, 3};  
        System.out.println(myNumbers[10]); // error!  
    }  
}
```

Error:

Exception in thread "main"
java.lang.ArrayIndexOutOfBoundsException: 10

at MyClass.main(MyClass.java:4)

Try catch

```
public class MyClass {  
    public static void main(String[ ] args) {  
        try {  
            int[] myNumbers = { 1, 2, 3};  
            System.out.println(myNumbers[10]);  
        } catch (Exception e) {  
            System.out.println("this is error ");  
            Out put: this is error .  
        }  
    }  
}
```

Try catch

```
public class MyClass {  
    public static void main(String[ ] args) {  
        try {  
            int[] myNumbers = { 1, 2, 3};  
            System.out.println(myNumbers[10]);  
        } catch (ArrayIndexOutOfBoundsException e) {  
            System.out.println("this is error");  
            Out put: this is error .  
        }  
    }  
}
```

Try catch

```
public class MyClass {  
  
    public static void main(String[ ] args) {  
  
        try {  
  
            int[] myNumbers = { 1, 2, 3 };  
  
            a=1; b= 0;  
  
            c = a/b;  
  
            System.out.println("Value =" + c);  
  
            System.out.println(myNumbers[1]);  
        } catch (ArrayIndexOutOfBoundsException | ArithmeticException) {  
            System.out.println("this is error");  
        }  
    }  
}
```


Try catch

Sinh viên tự viết cách chỉnh sửa đoạn code trên.

Try catch

Dùng try có nhiều catch

Các lệnh catch thường được viết theo thứ tự xuất hiện của ngoại lệ.

Chú ý:

catch Exception phải được đặt ở vị trí sau cùng nếu như có nhiều catch.

Try catch

```
public class MyClass {  
  
    public static void main(String[] args) {  
  
        try {  
  
            int[] myNumbers = {1, 2, 3};  
  
            a=1; b= 0;  
  
            c = a/b;  
  
            System.out.println(myNumbers[10]);  
  
        } catch (ArithmeticException e){  
  
            System.out.println("Lỗi chia cho 0");  
  
        } catch (Exception e) {  
  
            System.out.println("lỗi khác");  
  
