

# Web Scraping with BeautifulSoup and Selenium

**Web Sitesinin Kullanım Şartları:** Birçok web sitesi, web kazımanın açıkça yasaklandığını belirten bir hizmet şartları veya kullanım şartlarına sahiptir. Bir web sitesinin hizmet şartları web kazımanın izin verilmediğini açıkça belirtiyorsa, bu şartlara saygı göstermeli ve bu web sitesini kazımdan kaçınmalısınız.

**Robots.txt Dosyası:** Web siteleri genellikle arama motorlarının ve diğer web kazıyıcıların hangi bölümlerini taramasına izin verildiğini veya kısıtlandığını iletmek için bir robots.txt dosyası kullanır. Bir web sitesinin robots.txt dosyasını kontrol ederek kazımanın izin verilip verilmediğini görmelisiniz.

([www.example.com/robots.txt](http://www.example.com/robots.txt))([www.miuul.com/robots.txt](http://www.miuul.com/robots.txt))

**API'lar:** Birçok web sitesi, verilerine yapılandırılmış ve yasal bir şekilde erişmenizi sağlayan Uygulama Programlama Arayüzleri (API'lar) sunar. API'ları kullanmak genellikle veri çıkarmak için tercih edilen bir yöntemdir.

Web kazıma, internet üzerindeki web sitesinden veri toplama işlemidir.

API, farklı yazılım ve uygulamalar arasında bilgi ve işlem paylaşımını sağlayan bir arayüzdür. Biz API'lar kullanarak da verileri hızlı ve etkili bir şekilde alabiliriz. Peki o halde neden API yerine web kazıma kullanılır sorusu sorulabilir.

- İstenilen veriyi sunan bir API olmayabilir.
- API olsa dahi istenilen verinin tamamı sunulamayabilir.
- API maliyeti daha fazladır.

Veri bilimi, iş zekası, rekabetçi araştırma, pazar analizi ve raporlama gibi birçok disiplin web sitelerinden veri toplamak ve bu veriyi analiz etmekten büyük fayda sağlayabilir.

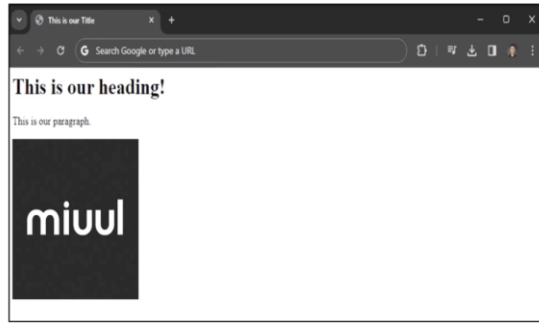
Microsoft, amazon ve google gibi büyük firmalar web kazımayı sıklıkla kullanmaktadır.

## HTML (HYPER TEXT MARKUP LANGUAGE)

Html, web sayfalarının oluşturulmasında kullanılan bir işaretleme dilidir. Htmlde iki ana yapı vardır: head ve body kısmı. Head, sayfa ayarlarının bulunduğu kısımken bod kısmı sayfa içerisinde gördüğümüz kısımlardır. Headi tam anlamıyla görüntüleyemediğimiz, body sayfada direkt olarak gördüğümüz içerik kısımlarını oluşturur.

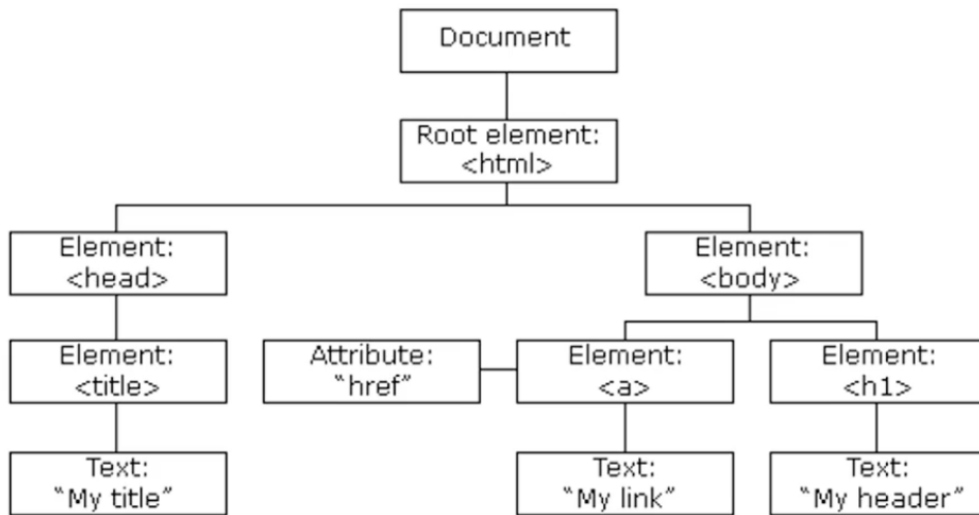
```
<head>
1 <title>This is our title</title>
2
3
4
5
6
7
8

<body>
1 <h1>This is our heading!</h1>
2 <p>This is our paragraph.</p>
3 
4
5
6
7
8
```



