

Replay

Learn More 00:15

U SECURITY

*Security Ultra Thin Regular Absorbency vs Always® Infinity Size 1 Ultra Thin Pads. Time lapse: two minutes between absorbencies. ©2012 4.3ml

Sponsored by Advertising Partner

Sponsored Video

Using XPath and Selenium to Find an Element in HTML Page

Selenium

Watch to learn more

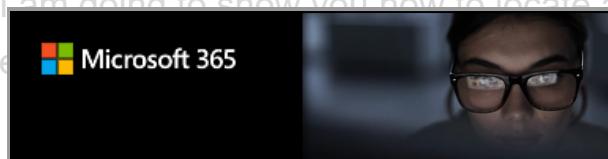
SEE MORE

4 months ago • by Shahriar Shovon

XPath, also known as XML Path Language, is a language for selecting elements from an XML document. As HTML and XML follow the same document structure, XPath can also be used to select elements from a web page.

Locating and selecting elements from the web page is the key to web scraping with Selenium. For locating and selecting elements from the web page, you can use XPath selectors in Selenium.

In this article, I am going to show you how to locate and select elements from web pages using XPath selector in Selenium.



Safeguard your business data

From breaches to backing up files,
help keep your business secured

LEARN MORE →

Prerequisites:



For examples of this article, you must have,

1. Mozilla Firefox or Google Chrome web browsers installed on your computer.

2. Computer.

3. Internet.

4. `selenium` package installed on your computer.

Sponsored by Advertising Partner

5. Mozilla Firefox or Google Chrome web browsers installed on your computer.

6. Must know how to install the Firefox Gecko Driver or Chrome Web Driver.

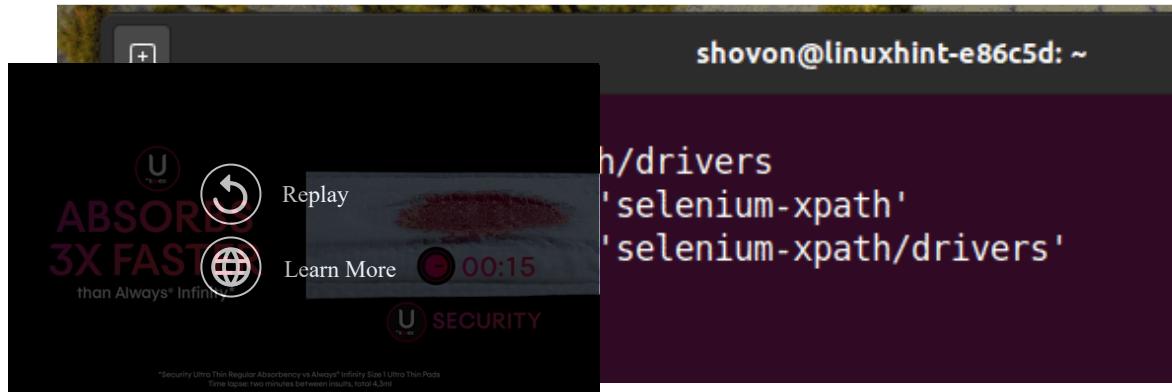
Watch to learn more

For full instructions on steps 4, 5 and 6, read my article [Introduction to Selenium in Python](#). You can also check out my articles on the other topics on [LinuxHint.com](#). Be sure to check them out if you need any assistance.

Setting Up a Project Directory:

To keep everything organized, create a new project directory `selenium-xpath/` as follows:

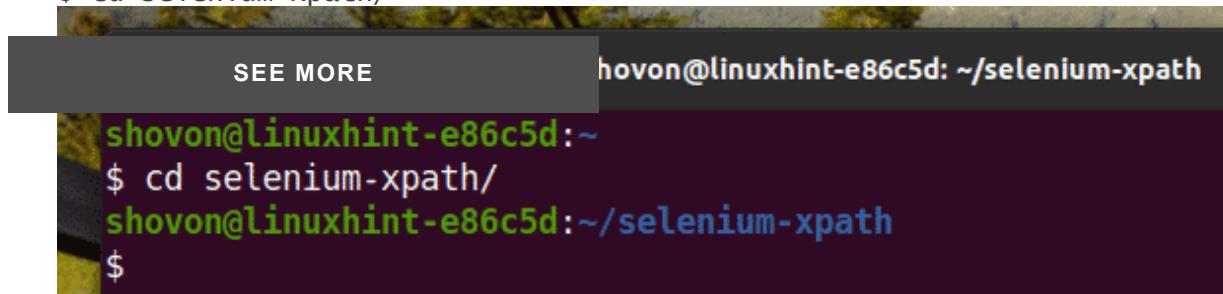
```
$ mkdir -pv selenium-xpath/drivers
```



Sponsored by Advertising Partner

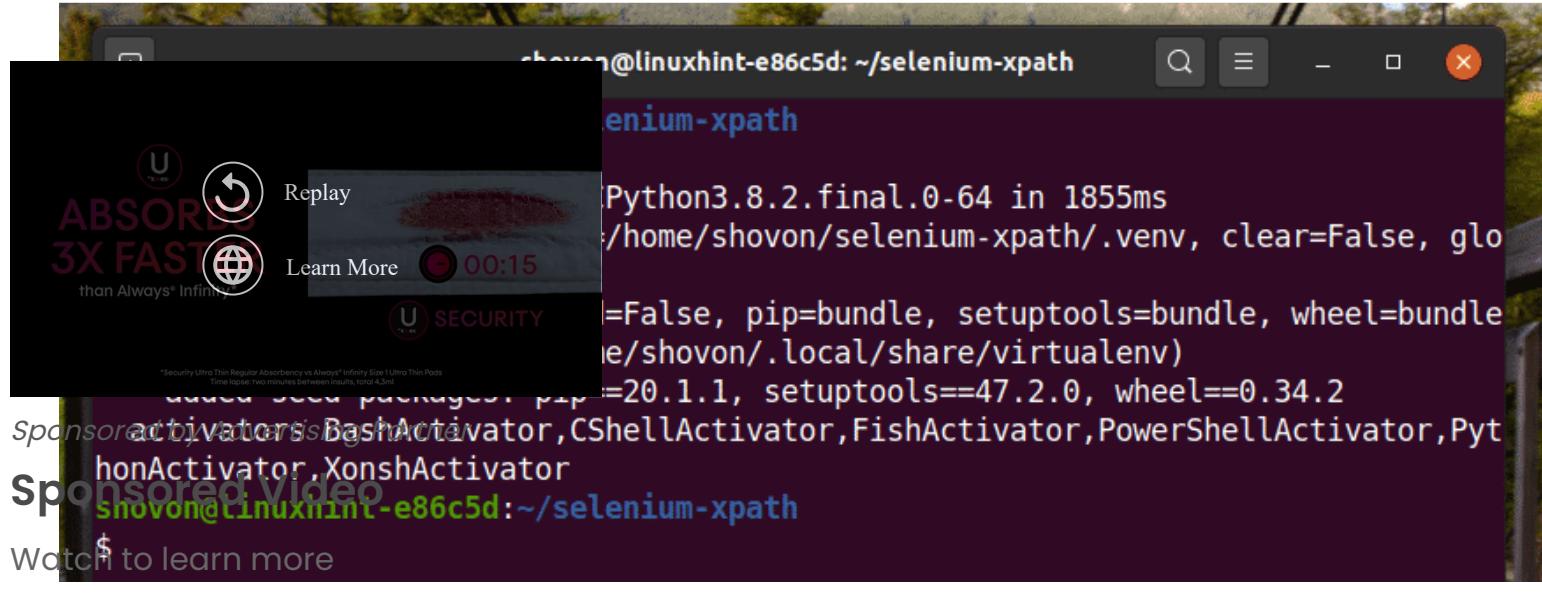
Navigate to the `selenium-xpath/` project directory as follows:
Sponsored Video

Watch to learn more.



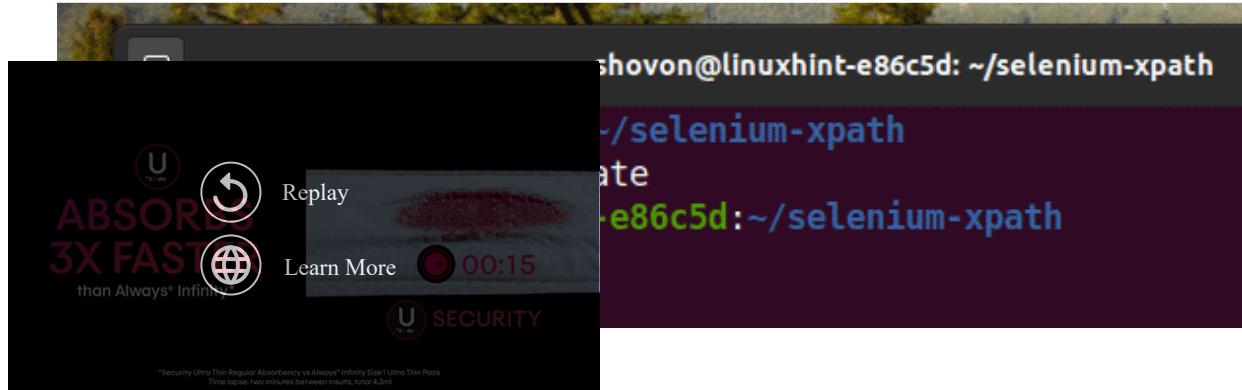
Create a Python virtual environment in the project directory as follows:

```
$ virtualenv .venv
```



Activate the virtual environment as follows:

```
$ source .venv/bin/activate
```



Install Selenium Python library using PIP3 as follows:

Sponsored by Advertising Partner

\$ pip3 install selenium

Sponsored video

Watch to learn more

shovon@linuxhint-e86c5d: ~/selenium-xpath

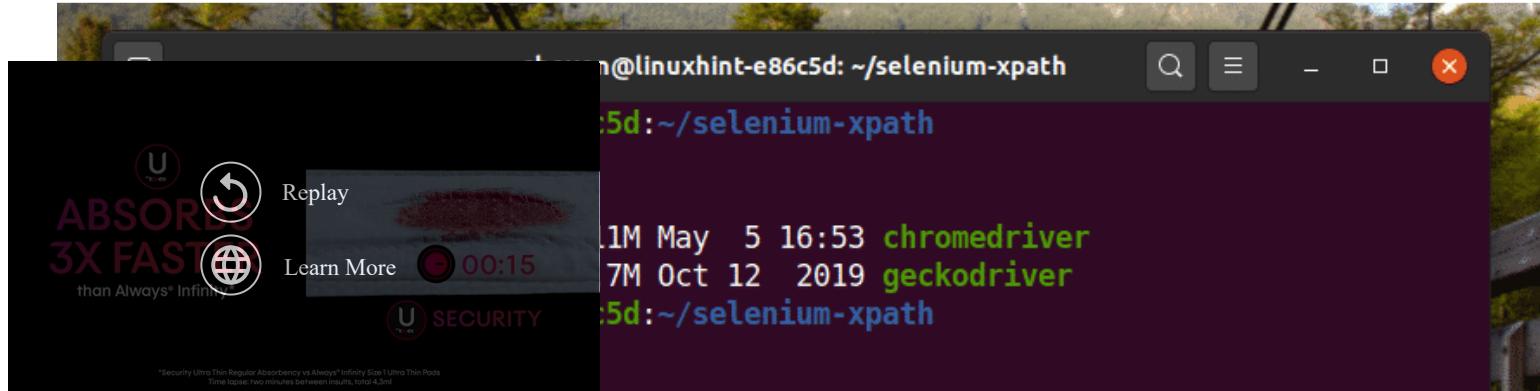


(.venv) shovon@linuxhint-e86c5d:~/selenium-xpath

SEE MORE

```
Using cached selenium-3.141.0-py2.py3-none-any.whl (904 kB)
Requirement already satisfied: urllib3 in ./.venv/lib/python3.8/site-packages (from selenium) (1.25.10)
Installing collected packages: selenium
Successfully installed selenium-3.141.0
(.venv) shovon@linuxhint-e86c5d:~/selenium-xpath
$
```

Download and install all the required web driver in the **drivers/** directory of the project. I have explained the process of downloading and installing web drivers in my article [Introduction to Selenium in Python 3](#).



Sponsored by Advertising Partner

Get the XPath Selector using Chrome Developer Tool:

In this section, I am going to show you how to find the XPath selector of the web page element you want to select with Selenium using the built-in Developer Tool of the Google

[SEE MORE](#)

An advertisement for ManageEngine OpManager Plus. It features the ManageEngine logo at the top left. The main heading is 'Manage all your IT Operations from a single console'. Below this is a list of five services: Network Monitoring, Bandwidth Analysis, Configuration Management, Firewall Analysis, and IP Address Management. At the bottom left is a screenshot of the software interface, which includes various charts and data tables. On the right side, there is a large yellow button with the text 'Try Opmanager Plus'.

