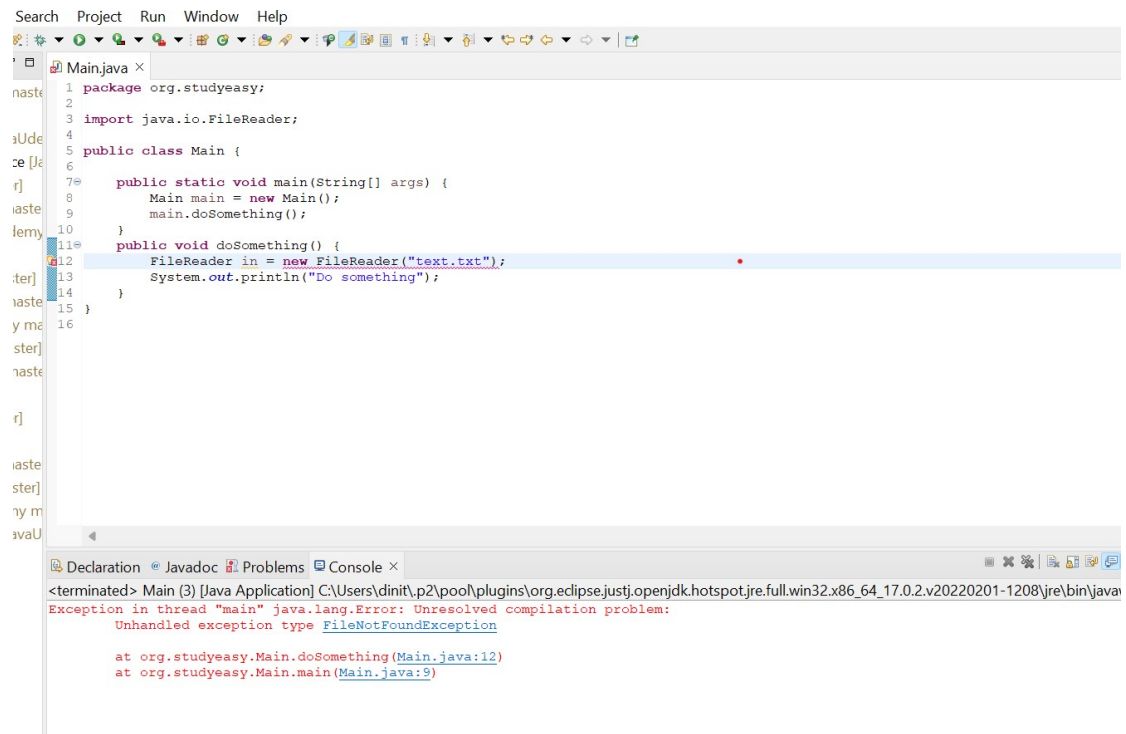


● THROWS

➤ Throws Example , And when you hover over the error you get options like below.