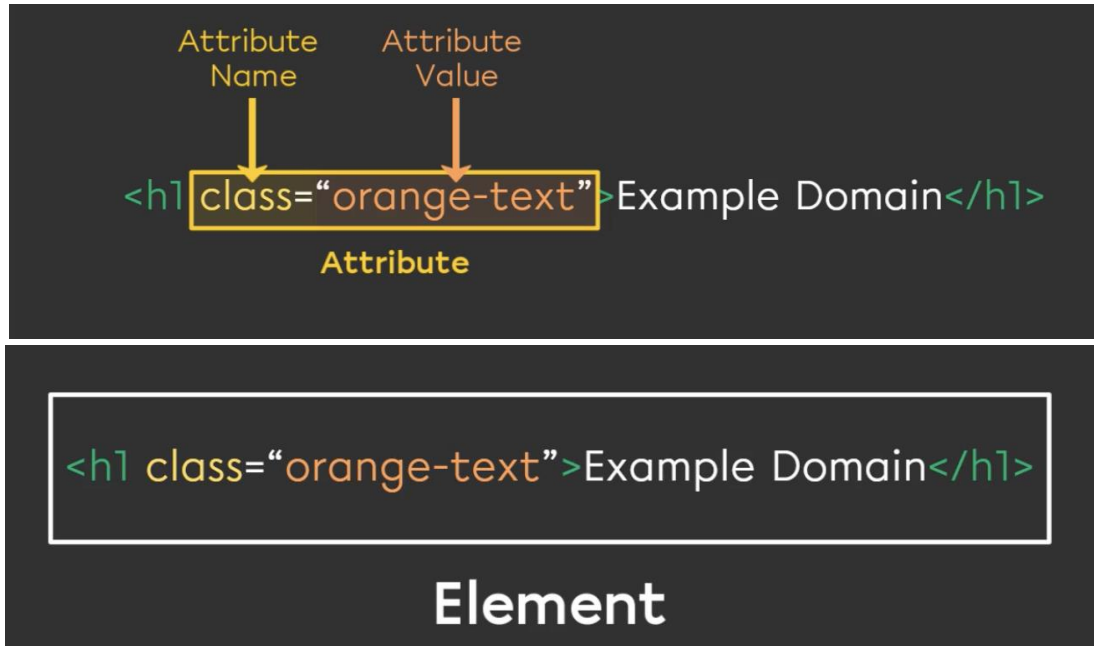
## HTML'in yapısı: DOM (Document Object Model)

Dom, bir web sayfasının yapısını hiyerarşik bir ağaç yapısı olarak temsil eder.



Sayfanın dom'unda bulunan her şeyi web kazanımıyla elde edebiliriz.

## HTML ETİKETLERİ VE ELEMENTLERİ



- **H** etiketi, başlıklar için kullanılır.
- **P** etiketi, paragraf gibi yazılan metinler için kullanılır.
- **Div** etiketi, çerçeveye gibi diğer etiketleri tek bir çatı altında tutup sınıflandırmak için kullanılır.
- **A** etiketi, linkler için kullanılır.
- **Ul** sıralı listeyi, **ol** sırasız listeyi, **li** ise listedeki elemanları temsil eder.
- **Table**, veri biliminde kullanılan dataframedir. Yapısal veriyi sayfada görmeyi sağlayan çerçeveye.
- **Tr**, bu çerçevedeki her bir satır; **td** ise sütunlardır.
- **Img** ise sayfadaki görselleri tutan etikettir.
- **Button** etiketi, tıklama için kullandığımız butonları sağlar.
- **Iframe**, herhangi bir sayfayı başka bir sayfa içerisinde görmemizi sağlar.

