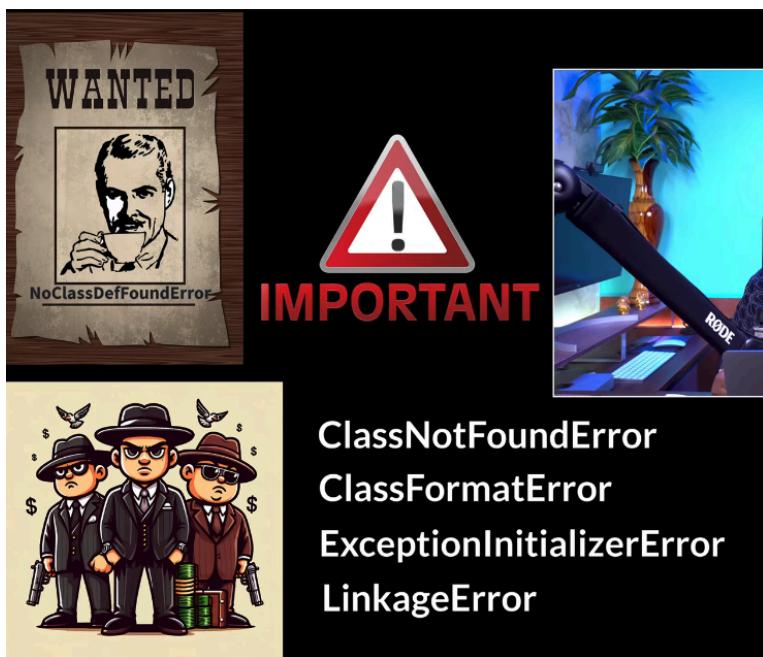


5 JAVA ERRORS you should learn today (NoClassDefFoundError) | JVM architecture - ClassLoader

Static class loading Vs Dynamic class loading



NoClassDefFoundError is main villain. The other errors his right hands

Video Contents :

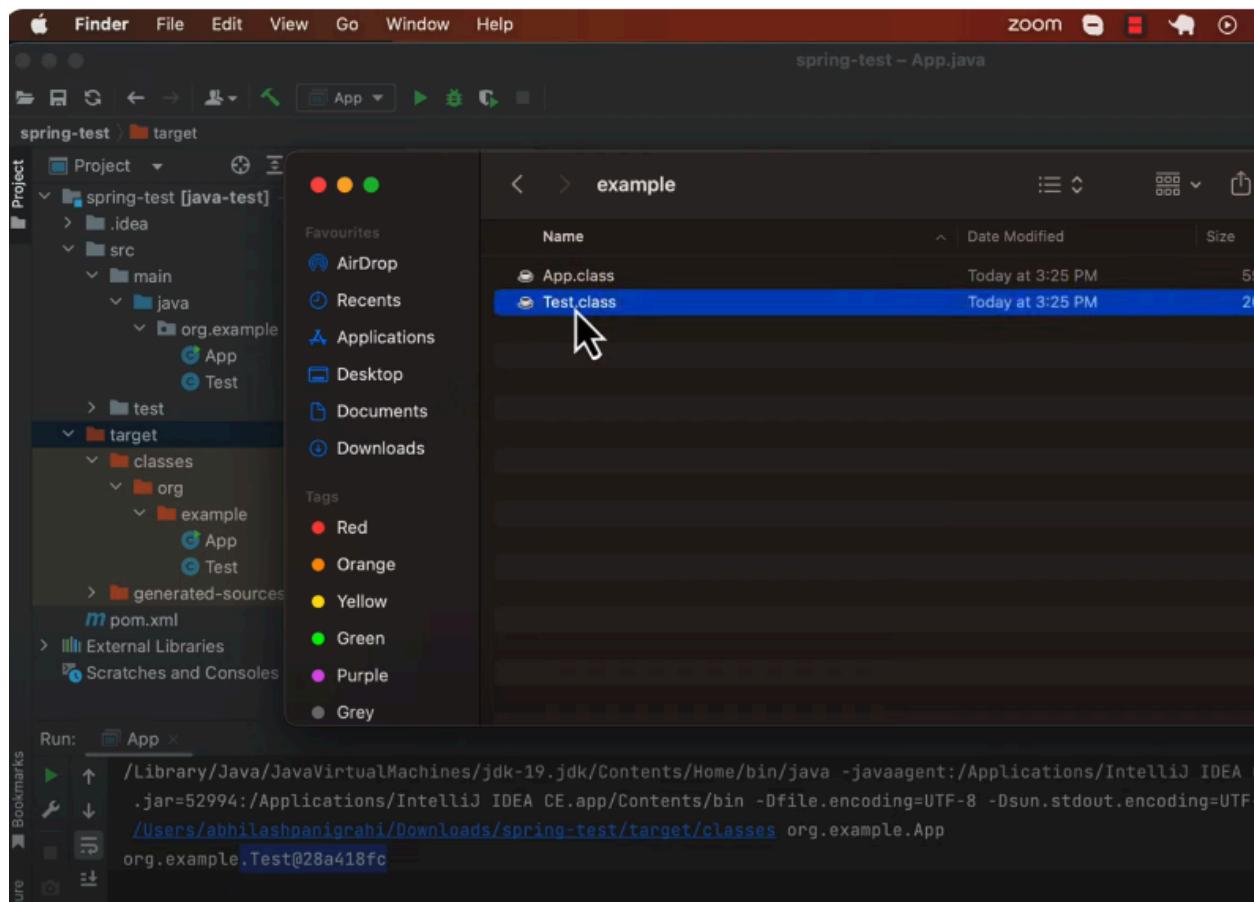
Static class Loading
Dynamic class loading
JVM architecture internals
Class loader subsystems
NoClassDefFoundError
ExceptionInitializerError
Real-Time problem solving

Just print test class object

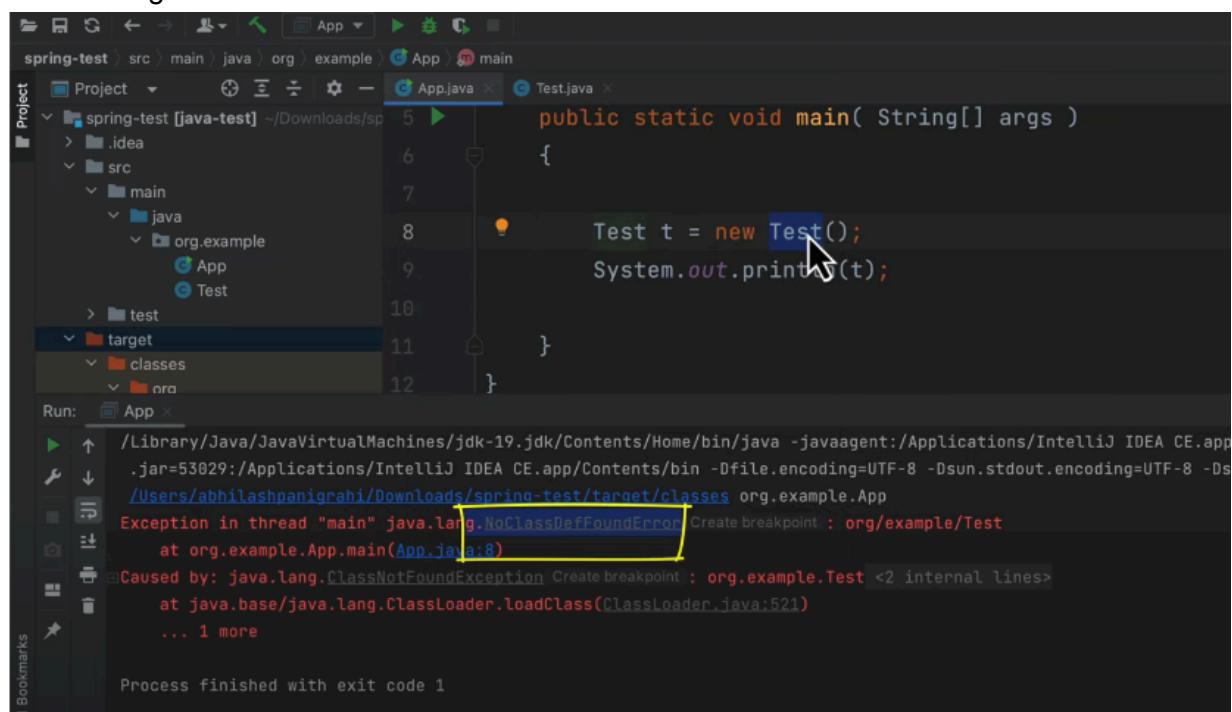
The screenshot shows the IntelliJ IDEA interface. On the left, the Project tool window displays a file structure with a 'main' directory containing 'java', 'org.example', 'App', and 'Test'. Below 'main' are 'test', 'target', 'pom.xml', 'External Libraries', and 'Scratches and Consoles'. The 'Run' tool window at the bottom shows a configuration for 'App' with the command: '/Library/Java/JavaVirtualMachines/jdk-19.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ .jar=52994:/Applications/IntelliJ IDEA CE.app/Contents/bin -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=ISO-8859-1 /Users/abhilashpanigrahi/Downloads/spring-test/target/classes org.example.App org.example.Test@28a418fc'. The status bar indicates 'Process finished with exit code 0'. The main editor window shows the following Java code:

```
public class App {  
    public static void main( String[] args ) {  
        Test t = new Test();  
        System.out.println(t);  
    }  
}
```

Just delete .class file from target folder and run



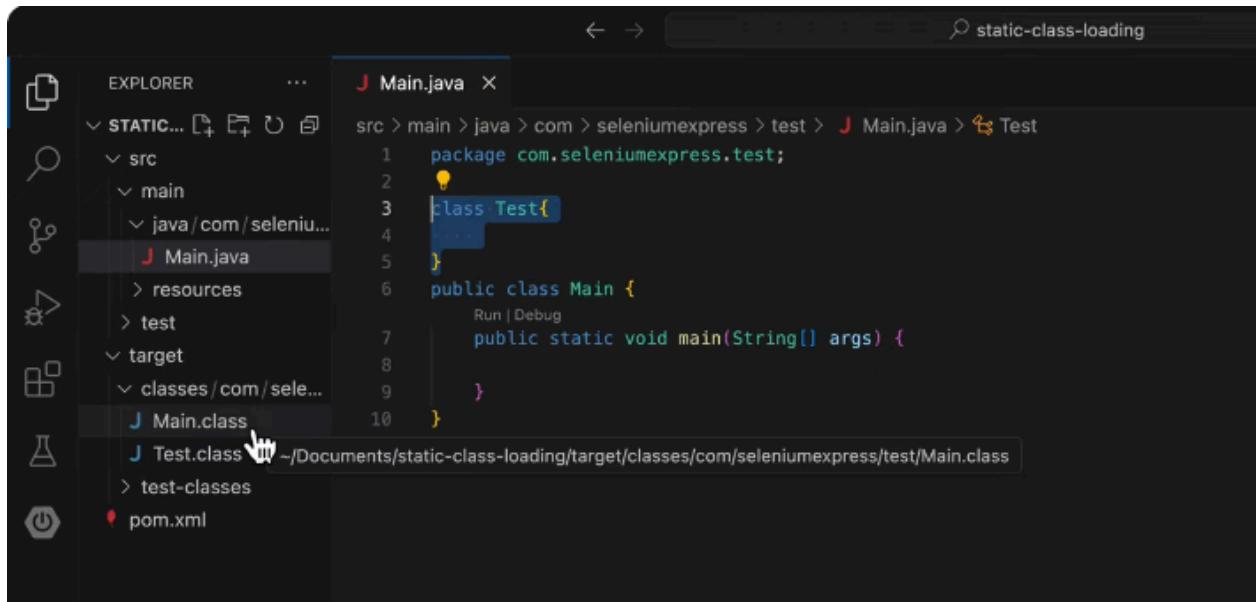
Now run the same code again. We get NoClassDefFoundError. If it is your understanding then you are wrong



