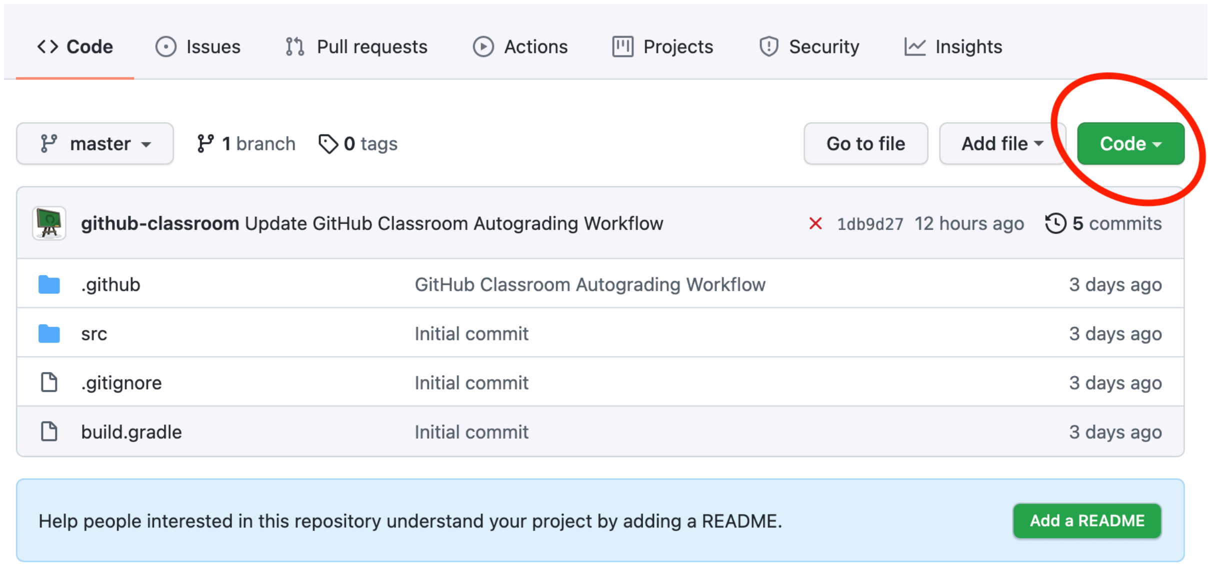
Github and IntelliJ

Please follow the directions below EXACTLY to complete this recitation successfully.

1. If you do not have a GitHub account, head over to https://github.com and sign up for an account.

2. If you have not install IntelliJ, go to IntelliJ Download and download and install the IntelliJ IDE Community Edition. This is the free version of the IDE.

3. Click on GitHub Repo. Click on the Use this template button. This will make a copy of your instructor's repo on your account.
4. You should now see your repository with a green Code button on the top right as shown in the image below.

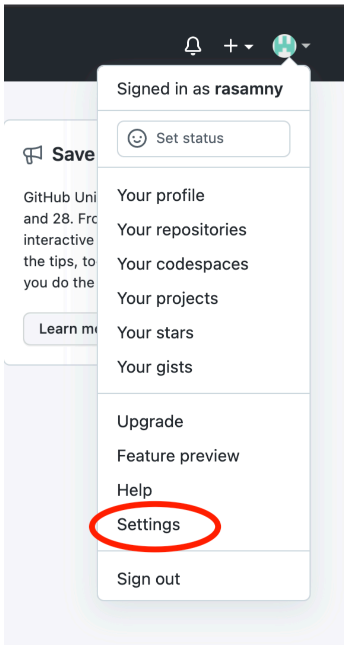


github-classroom Update GitHub Classroom Autograding Workflow ✕ 1db9d27 12 hours ago 5 commits

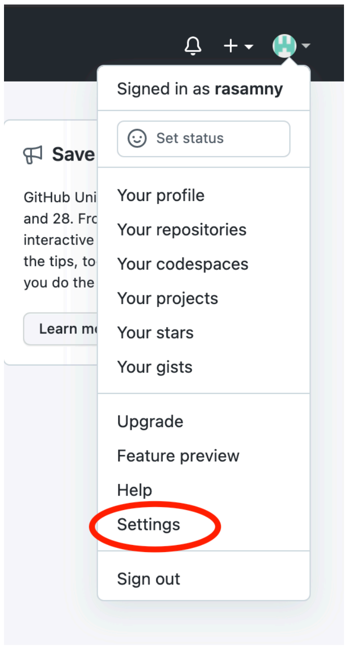
.github	GitHub Classroom Autograding Workflow	3 days ago
src	Initial commit	3 days ago
.gitignore	Initial commit	3 days ago
build.gradle	Initial commit	3 days ago

Help people interested in this repository understand your project by adding a README. Add a README

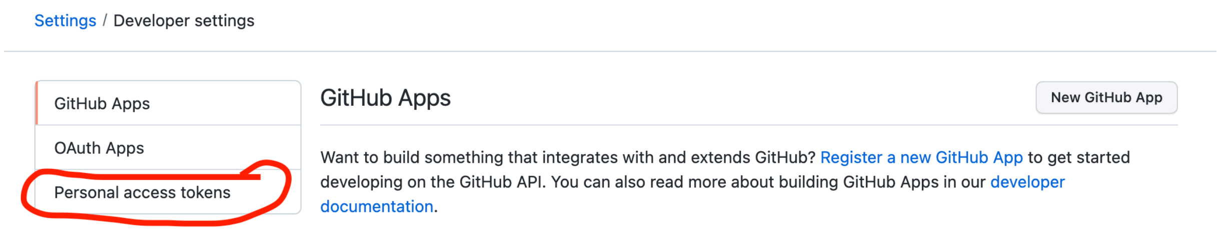
5. You will now need to create a Github authentication token to be used in IntelliJ. Click on your avatar on the top right of the screen. You should see a drop-down menu as shown in the image below. Select Settings.



