

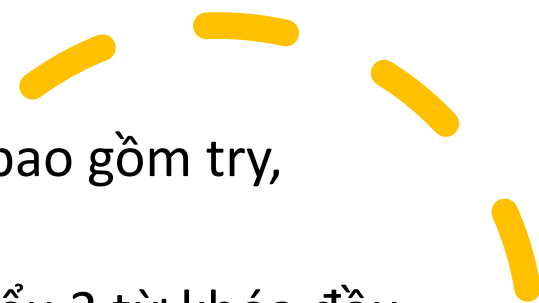
# Bài 60: Sử dụng try-catch-finally

---

- ✓ Mục đích sử dụng
- ✓ Sử dụng try-catch
- ✓ Sử dụng try-catch-finally
- ✓ Ví dụ minh họa



# Mục đích sử dụng

- 
- Các từ khóa trong xử lý ngoại lệ bao gồm try, catch, finally, throw, throws
  - Nội dung bài học này ta sẽ tìm hiểu 3 từ khóa đầu
  - Nội dung bài kế tiếp ta sẽ tìm hiểu 2 từ khóa cuối

# Mục đích sử dụng

Ý nghĩa cụ thể của từng từ khóa:

- Đoạn code có thể xảy ra ngoại lệ đặt trong khối try. Mỗi try có thể đi kèm với một vài khối catch/finally tương ứng
- Khi ngoại lệ xảy ra trong khối try thì các khối catch đi sau try sẽ được kiểm tra. Nếu kiểu của ngoại lệ trong catch nào trùng với kiểu ngoại lệ xảy ra trong try thì khối catch đó sẽ được gọi để xử lý ngoại lệ
- Khối finally luôn nằm ở cuối, sau catch. Nội dung của khối này luôn thực hiện cho dù có xảy ra ngoại lệ hay không. Ta thường dùng khối này để dọn dẹp, đóng kết nối, thực hiện các hành động luôn xảy ra trong chương trình

# Sử dụng try-catch

## ➤ Cú pháp tổng quát:

```
try {  
    // đoạn code có thể xảy ra ngoại lệ  
} catch (ExceptionType e) {  
    // đoạn code xử lý ngoại lệ  
}
```

## ➤ Trong đó:

- Thân của khối try sẽ chứa các đoạn code có thể xảy ra ngoại lệ
- ExceptionType là kiểu của ngoại lệ có thể xảy ra trong khối try mà khối catch sẽ xử lý
- Trong thân khối catch là đoạn chương trình xử lý ngoại lệ

# Ví dụ

## ➤ Ví dụ sau xử lý ngoại lệ ParseException:

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

# Đa khối catch

➤ Cú pháp tổng quát:

```
try {  
    // đoạn code có thể xảy ra ngoại lệ  
} catch (ExceptionType1 e1) {  
    // đoạn code xử lý ngoại lệ kiểu ExceptionType1  
} catch (ExceptionType2 e2) {  
    // đoạn code xử lý ngoại lệ kiểu ExceptionType2  
} catch (ExceptionType3 e3) {  
    // đoạn code xử lý ngoại lệ kiểu ExceptionType3  
} catch (...) {  
    // đoạn code xử lý ngoại lệ kiểu ...  
}
```

➤ Lưu ý: không để kiểu ngoại lệ cha trong catch phía trên ngoại lệ con.

## Gộp nhóm các catch

- Ta có thể gom nhiều kiểu ngoại lệ lại trong 1 catch như sau:

```
try {  
    // đoạn code có thể xảy ra ngoại lệ  
} catch (ExceptionType1 | ExceptionType2 | ExceptionType3 ex) {  
    // đoạn code xử lý ngoại lệ chung  
}
```

- Trong đó các ngoại lệ phân tách nhau bởi dấu | và chỉ dùng chung một biến của các kiểu ngoại lệ(ex) để sau ngoại lệ cuối
- Khi có một trong các ngoại lệ xảy ra tương ứng trong nhóm trên thì khối catch sẽ hoạt động

# Ví dụ

➤ Ví dụ sau kết hợp nhiều kiểu ngoại lệ trong 1 catch:

```
SimpleDateFormat dateFormat =  
    new SimpleDateFormat("dd/MM/yyyy");  
var dateString = "";  
try {  
    // dòng code dưới đây có thể xảy ra ngoại lệ:  
    Scanner fileReader = new Scanner(new File("input.txt"));  
    dateString = fileReader.nextLine(); // đọc dữ liệu từ file  
    Date myBirthday = dateFormat.parse(dateString);  
    System.out.println("Đối tượng date hoàn chỉnh: " + myBirthday);  
} catch (ParseException | FileNotFoundException e) {  
    System.out.println("Đã xảy ra ngoại lệ: " + e.getMessage());  
    System.out.println("Vui lòng kiểm tra lại!");  
}  
// các câu lệnh kế tiếp phía sau  
System.out.println("Thực hiện các câu lệnh tiếp theo...");
```



# try-catch- finally

## ➤ Tổng quát:

```
try {  
    // đoạn code có thể xảy ra ngoại lệ  
} catch (ExceptionType e) {  
    // đoạn code xử lý ngoại lệ  
} finally {  
    // đoạn code luôn được thực thi, vd: đóng kết nối, dọn dẹp  
}
```

## ➤ Trong đó: đoạn code trong khối finally luôn chạy dù có xảy ra ngoại lệ hay không

# Lưu ý

Lưu ý:

- Khối catch không thể tồn tại nếu không có try
- Khối try không thể tồn tại độc lập mà phải đi liền bởi catch hoặc finally
- Khối finally không có ý nghĩa nếu không có try-catch phía trước nó
- Không thể tồn tại bất kì dòng code nào xen giữa các khối try-catch-finally

# Ví dụ

## Ví dụ sử dụng try-catch-finally:

```
SimpleDateFormat dateFormat =
    new SimpleDateFormat("dd/MM/yyyy");
Date myBirthday = null;
var dateString = "";
try {
    // dòng code dưới đây có thể xảy ra ngoại lệ:
    Scanner fileReader = new Scanner(new File("input.txt"));
    dateString = fileReader.nextLine(); // đọc dữ liệu từ file
    myBirthday = dateFormat.parse(dateString);
} catch (ParseException | FileNotFoundException e) {
    System.out.println("Đã xảy ra ngoại lệ: " + e.getMessage());
    System.out.println("Vui lòng kiểm tra lại!");
} finally { // đoạn chương trình sau luôn chạy
    if (myBirthday == null) {
        myBirthday = new Date();
    }
    System.out.println("==> Sinh nhật của tôi là: " +
        dateFormat.format(myBirthday) + " <==");
}
// các câu lệnh kế tiếp phía sau
System.out.println("Thực hiện các câu lệnh tiếp theo...");
```



# Minh họa

➤ Thực hiện trong công cụ lập trình

# Nội dung tiếp theo

**Sử dụng throw và throws**