

TESTIRANJE SOFTVERA - VEŽBE 07

SELENIUM WEBDRIVER

E2E TESTIRANJE

- ▶ End-to-End testiranje je metodologija koja se koristi u životnom ciklusu razvoja softvera kako bi se testirale funkcionalnosti i performanse aplikacije u okolnostima koje su što sličnije produkcionim
- ▶ Cilj je simulirati korisnički scenario korišćenja aplikacije od početka do kraja
- ▶ Pored toga što se validira rad sistema koji se testira, potrebno je osigurati i da svi podsistemi u okviru aplikacije funkcionišu ispravno (softver, hardver, mreža, baze podataka itd.)
- ▶ U suštini, test prolazi kroz svaku operaciju koju aplikacija može da izvede da bi testirala kako aplikacija komunicira sa hardverom, mrežnom vezom, spoljnim zavisnostima, bazama podataka i drugim aplikacijama. Obično se E2E testiranje izvršava nakon što je funkcionalno testiranje završeno

BEST PRACTICES

- Dajte prioritet krajnjoj upotrebi
 - Tokom kreiranja test slučajeva, testirajte kao korisnik. Uđite u način razmišljanja nekoga ko prvi put koristi aplikaciju
 - Fokusirajte E2E testove na funkcije aplikacije čiji će neuspeh izazvati maksimalne probleme
- Izbegavajte testiranje izuzetaka
 - E2E testiranje se koristi za testiranje uobičajenih korisničkih scenarija. Za testiranje izuzetaka i specijalnih situacija koristite integracione ili jedinične testove
- Vodite računa o redosledu izvršavanja
 - Pošto E2E testiranje obuhvata celu aplikaciju, testni slučajevi teže ka tome da postanu suviše komplikovani
 - U ovoj vrsti testiranja je dozvoljeno da testovi zavise od izvršavanja drugih testova
- Optimizujte setup i teardown mehanizme
 - Uverite se da je okruženje za testiranje spremno za početak testiranja u svakom trenutku
 - Procesi podešavanja moraju biti što je moguće minimalniji
 - Kada se testovi završe, trebalo bi da bude potrebno jednako malo vremena da se izbrišu podaci testa kako bi se okruženje vratilo u prvobitno stanje – tako da je spremno za ponovno sprovođenje testova

ŠTA JE SELENIUM?

- Selenium je besplatan, open-source alat koji se koristi za automatsko testiranje web aplikacija
- Selenium WebDriver omogućava direktnu interakciju sa pretraživačima putem testnih skripti
- Selenium WebDriver podržava web pretraživače kao što su:
 - Mozilla Firefox
 - Google Chrome
 - Internet Explorer
 - Safari
 - Opera
- Takođe je kompatibilan sa nekoliko programskih jezika:
 - Java
 - Python
 - Ruby
 - Perl
 - JavaScript
- Što se tiče operativnih sistema, Selenium WebDriver podržava Windows, Linux, Mac OS i Solaris



SELENIUM TOOL SUITE

- ▶ Alati unutar Selenium-a:

- ▶ Selenium Integrated Development Environment (IDE)
 - ▶ Firefox ekstenzija koja omogućava snimanje i reprodukciju testova
- ▶ Selenium Remote Control (RC)
 - ▶ Preteča web driver-a (Selenium 2.0)
 - ▶ Zahtevao je dosta programerske veštine za pisanje testova
 - ▶ Testovi nisu bili razumljivi i čitki
- ▶ Selenium WebDriver
 - ▶ Naslednik RC-a I jedna od najznačajnijih komponenti Seleniuma
 - ▶ Omogućava izvršavanje testova nezavisno od platforme, jezika, browser-a
- ▶ Selenium Grid
 - ▶ Podrška za paralelno testiranje različitih testova na različitim platformama i u različitim browser-ima