6. On the Settings page, click on the Developer Setting menu item on the left as shown below.

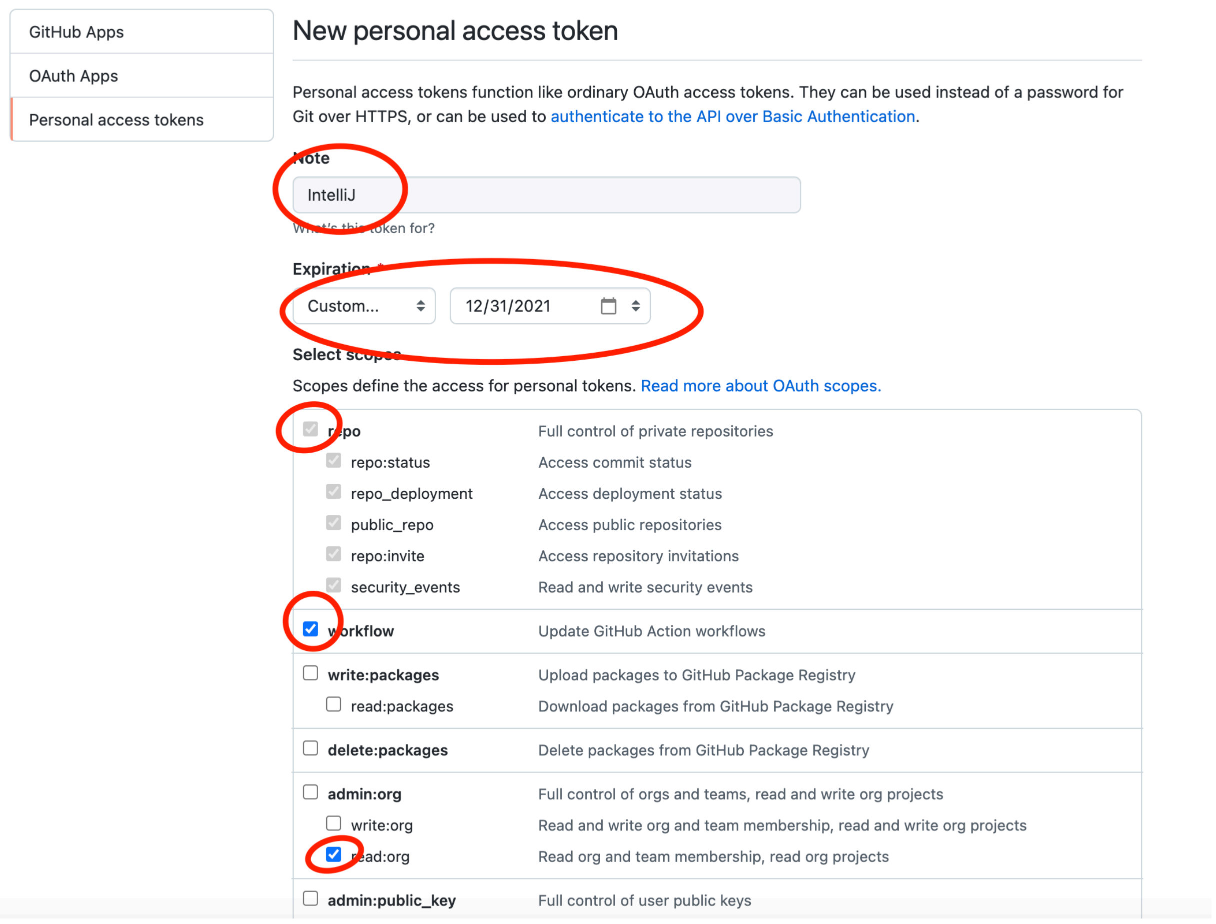


7. Select the **Personal Access Tokens** as shown in the image below. Make sure to select **Tokens (Classic)**.



Want to build something that integrates with and extends GitHub? Register a new GitHub App to get started developing on the GitHub API. You can also read more about building GitHub Apps in our developer documentation.

8. Click on **Generate new token** button and again select the classic option.
9. Enter **General Access** in the Note textbox and select the repo, workflow, gist, and read:org checkboxes as shown in the image below. Also, for the expiration, select custom and enter May 31, 2023. This will make your token valid for the entire semester.



- GitHub Apps
- OAuth Apps
- Personal access tokens

New personal access token

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

Note: IntelliJ

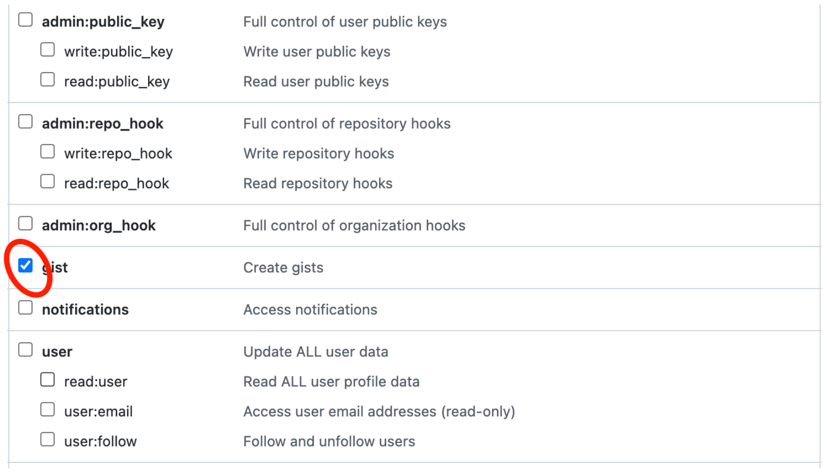
What's this token for?

Expiration *: Custom... 12/31/2021

Select scopes

Scopes define the access for personal tokens. Read more about OAuth scopes.

