
lf-releng-global-jjb

Release v0.37.0.dev0+460310f

May 01, 2019

Contents

1	Release Notes	3
2	Guides	5
2.1	Release Notes	5
2.2	Install	14
2.3	Configuration	18
2.4	Best Practices	22
2.5	Glossary	28
2.6	Appendix	28
3	Global JJB Templates	29
3.1	C/C++ Jobs	29
3.2	CI Jobs	32
3.3	INFO VOTE JOB	48
3.4	Global Macros	48
3.5	Maven Jobs	55
3.6	NodeJS Jobs	67
3.7	OpenStack Heat	69
3.8	Python Jobs	71
3.9	Release Announce	76
3.10	ReadTheDocs Jobs	76
3.11	Jenkins Views	79
3.12	WhiteSource Jobs	83
4	Indices and tables	85

Linux Foundation Release Engineering Global Jenkins Job Builder (JJB) Documentation.

Global-JJB is a library project containing reusable Jenkins Job Builder templates. Developed for LFCI to deploy management Jenkins jobs to an LF managed Jenkins instance, there are other jobs defined which may be helpful to projects that use the same build technology. The intention is to help projects save time from having to define their own job templates.

CHAPTER 1

Release Notes

Global JJB provides regular releases. The release notes for all releases are available in the relnotes directory in Git.

<https://github.com/lfit/releng-global-jjb/tree/master/relnotes>

2.1 Release Notes

2.1.1 v0.37.0

New Features

- Add verify, merge and stage templates for Java projects that build and wrap a JAR (e.g., a Spring-Boot application) inside a Docker image, and do not need to deploy any JAR libraries or POM files.

Upgrade Notes

- The next release of common-packer will require a minimum version of 1.3.2 for packer. The current release of packer is 1.4.0.

Bug Fixes

- The packer-merge job for Gerrit systems was improperly configured to use the Gerrit Trigger choosing strategy and not default. This caused issues unexpected issues with retriggering merged changes when the expectation was that it would pick up the latest change as per normal.
- This is a variable that does not need to be defined by the users of the jobs. The version needed in this builder step is inherited from versions.properties as “release_version” and it is fixed as that. This also helps teams not having to define this version in 2 places and just rely on version.properties.
- Projects using maven versions plugin let this plugin take care of updating their versions in the pom.xml. When maven-versions-plugin is set to “true”, skip the stripping of SNAPSHOTs from the pom.xml files. maven-versions-plugin is set to “false” by default.

Other Notes

- Update example Jenkins Init Script in README to redirect all output to a log file.

2.1.2 v0.36.0

Prelude

WhiteSource is a security and license compliance management platform. It is used to perform scans on a great variety of coding and scripting languages.

New Features

- New `comment-to-gerrit` builder will comment back to gerrit patchset if a file called `gerrit_comment.txt` is created by the build.
- Allows maven to run a clean install step before the WhiteSource scan runs the Unified Agent to fetch additional dependencies. Set to false by default.
- Job `{project-name}-whitesource-scan-{stream}` uses the WhiteSource Unified Agent scanner CLI tool to perform the code scan and report the results into the WhiteSource dashboard.

Bug Fixes

- Tag releases will now trigger a docs build to regenerate and update the release note link.
- Update `jenkins-cfg-verify` job to validate new images names obtained from `$GERRIT_REFSPEC` instead of the master branch.
- Hardcode project version to the “`GERRIT_BRANCH`”. Follow previous convention from CLM where reports were versioned after the branch name. Fix minor nits with bash variables.
- `wss-unified-agent.config` file should not be opened for configuration to tech teams. The config file should be part of Jenkins Settings Files and called via Managed Files. `wss-unified-agent.config` must be created in Jenkins config files based on `wss-unified-agent.config.example`.

Other Notes

- To run this job, a configuration file is needed (`wss-unified-agent.config.example`). A new secret text credential will need to be created. (ID=`wss-apiKey` Secret=WhiteSource organization API key)
- Update `lftools` version to **v0.23.1**.

2.1.3 v0.35.0

New Features

- The **jjb-merge** job now has a new parameter `jjb-workers` to allow configuration of the number of threads to run update with. Default is 0 which is equivalent to the number of CPU cores available on the system.
- New `info-vote-verify` macro Will count votes against an `INFO.yaml` change and submit automatically if a majority of committers vote +1 or +2 on the change. Job is triggered by +2 votes or a comment of “vote”

Other Notes

- The Maven Verify job will now call `-Dmaven.source.skip` to skip source jar generation in the verify job. This saves us some time in the verify build as the source artifacts are not useful in a verify job.

2.1.4 v0.34.0

New Features

- **jenkins-init-scripts** The ‘ciman’ repo is not longer required to be located in ‘/opt/ciman’.
- If-maven-set-version conditional step for If-maven-stage to allow teams to run Maven versions plugin to update their artifact versions. Step will run if maven-versions-plugin is set to true.
- Support for the [Throttle Plugin](#) is added to JJB jobs so static build servers can restrict the number of concurrent JJB jobs ran at the same time.

This must be explicitly enabled by setting *throttle-enabled* on the jobs.

Bug Fixes

- Adapt maven path search for files and dirs. The “-f” maven param can specify both a directory, in which case it will look for “pom.xml” in the directory, or a specific file. The original version of this search was only compatible with directories that contain a pom.xml file.
- Update the If-maven-central macro documentation and example templates with the missing required params.
- Fix JAVA_HOME for openjdk11 on CentOS 7 to use the OpenJDK version installed in /usr/lib/jvm/java-11-openjdk.
- The JJB Deploy Job is configured to trigger only if the Gerrit comment starts with the *jjb-deploy* keyword.

Without the regex being optimized the job triggers on any occurrence of the *jjb-deploy* keyword in a Gerrit comment, which is a waste of infrastructure resources.

Example of a valid command in Gerrit comment that triggers the job:

```
jjb-deploy builder-jjb-*
```

Example of an invalid command in Gerrit comment that would *not* trigger the job:

```
Update the job. jjb-deploy builder-jjb-*
```

2.1.5 v0.33.0

New Features

- **jenkins-init-scripts** If the environmental variable ‘SWAP_SIZE’ is set when the ‘init.sh’ script is called, then a ‘SWAP_SIZE’ GB swap space will be configured. If ‘SWAP_SIZE’ is ‘0’ or is not a valid integer, then no swap space is configured. If it is unset then 1GB of swap will be configured. Previously the swap size was fixed at 1GB.
- **jenkins-init-scripts** If the work directory or volume (/w) already exists, the ownership will be recursively set to ‘jenkins:jenkins’. Previously only the top directory /w was owned by ‘jenkins:jenkins’.
- **If-sign-sign-dir** macros now supports a *sign-mode* parameter which allows jobs to choose to sign artifacts using either *parallel* mode or *serial* mode (default).

Upgrade Notes

- **If-sigul-sign-dir** users need to add a new parameter `sign-mode` to their job-templates setting either *parallel* or *serial* as the value, we recommend setting serial mode for this setting.

{project-name}-maven-stage-{stream}'s Sigul signer now defaults to *serial* mode instead of the previous *parallel* behaviour. To change this back to the previous behaviour pass the “sign-mode” parameter to the job template:

```
- project:
  name: parallel-sign
  jobs:
    - gerrit-maven-stage:
      sign-mode: parallel
```

2.1.6 v0.31.0

New Features

- New job-template **{project-name}-release-announce** for If-releng projects to automate release announcement emails.
- Add support for pushing Sonar results to SonarCloud. Refer to [Maven Sonar docs](#) for details.

Upgrade Notes

- Jobs using the **If-maven-stage** macro now need to update to the new usage. Preparation calls to **If-provide-maven-settings**, **If-infra-create-netrc**, and **If-provide-maven-settings-cleanup** are no longer necessary to prepare the **If-maven-stage** macro.

Usage:

```
- lf-maven-stage:
  mvn-global-settings: 'global-settings'
  mvn-settings: 'settings'
  mvn-staging-id: 'staging profile id'
```

2.1.7 v0.30.0

New Features

- Packer merge jobs now include the image name in the Jenkins build description.

Bug Fixes

- Extend `${JOB_NAME}` to include `{java-version}` parameter to support jobs to build with multiple versions of `openjdk{8,11}`.
- Modified `lf-maven-jobs.yaml` sonar cron entry to `'{obj:cron}'` to pass value from custom user config file.

2.1.8 v0.29.0

New Features

- Add a puppet-verify job to lf-ci-jobs. This job will perform Puppet linting on the specified repository.

```
- project:
  name: lf-infra-puppet-mymodule
  project-name: lf-infra-puppet
  project: puppet/modules/mymodule
  jobs:
    - gerrit-puppet-verify
```

Bug Fixes

- maven-fetch-metadata.sh was not respecting the “-f” (for file path) flag in MAVEN_PARAMS, causing lf-maven-merge jobs that utilize this flag to fail. It will now set a path based on this flag if it is present, or default to the current working directory.
- Check openjdk \$VERSION before setting \$JAVA_HOME. This enables jobs to pass “openjdk10” or “openjdk11” on CentOS 7 images to use the OpenJDK version installed in /opt.

2.1.9 v0.28.3

Bug Fixes

- Compress and upload all jjb-verify XML files to Nexus, to ease out the IOPs on cron jobs that manage the logs on Nexus and optimize job performance by ~8 mins. This is because the job generates around ~2.3K XML files (small files) which is uploaded to Nexus in every run of jjb-verify. Doing this is faster as compared to the Nexus Unpack plugin in the Nexus end unpacking the zip file we upload takes longer.

2.1.10 v0.28.1

Other Notes

- Update lftools version to **v0.19.0**.

2.1.11 v0.28.0

New Features

- New lf-stack-create macro allows job-templates to setup a OpenStack Heat stack, useful for spinning up CSIT labs to run integration tests against. Use with the lf-stack-delete macro.
- Concurrency for the gerrit-jjb-verify job can now be configured by setting the ‘build-concurrent’ parameter.
- New macro **lf-maven-central** is available to deploy artifacts to OSSRH staging for jobs that want to eventually deploy to Maven Central.

```
---
- job-template:
  name: lf-maven-central-macro-test

  #####
  # Default variables #
  #####

  mvn-central: true
  mvn-global-settings: ''
  mvn-settings: ''
  ossrh-profile-id: ''

  #####
  # Job configuration #
  #####

  builders:
    - lf-maven-central:
      mvn-central: '{mvn-central}'
      mvn-global-settings: '{mvn-global-settings}'
      mvn-settings: '{mvn-settings}'
      ossrh-profile-id: '{ossrh-profile-id}'
```

- The GERRIT_REFSPEC build parameter can now be used to trigger a test build from the Jenkins Sandbox system against a work in progress packer image patch from a GitHub Pull Request.

Upgrade Notes

- `lf-stack-delete` has been modified to be a companion macro to `lf-stack-create` in order to cleanup the stack at the end of a job run. It now includes a required parameter **openstack-cloud** to choose the `clouds . yaml` cloud configuration for the project. Existing users of this macro will need to update their job templates accordingly.
- Requires JJB 2.8.0 for the `jenkins-sandbox-cleanup` job to not fail.

Note: Despite the failure if JJB 2.8.0 is not available the job will successfully delete all jobs and views, the primary purpose of this job.

Bug Fixes

- [RELENG-1450](#) All view disappears on Jenkins Sandbox after views are deleted. The **All** view is now recreated after `delete-all` is run.

2.1.12 v0.27.0

New Features

- Add the ability to configure the location of JJB's cache directory for CI jobs.
- New view-templates `project-view`, `common-view`, and `csit-view` are available for projects to manage Jenkins views through code.

To use the `project-view` template in a project:

```
- project:
  name: aaa-view
  views:
    - project-view

  project-name: aaa
```

To use the `common-view` template in a project:

```
- project:
  name: daily-builds
  views:
    - common-view

  view-name: Periodic
  view-regex: '.*-periodic-.*'
```

To use the `csit-view` template in a project:

```
- project:
  name: csit
  views:
    - csit-view

  view-name: CSIT
  view-regex: '.*csit.*'

- project:
  name: csit-lnode
  views:
    - csit-view

  view-name: CSIT-lnode
  view-regex: '.*-csit-lnode-.*'
```

- Add support to maven-stage jobs to publish to Maven Central via OSSRH.

This is accomplished by adding these 2 new optional parameters to the job configuration.

```
- gerrit-maven-stage:
  mvn-central: true
  ossrh-profile-id: 7edbe315063867
```

- The **openstack-cron** job now has the ability to remove images older than a specified age (default: 30).
- The **openstack-cron** job now has the ability to remove orphaned servers.
- The **openstack-cron** job now has the ability to remove orphaned stacks.
- The **openstack-cron** job now has the ability to remove orphaned volumes.
- **lf-infra-gerrit-scm** and **lf-infra-github-scm** now require a `submodule-timeout` parameter to provide a timeout value (in minutes) for git fetch operations.
- All job-templates now provide an optional `submodule-timeout` parameter for git fetch operations, defaulting to 10 minutes.

Upgrade Notes

- Some LF projects are already using a `common-view` template in their local ci-management repo. This `common-view` is called `project-view` in `global-jjb` so rename all instances of `common-view` to `project-view` when upgrading and remove the local `common-view` `view-template` definition from ci-management.
- The **openstack-cron** job now requires a new parameter configured `jenkins-urls` in order to use the job.
- Any project using the **lf-infra-gerrit-scm** and **lf-infra-github-scm** macros in `global-jjb` should need to add a `submodule-timeout` value. It is recommended to default this value to 10 since it is the default used by the Jenkins Git Plugin.

Bug Fixes

- The `jenkins-init` scripts dir is now updated to reflect changes recommended for the v0.25.0 release. We unfortunately missed this critical piece in the v0.25.0 release.
- Specify `refspec` to be blank for SCM config on `github-maven-merge` job. Setting the `refspec` to `+refs/pull/*:refs/remotes/origin/pr/*` causes there to be no merge job triggered.
- New `detox` version requires `tox` ≥ 3.5 and < 4 . Instead of explicitly pulling `tox`, we are now implicitly pulling via `detox`. (<https://jira.opendaylight.org/browse/RELENG-136>)

Other Notes

- Iftools' `openstack` module will now be installed as part of pre-build.
- The **openstack-cron** job now runs every hour instead of daily. This is because stack cleanup should happen more regularly.

2.1.13 v0.26.0

New Features

- Add a new `nexus-iq-namespace` optional parameter to insert a namespace into Nexus IQ AppID. This is useful for shared Nexus IQ systems where projects might have concern about namespace collision.

Note: We recommend when using the namespace to add a trailing `-` to the value. Eg. `'odl-'`, this is to make the namespace look nice for example `"odl-aaa"` is the result of namespace `odl-`, and project name `aaa`.