The Foundation

Static Class Loading

[Inheritance, Association & errors]



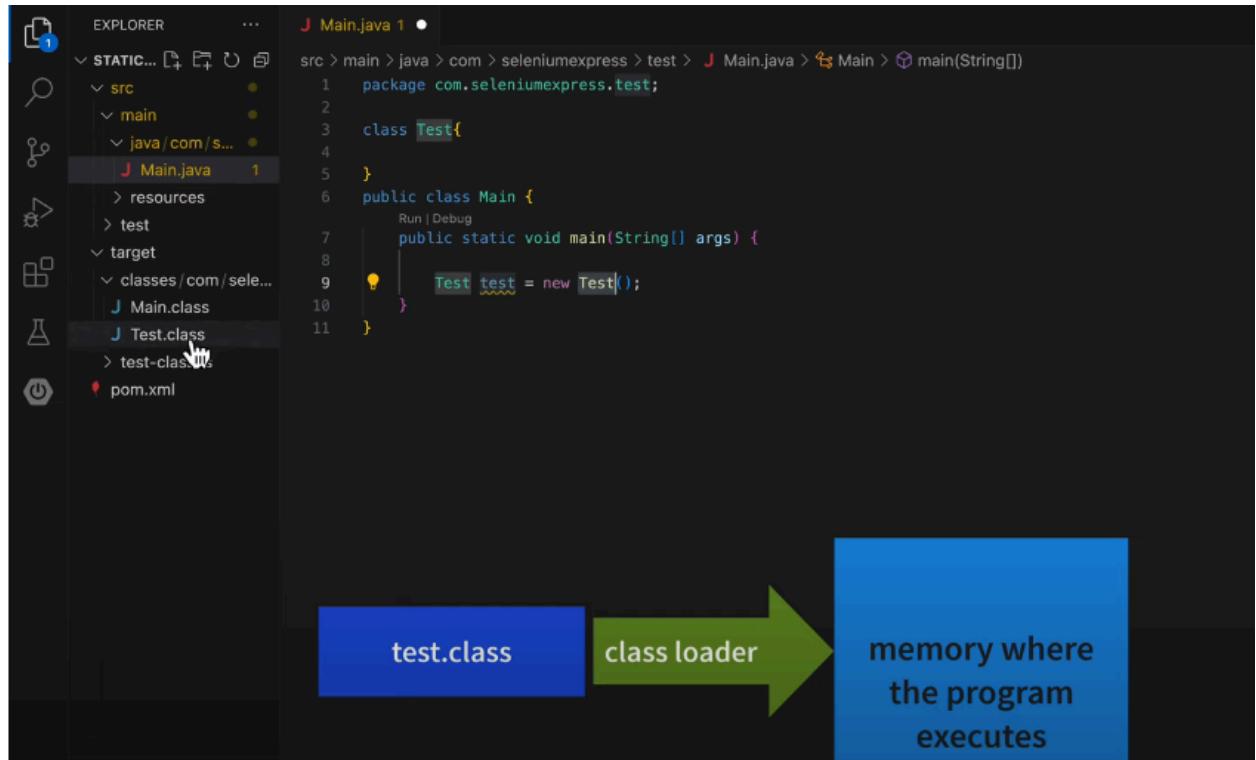
The screenshot shows an IDE interface with the following details:

- EXPLORER** view: Shows a project structure with a package named `com.seleniumexpress.test`. It contains a `src` folder with `main`, `test`, and `target` subfolders. Inside `main`, there is a `java/com/seleniumexpress/test` folder containing `J Main.java`. Inside `target`, there is a `classes/com/sele...` folder containing `J Main.class` and `J Test.class`.
- Main.java** file content:

```
1 package com.seleniumexpress.test;
2
3 class Test{
4 }
5
6 public class Main {
7     Run | Debug
8     public static void main(String[] args) {
9         }
10 }
```

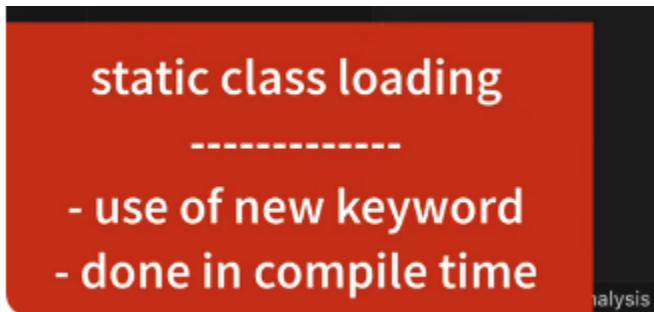
- Main.class** file content (shown in a tooltip):
`~/.Documents/static-class-loading/target/classes/com/sele.../test/Main.class`
- pom.xml**: A red error icon is present.

We have got two class files which contains bytecode and will load into memory with help of java class loader.



```
5  }
6  public class Main {
7      Run | Debug
8      public static void main(String[] args) {
9          Test test = new Test(); //static class loading
10     }
11 }
```

Test class will be loaded into memory when line 9 executes. From next time onward wen we refer the object it will be very fast.



Sample class is not loaded even though we have a static block in it.

The screenshot shows the VS Code interface with the following details:

- EXPLORER:** Shows a project structure with a file named **J Main.java** selected.
- MAIN CODE (J Main.java):**

```
src > main > java > com > seleniumexpress > test > J Main.java > Main > main(String[])
3   class Sample{
4       static{
5           System.out.println("test class loaded..");
6       }
7   }
8 }
9 class Test{
10    static{
11        System.out.println("test class loaded..");
12    }
13 }
14 }
15 public class Main {
16     Run | Debug
17     public static void main(String[] args) {
18         Test test = new Test(); //static class loading
19     }
20 }
```
- OUTPUT:** Shows command-line output:

```
abhilashpanigrahi@Abhilash-Mac-mini static-class-loading % /usr/bin/java -XX:+ShowCodeDetailsInExceptionMessages -cp /Users/abhilashpanigrahi/Documents/staticexpress/test.Main
test class loaded..
abhilashpanigrahi@Abhilash-Mac-mini static-class-loading %
```
- PROBLEMS:** Shows one error: **Java: Ready**.
- SIDE BAR:** Shows icons for OUTLINE, TIMELINE, JAVA PROJECTS, and MAVEN.
- Bottom Status Bar:** Shows icons for search, refresh, and other status information.

A large blue speech bubble on the right side of the interface contains the text "will it load Sample class ?".

After adding a sample class into the main method then only it runs. Refer below screenshot

The screenshot shows the VS Code interface with the following details:

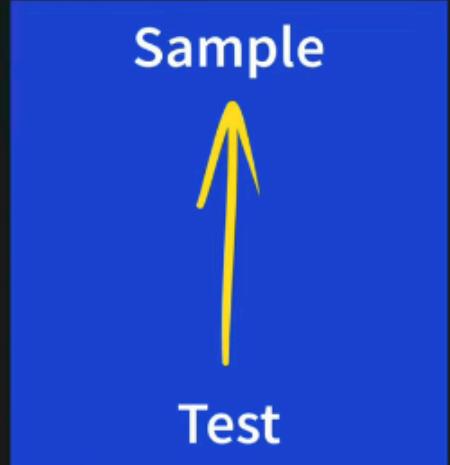
- EXPLORER**: Shows the project structure under "STATIC-CLASS-LOADING".
 - src
 - main
 - java/com/s...
 - J Main.java (selected)
 - resources
 - test
 - target
 - classes/com/sele...
 - J Main.class
 - J Sample.class
 - J Test.class
 - test-classes
 - pom.xml
- J Main.java**: The code editor window displays the following Java code:

```
src > main > java > com > seleniumexpress > test > J Main.java > Sample
1 package com.seleniumexpress.test;
2
3 class Sample{
4     static{
5         System.out.println("Sample class loaded..");
6     }
7 }
8
9 class Test{
10    static{
11        System.out.println("test class loaded..");
12    }
13 }
14
15 public class Main {
16     public static void main(String[] args) {
17
18         Test test = new Test(); //static class loading
19         Sample sample = new Sample();
20     }
21 }
```
- TERMINAL**: The terminal window shows the execution of the Java code, printing "test class loaded.." and "Sample class loaded.." to the console.
- PROBLEMS**: Shows 2 errors.
- OUTPUT**: Shows Java: Ready.
- DEBUG CONSOLE**: Shows RHDA analysis has failed.
- OUTLINE**, **TIMELINE**, **JAVA PROJECTS**, and **MAVEN** are also visible in the sidebar.

J Main.java 1 X

src > main > java > com > seleniumexpress > test > J Main.java > Sample

```
1 package com.seleniumexpress.test;
2
3 class Sample{
4     static{
5         System.out.println("Sample class loaded..");
6     }
7 }
8
9 class Test extends Sample{
10    static{
11        System.out.println("test class loaded..");
12    }
13 }
14
15 public class Main {
16     Run | Debug
17     public static void main(String[] args) {
18         Test test = new Test(); //static class loading
19     }
20 }
```



The screenshot shows a Java code editor with the file `Main.java` open. The code contains several static blocks that print class loading messages. Line 5 has a syntax error, indicated by a red squiggle under `System.out.println("Sample class loaded..");`. The terminal below shows the execution of the code, where the first two prints are successful, but the third one fails with an `ExceptionInInitializerError`.

```
... J Main.java 1 X
ING src > main > java > com > seleniumexpress > test > J Main.java > Sample
  1 package com.seleniumexpress.test;
  2
  3 class Sample{
  4     static{
  5         System.out.println("Sample class loaded..");
  6     }
  7
  8 }
  9 class Test extends Sample{
10     static{
11         System.out.println("test class loaded..");
12     }
13
14 }
15 public class Main {
16     Run | Debug
17     public static void main(String[] args) {
18         Test test = new Test(); //static class loading
19     }
20 }
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
● abhilashpanigrahi@Abhilashs-Mac-mini static-class-loading % /usr/bin/env /Users/abhilashpanigrahi/.java/bin/java -XX:+ShowCodeDetailsInExceptionMessages -cp /Users/abhilashpanigrahi/Downloads/seleniumexpress/test Sample class loaded.. ←
test class loaded..
○ abhilashpanigrahi@Abhilashs-Mac-mini static-class-loading %
```

Whenever we have exception in super class static block we will get `ExceptionInitializerError`.
In line number 5 contains error.

