

# REFERENSI CEPAT SELENIUM WEBDRIVER

Otomasi browser, interaksi elemen, wait, dan assertions

## Setup

### Instalasi

```
pip install selenium webdriver-manager
# webdriver-manager auto-downloads browser drivers
```

### Setup Driver Dasar

```
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager
driver = webdriver.Chrome(
    service=Service(ChromeDriverManager().install()))
```

### Mode Headless

```
options = webdriver.ChromeOptions()
options.add_argument("--headless=new")
options.add_argument("--no-sandbox")
driver = webdriver.Chrome(options=options)
```

### Browser yang Didukung

**webdriver.Chrome()** Google Chrome / Chromium  
**webdriver.Firefox()** Mozilla Firefox (GeckoDriver)  
**webdriver.Edge()** Microsoft Edge (Chromium)  
**webdriver.Safari()** Apple Safari (hanya macOS)

## Browser & Navigasi

### Navigasi

```
driver.get("https://example.com")
driver.back() # browser back
driver.forward() # browser forward
driver.refresh() # reload page
```

### Properti Browser

**driver.title** Judul halaman saat ini  
**driver.current\_url** URL halaman saat ini  
**driver.page\_source** Sumber HTML halaman lengkap  
**driver.get\_cookies()** Daftar semua cookie

### Manajemen Window

```
driver.set_window_size(1920, 1080)
driver.maximize_window()
driver.minimize_window()
driver.quit() # close all windows, end session
```

## Menemukan Elemen

### Strategi Locator

```
from selenium.webdriver.common.by import By
driver.find_element(By.ID, "login-btn")
driver.find_element(By.CLASS_NAME, "nav-item")
driver.find_element(By.CSS_SELECTOR, "div.card > h2")
driver.find_element(By.XPATH, "//input[@name='q']")
```

### Strategi By

**By.ID** Cocokkan atribut id elemen  
**By.NAME** Cocokkan atribut name elemen  
**By.CLASS\_NAME** Cocokkan class CSS (satu class)  
**By.TAG\_NAME** Cocokkan nama tag HTML  
**By.CSS\_SELECTOR** CSS selector (paling fleksibel)  
**By.XPATH** Ekspresi XPath  
**By.LINK\_TEXT** Teks anchor persis  
**By.PARTIAL\_LINK\_TEXT** Pencocokkan sebagian teks anchor

### Temukan Beberapa Elemen

```
items = driver.find_elements(By.CSS_SELECTOR, "li.item")
for item in items:
    print(item.text)
# Returns empty list if none found (no exception)
```

## Berinteraksi

### Klik & Ketik

```
elem = driver.find_element(By.ID, "search")
elem.clear() # clear existing text
elem.send_keys("selenium python")
elem.submit()
```

### Dropdown

```
from selenium.webdriver.support.ui import Select
select = Select(driver.find_element(By.ID, "country"))
select.select_by_visible_text("Canada")
select.select_by_value("ca")
select.select_by_index(2)
```

### Properti Elemen

**text** Konten teks yang terlihat  
**get\_attribute('href')** Nilai atribut HTML  
**is\_displayed()** True jika elemen terlihat  
**is\_enabled()** True jika elemen interaktif  
**is\_selected()** True jika checkbox/radio dipilih

**tag\_name** Tag HTML (mis. 'input', 'div')  
**value\_of\_css\_property('color')** Nilai properti CSS yang dihitung

## Wait

### Explicit Wait

```
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
elem = WebDriverWait(driver, 10).until(
    EC.presence_of_element_located((By.ID, "result")))
```

### Expected Conditions

**presence\_of\_element\_located** Elemen ada di DOM  
**visibility\_of\_element\_located** Elemen terlihat di halaman  
**element\_to\_be\_clickable** Elemen terlihat dan aktif  
**text\_to\_be\_present\_in\_element** Elemen memuat teks yang diharapkan

**alert\_is\_present** Alert JavaScript sedang tampil

**staleness\_of** Elemen tidak lagi ada di DOM  
**title\_contains** Judul halaman memuat teks

## Implicit Wait

```
driver.implicitly_wait(10) # seconds, applies globally
# Explicit waits are preferred - more precise control
```

## Frame & Window

### Frame

```
driver.switch_to.frame("frame-name") # by name/id
driver.switch_to.frame(0) # by index
driver.switch_to.frame(elem) # by element
driver.switch_to.default_content() # back to main
```

### Window & Tab

```
original = driver.current_window_handle
driver.switch_to.new_window("tab") # open new tab
driver.switch_to.window(original) # switch back
driver.close() # close current tab
```

### Alert

```
alert = driver.switch_to.alert
print(alert.text)
alert.accept() # click OK
alert.dismiss() # click Cancel
alert.send_keys("input text")
```

## Screenshot

### Ambil Screenshot

```
driver.save_screenshot("page.png") # full page
elem = driver.find_element(By.ID, "chart")
elem.screenshot("chart.png") # single element
```

### Screenshot sebagai Base64

```
b64 = driver.get_screenshot_as_base64()
png = driver.get_screenshot_as_png() # bytes
```

## Actions

### Action Chains

```
from selenium.webdriver.common.action_chains import ActionChains
actions = ActionChains(driver)
actions.move_to_element(menu).click().perform()
```

### Aksi Keyboard

```
from selenium.webdriver.common.keys import Keys
elem.send_keys(Keys.ENTER)
elem.send_keys(Keys.CONTROL, "a") # select all
actions.key_down(Keys.SHIFT).click(elem).perform()
```

### Aksi Mouse

**click(elem)** Klik elemen  
**double\_click(elem)** Klik ganda elemen  
**context\_click(elem)** Klik kanan elemen  
**move\_to\_element(elem)** Arahkan kursor ke elemen  
**drag\_and\_drop(src, dst)** Seret dari sumber ke tujuan  
**click\_and\_hold(elem)** Tekan dan tahan tombol mouse  
**release()** Lepas tombol mouse

## Assertions

### Assertions Umum (pytest)

```
assert "Dashboard" in driver.title
assert driver.find_element(By.ID, "msg").text == "Done"
assert driver.current_url.endswith("/home")
assert len(driver.find_elements(By.CSS_SELECTOR, "tr")) > 0
```

### Assertions Berbasis Wait

```
WebDriverWait(driver, 5).until(
    EC.visibility_of_element_located((By.ID, "success")))
WebDriverWait(driver, 5).until(
    EC.invisibility_of_element_located((By.ID, "spinner")))
```

## Eksekusi JavaScript

```
result = driver.execute_script("return document.title")
driver.execute_script(
    arguments[0].scrollIntoView(true);", elem)
```

## Pola Umum

### Pola Page Object

```
class LoginPage:
    URL = "/login"
    user_loc = (By.ID, "username")
    def login(self, drv, user, pwd):
        drv.find_element(*self.user_loc).send_keys(user)
```

### Context Manager

```
from selenium import webdriver
with webdriver.Chrome() as driver:
    driver.get("https://example.com")
    print(driver.title)
# driver.quit() called automatically
```

### Retry & Cleanup

```
try:
    driver.get("https://example.com")
    WebDriverWait(driver, 10).until(
        EC.element_to_be_clickable((By.ID, "btn")))
finally: driver.quit()
```