* What is reflection in java?

Reflection is an API which is used to examine or modify the behavior of methods, classes, interfaces at runtime.

* The required classes for reflection are provided under **java.lang.reflect package.**
* Reflection gives us information about the class to which an object belongs and also the methods of that class which can be executed by using the object.
* Through reflection we can invoke methods at runtime irrespective of the access specifier used with them.

Reflection can be used to get information about –

* Class The getClass() method is used to get the name of the class to which an object belongs.
* Constructors The getConstructors() method is used to get the public constructors of the class to which an object belongs.
* Methods The getMethods() method is used to get the public methods of the class to which an objects belongs.
* **import** java.lang.reflect.Method;
* **import** java.lang.reflect.Field;
* **import** java.lang.reflect.Constructor;
* // class whose object is to be created
* **class** Test
* {
* // creating a private field
* **private** String s;
* // creating a public constructor
* **public** Test()  {  s = "GeeksforGeeks"; }
* // Creating a public method with no arguments
* **public** **void** method1()  {
* System.out.println("The string is " + s);
* }
* // Creating a public method with int as argument
* **public** **void** method2(**int** n)  {
* System.out.println("The number is " + n);
* }
* // creating a private method
* **private** **void** method3() {
* System.out.println("Private method invoked");
* }
* }

1. **class** Demo
2. {
3. **public** **static** **void** main(String args[]) **throws** Exception
4. {
5. // Creating object whose property is to be checked
6. Test obj = **new** Test();
8. // Creating class object from the object using
9. // getclass method
10. Class cls = obj.getClass();
11. System.out.println("The name of class is " +
12. cls.getName());
14. // Getting the constructor of the class through the
15. // object of the class
16. Constructor constructor = cls.getConstructor();
17. System.out.println("The name of constructor is " +
18. constructor.getName());
20. System.out.println("The public methods of class are : ");
22. // Getting methods of the class through the object
23. // of the class by using getMethods
24. Method[] methods = cls.getMethods();
26. // Printing method names
27. **for** (Method method:methods)
28. System.out.println(method.getName());
30. // creates object of desired method by providing the
31. // method name and parameter class as arguments to
32. // the getDeclaredMethod
33. Method methodcall1 = cls.getDeclaredMethod("method2",
34. **int**.**class**);
36. // invokes the method at runtime
37. methodcall1.invoke(obj, 19);
39. // creates object of the desired field by providing
40. // the name of field as argument to the
41. // getDeclaredField method
42. Field field = cls.getDeclaredField("s");
44. // allows the object to access the field irrespective
45. // of the access specifier used with the field
46. field.setAccessible(**true**);
48. // takes object and the new value to be assigned
49. // to the field as arguments
50. field.set(obj, "JAVA");
52. // Creates object of desired method by providing the
53. // method name as argument to the getDeclaredMethod
54. Method methodcall2 = cls.getDeclaredMethod("method1");
56. // invokes the method at runtime
57. methodcall2.invoke(obj);
59. // Creates object of the desired method by providing
60. // the name of method as argument to the
61. // getDeclaredMethod method
62. Method methodcall3 = cls.getDeclaredMethod("method3");
64. // allows the object to access the method irrespective
65. // of the access specifier used with the method
66. methodcall3.setAccessible(**true**);
68. // invokes the method at runtime
69. methodcall3.invoke(obj);
70. }
71. }

Output:

1. The name of **class** is Test
2. The name of constructor is Test
3. The **public** methods of **class** are :
4. method2
5. method1
6. wait
7. wait
8. wait
9. equals
10. toString
11. hashCode
12. getClass
13. notify
14. notifyAll
15. The number is 19
16. The string is JAVA
17. Private method invoked