To get the XPath selector using the Google Chrome web browser, open Google Chrome, and visit the web site from which you want to extract data. Then, press the right mouse button

(RMB) on an empty area of the page and click on **Inspect** to open the **Chrome Developer**



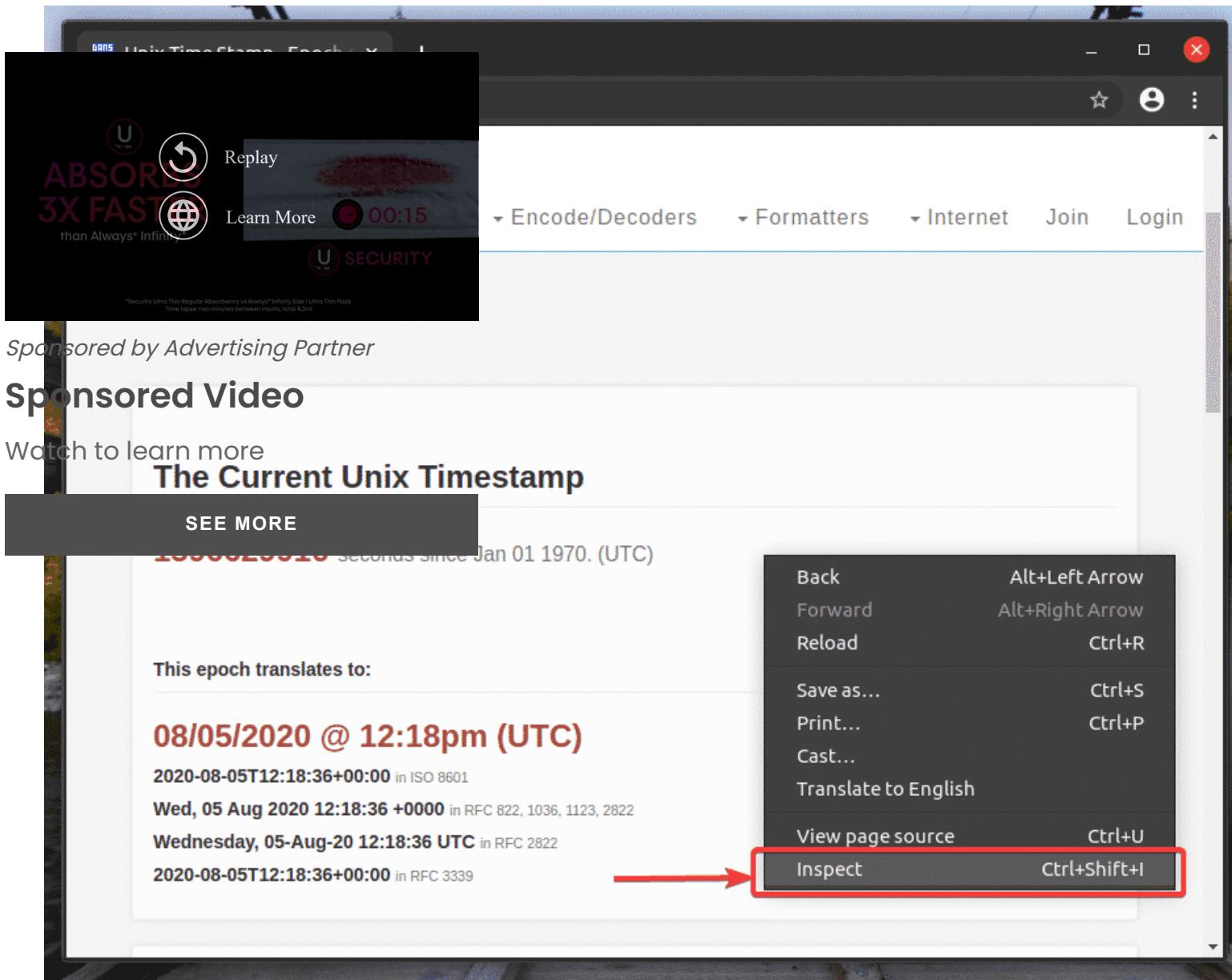
+ I to open the **Chrome Developer Tool**.

Sponsored by Advertising Partner

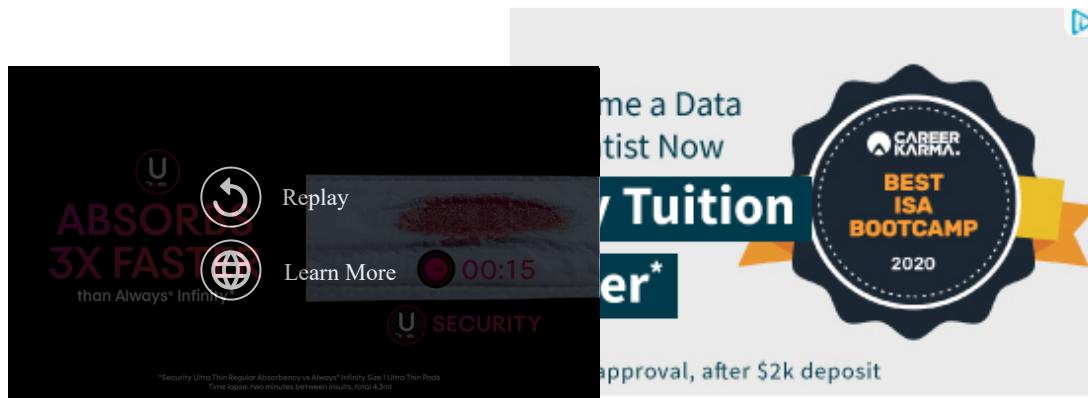
Sponsored Video

Watch to learn more

SEE MORE



Chrome Developer Tool should be opened.



Sponsored by Advertising Partner

galvanize

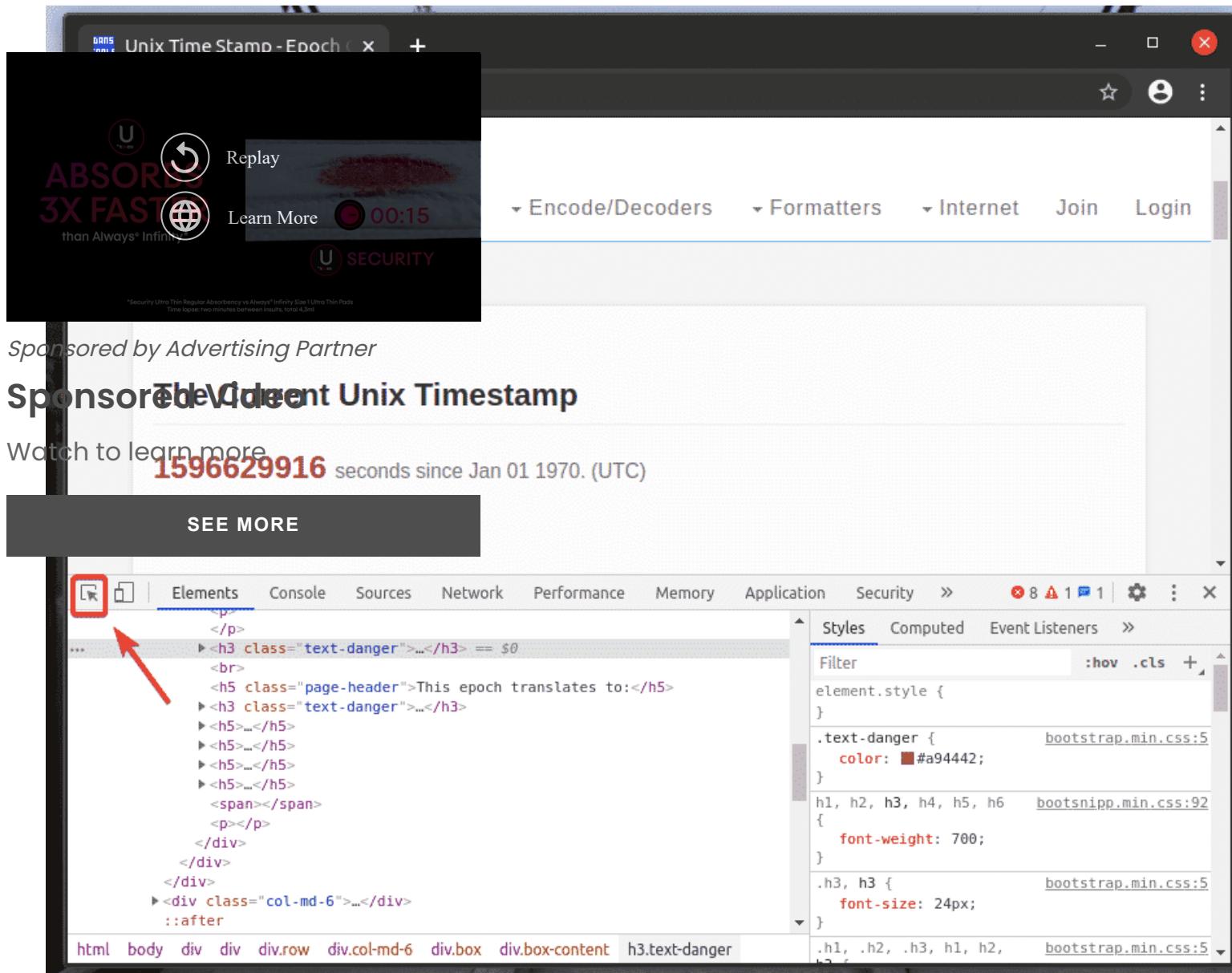
START CODING

Sponsored Video

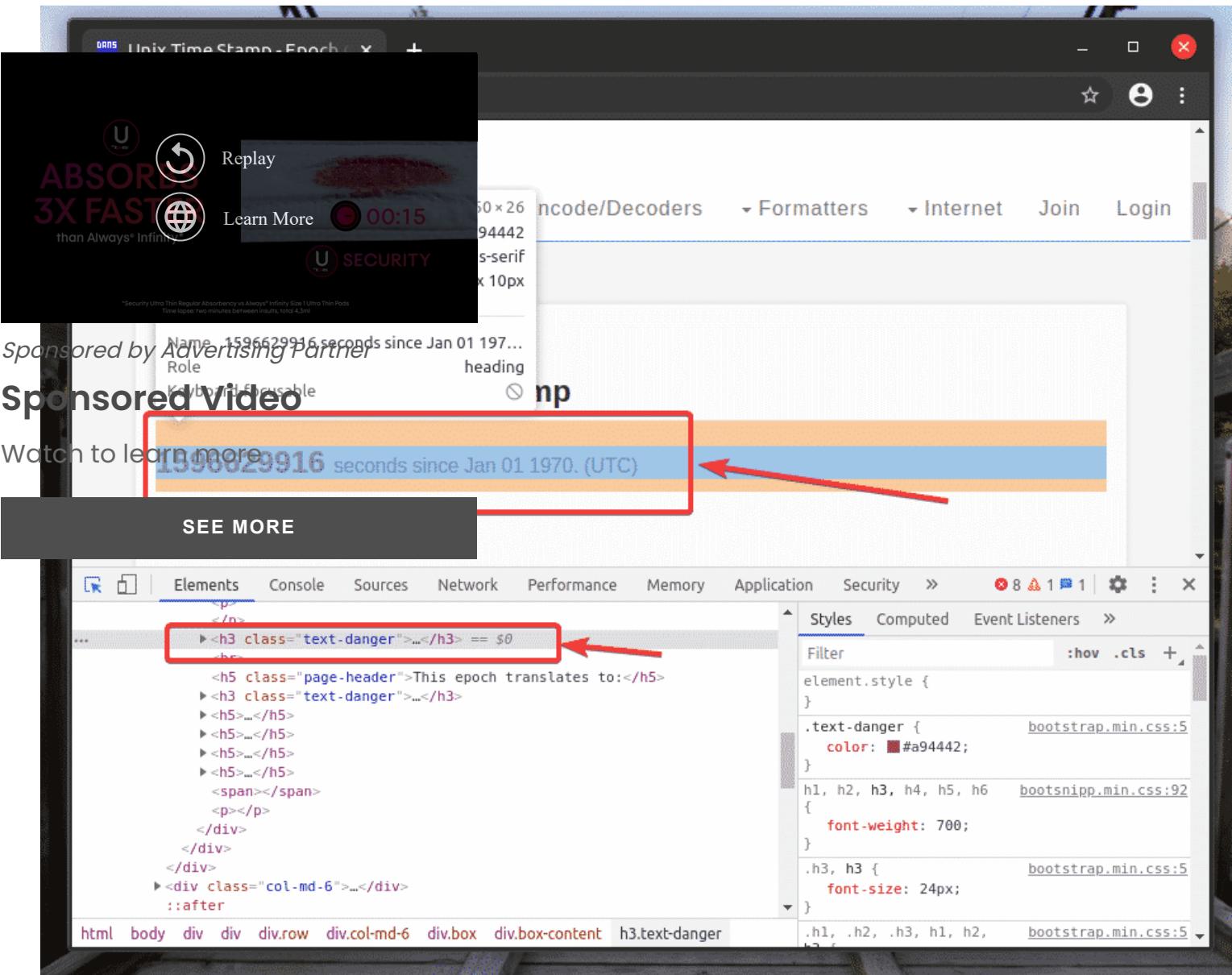
To find the HTML representation of your desired web page element, click on the **Inspect**(
watch to learn more)

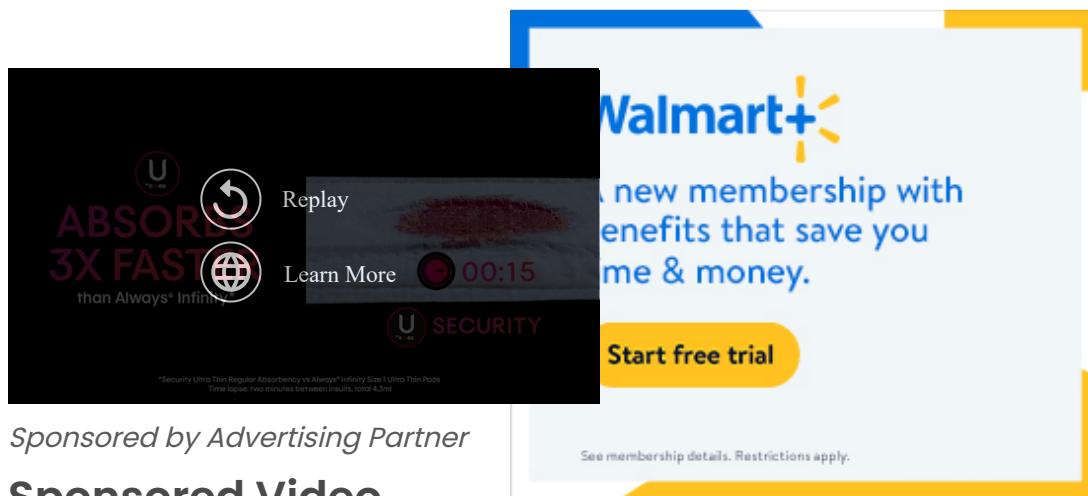
SEE MORE

bot below.



