LabVIEW CI Service

The LabVIEW Continuous Integration Server and Project Builder require LabVIEW 2014 or later.

Setting up the CI Server

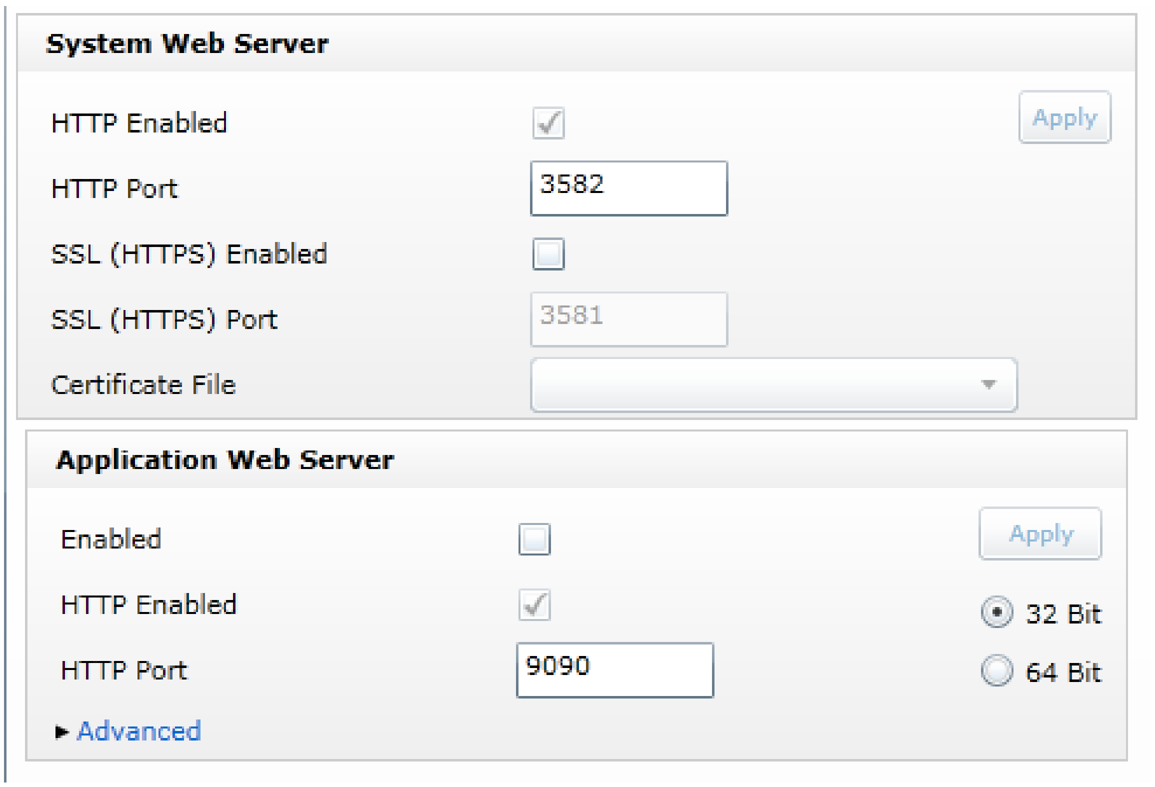
Install LabVIEW

The CI Server needs a copy of the LabVIEW development environment to build applications. We assume you know how to do that.

However, you must change the LabVIEW Web Server port.

Jenkins and the LabVIEW Web Server use the same port. This is known to cause problems such as the Jenkins page not being able to load. To rectify this, the port of the LabVIEW Web Manager needs to be changed.

1. Open the LabVIEW Application Web Server in internet explorer window
2. Navigate to the "Web Server Configuration" Page
3. Under System Web Server and Application Web Server. Verify that the HTTP or HTTPS ports are **NOT** 8080. (They can be any other number not being used ex 9090)



System Web Server

HTTP Enabled	✓
HTTP Port	3582
SSL (HTTPS) Enabled	☐
SSL (HTTPS) Port	3581
Certificate File	

Application Web Server

Enabled	☐
HTTP Enabled	✓
HTTP Port	9090

[✓ 32 Bit] 64 Bit

Installing Jenkins

1. Navigate to https://jenkins-ci.org/
2. Download the "Latest and greatest" version.
3. Once downloaded run the installer

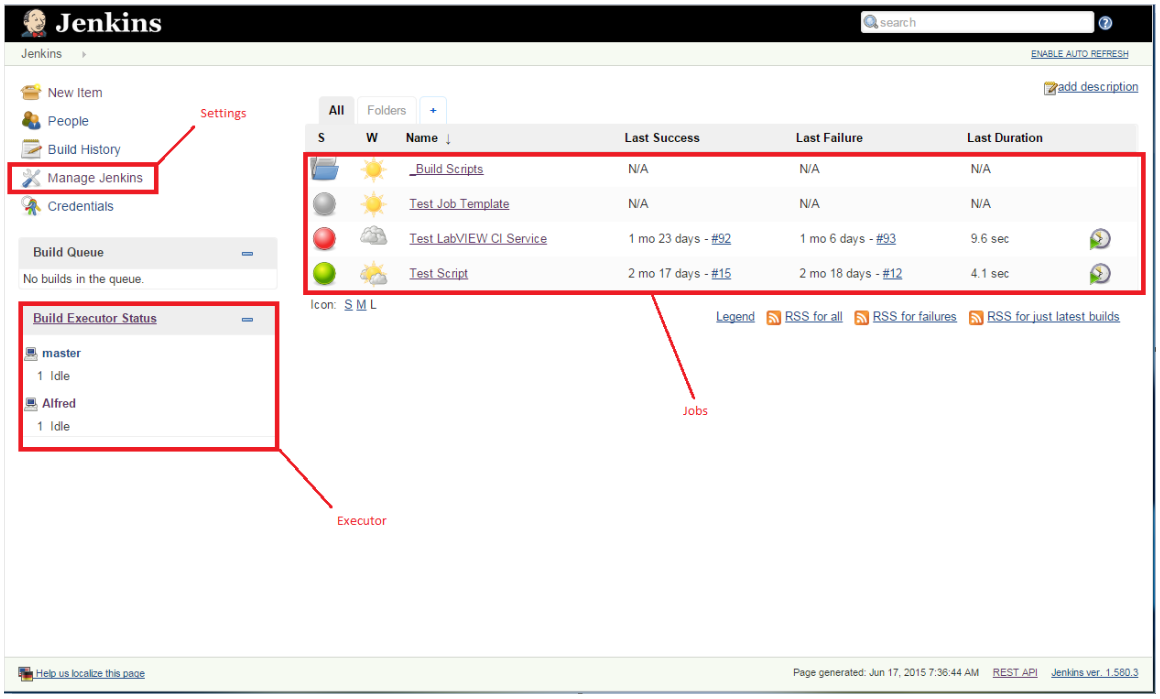
Removing Read Only

1. Navigate to the Jenkins folder in Program files
2. Make Sure the folder is not set to Read Only and All users have Full Control over the folder.

Launching Jenkins

1. Open Jenkins by using the default URL: http://localhost:8080

Note: The address can be changed in the "Manage Jenkins" Settings page



No builds in the queue.