✓	repo	Full control of private repositories
✓	repo:status	Access commit status
✓	repo_deployment	Access deployment status
✓	public_repo	Access public repositories
✓	repo:invite	Access repository invitations
✓	security_events	Read and write security events
✓	workflow	Update GitHub Action workflows
☐	write:packages	Upload packages to GitHub Package Registry
☐	read:packages	Download packages from GitHub Package Registry
☐	delete:packages	Delete packages from GitHub Package Registry
☐	admin:org	Full control of orgs and teams, read and write org projects
☐	write:org	Read and write org and team membership, read and write org projects
✓	read:org	Read org and team membership, read org projects
☐	admin:public_key	Full control of user public keys

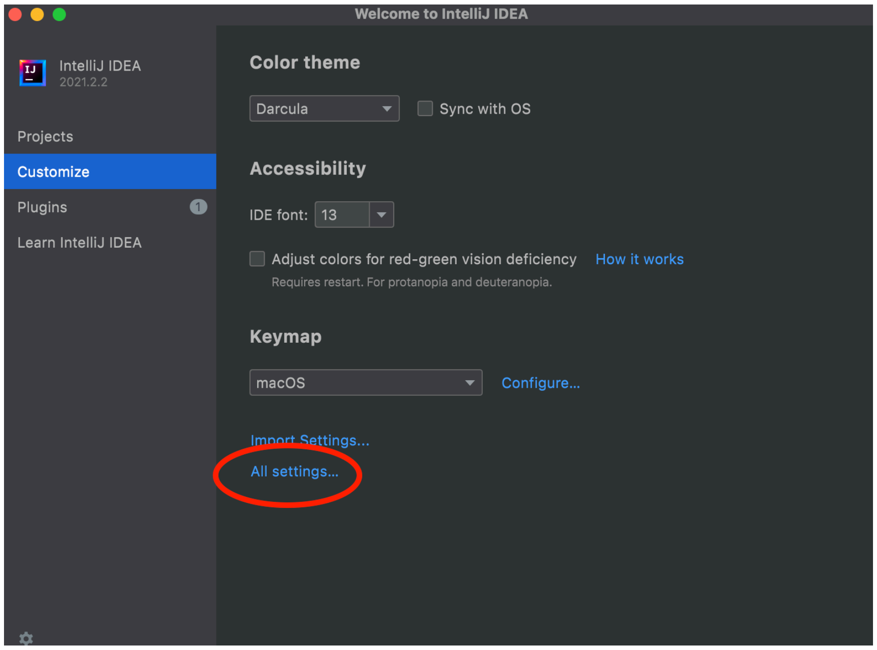


☐	admin:public_key	Full control of user public keys
☐	write:public_key	Write user public keys
☐	read:public_key	Read user public keys
☐	admin:repo_hook	Full control of repository hooks
☐	write:repo_hook	Write repository hooks
☐	read:repo_hook	Read repository hooks
☐	admin:org_hook	Full control of organization hooks
✓	gist	Create gists
☐	notifications	Access notifications
☐	user	Update ALL user data
☐	read:user	Read ALL user profile data
☐	user:email	Access user email addresses (read-only)
☐	user:follow	Follow and unfollow users

10. Click the green Generate Token button at the bottom of the screen.
11. Copy the token by clicking on the copy button as indicated in the image below. You should store the token somewhere because you will not be able to recover it. If you should forget it,

you will need to regenerate a new token.

12. Now, open up IntelliJ and click on Customize and then All Settings as shown in the image below.

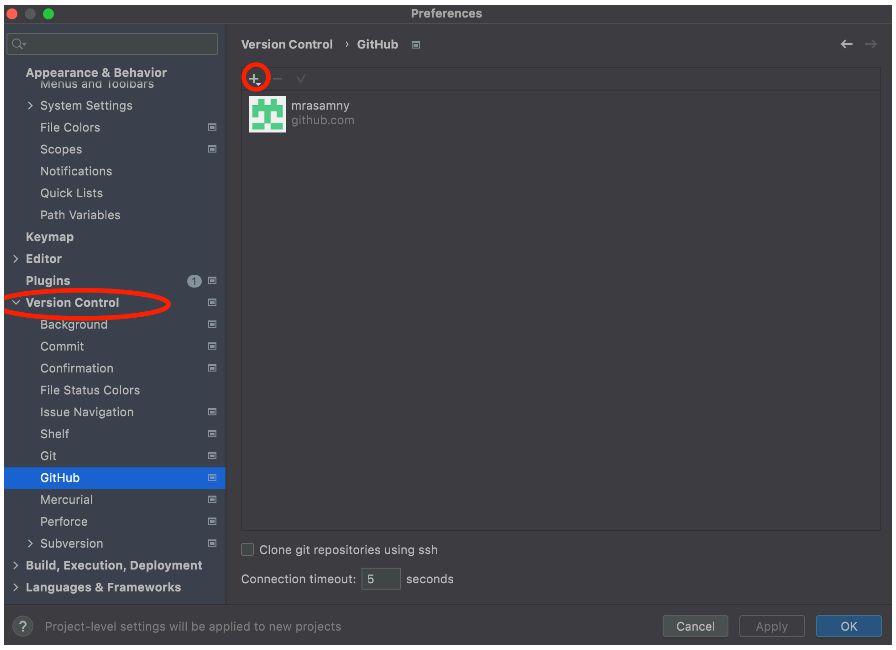


Color theme
Darcula
☐ Sync with OS

Accessibility
IDE font: 13
☐ Adjust colors for red-green vision deficiency
Requires restart. For protanopia and deuteranopia.

Keymap
macOS

13. Expand the Version Control item on the left of the window and select Github. Then click on the + as shown in the image below.