- Add `lf-infra-publish-windows`. A publisher for use at the end of Windows based job-templates.

Bug Fixes

- Fix `packer-verify` job to correctly work with `clouds.yaml` config model implemented in `global-jjb` v0.25.0.

2.1.14 v0.25.1

Bug Fixes

- `jjb-cleanup.sh` may be merged with other shell scripts that `set -u` which causes Jenkins to fail when activating `virtualenvs`

2.1.15 v0.25.0

New Features

- Add support to the `packer-build` job to use `clouds.yaml` for openstack builder configuration rather than through the `cloud-env` file. This allows us to simplify the template configuration for openstack builders moving forward.
- New macro `lf-sigul-sign-dir` available to sign artifacts in a provided directory using Sigul.

Usage:

```
- lf-sigul-sign-dir:
  sign-dir: '$WORKSPACE/m2repo'
```

This macro also requires a boolean variable to `SIGN_ARTIFACTS` to be set to true to activate the macro. We recommend the job-template that uses this macro to define it in the job parameters section.

Example:

```
- bool:
  name: SIGN_ARTIFACTS
  default: '{sign-artifacts}'
  description: Use Sigul to sign artifacts.
```

- Add Sigul signing support to the `maven-staging` job. To activate Sigul signing make sure to set `sign-artifacts: true`. Example:

```
- project:
  name: abc
  jobs:
    - gerrit-maven-stage

  sign-artifacts: true
```

- Add `lf-stack-delete` macro to delete an openstack heat stack at the end of the job.

This macro requires a parameter defined in the job named `STACK_NAME` containing the name of the stack to delete.

- Add `lf-infra-wrappers-windows` to handle Windows specific wrapper configuration.
- Refactor `lf-infra-wrappers` to be for Linux systems and split out the non-linux specific components into a new `lf-infra-wrappers-common`. This change is seamless for current users of `lf-infra-wrappers`.

Upgrade Notes

- Upgrade to `global-jjb v0.24.6` before performing this upgrade. This ensures that `jjb-verify` job pulls in a regex fix that will allow it to verify the `v0.25.0` upgrade.

- Global JJB now has non-JJB YAML configuration and requires action on the ci-management repo when upgrading to this version of Global JJB to prevent JJB from picking up these YAMLs as config. Follow the instructions below BEFORE upgrading global-jjb:

```
cd <git-root>
git mv jjb/global-jjb global-jjb
mkdir jjb/global-jjb
ln -s ../../global-jjb/shell jjb/global-jjb/shell
ln -s ../../global-jjb/jjb jjb/global-jjb/jjb
git add jjb/global-jjb
git commit -sm "Prepare repo for global-jjb v0.25.0"
```

- Minimum packer version 1.2.5 is now required for the packer-build job.
- **If-infra-packer-build** macro now requires 2 new variables to be passed.
 1. **openstack:** Set to true if template is built using the openstack builder
 2. **openstack-cloud:** The clouds.yaml cloud to use when running packer build

Deprecation Notes

- lftools-install.sh is deprecated and will be removed in a future release. We recommend installing lftools via *pip install --user lftools* to install instead of using this script.

Bug Fixes

- Fix *pip install pip setuptools* which seems to fail against the Nexus 3 proxy. Run them as separate calls to make things happier.
- jjb-verify will now test on all changes in the jjb directory. The previous pattern was too specific and sometimes missed verifying patches that should be verified.
- Replace jjb-verify to test on all changes in the shell/* directory.
- Fix the lftools virtualenv workaround we had to put in place in the tox-verify job by using *pip install --user* for global tool installs.
- Fix jobs failing with UNSTABLE build due to install pip==18.0 missing. This change moves all the jobs to using If-infra-pre-build to install lftools via *--user* command.
- Use *python -m pip* to ensure that we are using the pip version that was installed rather than the OS wrapper version of pip.
- Fix package listing script in post-builder from causing UNSTABLE build due to difference in the two files being compared.
- Fix RTD job failing to find PBR install.

Other Notes

- Update lftools to ~ 0.17.1

2.2 Install

global-jjb requires configuration in 2 places; Jenkins and the *ci-management* repository.

2.2.1 Jenkins configuration

On the Jenkins side, we need to prep environment variables and plugins required by the jobs in global-jjb before we can start our first jobs.

Install Jenkins plugins

Install the following required Jenkins plugins and any optional ones as necessary by the project.

Required

- Config File Provider
- Description Setter
- Environment Injector Plugin
- Git plugin
- Post Build Script
- SSH Agent
- Workspace Cleanup

Required for Gerrit connected systems

- Gerrit Trigger

Required for GitHub connected systems

- GitHub plugin
- GitHub Pull Request Builder

Optional

- Mask Passwords
- MsgInject
- OpenStack Cloud
- Timestampper

Environment Variables

The *Jenkins Configuration Merge* job can manage environment variables job but we must first bootstrap them in Jenkins so that the job can run and take over.

Required:

```
GIT_URL=ssh://jenkins-$SILO@git.opendaylight.org:29418
JENKINS_HOSTNAME=jenkins092
NEXUS_URL=https://nexus.opendaylight.org
SILO=production
SONAR_URL=https://sonar.opendaylight.org
```

Gerrit:

```
GERRIT_URL=https://git.opendaylight.org/gerrit
```

GitHub:

```
GIT_URL=https://github.com
GIT_CLONE_URL=git@github.com:
```

Note: Use `GIT_CLONE_URL` for GitHub projects as this will be different from the URL used in the properties configuration.

Optional:

```
LOGS_SERVER=https://logs.opendaylight.org
```

Steps

1. Navigate to <https://jenkins.example.org/configure>
2. Configure the environment variables as described above
3. Configure the same environment variables in the *ci-management* repo

2.2.2 ci-management

ci-management is a git repository containing *JJB* configuration files for Jenkins Jobs. Deploying Global JJB here as a submodule allows us easy management to install, upgrade, and rollback changes via git tags. Install Global JJB as follows:

1. Install Global JJB

```
GLOBAL_JJB_VERSION=v0.1.0
git submodule add https://github.com/lfit/releng-global-jjb.git global-jjb
cd global-jjb
git checkout $GLOBAL_JJB_VERSION
cd ..

# Setup symlinks
mkdir -p jjb/global-jjb
ln -s ../../global-jjb/jenkins-init-scripts jjb/global-jjb/jenkins-init-scripts
ln -s ../../global-jjb/shell jjb/global-jjb/shell
ln -s ../../global-jjb/jjb jjb/global-jjb/jjb
git add jjb/global-jjb

git commit -sm "Install global-jjb $GLOBAL_JJB_VERSION"
```

Note: We are purposely using github for production deploys of global-jjb so that uptime of LF Gerrit does not affect projects using global-jjb. In a test environment we can use <https://gerrit.linuxfoundation.org/infra/releng/global-jjb> if desired.

2. Setup jjb/defaults.yaml

Create and configure the following parameters in the `jjb/defaults.yaml` file as described in the *defaults.yaml configuration docs <defaults.yaml>*.

Once configured commit the modifications:

```
git add jjb/defaults.yaml
git commit -sm "Setup defaults.yaml"
```

3. Push patches to Gerrit / GitHub using your favourite push method

At this point global-jjb installation is complete in the *ci-management* repo and is ready for use.

2.2.3 Deploy ci-jobs

The CI job group contains jobs that should deploy in all LF Jenkins infra. The minimal configuration to deploy the **{project-name}-ci-jobs** job group as defined in **lf-ci-jobs.yaml** is as follows:

jjb/ci-management/ci-management.yaml:

```
- project:
  name: ci-jobs

  jobs:
    - '{project-name}-ci-jobs'

  project: ci-management
  project-name: ci-management
  build-node: centos7-builder-2c-1g
```

Required parameters:

- project** The project repo as defined in source control.
- project-name** A custom name to call the job in Jenkins.
- build-node** The name of the builder to use when building (Jenkins label).

Optional parameters:

- branch** The git branch to build from. (default: master)
- jjb-version** The version of JJB to install in the build minion. (default: <defined by the global-jjb project>)

2.2.4 Deploy packer-jobs

The packer job group contains jobs to build custom minion images. The minimal configuration needed to deploy the packer jobs is as follows which deploys the **{project-name}-packer-jobs** job group as defined in **lf-ci-jobs.yaml**.

jjb/ci-management/packer.yaml:

```
- project:
  name: packer-builder-jobs

  jobs:
    - '{project-name}-packer-jobs'

  project: ci-management
  project-name: ci-management
  branch: master
  build-node: centos7-builder-2c-1g

  platforms:
    - centos
    - ubuntu-16.04

  templates: builder
```

(continues on next page)

(continued from previous page)

```
- project:
  name: packer-docker-jobs

  jobs:
    - '{project-name}-packer-jobs'

  project: ci-management
  project-name: ci-management
  branch: master
  build-node: centos7-builder-2c-1g

  templates: docker

  platforms:
    - centos
    - ubuntu-16.04
```

Required parameters:

project The project repo as defined in source control.

project-name A custom name to call the job in Jenkins.

build-node The name of the builder to use when building (Jenkins label).

platforms A list of supported platforms.

templates A list of templates to build. We recommend setting one template per `project` section so that we can control which platforms to build for specific templates.

Optional parameters:

branch The git branch to build from. (default: master)

packer-version The version of packer to install in the build minion, when packer is not available. (default: <defined by global-jjb>)

2.3 Configuration

2.3.1 defaults.yaml

This file lives in the ci-management repo typically under the path `jjb/defaults.yaml`. The purpose of this file is to store default variable values used by global-jjb templates.

Required

jenkins-ssh-credential The name of the Jenkins Credential to use for ssh connections. (ex: jenkins-ssh)

lftools-version Version of lftools to install. Can be a specific version like '0.6.1' or a [PEP-440 definition](#)
For example `<1.0.0` or `>=1.0.0,<2.0.0`.

mvn-site-id Maven Server ID from settings.xml containing the credentials to push to a Maven site repository.

mvn-staging-id Maven Server ID from settings.xml containing the credentials to push to a Maven staging repository.

Gerrit required parameters:

gerrit-server-name The name of the Gerrit Server as defined in Gerrit Trigger global configuration. (ex: Primary)

GitHub required parameters:

git-url Set this to the base URL of your GitHub repo. In general this should be <https://github.com>. If you are using GitHub Enterprise, or some other GitHub-style system, then it should be whatever your installation base URL is. This sets a job property that GitHub Pull Request Builder requires to work. Note that this is the web url to your project: (eg. <https://github.com/protect\TI\textdollarORG/protect\TI\textdollarPROJECT>)

git-clone-url This is the clone prefix used by GitHub jobs. Set this to either the same base url as **git-url**, or to '[git@github.com](https://github.com):' including the trailing ':'. Determined by your clone method (https or git).

github-org The name of the GitHub organization interpolated into the scm config.

github_pr_org The name of the GitHub organization. All members of this organization will be able to trigger jobs.

github_pr_whitelist List of GitHub members you wish to be able to trigger jobs.

github_pr_admin_list List of GitHub members that will have admin privileges on the jobs.

Example Gerrit Infra:

```
- defaults:
  name: global

  # lf-infra defaults
  jenkins-ssh-credential: jenkins-ssh
  gerrit-server-name: OpenDaylight
  lftools-version: '<1.0.0'
  mvn-site-id: opendaylight-site
  mvn-staging-id: opendaylight-staging
```

Example GitHub Infra:

```
- defaults:
  name: global

  # lf-infra defaults
  jenkins-ssh-credential: jenkins-ssh
  github-org: lfit
  github_pr_whitelist:
    - jpwku
    - tykeal
    - zxiiro
  github_pr_admin_list:
    - tykeal
  lftools-version: '<1.0.0'
  mvn-site-id: opendaylight-site
  mvn-staging-id: opendaylight-staging
```

2.3.2 Jenkins Files

global-jjb makes use of the Jenkins Config File Provider plugin to provide some default configurations for certain tools. This section details the files to define in Jenkins' **Manage Files** configuration.

npmrc

This file contains default npmrc configuration and lives in \$HOME/.npmrc. Documentation for npmrc is available via the [npm project](#).

Required This file MUST exist. An empty file is acceptable if a proxy is not available for the project.

type Custom file

Create a **Custom file** with contents:

```
registry = https://nexus.opendaylight.org/content/repositories/npmjs/
```

clouds-yaml

Needed by `openstack client` and `packer` to fetch OpenStack credentials and configuration. This file is OpenStack's `clouds.yaml` file.

Optional Needed for jobs that use `openstack client.packer` if building against OpenStack infra.

type Custom file

Create a **Custom file** with contents:

```
clouds:
vex:
  auth:
    project_name: OS_PROJECT_NAME
    username: OS_USERNAME
    password: OS_PASSWORD
    auth_url: 'https://auth.vexxhost.net/v3/'
    user_domain_name: Default
    project_domain_name: Default
    region_name: ca-ymq-1
```

Warning: If using packer 1.3.0 make sure that the clouds.yaml **profile** configuration is **NOT** configured. Using **profile** causes packer to look for another file called `clouds-public.yaml` for configuration.

pipconf

This file contains default configuration for the python-pip tool and lives in \$HOME/.config/pip/pip.conf. Documentation for pip.conf is available via the [pip project](#).

Required This file MUST exist. An empty file is acceptable if a proxy is not available for the project.

type Custom file

Create a **Custom file** with contents:

```
[global]
timeout = 60
index-url = https://nexus3.opendaylight.org/repository/PyPi/simple
```


jjbini

This file contains the Jenkins Job Builder configuration for *CI Jobs*.

Required This file MUST exist.

type Custom file

Create a **Custom file** with contents:

```

[job_builder]
ignore_cache=True
keep_descriptions=False
include_path=.:scripts:~/git/
recursive=True

[jenkins]
user=jenkins-jobbuilder
password=1234567890abcdef1234567890abcdef
url=https://jenkins.example.org
query_plugins_info=False

[production]
user=jenkins-jobbuilder
password=1234567890abcdef1234567890abcdef
url=https://jenkins.example.org
query_plugins_info=False

[sandbox]
user=jenkins-jobbuilder
password=1234567890abcdef1234567890abcdef
url=https://jenkins.example.org/sandbox
query_plugins_info=False

```

The last 2 sections are for the `jenkins-cfg` job use, they should match the `silo` names for the respective Jenkins systems, typically `production` and `sandbox`.

jenkins-log-archives-settings

See *lf-infra-ship-logs* for usage. If not archiving logs then keep this file with default settings, `global-jjb` needs the file to exist to function.

Requires a *credential* named 'logs' of type 'Username and Password' created in the Jenkins Credentials system.