        }  
  
    }  
  
}
```

Lỗi chia cho 0 của
catch (ArithmeticException)
sẽ được xử lý trước.
Các lỗi khác sẽ được xử lý
trong catch (Exception).
Nếu thay đổi vị trí của 2 catch
thì điều gì xảy ra?

Try catch finally

Mỗi try phải có ít nhất một catch hoặc một finally. Khối finally sẽ luôn thực hiện.

try \Rightarrow catch \Rightarrow finally

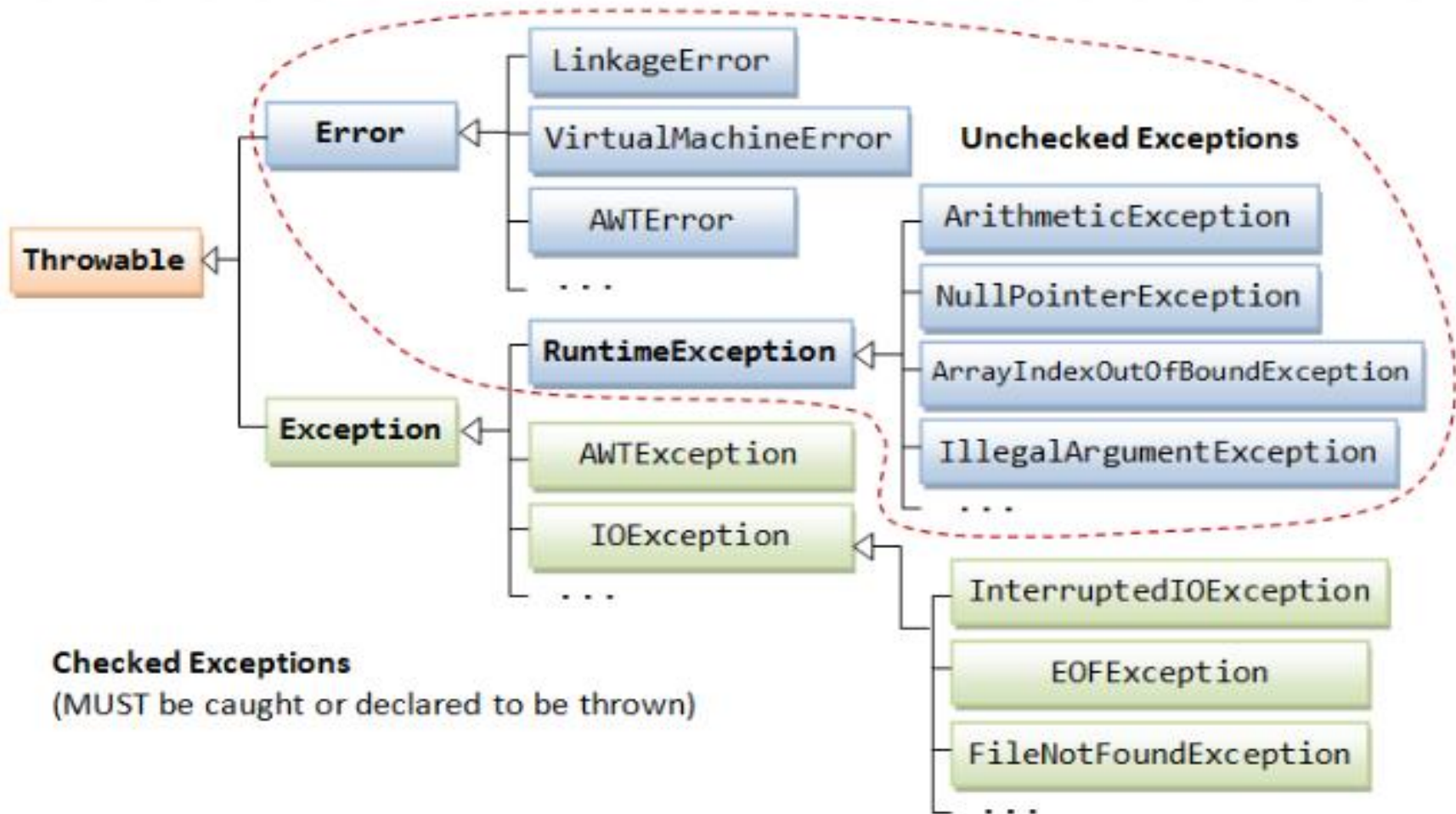
try \Rightarrow catch

try \Rightarrow finally

Khởi try-catch-finally

```
public class MyClass {  
  
    public static void main(String[ ] args) {  
  
        try {  
  
            int[] myNumbers = {1, 2, 3};  
  
            System.out.println(myNumbers[10]);  
  
        } catch (Exception e) {  
  
            System.out.println("this is error ");  
  
        } finally {  
  
            System.out.println("The 'try catch' is finished.");  
  
        }  
    }  
}
```

sơ đồ phân cấp Exception



Sơ đồ phân cấp Exception

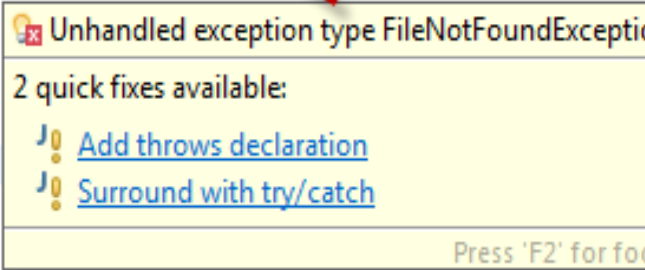
- `RuntimeException`: không được java kiểm tra trong thời điểm biên dịch.
- `Error`: lỗi hệ thống, nghiêm trọng liên quan tới môi trường thực thi hoặc hệ thống mà người lập trình khó kiểm soát, thường làm chết chương trình. (Ví dụ: `OutOfMemoryError`, `VirtualMachineError`, `StackOverflowError`, ...)
- Hầu hết các lỗi khác không nghiêm trọng thường được xét là `Exception`.
- Chú ý: không viết lớp kế thừa lớp `Throwable`



Sơ đồ phân cấp Exception

- Checked Exception: là các lỗi được kiểm tra ngay tại lúc code, bắt buộc ta phải xử lý (handle) nó.
- Ví dụ: IOException, FileNotFoundException, NoSuchFieldException,

