Automation User Guide

Version 10.2

WebLOAD Automation User Guide

March, 2014, RadView Publication Number WLPRO-1013-AUG84

For product assistance or information, contact:	
Toll free in the US:	1-888-RadView
Fax:	+1-908-864-8099
World Wide Web:	www.RadView.com

North American Headquarters:	**International Headquarters:**
RadView Software Inc. 991 Highway 22 West, Suite 200 Bridgewater, NJ 08807 Email: info@RadView.com Phone: 908-526-7756 Fax: 908-864-8099 Toll Free: 1-888-RadView	RadView Software Ltd. 14 Hamelacha Street, Park Afek Rosh Haayin, Israel 48091 Email: info@RadView.com Phone: +972-3-915-7060 Fax: +972-3-915-7683

Table of Contents

Chapter 1

Introduction

Welcome to WebLOAD Professional, the premier performance, scalability, reliability testing solution for internet applications.

WebLOAD is easy to use and delivers maximum testing performance and value. WebLOAD verifies the scalability and integrity of internet applications by generating a load composed of Virtual Clients that simulate real-world traffic. Probing Clients let you refine the testing process by acting as a single user that measures the performance of targeted activities, and provides individual performance statistics of the internet application under load.

This section provides a brief introduction to WebLOAD technical support, including both documentation and online support.

WebLOAD Documentation

WebLOAD is supplied with the following documentation:

WebLOAD™ Installation Guide

Instructions for installing WebLOAD and its add-ons.

WebLOAD™ IDE User Guide

Instructions for recording, editing, and debugging load test Agendas to be executed by WebLOAD to test your Web-based applications.

WebLOAD™ Console User Guide

A guide to using WebLOAD console, RadView's load/scalability testing tool to easily and efficiently test your Web-based applications. This guide also includes a quick start section containing instructions for getting started quickly with WebLOAD using the RadView Software test site.

WebLOAD™ Analytics User Guide

Instructions on how to use WebLOAD Analytics to analyze data and create custom, informative reports after running a WebLOAD test session.

WebRM™ User Guide

Instructions for managing testing resources with the WebLOAD Resource Manager.

WebLOAD™ Scripting Guide

Complete information on scripting and editing JavaScript Agendas for use in WebLOAD and WebLOAD IDE.

WebLOAD™ JavaScript Reference Guide

Complete reference information on all JavaScript objects, variables, and functions used in WebLOAD and WebLOAD IDE test Agendas.

WebLOAD™ Extensibility SDK

Instructions on how to develop extensions to tailor WebLOAD to specific working environments.

WebLOAD™ Automation Guide

Instructions for automatically running WebLOAD tests and reports from the command line, or by using the WebLOAD plugin for Jenkins.

The guides are distributed with the WebLOAD software in online help format. The guides are also supplied as Adobe Acrobat files. View and print these files using the Adobe Acrobat Reader. Install the Reader from the Adobe Web site http://www.adobe.com.

Typographical Conventions

Before you start using this guide, it is important to understand the terms, icons, and typographical conventions used in the documentation.

The following icons appear next to the text to identify special information.

Table 1: Icon Conventions

Icon	Type of Information
✎	Indicates a note.
🧩	Indicates a feature that is available only as part of a WebLOAD Add-on.

The following kinds of formatting in the text identify special information.

Table 2: Typographical Conventions

Formatting convention	Type of Information
Special Bold	Items you must select, such as menu options, command buttons, or items in a list.
Emphasis	Use to emphasize the importance of a point or for variable expressions such as parameters.
CAPITALS	Names of keys on the keyboard. for example, SHIFT, CTRL, or ALT.
KEY+KEY	Key combinations for which the user must press and hold down one key and then press another, for example, CTRL+P or ALT+F4.

Where to Get More Information

This section contains information on how to obtain technical support from RadView worldwide, should you encounter any problems.

Online Help

WebLOAD provides a comprehensive online help system with step-by-step instructions for common tasks.

You can press the **F1** key on any open window for an explanation of the options or select **Help** ➤ **Contents** to open the online help contents and index.

Technical Support Website

The technical support pages on our website contain:

- FAQ (Frequently Asked / Answered Questions)
- Agenda Center
- Documentation
- RadView's Product Resource Center, where you can find prepared test scripts, product information, and industry related news.
- http://radview.com/support/index.asp

Technical Support

For technical support in your use of this product, contact:

North American Headquarters	International Headquarters
e-mail: support@RadView.com Phone: 1-888-RadView (1-888-723-8439) (Toll Free) 908-526-7756 Fax: 908-864-8099	e-mail: support@RadView.com Phone: +972-3-915-7060 Fax: +972-3-915-7683

Note: We encourage you to use e-mail for faster and better service.

When contacting technical support please include in your message the full name of the product, as well as the version and build number.

Chapter 2

WebLOAD Automation Overview

This guide describes the two methods of automating WebLOAD tasks:

- Running WebLOAD components from a command line interface.
- Running a WebLOAD job in Jenkins.

Chapter 3

Running WebLOAD from the Command Line

You can run the following WebLOAD components from the command line:

- WebLOAD Console – You can launch a WebLOAD test that invokes the WebLOAD Console and runs a specified test according to the specified parameters.
- WebLOAD Analytics – You can launch WebLOAD Analytics and generate a report for a specified session, then publish or print it.
- WebLOAD IDE – You can launch an IDE test that invokes the WebLOAD IDE and runs a specified test according to the specified parameters.

Running a WebLOAD Console Test through the CLI

You can perform load testing through a command line interface. You can enter the WebLOAD Console launch command into a batch file or into an external script and WebLOAD Console will run directly, without user intervention, using the parameters specified.

To run WebLOAD Console through the command line interface:

Enter the `WebLOAD.exe` command together with a series of optional parameters (described below) into your external script to automatically launch a WebLOAD test. When your script runs, the executable file will invoke WebLOAD Console and run the specified test according to the specified parameters.

Note: Verify that the Agenda used with the template specified, and any included files, are accessible to the Load Template or Load Session file that will be run.

Syntax

Use the following syntax to define the parameters for running a WebLOAD test through a command line interface:

```
WebLOAD.exe [<file name to open>][<file name to save>]
[<flags>][<time>][/ag <Agenda name>]
```

Parameters

Parameter	Description	Comments
File name to open	The name of the `*.tpl` or `*.ls` file (Load Template or Load Session file) to run.	Optional parameter
File name to save	The name of the `*.ls` file containing the test data. This file will be saved in the current directory unless otherwise specified.	Optional parameter
Flags	• /ar – Automatically run the WebLOAD test without waiting for user input. If this flag is not specified, the Console is called up with the specified Load Template/Load Session but the test is not automatically run. The system waits for user input. • /ar *<time>* – Automatically run the test for the length of time specified in *<time>*/ • /vc – The number of Virtual Client licenses to allocate when using WebRM License Server. • /pc – The number of Probing Client licenses to allocate when using WebRM License Server. • /rc *<results_file_name>* – Place the results in the specified file (an XML file).	Optional parameter
~~Time~~	~~The length of time (in seconds) that the test is to run.~~	~~Optional parameter~~
/ag <Agenda name>	The name of an existing Agenda (*.wlp) to open.	Optional parameter

The parameters are all optional. If no parameters are entered, the executable launches the Console and does not run a test. If the autorun flag `</ar>` flag is not set, the `<file name to save>` and the `<time>` parameters are ignored.

