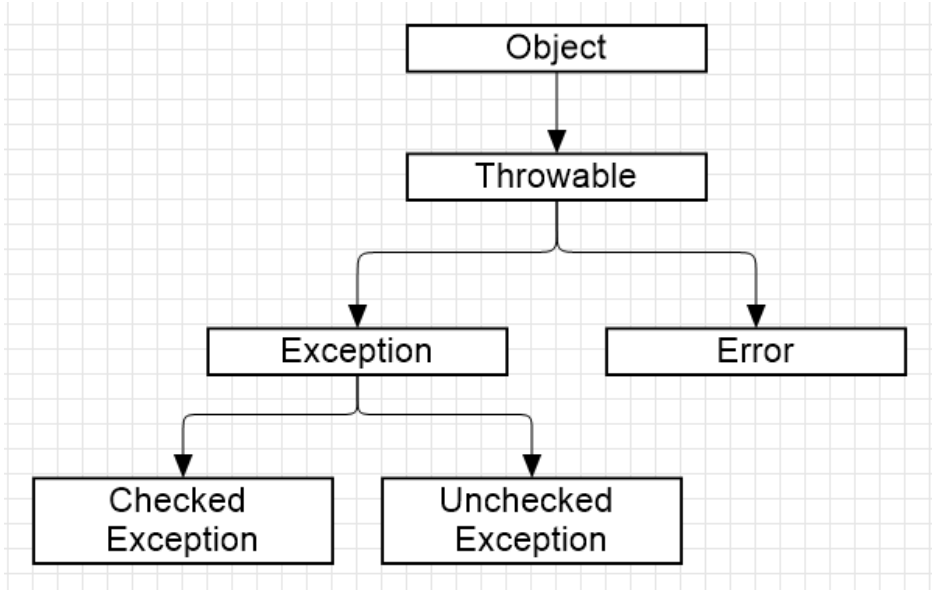
Bài 59: Checked vs unchecked exception

- ✓ Checked exception
- ✓ Unchecked exception
- ✓ Tìm hiểu lớp Exception
- ✓ Ví dụ minh họa

Phân loại

- Sơ đồ phân cấp ngoại lệ trong Java:



- Ngoại lệ trong Java chia làm hai loại: checked và unchecked exception

Checked exception

- Checked exception là một ngoại lệ được kiểm tra và cảnh báo bởi trình biên dịch chương trình tại thời điểm biên dịch code
- Còn có tên là compile time exception
- Ngoại lệ loại này không thể bị bỏ qua và người lập trình bắt buộc phải xử lý nó
- Để xử lý ngoại lệ loại này ta có hai cách: xử lý trực tiếp với try-catch hoặc hoãn xử lý với throws
- Một số ngoại lệ loại này: FileNotFoundException, ParseException, ClassNotFoundException, InvalidClassException…

Ví dụ 1

➢Ví dụ xử lý ngoại lệ ParseException:

```
SimpleDateFormat dateFormat =
        new SimpleDateFormat("dd/MM/yyyy");
var dateString = "27-05-2025";
// chuyển đổi string sang đối tượng date:
try {
    // dòng code dưới đây có thể xảy ra ngoại lệ:
    Date myBirthday = dateFormat.parse(dateString);
    System.out.println("Đối tượng date hoàn chỉnh: "
            + myBirthday);
} catch (ParseException e) {
    System.out.println("Không thể chuyển đổi String sang Date");
    e.printStackTrace();
}
// các câu lệnh kế tiếp phía sau
System.out.println("Thực hiện các câu lệnh tiếp theo...");
```

Ví dụ 2

➢Ví dụ xử lý ngoại lệ FileNotFoundException:

```
File input = new File("input.txt");
try {
    Scanner readFile = new Scanner(input);
    System.out.println("Mở file thành công");
    // thực hiện đọc dữ liệu trong file
} catch (FileNotFoundException e) {
    e.printStackTrace();
}
// các lệnh phía sau nơi xảy ra ngoại lệ
System.out.println("Thực hiện các lệnh kế tiếp...");
```

Unchecked exception

- Là ngoại lệ xảy ra tại thời điểm chương trình đang chạy
- Còn có tên là runtime exception
- Thường được bỏ qua tại thời điểm biên dịch chương trình
- Nguyên nhân chủ yếu dẫn đến ngoại lệ này là do lỗi logic của lập trình viên
- Đây là các ngoại lệ có thể tránh được bằng cách kiểm soát tốt logic chương trình
- Chúng ta cũng có thể xử lý các ngoại lệ này với try-catch

Ví dụ 1

- Một số ngoại lệ loại này: NullPointerException, ArrayIndexOutOfBoundsException, ArithmeticException, ClassCastException…
- Ví dụ: ngăn ngừa ngoại lệ NullPointerException bằng if-else.

```
int[] numbers = null;
int index = 100;
if (numbers != null) {
    // dòng code có ngoại lệ
    System.out.println("Số phần tử của mảng numbers là: " +
            numbers.length);
} else {
    System.out.println("Đối tượng numbers là null");
}

// các lệnh phía sau nơi xảy ra ngoại lệ
System.out.println("Thực hiện các lệnh kế tiếp...");
```

Ví dụ 2

- Ví dụ sau xử lý ngoại lệ NullPointerException bằng try-catch:

```
int[] numbers = null;
int index = 100;

try {
    // dòng code có ngoại lệ
    System.out.println("Số phần tử của mảng numbers là: " +
        numbers.length);
} catch (NullPointerException e) {
    System.out.println(e.getMessage());
}

// các lệnh phía sau nơi xảy ra ngoại lệ
System.out.println("Thực hiện các lệnh kế tiếp...");
```

Ví dụ 3

- Ngăn ngừa ArrayIndexOutOfBoundsException bằng if-else:

```
int[] numbers = {1, 2, 3, 6, 5, 4, 7, 8, 9};
int index = 100;

if (index >= 0 && index < numbers.length) {
    System.out.println("Phần tử thứ " + index +
            " của mảng numbers là: " + numbers[index]);
} else {
    System.out.println("Chỉ số mảng " + index + " không hợp lệ");
    System.out.println("Chỉ số phần tử tối đa của mảng: " +
            (numbers.length - 1));
}

// các lệnh phía sau nơi xảy ra ngoại lệ
System.out.println("Thực hiện các lệnh kế tiếp...");
```

Lớp Exception

- Một số phương thức của lớp Exception và mô tả:

Phương thức	Mô tả
String getMessage()	Trả về thông điệp chi tiết về ngoại lệ đã xảy ra. Thông điệp này được khởi tạo trong constructor của Throwable.
Throwable getCause()	Trả về nguyên nhân gây ra ngoại lệ trong đối tượng của Throwable.
void printStackTray()	In ra màn hình loại, thông điệp của ngoại lệ xảy ra và thứ tự các lời gọi lưu trong stack đến khi gặp ngoại lệ.
StackTraceElement [] getStackTrace()	Trả về mảng chứa các lời gọi trong khay stack. Phần tử tại vị trí 0 là phần tử trên top của stack. Phần tử cuối là lời gọi đầu tiên trong khay stack dẫn đến ngoại lệ.

Minh họa

- Thực hiện trong công cụ lập trình

Nội dung tiếp theo

Sử dụng try-catch-finally