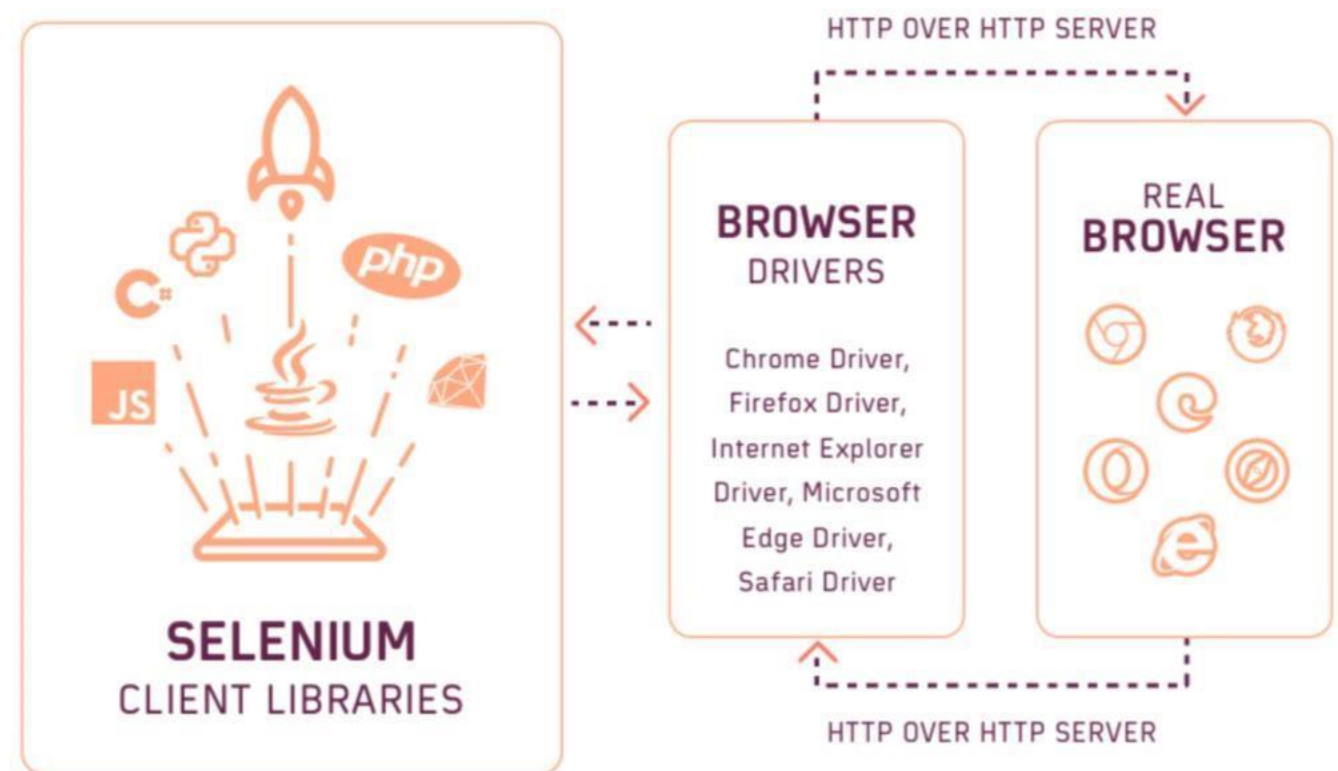


WEBDRIVER

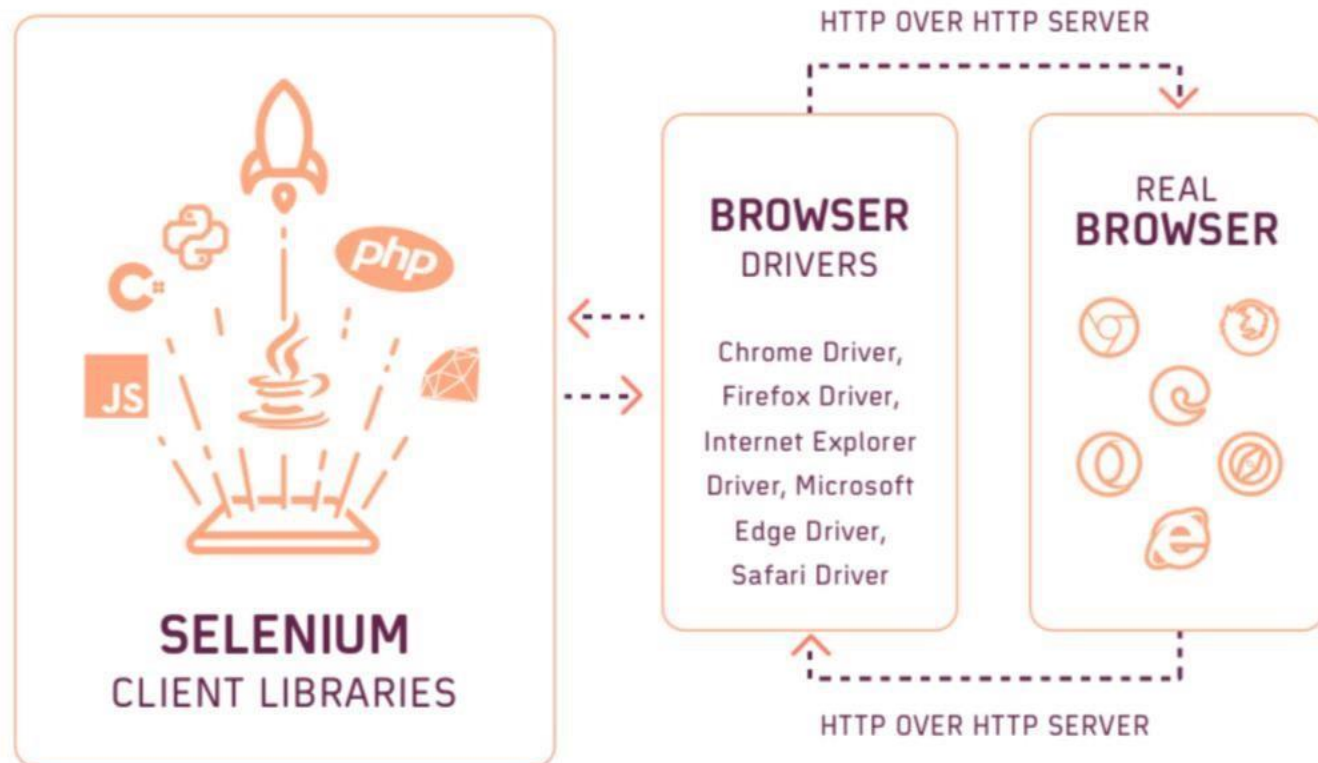
- ▶ Selenium Web Driver predstavlja jednu od najvažnijih komponenti Selenium Tool Suite-a
- ▶ Poznatiji je pod nazivom Selenium 2.0
- ▶ Posедуje ugrađenu podršku za Firefox, dok je za druge browser-e potrebno omogućiti odgovarajuće driver-e
- ▶ Selenium WebDriver se sastoji od 4 glavne komponente:
 1. Selenium klijentske biblioteke
 2. JSON Wire Protocol
 3. Upravljački programi pretraživača
 4. Pravi pretraživači
- ▶ Driver za [Chrome](#)
- ▶ Driver za [Mozzilu](#)

WEBDRIVER

- ▶ Selenium Client Libraries
 - ▶ omogućava izvršavanje automatskih skripti nad web pretraživačima
 - ▶ ponašaju se kao interpreteri između test skripti i pretraživača
 - ▶ zadužen je za mogućnost pisanja Selenium skripti na raznim jezicima
- ▶ JSON Wire Protocol
 - ▶ odgovoran za komunikaciju sa drajverima pretraživača
 - ▶ preuzima informacije iz Selenium klijentskih biblioteka, a zatim ih prenosi odgovarajućem drajveru pretraživača



WEBDRIVER



- ▶ Browser Drivers

- ▶ svaki pretraživač ima drajver koji je odgovoran za kontrolu radnji koje se obavljaju u tom pretraživaču
- ▶ nakon što se putem JSON Wire Protocol prenese informacije u drajver, te iste naredbe se prosleđuju pravom pretraživaču putem HTTP protokola

- ▶ Browsers

- ▶ Selenium WebDriver omogućava interakciju sa različitim web pretraživačima, kao što su Google Chrome, Mozilla Firefox, Safari, Opera, Microsoft Edge...

