XPath

- Xpath = //(tagname)([@(Attribute)='(Value of attribute)'])
- // ---> Select current node
- @ ---> Selects attribute

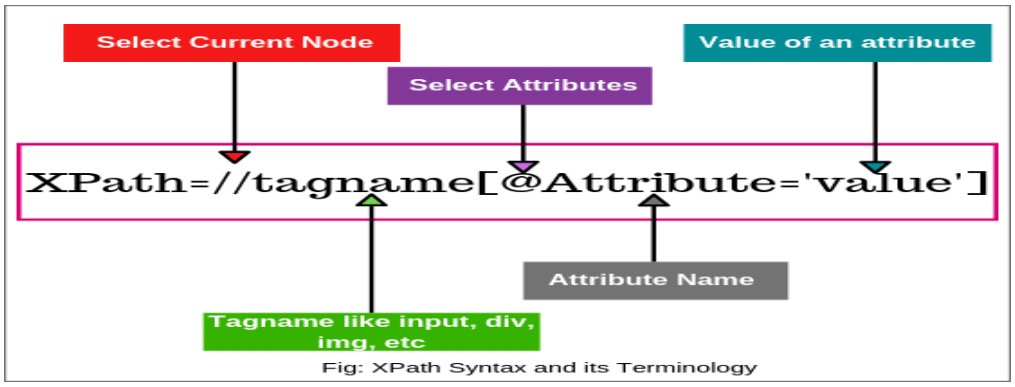


Fig: XPath Syntax and its Terminology

- Absalute xpath(/) ögeyi kök düğümden seçebileceğiniz anlamına gelir

Örn /html/body/div[2]/div[2]/...

- Relative xpath(//) Web sayfasının herhangi bir yerindeki ögeleri arayabilir, uzun bir xpath yazmaya gerek olmadığı anlamına gelir ve HTML DOM yapısının ortasından başlayabilirsiniz.

Örn //div[@class='featured-box cloumsize1']//h4[1]//b[1]

1) Standart xpath ---> Örn: ---> //input[@id='login-usename']
2) Contains ---> //*[contains(@ attribute,'value of attribute')]

   //*[contains(text(),'value of attribute')]

   //input[contains(@attribute,'value of attribute')]

3) Starts-with ---> //tagname[starts-with(@attribute,'partical static value of attribute')]
4) And & Or ---> Örn; LOGIN RESET olarak ayrı iki obje olsun

   //*[@type='submit' and @name='btReset'] LOGIN = type'submit'

   //*[@type='submit' or @name='btReset'] RESET = name='btReset'

   //input[type='submit' and @name'btReset']

   //input[type='submit' or @name'btReset']

5) Text ---> Örn --> Xpath=//div[text()='value of text']

Örn: (//img[@class='s-img'])[11]

NOT= (//img[@class='s-img'])--->53 tane objeden varsa x'inciyi almak için [x] methodu kullanılır.

CSS Selector

Tag ve ID	Tag ve Class	Tag ve Attribute	Tag ve ID/Class &Attribute	Sub-String	innertext

Matches(Starts with, Ends with, Contains)

---> id

- ---> class

1) Tag ID -----> Syntax : css=(Html etiketi)#(ID'nin değeri) --> Örnek ---> a#login
2) Tag ve Class -----> Syntax : css=(Html etiketi).(Class'ın değeri) --> Örnek ---> div.SearchBoxOld-buttonContain
3) Tag ve Attribute -----> Syntax : css=(Html etiketi)[attribute='Value of attribute']
4) Tag, Class ve Attribute -----> Syntax : css=(Html etiketi).(class)([attribute='Value of attribute'])
5) Matches --->

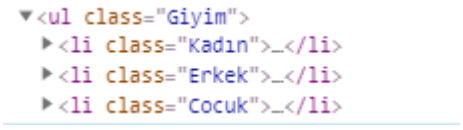
- Starts with(^) -----> Syntax : css=(Html etiketi)([attribute^='Value of attribute'])
- Ends with($) -----> Syntax : css=(Html etiketi)([attribute$='end of the string'])