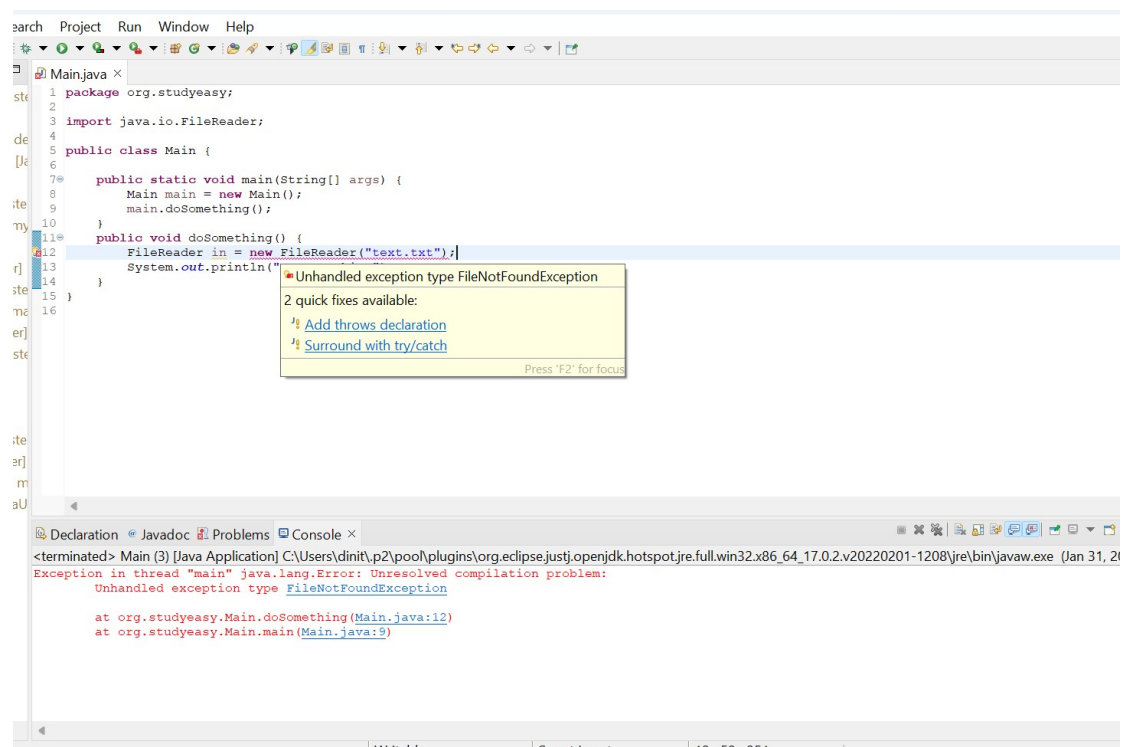


```
1 package org.studyeasy;
2
3 import java.io.FileReader;
4
5 public class Main {
6
7     public static void main(String[] args) {
8         Main main = new Main();
9         main.doSomething();
10    }
11    public void doSomething() {
12        FileReader in = new FileReader("text.txt");
13        System.out.println("Do something");
14    }
15 }
16
```

Declaration Javadoc Problems Console

<terminated> Main (3) [Java Application] C:\Users\dinit\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bin\java.exe
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
 Unhandled exception type [FileNotFoundException](#)

 at org.studyeasy.Main.doSomething(Main.java:12)
 at org.studyeasy.Main.main(Main.java:9)



```
1 package org.studyeasy;
2
3 import java.io.FileReader;
4
5 public class Main {
6
7     public static void main(String[] args) {
8         Main main = new Main();
9         main.doSomething();
10    }
11    public void doSomething() {
12        FileReader in = new FileReader("text.txt");
13        System.out.println("Do something");
14    }
15 }
16
```