OSNOVNI KORACI PISANJA SELENIUM TESTOVA

1. Kreiranje instance Selenium WebDriver-a
2. Konfiguracija pretraživača (opciono)
 - na primer, maximizacija pretraživač, onemogućavanje notifikacija itd.
3. Navigacija do tražene URL stranice
4. Lociranje web elemenata
5. Izvršavanje akcija nad lociranim elementima
6. Verifikacija i validacija izvršenih akcija
7. Slikanje stranice i generisanje izveštaja

WEBDRIVER INSTANCE

For Firefox

```
1 System.setProperty("WebDriver.gecko.driver", "C:\\Users\\shalini\\Downloads\\geckodriver-v0.26.0-win64\\geckodriver.exe");  
2 WebDriver driver = new FirefoxDriver();
```

For Chrome

```
1 System.setProperty("WebDriver.chrome.driver", "C:\\Users\\shalini\\Downloads\\chromedriver\\chromedriver.exe");  
2 WebDriver driver = new ChromeDriver();
```

For Edge

```
1 System.setProperty("WebDriver.edge.driver", "C:\\Users\\shalini\\Downloads\\edgedriver_win64\\msedgedriver.exe");  
2 WebDriver driver = new EdgeDriver();
```

For Opera

```
1 System.setProperty("WebDriver.opera.driver", "C:\\Users\\shalini\\Downloads\\operadriver\\operadriver_win64\\operadriver.exe");  
2 WebDriver driver = new OperaDriver();
```

For Internet Explorer

```
1 System.setProperty("WebDriver.ie.driver", "C:\\Users\\shalini\\Downloads\\ieDriver\\IEDriverServer.exe");  
2 WebDriver driver = new InternetExplorerDriver();
```

KOMANDE

Komanda	Opis
<code>driver.get("URL")</code>	To navigate to an application.
<code>element.sendKeys("inputtext")</code>	Enter some text into an input box.
<code>element.clear()</code>	Clear the contents from the input box.
<code>select.deselectAll()</code>	Deselect all OPTIONS from the first SELECT on the page.
<code>select.selectByVisibleText("some text")</code>	Select the OPTION with the input specified by the user.
<code>driver.switchTo().window("windowName")</code>	Move the focus from one window to another.
<code>driver.switchTo().frame("frameName")</code>	Swing from frame to frame.
<code>driver.switchTo().alert()</code>	Helps in handling alerts.
<code>driver.navigate().to("URL")</code>	Navigate to the URL.
<code>driver.navigate().forward()</code>	To navigate forward.
<code>driver.navigate().back()</code>	To navigate back.
<code>driver.close()</code>	Closes the current browser associated with the driver.
<code>driver.quit()</code>	Quits the driver and closes all the associated window of that driver.
<code>driver.refresh()</code>	Refreshes the current page.

LOKATORI

- ▶ Jedna od najvažnijih stvari u korišćenju WebDriver-a je pronalaženje elemenata na stranici

```
WebElement findElement(By by)
```

- ▶ Ulazni parametar je instanca `By` objekta, koji predstavlja mehanizam za lociranje elemenata stranice
- ▶ Povratna vrednost je objekat klase `WebElement` koji je reprezentacija HTML elementa
- ▶ **NAPOMENA:** metoda vraća prvi web element koji zadovoljava kriterijum pretrage
- ▶ Ukoliko WebDriver ne pronade element, baciće `NoSuchElementException`
- ▶ Primer:

```
WebElement cheese = driver.findElement(By.id("cheese"));
```

LOKATORI

- Za pronalaženje kolekcije elemenata, koristi se metoda

```
java.util.List findElements(By by)
```

- Ulazni parametar je instanca `By` objekta, kao i u prethodnoj metodi
- Povratna vrednost lista `WebElement` objekata
- **NAPOMENA:** metoda će vratiti listu svih elemenata koji zadovoljavaju kriterijum pretrage
- Ukoliko `WebDriver` ne pronađe nijedan element, povratna vrednost će biti prazna lista
- Primer:

```
List<WebElement> muchoCheese = driver.findElements(By.cssSelector("#cheese li"));
```