Note: If there is a conflict between the time defined in this command and the time defined in the WebLOAD Scheduler, the load test runs for the shorter of the two periods defined.

Examples

Example 1:

```
WebLOAD.exe test1.tpl
```

This command opens the Console and the `test1.tpl` template. The Console waits for user input.

Example 2:

```
WebLOAD.exe test1.tpl march9.ls /ar 30
```

This command opens the Console and automatically runs a test using the `test1.tpl` template file. The test results are saved in the Load Session file `march9.ls`, which includes all of the test data and results. This file is saved in the current directory, unless otherwise specified. The autorun flag is set, meaning that the test runs without user intervention. The test will run for 30 seconds.

Example 3:

```
WebLOAD.exe /ag c:\agendas\MyAgenda.wlp
```

This command opens the Console and the WebLOAD Wizard to the Agenda/Mix Selection dialog box. The `MyAgenda.wlp` Agenda is automatically selected and the WebLOAD Wizard waits for user input.

Example 4:

```
WebLOAD.exe test1.tpl march9.ls /ar 30 /rc result1.xml
```

This command performs all the actions described in *Example 2* above, and in addition the execution return code is saved in `result1.xml`.

Example 5:

```
WebLOAD.exe test1.tpl march9.ls /ar 30 /vc 100 /pc 3
```

This command performs all the actions described in *Example 2* above, and in addition it allocates 100 virtual clients and 3 probing clients from the WebRM server.

Generating an Analytics Report through the CLI

WebLOAD Analytics can be executed in command line mode. This enables incorporating WebLOAD Analytics in scripts. Two executables are available:

- **`WLAnalyticsCMD.exe`** – Automatically generates a report for a specified session, and publishes or prints it. WebLOAD Analytics then closes.
- **`WLAnalytics.exe`** – Launches the WebLOAD Analytics UI, and generates a report for a specified session.

The executables are located in `<Installation dir>\bin`. For example: `C:\Program Files\RadView\WebLOAD\bin`.

Running WLAnalyticsCMD.exe

Use this executable to generate a report for a specified session, and publish or print it.

Syntax

```
WLAnalyticsCmd.exe -m U|P {-t template_path}|{-p portfolio_path}
{-s session_name}|{-ls session_path} [-f
DOC|ODT|HTML|XLS|RTF|PDF] [-l report_location] [-n
output_report_name] [-h]
```

Parameters

Parameter	Description	Comments
`-m`	Indicates the action. Specify one of the following **U** – Publish. **P** – Print.	Mandatory parameter.
`-t` *`template_path`*	Generates a chart from a specified template. You must specify the path to the template directory (either absolute or relative to the gallery).	You must specify one of the two options: **-t** or **–p**.
`-p` *`portfolio_path`*	Generates a report from a portfolio. You must specify the path to the portfolio directory (either absolute or relative to the Portfolio category).	
`-s` *`session_name`*	Specifies a session already loaded into WebLOAD. You must specify the session name. **Note**: You can use this parameter multiple times to specify multiple sessions. This is necessary if you are generating a regression chart.	You must specify one of the two options: **-s** or **–ls**.

Parameter	Description	Comments
-ls *session_path*	Specifies a load session file to import into WebLOAD. You must specify the full path. **Note**: You can use this parameter multiple times to load multiple sessions. This is necessary if you are generating a regression chart.	
-f	Specifies the output format for a published report. Select one of the following: **DOC**, **ODT**, **HTML**, **XLS**, **RTF**, or **PDF**. If you do not specify an output format, the default format, specified in Analytics Preferences, is used.	Optional parameter.
-l *report_location*	Specifies the location of the published report. If you do not specify a location, the default location, specified in Analytics Preferences, is used.	Optional parameter.
-n *output_report_name*	Specifies a name for the newly created report. If you do not specify a name, the application provides a default name.	Optional parameter.
-h	Displays the help.	Optional parameter.

Note: Note that you must specify:
– Publish or print.
– A template or portfolio.
– A session, either previously loaded or to be imported.

Examples:

Example 1:

To load the `mysession.ls` Load Session, generate a ‘General/Load Size Summary’ chart, and publish it in the default file format, in the default location, under the name `test-report`:

```
WLAnalyticsCmd.exe -m U -t "General\Load Size Summary" -ls
"C:\mysession.ls" -n "test-report"
```

Example 2:

To use the loaded `first-session` Load Session, generate a ‘Summary Portfolio’ portfolio, and print it:

```
WLAnalyticsCmd.exe -m P -p "Summary Portfolio" -s “first-
session"
```

Example 3:

To use the loaded `first-session` and `second-session` Load Sessions, generate a ‘Regression/Load Size Summary’ regression chart, and publish it as a PDF file in `C:\myreports`, using a default name:

```
WLAnalyticsCmd.exe -m U -t "Regression\Load Size Summary" -s
‟first-session" -s "second-session" –f PDF –l ‟c:\myreports”
```

Running WLAnalytics.exe

Use this executable to open the WebLOAD Analytics UI, and open a report or generate a report for a specified session.

Syntax

```
WLAnalytics.exe {-t template_path}|{- p portfolio_path}
{-s session_name}|{-ls session_path} [-h] [-noSplash]
```

Parameters

Parameter	Description	Comments
-t *template_path*	Generates a chart from a specified template. You must specify the path to the template directory (either absolute or relative to the gallery).	You must specify one of the two options: **-t** or **–p**.
-p *portfolio_path*	Generates a report from a portfolio. You must specify the path to the portfolio directory (either absolute or relative to the Portfolio category).	
-s *session_name*	Specifies a session already loaded into WebLOAD. You must specify the session name. **Note**: You can use this parameter multiple times to specify multiple sessions. This is necessary if you are generating a regression chart.	You must specify one of the two options: **-s** or **–ls**.
-ls *session_path*	Specifies a load session file to import into WebLOAD. You must specify the full path. **Note**: You can use this parameter multiple times to load multiple sessions. This is necessary if you are generating a regression chart.	
-h	Displays the help.	Optional parameter.
-noSplash	Launches without a Splash screen.	Optional parameter.

Note: Note that you must specify:
– A template, report, or portfolio.
– A session, either previously loaded or to be imported.

Examples

Example 1:

To open the WebLOAD Analytics UI, load the `mysession.ls` Load Session, and generate a ‘General/Load Size Summary’ chart:

```
WLAnalytics.exe -t "General\Load Size Summary" -ls
"C:\mysession.ls"
```

Example 2:

To open the WebLOAD Analytics UI, use the loaded `first-session` Load Session, and generate a ‘Summary Portfolio’ portfolio:

```
WLAnalytics.exe -p "Summary Portfolio" -s “first-session"
```

Running WebLOAD IDE Testing through the CLI

You can initiate WebLOAD IDE testing directly through the CLI. You can enter the WebLOAD IDE launch command into a batch file or into an external script and WebLOAD IDE will run directly, without user intervention, using the parameters specified.