## CSS VE JAVASCRIPT

CSS, web sayfalarının görünümünü düzenlemek için kullanılır. Renkler, yazı tipleri, boyutlar ve düzenlemeleri kontrol eder. CSS web kazıma işlemlerinde önemli bir yeri vardır. Örneğin bir haber sitesine girildiğinde yalnızca başlıkları bulmak istiyorsak (genellikle başlıklar koyu ve büyük yazılmıştır) hedeflediğimiz şey bu başlığa uygun biçimlendirilmiş yazıları bulmak ve almak olacaktır.

Javascript, web sayfalarını etkileşimli hale getiren bir programlama dilidir. Veri kazıma için önemli bir süreçtir. Elementler ile etkileşim işlemi JS sayesinde yapılır. Örneğin sayfanın aşağı veya yukarı kaydırıldığı durumlarda JS kodu yazılması gerekmektedir.

**Bilgi:** Çoğu sayfada h1 etiketinden sadece bir tane bulunur. Bunun sebebi, Google gibi arama motorları kendini daha iyi hale getirmek için sitelerin sayfalarını tarar. O sayfaların ne ile ilgili olduğunu h1 etiketine bakarak anlar. H1 etiketinden sadece bir tane bulmasını sever ve bunu yapan sayfaların puanlarını artırır. Yani arama motoru optimizasyonuna dikkat eden sitelerin herhangi bir sayfasına bakıldığında yalnızca bir tane h1 etiketi olduğu görülebilir.

## XPath ve CSS Seçicilere Giriş

Sektörde en sık görülen element bulma yöntemlerinden XPath ve CSS Seçiciler sayesinde elementleri farklı yollardan buluyor ve istenilen içeriğe erişiyor olunabilecek.

### XPath (XML Path Language):

XPath, XML veya HTML belgelerinde belirli öğeleri bulmak ve seçmek için kullanılan bir dil ve yol belirleme tekniğidir. Ağaç yapısı kullanarak öğeleri belirler ve sorguları bu yapıya göre oluşturur. Belirli bir syntaxı vardır. ‘Şununla başlayan’ veya ‘içerisinde şunu barındıran’ gibi fonksiyonlara da sahip olduğundan element bulma konusunda kuvvetli olmaktadır.

//p yazarak p elementlerine erişim sağlanabilir.

```
//p
```

```
<html>
  <head>
    <title>Miouul XPath and CSS Selectors</title>
  </head>
  <body>
    <h1>This is the h1 element</h1>
    <p class="text-red">This is the first paragraph</p>
    <p>This is the second paragraph</p>
  </body>
</html>
```

Yalnızca ilk p elementine erişilmek isteniyorsa

```
//p[@class='text-red']
```

```
<html>
  <head>
    <title>Miouu XPath and CSS Selectors</title>
  </head>
  <body>
    <h1>This is the h1 element</h1>
    <p class="text-red">This is the first paragraph</p>
    <p>This is the second paragraph</p>
  </body>
</html>
```

Div'in içerisinde p elementlerine ulaşmak istenirse

```
//div//p
```

```
<html>
  <head>
    <title>Miouu XPath and CSS Selectors</title>
  </head>
  <body>
    <h1>This is the h1 element</h1>
    <div>
      <p class="text-red">This is the first paragraph</p>
      <p>This is the second paragraph</p>
    </div>
    <p>Outside of container</p>
  </body>
</html>
```

## CSS Selectors:

CSS Seçiciler, web sayfalarındaki HTML öğelerini seçmek ve stil uygulamak için kullanılan bir yöntemdir.

p.text-red

```
<html>
  <head>
    <title>Miouul XPath and CSS Selectors</title>
  </head>
  <body>
    <h1>This is the h1 element</h1>
    <p class="text-red">This is the first paragraph</p>
    <p>This is the second paragraph</p>
  </body>
</html>
```

Nokta classı temsil etmektedir. Id'si p olan etiketleri bulmak için p.text yerine p#text yazılmalıdır.