Örnek ---> input[name$='ail']

- Contains(*) -----> Syntax : css=(Html etiketi)([attribute*='partial string'])

Örnek ---> inpt[class*='control']

6) Child Element -----> Syntax : css=(Html etiketi).(class name) li :nth-of-child(x)



-----> Syntax : css=(Html etiketi).(class name) li :nth-of-child(x)

. ---> class olduğu için).(arada . Kullanıldı

< li ---> olduğu için) li : Kullanıldı

X --->child elementtir

' ' ---> yazılacak attributeler '' içinde yazılmalıdır

-----> attribute : html sayfalarında kullanılan protokollerin (tag) aldığı parametrelerdir. Her bir parametre kontrole yeni bir özellik ekler. Bazı parametreler değer alır bazıları ise olduğu gibi kalır.

HTML unsurlari nelerdir ?

Bir HTML oge temelde 3 unsurdan olusur.

1- Tag

2- Attribute

3- Attribute value

1- tag :

1) tag nedir ? Bir html ogesinin baslangıcını ve bitisini belirtmek için kullanılan bir işaretleme dili parçasıdır.

```
<script>
GwInstrumentation.markH1Af({ uri: "/ah/ajax/counter?
    ctr=desktop_ajax_atf&exp=13&rId=T4M&mkId=&h=4cc3
})
</script>
```

Bir HTML ogesi yukaridaki ornekteki gibi tek bir tag icerisinde baslayip bitecegi gibi, asagidaki ornekteki gibi tag baslangic ve bitisi ayri ayri da olabilir.

<script> baslangic

</script> bitis

2- attribute :

2) attribute nedir ? Bir html attribute'u Html ögenin davranışlarını kontrol etmek üzere acilis tag'i icine yazılan ozel kelimelerdir. Eşittirden önceki şeylere attribute denir. Esittirden öncekilere attribute denir.

```
<input type="text" id="twotabsearchtextbox" value=""
name="field-keywords" autocomplete="off" placeholder=""
class="nav-input nav-progressive-attribute" dir="auto"
tabindex="0" aria-label="Search">
```

3- attribute value :

3) Attribute value nedir ? Html'deki value, birlikte kullanıldığı ögenin değerini belirtmek için kullanılır. Farklı html ögeleri için farklı anlamlara sahip olabilir.

```
<input type="text" id="twotabsearchtextbox" value=""
name="field-keywords" autocomplete="off" placeholder=""
class="nav-input nav-progressive-attribute" dir="auto"
tabindex="0" aria-label="Search">
```

Not : Birden fazla html oge icin ayni tag, attribute ve attribute value kullanılabilir.

-----> Locate islemi ise unique(benzersiz) olmalıdır.

-----> Locate unique(benzersiz) olmazsa driver islem icin kendisine locate edilen elementlerden hangisine gidecegini bilemeyecegi icin NoSuchElemtExeption verecek ve işlemi yapmayacaktır.

Özetle : "Locate" islemi, birbirine benzer ozelliklerde olabilen Html ogesini "Unique" olarak "belirleme" islemidir.

-----> Selenium'da 8 adet locator vardır.

Bizim test yaparken bu 8 locator'dan hangisinin kullanacağımız WebElementi unique olarak tarif edebileceğini bulup onu kullanmamız gerekir.

Her bir locate işlemi için ilgili Web sayfasına manual olarak gidip, Html elementini incelememiz,uygun locator'i bularak webElement'i locaete etmemiz gerekir.
Ornek : Facebook giriş butonu, Amazonda Nutella ornegi gibi..

Locaterlerden 4 tanesi çok kullanılan html ogeleri ile yapılır.