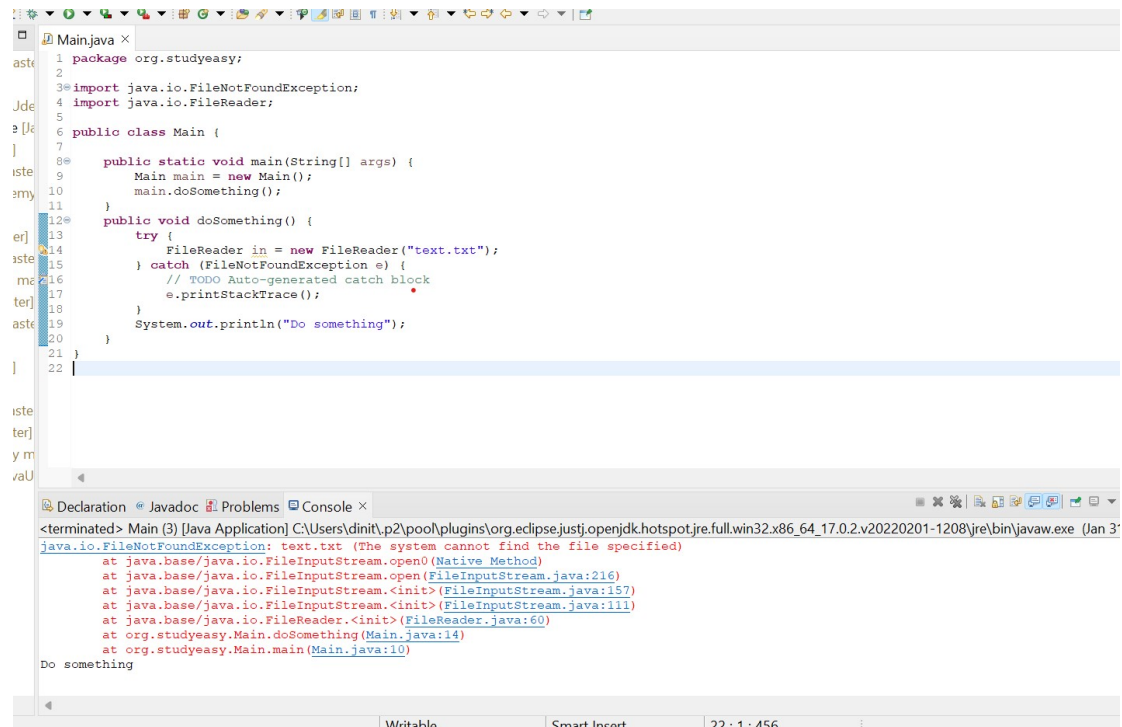
Declaration Javadoc Problems Console

<terminated> Main (3) [Java Application] C:\Users\dinit\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bin\javaw.exe (Jan 31, 2022)
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
 Unhandled exception type [FileNotFoundException](#)

 at org.studyeasy.Main.doSomething(Main.java:12)
 at org.studyeasy.Main.main(Main.java:9)

Unhandled exception type [FileNotFoundException](#)
2 quick fixes available:
1. [Add throws declaration](#)
2. [Surround with try/catch](#)
Press 'F2' for focus

Case 1: Surround with try/catch



```
1 package org.studyeasy;
2
3 import java.io.FileNotFoundException;
4 import java.io.FileReader;
5
6 public class Main {
7
8     public static void main(String[] args) {
9         Main main = new Main();
10        main.doSomething();
11    }
12    public void doSomething() {
13        try {
14            FileReader in = new FileReader("text.txt");
15        } catch (FileNotFoundException e) {
16            // TODO Auto-generated catch block
17            e.printStackTrace();
18        }
19        System.out.println("Do something");
20    }
21 }
22
```

Declaration Javadoc Problems Console

<terminated> Main (3) [Java Application] C:\Users\dinit\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64.17.0.2.v20220201-1208\jre\bin\javaw.exe (Jan 3'

java.io.FileNotFoundException: text.txt (The system cannot find the file specified)

at java.base/java.io.FileInputStream.open0(Native Method)

at java.base/java.io.FileInputStream.open(FileInputStream.java:216)

at java.base/java.io.FileInputStream.<init>(FileInputStream.java:157)

at java.base/java.io.FileReader.<init>(FileReader.java:111)

