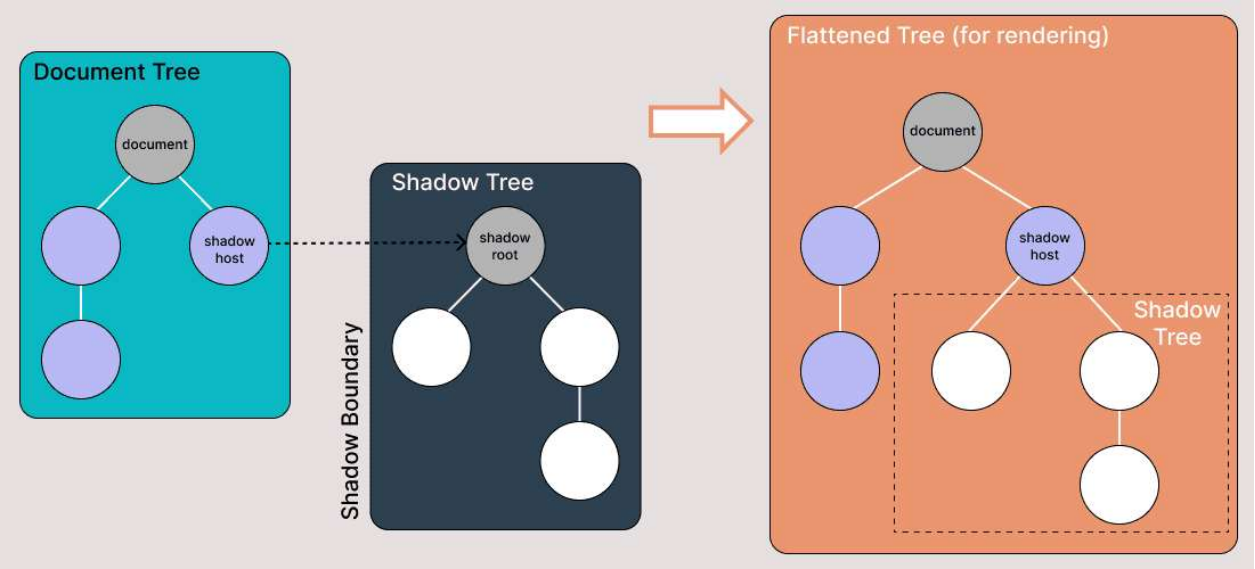
36. Handle Shadow DOM Elements

Shandow DOM

- DOM is Document Object Model generated by browser when loading a website.
- Shadow DOM provides encapsulation in HTML by isolating styles and behavior for specific parts of a document.
- It allows hidden DOM trees, starting with a Shadow Root, where elements can be added like in the regular DOM.
- Shadow DOM acts as a separate DOM within the main DOM, preventing interference with other code.

Document ⇒ Shadow host ⇒ Shadow root ⇒ Shadow Tree(Elements)



- XPath cannot handle shadow dom elements.Only CSS can handle shadow dom elements.

Shadow Host

➔ The Shadow Host is the regular DOM element to which the Shadow DOM is attached.

➔ This is the element where the shadow tree starts.

Shadow Tree

➔ The DOM tree inside the Shadow DOM.

Shadow Root

➔ The Shadow Root is root node of the Shadow DOM tree. It's the point where the shadow tree begins.

➔ When you attach a Shadow DOM to an element, the shadow root is created as the entry point to that hidden DOM structure.

Shadow Boundary

➔ The place where the Shadow DOM ends and the regular DOM begins.

Note

➔ Inside the Shadow Host, Shadow Root serves as parent element for all nodes in the Shadow DOM.