1) id:

```
<input type="text" id="twotabsearchtextbox" value=""
```

WebELement aramaKutusu=
driver.findElement(By.id("twotabsearchtextbox"));

Web ogesini tanımlanana en populer yolu id kullanmaktır.

id en güvenli ve en hızlı locator seçeneği olarak kabul edilir ve her zaman birden çok locator arasında ilk öncelik olarak denenir.

id genelde unique olarak kullanılsa da developer'lar unique yapmayabilir. Locator olarak id sececeksek unique oladugundan emin olmamız gerekir.

```
<input type="text" id="twotabsearchtextbox" value=""
name="field-keywords" autocomplete="off" placeholder=""
class="nav-input nav-progressive-attribute" dir="auto"
tabindex="0" aria-label="Search">
```

2) name

```
WebELement aramaKutusu=
        driver.findElement(By.name("field-keywords"));
```

3) classname

```
WebELement aramaKutusu=
        driver.findElement(By.classname("nav-input nav-progressive-attribute"));
```

Class attribute'u genelde aynı islevi yapan bir grup Web Elementi tanımlamak icin kullanılır ve unique olmaz.

Class attribute'nun degeri bosluk iceriyorsa, By.classname() ile yapılan locator'lar sağlıklı calismayabilir.

4) tagname

Tag isimleri genelde aynı oldugundan unique degere ulasmak zordur.

Not : Unique olduktan sonra hangisinin kullandıldıgı onemli degil.

-----> 8 Locatorlerimden "2" tanesi link olarak tanımlanmış webElementlerde kullanılır. Sadece linkte geçerli.

```
<a class="nav-item nav-link" data-test="addresses" href="/addresses">Addresses</a>
```

5) linkText

Kullancağımız webElement bir link ise üzerindeki yazinin tamamini kullanarak locate edebiliriz.

```
WebELement aramaKutusu=
        driver.findElement(By.linkText("Addresses"));
```

Not : Link üzerindeki bir String oldugundan buyuk,kucuk harf, bosluk gibi durumlara dikkat edilmelidir.

6) partialLinkText

Link üzerindeki yazının tamami degil bir kismini kullanarak da unique bir sonuca ulasabiliyorsak partialLinkText kullanılabilir.

```
WebELement aramaKutusu=
        driver.findElement(By.partialLinkText("esses"));
```

Not : 7.ve 8. locaterler her webELement icin bu yontemlerle unique bir sonuca ulasmak mumkundur. En güclü webELementlerdir.


```
<input type="text" id="twotabsearchtextbox" value=""
name="field-keywords" autocomplete="off" placeholder=""
class="nav-input nav-progressive-attribute" dir="auto"
tabindex="0" aria-label="Search">
```


7) xpath()

En güclü locator'dir ve tum webElementleri unique olarak belirleyebilir.

WebELement aramaKutusu=
driver.findElement(By.xpath("//input[@type='text']"));

//tagName[@attributeismi='attributeValue']
//img[@class='s-image'] Amazondan örnek verdik, burda tüm resimleri gösterir
(//img[@class='s-image'])[57] 57 yazan yere ne yazarsan sırayla o resimleri gösterir.

Her attribute kullanılabileceği icin bir Html ogesi icin birden fazla xpath yazılıp, bunlardan unique olan kullanılabilir.

8) cssSelector

Xpath'e benzer ve tum webElementler icin kullanılabilir. Farkı xpath'de kullandığımız // ve @ isaretinin kullanılmamasıdır.

