

Custom Git Repository Feature Overview

Last Modified on 06/12/2025 5:04 pm EDT

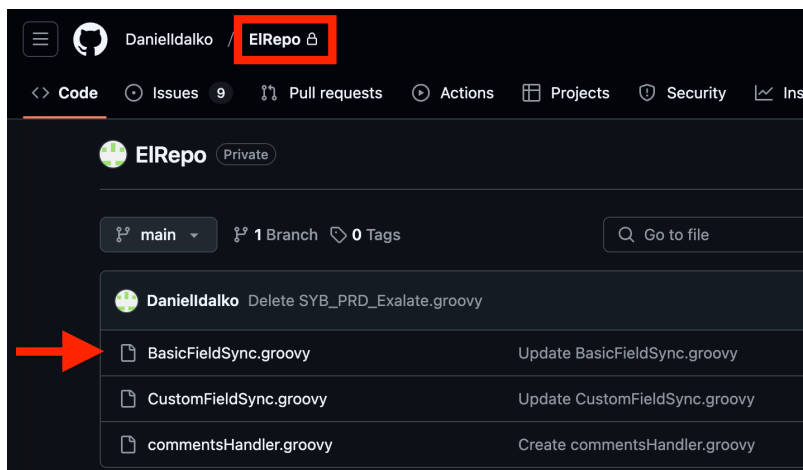
As your Exalate integration expands you may encounter the need of a centralized version control solution for your scripts, no worries, we got you covered with industry standards.

Custom Git Repository Feature Overview

Github is a platform for code version control and deployment widely used by developers all around the globe, given that Exalate uses a script based console it makes sense that some you might want to use Git to control your Exalate scripts versions.

Exalate provides a solution to integrate Github repositories into the Exalate console by using what we know as “External scripts”. This is basically a library which you can reference inside your Groovy script and import classes from.

Here is an example of a Github repository(EIRepo) which contains a script that we wish to use in the Exalate console(BasicFieldSync.groovy):



This is how the class definition looks on that script:

ElRepo / BasicFieldSync.groovy

Danielldalko Update BasicFieldSync.groovy

Code Blame 17 lines (16 loc) · 537 Bytes Code 55% faster with GitHub Copilot


```
1 class BasicFieldSync
2 {
3     static receive(issue,
4                     replica,
5                     nodeHelper,
6                     commentHelper
7                 ) {
8
9         issue.summary = replica.summary
10        issue.description = replica.description
11        issue.assignee = nodeHelper.getUserByUsername(replica.assignee?.username)
12        issue.reporter = nodeHelper.getUserByUsername(replica.reporter?.username)
13        issue.labels = replica.labels
14        issue.comments = commentHelper.mergeComments(issue, replica)
15        //issue.attachments = attachmentHelper.mergeAttachments(issue, replica)
16    }
17 }
```

Given that the node has been correctly configured to access this repository(see notes below) you can then reference it inside the Exalate console as follows:

✓ Incoming sync

Incoming sync rules define how received data can be interpreted on the source side. Check [the docs](#)

```
1 BasicFieldSync.receive(issue,
2                         replica,
3                         nodeHelper,
4                         commentHelper)
5
6 //Comment these lines out if you are interested in sending the full list
```



Once this is implemented I can now make changes to the script rules directly in the Git repository Class script instead of accessing the Exalate console, and enjoy the flexibility that comes with it, along with the version control and other Git features.

Important notes:

- There's a 15min update time between changes performed in the git repository to be reflected on the Exalate console itself.
- Configuration of custom git repo is performed by our cloud team on each node separately.

- **Product**
 - **Its possible** to configure the node to fetch from private repositories.
- **Example** scripts and more information on the scripting side of the action can be found [here](#) and [here](#).

[Security](#)

[Pricing and Licensing](#)

Resources

[Subscribe for a weekly Exalate hack](#)

[Academy](#)

[Blog](#)

[YouTube Channel](#)

[Ebooks](#)

Still need help?

[Join our Community](#) ⓘ

[Visit our Service Desk](#) ⓘ

[Find a Partner](#) ⓘ