Integration Between Jenkins and GitHub (with the GitHub Jenkins Plugin)

Jenkins (and in fact any CI systems) works best when builds are triggered after each and every commit. That provides you, the developer, with the most instant feedback possible. That, in turn, means you get fix(es) to the problems you introduced before you leave the context of what you were working on.

There are two general techniques for detecting commits in any source control system:

* Polling - where you ask the source control system at regular intervals.
* Push - where the source control system tells you there has been a change.

The first technique requires no support from the source control system but puts a lot more load on both the CI system and the source control system. As well as that, it can only pick up changes as often as the polling frequency.

The second technique requires support, in both the source control system and the CI system, for the sending and receiving of notifications.

The Jenkins GitHub plugin provides support for receiving push notifications from Github post-commit hooks and using those notifications to trigger jobs within Jenkins. It can set-up the post-commit hooks in GitHub for you.

In addition, GitHub decorates the Jenkins “Changes” pages with links to the pretty GitHub commit, issues pages and adds a sidebar link to the corresponding GitHub project page.

Requirement for Plug-in Use

There are not too many requirements for using this plugin:

* Jenkins 1.400 or newer
* Jenkins Git Plugin version 1.1.12 or newer
* At least one GitHub hosted project to build ;-)

Step-by-Step Instructions on How to Use it

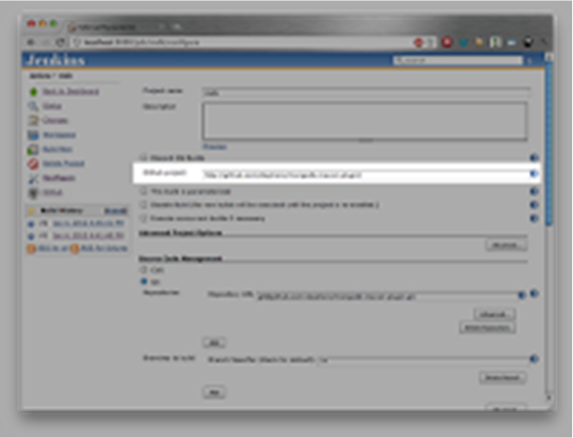
 Installation

[](http://3.bp.blogspot.com/-A5jxC7YqxeY/TwSAbFqlwrI/AAAAAAAAABE/bkEYoA1JtBw/s1600/Screen+shot+2012-01-04+at+16.33.22.png)

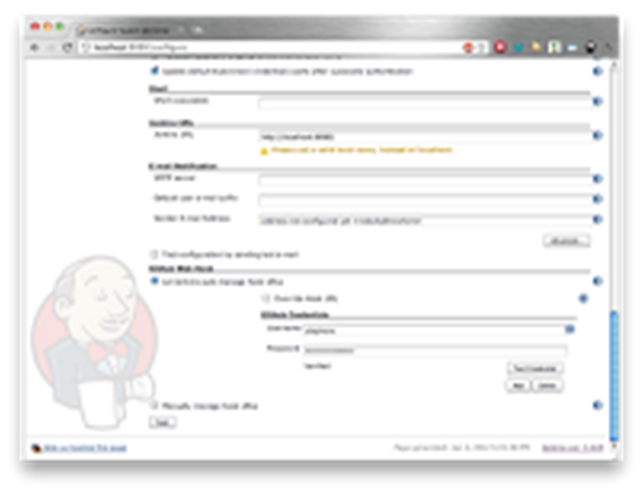
1. Go to your Jenkins instances root page.
2. If your Jenkins instance has security enabled, login as a user who has the Overall | Administer permission.
3. Select the Manage Jenkins link on the left-hand side of the screen.
4. Select the Manage Plugins link.
5. On the Available tab, select the Github Plugin and click the Download and Install button at the bottom of the page (if you do not got the Git Plugin installed, do not worry, Jenkins is smart enough to install/upgrade the Git plugin, where required).
6. Restart Jenkins once the plugins are downloaded (Note: users of Jenkins 1.442 or newer should be aware that the plugin currently requires a restart to function correctly).