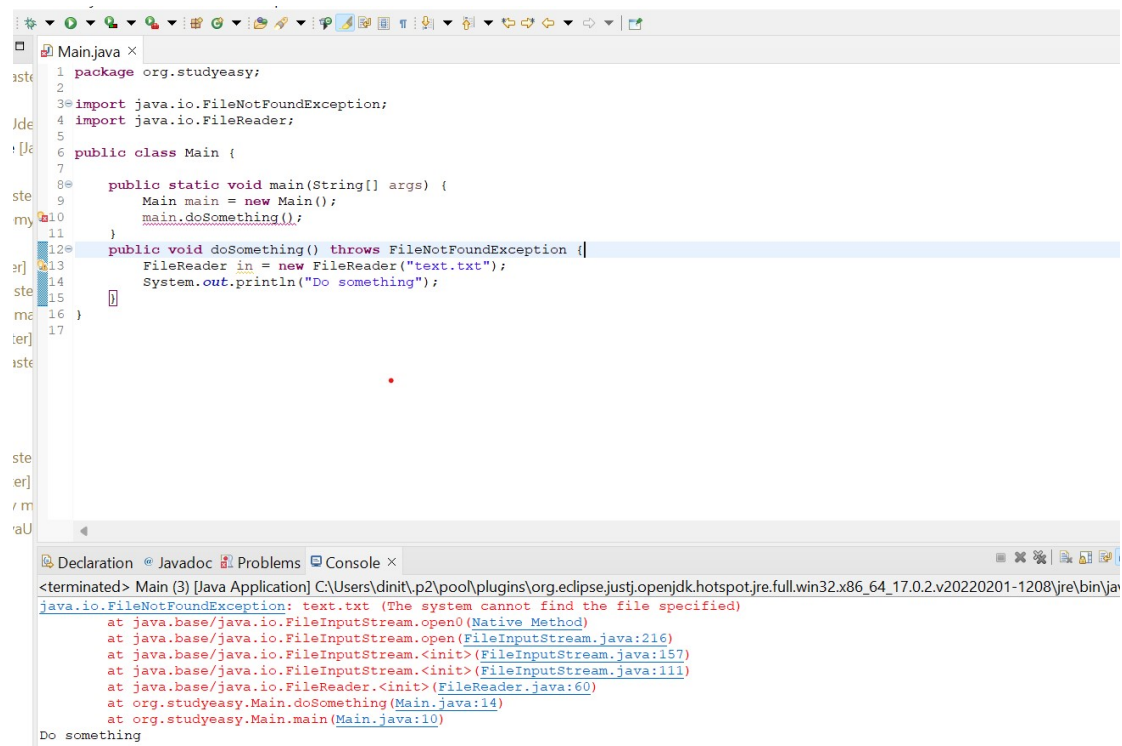
at java.base/java.io.FileReader.<init>(FileReader.java:60)

at org.studyeasy.Main.doSomething(Main.java:14)

at org.studyeasy.Main.main(Main.java:10)

Do something

Case 2 : Add throws declaration



```
1 package org.studyeasy;
2
3 import java.io.FileNotFoundException;
4 import java.io.FileReader;
5
6 public class Main {
7
8     public static void main(String[] args) {
9         Main main = new Main();
10        main.doSomething();
11    }
12    public void doSomething() throws FileNotFoundException {
13        FileReader in = new FileReader("text.txt");
14        System.out.println("Do something");
15    }
16 }
17
```

Declaration Javadoc Problems Console

<terminated> Main (3) [Java Application] C:\Users\dinit\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64.17.0.2.v20220201-1208\jre\bin\ja

java.io.FileNotFoundException: text.txt (The system cannot find the file specified)

at java.base/java.io.FileInputStream.open0(Native Method)

at java.base/java.io.FileInputStream.open(FileInputStream.java:216)

at java.base/java.io.FileInputStream.<init>(FileInputStream.java:157)

at java.base/java.io.FileReader.<init>(FileReader.java:111)

at java.base/java.io.FileReader.<init>(FileReader.java:60)

at org.studyeasy.Main.doSomething(Main.java:14)

at org.studyeasy.Main.main(Main.java:10)

Do something

In Java, the throws keyword is used in the method signature to declare that a method may throw certain exceptions. This keyword is part of the method declaration and is used to indicate that the method might generate exceptions of a particular type during its execution.

```
public void exampleMethod() throws SomeException {
    // Method implementation that may throw SomeException
}
```