Build Executor Status
master
1 Idle
Alfred
1 Idle

S	W	Name ↓	Last Success	Last Failure	Last Duration
		_Build Scripts	N/A	N/A	N/A
		Test Job Template	N/A	N/A	N/A
🔴		Test LabVIEW CI Service	1 mo 23 days - #92	1 mo 6 days - #93	9.6 sec
🟢		Test Script	2 mo 17 days - #15	2 mo 18 days - #12	4.1 sec

Icon: S M L

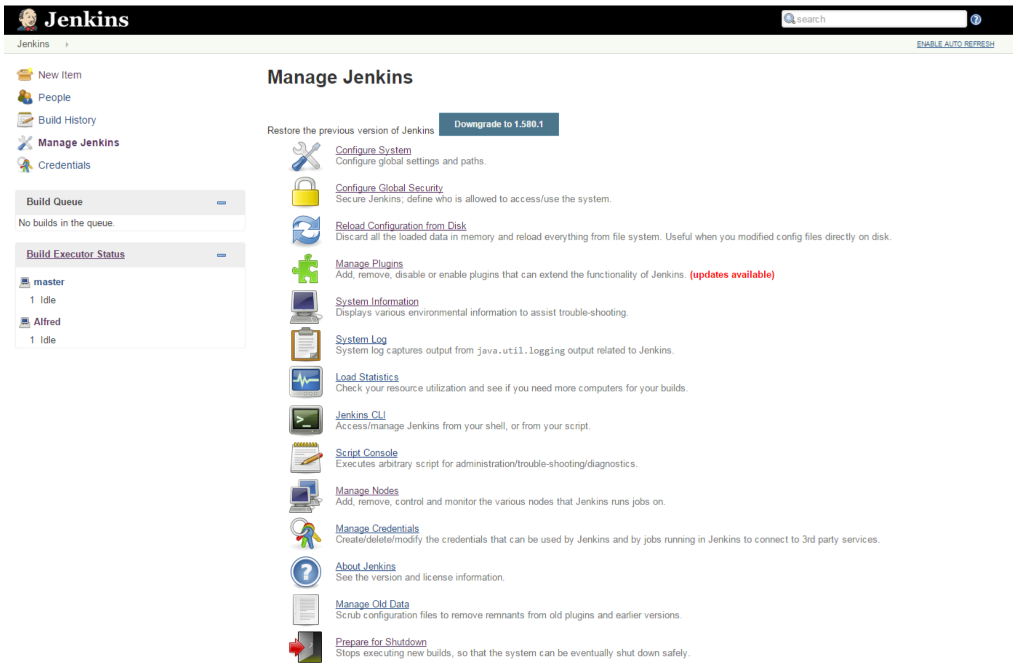
Page generated: Jun 17, 2015 7:36:44 AM REST API Jenkins ver. 1.580.3

The "Jobs" section shows a list of all of the Jenkins jobs. If you use the Alfred tool, this section will show your jobs in folders.

The "Executor" section shows the number of Executors and Nodes. A Jenkins job runs on a single executor within a node. Nodes can be on the same computer on or on a different computer. By default, there is one executor and one node, but this can be configured.

Installing Plugins

1. Navigate to Manage Jenkins



No builds in the queue.

master
1 Idle
Alfred
1 Idle

Manage Jenkins

Restore the previous version of 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. **(updates available)**

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.

Jenkins CLI
Access/manage Jenkins from your shell, or from your script.

Script Console
Executes arbitrary script for administration/trouble-shooting/diagnostics.

Manage Nodes
Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

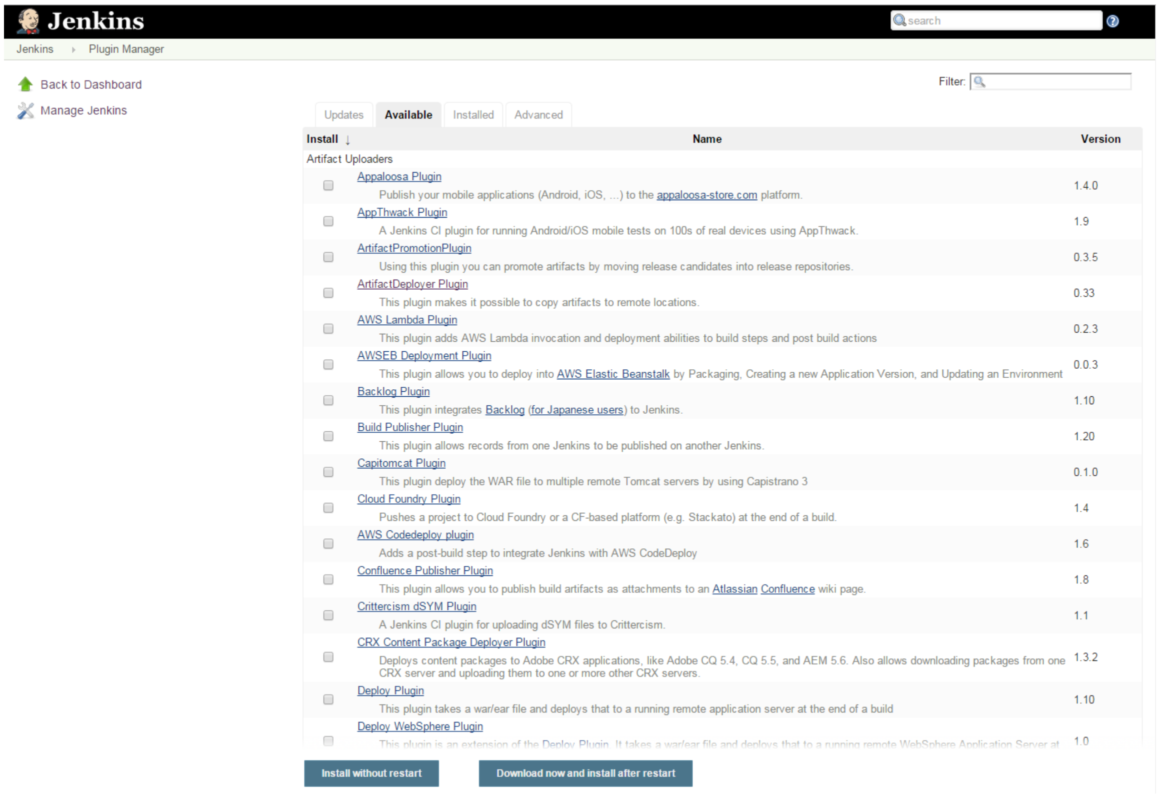
Manage Credentials
Create/delete/modify the credentials that can be used by Jenkins and by jobs running in Jenkins to connect to 3rd party services.

About Jenkins
See the version and license information.

Manage Old Data
Scrub configuration files to remove remnants from old plugins and earlier versions.

Prepare for Shutdown
Stops executing new builds, so that the system can be eventually shut down safely.

