Committer notes

This is information for project committers.

Git repository:

The project code repository allowing git pushes is:

```
https://git-wip-us.apache.org/repos/asf/giraph.git
```

This is the repository that should be used by committers to push contributions back to trunk. More information on how to get started can be found here and the workflow is presented here here

How to update the website (http://giraph.apache.org):

Any commits that modify the src/site directory should be accompanied by an update to the website. svnpubsub is used to sync the website with the compiled website in your target/site directory.

The first step is to committing your changes to src/site in the Giraph tree. You must then build the site using mvn, which compiles into your local site directory. Finally, you must commit the results to separate SVN *site repository*. To explain in more detail:

First, in your giraph tree, generate the site via

```
mvn compile site:site
```

This will populate the target/site directory with fresh site content.

Notes:

- Don't use `-DskipTests` or it will appear we have no tests in the coverage report.
- You must include `compile` before `site:site` due to Apache munge complications - see this JIRA comment for more.

Next, in a separate directory, checkout the site repository:

```
svn co https://svn.apache.org/repos/asf/giraph/site
cd site
```

Note carefully the 'https': if you simply use 'http', you won't be able to commit back to the site repository.

Next, copy the site contents that you generated via the above mvn command into your site repository directory:

```
cp -r $GIRAPH_TREE/target/site/* .
```

Note that you must explicitly `svn add` any new files, otherwise they won't be included in the commit.

Commit with an informative comment, such as:

```
svn commit -m "regenerated from git-hash: git://git.apache.org/giraph.git@$GITHASH"
```

You should see output similar to:

```
Sending        apidocs/allclasses-frame.html
...
Sending        xref-test/overview-summary.html
Transmitting file data
...
Committed revision 1372694.
```

You can now do:

```
svn update
svn log | head
```

to confirm that your commit is the top (newest) commit.

If you forgot to `svn add` any new files, now you can `svn add` them and re-`svn commit`.

The latest site changes should be visible to the world within a few minutes of your commit.

How to roll a release

Broadly the process is defined here, but that page is pretty verbose.

1. Before rolling a release, be sure you've added your PGP key to your Apache account so you can sign the release. Paste the public key into the "OpenPGP Public Key Primary Fingerprint" text box. Your key should automatically be picked up and included in the giraph.asc file
2. Before rolling a release candidate, merge in any remaining changes or bugfixes.
3. Verify all tests are passing and documentation is up-to-date.
4. Create a new version in JIRA and assign any open bugs that are assigned to the current version to that one so they don't appear in the release notes.
5. Announce on dev you're working on rolling a new release so other committers can pitch in and don't assign new commits to that release during the process.
6. Verify that any new dependencies that have been introduced since the last release are properly accounted for in the NOTICE and all licensing issues have been resolved. ASF guidance is provided here
7. Update CHANGELOG to take into account the new release (create a new header for it). This is a minor change and doesn't require a patch.
8. Create a new branch for the release under the branches svn tree. Here is a good how-to.
9. When you're satisfied the branch represents what you'd like to release, tag that revision into the svn tags directory naming it release-VERSION-RC{starting at 0}. Here is a how-to for tagging.
10. Generate a release notes of the changes via JIRA via the Versions/{Version you're releasing}/Release Notes/<copy and paste the bottom section into an html file>, to be included with the artifacts
11. Create a tar.gz of the revision you've tagged and place that somewhere accessible (your people.apache.org/~you site is good choice). Additionally add a file with the md5 of the .tar.gz and sign the .tar.gz. Upload these four files (release notes, .tar.gz, .md5 and .asc) to your host.
12. Call a 72-hour vote on the release on the dev and public lists, pointing everyone to the artifacts. If needing to iterate on more release candidates, just keep working on the new branch, tagging each release candidate as you go. Once a release has been approved by the PMC, that tag can be renamed to the actual release name (since it's gone from being a release candidate to the actual RC).
13. ASF has instructions on distributing the new release artifact to the mirrors. It's a good idea to wait until the release has been mirrored before announcing the it on general@.