

Set up Automatic Builds and Testing of Pull Requests

To set up automatic builds on “jenkins” (A continuous integration server). Installation of jenkins and Jenkins plugins, GitHub and jdk is need full.

How to install Jenkins on ubuntu ?

The following commands are used ,

```
// adds the repository key to the system.
```

```
$ wget -q -https://pkg.jenkins.io/debian/jenkins-ci.org.key
```

```
// Append the Debian package repository address to the server's source list.
```

```
$ sudo apt-key add jenkins_ci.org.key
```

```
// update apt-get will use new repository .
```

```
$ sudo apt-get update
```

```
// Installs jenkins and its dependencies , including java
```

```
$ sudo apt-get install jenkins
```

```
// starts jenkins and provides the path
```

```
$ sudo systemctl start jenkins
```

```
$ sudo cat ( given path )
```

Once the jenkins admin user created , enter username , password, Full name and E-mail address

How to install GitHub on Ubuntu ?

```
$ sudo apt-get install git
```

How to install Java (or) jdk ?

```
// displays the available idk packages
```

```
$ java -version
```

```
//installs available jdk package
```

```
$ sudo apt-get install openjdk-8-home...
```

How to install Jenkins plugins ?

Run Jenkins. using a browser go to Jenkins running on port 8080

For example , <http://nnn.nnn.nnn.nnn:8080> , where nnn.nnn.nnn.nnn is your system's IP address.

1. log in with your user name and password .
2. From the Jenkins Dashboard Jenkins menu, select on the Manage Jenkins.

Figure 1: Select Manage Jenkins



3. Click on Available and select all the required plugin
4. Click on Install without Restart.

Set Up Automatic Builds

1. Clone Git Repository ECL attributes

1.1 create Job

From the Jenkins Dashboard Jenkins menu, select on New Item (New Item >> enter Item Name >> Freestyle Project>>OK)

1.2 Configure the jenkins project (or) Item created to clone Git Repository

- Click on the jenkins project created
- From the project menu select on Configure
 - General section
 - provide Description about the project
 - Enter the GitHub Project Url
 - Source Code Management
 - Provide Git repository URL and Git Credentials
 - Specify branch to be clone (or) build (by default it will be 'Master' branch)

Figure 2: Provide Git Project details

General

Source Code Management

Build Triggers

Build Environment

Build

Post-build Actions

☒ GitHub project

Project url

https://github.com/Git_UserName/Repository_Name/

Advanced...

☐ This project is parameterized

☐ Throttle builds

☐ Disable this project

☐ Execute concurrent builds if necessary

Advanced...

Source Code Management

☐ None

☒ Git

Repositories

Repository URL

https://github.com/Git_UserName/Repository_Name.git

?

Credentials

Git_Uname/*****

Add

Advanced...

Add Repository

Branches to build

Branch Specifier (blank for 'any')

*/master

X

?

Add Branch

4 of 26

2. Automatically trigger the jenkins project

To build the jenkins project automatically there are several ways ,

- Build Periodically
- Poll SCM
- GitHub Pull Requests and so on

2.1. To Build the jenkins project automatically when the changes are made in git repository (polling every 5 minutes)

- From the project menu select on Configure
 - Build Triggers
 - select GitHub hook trigger for GITScm polling
 - Poll SCM >> write a expression to specify the schedule . Its basically 5 fields separated by TAB or whitespace: MINUTE HOUR DOM MONTH DOW

Figure 3: Automatic trigger_Polling every 5 minutes

Build Triggers

- ☐ Trigger builds remotely (e.g., from scripts)
- ☐ Build after other projects are built
- ☐ Build periodically
- ☐ GitHub Branches
- ☐ GitHub Pull Request Builder
- ☐ GitHub Pull Requests
- ☒ GitHub hook trigger for GITScm polling

The webhook for repo JUYayashree/jenkins_ecl on github.com failed to be registered or was removed. More info can be found on the global configuration page. This message will be dismissed if Jenkins receives a PING event from repo webhook or if you add the repo to the ignore list in the global configuration.