 Configuration

If all you want to do is enable the hyperlinks in the Recent Changes for the build, it is just a case of providing the GitHub project URL in the Jenkins job configuration, e.g.

[](https://www.cloudbees.com/sites/default/files/blogger_importer/s1600/Screen%2Bshot%2B2012-01-04%2Bat%2B16.47.28.png)

1. Goto your Jenkins instance job.
2. Select the Configure link on the left hand side of the screen.
3. In the GitHub project field, enter the URL of the GitHub project. If your GitHub project’s git URL looks like: [git@github.com:username/project.git](mailto:git@github.com:username/project.git),  
     
   then the GitHub project should be: <http://github.com/username/project/or> if the project is private, you can get faster navigation with: <https://github.com/username/project/>

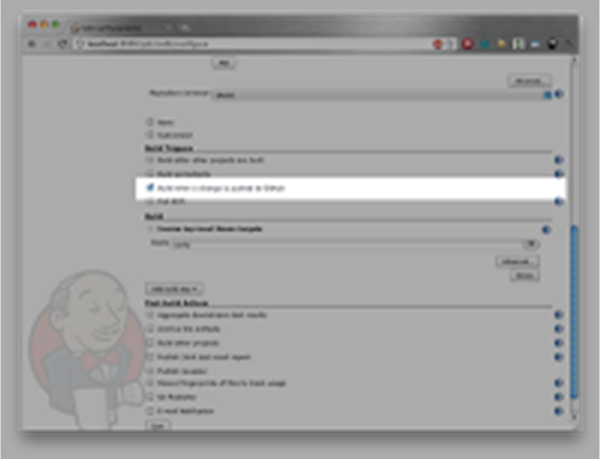
If you want to enable build triggering, you need to configure your Jenkins instance for receiving the push notifications from GitHub. There are two ways you can achieve this. The first way is to let Jenkins manage the Post-Receive URLs for you: [](https://www.cloudbees.com/sites/default/files/blogger_importer/s1600/Screen%2Bshot%2B2012-01-04%2Bat%2B17.05.25.png)

1. Go to your Jenkins instances root page.
2. If your Jenkins instance has security enabled, login as a user who has the Overall | Administer permission.
3. Select the Manage Jenkins link on the left hand side of the screen.
4. Select the Configure System link.
5. In the GitHub Web Hook section select the Let Jenkins auto-manage hook URLs option.
6. Ensure you have provided at least one username and password for connecting to GitHub (the password is required as GitHub does not expose an API for managing the Post-Receive URLs).

The second way is to manage the Post-Receive URLs yourself:

[](https://www.cloudbees.com/sites/default/files/blogger_importer/s1600/Screen%2Bshot%2B2012-01-04%2Bat%2B17.13.18.png)

1. Go to your Jenkins instances root page.
2. If your Jenkins instance has security enabled, login as a user who has the Overall | Administer permission.
3. Select the Manage Jenkins link on the left hand side of the screen.
4. Select the Configure System link.
5. In the GitHub Web Hook section select the Manually manage hook URLs option.
6. For each project that you want to have triggering builds, you need to open the Repository Administration screen on that GitHub project’s page.
7. Select the Service Hooks tab.
8. Select the Post-Receive URLs hook.
9. Add the URL, which will be the root URL of your Jenkins instance with /github-webhook appended.

[](https://www.cloudbees.com/sites/default/files/blogger_importer/s1600/Screen%2Bshot%2B2012-01-04%2Bat%2B16.57.07.png)

Once you have configured your Jenkins instance for receiving the push notifications, you can enable jobs being triggered via the push notifications:

1. Goto your Jenkins instance job.
2. Select the Configure link on the left hand side of the screen.
3. Select the Build when a change is pushed to GitHub checkbox and save the configuration.