Data Journalism

Web Scraping - Selenium

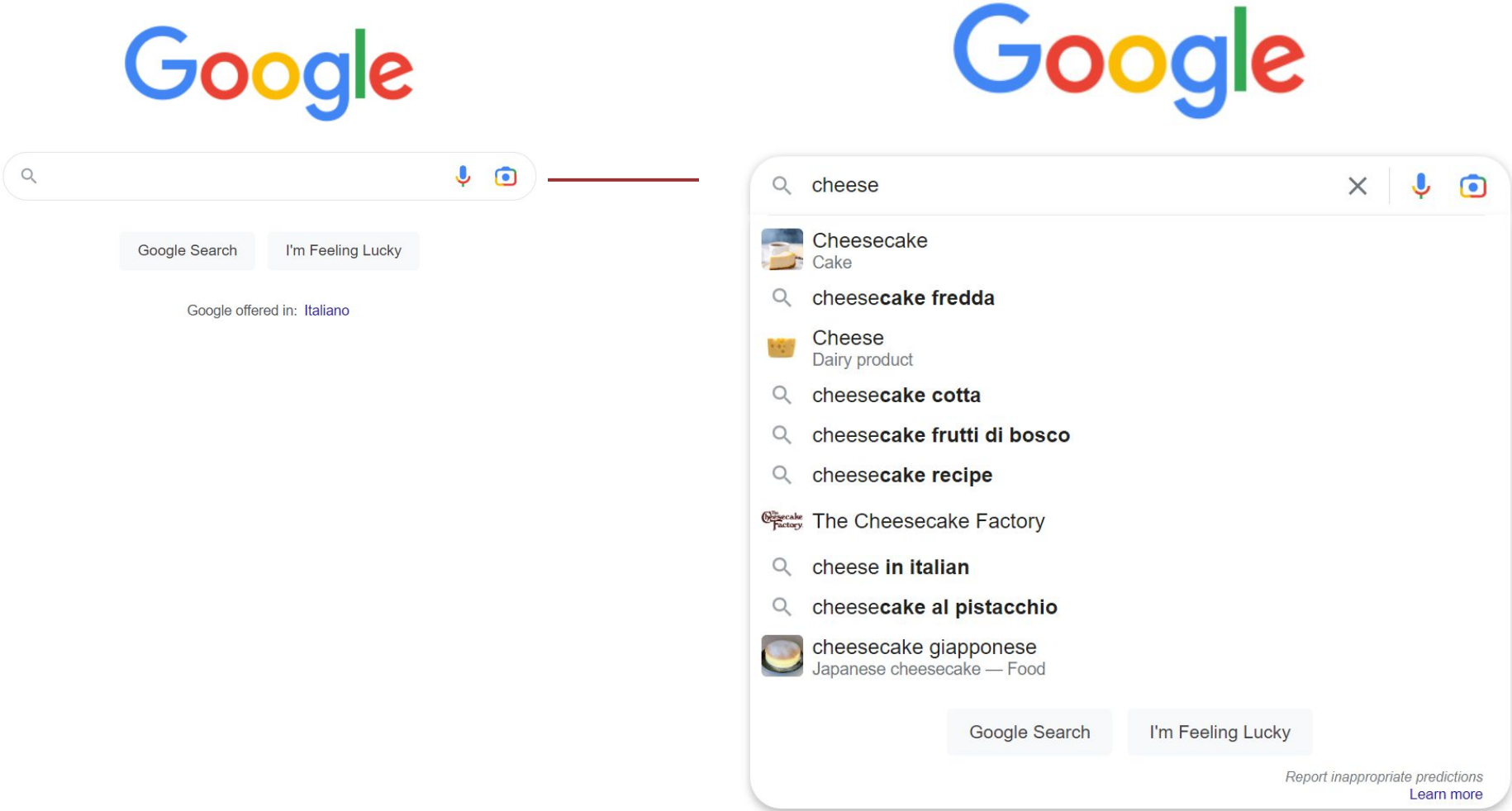
InfoUma 2023 ***Andrea Marchetti***

Selenium WebDriver

Selenium è un framework per automatizzare il comportamento di un browser

Posso scrivere un programma per

- aprire una finestra del browser,
- navigare in una pagina web
- compilare campi di input
- fare click sui pulsanti
- gestire le finestre di dialogo