☒ Poll SCM

Schedule:

Would last have run at Saturday, 1 September, 2018 9:10:41 PM IST; would next run at Saturday, 1 September, 2018 9:15:41 PM IST.

Ignore post-commit hooks ☐

2.2 To Build the jenkins project automatically when the pull Request is created to any git repository

To trigger the jenkins job automatically when the pull request is created to git repository, requires several prerequisites.

- Installation of GitHub Pull Request Builder plugin
- Configuration of GitHub Pull Request Builder plugin in jenkins.
- Add webhook in GitHub Repository.

2.2.1 Install GitHub Pull Request Builder plugin

1. From the Jenkins Dashboard Jenkins menu, select on the Manage Jenkins.
2. Click on Available and select GitHub Pull Request Builder plugin
3. Click on Install without Restart.

2.2.2 Configuration of GitHub Pull Request Builder plugin in jenkins.

1. From the Jenkins Dashboard Jenkins menu, select on the Configure system.
 - Goto GitHub Pull Request Builder
 - Enter GitHub server API url as "<https://api.github.com>"
 - Add Git hub credentials i.e. your git account username and password .
 - Create API token in GitHub i.e 'personal Access Token' and paste the same into Shared secret field.
 - select Auto-manage webhooks
 - Goto GitHub Plugin
 - Add GitHub Servers
 - Add GitHub account credentials
 - Advanced >>select Override Hook URL >> copy git hub web hook Url
 - Save

Figure 4: System configuration of GitHub Pull Request Builder

GitHub Pull Request Builder

GitHub Auth

GitHub Server API URL

https://api.github.com

Jenkins URL override

Shared secret

.....

Credentials

Git_Uname/*****

Description

Add

Auto-manage webhooks

☒

Use comments to report results when updating commit status fails

☐

Use comments to report intermediate phases when build fails

☐

GitHub

GitHub Servers

GitHub Server

Name

API URL

https://api.github.com

Credentials

https://api.github.com GitHub auto generated token credentials

Manage hooks ☒

Add GitHub Server

Override Hook URL

☒ Specify another hook URL for GitHub configuration

http:// IP_address_of_jenkins_server:8080/github-webhook/

Shared secret

- none -

Add

Additional entries

2.2.3 Add Web Hook in GitHub Repository.

To trigger the jenkins server project it is required to add web hook in GitHub Repository and the following steps are followed,

- Goto your GitHub Repository
 - Click settings >> select Webhooks from menu >> Click Add webhook
 - Enter **GitHub webhook Url** copied from github plugin of jenkins.
 - Select Content Type as '**application/x-www-form-urlencoded**'
 - Add 'personal Access Token' into secret field
 - Goto Which events would you like to trigger this webhook?
 - Select Let me select individual events>> select 'Pull requests' , 'Pull request review comments' and 'Pull request reviews'.
 - Click Add web hook to create webhook.

Figure 5: Web hook setting at GitHub Repo

<> Code

Pull requests 0

Projects 0

Wiki

Insights

Settings

Options

Collaborators

Branches

Webhooks

Integrations & services

Deploy keys

Moderation

Interaction limits

Webhooks / Manage webhook

We'll send a POST request to the URL below with details of any subscribed events. You can choose which data format you'd like to receive (JSON, x-www-form-urlencoded, etc). More information in our [developer documentation](#).

Payload URL *

http://Jenkins_URL/github-webhook/

Content type

application/json

Secret

..... — Edit

Which events would you like to trigger this webhook?

☐ Just the push event.

☐ Send me everything.

☒ Let me select individual events.

☐ Check runs
Check run created from the API.

☐ Check suites
Check suite created from the API.

☐ Commit comments
Commit or diff commented on.

☐ Branch or tag creation
Branch or tag created.

☐ Project cards
Project card created, updated, or deleted.

☐ Project columns
Project column created, updated, moved or deleted.

