

How to Manage Test Data Files using GitHub

Table of Contents

| | |
|---|---|
| Software Requirements (install from jdsrs)..... | 1 |
| Security Requirements..... | 1 |
| Link to GitHub Repository..... | 2 |
| Setup of Git..... | 2 |
| Steps to Clone the GitHub Repo..... | 3 |
| Steps to Sync the Latest Version of Test Files from GitHub to Your Local Cloned Folder..... | 4 |
| Steps to Merge the Changes from Your Local to GitHub..... | 5 |
| Additional Links & References..... | 6 |

Reminder -

Software Requirements (install from jdsrs)

1. [GitHub Desktop Client Latest \(64 bit\)](#) (if you encounter problems, check for the latest version in jdsrs)
2. [Git 2.29.2.2](#) (if you encounter problems, check for the latest version in jdsrs)

Security Requirements

1. You would need access to your team's GitHub AD group (connect with your EM/PO if you don't find the AD Group because they may be using a different naming convention)
2. Standard naming convention is "GDY_REPO_<PRODUCTID>_WORKSOFTCERTIFY"

Link to GitHub Repository

| Product Team | GitHub Repository Link |
|---|---|
| Parts SAP EWM | https://github.deere.com/parts/partssapewm-worksoftcertify |
| In case the above repository link for your team is not working or you are facing an issue, check if you are added to the correct AD Group (Security Requirements section). Else please connect with the API Enablement and Delivery Automation team through the SharePoint Form. | |

Setup of Git

1. After installation of Git 2.29.2.2, validate the git version from command prompt.
2. Launch cmd and enter `git --version`. You should get a return of the git version. If not, restart your system and repeat the step. You should see a value like below screenshot

```
C:\Users\rc47033>git --version
git version 2.29.2.windows.2
```

3. Check the sub-point below and follow the steps mentioned [here](#) to generate the Personal Access Token.
 - **For Step 7:** Selecting the scopes - select only the repo option

Select scopes

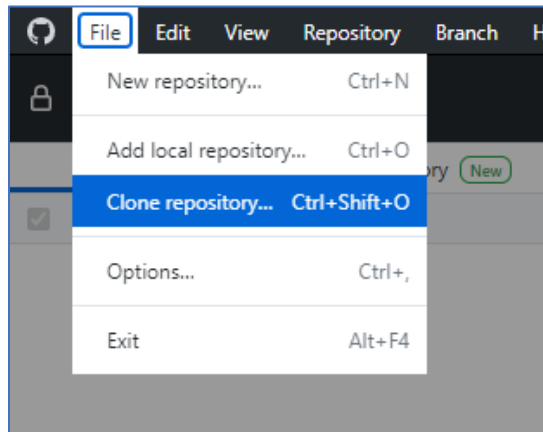
Scopes define the access for personal tokens. [Read more about OAuth scopes.](#)

| | | |
|-------------------------------------|-----------------|--------------------------------------|
| <input checked="" type="checkbox"/> | repo | Full control of private repositories |
| <input checked="" type="checkbox"/> | repo:status | Access commit status |
| <input checked="" type="checkbox"/> | repo_deployment | Access deployment status |
| <input checked="" type="checkbox"/> | public_repo | Access public repositories |
| <input checked="" type="checkbox"/> | repo:invite | Access repository invitations |
| <input checked="" type="checkbox"/> | security_events | Read and write security events |