Then, hover over your desired web page element and press the left mouse button (LMB) to select it.

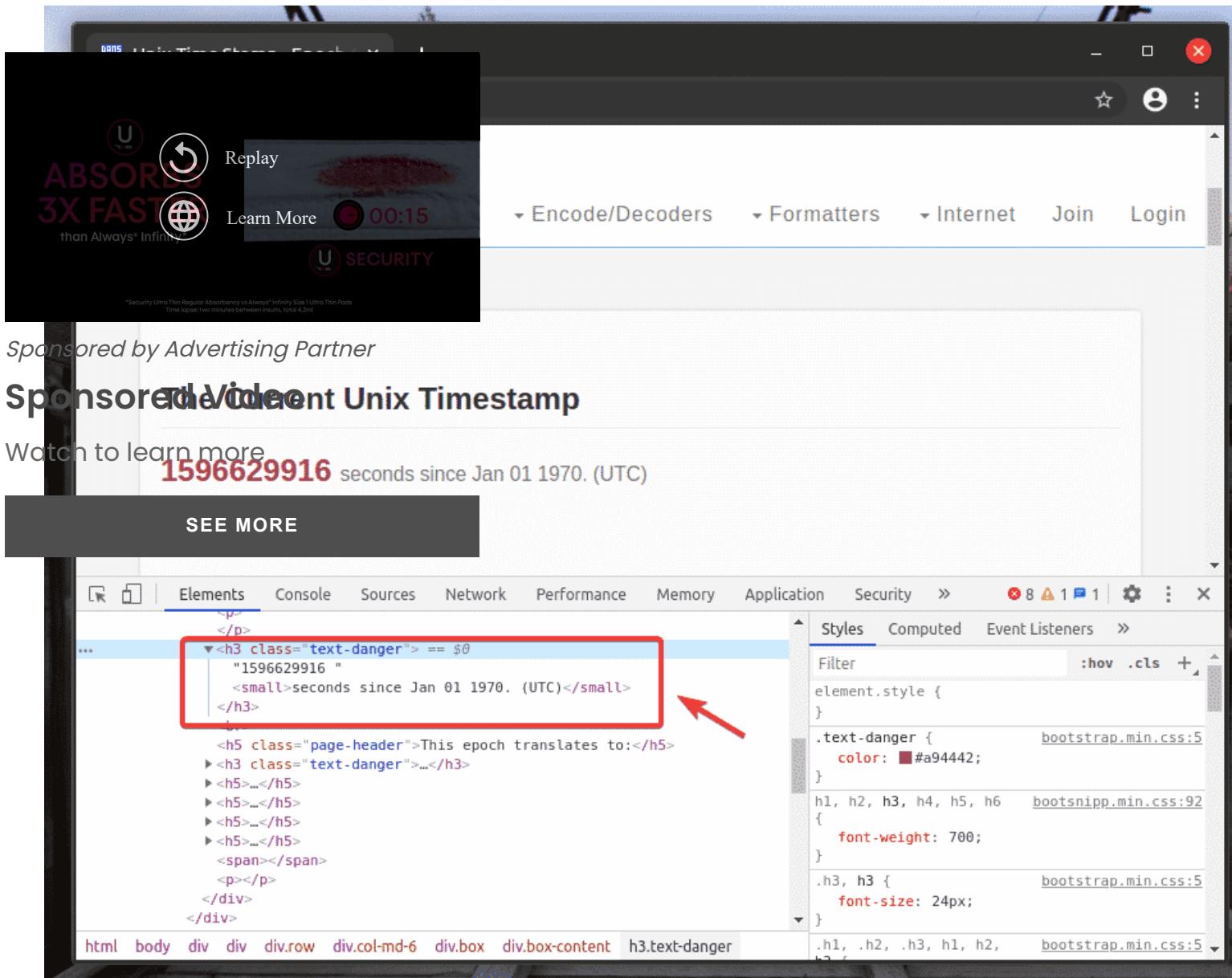




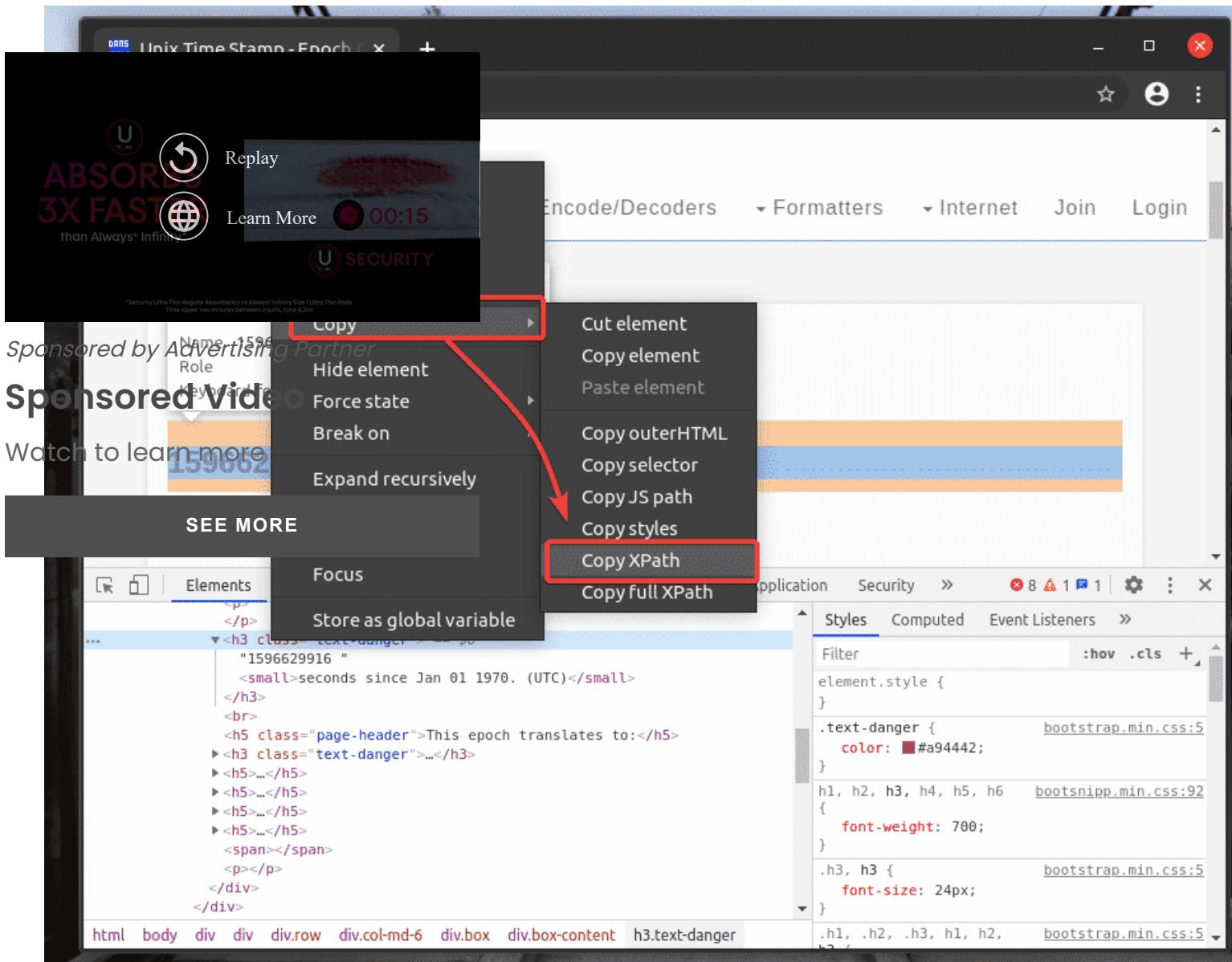
Sponsored Video

The HTML representation of the web element you have selected will be highlighted in the [Elements tab of the Chrome Developer Tool](#), as you can see in the screenshot below.

[SEE MORE](#)



To get the XPath selector of your desired element, select the element from the **Elements** tab of **Chrome Developer Tool** and right-click (RMB) on it. Then, select **Copy > Copy XPath**, as marked in the screenshot below.



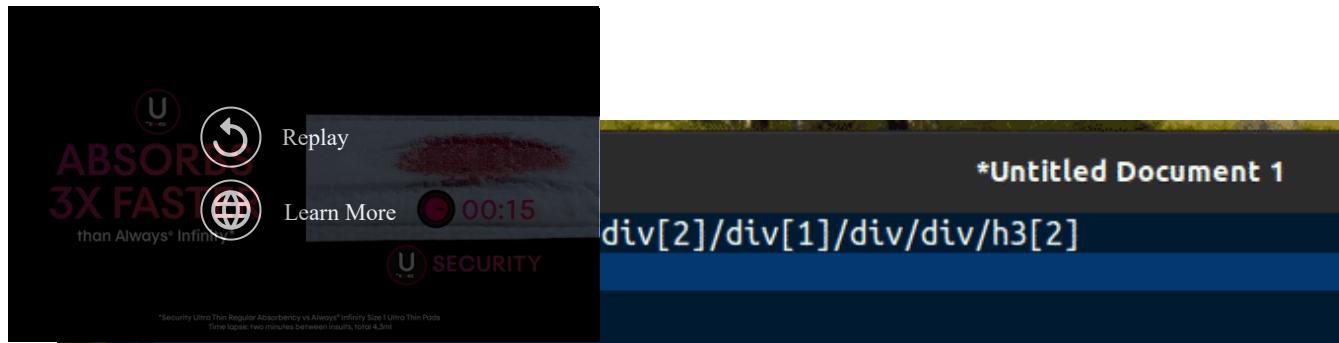
Study Data Science

Recession proof your career

galvanize

START CODING

I have pasted the XPath selector in a text editor. The XPath selector looks as shown in the



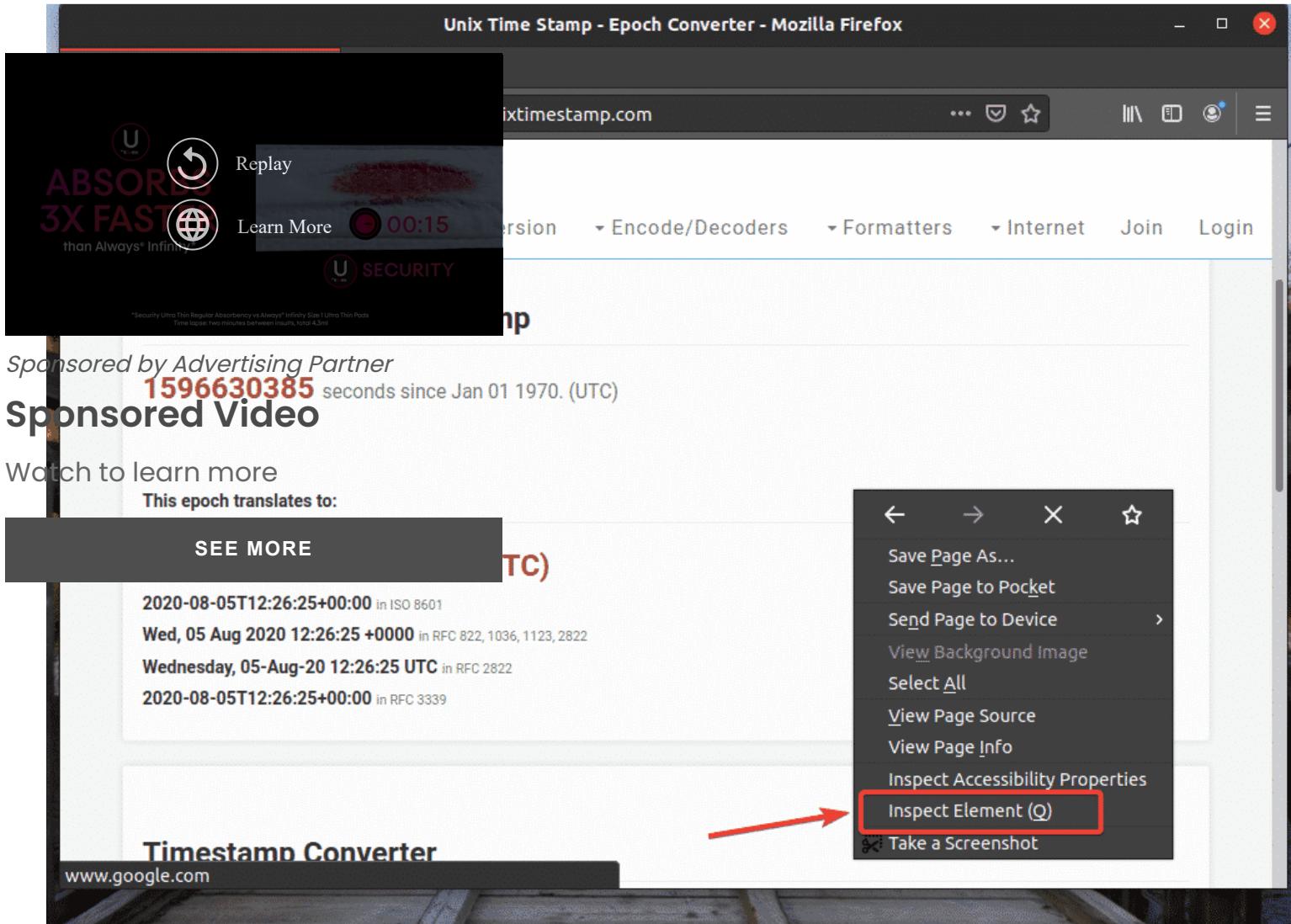
Sponsored by Advertising Partner

Get the XPath Selector using Firefox Developer Tool:

In this section, I am going to show you how to find the XPath selector of the web page elements to interact with Selenium using the built-in Developer Tool of the Mozilla

[SEE MORE](#)

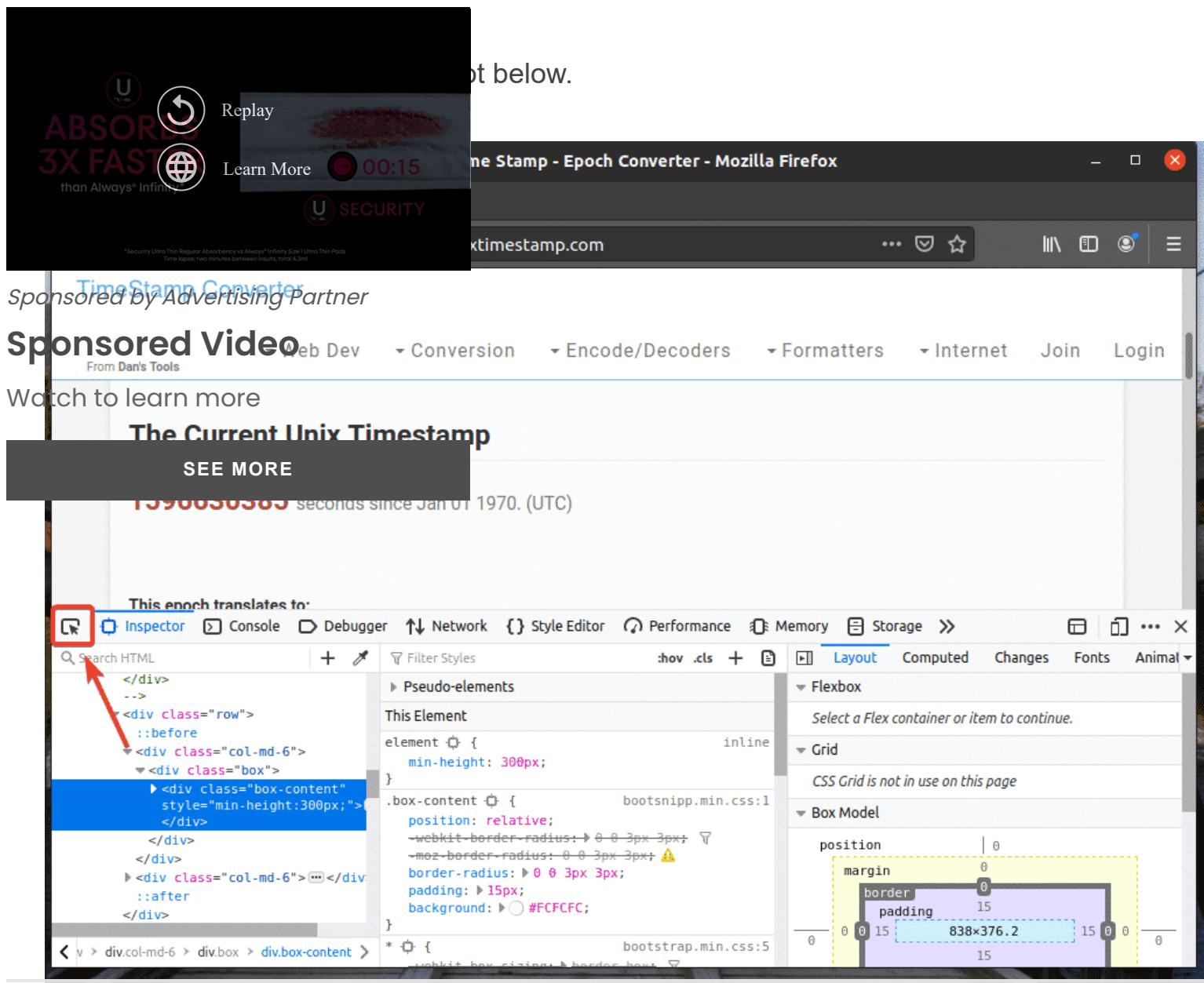
To get the XPath selector using the Firefox web browser, open Firefox and visit the web site from which you want to extract data. Then, press the right mouse button (RMB) on an empty area of the page and click on **Inspect Element (Q)** to open the **Firefox Developer Tool**.



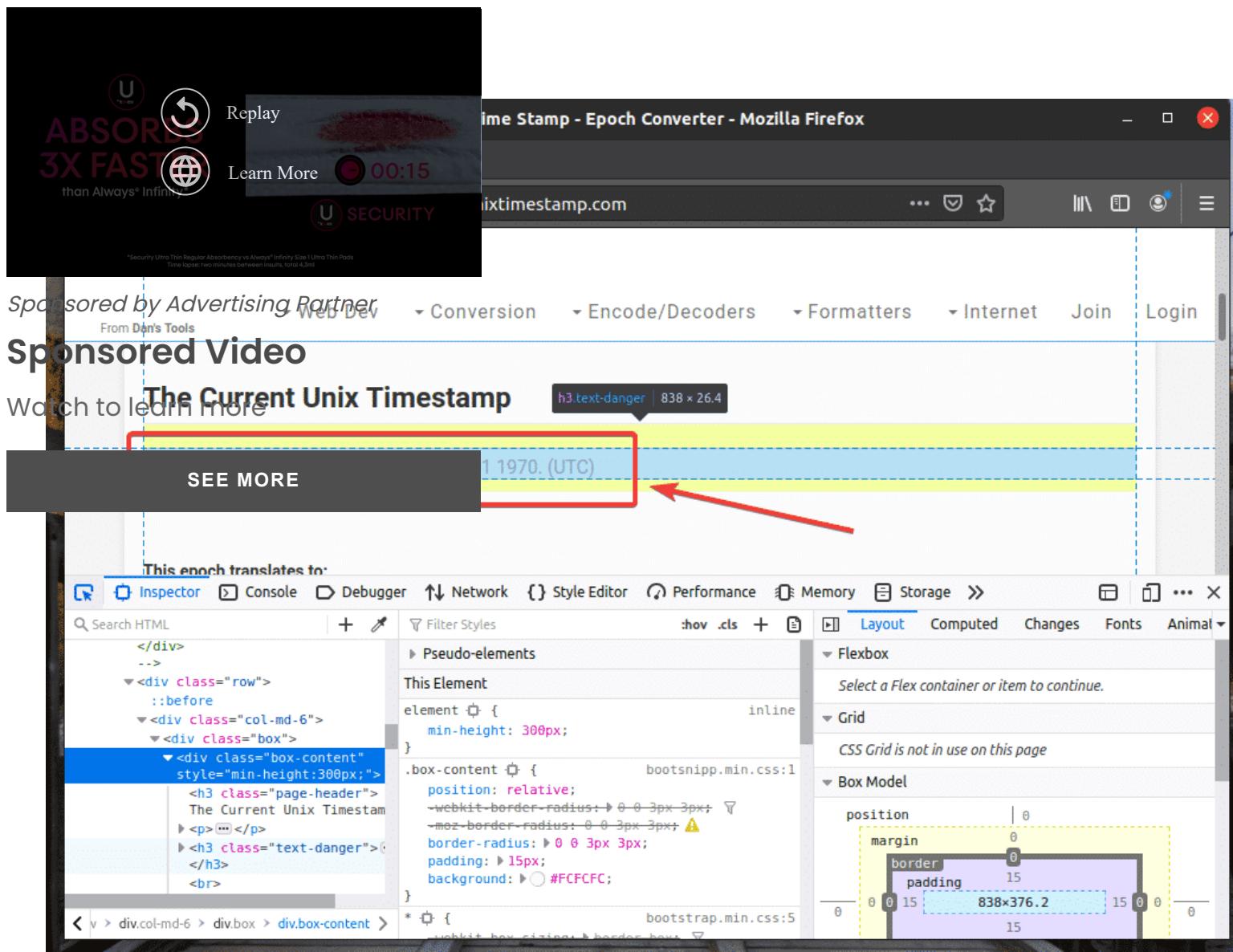
Firefox Developer Tool should be opened.



To find the HTML representation of your desired web page element, click on the **Inspect**(



Then, hover over your desired web page element and press the left mouse button (LMB) to



