Pipeline Shared Groovy Libraries Plugin

Plugin Information
View Pipeline: Shared Groovy Libraries on the plugin site for more information.

Provides capability to extend pipeline scripts using shared libraries.

When you have multiple Pipeline jobs, you often want to share common parts of the Pipeline scripts to keep them DRY. A typical use case is that you have many projects that are built in the similar way.

This plugin adds that functionality by allowing you to create “shared library script” SCM repositories. It can be used in two modes:

- A legacy mode in which there is a single Git repository hosted by Jenkins itself, to which you may push changes;
- A more general mode in which you may define libraries hosted by any SCM in a location of your choice.

Comprehensive user documentation can be found in the Pipeline chapter of the User Handbook.

A component of Pipeline Plugin.

Source code README.md

Configure plugin via Groovy script

Either automatically upon Jenkins post-initialization or through Jenkins script console, example:

```
#!groovy

// imports
import hudson.scm.SCM
import jenkins.model.Jenkins
import jenkins.plugins.git.GitSCMSource
import org.jenkinsci.plugins.workflow.libs.*
import org.jenkinsci.plugins.workflow.libs.LibraryConfiguration
import org.jenkinsci.plugins.workflow.libs.SCMSourceRetriever

// parameters
def globalLibrariesParameters = [
  branch:               "master",
  credentialId:         "global-shared-library-key",
  implicit:             false,
  name:                 "Your Global Shared Library name here",
  repository:           "git@bitbucket.org:your-company/your-repo.git"
]

// define global library
GitSCMSource gitSCMSource = new GitSCMSource(
  "global-shared-library",
  globalLibrariesParameters.repository,
  globalLibrariesParameters.credentialId,
  "*",
  "",
  false
)

// define retriever
SCMSourceRetriever sCMSourceRetriever = new SCMSourceRetriever
(gitSCMSource)

// get Jenkins instance
Jenkins jenkins = Jenkins.getInstance()

// get Jenkins Global Libraries
def globalLibraries = jenkins.getDescriptor("org.jenkinsci.plugins.
workflow.libs.GlobalLibraries")

// define new library configuration
LibraryConfiguration libraryConfiguration = new LibraryConfiguration
(globalLibrariesParameters.name, sCMSourceRetriever)
libraryConfiguration.setDefaultVersion(globalLibrariesParameters.branch)
libraryConfiguration.setImplicit(globalLibrariesParameters.implicit)

// set new Jenkins Global Library
globalLibraries.get().setLibraries([libraryConfiguration])

// save current Jenkins state to disk
jenkins.save()
```

Changelog

2.13 (Feb 1, 2019)

- Fix: (PR 59) - Support for SCM retry count added in 2.12 did not apply to some SCM operations.
- Internal: (PR 57) - Avoid use of deprecated APIs.
- Internal: (PR 44, PR 56) - Add additional tests and update tests to run correctly on Windows

2.12 (Oct 2, 2018)

- Fix: JENKINS-40109 - Make compilation errors in shared libraries serializable so that the actual compilation error is reported instead of a `NotSerializableException` in some cases.
- Improvement: Implement support for SCM retry count.

2.11 (Sep 8, 2018)

- JENKINS-53485 - Fix a file leak introduced in version 2.10 of this plugin affecting all uses of the `libraryResource` step.

2.10 (Aug 21, 2018)

- **Important: As of this release, the plugin requires Java 8 and Jenkins 2.60.3 or newer.**
- JENKINS-52313 - Add an optional encoding argument to the `libraryResource` step. `Base64` is a supported encoding, and will cause the resource to be loaded as a Base64-encoded string, which is useful for copying binary resources such as images when combined with Pipeline: Basic Steps 2.8.1 or higher.

2.9 (Sept 13, 2017)

- JENKINS-41497 - allow excluding shared libraries from changelogs (and therefore from SCM polling as well) via global configuration option and/or `@Library(value="some-lib@master", changelog=false)`.

2.8 (Apr 24, 2017)

- Fixing some bugs affecting Windows-based masters (agent platform irrelevant):
  - improper handling of CRNL in `*.txt` global variable help files
  - incorrect display of class names in **Replay** when using class libraries
  - failure of class library access from `library` step depending on filesystem canonicalization

2.7 (Mar 03, 2017)

- JENKINS-39450 Added a `library` step as a dynamic alternative to `@Library` used since 2.3.

2.6 (Feb 10, 2016)

- JENKINS-40408 Race condition introduced in 2.5.

2.5 (Nov 21, 2016)

- Related to JENKINS-38517, checking out distinct libraries each into their own local workspaces, and improving parallelism in the case of concurrent builds.

2.4 (Oct 05, 2016)

- JENKINS-38550 The **Modern SCM** option should not be shown unless some matching plugin is actually installed.
- JENKINS-38712 Library configuration screens used deep horizontal indentation.
- JENKINS-38048 Obsolete query parameter caused a warning in the JavaScript console.

2.3 (Sep 07, 2016)

- JENKINS-31155 New system of external shared libraries.
- JENKINS-26192 Supporting Grape (the `@Grab` annotation) from global shared libraries (internal or external).

2.2 (Aug 09, 2016)

- JENKINS-34650 Global library code now runs without the Groovy sandbox, so may provide safe encapsulations of privileged operations such as Jenkins API accesses. (Pushes to the library always required Overall/RunScripts anyway.)
- JENKINS-34008 API allowing plugins to be notified of changes to the library.

2.1 (Jun 30, 2016)

- JENKINS-34517 Use of global variables from the shared library would result in errors when resuming a build.

2.0 (Apr 05, 2016)

- First release under per-plugin versioning scheme. See 1.x changelog for earlier releases.