Google
Google Search
I'm Feeling Lucky
Google offered in: Italiano
Google
cheese
Cheesecake
Cake
cheesecake fredda
Cheese
Dairy product
cheesecake cotta
cheesecake frutti di bosco
cheesecake recipe
The Cheesecake Factory
cheese in italian
cheesecake al pistacchio
cheesecake giapponese
Japanese cheesecake — Food
Google Search
I'm Feeling Lucky
Report inappropriate predictions
Learn more

Google

cheese

Sign in

All | Images | Shopping | Maps | News | More | Tools

About 1,970,000,000 results (0.48 seconds)

Results for **57014 Collesalvetti LI** · Choose area

Wikipedia
https://en.wikipedia.org › wiki › Cheese

Cheese - Wikipedia

Cheese is a dairy product produced in wide ranges of flavors, textures, and forms by coagulation of the milk protein casein. It comprises proteins and fat ...

Types of cheese · History of cheese · Cheddar cheese · List of

People also ask

- What are the top 10 cheeses?
- What are 7 types of cheese?
- What are the 20 types of cheese?
- Is cheese Italian or French?

Feedback

Cheese.com
https://www.cheese.com

Cheese.com - World's Greatest Cheese Resource

The database includes information on most famous sorts of **cheese** such as Cheddar, Camembert, Stilton or Parmesan, as well as rarities like Crottin de Chavignol.

Alphabetical list · American Cheese · Cheese and Wine Pairings · Cheeses by type

slowfood.it
https://cheese.slowfood.it › ...

Cheese: Home

The world's largest and most important festival dedicated to raw milk **cheese** and dairy products is in Bra, Italy, from September 17 to 20.

Encyclopedia Britannica
https://www.britannica.com › ... › Food

Cheese | Description, Types, & Production | Britannica

cheese, nutritious food consisting primarily of the curd, the semisolid substance formed when milk curdles, or coagulates. Curdling occurs naturally if milk ...

Sponsored · Shop now

Parmigiano Reggiano 36 Mesi | 3kg | Caseificio Bazzanese
€64.80
Shop Parmigiano Reggiano
€21.60/1kg
(€21.60/1kg)
Free shipping
By Google

Parmigiano Reggiano 36 Mesi | 4kg | Caseificio Barani
€74.00
Shop Parmigiano Reggiano
€18.50/1kg
(€18.50/1kg)
Free shipping
By Google

Cheese

Dairy product

Cheese is a dairy product produced in wide ranges of flavors, textures, and forms by coagulation of the milk protein casein. It comprises proteins and fat from milk.
Wikipedia

Make

```
driver = Chrome()
driver.get("http://www.google.com/ncr")
queryInput = driver.find_element(By.NAME,"q")    # trovo nel codice DOM l'elemento che ha il nome "q"
queryInput.clear()                               # cancello eventuali ricerche precedenti
queryInput.send_keys("milk" + Keys.RETURN)       # scrivo "milk" nel campo di input e invio la ricerca
driver.implicitly_wait(10)                       # aspetto che la pagina si ricarichi
result=driver.find_element(By.CSS_SELECTOR,"h3") # trovo il primo risultato
print(result.text)                               # stampo il testo del primo risultato
driver.quit()                                    # chiudo il driver, la pagina sparisce
```

