**Xử Lý Ngoại Lệ (Exception Handling) trong Java**

Ngoại lệ là các tình huống lỗi xảy ra trong quá trình thực thi chương trình. Java cung cấp cơ chế để xử lý các ngoại lệ thông qua các khối try-catch-finally.

**1. Sử Dụng try-catch-finally**

Cấu trúc try-catch-finally cho phép chương trình bắt ngoại lệ và xử lý chúng một cách an toàn mà không làm chương trình dừng đột ngột.

**Cú pháp:**

try {

// Khối lệnh có thể gây ra ngoại lệ

} catch (ExceptionType e) {

// Khối lệnh xử lý ngoại lệ

} finally {

// Khối lệnh luôn được thực thi, dù có xảy ra ngoại lệ hay không

}

**Ví dụ:**

try {

int result = 10 / 0; // Gây ra ngoại lệ ArithmeticException

} catch (ArithmeticException e) {

System.out.println("Không thể chia cho 0.");

} finally {

System.out.println("Khối finally luôn được thực thi.");

}

* **try**: Chứa các khối lệnh có thể gây ra ngoại lệ.
* **catch**: Bắt ngoại lệ và xử lý nó.
* **finally**: Luôn thực thi sau khối try và catch, dùng để đóng tài nguyên hoặc thực hiện các tác vụ dọn dẹp.

**2. Ngoại Lệ Tùy Chỉnh (Custom Exceptions)**

Java cho phép tạo ra các ngoại lệ tùy chỉnh bằng cách kế thừa từ lớp Exception.

**Tạo Ngoại Lệ Tùy Chỉnh:**

class CustomException extends Exception {

public CustomException(String message) {

super(message);

}

}

**Sử dụng Ngoại Lệ Tùy Chỉnh:**

public class Main {

public static void checkNumber(int number) throws CustomException {

if (number < 0) {

throw new CustomException("Số không được nhỏ hơn 0.");

}

}

public static void main(String[] args) {

try {

checkNumber(-5);

} catch (CustomException e) {

System.out.println(e.getMessage());

}

}

}

Trong ví dụ trên, khi checkNumber() nhận số âm, nó sẽ ném ngoại lệ tùy chỉnh CustomException với thông báo tương ứng.

**Luồng Vào/Ra (I/O) trong Java**

Java cung cấp các lớp và giao diện để thực hiện các thao tác vào/ra (I/O), đặc biệt là đọc và ghi tệp.

**1. Làm Việc Với Tệp: FileReader và FileWriter**

Các lớp FileReader và FileWriter trong Java dùng để đọc và ghi các tệp văn bản.

**1.1. FileReader - Đọc Tệp**

Lớp FileReader được sử dụng để đọc dữ liệu từ tệp văn bản.

**Cú pháp:**

FileReader reader = new FileReader("file.txt");

**Ví dụ:**

import java.io.FileReader;

import java.io.IOException;

public class Main {

public static void main(String[] args) {

try {

FileReader reader = new FileReader("example.txt");

int character;

while ((character = reader.read()) != -1) {

System.out.print((char) character); // In ra nội dung tệp

}

reader.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

**1.2. FileWriter - Ghi Tệp**

Lớp FileWriter được sử dụng để ghi dữ liệu vào tệp văn bản.

**Cú pháp:**

FileWriter writer = new FileWriter("file.txt");

**Ví dụ:**

import java.io.FileWriter;

import java.io.IOException;

public class Main {

public static void main(String[] args) {

try {

FileWriter writer = new FileWriter("example.txt");

writer.write("Hello, World!"); // Ghi dữ liệu vào tệp

writer.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

**Lưu ý:**

* **Ghi đè dữ liệu**: Nếu tệp đã tồn tại, FileWriter sẽ ghi đè nội dung cũ.
* **Thêm dữ liệu (Append)**: Để thêm dữ liệu mà không ghi đè, có thể khởi tạo FileWriter với tham số thứ hai là true.

FileWriter writer = new FileWriter("example.txt", true); // Thêm dữ liệu vào tệp

**2. Đọc và Ghi Tệp Văn Bản**

**Đọc Tệp Bằng BufferedReader**

BufferedReader kết hợp với FileReader để đọc dữ liệu từ tệp theo từng dòng, giúp cải thiện hiệu suất.

**Ví dụ:**

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;

public class Main {

public static void main(String[] args) {

try {

BufferedReader reader = new BufferedReader(new FileReader("example.txt"));

String line;

while ((line = reader.readLine()) != null) {

System.out.println(line); // In ra từng dòng của tệp

}

reader.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

**Ghi Tệp Bằng BufferedWriter**

BufferedWriter kết hợp với FileWriter để ghi dữ liệu vào tệp theo từng dòng.

**Ví dụ:**

import java.io.BufferedWriter;

import java.io.FileWriter;

import java.io.IOException;

public class Main {

public static void main(String[] args) {

try {

BufferedWriter writer = new BufferedWriter(new FileWriter("example.txt"));

writer.write("First line");

writer.newLine(); // Ghi dòng mới

writer.write("Second line");

writer.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

**Tóm Tắt:**

* try-catch-finally dùng để xử lý ngoại lệ và bảo vệ chương trình khỏi lỗi runtime.
* Tạo ngoại lệ tùy chỉnh giúp dễ dàng quản lý và thông báo lỗi.
* FileReader và FileWriter là các lớp cơ bản để đọc và ghi tệp văn bản.
* BufferedReader và BufferedWriter giúp tối ưu hóa việc đọc và ghi tệp.

Trên đây là kiến thức về xử lý ngoại lệ và làm việc với tệp trong Java.