2. Navigate to Manage Plugins



Install ↓	Name	Version
	Artifact Uploaders	
☐	Appaloosa Plugin Publish your mobile applications (Android, iOS, ...) to the appaloosa-store.com platform.	1.4.0
☐	AppThwack Plugin A Jenkins CI plugin for running Android/iOS mobile tests on 100s of real devices using AppThwack.	1.9
☐	ArtifactPromotionPlugin Using this plugin you can promote artifacts by moving release candidates into release repositories.	0.3.5
☐	ArtifactDeployer Plugin This plugin makes it possible to copy artifacts to remote locations.	0.33
☐	AWS Lambda Plugin This plugin adds AWS Lambda invocation and deployment abilities to build steps and post build actions	0.2.3
☐	AWSEB Deployment Plugin This plugin allows you to deploy into AWS Elastic Beanstalk by Packaging, Creating a new Application Version, and Updating an Environment	0.0.3
☐	Backlog Plugin This plugin integrates Backlog (for Japanese users) to Jenkins.	1.10
☐	Build Publisher Plugin This plugin allows records from one Jenkins to be published on another Jenkins.	1.20
☐	Capitomcat Plugin This plugin deploy the WAR file to multiple remote Tomcat servers by using Capistrano 3	0.1.0
☐	Cloud Foundry Plugin Pushes a project to Cloud Foundry or a CF-based platform (e.g. Stackato) at the end of a build.	1.4
☐	AWS Codedeploy plugin Adds a post-build step to integrate Jenkins with AWS CodeDeploy	1.6
☐	Confluence Publisher Plugin This plugin allows you to publish build artifacts as attachments to an Atlassian Confluence wiki page.	1.8
☐	Crittercism dSYM Plugin A Jenkins CI plugin for uploading dSYM files to Crittercism.	1.1
☐	CRX Content Package Deployer Plugin Deploys content packages to Adobe CRX applications, like Adobe CQ 5.4, CQ 5.5, and AEM 5.6. Also allows downloading packages from one CRX server and uploading them to one or more other CRX servers.	1.3.2
☐	Deploy Plugin This plugin takes a war/ear file and deploys that to a running remote application server at the end of a build	1.10
☐	Deploy WebSphere Plugin This plugin is an extension of the Deploy Plugin. It takes a war/ear file and deploys that to a running remote WebSphere Application Server at	1.0

3. Click on the Available Tab
4. Search for and select:
    a. CloudBees Folders Plugin
    b. HTTP Request Plugin
    c. Job DSL Plugin
    d. Run Condition Extras Plugin
    e. Windows Slaves Plugin
    f. Workspace Cleanup Plugin
    g. (Whatever Source Code Control you use)
    h. Green Balls

    Note: The Windows Slaves Plugin depends on one or more of these other plugins. You will not be able to find or select this plugin until you have installed the others, and restarted Jenkins.

5. Select "Download now and install after restart"

Setting up Email Notification

1. Navigate to Manage Jenkins
2. Select Configure System
3. Scroll down to the Email Notification section and enter your company's SMTP server and default user email suffix
4. Click Save

Setting Up Slave Node for Alfred

Note: Setting this up is optional but highly recommended. Since the LabVIEW CI Project builder tool uses Jenkins scripts to build Jenkins jobs, these scripts run in Jenkins. If, you do not setup a slave node dedicated for Alfred, there may be a delay in the creation of jobs if another job is already running.

1. Follow these steps: https://wiki.jenkins-ci.org/display/JENKINS/Step+by+step+guide+to+set+up+master+and+slave+machines
2. Name the Slave node "Alfred"
3. After creating this node, make sure you have read and write access to the root folder you specified.

Note: It is recommended that you install the slave node as a Windows service. If you attempt to do so, and encounter an access error, stop the node and exit your browser. Run your browser again as an administrator, restart the slave node, and try again to run it as a service.

Configuring Number of Executors

Change the number of executors to 1

Restart Jenkins before attempting to install the LabVIEW CI Server. Navigate to http://localhost:8080/restart.

LabVIEW CI Installation

The LabVIEW CI Installation has two components. The LabVIEW CI Web Service runs on the build service machine, and processes build step requests from Jenkins. The Alfred tool is installed in your LabVIEW tools menu, and runs from your local development environment.

The components are distributed as VI Packages.

Installing the LabVIEW CI Web Service

You will need to install two VI Packages on your CI server computer. Depending on how your server's security is configured, it may be necessary to run VI Package Manager as an administrator.

Download and install the NI Launch Remote Actor package. You can find it here: https://decibel.ni.com/content/message/46210#46210

Install the package ni_lib_labview_ci_server_source package.

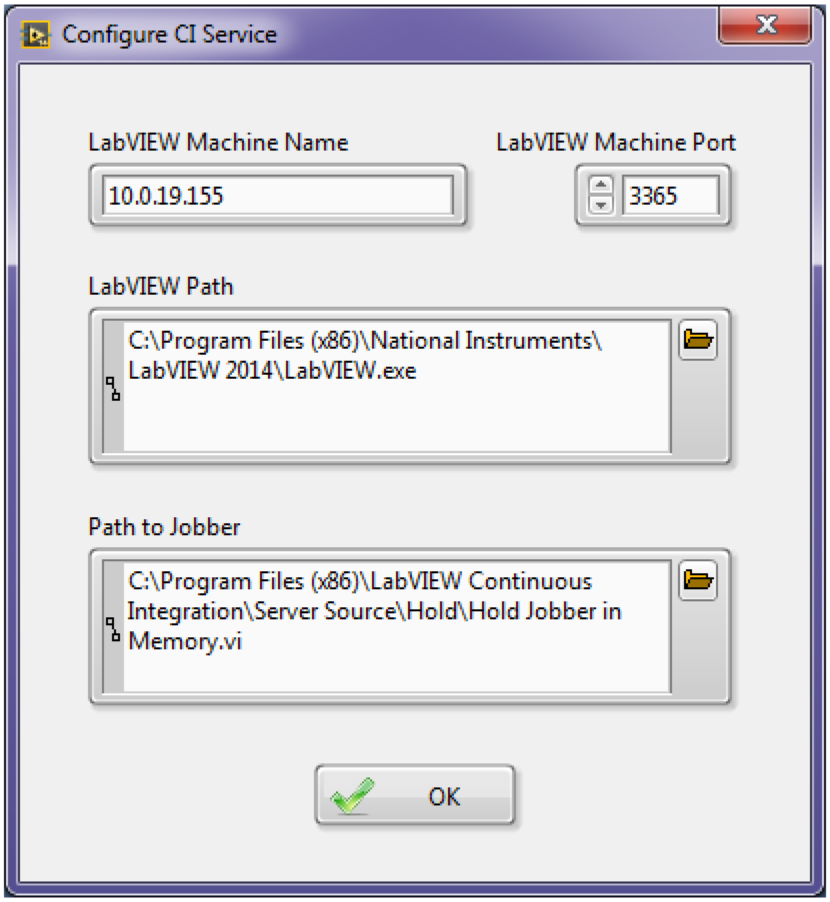
Once installed, navigate to the installation folder, which can be found at <applications folder>\National Instruments\LabVIEW Continuous Integration. Make sure you have read and write access to this folder.

Open this folder, navigate to the Server Source folder, and open LabVIEW Continuous Integration.lvproj.

Open and run Build Server.vi. This VI builds the various components of the CI Server.

Build Server.vi will automatically launch the installer for the CI Web Service. Follow the instructions in the installer dialog.

After the web service has been installed, you will be presented with a setup dialog, as shown.