To run WebLOAD IDE testing through the CLI:

Enter the `webloadIDE.exe` command together with a series of optional parameters (described below) into your external script to automatically launch a WebLOAD IDE test. When your script runs, the executable file will invoke WebLOAD IDE and run the specified test according to the specified parameters.

Syntax

Use the following syntax to define the parameters for running a WebLOAD IDE test **through a Command Line Interface.:**

```
webloadide.exe [<flags>][<project or session name to open>]
[<session name to save to>][<Number of rounds to run>]
```

To run more than one session, append all relevant parameters at the end of the syntax. See examples 2 and 3 in *Examples* (on page 14).

Parameters

When running a test invoked by the executable, you can specify the following parameters:

Parameter	Description	Comments
Flags	/a - auto run Automatically run the WebLOAD IDE test without waiting for user input. If this flag is not specified, WebLOAD IDE is opened with the specified project / session but the test is not automatically run. The system waits for user input.	Optional parameter
Project or session name to open	The name of the `.wlp` file or `.wls` file (Project file or Session file) to run.	Optional parameter
Session name to save to	The name of the `.wls` file containing the test data. This file will be saved in the current directory unless otherwise specified.	Optional parameter
Number of rounds to run	The number of iterations to run during runtime. The default value is 1.	Optional parameter

The parameters are all optional. If no parameters are entered, the executable launches WebLOAD IDE and does not run a test. If the autorun flag </a> flag is not set, the < Session name to save to >, and the < Number of rounds to run > parameters are ignored.

Examples

Example 1:

```
webloadide.exe test1.wlp
```

This command opens WebLOAD IDE with the `test1` project file and waits for user input.

Example 2:

```
webloadide.exe /a test1.wlp test2.wlp 3
```

This command:

- Opens WebLOAD IDE and automatically runs a test using the `test1.wlp` project file.
- Runs the project for three iterations.
- Saves the test results in the WebLOAD IDE session file `test1.wls`, which includes all of the test data and results.

Example 3:

```
webloadide.exe /a test1.wlp test1.wls 3 /a test2.wlp test2.wls 2
```

This command:

- Opens WebLOAD IDE and automatically runs a test using the `test1.wlp` project file.
- Runs the project test1.wlp for three iterations.
- Saves the test results in the WebLOAD IDE session file `test1.wls`, which includes all of the test data and results.
- Opens the WebLOAD IDE project file `test2.wlp`.
- Runs the project test2.wlp for two iterations.
- Saves the test results in the WebLOAD IDE session file `test2.wls`, which includes all of the test data and results.

Chapter 4

Running WebLOAD in Jenkins

Jenkins CI is a Continuous Integration server that is growing in popularity. Agile development teams use Jenkins to run their builds, tests and deployments as well as their performance tests if possible. Jenkins is extendable by means of plugins. As of WebLOAD 10.1, a WebLOAD plugin is available for Jenkins.

To run WebLOAD jobs in Jenkins, you must:

1. Install a Jenkins CI server.
   Jenkins needs to be able to control a full WebLOAD installation (in which both the WebOLAD Console and WebOLAD Analytics are installed, and which has an active license). This can be done either by installing the Jenkins Master on the WebLOAD machine, or by setting up a Jenkins Slave (recommended).
   For instructions, refer to the Jenkins documentation at *http://jenkins-ci.org/*.

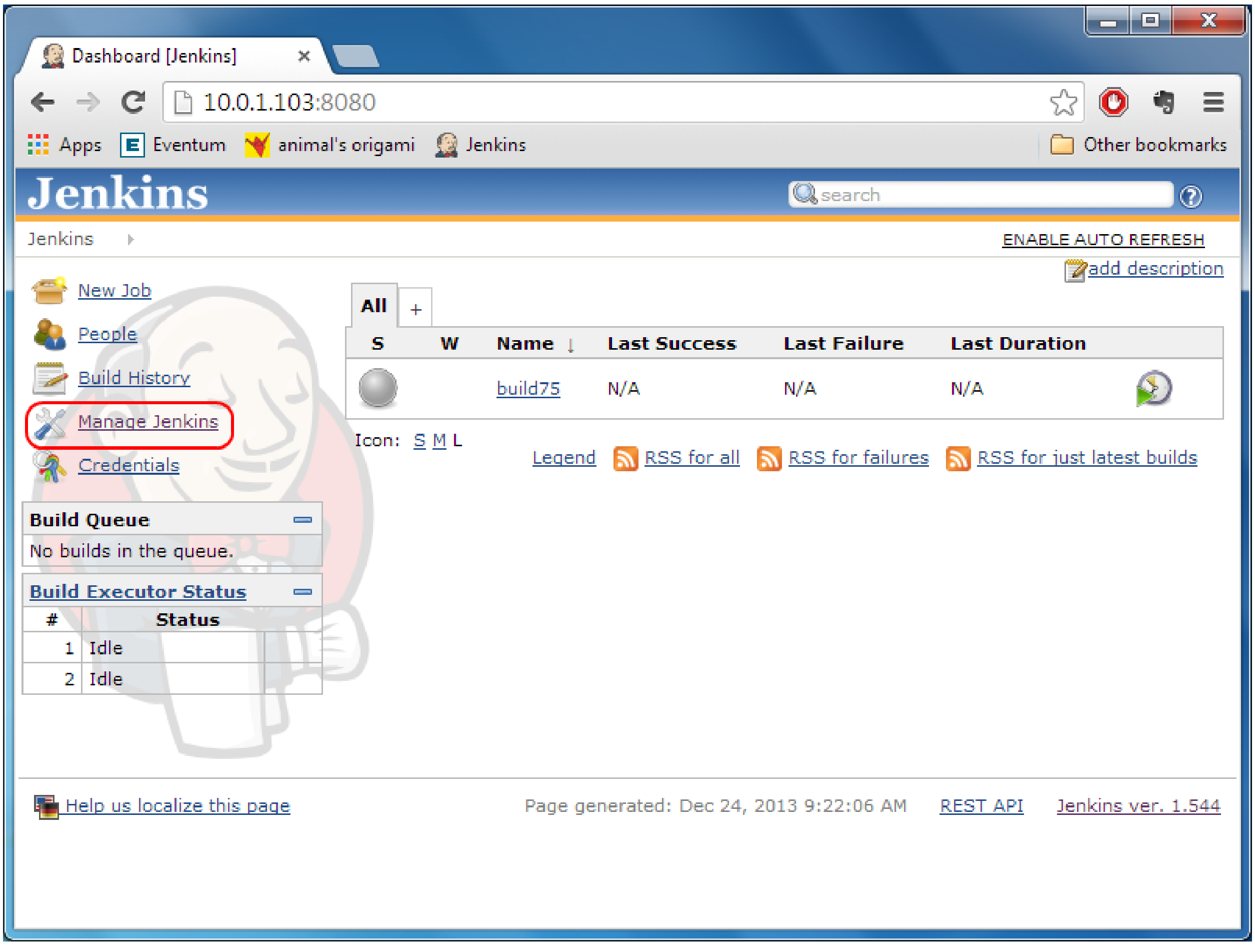
   Note: If Jenkins is running as a Service, make sure that you are using a logged in user, and not the Local System Account.

