Java Reflection API

**Java Reflection** is a *process of examining or modifying the run time behavior of a class at run time*.

The **java.lang.Class** class provides many methods that can be used to get metadata, examine and change the run time behavior of a class.

The java.lang and java.lang.reflect packages provide classes for java reflection.

Where it is used

The Reflection API is mainly used in:

* IDE (Integrated Development Environment) e.g. Eclipse, MyEclipse, NetBeans etc.
* Debugger
* Test Tools etc.

Do You Know ?

* How many ways we can get the instance of Class class ?
* How to create the javap tool ?
* How to create the appletviewer tool ?
* How to access the private method from outside the class ?

# newInstance() method

The **newInstance()** method of **Class** class and **Constructor** class is used to create a new instance of the class.

The newInstance() method of Class class can invoke zero-argument constructor whereas newInstance() method of Constructor class can invoke any number of arguments. So Constructor class is preferred over Class class.

#### **Syntax of newInstance() method of Class class**

**public T newInstance()throws InstantiationException,IllegalAccessException**

Here T is the generic version. You can think it like Object class. You will learn about generics later.

### Example of newInstance() method

Let's see the simple example to use newInstance() method.

**class** Simple{

**void** message(){System.out.println("Hello Java");}

}

**class** Test{

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

**try**{

 Class c=Class.forName("Simple");

 Simple s=(Simple)c.newInstance();

 s.message();

 }**catch**(Exception e){System.out.println(e);

}  }  }

Output:Hello java