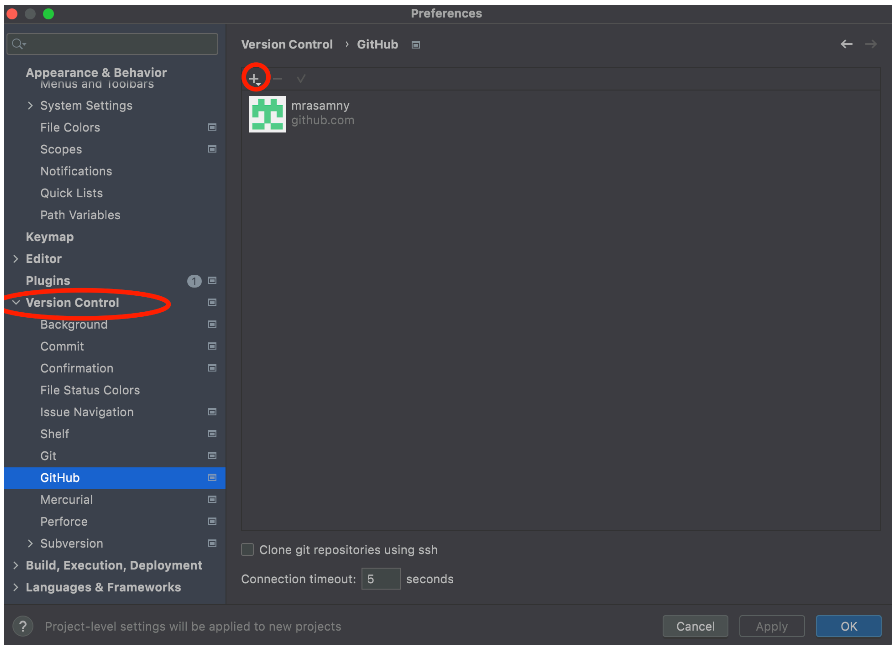


mrasamny
github.com

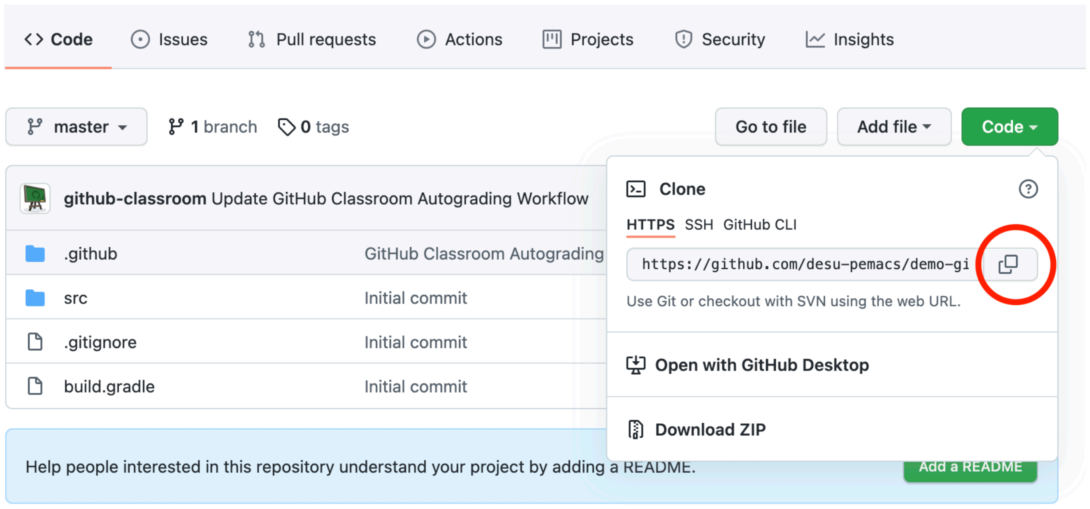
☐ Clone git repositories using ssh
Connection timeout: 5 seconds

Project-level settings will be applied to new projects

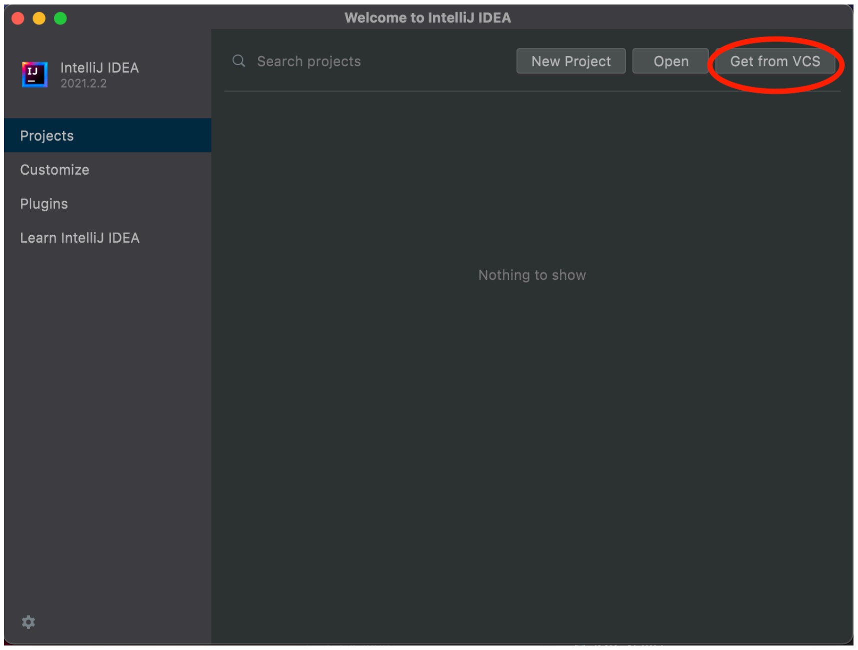
14. Select the Login with Token and paste your token in the Token textbox and click the Add Account button as shown in the image below.



15. Now, go back to **YOUR** repo and click the green Code button and then copy the URL by clicking on the copy button as indicated in the image below.