2. Install the WebLOAD plugin in Jenkins. This is a one-time task. Refer to *Installing the WebLOAD Plugin for Jenkins* below.
3. Define a WebLOAD job in Jenkins whenever desired. Refer to *Creating a WebLOAD Job in Jenkins* on page 19.

Installing the WebLOAD Plugin for Jenkins

To install the WebLOAD Plugin for Jenkins:

1. In the Jenkins dashboard (homepage), click **Manage Jenkins**.



S	W	Name ↓	Last Success	Last Failure	Last Duration
●		build75	N/A	N/A	N/A

Build Queue

No builds in the queue.

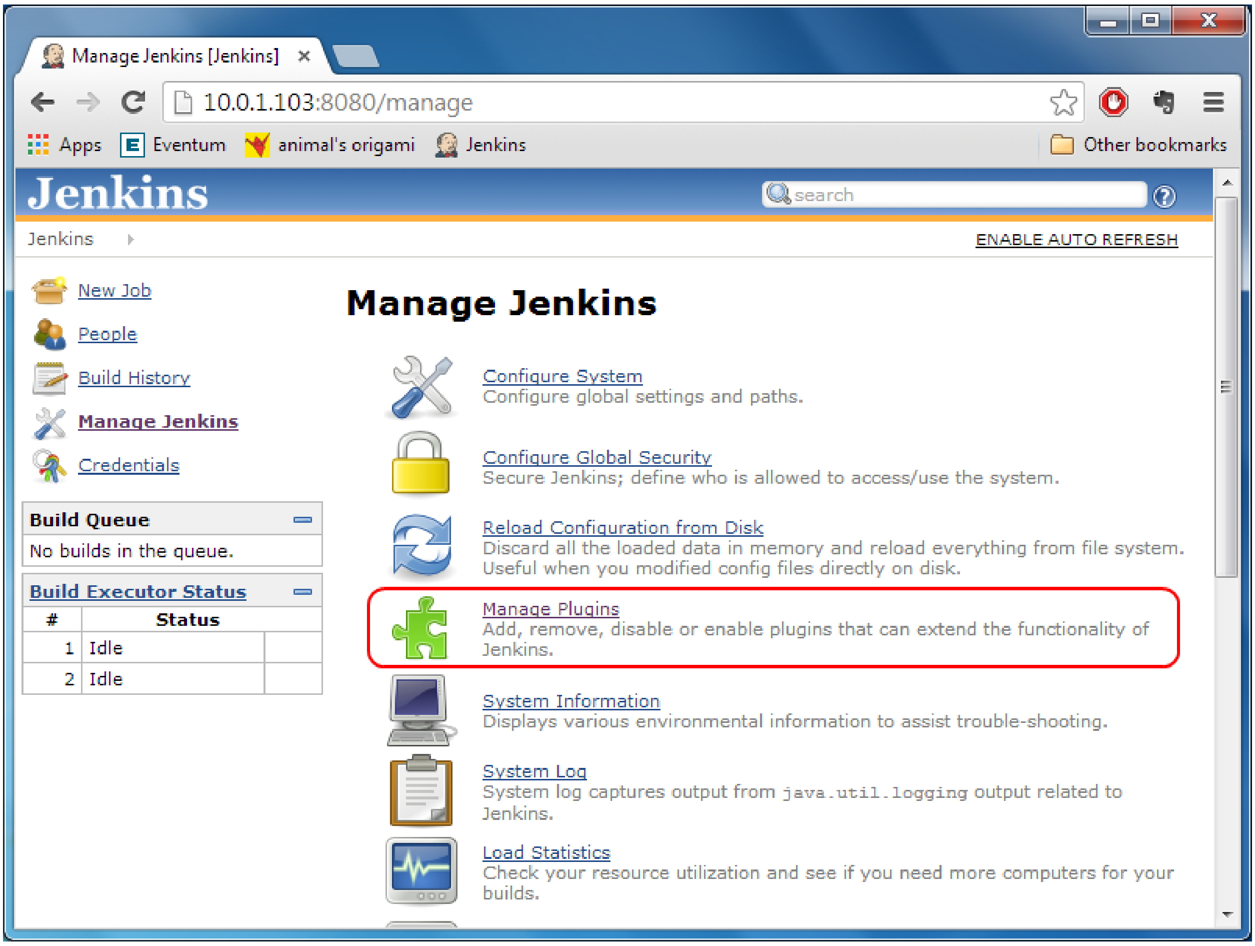
Build Executor Status

#	Status
1	Idle
2	Idle

Page generated: Dec 24, 2013 9:22:06 AM

Figure 1: Selecting Manage Jenkins

2. In the Manage Jenkins page, click **Manage Plugins**.



Build Queue

No builds in the queue.

Build Executor Status

#	Status
1	Idle
2	Idle

Manage Jenkins

Configure System
Configure global settings and paths.

Configure Global Security
Secure Jenkins; define who is allowed to access/use the system.

Reload Configuration from Disk
Discard all the loaded data in memory and reload everything from file system. Useful when you modified config files directly on disk.

Manage Plugins
Add, remove, disable or enable plugins that can extend the functionality of Jenkins.

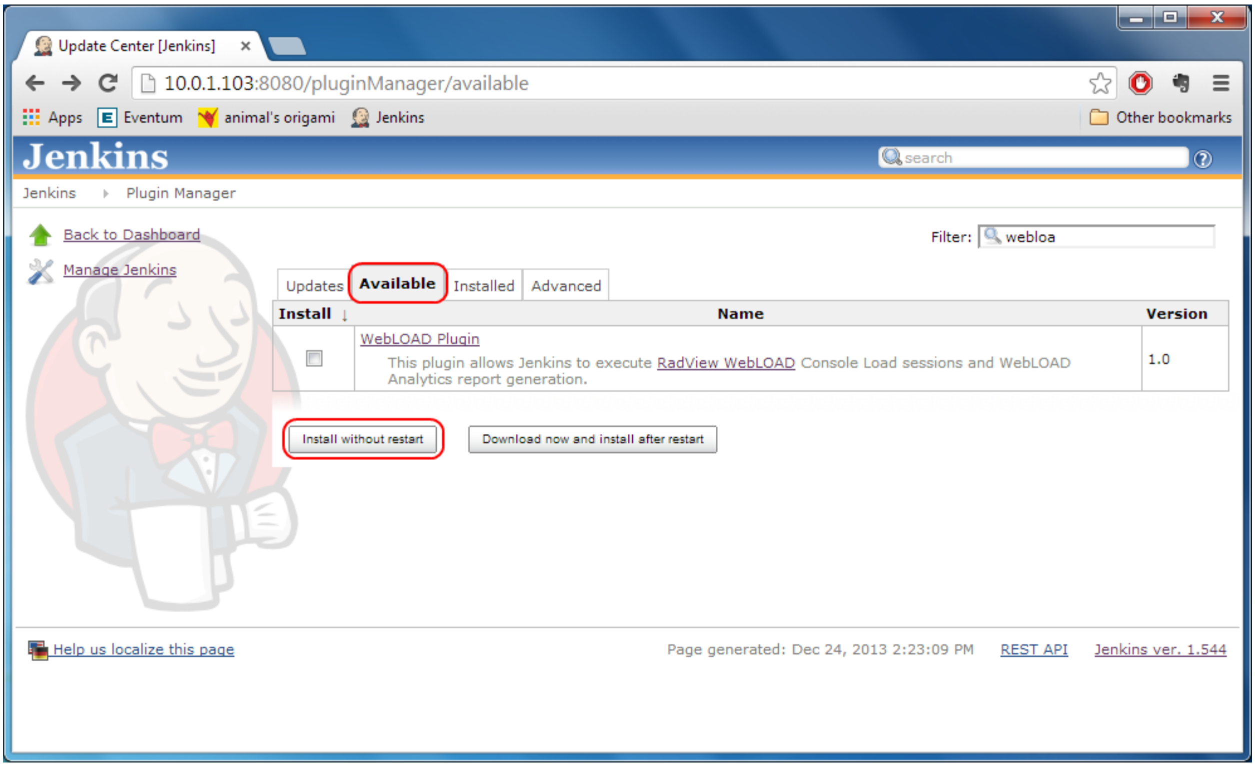
System Information
Displays various environmental information to assist trouble-shooting.