Locating Shadow Dom Elements

➔ Selenium cannot directly access Shadow DOM elements with default methods we need to use **Javascript** or **Shadow Dom selectors**

```
shadow_host = driver.find_element_by_css_selector('#shadow_host')
```

Approach - 1 ⇒ by Javascript

```
shadow_root = driver.execute_script('return arguments[0].shadowRoot', shadow_host)
```

Approach - 2 ⇒ by Shadow Dom selectors

```
shadow_root = shadow_host.shadow_root
shadow_content = shadow_root.find_element(By.CSS_SELECTOR, '#shadow_content')
```

ShadowDOMDemo_1.py

```
import time
from selenium import webdriver
from selenium.webdriver.common.by import By
opt = webdriver.ChromeOptions()
opt.add_experimental_option("detach",True)
opt.add_argument("--start-maximized")
driver = webdriver.Chrome(options=opt)
driver.implicitly_wait(10)
driver.get("https://dev.automationtesting.in/shadow-dom")
```

loacting shadow dom element

locate shadow host

```
shadow_host = driver.find_element(By.CSS_SELECTOR,"#shadow-root")
```

locate shadow root

Approach 1 - javascript

```
shadow_root = driver.execute_script("return arguments[0].shadowRoot",shadow_host)
```

Approach -2 - shadow dom selectors

```
shadow_root = shadow_host.shadow_root
shadow_element =shadow_root.find_element(By.CSS_SELECTOR,"#shadow-element")
print(shadow_element.is_displayed())
```

loacting Nested shadow dom element

```
shadow_host_1 = driver.find_element(By.CSS_SELECTOR,"#shadow-root")
shadow_root_1 = shadow_host_1.shadow_root
shadow_host_2 = shadow_root_1.find_element(By.CSS_SELECTOR,"#inner-shadow-dom")
shadow_root_2 = shadow_host_2.shadow_root
ele =shadow_root_2.find_element(By.CSS_SELECTOR,"#nested-shadow-element")
print(ele.text)
```

loacting Multi-nested Shadow Element

```
shadow_host_1 = driver.find_element(By.CSS_SELECTOR,"#shadow-root")
shadow_root_1 = shadow_host_1.shadow_root
shadow_host_2 = shadow_root_1.find_element(By.CSS_SELECTOR,"#inner-shadow-dom")
shadow_root_2 = shadow_host_2.shadow_root
shadow_host_3 = shadow_root_2.find_element(By.CSS_SELECTOR,"#nested-shadow-dom")
shadow_root_3 = driver.execute_script("return arguments[0].shadowRoot",shadow_host_3)
ele =shadow_root_3.find_element(By.CSS_SELECTOR,"#multi-nested-shadow-element")
print(ele.text)
driver.quit()
```

ShadowDOMDemo_2.py

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

```
options = webdriver.ChromeOptions()
options.add_experimental_option("detach", True)
driver = webdriver.Chrome(options=options)
driver.get("https://books-pwakit.appspot.com/")
driver.maximize_window()
```

Locate the shadow Dom element by CSS Selector → NoSuchElement Exception

```
driver.find_element(By.CSS_SELECTOR, "#input").send_keys("WELCOME")
```

1.Locate the shadow host element

```
shadow_host = driver.find_element(By.CSS_SELECTOR, "book-app[apptitle='BOOKS']")
```

2.Access the shadow root

Approach - 1 ⇒ by Javascript

```
shadow_root = driver.execute_script("return arguments[0].shadowRoot", shadow_host)
```

Approach - 2 ⇒ by Shadow Dom selectors

```
shadow_root = shadow_host.shadow_root
```

3.Find the actual input element within the Shadow DOM and interact with it

```
input_box = shadow_root.find_element(By.CSS_SELECTOR, "#input")
input_box.send_keys("Welcome")
driver.quit()
```

NestedShadowDOMDemo.py

```
from selenium import webdriver
from selenium.webdriver.common.by import By
options = webdriver.ChromeOptions()
options.add_experimental_option("detach", True)
driver = webdriver.Chrome(options=options)
driver.get("https://shop.polymer-project.org/")
driver.implicitly_wait(10)
driver.maximize_window()
```

Shadow Host 1

```
shadow_host1 = driver.find_element(By.CSS_SELECTOR, "shop-app[page='home']")
```

Shadow Root 1

```
shadow_root1 = shadow_host1.shadow_root
```

Shadow Host 2 within Shadow Root 1

```
shadow_host2 = shadow_root1.find_element(By.CSS_SELECTOR, ".iron-selected")
```

Shadow Root 2

```
shadow_root2 = shadow_host2.shadow_root
```

Find the button within Shadow Root 2 and click it

```
button = shadow_root2.find_element(By.CSS_SELECTOR, "a[aria-label=\"Men's Outerwear Shop Now\"]")
button.click()
driver.quit()
```