`ExceptionInitializerError` error will come whenever static block fails.

guess the output :

The screenshot shows a Java project structure in the left sidebar with a file named Main.java selected. The code defines a Sample class with a static block that prints "Sample class loaded..". It also defines a Test class that extends Sample and a Main class with a main method that creates an instance of Test. The terminal tab shows the output of running the program, which includes the static class loading message and an exception stack trace for an ArithmeticException.

```
src > main > java > com > seleniumexpress > test > Main.java > Main > main(String[])
1 package com.seleniumexpress.test;
2
3 class Sample{
4     static{
5         int i = 10/0;
6         System.out.println("Sample class loaded..");
7     }
8 }
9
10 class Test extends Sample{
11     static{
12         System.out.println("test class loaded..");
13     }
14 }
15
16 public class Main {
17     Run | Debug
18     public static void main(String[] args) {
19         Test test = new Test(); //static class loading
20     }
21 }
```

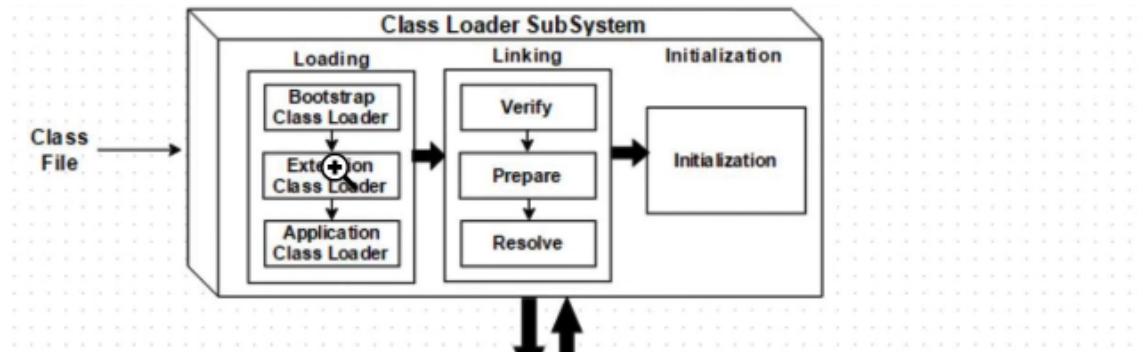
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
test class loaded..
○ abhilashpanigrahi@Abhilashs-Mac-mini static-class-loading %

○ abhilashpanigrahi@Abhilashs-Mac-mini static-class-loading %
● abhilashpanigrahi@Abhilashs-Mac-mini static-class-loading % cd /Users/abhilashpanigrahi/Documents/static-class-loading/target/classes com.seleniumexpress.test.Main
Exception in thread "main" java.lang.ExceptionInInitializerError
    at com.seleniumexpress.test.Sample.<clinit>(Main.java:19)
    Caused by: java.lang.ArithmetricException: / by zero
        at com.seleniumexpress.test.Sample.<clinit>(Main.java:5)
        ... 1 more
○ abhilashpanigrahi@Abhilashs-Mac-mini static-class-loading %
```

OUTLINE TIMELINE JAVA PROJECTS MAVEN

How ClassLoader works ? (JVM Internals)



ExceptionInitializerError error happens in the initialization phase in JVM.

Loading stage : The application class loader will load user defined class files and BootStrap and Extension class loaders load system class and builtin class files.

In the linking stage: memory will be allocated.

Initialization stage : static blocks and other object initialization will happen

Now create has a relationship between test and sample class as shown below.

A screenshot of a Java code editor showing the file `Main.java`. The code defines three classes: `Sample`, `Test`, and `Main`. The `Test` class contains a static block that creates an instance of `Sample`. A purple circle with the text "Association (has-a relationship)" points to the line where `Sample sample = new Sample();` is written. The code editor interface includes tabs for `Main.java` and `Test`, and a search bar at the top.

```

J Main.java 2 •
src > main > java > com > seleniumexpress > test > J Main.java > Test
1 package com.seleniumexpress.test;
2
3 class Sample{
4
5     static{
6         int i = 10/0;
7         System.out.println("Sample class loaded");
8     }
9 }
10 }
11 class Test {
12     Sample sample = new Sample();
13     static{
14         System.out.println("test class loaded");
15     }
16 }
17 }
18 public class Main {
19     Run | Debug
20     public static void main(String[] args) {
21
22         Test test = new Test(); //static class loading
23     }
}

```

Refer below console where catch block is not printed. This is because we have runtime exception.

The screenshot shows a Java code editor and a terminal window. The code editor has a file named Main.java open, showing the following code:

```
src > main > java > com > seleniumexpress > test > J Main.java > Main > main(String[])
11  class Test {
12
13  }
14  public class Main {
15      Run | Debug
16      public static void main(String[] args) {
17          try{
18              Test test = new Test(); //static class loading
19          }catch(Exception e){
20              System.out.println("inside catch block");
21              System.out.println(e);
22          }
23          System.out.println("*****");
24      }
25  }
26
27
28 }
```

The terminal window below shows the output of the application. It includes a stack trace for an `java.lang.ArithmaticException` and some command-line arguments.

```
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS
Caused by: java.lang.ArithmaticException: / by zero
    at com.seleniumexpress.test.Sample.<clinit>(Main.java:6)
    ... 2 more
○ abhilashpanigrahi@Abhilashs-Mac-mini static-class-loading %
○ abhilashpanigrahi@Abhilashs-Mac-mini static-class-loading %
○ abhilashpanigrahi@Abhilashs-Mac-mini static-class-loading % cd /Users/abhilashpanigrahi/Documents/static-class-load
grahi/.vscode/extensions/redhat.java-1.30.0-darwin-arm64/jre/17.0.10-macosx-aarch64/bin/java -XX:+ShowCodeDetailsInM
nigrahi/Documents/static-class-loading/target/classes com.seleniumexpress.test.Main
test class loaded..
Exception in thread "main" java.lang.ExceptionInInitializerError
    at com.seleniumexpress.test.Test.<init>(Main.java:12)
    at com.seleniumexpress.test.Main.main(Main.java:21)
Caused by: java.lang.ArithmaticException: / by zero
```

An error that happen in linking phase then it will come under linkageException

The screenshot shows the Java code for the `ExceptionInInitializerError` class. The code is as follows:

```
38  *
39  * @author  Frank Yellin
40  * @since   1.1
41  */
42  public class ExceptionInInitializerError extends LinkageError {
43      /**
44      * Use serialVersionUID from JDK 1.1.X for interoperability
45      */
46      @java.io.Serial
47      private static final long serialVersionUID = 1521711792217232256L;
```

After changing throwable it is printed catch block.

A screenshot of the VS Code interface. The code editor shows a Java file with the following code:

```
17 }
18 public class Main {
19     Run | Debug
20     public static void main(String[] args) {
21         try{
22             Test test = new Test(); //static class loading
23         }catch(Throwable e){
24             System.out.println("inside catch block");
25             System.out.println(e);
26         }
27     }
28 }
```

The terminal below shows the execution of the code. A yellow dot indicates the current line of execution. The output is:

```
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

Exception in thread "main" java.lang.ExceptionInInitializerError
    at com.seleniumexpress.test.Test.<init>(Main.java:12)
    at com.seleniumexpress.test.Main.main(Main.java:21)
Caused by: java.lang.ArithmetricException: / by zero
    at com.seleniumexpress.test.Sample.<clinit>(Main.java:6)
    ... 2 more
○ abhilashpanigrahi@Abhilash-Mac-mini static-class-loading %
○ abhilashpanigrahi@Abhilash-Mac-mini static-class-loading %
● abhilashpanigrahi@Abhilash-Mac-mini static-class-loading % cd /Users/abhilashpanigrahi/Documents/static-class-loading/target/classes com.seleniumexpress.test.Main
test class loaded..
inside catch block
java.lang.ExceptionInInitializerError
*****
```

Now create one more test object after catch block. We will get NoClassDefFoundError.

A screenshot of the VS Code interface. The code editor shows the same Java file as before, but with an additional line of code added after the catch block:

```
11 class Test {
12 }
13
14
15
16
17 }
18 public class Main {
19     Run | Debug
20     public static void main(String[] args) {
21         try{
22             Test test = new Test(); //static class loading
23         }catch(Throwable e){
24             System.out.println("inside catch block");
25             System.out.println(e);
26         }
27         Test test = new Test();
28     }
29 }
```

The terminal shows the same initial output as before, followed by a modal dialog box with the text "Let's welcome the VIP". After the dialog, the output continues with:

```
PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS

○ abhilashpanigrahi@Abhilash-Mac-mini static-class-loading %
○ abhilashpanigrahi@Abhilash-Mac-mini static-class-loading %
● abhilashpanigrahi@Abhilash-Mac-mini static-class-loading % cd /Users/abhilashpanigrahi/Documents/static-class-loading/target/classes com.seleniumexpress.test.Main
test class loaded..
inside catch block
java.lang.ExceptionInInitializerError
*****
Exception in thread "main" java.lang.NoClassDefFoundError: Could not initialize class com.seleniumexpress.te
    at com.seleniumexpress.test.Test.<init>(Main.java:12)
    at com.seleniumexpress.test.Main.main(Main.java:27)
```

When it was previously tried we got an exception. So JVM imagine now also we will get this error so through NoClassDefFoundError



Cause of
NoClassDefFoundError

```
... J Main.java 3 ...
NG src > main > java > com > seleniumexpress > test > J Main.java > Main > main(String[])
11  class Test {
12      }
13  }
14
15  public class Main {
16      Run | Debug
17      public static void main(String[] args) {
18          try{
19              Test test = new Test(); //static class loading
20          }catch(Throwable e){
21              System.out.println("inside catch block");
22              System.out.println(e);
23          }
24          System.out.println("*****");
25          Test test = new Test(); //static class loading
26      }
27  }
28
29 }
```

If you change .class file (delete some characters) we will get ClassFormatError