System Log
System log captures output from `java.util.logging` output related to Jenkins.

Load Statistics
Check your resource utilization and see if you need more computers for your builds.

Figure 2: Selecting Manage Plugins

3. Select the **Available** tab.



Install ↓	Name	Version
□	WebLOAD Plugin This plugin allows Jenkins to execute RadView WebLOAD Console Load sessions and WebLOAD Analytics report generation.	1.0

Figure 3: Available Plugins

4. In the Available page:

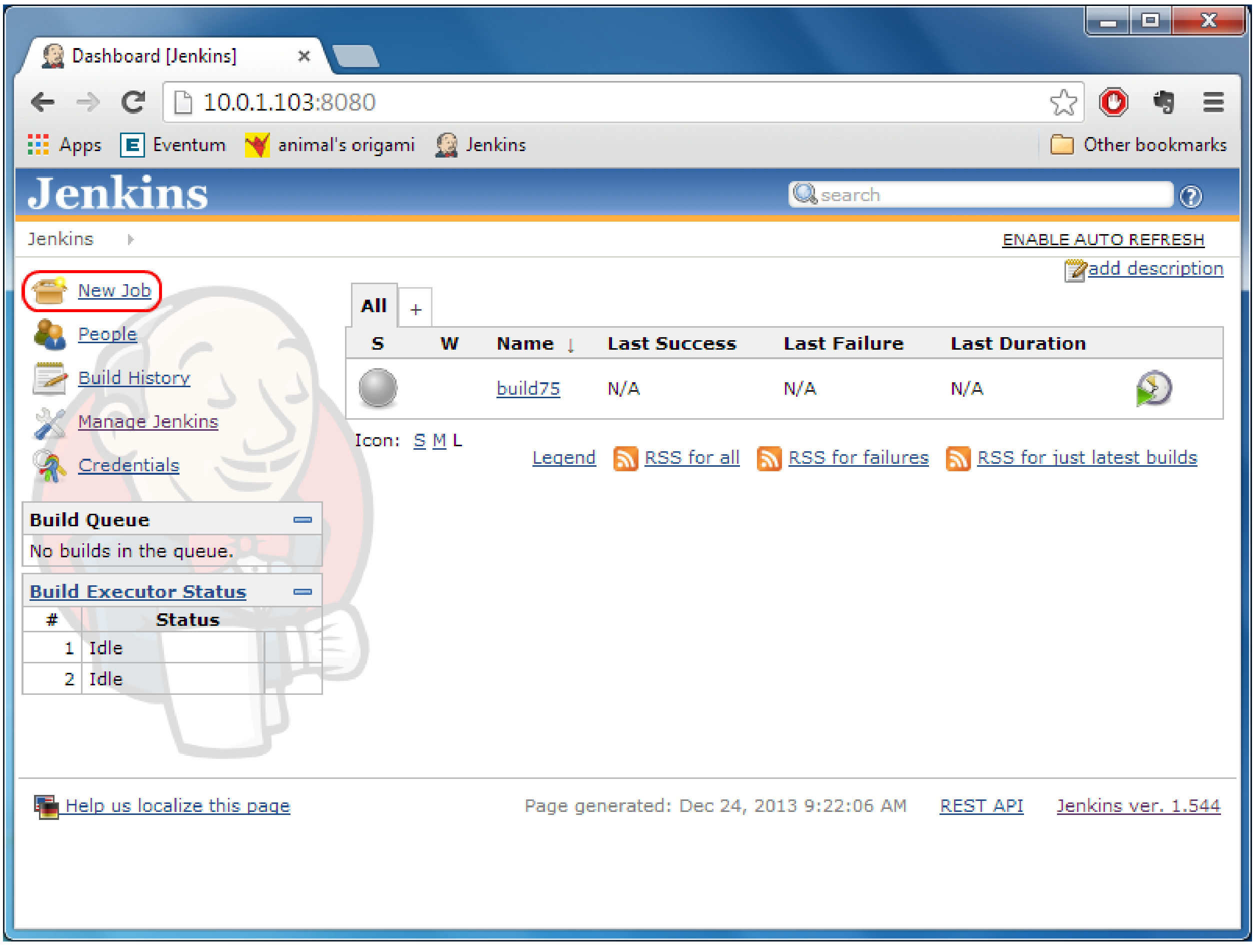
   a. Select the **WebLOAD Plugin** in the list.

   b. Click **Install without restart**.

You are returned to the Jenkins dashboard.

Creating a WebLOAD Job in Jenkins

1. In the Jenkins dashboard (homepage), click **New Job**.



S	W	Name ↓	Last Success	Last Failure	Last Duration
●		build75	N/A	N/A	N/A

Build Queue

No builds in the queue.

