[Introduction: 2](#_Toc447700190)

[Feature: 2](#_Toc447700191)

[Installation: 3](#_Toc447700192)

[Windows: 3](#_Toc447700193)

[Mac: 3](#_Toc447700194)

[Linux/Ubuntu 4](#_Toc447700195)

[Configuration 4](#_Toc447700196)

[Configuration 4](#_Toc447700197)

[License 4](#_Toc447700198)

[Test Environment 5](#_Toc447700199)

[Pre-requirement 5](#_Toc447700200)

[Process To Run Automated Script 5](#_Toc447700201)

[Automated Test Steps 5](#_Toc447700202)

# Introduction:

Sahi is a mature, business-ready tool for automation of web application testing. It is available as an Open Source free product and as Sahi Pro, the commercial version. For testing teams in product companies and captive IT units which need rapid reliable web automation, Sahi would be the best choice among web automation tools.

Sahi is especially suited for cross-browser/multi-browser testing of complex web 2.0 applications with lots of AJAX and dynamic content. Sahi works well in Agile development environments, enabling rapid automation and maintenance and easily integrating with build systems. Sahi saves time and effort with faster development, less maintenance and fast distributed playback. Sahi runs on any modern browser which supports javascript.

# Feature:

* Works on any browser which supports Javascript and Proxy; this includes almost all desktop and mobile browsers
* Powerful recorder and Object Identifier which works across browsers
* Intuitive and simple APIs
* Javascript based scripts for good programming control
* Version Controllable text-based scripts
* In-built reports
* In-built multithreaded or parallel playback of tests
* Scripts do not need the browser window to be in focus, making scripts very stable
* Command line and ant support for integration into build processes
* Supports external proxy, HTTPS, 401 & NTLM authentications
* Supports browser popups and modal dialogs
* Supports AJAX and highly dynamic web applications
* Scripts very robust
* No wait statements required even for AJAX and page loads
* Intuitive APIs like \_near and \_in eliminates need for XPaths
* Works on applications with random auto-generated ids
* Very lightweight and scalable
* Can directly invoke Java code from scripts. This is used to access databases, read pdf files, connect to another machine via ssh etc.
* Supports data-driven testing. Can connect to database, Excel or CSV file.
* Can call any command line program

# Installation:

## Windows:

To Install SahiPro in Windows following Steps should be done:

1. Confirm Java is configured: Java -version
2. Download Sahi pro
   1. SahiPro runner: (Not required license) <http://sahipro.com/static/builds/pro/install_sahi_pro_runner_v620_20160118.jar>
   2. SahiPro: (License required) <http://sahipro.com/static/builds/pro/install_sahi_pro_v620_20160118.jar>
3. Double click on install\_sahi\_pro\_xxx.jar to start the installer.
4. Install Sahi in a folder path without spaces. (If there are spaces, Chrome will have trouble launching correctly.)

## Mac:

To Install SahiPro in Mac following Steps should be done:

1- Confirm Java is configured: Java -version

2- Extract Sahi into PLACES\<your user name>.

3- Modify the sahi.sh file; manually give the variable class paths:

*#if this does not work, try to run dos2unix on this file, or delete and reintroduce newlines.*

*#EXT\_CLASS\_PATH=;*

*#EXT\_CLASS\_PATH=$EXT\_CLASS\_PATH:../extlib/mysql-connector-java-5.0.4-bin.jar*

*#SAHI\_CLASS\_PATH=/Users/YOUR\_USER\_NAME/sahi/lib/sahi.jar*

*MOZ\_NO\_REMOTE=1*

*java -classpath /Users/YOUR\_USER\_NAME/sahi/lib/sahi.jar:/Users/YOUR\_USER\_NAME/sahi/extlib/rhino/js.jar:/Users/YOUR\_USER\_NAME/sahi/extlib/apc/commons-codec-1.3.jar net.sf.sahi.Proxy*

- do not forget to modify "YOUR\_USER\_NAME" with your Mac user name.

4- Start Firefox Browser. Setup the Proxy address: instead of localhost type 127.0.0.1.

*Manually Proxy Configuration:*

*HTTP Proxy: 127.0.0.1; Port: 9999*

*SSL Proxy: 127.0.0.1; Port: 9999*

*SOCKS v5 - checked*

5- Open Terminal Window. Go to Sahi\bin. Type: ./Dashboard.sh

6- A- Select the browser from dashboard and open the controller by Alt+Double click

6- B- Select the Bin link. It will reopen a new terminal to /userdata/bin folder. Use following command to run the script through command line

./testrunner.sh traderev/choritotext.sah http://rc.devops.traderev.com firefox

## Linux/Ubuntu

To Install SahiPro in Linux run following command in Terminal:

* Open a Terminal
* cd ~ *//To Confirm back to root*
* sudo mkdir SahiPro *//To Create a folder*
* cd SahiPro *//Navigate to new folder*
* sudo wget http://sahipro.com/static/builds/pro/install\_sahi\_pro\_v620\_20160118.jar *// Get current latest Sahipro version*
* java -jar install\_sahi\_pro\_runner\_v620\_20160118.jar *//Run Jar file to install SahiPro*

It will bring up GUI installer; by following the instruction, SahiPro will be installed properly

In some case it is recommending to install with a “silent\_install.xml” file. But this file’s content could be different based on in different system.

# Configuration

## Configuration

To configure and any possible browser setup, please have a look on this page:

<http://sahipro.com/docs/using-sahi/sahi-configuration-basic.html>

## License

The license only requires for that versions which need to script and create Automation script. For the runner version no license is required.

To use your license:

* Download the "license.data" file
* Start Sahi Pro, and when prompted for a license, point to the downloaded license.data file.
* Or alternatively,
* Copy the downloaded "license.data" file into SahiPro/userdata/config folder and then start Sahi Pro.

# Test Environment

## Pre-requirement

To have a success automation test result, pre-data is the most important thing in this process. Most of automated test cases/scripts, will use pre-data which are available in DB. Therefore, to keep them available with no changes, there is required to keep a backup from current DB and restore it every time before running the automation suites.

In addition, to prevent happening any interrupt during running the test, it recommends to have a separate test environment than Dev, Demo, QA, Sales, Support, and so on.

Furthermore, to have the latest application changes by Dev, and the latest automated script, it recommends to schedule to run the script after work and even through the night.

## Process to Run Automated Script

Before running the Automated test script, following steps should be done:

1. Deploy latest changes and re-build test environment
2. Restore DB from backup
3. Pull the latest automation scripts from GitHub
4. Schedule and run automation script
5. Retrieve the results and email to …
6. Log possible bugs which has been found via test running

## Automated Test Steps

To have an acceptable test result, it recommends to do following steps:

1. Run a quick environment verification script - Level 1 (TBD)
   1. Navigate Web
   2. Quick login
   3. Navigate between pages
2. Run a quick data verification script - Level 2 (TBD)
   1. Quick data verification in each page
3. Run entire Automated Scripts based on test cases