LabVIEW Machine Name: 10.0.19.155

LabVIEW Machine Port: 3365

LabVIEW Path: C:\Program Files (x86)\National Instruments\LabVIEW 2014\LabVIEW.exe

Path to Jobber: C:\Program Files (x86)\LabVIEW Continuous Integration\Server Source\Hold\Hold Jobber in Memory.vi

Make sure the information shown is correct, and click OK.

You should have an icon on your desktop for the LabVIEW CI Web Service. Launch the service. Depending on your security configuration, you may need to launch the service as an administrator.

Restart Jenkins, as described above.

Your CI Server is now installed and ready, but you must define a Jenkins project to perform builds. This is done using the LabVIEW CI Project Builder.

Installing and Using the LabVIEW CI Project Builder

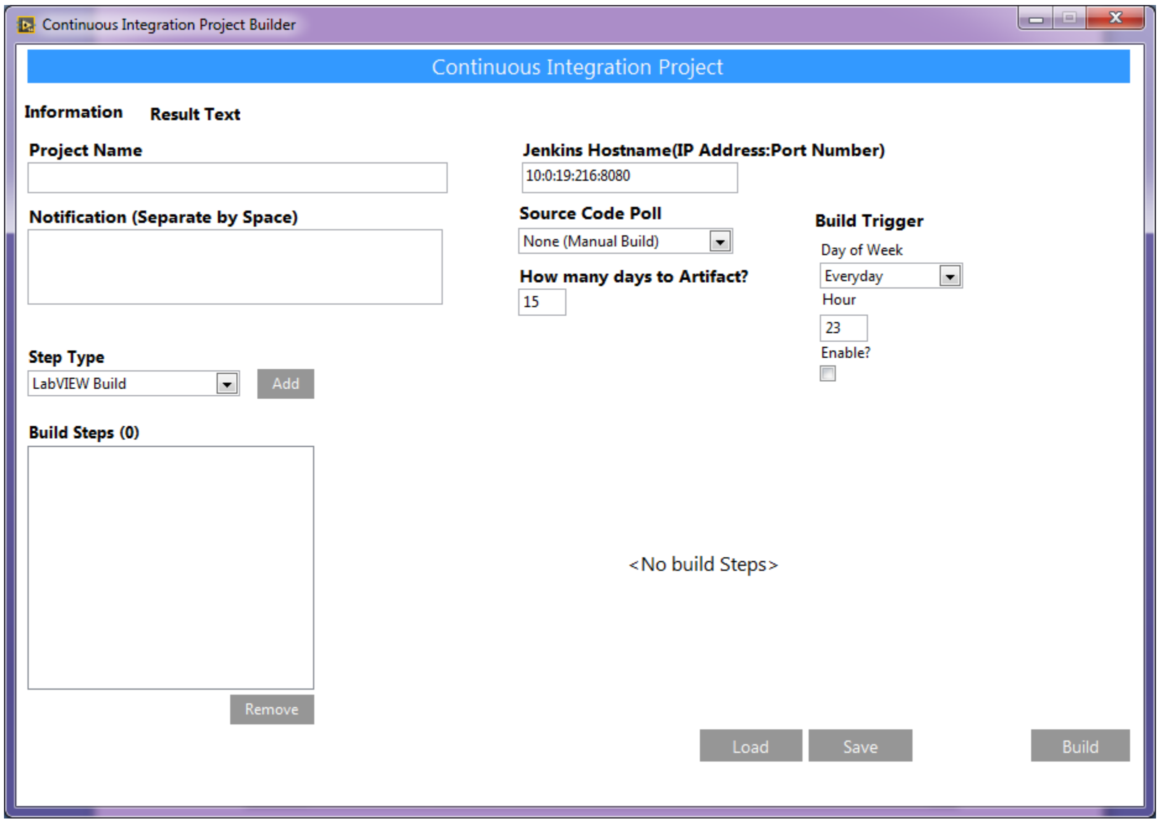
The CI Project Builder is an addition to the tools menu of your LabVIEW development environment.

While you *can* install this tool on the CI Server's copy of LabVIEW, it is intended to be installed and used on your developers' machines.

The tool is distributed as a VI Package, and installs in the LabVIEW\Projects folder. From VI Package Manager, install the package ni_lib_labview_ci_client_source. The CI Project Builder is ready for immediate use.

To launch the CI Builder, select Tools » CI Project Builder… from the LabVIEW menu.

You will be presented with the following dialog box:


Continuous Integration Project Builder

Continuous Integration Project

Information   Result Text

Project Name

Jenkins Hostname(IP Address:Port Number): 10:0:19:216:8080

Notification (Separate by Space)

Source Code Poll: None (Manual Build)

How many days to Artifact?: 15

Build Trigger
Day of Week: Everyday
Hour: 23
Enable?

Step Type: LabVIEW Build

Build Steps (0)

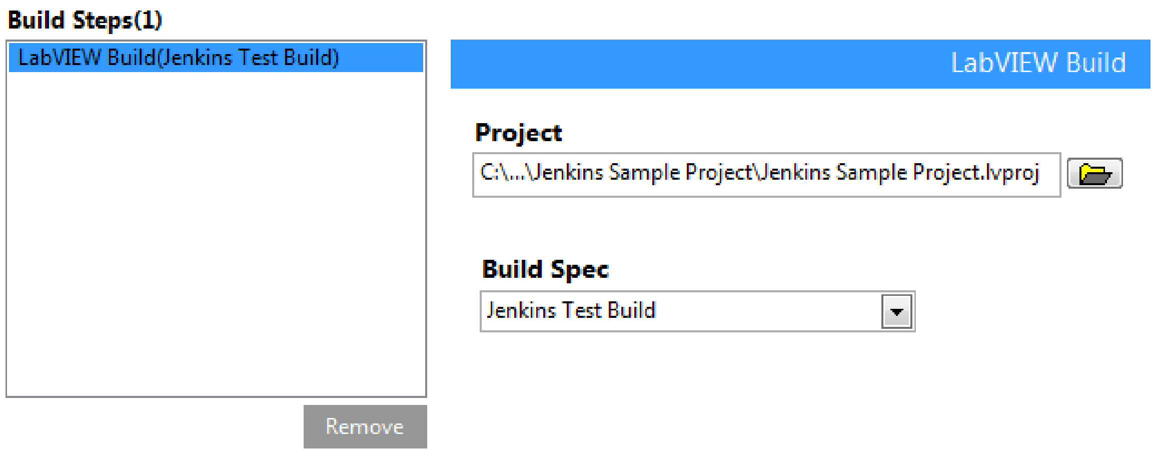
<No build Steps>

Fill in the indicated fields to define your CI project.

Important Note: The dialog box auto-populates the Jenkins Hostname field with your local IP address. Presumably, you will want to use a CI Server on a different machine. To do so, update this field with the IP address of your CI Server.

You can define as many build steps as you wish. Currently, only LabVIEW Build and Run VI are defined.

