**ClassNotFoundException** and **NoClassDefFoundError** occur when a particular class is not found at runtime. However, they occur at different scenarios.

**ClassNotFoundException** is an exception that occurs when you try to load a class at run time using **Class.forName()** or **loadClass()** methods and mentioned classes are not found in the classpath.

**NoClassDefFoundError** is an error that occurs when a particular class is present at compile time, but was missing at run time.

**ClassNotFoundException**

ClassNotFoundException is a runtime exception that is thrown when an application tries to load a class at runtime using the **Class.forName()** or **loadClass()** or **findSystemClass()** methods ,and the class with specified name are not found in the classpath. For example, you may have come across this exception when you try to connect to MySQL or Oracle databases and you have not updated the classpath with required JAR files. Most of the time, this exception occurs when you try to run an application without updating the classpath with required JAR files.

For example, the below program will throw ClassNotFoundException if the mentioned class **“oracle.jdbc.driver.OracleDriver”** is not found in the classpath.

public class MainClass

{

public static void main(String[] args)

{

try

{

Class.forName("oracle.jdbc.driver.OracleDriver");

}catch (ClassNotFoundException e)

{

e.printStackTrace();

}

}

}

If you run the above program without updating the classpath with required JAR files, you will get an exception akin to:

java.lang.ClassNotFoundException: oracle.jdbc.driver.OracleDriver

at java.net.URLClassLoader.findClass(Unknown Source)

at java.lang.ClassLoader.loadClass(Unknown Source)

at sun.misc.Launcher$AppClassLoader.loadClass(Unknown Source)

at java.lang.ClassLoader.loadClass(Unknown Source)

at java.lang.Class.forName0(Native Method)

at java.lang.Class.forName(Unknown Source)

at pack1.MainClass.main(MainClass.java:17)

**NoClassDefFoundError**

NoClassDefFoundError is an error that is thrown when the Java Runtime System tries to load the definition of a class, and that class definition is no longer available. The required class definition was present at compile time, but it was missing at runtime. For example, compile the program below.

class A

{

// some code

}

public class B

{

public static void main(String[] args)

{

A a = new A();

}

}

When you compile the above program, two .class files will be generated. One is **A.class** and another one is **B.class**. If you remove the **A.class** file and run the **B.class**file, Java Runtime System will throw NoClassDefFoundError like below:

Exception in thread "main" java.lang.NoClassDefFoundError: A

at MainClass.main(MainClass.java:10)

Caused by: java.lang.ClassNotFoundException: A

at java.net.URLClassLoader.findClass(URLClassLoader.java:381)

at java.lang.ClassLoader.loadClass(ClassLoader.java:424)

at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:331)

at java.lang.ClassLoader.loadClass(ClassLoader.java:357)

**Recap**

|  |  |
| --- | --- |
| **ClassNotFoundException** | **NoClassDefFoundError** |
| It is an exception. It is of type java.lang.Exception. | It is an error. It is of type java.lang.Error. |
| It occurs when an application tries to load a class at run time which is not updated in the classpath. | It occurs when java runtime system doesn’t find a class definition, which is present at compile time, but missing at run time. |
| It is thrown by the application itself. It is thrown by the methods like Class.forName(), loadClass() and findSystemClass(). | It is thrown by the Java Runtime System. |
| It occurs when classpath is not updated with required JAR files. | It occurs when required class definition is missing at runtime. |