**Interface:**

Using Interface, We can create a contract or we can say set of rules for behavior of application. Interface is looks like class but It Is not class. When you implements that Interface In any class then all those Interface rules must be applied on that class.

It is a set of rules defined by us which has only declaration of functions, no body of the function and it will force us to override all the functions in the class which implements this interface.

Interface can be implemented with any class using *implements* keyword. There are set of rules to be followed for creating an Interface.

* Interface cannot hold instance fields/variables.
* Interface cannot hold static methods.
* You cannot instantiate/create object of an Interface.
* Any class can Implement Interface but cannot extend Interface.
* Can write body less methods Inside Interface.
* By default all the methods and variables of Interface are public so no need to provide access modifiers.

Syntax: [modifier] **interface** InterfaceName {

[Modifier] [Return type] methodName1();

[Modifier] [Return type] methodName2();

….. }

* **Syntax** : interface InterfaceName =new ClassName()
* **E.g**.: WebDriver wd= new FireFoxDriver();
* By declaring like this one can access **all the WebDriver’s stuff** as well as **only overridden stuff from** ClassName **FireFoxDriver**

**Note:**

* One can implement one or more interfaces at a time**.**
* We can’t create an object for an interface.
* We can create an object for a class which is implementing the interface and then give the reference of the interface.