☐ Visibility changes
Repository changes from private to public.

☒ Pull requests
Pull request opened, closed, reopened, edited, assigned, unassigned, review requested, review request removed, labeled, unlabeled, or synchronized.

☒ Pull request reviews
Pull request review submitted, edited, or dismissed.

☒ Pull request review comments
Pull request diff comment created, edited, or deleted.

☐ Pushes
Git push to a repository.

☐ Releases
Release published in a repository.

☐ Repositories
Repository created, deleted, archived, unarchived, publicized, or privatized.

☐ Repository imports
Repository import succeeded or failed.

After configuring prerequisites in Jenkins project, the following configuration need to be done to trigger the project for build on pull request to GitHub repo.

- From the project menu select on Configure
 - **Build Triggers**
 - Select 'GitHub Pull Request Builder'
 - Select Use github hooks for build triggering
 - select 'Advanced' Enter github user account name in 'white list' field so that it helps the web hook to trigger the project and upload the file as comment to pull request.
 - select 'Trigger Setup' >> Add comment file >> Enter comment file path, by this its possible to upload file as comment to pull request as post build action.

Figure 6: Project configuration of GitHub Pull Request Builder

The screenshot shows the Jenkins configuration interface for the 'GitHub Pull Request Builder'. The 'Build Triggers' tab is selected. The configuration includes:

- GitHub Pull Request Builder** (checked)
- GitHub API credentials**: A dropdown menu showing 'https://api.github.com :'. A red 'X' icon is visible on the right.
- Admin list**: An empty text area.
- Use github hooks for build triggering**: A checked checkbox.
- Advanced...**: A button with a pencil icon.
- Trigger Setup**: A section containing a 'Comment File' field with a red 'X' icon, a 'Comment file path' text input field containing '/path/to/comment/file', and an 'Add' button with a dropdown arrow.

3. Perform build operation and Post Build Actions

- As a part of build operation , ECL query syntax check of all “.ecl” files which are cloned from GitHub repository and the executable ecl file build can be performed.
- To perform ECL query ‘syntax check’ and ‘excitable ECL file build’ it requires the following steps
 - Installation of HPCC system
 - Set Up single node on system
 - install client tools
 - To perform the above steps follow the link “ http://cdn.hpccsystems.com/releases/CE-Candidate-6.4.22/docs/Installing_and_RunningTheHPCCPlatform-6.4.22-1.pdf “.
- **As a post build actions , several operations can be performed, like**
 - send E-mail notification about the build Result.
 - Send E-mail Notification with attachment of jenkins log file.
 - Send file as comment to pull Request on GitHub Repo.

3.1 Perform ECL query syntax check of all “.ecl” files as build operation and send E-mail notification as post build.

3.1.1. How to perform syntax check operation as build operation?

- From the project menu select on Configure
 - Goto Build
 - Add build step
 - Select Execute shell and write a shell script to perform ECL query syntax check and display result in the jenkins log

NOTE :Similarly shell script can be written to perform many other operation

- Build the executable ECL files .
- Build the executable ECL files and push the errors encountered to log file.
- Check the syntax of ECL file queries and push the errors encountered to log file
- To generate the log file with Jenkins Build Result, Build URL and summary of Errors and warning and so on .

Figure 7: Build _ shell script to check syntax of ECL file queries.

Build

Execute shell

Command

```
#!/bin/bash

#declare a variable to indicate path where all cloned files
#are present
QUERY_PATH = "/var/lib/jenkins/workspace/ecl_syn_check_dynamic"

# extract path of all ecl files using find command
# check syntax of all ecl files using eclcc -syntax command

find $QUERY_PATH -iname '*.ecl' | while read -r out
do
    # 'out' is variable which holds the ecl file name
    eclcc -syntax $out
done
```

[See the list of available environment variables](#)

Advanced...