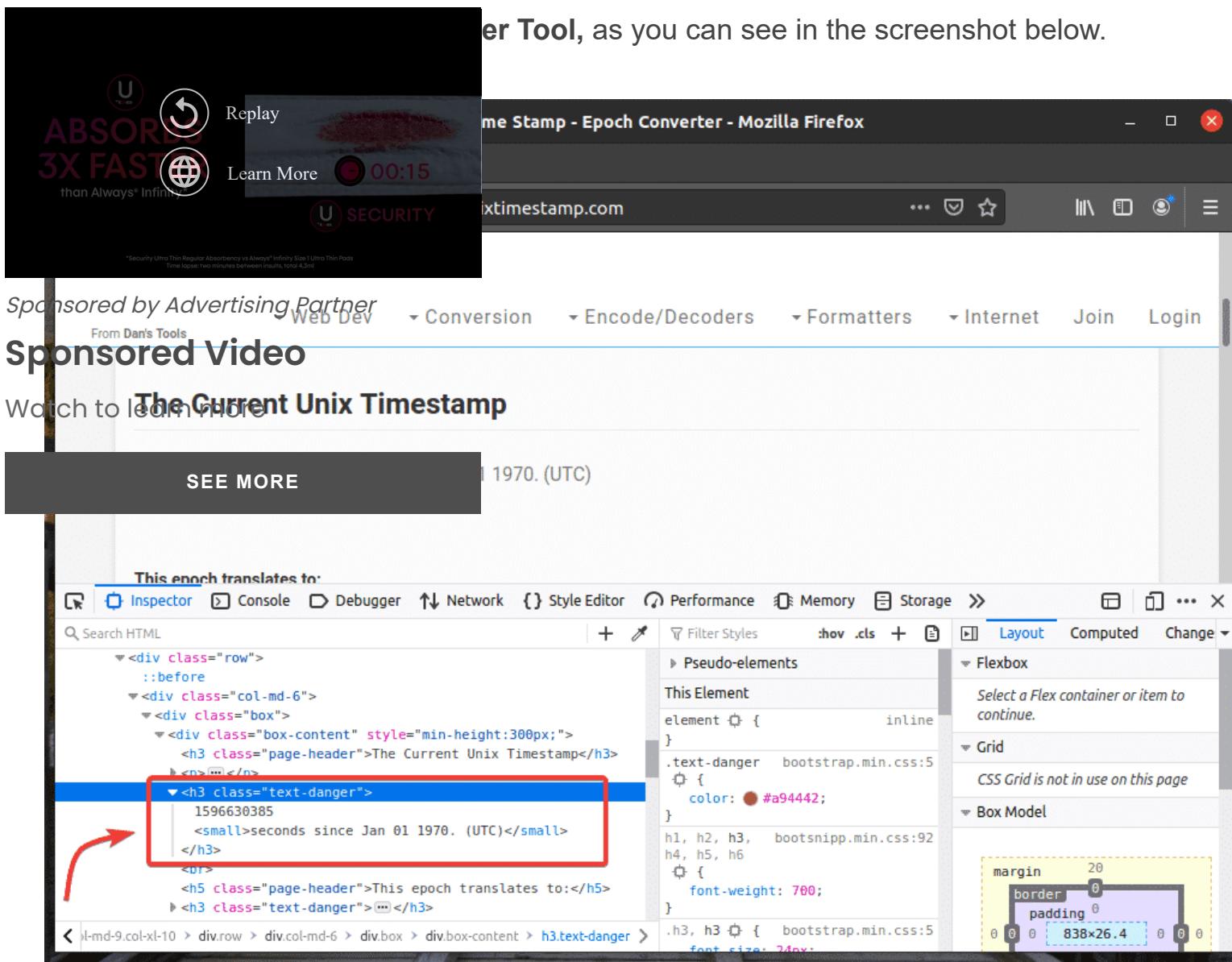
Study Data Science

Recession proof your career with our 13-week, live online bootcamp

galvanize

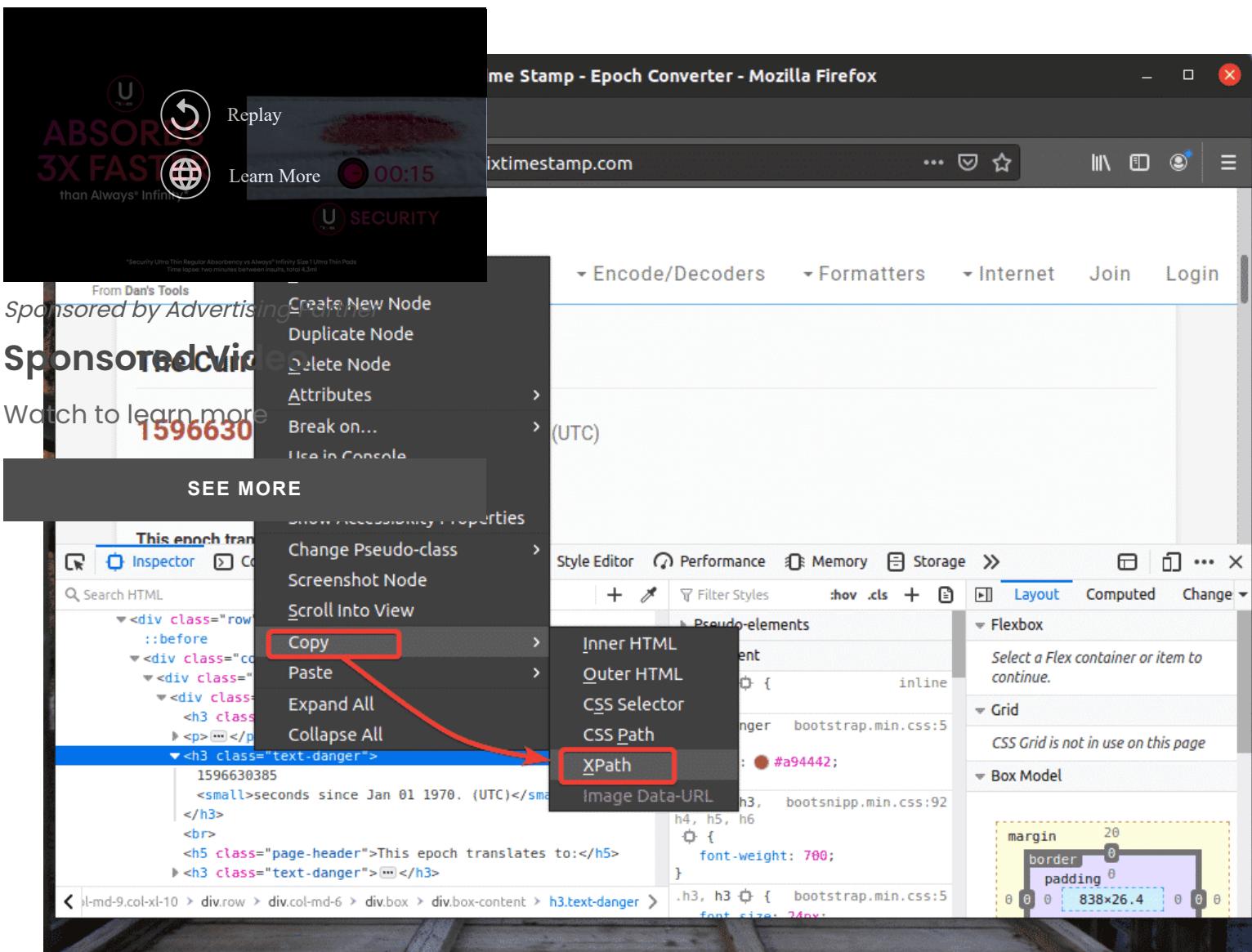
START CODING

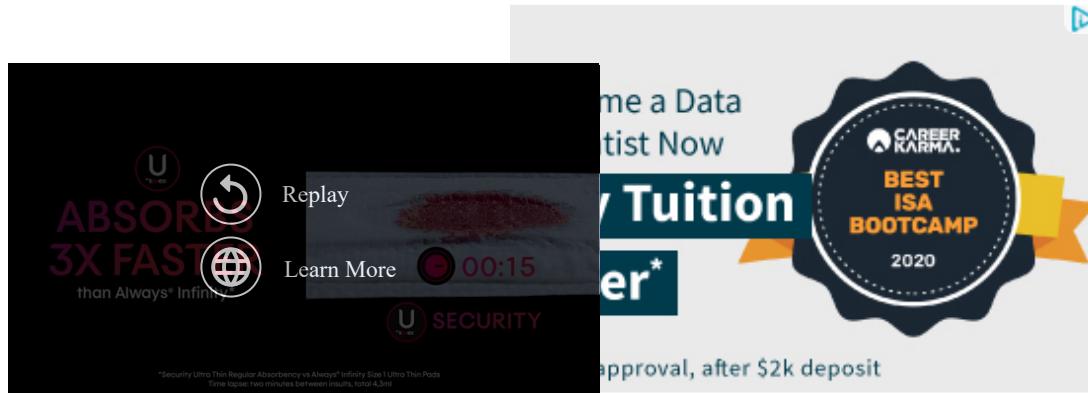
The HTML representation of the web element you have selected will be highlighted in the **Inspector Tool**, as you can see in the screenshot below.



To get the XPath selector of your desired element, select the element from the **Inspector** tab of **Firefox Developer Tool** and right-click (RMB) on it. Then, select **Copy > XPath** as marked

in the screenshot below.





Sponsored by Advertising Partner

galvanize

START CODING

Sponsored Video

The XPath selector of your desired element should look something like this.
watch to learn more

SEE MORE

*Untitled Document 1

```
1 /html/body/div[1]/div[1]/div[2]/div[1]/div/div/h3[2]
2
3 |
```

Extracting Data from Web Pages using XPath Selector:

In this section, I am going to show you how to select web page elements and extract data from them using XPath selectors with the Selenium Python library.

First, create a new Python script **ex01.py** and type in the following lines of codes.

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
options = webdriver.ChromeOptions()
options.headless = True
browser = webdriver.Chrome(executable_path=".//drivers/chromedriver",
options=options)
```

```
browser.get("https://www.unixtimestamp.com/")
timestamp = browser.find_element_by_xpath('/html/body/div[1]/div[1]
                                            [1]/div[2]/div[1]/div[1]/div[2]/div[1]/div[1]/div[1])
print('Current timestamp: %s' % (timestamp.text.split(' ')[0]))
```

Copy Python script.

ex01.py - selenium-xpath - Visual Studio Code



```
1  from selenium import webdriver
2  from selenium.webdriver.common.keys import Keys
3  from selenium.webdriver.common.by import By
4
5  options = webdriver.ChromeOptions()
6  options.headless = True
7
8  browser = webdriver.Chrome(executable_path="./drivers/chromedriver", options=options)
9
10 browser.get("https://www.unixtimestamp.com/")
11
12 timestamp = browser.find_element_by_xpath('/html/body/div[1]/div[1]/div[2]/div[1]/div[1]/div[2]/div[1]/div[1]/div[1])
13 print('Current timestamp: %s' % (timestamp.text.split(' ')[0]))
14
15 browser.close()
16
```

Test Case Management Software

Use Cloud or Download version. Highly customizable & flexible
for optimized test results.

gurock.com

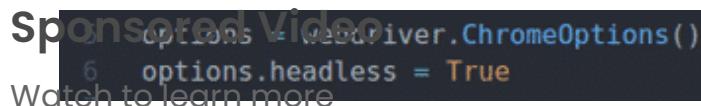
OPEN

Line 1-3 imports all the required Selenium components.



object, and line 6 enables headless mode for the Chrome

Sponsored by Advertising Partner



SEE MORE

object using the **chromedriver** binary from the **drivers/** directory of the project.

```
    /  
8   browser = webdriver.Chrome(executable_path='./drivers/chromedriver', options=options)  
9
```

Line 10 tells the browser to load the website unixtimestamp.com.

```
10  browser.get("https://www.unixtimestamp.com/")  
11
```

Line 12 finds the element that has the timestamp data from the page using the XPath selector and stores it in the **timestamp** variable.

Line 13 parses the timestamp data from the element and prints it on the console.

```
12 timestamp = browser.find_element_by_xpath('/html/body/div[1]/div[1]/div[2]/div[1]/div/div/h3[2]')
13 print('Current timestamp: %s' % (timestamp.text.split(' ')[0]))
```

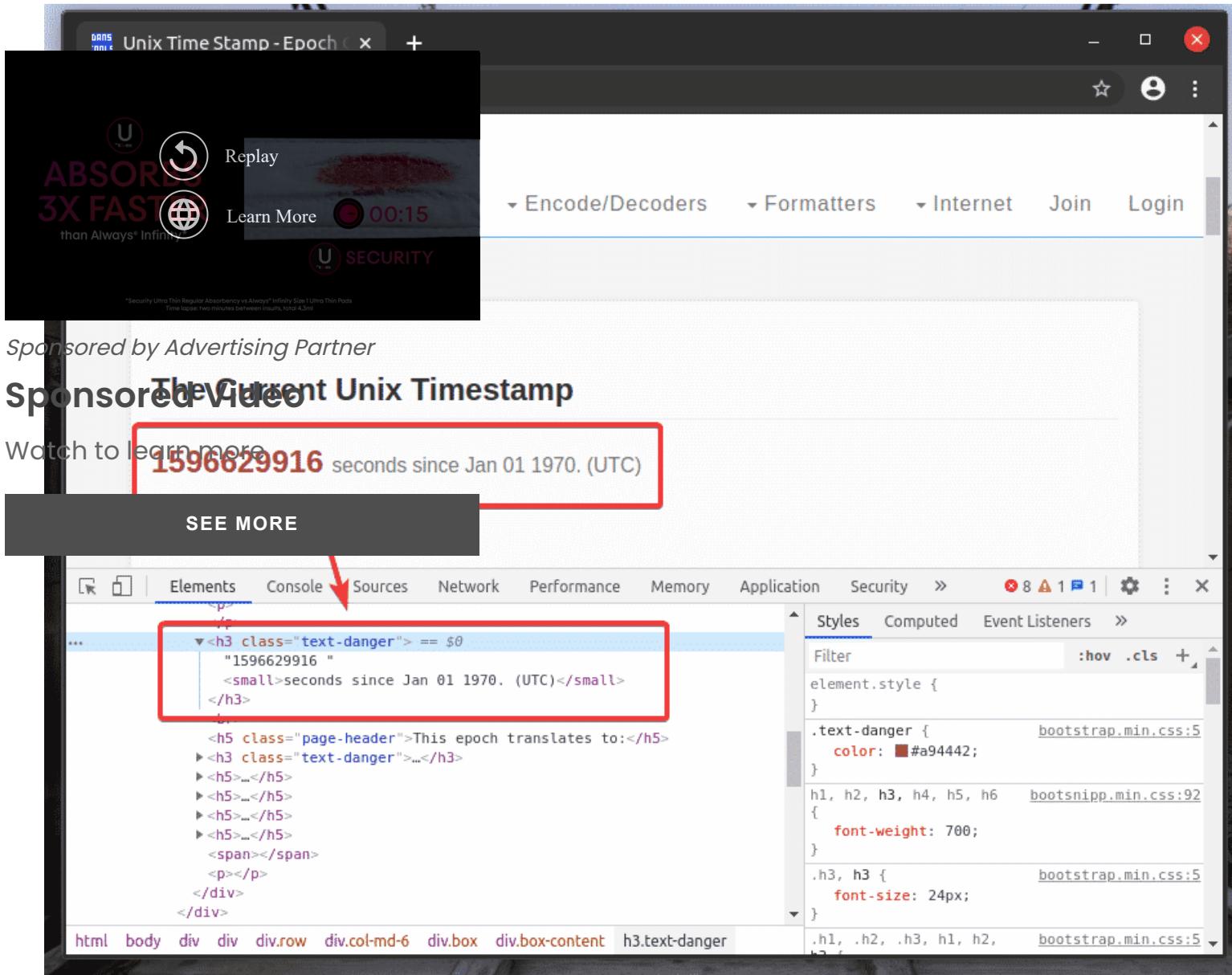


Sponsored by Advertising Partner

Sponsored Video

Watch to learn more

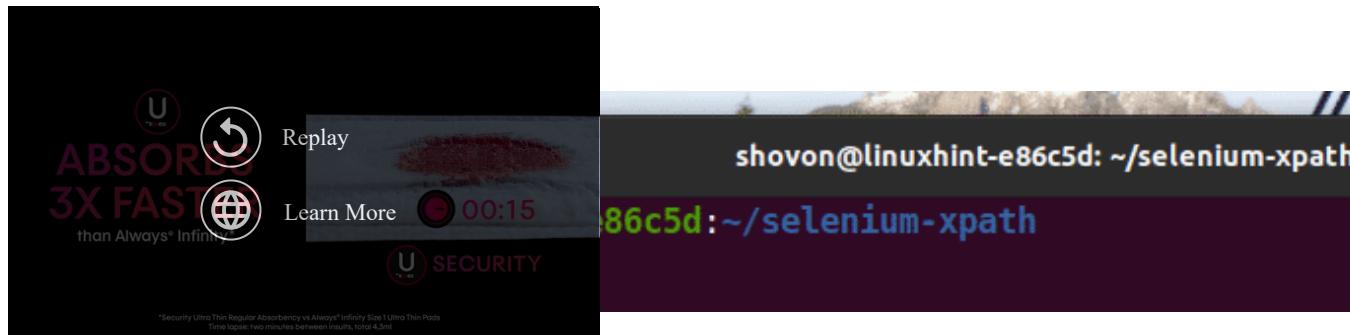
SEE MORE



Line 14 closes the browser.

15 browser.close()

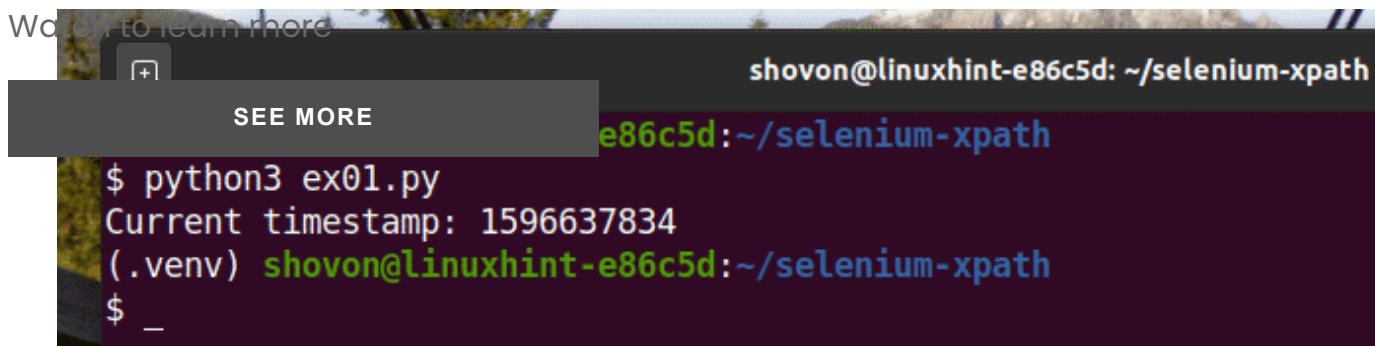
Run the Python script **ex01.py** as follows:



Sponsored by Advertising Partner

As you can see, the timestamp data is printed on the screen.

Sponsored Video

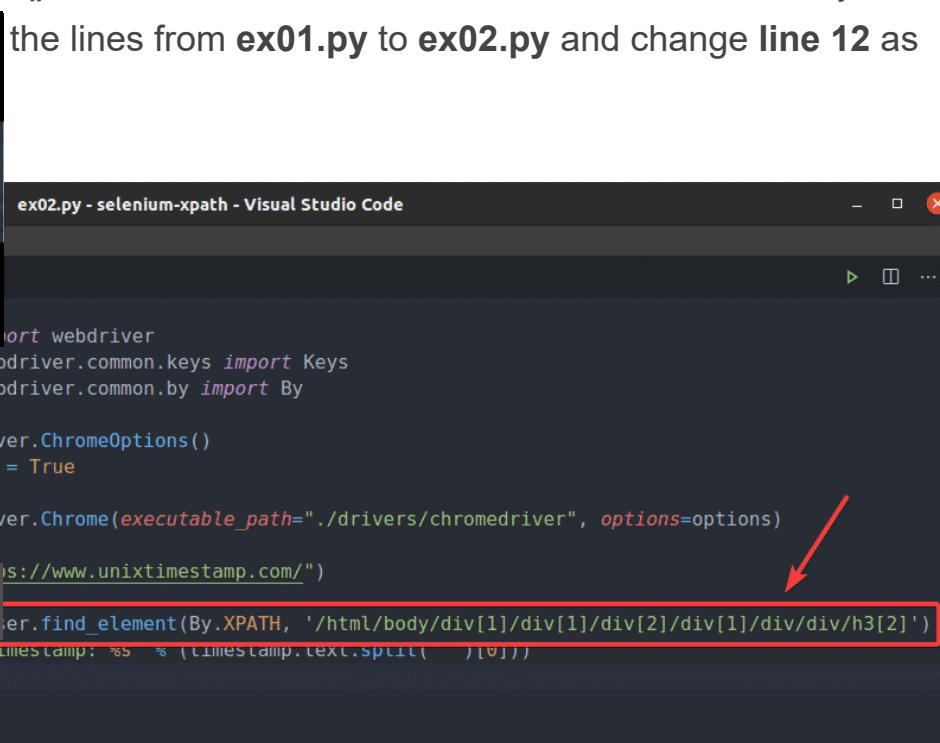
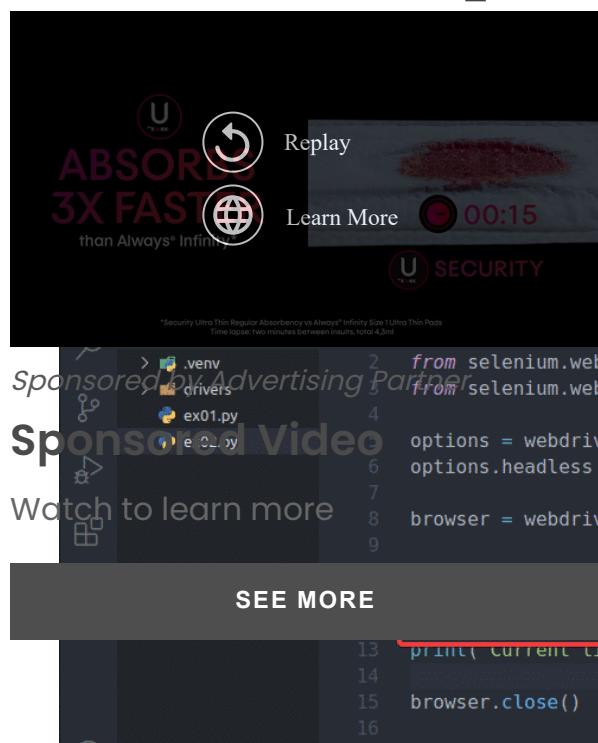


Here, I have used the **browser.find_element_by_xpath(selector)** method. The only parameter of this method is the **selector**, which is the XPath selector of the element.

Instead of **browser.find_element_by_xpath()** method, you can also use **browser.find_element(By, selector)** method. This method needs two parameters. The first parameter **By** will be **By.XPATH** as we will be using the XPath selector, and the second parameter **selector** will be the XPath selector itself. The result will be the same.

To see how `browser.find_element()` method works for XPath selector, create a new Python

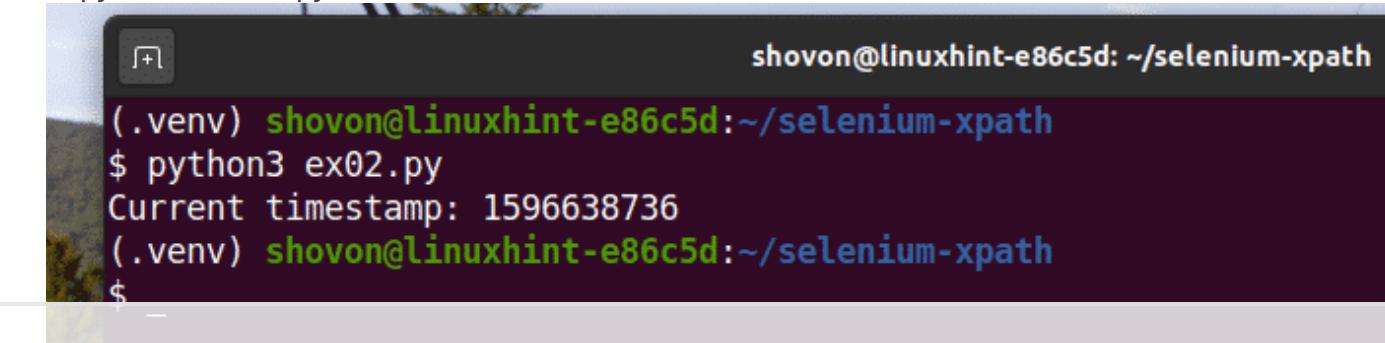
the lines from `ex01.py` to `ex02.py` and change **line 12** as