Selenium

Pensato per testare una applicazione web

Ottimo tool per progettare attività di scraping

www.selenium.dev/documentation/webdriver/

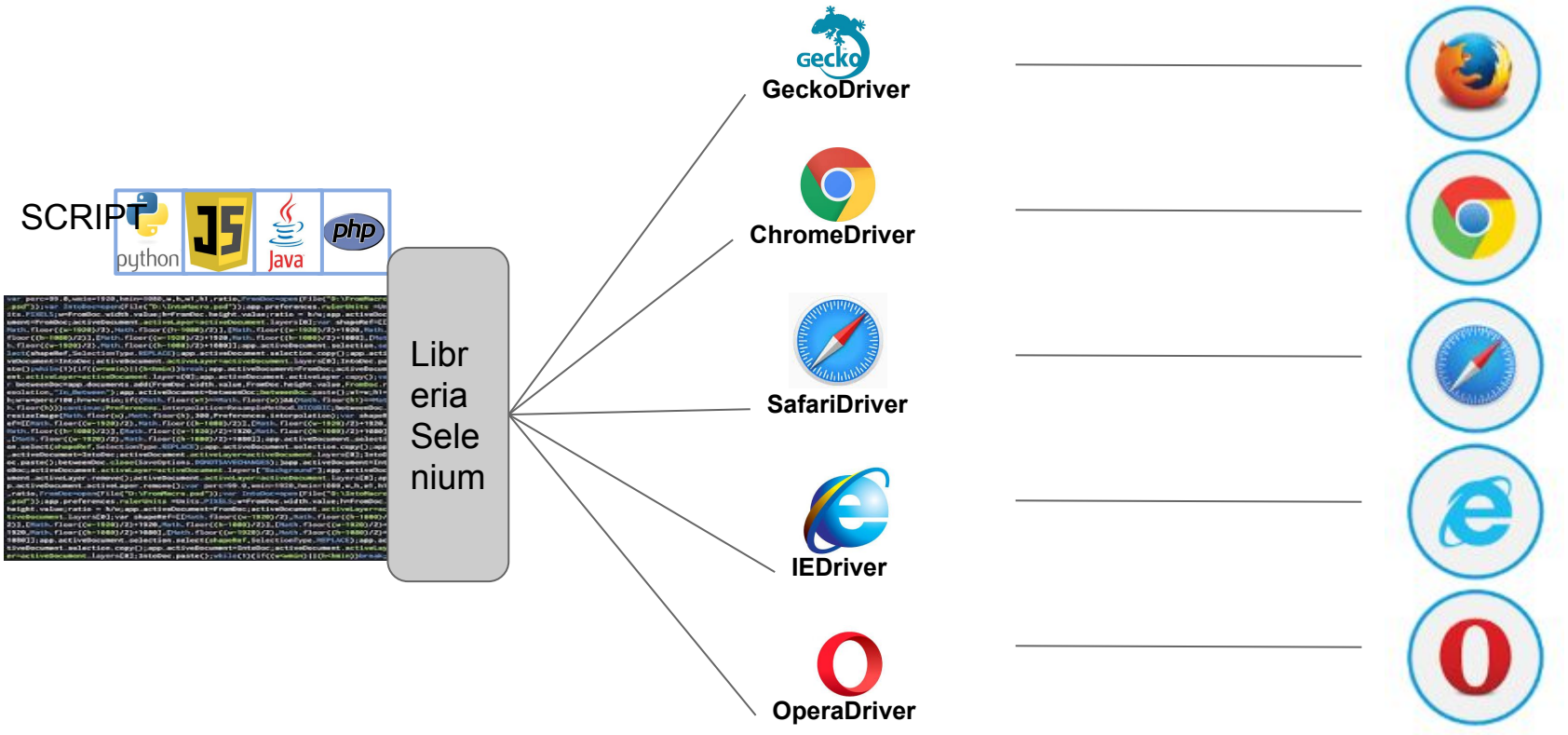
Overview

Installazione

Navigare e interagire con pagine HTML

Gestire le attese asincrone

Script WebDriver Browser



Installazione Libreria

Libreria API per il linguaggio scelto - **Python**

```
pip install selenium
```

[Install a Selenium library]

Installazione WebDriver

Il driver è un eseguibile che va solo piazzato in una directory

occorre assicurarsi che la variablie di sistema $PATH contenga la directory dove inseriamo il webdriver

[Install Web Drivers]

Dove trovare il Web Driver

Browser	Supported OS	Maintained by	Download	Issue Tracker
Chromium/Chrome	Windows/macOS/Linux	Google	Downloads	Issues
Firefox	Windows/macOS/Linux	Mozilla	Downloads	Issues
Edge	Windows/macOS/Linux	Microsoft	Downloads	Issues
Internet Explorer	Windows	Selenium Project	Downloads	Issues
Safari	macOS High Sierra and newer	Apple	Built in	Issues