XPath'de CSS seçicilere kıyasla nokta veya hashtag kullanmadan kelimelerle istenilen öğelere erişim sağlanabilmektedir, bu da okunurluk açısından daha iyi bir durumdur denilebilir. İki yöntemle de ileri seviyede sorgular yazılarak istenen elementlere ulaşılabilir.

### Web kazımada kullanılan CSS Seçiciler

- **Simple Selectors:** Öğeleri adlarına, id'lerine veya class'larına göre seçmek için kullanılırlar. Id için #, text için ise nokta (.) kullanılır. Element sadece adına göre seçilmek istendiğinde ise direkt olarak elementin adı girilir.
- **Combinator Selectors:** Elementleri diğer elementlerle olan özel ilişkilerine ("div içerisindeki p" gibi) göre seçmek için kullanılırlar. Kapsayıcı elementin adı boşluk kapsanan element şeklinde yazılır. Örneğin 'div p'. 'Şu elementin kardeş elementi' veya 'şu elementten sonra gelen element' şeklinde kullanımları da mevcuttur.
- **Attribute Selectors:** Elementleri attribute'u olup olmamasına göre veya attribute değerine göre seçmek için kullanılırlar.

@class='text-red' --> 'classı şu olan attribute' şeklinde kullanım

@class='text-red' --> 'attribute değerinde bu kelime geçen' şeklinde kullanım

## XPath'in Sytnaxı

İnternette araştırıldığında elementlere 'node' de dendiği görülebilir. Etiket, element, node hepsi bulmak istediğimiz öğeyi temsil eder.

- **Node, Element, Tag:** Bulmak istediğimiz öğe
- **Attribute:** Elementlerin özellikleri. (class, id, href gibi..)
- **Text:** Element içerisindeki asıl içerik
- **Child:** Bir elementin içerisindeki element
- **Parent:** Elementi kapsayan element. İçerisinde bulunduğumuz element.
- **Ancestor:** Tüm atalarımız, yani elementimizi içerisine alıp kapsayan tüm elementler.

```
<div class="parent">
  <div class='sibling'></div>
  <div class="child">I'm the child</div>
</div>
```

ancestor

```
<div class="grandparent">
  <div class="parent">
    <div class="child">
      <p>This is the target element.</p>
    </div>
  </div>
</div>
```

```
<div class="grandparent">
  <div class="parent">
    <span class="foreigner">Hi</span>
    <div class="child">
      <p>This is the target element.</p>
    </div>
  </div>
</div>
```

span bizi ilgilendirmiyor.

- **//:** Belgenin herhangi bir yerindeki elementleri aramak

için kullanılır

- **/:** Sadece child elementleri aramamızı sağlar.
- **elementName:** Bu element adına sahip olan tüm elementleri seçer
- **@attributeName:** Bir attribute'u seçer. (Elementin < ve > işaretleri arasında kalan herhangi bir özelliği temsil eder.)
- **text():** Bir elementin içindeki metni/içeriği seçer.

```
//div[@class='child']/p/text()
```

Yukarıdaki görsel incelendiğinde ilk kısımda çift slash ile tüm doküman içerisinde class attribute'ı child olan div var denmiştir. Sonrasında ise bu div'in p adında bir çocuğunun textini getirir misin demek istenilmiştir.

```
<div class="grandparent">
  <div class="parent">
    <div class="child">
      <p>This is the target element.</p>
    </div>
  </div>
</div>
```

```
//div[@class='child']/p/text()
```

Bunun yanı sıra '//p/text' de denilebilirdi fakat yapıda birden fazla p olsaydı bulunan tüm p'ler gelecekti. En yalın haliyle istenen p'nin gelmesi için gereken kod yukarıdaki gibidir.

## XPath Fonksiyonları ve Operatörleri

- "|" (Pipe) Operatörü: Birden fazla elementi aynı anda seçmek için kullanılır. Örnekte tüm h2 başlıklarının ve tüm paragrafların seçildiği gözlemlenebilir.
- **and Operatörü:** İki farklı attribute koşulunun tek sorguda yazılabilmesini sağlar.
- **or Operatörü:** Veya gibi çalışma gösterir.