1. Add Server Credentials
2. Set `ServerId` to `logs`
3. Set Credentials to the `logs` user created in the Credentials System

Required This file MUST exist if using log archiving.

type Maven settings.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0 http://maven.
↵apache.org/xsd/settings-1.0.0.xsd">
</settings>

```

Note: This example is the default boilerplate generated by Jenkins with the comments stripped out. We can also use the default generated by Jenkins without modifying it.

packer-cloud-env

Cloud environment configuration variables for Packer jobs. These can contain credentials and configuration for whichever clouds packer jobs are using.

Required This file **MUST** exist to use packer jobs.

type Json file

```
{
  "cloud_auth_url": "https://auth.vexxhost.net/v3/",
  "cloud_tenant": "TENANT_ID",
  "cloud_user": "CLOUD_USERNAME",
  "cloud_pass": "CLOUD_PASSWORD",
  "cloud_network": "CLOUD_NETWORK",
  "ssh_proxy_host": ""
}
```

2.3.3 Jenkins CI Jobs

jenkins-cfg-merge

This job manages Jenkins Global configuration. Refer to the [CI Documentation](#) for job configuration details.

2.3.4 Log Archiving

The logs account requires a Maven Settings file created called **jenkins-log-archives-settings** with a server ID of **logs** containing the credentials for the logs user in Nexus.

2.4 Best Practices

2.4.1 JJB YAML Layout

Note: While some of this applies to the Global JJB project other recommendations are generally useful to projects that might be defining JJB templates.

The Global JJB project is a useful example project to look at so we recommend referring to the Maven job definitions as an example as you read the documentation below:

<https://github.com/lf/releng-global-jjb/blob/master/jjb/lf-maven-jobs.yaml>

We recommend sectioning off the template into 3 general sections in order:

1. Job Groups (optional)
2. Common Functions

3. Job Template Definitions

In section 1) not all configurations need this so is an optional section. Job groups are useful in cases where there are jobs that are generally useful together. For example the OpenDaylight uses a lot of Merge and Verify job combinations so every new project will want both job types defined in their project.

In section 2) we want to define all common functions (anchors, aliases, macros) that are generally useful to all jobs in the file. This allows job template developers to look at the top of the file to see if there are useful functions already defined that they can reuse.

In section 3) we can declare our job definitions. In the Global JJB project we create Gerrit and GitHub versions of the jobs so the format we use here might look strange at first but is well layed out for code reuse if we need to define 2 or more versions of the same job template for different systems. We will define this in more detail in the next section.

Job Template Layout

1. Comment of Job Template Name
2. Macro containing build definition of the job a. Macro named after job b. Complete documentation of the job parameters c. Default parameters defined by the job d. Job configuration
3. job-template definition containing build triggers

In section 1) we need to declare a in large comment text to identify the job section.

In section 2) we declare the actual job definition. This is so that we have a single macro that we call in all the real job-template sections that is reusable and not duplicating any code. First we declare the macro as the job name. Then in 2.b) we provide the complete documentation of the job parameters this is so that we can link users of the job to this file and they can understand fully what options they can configure for this particular job. Then we define defaults for any parameters that are optional. The last section we define the job configuration which completes the macro.

In section 3) we declare the actual job-template. Because of all the preparations above job template definitions should be small and simple. It needs to define the scm and job triggers. The Global JJB project needs to support both Gerrit and GitHub versions of the same job so the job definitions there have 2 templates for each job defined.

2.4.2 Passing parameters to shell scripts

There are 2 ways to pass parameters into scripts:

1. JJB variables in the format {var}
2. Environment variables in the format \${VAR}

We recommend avoiding using method 1 (Pass JJB variables) into shell scripts and instead always use method 2 (Environment variables). This makes troubleshooting JJB errors easier and does not require escaping curly braces.

This method requires 3 steps:

1. Declare a parameter section or inject the variable as properties-content.
2. Invoke the shell script with *include-raw-escape* instead of *include-raw*.
3. Use the shell variable in shell script.

The benefit of this method is that parameters will always be at the top of the job page and when clicking the Build with Parameters button in Jenkins we can see the parameters before running the job. We can review the parameters retroactively by visiting the job parameters page `job/lastSuccessfulBuild/parameters/`. Injecting variables as properties-content makes the variable local to the specific macro, while declaring it as parameter makes the variable global.

Note: When a macro which invokes a shell script has no JJB parameters defined *!include-raw-escape* will insert extra curly braces, in such cases its recommended to use *!include-raw*.

2.4.3 Shell scripts

When developing shell scripts for JJB we recommend to create shell scripts as a separate file instead of inlining in YAML. This way we can ensure that the ShellCheck linter can catch potential issues with the scripts.

When writing the script itself, we recommend to redeclare all expected inputs at the top of the file using lowercase variable names before setting `set -u` after the inputs section. This ensures that all variables the script expects are at the top of the file which is useful for others to review and debug the script at a later stage. The `set -u` configuration before the start of the script code ensures that we catch any of these undeclared variables at the top of the file.

Example:

```
#!/bin/bash

# Inputs
tox_dir="${TOX_DIR:-$WORKSPACE}"
tox_envs="${TOX_ENVS:-}"

# Script start
set -eux -o pipefail

# ... script code goes here
```

2.4.4 Usage of config-file-provider

When using the config-file-provider plugin in Jenkins to provide a config file. We recommend using a macro so that we can configure the builder to remove the config file as a last step. This ensures that credentials do not exist on the system for longer than it needs to.

ship-logs example:

```
- builder:
  name: lf-ship-logs
  builders:
    - config-file-provider:
      files:
        - file-id: jenkins-log-archives-settings
          variable: SETTINGS_FILE
    - shell: !include-raw:
      - ../shell/logs-get-credentials.sh
    - shell: !include-raw:
      - ../shell/logs-deploy.sh
    - shell: !include-raw:
      - ../shell/logs-clear-credentials.sh
    - description-setter:
      regexp: '^Build logs: .*'
```

In this example the script logs-deploy requires a config file to authenticate with Nexus to push logs up. We declare a macro here so that we can ensure that we remove credentials from the system after the scripts complete running via the logs-clear-credentials.sh script. This script contains 3 basic steps:

1. Provide credentials via config-file-provider
2. Run logs-deploy.sh
3. Remove credentials provided by config-file-provider

2.4.5 Preserving Objects in Variable References

JJB has an option to preserve a data structure object when you want to pass it to a template. <https://docs.openstack.org/infra/jenkins-job-builder/definition.html#variable-references>

One thing that is not explicitly covered is the format of the variable name that you pass the object to. When you use the `{obj:key}` notation to preserve the original data structure object, it will not work if the variable name has a dash - in it. The standard that we follow, and recommend, is to use an underscore _ instead of a dash.

Example:

```
.. literalinclude:: _static/github-pr-trigger.example
```

In the above example note the use of underscores in `github_pr_whitelist`, `github_pr_admin_list`, and `github_included_regions`.

2.4.6 Using single quotes around variables

Its recommended to use single quotes around JJB variables '`{variable}-field`' during variable substitution or when using a variable in a string field, in other cases its recommended to drop the single quotes.

Example:

```
- builder:
  name: lf-user-logs
  builders:
    - inject:
      properties-content: |
        'HOME={user-home}'
    - build-file:
      settings: '{settings-file}'
      file-version: '{file-version}'
```

2.4.7 Variable expansion and Defaults

JJB has a concept called **Defaults** which is what JJB will replace a variable with if unset. Variables can configure dynamic content in **job-template** sections and allow certain options in these sections to be configurable.

The section that expands Defaults is **Job Templates** no other sections will expand a default. This documentation will explain how variables and defaults expansion works and which take precedence in JJB's variable expansion logic for the following configuration sections.

- macro
- job-template
- project
- default

Macro sections

Macro sections can contain variables but do **NOT** support default values getting filled in both at the macro definition level and at the defaults configuration level. [Macros](#) and [Job Templates](#) can use Macros but any variables defined in a Macro needs to pass a value or a new variable redefined in the [Job Template](#) if you want to pass on the configuration.

So for example if you have a macro that has a '{msg}' variable:

Example:

```
- builder:
  name: echo-msg
  builders:
    - shell: "echo {msg}"
```

Any downstream job-templates or macros that use this macro **MUST** pass in a *msg: Hello* definition or redefine the msg variable *msg: {msg}*.

Job Template sections

[Job Template](#) sections can use defaults in two ways.

1. Configure the message:

```
- job-template:
  name: echo-hello-world
  builders:
    - echo-msg:
      msg: 'Hello World'
```

2. Re-define '{msg}' variable

```
- job-template:
  name: echo-message
  builders:
    - echo-msg:
      msg: '{message}'
```

In option 2, we redefine the variable msg as *{message}* which a user of the job-template can now pass into the job their own custom message which is different than option 1, where we set a static message to pass in. We purposely redefined the **{msg}** to **{message}** here to show that you do not need to redefine it with the same name but we could have used the same name *{msg}* in the template too if we wanted to keep it the same.

Job Templates can also default a default variable for the variables it defines.

Example:

```
- job-template:
  name: echo-message
  message: 'Hello World'
  builders:
    - echo-msg:
      msg: '{message}'
```

This creates a job template variable called '{message}' which will default to "Hello World" if the user of the template does not explicitly pass in a message.

We should be aware of 2 Defaults concepts:

1. Default as defined in the [job-template](#)
2. Default as defined in a [defaults](#) configuration (typically defaults.yaml)

In this case there is a default ‘{message}’ set in the [job-template](#). JJB will use this default if the user (project section) does not declare a {message}.

If we do not declare a default in the [job-template](#) then JJB will fallback to checking the “defaults configuration”.

This means that the precedence of defaults is as follows:

1. User-provided
2. Job Template
3. Defaults.yaml

Project sections

[Project](#) sections define real jobs and pass in variables as necessary. Projects sections do NOT expand defaults.yaml. So you cannot configure a setting with {var} in here and expect defaults.yaml to fill it in for you. Define required configuration here.

Example:

```
- project
  name: foo
  jobs:
    - 'echo-message'
  message: 'I am foo'
```

Defaults sections

[Defaults](#) sections are the absolute last thing JJB checks if a variable is not configured in a job-template and user did not pass in the variable. JJB will fill in whatever is in the defaults configuration.

Variable expansion order of precedence seems to be:

1. job-group section definition
2. project section definition
3. job-template variable definition
4. defaults.yaml variable definition

Note: Defaults set variables in job-templates and are NOT used in Macros.

global-jjb should not provide job-group definitions and leave it up to users of global-jjb to create their own as a job-group as a variable defined in a job group the highest precedence. Global JJB should strive to be purely a job-template and macro library for downstream consumers.

Final thoughts

For any [Basic Job Configuration](#) for example “concurrent”, “jdk”, “node” etc. . . we cannot set defaults with the same name as JJB will not expand them. To use “node” we need to give the variable for that setting a different name such as “build-node” instead, if we want JJB to perform expansion for those settings. This issue affects top level job configuration, it does not appear to affect items below the top level such as calling a builder, wrapper or parameter.

2.5 Glossary

ciman Short for *ci-management*.

ci-management Refers to the SCM repository containing the *JJB* configuration files. In most LF Projects this is the repository named *ci-management*, but is *releng/builder* in the OpenDaylight project and *releng* in the OPNFV project.

JJB Short for Jenkins Job Builder (JJB) a tool used to convert YAML definitions into XML as a way to define Jenkins job configuration.

2.6 Appendix

2.6.1 ShellCheck

When using ShellCheck to lint *global-jjb* or any projects that include *global-jjb* as part of their project (common with *ci-management* repos) then we require version 0.4.x of ShellCheck installed on the build vms. This version introduces annotations used by shell scripts in this repo.

Job template code is in the *jjb/* directory but documentation is in the *docs/jjb/* directory of this project.

3.1 C/C++ Jobs

3.1.1 Job Templates

CMake Sonar

Sonar job which runs cmake && make then publishes to Sonar.

This job purposely runs on the master branch as there are configuration needed to support multi-branch.

Template Names

- {project-name}-cmake-sonar
- gerrit-cmake-sonar
- github-cmake-sonar

Comment Trigger run-sonar

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Configure in defaults.yaml)

sonar-scanner-version Version of sonar-scanner to install.

sonarcloud-project-key SonarCloud project key.

sonarcloud-project-organization SonarCloud project organization.

sonarcloud-api-token SonarCloud API Token.

Optional parameters

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-timeout Timeout in minutes before aborting build. (default: 60)

cmake-opts Parameters to pass to cmake. (default: ‘‘)

cron Cron schedule when to trigger the job. This parameter also supports multiline input via YAML pipe | character in cases where one may want to provide more than 1 cron timer. (default: ‘@daily’)

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

install-prefix CMAKE_INSTALL_PREFIX to use for install. (default: \$BUILD_DIR/output)

Listing 1: Example

```
install-prefix: |
  #!/bin/bash
  echo "Hello World."
```

make-opts Parameters to pass to make. (default: ‘‘)

pre-build Shell script to run before performing build. Useful for setting up dependencies. (default: ‘‘)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

gerrit_sonar_triggers Override Gerrit Triggers.

CMake Stage

Stage job which runs cmake && make && make install and then packages the project into a tar.xz tarball to produce a release candidate.

Template Names

- {project-name}-cmake-stage-{stream}
- gerrit-cmake-stage
- github-cmake-stage

Comment Trigger stage-release

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Configure in defaults.yaml)

nexus-group-id The Maven style Group ID for the namespace of the project in Nexus.

staging-profile-id The unique Nexus Staging Profile ID for the project. Contact your infra admin if you do not know it.

Optional parameters

branch Git branch to fetch for the build. (default: master)

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-dir Directory to build the project in. (default: \$WORKSPACE/target)

build-timeout Timeout in minutes before aborting build. (default: 60)

cmake-opts Parameters to pass to cmake. (default: '')

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

install-prefix CMAKE_INSTALL_PREFIX to use for install. (default: \$BUILD_DIR/output)

Listing 2: Example

```
install-prefix: |
    #!/bin/bash
    echo "Hello World."
```

make-opts Parameters to pass to make. (default: '')

pre-build Shell script to run before performing build. Useful for setting up dependencies. (default: '')

stream Keyword that to represent a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

version (default: '') Project version to stage release as. There are 2 methods for using this value:

1. Defined explicitly here.
2. Leave this value blank and set /tmp/artifact_version

Use method 2 in conjunction with 'pre-build' configuration to generate the artifact_version automatically from files in the project's repository. For example with pre-build script:

```
#!/bin/bash
MAJOR_VERSION="$(grep 'set(OCIO_VERSION_MAJOR' CMakeLists.txt
↪ | awk '{{print $NF}}' | awk -F')' '{{print $1}}' )"
MINOR_VERSION="$(grep 'set(OCIO_VERSION_MINOR' CMakeLists.txt
↪ | awk '{{print $NF}}' | awk -F')' '{{print $1}}' )"
PATCH_VERSION="$(grep 'set(OCIO_VERSION_PATCH' CMakeLists.txt
↪ | awk '{{print $NF}}' | awk -F')' '{{print $1}}' )"
echo "${MAJOR_VERSION}.${MINOR_VERSION}.${PATCH_VERSION}"
↪ > /tmp/artifact_version
```

CMake Verify

Verify job which runs cmake && make && make install to test a project build..

Template Names

- {project-name}-cmake-verify-{stream}
- gerrit-cmake-verify
- github-cmake-verify

Comment Trigger rechecklreverify

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Configure in defaults.yaml)

Optional parameters

branch Git branch to fetch for the build. (default: master)

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-dir Directory to build the project in. (default: \$WORKSPACE/target)

build-timeout Timeout in minutes before aborting build. (default: 60)

cmake-opts Parameters to pass to cmake. (default: '')

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

install-prefix CMAKE_INSTALL_PREFIX to use for install. (default: \$BUILD_DIR/output)

Listing 3: Example

```
install-prefix: |
  #!/bin/bash
  echo "Hello World."
```

make-opts Parameters to pass to make. (default: '')

pre-build Shell script to run before performing build. Useful for setting up dependencies. (default: '')

stream Keyword that to represent a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

gerrit_verify_triggers Override Gerrit Triggers.

gerrit_trigger_file_paths Override file paths which to filter which file modifications will trigger a build.

3.2 CI Jobs

3.2.1 Job Groups

Job groups are a great tool to configure categories of jobs together at the same time. Below the example are some starting point job-groups but we recommend creating your own to ensure that the jobs configured reflect the project's needs.

An example project:

```
- job-group:
  name: odl-maven-jobs

  jobs:
    - gerrit-maven-clm
    - gerrit-maven-merge
    - gerrit-maven-release
```

(continues on next page)

(continued from previous page)

```

- gerrit-maven-verify
- gerrit-maven-verify-dependencies:
  build-timeout: 180

mvn-version: mvn35

- project:
  name: aaa
  jobs:
    - odl-maven-jobs

```

In this example we are using the job-group to assign a list of common jobs to the aaa project. The job-group also hardcodes mvn-version to *mvn35* and build-timeout to *180* for all projects using this job-group.

A benefit of this method is for example disabling entire category of jobs by modifying the job-group, insert `disable-job: true` parameter against the jobs to disable.

Below is a list of CI job groups:

```

---
- job-group:
  name: '{project-name}-ci-jobs'

  jobs:
    - gerrit-jenkins-cfg-merge
    - gerrit-jenkins-cfg-verify
    - gerrit-jenkins-sandbox-cleanup
    - gerrit-jjb-deploy-job
    - gerrit-jjb-merge
    - gerrit-jjb-verify

- job-group:
  name: '{project-name}-github-ci-jobs'

  jobs:
    - github-jenkins-cfg-merge
    - github-jenkins-cfg-verify
    - github-jenkins-sandbox-cleanup
    - github-jjb-deploy-job
    - github-jjb-merge
    - github-jjb-verify

- job-group:
  name: '{project-name}-info-yaml-jobs'

  jobs:
    - gerrit-info-yaml-verify

- job-group:
  name: '{project-name}-github-info-yaml-jobs'

  jobs:
    - github-info-yaml-verify

- job-group:
  name: '{project-name}-packer-jobs'

```

(continues on next page)

(continued from previous page)

```
jobs:
- job-group:
  name: '{project-name}-github-packer-jobs'

  jobs:
    - github-packer-merge
    - github-packer-verify
```

3.2.2 Macros

lf-infra-jjb-parameters

Required Parameters

jjb-cache Location of Jenkins Job Builder (JJB) cache used for jjb jobs.

jjb-version Version of Jenkins Job Builder (JJB) to install and use in the jjb jobs.

lf-jenkins-cfg-clouds

Deploys Jenkins Cloud configuration read from the `jenkins-clouds` directory in ci-management repositories.

Note: Requires the `jjbini` file in Jenkins CFP to contain JJB 2.0 style config definitions for “production” and “sandbox” systems.

Required Parameters

jenkins-silos Space-separated list of Jenkins silos to update configuration for as defined in `~/config/jenkins_jobs/jenkins_jobs.ini`

lf-jenkins-cfg-global-vars

Manages the Global Jenkins variables. This macro will clear all exist macros in Jenkins and replaces them with the ones defined by the `ci-management/jenkins-config/global-vars-SILO.sh` script.

Note: Requires the `jjbini` file in Jenkins CFP to contain JJB 2.0 style config definitions for “production” and “sandbox” systems.

Required parameters

jenkins-silos Space-separated list of Jenkins silos to update configuration for as defined in `~/config/jenkins_jobs/jenkins_jobs.ini`

lf-infra-jjbini

Provides `jenkins_jobs.ini` configuration for Jenkins.

lf-infra-jjbini-sandbox

Provides jenkins_jobs.ini configuration for Jenkins sandbox.

Todo: This needs to be consolidated into lf-infra-jjbini when JJB 2.0 is available

lf-packer-common

Common packer configuration.

lf-packer-file-paths

Gerrit file-paths for packer jobs.

lf-packer-parameters

Parameters useful for packer related tasks.

Parameters

packer-version Version of packer to install / use. (shell: PACKER_VERSION)

lf-packer-verify-file-paths

Gerrit file-paths for packer verify jobs.

lf-puppet-parameters

Parameters useful for Puppet related tasks.

Parameters

puppet-lint-version Version of puppet-lint to install / use. (shell: PUPPET_LINT_VERSION)

3.2.3 Job Templates

Gerrit Branch Lock

Job submits a patch to lock or unlock a project's branch.

Template Names

- {project-name}-gerrit-branch-lock-{stream}
- gerrit-branch-lock

Comment Trigger

- lock branch
- unlock branch

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally should be configured in defaults.yaml)

Optional parameters

branch Git branch to build against. (default: master)

git-url URL to clone project from. (default: \$GIT_URL/\$GERRIT_PROJECT)

stream Keyword that can be used to represent a release code-name. Often the same as the branch. (default: master)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

gerrit_merge_triggers Override Gerrit Triggers.

Jenkins Configuration Merge

Jenkins job to manage Global Jenkins configuration.

Note: Requires the jjbini file in Jenkins CFP to contain JJB 2.0 style config definitions for “production” and “sandbox” systems.

Template names

- {project-name}-jenkins-cfg-merge
- gerrit-jenkins-cfg-merge
- github-jenkins-cfg-merge

Optional parameters

branch Git branch to build against. (default: master)

cron How often to run the job on a cron schedule. (default: @daily)

git-url URL to clone project from. (default: \$GIT_URL/\$GERRIT_PROJECT)

jenkins-silos Space separated list of Jenkins silos to update configuration for as defined in ~/.config/jenkins_jobs/jenkins_jobs.ini (default: production sandbox)

Typically this template is automatically pulled in by the “{project-name}-ci-jobs” job-group and does not need to be explicitly called if the job group is being used.

Minimal Example:

```
---
- project:
  name: jenkins-cfg-merge-minimal-test
  jobs:
    - 'gerrit-jenkins-cfg-merge'

  project-name: ci-management
```

Full Example:


```

---
- project:
  name: jenkins-cfg-merge-full-test
  jobs:
    - 'gerrit-jenkins-cfg-merge'

  project-name: builder
  jenkins-silos: releng sandbox

```

Global Environment Variables

Global Environment Variables are managed via the `jenkins-config/global-vars-SILO.sh` file in ci-management. Replace SILO with the name of the Jenkins silo the variable configuration is for.

The format for this file is `KEY=value` for example:

```

GERRIT_URL=https://git.opendaylight.org/gerrit
GIT_BASE=git://devvexx.opendaylight.org/mirror/$PROJECT
GIT_URL=git://devvexx.opendaylight.org/mirror
JENKINS_HOSTNAME=vex-yul-odl-jenkins-2
LOGS_SERVER=https://logs.opendaylight.org
NEXUS_URL=https://nexus.opendaylight.org
ODLNEXUSPROXY=https://nexus.opendaylight.org
SILO=sandbox
SONAR_URL=https://sonar.opendaylight.org

```

Cloud Configuration

This configuration requires the OpenStack Cloud plugin in Jenkins and is currently the only cloud plugin supported.

OpenStack Cloud plugin version supported:

- 2.30 - 2.34
- 2.35 - 2.37

Cloud configuration are managed via a directory structure in ci-management as follows:

- `jenkins-config/clouds/openstack/`
- `jenkins-config/clouds/openstack/cattle/cloud.cfg`
- `jenkins-config/clouds/openstack/cattle/centos7-builder-2c-2g.cfg`
- `jenkins-config/clouds/openstack/cattle/centos7-builder-4c-4g.cfg`
- `jenkins-config/clouds/openstack/cattle/centos7-docker-4c-4g.cfg`

The directory name inside of the “openstack” directory is used as the name of the cloud configuration. In this case “cattle” is being used as the cloud name.

The `cloud.cfg` file is a special file used to configure the main cloud configuration in the format `KEY=value`.

Cloud Parameters

CLOUD_URL API endpoint URL for Keystone. (default: “”)

CLOUD_IGNORE_SSL Ignore unverified SSL certificates. (default: false)

CLOUD_ZONE OpenStack region to use. (default: “”)

CLOUD_CREDENTIAL_ID Credential to use for authentication to OpenStack. (default: “os-cloud”)

INSTANCE_CAP Total number of instances the cloud will allow spin up. (default: null)

SANDBOX_CAP Total number of instances the cloud will allow to spin up. This applies to “sandbox” systems and overrides the INSTANCE_CAP setting. (default: null)

Template Parameters

Note: In the case of template definitions of a parameter below is not passed the one defined in default clouds will be inherited.

IMAGE_NAME The image name to use for this template. (required)

HARDWARE_ID OpenStack flavor to use. (required)

LABELS Labels to assign to the vm. (default: FILE_NAME)

NETWORK_ID OpenStack network to use. (default: “”)

USER_DATA_ID User Data to pass into the instance. (default: jenkins-init-script)

INSTANCE_CAP Total number of instances of this type that can be launched at one time. When defined in clouds.cfg it defines the total for the entire cloud. (default: null)

SANDBOX_CAP Total number of instances of this type that can be launched at one time. When defined in clouds.cfg it defines the total for the entire cloud. This applies to “sandbox” systems and overrides the INSTANCE_CAP setting. (default: null)

FLOATING_IP_POOL Floating ip pool to use. (default: “”)

SECURITY_GROUPS Security group to use. (default: “default”)

AVAILABILITY_ZONE OpenStack availability zone to use. (default: “”)

START_TIMEOUT Number of milliseconds to wait for the agent to be provisioned and connected. (default: 600000)

KEY_PAIR_NAME SSH Public Key Pair to use for authentication. (default: jenkins)

NUM_EXECUTORS Number of executors to enable for the instance. (default: 1)

JVM_OPTIONS JVM Options to pass to Java. (default: “”)

FS_ROOT File system root for the workspace. (default: “/w”)

RETENTION_TIME Number of minutes to wait for an idle slave to be used again before it’s removed. If set to -1, the slave will be kept forever. (default: 0)

CONNECTION_TYPE The connection type for Jenkins to connect to the build minion. Valid options: JNLP, SSH. (default: “SSH”)

For a live example see the OpenDaylight project jenkins-config directory. <https://github.com/opendaylight/releng-builder/tree/master/jenkins-config>

Troubleshooting

Cloud Configuration

The directory `groovy-inserts` contains the groovy script output that is used to push to Jenkins. In the event of a job failure this file can be inspected.

Jenkins Configuration Verify

Jenkins job to verify the Global Jenkins configuration.

Requires the `clouds.yaml` file to be setup on the Jenkins host.

Template names

- `{project-name}-jenkins-cfg-verify`
- `gerrit-jenkins-cfg-verify`
- `github-jenkins-cfg-verify`

Optional parameters

branch Git branch to build against. (default: `master`)

git-url URL to clone project from. (default: `$GIT_URL/$GERRIT_PROJECT`)

This job is not part of the “`{project-name}-ci-jobs`” group. It must be called explicitly.

Example:

```
---
- project:
  name: jenkins-cfg-verify
  jobs:
    - "gerrit-jenkins-cfg-verify"

  project-name: ci-management
```

Jenkins Sandbox Cleanup

Cleanup Jenkins Sandbox of jobs and views periodically.

Template names

- `{project-name}-jenkins-sandbox-cleanup`
- `gerrit-jenkins-sandbox-cleanup`
- `github-jenkins-sandbox-cleanup`

Comment Trigger `NONE`

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally should be configured in `defaults.yaml`)

Optional parameters

cron Schedule to run job. (default: `'0 8 * * 6'`)

JJB Deploy Job

Deploy jobs to jenkins-sandbox system via code review comment.

This job checks out the current code review patch and then runs a `jenkins-jobs update` to push a patch defined by the comment.

Template names

- {project-name}-jjb-deploy-job
- gerrit-jjb-deploy-job
- github-jjb-deploy-job

Comment Trigger jjb-deploy JOB_NAME

Note: The JJB Deploy Job is configured to trigger only if the Gerrit comment starts with the *jjb-deploy* keyword.

Example of a valid command in Gerrit comment that triggers the job:

```
jjb-deploy builder-jjb-*
```

Example of an invalid command in Gerrit comment that would *_not_* trigger the job:

```
Update the job. jjb-deploy builder-jjb-*
```

JOB_NAME can include the * wildcard character to push multiple jobs matching the pattern. For example `jjb-deploy builder-jjb-*` will push all `builder-jjb-*` jobs to the sandbox system.

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally should be configured in `defaults.yaml`)

Optional parameters

git-url URL clone project from. (default: `$GIT_URL/$PROJECT`)

gerrit_jjb_deploy_job_triggers Override Gerrit Triggers.

JJB Merge

Runs *jenkins-jobs update* to update production job configuration

Template Names

- {project-name}-jjb-merge
- gerrit-jjb-merge
- github-jjb-merge

Comment Trigger remerge

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally should be configured in `defaults.yaml`)

Optional parameters

branch Git branch to fetch for the build. (default: `master`)

build-days-to-keep Days to keep build logs in Jenkins. (default: `7`)

build-timeout Timeout in minutes before aborting build. (default: `10`)

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

jjb-cache JJB cache location. (default: \$HOME/.cache/jenkins_jobs)

jjb-workers Number of threads to run **update** with. Set to 0 by default which is equivalent to the number of available CPU cores. (default: 0)

jjb-version JJB version to install. (default: see job-template)

stream Keyword that can be used to represent a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

gerrit_merge_triggers Override Gerrit Triggers.

gerrit_trigger_file_paths Override file paths which can be used to filter which file modifications will trigger a build. (default defined by lf_jjb_common)

JJB Verify

Runs *jenkins-jobs test* to validate JJB syntax

Template Names

- {project-name}-jjb-verify
- gerrit-jjb-verify
- github-jjb-verify

Comment Trigger recheck|reverify

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally should be configured in defaults.yaml)

Optional parameters

branch Git branch to fetch for the build. (default: master)

build-concurrent Whether or not to allow this job to run multiple jobs simultaneously. (default: true)

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-timeout Timeout in minutes before aborting build. (default: 10)

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

jjb-cache JJB cache location. (default: \$HOME/.cache/jenkins_jobs)

jjb-version JJB version to install. (default: see job-template)

stream Keyword that can be used to represent a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

throttle_categories List of categories to throttle by.

- throttle-enabled** Whether or not to enable throttling on the job. (default: true)
- throttle-max-per-node** Max jobs to run on the same node. (default: 1)
- throttle-max-total** Max jobs to run across the entire project. - 0 means 'unlimited' (default: 0)
- throttle-option** Throttle by the project or by list of categories defined in the throttle plugin configuration. (options: 'project', 'category'; default: project)
- gerrit_verify_triggers** Override Gerrit Triggers.
- gerrit_trigger_file_paths** Override file paths which can be used to filter which file modifications will trigger a build. (default defined by lf_jjb_common)

JJB Verify Upstream Global JJB

Runs `jenkins-jobs test` to validate JJB syntax for upstream global-jjb patches. This job is useful to notify upstream that they may be breaking project level jobs.

Template Names

- {project-name}-jjb-verify-upstream-gjjb
- gerrit-jjb-verify-upstream-gjjb

Comment Trigger recheck|reverify

Required parameters

- build-node** The node to run build on.
- jenkins-ssh-credential** Credential to use for SSH. (Generally should be configured in defaults.yaml)

Optional parameters

- branch** Git branch to fetch for the build. (default: master)
- build-days-to-keep** Days to keep build logs in Jenkins. (default: 7)
- build-timeout** Timeout in minutes before aborting build. (default: 10)
- git-url** URL clone project from. (default: \$GIT_URL/\$PROJECT)
- jjb-cache** JJB cache location. (default: \$HOME/.cache/jenkins_jobs)
- jjb-version** JJB version to install. (default: see job-template)
- stream** Keyword that can be used to represent a release code-name. Often the same as the branch. (default: master)

Info YAML Verify

Info YAML Verify job validates that INFO.yaml file changes are kept isolated from other file changes. Verifies INFO.yaml files follow the schema defined in *global-jjb/info-schema*.

Template Names

- {project-name}-info-yaml-verify
- gerrit-info-yaml-verify
- github-info-yaml-verify

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally should be configured in defaults.yaml)

Optional parameters

branch Git branch to fetch for the build. (default: master)

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-timeout Timeout in minutes before aborting build. (default: 10)

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

stream Keyword that can be used to represent a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

gerrit_verify_triggers Override Gerrit Triggers.

License Checker

Job to scan projects for files missing license headers.

Template Names

- {project-name}-license-check
- gerrit-license-check
- github-license-check

Optional parameters

build-timeout Timeout in minutes before aborting build. (default: 15)

file-patterns Space-separated list of file patterns to scan. (default: *.go *.groovy *.java *.py *.sh)

spdx-disable Disable the SPDX-Identifier checker. (default: false)

lhc-version Version of LHC to use. (default: 0.2.0)

license-exclude-paths Comma-separated list of paths to exclude from the license checker. The paths used here will be matched using a contains rule so it is best to be as precise with the path as possible. For example a path of '/src/generated/' will be searched as '/src/generated/'. Example: org/opendaylight/yang/gen,protobuf/messages (default: '')

licenses-allowed Comma-separated list of allowed licenses. (default: Apache-2.0,EPL-1.0,MIT)

project-pattern The ANT based pattern for Gerrit Trigger to choose which projects to trigger job against. (default: '**')

OpenStack Cron

Cron job that runs regularly to perform periodic tasks against OpenStack.

This job requires a Config File Provider file named `clouds.yaml` available containing the credentials for the cloud.

Template Names

- `{project-name}-openstack-cron`
- `gerrit-openstack-cron`
- `github-openstack-cron`

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally should be configured in `defaults.yaml`)

jenkins-urls URLs to Jenkins systems to check for active builds.

Optional parameters

branch Git branch to fetch for the build. (default: `master`)

build-days-to-keep Days to keep build logs in Jenkins. (default: `7`)

build-timeout Timeout in minutes before aborting build. (default: `90`)

cron Time when the packer image should be rebuilt (default: `@hourly`)

git-url URL clone project from. (default: `$GIT_URL/$PROJECT`)

openstack-cloud `OS_CLOUD` setting to pass to openstack client. (default: `vex`)

openstack-image-cleanup Whether or not to run the image cleanup script. (default: `true`)

openstack-image-cleanup-age Age in days of image before marking it for removal. (default: `30`)

openstack-image-protect Whether or not to run the image protect script. (default: `true`)

openstack-server-cleanup Whether or not to run the server cleanup script. (default: `true`)

openstack-stack-cleanup Whether or not to run the stack cleanup script. (default: `true`)

openstack-volume-cleanup Whether or not to run the volume cleanup script. (default: `true`)

stream Keyword that can be used to represent a release code-name. Often the same as the branch. (default: `master`)

submodule-recursive Whether to checkout submodules recursively. (default: `true`)

submodule-timeout Timeout (in minutes) for checkout operation. (default: `10`)

Minimal Example:

```
---
- project:
  name: openstack-cron-minimal-test
  jobs:
    - 'gerrit-openstack-cron'

  project-name: ci-management
```

(continues on next page)

(continued from previous page)

```
jenkins-urls: >
  https://jenkins.example.org
  https://jenkins.example.org/sandbox
```

Full Example:

```
---
- project:
  name: openstack-cron-full-test
  jobs:
    - 'gerrit-openstack-cron'

  project-name: ciman-full

  jenkins-urls: >
    https://jenkins.example.org
    https://jenkins.example.org/sandbox
  openstack-cloud: example-cloud
  openstack-image-cleanup: false
  openstack-image-cleanup-age: 42
  openstack-image-protect: false
  openstack-server-cleanup: false
  openstack-stack-cleanup: false
  openstack-volume-cleanup: false
```

Packer Merge

Packer Merge job runs *packer build* to build system images in the cloud.

Template Names

- {project-name}-packer-merge-{platforms}-{templates}
- gerrit-packer-merge
- github-packer-merge

Comment Trigger remerge

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally should be configured in defaults.yaml)

mvn-settings The name of settings file containing credentials for the project.

platforms Platform or distribution to build. Typically json file found in the packer/vars directory. (Example: centos)

template System template to build. Typically shell script found in the packer/provision directory. (Example: java-builder)

Optional parameters

cron Time when the packer image should be rebuilt (default: @monthly)

branch Git branch to fetch for the build. (default: master)

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-timeout Timeout in minutes before aborting build. (default: 90)

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

openstack Packer template uses an OpenStack builder (default: true).

openstack-cloud Sets OS_CLOUD variable to the value of this parameter. (default: vex).

packer-cloud-settings Name of settings file containing credentials for the cloud that packer will build on. (default: packer-cloud-env)

packer-version Version of packer to install / use in build. (default: 1.0.2)

stream Keyword that can be used to represent a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

gerrit_verify_triggers Override Gerrit Triggers.

Test an in-progress patch

To test an in-progress patch from a GitHub Pull Request. Upload this job to the [Jenkins Sandbox](#). Then when manually building the job replace the GERRIT_REFSPEC parameter with the GitHub Pull Request number of the patch you would like to test.

Example GitHub:

GERRIT_REFSPEC: origin/pr/49/merge

Packer Verify

Packer Verify job runs *packer validate* to verify packer configuration.

Template Names

- {project-name}-packer-verify
- gerrit-packer-verify
- github-packer-verify

Comment Trigger rechecklreverify

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally should be configured in defaults.yaml)

mvn-settings The name of settings file containing credentials for the project.

Optional parameters

branch Git branch to fetch for the build. (default: master)

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-timeout Timeout in minutes before aborting build. (default: 10)

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

openstack Packer template uses an OpenStack builder (default: true).

openstack-cloud Sets OS_CLOUD variable to the value of this parameter. (default: vex).

packer-cloud-settings Name of settings file containing credentials for the cloud that packer will build on. (default: packer-cloud-env)

packer-version Version of packer to install / use in build. (default: 1.0.2)

stream Keyword that can be used to represent a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

gerrit_verify_triggers Override Gerrit Triggers.

gerrit_trigger_file_paths Override file paths which can be used to filter which file modifications will trigger a build.

Puppet Verify

Runs puppet-lint in the `puppet-dir` directory. puppet-lint runs recursively, the base directory is usually the best place to run from.

Template Names

- {project-name}-puppet-verify
- gerrit-puppet-verify
- github-puppet-verify

Comment Trigger `recheck|reverify`

Required Parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally set in defaults.yaml)

Optional Parameters

branch The branch to build against. (default: master)

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-timeout Timeout in minutes before aborting build. (default: 15)

gerrit_trigger_file_paths Override file paths which used to filter which file modifications will trigger a build. Refer to JJB documentation for “file-path” details. <https://docs.openstack.org/infra/jenkins-job-builder/triggers.html#triggers.gerrit>

git-url URL clone project from. (default: \$GIT_URL/\$GERRIT_PROJECT)

puppet-dir Directory containing the project’s puppet module(s) relative to the workspace. (default: “”)

puppet-lint-version Version of puppet-lint to use for testing. (default: 2.3.6)

stream Keyword representing a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

3.3 INFO VOTE JOB

Job counts the votes from the committers against a change to the INFO.yaml file

If needed, will also check for a majority of tsc voters (not yet implemented)

Auto-merges the change on a majority vote.

3.3.1 info-vote

Comment Trigger recheck|reverify|Vote

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally set in defaults.yaml)

3.4 Global Macros

3.4.1 Builders

comment-to-gerrit

This macro will post a comment to the gerrit patchset if the build creates a file named gerrit_comment.txt To use this macro add it to the list of builders.

lf-fetch-dependent-patches

Fetch all patches provided via comment trigger

This macro will fetch all patches provided via comment trigger and will create a list of projects from those patches via environment variable called `DEPENDENCY_BUILD_ORDER` which can be used if necessary to build projects in the specified order. The order is determined by first patch instance for a project in the patch list.

lf-license-check

Checks files for

Required parameters

file-patterns Space-separated list of file patterns to scan. For example: *.go *.groovy
*.java *.py *.sh

spdx-disable Disable the SPDX-Identifier checker.

lhc-version Version of LHC to use.

license-exclude-paths Comma-separated list of paths to exclude from the license checker.
The paths used here will be matched using a contains rule so it is best to be as precise with the path as possible. For example a path of `/src/generated/` will be searched as `/src/generated/`. Example: org/opendaylight/yang/gen,protobuf/messages

licenses-allowed Comma-separated list of allowed licenses. For example: Apache-2.0,EPL-1.0,MIT

If-infra-create-netrc

Create a ~/.netrc file from a Maven settings.xml

Required parameters

server-id The id of a server as defined in settings.xml.

Optional parameters

ALT_NEXUS_SERVER URL of custom nexus server. If set this will take precedence. Use this to point at nexus3.\$PROJECTDOMAIN for example.

If-infra-deploy-maven-file

Deploy files to a repository.

Required parameters

global-settings-file Global settings file to use.

group-id Group ID of the repository.

maven-repo-url URL of a Maven repository to upload to.

mvn-version Version of Maven to use.

repo-id Repository ID

settings-file Maven settings file to use.

upload-files-dir Path to directory containing one or more files

If-infra-docker-login

Login into a custom hosted docker registry and / or docker.io

The Jenkins system should have the following global variables defined

Environment variables

DOCKER_REGISTRY The DNS address of the registry (IP or FQDN) ex: nexus3.example.com (GLOBAL variable)

REGISTRY_PORTS Required if DOCKER_REGISTRY is set. Space separated list of the registry ports to login to. ex: 10001 10002 10003 10004 (GLOBAL variable)

DOCKERHUB_EMAIL If this variable is set then an attempt to login to DockerHub (docker.io) will be also made. It should be set to the email address for the credentials that will get looked up. Only _one_ credential will ever be found in the maven settings file for DockerHub. (GLOBAL variable)

If-infra-gpg-verify-git-signature

Verify gpg signature of the latest commit message in \$WORKSPACE. This command assumes that \$WORKSPACE is a git repo.

If-infra-pre-build

Macro that runs before all builders to prepare the system for job use.

If-infra-package-listing

Lists distro level packages.

If-infra-packer-build

Run *packer build* to build system images.

Required parameters

openstack Packer template uses an OpenStack builder (true/false).

openstack-cloud Sets OS_CLOUD variable to the value of this parameter.

packer-version Version of packer to use.

platform Build platform as found in the vars directory.

template Packer template to build as found in the templates directory.

If-infra-packer-validate

Run *packer validate* to verify packer configuration.

Required parameters

openstack Packer template uses an OpenStack builder (true/false).

openstack-cloud Sets OS_CLOUD variable to the value of this parameter.

packer-cloud-settings Cloud configuration file. Loaded on the build server as CLOUDENV environment variable.

packer-version Version of packer to use.

If-infra-push-gerrit-patch

Push a change through a Jenkins job to a Gerrit repository in an automated way using git-review.

Required parameters

gerrit-commit-message Commit message to assign.

gerrit-host Gerrit hostname.

gerrit-topic Gerrit topic.

gerrit-user Gerrit user-id used for submitting the change.

reviewers-email Reviewers email. Space-separated list of email addresses to CC on the patch.

project Gerrit project name.

lf-infra-ship-logs

Gather and deploy logs to a log server.

lf-infra-sysstat

Retrieves system stats.

lf-jacoco-nojava-workaround

Workaround for Jenkins not able to find Java in JaCoCo runs.

lf-maven-central

Publish artifacts to OSSRH (Maven Central) staging.

Requires that the project's settings.xml contains a ServerId 'ossrh' with the credentials for the project's OSSRH account.

This macro assumes the directory \$WORKSPACE/m2repo contains a Maven 2 repository which is to upload to OSSRH.

Required parameters

mvn-central Whether or not to upload to mvn-central. (true/false)

mvn-global-settings The name of the Maven global settings to use for Maven configuration. (default: global-settings)

mvn-settings The name of settings file containing credentials for the project.

ossrh-profile-id Nexus staging profile ID as provided by OSSRH.