Release Browser e Driver allineate

chrome://settings/help

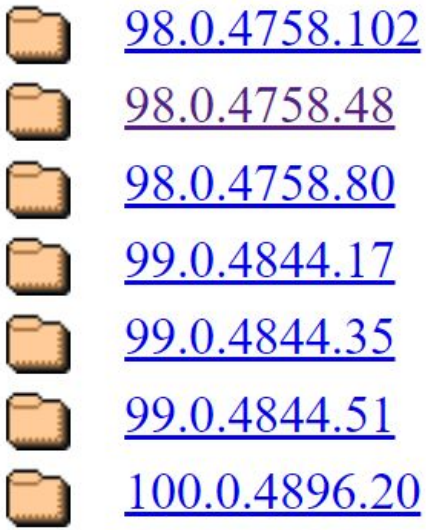
```
Versione 99.0.4844.82
```

da terminale

```
> chromedriver
Versione 97.0.4692.71
```

L'aggiornamento del browser è automatico.

Il driver deve essere aggiornato manualmente !!!



https://chromedriver.storage.googleapis.com/index.html

Lancio del driver da script python

```python
import selenium                          # Importo la libreria Selenium

driver = selenium.webdriver.Chrome()     # lancio il chromedriver di chrome
                                         # che fa partire una pagina vuota di Chrome
```

Possibili messaggi di errore

```
SessionNotCreatedException: Message: session not created: This version of ChromeDriver only supports Chrome version 97
Current browser version is 99.0.4844.82 with binary path C:\Program Files (x86)\Google\Chrome\Application\chrome.exe
```

```
WebDriverException: Message: 'chromedriver' executable needs to be in PATH. Please see https://sites.google.com/a/chromium.o
rg/chromedriver/home
```

Driver dei maggiori browser

```
from selenium import webdriver

# selenium ha le interfacce per i maggiori browser
driverChrome   = webdriver.Chrome()
driverFirefox  = webdriver.FireFox()
driverEdge     = webdriver.Edge()
driverSafari   = webdriver.Safari()
driverOpera    = webdriver.Opera()
driverExplorer = webdriver.Ie()
```

Caricare una pagina

```
from selenium import webdriver.Chrome

# Lancio del driver di Chrome che fa partire una pagina vuota del browser
driver  = Chrome()


# Istruisco il driver di Chrome a caricare una url
driver.get(“http://www.booking.com”)
```

Un primo esempio

```
from selenium import webdriver.Chrome
from selenium.webdriver.common.keys import Keys

driver = Chrome()
driver.get("http://www.google.com")

elem = driver.find_element_by_name("q")
elem.clear()
elem.send_keys("selenium")
elem.send_keys(Keys.RETURN)

driver.quit()
```

Un primo esempio

```
from selenium import webdriver.Chrome
from selenium.webdriver.common.keys import Keys

driver = Chrome()
driver.get("http://www.google.com")

elem = driver.find_element_by_name("q")
elem.clear()
elem.send_keys("selenium")
elem.send_keys(Keys.RETURN)

driver.quit()
```

Lo script, tramite un'istanza di Chrome apre la pagina http://www.google.com e attraverso la searchbar cerca il termine selenium.

Parsing e simulazione tastiera

```
elem = driver.find_element("q")
```

Il WebDriver offre diversi modi per trovare gli elementi presenti nella pagina. In questo caso localizziamo una element input text grazie al suo attributo name.

```
elem.clear()
elem.send_keys("selenium")
elem.send_keys(Keys.RETURN)
```

Per simulare la digitazione da tastiera utilizziamo il metodo send_keys(). Prima però, ci assicuriamo che non ci sia già del testo nell'input text svuotandolo tramite il metodo clear(). Tasti speciali come INVIO possono essere simulati grazie alla classe Keys.

Modalità Headless

Quando si invoca un driver di un browser questo fa partire una pagina vuota del browser che poi verrà pilotata.