- Throws keyword is used in order to throw an exception.
- If we try catch statement - in another block like below, then we get output like below.

```
1 package org.studyeasy;
2
3 import java.io.FileNotFoundException;
4 import java.io.FileReader;
5
6 public class Main {
7
8     public static void main(String[] args) {
9         Main main = new Main();
10        try {
11            main.doSomething();
12        } catch (FileNotFoundException e) {
13            // TODO Auto-generated catch block
14            e.printStackTrace();
15        }
16    }
17
18    public void doSomething() throws FileNotFoundException {
19        FileReader in = new FileReader("text.txt");
20        System.out.println("Do something");
21    }
22 }
```

Declaration @ Javadoc Problems Console x

```
<terminated> Main (3) [Java Application] C:\Users\dinit\p2\poo\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bin\javaw.exe (Jan 31
java.io.FileNotFoundException: text.txt (The system cannot find the file specified)
at java.base/java.io.FileInputStream.open0(Native Method)
at java.base/java.io.FileInputStream.open(FileInputStream.java:216)
at java.base/java.io.FileInputStream.<init>(FileInputStream.java:157)
at java.base/java.io.FileReader.<init>(FileReader.java:111)
at org.studyeasy.Main.doSomething(Main.java:18)
at org.studyeasy.Main.main(Main.java:11)
```

You won't get the output "Do something"

- If you don't know how to handle the exception by yourself and the exception handling will be unique for every case. It is better idea to throw the exception, so whenever the user will use the method, the user have to handle the exception according to the situation.

The screenshot shows an IDE with a Java file named `Main.java` and a console window. The code in `Main.java` is as follows:

```
1 package org.studyeasy;
2
3 import java.io.FileNotFoundException;
4 import java.io.FileReader;
5
6 public class Main {
7
8     public static void main(String[] args) {
9         Main main = new Main();
10        try {
11            main.doSomething();
12        } catch (FileNotFoundException e) {
13            // TODO Auto-generated catch block
14            e.printStackTrace();
15        }
16    }
17    public void doSomething() throws FileNotFoundException {
18        FileReader in = new FileReader("text.txt");
19        System.out.println("Do something");
20    }
21 }
22
```

The console window shows the following output:

```
<terminated> Main (3) [Java Application] C:\Users\dinit\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bin\javaw.exe (Ja
java.io.FileNotFoundException: text.txt (The system cannot find the file specified)
    at java.base/java.io.FileInputStream.open0(Native Method)
    at java.base/java.io.FileInputStream.open(FileInputStream.java:216)
    at java.base/java.io.FileInputStream.<init>(FileInputStream.java:157)
    at java.base/java.io.FileInputStream.<init>(FileInputStream.java:111)
    at java.base/java.io.FileReader.<init>(FileReader.java:60)
    at org.studyeasy.Main.doSomething(Main.java:18)
    at org.studyeasy.Main.main(Main.java:11)
```

An arrow points from the `throws FileNotFoundException` line in the code to the `FileNotFoundException` error in the console.

User is forced to handle the exception as the throws keyword forces it.

➤ JAVA THROW KEYWORD

- Imagine a scenario where we want to throw an exception, based on our own custom logic. So we have **throw** keyword.

Willing to throw an exception manually, we will use this keyword - throw.

```
1 package org.studyeasy;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         Main main = new Main();
7         try {
8             main.doSomething();
9         } finally {
10            System.out.println("This demo is very useless");
11        }
12    }
13    public void doSomething() throws RuntimeException {
14        String x = "zero";
15        if(x.equals("zero")) {
16            throw new RuntimeException();
17        }
18        System.out.println("Do something");
19    }
20 }
21
```

Declaration @ Javadoc Problems Console ×

<terminated> Main (4) [Java Application] C:\Users\dinit\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bin\javaw

Exception in thread "main" This demo is very useless
java.lang.RuntimeException
at org.studyeasy.Main.doSomething(Main.java:16)
at org.studyeasy.Main.main(Main.java:8)

➤ In the above example, String x = "one"

```
1 package org.studyeasy;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         Main main = new Main();
7         try {
8             main.doSomething();
9         } finally {
10            System.out.println("This demo is very useless");
11        }
12    }
13    public void doSomething() throws RuntimeException {
14        String x = "one";
15        if(x.equals("zero")) {
16            throw new RuntimeException();
17        }
18        System.out.println("Do something");
19    }
20 }
21
```

