Selenium – Day 2

November 2016

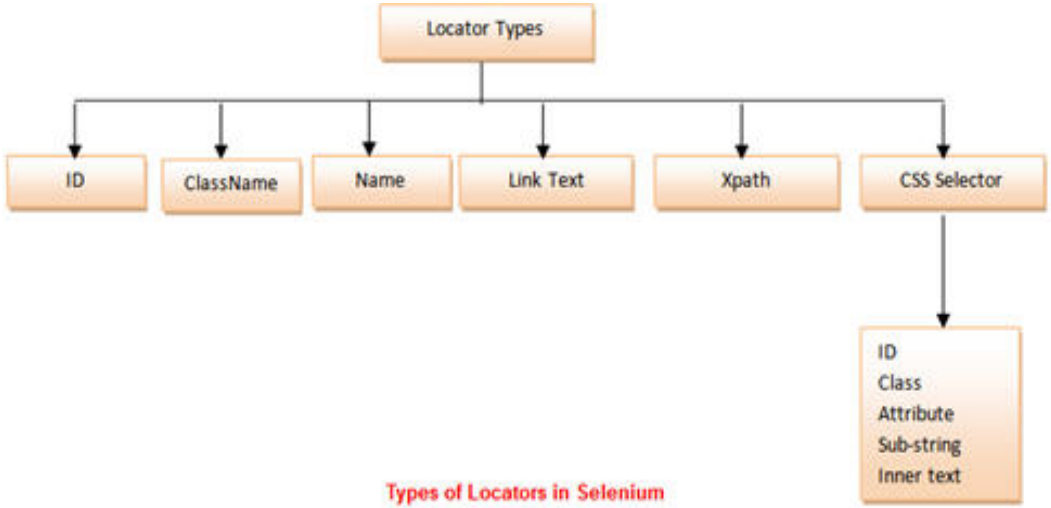
Learning Objectives– Day 2

- DAY 1 - Recap
- Web Elements – Locating & Performing Actions
- Locators in Detail – ID, Name, xpath, CSS
- Browser Plugins for Object Identification
- Chrome Developer Tools
- IE Developer Tools
- Exercise : Identify Web Elements, Simple Actions, Get Element Properties, Web Elements using different Locators

Object Identification | Locators

- Object identification is a very crucial part of any automation effort in selenium or any other tool for that matter. So it is very important to be able to identify the objects used in the automation.
- Following ways can be used for object identification

  - ID
  - Name
  - classname
  - Xpath – Relative
  - Xpath – Absolute
  - CSS Selector

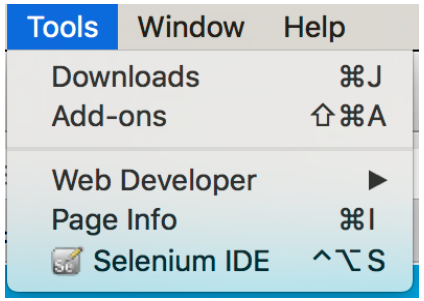


Firefox – Selenium IDE Plugin

Selenium IDE is an integrated development environment for Selenium tests. It is implemented as a Firefox extension, and allows you to record, edit, and debug tests.

Download selenium IDE - https://addons.mozilla.org/en-US/firefox/addon/selenium-ide/

