Contents

[Selenium Locators: 3](#_Toc421141914)

[LinkText 3](#_Toc421141915)

[Id 3](#_Toc421141916)

[className 4](#_Toc421141917)

[name 5](#_Toc421141918)

[tagName 6](#_Toc421141919)

[Xpath 6](#_Toc421141920)

[Few Browser based Tools to find the value for Selenium Supported locators 6](#_Toc421141921)

[I. Developer Tool: 6](#_Toc421141922)

[II. Firebug: 9](#_Toc421141923)

[III. Selenium IDE: 11](#_Toc421141924)

[Installing the IDE 11](#_Toc421141925)

[Opening the IDE 15](#_Toc421141926)

[How to Frame Xpath for the web element efficiently? 17](#_Toc421141927)

# Selenium Locators:

To interact with any element available on your application, it is important to identify the element first.

Selenium uses various locators to locate the target element.

1. linkText
2. id
3. className
4. name
5. tagName
6. xpath
7. css

Let’s discuss about each locator type in detail.

## LinkText

This type of locator applies only to hyperlink texts, selects the anchor element containing the specified text.



LinkText locator

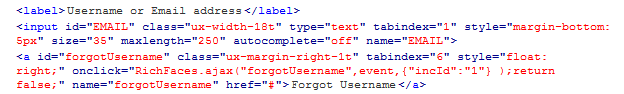
Above html code is part of Frontier.uhg.com. As a welcome message you will see your name appear as Hyperlink.

To interact with this element we can use linktext locator.

You can see it has anchor tag.

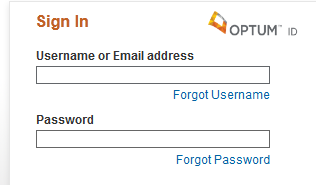
## Id

ID is the best locator to locate the element but you need to be careful that ID for the element is not changing dynamically at runtime.

****

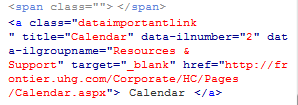
ID locator

In the above html code for username field id can be used as locator and even for forgot username link id can be used to interact with Forgot Username link.

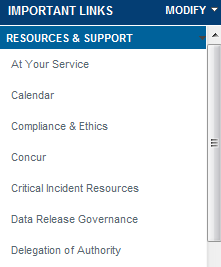
****

## className

To click on Calendar link we can use classname as locator.

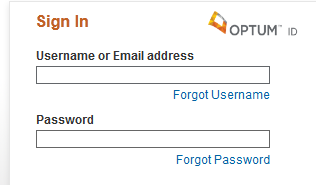
****

Class locator

****

## name

name is another locator which can be used to locate the element by selenium.

****

As you can see below for password field, we can see the name attribute, you can use its value as a locator to interact with password text box.



Name locator

## tagName

Tagname is also can be used as a locator to interact with element, but it need to be unique on the page.

Example of how to find an element that looks like this:

<iframe src="..."></iframe>

Here the iframe can be used as locator.

## Xpath

XPath is the language used when locating XML (Extensible Markup Language) nodes. Since HTML can be thought of as an implementation of XML, we can also use XPath in locating HTML elements.

It can access almost any element, even those without class, name, or id attributes.

You can take initially help of Firebug plugin to generate Xpath for the element for you, with initial few practice, it would be very easy to frame xpath for the target element.

While framing xpath you can put ‘and’ & ‘or’ condition as well.

We will discuss about framing xpath in detail with examples in subsequent section.

# Few Browser based Tools to find the value for Selenium Supported locators

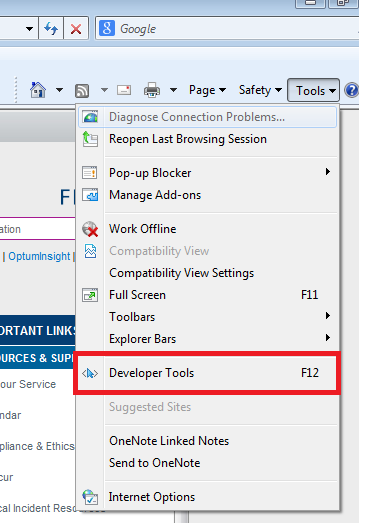
There are various tools available but below mentioned tool would be sufficient to begin with and would help you to find the value for locators.

