Micro Focus Application Automation Tools

This plugin integrates Micro Focus products with Jenkins. Using the plugin, you can create and use virtual services, run Performance Center or LoadRunner performance tests, run UFT functional tests, run tests in your lab, and run tests on mobile devices. This plugin also lets you upload test results to ALM. In addition, ALM Octane users can track and trigger Jenkins pipelines from within the user interface.

Jenkins versions

Beginning version 5.5, this plugin will only support the latest five LTS versions of Jenkins (currently 2.107.3). This is a result of the Jenkins policy to no longer support older update centers. Jenkins now requires you to have Java 8 installed on the machine. To recover any information lost during an upgrade relating to package names, backup the Jenkins folder and then run the Rebranding script.

Submit issues and feedback through JIRA

This plugin is open source. If you identify any issues or want to submit enhancement requests, please use JIRA. The component for this plugin is the hp-application-automation-tools-plugin component. Your feedback and involvement will allow us to stabilize and enhance the capabilities of the plugin. The latest, early access, beta versions of this plugin are available here.

A list of known bugs is available here.

- What's New in version 5.9
- Changelog
- Supported Integrations
- Prerequisites
- User Guide
  - Create an execution node
  - Support for Pipelines
  - Enable Non-English Languages
  - Tips and Troubleshooting

Content Security Policy

Starting with version 1.641 (or 1.625.3), Jenkins introduced the **Content-Security-Policy** header. This causes some of the integration links, such as links to reports, to become inoperable. For details, see Configuring Content Security Policy and Jenkins Security Advisory . For suggested workarounds until the issue is resolved, see Content Security Policy Header.

What's New in version 5.9

See the Changelog for information about beta versions and recent patches.

Version 5.9 provides the following enhancements:

ALM Octane

- Support for custom converters to test frameworks
- Improvements in logs
- Support for additional versions of the Sonarqube scanner plugin (2.6.1, 2.8.1, 2.9)
- Bug fixes

Mobile Center

- Rebrand from "Mobile Center" to "UFT Mobile"

ALM

- Improvements in the logic of parsing test case status.
  https://issues.jenkins-ci.org/browse/JENKINS-58134

UFT

- Support for ALM 15.0 with SSO
- Refactoring of HpToolsLauncher
- Bug fixes

Changelog

The Changelog page lists the changes in the versions of the plugin.

- Version 5.6.2
- Version 5.6.1
- Version 5.5.2
- Version 5.5.1

Supported Integrations

This plugin supports the following Micro Focus product versions:

Micro Focus tool	Supported versions	Find more information...
ALM (Application Lifecycle Management) 12.xx	12.xx, 15.0	ALM Integration Jenkins page
ALM Lab Management	12.xx	ALM Integration Jenkins page
ALM Octane	12.53.20 or later (12.55.4 or later required for direct UFT integration and for PC integration using pipelines)	ALM Octane Help
LoadRunner	12.xx	LoadRunner Integration Jenkins page
Mobile Center (uploading apps)	2.0 - 3.2	Mobile Center Help
Performance Center	12.xx (12.53 or higher required for trend reports)	Performance Center Help
Service Virtualization	3.80 - 5.1	Service Virtualization Integration Jenkins page
StormRunner Functional	1.40	StormRunner Functional Help Center
UFT Pro (LeanFT)	14.03 and higher	Lean Functional Testing Help Center Blog: Integrating LeanFT with Jenkins— in just a few simple steps
Unified Functional Testing (UFT)	12.xx and 14.03 or higher	UFT Help Center Mobile Center Help

Prerequisites

1. Install one of the five latest LTS versions of Jenkins, (Click here for list)
2. Install the Jenkins Micro Focus Application Automation Tools plugin.
3. **Java version 8 or higher.** To verify your Java version, go to http://www.java.com/en/download/installed.jsp.
4. **ALM/Quality Center client** installed the machine that will run the tests. To check if you have a client installed, follow the instructions on this page: http://<your_server>:8080/qcbin/TDConnectivity_index.html
5. **For running UFT tests from ALM** install the ALMClient in common registration mode by accessing the following link from an Internet Explorer browser on the UFT machine: http://<almserver>:8080/qcbin/start_a.jsp?Common=true
6. **For running file system scenarios with LoadRunner or UFT** you need to set up a node in Jenkins. For details, see Create an execution node.
7. **For building and tracking pipelines on ALM Octane:**
    - JQuery Plugin 1.7.2-1 or later (Required to enable the integration)
    - A plugin that enables publishing test results. For example, JUnit Plugin 1.10 or later, NUnit plugin, and so on (Required to enable ALM Octane to collect your automated test results.)
8. **For ALM Octane integration with UFT:**
    - Jenkins Git plugin (version 2.4.4 or later)

User Guide

You can run client side-or server-side (Default or Functional) test sets and build verification suites from Application Lifecycle Management (ALM) or functional tests from the file system. You can create and configure ALM Octane pipelines from the ALM Octane user interface, or on Jenkins.