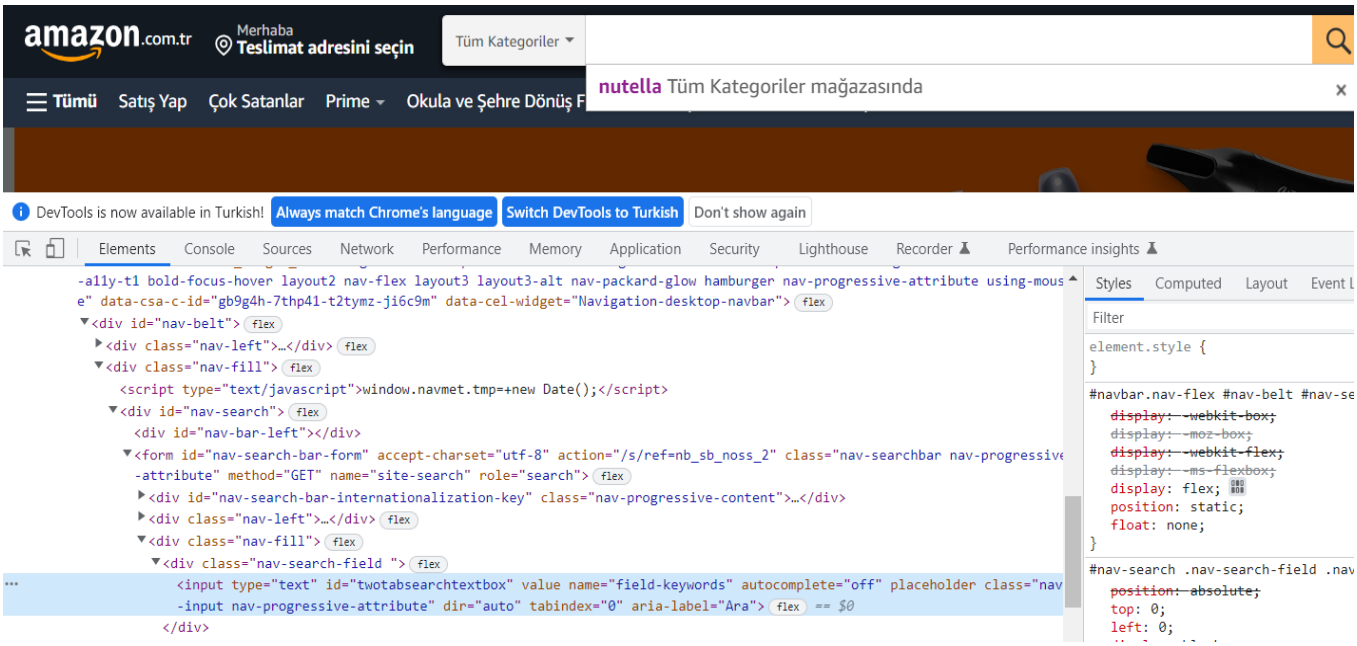
tagName[attributeismi='attributeValue']

Web Sayfasındaki Öğeleri Nasıl Buluruz?

1) İstediğimiz nesneye sağ tıkla ile incele(inspect) alanına tıklıyoruz. İncele(inspect) alanına tıkladıktan sonra objenin attribute'lerini göreceksiniz. Ya da otomasyon yazacağımız site içerisinde klavyeden F12'ye basarak geliştirici seçeneklerine ulaşıyoruz. Bu ekranı farklı bir pencerede ya da sayfanızın solunda, sağında, aşağısında duracak şekilde açabilirsiniz

2)Burada 'Elements' tabını açıyoruz. Web sayfamızdaki bütün elementleri burada görebiliriz.

3)Elementin bilgilerinden size uygun olan locator için gerekli bilgileri alıp otomasyonunuzda bu elemente ulaşmak için kullanabilirsiniz.



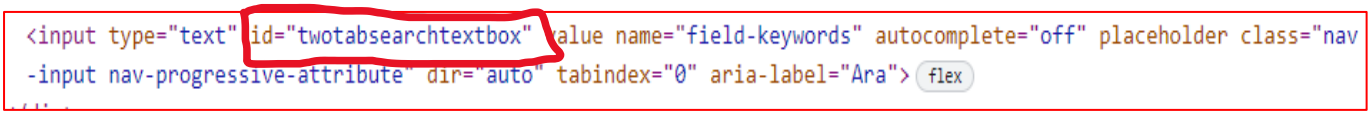
-------> Arama kutusuna sağ tıklayıp incele deyince yukarıdaki gibi bir ekran çıkar. Bizim için önemli olan çıkan ekrandaki tag'lar içinden unıque olanı bulmak ve işleme sokmaktır.

