**How to throw exception in java with example**

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

In java we have already defined exception classes such asArithmeticException, ArrayIndexOutOfBoundsException,NullPointerException etc. There are certain conditions defined for these exceptions and on the occurrence of those conditions they are implicitly thrown by JVM(java virtual machine).

**Do you know that a programmer can create a new exception and throw it explicitly**? These exceptions are known as [**user-defined exceptions**](http://beginnersbook.com/2013/04/user-defined-exception-in-java/). In order to throw user defined exceptions, [**throw keyword**](http://beginnersbook.com/2013/12/throw-keyword-example-in-java/) is being used. In this tutorial, we will see how to create a new exception and throw it in a program using **throw keyword.**

You can also throw an already defined exception like ArithmeticException,IOException etc.

**Syntax of throw statement**

throw AnyThrowableInstance;

**Example:**

//A void method

public void sample()

{

  //Statements

  //if (somethingWrong) then

  IOException e = new IOException();

  throw e;

  //More Statements

}

**Note:**

* A call to the above mentioned sample method should be always placed in a try block as it is throwing a [**checked exception**](http://beginnersbook.com/2013/04/java-checked-unchecked-exceptions-with-examples/) – IOException. This is how it the call to above method should be done:
* MyClass obj = new MyClass();
* try{
* obj.sample();
* }catch(IOException ioe)
* {
* //Your error Message here
* System.out.println(ioe);

}

* Exceptions in java are compulsorily of type Throwable. If you attempt to throw an object that is not throwable, the  compiler refuses to compile your program and it would show a compilation error.

**Flow of execution while throwing an exception using throw keyword**

Whenever a throw statement is encountered in a program the next statement doesn’t execute. Control immediately transferred to catch block to see if the thrown exception is handled there. If the exception is not handled there then next catch block is being checked for exception and so on. If none of the[**catch block**](http://beginnersbook.com/2013/04/try-catch-in-java/) is handling the thrown exception then a system generated exception message is being populated on screen, same what we get for un-handled exceptions.  
E.g.

class ThrowDemo{

public static void main(String args[]){

try{

char array[] = {'a','b','g','j'};

/\*I'm displaying the value which does not

\* exist so this should throw an exception

\*/

System.out.println(array[78]);

}catch(ArithmeticException e){

System.out.println("Arithmetic Exception!!");

}

}

}

Output:

Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException:

78 at beginnersbook.com.ThrowDemo.main(Details.java:9)

Since the exception thrown was not handled in the catch blocks the system generated exception message got displayed for that particular exception.

**Few examples of throw exception in Java**

**Example 1: How to throw your own exception explicitly using throw keyword**

package beginnersbook.com;

class MyOwnException extends Exception {

public MyOwnException(String msg){

super(msg);

}

}

class EmployeeTest {

static void employeeAge(int age) throws MyOwnException{

if(age < 0)

throw new MyOwnException("Age can't be less than zero");

else

System.out.println("Input is valid!!");

}

public static void main(String[] args) {

try {

employeeAge(-2);

}

catch (MyOwnException e) {

e.printStackTrace();

}

}

}

**Output:**

beginnersbook.com.MyOwnException: Age can't be less than zero

**Points to Note:**Method call should be in try block as it is throwing an exception.

**Example2: How to throw an already defined exception using throw keyword**

package beginnersbook.com;

class Exception2{

static int sum(int num1, int num2){

if (num1 == 0)

throw new ArithmeticException("First parameter is not valid");

else

System.out.println("Both parameters are correct!!");

return num1+num2;

}

public static void main(String args[]){

int res=sum(0,12);

System.out.println(res);

System.out.println("Continue Next statements");

}

}

**Output:**

Exception in thread main java.lang.ArithmeticException: First parameter is not valid

Similarly other exceptions, such as NullPointerException,ArrayIndexOutOfBoundsException etc. can be thrown.