Rebranding script

Due to rebranding in this version of the plugin to the "Micro Focus" name, certain job history and configurations from previous jobs may be not be recognized by Jenkins. Before you use this version of the plugin, back up all projects and jobs, and all job configurations related to the plugin that are included in the General Jenkins configuration. Run the following job to convert your earlier configurations:

- Create a new **Free-Style** project.
- Click **Add build step** and select **Fix old Micro Focus Jenkins builds**.
- Run the job.
- Restart the server after the build status is successful. You can restart the server by entering the following URL: **Jenkins_URL/safeRestart**

Create an execution node

1. Creating an execution node is only required when running scenarios from LoadRunner or UFT, that are stored on the file system. You only need to set an execution node if the Jenkins master machine is **not** the executing machine. If you are running LoadRunner/UFT on the master machine, you do not need to set and select a node.
2. Go to the Jenkins Server home page.
3. Click the **Manage Jenkins** link in the left pane.
4. In the Manage Jenkins Page click **Manage Nodes**.
5. Click **New Node**.
6. Enter a **Node name**, select **Permanent Agent**, and then click **OK**.
7. In the next screen, enter information in the required fields. Enter the full path of the **Remote Root directory**, to which the tests results will be saved.
8. Provide one or more labels for the node. Separate the values with a space. Through these labels, you will be able to identify the nodes used in the job.
9. In the **Usage** area, select **Only build jobs with label expressions matching this node**.
10. Click **Save**.
11. Connect the execution node to the Jenkins server as described below.

Connect an execution node to the Jenkins server

1. On the computer that you defined as an execution node, open a browser and go to the Jenkins Server home page.
2. Open **Manage Jenkins** > **Manage Nodes**.
3. If there is a warning mark adjacent to the node you want to connect, click the node's link. You are prompted to connect the agent to Jenkins.
4. Click **Launch** to download the agent and then run the downloaded agent. Alternatively, use one of the other methods suggested on the screen.

Support for Pipelines

Generate pipeline code

To set up a pipeline test job for your Micro Focus testing tool:

a. From Jenkins Dashboard. click **New Job** or select an existing one.
b. On the page that opens, enter a job name (for a new job), click **Build a Pipeline project**, and click **OK**.
c. In the Project Configuration page, scroll down to the Pipeline section.
d. Enter the **stage** and **node** arguments into the **Script** area. For example,

?

```
stage('RunUFTTestFromFS'){ // The stage name
    node('Test'){ //  The name of the node in which to run the test.
```

2. Prepare the code for your testing tool:
   a. Click the **Pipeline Syntax** link.
   b. In the Snippet Generator drop down, select the desired step, for example, **uftScenarioLoad: Run UFT scenario**.
   c. Fill in the fields as required. Fields marked in red are mandatory. **Note:** For fields that take multiple values, such as in the **Tests** field, separate multiple entries with a line break.
   d. If relevant, select one of the **Report archive modes** (below).
   e. Click **Generate Pipeline Script**. Copy the code to the clipboard.
3. Return to the Project Configuration page, and paste the generated Groovy script into the **Script** field in the Pipeline section.
4. Repeat the above steps to add other commands to your script.
5. Save the script and run or schedule the job as you would with any standard Jenkins job.
6. After the test run, click the **Console** link on the dashboard to see a link to your results. Copy the link to your browser (Internet Explorer required to view ALM test sets in ALM).

Supported Pipeline job types

The available Pipeline job types
are loadRunnerTest, uftScenarioLoad, runFromAlmBuilder, sseBuild, sseBuildAndPublish, pcBuild, svChangeModeStep, svDeployStep, svExportStep, and svUndeployStep.

Product	Pipeline step name	Description
LoadRunner	loadRunnerTest	Run LoadRunner performance tests from a file system scenario file
UFT	uftScenarioLoad	Run a UFT scenario from file system scenario
ALM	runFromAlmBuilder	Execute functional tests from ALM
ALM Lab Management	sseBuild	Execute tests using ALM Lab Management
ALM Lab Management	sseBuildAndPublish	Execute tests using ALM Lab Management and Publish tests result
Performance Center	pcBuild	Execute tests using Performance Center
Service Virtualization	svChangeModeStep	Change the mode of a Virtual Service
Service Virtualization	svDeployStep	Deploy a Virtual Service
Service Virtualization	svExportStep	Export a Virtual Service
Service Virtualization	svUndeployStep	Undeploy a Virtual Service

Pipeline jobs are not supported for Mobile Center uploads, ALM test uploader, and ALM AUT job types.

