stale_element_solution.py

Solution 內有一個迴圈，因為該測試頁面有 8 個按鈕。

迴圈內第一段有三個被註解的 `repeat_action`，
第一個採用內建的 `click_action`
第二個採用內建的 `checkbox_action`
第三個採用內建的 `sendkeys_action`

另外要注意的是，待測試頁面只有一個 checkbox 和 input 欄位，
所以會用 break 來只讓 `checkbox_action` 和 `sendkeys_action` 執行一次

```
if __name__ == '__main__':
    chrome_driver = webdriver.Chrome()
    chrome_driver.get('http://localhost:5000/stale_element')

    time.sleep(7)

    for i in range(1, 8+1):

        # Use Build-In Method
        repeat_action(click_action, chrome_driver, {'click': f'#main-table > tbody > tr:nth-child({i}) button'}, 5)
        #repeat_action(checkbox_action, chrome_driver, {'checkbox': '#exampleCheck'}, 5)
        #repeat_action(sendkeys_action, chrome_driver, {'sendkeys': '#exampleInputPassword', 'text': 'password'}, 5)
        #break
```

下面段落提供兩個錯誤的範例
第一個是故意把 dictionary 的 key 打錯，內建的 `click_action` 函式會提示 exception；
第二個則是故意送一個錯誤的 css_selector，`repeat_action` 則會持續 retry。

```
# Wrong Example 1: Build-In function With Wrong Key (Dictionary KeyError)
#repeat_action(click_action, chrome_driver, {'c': f'#main-table > tbody > tr:nth-child({i}) button'}, 5)
#break

# Wrong Example 2: No_Such_Element Exception
#repeat_action(click_action, chrome_driver, {'click': f'#main-table > tbody > tr:nth-child(777) button'}, 5)
#break
```

下面段落提供兩個呼叫自定義 function 的範例，
這邊使用了自定義 function `my_click_action`、`my_sendkey_and_checked_action`

```
# Use Personal Method
#repeat_action(my_click_action, chrome_driver, {}, 5)
#repeat_action(my_sendkey_and_checked_action, chrome_driver, {}, 5)
#break
```

這邊則是 `my_click_action` 的詳細內容，可以只帶 driver 參數，或是可以接 dictionary 的參數。
也提供一個可能仍然會發生重複執行的分支情況，可能造成意外，故不推薦用下面的 Wrong Example 寫法。

```python
def my_click_action(driver):

    btn = driver.find_element_by_css_selector('#main-table > tbody > tr:nth-child(1) button')
    btn.click()
    # print(btn.text) # To verify I get the element definitely


    # --- Wrong Exmaple: Repeatedly re-operate on manipulated objects ---
    # for i in range(1, 9):
    #     btn = driver.find_element_by_css_selector(f'#main-table > tbody > tr:nth-child({i}) button')
    #     btn.click()
```

hidden_element_solution.py

程式開頭有一些註解，是簡單的原理說明以及避免發生意外狀況
(第一段註解是讓頁面移到最上面，第二段註解則是簡單的原理說明)

```python
if __name__ == '__main__':
    driver = webdriver.Chrome()
    driver.get('http://localhost:5000/hidden_element2')

    # === for go to the top ===
    # driver.find_element_by_tag_name('body').send_keys(Keys.HOME)
    # time.sleep(5)

    # === Click Basic Script ===
    #btn = driver.find_element_by_id('target-btn')
    #driver.find_element_by_id('target-btn').click()
    #driver.execute_script("arguments[0].click();", btn)
```

其實呼叫只要一行 `click_hidden_by_selector`，
不過因為該測試頁面需要切換到下一頁，所以有一個 click next_button 的動作。

```python
# === Experiment ===
try:
    # === p=1 ===
    click_hidden_by_selector(driver, {'click': '#target-btn'})
    next_btn = WebDriverWait(driver, 2).until(
        EC.element_to_be_clickable((By.ID, "next-btn"))
    )
    next_btn.click()

    # === p=2 ===
    click_hidden_by_selector(driver, {'click': '#target-btn'})
    next_btn = WebDriverWait(driver, 2).until(
        EC.element_to_be_clickable((By.ID, "next-btn"))
    )
    next_btn.click()
```

有時候 selector 會因為回傳的是 array，無法簡單透過參數傳遞來指定你要第幾個
所以直接採用原理的方式來寫也很推薦。
原理的 Example 有在程式最一開始的註解內。

```
# === p=3 ===
btn_p3 = driver.find_elements_by_css_selector('a.dropdown-item.dropdown-item-warning')[0]
driver.execute_script("arguments[0].click();", btn_p3)
next_btn = WebDriverWait(driver, 2).until(
    EC.element_to_be_clickable((By.ID, "next-btn"))
)
next_btn.click()
```

也有對 xpath 的方式提供支持，如果用不習慣 css_selector 也可以試試這個 function
或是用原理下去撰寫，把抓 element 改成 xpath 抓取就可以。

```
# === p=4 ===
click_hidden_by_selector(driver, {'click': '#target-checkbox'})
next_btn = WebDriverWait(driver, 2).until(
    EC.element_to_be_clickable((By.ID, "next-btn"))
)
next_btn.click()

# === p=5 ===
click_hidden_by_xpath(driver, {'click': "//select/option[text()='Click Me!']"})
print('last')
```