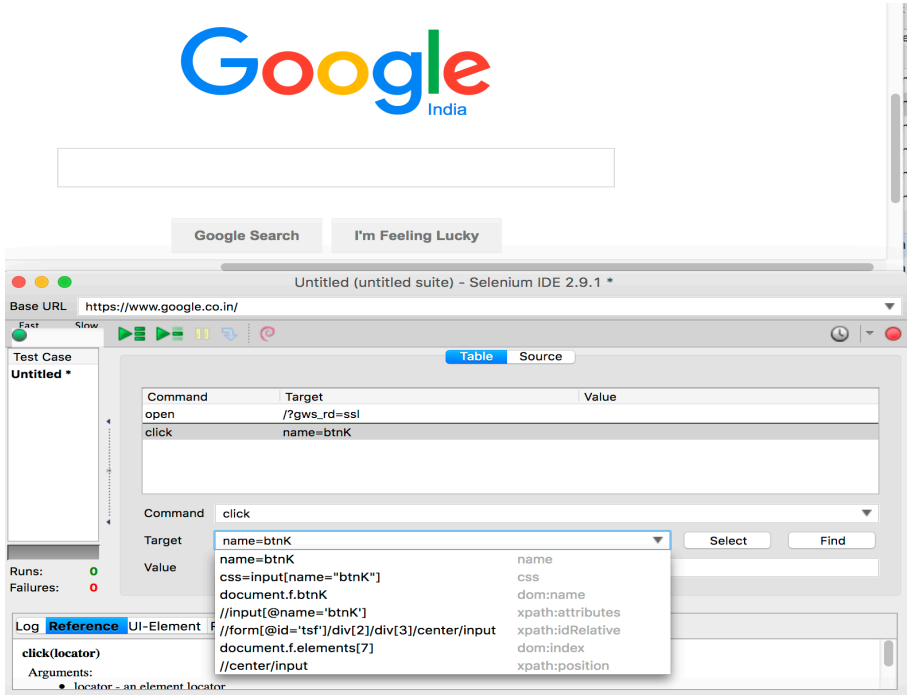
Step1: Open selenium IDE



Step 2:

Click on record button

Step3: Click on Google search button (element locator is captured in the target field)



Firefox – Firebug Plugin

Firebug integrates with Firefox to put a wealth of development tools at your fingertips while you browse. You can edit, debug, and monitor CSS, HTML, and JavaScript live in any web page

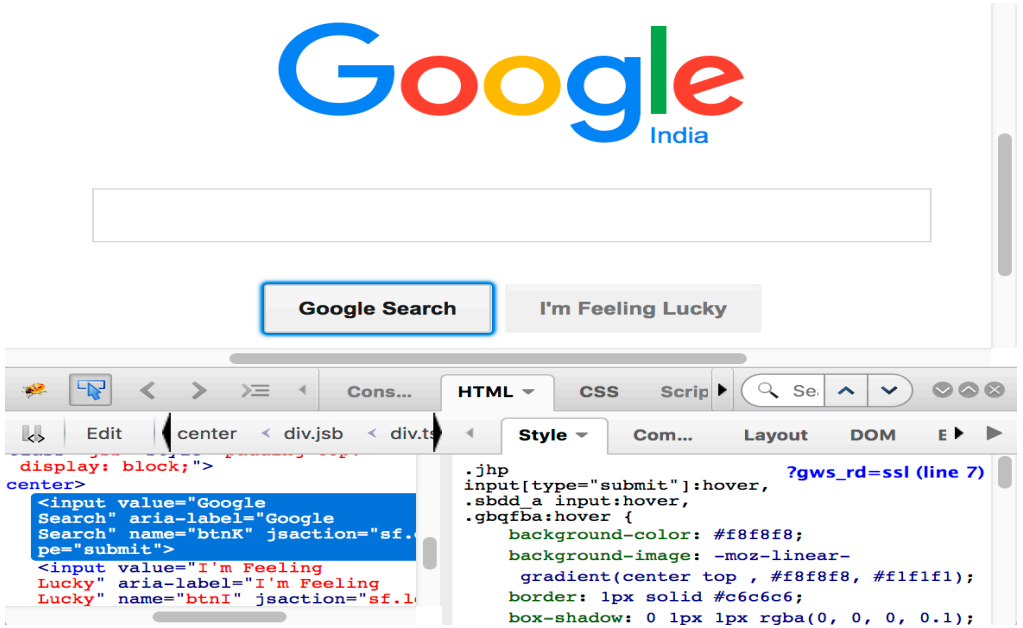
Download selenium IDE - https://addons.mozilla.org/en-US/firefox/addon/firebug/

Step1: Activate firebug info.