Report archive modes

The available archive modes are:

- Archive test report for failed tests
- Always archive test reports
- Always archive and publish test reports (LR only)
- Do not archive test reports

Enable Non-English Languages

In order to allow the add-in to support non-English languages, make the following changes to your Jenkins Master and nodes configuration:

Master configuration

1. Open the **Jenkins.xml** file in the Jenkins installation folder, for example: c:\Jenkins\Jenkins.xml.
2. Go to the <service> -> <arguments> section and add the **-Dsun.jnu.encoding=UTF-8 -Dfile.encoding=UTF-8** flags after the **-jar** flag.

For example:

```
<service>
<id>jenkins</id>
<name>Jenkins</name>
<description>This service runs Jenkins continuous integration system.</description>
<env name="JENKINS_HOME" value="%BASE%"/>
<!--
This example assumes that you have java in your PATH.
To run Jenkins with a specific version of Java, specify a full path to the desired java.exe.
-->
<executable>%BASE%\jre\bin\java</executable>
<arguments>-Xrs -Xmx256m -Dhudson.lifecycle=hudson.lifecycle.WindowsServiceLifecycle -jar -Dsun.jnu.
encoding=UTF-8 -Dfile.encoding=UTF-8 "%BASE%\jenkins.war" --httpPort=8080 --webroot="%BASE%\war"</arguments>
<!--
The *interactive* flag causes the empty black Java window to be displayed.
<interactive />
-->
<logmode>rotate</logmode>
<onfailure action="restart" />
</service>
```

Slave configuration

1. Open the **jenkins-slave.xml** file on the Jenkins slave (node) machine in the folder that you designated.
2. Go to the <service> -> <arguments> section and add the **-Dsun.jnu.encoding=UTF-8 -Dfile.encoding=UTF-8** flags after the **-jar** flag.

```
<service>
<id>jenkinsslave-c__jkns</id>
<name>jenkinsslave-c__jkns</name>
<description>This service runs a slave for Jenkins continuous integration system.</description>
<!--
This example assumes that you have java in your PATH.
To run Jenkins with a specific version of Java, specify a full path to the desired java.exe.
-->
<executable>C:\Program Files (x86)\Java\jre1.8.0_91\bin\java.exe</executable>
<arguments>-Xrs -jar "%BASE%\slave.jar" -Dsun.jnu.encoding=UTF-8 -Dfile.encoding=UTF-8 -jnlpUrl http://xxx.xxx.
xxx.xxx:8080/jenkins/computer/VM112233/slave-agent.jnlp -secret xyzabc</arguments>
<!--
The *interactive" flag causes the empty black Java window to be displayed.
<interactive />
-->
<logmode>rotate</logmode>
<onfailure action="restart" />
</service>
```

Configuration for Java Web Start clients

1. On the Jenkins master machine, go to **Manage Jenkins -> Manage Nodes**. Click the relevant node or slave. and select **Slave -> Configure -> Advanced**.
2. Add the following to the JVM options text box – **“-Dsun.jnu.encoding=UTF-8 -Dfile.encoding=UTF-8”**.
3. Restart your Jenkins master, node, and slave.
4. Verify that your changes took effect:
    a. For the Master: Go to **Manage Jenkins -> System information -> “file.encoding”, “sun.jnu.encoding”**. The value should be “UTF-8”.
    b. For the node/slave: On the master machine, go to **Manage Jenkins -> Manage Nodes**.
    Click on the relevant node or slave and select **System information-> “file.encoding”, “sun.jnu.encoding”**. The value should be “UTF-8”.
    Workaround: If you find that the slave machine still does not support localized encoding, launch the slave from the command line as shown in the image below:

For example:

```
java -Dsun.jnu.encoding=UTF-8 -Dfile.encoding=UTF-8 -jar slave.jar -jnlpUrl http://xxx.xxx.xxx.xxx:8080/jenkins
/computer/VM112233/slave-agent.jnlp -secret xyzabc
```

Tips and Troubleshooting

Integration Issues

- If your job includes UFT, QTP, or Service Test tests running on a remote ALM/QC machine (run mode = Run Remotely), you should manually stop the test execution.
- When UFT is installed on the slave machine (node), the LoadRunner test job will fail in the **Analyze Result** stage. **Workaround:** Add the path of the LoadRunner **bin** folder (**%LR_PATH%/bin**) to the **PATH** environment variable.

Content Security Policy Header

Starting with version 1.641 (or 1.625.3), Jenkins introduced the **Content-Security-Policy** header. This prevents some of the links that appear in the integration to become inoperable. For example, the links to the LoadRunner Performance and UFT HTML reports will not work.

For workarounds to enable viewing UFT HTML Reports, see the UFT Help Center.