Declaration @ Javadoc Problems Console ×

<terminated> Main (4) [Java Application] C:\Users\dinit\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\

Do something
This demo is very useless

➤ User defined Exception

```
1 package org.studyeasy;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         Main main = new Main();
7         try {
8             main.doSomething();
9         } finally {
10            System.out.println("This demo is very useless");
11        }
12    }
13    public void doSomething() throws RuntimeException, ArithmeticException {
14        String x = "one";
15        if(x.equals("zero")) {
16            throw new RuntimeException();
17        } else if(x.equals("one")) {
18            throw new ArithmeticException();
19        }
20        System.out.println("Do something");
21    }
22 }
23 }
```

Declaration Javadoc Problems Console

<terminated> Main (4) [Java Application] C:\Users\dinit\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bin\javaw.exe (Jan :
This demo is very useless
Exception in thread "main" java.lang.ArithmeticException
at org.studyeasy.Main.doSomething(Main.java:18)
at org.studyeasy.Main.main(Main.java:8)

When we use Exception instead of ArithmeticException. Solution is below also.

```
1 package org.studyeasy;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         Main main = new Main();
7         try {
8             main.doSomething();
9         } finally {
10            System.out.println("This demo is very useless");
11        }
12    }
13    public void doSomething() throws RuntimeException, ArithmeticException {
14        String x = "one";
15        if(x.equals("zero")) {
16            throw new RuntimeException();
17        } else if(x.equals("one")) {
18            throw new Exception();
19        }
20        System.out.println("Do something");
21    }
22 }
23 }
```

Unhandled exception type Exception
1 quick fix available:
Add throws declaration
Press 'F2' for focus

Declaration Javadoc Problems Console

<terminated> Main (4) [Java Application] C:\Users\dinit\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bin\javaw.exe
This demo is very useless
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
Unhandled exception type Exception
at org.studyeasy.Main.doSomething(Main.java:18)
at org.studyeasy.Main.main(Main.java:8)


```
1 package org.studyeasy;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         Main main = new Main();
7         try {
8             main.doSomething();
9         } finally {
10             System.out.println("This demo is very useless");
11         }
12     }
13
14     public void doSomething() throws RuntimeException, Exception {
15         String x = "one";
16         if(x.equals("zero")) {
17             throw new RuntimeException();
18         } else if(x.equals("one")) {
19             throw new Exception();
20         }
21         System.out.println("Do something");
22     }
23 }
```

<terminated> Main (4) [Java Application] C:\Users\dinit\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bin\javaw.exe
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
Unhandled exception type Exception
at org.studyeasy.Main.main(Main.java:8)

But Exception does not make any sense. So we can create a custom Exception, like Zero Exception.

```
1 package org.studyeasy;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         Main main = new Main();
7         try {
8             main.doSomething();
9         } catch (ZeroException e) {
10             // TODO Auto-generated catch block
11             e.printStackTrace();
12         }
13     }
14
15     class ZeroException extends Exception {
16     }
17
18     public void doSomething() throws ZeroException {
19         String x = "one";
20         if(x.equals("zero")) {
21             throw new ZeroException();
22         }
23         System.out.println("Do something");
24     }
25 }
```

<terminated> Main (4) [Java Application] C:\Users\dinit\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bin\javaw.exe (Jar Do something

If `String x = "zero";` you get the below output.

```
1 package org.studyeasy;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         Main main = new Main();
7         try {
8             main.doSomething();
9         } catch (ZeroException e) {
10             // TODO Auto-generated catch block
11             e.printStackTrace();
12         }
13     }
14
15     class ZeroException extends Exception{
16     }
17
18     public void doSomething() throws ZeroException{
19         String x = "zero";
20         if(x.equals("zero")) {
21             throw new ZeroException();
22         }
23         System.out.println("Do something");
24     }
25 }
```