Add build step ▾

3.1.2 How to send E-mail notification on 'Unstable' build as post build operation ?

- It requires following steps,
 - Email Extension plugin - It is a basic plugin which is installed at the time of default plugin installation .
 - System configuration of Email Notification
 - Project configuration of Email Notification
- **System configuration of Email Notification**

To make use of Email Extension plugin to send email notification from Jenkins, it is required to make setting in Jenkins . It is general for all project which will be configured to send E-mail Notification.

 1. From the Jenkins Dashboard Jenkins menu, select on the Manage Jenkins.
 2. select on the Configure system.
 3. Goto E-mail Notification
 - Enter SMTP server of the E-mail address . For example ,Gmail account SMTP server is 'smtp.gmail.com'.
 - Give email suffix of the E-mail address
 - Click Advance
 - select SMTP Authentication
 - Provide sender E-mail id credentials
 - Enter SMTP port
 4. Save

Figure 8: System configuration of Email Notification

E-mail Notification

SMTP server	<input type="text" value="smtp.domainname.com"/>
Default user e-mail suffix	<input type="text" value="@domainname.com"/>
<input checked="" type="checkbox"/> Use SMTP Authentication	
User Name	<input type="text" value="example@domainname.com"/>
Password	<input type="password" value=""/>
Use SSL	<input checked="" type="checkbox"/>
SMTP Port	<input type="text" value="Enter domain's port num"/>
Reply-To Address	<input type="text" value=""/>
Charset	<input type="text" value="UTF-8"/>
<input type="checkbox"/> Test configuration by sending test e-mail	

[Click Bullseye](#)

- Project configuration of Email Notification

To send E-mail notification from any Jenkins project the following settings need to be done.

- From the project menu select on Configure
 - Goto Post-build Actions
 - click add Post-build Actions >> select 'Email-Notification'.
 - Enter recipient E-mail Address >> select 'Send e-mail for every unstable build'
 -

Figure 9: E-mail notification setting in individual project



The screenshot shows the 'Post-build Actions' configuration page in Jenkins. The 'E-mail Notification' section is expanded, showing a text input for 'Recipients' with the value 'example@domainname.com'. Below this is a description: 'Whitespace-separated list of recipient addresses. May reference build parameters like \$PARAM. E-mail will be sent when a build fails, becomes unstable or returns to stable.' There are two radio buttons: 'Send e-mail for every unstable build' (which is selected) and 'Send separate e-mails to individuals who broke the build'. At the bottom left of the configuration area is a button labeled 'Add post-build action' with a dropdown arrow. The top right of the configuration area has a red 'X' icon and a blue information icon.

3.1.3. How to send E-mail notification on 'Stable' build as post build operation ?

It requires following steps,

- Email Extension plugin - It is a basic plugin which is installed at the time of default plugin installation .
- System configuration of Extended Email Notification
- Project configuration of Extended Email Notification

- **System configuration of Extended Email Notification**

To make use of Email Extension plugin to send email notification on stable build or to upload console log as attached file to e-mail from jenkins, it is required to make setting in jenkins . It is general for all project which will be configured to send E-mail Notification.

1. From the Jenkins Dashboard Jenkins menu, select on the Manage Jenkins.
2. select on the Configure system.
3. Goto Extended E-mail Notification
 - Enter SMTP server of the E-mail address . For example ,Gmail account SMTP server is 'smtp.gmail.com'.
 - Give email suffix of the E-mail address
 - Click Advance
 - select SMTP Authentication
 - Provide sender E-mail id credentials
 - Enter SMTP port . For example, Default port for gmail is '465'
 - click Default Triggers >> select 'Always'. There are many options and can be selected as per the need.
4. Save