```

---
- job-template:
  name: lf-maven-central-macro-test

  #####
  # Default variables #
  #####

  mvn-central: true
  mvn-global-settings: ''
  mvn-settings: ''
  ossrh-profile-id: ''

  #####
  # Job configuration #
  #####

  builders:
    - lf-maven-central:
      mvn-central: '{mvn-central}'
      mvn-global-settings: '{mvn-global-settings}'
      mvn-settings: '{mvn-settings}'
      ossrh-profile-id: '{ossrh-profile-id}'

```

lf-maven-install

Call maven-target builder with a goal of `-version` to force Jenkins to install the need provided version of Maven. This is needed for any shell scripts that want to use Maven.

Required parameters

mvn-version Version of Maven to install.

lf-pip-install

Call pip install to install packages into a virtualenv located in `/tmp/v/VEENV`

Note: The first package listed in `PIP_PACKAGES` is used as the VENV name.

lf-provide-maven-settings

Push a global settings and user settings maven files to the build node.

lf-provide-maven-settings-cleanup

Cleanup maven settings.xml configuration. This should be called at the end of any macros that calles the *lf-provide-maven-settings* macro.

lf-rtd-trigger-build

Script to trigger a build on <http://readthedocs.org>

lf-rtd-verify

ReadTheDocs verify script.

check-info-votes

Calls shell script to validate votes on a change to an INFO.yaml

lf-sigul-sign-dir

Use Sigul to sign a directory via `{sign-dir}`.

Requires `SIGUL_BRIDGE_IP` configured as a global envvar.

Required Parameters

sign-artifacts Whether or not to sign artifacts with Sigul.

sign-dir Directory to sign.

sign-mode serialparallel

lf-infra-provide-docker-cleanup

Forcibly removes all of the docker images.

3.4.2 Parameters

lf-clm-parameters

Provides the policy evaluation stage to run against Nexus IQ Server. Valid values include: 'build', 'stage-release', 'operate'.

lf-cmake-parameters

Provides parameters needed by CMake. Should be used by any jobs that need to call the `cmake` && `make` && `make install` pattern.

lf-infra-maven-parameters

Provides parameters needed by Maven. Should be used by any jobs that need to call the `mvn` cli.

lf-infra-openstack-parameters

Provides parameters needed by OpenStack client CLI. Use in jobs that need to call the `openstack` cli.

Required Parameters

os-cloud Configures `OS_CLOUD` envvar as used by `openstack` cli.

lf-infra-parameters

Standard parameters used in the LF CI environments. Gerrit variables are not used by GitHub projects, but defining them is not harmful. Should be used in every job template.

lf-infra-node-parameters

Provides parameters needed by NodeJS and NPM. Should be used by any jobs that need to run NodeJS or NPM.

lf-infra-tox-parameters

Provides parameters needed by `python-tox`. Should be used by any jobs that need to run `tox` <<https://tox.readthedocs.io>>.

3.4.3 Properties

lf-infra-properties

Configures the build-discarder plugin for Jenkins with the recommended lf-infra settings. Should be used in all job-templates.

3.4.4 Publishers

If-jacoco-report

Provides basic configuration for the JaCoCo plugin.

If-infra-publish

Provides basic If-infra recommended publisher configurations which should be used in all job templates. This primary objective of this trigger is to gather build logs and copy them to a log server.

If-infra-publish-windows

Windows publisher for use at the end of Windows job templates. Takes care of cleaning out the workspace at the end of a job.

3.4.5 SCM

If-infra-gerrit-scm

Basic SCM configuration for Gerrit based projects.

Required parameters

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

If-infra-github-scm

Basic SCM configuration for GitHub based projects.

On the *branch* variable you can assign *\$sha1* or *\$ghprbActualCommit* as the value. This will require that the job be triggered via the GHPRB plugin and not manually.

Required parameters

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

3.4.6 Wrappers

If-infra-wrappers-common

Provides If-infra recommended wrappers which should be used in every job-template. It's meant to be used by more specific wrappers below.

If-infra-wrappers

Provides If-infra recommended wrappers which should be used in every job-template that's run on Linux systems.

This wrapper requires that a managed file called *npmrc* exists in the Jenkins. The main use case here is to point to a npm proxy, on Nexus for example. The type of the file should be "Custom file". You can set various npmrc settings in it. Documentation on npm configuration can be found at <https://docs.npmjs.com/files/npmrc>. If you are not using npm then it is fine for the file to be empty.

Example npmrc:

```
registry=https://nexus3.onap.org/repository/npm.public/
```

lf-infra-wrappers-windows

Provides lf-infra recommended wrappers which should be used in every job-template that's run on Windows systems.

3.5 Maven Jobs

3.5.1 Job Groups

Job groups are a great tool to configure categories of jobs together at the same time. Below the example are some starting point job-groups but we recommend creating your own to ensure that the jobs configured reflect the project's needs.

An example project:

```
- job-group:
  name: odl-maven-jobs

  jobs:
    - gerrit-maven-clm
    - gerrit-maven-merge
    - gerrit-maven-release
    - gerrit-maven-verify
    - gerrit-maven-verify-dependencies:
      build-timeout: 180

  mvn-version: mvn35

- project:
  name: aaa
  jobs:
    - odl-maven-jobs
```

In this example we are using the job-group to assign a list of common jobs to the aaa project. The job-group also hardcodes mvn-version to *mvn35* and build-timeout to *180* for all projects using this job-group.

A benefit of this method is for example disabling entire category of jobs by modifying the job-group, insert `disable-job: true` parameter against the jobs to disable.

Below is a list of Maven job groups:

```
---
- job-group:
  name: '{project-name}-maven-jobs'

  # This job group contains all the recommended jobs that should be deployed
  # for any project ci.

  jobs:
    - gerrit-maven-clm
    - gerrit-maven-merge
```

(continues on next page)

(continued from previous page)

```
- gerrit-maven-stage
- gerrit-maven-verify
- gerrit-maven-verify-dependencies

- job-group:
  name: '{project-name}-github-maven-jobs'

  # This job group contains all the recommended jobs that should be deployed
  # for any project ci that is using github.

  jobs:
    - github-maven-clm
    - github-maven-merge
    - github-maven-stage
    - github-maven-verify

- job-group:
  name: '{project-name}-maven-javadoc-jobs'

  # This job group contains all the recommended jobs that should be deployed
  # for any project ci.

  jobs:
    - gerrit-maven-javadoc-publish
    - gerrit-maven-javadoc-verify

- job-group:
  name: '{project-name}-github-maven-javadoc-jobs'

  # This job group contains all the recommended jobs that should be deployed
  # for any project ci.

  jobs:
    - github-maven-javadoc-publish
    - github-maven-javadoc-verify
```

3.5.2 Macros

If-infra-maven-sonar

Runs Sonar against a Maven project.

Required Parameters

java-version Version of Java to execute Sonar with.

mvn-version Version of Maven to execute Sonar with.

mvn-settings Maven settings.xml file containing credentials to use.

If-infra-maven-sonarcloud

Runs Sonar against a Maven project and pushes results to SonarCloud.

Required Parameters

java-version Version of Java to execute Sonar with.

mvn-version Version of Maven to execute Sonar with.

mvn-settings Maven settings.xml file containing credentials to use.

sonarcloud-project-key SonarCloud project key.

sonarcloud-project-organization SonarCloud project organization.

sonarcloud-api-token SonarCloud API Token.

If-maven-build

Calls the maven build script to perform a maven build.

Required parameters

mvn-goals The maven goals to perform for the build. (default: clean deploy)

If-maven-common

Common Jenkins configuration for Maven jobs.

If-maven-deploy

Calls the maven deploy script to push artifacts to Nexus.

If-maven-versions-plugin

Conditionally calls Maven versions plugin to set, update and commit the maven *versions:set*.

Required Parameters

maven-versions-plugin Whether to call Maven versions plugin or not. (default: false)

mvn-version Version of Maven to execute Sonar with.

mvn-pom Location of pom.xml.

mvn-settings Maven settings.xml file containing credentials to use.

If-maven-stage

Calls the maven stage script to push artifacts to a Nexus staging repository.

Required Parameters

mvn-global-settings The name of the Maven global settings to use for Maven configuration.

mvn-settings The name of settings file containing credentials for the project.

If-update-java-alternatives

Setup Java alternatives for the Distro.

Required Parameters

java-version Version of Java to set as the default Java. Eg. openjdk8

3.5.3 Job Templates

Maven CLM

Produces a CLM scan of the code into Nexus IQ Server.

Template Names

- {project-name}-maven-clm-{stream}
- gerrit-maven-clm
- github-maven-clm

Comment Trigger run-clm

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally should be configured in defaults.yaml)

mvn-settings The name of settings file containing credentials for the project.

Optional parameters

branch The branch to build against. (default: master)

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-timeout Timeout in minutes before aborting build. (default: 60)

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

java-version Version of Java to use for the build. (default: openjdk8)

mvn-global-settings The name of the Maven global settings to use for Maven configuration. (default: global-settings)

mvn-opts Sets MAVEN_OPTS. (default: '')

mvn-params Additional mvn parameters to pass to the cli. (default: '')

mvn-version Version of maven to use. (default: mvn35)

nexus-iq-namespace Insert a namespace to project AppID for projects that share a Nexus IQ system to avoid project name collision. We recommend inserting a trailing - dash if using this parameter. For example 'odl-'. (default: '')

nexus-iq-stage Stage the policy evaluation will be run against on the Nexus IQ Server. (default: 'build')

stream Keyword that can be used to represent a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

gerrit_merge_triggers Override Gerrit Triggers.

Maven JavaDoc Publish

Produces and publishes javadocs for a Maven project.

Expects javadocs to be available in \$WORKSPACE/target/site/apidocs

Template Names

- {project-name}-maven-javadoc-publish-{stream}-{java-version}
- gerrit-maven-javadoc-publish
- github-maven-javadoc-publish

Comment Trigger remerge

Required parameters

build-node The node to run build on.

javadoc-path The path in Nexus to deploy javadoc to.

jenkins-ssh-credential Credential to use for SSH. (Generally should be configured in defaults.yaml)

mvn-settings The name of settings file containing credentials for the project.

mvn-site-id Maven Server ID from settings.xml to pull credentials from. (Note: This setting should be configured in defaults.yaml.)

Optional parameters

branch The branch to build against. (default: master)

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-timeout Timeout in minutes before aborting build. (default: 60)

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

java-version Version of Java to use for the build. (default: openjdk8)

mvn-global-settings The name of the Maven global settings to use for Maven configuration. (default: global-settings)

mvn-opts Sets MAVEN_OPTS. (default: “")

mvn-params Additional mvn parameters to pass to the cli. (default: “")

mvn-version Version of maven to use. (default: mvn35)

stream Keyword that can be used to represent a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

gerrit_merge_triggers Override Gerrit Triggers.

Maven JavaDoc Verify

Produces javadocs for a Maven project.

Expects javadocs to be available in \$WORKSPACE/target/site/apidocs

Template Names

- {project-name}-maven-javadoc-verify-{stream}-{java-version}
- gerrit-maven-javadoc-verify
- github-maven-javadoc-verify

Comment Trigger recheck|reverify

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally should be configured in defaults.yaml)

mvn-settings The name of settings file containing credentials for the project.

Optional parameters

branch The branch to build against. (default: master)

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-timeout Timeout in minutes before aborting build. (default: 60)

deploy-path The path in Nexus to deploy javadoc to. (default: \$PROJECT/\$STREAM)

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

java-version Version of Java to use for the build. (default: openjdk8)

mvn-global-settings The name of the Maven global settings to use for Maven configuration. (default: global-settings)

mvn-opts Sets MAVEN_OPTS. (default: “)

mvn-params Additional mvn parameters to pass to the cli. (default: “)

mvn-version Version of maven to use. (default: mvn35)

stream Keyword that can be used to represent a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

gerrit_verify_triggers Override Gerrit Triggers.

Maven Merge

Merge job which runs *mvn clean deploy* to build a project.

This job pushes files to Nexus using cURL instead of allowing the Maven deploy goal to push the upload. This is to get around the issue that Maven deploy does not properly support uploading files at the end of the build and instead pushes as it goes. There exists a *-Ddeploy-at-end* feature however it does not work with extensions.

This job uses the following strategy to deploy jobs to Nexus:

1. *wget -r* to fetch maven-metadata.xml from Nexus
2. *mvn deploy -DaltDeploymentRepository* to prepare files for upload
3. Removes untouched maven-metadata.xml files before upload
4. Use lftools (cURL) upload script to push artifacts to Nexus

Template Names

- {project-name}-maven-merge-{stream}
- gerrit-maven-merge
- github-maven-merge

Comment Trigger remerge

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally should be configured in defaults.yaml)

mvn-settings The name of settings file containing credentials for the project.

mvn-snapshot-id Maven Server ID from settings.xml to pull credentials from. (Note: This setting should be configured in defaults.yaml.)

nexus-snapshot-repo The repository id of the Nexus snapshot repo to deploy to.

Optional parameters

branch Git branch to fetch for the build. (default: master)

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-timeout Timeout in minutes before aborting build. (default: 60)

cron Cron schedule when to trigger the job. This parameter also supports multiline input via YAML pipe | character in cases where one may want to provide more than 1 cron timer. (default: 'H H * * 0' to run weekly)

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

java-version Version of Java to use for the build. (default: openjdk8)

mvn-global-settings The name of the Maven global settings to use for Maven configuration. (default: global-settings)

mvn-opts Sets MAVEN_OPTS. (default: '')

mvn-params Additional mvn parameters to pass to the cli. (default: '')

mvn-version Version of maven to use. (default: mvn35)

nexus-cut-dirs Number of directories to cut from file path for *wget -r*.

stream Keyword that can be used to represent a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

gerrit_merge_triggers Override Gerrit Triggers.

gerrit_trigger_file_paths Override file paths which can be used to filter which file modifications will trigger a build.

Maven Merge for Docker

Produces a snapshot docker image in a Nexus registry. Appropriate for Java projects that do not need to deploy any POM or JAR files.

Similar to Maven Merge as described above but logs in to Docker registries first and skips the If-maven-deploy builder. The project POM file should invoke a plugin to build and push a Docker image. The base image should be pulled from the registry in the environment variable `CONTAINER_PULL_REGISTRY`. The new image should be pushed to the registry in the environment variable `CONTAINER_PUSH_REGISTRY`.

Template Names

- {project-name}-maven-docker-merge-{stream}
- gerrit-maven-docker-merge
- github-maven-docker-merge

Required parameters

container-public-registry Docker registry source with base images.

container-snapshot-registry Docker registry target for the deploy action.

All other required and optional parameters are identical to the Maven Merge job described above.

Maven Stage

Produces a release candidate by creating a staging repo in Nexus.

The staging repo name is in the format PROJECT-NUMBER for example “aaa-1234”, “autorelease-2000”, “odlparent-1201”, etc...

This job runs a Maven build and deploys to \$WORKSPACE/m2repo directory. This directory is then used later to deploy to Nexus.

Template Names

- {project-name}-maven-stage-{stream}
- gerrit-maven-stage
- github-maven-stage

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally should be configured in defaults.yaml)

mvn-settings The name of settings file containing credentials for the project.

mvn-staging-id Maven Server ID from settings.xml to pull credentials from. (Note: This setting should be configured in defaults.yaml.)

staging-profile-id Profile ID of the project’s Nexus staging profile.

Optional parameters

archive-artifacts Artifacts to archive to the logs server (default: “”).

branch The branch to build against. (default: master)

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-timeout Timeout in minutes before aborting build. (default: 60)

cron Cron schedule when to trigger the job. This parameter also supports multiline input via YAML pipe | character in cases where one may want to provide more than 1 cron timer. (default: ‘’)

deploy-path The path in Nexus to deploy javadoc to. (default: \$PROJECT/\$STREAM)

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

java-version Version of Java to use for the build. (default: openjdk8)

mvn-central Set to ‘true’ to also stage to OSSRH. This is for projects that want to release to Maven Central. If set the parameter `ossrh-profile-id` also needs to be set. (default: false)

maven-versions-plugin Whether to call Maven versions plugin or not. (default: false)

mvn-global-settings The name of the Maven global settings to use for Maven configuration. (default: global-settings)

mvn-opts Sets MAVEN_OPTS. (default: ‘’)

mvn-params Additional mvn parameters to pass to the cli. (default: ‘’)

mvn-version Version of maven to use. (default: mvn35)

ossrh-profile-id Profile ID for project as provided by OSSRH. (default: ‘’)

sign-artifacts Sign artifacts with Sigul. (default: false)

stream Keyword that can be used to represent a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

gerrit_release_triggers Override Gerrit Triggers.

Maven Stage for Docker

Produces a release candidate docker image in a Nexus registry. Appropriate for Java projects that do not need to deploy any POM or JAR files.

Similar to Maven Stage as described above but logs in to Docker registries first and skips the If-maven-deploy builder. The project POM file should invoke a plugin to build and push a Docker image. The base image should be pulled from the registry in the environment variable `CONTAINER_PULL_REGISTRY`. The new image should be pushed to the registry in the environment variable `CONTAINER_PUSH_REGISTRY`.

Template Names

- {project-name}-maven-docker-stage-{stream}
- gerrit-maven-docker-stage
- github-maven-docker-stage

Required parameters

container-public-registry Docker registry source with base images.

container-staging-registry Docker registry target for the deploy action.

All other required and optional parameters are identical to the Maven Stage job described above.

Maven Sonar

Sonar job which runs mvn clean install then publishes to Sonar.

This job purposely only runs on the master branch as there are Additional configuration needed to support multiple branches and there's not much interest in that kind of support.

Template Names

- {project-name}-sonar
- gerrit-maven-sonar
- github-maven-sonar

Comment Trigger run-sonar

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally should be configured in defaults.yaml)

mvn-settings The name of settings file containing credentials for the project.

Optional parameters

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-timeout Timeout in minutes before aborting build. (default: 60)

cron Cron schedule when to trigger the job. This parameter also supports multiline input via YAML pipe | character in cases where one may want to provide more than 1 cron timer. (default: 'H H * * 6' to run weekly)

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

java-version Version of Java to use for the build. (default: openjdk8)

mvn-global-settings The name of the Maven global settings to use for Maven configuration. (default: global-settings)

mvn-opts Sets MAVEN_OPTS. (default: '')

mvn-params Additional mvn parameters to pass to the cli. (default: '')

mvn-version Version of maven to use. (default: mvn35)

sonar-mvn-goals Maven goals to run for sonar analysis. (default: sonar:sonar)

sonarcloud Whether or not to use SonarCloud `true|false`. (default: false)

sonarcloud-project-key SonarCloud project key. (default: '')

sonarcloud-project-organization SonarCloud project organization. (default: '')

sonarcloud-api-token SonarCloud API Token. (default: '')

stream Keyword that can be used to represent a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

gerrit_sonar_triggers Override Gerrit Triggers.

SonarCloud Example:

```
---
- project:
  name: example-sonarcloud
  jobs:
    - gerrit-maven-sonar

  project: 'sonarcloud'
  project-name: 'sonarcloud'
  branch: 'master'
  mvn-settings: 'sonarcloud-settings'
  mvn-opts: '-Xmx1024m'
  sonarcloud: true
  sonarcloud-project-key: KEY
  sonarcloud-project-organization: ORGANIZATION
  sonarcloud-api-token: TOKEN
```

Maven Verify

Verify job which runs mvn clean install to test a project build..

Template Names

- {project-name}-maven-verify-{stream}-{mvn-version}-{java-version}
- gerrit-maven-verify
- github-maven-verify

Comment Trigger recheck|reverify

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally should be configured in defaults.yaml)

mvn-settings The name of settings file containing credentials for the project.

Optional parameters

branch Git branch to fetch for the build. (default: master)

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-timeout Timeout in minutes before aborting build. (default: 60)

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

java-version Version of Java to use for the build. (default: openjdk8)

mvn-global-settings The name of the Maven global settings to use for Maven configuration. (default: global-settings)

mvn-opts Sets MAVEN_OPTS. (default: “)

mvn-params Additional mvn parameters to pass to the cli. (default: “)

mvn-version Version of maven to use. (default: mvn35)

stream Keyword that can be used to represent a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

gerrit_verify_triggers Override Gerrit Triggers.

gerrit_trigger_file_paths Override file paths which can be used to filter which file modifications will trigger a build.

Maven Verify for Docker

Similar to Maven Verify as described above but logs in to Docker registries first. The project POM file should invoke a plugin to build a Docker image. The base image should be pulled from the registry in the environment variable `CONTAINER_PULL_REGISTRY`.

Template Names

- {project-name}-maven-docker-verify-{stream}-{mvn-version}-{java-version}
- gerrit-maven-docker-verify
- github-maven-docker-verify

Required parameters

container-public-registry Docker registry source with base images.

All other required and optional parameters are identical to the Maven Verify job described above.

Maven Verify w/ Dependencies

Verify job which runs `mvn clean install` to test a project build /w deps

This job can be used to verify a patch in conjunction to all of the upstream patches it depends on. The user of this job can provide a list via comment trigger.

Template Names

- {project-name}-maven-verify-deps-{stream}-{mvn-version}-{java-version}
- gerrit-maven-verify-dependencies

Comment Trigger recheck: SPACE_SEPARATED_LIST_OF_PATCHES

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally should be configured in defaults.yaml)

mvn-settings The name of settings file containing credentials for the project.

Optional parameters

branch Git branch to fetch for the build. (default: master)

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-timeout Timeout in minutes before aborting build. (default: 60)

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

- java-version** Version of Java to use for the build. (default: openjdk8)
- mvn-global-settings** The name of the Maven global settings to use for Maven configuration. (default: global-settings)
- mvn-opts** Sets MAVEN_OPTS. (default: “)
- mvn-params** Additional mvn parameters to pass to the cli. (default: “)
- mvn-version** Version of maven to use. (default: mvn35)
- stream** Keyword that can be used to represent a release code-name. Often the same as the branch. (default: master)
- submodule-recursive** Whether to checkout submodules recursively. (default: true)
- submodule-timeout** Timeout (in minutes) for checkout operation. (default: 10)
- gerrit_verify_triggers** Override Gerrit Triggers.
- gerrit_trigger_file_paths** Override file paths which can be used to filter which file modifications will trigger a build.

3.6 NodeJS Jobs

3.6.1 Job Groups

Job groups are a great tool to configure categories of jobs together at the same time. Below the example are some starting point job-groups but we recommend creating your own to ensure that the jobs configured reflect the project's needs.

An example project:

```
- job-group:
  name: odl-maven-jobs

  jobs:
    - gerrit-maven-clm
    - gerrit-maven-merge
    - gerrit-maven-release
    - gerrit-maven-verify
    - gerrit-maven-verify-dependencies:
      build-timeout: 180

  mvn-version: mvn35

- project:
  name: aaa
  jobs:
    - odl-maven-jobs
```

In this example we are using the job-group to assign a list of common jobs to the aaa project. The job-group also hardcodes mvn-version to *mvn35* and build-timeout to *180* for all projects using this job-group.

A benefit of this method is for example disabling entire category of jobs by modifying the job-group, insert `disable-job: true` parameter against the jobs to disable.

Below is a list of Node job groups:

```
---
- job-group:
  name: '{project-name}-github-node-jobs'

  # Job group containing recommended jobs to deploy for a Node Project.

  node-version: 6.11.4

  jobs:
    - github-node-verify
- job-group:
  name: '{project-name}-node-jobs'

  # Job group containing recommended jobs to deploy for a Node Project.

  node-version: 6.11.4

  jobs:
    - gerrit-node-verify
```

3.6.2 Job Templates

Node Verify

Verify job for NodeJS projects

Template Names

- {project-name}-node-verify-{stream}
- gerrit-node-verify
- github-node-verify

Comment Trigger rechecklreverify

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally set in defaults.yaml)

node-version Version of NodeJS to install. Default defined in job-group.

Optional parameters

branch The branch to build against. (default: master)

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-timeout Timeout in minutes before aborting build. (default: 10)

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

node-dir Path to a NodeJS project to run node test against (default: “)

stream Keyword representing a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

gerrit_verify_triggers Override Gerrit Triggers.

gerrit_trigger_file_paths Override file paths filter which checks which file modifications will trigger a build.

3.7 OpenStack Heat

This section contains a series of macros for projects that need to spin up full test labs using HEAT scripts.

3.7.1 Job Setup

The 2 macros *lf-stack-create* & *lf-stack-delete* are companion macros and used together when constructing a job template that needs to spin up a full integration lab using Heat Orchestration Templates (HOT).

Example Usage:

```
- job-template:
  name: csit-test

  #####
  # Default variables #
  #####

  openstack-cloud: vex
  openstack-heat-template: csit-2-instance-type.yaml
  openstack-heat-template-dir: 'openstack-hot'

  odl_system_count: 1
  odl_system_flavor: odl-highcpu-4
  odl_system_image: ZZCI - CentOS 7 - builder - x86_64 - 20181010-215635.956
  tools_system_count: 1
  tools_system_flavor: odl-highcpu-2
  tools_system_image: ZZCI - Ubuntu 16.04 - mininet-ovs-25 - 20181029-223449.514

  #####
  # Job configuration #
  #####

  builders:
    - lf-infra-pre-build
    - lf-stack-create:
        openstack-cloud: '{openstack-cloud}'
        openstack-heat-template: '{openstack-heat-template}'
        openstack-heat-template-dir: '{openstack-heat-template-dir}'
        openstack-heat-parameters: |
          vm_0_count: '{odl_system_count}'
          vm_0_flavor: '{odl_system_flavor}'
          vm_0_image: '{odl_system_image}'
          vm_1_count: '{tools_system_count}'
          vm_1_flavor: '{tools_system_flavor}'
          vm_1_image: '{tools_system_image}'

  publishers:
```

(continues on next page)

(continued from previous page)

```
- lf-stack-delete:
  openstack-cloud: '{openstack-cloud}'
```

3.7.2 Macros

lf-stack-create

Creates an OpenStack stack as configured by the job. Name pattern of stack is \$SILO-\$JOB_NAME-\$BUILD_NUMBER.

Requires lf-infra-pre-build macro to run first to install the openstack and lftools packages.

Requires a Config File Provider configuration for clouds.yaml named clouds.yaml.

Required Parameters

openstack-cloud The OS_CLOUD variable to pass to OpenStack client. (Docs: <https://docs.openstack.org/python-openstackclient>)

openstack-heat-template Name of template file to use when running stack create.

openstack-heat-template-dir Directory in the ci-management repo containing the OpenStack heat templates.

Example:

```
---
- job-template:
  name: stack-create-test

  #####
  # Default variables #
  #####

  openstack-cloud: lf-cloud
  openstack-heat-template: csit-2-instance-type.yaml
  openstack-heat-template-dir: 'openstack-hot'

  odl_system_count: 1
  odl_system_flavor: odl-highcpu-4
  odl_system_image: ZZCI - CentOS 7 - builder - x86_64 - 20181010-215635.956
  tools_system_count: 2
  tools_system_flavor: odl-highcpu-2
  tools_system_image: ZZCI - Ubuntu 16.04 - mininet-ovs-25 - 20181029-223449.514

  #####
  # Job configuration #
  #####

  builders:
  - lf-infra-pre-build
  - lf-stack-create:
    openstack-cloud: '{openstack-cloud}'
    openstack-heat-template: '{openstack-heat-template}'
    openstack-heat-template-dir: '{openstack-heat-template-dir}'
    openstack-heat-parameters: |
      vm_0_count: '{odl_system_count}'
```

(continues on next page)

(continued from previous page)

```

vm_0_flavor: '{odl_system_flavor}'
vm_0_image: '{odl_system_image}'
vm_1_count: '{tools_system_count}'
vm_1_flavor: '{tools_system_flavor}'
vm_1_image: '{tools_system_image}'

```

lf-stack-delete

Deletes the stack associated with this job. Name pattern of stack is \$SILO-\$JOB_NAME-\$BUILD_NUMBER.

Requires lf-infra-pre-build macro to run first to install the openstack and lftools packages.

Requires a Config File Provider configuration for clouds.yaml named clouds-yaml.

Required Parameters

openstack-cloud The OS_CLOUD variable to pass to OpenStack client. (Docs: <https://docs.openstack.org/python-openstackclient>)

Example:

```

---
- job-template:
  name: stack-delete-test

  #####
  # Default variables #
  #####

  openstack-cloud: lf-cloud

  #####
  # Job configuration #
  #####

  publishers:
    - lf-stack-delete:
      openstack-cloud: '{openstack-cloud}'

```

3.8 Python Jobs

3.8.1 Job Groups

Job groups are a great tool to configure categories of jobs together at the same time. Below the example are some starting point job-groups but we recommend creating your own to ensure that the jobs configured reflect the project's needs.

An example project:

```

- job-group:
  name: odl-maven-jobs

  jobs:
    - gerit-maven-clm

```

(continues on next page)

(continued from previous page)

```
- gerrit-maven-merge
- gerrit-maven-release
- gerrit-maven-verify
- gerrit-maven-verify-dependencies:
    build-timeout: 180

mvn-version: mvn35

- project:
    name: aaa
    jobs:
        - odl-maven-jobs
```

In this example we are using the job-group to assign a list of common jobs to the aaa project. The job-group also hardcodes mvn-version to *mvn35* and build-timeout to *180* for all projects using this job-group.

A benefit of this method is for example disabling entire category of jobs by modifying the job-group, insert `disable-job: true` parameter against the jobs to disable.

Below is a list of Maven job groups:

```
---
- job-group:
    name: '{project-name}-python-jobs'

    # This job group contains all the recommended jobs that should be deployed
    # for any project ci that is using Gerrit.