▪ "|" (Pipe) Operatörü

```
//h2 | //p
```

▪ and Operatörü

```
//input[@type='text' and @name='username']
```

▪ or Operatörü

```
//a[@class='external' or @target='_blank']
```

**XPath Fonksiyonları**, diğer dillerde olduğu gibi built-in (gömülü) fonksiyonlardır. Gömülü fonksiyonu hatırlamak gerekirse örneğin python'da bir listenin uzunluğunun görülebilmesi için 'len()' fonksiyonu kullanılır veya python'da bir değişken stringe çevrilmek istendiğinde 'str()' kullanılır. Bunlar python ile birlikte gelen ve herhangi bir paket yüklenmeden kullanılabilecek gömülü fonksiyonlardır.

- **contains()** : 'Bir elementin x kısmında y kelimesi geçiyor mu?' sorusunun cevabının alınmasını sağlayan fonksiyon. Örnek olarak ürünlerin bulunduğu bir sayfaya girilip arka planı açılıp ürünlerin tutulduğu elementler incelendiğinde,



ürünlerin classında 'products container' yazıldığı görülürken öne çıkarılan ürünlerin classında ise 'promoted-products container' yazdığı görülüyor. Böyle bir durumda tüm ürünler alınmak istendiğinde contains ile birlikte classı 'products' stringini barındıran elementleri getir denir. Sadece öne çıkarılan ürünler alınmak istenirse classında 'promoted-products' içeren stringleri getir denilebilir.

- **Starts-with():** Classındaki değer 'promoted-products' ile başlayan elementleri getir denilebilir. İşleyiş bakıldığında contains ile aynı.

Contains ve starts-with fonksiyonları sayesinde herhangi bir attribute veya texti spesifik bir string ile başlayan ve o stringi barındıran elementler yakalanılabilir. Bu fonksiyonlar iki tane argüman isterler. Birinci argümanda aranılan stringin (classta mı, idde mi, textte mi gibi) nerede aranıldığı belirtilir, ikinci argümanda ise aranılan string yani metnin kendisi belirtilir.

- contains(@class, 'products')
- starts-with(@class, 'products')

- **Not():** X elementi isteniyor fakat şöyle bir şart var 'y elementini kapsamassın'

```
<div class="container">
  <p>This is some <b>bold</b> text.</p>
  <p>Another paragraph with
  <a href="https://example.com">a link</a>.
</p>
</div>
```

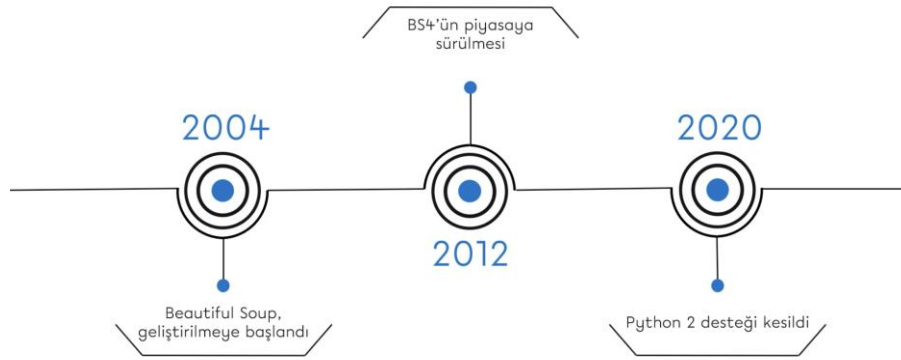
Yukarıdaki kodda ilk p etiketi b'nin parenti, ikinci p etiketi ise a'nın parenti durumunda. Böyle bir durumda b adında childi olmayanın seçilmesi için '//p[not(b)]' sorgusu yazılabilir.

- **Last():** Ul'un içindeki li'lerden sonuncusu bu fonksiyon ile getirilebilir. Index kullanılmadan bu iş halledilir.
- **Text():** Örneğin div etiketinin textinin bulunması istenildiğinde '//div/text()' şeklinde sorgu yazılabilir.

Tarayıcı geliştirici araçlarla web sayfaları incelenirken sağ tık incele seçeneğiyle yan tarafta açılacak olan konsol alanıyla inceleme gerçekleştirilebilir. Bu zamana kadar öğrenilen XPath yöntemleriyle sorgulama yapılmak istendiğinde ise yan tarafta açılan alanda herhangi bir elementin üstüne tıkladıktan sonra 'command+F' (windows kullanıcıları için ctrl+f) yaparak çıkan arama sekmesi benzeri çubukta istenilen sorgular

yapılabilir. Aynı zamanda bu alanda sağ tık yaparak 'add attributes' seçeneğinden eklemeler veya düzenlemeler yapılabilir.