If you add a LabVIEW Build step, you will see the following, additional fields:


Build Steps(1)
LabVIEW Build(Jenkins Test Build)

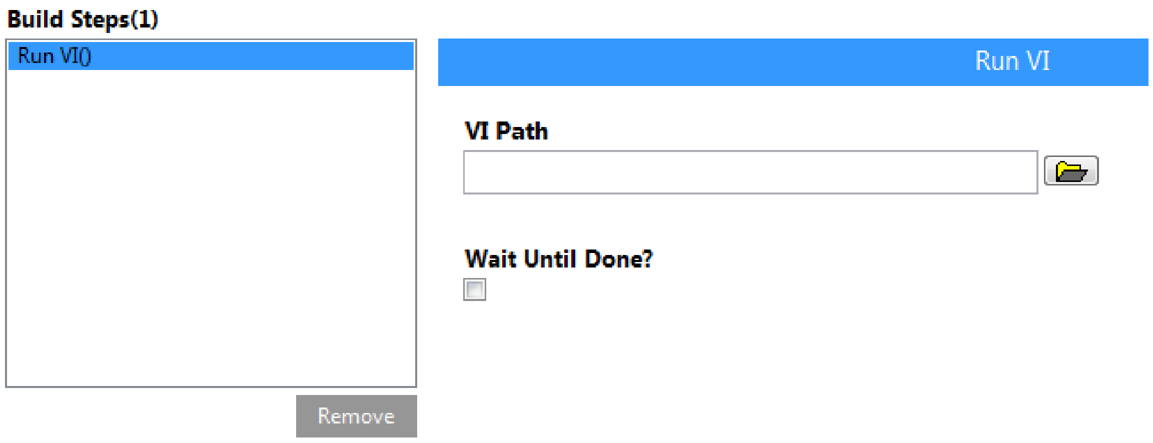
LabVIEW Build

Project: C:\...\Jenkins Sample Project\Jenkins Sample Project.lvproj

Build Spec: Jenkins Test Build

Select the Build Spec to build from the Build Spec pulldown menu.

Adding a Run VI step presents these fields:



In both cases, the path field should point to **the path where the project or VI resides on the CI Server**. For convenience, we strongly recommend that the local copy of your source code reside in the same folder structures on both the developer's machine and the CI Server machine. Virtual drives are handy for this.

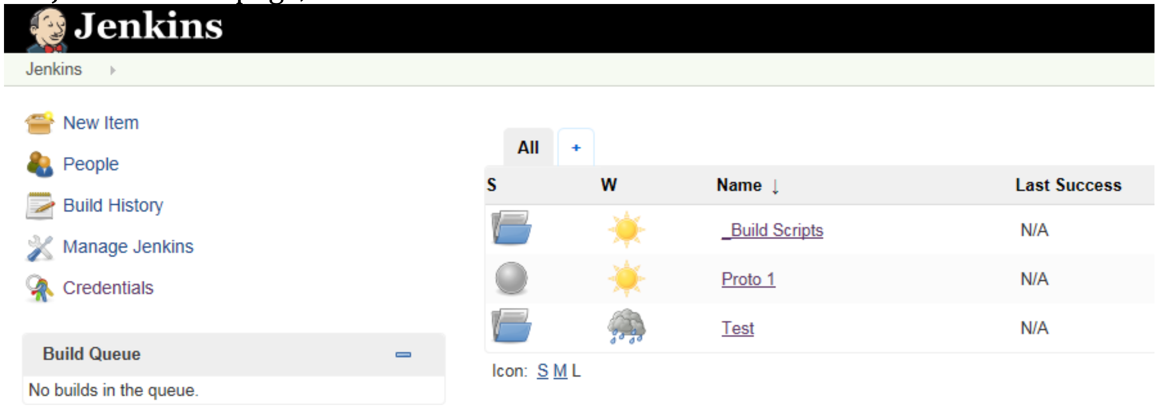
Additional step types will be added to this list in the future.

Save your configuration by clicking "Save." You can have more than one saved configuration, and you can retrieve them by clicking "Load."

Click "Build" to configure this project in Jenkins.

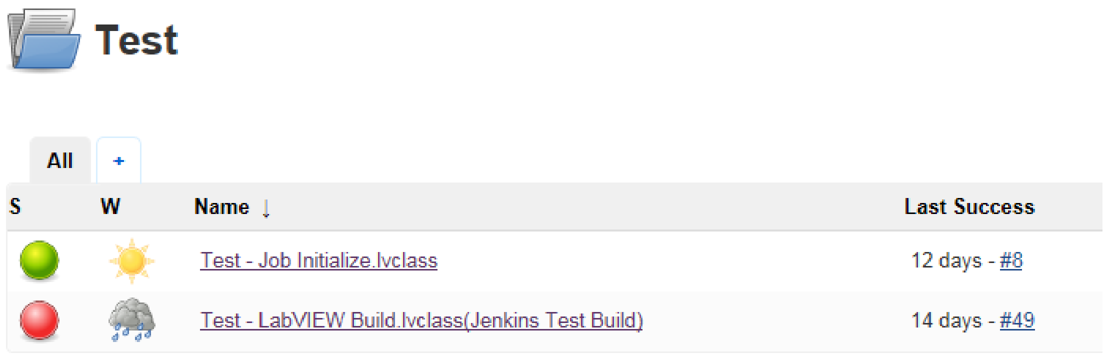
Configuring Source Code Control

When the CI Project Builder tool builds your Jenkins project, it creates a folder on the Jenkins main page, as shown here.



S	W	Name ↓	Last Success
		_Build Scripts	N/A
		Proto 1	N/A
		Test	N/A

In this illustration, the CI Project Builder created the "Test" folder. The new folder will contain all of the build steps you defined, plus a Job Initialize step, as shown here.

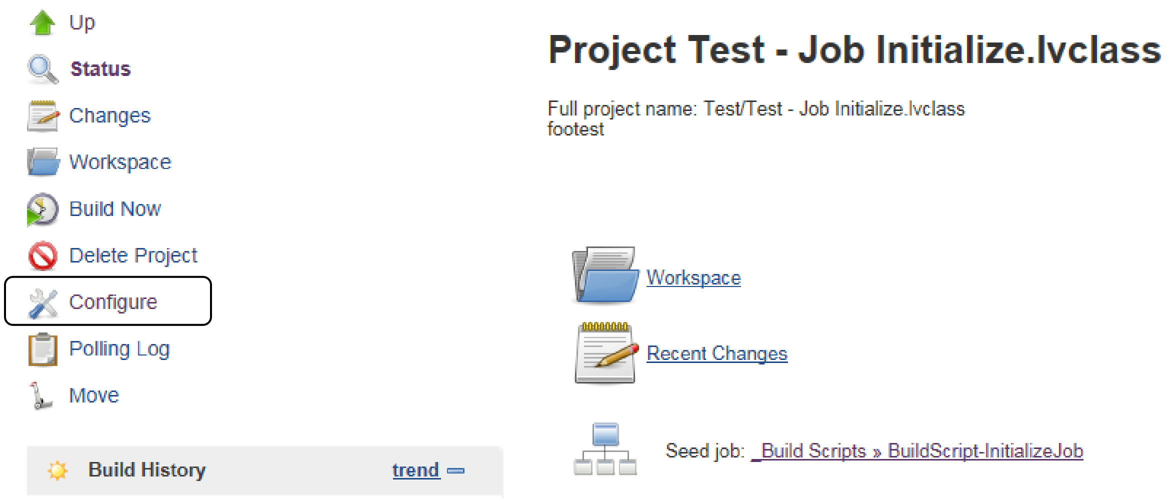


Test

All | +

S	W	Name ↓	Last Success
●	☀	Test - Job Initialize.lvclass	12 days - #8
●	☂	Test - LabVIEW Build.lvclass(Jenkins Test Build)	14 days - #49

Job Initialize is the step that performs source code synchronization. You must customize this step in Jenkins to interact with your source code control system. Click on the step name to open the step, and then click on Configure.