Con Chome e Firefox si può evitare questo comportamento con l'opzione Headless

Utile per velocizzare lo scraping

```
from selenium.webdriver import Chrome
from selenium.webdriver.chrome.options import Options

chromeOptions = Options()
chromeOptions.add_argument("--headless")

driver    = Chrome(options=chromeOptions)
driver.get("http://www.repubblica.ir")
```

Documentazione API for Python

https://selenium-python.readthedocs.io/

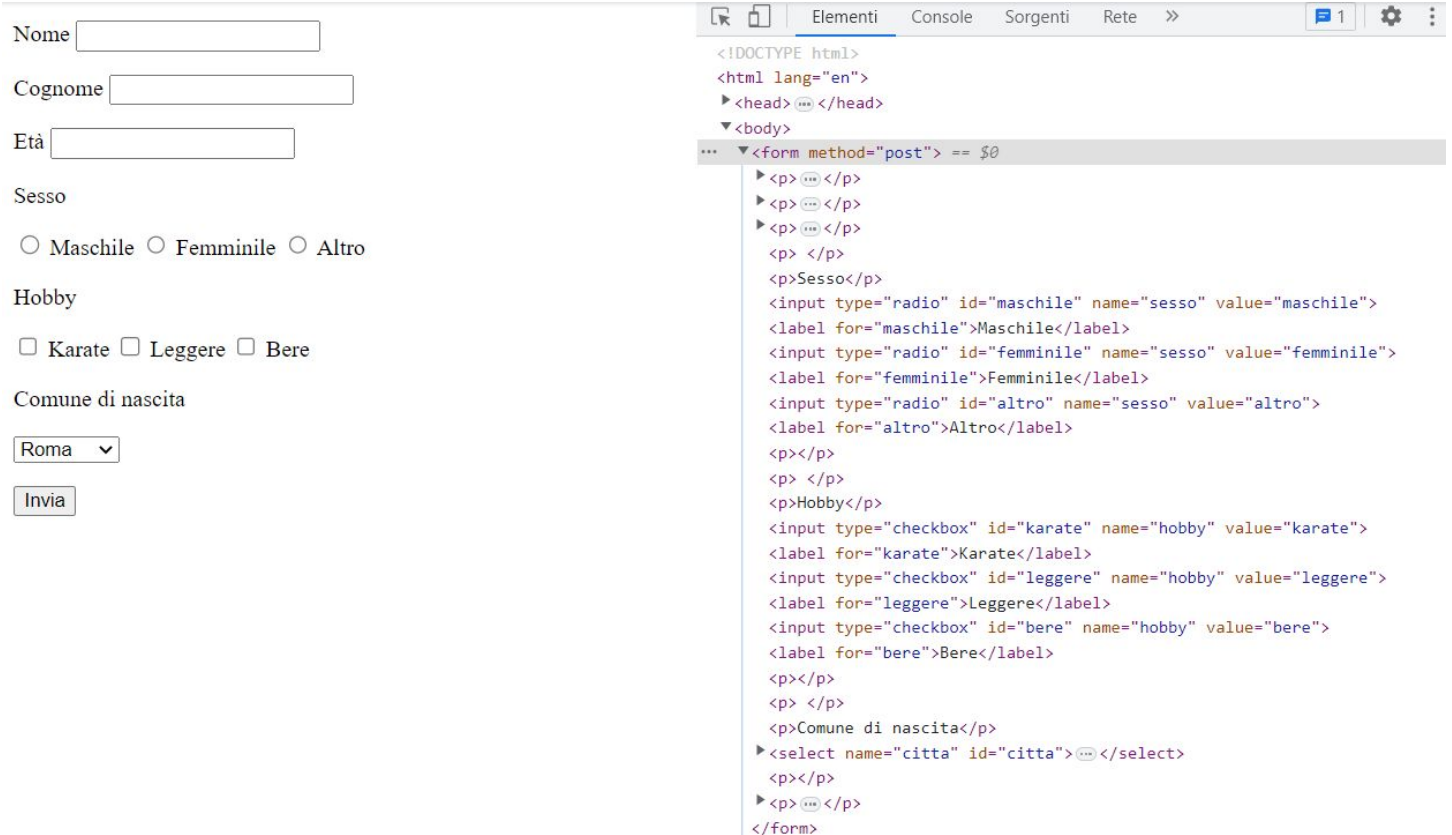
Trovare un elemento nella pagina

Il WebDriver fornisce diverse strategie per localizzare un elemento presente nella pagina.

```
driver.find_element(By.ID, '')
driver.find_element(By.NAME, '')
driver.find_element(By.XPATH, '')
driver.find_element(By.LINK_TEXT, '')
driver.find_element(By.PARTIAL_LINK_TEXT, '')
driver.find_element(By.TAG_NAME '')
driver.find_element(By.CLASS_NAME, '')
driver.find_element(By.CSS_SELECTOR, '')
```

Tutti i metodi restituiscono un oggetto di tipo WebElement. Nel caso non esista un elemento corrispondente ai criteri della ricerca viene generata l'eccezione NoSuchElementException.

