Git Tag Message Plugin

Plugin Information
View Git Tag Message on the plugin site for more information.

Exports the name and message for a git tag as environment variables during a build.If the revision checked out during a build has a git tag associated with it, its name will be exported during the build as the `GIT_TAG_NAME` environment variable.
If a message was specified when creating the tag (e.g. via `git tag -m "..."`), then that message will also be exported during the build, as the `GIT_TAG_MESSAGE` environment variable.

If the revision has more than one tag associated with it, only the most recent tag will be taken into account. However, if your refspec includes "refs/tags" — i.e. builds are only triggered when certain tag names or patterns are matched — then the exact tag name that triggered the build will be used, even if it's not the most recent tag for this commit.

You can optionally choose the "use most recent tag" option, which will then export the tag name and message from the nearest tag in the history of the commit being built, if any.

Usage

Freestyle

Under the Source Control Management section in your job configuration, if you have selected "Git", then there should be a section labelled "Additional Behaviours".

Click "Add" and select "Export git tag and message as environment variables".

If you are also using the "Create a tag for every build" behaviour, use the drag-and-drop handles to ensure that it happens **after** the "Export git tag and message as environment variables" behaviour. Otherwise, the git tag message will likely end up being the auto-generated Jenkins tag message, e.g. "Jenkins Build #1".

Pipeline

Currently, this plugin does not work in Pipeline, as plugins can't contribute environment variables in the same way as in Freestyle.

However, until this plugin is fixed, you could use these Scripted Pipeline functions to achieve pretty much the same result:

```
// Example usage
node {
    git url: 'https://github.com/jenkinsci/git-tag-message-plugin'
    env.GIT_TAG_NAME = gitTagName()
    env.GIT_TAG_MESSAGE = gitTagMessage()
}

/** @return The tag name, or `null` if the current commit isn't a tag. */
String gitTagName() {
    commit = getCommit()
    if (commit) {
        desc = sh(script: "git describe --tags ${commit}", returnStdout: true)?.trim()
        if (isTag(desc)) {
            return desc
        }
    }
    return null
}

/** @return The tag message, or `null` if the current commit isn't a tag. */
String gitTagMessage() {
    name = gitTagName()
    msg = sh(script: "git tag -n10000 -l ${name}", returnStdout: true)?.trim()
    if (msg) {
        return msg.substring(name.size()+1, msg.size())
    }
    return null
}

String getCommit() {
    return sh(script: 'git rev-parse HEAD', returnStdout: true)?.trim()
}

@NonCPS
boolean isTag(String desc) {
    match = desc =~ /.+-[0-9]+-g[0-9A-Fa-f]{6,}$/
    result = !match
    match = null // prevent serialisation
    return result
}
```

Example

At iosphere, we use this in the process of automatically building and distributing beta versions of our mobile apps:

1. Commit some code
2. Write or generate release notes, e.g. `./generateChangelog.sh > /tmp/changes`
3. Create a tag, annotating it with the release notes, e.g. `git tag -F /tmp/changes beta/123`
4. Push the commit and tag to the remote repo

Jenkins will then, having received a git webhook notification:

1. Trigger a build of the relevant job, e.g. 'trails-beta-distribution'
2. Check out the committed code, i.e. the revision at tag beta/123
3. Export a GIT_TAG_MESSAGE variable — this will contain the release notes
4. Build the mobile app
5. Use the ${GIT_TAG_MESSAGE} value to fill out the release notes text field in a post-build step, e.g. the HockeyApp or Google Play Android Publisher plugin
6. Upload and deploy the app to the relevant service for distribution to users

Version history

Version 1.6.1 (December 24, 2017)

- Removed an unnecessary dependency on a Pipeline plugin, added in version 1.6
- Fixed integration tests so that they can run on Windows

Version 1.6 (December 20, 2017)

- Added the ability to use the most recent tag to the commit being built (JENKINS-32208)
    - Thanks to Arnaud Tamaillon
- Fixed the plugin to work with repos where git returns commit hash prefixes longer than seven characters

Version 1.5 (April 25, 2016)

- Fixed crash which could happen with a detached HEAD, i.e. no branch name was associated with a commit (JENKINS-34429)

Version 1.4 (June 21, 2015)

- The git tag name is now also exported as `GIT_TAG_NAME` (JENKINS-28705)
    - Thanks to Thomas Blitz

Version 1.3 (March 12, 2015)

- Fixed crash when no tag name could be determined (JENKINS-27383)
- The outcome of this plugin (i.e. whether a tag message was found and exported) is now written to the build log
- Increased Jenkins requirement to 1.565.1

Version 1.2 (February 1, 2015)

- Ensured that, if building from a tag, the message from that tag is used (rather than a newer tag pointing at the same commit)

Version 1.1 (October 31, 2014)

- Fixed crash when a detached HEAD or repo with no tags was used

Version 1.0 (October 22, 2014)

- Initial release