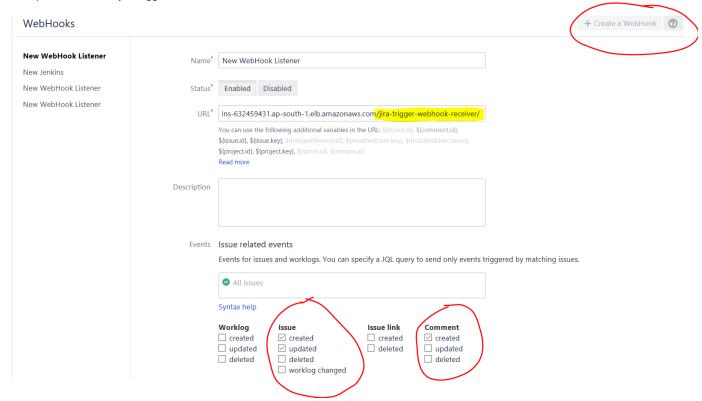
How To - Automated Issue Tracking with JIRA and Jenkins

Part 1

Jira side customization

Jira webhook integration:

Click on the "Cog wheel" on the right hand side, then click on "System" this time, then, scroll down (in the very bottom actually), and on the left hand side pane, look for "Webhooks". Click on that. Then, click on "+Create a Webhook" on the right hand side. Here, add following SUFFIX after your Jenkins server url: /jira-trigger-webhook-receiver/. So for example, your Jenkins server is running on http://10.0.0.1:8080, the webhook will be http://10.0.0.1:8080/jira-trigger-webhook-receiver/.

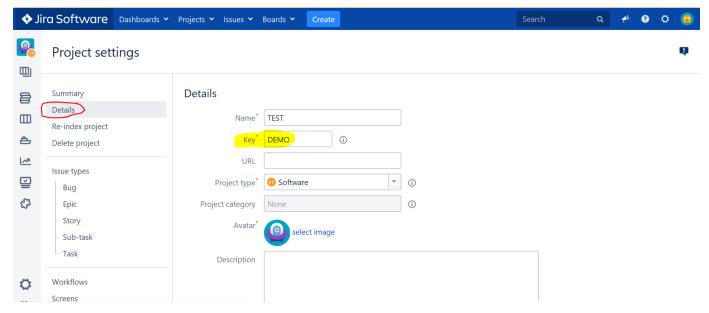


Additionally, tick the checkbox "created" and "updated" under Issue, and "created" under Comment in the Events section. What we are essentially configuring is that whenever new issue is created or updated, or a comment is created, Jenkins is triggered by Jira. So essentially, by updating our issues or by adding comments, we can control Jenkins pipelines. This is how Jira triggers Jenkins pipeline.

Finally, scroll down and click on CREATE to create this webhook. We can integrate multiple Jenkins pipeline using this feature, as we can create webhooks for multiple Jenkins server URLs.

Unfortunately, webhooks is just one part. You would also need TICKET PATTERN. As we know, in Jira, everything is an issue (including stories, epics etc.). So to keep track of it, Jira system generates a prefix before every issue. Think of it as a Primary Key value for Jira.

To get this Ticket pattern, click on the "Cog Wheel" on upper right side, then click on Project. Click on the project which you want to integrate. Then click on "Details" on the left side. The KEY is the TICKET PATTERN required to integrate with Jenkins.



You would also be needing the Jira server URL, Jira server name and the admin username/password. Except the admin username/password (which you should remember), server URL and name can be found in System Settings.

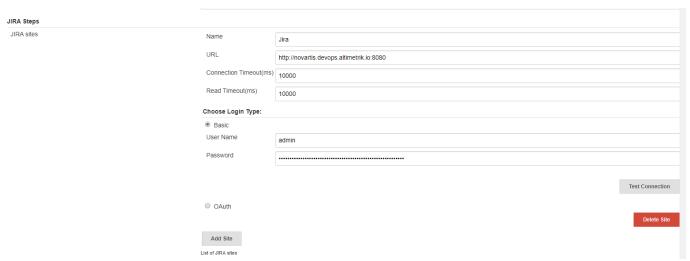
Plugins to be installed to achieve this

- Jira Plugin
- Generic Webhook Trigger Plugin
- Jira Trigger Plugin

Jenkins side customization

Ensure you have the below configuration done inside Manage Jenkins > Configure System.