Workarounds to enable viewing LoadRunner Performance reports:

- View the reports locally on the slave machine, in your Jenkins results folder under the relevant build ID, or on the master machine in the Jenkins builds folder. Go to the **%jenkins% \jobs\<job_name>\builds\<job_count>\PerformanceReport** folder. For example, **C:\Program Files (x86)\Jenkins\jobs\Plug_Debug\builds\5\PerformanceReport\index.html**.
- Click **Manage Jenkins > Script Console**, and run one of the following scripts from the console:
  - Type the following line and press **Run**.

    ```
    System.setProperty("hudson.model.DirectoryBrowserSupport.CSP", "")
    ```

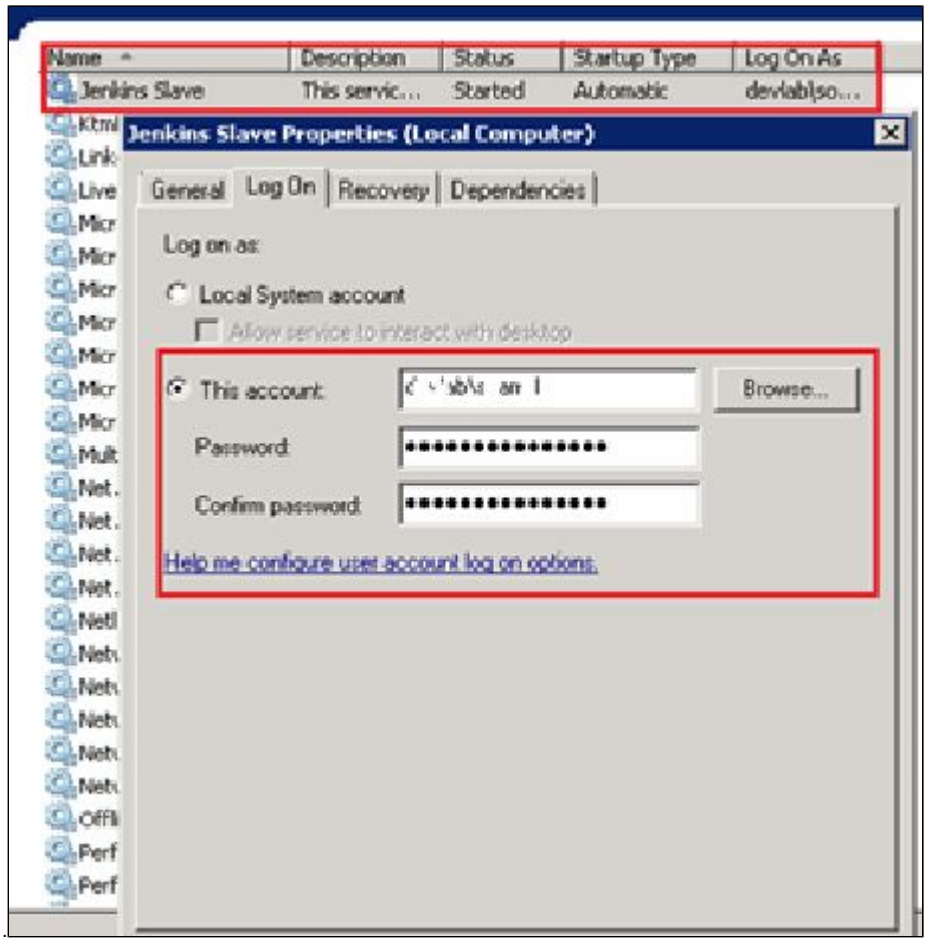
  - Alternatively, type the following line and press **Run**.

    ```
    System.setProperty("hudson.model.DirectoryBrowserSupport.CSP", "sandbox; default-src 'none'; img-src 'self'; style-src 'self' 'unsafe-inline'; script-src 'self' 'unsafe-inline'; child-src 'self';")
    ```

  - Another option that is more secure, but it disables the left pane menu and embedded Javascript:

    ```
    System.setProperty("hudson.model.DirectoryBrowserSupport.CSP", "sandbox; default-src 'none'; img-src 'self'; style-src 'self'; script-src 'self'; child-src 'self';")
    ```

For best performance, it is recommended to install the testing tools and ALM/QC as nodes on slave machines, and not directly on the Jenkins server. For instructions on configuring nodes, see https://wiki.jenkins.io/display/JENKINS/Step+by+step+guide+to+set+up+master+and+slave+machines+on+Windows.
On the Jenkins slave machine, make sure the the Jenkins Slave service is not logged in with the **Local System account**. To function properly, it should log in with an account the has administrator privileges.


Name
Description
Status
Startup Type
Log On As
Jenkins Slave
This servic...
Started
Automatic
devlab\so...
Jenkins Slave Properties (Local Computer)
General
Log On
Recovery
Dependencies
Log on as:
Local System account
Allow service to interact with desktop
This account:
Browse...
Password:
Confirm password:
Help me configure user account log on options.