Fpt Polytechnic

BÀI 2
Xử lý ngoại lệ

FPT Fpt University
FPT POLYTECHNIC

Nhắc lại bài cũ

Package và Interface

- Sử dụng package
- Các từ khóa public, protected, private
- Các package chuẩn của Java
- Khái niệm inteface
- Thực thi interface
- Kế thừa interface

FPT Fpt University
FPT POLYTECHNIC

Nội dung bài học

1. Sử dụng khối try...catch để xử lý ngoại lệ

2. Sử dụng final trong khối try...catch

3. Sử dụng từ khóa throws và throw

FPT POLYTECHNIC

1. Sử dụng khối try... catch để xử lý ngoại lệ

Trong phần này có các nội dung:

1.1. Cơ bản về ngoại lệ (Exception)

1.2. Sử dụng try... catch để xử lý ngoại lệ

- try có nhiều catch
- khối try lồng nhau

1.1 Cơ bản về ngoại lệ

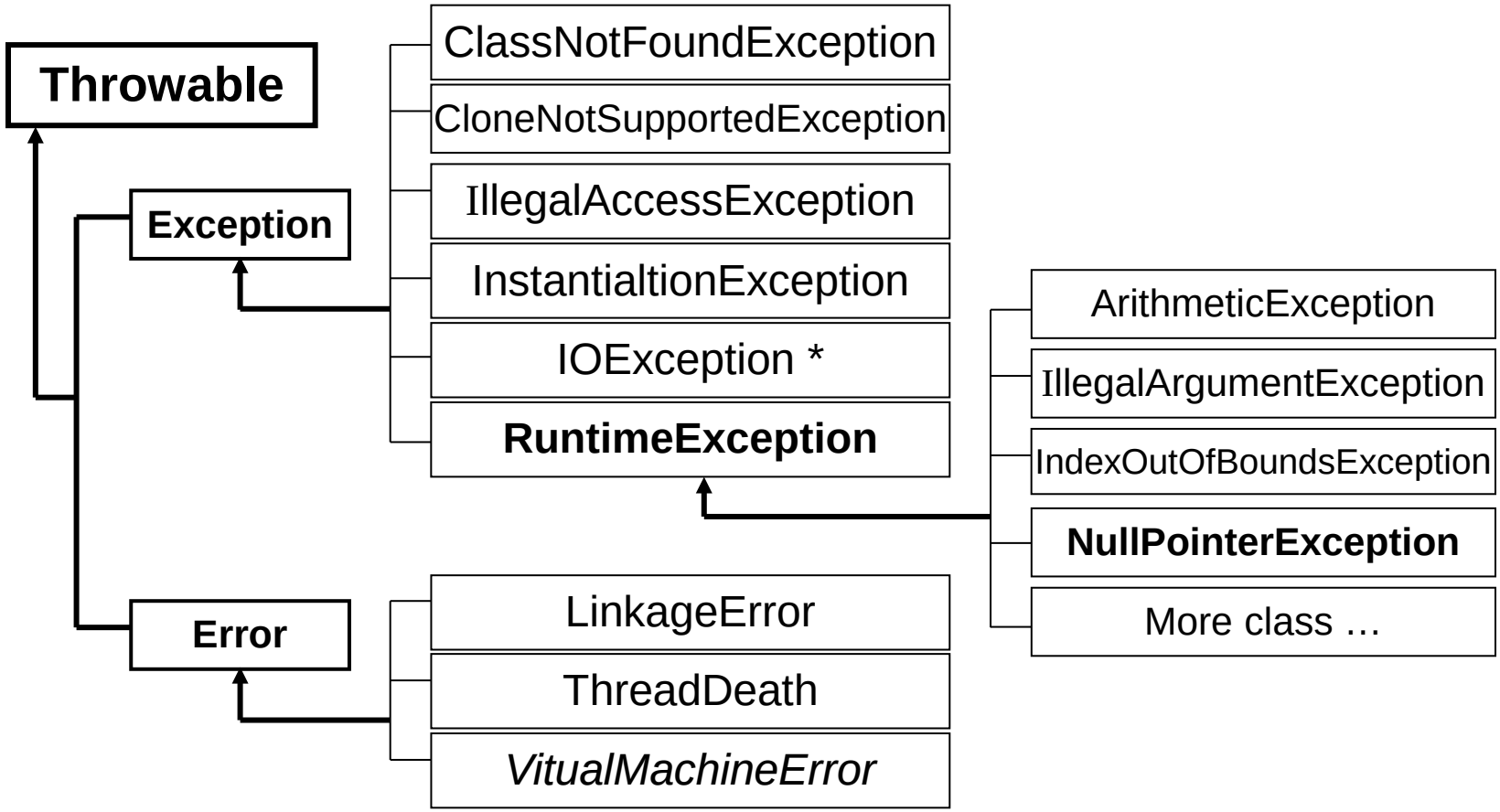
Ngoại lệ là gì?

