**Flow control in try-catch-finally blocks**

[**EXCEPTION HANDLING**](http://beginnersbook.com/category/technology/java-guide/exception-handling/)

In this tutorial we will see how to use try-catch-finally blocks for [**exception handling**](http://beginnersbook.com/2013/04/java-exception-handling/). Lets see  the [**try-catch**](http://beginnersbook.com/2013/04/try-catch-in-java/) alone first and then we will discuss **try-catch-finally**.

**Flow of control in try/catch blocks:**

**when exception doesn’t occur:**  
When the code which is present in try block’s body doesn’t [**throw any exception**](http://beginnersbook.com/2013/04/throw-in-java/) then first, the body of try block executes and then the code after catch blocks. In this case catch block never runs as they are meant to be run when an exception occurs. For example-

.....

int x = 10;

int y = 10;

try{

int num= x/y;

System.out.println("next-statement: Inside try block");

}catch(Exception ex)

{

System.out.println("Exception");

}

System.out.println("next-statement: Outside of try-catch");

...

**Output:**

next-statement: Inside try block

next-statement: Outside of try-catch

In the above example exception didn’t occur in try block so catch block didn’t run.

**when exception occurs:**  
First have a look at the below example and then we will discuss it –

int x = 0;

int y = 10;

try{

int num= y/x;

System.out.println("next-statement: Inside try block");

}catch(Exception ex)

{

System.out.println("Exception Occurred");

}

System.out.println("next-statement: Outside of try-catch");

...

**Output:**

Exception Occurred

next-statement: Outside of try-catch

**Point to note in above example:** There are two statements present inside try block. Since exception occurred because of first statement, the second statement didn’t execute. Hence we can conclude that if an exception occurs then the rest of the try block doesn’t execute and control passes to catch block.

**Flow of control in try/catch/finally blocks:**

1. If exception occurs in try block’s body then control immediately transferred(**skipping rest of the statements in try block**) to the catch block. Once catch block finished execution then [**finally block**](http://beginnersbook.com/2013/04/java-finally-block/) and after that rest of the program.
2. If there is no exception occurred in the code which is present in try block then first, the try block gets executed completely and then control gets transferred to finally block (**skipping catch blocks**).
3. If a [**return statement**](http://beginnersbook.com/2013/05/java-finally-return/) is encountered either in try or catch block. In such case also **finally runs**. Control first goes to finally and then it returned back to **return statement.**

**Consider the below example to understand above mentioned points:**

class TestExceptions {

static void myMethod(int testnum) throws Exception {

System.out.println ("start - myMethod");

if (testnum == 12)

throw new Exception();

System.out.println("end - myMethod");

return;

}

public static void main(String args[]) {

int testnum = 12;

try {

System.out.println("try - first statement");

myMethod(testnum);

System.out.println("try - last statement");

}

catch ( Exception ex) {

System.out.println("An Exception");

}

finally {

System. out. println( "finally") ;

}

System.out.println("Out of try/catch/finally - statement");

}

}

**Output:**

try - first statement

start - myMethod

An Exception

finally

Out of try/catch/finally - statement