Figure 10: System configuration of Extended Email Notification

Extended E-mail Notification	
SMTP server	<input type="text" value="smtp.Domain_name.com"/>
Default user E-mail suffix	<input type="text" value="@Domain_name.com"/>
<input checked="" type="checkbox"/> Use SMTP Authentication	
User Name	<input type="text" value="example@Domain_name.com"/>
Password	<input type="password" value="....."/>
Advanced Email Properties	<div></div>
Use SSL	<input checked="" type="checkbox"/>
SMTP port	<input type="text" value="Domain_name_port_Num"/>
Charset	<input type="text" value="UTF-8"/>
Additional accounts	<div>Add</div>
Default Content Type	<div>Plain Text (text/plain)</div>
Default Triggers	<div><div><input type="checkbox"/> Aborted</div><div><input checked="" type="checkbox"/> Always</div><div><input type="checkbox"/> Before Build</div><div><input type="checkbox"/> Failure - 1st</div><div><input type="checkbox"/> Failure - 2nd</div><div><input checked="" type="checkbox"/> Failure - Any</div><div><input type="checkbox"/> Failure - Still</div><div><input type="checkbox"/> Failure - X</div><div><input type="checkbox"/> Failure -> Unstable (Test Failures)</div><div><input type="checkbox"/> Fixed</div><div><input type="checkbox"/> Not Built</div><div><input type="checkbox"/> Script - After Build</div><div><input type="checkbox"/> Script - Before Build</div><div><input type="checkbox"/> Status Changed</div><div><input type="checkbox"/> Success</div><div><input type="checkbox"/> Test Improvement</div><div><input type="checkbox"/> Test Regression</div><div><input type="checkbox"/> Unstable (Test Failures)</div><div><input type="checkbox"/> Unstable (Test Failures) - 1st</div><div><input type="checkbox"/> Unstable (Test Failures) - Still</div><div><input type="checkbox"/> Unstable (Test Failures)/Failure -> Success</div></div>

- Project configuration of Email Notification

To send E-mail notification from any jenkins project on successful build the following setting need to be done.

- From the project menu select on Configure
 - Goto Post-build Actions
 - click add Post-build Actions >> select 'Editable Email-Notification'.
 - Enter recipient E-mail Address
 - Click on Advanced >> Goto Trigger >> add Trigger >> select Always>> add Recipient List.

Figure 11: Editable E-mail notification setting in individual project

The screenshot displays the 'Post-build Actions' configuration page in Jenkins. The 'Editable Email Notification' section is active, showing options to 'Disable Extended Email Publisher' (unchecked) and a description: 'Allows the user to disable the publisher, while maintaining the settings'. Below this, the 'Project From' field is empty, and the 'Project Recipient List' field contains the email address 'example@domainname.com'. A note below the list field states: 'Comma-separated list of email address that should receive notifications for this project.' The 'Triggers' section is also visible, showing an 'Always' trigger with a 'Send To' dropdown menu. The dropdown is open, showing 'Developers' and 'Recipient List' as options, each with a red 'X' icon and a help icon. An 'Add' button is located below the dropdown. An 'Advanced...' button is at the bottom right of the triggers section. An 'Add Trigger' button is at the bottom left of the triggers section.

3.1.4. How to send File as comment to pull request on GitHub Repo with Build Status and Build Url information ?

To send file which consists of Build Status, Build Url and Syntax check (or) ECL file build result summary as comment to pullRequest created in GitHub, it is required to write shell script in build part and groovy script to append Jenkins build status at post-build Actions and upload the file after build using GitHub Pull Request Builder setting in build trigger section.

Settings to be made is as follows

1. Write ShellScript to Append 'Syntax check' & 'ECL file build' Errors to a file. for example, '.log' file
2. Write groovy Script in Post-build Actions to append Build status to the same file which contains Error information. (or) upload Build status, Build Url and Job name to comment using messages of Pull Request Builder.
3. GitHub Pull Request Builder setting to upload file to pull request.

- How to write ShellScript to Append 'Syntax check' & 'ECL file build' Errors to a file.

- Write the code on build section of project configuration.
- The following is the example of shell script to check all cloned Ecl file syntax and append the error to a file ,It intern calls python code to identify Number of total Number of errors and warning and append information to the file.