Declaration @ Javadoc Problems Console ×

```
<terminated> Main (4) [Java Application] C:\Users\dinit\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre
org.studyeasy.Main$ZeroException
    at org.studyeasy.Main.doSomething(Main.java:20)
    at org.studyeasy.Main.main(Main.java:8)
```

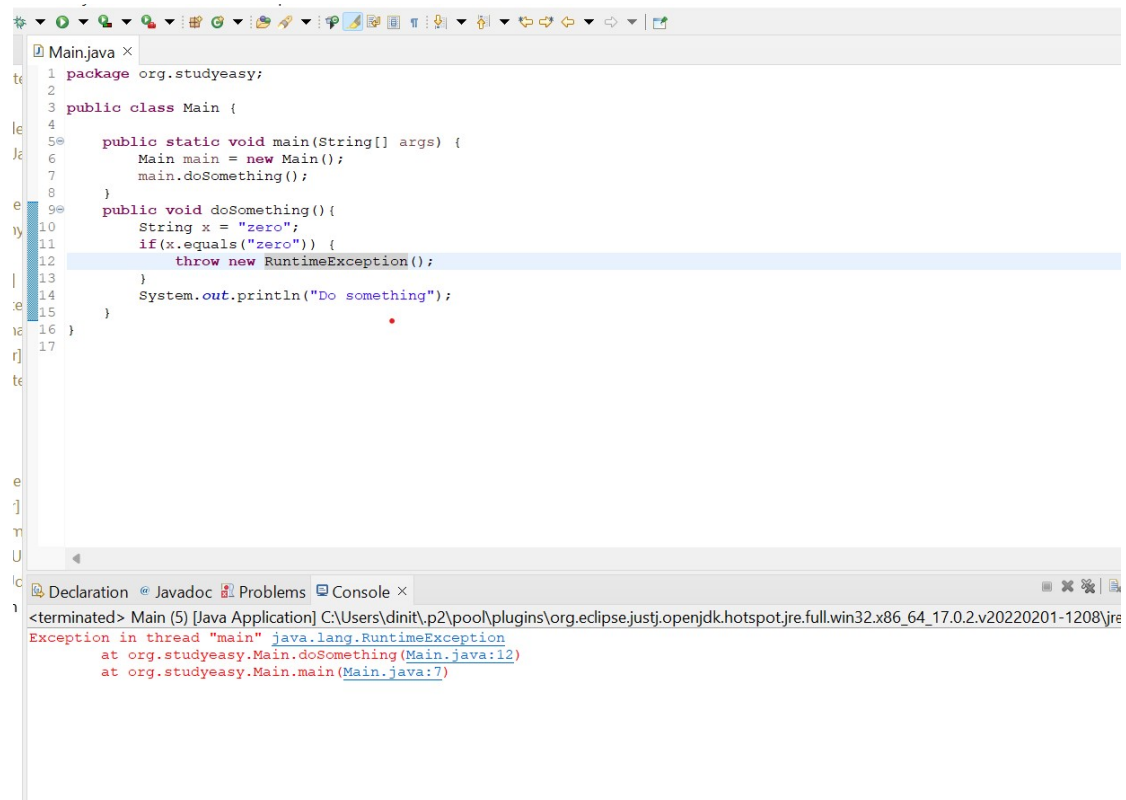
This is the how Standard Custom Exception structure looks like below.

```
class ZeroException extends Exception{
    }
}
```

Custom exception is not widely used.

➤ CHECKED AND UNCHECKED EXCEPTIONS JAVA

- ✓ Unchecked Exception are casual or less serious exception. Runtime Error is not seriously error. So it is unchecked.