Configuring Jira sites:

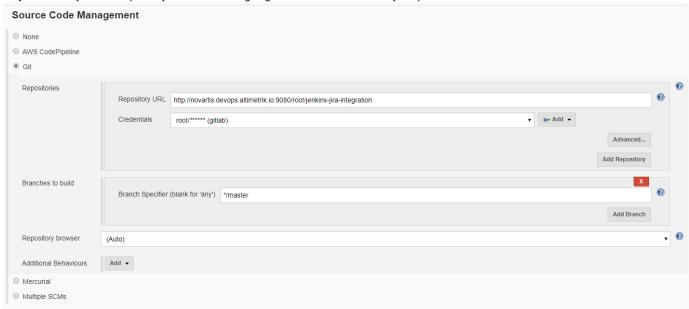


Jira trigger related configuration:

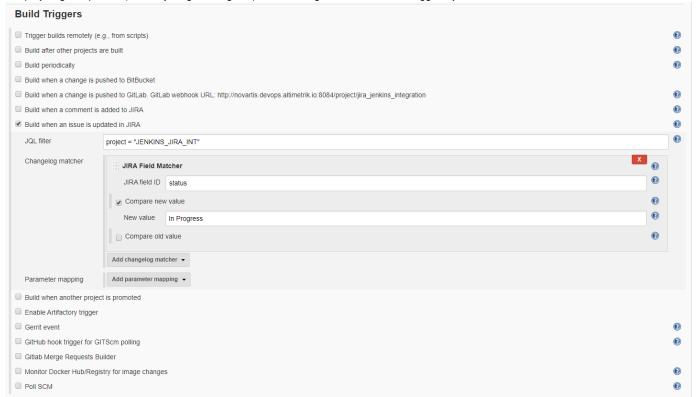


Creating Jenkins Freestyle Job:

Step 1 Provide your GIT repository details which is going to be used here to run any script etc.



Step 2 Provide what do you want to do with Jira using this job for example here we are filtering particular project and then if the issue status from that project gets updated (here anything In Progress) then a configured web hook will trigger a job.



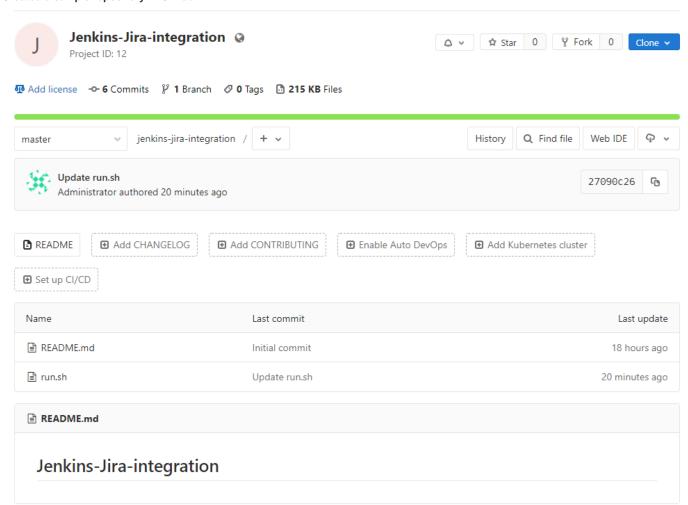
Step 3 This step is just for a demo purpose here and trying to execute a script from repository.



After configuring as shown above, you will need to go to any issue from the above configured project and change its status to see if the webhook is working as expected and triggering a particular job in Jenkins.

GitLab side customization:

Created a sample repository in GitLab.



Sample shell script which is creating some folder structure there and providing permissions to it.



Automated it using Jenkinsfile as below:

```
timestamps {
node () {
stage ('jira_jenkins_integration - Checkout') {
   checkout([$class: 'GitSCM', branches: [[name: '*/master']],
doGenerateSubmoduleConfigurations: false, extensions: [], submoduleCfg:
[], userRemoteConfigs: [[credentialsId: 'gitlab', url:
'http://novartis.devops.altimetrik.io:9080/root/jenkins-jira-integration
']]])
 }
stage ('jira_jenkins_integration - Build') {
// Shell build step
sh """
echo "This job will be triggered when there is change in JIRA ticket
status"
sh ./run.sh
 11 11 11
 }
```

References docs and links:

https://www.youtube.com/watch?v=DktKQZVLhtE

https://medium.com/@shrut_terminator/devops-usecase-jira-jenkins-integration-4051413446a9