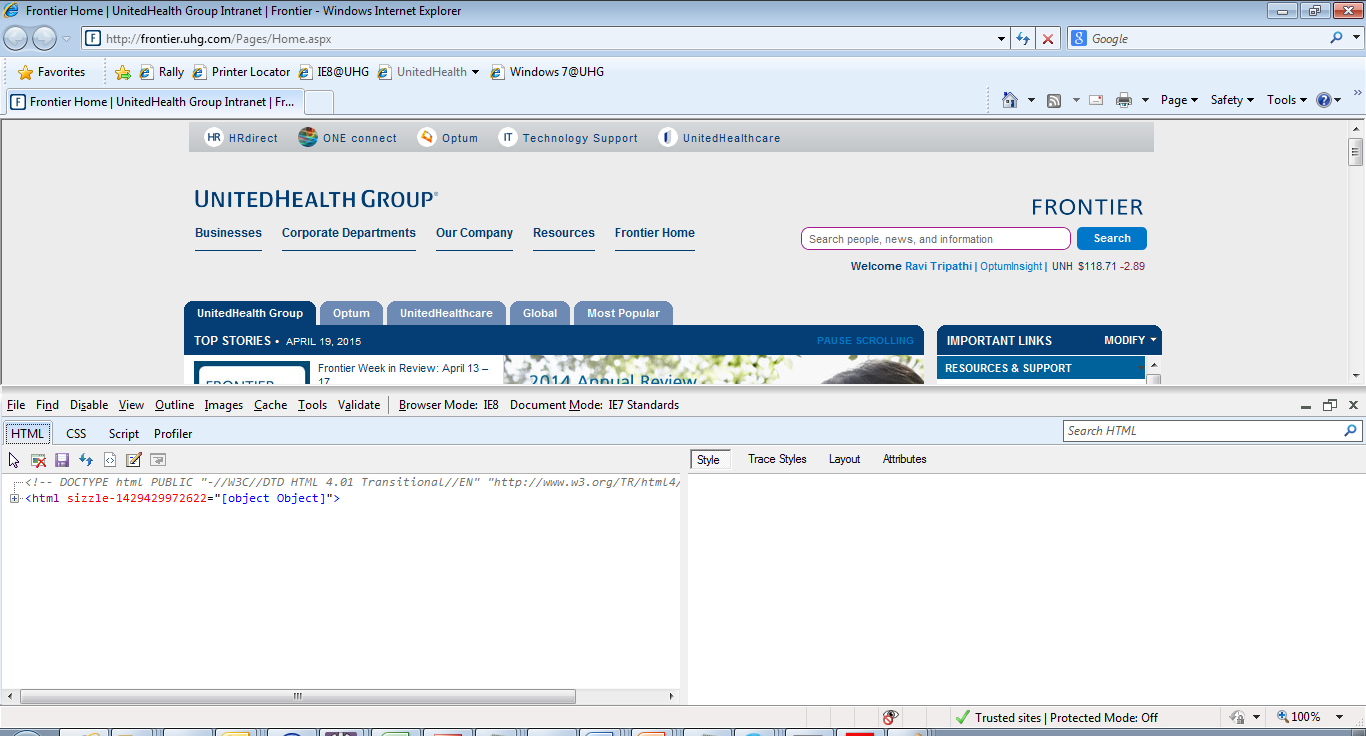
1. Developer Tool in IE(workswith IE only)
2. Firebug(works with Firefox only )

## Developer Tool:

There is a inbuilt tool comes with Internet explorer 8 and higher version of IE,

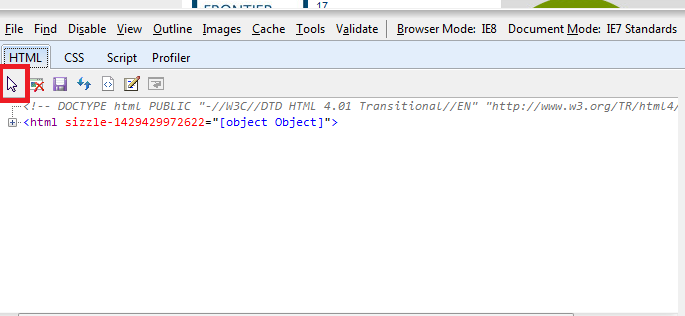
You can find this in Tools. Alternatively even you can press F12 and developer tool will open.