LOKATORI

Locator	Description	Syntax (in Java)
ID	Identify the WebElement using the <i>ID</i> attribute	<code>driver.findElement(By.id("IdValue"));</code>
Name	Identify the WebElement using the <i>Name</i> attribute	<code>driver.findElement(By.name("nameValue"));</code>
ClassName	Use the <i>Class</i> attribute for identifying the object	<code>driver.findElement(By.className("classValue"));</code>
LinkText	Use the text in hyperlinks to locate the WebElement	<code>driver.findElement(By.linkText("textofLink"));</code>
Partial LinkText	Use a part of the text in hyperlinks to locate the desired WebElement	<code>driver.findElement(By.partialLinkText("PartialTextofLink"));</code>
TagName	Use the TagName to locate the desired WebElement	<code>driver.findElement(By.tagName("htmlTag"));</code>
CssSelector	CSS used to create style rules in web page is leveraged to locate the desired WebElement	<code>driver.findElement(By.cssSelector("cssValue"));</code>
XPath	Use XPath to locate the WebElement	<code>driver.findElement(By.xpath("xpathValue"));</code>

PROVERA OSOBINA ELEMENTA

- Selenium WebDriver nam omogućava preuzimanje i proveru osobina HTML elemenata
- *String getAttribute(String name)*
 - Omogućava preuzimanje vrednosti bilo kog atributa
- *String getText()*
 - Vraća vidljivi deo texta WebElement-a ukoliko postoji
- *String getCssValue(String propertyName)*
 - Omogućava preuzimanje bilo koje vrednosti CSS-a nad elementom (font-family, background-color, color...)
- *Point getLocation()*
 - Vraća koordinate elementa na web stranici. Pozicija se određuje relativno u odnosu na gornji levi ugao, čije su koordinate (0,0)
- *Dimension getSize()*
 - Vraća dimenzije, tj. širinu i visinu elementa
- *String getTagName()*
 - Vraća naziv HTML elementa odgovarajućeg WebElementa

IZVRŠAVANJE AKCIJA NAD ELEMENTIMA

- ▶ `void sendKeys(CharSequence keysToSend)`
 - ▶ Simulira unos teksta sa tastature od strane korisnika
 - ▶ Ukoliko je potrebno simulirati unos posebnih tipki (Backspace, Enter, Tab, Shift...), koristi se enumeracija *Keys*
 - ▶ Primenjiv nad `textBox` i `textArea` elementima
- ▶ `void clear()`
 - ▶ Briše tekstualne unose nad `textBox` i `textArea` elementima

PROVERA STANJA ELEMENATA

- ▶ `boolean isDisplayed()`
 - ▶ Proverava prisutnosti elementa na stranici
 - ▶ Primenjiv nad svim HTML elementima
- ▶ `boolean isEnabled()`
 - ▶ Proverava da li je omogućen pristup elementu (u smislu interakcije nad njim)
 - ▶ Primenjiv nad svim HTML elementima
- ▶ `boolean isSelected()`
 - ▶ Proverava da li je element selektovan
 - ▶ Primenjiv samo nad radio button, option i checkbox elementima

INTERAKCIJE

- ▶ Selenium WebDriver nam omogućava direktnu interakciju sa elementima HTML stranice

- 1. Text Box Interaction**

- 2. Radio Button Selection**

- 3. Drop Down Item Selection**

- 4. Check Box Selection**

5. Drag & Drop

6. Keyboard Actions

7. Mouse Actions

8. Multi Select

9. Find All Links

INTERAKCIJE

- ▶ Selenium WebDriver nam omogućava direktnu interakciju sa elementima HTML stranice
 - 1. Text Box Interaction
 - 2. Radio Button Selection
 - 3. Drop Down Item Selection
 - 4. Check Box Selection
 - 5. Drag & Drop**
 - 6. Keyboard Actions
 - 7. Mouse Actions
 - 8. Multi Select
 - 9. Find All Links

INTERAKCIJE

- ▶ Keyboard Actions se izvodi pomoću jedne od 3 metode:
 - ▶ `sendKeys` – simulacija unosa teksta preko tastature. Posebni karakteri koji nisu tekst, takođe mogu biti uneti putem ove metode
 - ▶ `pressKey` – simulacija pritiskanja karaktera koji nisu tekst. Pod `pressKey` podrazumevamo korišćenje dugmića kao što su "F1", "F2", "Tab", "Control" itd.
 - ▶ `releaseKey` – simulacija otpuštanja dugmeta na tastaturi nakon `keyPress` događaja

```
void sendKeys(java.lang.CharSequence keysToSend)
void pressKey(java.lang.CharSequence keyToPress)
void releaseKey(java.lang.CharSequence keyToRelease)
```