Yer Bulucuların Özellikleri ve Kullanımları

Selenium Web Driver'da ilgili elementleri bulabilmek için 'findElement/findElements' syntaxını kullanmalıyız.

1)ID

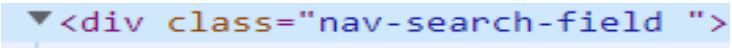
ID, web sayfasında her öğeye özgü olduğu düşünülerek öğeleri bulmanın en yaygın yoludur. Id'nin değişken olup olmadığı sayfa yenilenerek tekrar ilgili elemente ulaşıp kontrol ederek anlaşılabilir.



driver.findElement(By.id("createacc"))

2)ClassName

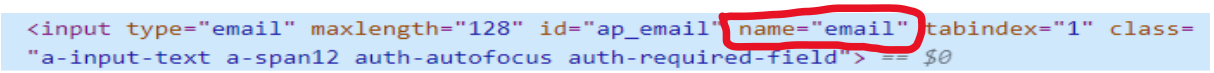
ClassName locator, elementin class özelliği kullanılarak bulunmasını sağlar.



driver.findElement(By.className("btn btn-lg btn-primary"))

3)Name:

Selenium WebDriver'daki Name locator, ID gibi kullanılabilir.



driver.findElement(By.name("login-password"))

4)Tag Name

Selenium WebDriver'daki bu bulucu, div etiketi, etiket vb. gibi etiket adlarına sahip öğeleri tanımlamak için kullanılır.

driver.findElements(By.tagName());

Örnekler;

driver.findElements(By.tagName(a));

driver.findElements(By.tagName(div));

5)LinkText

Elementler, bağlantı metni aracılığıyla yerleştirilebilir. Aynı metnin birden çok bağlantısının bulunduğu bir senaryoda, ilk bağlantı seçilir.

Örnek element;

Blog

Elementi bulmak için linktext kullanımı;

Syntax: driver.findElement(By.linkText("Blog"))

6)Xpath

Xpath, XML ifadelerini kullanarak web sayfasındaki öğeleri bulmaya yardımcı olur.

Xpath=//tagname[@Attribute='Value']

tagname= Hedeflediğiniz elementin etiketi, örneğin bir giriş(input) etiketini ve ya bağlantı(anchor) etiketini, vb. belirtir.

attribute= '@' ön eki ve karşılık gelen değerleri ile tanımlanır. Name, ID, Class vb.olabilir.

Xpath Kullanımları

Xpath Seçicileri çeşitli biçimlerde bulunabilir:

- Standard Xpath
- Contains
- AND & OR
- Starts-with
- Text

1.1)Standard Xpath

Xpath'in standart syntax'ı ile kullanımıdır.

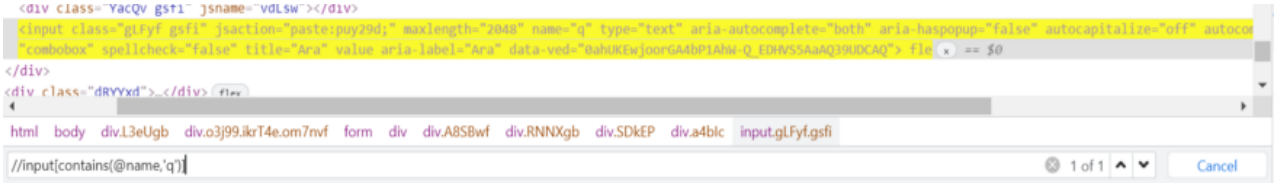
driver.findElement(By.xpath("//input[@id= 'login-username']"))

1.2)Contains

Herhangi bir özelliğin değeri dinamik olarak değiştiğinde kullanılır.