```
grahvi/.vscode/extensions/redhat.java-1.30.0-darwin-arm64/jre/17.0.10-macosx-aarch64/bin/java -XX:+Show
nigrahi/Documents/static-class-loading/target/classes com.seleniumexpress.test.Main
inside catch block
java.lang.ClassFormatError: Illegal exception table range in class file com/seleniumexpress/test/Test
*****
```

It is linked to LinkageError

```
34  /*
35  public class ClassFormatError extends LinkageError {
36      @java.io.Serial
37      private static final long serialVersionUID = -8420114879011949195L;
38
39  /**
```

The Intermediate Dealing With Multiple Dependencies

The below concept is missing dependencies

We have three projects with single class like below

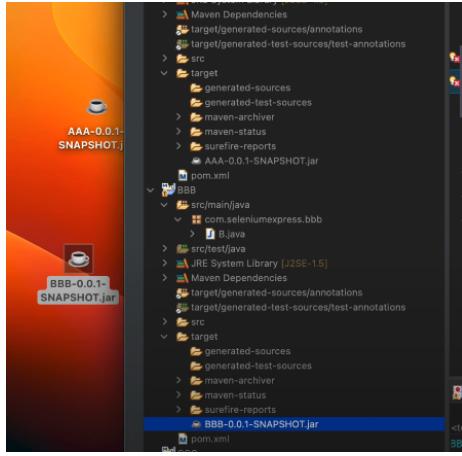
The screenshot shows the Eclipse IDE interface. On the left is the 'Package Explorer' view, which lists several projects: AAA, BBB, CCC, Project-test, and Servers. Project BBB is expanded, showing its src/main/java and src/test/java directories, along with JRE System Library [J2SE-1.5], Maven Dependencies, and a pom.xml file. The 'src/main/java/com.seleniumexpress.bbb' folder contains a file named B.java. On the right is the code editor window titled 'B.java', displaying the following Java code:

```
1 package com.seleniumexpress.bbb;
2
3 /**
4  * Hello world!
5  */
6 public class B
7 {
8 }
9
10 }
11 }
```

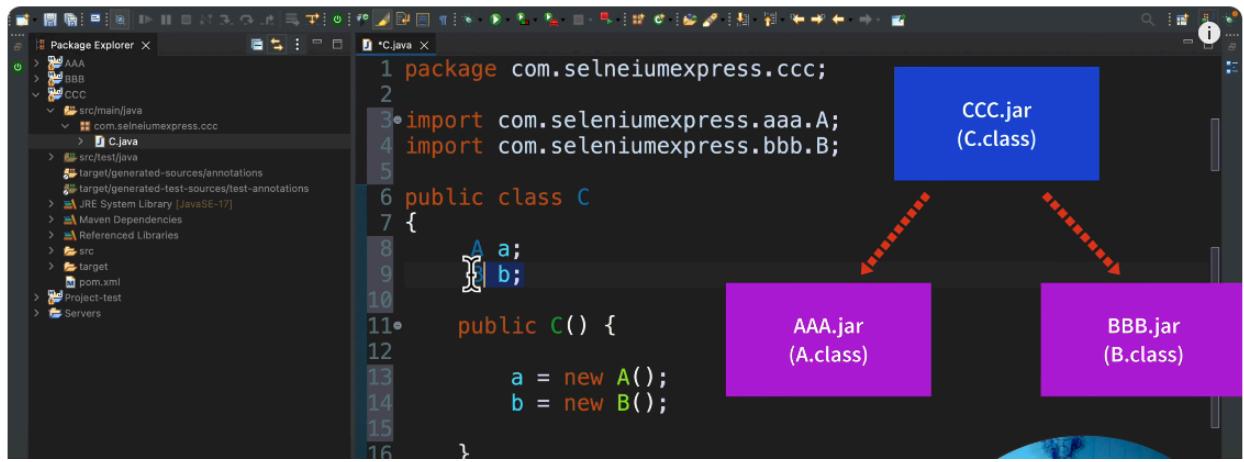
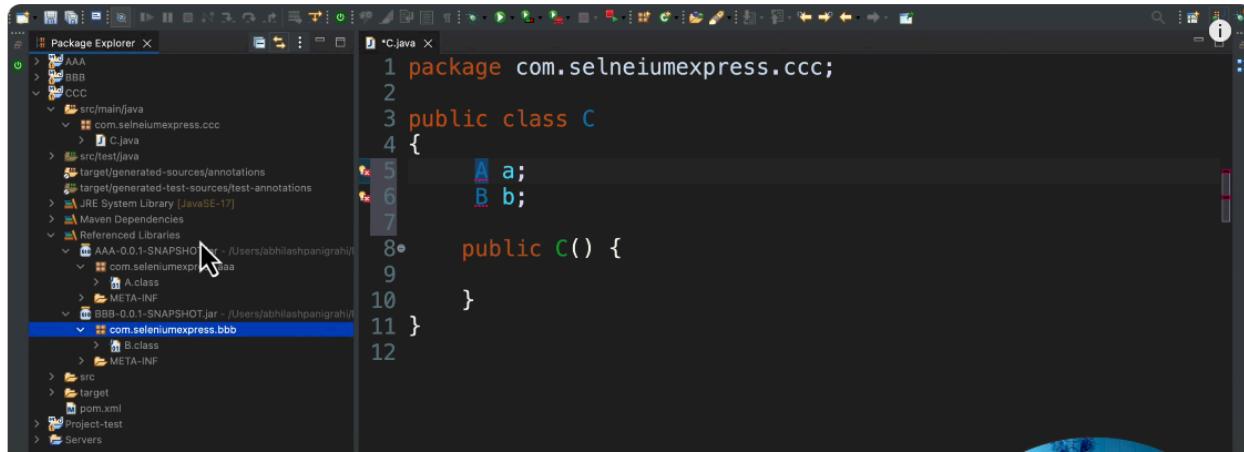
Build jars and include in project CCC

The screenshot shows the Spring Tool Suite 4 interface. On the left is the 'Project Explorer' view, which lists projects AAA, BBB, CCC, Project-test, and Servers. Project CCC is expanded, showing its src/main/java and src/test/java directories, along with JRE System Library [JavaSE-17], Maven Dependencies, and a pom.xml file. The 'src/main/java/com.seleniumexpress.ccc' folder contains a file named C.java. On the right is the code editor window titled 'se-test - CCC/src/main/java/com.seleniumexpress.ccc/C.java - Spring Tool Suite 4', displaying the following Java code:

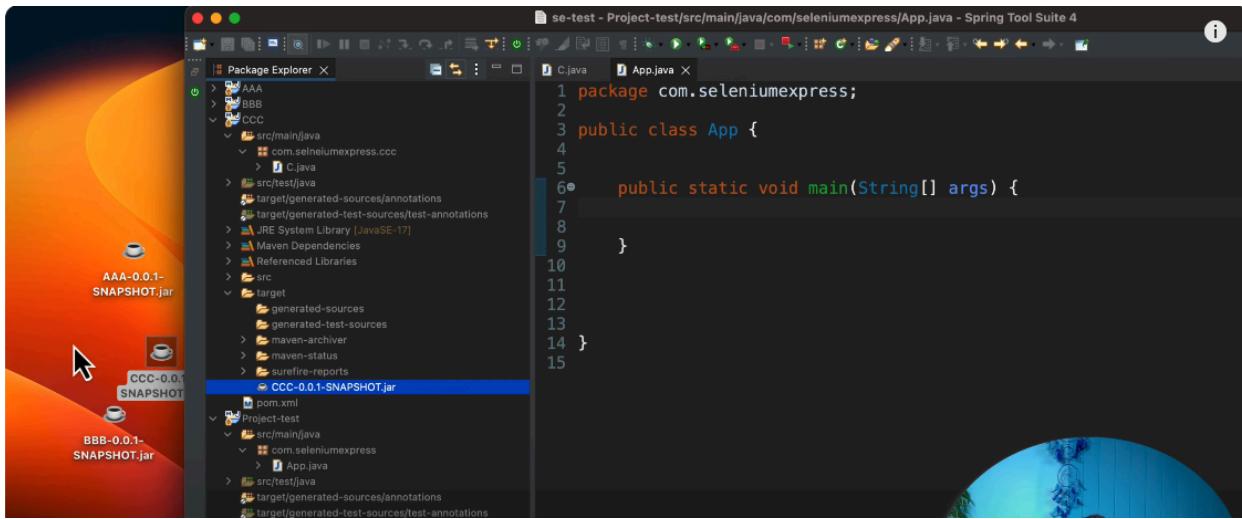
```
1 package com.seleniumexpress.ccc;
2
3 public class C
4 {
5     A a;
6     B b;
7
8     public C() {
9
10    }
11 }
```



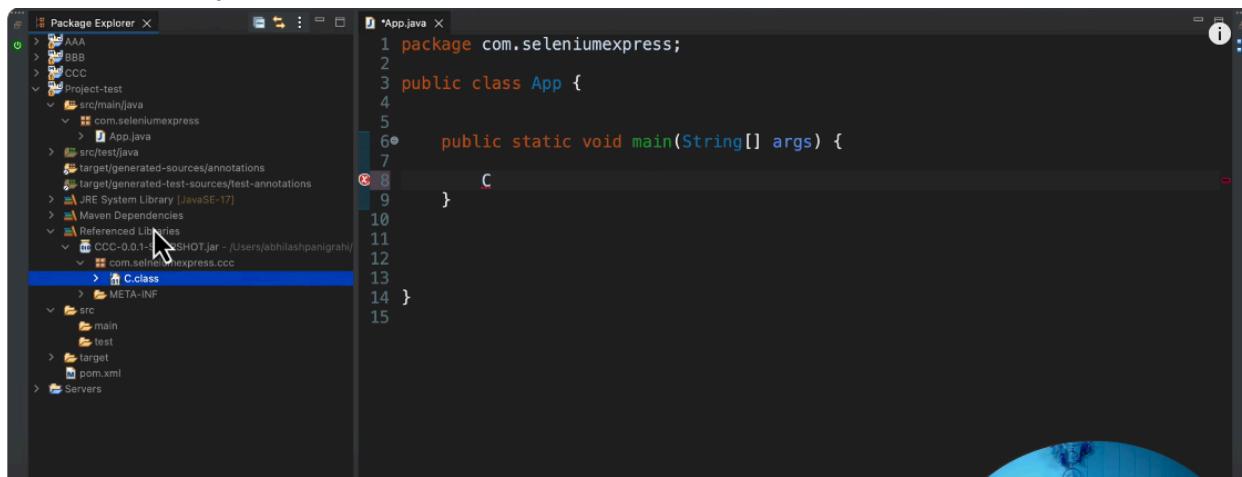
Include them as external jar file in classpath



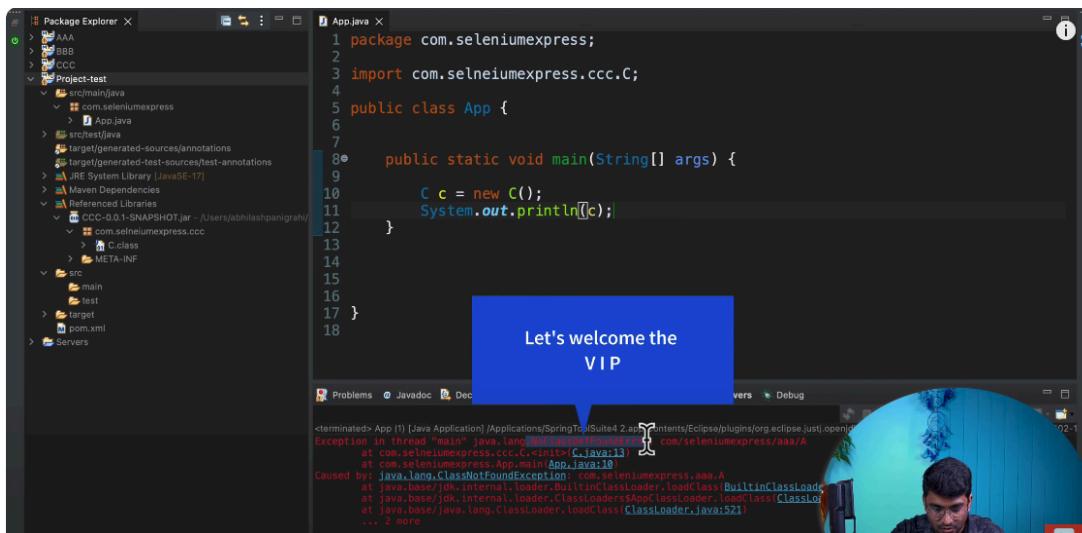
Now create jar file project ccc



Add it in new project



Now we got class not found exception



Now add project AAA.jar as it is showing noClassDefFoundError for A and now we got same error for B as well.