- Có những lỗi chỉ khi chạy chương mới xuất hiện và chương trình đang chạy lập tức ngừng lại và xuất hiện thông báo lỗi – đó chính là ngoại lệ (exception).
- Ví dụ: Chương trình chia 2 số. Nếu ta cho mẫu số =0 thì phát sinh lỗi và đó được coi là 1 ngoại lệ.

5/0 ⇨ error !

1.2. Sử dụng khối try... catch để xử lý ngoại lệ

- Class Throwable xử lý lỗi và ngoại lệ (Error, Exception).
- Tất cả các class dưới đây đều nằm trong gói java.lang, ngoại trừ class IOException là nằm trong gói java.io



FPT POLYTECHNIC

1.1 Cơ bản về ngoại lệ

Class Exception

- Có nhiều ngoại lệ là lớp con của lớp Exception
  - RuntimeErrorException là lớp con của lớp Exception
  - RuntimeErrorException là các ngoại lệ chỉ xảy khi chạy chương trình.
- Người lập trình có thể tự tạo các class kế thừa từ class Exception.

Class Error

- Chỉ những lỗi nghiêm trọng và không dự đoán trước được như VirtualMachineError, LinkageError, ThreadDead…
- Các ngoại lệ Error ít được xử lý

1.2. Sử dụng khối try… catch để xử lý ngoại lệ

Ngoại lệ '*unchecked*':

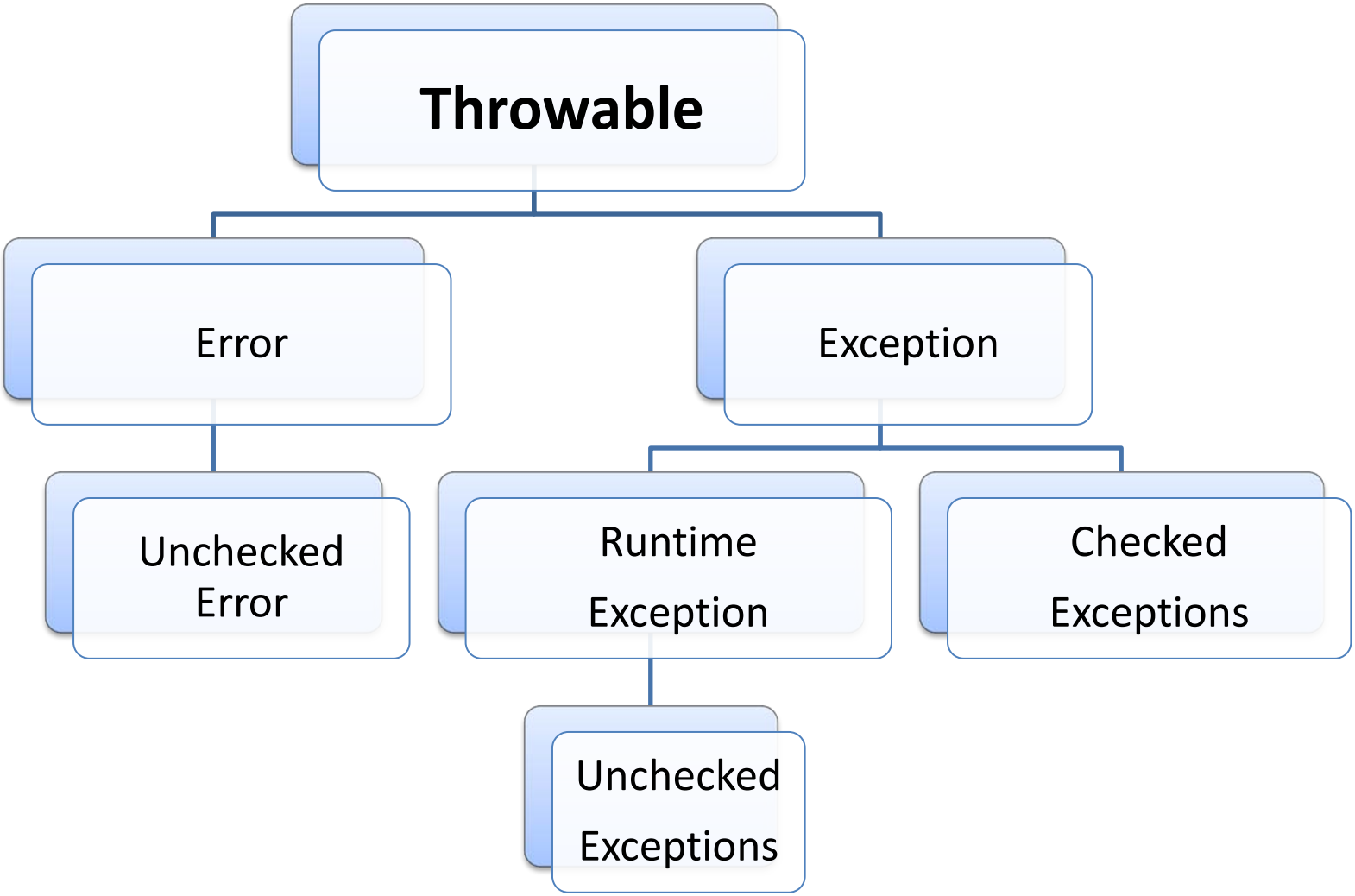
- Là các ngoại lệ không cần phải 'catch' khi viết mã
- Là các class Error, RuntimeException và các lớp con của chúng

