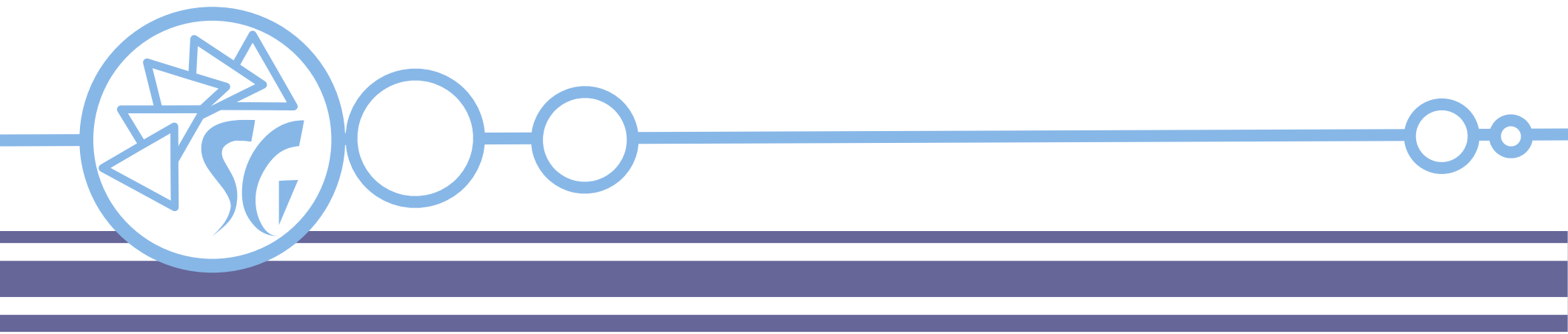


Selenium IDE

Ing. Simone Giustetti

www.giustetti.net

Selenium IDE

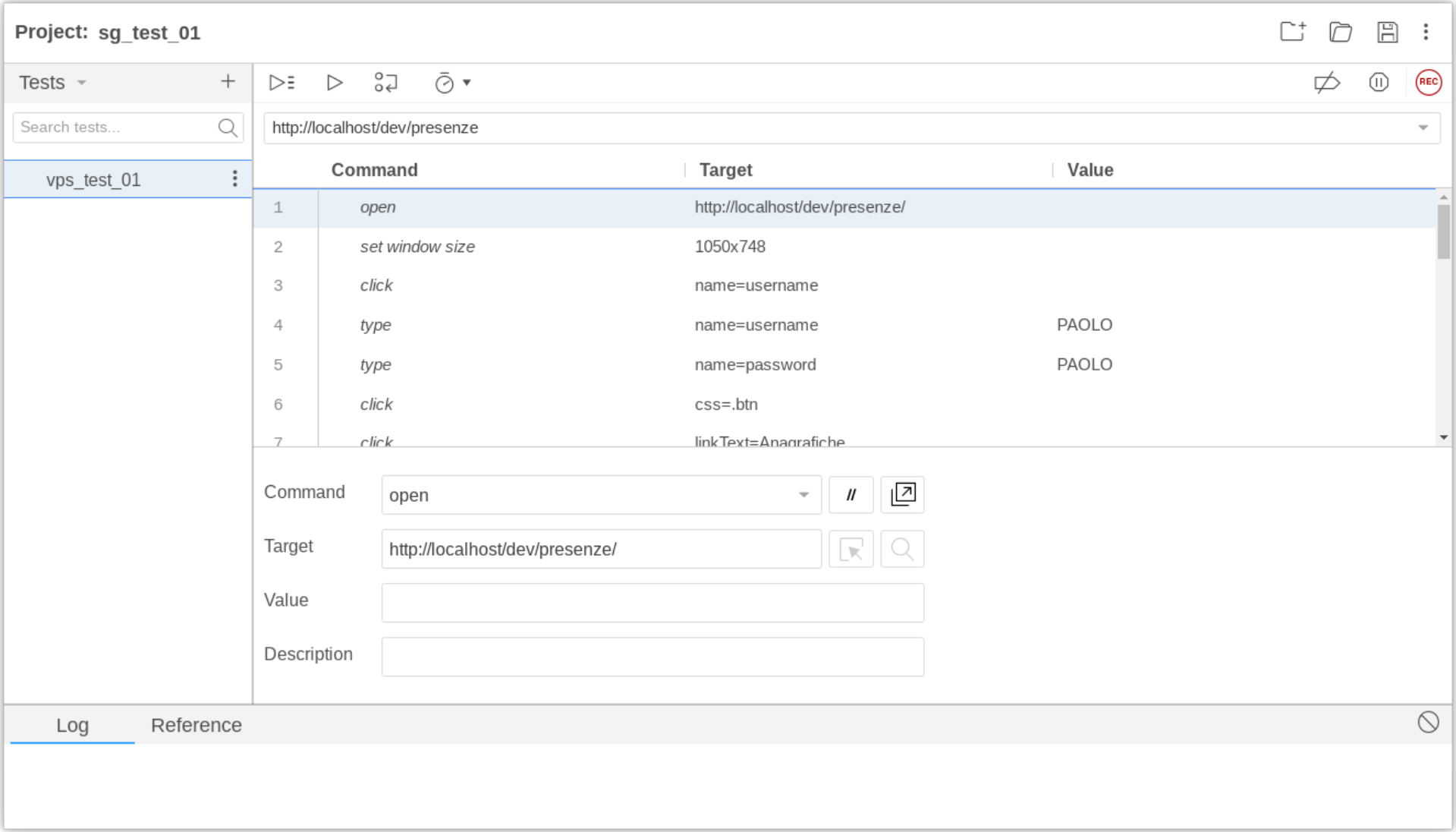
Interfaccia grafica che si installa su un browser.