```

1 package com.seleniumexpress;
2
3 import com.seleniumexpress.ccc.C;
4
5 public class App {
6
7     public static void main(String[] args) {
8         C c = new C();
9         System.out.println(c);
10    }
11
12 }
13
14
15
16
17 }
18

```

AAA.jar (A.class)

BBB.jar (B.class)

After adding bbb.jar it ran well.

```

1 package com.seleniumexpress;
2
3 import com.seleniumexpress.ccc.C;
4
5 public class App {
6
7     public static void main(String[] args) {
8         C c = new C();
9         System.out.println(c);
10    }
11
12 }
13
14
15
16
17 }
18

```

Loaded fine..

CCC.jar (C.class)

AAA.jar (A.class)

BBB.jar (B.class)

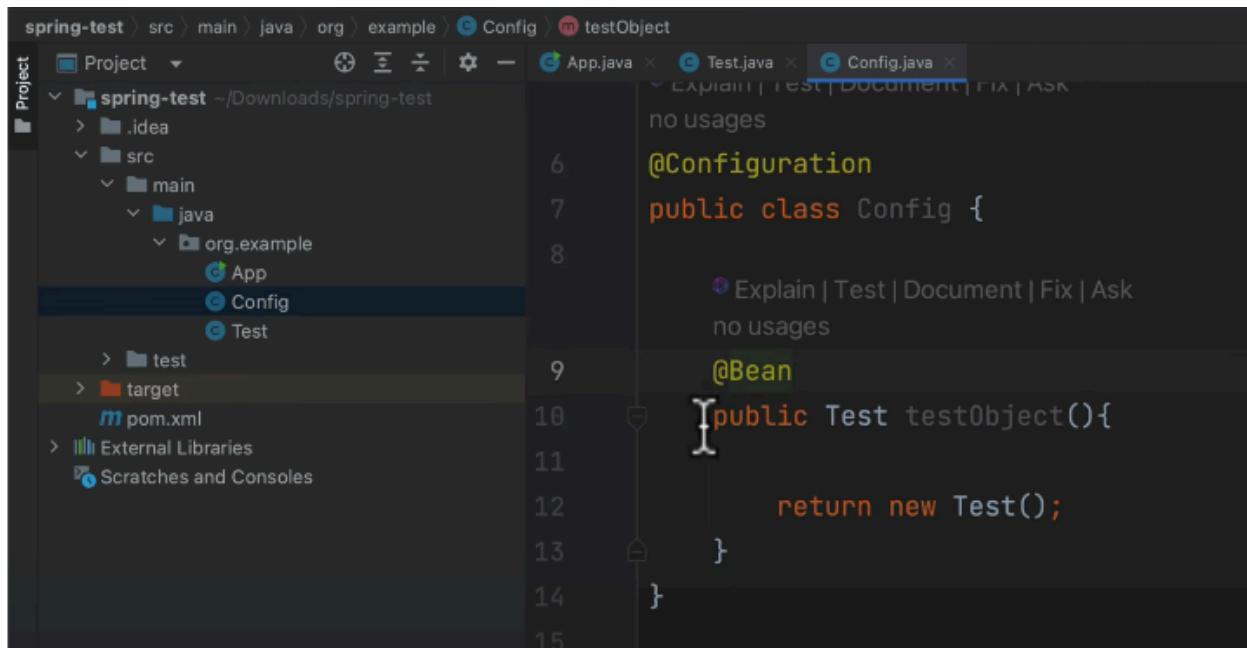
Advanced Framework Issues

(demo using spring framework)

The below concept is also called - the jar conflict

Most occurrences in our day to day. Wasting lot of time without knowing something wrong with our dependencies

Assume that we have below spring boot application



The screenshot shows a Java-based Spring Boot application structure in an IDE. The project tree on the left lists the following structure:

- Project: spring-test (~/Downloads/spring-test)
- src/main/java/org/example

 - App.java
 - Config.java (selected)
 - Test.java

- src/test
- target
- pom.xml
- External Libraries
- Scratches and Consoles

The code editor on the right displays the `Config.java` file:

```
no usages
@Configuration
public class Config {
    @Bean
    public Test testObject(){
        return new Test();
    }
}
```

```
spring-test / src / main / java / org / example / App.java
no usages
public class App {
}
no usages
public static void main( String[] args ) {
    ApplicationContext app = new AnnotationConfigApplicationContext(Config.class);
    System.out.println("container created..");
    System.out.println(app.getBean(Test.class));
}
```

All we want to print the bean

When we run we get NoClassDefFoundError

```
Run: App
Exception in thread "main" java.lang.NoClassDefFoundError: Create breakpoint : org.springframework.core.NestedIOException
    at org.springframework.context.annotation.ConfigurationClassPostProcessor.processConfigBeanDefinitions(ConfigurationClassPostProcessor.java:311)
    at org.springframework.context.annotation.ConfigurationClassPostProcessor.postProcessBeanDefinitionRegistry(ConfigurationClassPostProcessor.java:300)
    at org.springframework.context.support.PostProcessorRegistrationDelegate.invokeBeanDefinitionRegistryPostProcessors(PostProcessorRegistrationDelegate.java:105)
    at org.springframework.context.support.AbstractApplicationContext.invokeBeanFactoryPostProcessors(AbstractApplicationContext.java:694)
    at org.springframework.context.support.AbstractApplicationContext.refresh(AbstractApplicationContext.java:564)
    at org.springframework.context.annotation.AnnotationConfigApplicationContext.<init>(AnnotationConfigApplicationContext.java:145)
    at org.example.App.main(App.java:14)
```

Caused by ClassNotFoundError

```
Caused by: java.lang.ClassNotFoundException: Create breakpoint : org.springframework.core.NestedIOException <2 internal
    at java.base/java.lang.ClassLoader.loadClass(ClassLoader.java:576)
    ... 8 more
```

Why are we getting this error when the code is clean?

Because it has some mismatched versions of jar. Someone did some changes build.gradle / pom.xml

The screenshot shows the IntelliJ IDEA interface with the pom.xml file open. A tooltip appears over the dependency line for 'org.springframework:spring-core:6.1.6' with the text 'but the spring-beans is 5.3.30'. The dependency line is highlighted in blue.

```
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-core</artifactId>
    <version>6.1.6</version>
</dependency>
```

Clean and rebuild the project and run the project you will get our print statement

The screenshot shows the IntelliJ IDEA interface with the code being run in the terminal. The output shows the container created and the bean name printed.

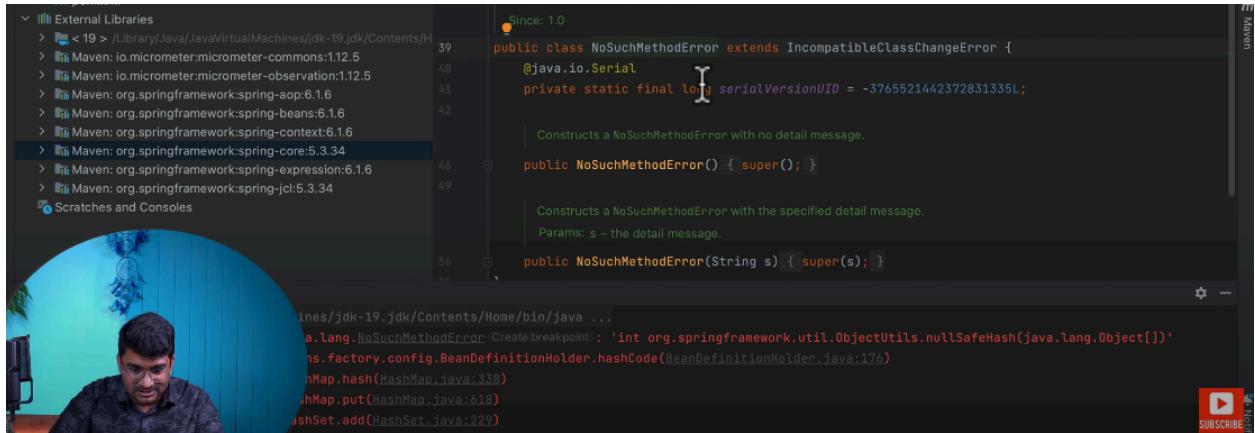
```
ApplicationContext app = new AnnotationConfigApplicationContext();
System.out.println("container created..");
System.out.println(app.getBean(Test.class));
```

Run: App
Process finished with exit code 0

The screenshot shows the IntelliJ IDEA interface with the pom.xml file open, displaying multiple versions of Spring dependencies.

```
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-core</artifactId>
    <version>6.1.6</version>
</dependency>
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>6.1.6</version>
</dependency>
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-core</artifactId>
    <version>5.3.34</version>
</dependency>
```

When we have different versions of spring jars we will get errors like NoSuchMethodError.