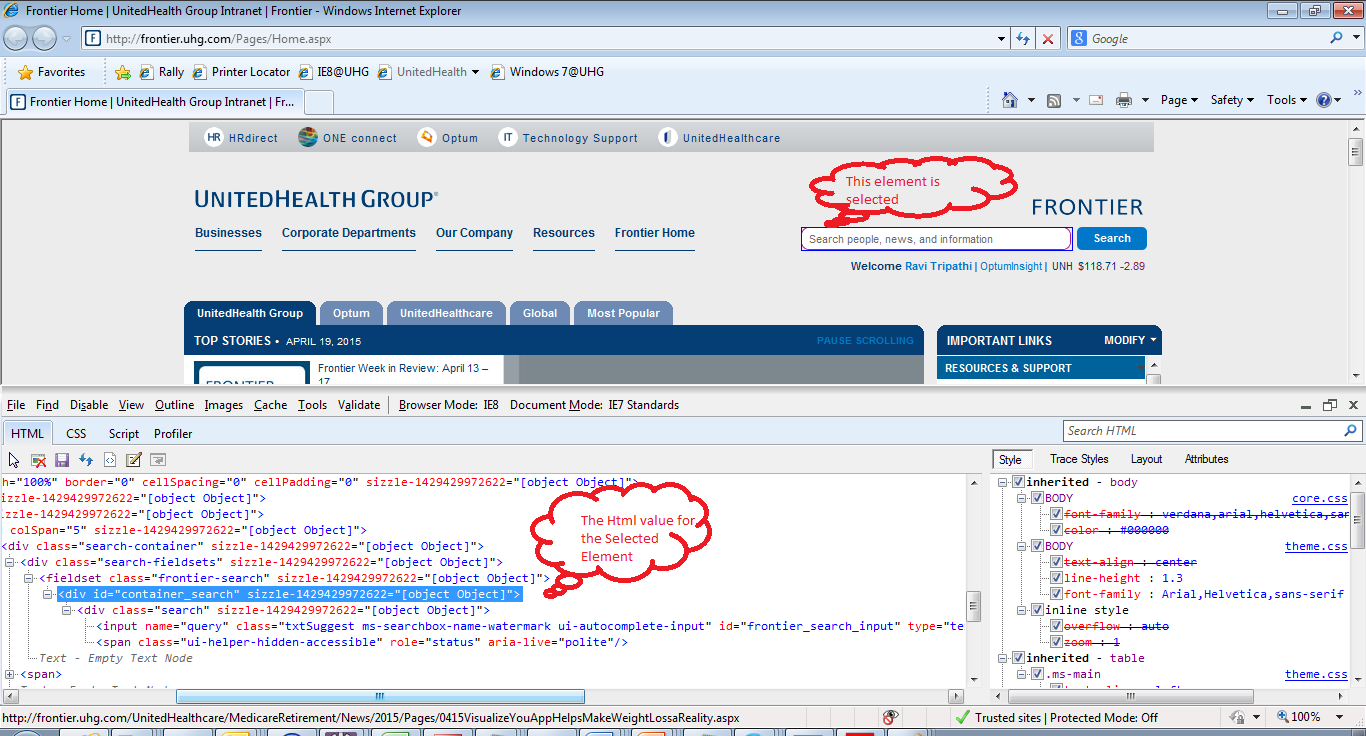
Once you will open the developer tool, you can see it open in in the bottom of the page, it will have DOM (Document Object Model) hierarchy of HTML element on the particular page.

You can see the pointer in left side of the Developer tool window.



To get the element’s HTML attribute , you need to click on the pointer and then click on the element.

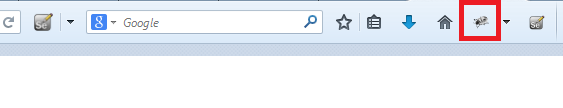
In below image, Search text box has been selected.



