*What is an Interface?*  
Interface is like a blueprint of Class. It contains variables and body less methods(Abstract methods), where we just declare methods but we implement them inside the class which inherit Interface.  
  
*Note:-***Java 8** introduced “*Default Method*” or (Defender methods), which allows us to add new methods with*declaration and definition (like a normal method)*to the Interfaces without breaking the existing implementation of these Interface.  
  
***Example:-***

Public Interface WebDriver()

{

int iNum = 10;

public void get(String URL);

}

Above example just contain the structure of a *get* method, now a class which would implement this interface need to define the method.  
  
*Note:-* By default all the variables and methods inside an Interface are public-ally accessible.   
  
*Example:-*

Public class FirefoxDriver implements WebDriver()

{

public void get(String sURL)

{

System.out.println("URL="+sURL);

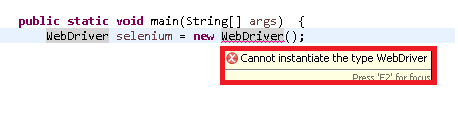
}

}

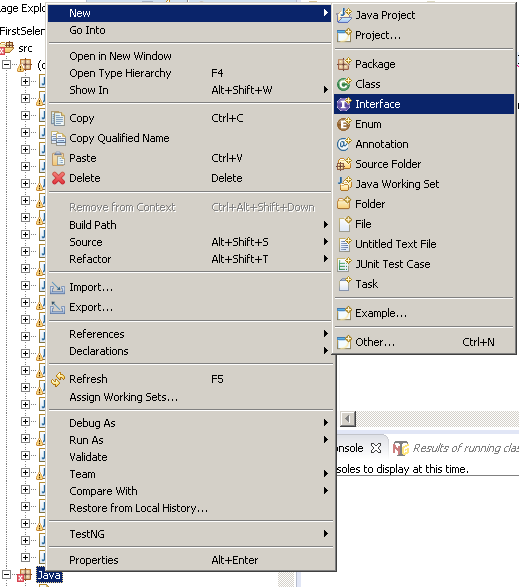
Now we know that incase of Interface, "*If we define a reference variable whose type is an interface,any object we assign to it must be an instance of a class that implements the interface*."  
  
In our example that Class is FireFoxDriver and Interface is WebDriver, so we can say *WebDriver driver = New FirefoxDriver();*

|  |
| --- |
| [WebDriver interface implements Firefoxdriver class](http://1.bp.blogspot.com/-lV8KF49n6Wo/VL5mH7IHu2I/AAAAAAAABgw/q5FBJOhTM_g/s1600/WebDriver%2Binterface%2Bimplemented%2Bin%2BFirefoxDriver%2BClass.png) |
|  |

*Note:-*We need to use the Implements Keyword to consume Interface inside a Class.  
  
*Lets learn more about an Interface*  
  
*Interface instance:-*  
We can create a reference variable of an interface but we can't instantiate any interface since it is just a contract to be implemented in a Class.  
  
WebDriver driver = New WebDriver ...Not allowed,

[](http://2.bp.blogspot.com/-LAo76rHE2e4/VL5nLMqNYqI/AAAAAAAABg8/0YF7v9ZXv8c/s1600/Cannot%2Bcreate%2Ba%2Bobject%2Bof%2BInterface.png)

*but below code is allowed....* ***Example:-***  
*WebDriver driver = New FireFoxDriver();*  
*driver.get(testUrl);*  
  
***Creating Interface in Eclipse:-***  
Lets create a sample interface in Eclipse and implement it inside a Class.  
Right click on package and go to -> New -> Interface.

[](http://2.bp.blogspot.com/-jf_elt7gYwM/VL_Vh-mS7XI/AAAAAAAABhs/_zHU80PD494/s1600/Creatint%2BInterface%2Bin%2BEclipse.png)

*Important points:-*

* 1. Class that implements an interface must implement all the methods declared in the interface.Now our FirefoxDriver class should implement all the methods declared inside an WebDriver interface, same is the case with ChromeDriver or IEDriver classes.
  2. While implementing methods, we must follow the *exact same signature (name + parameters)* as declared in the interface.
  3. We *can not instantiate/create object of an Interface*. webdriver driver = new webdriver ();
  4. All the variable inside an interface are by default [Final](http://www.ufthelp.com/2015/01/Final-in-Java-Selenium.html).
  5. Class cannot Extend Interface only Implements it.
  6. Interface an Extend another Interface but then the class which implements the interface need to implemented the methods of both interface .
  7. *Class can implement multiple Interface*(*Remember class cannot extend multiple classes,multiple inheritance in class is not possible*)

|  |
| --- |
| [Class inherit from multiple Interface](http://3.bp.blogspot.com/-AgZEqUwcH-g/VL6H5bU4FYI/AAAAAAAABhM/9z6x1Tvk2hU/s1600/Multiple%2Binheritence%2Bin%2BInterface.png) |
|  |

* 1. Interface can not hold constructor.
  2. Interface can not hold instance fields/variables.
  3. By default all the methods of Interface are public so no need to provide access modifiers.
  4. An interface can have another interface i.e. known as nested interface.

***Abstract Class v/s Interface***  
Abstract class is similar to interface, like we cannot instantiate them, and they may contain a mix of methods declared with or without an implementation.But in abstract class, we can declare fields that are not static and final, and define public, protected, and private concrete methods. With interfaces, all fields are automatically public, static, and final, and all methods that are  declared or defined (as default methods) are public. We can implement any number of interfaces unlike abstract class which can extend only one class.  
  
  
*Note:-* We could have implemented WebDriver driver = New FirefoxDriver(); as FirefoxDriver driver = new Firefoxdriver();  
  
But we would face problem incase we need to run the same code for*chromedriver* then we need to create the object as ChromeDriver driver = new ChromeDriver();.  
  
Thus to avoid this we are creating reference of WebDriver and assigning the object of implementing class i.e webdriver driver; driver = new firefoxdriver or driver = new chromedriver();