Pensata per utenti non programmatori.

Consente di registrare, salvare ed eseguire test di applicazioni web mediante mouse e tastiera.

Tutti i test sono eseguiti direttamente sul browser.

Selenium IDE



Installazione

Selenium IDE è distribuito come un componente di un browser.

Esiste sotto forma di:

- Estensione per Chrome.
- Add-on per Firefox.

Chrome: chrome://extensions

Firefox: about:addons

Registrazione di un Test

Prima di poter registrare un test, è necessario impostare lo **URL** dell'applicazione che si andrà a testare.

- Fare click sul pulsante "**REC**".
- Verrà aperta una finestra del browser che punta allo URL impostato.
- Eseguire tutte le interazioni come se si stesse navigando il sito.
- Fermare la registrazione.
- Salvare il test registrato.

Salvare i Test

I **test** possono essere salvati in appositi file aventi estensione ***.side**.

Le **suite dei test** posso essere esportate in un linguaggio di programmazione tra:

- C# + NUnit
- C# + xUnit
- Java + JUnit
- JavaScript + Mocha
- Python + pytest
- Ruby + RSpec

Categorie dei Comandi

I comandi di Selenium IDE ricadono in 3 categorie:

- **Azioni**: Cambiano lo stato dell'applicazione;
- **Controlli**: Verificano lo stato dell'applicazione e immagazzinano la risposta;
- **Affermazioni**: Eseguono comparazioni tra stato atteso e rilevato.

Categorie delle Affermazioni

Le affermazioni utilizzabili in Selenium IDE ricadono in 3 categorie:

- **Assert**: Causa la terminazione immediata dello script in caso di test fallito;
- **Verify**: Consente di proseguire lo script in caso di test fallito;
- **Wait for**: Attende che una condizione sia verificata prima di proseguire con il passo successivo dello script.

Alcuni Comandi di Uso Comune

- assertElementPresent, verifyElementPresent;
- assertTextPresent, verifyTextPresent;
- assertTitle, VerifyTitle;
- chooseOkOnNextConfirmation, chooseCancelOnNextConfirmation;
- click, clickAt, clickAndWait;
- open;
- type, typeKeys, sendKeys;
- waitForElementPresent;
- waitForPageToLoad.

Istruzioni Condizionali

Selenium IDE dispone di comandi in grado di controllare il flusso di un test.

```
if
else if
else
end
```

Le condizioni possono essere espresse nella forma di codice JavaScript.

