Login to DFD(Datastage flow designer) using the below link.

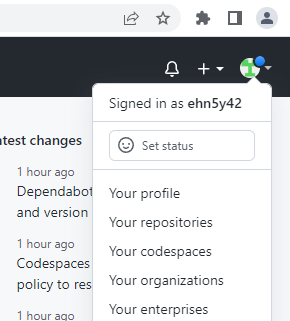
<https://sappoc-pod1-018.oc.mckesson.com:9445/ibm/iis/dscdesigner>

**🡺 Setting up a Git user in IBM DataStage Flow Designer (DFD):**

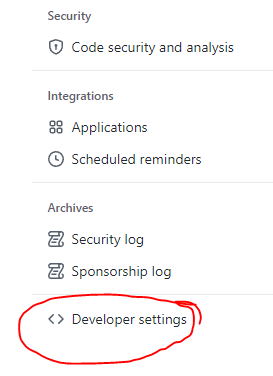
1. Click the person icon in the toolbar, select GitHub User🡪The set up user window will open.
2. In the Git email address field, enter your email address.
3. In the Git username field, enter your user name which you have for your GitHub account.
4. In the Git personal access token field, enter the personal access token that was generated by Git.

For instructions on creating a token, see [the](https://www.ibm.com/links?url=https%3A%2F%2Fhelp.github.com%2Farticles%2Fcreating-a-personal-access-token-for-the-command-line%2F) below steps.

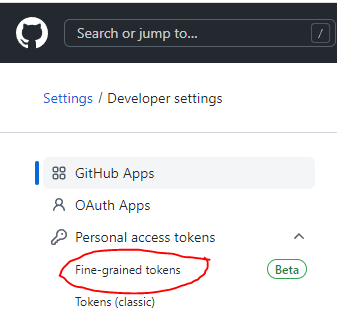
* Login to your GitHub account.
* [Verify your email address](https://docs.github.com/en/github/getting-started-with-github/verifying-your-email-address) in Git, if it hasn't been verified yet.
* In the upper-right corner of any page, click your profile photo, then click **Settings**.



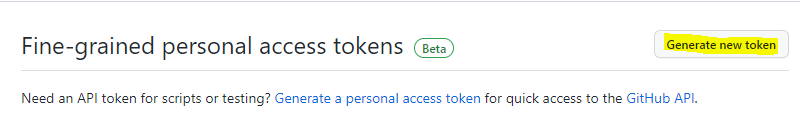
* In the left sidebar, click ***Developer settings***.



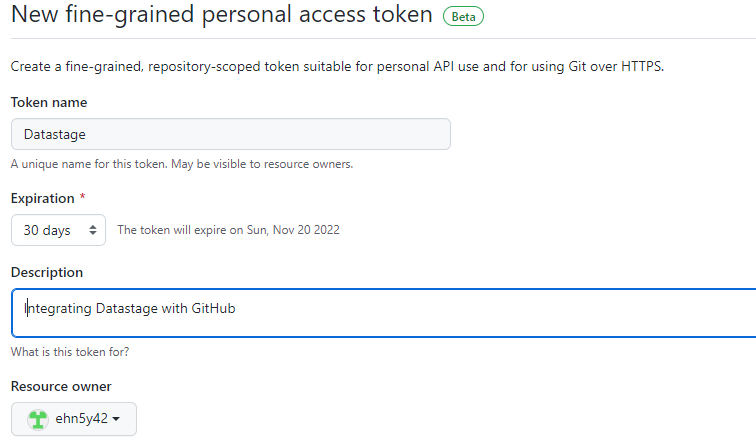
* In the left sidebar, under***Personal access tokens***, click ***Fine-grained tokens***.



* Click **Generate new token**.



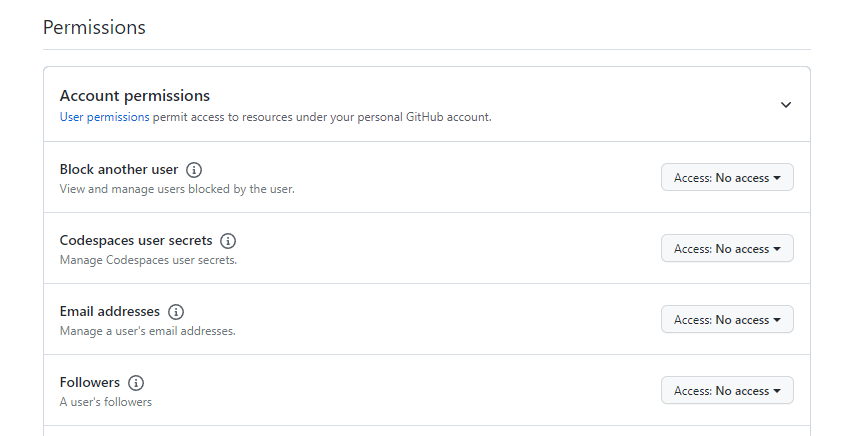
* Optionally, under **Token name**, enter a name for the token.
* Under **Expiration**, select an expiration for the token.
* Optionally, under **Description**, add a note to describe the purpose of the token.
* Under **Resource owner**, select a resource owner. The token will only be able to access resources owned by the selected resource owner. Organizations that you are a member of will not appear unless the organization opted into fine-grained personal access tokens.