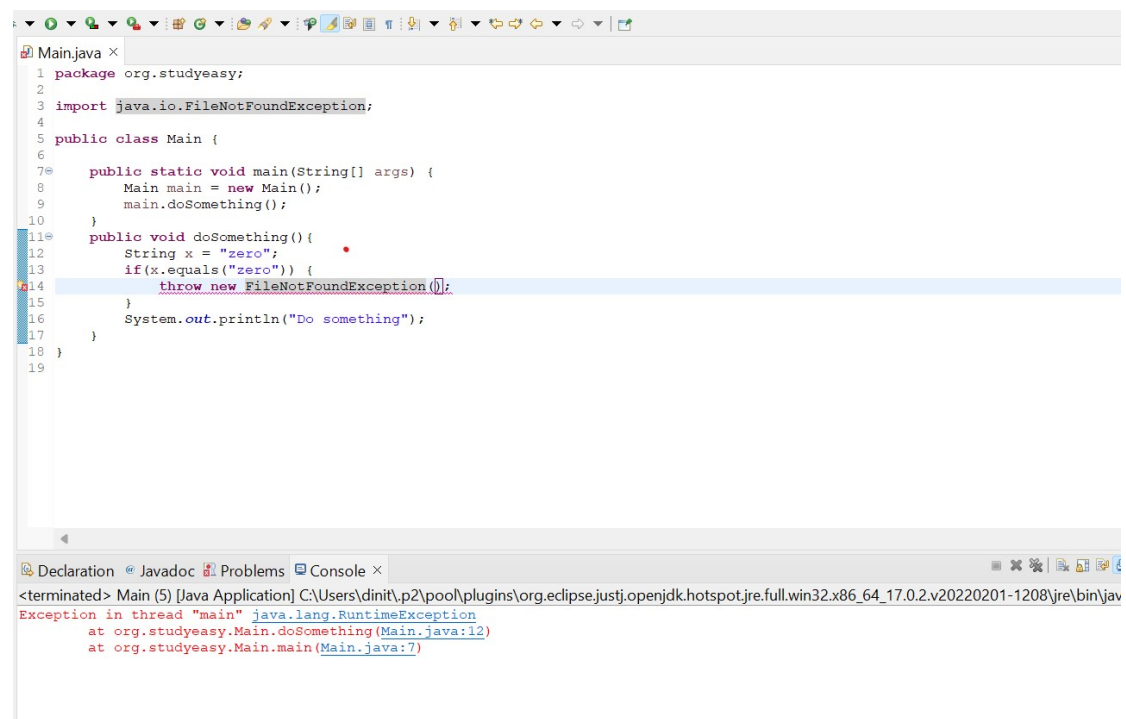
```
1 package org.studyeasy;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         Main main = new Main();
7         main.doSomething();
8     }
9     public void doSomething(){
10         String x = "zero";
11         if(x.equals("zero")) {
12             throw new RuntimeException();
13         }
14         System.out.println("Do something");
15     }
16 }
17
```

Declaration @ Javadoc Problems Console ×

<terminated> Main (5) [Java Application] C:\Users\dinit\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre

Exception in thread "main" java.lang.RuntimeException
at org.studyeasy.Main.doSomething(Main.java:12)
at org.studyeasy.Main.main(Main.java:7)

✓ Checked Exception - Not serious type exceptions - Example : **FileNotFoundException**



```
1 package org.studyeasy;
2
3 import java.io.FileNotFoundException;
4
5 public class Main {
6
7     public static void main(String[] args) {
8         Main main = new Main();
9         main.doSomething();
10    }
11    public void doSomething(){
12        String x = "zero";
13        if(x.equals("zero")) {
14            throw new FileNotFoundException();
15        }
16        System.out.println("Do something");
17    }
18 }
19
```

Declaration @ Javadoc Problems Console ×

<terminated> Main (5) [Java Application] C:\Users\dinit\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bin\jav

Exception in thread "main" java.lang.RuntimeException
at org.studyeasy.Main.doSomething(Main.java:12)
at org.studyeasy.Main.main(Main.java:7)

✓ Custom Exceptions by default are Checked or Serious Exceptions.