Project Test - Job Initialize.lvclass

Full project name: Test/Test - Job Initialize.lvclass
footest

Seed job: Build Scripts » BuildScript-InitializeJob

A future release of the build tool may allow you to customize this step from within the tool.

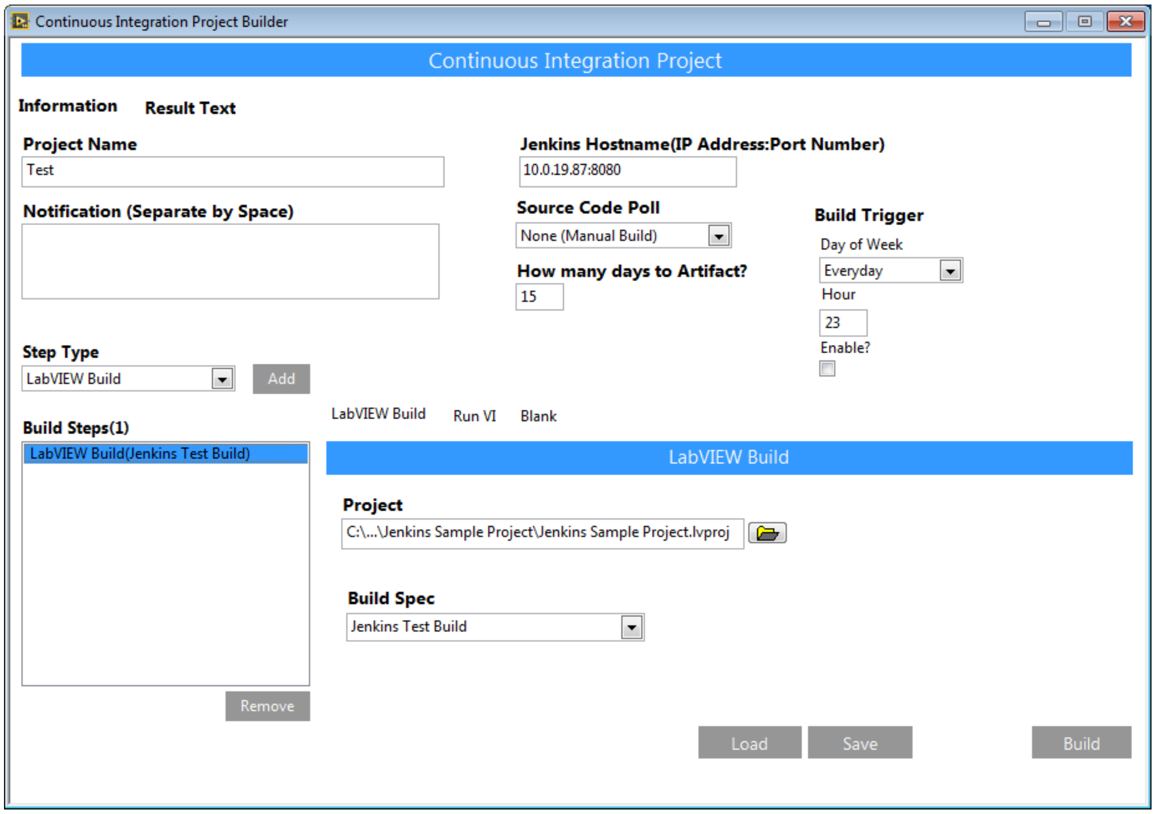
We intend to provide examples showing how to configure this step for common source code control tools at a future date.

Example

We have provided a sample project to test your installation. You can find it in the folder <applications>\National Instruments\LabVIEW Continuous Integration\Sever Source\Jenkins Sample Project.

The first step is to build the CI Project. Open LabVIEW, and select Tools » CI Project Builder…

Complete the dialog box as shown, using the IP address of your Jenkins server.


Continuous Integration Project Builder

Continuous Integration Project

Information Result Text

Project Name: Test

Jenkins Hostname(IP Address:Port Number): 10.0.19.87:8080

Notification (Separate by Space):

Source Code Poll: None (Manual Build)

How many days to Artifact?: 15

Build Trigger

Day of Week: Everyday

Hour: 23

Enable? ☐

Step Type: LabVIEW Build

Build Steps(1)

LabVIEW Build(Jenkins Test Build)

LabVIEW Build Run VI Blank

LabVIEW Build

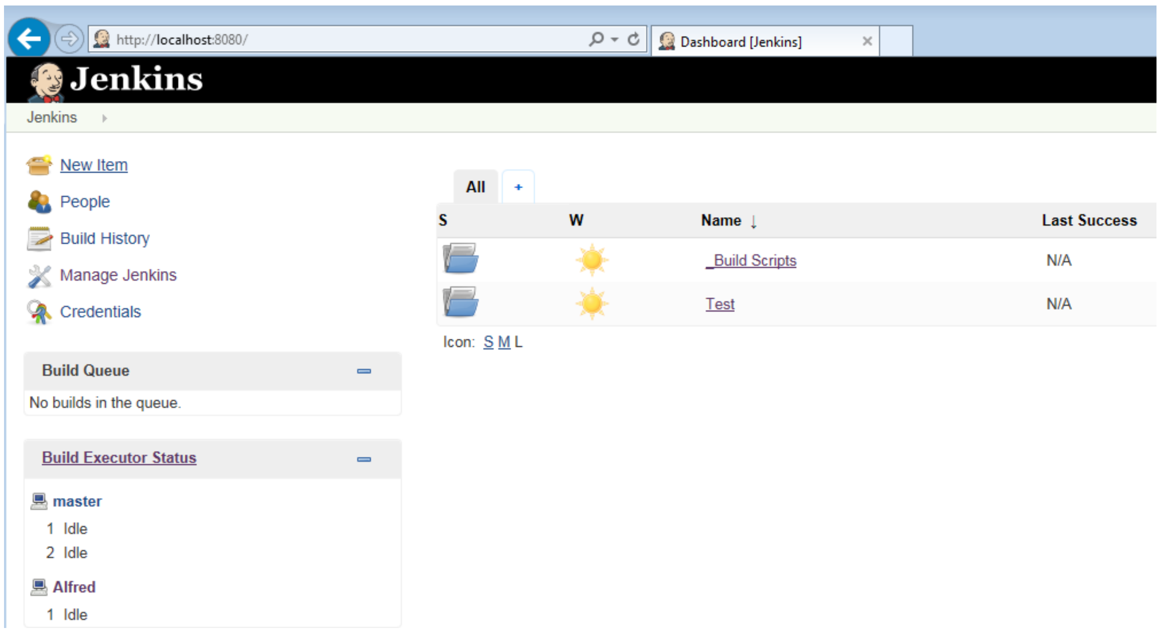
Project: C:\...\Jenkins Sample Project\Jenkins Sample Project.lvproj

Build Spec: Jenkins Test Build

For the project path, select the Jenkins Sample Project. Be sure to use the path to the project located **on the CI server.**