Step 2:

Click on pointer icon

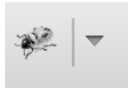
Step3: Select an element and firebug shows the locator



Firefox – FirePath Plugin

FirePath is a Firebug extension that adds a development tool to edit, inspect and generate XPath 1.0 expressions, CSS 3 selectors and JQuery selectors (Sizzle selector engine).

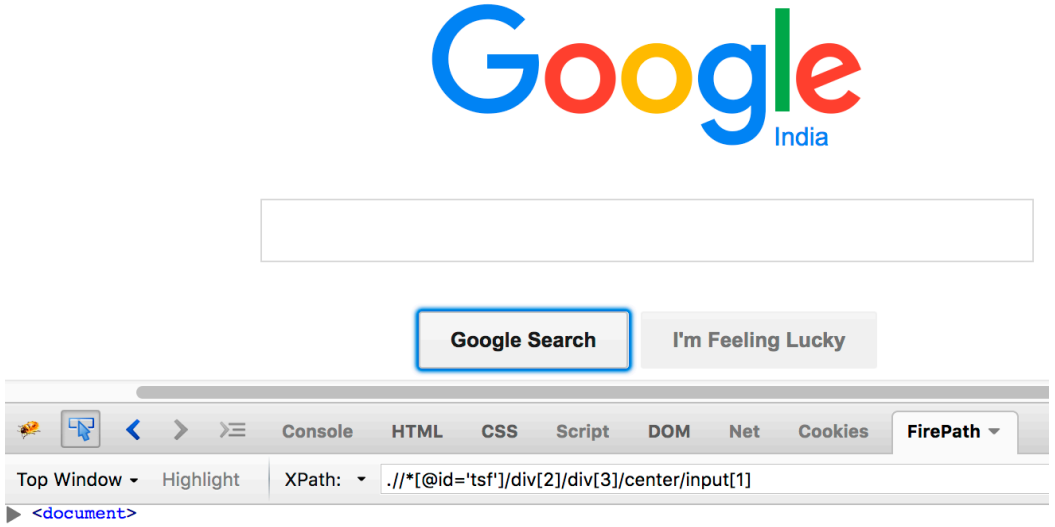
Download selenium IDE - https://addons.mozilla.org/en-us/firefox/addon/firepath/

Step1: Activate firebug 

Step 2:

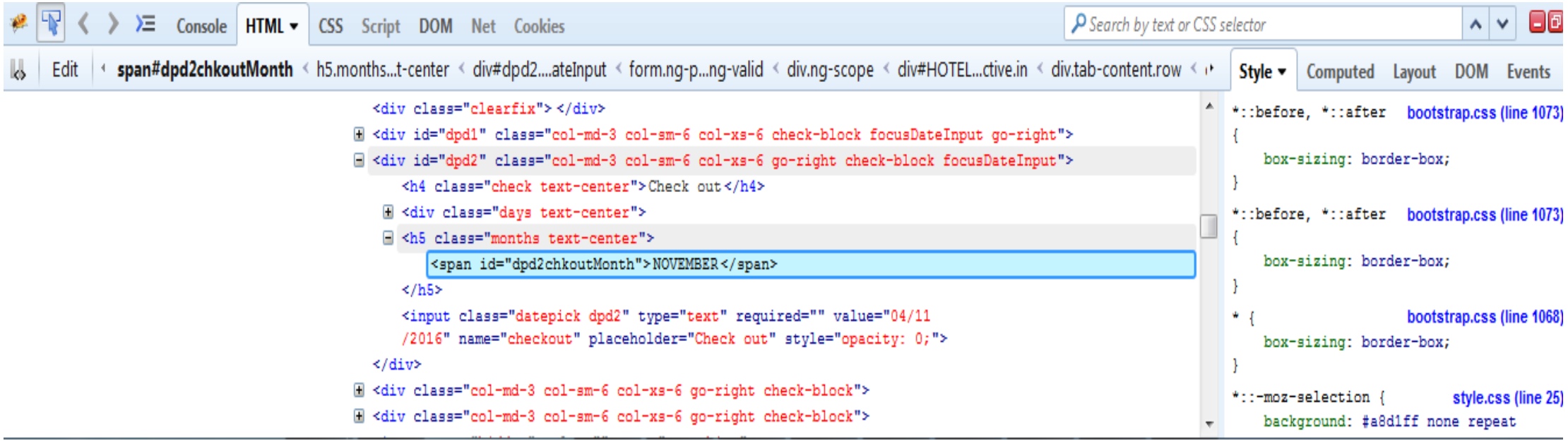
Click on pointer icon

Step3: Select an element and firebug shows the locator info.
Select FirePath tab to get XPath info.



Object Identification | Locate by ID

- View Properties of page
- The Best and most popular method to identify element is using ID. They are Unique in nature



Object Identification – Locate by Name

Object Identification - XPath

- Xpath is a syntax for defining parts of an XML document. XML stands for Extensible Markup Language and is used to store, organize and transport arbitrary data. XML stores data in Kay-Value Pair which is similar to HTML tags.
- The fundamental behind locating elements using Xpath is the traversing between various elements across the entire page and thus enabling a user to find an element with the reference of another element
- XPath can be created in 2 ways.
- Relative Xpath Method:
  - Relative Xpath begins from the current location and is prefixed with a "//"
  - For Example:

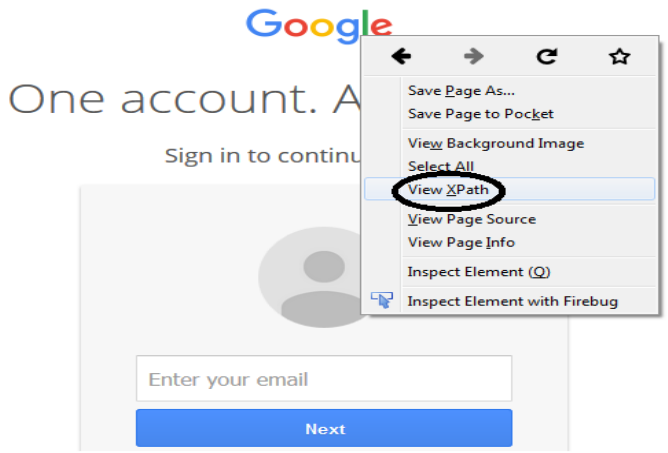
    ```
    //*[@id="link-signup"]/a
    ```

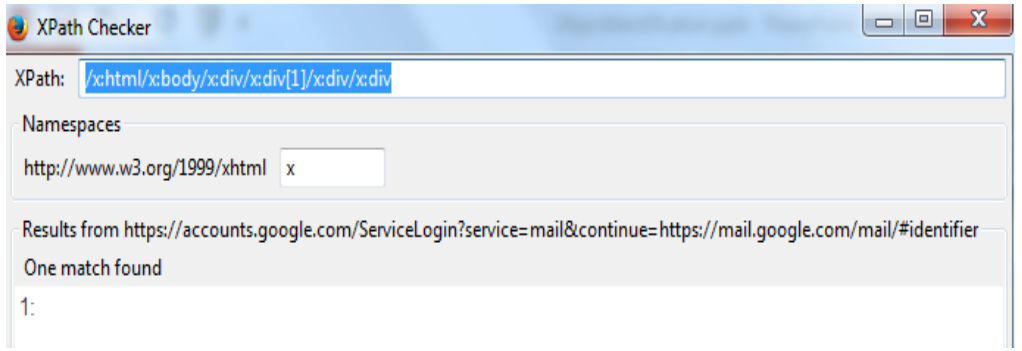
- Absolute Xpath method:
  - Absolute Xpath begins with a root path and is prefixed with a "/".
  - For example:

    ```
    /html/body/div/div[@id='Email']
    ```

Object Identification - Keypoints

- The success rate of finding an element using Xpath is too high.
- Xpath can find relatively all the elements within a web page. Thus, Xpaths can be used to locate elements having no id, class or name.
- Creating a valid Xpath is a tricky and complex process.
- While creating Xpath, user should be aware of the various nomenclatures and protocols
-
- Tools for Xpath Generation:
  - Xpath Checker Plugin:
    - Xpath is plugin available in firefox and makes easy to retrieve the Xpath.
  - FirePath Plugin:
    - When View FirePath is selected for Google logo, below Absolute Xpath gets generated.

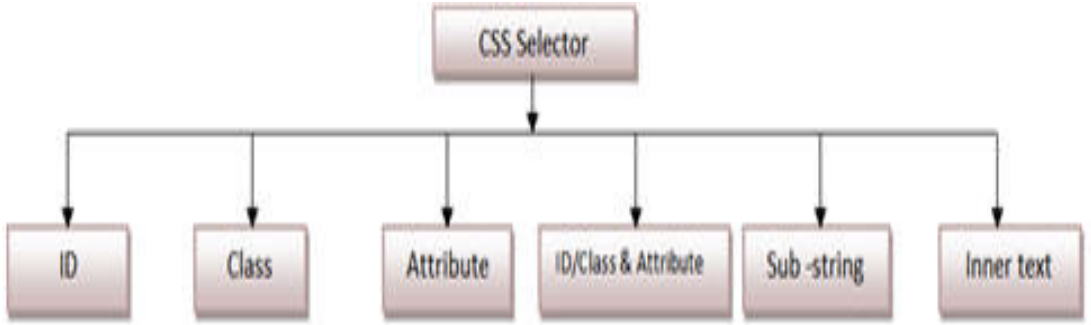




Object Identification - CSS Selector

CSS Selector: ID

- We would access “Email” text box present in the login form at Gmail.com.
- The Email textbox has an ID attribute whose value is defined as “Email”. ID attribute and its value can be used to create CSS Selector to access the email textbox.



Identify Element using CSS Selector:

- CSS Selector is combination of an element selector and a selector value which identifies the web element within a web page. The composite of element selector and selector value is known as Selector Pattern
- It is constructed using HTML tags, attributes and their values.
- Similar to XPath, CSS selector can locate web elements without ID, class or name.
- Following flow shows how the CSS selector is organized.

Object Identification - CSS Selector

- Inspect the web element (“email” textbox of gmail)
  - Observe the html tag is “input” and value of ID attribute is “Email”
  - Both of them collectively make a reference to the “Email Text box”.
  - Hence the Syntax for above data would be used to create CSS Selector.

    css=<HTML tag><#><Value of ID attribute>

  - HTML tag: tag used to denote the web element which we want to access.

  - # : The hash sign is used to symbolize ID attribute. It is mandatory to use hash sign if ID attribute is being used to create CSS Selector

  - Value of ID attribute – It is the value of an ID attribute which is being accessed.
    The value of ID is always preceded by a hash sign.

  - css Selector for email Text Box would be

    css=input#Email

Object Identification - CSS Selector

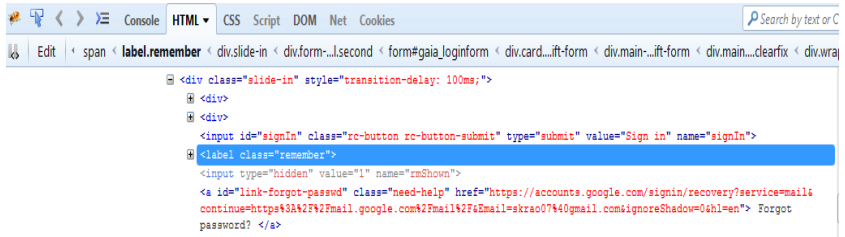
Access “Stay signed in” check box the login form at gmail.com.