```
Since: 1.0
public class NoSuchMethodError extends IncompatibleClassChangeError {
    @java.io.Serial
    private static final long serialVersionUID = -3765521442372831335L;

    Constructs a NoSuchMethodError with no detail message.

    public NoSuchMethodError() { super(); }

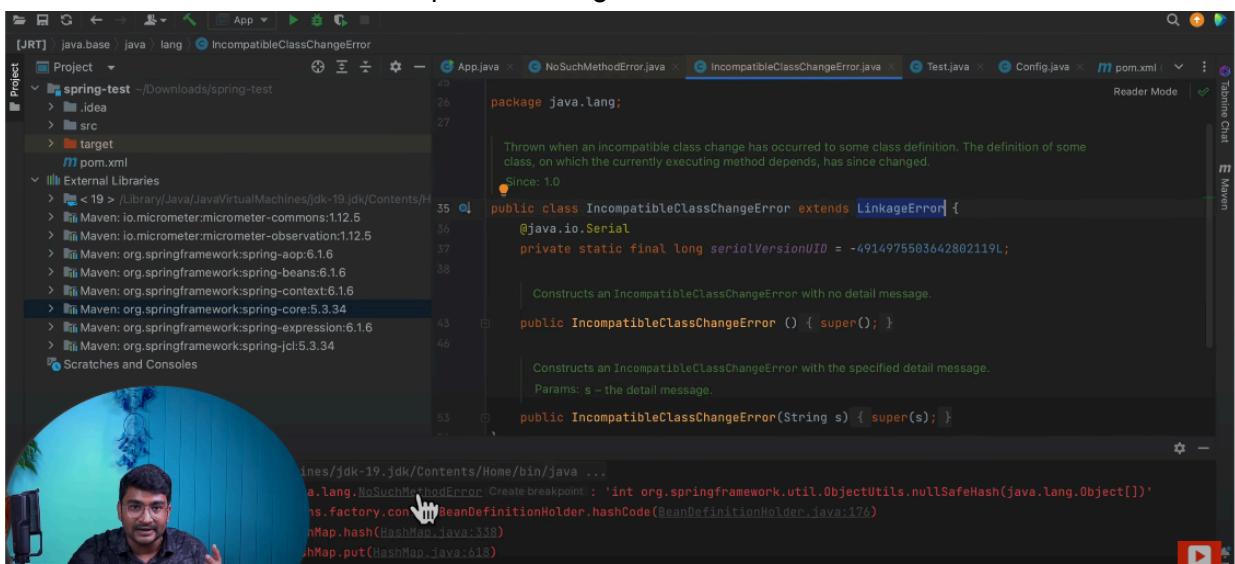
    Constructs a NoSuchMethodError with the specified detail message.

    Params: s – the detail message.

    public NoSuchMethodError(String s) { super(s); }

    ...
}
```

NoSuchMethodError will come as part of Linkage Error.



```
Thrown when an incompatible class change has occurred to some class definition. The definition of some class, on which the currently executing method depends, has since changed.

Since: 1.0
public class IncompatibleClassChangeError extends LinkageError {
    @java.io.Serial
    private static final long serialVersionUID = -4914975503642802119L;

    Constructs an IncompatibleClassChangeError with no detail message.

    public IncompatibleClassChangeError() { super(); }

    Constructs an IncompatibleClassChangeError with the specified detail message.

    Params: s – the detail message.

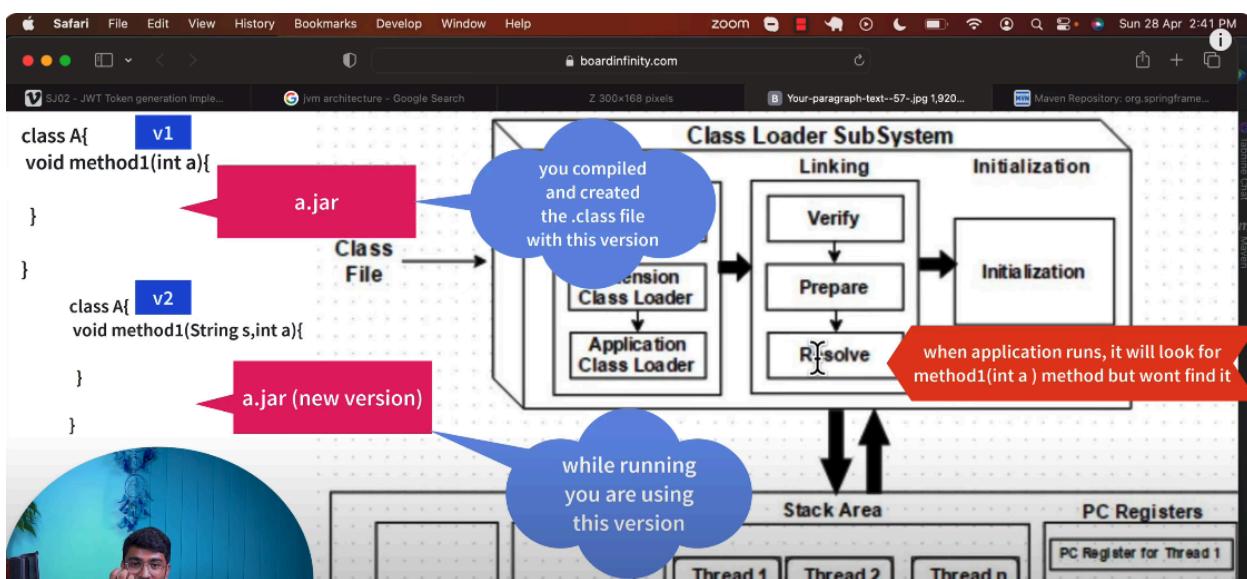
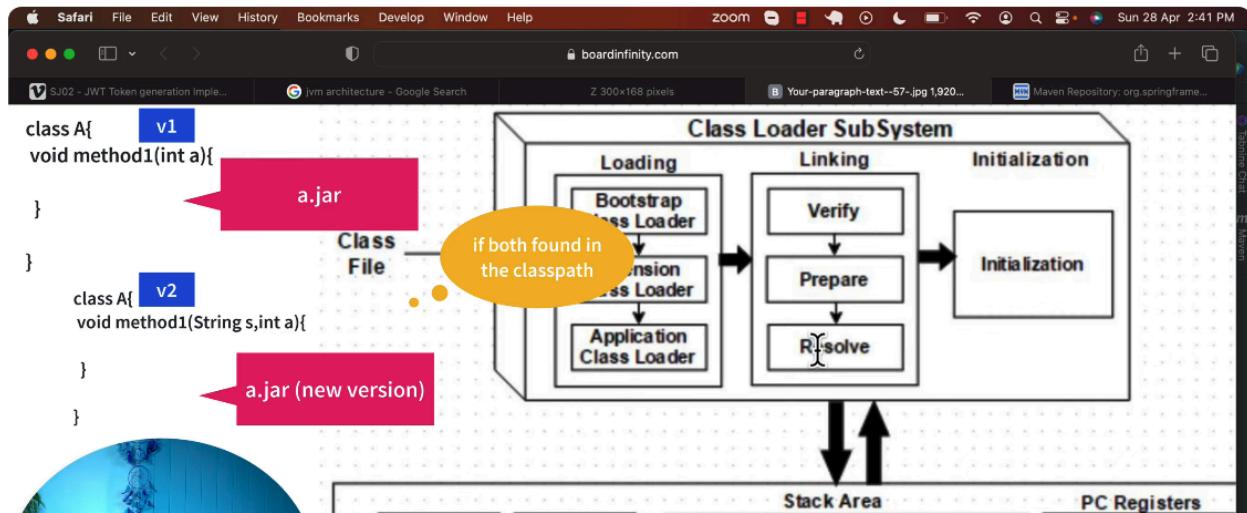
    public IncompatibleClassChangeError(String s) { super(s); }

    ...
}
```

Because we have two versions

During compile time it take one version and During Runtime it takes others version

Methods argument varies so in JVM during it is not able to Resolve in finding correct method



```

External Libraries
> < 19 > /Library/Java/JavaVirtualMachines/jdk-19.jdk/Contents/H 35 o
> Maven: io.micrometer:micrometer-commons:1.12.5 36
> Maven: io.micrometer:micrometer-observation:1.12.5 37
> Maven: org.springframework:spring-aop:6.1.6 38
> Maven: org.springframework:spring-beans:6.1.6
> Maven: org.springframework:spring-context:6.1.6
> Maven: org.springframework:spring-core:5.3.34 43
> Maven: org.springframework:spring-expression:6.1.6
> Maven: org.springframework:spring-jcl:5.3.34 46
Scatches and Consoles

```

```

public class IncompatibleClassChangeError extends LinkageError {
    @java.io.Serial
    private static final long serialVersionUID = -4914975503642802119L;
    Constructs an IncompatibleClassChangeError.
    public IncompatibleClassChangeError()
    Constructs an IncompatibleClassChangeError.
    Params: s - the detail message.
    public IncompatibleClassChangeError(String s)
}

```

we have compiled our code before with 6.x version and the .class file was generated. Now when we changed the version, same jar of different version(5.x) is available to our class path during runtime, where the below method signature could not be found.

The screenshot shows the IntelliJ IDEA interface with the code editor open. A tooltip is displayed over a section of the XML code, specifically the dependency declarations for Spring Core and Spring Context. The tooltip contains the following text:

Load Maven Changes

Maven project structure has been changed.
Load changes into IntelliJ IDEA to make it work correctly.

The code editor shows the following XML snippet:

```
<dependencies>
    <!-- https://mvnrepository.com/artifact/org.springframework/spring-core -->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-core</artifactId>
        <version>6.1.6</version>
    </dependency>
    <!-- https://mvnrepository.com/artifact/org.springframework/spring-context -->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-context</artifactId>
        <version>6.1.6</version>
    </dependency>
</dependencies>
```

Below the code editor, there is a blue banner with white text containing instructions for handling duplicate dependencies:

when you encounter this issue:
check for the duplicate dependencies.
maven clean to get rid of the old .class files.
create a fresh maven build to generate new .class file with modified pom

At the bottom right of the banner is a red YouTube-like button with the text "SUBSCRIBE".

Advantages of static class loading..

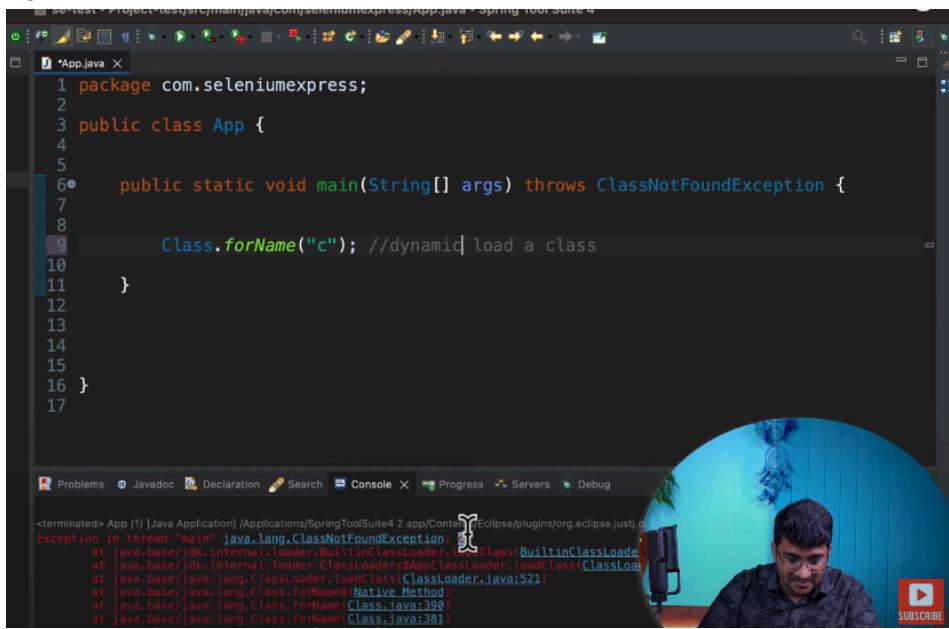
When ever we want to use same class object again and again
Static class loading happens during compile time. Whatever we discussed so far is static class loading.

Dynamic Class Loading When to use ?

Dynamic class loading happens due to a particular method

ClassNotFoundException happens during Dynamic class loading
ClassDefNotFoundException happens during Static class loading

ClassNotFoundException throws ClassNotFoundException so we have to add to method signature.



The screenshot shows the Spring Tool Suite 4 interface. In the top-left, there's a code editor window titled "App.java X" containing the following Java code:

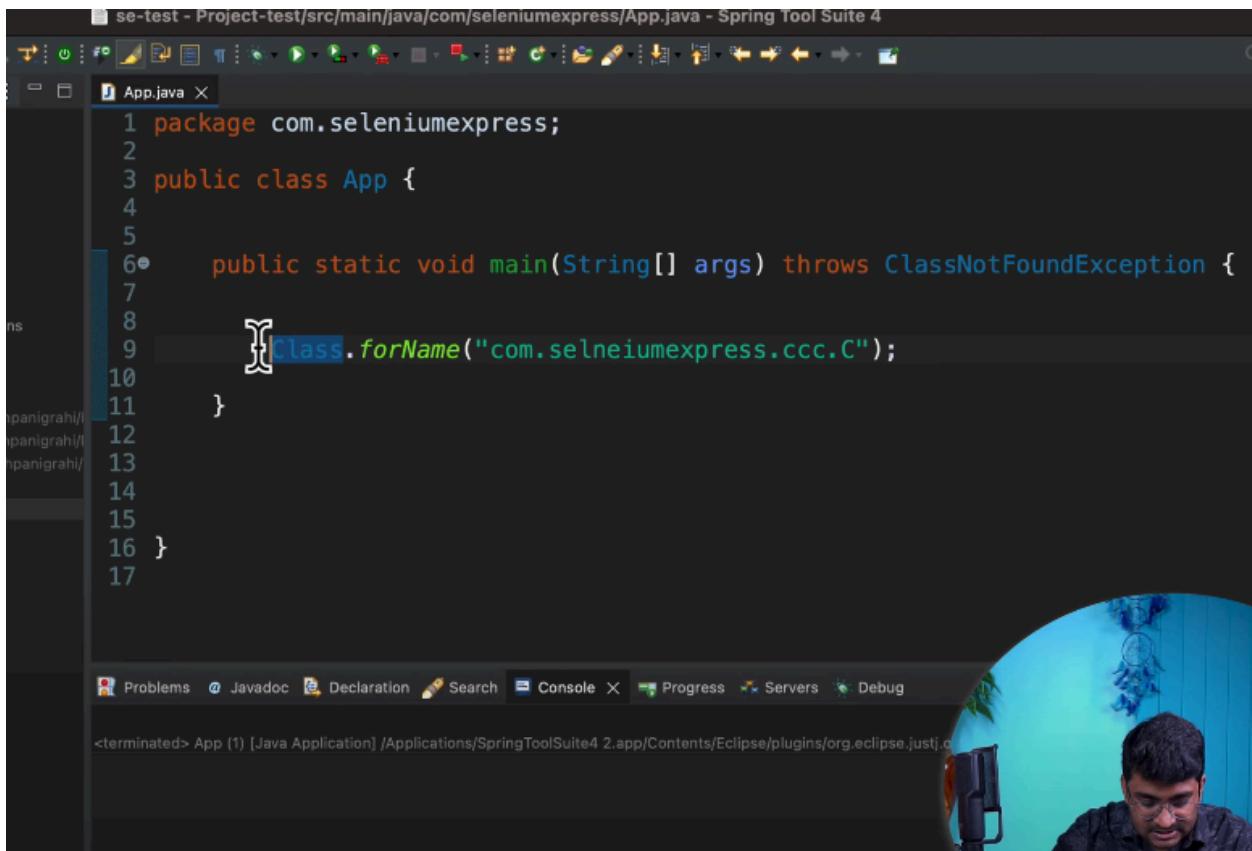
```
1 package com.seleniumexpress;
2
3 public class App {
4
5
6     public static void main(String[] args) throws ClassNotFoundException {
7
8         Class.forName("c"); //dynamic| load a class
9
10    }
11
12
13
14
15
16 }
17
```

In the bottom-right corner of the code editor, there is a circular video overlay featuring a person speaking. Below the video, there is a "SUBSCRIBE" button.

In the bottom-left, the "Console" tab is selected, showing the following exception output:

```
<terminated> App (1) [Java Application] /Applications/SpringToolSuite4 2.app/Contents/Eclipse/plugins/org.eclipse.justj.lexer/ast/api/Error.java:1: error: cannot find symbol
Exception in thread "main" java.lang.ClassNotFoundException: c
at java.base/jdk.internal.loader.BuiltinClassLoader.loadClass(BuiltinClassLoader.java:582)
at java.base/java.lang.ClassLoader.loadClass(ClassLoader.java:521)
at java.base/java.lang.Class.forName0(Native Method)
at java.base/java.lang.Class.forName(Class.java:398)
at java.base/java.lang.Class.forName(Class.java:381)
```

In order to avoid above exception we have to load complete path.



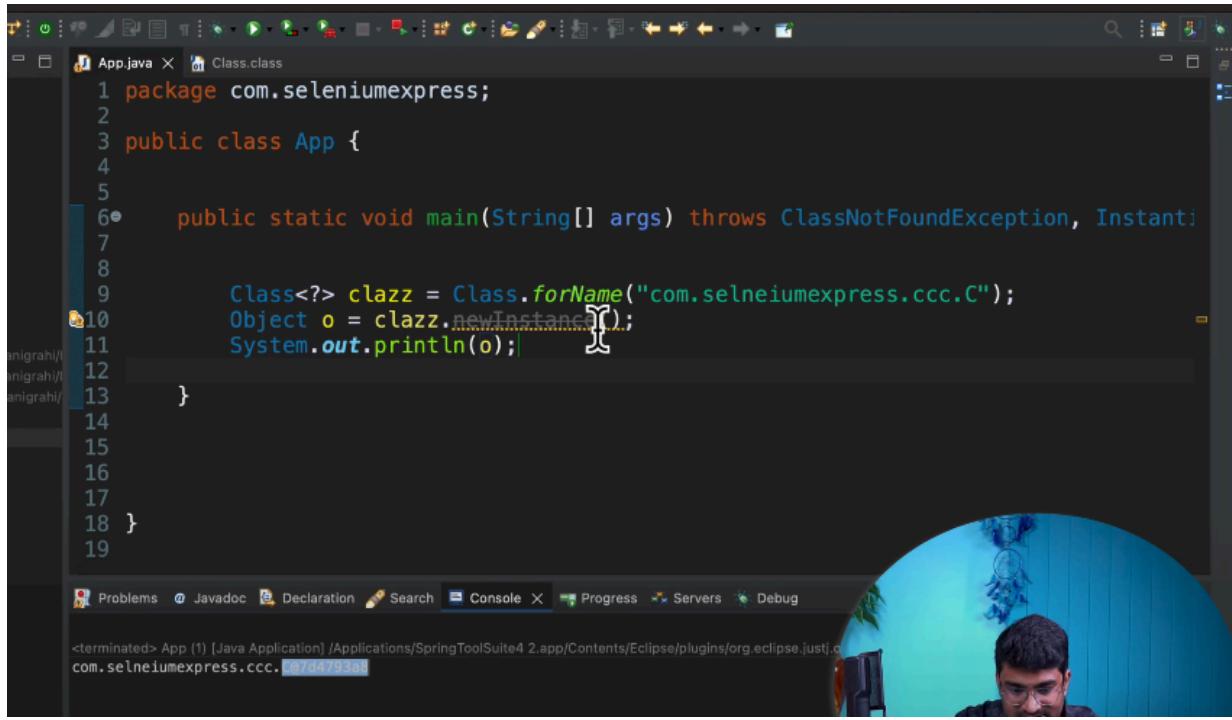
The screenshot shows the Spring Tool Suite 4 interface with the same code editor and video overlay as the previous screenshot. The code has been modified to include a fully qualified class name:

```
1 package com.seleniumexpress;
2
3 public class App {
4
5
6     public static void main(String[] args) throws ClassNotFoundException {
7
8         Class.forName("com.seleniumexpress.ccc.C");
9
10    }
11
12
13
14
15
16 }
17
```

The "Console" tab shows the output of the application running, which is now successful:

```
<terminated> App (1) [Java Application] /Applications/SpringToolSuite4 2.app/Contents/Eclipse/plugins/org.eclipse.justj.lexer/ast/api/Error.java:1: error: cannot find symbol
```

Another way of dynamic loading using class Class.



The screenshot shows the Eclipse IDE interface with two files open: App.java and Class.class. The App.java file contains the following code:

```
1 package com.seleniumexpress;
2
3 public class App {
4
5
6     public static void main(String[] args) throws ClassNotFoundException, InstantiationException, IllegalAccessException {
7
8         Class<?> clazz = Class.forName("com.seleniumexpress.ccc.C");
9         Object o = clazz.newInstance();
10        System.out.println(o);
11    }
12
13 }
14
15
16
17
18 }
19
```

The console tab at the bottom shows the output of the program:

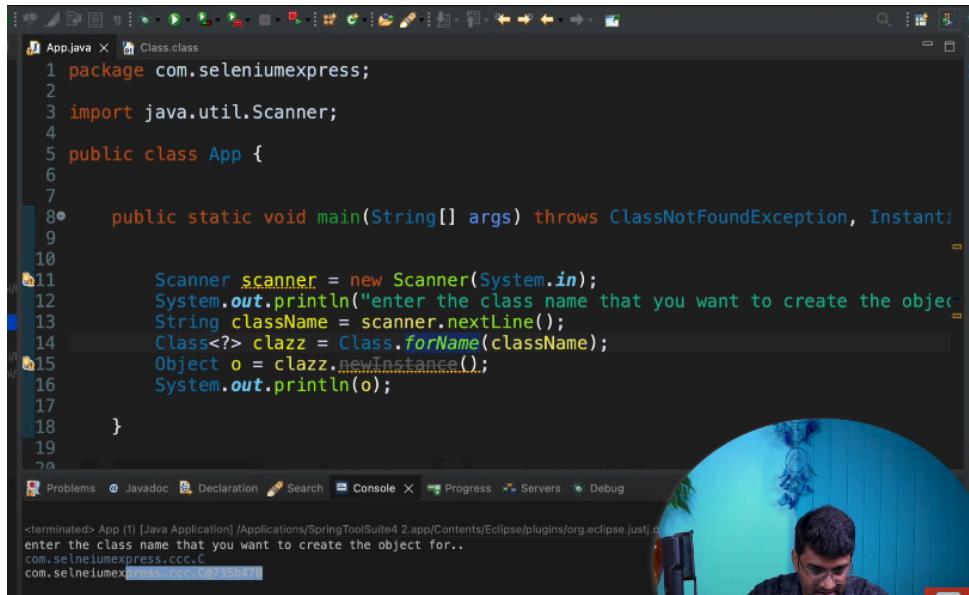
```
<terminated> App (1) [Java Application] /Applications/SpringToolSuite4 2.app/Contents/Eclipse/plugins/org.eclipse.justj.compiler/com.seleniumexpress.ccc.C@7d4791d
```

A circular inset image in the bottom right corner shows a person with glasses looking at the screen.

When it is useful

When we have a lot of classes. And we want to create object dynamically

Create C class object.



The screenshot shows the Eclipse IDE interface with two files open: App.java and Class.class. The App.java file contains the following code:

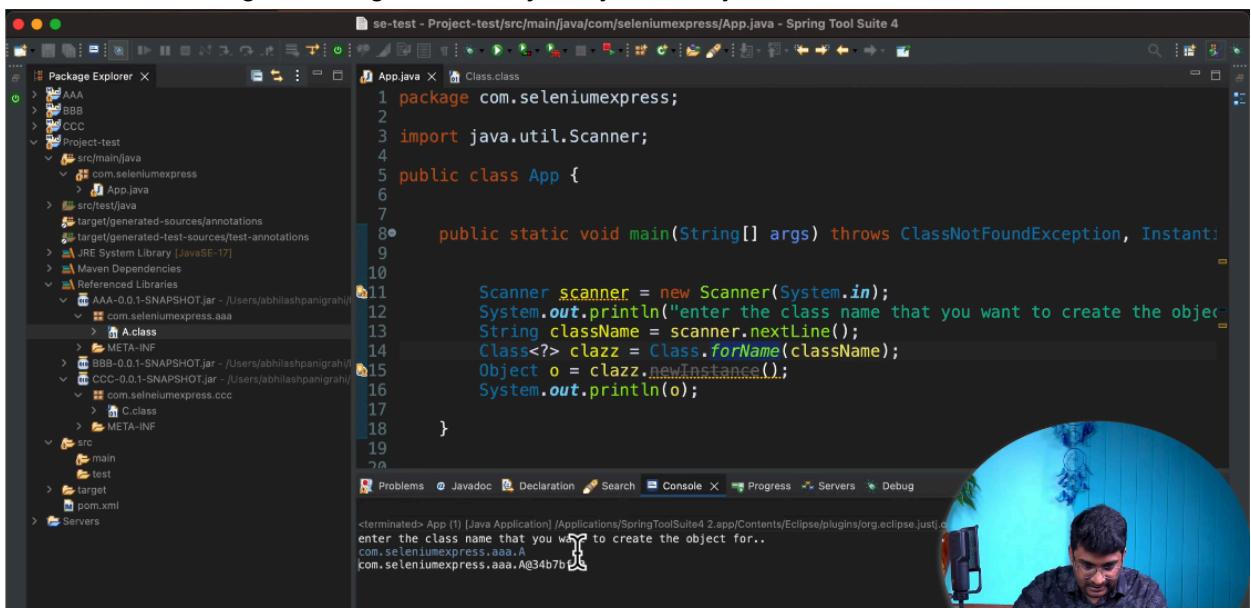
```
1 package com.seleniumexpress;
2
3 import java.util.Scanner;
4
5 public class App {
6
7
8     public static void main(String[] args) throws ClassNotFoundException, InstantiationException, IllegalAccessException {
9
10        Scanner scanner = new Scanner(System.in);
11        System.out.println("Enter the class name that you want to create the object for..");
12        String className = scanner.nextLine();
13        Class<?> clazz = Class.forName(className);
14        Object o = clazz.newInstance();
15        System.out.println(o);
16
17 }
18
19 }
```

The console tab at the bottom shows the output of the program, prompting the user for a class name:

```
<terminated> App (1) [Java Application] /Applications/SpringToolSuite4 2.app/Contents/Eclipse/plugins/org.eclipse.justj.compiler/com.seleniumexpress.ccc.C
Enter the class name that you want to create the object for..
com.seleniumexpress.ccc.C
com.seleniumexpress.ccc.C@735b478
```

A circular inset image in the bottom right corner shows a person with glasses looking at the screen.

Without code change creating A class object dynamically



The screenshot shows the Spring Tool Suite interface with a Java project named "se-test". The "App.java" file is open in the editor, containing the following code:

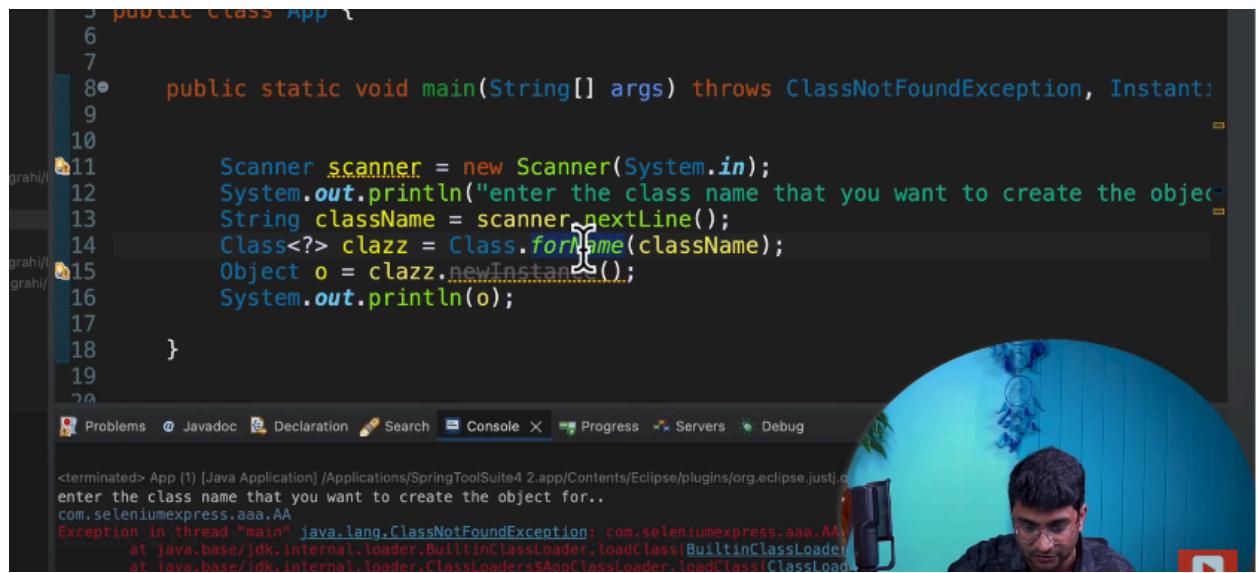
```
1 package com.seleniumexpress;
2
3 import java.util.Scanner;
4
5 public class App {
6
7
8    public static void main(String[] args) throws ClassNotFoundException, InstantiationException, IllegalAccessException {
9
10        Scanner scanner = new Scanner(System.in);
11        System.out.println("enter the class name that you want to create the object for..");
12        String className = scanner.nextLine();
13        Class<?> clazz = Class.forName(className);
14        Object o = clazz.newInstance();
15        System.out.println(o);
16
17    }
18
19
20}
```

The "Console" tab at the bottom shows the output of running the application:

```
<terminated> App (1) [Java Application] /Applications/SpringToolSuite4.app/Contents/Eclipse/plugins/org.eclipse.justj.console
enter the class name that you want to create the object for..
com.seleniumexpress.aaa.AA
Exception in thread "main" java.lang.ClassNotFoundException: com.seleniumexpress.aaa.AA
at java.base/jdk.internal.loader.BuiltinClassLoader.loadClass(BuiltinClassLoader.java:583)
at java.base/jdk.internal.loader.ClassLoaders$AppClassLoader.loadClass(ClassLoaders.java:178)
at java.base/java.lang.ClassLoader.loadClass(ClassLoader.java:519)
```



Whenever we enter wrong input - it will give ClassNotFoundException. (this exception comes only during dynamic loading of class)



The screenshot shows the Spring Tool Suite interface with the same Java code as before. The "Console" tab at the bottom shows the output of running the application, but this time it highlights the line where the exception occurs:

```
<terminated> App (1) [Java Application] /Applications/SpringToolSuite4.app/Contents/Eclipse/plugins/org.eclipse.justj.console
enter the class name that you want to create the object for..
com.seleniumexpress.aaa.AA
Exception in thread "main" java.lang.ClassNotFoundException: com.seleniumexpress.aaa.AA
at java.base/jdk.internal.loader.BuiltinClassLoader.loadClass(BuiltinClassLoader.java:583)
at java.base/jdk.internal.loader.ClassLoaders$AppClassLoader.loadClass(ClassLoaders.java:178)
at java.base/java.lang.ClassLoader.loadClass(ClassLoader.java:519)
```



The newInstance method is deprecated. The below one is a new way / method.

```
13     System.out.println("enter the class name that you want to cre
14     String className = scanner.nextLine();
15     Class<?> clazz = Class.forName(className);
16     Object o = clazz.getDeclaredConstructor().newInstance();
17     System.out.println(o);
18 }
19
20
21
22
```

Problems Javadoc Declaration Search Console <terminated> App (1) [Java Application] /Applications/SpringToolSuite4 2.app/Contents/Eclipse/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx-x86_64/enter the class name that you want to create the object for.. com.seleniumexpress.App |com.seleniumexpress.App@4411d978

This Class.forName in real time example could be if want to download DB driver which can JDBC Driver/SQL Drive/Portugues Driver/noSQL driver/etc