# Checked vs Unchecked Exceptions in Java

**Checked exceptions are checked at compile time. If some code in the method throws checked exception, then method must throw the exception using throws keyword or handle it.**

import java.io.\*;

class Main {

    public static void main(String[] args) {

        FileReader file = new FileReader("C:\\test\\a.txt");

        BufferedReader fileInput = new BufferedReader(file);

        // Print first 3 lines of file "C:\test\a.txt"

        for (int counter = 0; counter < 3; counter++)

            System.out.println(fileInput.readLine());

        fileInput.close();

    }

}

In the code above, FileReader while opening file may throw **FileNotFoundException,** FileReader uses readLine() and close() methods. These methods throw **IOException**.

In above code, exception is not thrown or handled. So complier has problem here.

Output:

Exception in thread "main" java.lang.RuntimeException: Uncompilable source code -

unreported exception java.io.FileNotFoundException; must be caught or declared to be

thrown

at Main.main(Main.java:5)

To fix this problem, either we need to throw list of exceptions or we need to handle using try- catch block. Since FileNotFoundException is sub class of IOException. So we can throw IOException only using throws keyword.

import java.io.\*;

class Main {

public static void main(String[] args) throws IOException {

FileReader file = new FileReader("C:\\test\\a.txt");

BufferedReader fileInput = new BufferedReader(file);

// Print first 3 lines of file "C:\test\a.txt"

for (int counter = 0; counter < 3; counter++)

System.out.println(fileInput.readLine());

fileInput.close();

}

}

**Unchecked exceptions** are not checked at compile time, so compiler will not force to throw or handle exceptions. It is up to programmer to specify or catch the exception.

In java all exceptions under **ERROR** and **RunTimeException** are unchecked exceptions, every thing under throwable class are checked exceptions.

Consider the following Java program. It compiles fine, but it throws *ArithmeticException* when program is executed. The compiler allows it to compile, because *ArithmeticException* is an unchecked exception.

class Main {

   public static void main(String args[]) {

      int x = 0;

      int y = 10;

      int z = y/x;

  }

}

Output:

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

at Main.main(Main.java:5)

Java Result: 1

**Should we make our exceptions checked or unchecked?**

**Java Doc recommendation:**

***If a client may recover from exception then make it checked exception, if the client cannot do anything to recover from the exception then make it unchecked exception.***