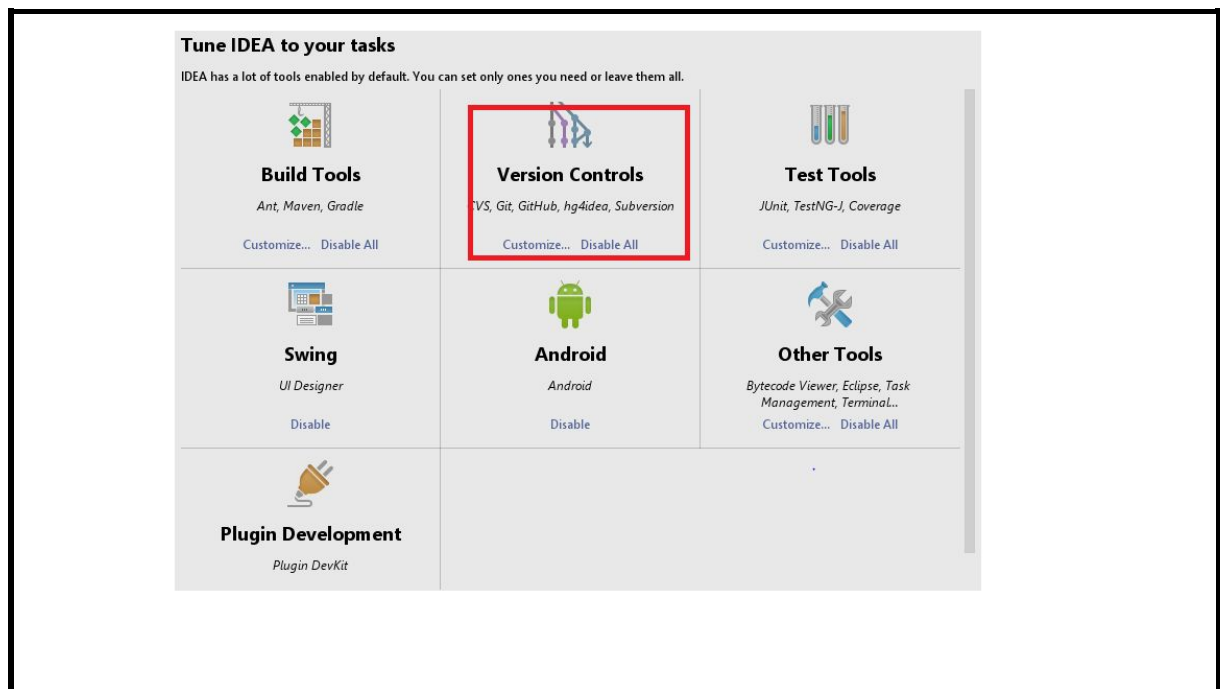


GitHub Integration in IntelliJ

IntelliJ IDEA supports integration with the GitHub remote storage.

IntelliJ Plugin:

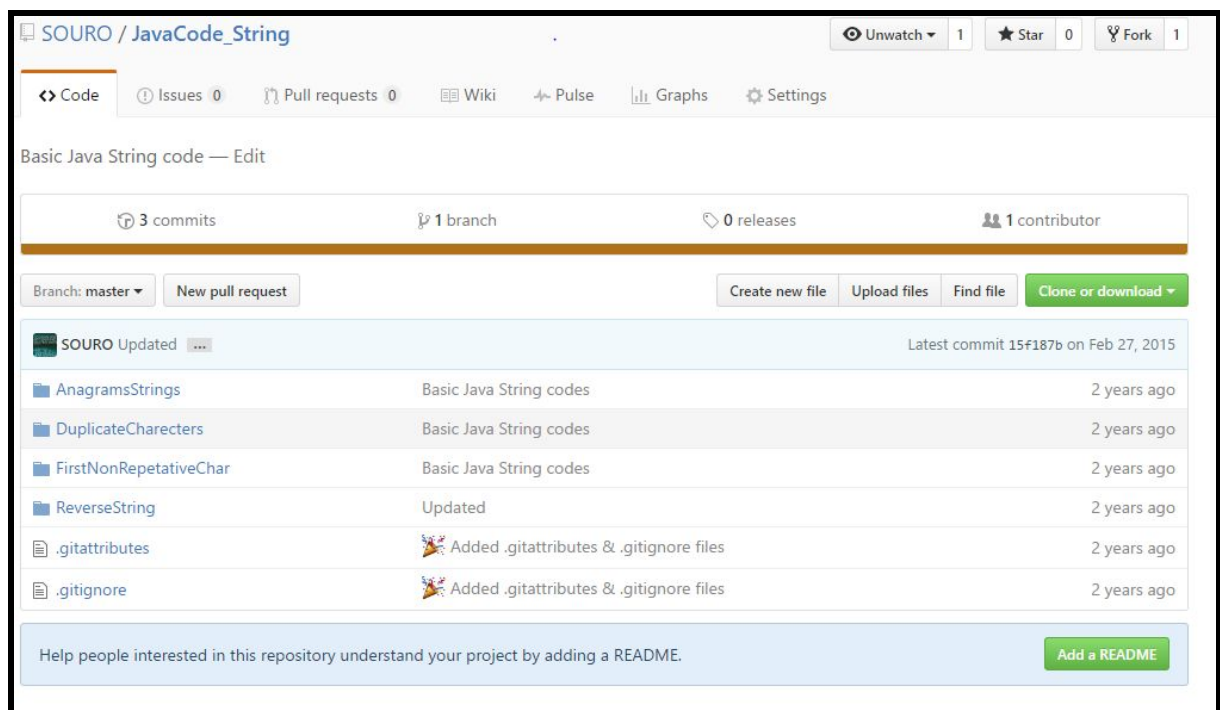
IntelliJ comes with pre installed GIT plugin.

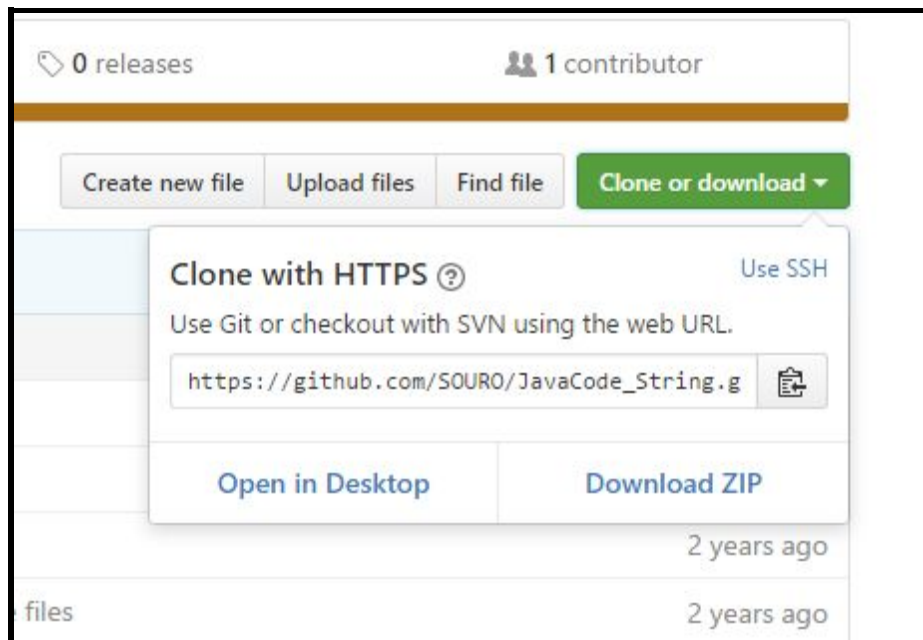


Checkout latest Code from GitHub remote repository:

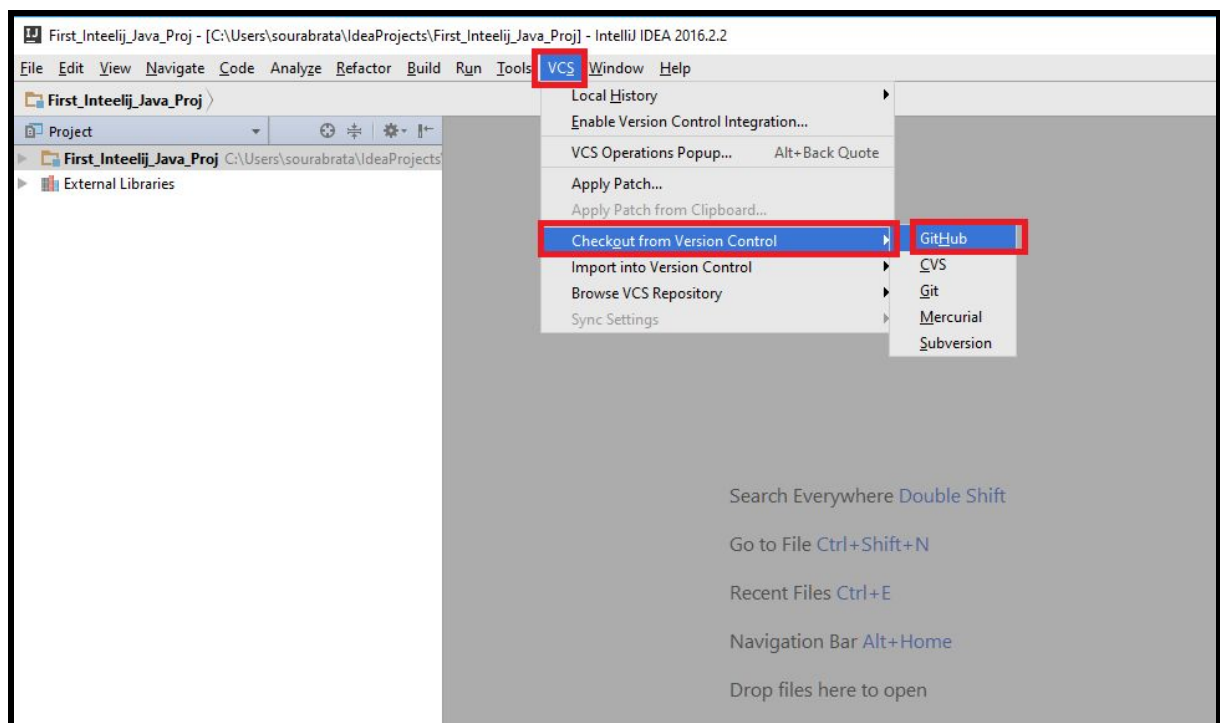
Let's say, we have a repository called "JavaCode_String" in remote GitHub.

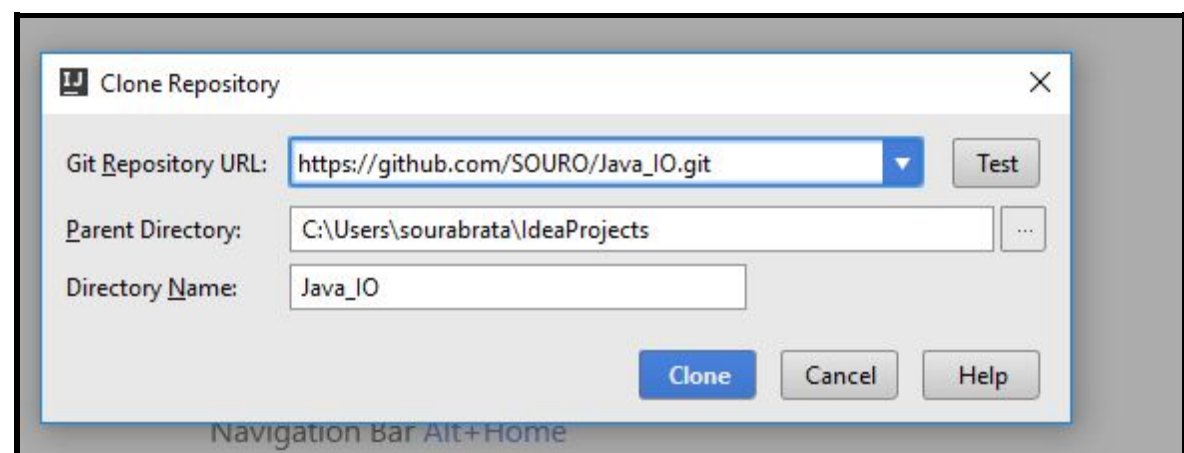
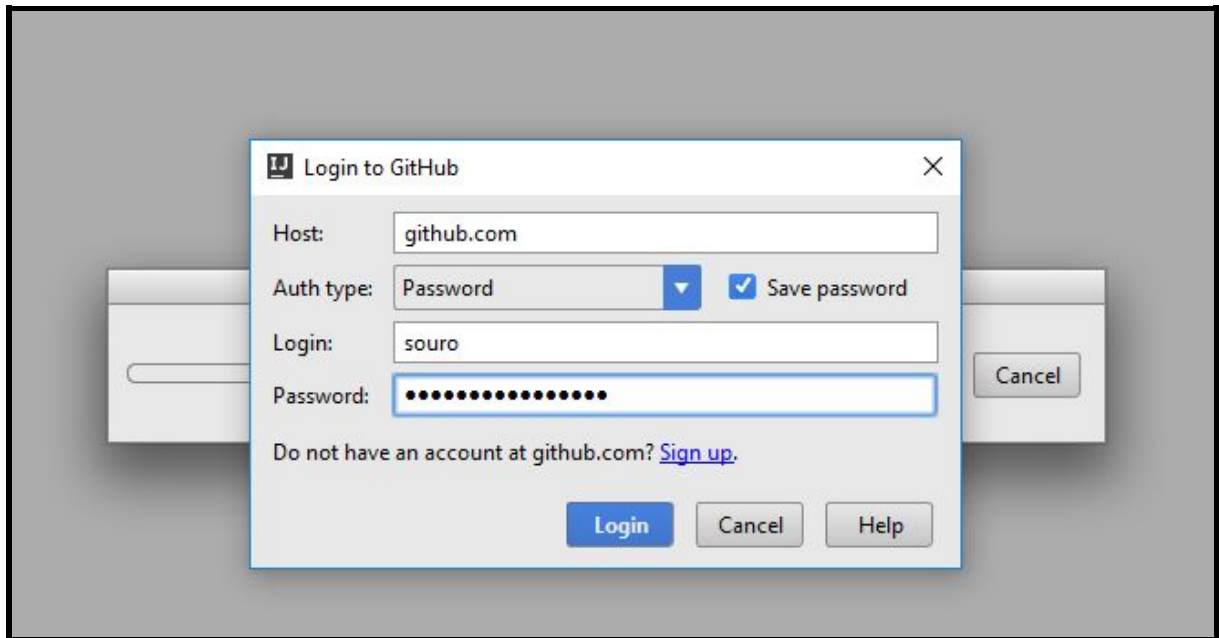
1.Copy the repository URL by clicking on “Clone or download” or later you can select from the drop down,

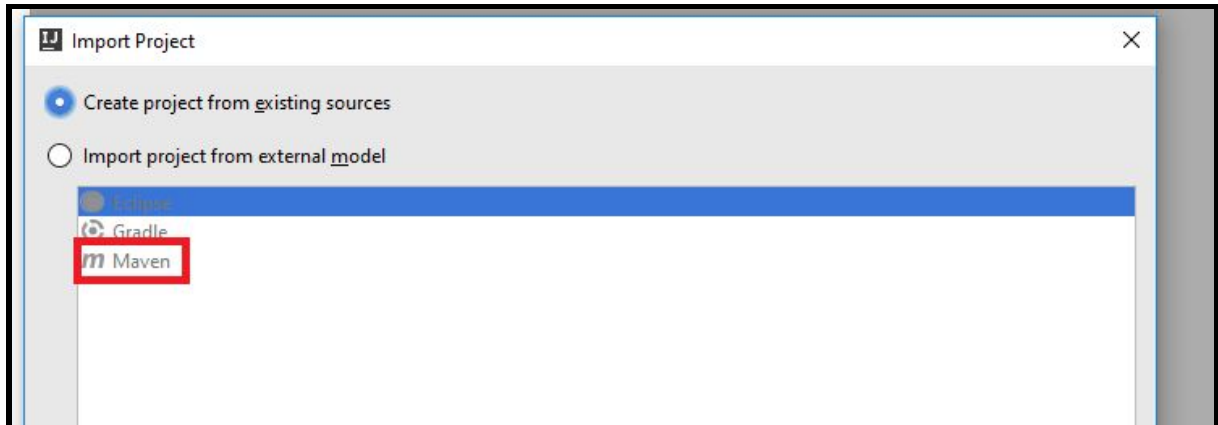
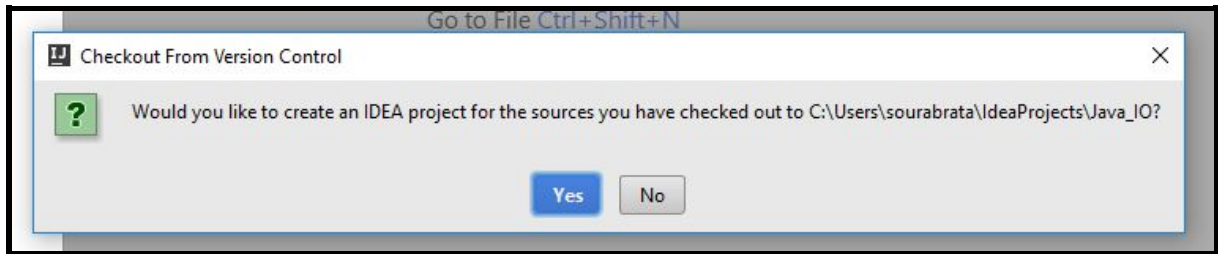