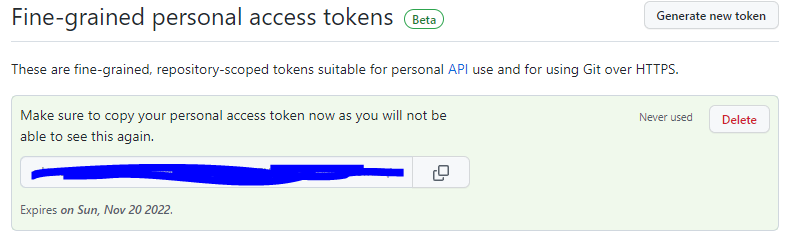
* Under **Repository access**, select which repositories you want the token to access. You should choose the minimal repository access that meets your needs. Tokens always include read-only access to all public repositories on GitHub.
* If you selected **Only select repositories** in the previous step, under the **Selected repositories** dropdown, select the repositories that you want the token to access.



* Under **Permissions**, select which permissions to grant the token. Depending on which resource owner and which repository access you specified, there are repository, organization, and account permissions. You should choose the minimal permissions necessary for your needs.

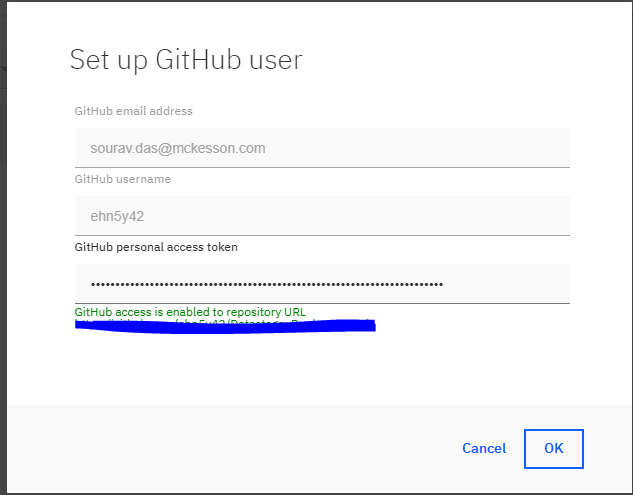


* Click **Generate token**.



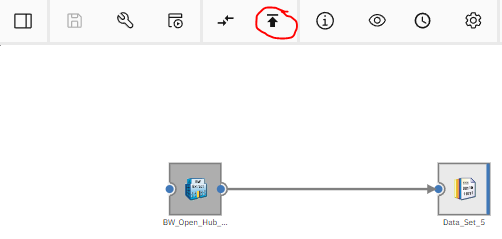
To access the full functionality of Git with IBM DataStage Flow Designer, make sure that you set the permissions for the token to "Full control of repositories" in Git.

1. Click OK to save your changes and complete the setup of the Git user in DFD.

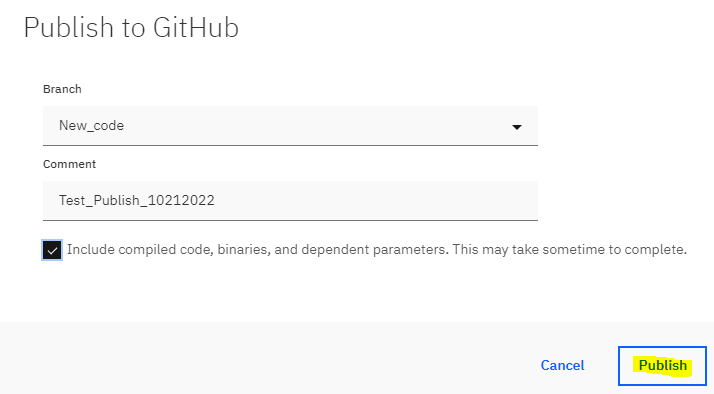


**🡺 Publishing jobs to Git by using IBM DataStage Flow Designer (DFD)** :

1. Log in to IBM DataStage Flow Designer and open the job that you want to publish.
2. Click the compile icon to compile the job. Then, save it.
3. Click the publish to Git icon to publish the file.

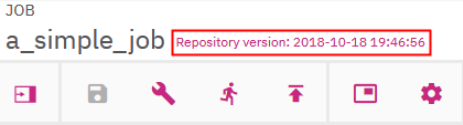


1. Select the branch that you want to add the file to and add a comment. Select Include compiled code, binaries, and dependent parameters if you want the binaries to be published to Git. This publishes the .isx package, which can be moved to other InfoSphere Information Server environments, such as QA or production.
2. Click Publish.



🡺 Loading jobs from Git by using IBM DataStage Flow Designer (DFD):

1. Log in to IBM DataStage Flow Designer and open the job that you want to load a different version of from Git.
2. Click the Repository version link next to the job name. For example:



1. Select the branch that you want to load the job from and select the version number.
2. Click Load.

***Scenario:*** You need to make a change to the job in the Job1\_branch in Git and work on it in the Job2\_branch. After you fix a bug, you need to push the change back to the Job1\_branch. Your administrator created a Git configuration, and you created a Git profile. You also pushed Job1 and Job2 to master prior to this scenario.

1. You load Job1 to develop a requirement that is complex to implement.
2. You get a defect due to a CritSit for which you need to make an immediate update to Job2 and deliver it to the customer.
3. You push the changes made so far to Job1 to a branch Job1\_branch.
4. You load Job2, fix the defect, and push to Job2 \_branch.
5. You get the changes reviewed and push the fix to the master branch.
6. You load Job1 from the Job1\_branch and continue to develop the requirement.
7. When you are done, you push directly to master or go through the branch/review process, depending on your company policy.