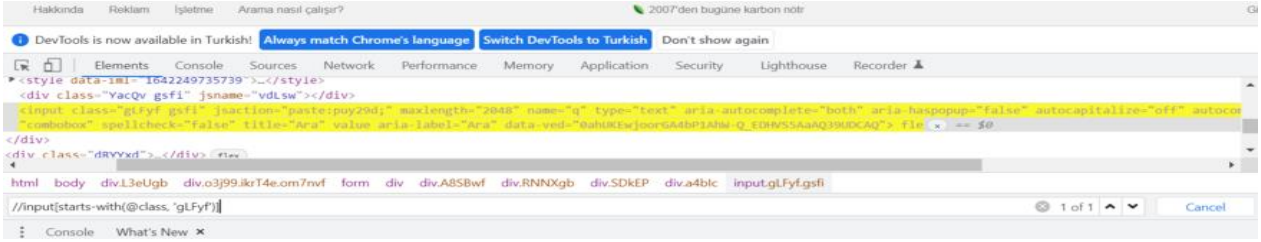
Contains kullanarak xpath'ini yazarsak;

driver.findElement(By.xpath("//input[contains(@name, 'q')]"))



1.3) Starts-With

Yenilenen veya web sayfasındaki diğer dinamik işlemlerle değiştirilen web öğesini bulmak için kullanılan bir işlevdir. Aramalarımızda uzun kodları daha kolay bulmamızı sağlar.



driver.findElement(By.xpath("//input[starts-with(@class,'gLFyf')]"))

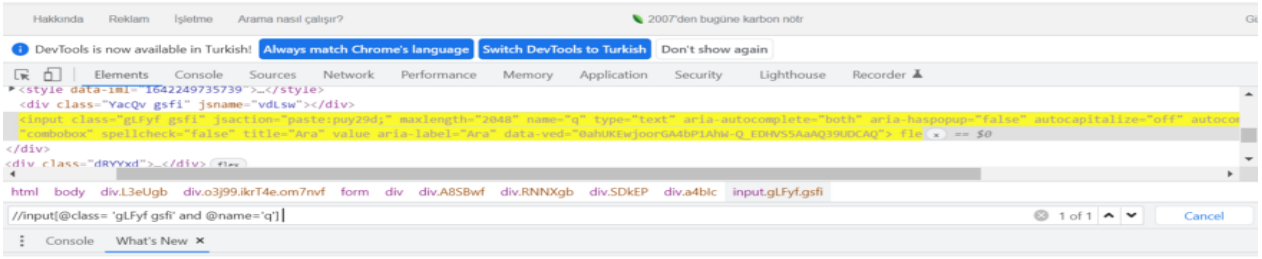
1.4) AND ve OR Kullanımı ile Xpath

OR ifadesinde 1.koşulun ya da 2. Koşulun doğru olması yeterlidir. Her iki koşulun doğru olması durumunda da geçerlidir. AND ifadesinde iki koşul kullanılır, öğeyi bulmak için iki koşul da doğru olmalıdır. Herhangi bir koşul yanlışsa öne bulunamaz.

OR için syntax: Xpath=//input[@class='value of id' OR @name='value of name']

-----> İfadelerden birisinin doğru olması yeterlidir yani ikinci ifade yanlış olsa bile doğruyu bulur. OR öperatörü birinin doğru olasını kabul eder ve öyle çalışır.

