Why Do We Up cast Browser Driver Class Object To WebDriver?

Hello Folks,

In this post, we will learn about a frequently asked interview question in selenium that “Why we upcast a browser driver class object to WebDriver type?”.

OR,

Why do we write as below:

WebDriver driver= new FirefoxDriver();
WebDriver driver= new ChromeDriver ();

Why not as below:

FirefoxDriver driver= new FirefoxDriver();
ChromeDriver driver= new ChromeDriver ();

Let’s discuss about this.

First of all, all above statements are correct. It is **not mandatory** to up cast a browser driver class object to WebDriver or any higher class/interface in hierarchy.

Now question is if it is not mandatory, why we do so?

We should be aware hierarchy of Selenium webdriver classes and interfaces. You can learn about it from here.

WebDriver Interface is super interface (Not super most) of every browser driver classes like **ChromeDriver , FirefoxDriver** etc. And as per java concept, a super class/interface reference can hold a its sub class object but vice versa is not possible.

We will see two approaches here:

1. When we do not upcast to WebDriver

2. When we upcast to WebDriver

When we do not upcast to WebDriver:

Suppose, you need to create a base class which need to be extended in every class and perform some actions. User should be able to launch any browser.

You will create a base class as below:

```
package UpCast;

import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.edge.EdgeDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.ie.InternetExplorerDriver;
import org.openqa.selenium.safari.SafariDriver;

public class BaseClassWithoutUpcasting {

        // Since user should be able to launch any browser , we need to create reference object
        public static ChromeDriver cDriver;
        public static FirefoxDriver fDriver;
        public static InternetExplorerDriver iDriver;
        public static EdgeDriver eDriver;
        public static SafariDriver sDriver;

        // We need to keep adding browser driver reference variable
}
```

gistfile1.txt hosted with ❤ by GitHub view raw

You will create test scripts as below:

```
package UpCast;

import java.io.File;
import java.io.IOException;

import org.apache.commons.io.FileUtils;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.chrome.ChromeDriver;

public class ChromeBrowser extends BaseClassWithoutUpcasting
{
```

```
        public static void main(String[] args) throws IOException {

                // Since you need to launch chrome browser , you need to use ChromeBrowser refe
                System.setProperty("webdriver.chrome.driver","./exefiles/chromedriver.exe");
                cDriver= new ChromeDriver();
                cDriver.get("http://makeseleniumeasy.com/");

                // Taking screenshot
                // To take screenshot no need to downcast to TakesScreenshot interface

                File screenshotSRC= cDriver.getScreenshotAs(OutputType.FILE);
                // Defining path and extension of image
                String path=System.getProperty("user.dir")+"/ScreenCapturesPNG/"+System.current
                // copying file from temp folder to desired location
                File screenshotDest= new File(path);
                FileUtils.copyFile(screenshotSRC, screenshotDest);

                // Running javascript command
                // No need to downcast to JavascriptExecutor
                cDriver.executeScript("window.scrollBy(0,250)");

                // Closing browser
                cDriver.quit();

        }
    }
```

gistfile1.txt hosted with ❤ by GitHub view raw

Disadvantages:

- In the same style you need to create other browser driver scripts and use proper reference variable of browser type.
- In this approach, if a new browser comes in market, you need to change Base class because we need to define a new reference variable of type new browser class.
- Modification is not easy because it require changes at framework level.
- You can not use methods defined in above classes in hierarchy.
- It will problem to maintain a same reference of browser driver. You might get null pointer exception.

When we upcast to WebDriver:

We will create a very simple base class as below:

```
package UpCast;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.edge.EdgeDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.ie.InternetExplorerDriver;
import org.openqa.selenium.safari.SafariDriver;

public class BaseClassWithUpcasting {

        // Just we need to create a single reference variable of type WebDriver.It can hold all
        // indirectly implements WebDriver interface.
        public static WebDriver _driver;

}
```

gistfile1.txt hosted with ❤ by GitHub view raw

We will create scripts as below:

```
package UpCast;

import java.io.File;
import java.io.IOException;

import org.apache.commons.io.FileUtils;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.chrome.ChromeDriver;

public class ChromeBrowserWithUpcast extends BaseClassWithUpcasting
{

        public static void main(String[] args) throws IOException {

                // Whatever browser you want to launch, just creata an object of that particula
                System.setProperty("webdriver.chrome.driver","./exefiles/chromedriver.exe");
                _driver= new ChromeDriver();
                _driver.get("http://makeseleniumeasy.com/");

                // Taking screenshot
                // To take screenshot we need to downcast to TakesScreenshot interface
                // down casting WebDriver to TakesScreenshot to use getScreenshotAs method.
```

```
TakesScreenshot ts= (TakesScreenshot)_driver;
File screenshotSRC= ts.getScreenshotAs(OutputType.FILE);
// Defining path and extension of image
String path=System.getProperty("user.dir")+"/ScreenCapturesPNG/"+System.current
// copying file from temp folder to desired location
File screenshotDest= new File(path);
FileUtils.copyFile(screenshotSRC, screenshotDest);

// Running javascript command
// Need to downcast to JavascriptExecutor
JavascriptExecutor jse = (JavascriptExecutor)_driver;
jse.executeScript("window.scrollBy(0,250)");

// Closing browser
_driver.quit();
```

gistfile1.txt hosted with ❤ by GitHub view raw

In this approach, you can use same reference variable of type WebDriver to launch any browser.

Advatnages:

- If you compare both approach, you will understand that we upcast for better architecture , best coding practice, ease of modification and enhancements. There is no hard core rule that we need to do it.
- We can launch any browser using same reference type.
- If new browser comes in market, there is no need to do any changes in framework (Base class).
- You can use methods of classes in hierarchy which requires downcasting. For ex: To take screenshot and execute java script.
- We achieve run time polymorphism here.