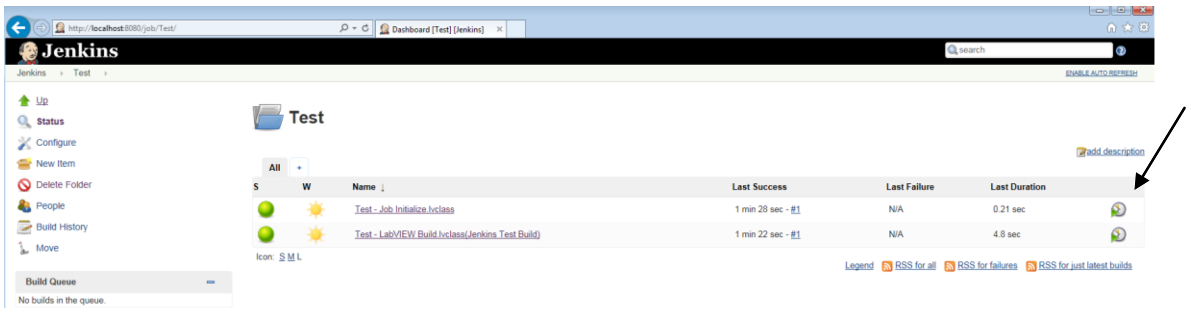
Save and Build the CI project. Close this dialog box after the project has been built.

The Jenkins main page should now look like this.



S	W	Name ↓	Last Success
		_Build Scripts	N/A
		Test	N/A

“Test” is the project folder for the sample project. Click on “Test” to open the Test folder.



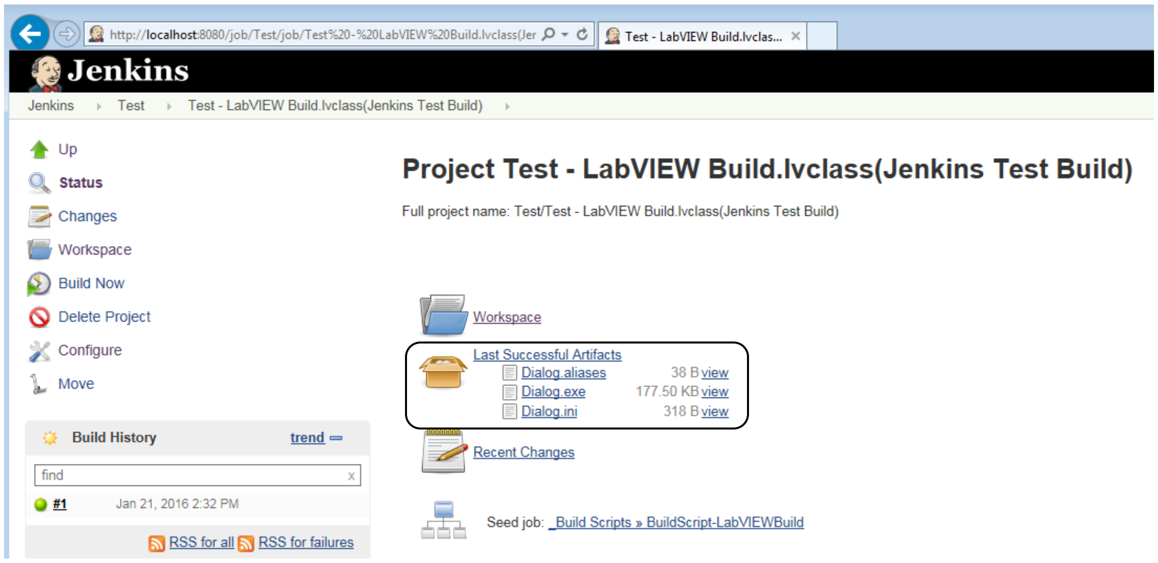
S	W	Name ↓	Last Success	Last Failure	Last Duration
		Test - Job Initialize.lvclass	1 min 28 sec - #1	N/A	0.21 sec
		Test - LabVIEW Build.lvclass(Jenkins Test Build)	1 min 22 sec - #1	N/A	4.8 sec

Click the run button to the far right of the “Test – Job Initialize.lvclass” jobs.

You should see the LabVIEW Build job appear in the queue, and then execute. LabVIEW should open and display the main panel for the LabVIEW Build job. After a short period, LabVIEW will exit, and the Build Executor status should be Idle.

Refresh the display to see the final status of these jobs.

To retrieve your artifacts, click on “Test – LabVIEW Build.lvclass (Jenkins Test Build).” The job page will contain links to your last successful build artifacts. Click to download the artifacts and test your results.



http://localhost:8080/job/Test/job/Test%20-%20LabVIEW%20Build.lvclass(Jer

Test - LabVIEW Build.lvclas...

Jenkins

Jenkins › Test › Test - LabVIEW Build.lvclass(Jenkins Test Build) ›

- Up
- Status
- Changes
- Workspace
- Build Now
- Delete Project
- Configure
- Move

Build History trend

find

#1 Jan 21, 2016 2:32 PM

RSS for all RSS for failures

Project Test - LabVIEW Build.lvclass(Jenkins Test Build)

Full project name: Test/Test - LabVIEW Build.lvclass(Jenkins Test Build)

Workspace

Last Successful Artifacts

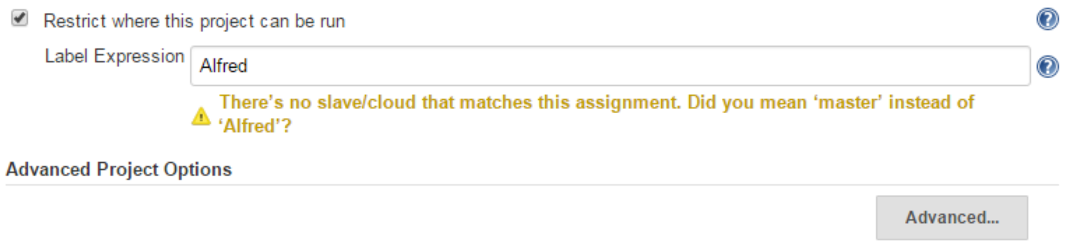
Dialog.aliases	38 B	view
Dialog.exe	177.50 KB	view
Dialog.ini	318 B	view

Recent Changes

Seed job: _Build Scripts » BuildScript-LabVIEWBuild

Common Issues:

1. If you did not setup a slave node, go into the Build Scripts configuration and Uncheck "Restrict where this project can run



☑ Restrict where this project can be run

Label Expression: Alfred

There's no slave/cloud that matches this assignment. Did you mean 'master' instead of 'Alfred'?