- The “Stay signed in” check box has a Class attribute with value defined as “remember”.
- Thus Class attribute and its value can be used to create CSS Selector to access the designated web element.

Locating an element using Class as a CSS Selector is very much similar to using ID, the lone difference lies in their syntax formation

Creating CSS Selector for web element

- Inspect the web element (“Stay signed in” check box)
- Observe that the html tag is “label” and value of attribute is “remember”
- Both of them collectively make a reference “Stay signed in check box”.
- Hence the Syntax for above data would be used to create CSS Selector.



css=<HTML tag><.><Value of Class attribute>

- . => The dot sign is used to symbolize Class attribute. It is mandatory to use dot sign if Class attribute is being used to create CSS Selector.
- The value of Class is always preceded by a dot sign.
- css Selector for email Text Box would be

css=label.remember

In the similar way as done for ID, Class rest of the locators “attribute, ID/Class and attribute, Sub-string, Inner text etc are used.

Assertion

What is Assertion?

- Asserts helps us to verify the conditions of the test and decide whether test has failed or passed. A test is considered successful ONLY if it is completed without throwing any exception
- Assertions are very similar to Accessors as they do not interact with the application directly.
- Assertions are used to verify the current state of the application with an expected state
- There are different frameworks used with Selenium Webdriver for doing Assertion, verify etc. Junit and TestNG are the popular and widely used frameworks.

Assertion – Continued…

```
@Test
public static void verify_Flight_HomePageDisplay() throws InterruptedException {

    System.setProperty("webdriver.gecko.driver", ”<Firefox driver path>");
    WebDriver driver=new FirefoxDriver();
    driver.navigate().to("http://www.phptravels.net");
   Assert.assertEquals(”phptravels", driver.getTitle());
```

The above example does the following things:

1. *@Test* annotation is used to run Junit/testng test.
2. Creates the firefox driver
3. Navigates to phptravels.net
4. Asserts the page title

Now lets look at failing the test case.

Assertion – Continued…

```
@Test
public void testCaseVerifyHomePage() {
        driver= new FirefoxDriver();
        driver.navigate().to("http://google.com");
        Assert.assertEquals("Gooogle", driver.getTitle());
}
```

In the example, asserting for the title Gooogle, which doesn't exist on the page

It would throw an Assertion error as below:

java.lang.AssertionError: expected [Google] but found [Goooooogle]

Like wise there are many assertions present in TestNG framework.

Following are few which are used quite often.

- Assert.assertEquals(String actual,String expected) :- Takes two string arguments and checks whether both are equal, if not it will fail the test.
- Assert.assertTrue(condition) :- Takes one boolean arguments and checks that a condition is true, If it isn't, an AssertionError is thrown.
- Assert.assertFalse(condition) :- Takes one boolean arguments and checks that a condition is false, If it isn't, an AssertionError is thrown.

Assertion – Continued…

Following is the example which uses few of the assertions.

```
@Test  public static void verify_Flight_HomePageDisplay() throws InterruptedException {
    //Firefox Driver
    System.setProperty("webdriver.gecko.driver", "/tmp/Selenium/geckodriver");
    WebDriver driver=new FirefoxDriver();

    driver.navigate().to("http://www.phptravels.net");
    driver.manage().window().maximize();

    //Identify element using id
    //Get default passenger count value
    String strPassengerCount = driver.findElement(By.id("adults")).getAttribute("Value");       //Identify element
                                                                                                    using name

    //Get default checkin date
    String strCheckinDate = driver.findElement(By.name("checkin")).getAttribute("Value");       //output values to
                                                                                                    console

    System.out.println (strPassengerCount);
    System.out.println (strCheckinDate);       //Assert if default values are correct

    Assert.assertEquals(strPassengerCount,"2","Default passenger count needs to be 2");
    Assert.assertTrue(strCheckinDate.equals("14/11/2016"),"Check-in date has to be today's date");
```

Browser Plugins for Object Identification

- Firefox – Selenium IDE
- Firefox – Firebug
- Firefox – FirePath
- Google Chrome Developer Tools
- IE Developer tool

Google Chrome – Developer Tools

The Chrome Developer Tools (DevTools for short), are a set of web authoring and debugging tools built into Google Chrome. The DevTools provide web developers deep access into the internals of the browser and their web application.