Code
Issues
Pull requests
Actions
Projects
Security
Insights
master
1 branch
0 tags
Go to file
Add file
Code
github-classroom Update GitHub Classroom Autograding Workflow
.github
GitHub Classroom Autograding
src
Initial commit
.gitignore
Initial commit
build.gradle
Initial commit
Help people interested in this repository understand your project by adding a README.
Add a README
Clone
HTTPS SSH GitHub CLI
https://github.com/desu-pemacs/demo-gi
Use Git or checkout with SVN using the web URL.
Open with GitHub Desktop
Download ZIP

16. Open up IntelliJ and use the Get from VCS button to clone the repository.

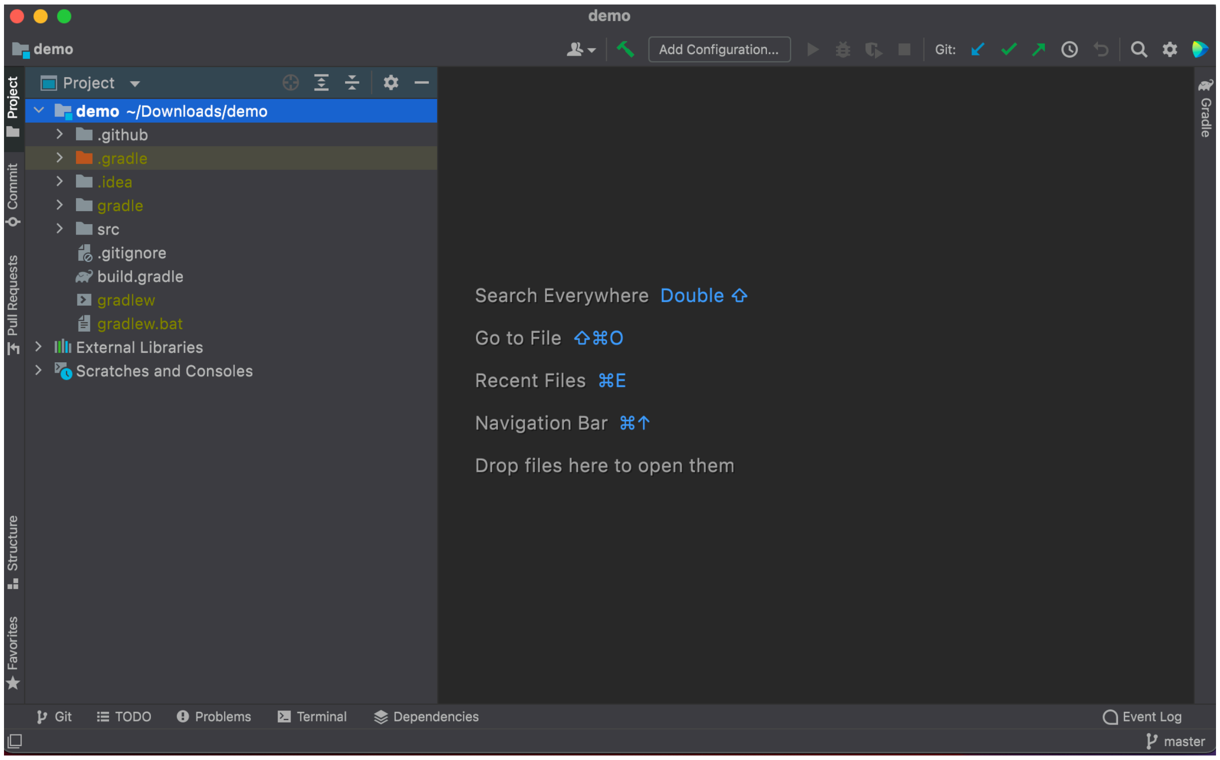


17. Paste the assignment repository URL in the URL field and make sure to select the directory in which the assignment will sit. IntelliJ will add the name of repo to the Directory path. If it does not, please add the name of the assignment directory. For example, if the directory is C:\Users\JohnSmith\Documents, make sure to add the name of the assignment. In my case, this should be changed to C:\Users\JohnSmith\Documents\recitation0

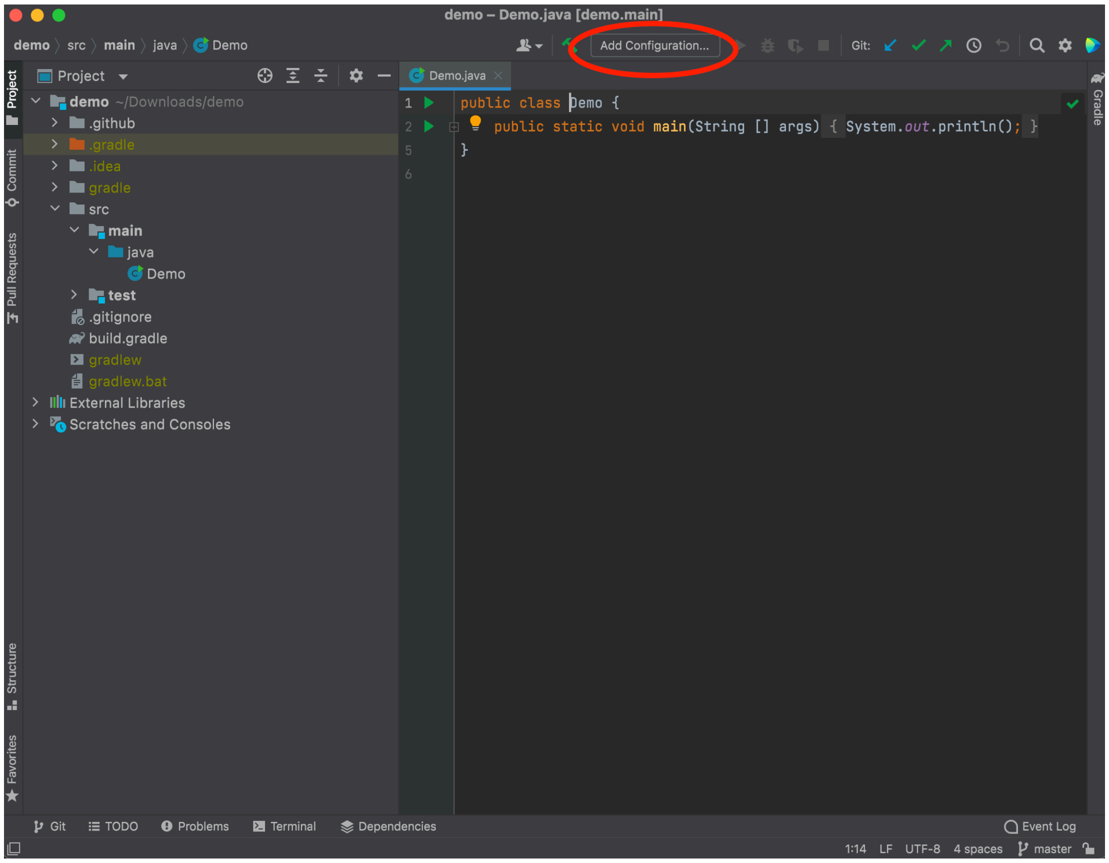
    Click the link to download and install Git, if asked.