```
import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
options = webdriver.ChromeOptions()
options.headless = True
browser = webdriver.Chrome(executable_path='./drivers/chromedriver", options=options)
url = "https://www.unixtimestamp.com/"
browser.find_element(By.XPATH, '/html/body/div[1]/div[1]/div[2]/div[1]/div/div/h3[2]')
print("Current timestamp: %s" % (timestamp.text.split(' ')[0]))
browser.close()
```

As you can see, the Python script `ex02.py` gives the same result as `ex01.py`.

```
$ python3 ex02.py
```



```
shovon@linuxhint-e86c5d: ~/selenium-xpath
(.venv) shovon@linuxhint-e86c5d:~/selenium-xpath
$ python3 ex02.py
Current timestamp: 1596638736
(.venv) shovon@linuxhint-e86c5d:~/selenium-xpath
$
```

The `browser.find_element_by_xpath()` and `browser.find_element()` methods are used to

select elements from web pages. If you want to find and select multiple elements, then you have to use `browser.find_elements_by_xpath()` and `browser.find_elements()`.

`browser.find_element_by_xpath()` method takes the same argument as the `browser.find_element()` method.

Sponsored by Advertising Partner

Sponsored Video `browser.find_elements()` method takes the same arguments as the

[Watch to learn more](#)

[SEE MORE](#)

elements from the page. We can select a list of names using XPath selector from `random-name-generator.info` with the Selenium Python library.

The unordered list (`ol` tag) has a 10 `li` tags inside each containing a random name. The XPath to select all the `li` tags inside the `ol` tag in this case is `//*[@id="main"]/div[3]/div[2]/ol//li`

The screenshot shows a web page from generator.info featuring a sponsored video for Always Ultra Thin Pads. The video player has a progress bar at 00:15. Below the video, there's a list of names under the heading "Sponsored Video". The names are:

- 2. Santiago Hubbard
- 3. Homer Nunez
- 4. Javier Blair
- 5. Gretchen Wilson
- 6. Audrey Hamilton

A "SEE MORE" button is present, followed by another list:

- 9. Noah Lawson
- 10. Grady Hardy

The "SEE MORE" button and the second list are highlighted with a red rectangle. The developer tools' Elements tab is active, showing the corresponding HTML structure:

```
<div class="results">
  ...
  <ol class="nameList">
    <li>Kerry Richardson</li>
    <li>Santiago Hubbard</li>
    <li>Homer Nunez</li>
    <li>Javier Blair</li>
  </ol>
</div>
```

The "nameList" ol element is also highlighted with a red rectangle. The developer tools' Styles tab shows the CSS for the "nameList" class:

```
ol.nameList li {
  font-family: 'Open Sans', sans-serif;
  font-size: 18px;
  line-height: 26px;
  font-weight: 400;
}
```

Red arrows point from the "SEE MORE" button and the second list back to the "nameList" element in the DOM tree.

Let's go through an example of selecting multiple elements from the web page using XPath selectors.

Create a new Python script **ex03.py** and type in the following lines of codes in it.



```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
```

```
options = webdriver.ChromeOptions()
options.headless = True
```

```
browser = webdriver.Chrome(executable_path='./drivers/chromedriver', options=options)
```

```
browser.get("http://random-name-generator.info/")
```

```
names = browser.find_elements_by_xpath('//*[@id="main"]/div[3]/div[2]/ol//li')
```

```
for name in names:
    print(name.text)
```

```
browser.close()
```

Sponsored Video

Once you're done, save the **ex03.py** Python script.

Watch to learn more

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By

options = webdriver.ChromeOptions()
options.headless = True

browser = webdriver.Chrome(executable_path='./drivers/chromedriver', options=options)

browser.get("http://random-name-generator.info/")

names = browser.find_elements_by_xpath('//*[@id="main"]/div[3]/div[2]/ol//li')
for name in names:
    print(name.text)

browser.close()
```

Line 1-8 is the same as in **ex01.py** Python script. So, I am not going to explain them here again.

```
1 from selenium import webdriver
2 from selenium.webdriver.common.keys import Keys
3
4 option = webdriver.ChromeOptions()
5 option.add_argument('headless')
6
7 options = option=options)
8
9 executable_path='./drivers/chromedriver', options=options)
```



Line 10 tells the browser to load the website random-name-generator.info.

Sponsored Video

```
10 browser.get("http://random-name-generator.info/")
```

Watch to learn more

SEE MORE

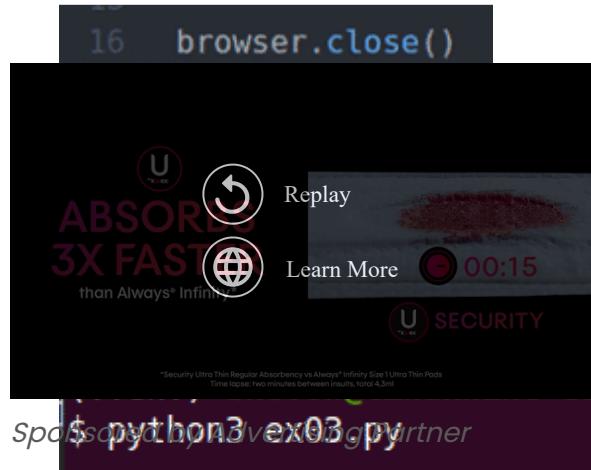
g the `browser.find_elements_by_xpath()` method. This method uses the XPath selector `//*[@id="main"]/div[3]/div[2]/ol//li` to find the name list. Then, the name list is stored in the `names` variable.

```
12 names = browser.find_elements_by_xpath('//*[@id="main"]/div[3]/div[2]/ol//li')
```

In lines 13 and 14, a `for` loop is used to iterate through the `names` list and print the names on the console.

```
13 for name in names:
14     print(name.text)
```

Line 16 closes the browser.



follows:

\$ python3 ex03.py

Sponsored Video

Watch to learn more
As you can see, the names are extracted from the web page and printed on the console.

```
shovon@linuxhint-e86c5d: ~/selenium-xpath
6c5d:~/selenium-xpath
$ python3 ex03.py
Ross Owen
Audrey Hubbard
Brad Dunn
Elsa Brown
Jon Walker
Leo Black
Bert George
Gregory Robbins
Ernest Brooks
Kelly Greene
(shovon@linuxhint-e86c5d:~/selenium-xpath)
$
```

A screenshot of a terminal window. The title bar says "shovon@linuxhint-e86c5d: ~/selenium-xpath". The command "\$ python3 ex03.py" is run, followed by a series of names: Ross Owen, Audrey Hubbard, Brad Dunn, Elsa Brown, Jon Walker, Leo Black, Bert George, Gregory Robbins, Ernest Brooks, and Kelly Greene. The terminal prompt ends with "(.venv)". There are some UI elements like a "SEE MORE" button and a "F" icon on the left side of the terminal window.

Instead of using the `browser.find_elements_by_xpath()` method, you can also use the `find_elements()` method. The first argument of this method is **By.XPATH**, Path selector.