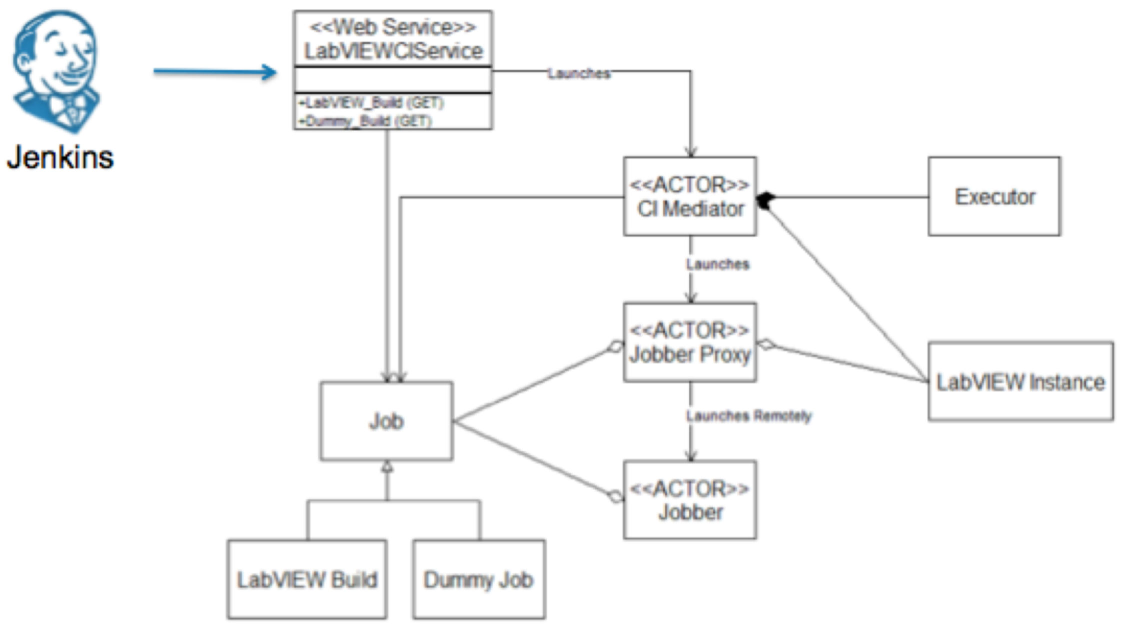
Advanced Project Options

Advanced...

LabVIEW Service Architecture

(This section is incomplete. It needs more detail on how to add step types to the CI Server. Adding step types to the CI Project Build tool is not discussed at all, partly because we anticipate significant changes to the underlying architecture of that tool. At the moment, it is not modular, and adding new step types requires more knowledge of its construction that we wish. We plan to change this in a future release. – ACS)

We have chosen to include this section for those who wish to extend the CI tools to meet their particular requirements.



Overview

Jenkins interacts with the LabVIEW Service via HTTP Plugin. The Service Executable contains the CI Mediator and Jobber Proxy.

Using the VI Job Type

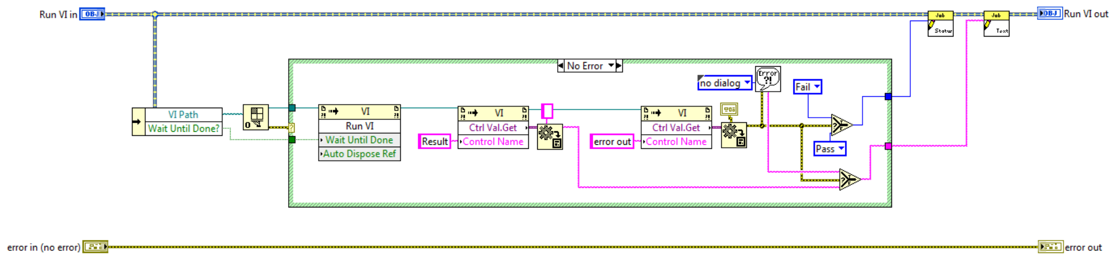
1. When using the VI Job Type to Run a VI, the only inputs that are needed are the VI Path and The Wait until Done Boolean. The Job Type have a built in reporting and error handling Capability. If you want text to be reported in Jenkins, the VI that is

ran needs to have an output terminal called “Result” and the Error Terminal should be named “error out”.

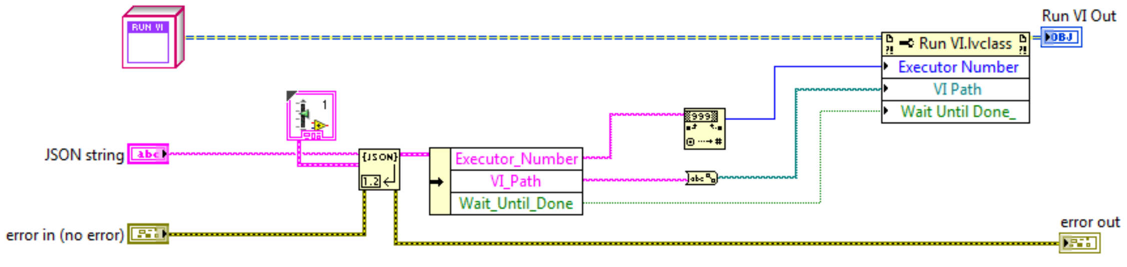
Add a Job Type

Adding to Web Service

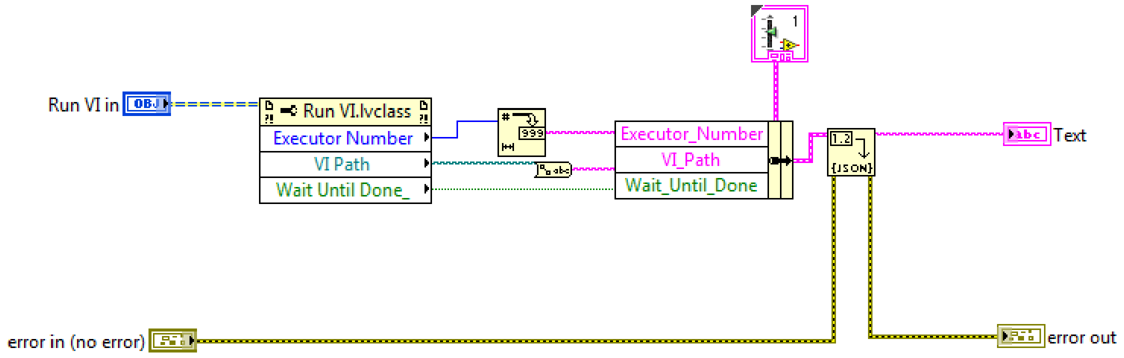
1. Create the Job Class of the Type you want.
2. Create a Type Def Cluster containing the Data. (Please Ensure the name of the controls do NOT have any spaces in them). The Cluster must also have the Executor Number in it.
3. Inherit from the Job Class
4. Override the Do This and add in Code to perform action. Please ensure you write the Status and the Result Text.



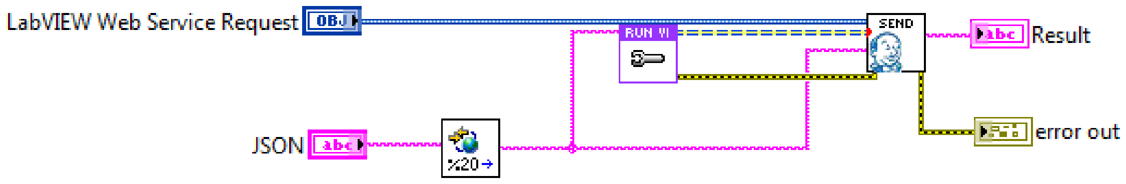
5. Create a Configure VI that takes the JSON string, converts to the Strict Type def cluster and places data into the class.



6. Override the Serialize Method and create a method that takes the Class data and flatten to a string.



7. Place class onto the VI called “Available Jobs”
8. Create an HTTP Get method in the Web Service and add the Job Type Method.


LabVIEW Web Service Request
RUN VI
SEND
Result
JSON
error out