    WARNING: If you do not add a folder name, IntelliJ will put all the project files in the Documents folder.

18. Press the Clone button. IntelliJ will create the folder for you and place the assignment code in the demo-github-classroom-rasamny folder under the Documents folder.
19. Your workspace should now look like the image shown below.



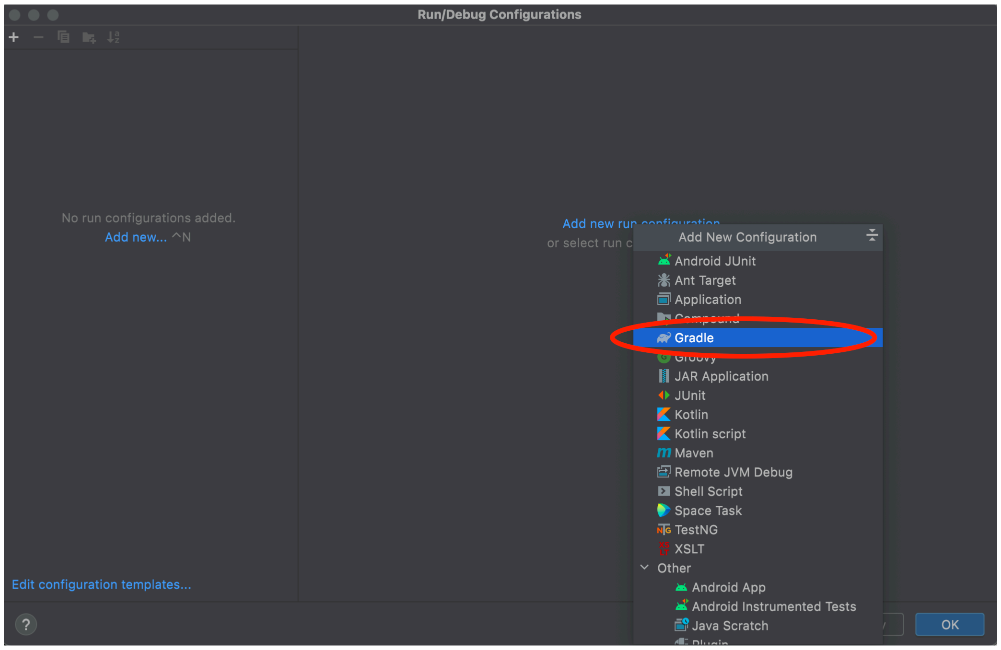
20. Now, to run and test the provided Java code, you will need to create a run configuration. Click on the run configuration button as shown in the image below.


demo – Demo.java [demo.main]
demo › src › main › java › Demo
Add Configuration...
Project
Demo.java
demo ~/Downloads/demo
main
java
Demo
test
1:14 LF UTF-8 4 spaces master

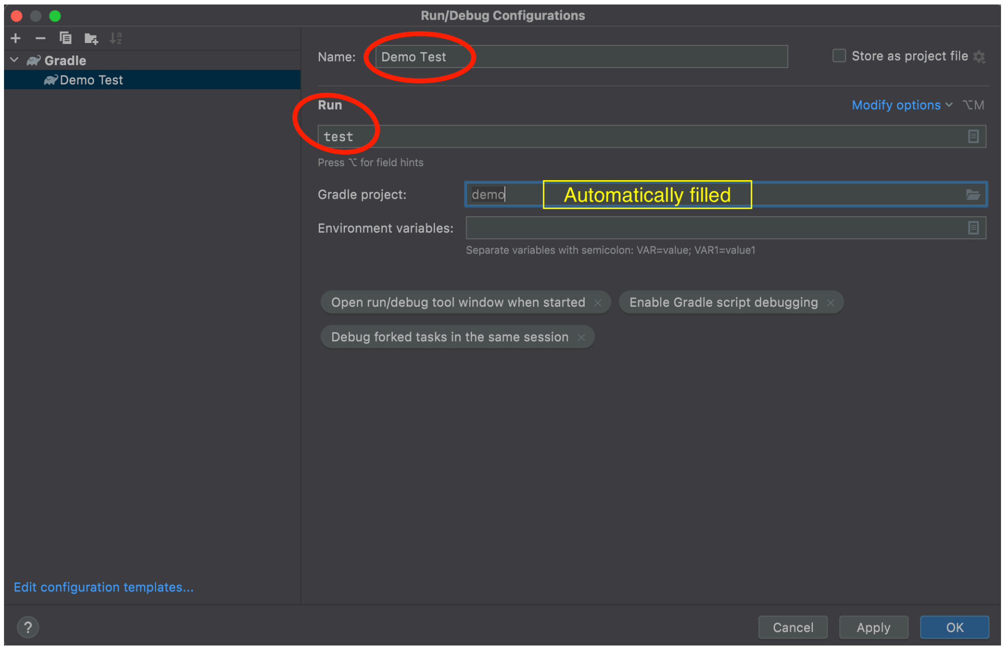
```
public class Demo {
    public static void main(String [] args) { System.out.println(); }
}
```

21. Select Add New Configuration and then select Gradle from the menu as shown in the

image below.