Example Code :

```
#!/bin/bash
#!/usr/bin/python

QUERY_PATH="/var/lib/jenkins/workspace/pr_buildstatus_comment/"
COMMENT_LOG_PATH="/home/jenkins/scripts/pr_buildstatus_comment/pr_buildstatus_comment.log"

if [ -f $COMMENT_LOG_PATH ]; then
    rm $COMMENT_LOG_PATH
fi

echo "Automated ECL syntax check :"$'\n' >> $COMMENT_LOG_PATH
#Execute ECL queries
for ecl_file in $( find $QUERY_PATH -name *.ecl)
do
    if [ -f "$ecl_file" ]; then
        eclcc -syntax "$ecl_file" 2>> $COMMENT_LOG_PATH | >1
    fi
done
```

```
#Sleep for some time
sleep 20
```

```
python3 pr_buildstatus_comment.py >> $COMMENT_LOG_PATH
```

Example Python Code :

```
#!/usr/bin/python
import re
import jenkins
import time
import sys

COMMENT_LOG_PATH="/home/jenkins/scripts/pr_buildstatus_comment/pr_buildsta
tus_comment.log"
pattern = r"(\d+)\s+errors?\,\s+(\d+)\s+warnings?"

errors=0
warnings=0
with open(COMMENT_LOG_PATH, 'r') as f:
    for line in f:
        m=re.match(pattern,line)
        if m:
            errors += int(m.group(1))
            warnings += int(m.group(2))

print('***** After checking ECL file syntax *****')
print('***** Error and Warning Summary***')
print('Total Errors: {}'.format(errors))
print('Total warnings: {}'.format(warnings))
print('*****')

if errors == 0:
    pass
else:
    sys.exit(5)
```

- How to upload Build status, Build Url and Job name to comment using messages of Pull Request Builder

To send Build information to comment using pull request Builder

- From the project menu select on Configure
 - Goto Build Triggers
 - Select GitHub Pull Request Builder
 - click Trigger setup >> add
 - select Build status messages
 - Add Build Result Messages >> select “build result as Success”>> write message which you wish to display as comment if build is successful.
 - Add Build Result Messages >> select “build result as Failure”>>write message which you wish to display as comment if build is unsuccessful.

Figure 12 : Message indicating ‘build results’ to be displayed as comment

The screenshot shows the 'Trigger Setup' interface for 'Build Status Messages'. It contains two sections: 'Build Result Message' and 'Comment File'.

Build Result Message

- Build Result:** A dropdown menu with 'SUCCESS' selected.
- Message:** A text area containing the following text:
Build Status: SUCCESS
Build Url: \$BUILD_URL
Job:\$JOB_URL
- Add:** A button to add a new message.

Comment File

- Comment file path:** A text input field containing the path: / path / to / comment / file
- Add:** A button to add a new comment file.

- **Write groovy Script in Post-build Actions to append Build status to the same file which contains Error information.**

To write groovy script , need to install groovy postbuild plugin and write code by adding groovy post build on post built Action.

Figure 13: Groovy code on post build action to append build status to comment file

The screenshot shows the Jenkins 'Post-build Actions' configuration page. The 'Groovy Postbuild' plugin is selected. The 'Groovy Script' field contains the following code:

```
build = manager.build.result
url = manager.build.url

File file = new
File('/home/jenkins/scripts/pr_buildstatus_comment/pr_buildstatus_comment.log')

def lines = file.readLines()
lines = lines.plus(1, 'Build Status ::' + build)
file.text = lines.join('\n')
lines = lines.plus(2, 'Build URL ::' + url)
file.text = lines.join('\n')
```