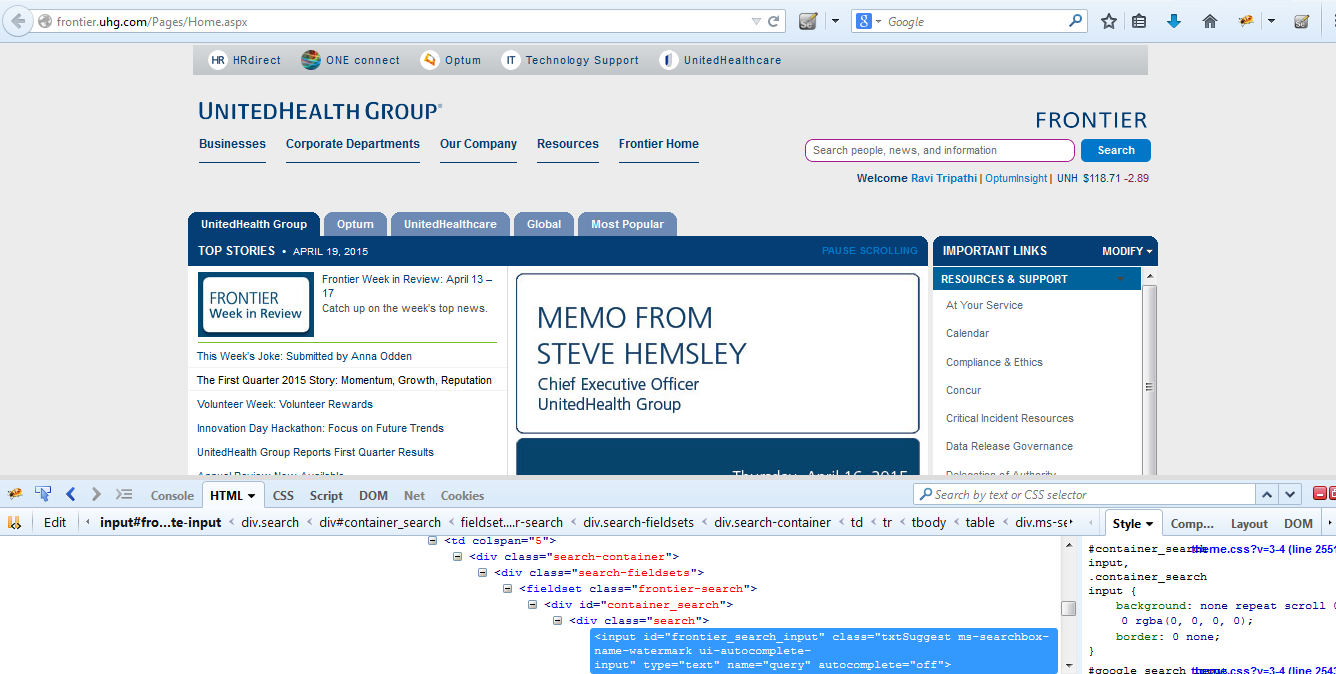
## Firebug:

Firebug is an add-on tool available with Firefox to inspect the element

Once installed this add on you can see the icon in your firefox browser as highlihted in below image.