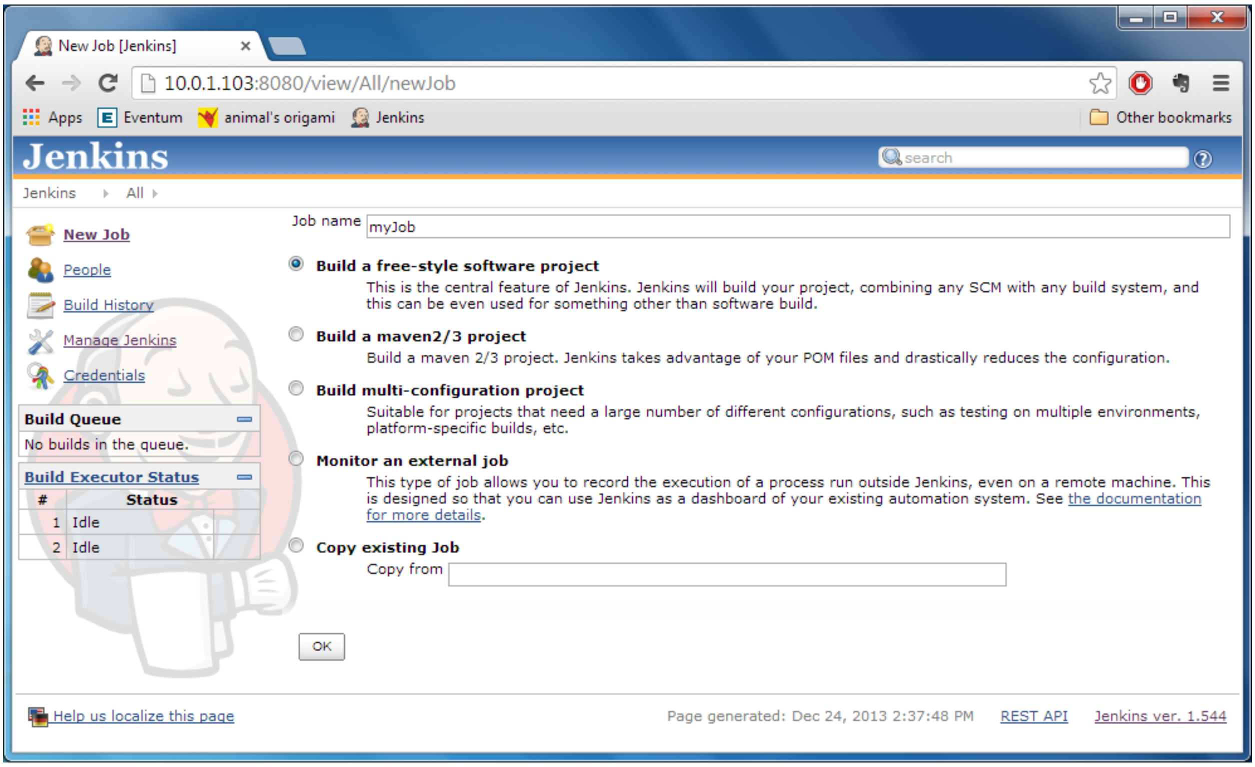
Build Executor Status

#	Status
1	Idle
2	Idle

Page generated: Dec 24, 2013 9:22:06 AM

Figure 4: Selecting New Job

2. Enter a name for the new job, and define the type of job by selecting from the options displayed under the job name. Typically you might select **Build a free-style software project** or **Copy existing Job**.



Build Queue

No builds in the queue.

Build Executor Status

#	Status
1	Idle
2	Idle

Job name: myJob

● **Build a free-style software project**

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

○ **Build a maven2/3 project**

Build a maven 2/3 project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

○ **Build multi-configuration project**

Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

○ **Monitor an external job**

This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed so that you can use Jenkins as a dashboard of your existing automation system. See the documentation for more details.

○ **Copy existing Job**

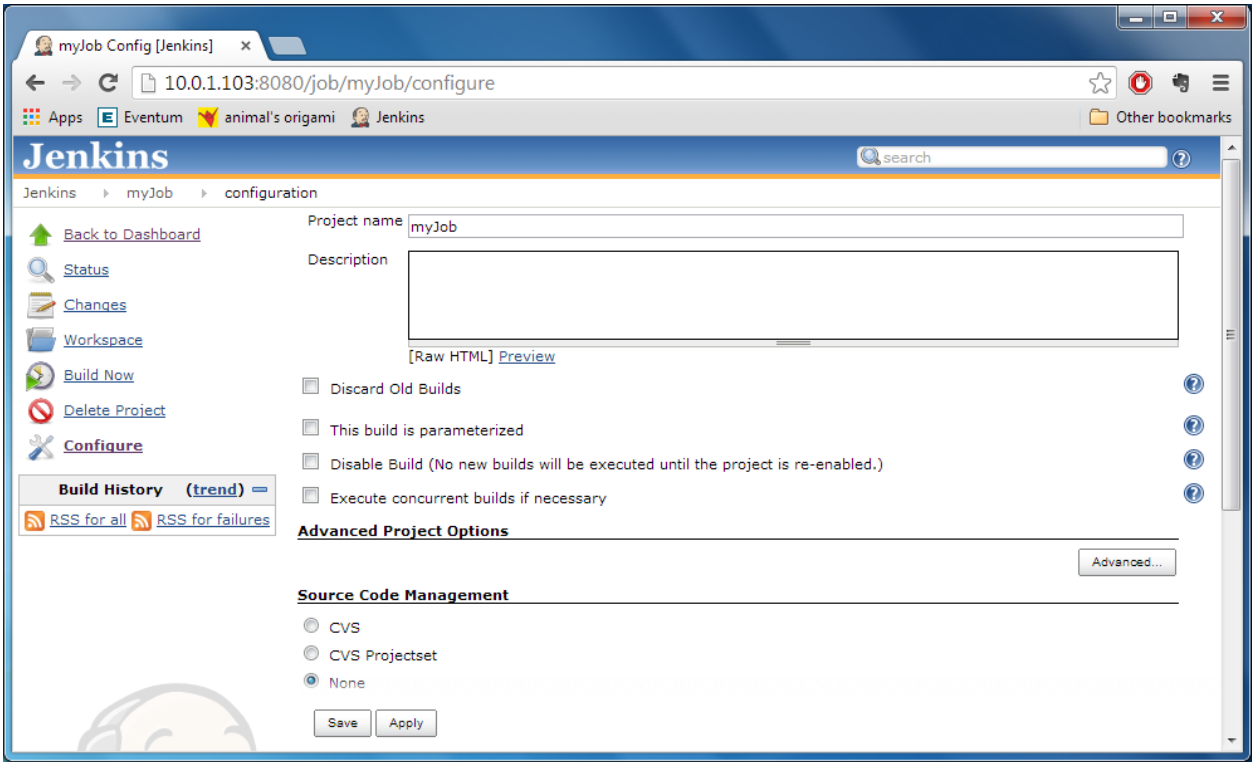
Copy from

Page generated: Dec 24, 2013 2:37:48 PM REST API Jenkins ver. 1.544

Figure 5: Defining a Job Name and Type

3. Click **OK**.

   A standard Jenkins New Job page appears.



Project name: myJob

Description

[Raw HTML] Preview

□ Discard Old Builds

□ This build is parameterized

□ Disable Build (No new builds will be executed until the project is re-enabled.)

□ Execute concurrent builds if necessary

Advanced Project Options

Source Code Management

○ CVS

○ CVS Projectset

● None

Figure 6: New Job Page

4. Define the project (for help, refer to Jenkins documentation). For example, you might click **Build periodically** to define a project that runs automatically every Thursday at 2 AM.

5. In the **Build** section, click **Add build step** and select a step. The two steps offered by the WebLOAD plugin are **Execute WebLOAD load session** and **Generate WebLOAD Analytics Report**.