Sponsored by Advertising Partner

Sponsored Video

Watch to learn more

File Edit Selection View Go Run Terminal Help

EXPLORER ex04.py

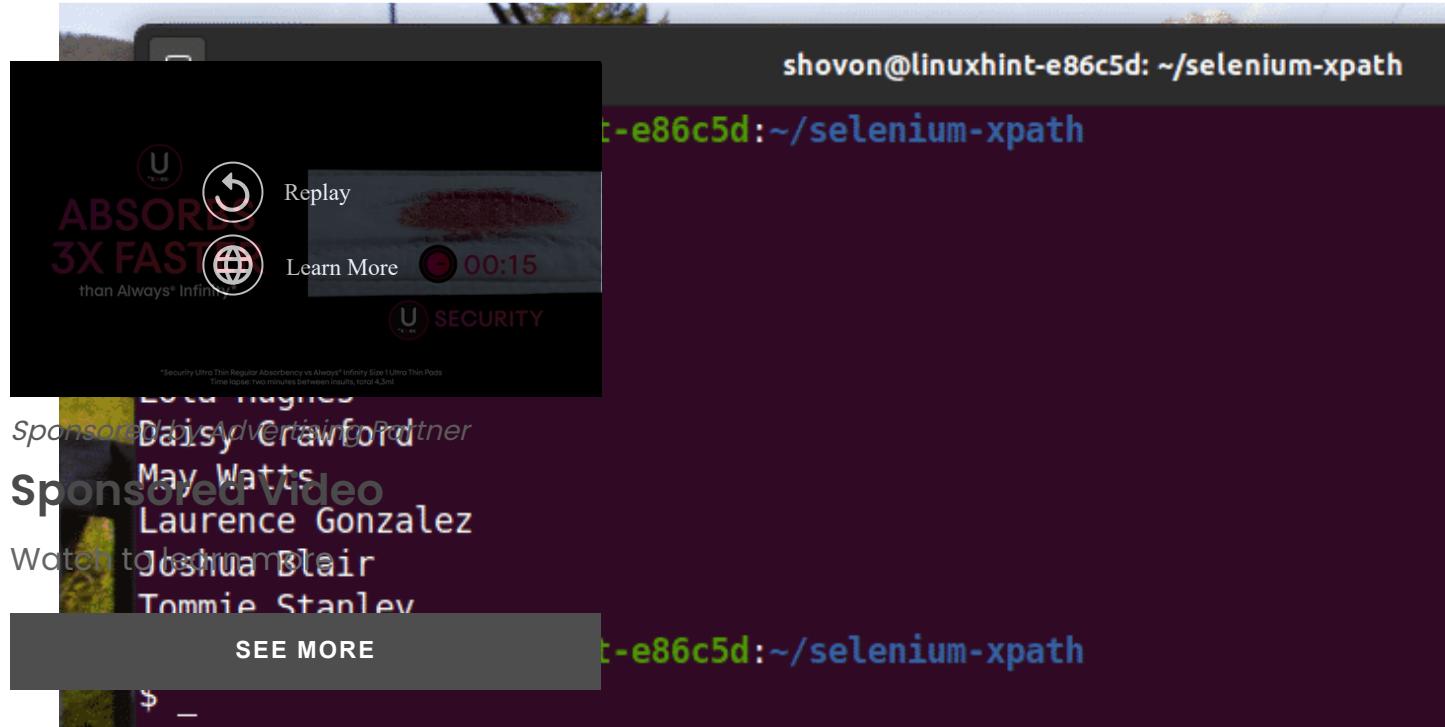
SEE MORE

```
1 import webdriver
2 from selenium.webdriver.common.keys import Keys
3
4 options = webdriver.ChromeOptions()
5 options.headless = True
6
7 browser = webdriver.Chrome(executable_path='./drivers/chromedriver', options=options)
8
9 browser.get("http://random-name-generator.info/")
10
11
12 names = browser.find_elements(By.XPATH, '//*[@id="main"]/div[3]/div[2]/ol//li')
13 for name in names:
14     print(name.text)
15
16 browser.close()
17
```

A red arrow points from the text "You should get the same result as before." to the line of code "names = browser.find_elements(By.XPATH, '//*[@id="main"]/div[3]/div[2]/ol//li')".

You should get the same result as before.

```
$ python3 ex04.py
```

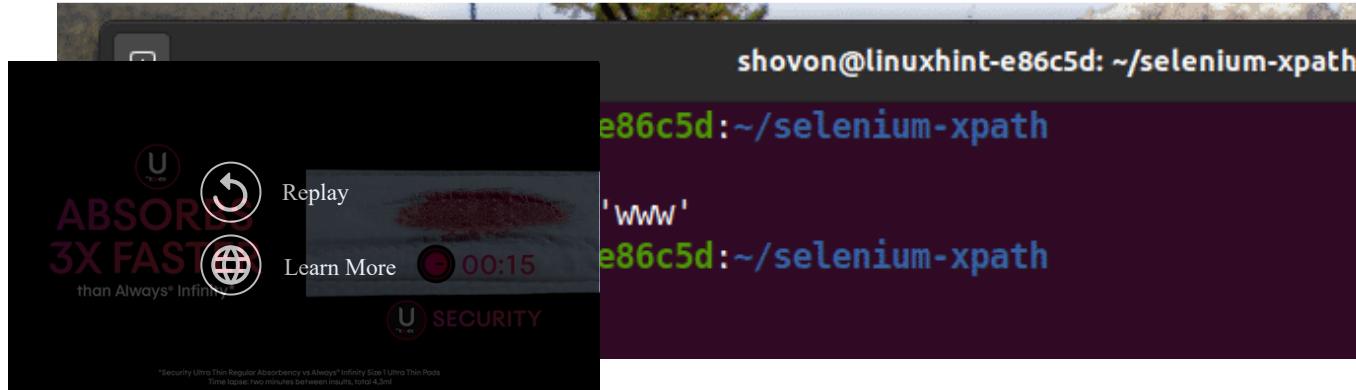


Basics of XPath Selector:

The Developer Tool of Firefox or Google Chrome web browser does generate XPath selector automatically. But these XPath selectors are sometimes not sufficient for your project. In that case, you must know what a certain XPath selector does to build your XPath selector. In this section, I am going to show you the basics of XPath selectors. Then, you should be able to build your own XPath selector.

Create a new directory **www/** in your project directory as follows:

```
$ mkdir -v www
```



Sponsored by Advertising Partner

Create a new file **web01.html** in the **www/** directory and type in the following lines in that file.

Sponsored Video

Watch to learn more

```
<!DOCTYPE html>
<html lang="en">
<head>
    <a href="#" class="button" style="background-color: #555; color: white; padding: 10px 20px; text-decoration: none; font-size: 16px; font-weight: bold; border-radius: 5px; text-align: center; width: fit-content; margin: auto; font-family: sans-serif;">SEE MORE</a>
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta charset="UTF-8" /> Basic HTML Document
    <title>Hello World</title>
</head>
<body>
    <h1>Hello World</h1>
</body>
</html>
```

Once you're done, save the **web01.html** file.

A screenshot of Visual Studio Code. On the right, there's a dark-themed code editor window titled "web01.html - selenium-xpath - Visual Studio Code". It displays the following HTML code:

```
<html>
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <h1>Hello World</h1>
  </body>
</html>
```

To the left of the code editor, there's a semi-transparent "Sponsored Video" overlay. It features a thumbnail of a video showing a person's face, with text overlays: "Replay", "Learn More", and "00:15". The overlay also includes a "U SECURITY" logo and some promotional text about absorbency.

[SEE MORE](#)

Run a simple HTTP server on port 8080 using the following command:

```
$ python3 -m http.server --directory www/ 8080
```

A screenshot of a terminal window. The prompt shows the user is in a virtual environment named ".venv" on a machine named "shovon@linuxhint-e86c5d". The user has run the command:

```
(.venv) shovon@linuxhint-e86c5d:~/selenium-xpath$ python3 -m http.server --directory www/ 8080
```

The HTTP server should start.

A screenshot of a terminal window. The prompt shows the user is in a virtual environment named ".venv" on a machine named "shovon@linuxhint-e86c5d". The user has run the command:

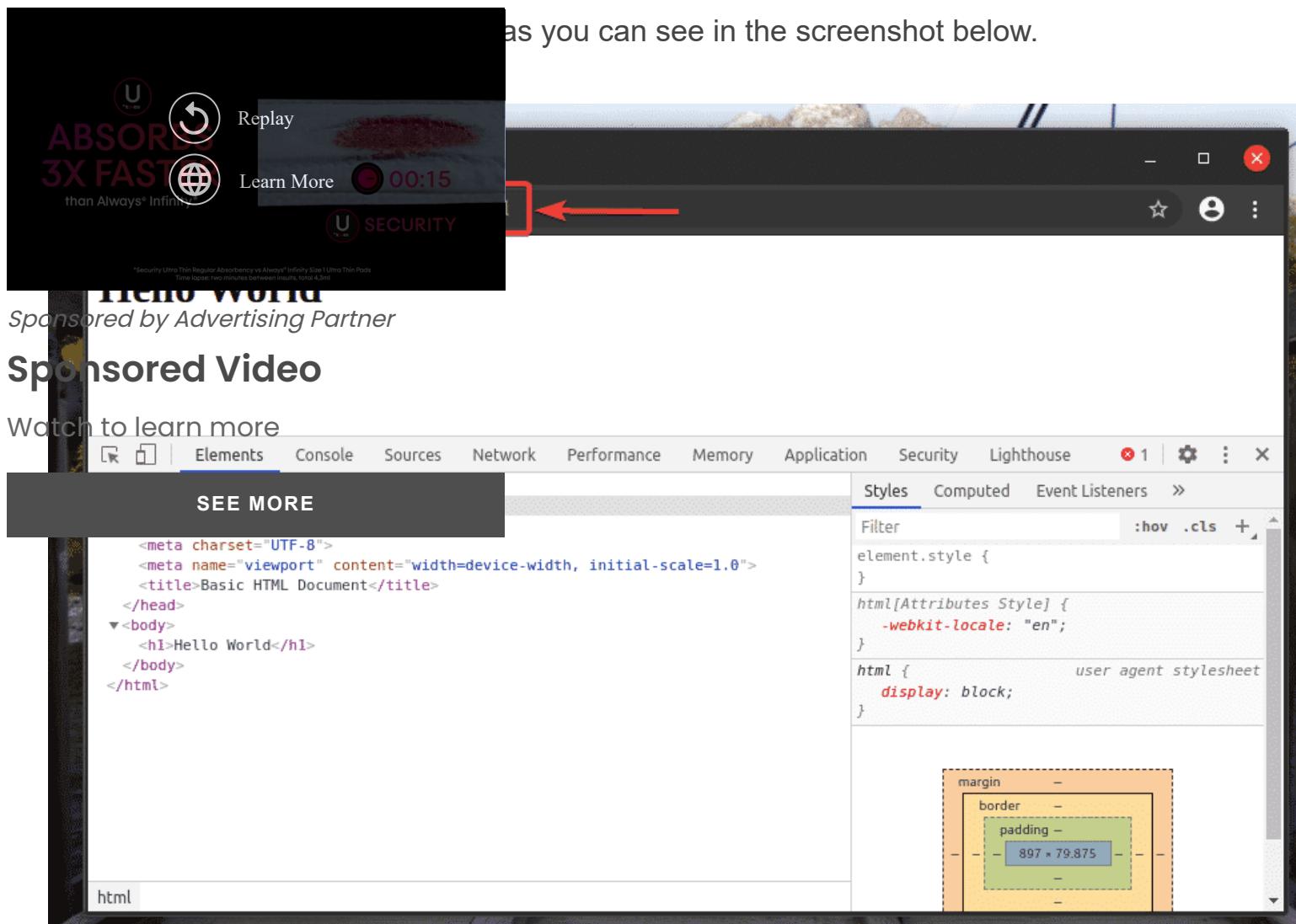
```
(.venv) shovon@linuxhint-e86c5d:~/selenium-xpath$ python3 -m http.server --directory www/ 8080
```

Output from the command:

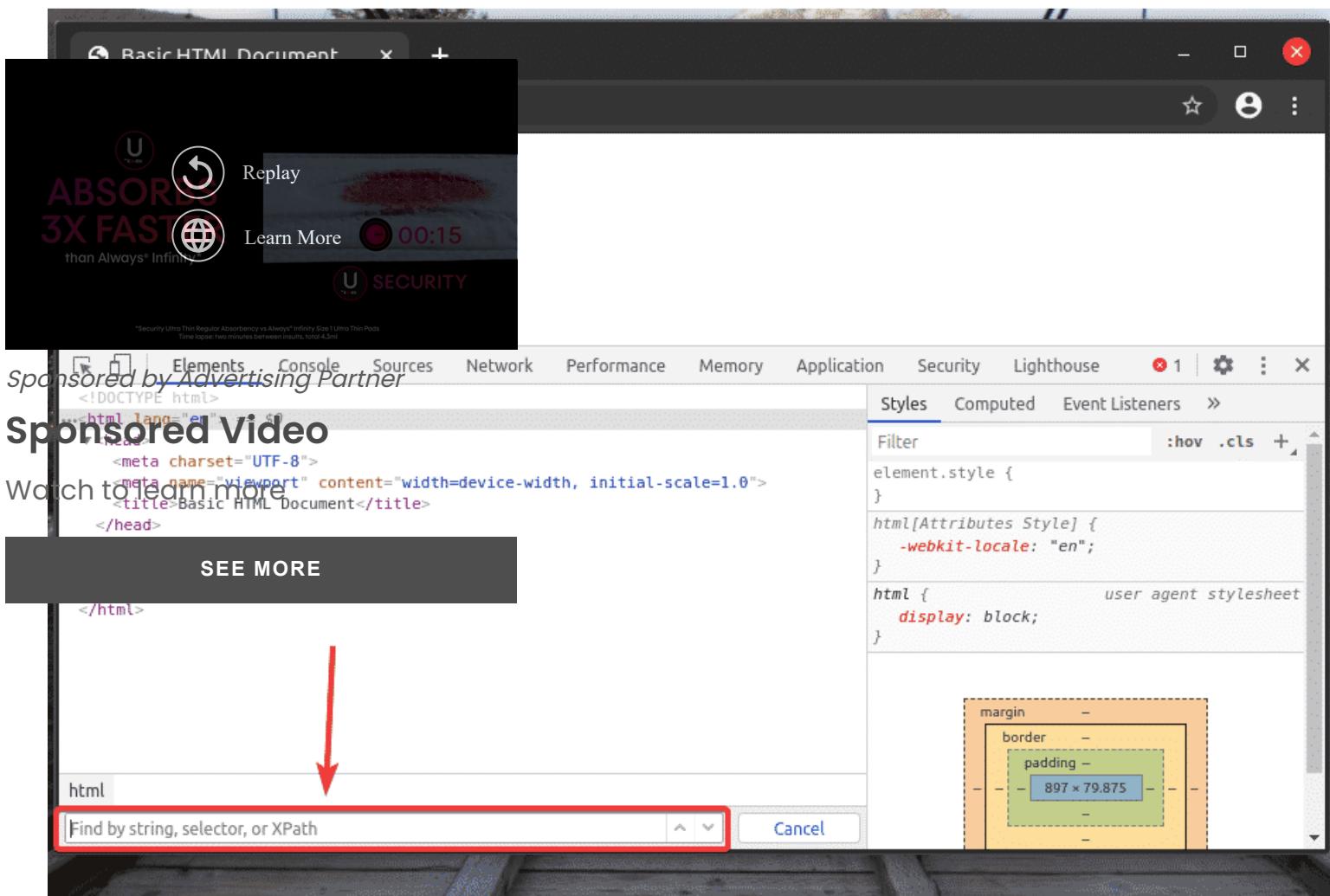
```
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
```