Ngoại lệ '*checked*':

- Là các ngoại lệ phải được 'catch' khi viết mã
- Là các class còn lại

FPT POLYTECHNIC

1.2. Sử dụng khối try... catch để xử lý ngoại lệ



FPT POLYTECHNIC

1.2. Sử dụng khối try… catch để xử lý ngoại lệ

Một số ngoại lệ 'checked':

- ClassNotFoundException
- IOException
  - FileNotFoundException
  - EOFException

Một số ngoại lệ 'unchecked'

- ArithmeticException
- IllegalArgumentException
- IndexOutOfBoundException
- NullPointerException
- InputMismatchException

1.2. Sử dụng khối try… catch để xử lý ngoại lệ

Sử dụng từ khóa try và catch

```
try{
    //Khối lệnh
}catch(…){
    //Khối lệnh xử lý ngoại lệ
}
```

1.2. Sử dụng khối try… catch để xử lý ngoại lệ

Ví dụ: Nếu không dùng try… catch, xét ví dụ sau:

```
c=a/b;
System.out.println(“Sau phep chia !”);(*)
```

Câu lệnh (*) sẽ không được thực hiện nếu mẫu số b=0, chương trình lập tức ngừng lại và xuất hiện thông báo lỗi của hệ thống

1.2. Sử dụng khối try... catch để xử lý ngoại lệ

Ví dụ:

```
try{
    c=a/b;
}catch(Exception e){
          System.out.println("Có lỗi "+e);
}
System.out.println("Sau phép chia !"); (*)
```

Câu lệnh (*) sẽ luôn được thực hiện dù mẫu số b=0 hay b!=0.

1.2. Sử dụng khối try... catch để xử lý ngoại lệ

Dùng try có nhiều catch

- Trong một đoạn code có thể có nhiều ngoại lệ xảy ra nên ta sẽ dùng nhiều catch để xử lý các ngoại lệ đó.
- Các lệnh catch thường được viết theo thứ tự xuất hiện của ngoại lệ.
- Chú ý: Tất cả các ngoại lệ sẽ là lớp con của class Exception nên catch cuối cùng sẽ là Exception.

1.2. Sử dụng khối try... catch để xử lý ngoại lệ

Dùng try có nhiều catch

```
public static void main(String[] args) {
    {
        try {
            int b = 0;
            int a = 10 / b;
            int c[] = {2};
            c[5] = 3;
            System.out.println("a = "+a);
        } catch (ArithmeticException e) {
            System.out.println("Phep chia cho 0 : " +e);
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Vuot ngoai chi so mang : "+e);
        }
    }
}
```

FPT POLYTECHNIC

1.2. Sử dụng khối try... catch để xử lý ngoại lệ

```
public static void main(String[] args) {
    {
        try {
            int b = 0;
            int a = 10 / b;
            int c[] = {2};
            c[5] = 3;
            System.out.println("a = "+a);
        } catch (ArithmeticException e) {    // (1)
            System.out.println("Phep chia cho 0 : " +e);
        } catch (Exception e) {              //(2)
            System.out.println("Co loi : "+e);
        }
    }
}
```

Nếu xuất hiện ngoại lệ phép chia cho 0 thì lệnh (1) sẽ xử lý, còn các ngoại lệ khác sẽ được xử lý bởi lệnh (2). Nếu đặt (2) đổi chỗ cho (1) thì (2) sẽ xử lý luôn ngoại lệ chia cho 0 vì như thế không cần (1) nữa. Vì thế không thể thay đổi vị trí giữa lệnh (1) và lệnh (2)

1.2. Sử dụng khối try... catch để xử lý ngoại lệ

Khối try lồng nhau

```
try {
    int a = 2;  //cho a=0 hoặc a=1 hoặc a=2 để test
    int b = 42 / a;
    System.out.println("Ket qua phep chia b=" + b);
    try {
        if (a == 1) {
            a = a / (a - a);
        }
        if (a == 2) {
            int c[] = {2};
            c[5] = 3;
        }
    } catch (ArrayIndexOutOfBoundsException e) {
        System.out.println("Có lỗi vượt quá chỉ số mảng");
    }
} catch (ArithmeticException e) {
    System.out.println("Có lỗi: mẫu số = 0");
```

1.2. Sử dụng khối try... catch để xử lý ngoại lệ

Trong khối finally sẽ chứa một khối mã sẽ thực hiện sau khối try/catch. Khối finally sẽ được thực hiện dù ngoại lệ có xuất hiện hay không. Tuy nhiên, mỗi try sẽ yêu cầu có ít nhất 1 catch hoặc 1 finally.