INTERAKCIJE

- ▶ Selenium WebDriver nam omogućava direktnu interakciju sa elementima HTML stranice
 - 1. Text Box Interaction
 - 2. Radio Button Selection
 - 3. Drop Down Item Selection
 - 4. Check Box Selection
 - 5. Drag & Drop
 - 6. Keyboard Actions**
 - 7. Mouse Actions
 - 8. Multi Select**
 - 9. Find All Links

INTERAKCIJE

- ▶ Mouse Actions se izvodi pomoću jedne od 6 metoda:
 - ▶ `click` – simulira klik na element ili na tačno određene koordinate
 - ▶ `contextClick` – kontekstni, desni klik na element ili na tačno određene koordinate
 - ▶ `doubleClick` – simulira dvoklik na element ili na tačno određene koordinate
 - ▶ `mouseDown` – simulira pritisak na taster miša
 - ▶ `mouseMove` – simulira pomeranje miša
 - ▶ `mouseUp` – simulira otpuštanje tastera miša

```
void click(WebElement onElement)
void contextClick(WebElement onElement)
void doubleClick(WebElement onElement)
void mouseDown(WebElement onElement)
void mouseUp(WebElement onElement)
void mouseMove(WebElement toElement)
void mouseMove(WebElement toElement, long xOffset, long yOffset)
```

INTERAKCIJE

- ▶ Selenium WebDriver nam omogućava direktnu interakciju sa elementima HTML stranice
 - 1. Text Box Interaction
 - 2. Radio Button Selection
 - 3. Drop Down Item Selection
 - 4. Check Box Selection
 - 5. Drag & Drop
 - 6. Keyboard Actions
 - 7. Mouse Actions**
 - 8. Multi Select
 - 9. Find All Links

INTERAKCIJE

- ▶ Selenium WebDriver nam omogućava direktnu interakciju sa elementima HTML stranice
 - 1. Text Box Interaction
 - 2. Radio Button Selection
 - 3. Drop Down Item Selection
 - 4. Check Box Selection
 - 5. Drag & Drop
 - 6. Keyboard Actions
 - 7. Mouse Actions
 - 8. Multi Select
 - 9. Find All Links**

HEADLESS MODE

- ▶ Headless mode omogućava izvršavanje testova bez prikazivanja UI komponenti (testovi se izvršavaju u pozadini)
- ▶ Ovim je omogućeno brže izvršavanje testova, naročito u CI okruženjima

```
@BeforeMethod
public void setup() {
    System.setProperty("webdriver.chrome.driver",
        "./src/test/resources/drivers/chromedriver");

    ChromeOptions chromeOptions = new ChromeOptions();
    chromeOptions.setHeadless(true);

    driver = new ChromeDriver(chromeOptions);

    driver.get("http://demo-store.seleniumacademy.com/");
}
```

IZUZECI PRILIKOM SELENIUM TESTIRANJA

- ▶ Izuzeci su događaji koji uzrokuju da se Java program naglo završi bez davanja očekivanog rezultata. Ispod su najčešći izuzeci koji se javljaju prilikom izvršavanja testova u Selenium WebDriver-u:
- ▶ `org.openqa.selenium.NoSuchElementException`
 - ▶ javlja kada Selenium WebDriver ne može da locira element
- ▶ `java.lang.IllegalStateException: The driver executable does not exist:`
 - ▶ javlja kada driver nije pronađen na navedenoj putanji
- ▶ `org.openqa.selenium.firefox.NotConnectedException: Unable to connect to host 127.0.0.1 on port 7055 after 45000ms`
 - ▶ javlja se kod nekompatibilnosti verzije pretraživača i Seleniuma
 - ▶ proverite kompatibilnost pretraživača i drajvera na stranici za preuzimanje drajvera
- ▶ `org.openqa.selenium.StaleElementReferenceException`
 - ▶ ovaj izuzetak može nastati kada web element više nije prisutan na web stranici. Često zna biti i posledica izmene DOM stabla