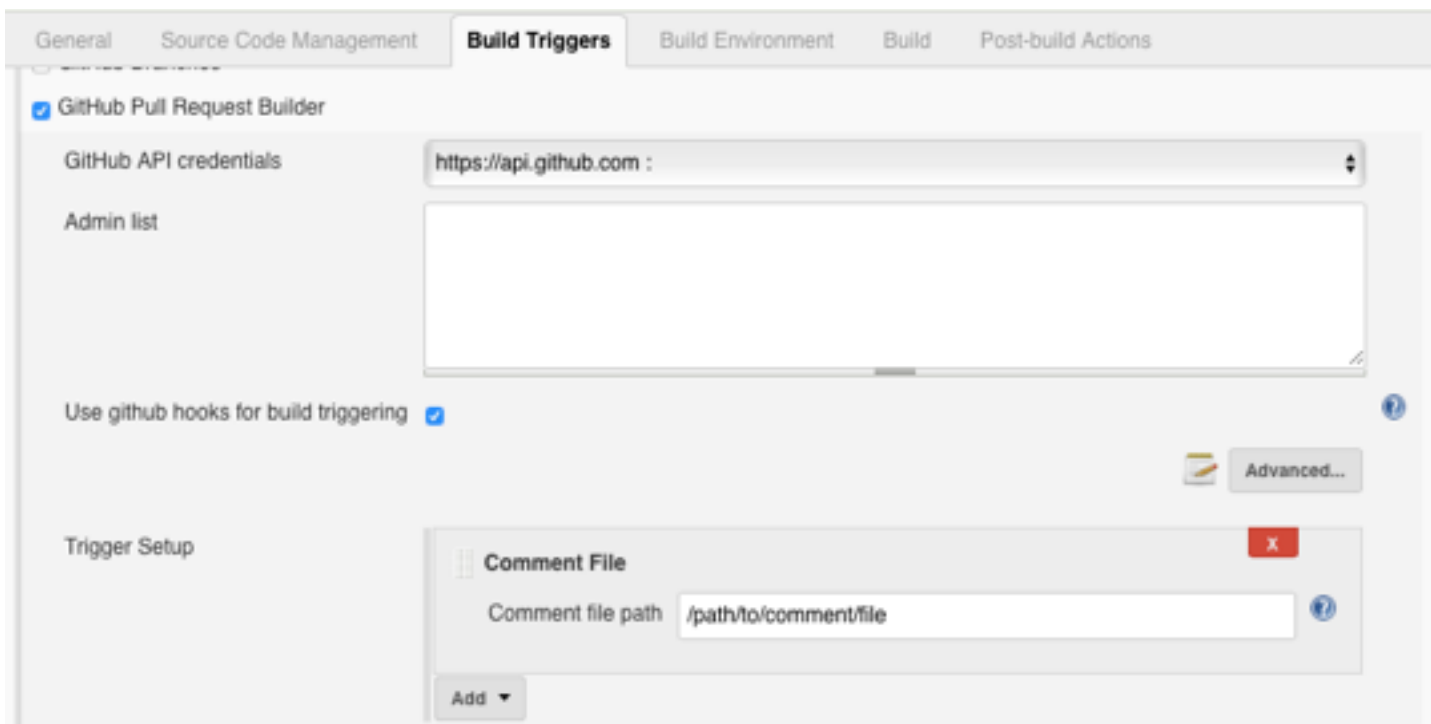
Below the script, there is a checkbox for 'Use Groovy Sandbox' which is unchecked. There is also an 'Additional classpath' section with an 'Add entry' button. At the bottom, the 'If the script fails:' dropdown is set to 'Do nothing'. A button at the bottom left says 'Add post-build action'.

- How to upload comment file to pull request using GitHub Pull Request Builder.

To upload comment , jenkins project has to be triggered when pull request is created and the “GitHub Pull Request Builder” plugin settings are configured accordingly.

- The following are the steps,
- From the project menu select on Configure
 - **Build Triggers**
 - Select the ‘GitHub Pull Request Builder’
 - Select Use github hooks for build triggering
 - select ‘Advanced’ Enter github user account name in ‘white list’ field so that it helps the web hook to trigger the project and upload the file as comment to pull request.
 - select ‘Trigger Setup ‘ >> Add comment file >> Enter/comment/file/path, by this its possible to upload file as comment to pull request as post build action.

Figure 14: Setting to upload comment file to pull request after build



Note : How to send comment file where content of file are to be displayed using HTML tags ?