22. Give the configuration a name, for example **Demo Test**, and type **test** in the run textbox then click OK to complete the run configuration.


Run/Debug Configurations
Gradle
Demo Test
Store as project file
Modify options ⌥M
Press ⌥ for field hints
Automatically filled
Separate variables with semicolon: VAR=value; VAR1=value1
Open run/debug tool window when started
Enable Gradle script debugging
Debug forked tasks in the same session
Edit configuration templates...
Cancel
Apply
OK

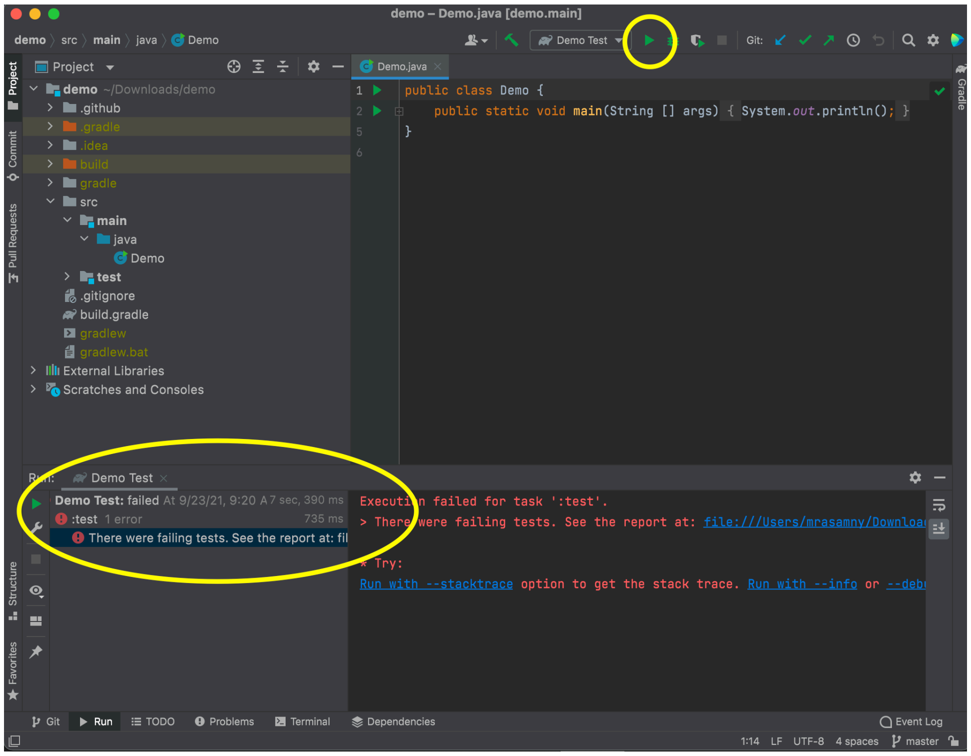
Name: Demo Test

Run: test

Gradle project: demo

Environment variables:

23. Now press the green play button and notice that the test will fail.



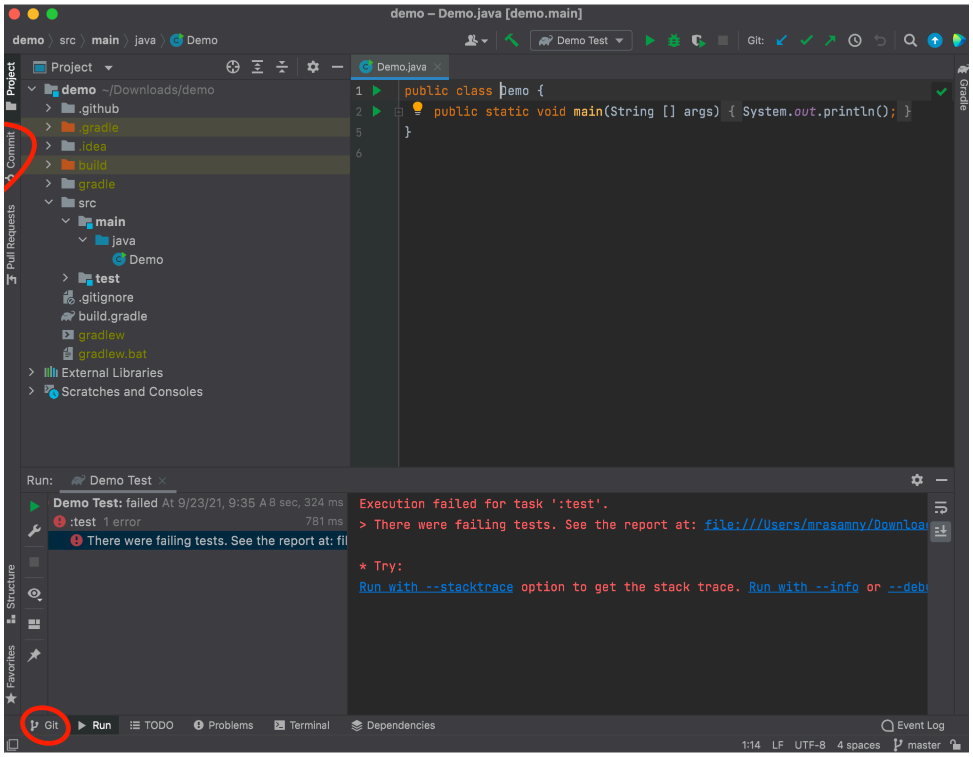
```
public class Demo {
    public static void main(String [] args) { System.out.println(); }
}
```

```
Demo Test: failed At 9/23/21, 9:20 A 7 sec, 390 ms
  :test 1 error                              735 ms
    There were failing tests. See the report at: fil
```

```
Execution failed for task ':test'.
> There were failing tests. See the report at: file:///Users/mrasamny/Downloa

* Try:
Run with --stacktrace option to get the stack trace. Run with --info or --debu
```

24. Add "Hello World!" so that the following line of code reads `System.out.println("Hello World");` and then run the test again by clicking on the green play button. You should now see that the test suucceeds!
25. You now need to commit the changes you made and push the changes to your repository on Github. Select the Commit tab as shown in the image below.