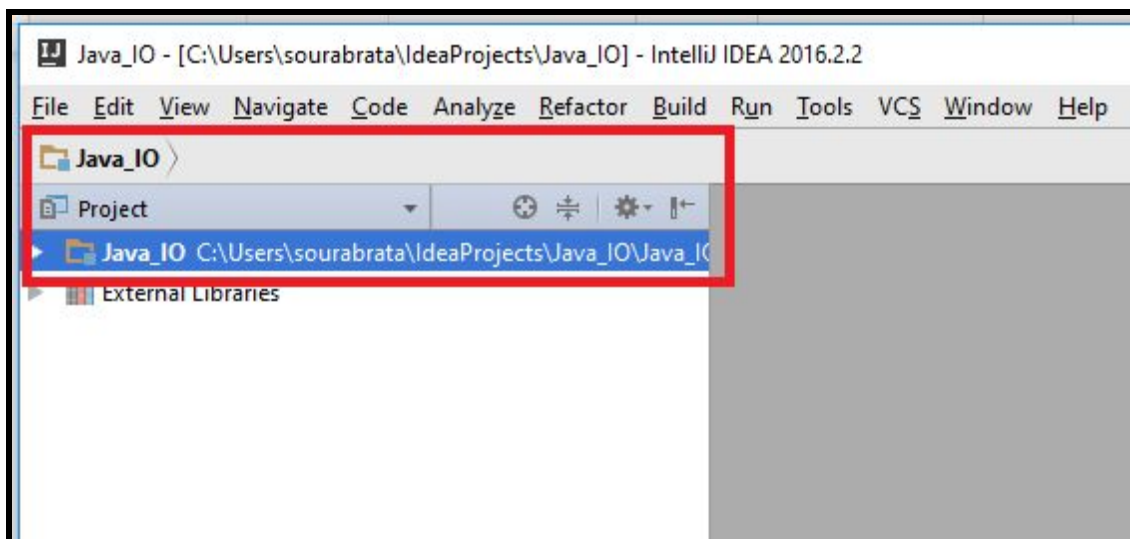
2. Go to IntelliJ and do the following to check out the code





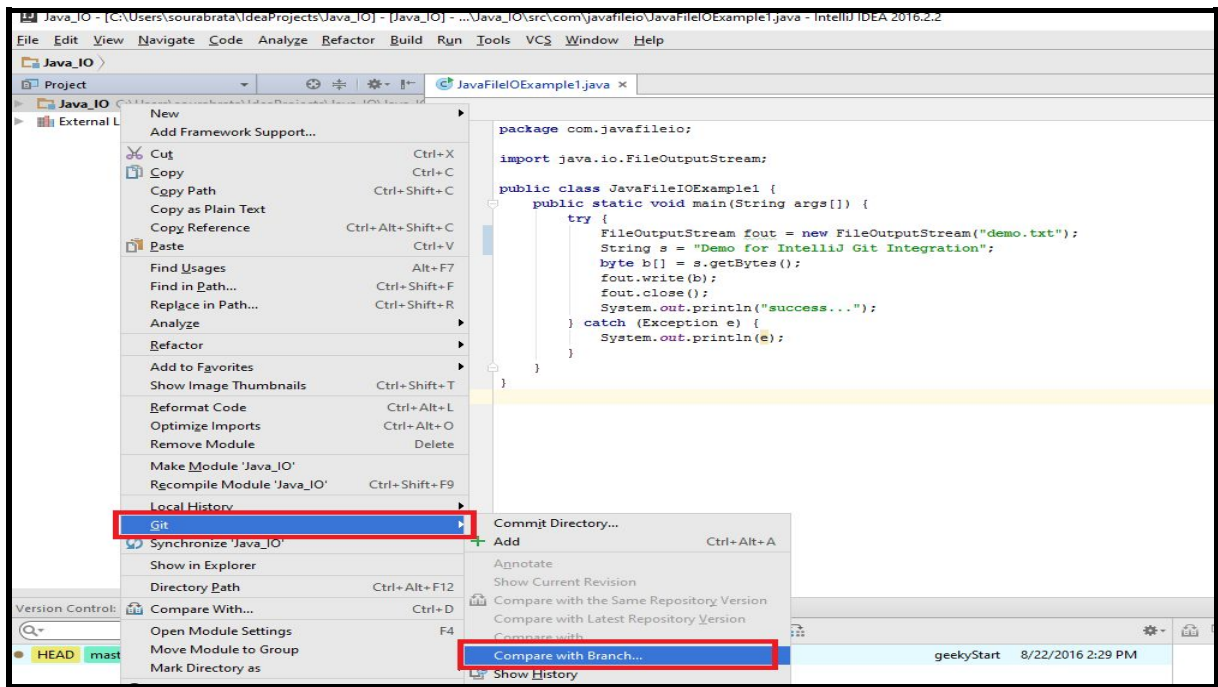


3. Once you are done with the above steps, you can see the code you have checked - out in your local.

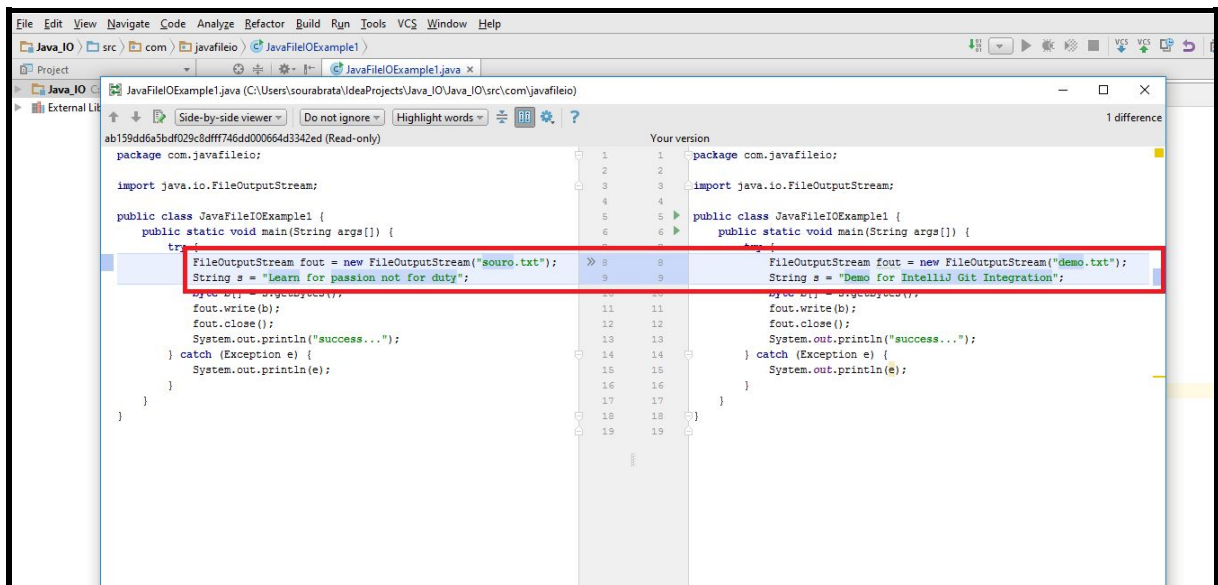


Code Development:

4. Now made required changes in your local and compare it like below,

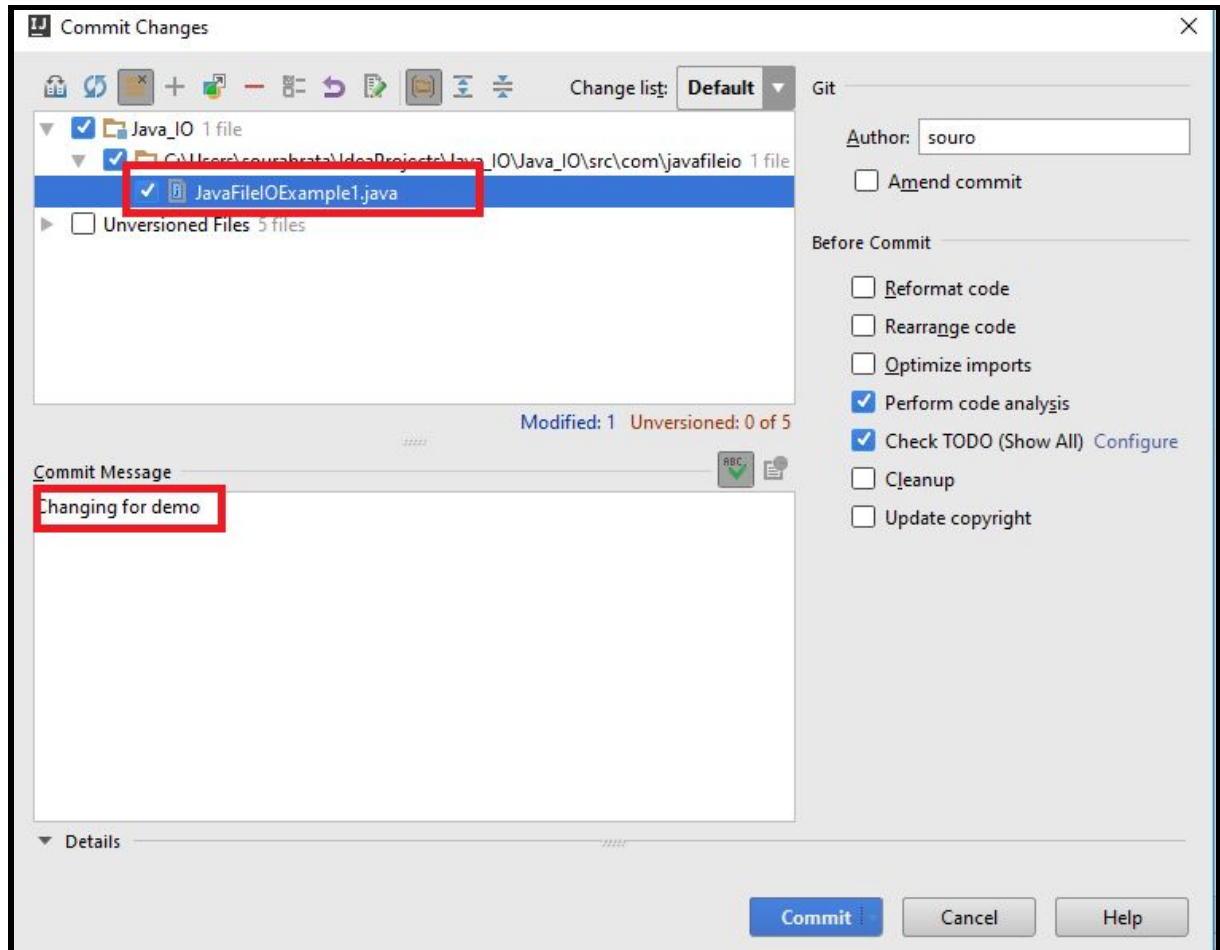


Verify that your changes are as expected,

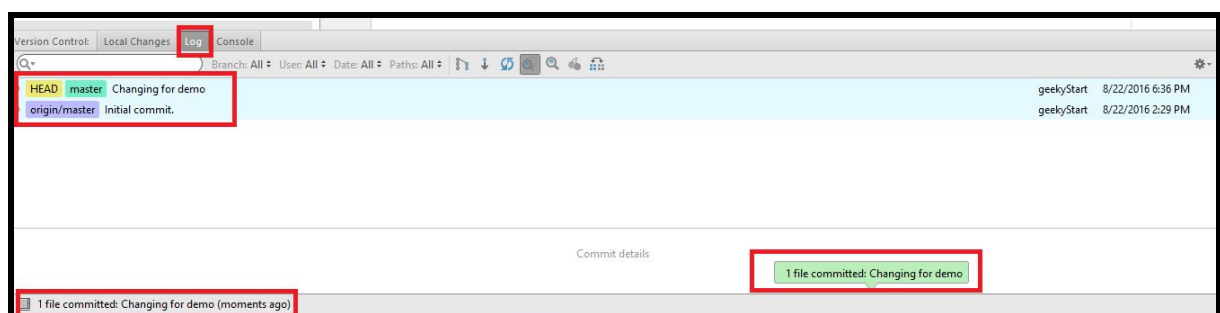


Commit in Local Repository:

5. Once done, then commit the changes locally,

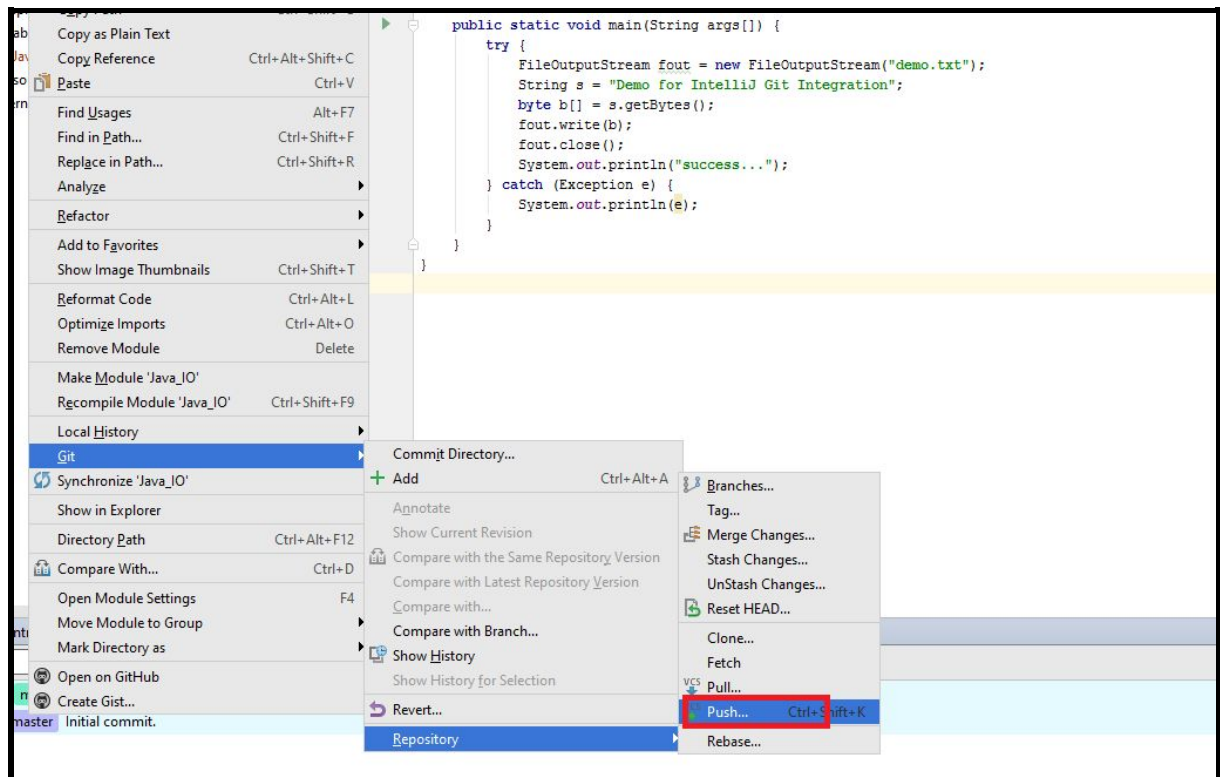


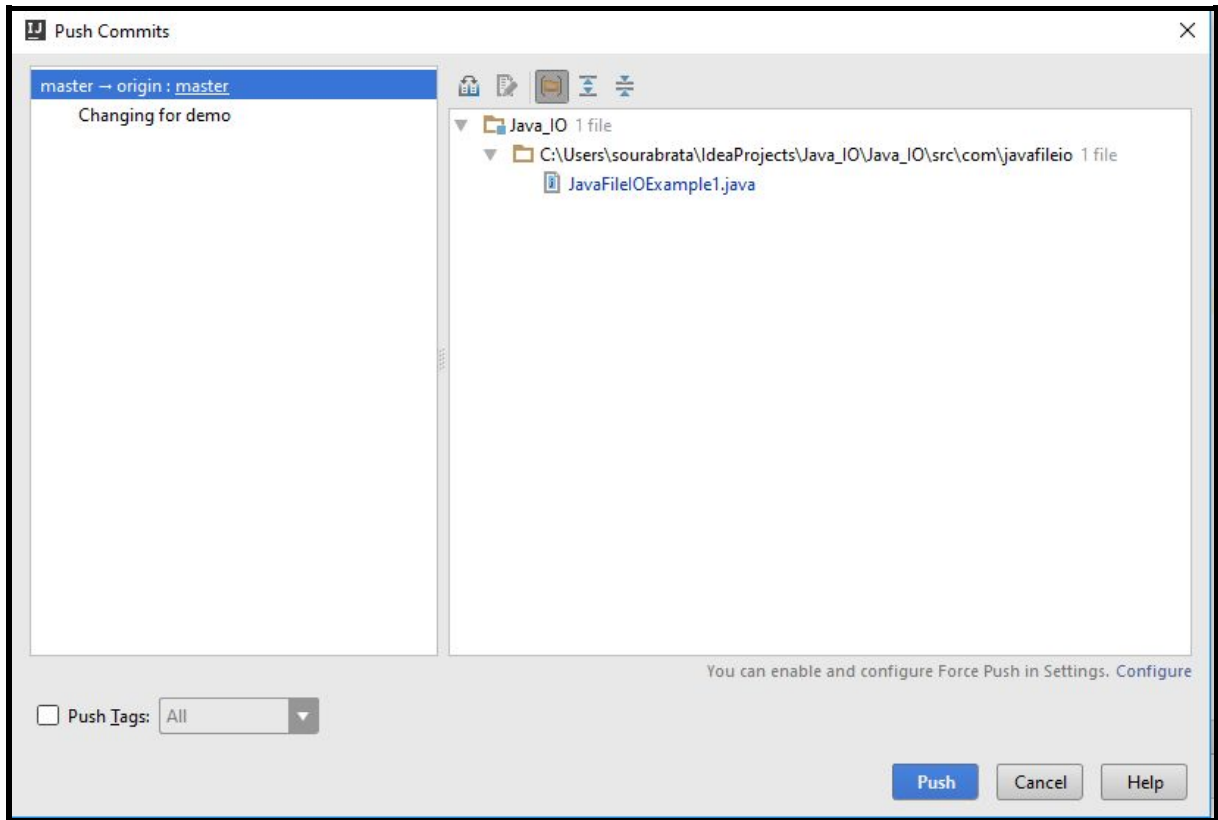
You can see message like below once it is committed in local repository.



Push Changes to Remote Branch:

6. Once you are done with the local commit, let's push it to remote(in the respective branch),





On successful completion you will get one message,



Verify changes in remote repository:

Now you can verify that your changes are pushed into the remote repository,

History for **Java_IO** / Java_IO / src / com / javafileio / JavaFileIOExample1.java

Commits on Aug 22, 2016

 **Changing for demo**
SOURO committed 10 minutes ago

 c8cf826 

 **Initial commit.**
SOURO committed 4 hours ago

 ab159dd 


SOURO committed 9 minutes ago 1 parent ab159dd commit c8cf826734149052ebf4664ec403a1d4fc2ad252

Showing 1 changed file with 2 additions and 2 deletions. Unified Split

4 Java_IO/src/com/javafileio/JavaFileIOExample1.java View

```
@@ -5,8 +5,8 @@
5      public class JavaFileIOExample1 {
6          public static void main(String args[]) {
7              try {
8                  -   FileOutputStream fout = new FileOutputStream("souru.txt");
9                  -   String s = "Learn for passion not for duty";
10                 +   FileOutputStream fout = new FileOutputStream("demo.txt");
11                 +   String s = "Demo for IntelliJ Git Integration";
12                 byte b[] = s.getBytes();
13                 fout.write(b);
14                 fout.close();
15             }
16         }
17     }
18 }
```

0 comments on commit c8cf826 Lock conversation



Write Preview

AA B i “ <> ☰ ☷ ☹ ↶ @ 📌

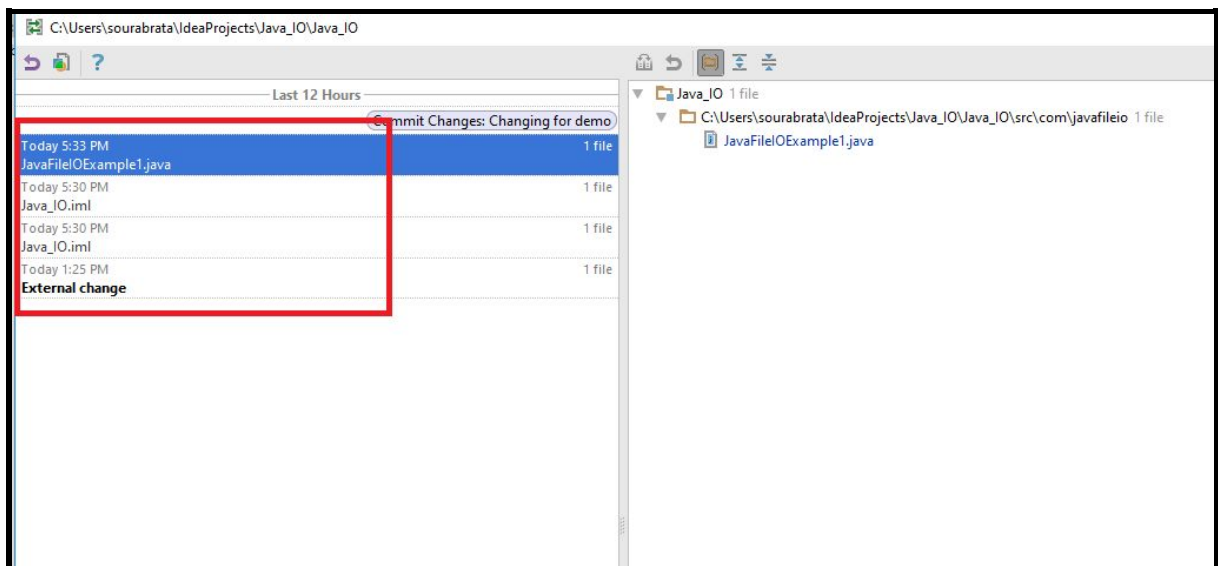
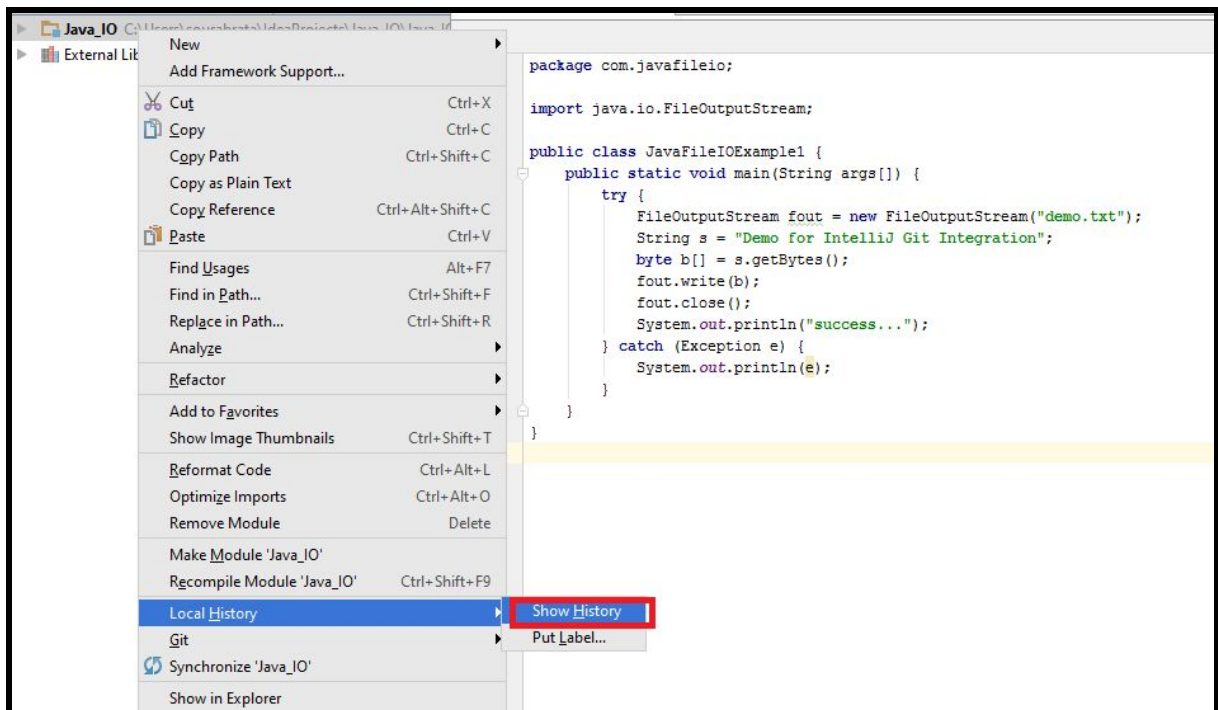
Leave a comment