Step 1:

Select the Chrome menu at the top-right of your browser window, then select Tools > Developer Tools.

Or

Right-click on any page element and select Inspect Element.

Step 2:

Click on pointer icon on any element

IE – Developer (F12) Tools

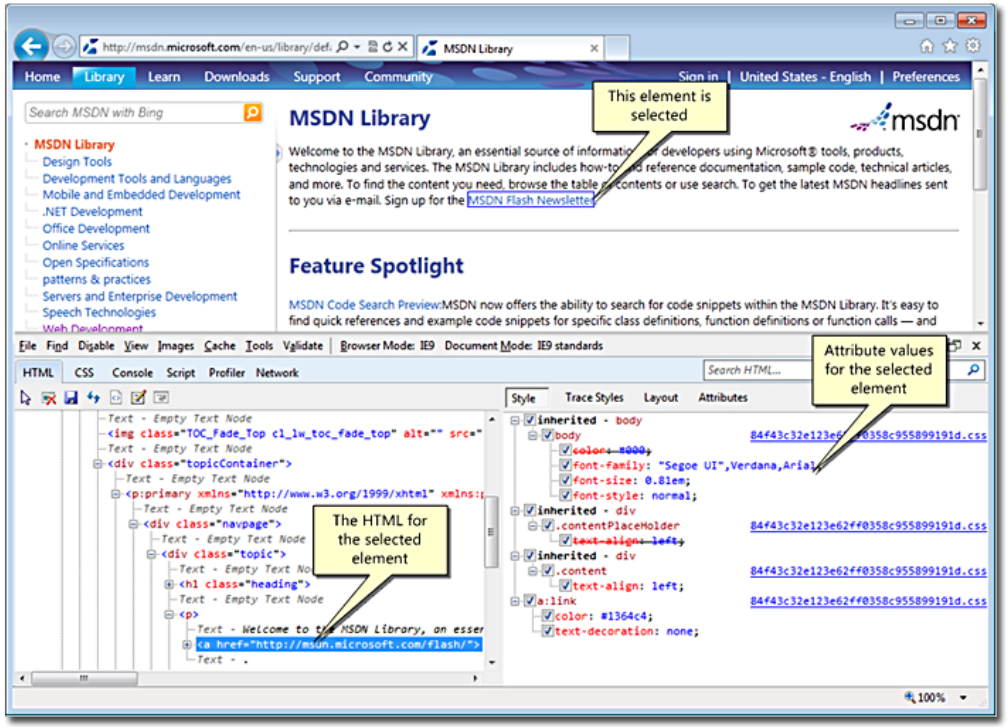
Every installation of Windows Internet Explorer 8 comes with the Developer (F12) Tools. This tool help identify and report about elements on the page such as links and images.

Step1:

F12 tools are available on any page you are viewing from within Internet Explorer by pressing F12, or by clicking the tools button , and then selecting F12 tools

Step 2:

Click on pointer icon and select any element



Exercise

- Identify Web Elements
- Perform Simple Actions
- Get Element Properties for Verification
- Identify Web Elements using different Locators
  - ID
  - Name
  - Xpath
  - CSS

Day 2 - Summary

Today we learnt :

- What is Web Elements and Locating & Performing Actions
- Locators in Detail – ID, Name, xpath, CSS
- Browser Plugins for Object Identification
- Chrome Developer Tools
- IE Developer Tools