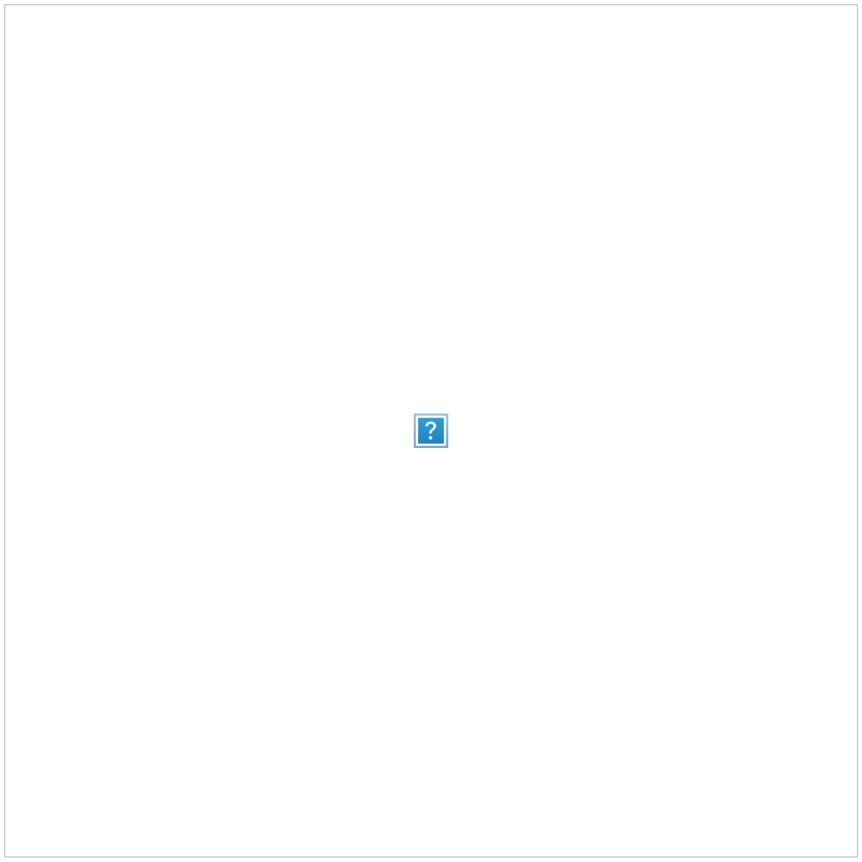
```
public class Demo {
    public static void main(String [] args) { System.out.println(); }
}
```

```
Demo Test: failed At 9/23/21, 9:35 A 8 sec, 324 ms
  :test 1 error                              781 ms
    There were failing tests. See the report at: fil
```

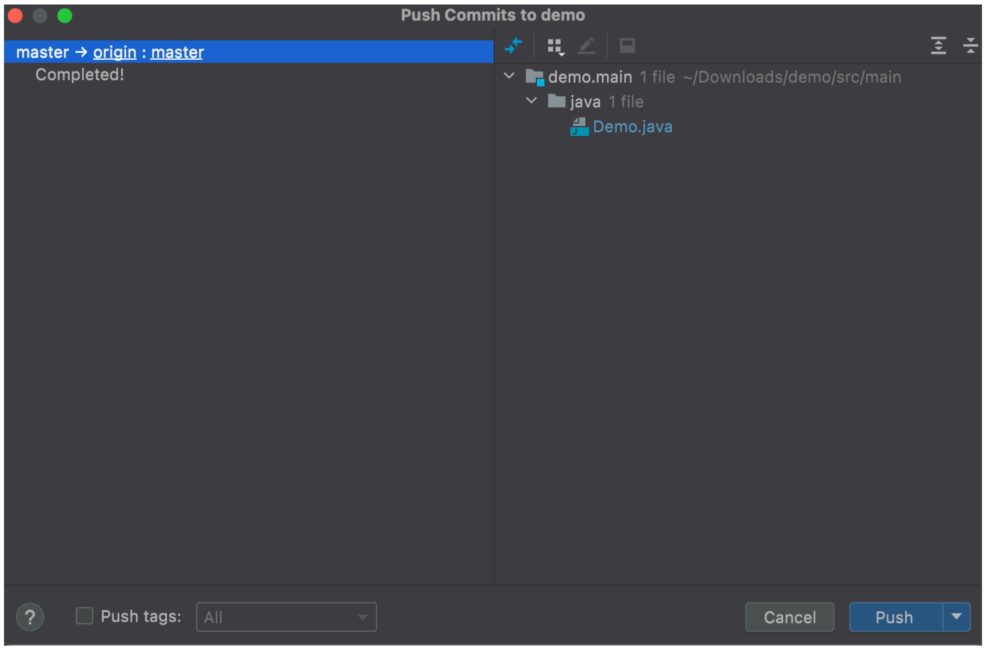
```
Execution failed for task ':test'.
> There were failing tests. See the report at: file:///Users/mrasamny/Downloa

* Try:
Run with --stacktrace option to get the stack trace. Run with --info or --debu
```

26. Select all the files that have changed and type a message in th text box as shown in the image, the message should summarize the work done at that point, and click commit and push button.



27. Click on the push button in the next window to complete the submission.



Push Commits to demo

master → origin : master
Completed!

demo.main 1 file ~/Downloads/demo/src/main
java 1 file
Demo.java

Congratulations! You now have completed your first Java

assignment!