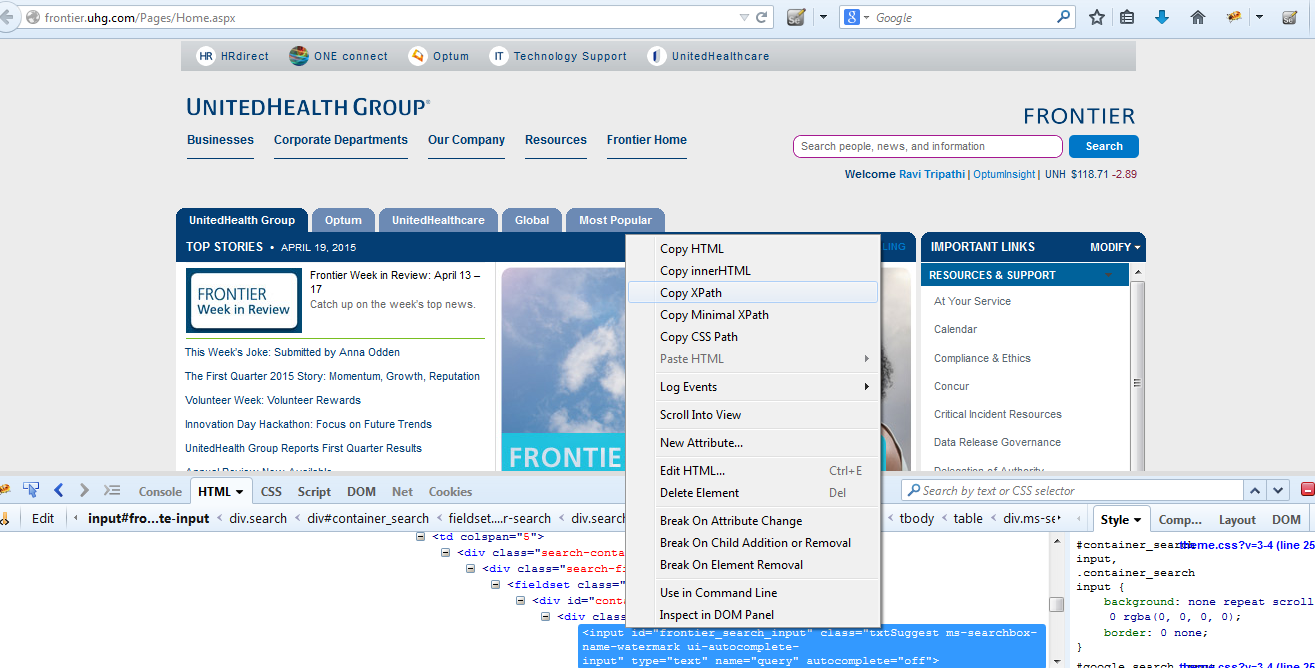


You can click the icon or you can press F12, it will open FireBug tool in your browser, it is similar to developer tool as you have seen in IE browser.

****

You can find the xpath as well using this tool.

Right click on the HTML code of inspected element; there is an option copy xpath.

****

# How to Frame Xpath for the web element efficiently?

I will explain the different ways of choosing Xpaths and choosing the most effective xpaths.

1. Technique 1 | Absolute XPath
2. Technique 2 | Relative XPath
3. Technique 3 | Relative XPath | Combination of Double Slash:
4. Technique 4 | Partial XPath | Contains Keyword
5. Technique 5 | Partial XPath | Starts-With Keyword
6. Technique 6 | Partial XPath | Text Keyword

**Technique 1 | Absolute XPath**

The easiest way of finding the xpath is to use the Browser Inspector tool to locate an element and  get the xpath of it:

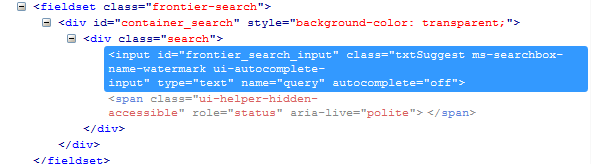
XPath Generated by the tool for search text box on Frontier application is

: /html/body/form/table/tbody/tr[2]/td/table/tbody/tr/td/div/div[4]/div/table/tbody/tr/td/div/div/fieldset/div/div/input

**Technique 2 | Relative XPath**

At times XPath generated by Firebug are too lengthy and you see there is a possibility of getting a shorter XPath. Above xpath will technically work, but each of those nested relationships will need to be present 100% of the time, or the locator will not function.  Above taken xpath is known as **Absolute xpath**. There is a good chance that your xpath will vary in every release. It is always better to choose **Relative xpath**, as it helps us to reduce the chance of element not found exception.

To choose the relative xpath, it is advisable to look for the recent **Id**attribute or any element which has certain attribute which you feel is unique. Look below at the HTML code of the above screen shot.



You can see the recent or last Id produced is ‘**container\_search**‘. This id would be appropriate in this case, so a quality xpath will look like this:  **//\*[@id= container\_search’]/div/input**

**Did you notice the difference between the Absolute and Relative xpaths?**

**Absolute** Xpath: /html/body/form/table/tbody/tr[2]/td/table/tbody/tr/td/div/div[4]/div/table/tbody/tr/td/div/div/fieldset/div/div/input

**Relative** xpath: :  //\*[@id= container\_search’]/div/input

Absolute xpath is using single slash at the start of the xpath and relative is using double slash.

**Difference between single ‘/’ or double ‘//’**

A **single slash** at the start of Xpath instructs XPath engine to look for element starting from **root node**.A **double slash** at the start of Xpath instructs XPath engine to search look for matching element **anywhere** in the XML document.

**Technique 3 | Relative XPath | Combination of Double Slash:**

Relative xpath can be choose in many ways and to understand that, it is required to understand the usage of single & double slashes in the xpaths.

**Usage of Single ‘/’ and double ‘//’ in the xpath**

A **single slash** ‘/’ anywhere in Xpath signifies to look for the element **immediately inside** its parent element.