AND için syntax: Xpath=//input[@class='value of id' AND @name='value of name']

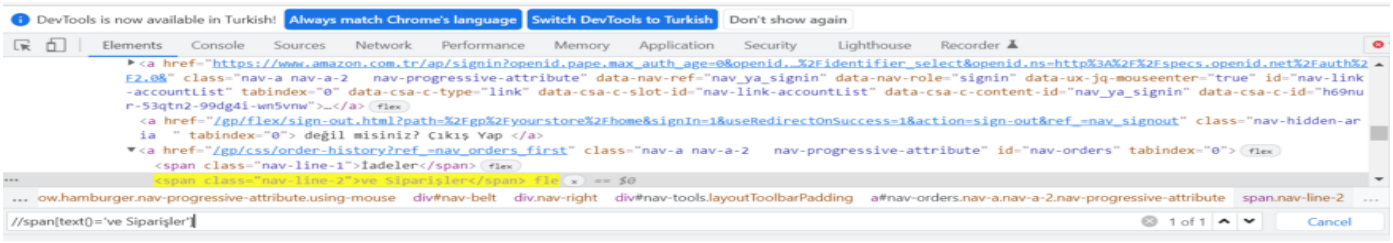


-----> Burada ise her iki ifade de doğru olmak zorundadır çünkü AND operatötü diğer türlü sonuç vermez.

1.5) Text

Metin eşleşmesini kullanarak elementi bulmak için kullanılan bir yöntemdir. Bazen elementler id, class gibi özelliklere sahip değilken sadece text barındırabilir. Bu elementleri bulmamız için text yöntemiyle xpathini yazmak bize yardımcı olmuş olur.

Syntax: Xpath=//div[text()='value of text']



Xpath=//div[text()='ve Siparişler']

7)CSS Selector

Bir elementte ID ya da name ile ilgili bir bilgi yoksa veya bunlar değişken ise CSS selector ve Xpath ile elementi bulmaya çalışırız. Xpath ile karşılaştırıldığında CSS selector daha hızlı çalışmaktadır.

CSS Selector Kullanımları

Bir elementte ID ya da name ile ilgili bir bilgi yoksa veya bunlar değişken ise genellikle CSS Selector veya xpath kullanılır. xpath ile karşılaştırıldığında CSS Selector daha hızlı çalışmaktadır. Bu nedenle bir engel yoksa öncelikle CSS Selector tercih edilir.

CSS Seçicileri çeşitli biçimlerde bulunabilir:

- Tag ve ID
- Tag ve Class
- Tag ve Attribute
- Tag, Class ve Attribute
- Matches (Starts with, Ends with, Contains)
- Child elementler

2.1)Tag ve ID

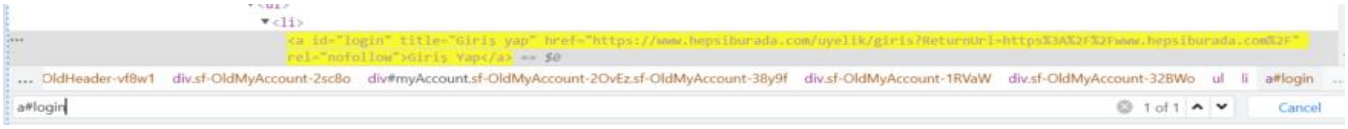
Tag ve ID kullanarak öğeyi bulmak için üç bileşen kullanırız.

Syntax: css = (Html etiketi) (#) (ID'nin değeri)

Html Etiketi: Bulmak istediğimiz etiketi sağlamak için kullanılır, örnek giriş etiketi.

#: Bu işaret ID değerini temsil etmek için kullanılır. Bir öğeyi CSS seçici aracılığıyla kimlik yoluyla bulmak istediğinizde, aynı şekilde bir # işaretinin olması gerektiğini unutmayın. Diğer özellikler için # işaretini kullanmamız gerekmez.

ID: Bulmak istediğimiz elementin kimlik değerini temsil eder.

