Normal program execution flow.

Note:

1. new will create a memory on the heap area
2. Demo (a class name ) => jvm will search for the Demo.class file in the current working directory

If found load the .class file data into the method area

1. During the loading of the .class file
2. static variables will get memory set with default value
3. static block gets executed
4. in the heap area , for the required object , memory for instance variables is given, jvm will set the default values to it
5. execute the instance block if available
6. call the constructor to set the meaningful values to the instance variables

jvm will give the address of the object to the hashing algorithm , which generates the hash code for the object and that hash code will be returned as the reference to the programmer .

new vs newInstance() :

1. new is an operator to create objects , if we know class name at the beginning then we can create the object by using new keyword.
2. newInstance() is a method presenting class “ Class “ , which can be used to create the object
3. if we don’t know the class name at the beginning and it is available dynamically at runtime then we should go for newInstance()

Eg: newInstance // used method chaining

Eg: newInstance\_Eg2

// go through the code

If dynamically loaded class is not available then we will get a runtime exception ClassNotFouondException

To use newinstance() method compulsory corresponding class should contain no argument constructor, otherwise we will get runtime exception saying “InstantiationException”

If the argument constructor is private then it would result in the “IllegalAcessException”.

Note:

During typecasting if there is no relationship between 2 classes , then it would result in the “ClassCastException”

Difference between new and newInstance():

new :

new is an operator which can be used to crate the object

we can use new operator if we know the class name at the beginning

Test t = new Test();

If the corresponding class is not found at the runtime then we will get a runtime exception saying NoClassDefFoundError, it is unchecked .

To use the new operator the corresponding class is not required to contain no argument constructor

newInstance() :

newInstance() is a method, present in class Class , which can be used to create an object

we can use the newInstance() method , if we don’t know the class name at the beginning and available dynamically at runtime

Object o = Class.forName(args[0]).newInstance();

It the corresponding class is not available at the runtime , then it will lead to ClassNotFoundException , it is checked .

To use newInstance() the corresponding class should compulsory contain no argument constructor, Otherwise we will get a runtime exception called InstantiationException .

Difference between ClassNotFoundException & NoClassDefFoundError :

1. For hard coded class names ( .class file names can’t be changed so they are called as hard coded) at runtime in the corresponding .class files , if it is not available we will get NoClassDefFoundError , which is unchecked

Eg: Demo t = new Demo();

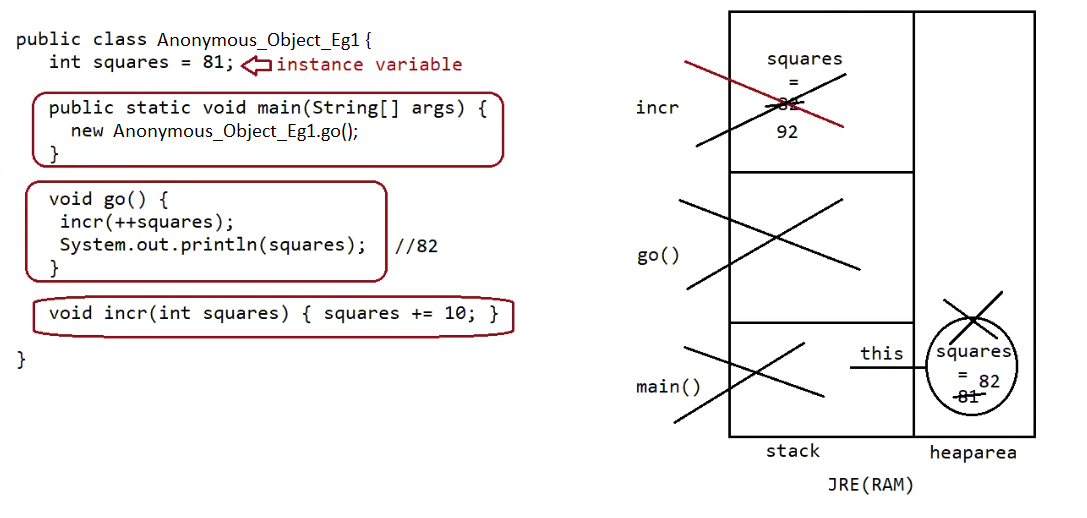
In runtime Test.class file is not available then we will get NoClassDefFoundError

1. For dynamically provided class names at runtime , if the corresponding .class file is not available , then we will get a runtime exception saying “ClassNotFounfException”

Eg: Object obj = Class.forName(“Demo”).newInstance();

At runtime if Demo.class file is not available then we will get an exception called ClassNotFoundException , it is checked .

Eg: Anonymous\_Object\_Eg1



Here first main() will be loaded in the stack frame , and squares is an instance variables , its memory is allocated during object creation .

object for squares is created but it is not collected by reference , so jvm uses implicit reference “this” to hold its address .

since go() is called a new stack frame for go() will be created .

go() is a method , which is a method called with respect to object and go() is present in same class , and squares is a instance variable, so it can be accessed directly in the instance method , so squares is incremented to 82.

And incr() is called from go() method . a new stack frame will be created and incr() is executed.

After the execution of incr() its stack frame will be deleted.

Now go() will be executed , after the completion of its execution its stack frame will be deleted.

Similarly after the completion of main() execution its stack frame will be deleted.

Now since there is no reference to store the object , garbage collector will clear it.