```
3 import java.io.FileReader;
4
5 public class ExceptionExample2 {
6
7     public static void main(String[] args) {
8
9         FileReader f = new FileReader("File is not exists");
10    }
11
12 }
13
14
15
16
17
18
```



Unhandled exception type FileNotFoundException

2 quick fixes available:

- [Add throws declaration](#)
- [Surround with try/catch](#)

Press 'F2' for focus

Sơ đồ phân cấp Exception

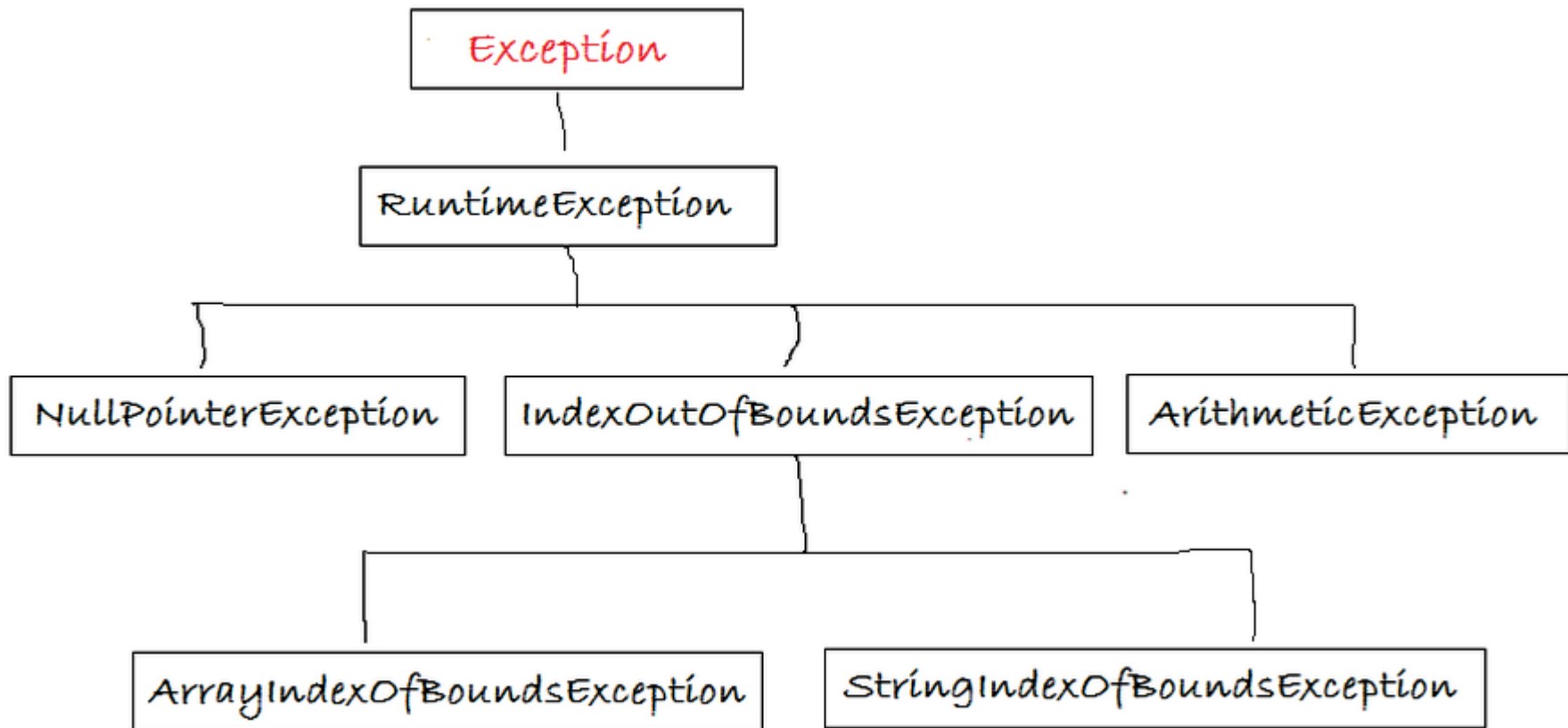
• **Unchecked Exception**: là các lỗi xuất hiện khi đang chạy chương trình. Các exception này còn gọi là runtime exceptions, đó chính là programming bugs, lỗi logic của chương trình. Không bắt buộc phải handle.

•- Ví dụ:

• **NullPointerException**, **NumberFormatException**,
ArrayIndexOutOfBoundsException,
DivideByZeroException, ...



Sơ đồ phân cấp Exception



Sơ đồ phân cấp Exception

Trong đó:

- `NullPointerException`: phương thức hoặc truy cập vào các trường của một đối tượng chưa được khởi tạo (đối tượng null).

Ví dụ:

```
String obj = null;
```

```
System.out.println(obj.length()); // NullPointerException
```

- `ArrayIndexOutOfBoundsException`, `StrIndexOutOfBoundsException`: truy cập vào phần tử có chỉ số không hợp lệ trên mảng.

- `ArithmeticException`: lỗi liên quan tới xử lý số học. Ví dụ như chia cho 0.

Chú ý: khi có nhiều exception thì exception con phải được đặt ở trước exception cha.

Throw, Throws

Từ khóa Throw dùng để ném (throw) ra một ngoại lệ cụ thể (ngoại lệ do người dùng tự định nghĩa).

Cú pháp: `throw exception;`

Ví dụ: `throw new IOException("File không tồn tại");`

Throw, Throws

Ví dụ:

```
public class TestThrow2 {  
    static void validate(int age) {  
        try {  
            if (age < 18)  
                throw new ArithmeticException("not valid");  
            else  
                System.out.println("welcome");  
        } catch (ArithmeticException ex) {  
            System.out.println(ex.getMessage()); } }  
    public static void main(String args[]) {  
        validate(13);  
        System.out.println("rest of the code..."); } }
```

Out put:

not valid

rest of the code...

Throw, Throws

Từ khóa Throws dùng để xử lý những ngoại lệ phát sinh trong method. Có những method khi viết sẽ phát sinh ra lỗi “checked” nên chúng ta bắt buộc phải xử lý.

```
public void ghifile() throws IOException{
    FileWriter file = new FileWriter("data.txt");
    file.write("Xu ly ngoai le trong java");
    file.write(100);
    System.out.println("Da ghi xong !");
    file.close();
}

public static void main(String[] args) throws IOException {
    throwsexampel obj = new throwsexampel();
    obj.ghifile();
    System.out.println("Su dung tu khoa throws");
}
}
```

Throw, Throws

Chúng ta có thể viết class xử lý ngoại lệ của riêng mình bằng cách kế thừa class Exception.

```
public class myexception extends Exception {  
  
    private int message;  
  
    myexception(int a) {  
        message = a;  
    }  
  
    @Override  
    public String toString() {  
        return "My exception " + message;  
    }  
}
```

Throw, Throws

```
static void tinhtoan(int a) throws myexception {  
    if (a > 10) {  
        throw new myexception(a);  
    }  
    System.out.println("Normal exit");  
}  
  
public static void main(String args[]) {  
    try {  
        tinhtoan(1);           //khong tao ra exception  
        tinhtoan(20);          //tao ra exception  
    } catch (myexception e) {  
        System.out.println("Caugh " + e);  
    }  
}  
}
```


Bài tập tuần 8

Throw, Throws

Đọc 1 file sau đó đếm các từ có trong file. Nếu số từ lớn hơn 50 từ thì xem như là lỗi (throw vào myexception tự tạo). Viết hàm main test chương trình.