You should be able to access the **web01.html** file using the URL

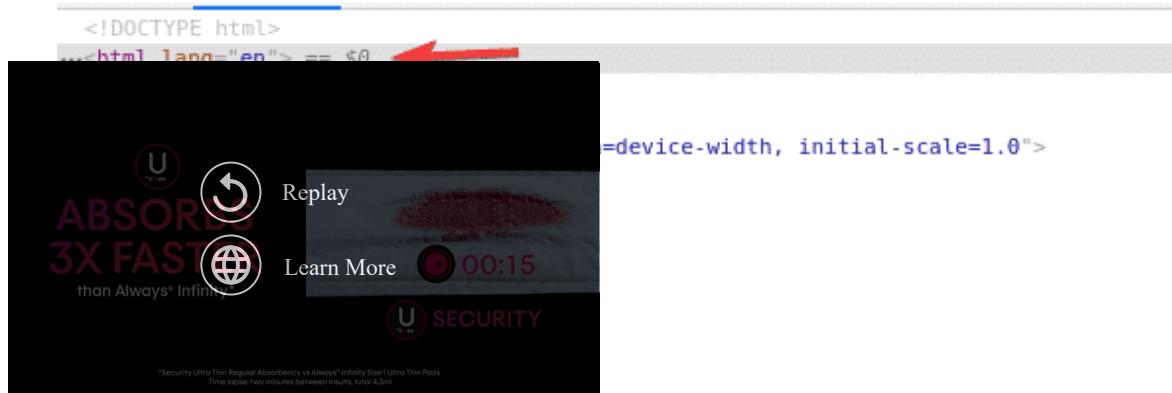
as you can see in the screenshot below.



While the Firefox or Chrome Developer Tool is opened, press **<Ctrl> + F** to open the search box. You can type in your XPath selector here and see what it selects very easily. I am going to use this tool throughout this section.



An XPath selector starts with a **forward slash (/)** most of the time. It's like a Linux directory tree. The / is the root of all elements on the web page.



Sponsored by Advertising Partner

The first child of the html tag is the head. So, the XPath selector /html selects the entire html tag.

Watch to learn more

A screenshot of the Chrome DevTools Elements tab. The top navigation bar has tabs: Elements (which is selected), Console, Sources, Network, Performance, Memory, and Application. Below the tabs, there is a "SEE MORE" button. The main content area shows the DOM tree. The root node is <html>. Under it are <head> and <body>. The <head> node contains <meta charset="UTF-8">, <meta name="viewport" content="width=device-width, initial-scale=1.0">, and <title>Basic HTML Document</title>. The <body> node contains <h1>Hello World</h1>. At the bottom of the DevTools interface, there is a search bar with "html" typed into it, and a results panel showing "/html" with "1 of 1" results. There are also "Cancel" and navigation buttons.

Inside the html tag, we have a body tag. The body tag can be selected with the XPath selector /html/body

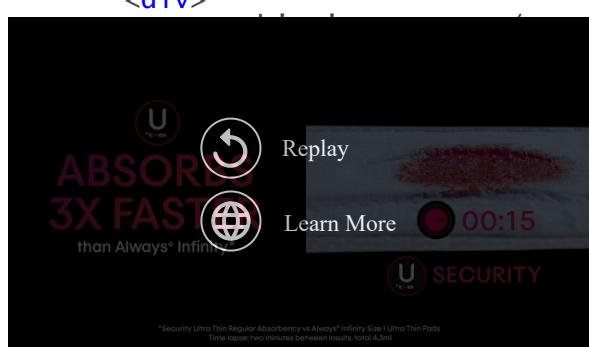
The screenshot shows a browser's developer tools with the "Elements" tab selected. The main content area displays the HTML structure of a video player. At the top, there are icons for zooming in and out, followed by tabs for Elements, Console, Sources, Network, Performance, Memory, and Application. Below the tabs, the page content includes a banner for "U by Verizon" with the text "ABSORBS 3X FAST than Always® Infinity" and "SECURITY". A video player interface is shown with a "Replay" button, a blurred video thumbnail, a "Learn More" button, a timer at "00:15", and a "SEE MORE" button. The bottom of the page has a footer with links for "About Us", "Privacy Policy", "Terms of Service", and "Help". The developer tools sidebar on the right shows sections for "Elements", "Console", "Sources", "Network", "Performance", "Memory", and "Application". A status bar at the bottom right shows a magnifying glass icon and the number "40/53".

The screenshot shows a web browser window with the developer tools open, specifically the Elements tab. In the background, there is a sponsored video player for 'Ultra Thin Pads' with a thumbnail showing a product and the text 'Replay' and 'Learn More'. The video duration is 00:15. The main content area has a heading 'Sponsored by Advertising Partner' and a section titled 'Sponsored Video' with the sub-instruction 'Watch to learn more'. Below this is a dark button labeled 'SEE MORE'. At the bottom of the page, there is a status bar with three icons: a square with an 'X', a square with an upward arrow, and a square with a downward arrow.

This type of XPath selector is called an absolute path selector. In absolute path selector, you must traverse the web page from the root (/) of the page. The disadvantage of an absolute path selector is that even a slight change to the web page structure may make your XPath selector invalid. The solution to this problem is a relative or partial XPath selector.

To see how relative path or partial path works, create a new file **web02.html** in the **www/** directory and type in the following lines of codes in it.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Basic HTML Document</title>
</head>
<body>
    <h1>Hello world</h1>
```



2.html file and load it in your web browser.

```
File Edit Selection View Go Run Terminal Help
Sponsored Video
OPEN EDITORS
SELENIUM-XPATH
.venv
SEE MORE
arset="UTF-8">
    me="viewport" content="width=device-width, initial-scale=1.0">
        <title>Basic HTML Document</title>
    </head>
    <body>
        <h1>Hello World</h1>
        <div>
            <p>this is message</p>
        </div>
        <div>
            <span>hello world</span>
        </div>
    </body>
</html>
```

Python 3.8.2 64-bit ('.venv': venv) 0 ▲ 0 Ln 19, Col 8 Spaces: 4 UTF-8 LF HTML ⚡

As you can see, the XPath selector `//div/p` selects the `p` tag inside the `div` tag. This is an example of a relative XPath selector.

Relative XPath selector starts with `//`. Then you specify the structure of the element you want



ment inside a `div` element, does not matter what comes

Sponsored by Advertising Partner

Sponsored Video

Watch to learn more

[SEE MORE](#)

The screenshot shows a browser window titled "Basic HTML Document" displaying a page with a video player at the top and some text below it. The developer tools are open, specifically the "Console" tab, which is highlighted in blue. The console interface includes tabs for "Sources", "Network", "Performance", "Memory", and "Application". In the bottom right corner of the developer tools, there are three small icons: a close button (X), a back arrow, and a forward arrow.

The main content area shows the HTML structure:

```
<!DOCTYPE html>
<html>
  <head>
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <h1>Hello World</h1>
    <div>
      <p>this is message</p>
    </div>
    <div>
      <span>hello world</span>
    </div>
  </body>
</html>
```

A red arrow points from the text "this is message" in the `<p>this is message</p>` line to the `//div/p` selector in the developer tools' search bar. Another red arrow points from the search bar to the `//div/p` text itself, which is highlighted with a red border.

You can also select elements by different attributes like **id**, **class**, **type**, etc. using XPath selector. Let's see how to do that.

Create a new file **web03.html** in the **www/** directory and type in the following lines of codes in



```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>
<h1>Hello world</h1>
Sponsored by Advertising Partner
<div class="container1">
    <p>this is message</p>
    <span>another message</span>
</div>
<div class="container1">
    <h2>heading 2</h2>
    <p>Lorem ipsum dolor sit amet consectetur, adipisicing elit. Quibusdam
       estias quos quae non nam incident quis delectus
       atque fuga? Unde, aut natus?</p>
    <a href="#">SEE MORE</a>
</div>
<footer>
    <span id="footer-msg">this is a footer</span>
</footer>
</body>
</html>
```

Once you're done, save the **web03.html** file and load it in your web browser.

```
html
ml>
en">

charset="UTF-8">
name="viewport" content="width=device-width, initial-scale=1.0">
basic HTML Document</title>

<h1>Hello World</h1>
<div class="container1">
    <p>this is message</p>
    <span>this is another message</span>
</div>
<div class="container1">
    <h2>heading 2</h2>
    <p>Lorem ipsum dolor sit amet consectetur, adipisicing elit. Quibusdam
        endi doloribus sapiente, molestias quos quae non nam incidunt quis
        ectus facilis magni officiis alias neque atque fuga? Unde, aut natus?</p>
</div>
<footer>
    <span id="footer-msg">this is a footer</span>
</footer>
</body>
</html>
```

Let's say you want to select all the **div** elements which have the **class** name **container1**. To do that, you can use the XPath selector **//div[@class='container1']**

As you can see, I have 2 elements which match the XPath selector
//div[@class='container1']

The screenshot shows a browser window with a video player at the top. Below it is a sponsored video section with a heading "Sponsored Video". The main content area contains a "SEE MORE" button and some sample HTML code.

The browser's developer tools are open, specifically the Elements tab. A red arrow points from the text "this is a footer" in the sample HTML to the corresponding element in the DOM tree. Another red arrow points from the highlighted element in the tree down to the search results in the bottom panel. The search bar at the bottom contains the XPath expression `//div[@class='container1']`. The results panel shows "1 of 2" results, with a red box highlighting the first result.

```
<body>
  <h1>Hello World</h1>
  <div class="container1">
    <p>this is message</p>
    <span>this is another message</span>
  </div>
  <div class="container1">
    <h2>heading 2</h2>
    > <p>...</p>
  </div>
  <footer>
    <span id="footer-msg">this is a footer</span>
  </footer>
</body>
</html>
```

To select the first `div` element with the `class` name `container1`, add `[1]` at the end of the XPath select, as shown in the screenshot below.

The screenshot shows the Chrome DevTools interface with the 'Elements' tab selected. The main pane displays the HTML structure of a webpage. A red box highlights the second `div` element with the class name `container1`. The element's content is shown in the preview area:

```
<div[@class='container1'][2]>
```

The page content includes a video player with a play button, a progress bar, and a timer at 00:15. Below the video, there is a 'Learn More' button and a 'SECURITY' section. The footer contains a message: "this is a footer == \$0". The page title is "Sponsored by Advertising Partner" and the overall heading is "Sponsored Video". A 'SEE MORE' button is also visible.

The screenshot shows the Chrome DevTools interface with the 'Elements' tab selected. The main pane displays the HTML structure of a sponsored video ad. The DOM tree highlights several elements:

- A header section with icons for a timer (00:15), a play button, and a 'Learn More' link.
- A main content area containing text and a small image.
- An footer section containing a message: `<p id="footer-msg">this is a footer == $0`.
- A 'SEE MORE' button at the bottom.

The bottom of the DevTools window shows the XPath selector `//div[@class='container'][2]` and a 'Cancel' button.

You can select elements by **id** as well.

For example, to select the element which has the **id** of **footer-msg**, you can use the XPath selector **`//*[@id='footer-msg']`**

Here, the * before `[@id='footer-msg']` is used to select any element regardless of their tag.

The screenshot shows a browser window with developer tools open. The main content area displays a page with a header, a video player, and a footer containing the text "this is a footer". The developer tools interface includes tabs for Elements, Console, Sources, Network, Performance, Memory, Application, Security, Lighthouse, and a settings gear icon. The Elements tab is active, showing the DOM tree. A red arrow points to the XPath selector `//*[@id='footer-msg']` in the search bar at the bottom of the Elements panel. To the right, the Styles panel is open, showing a visual representation of element padding, border, and margin. Another red arrow points to the highlighted element `this is a footer` in the DOM tree.

That's the basics of the XPath selector. Now, you should be able to create your own XPath selector for your Selenium projects.

Conclusion:

In this article, I have shown you how to find and select elements from web pages using the XPath selector with the Selenium Python library. I have also discussed the most common

XPath selectors. After reading this article, you should feel pretty confident selecting elements

selector with the Selenium Python library.



ABOUT THE AUTHOR

Sponsored by Advertising Partner

Shahriar Shovon

Sponsored Video

Watch to learn more Freelancer & Linux System Administrator. Also loves Web API development with Node.js and JavaScript. I was born in Bangladesh. I am currently studying

[SEE MORE](#)

Communication Engineering at Khulna University of Engineering & Technology (KUET), one of the demanding public engineering universities of Bangladesh.

[View all posts](#)

RELATED LINUX HINT POSTS

The screenshot shows a web page with a video player at the top and a sidebar on the left.

Video Player:

- Title: How to Do Testing with Selenium
- Controls: Replay, Learn More, 00:15
- Text: ABSORB 3X FAST than Always® Infinity
- Small text: *Security Ultra Thin Regular Absorbency vs Always® Infinity Size 1 Ultra Thin Pads
Time lapse: two minutes

Sidebar Content:

- Sponsoring Elements by CSS**
Sponsored by Advertising Partner
- Selectors with Selenium**
- Sponsored Video**
- Watch to learn more**
- How to Get the Current URL with Selenium**
- SEE MORE**
- How to Refresh the Page with Selenium**
- How to Wait for a Page to Load with Selenium**

Bottom Right Controls:

- Close button (X)
- Up arrow

