Khối try-catch trong Java

Khối try trong Java

Khối try trong Java được sử dụng để bao quanh code mà có thể ném một Exception. Nó phải được sử dụng bên trong phương thức. Khối try phải được theo sau bởi hoặc khối catch hoặc khối finally.

Cú pháp của khối try-catch trong Java

```
try{
//code ma co the nem exception
}catch(Exception_class_Name ref){}
```

Cú pháp của khối try-finally trong Java

```
try{
//code ma co the nem exception
}finally{}
```

Khối catch trong Java

Khối catch trong Java được sử dụng để xử lý các Exception. Nó phải được sử dụng chỉ sau khối try. Bạn có thể sử dụng nhiều khối catch với một khối try đơn.

Vấn đề khi không có Exception Handling

Chúng ta cùng tìm hiểu vấn đề nếu không sử dụng khối try-catch.

```
public class Testtrycatch1{
  public static void main(String args[]){
      int data=50/0;//co the nem exception
      System.out.println("Phan code con lai...");
}
}
```

Chương trình sẽ cho kết quả sau:

```
Exception in thread "main" java.lang.ArithmeticException: / by zero
```

Như trong ví dụ trên, phần còn lại của code không được thực thi (trong ví dụ này là lệnh in*Phan code con lai…* không được in). Giả sử nếu có khoảng 100 dòng code sau exception, thì tất cả các dòng code này sẽ không được thực thi.

Xử lý vấn đề trên bởi Exception Handling trong Java

Bây giờ, cùng ví dụ trên, chúng ta sử dụng khối try-catch để xử lý vấn đề trên.

```
public class Testtrycatch2{
  public static void main(String args[]){
   try{
      int data=50/0;
   }catch(ArithmeticException e){System.out.println(e);}
   System.out.println("Phan code con lai...");
}
}
```

Chương trình sẽ cho kết quả sau:

```
java.lang.ArithmeticException: / by zero
Phan code con lai...
```

Lúc này, phần còn lại của code đã được thực thi.

Chế độ làm việc nội tại của khối try-catch trong Java

Đầu tiên, JVM kiểm tra xem exception đã được xử lý hay chưa. Nếu exception chưa được xử lý, JVM cung cấp một Exception Handler mặc định, để thực hiện các tác vụ sau:

- In ra miêu tả của exception đó.
- In ra stack trace (cấu trúc thứ bậc của phương thức nơi mà exception xuất hiện).
- Làm cho chương trình ngừng lại.

Nhưng nếu exception đã được xử lý bởi Lập trình viên, thì luồng chuẩn của ứng dụng được duy trì (hay là phần còn lại của code được thực thi).

Nhiều khối catch trong Java

Nếu bạn phải thực hiện các tác vụ khác nhau mà có thể xảy ra các exception khác nhau, bạn sử dụng nhiều khối catch trong Java. Bạn theo dõi ví dụ đơn giản sau:

```
public class TestMultiCatchBlock{
  public static void main(String args[]){
   try{
    int a[]=new int[5];
    a[5]=30/0;
   }
   catch(ArithmeticException e){System.out.println("Task1 duoc hoan thanh");}
   catch(ArrayIndexOutOfBoundsException e){System.out.println("Task2 duoc hoan thanh");}
   catch(Exception e){System.out.println("Task chung duoc hoan thanh");}

   System.out.println("Phan code con lai...");
 }
}
```

Chương trình sẽ cho kết quả sau:

```
Task1 duoc hoan thanh
Phan code con lai...
```

Qui tắc 1: Tại một thời điểm, chỉ một exception được xuất hiện và tại một thời điểm chỉ có một khối catch được thực thi.

Qui tắc 2: Tất cả khối catch phải được sắp xếp từ cụ thể nhất tới chung nhất, ví dụ: việc bắt ArithimeticExption phải ở trước việc bắt Exception.

```
class TestMultipleCatchBlock1{
  public static void main(String args[]){
   try{
    int a[]=new int[5];
    a[5]=30/0;
   }
   catch(Exception e){System.out.println("Task chung duoc hoan thanh");}
```

```
   catch(ArithmeticException e){System.out.println("Task1 duoc hoan thanh");}
   catch(ArrayIndexOutOfBoundsException e){System.out.println("Task2 duoc hoan thanh");}
   System.out.println("Phan code con lai...");
 }
}
```

Chạy chương trình trên sẽ cho một lỗi compile time error.

Lồng khối try trong Java

Việc một khối try bên trong một khối try khác thì được gọi là các khối try được lồng vào nhau, hay là lồng các khối try trong Java.

Tại sao sử dụng các khối try lồng nhau trong Java

Đôi khi có một tình huống là một phần của một khối code có thể gây ra một lỗi và toàn bộ khối lại có thể gây ra lỗi khác. Trong các tình huống đó, Exception Handler phải được lồng vào nhau. Cú pháp:

```
....
try
{
    lenh 1;
    lenh 2;
    try
    {
        lenh 1;
        lenh 2;
    }
    catch(Exception e)
    {
    }
}
catch(Exception e)
{
}
```

```
....
```

Ví dụ về các khối try lồng nhau trong Java

```
class Excep6{
 public static void main(String args[]){
  try{
    try{
     System.out.println("Thuc hien phep chia");
     int b =39/0;
    }catch(ArithmeticException e){System.out.println(e);}

    try{
    int a[]=new int[5];
    a[5]=4;
    }catch(ArrayIndexOutOfBoundsException e){System.out.println(e);}

    System.out.println("Lenh khac");
  }catch(Exception e){System.out.println("Da xu ly");}

  System.out.println("Luong chuan..");
 }
}
```

Chương trình sẽ cho kết quả sau:

```
Thuc hien phep chia
java.lang.ArithmeticException: / by zero
java.lang.ArrayIndexOutOfBoundsException: 5
Lenh khac
Luong chuan..
```