## **Java throw**

The **throw** keyword in Java is used to explicitly throw an exception from a method or any block of code. We can throw either checked or unchecked exceptions. The **throw** keyword is mainly used to throw custom exceptions.

**Example:**

***throw new ArithmeticException(“/ by zero”);***

But this exception i.e., Instance must be of type **Throwable** or a subclass of **Throwable**.

The flow of execution of the program stops immediately after the throw statement is executed and the nearest enclosing **try** block is checked to see if it has a **catch** statement that matches the type of exception. If it finds a match, control is transferred to that statement. If no matching **catch** is found then the default exception handler will halt the program.

## **Java throws**

**throws** is a keyword in Java that is used in the signature of a method to indicate that this method might throw one of the listed type exceptions. The caller to these methods has to handle the exception using a try-catch block.

### **Syntax of Java throws**

*type method\_name(parameters) throws exception\_list*

where, **exception\_list** is a comma separated list of all the exceptions which a method might throw.

In a program, if there is a chance of raising an exception then the compiler always warns us about it and compulsorily we should handle that checked exception, Otherwise, we will get a compile time error saying **unreported exception XXX must be caught or declared to be thrown**.

To prevent this compile time error we can handle the exception in two ways:

1. By using **try catch**
2. By using the **throws** keyword

We can use the throws keyword to **delegate the responsibility** of exception handling to the caller (It may be a method or JVM) then the caller method is responsible to handle that exception.

## **Difference Between throw and throws**

The main **differences between throw and throws in Java** are as follows:

| **Feature** | **throw** | **throws** |
| --- | --- | --- |
| **Definition** | It is used to explicitly throw an exception. | It is used to declare that a method might throw one or more exceptions. |
| **Location** | It is used inside a method or a block of code. | It is used in the method signature. |
| **Usage** | It can throw both checked and unchecked exceptions. | It is only used for checked exceptions. Unchecked exceptions do not require **throws** |
| **Responsibility** | The method or block throws the exception. | The method’s caller is responsible for handling the exception. |
| **Flow of Execution** | Stops the current flow of execution immediately. | It forces the caller to handle the declared exceptions. |
| **Example** | throw new ArithmeticException(“Error”); | public void myMethod() throws IOException {} |