## BEATIFUL SOUP'A GİRİŞ



- **Kolay Kullanım:** BeautifulSoup'u kullanmak oldukça basittir.
- **HTML ve XML Desteği:** BeautifulSoup, HTML ve XML belgelerini işlemek için özel olarak tasarlanmıştır.
- **Güçlü Veri Ayıklama:** BeautifulSoup ile web sayfalarındaki metin, bağlantılar, resimler ve diğer öğeler kolayca ayıklanabilir.
- **Dinamik Element Eksikliği:** Sayfalarda sonradan yüklenen, yani dinamik olarak gelen bilgileri alamıyoruz.
- **Etkileşim:** Sayfalar ile etkileşim halinde olamıyoruz.

Beatifulsoup() iki tane argüman ister. Bunlardan ilki içeriğin ne olduğu (html sitesi), ikincisi ise beatifulsoup hem xml hem de html içeriklerinde çalışabilir; argüman hangisi olduğunu belirtmeye yarar (ör "html.parser").

Örneğin; `soup = BeautifulSoup(html, "html.parser")`. Uygulamalı örnekler .py uzantılı dosyalarda mevcuttur.

## SELENIUM'A GİRİŞ

Beatifulsoup'un yetersiz kaldığı zamanlarda yani dinamik olarak yüklenen web sayfalarının nasıl kazınıldığı görülecek.

Selenium, web tarayıcılarını otomatize etmek için kullanılan açık kaynaklı bir test otomasyon aracıdır. Seleniumun doğru amacı, web uygulamalarını test etmektir. Zamanla web kazanım dünyasında da önemli bir rol oynamaya başladı.

## Neden Selenium?

- JavaScript ile etkileşim
- Tarayıcıyı otomatik olarak kontrol eder
- Popüler

Selenium'un tek **dezavantajı** BeautifulSoup'a göre daha yavaş olmasıdır diyebiliriz.

Webdriver, istenilen tarayıcıyı (chrome, firefox, edge..) istenilen seçeneklerle açmaya yarar.

## HEADERS VE PROXY

Headers, bir web tarayıcısı ile iletişimde bulunurken gönderilen bilgilerdir. Bu bilgilerde kullanıcının tarayıcı türü, dil tercihi, isteğin tipi gibi bilgiler bulunabilir. Web kazınması sırasında uygun headers kullanılarak tarayıcı gibi davranılabilir, web sitesinden veri çekerken tarayıcıya benzer bir izlenim bırakılabilir. Bu, web sitelerinin botları engelleme veya tarayıcı benzeri davranışları algılama girişimlerini atlatmaya yardımcı olabilir.

Proxy, bir ağdaki istemcinin, sunucuyla iletişimini sağlayan bir araçtır. Proxy'nin kullanılma sebebi, gizliliği artırılabilir olmasıdır ve içerikleri yönetmeye olanak tanır. Farklı bir ip adresinden bağlantı yapılmaya çalışılıyor gibi görünülmesini sağlar, bu da gizliliği artırır. Coğrafi sınırlamaları ortadan kaldırır.

Örnek headers kullanımı:

```
1 headers = {  
2     "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64).....",  
3     "Accept-Language": "en-US,en;q=0.5"  
4 }
```

- User-Agent ile tarayıcımızın Mozilla olduğu belirtildi
- Accept-Language ile tercih edilen dilin en-US olduğu belirtilerek içeriklerin İngilizce gelmesi sağlandı.

Örnek proxy kullanımı:

Örneğin popüler bir film sitesi var. Bu siteye gelen yeni filmlerin bilgisini almak istiyoruz. Bu site farklı bölgelerdeki kullanıcılara farklı içerikler sunuyor (örneğin Avrupa'daki bir kullanıcı içeriklere erkenden ulaşabilirken başka yerlere ulaşması uzun sürüyor).

```
1 proxy = {"http": "http://192.0.0.0:0000", "https":  
           "http://192.0.0.10:0000"}
```

- Coğrafi sınırlar kaldırılmak için Avrupa Sunucusunda bulunan proxy kullanıldı.