- try ⇨ catch ⇨ finally
- try ⇨ catch
- try ⇨ finally

FPT POLYTECHNIC

2. Sử dụng từ khóa final trong try... catch

```
static void proA(){
	try{
		System.out.println(“Trong phương thức proA");
		throw new RuntimeException("Demo");	(1)
	}
	finally{
		System.out.println(“Trong khối finally của proA”);
	}

}
```

FPT Fpt University
FPT POLYTECHNIC

2. Sử dụng từ khóa final trong try... catch

```
static void proB(){
      try{
            System.out.println("Trong phương thức proB");
            return;
      }
      finally{
            System.out.println("Trong khối finally của proB");
      }

}
```

FPT Fpt University
FPT POLYTECHNIC

2. Sử dụng từ khóa final trong try... catch

```
static void proC(){
      try{
            System.out.println("Trong phương thức proC");

      }
      finally{
            System.out.println("Trong khối finally của proC");
      }
}
```

Nhận xét:

- ở procA() có tạo ra ngoại lệ mà vẫn chạy khối finally
- ở procB() có return mà vẫn chạy khối finally.

FPT POLYTECHNIC

3. Sử dụng từ khóa throws và throw

Từ khóa throws

Từ khóa throws được sử dụng trong method dùng để đề xuất các ngoại lệ có thể xảy ra trong method đó. Có những method sử dụng một số lệnh mà các lệnh đó có thể xảy ra ngoại lệ 'checked' nên chúng ta bắt buộc phải xử lý ngoại lệ đó. Ví dụ khi xử lý các lệnh thao tác với file, phải xử lý ngoại lệ 'checked' FileNotFoundException. Tất cả các ngoại lệ được khai báo bởi throws đều phải được xử lý, nếu không có đủ sẽ bị thông báo lỗi.

FPT POLYTECHNIC

3. Sử dụng từ khóa throws và throw

Ví dụ 1:

```
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();
    }
```

3. Sử dụng từ khóa throws và throw

Ví dụ 1 (tiếp):

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

FPT POLYTECHNIC

3. Sử dụng từ khóa throws và throw

Ví dụ 2: Dùng cách throws trong phương thức main

```
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");
    }
}
```

FPT POLYTECHNIC

3. Sử dụng từ khóa throws và throw

- Thông thường các exception sẽ được 'ném' ra bởi hệ thống Java runtime. Tuy vậy ta vẫn có thể lập trình để 'ném' ra các ngoại lệ khi gặp một tình huống nào đó trong khi lập trình.
- Trong một phương thức có thể throw nhiều ngoại lệ.
- Có 2 cách để 'ném' (throw) ra các ngoại lệ:
  - Dùng toán tử new
  - Đưa 1 tham số vào mệnh đề catch.
- Ví dụ:

```
if (check==0)
    throw new NullPointerException();
```

FPT Fpt University
FPT POLYTECHNIC

3. Sử dụng từ khóa throws và throw

```
public class throwDemo {
    static void demoProc() {
        try {
            throw new NullPointerException("demo");
        } catch (NullPointerException e) {
            System.out.println("Ben trong xu ly ngoai lệ demoPro");
            throw e;
        }
    }
    public static void main(String args[]) {
        try {
            demoProc();
        } catch (NullPointerException e) {
            System.out.println("Trong main, tiep tuc xu ly ngoai le");
        }
    }
}
```

FPT POLYTECHNIC

3. Sử dụng từ khóa throws và throw

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

```
public class myexception extends Exception {

    private int message;

    myexception(int a) {
        message = a;
    }

    @Override
    public String toString() {
        return "My exception " + message;
    }
```

FPT POLYTECHNIC

3. Sử dụng từ khóa throws và throw

```
    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);
        }
    }
}
```

FPT Fpt University
FPT POLYTECHNIC

Tổng kết bài học

- Ngoại lệ là các lỗi chỉ xảy ra khi chạy chương trình
- Khi gặp ngoại lệ thì chương trình lập tức dừng lại
- Dùng try… catch để xử lý ngoại lệ theo ý đồ của người lập trình.
- Dùng try có nhiều catch
- Dùng try lồng nhau
- Sử dụng try-catch-finally
- Sử dụng từ khóa throws
- Sử dụng từ khóa throw