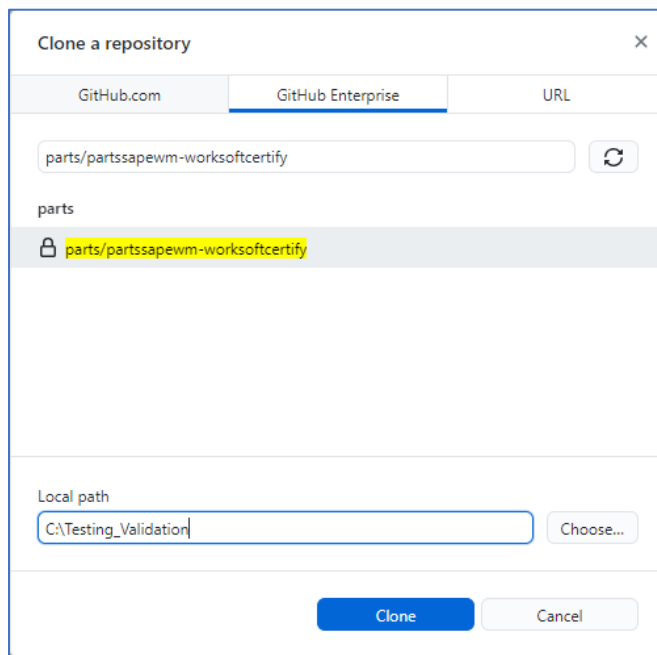
If a pop-up appears to enter the ID and password, enter your RACF and your Personal Access Token (PAT). Entering the ID and PAT should be a one-time activity as the system stores the data in its cache persistent across logins.

Steps to Clone the GitHub Repo

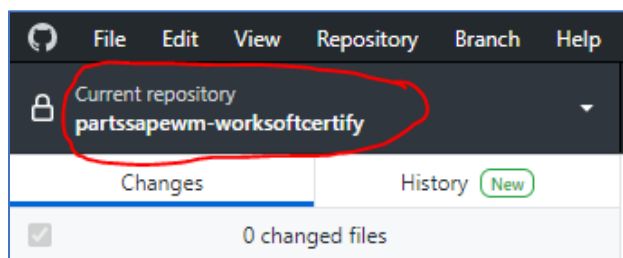
1. Open GitHub Desktop Client and navigate to **File >> Clone repository**



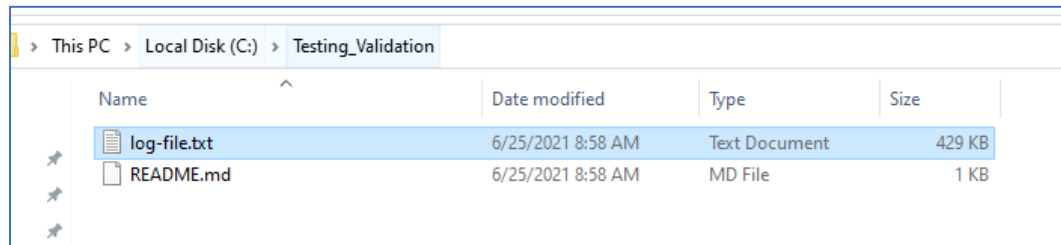
2. In the next screen that appears, enter your team's repository name and in the local path field, mention "C:\Testing_Validation"