Popolare le Variabili

È possibile definire e popolare variabili mediante le istruzioni **store**<...>:

- **store**: Salva una stringa di testo in una variabile;
- **store attribute**: Salva il valore a cui è impostato l'attributo di un elemento;
- **store text**: Salva il testo di un elemento;
- **store title**: Salva il titolo della pagina;
- **store value**: Salva il valore di un elemento.

Risultati di uno Script

È possibile eseguire script JavaScript e salvare i risultati in variabili.

Gli script possono essere richiamati mediante i comandi:

- execute script
- execute async script

Cicli

È possibile eseguire parti di un test più volte utilizzando apposite istruzioni:

- times … end (for)
- do … repeat if
- while … end
- for each … end

Cicli ed istruzioni condizionali possono essere annidati.

Command-Line Runner

Esiste una interfaccia a riga di comando per Selenium IDE che si chiama **Command-line Runner**.

Consente di eseguire test senza appoggiarsi al browser.

Può eseguire test su più browser in parallelo.

Deve essere installato indipendentemente da IDE.

Command-line Runner - Installazione

Selenium CLR necessita la presenza di **NodeJS** sulla macchina e di **npm** o altro gestore di pacchetti JavaScript.

Per installare CLR per l'intero sistema:

npm install *-g* selenium-side-runner

Per installare CLR in un singolo progetto:

npm install selenium-side-runner

Installazione dei Driver per Browser

Selenium CLR necessita dei driver per i singoli browser al fine di eseguire i test localmente.

npm install *-g* chromedriver
npm install *-g* edgedriver
npm install *-g* geckodriver (FireFox)
npm install *-g* iedriver

Lanciare un Test da Riga di Comando

I test possono essere controllati mediante il comando **selenium-side-runner**.

```
selenium-side-runner <path>/<file>.side
```

È possibile utilizzare i caratteri speciali per lanciare molti test in parallelo aprendo più finestre del browser:

```
selenium-side-runner <path>/*.side
```

Imporre un Browser

Selenium-side-runner esegue automaticamente i test per tutti i browser. È possibile limitarsi all'uso di un singolo browser mediante l'opzione *-c*.

```
selenium-side-runner -c "browserName=chrome"
selenium-side-runner -c "browserName='internet explorer'"
selenium-side-runner -c "browserName=edge"
selenium-side-runner -c "browserName=firefox"
selenium-side-runner -c "browserName=safari"
```

Salvare l’Esito dei Test

Selenium-side-runner salva gli esiti dei test in appositi file aventi formato **json**. Per imporre la cartella ove salvare i file si usa l’opzione *-o* (*--output-directory*).

```
selenium-side-runner -o <path>/<directory>
```

Es:

```
selenium-side-runner -c "browserName=firefox" -o /tmp/
  ./studiosg_test.side
```

Formato dei Rapporti

Il formato dei rapporti ottenuti dai test può essere impostato con l'opzione *--output-format*. I formati supportati sono:

- jest (JSON);
- junit (XML. Deprecato in Selenium 4).

Es:

```
selenium-side-runner -c "browserName=firefox" -o /tmp/
  ./studiosg_test.side --output-format junit
```

Eseguire i Test in Selenium Grid

Per eseguire i test su macchine remote attraverso **Selenium Grid**, imporre lo URL del server mediante l'opzione *-s* (*--server*).

```
selenium-side-runner \
   --server http://localhost:4444/wd/hub \
   -c "browserName='internet explorer' \
   version='11.0' \
   platform='Windows 8.1'" \
   <test>.side
```

Configurazione Predefinita

È possibile salvare le opzioni della riga di comando in un file di configurazione.

Esistono 2 tipologie di file di configurazione:

- Un file chiamato *.side.yml* nella cartella da cui sono avviati i test. Il file verrà caricato automaticamente dal browser.
- Un file in formato YAML impostato mediante l'opzione *--config-file.* La presenza dell'opzione comporta che il file *.side.yml* venga ignorato.

Esempio di File di Configurazione

```
capabilities:
    browserName: "firefox"
baseUrl: "https://www.studiosg.net"
server:  "http://localhost:4444/wd/hub"
```

Informazioni & Licenze