A **double slash** ‘//’ signifies to look for **any child** or **any grand-child** element inside the parent element.

Finding it confusing, just look at the xpath of the same Search text box with using double slashes in the middle of the xpath:

**Absolute xpath** : /html/body/form/table/tbody/tr[2]/td/table/tbody/tr/td/div/div[4]/div/table/tbody/tr/td/div/div/fieldset/div/div/input

**Relative xpath** :  //\*[@id= container\_search’]/div/input

**New relative xpath**: //body/form/table/tbody//div/table/tbody/tr/td/div/div/fieldset/div/div/input

**Another relative xpath**: //body //div/table/tbody/tr/td/div/div/fieldset/div/div/input

Better you try it yourself, so that you can understand it more efficiently.

**Technique 4 | Partial XPath | Contains Keyword**

Most of the times tester face issues when the locator’s properties are dynamically generating. Let’s take the example of image element and assume that the ‘src’ of the image is dynamically generating. The html code of the div looks like this:

<div class="profile\_cont" itemtype="http://schema.org/Article" itemscope="">

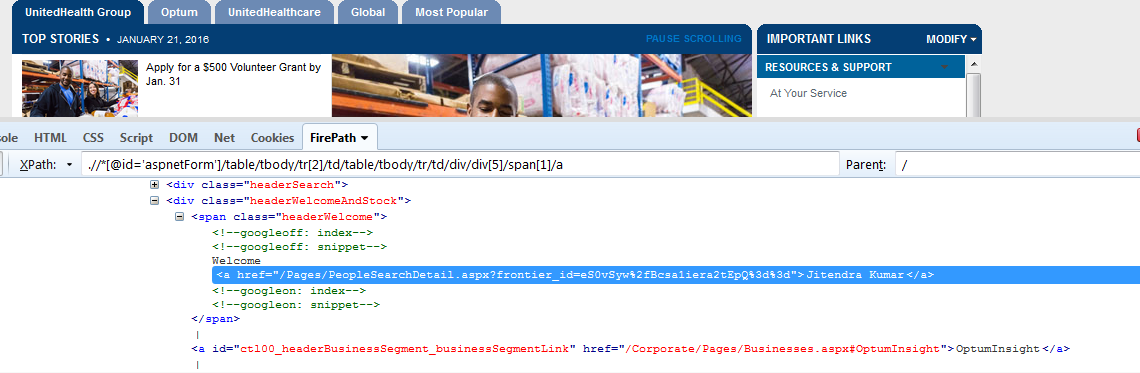
<img src="http://toolsqa.com/wp-content/uploads/2014/02/**Profile**.jpg?51a507" itemprop="image">

</div>

The only thing we are sure here is that the text ‘Profile’ will always be included in the src of this image, so we can utilize this hint in our xpath like this: **//img[contains(@src,’Profile’)]**

**Technique 5 | Partial XPath | Starts-With Keyword :**

Now let’s take another example of the Welcome <Username> on this Frontier home page at the right side of the screen just under search text box and again you can see with different login link text would be different , in my case it is “Ravi Tripathi” and href property of the link is also generating dynamically, in the href property initial value “/Pages/PeopleSearchDetail.aspx?frontier\_id” would be constant for all the login

****

So now you can frame xpath like this for this element

//a[starts-with(@href,'/Pages/PeopleSearchDetail.aspx?frontier\_id' )]

**Technique 6 | Partial XPath | Text Keyword :**

We can frame xpath for the Welcome<User> element using text keyword like this:

1. //a[text()=’UHG’ ]
2. //a[contains(text(),'UHG)]

You can practice this on various element you have in your application and with few practice , you would be able to frame Xpath efficiently.

**CSS Selector:**

**Direct child**

A direct child in XPATH is defined by the use of a "/", while on CSS, it’s defined using ">"

**Examples:**

//div/a

css=div > a

**Child or subchild**

If an element could be inside another or one its childs, it’s defined in XPATH using “//” and in CSS just by a whitespace.

**Examples:**

//div//a

css=div a

**Id**

An element’s id in XPATH is defined using: “[@id='example']” and in CSS using: “#”

**Examples:**

//div[@id='example']//a

css=div#example a

**Class**

For class, things are pretty similar in XPATH: “[@class='example']” while in CSS it’s just “.”

**Examples:**

//div[@class='example']//a

css=div.example a