Proxy'yi kullanmak için satın almak gereklidir.

**Bilgi:** Seleniumda kimlik doğrulama olmadan (user name ve password olmadan) proxy kullanmak kolaydır fakat işin içine user name ve password girdiğinde bu yöntem çalışmaz. Başka bir yöntem uygulamak gerekmektedir (extension.py uzantılı dosyada).

## **BOT TESTLERİNİ GEÇMEK İÇİN UNDETECTED-CHROMEDRIVER**

## **DEĞERLENDİRME SORULARI**

Beautiful Soup ile bir proxy kullanmak için aşağıdakilerden hangisi gereklidir?

Yalnızca en uygun TEK bir yanıt seçin.

- ☒ A requests kütüphanesi
- ☐ B urllib modülü
- ☐ C proxies modülü
- ☐ D dotenv kütüphanesi



Selenium ile proxy kullanmak için requests kütüphanesine ihtiyaç duyulur.

Yalnızca en uygun TEK bir yanıt seçin.

- ☐ A Doğru
- ☒ B Yanlış


Bir HTML sayfasındaki tüm <div> elementlerini  
CSS Seçici ile seçmek için hangi ifadeyi  
kullanırsınız?

Yalnızca en uygun TEK bir yanıt seçin.

- ☐ A .div
- ☒ B #div 
- ☐ C div 
- ☐ D \*div\*

## Bir elementin class'ına göre CSS Seçici ifadesi hangi şıkta doğru verilmiştir?

Yalnızca en uygun TEK bir yanıt seçin.

- A** `class:elementClass`
- B** `elementClass`
- C** `#elementClass`
- D** `.elementClass` 

## Beautiful Soup kullanarak bir elementin bir önceki kardeşini seçmek için hangi ifadeyi kullanırsınız?

Yalnızca en uygun TEK bir yanıt seçin.

- A** `element.find_previous_sibling()`
- B** `element.find_previous()`
- C** `element.find_previous_element()`
- D** `element.find_previous_sibling("element")`

## Selenium ve XPath kullanarak id'si name olan elementin altındaki tüm çocukları seçmek için hangi ifadeyi kullanırsınız?

Yalnızca en uygun TEK bir yanıt seçin.

- |   |   |   |
|---|---|---|
| A | <code>//*[@id="name"]/child::p</code>           |   |
| B | <code>//*[@id="name"]/*</code>                  | ✓ |
| C | <code>//*[@id="name"]/descendant::*</code>      |   |
| D | <code>//*[@id="name"]/descendant::node()</code> |   |

## Soup objesi ile bir img öğesinin src özelliğini almak için aşağıdakilerden hangisi doğrudur?

Yalnızca en uygun TEK bir yanıt seçin.

- |   |  |   |
|---|--|---|
| A | <code>img["src"]</code>                | ✓ |
| B | <code>soup.get_attribute("src")</code> | ✗ |
| C | <code>img["href"]</code>               |   |
| D | <code>img.get_attribute("href")</code> |   |

## Selenium ile ul elementi içerisinde aşağı doğru kaydırma yapmak için aşağıdakilerden hangisi kullanılır?

Yalnızca en uygun TEK bir yanıt seçin.

- A send\_keys()
- B action\_chains()
- C scroll()
- D click()

## Aşağıdaki seçeneklerden hangisinde proxy için syntax doğrudur?

Yalnızca en uygun TEK bir yanıt seçin.

- A username:password@port:ip\_address
- B ip\_address:port@:username:password
- C username:password@ip\_address:port ✓
- D port:ip\_address@username:password

Text'inde name geçen ve p olmayan element'i bulmak için aşağıdaki XPATH sorgularından hangisi kullanılmalıdır?

Yalnızca en uygun TEK bir yanıt seçin.

- A `//*[contains(text(),'name') and not(p)]` ✗
- B `//*[contains(text(),'name') and not(self::p)]` ✓
- C `//*[starts-with(text(),'name') and not(p)]`
- D `//*[contains(text()='name') and not(self::p)]`



**Selenium ve Beautiful Soup birlikte çalışabilir.**

Yalnızca en uygun TEK bir yanıt seçin.

<b>A</b>	Doğru
<b>B</b>	Yanlış