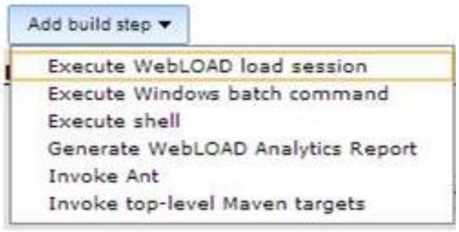

Add build step
Execute WebLOAD load session
Execute Windows batch command
Execute shell
Generate WebLOAD Analytics Report
Invoke Ant
Invoke top-level Maven targets

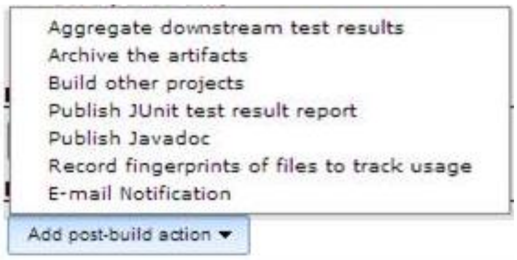
Figure 7: Plugins List

The screen refreshes to enable defining the settings of the build step. Refer to:

- *Defining the Execute Load Session Build Step* on page 22
- *Defining the Generate Analytics Report Build Step* on page 24

6. Repeat the previous step as many times as desired to add as many build actions as desired.

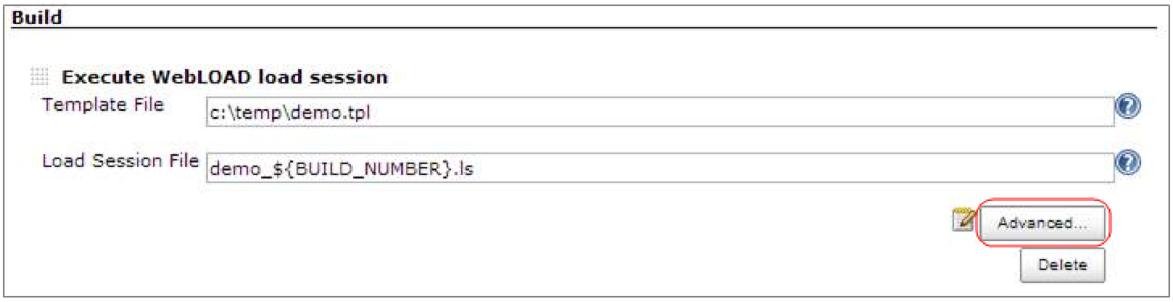
7. Click **Add post-build action** and specify a post-build action or actions. For example, if you specified the JUNIT output format for an Analytics report, you can select **Publish JUnit test result report.**


Aggregate downstream test results
Archive the artifacts
Build other projects
Publish JUnit test result report
Publish Javadoc
Record fingerprints of files to track usage
E-mail Notification
Add post-build action

8. Click **Save** or **Apply**.

Defining the Execute Load Session Build Step

If you select **Execute WebLOAD load session** as a build step, the following appears:



Build

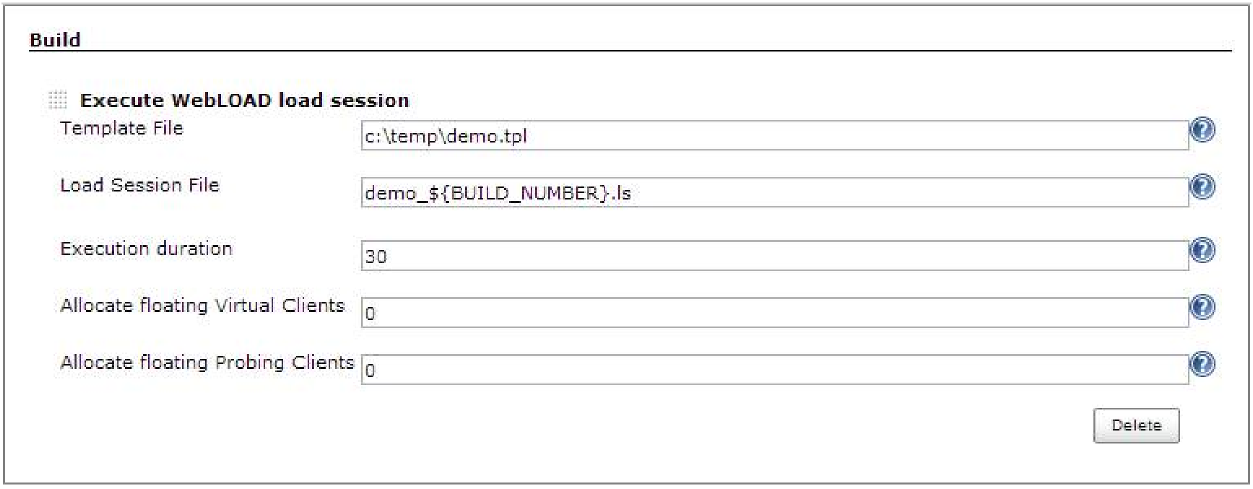
Execute WebLOAD load session

Template File: c:\temp\demo.tpl

Load Session File: demo_${BUILD_NUMBER}.ls

*Figure 8: **Execute WebLOAD load session – Defining***

1. Optionally click **Advanced** to further refine your definitions. The following appears:



Build

Execute WebLOAD load session

Template File: c:\temp\demo.tpl

Load Session File: demo_${BUILD_NUMBER}.ls

Execution duration: 30

Allocate floating Virtual Clients: 0

Allocate floating Probing Clients: 0

Figure 9: Execute WebLOAD load session – Advanced Definition

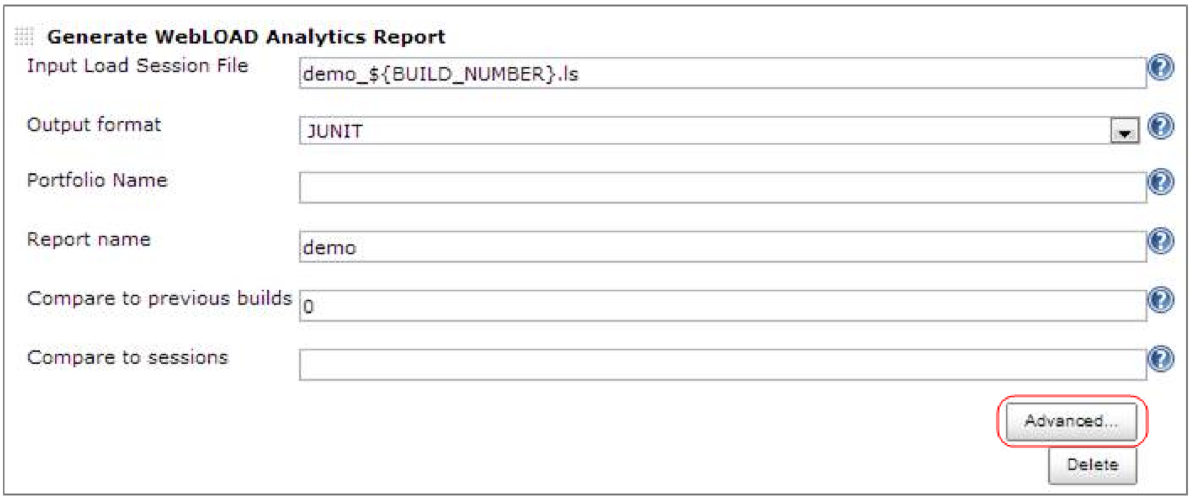
2. Enter information in the fields, as described in Table 3.

