SCM Filter Branch PR Plugin

About SCM Filter Branch PR Plugin

This plugin provides wildcard and regex filters for Pipeline Multibranch Plugin pipelines. It will match branches and pull requests destined for the matched branches.

Setup

After installing this plugin two new options will appear in Pipeline Multibranch Plugin jobs for configuring branch filters. In additional behaviors, click on Add and you'll see the following additional filters:

This filter will match branches and tags to be built. It will also match any pull request destination branch that matches a filtered branch to also be built as a pull request by the Pipeline Multibranch Plugin job.

Configuring tag building

This plugin supports filtering both for branches, pull requests destined for matched branches, and tags. This plugin is meant to satisfy a typical tag-based development workflow. In order to build tags, tag discovery must be configured in addition to specifying matching tags.

Here's a screenshot to filter tags with a regular expression (notice Discover tags trait).

Here's a screenshot to filter tags with wildcards (notice Discover tags trait).
What is a PR?

Matching and building a PR, is what is called a Peer Review build in generic terms of SCM. Depending on the platform you’re using it goes by different names. For example, GitHub (for Git) and Mercurial (another SCM) has Pull Requests. GitLab has Merge Requests. In essence, these are all the same thing. This plugin will match branches which are destined for the target branch (usually master branch in Git) and build them as part of an automated peer review.

Pipeline scripting for branch build vs PR build vs tag build

In multibranch pipelines, it’s not straight forward to tell the difference between tags, pull requests, and regular branch builds. Create a pipeline shared library and add the following pipeline variables to it.

- vars/isTagBuild.groovy
- vars/isPRBuild.groovy

The source of vars/isTagBuild.groovy:

```groovy
import hudson.model.Job
import jenkins.scm.api.mixin.TagSCMHead
import org.jenkinsci.plugins.workflow.multibranch.BranchJobProperty

@NonCPS
Boolean isTagBuild(Job build_parent) {
    build_parent.getProperty(BranchJobProperty).branch.head in TagSCMHead
}

Boolean call() {
    isTagBuild(currentBuild.rawBuild.parent)
}
```

The source of vars/isPRBuild.groovy:

```groovy
import hudson.model.Job
import jenkins.scm.api.mixin.ChangeRequestSCMHead
import org.jenkinsci.plugins.workflow.multibranch.BranchJobProperty

@NonCPS
Boolean isPRBuild(Job build_parent) {
    build_parent.getProperty(BranchJobProperty).branch.head in ChangeRequestSCMHead
}

Boolean call() {
    isPRBuild(currentBuild.rawBuild.parent)
}
```

Now scripted and declarative pipelines can be written which can detect what kind of build is occurring: branch build, tag build, PR build. Here’s some example usage in scripted pipeline.
Shell scripting for main build or PR build

`CHANGE_ID` environment variable is populated for pull requests. It is not set when the triggered build is not a peer review.

Shell scripting logic for branches vs pull requests

One can modify their build scripts to behave differently if a peer review is being built vs a matched branch. Here's an example in bash:

```
Example build script

if [ -n "${CHANGE_ID}" ]; then
  # do something because it's a pull request
else
  # not a pull request
fi
```

Additionally, it may be desired that a Jenkinsfile not run certain stages when evaluating branches vs peer reviews (such as deploys). This can be accomplished two ways.

Scripted pipeline syntax logic for branches vs pull requests

```
Pipeline script

if(env.CHANGE_ID) {
  // do something because it's a pull request
} else {
  // not a pull request
}
```

Declarative pipeline syntax logic for branches vs pull requests

Declarative pipelines provide a "when" condition. Therefore, similar logic can be applied. For example, deploy only when not a peer review and it is the master branch.
Declarative pipeline script

```java
pipeline {
    stages {
        stage('Example Deploy') {
            when {
                allOf {
                    environment name: 'CHANGE_ID', value: ''
                    branch 'master'
                }
            }
            steps {
                // not a pull request so do something
            }
        }
    }
}
```