    jobs:
        - gerrit-python-xc-clm
        - gerrit-tox-verify

- job-group:
    name: '{project-name}-github-python-jobs'

    # This job group contains all the recommended jobs that should be deployed
    # for any project ci that is using GitHub.

    jobs:
        - github-python-xc-clm
        - github-tox-verify
```

3.8.2 Macros

If-infra-clm-python

Run CLM scanning against a Python project.

Required Parameters

clm-project-name Project name in Nexus IQ to send results to.

If-infra-tox-install

Install Tox into a virtualenv.

Required Parameters

python-version Version of Python to install into the Tox virtualenv. Eg. python2 / python3

lf-infra-tox-sonar

Runs Sonar scanning against a Python project.

Required Parameters

java-version Version of Java to use to run Sonar.

mvn-version Version of Maven to use to run Sonar.

lf-tox-install

Runs a shell script that installs tox in a Python virtualenv.

Required Parameters

python-version Base Python version to use in the virtualenv. For example python2 or python3.

3.8.3 Job Templates

Python XC CLM

CLM scans for Python based repos. This job will call the Nexus IQ CLI directly to run the scans.

A new credential named “nexus-iq-xc-clm” needs to exist in the Jenkins credentials. The credential should contain the username and password to access Nexus IQ Server.

Template Names

- {project-name}-python-clm-{stream}
- gerrit-python-xc-clm
- github-python-xc-clm

Comment Trigger run-clm

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally should get configured in defaults.yaml)

Optional parameters

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

nexus-iq-cli-version Nexus IQ CLI package version to download and use. (default: 1.44.0-01)

nexus-iq-namespace Insert a namespace to project AppID for projects that share a Nexus IQ system to avoid project name collision. We recommend inserting a trailing - dash if using this parameter. For example ‘odl-’. (default: ‘’)

build-timeout Timeout in minutes before aborting build. (default: 60)

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

java-version Version of Java to use for the build. (default: openjdk8)

stream Keyword used to represent a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

gerrit_clm_triggers Override Gerrit Triggers.

gerrit_trigger_file_paths Override file paths which used to filter which file modifications will trigger a build. Refer to JJB documentation for “file-path” details. <https://docs.openstack.org/infra/jenkins-job-builder/triggers.html#triggers.gerrit>

Python Sonar with Tox

Sonar scans for Python based repos. This job will perform a tox call which runs the coverage command to gather tests results. These test results get published back into Sonar after running the Sonar goals.

To get the Sonar coverage results, tox.ini needs to exist and configured with coverage commands to run.

The coverage commands define the code that gets executed by the test suites. It does not guarantee that the code tests executed properly, but it will help pointing out the code that is not tested at all.

For example:

```
[testenv:py27]
commands =
    coverage run --module pytest --junitxml xunit-results.xml
    coverage xml --omit=".tox/py27/*","tests/*"
    coverage report --omit=".tox/py27/*","tests/*"
```

For more details refer to coverage and sonar documentation:

<https://coverage.readthedocs.io/>

<https://docs.sonarqube.org/display/PLUG/Python+Coverage+Results+Import>

Template Names

- {project-name}-tox-sonar
- gerrit-tox-sonar
- github-tox-sonar

Comment Trigger run-sonar

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally should get configured in defaults.yaml)

mvn-settings The name of settings file containing credentials for the project.

Optional parameters

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-timeout Timeout in minutes before aborting build. (default: 60)

cron Cron schedule when to trigger the job. This parameter also supports multiline input via YAML pipe | character in cases where one may want to provide more than 1 cron timer. (default: H 11 * * * to run once a day)

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

java-version Version of Java to use for the build. (default: openjdk8)

mvn-global-settings The name of the Maven global settings to use for Maven configuration. (default: global-settings)

mvn-version Version of maven to use. (default: mvn35)

stream Keyword used to represent a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

gerrit_sonar_triggers Override Gerrit Triggers.

gerrit_trigger_file_paths Override file paths which used to filter which file modifications will trigger a build. Refer to JJB documentation for “file-path” details. <https://docs.openstack.org/infra/jenkins-job-builder/triggers.html#triggers.gerrit>

Tox Verify

Tox runner to verify a project. This job is pyenv aware so if the image contains an installation of pyenv at /opt/pyenv it will pick it up and run Python tests with the appropriate Python versions. This job will set the following pyenv variables before running.

```
export PYENV_ROOT="/opt/pyenv"
export PATH="$PYENV_ROOT/bin:$PATH"
```

Template Names

- {project-name}-tox-verify-{stream}
- gerrit-tox-verify
- github-tox-verify

Comment Trigger recheck|reverify

Required Parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally set in defaults.yaml)

Optional Parameters

branch The branch to build against. (default: master)

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-timeout Timeout in minutes before aborting build. (default: 10)

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

python-version Version of Python to configure as a base in virtualenv. (default: python3)

stream Keyword representing a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

tox-dir Directory containing the project's tox.ini relative to the workspace. Empty works if tox.ini is at project root. (default: '')

tox-envs Tox environments to run. If blank run everything described in tox.ini. (default: '')

gerrit_trigger_file_paths Override file paths which used to filter which file modifications will trigger a build. Refer to JJB documentation for "file-path" details. <https://docs.openstack.org/infra/jenkins-job-builder/triggers.html#triggers.gerrit>

— Job Templates =====

3.9 Release Announce

Job for lf-releng projects to automate release announcement emails.

Template Names

- {project-name}-release-announce
- gerrit-release-announce

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Configured in defaults.yaml)

3.10 ReadTheDocs Jobs

3.10.1 Job Groups

Job groups are a great tool to configure categories of jobs together at the same time. Below the example are some starting point job-groups but we recommend creating your own to ensure that the jobs configured reflect the project's needs.

An example project:

```
- job-group:
  name: odl-maven-jobs

  jobs:
    - gerrit-maven-clm
    - gerrit-maven-merge
    - gerrit-maven-release
    - gerrit-maven-verify
    - gerrit-maven-verify-dependencies:
      build-timeout: 180

  mvn-version: mvn35
```

(continues on next page)

(continued from previous page)

```
- project:
  name: aaa
  jobs:
    - odl-maven-jobs
```

In this example we are using the job-group to assign a list of common jobs to the aaa project. The job-group also hardcodes `mvn-version` to `mvn35` and `build-timeout` to `180` for all projects using this job-group.

A benefit of this method is for example disabling entire category of jobs by modifying the job-group, insert `disable-job: true` parameter against the jobs to disable.

Below is a list of Maven job groups:

```
---
- job-group:
  name: '{project-name}-rtd-jobs'

  jobs:
    - gerrit-rtd-merge
    - gerrit-rtd-verify

- job-group:
  name: '{project-name}-github-rtd-jobs'

  jobs:
    - github-rtd-merge
    - github-rtd-verify
```

3.10.2 Macros

lf-rtd-common

RTD verify and merge jobs are the same except for their scm, trigger, and builders definition. This anchor is the common template.

3.10.3 Job Templates

ReadTheDocs Merge

Merge job which triggers a POST of the docs project to readthedocs.

To use this job first configure the Generic API incoming webhook in ReadTheDocs. To do that follow these steps:

1. Browse to <https://readthedocs.org/dashboard/PROJECT/integrations/>
2. Click on Generic API incoming webhook

Note: If not available click on Add integration and add the Generic API incoming webhook.

3. Copy the custom webhook URL, this is your `rtd-build-url`

For example: <https://readthedocs.org/api/v2/webhook/opendaylight/32321/>

4. Copy the token, this is your `rtd-token`

Template Names

- `{project-name}-rtd-merge-{stream}`
- `gerrit-rtd-merge`
- `github-rtd-merge`

Comment Trigger `remerge`

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally set in `defaults.yaml`)

rtd-build-url This is the generic webhook url from readthedocs.org. Refer to the above instructions to generate one. (Check Admin > Integrations > Generic API incoming webhook)

rtd-token The unique token for the project Generic webhook. Refer to the above instructions to generate one. (Check Admin > Integrations > Generic API incoming webhook)

Optional parameters

branch Git branch to fetch for the build. (default: `master`)

build-days-to-keep Days to keep build logs in Jenkins. (default: `7`)

build-timeout Timeout in minutes before aborting build. (default: `15`)

git-url base URL of git project. (default: <https://github.com>)

project-pattern Project to trigger build against. (default: `**`)

stream Keyword representing a release code-name. Often the same as the branch. (default: `master`)

submodule-recursive Whether to checkout submodules recursively. (default: `true`)

submodule-timeout Timeout (in minutes) for checkout operation. (default: `10`)

gerrit_merge_triggers Override Gerrit Triggers.

gerrit_trigger_file_paths Override file paths filter which checks which file modifications will trigger a build. **default:**

```
- compare-type: ANT
  pattern: '**/*.rst'
- compare-type: ANT
  pattern: '**/conf.py'
```

ReadTheDocs Verify

Verify job which runs a tox build of the docs project

Template Names

- `{project-name}-rtd-verify-{stream}`
- `gerrit-rtd-verify`
- `github-rtd-verify`

Comment Trigger recheck|reverify

Required Parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Generally set in defaults.yaml)

Optional Parameters

branch Git branch to fetch for the build. (default: master)

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-node The node to run build on.

build-timeout Timeout in minutes before aborting build. (default: 15)

doc-dir Directory where tox will place built docs. as defined in the tox.ini (default: docs/_build/html)

gerrit-skip-vote Skip voting for this job. (default: false)

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

project-pattern Project to trigger build against. (default: **)

stream Keyword representing a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

gerrit_verify_triggers Override Gerrit Triggers.

gerrit_trigger_file_paths Override file paths filter which checks which file modifications will trigger a build. **default:**

```
- compare-type: ANT
  pattern: '**/*.rst'
- compare-type: ANT
  pattern: '**/conf.py'
```

3.11 Jenkins Views

3.11.1 View Templates

JJB view-templates provides a way to manage Jenkins views through code. Using view-templates we can define common views configuration that are interesting to a project.

We recommend creating separate project sections for views apart from job configuration such that job configuration does not overlap with the view configuration.

Example Usage:

```
---
- project:
  name: project-view
  views:
    - common-view
```

(continues on next page)

(continued from previous page)

```
    project-name: project
- project:
  name: project-stream1
  jobs:
    - '{project-name}-{seq}'

  project: project
  project-name: project
  seq:
    - a
    - b
- project:
  name: project-stream2
  jobs:
    - '{project-name}-{seq}'

  project: project
  project-name: project
  seq:
    - x
    - y
- job-template:
  name: '{project-name}-{seq}'
```

Project view

Groups all jobs owned by a project under one view by capturing jobs with the prefix of `project-name`.

This view uses the following columns:

Columns

- status
- weather
- job
- last-success
- last-failure
- last-duration
- build-button
- jacoco
- find-bugs

Template Names

- {project-name}
- project-view

Required parameters

project-name The name of the project utilizing the view.

Optional parameters

view-filter-executors View filter executor. (default: false)

view-filter-queue View filter queue. (default: false)

view-recurse View recurse. (default: false)

Example:

```
---
- project:
  name: project-view-test
  views:
    - project-view

  project-name: project-view-test
```

Common view

Groups all jobs owned by a project under one view by capturing jobs with the prefix of `project-name`.

This view uses the following columns:

Columns

- status
- weather
- job
- last-success
- last-failure
- last-duration
- build-button
- jacoco
- find-bugs

Template Names

- {view-name}
- common-view

Required parameters

view-name The name of the view.

view-regex Regex to match the jobs.

Optional parameters

view-filter-executors View filter executor. (default: false)

view-filter-queue View filter queue. (default: false)

view-recurse View recurse. (default: false)

Example:

```
---
- project:
  name: common-view-test
  views:
    - common-view

  view-name: Daily
  view-regex: '.*-daily-.*'
```

CSIT view template

View template that loads columns useful for CSIT jobs.

This view uses the following columns:

Columns

- status
- weather
- job
- last-success
- last-failure
- last-duration
- build-button
- robot-list

Template Names

- {view-name}
- csit-view

Required parameters

view-name The name of the view.

view-regex Regex to match the jobs.

Optional parameters

view-description View description. (default: 'CSIT Jobs.')

view-filter-executors View filter executor. (default: false)

view-filter-queue View filter queue. (default: false)

view-recurse View recurse. (default: false)

Example:

```
---
- project:
  name: csit-view-test
  views:
    - csit-view
```

(continues on next page)

(continued from previous page)

```
view-name: CSIT-1node
view-regex: '.*csit-1node.*'
```

3.12 WhiteSource Jobs

3.12.1 Macros

lf-infra-wss-mvn-clean-install

Run maven clean install. Applicable to Maven based repos.

lf-infra-wss-unified-agent-scan

Run WhiteSource Unified Agent for a project.

3.12.2 Job Templates

WhiteSource Unified Agent scan

Trigger WhiteSource code scans using Unified Agent. For more details: <https://whitesource.atlassian.net/wiki/spaces/WD/pages/33718339/Unified+Agent>

The WhiteSource Unified Agent scanner runs using a configuration file: <https://s3.amazonaws.com/unified-agent/wss-unified-agent.config>

Template Names

- {project-name}-whitesource-scan-{stream}
- gerrit-whitesource-scan
- github-whitesource-scan

Comment Trigger run-whitesource

Required parameters

build-node The node to run build on.

jenkins-ssh-credential Credential to use for SSH. (Set in defaults.yaml)

wss-product-name Product to associate the WhiteSource report in the dashboard.

wss-unified-agent-config Path to wss-unified-agent.config.

Optional parameters

build-days-to-keep Days to keep build logs in Jenkins. (default: 7)

build-timeout Timeout in minutes before aborting build. (default: 60)

git-url URL clone project from. (default: \$GIT_URL/\$PROJECT)

java-version Version of Java to use for the build. (default: openjdk8)

mvn-clean-install Run maven clean install before the code scan. (default: false)

mvn-global-settings The name of the Maven global settings to use for Maven configuration. (default: global-settings)

mvn-version Version of maven to use. (default: mvn35)

pom Path of the pom.xml file.

stream Keyword used to represent a release code-name. Often the same as the branch. (default: master)

submodule-recursive Whether to checkout submodules recursively. (default: true)

submodule-timeout Timeout (in minutes) for checkout operation. (default: 10)

wss-unified-agent-version WhiteSource Unified Agent version package to download and use.

gerrit_trigger_file_paths Override file paths which used to filter which file modifications will trigger a build. Refer to JJB documentation for “file-path” details. <https://docs.openstack.org/infra/jenkins-job-builder/triggers.html#triggers.gerrit>

gerrit_wss_triggers Override Gerrit Triggers.

CHAPTER 4

Indices and tables

- `genindex`
- `modindex`
- `search`

C

ci-management, [28](#)
ciman, [28](#)

J

JJB, [28](#)