Table 3: WebLOAD Load Session Parameters

Parameter	Description
Template File	The name and path of the `*.tpl` (Template file) or `*.ls` file (Load Session file) to run. The template defines which agenda to run, on which load generator, and for how long.
Load Session File	The name of the Load Session results file (*.ls) that will store the load session results. If you do not enter a name, the template file name will be used (and the .tpl suffix will be replaced with .ls) This file will be saved in the workspace unless you enter a different path.
Execution duration	The length of time (in seconds) that the test is to run. This setting overrides the template definitions.
Allocate floating Virtual Clients	The number of Virtual Client licenses to allocate when using WebRM License Server.
Allocate floating Probing Clients	The number of Probing Client licenses to allocate when using WebRM License Server.

Defining the Generate Analytics Report Build Step

If you select **Generate WebLOAD Analytics Report** as a build step, the following appears:

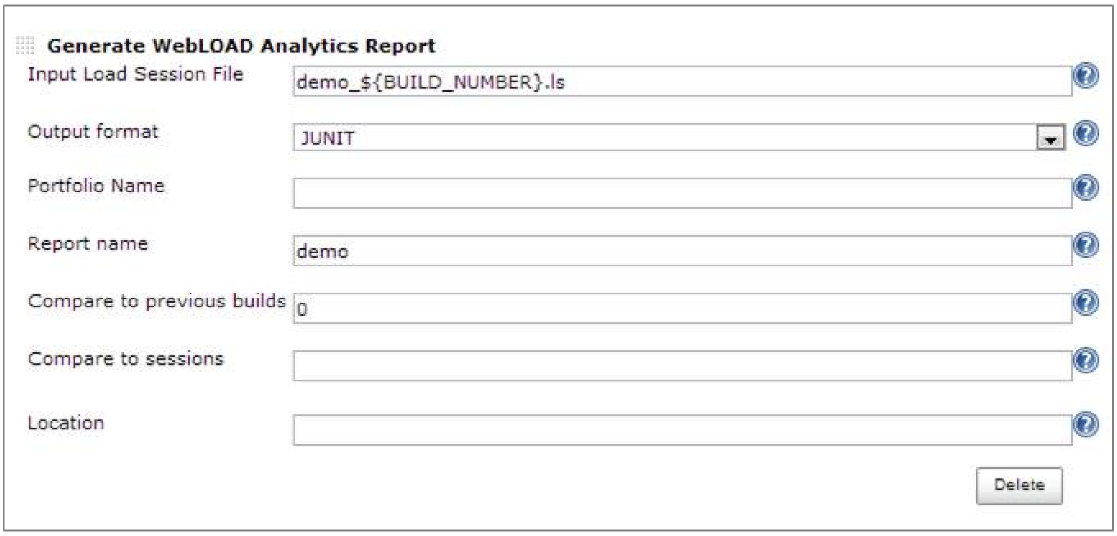


Generate WebLOAD Analytics Report

Input Load Session File	demo_${BUILD_NUMBER}.ls
Output format	JUNIT
Portfolio Name	
Report name	demo
Compare to previous builds	0
Compare to sessions	

Figure 10: Generate WebLOAD Analytics Report – Defining

1. Optionally click **Advanced** to further refine your definitions. The following appears:



Generate WebLOAD Analytics Report

Input Load Session File	demo_${BUILD_NUMBER}.ls
Output format	JUNIT
Portfolio Name	
Report name	demo
Compare to previous builds	0
Compare to sessions	
Location	

Figure 11: Generate WebLOAD Analytics Report – Advanced Definition

2. Enter information in the fields, as described in Table 4.

Table 4: WebLOAD Analytics Report Parameters

Parameter	**Description**
Input Load Session File	The Load Session file name (*.ls) for which to generate a report. The Load Session file was created by executing a WebLOAD Console load session.
Output format	The output format of the report. Select one of the following: • **JUNIT** - JUnit XML Report. Can be used in conjunction with the Post Build step **Publish JUnit test result report**. Available from WebLOAD 10.1. • **HTML** - HTML Report • **DOC** - Microsoft Word document • **ODT** - OpenDocument format • **XLS** - Microsoft Excel spreadsheet • **XLSX** - Microsoft Excel spreadsheet (2007 and newer) • **RTF** - Rich Text Format • **PDF** - Portable Document Format • **CSV** - Comma Separated Values (text) • **RAW** - Export the report raw data in CSV (text) format
Portfolio Name	The templates portfolio to use in order to generate the report. You can specify any portfolio created in the WebLOAD Analytics tool. Note that the predefined portfolios are: • **Summary Portfolio** – used by default • **Regression Portfolio** – use when comparing several sessions. Available from WebLOAD 10.1 • **Session Comparison Portfolio** – use when comparing to a single baseline session • **Extended Summary Portfolio**
Report name	The name of the generated report. The name is used both as the report title and as the output filename. If you do not specify a name, the application provides a default name.
Compare to previous builds	Compare to the session results of the defined number of previous builds. For example, if you enter 8, the report will compare the current build with the past eight builds. The previous result files (.ls) are expected to have the same name, in the previous build artifacts.
Compare to sessions	Compare to these specified Load Session files. The sessions should be entered as a comma separated list of *.ls files. Use absolute paths, or paths that are relative to the workspace.
Location	The location of the generated report, relative to the workspace.

Common Usage Scenarios of an Analytics Build Action

Generate a summary report of the last build

Used to present the results of the recently-run load session.

- Input Load Session File: The Load Session from the Execute Console command.
- Output format: PDF, DOC or HTML
- Portfolio Name: Summary Portfolio

Compare last build to X number of previous builds

Used for regression testing against several previous runs of the Jenkins job (builds).

- Input Load Session File: The Load Session from the Execute Console command.
- Output format: PDF, DOC or HTML
- Portfolio Name: Regression Portfolio
- Compare to previous builds: Number of previous builds to compare with, for example 8.

Compare last build to a known session

Used to compare the results to a base / benchmark load session.

- Input Load Session File: The Load Session from the Execute Console command.
- Output format: PDF, DOC or HTML
- Portfolio Name: Session Comparison Portfolio
- Compare to sessions: Path to the base session file, for example C:\base_results.ls

Generate a report in JUnit format for the last build

Used to export the validation results in JUnit format.

- Input Load Session File: The Load Session from the Execute Console command.
- Report Name: Enter a name for the report. For example, entering ‘result’ will create a ‘result.xml’ file.
- Output format: JUNIT
- Portfolio Name: Summary Portfolio

Note: Only templates with Validation Rules are exported.

Note: You can optionally instruct Jenkins to display the generated report. To do so, click **Add post-build action**, select **Publish JUnit test result report** and specify the report file name (result.xml in our example).