It is required to store the comment file to be sent to pull request as comment , in the workspace of jenkins (or) in the respective projects folder which is present inside the jenkins workspace.

For instance : if the jenkins project name is pr_buildstatus_comment then it is required to store comment file in the path “/var/lib/jenkins/workspace/pr_buildstatus_comment/ Comment_filename.log “.

3.1.5. How to trigger (or) pass control to a job from another job on successful build

Upstream job: The Jenkins project which triggers or pass control to another job is considered as upstream job.

Downstream job: The Jenkins project which is triggered by another job is considered as downstream job.

To pass control from one job to another job the following steps can be used

- In Upstream job should include script to pass control to another job on successful build or depend on required condition.
- In downstream job need to configure the build trigger settings.

- **pass control to another job on successful build or depend on required condition of present job .**

The following code can be used in python script to pass control depend on the number of errors encountered while executing the code

```
#!/usr/bin/python
```

```
import re
```

```
import jenkins
```

```
import time
```

```
import sys
```

```
// write code to find number of errors encountered .
```

```
if errors == 0:
```

```
    pass // pass control to another job or make the job's build status as Successful
```

```
else:
```

```
    sys.exit(5) // make the job's build status as Unstable
```

- **configuration of build trigger settings in downstream job.**

The following configuration is made in 'build trigger' block of project configuration.

From the project menu select on Configure

• **Build Triggers**

- Select 'Build after other projects are built'
- enter project or projects name after which this project to be build.
- select the condition on which the build should be triggered, 3 conditions are provided
 - Trigger only if build is stable
 - Trigger even if the build is unstable
 - Trigger even if the build fails

Figure 15: Build Trigger setting in DownStream job

Build Triggers

☐ Trigger builds remotely (e.g., from scripts)

☒ Build after other projects are built

Projects to watch

☒ Trigger only if build is stable

☐ Trigger even if the build is unstable

☐ Trigger even if the build fails

☐ Build periodically

☐ GitHub Branches

☐ GitHub Pull Request Builder

☐ GitHub Pull Requests

☐ GitHub hook trigger for GITScm polling

☐ Poll SCM

Note : To make the upstream job build status as downstream build state or to make the upstream job to be blocked for the downstream job to build before sending comment file to pull request of GitHub as post build action the following steps can be used.

- Install Parameterized Trigger Plugin
- Configure the Parameterized Trigger Plugin settings in upstream job

From the project menu select on Configure

- **Build**
 - click Add build setup
 - select 'Trigger/call builds on other projects'
 - Enter triggered project name as projects to build
 - select 'Block until the triggered projects finish their builds'

Figure 16: Parameterized Trigger Plugin setting in UpStream job

Build

Trigger/call builds on other projects

Build Triggers

Projects to build

Triggered_project_name

No project specified

☒ Block until the triggered projects finish their builds

Fail this build step if the triggered build is worse or equal to

FAILURE

Mark this build as failure if the triggered build is worse or equal to

FAILURE

Mark this build as unstable if the triggered build is worse or equal to

UNSTABLE

Add Parameters

Add ParameterFactories

Add trigger...

Add build step

List of Figures :

Figure Number	Name	Page Number
1	Select Manage Jenkins	2
2	Provide Git Project details	4
3	Automatic trigger_Polling every 5 minutes	5
4	System configuration of GitHub Pull Request Builder	7
5	Web hook setting at GitHub Repo	9
6	Project configuration of GitHub Pull Request Builder	10
7	Build _ shell script to check syntax of ECL file queries.	12
8	System configuration of Email Notification	13
9	E-mail notification setting in individual project	14
10	System configuration of Extended Email Notification	16
11	Editable E-mail notification setting in individual project	17
12	Message indicating 'build results' to be displayed as comment	20
13	Groovy code on post build action to append build status to comment file	21
14	Setting to upload comment file to pull request after build	22
15	Build Trigger setting in DownStream job	24
16	Parameterized Trigger Plugin setting in UpStream job	25