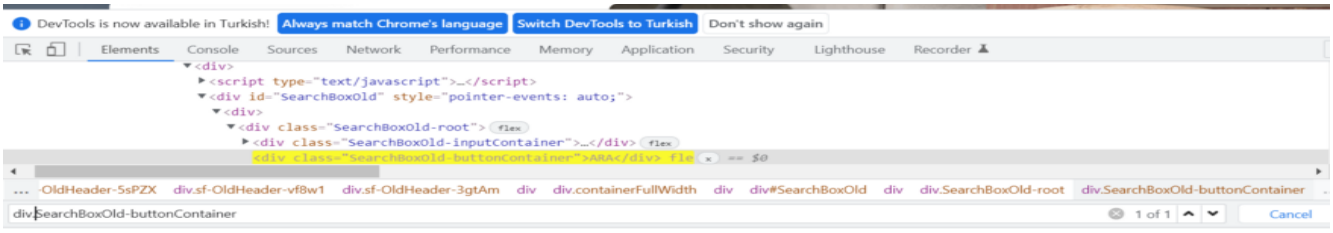


driver.findElement(By.cssSelector("a#login"))

2.2) Tag ve Class

Bu locator da ID gibi çalışır tek farkı syntaxında # işareti yerine nokta kullanılır.

Syntax: css = (Html etiketi) (.) (Class'ın değeri)



Bu alanı Tag ve Class ile bulmak için;

driver.findElement(By.cssSelector("div.SearchBoxOld-buttonContainer"))

2.3)Tag ve Attribute

Bu yöntemde de elementin tag name'i ve elemente özel olan bir özelliği(type, name vb.) kullanılarak css selectoru yazılır. Birden fazla öğenin aynı etikete ve niteliğe sahip olması durumunda, ilk öğe seçilecektir.

Syntax: css= (HTML Tag)[Attribute=Value]

2.4)Tag, Class ve Attribute

Bu locator; Tag, Class ve seçilen bir Attribute değerinin kombini olarak yazılır.

Syntax: css=(HTML Tag)(.)(Class)([attribute='Value of attribute'])

Görseldeki Facebook Şifre alanı Css Selectorunu Tag, Class ve Attribute kullanarak yazalım.

driver. findElement(By.cssSelector(“input. inputtext login_form_input_box [name = ‘pass’]”))

2.5) Elementleri Eşleşme ile bulma(Matches)

Elementleri bulmak için her zaman ilgili Attribute değerinin tamamını yazmaya gerek yoktur. Başını, sonunu yada içerdiği bir bölümü yazarak da css selectorunu oluşturmak mümkündür. Bunu bazı semboller kullanarak yapabiliriz.

2.5.1) Starts With (^)

Syntax: css=(HTML tag)([attribute^=’start of the string’])

Element şu şekilde tanımlanmış olsun;

Bu elementin Css Selectorunu Starts With yöntemiyle yazalım;

driver.findElement(By.cssSelector(“input[name^=’em’]”))

2.5.2)Ends With ($)

Syntax: css=(HTML tag)([attribute$=’end of the string’])

Bu elementin Css Selectorunu Ends With yöntemiyle yazalım;

driver.findElement(By.cssSelector(“input[name$=’ail’]”))

2.5.3) Contains (*)

Syntax: css=(HTML tag)([attribute*=’partial string’])

Bu elementin Css Selectorunu Contains yöntemiyle yazalım;

driver.findElement(By.cssSelector(“input[class*=’control’]”))

2.6) Child Elementler

Child Elementler, başka bir elementin altında tanımlanan elementlerdir. Genellikle bir liste yada tablonun altındaki verileri ulaşmak istediğimizde kullanırız.

Syntax: Css= tagname . class name li : nth-of-child(child element indexi)

Kategoriler Listesi görselindeki gibi tanımlanmış bir listemiz olsun

```
▼<ul class="Giyim">
  ▶<li class="Kadın">…</li>
  ▶<li class="Erkek">…</li>
  ▶<li class="Cocuk">…</li>
```

Kategoriler Listesi

Bu listedeki ‘Erkek’ kategorisine erişmek için yazağımız css selector;

driver.findElement(By.cssSelector(“ul. Giyim li:nth-of-child(2)”);