3. After a couple of minutes, validate the repo is cloned.



4. Navigate to “C:\Testing_Validation” and verify if all the test files are present

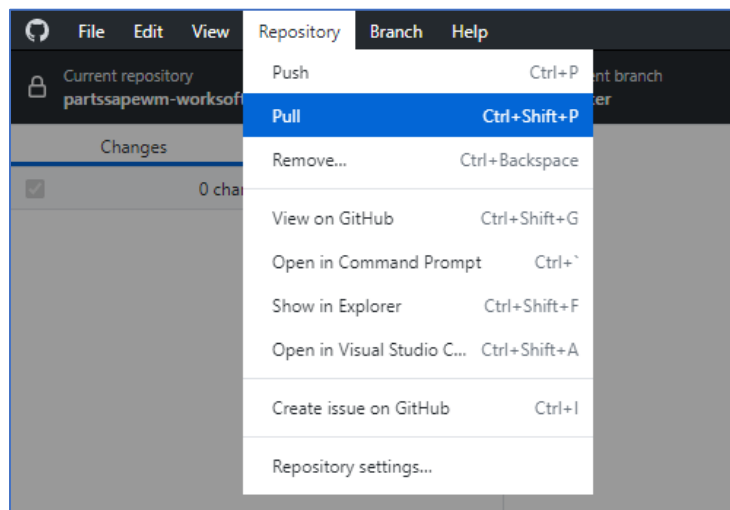


5. You are now good to start your testing work by creating new test files or editing the existing ones.

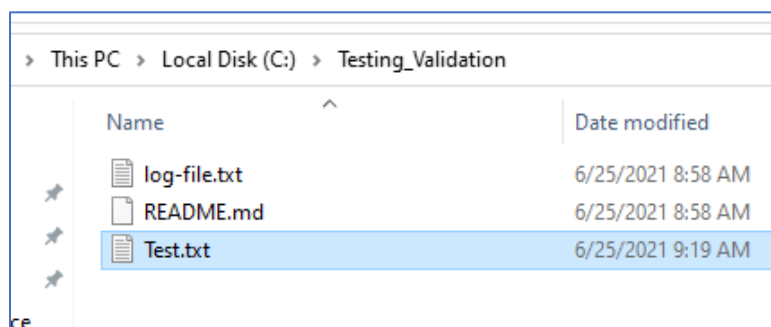
Steps to Sync the Latest Version of Test Files from GitHub to Your Local Cloned Folder

Pulling the latest changes from GitHub

1. Launch GitHub Client Desktop and navigate to **Repository >> Pull**



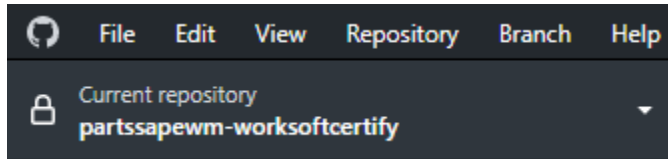
2. Validate in the local drive



Steps to Merge the Changes from Your Local to GitHub

Pushing the changes to GitHub (Committing to master branch)

1. Make the required changes to the test files and save the file.
2. Launch GitHub Desktop Client
3. Verify the repository selected is correct.



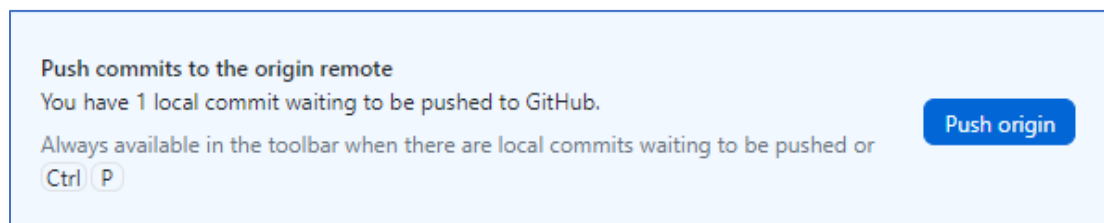
4. The changes will automatically be fetched and displayed in red and green highlight combination as seen below

| Changes 1 | | History New | Test_2.txt | |
|-------------------------------------|----------------|--------------------------|------------|---------------------------------------|
| <input checked="" type="checkbox"/> | 1 changed file | | | @@ -1 +1 @@ |
| <input checked="" type="checkbox"/> | Test_2.txt | | 1 | -Testing +Testing Commit to Master |

5. Enter the required comments and click on **Commit to master** button

A screenshot of the GitHub Desktop commit dialog. It features a text input field with the placeholder text 'Test Commit to Master'. Below this is a larger text area labeled 'Description'. At the bottom left, there is a small icon of a person with a plus sign. A prominent blue button at the bottom is labeled 'Commit to master'.

6. Click on **Push origin** button



7. Your changes have been merged with the master branch. During the script execution, the recently changed test data will be picked by your automated scripts.

Additional Links & References

[Changes to Worksoft Scripts to Implement GitHub Solution](#)

[Sync Test Data Files Between GitHub and Local System Using Command Prompt](#)

[Generating Personal Access Token for an Application ID](#)

[Steps to Setup Amazon Workspaces](#)

[Autotestpartsbs1 User's Manual](#)