How to use try .. catch... finally

Tags	Progress

Trong Java,
`try`, `catch`, và `finally` là các khối lệnh quan trọng trong xử lý ngoại lệ. Chúng giúp chương trình duy trì luồng xử lý bình thường khi gặp lỗi runtime. Cụ thể:

1. **Try**: Dùng để chứa code có thể phát sinh ngoại lệ. Nếu có lỗi xảy ra, chương trình sẽ chuyển sang khối `catch` tương ứng.
2. **Catch**: Dùng để xử lý ngoại lệ cụ thể sau khối `try`. Có thể có nhiều khối `catch` cho một khối `try`.
3. **Finally**: Luôn được thực thi dù có ngoại lệ hay không. Hữu ích trong việc dọn dẹp tài nguyên.

Cú Pháp:

- Try Catch:

```
try {
    // code có thể ném ra ngoại lệ
} catch(Exception_class_Name ex) {
    // code xử lý ngoại lệ
}
```

- Try Finally:

```
try {
    // code có thể ném ra ngoại lệ
} finally {
    // code luôn được thực thi
```

```
}
```

- Try Catch Finally:

```
try {
    // code có thể ném ra ngoại lệ
} catch(Exception_class_Name_1 ex) {
    // xử lý ngoại lệ 1
} // có thể có nhiều khối catch
finally {
    // code luôn được thực thi
}
```

2 Kiểu Ngoại Lệ Trong Java:

1. *Checked Exceptions*: Bắt buộc phải xử lý hoặc khai báo. Nếu không, chương trình sẽ không biên dịch.
2. *Unchecked Exceptions*: Thường do lỗi code, null reference, hoặc đối số không hợp lệ.

Các Ngoại Lệ Phổ Biến:

- Checked: `IOException`, `SQLException`, `ClassNotFoundException`, `InstantiationException`.
- Unchecked: `NullPointerException`, `ArrayIndexOutOfBound`, `IllegalArgumentException`, `IllegalStateException`, `NumberFormatException`, `ArithmeticException`.

Ví Dụ Về Cách Sử Dụng Try Catch Trong Java

1. **Chia Cho 0 (Chỉ Một Khối Catch):**

```
public class TryCatchDemo {
    public static void main(String[] args) {
        try {
```

```
            int data = 5 / 0;
        } catch (ArithmeticException e) {
            System.out.println(e);
        }
        System.out.println("Phép chia cho 0");
    }
}
```

Kết quả: In ra ngoại lệ và tiếp tục chương trình.

2. **Nhiều Khối Catch:**

```
public class TryCatchDemo {
    public static void main(String[] args) {
        try {
            // code gây ra nhiều loại ngoại lệ khác nhau
        } catch (NullPointerException ex) {
            System.out.println(ex);
        } catch (ArithmeticException ex) {
            // và các khối catch khác
        }
        System.out.println("Finished!");
    }
}
```

Kết quả: Bắt từng loại ngoại lệ cụ thể.

3. **Sử Dụng Try Finally (Không Xử Lý Ngoại Lệ):**

```
public class TryCatchDemo {
    public static void main(String[] args) {
        try {
            int data = 5 / 0;
```

```
        } finally {
            System.out.println("Khối lệnh finally luôn được
        }
        System.out.println("Finished!");
    }
}
```

Kết quả: `finally` thực thi, nhưng ngoại lệ không được xử lý do không có `catch`.

4. **Try Finally Với Ngoại Lệ Được Xử Lý:**

```
public class TryCatchDemo {
    public static void main(String[] args) {
        try {
            int data = 5 / 0;
        } catch (ArithmeticException ex) {
            System.out.println(ex);
        } finally {
            System.out.println("Khối lệnh finally luôn được
        }
        System.out.println("Finished!");
    }
}
```

Kết quả: Cả `catch` và `finally` đều thực thi.