**Everything About ClassNotFoundException Vs NoClassDefFoundError**

posted by [Naresh Joshi](https://plus.google.com/108302142446482002391) on April 08, 2017  [No Comments](https://programmingmitra.blogspot.com/2017/04/Difference-Between-ClassNotFoundException-and-NoClassDefFoundError.html#comment-form)

We know Java is an Object Oriented Programming Language and almost everything is an object in Java and in order to create an object we need a class.  
  
While executing our program whenever JVM find a class, First JVM will try to load that class into memory if it has not done it already.  
  
For Example, If JVM is executing below the line of code, before creating the object of Employee class JVM will load this class into memory using a ClassLoader.

Employee emp = **new** Employee();

In above example, JVM will load the Employee class because it is present in the execution path and JVM want to create an object of this class.  
  
But we can also ask JVM to just load a class through its string name using Class.forName() or ClassLoader.findSystemClass() or ClassLoader.loadClass() methods. For Example below the line of code will only load the Employee class into memory and do nothing else.

Class.forName("Employee");

Both **ClassNotFoundException**and **NoClassDefFoundError**occur when a particular class is not found at run time but under different scenarios. And here in this article, we going to study these different scenarios.

**ClassNotFoundException**

Is a checked exception that occurs when we tell JVM to load a class by its string name usingClass.forName() or ClassLoader.findSystemClass() or ClassLoader.loadClass() methods and mentioned class is not found in the classpath.  
  
Most of the time, this exception occurs when you try to run an application without updating the classpath with required JAR files. For Example, You may have seen this exception when doing the JDBC code to connect to your database i.e.MySQL but your classpath does not have JAR for it.  
  
If we compile below example, the compiler will produce two class files Test.class and Person.class. And Now if we execute the program it will successfully print Hello. But if we delete Person.class file and again try to execute the program we will receive ClassNotFoundException.

**public** **class** Test {

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

*// ClassNotFoundException Example*

*// Provide any class name to Class.forName() which does not exist*

*// Or compile Test.java and then manually delete Person.class file so Person class will become unavailable*

*// Run the program using java Test*

Class clazz = Class.forName("Person");

Person person = (Person) clazz.newInstance();

person.saySomething();

}

}

class Person {

**void** saySomething() {

System.**out**.println("Hello");

}

}

**NoClassDefFoundError**

Is a subtype of java.lang.Error and Error class indicates an abnormal behavior which really should not happen with an application but and application developers should not try to catch it, it is there for JVM use only.  
  
NoClassDefFoundError occurs when JVM tries to load a particular class that is the part of your code execution (as part of a normal method call or as part of creating an instance using the newkeyword) and that class is not present in your classpath but was present at compile time because in order to execute your program you need to compile it and if you are trying use a class which is not present compiler will raise compilation error.  
  
Similar to above example if we try to compile below program, we will get two class files Test.classand Employee.class. A and on execution it will print Hello.

**public** **class** Test {

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

*// NoClassDefFoundError Example*

*// Do javac on Test.java,*

*// Program will compile successfully because Empoyee class exits*

*// Manually delete Employee.class file*

*// Run the program using java Test*

Employee emp = **new** Employee();

emp.saySomething();

}

}

class Employee {

**void** saySomething() {

System.**out**.println("Hello");

}

}

But if we delete Employee.class and try to execute the program we will get NoClassDefFoundError.

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

at Test.main(Test.java:9)

Caused by: java.lang.ClassNotFoundException: Employee

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

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

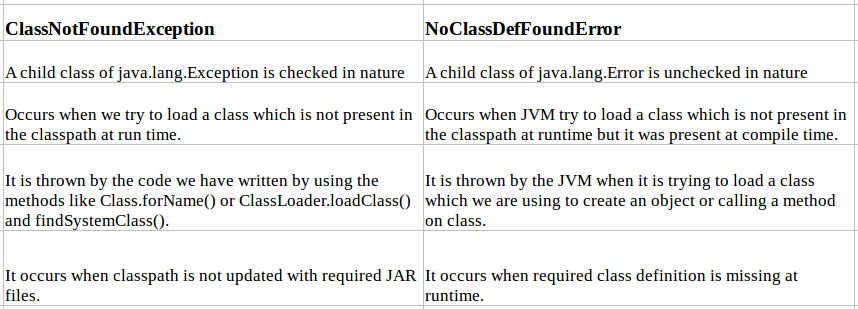
at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:331)

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

... 1 more

As you can see in above stack trace NoClassDefFoundError is caused by ClassNotFoundException, because JVM is not able to find the Employee class in the class path.

**Conclusion**

[[](https://4.bp.blogspot.com/-C_sn-jti8HE/WOioqaoU0UI/AAAAAAAAK7w/ps_eyJ3QVHkju5CAWdoQM_2IvgneT21nwCK4B/s1600/Difference-Between-ClassNotFoundException-and-NoClassDefFoundError.png)](https://4.bp.blogspot.com/-C_sn-jti8HE/WOioqaoU0UI/AAAAAAAAK7w/ps_eyJ3QVHkju5CAWdoQM_2IvgneT21nwCK4B/s1600/Difference-Between-ClassNotFoundException-and-NoClassDefFoundError.png)