Controllare la sorgente HTML



NoSuchElementException

```
# import NoSuchElementException
from selenium.common.exceptions import NoSuchElementException


try:
    elem = driver.find_element(By.ID, "data")
    elem.clear()
    elem.send_keys("01/01/2020")
except NoSuchElementException:
    print("Elemento non trovato")
```

WebElement Methods

Il WebElement è un interfaccia che rappresenta un elemento nella pagina. Ci permette di interagire con l'elemento e di estrarne informazioni attraverso vari metodi.

```
webelement.clear()
webelement.click()
webelement.get_attribute('')
webelement.get_property('')
webelement.is_displayed()
webelement.is_enabled()()
webelement.is_selected()
webelement.screenshot('')
webelement.send_keys('')
webelement.submit()
webelement.value_of_css_property('')
```

WebElement Attributes

È fornito anche di attributi.

```
webelement.id
webelement.location
webelement.location_once_scrolled_into_view
webelement.parent
webelement.rect
webelement.screenshot_as_base64
webelement.screenshot_as_png
webelement.size
webelement.tag_name
webelement.text
```

Trovare più elementi nella pagina

È possibile localizzare anche più elementi contemporaneamente.

```
driver.find_elements_by_name('')
driver.find_elements_by_xpath('')
driver.find_elements_by_link_text('')
driver.find_elements_by_partial_link_text('')
driver.find_elements_by_tag_name('')
driver.find_elements_by_class_name('')
driver.find_elements_by_css_selector('')
```

Tutti i metodi restituiscono una **lista** di oggetti di tipo WebElement.
Nel caso non esista nemmeno un elemento corrispondente ai criteri della ricerca viene restituita una lista di lunghezza 0.

WebElement Navigation

La classe **WebElement** eredita tutti i metodi di localizzazione presenti nella classe **WebDriver**.

Quindi è possibile (consigliato) ricercare un elemento a partire da un elemento trovato

Altro

Drag and drop

Moving between windows and frames

Popup dialogs

Navigation: history and location

Cookies

Attese in Selenium

documentazione

- contenuti diversi possono essere caricati sulla pagina con tempi diversi
  - la diffusa presenza di tecnologie come AJAX rende frequente questa possibilità
- la ricerca di un elemento non ancora presente solleva un'eccezione
- la gestione delle attese durante lo scraping può risolvere questi problemi
- inserire attese può prevenire anche possibili **banning** da parte del server

Selenium WebDriver

Fornisce due tipi di waits

Esplicita

Implicita

WebDriverWait

Selenium WebDriver fornisce due tipi di wait;

Esplicita

Aspetta finché si verifica una certa condizione prima di procedere.

Implicita

Aspetta per un certo periodo e poi prosegue

WebDriverWait Esplicita

```
from selenium.webdriver.support.ui import WebDriverWait

elem = WebDriverWait(driver, tempo).until(condizione)
```

nel dettaglio, l'esecuzione riprende se:

- si verifica la condizione specificata
- si raggiunge un tempo massimo per l'attesa

Expected Conditions

- è possibile specificare la propria condizione andando a definire delle classi Python opportune
- sono già previste una serie di condizioni di uso comune che possono essere usate direttamente:
  - *visibility_of_element_located*
  - *element_to_be_clickable*
  - *text_to_be_present_in_element*
  - *…*

```
from selenium.webdriver.support import expected_conditions as EC

elem = WebDriverWait(driver, tempo)
```

WebDriverWait Esplicita

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.wait import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC

driver = webdriver.Chrome()
driver.get("url")

try:
    element = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.TAG_NAME, "p"))
    )
    print(driver.find_element(By.TAG_NAME,"p").text)
except Exception as e:
    print(e)
finally:
    driver.quit()
```

WebDriverWait Implicita

```
from selenium import webdriver


driver = webdriver.Firefox()
driver.implicitly_wait(10) # 10 seconds
driver.get("http://somedomain/url_that_delays_loading")
myDynamicElement = driver.find_element_by_id("myDynamicElement")
```