Groovy Label Assignment plugin

Plugin Information
View Groovy Label Assignment on the plugin site for more information.

Older versions of this plugin may not be safe to use. Please review the following warnings before using an older version:

- Arbitrary code execution vulnerability

Provides "Groovy script to restrict where this project can be run" in job configuration pages.

What's this?

This plugin provides "Groovy script to restrict where this project can be run" to the job configuration page:

- The value returned from the script is treated as a label expression.
    - This label expression overrides "Restrict where this project can be run", and "Slaves"/"Label-expression" axes of multi-configuration projects.
    - A non-string values is converted into a string with `toString()`.
    - Returning null or empty string does not override existing label expressions.
- Following variables are binded to the Groovy script:
    - Parameters defined with "This build is parameterized".
    - Axes defined with a multi-configuration project.
    - Environment variables defined with plugins.
- A build does not start (trigger is ignored) in following cases:
    - No groovy script is configured even though Groovy Label Assignment is enabled in the job.
    - The groovy script contains syntax errors.
    - The groovy script throws an exception at the runtime.

Use cases

Use case 1

Think a following scenario:

- You have to build a project for multiple platforms: arm, win, linux
- There are following nodes:

Node	Label	arm	win	linux
win1	vs2010,armcc	O	O	X
win2	armcc	X	O	X
linux	gcc	X	X	O

You can manage this by using multi-configuration project as followings:

- Define a User-defined axis "platform": arm, win, linux
- Define a Slaves axis "slave": armcc, vs2010, gcc
- Define "Combination Filter" as following:

```
(platform == "arm" && slave=="armcc") || (platform == "win" && slave=="vs2010") || (platform == "linux"
&& slave=="gcc")
```

Groovy Label Assignment plugin provides following alternate solution:

- Define a User-defined axis "platform": arm, win, linux
- Define "Groovy script to restrict where this project can be run":

```
def labelMap = [
    arm: "armcc",
    win: "vs2010",
    linux: "gcc",
];
return labelMap.get(binding.getVariables().get("platform"));
```

Use case 2

Consider to create a job with which developers build a source tree.

- You want developers can build both a release build and a snapshot with that job. Developers select release or snapshot when they trigger a build.
- Release build must be built on nodes labeled "RELEASE" for releasing. Snapshot build must be built not on those nodes, but on other nodes.

You can create a satisfying job by using Groovy Label Assignment plugin:

- Parameterize the job.
- Define a Boolen Value parameter "release", which specifies the triggering build is for release.
- Define "Groovy script to restrict where this project can be run":

```
return (release == "true")?"RELEASE":"!RELEASE"
```

Limitations

- Some variables may not be properly binded:
  - Some type of parameters may be not properly binded.
  - Environment variables of some type of plugins may be not properly binded.
  - This is for Groovy Label Assignment plugin works when a build is going to be created, and is not created. Parameters and plugins that refers build information does not work properly.
- When Groovy Label Assignment plugin fails, a build is rejected silently. Failures happen in following cases. You can refer the system log to see why Groovy Label Assignment plugin failed.
  - Groovy script is not defined.
  - Groovy script contains syntax errors.
  - Groovy script failed at the runtime.
    - Especially in case referring non-binded variables. It often happens when running with multi-configuration project. In that case, you can access the variable safely as following:

      ```
      binding.getVariables().get("variable-name");
      ```

  - Returned value cannot be parsed as a label expression.

Screenshots

TODO

Issues

To report a bug or request an enhancement to this plugin please create a ticket in JIRA (you need to login or to sign up for an account). Also have a look on How to report an issue

- Bug report
- Request or propose an improvement of existing feature
- Request or propose a new feature

T	P	Key	Summary
No issues found			

How does this work?

This plugin works as following:

1. When a new build is triggerd, `GroovyLabelAssignmentQueueDecisionHandler` is called.
2. If `GroovyLabelAssignmentProperty` is assigned to the job, call it.
3. `EnvironmentContributingAction#buildEnvVars()` is called for retrieving variables to bind to the Groovy script.
    - Parameters are defined here.
4. Retrieve axes values configured to that job and bind to the Groovy script.
5. Run Groovy script.
6. Parse returned value as a label expression.
7. Assign it with `LabelAssignmentAction`.

Change Log

Version 1.2.0 (May 8, 2016)

- Now targets Jenkins 1.509 and later (was 1.466).
- **Groovy scripts run with Script Security Plugin** (JENKINS-27535)
    - Existing scripts are configured to run in the Groovy sandboxes.
    - You may have to approve some methods to allow run in the sandbox, or approve your scripts to allow run out of the sandbox.
    - See Script Security Plugin for details.

Version 1.1.1 (Sep 13, 2015)

- Fixed: fails to find nodes with a specified label when the label is once removed from all nodes (JENKINS-30135)

Version 1.1.0 (Mar 21, 2015)

- Expose current Jenkins job to the Groovy script as "currentJob" variable (JENKINS-27424)

Version 1.0.0 (Jun 05, 2013)

- Initial release.