Attach files by dragging & dropping, selecting them, or pasting from the clipboard.

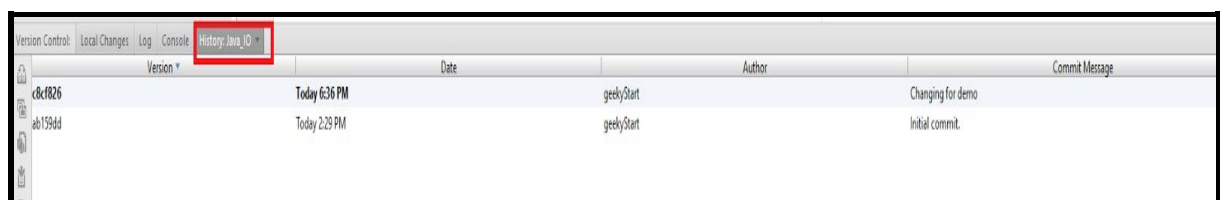
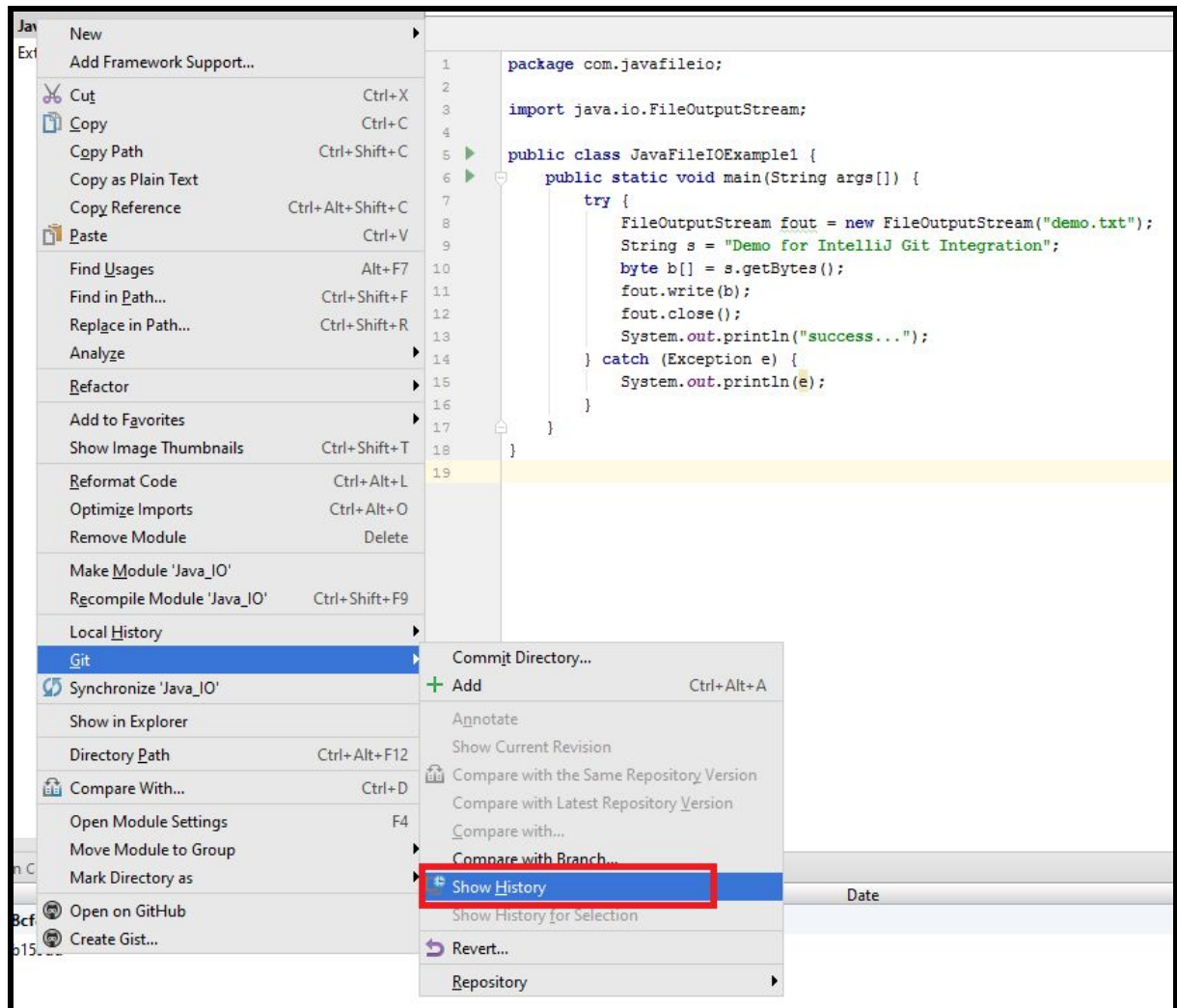
Styling with Markdown is supported Comment on this commit

History for details about the check in:

Local History,



Remote History,

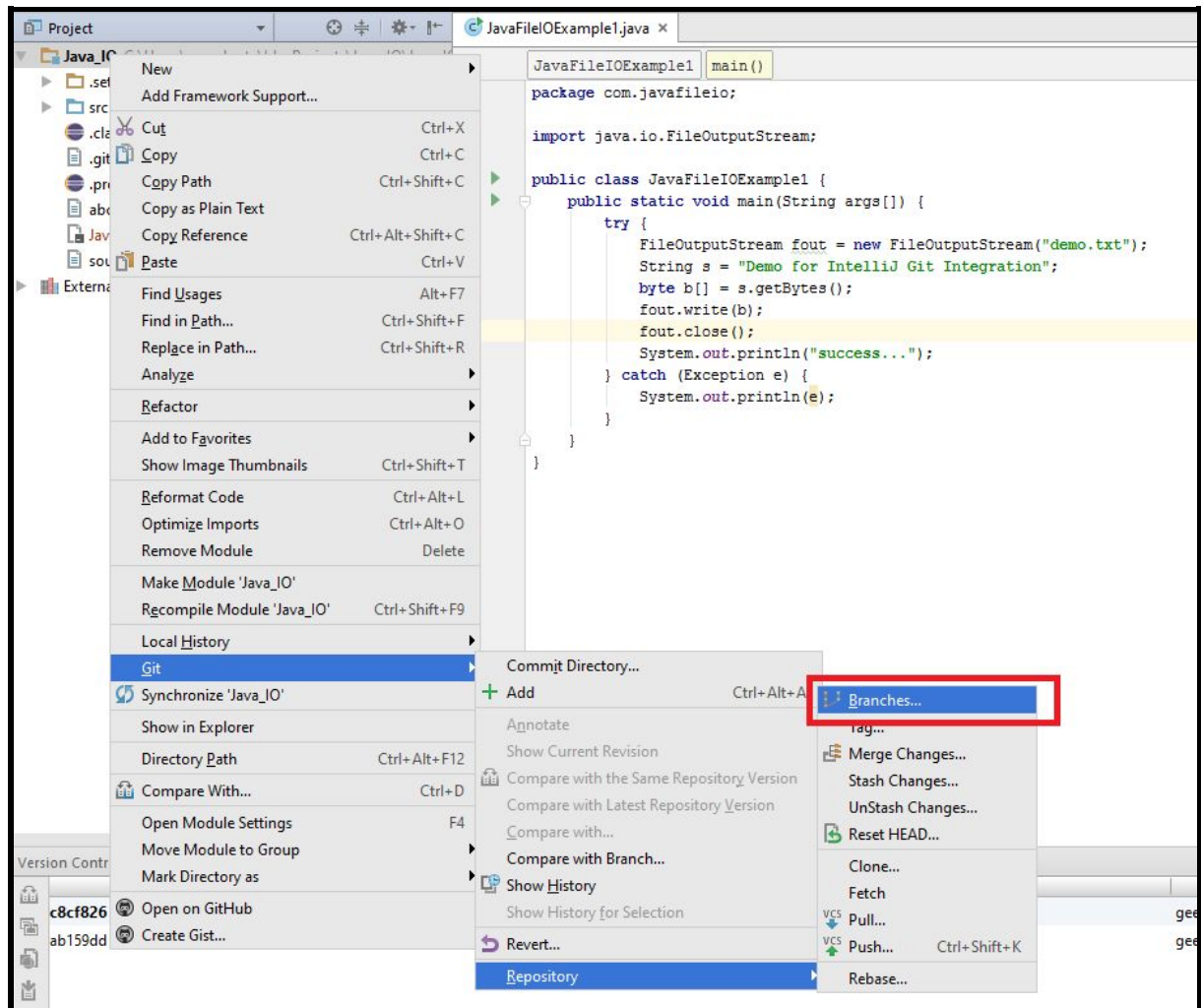


New Branch Creation:

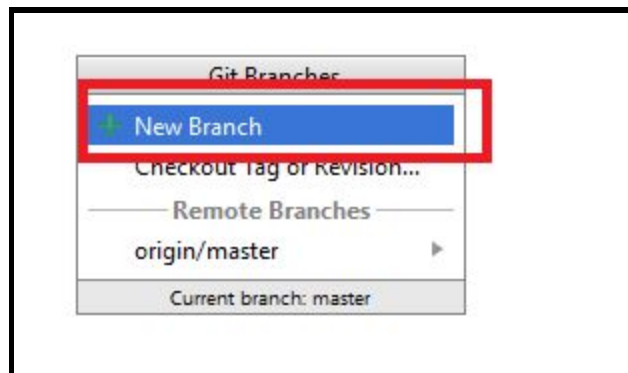
Local Branch Creation:

Please follow the below steps to create new local branch:

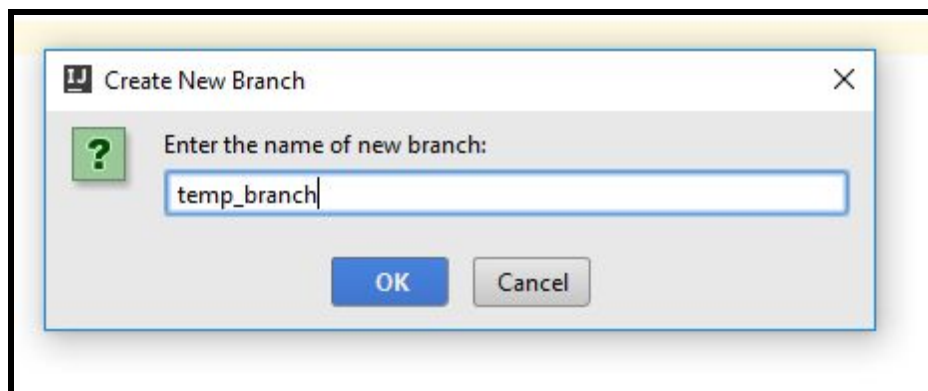
Click : Git -> Repository -> Branches



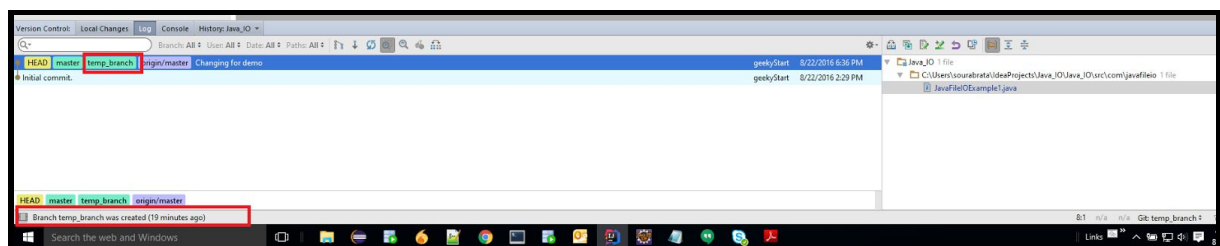
Click : New Branch



Give your branch a name:

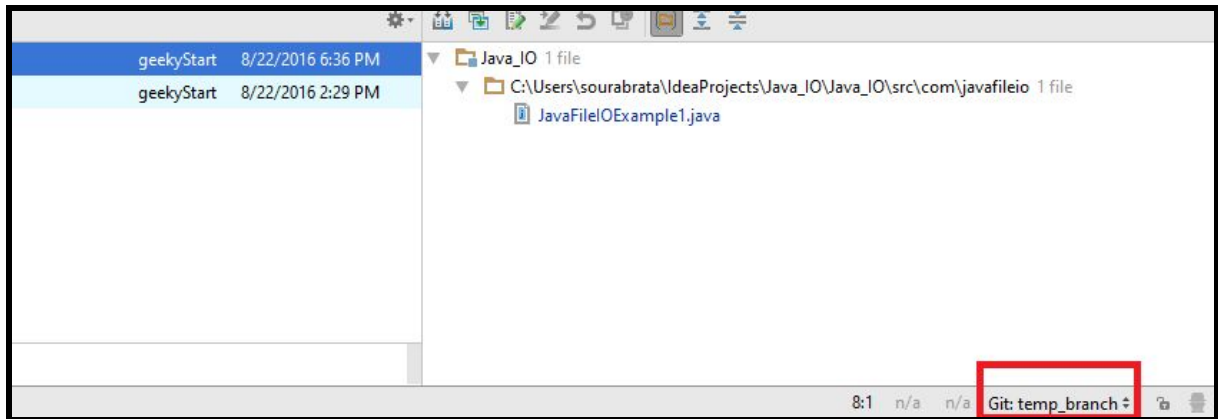


Now you can see, that your new branch has been created,

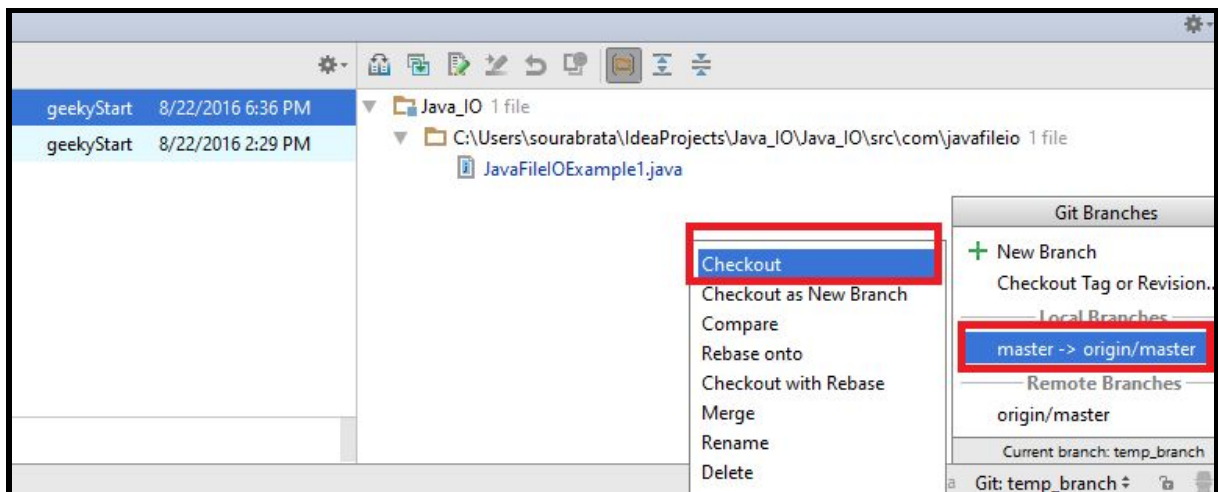


Switch between Branches:

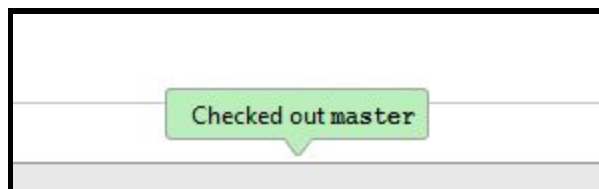
Now we are in “temp_branch”,



Click: origin/master -> Checkout

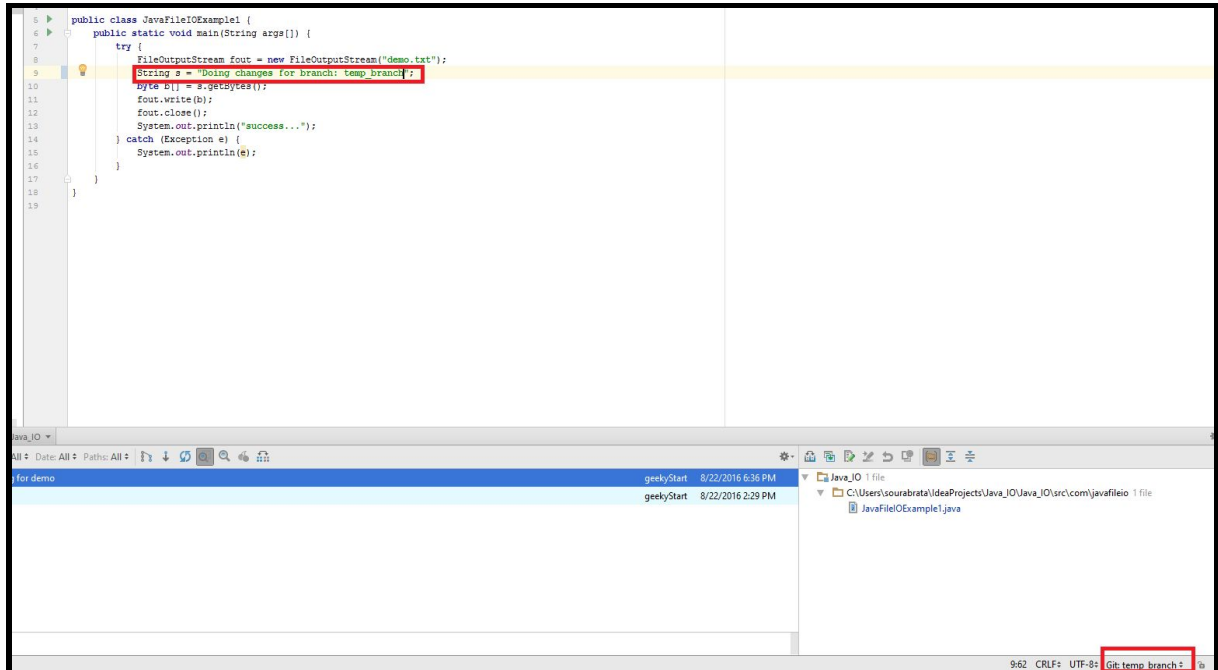


Now we successfully checked out to “master” branch,

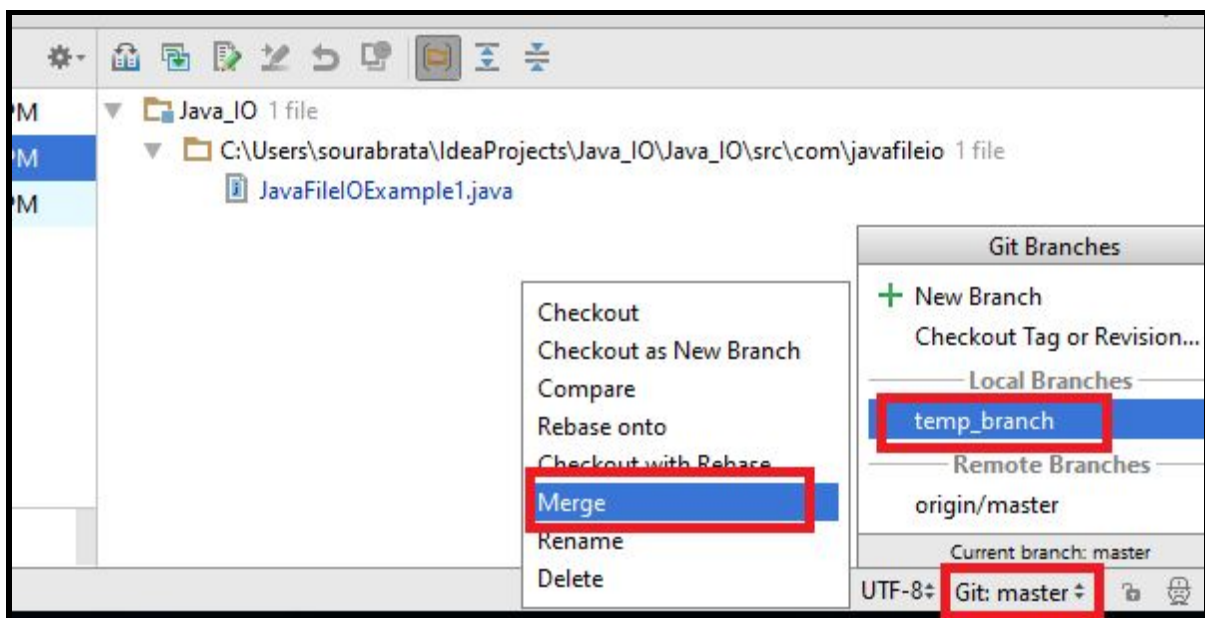


Made some changes in local branch and merge it to local master branch:

Doing some changes,

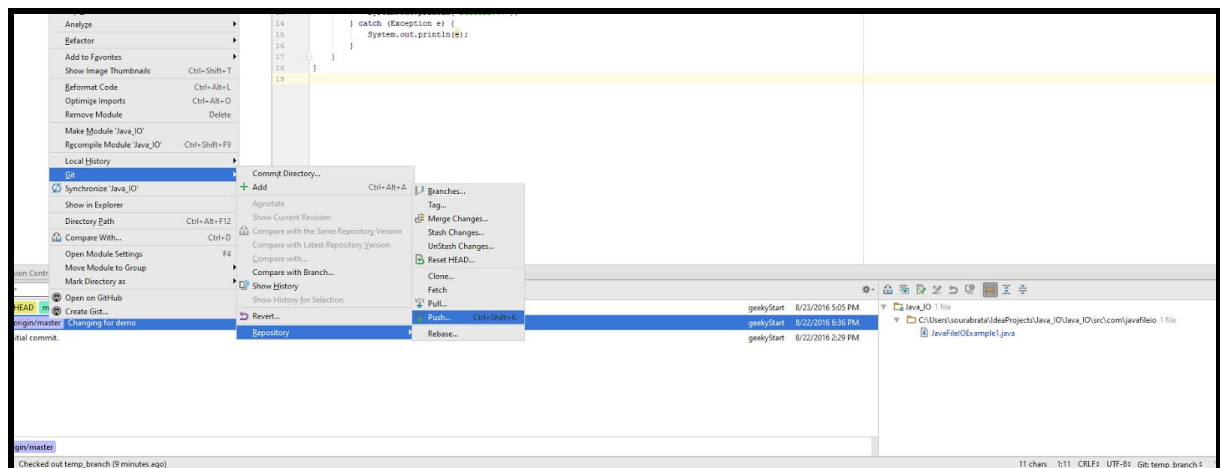


We are merging "temp_branch" changes to local "master" branch,
For that you need to be in master branch and **Click:** temp_branch -> Merge

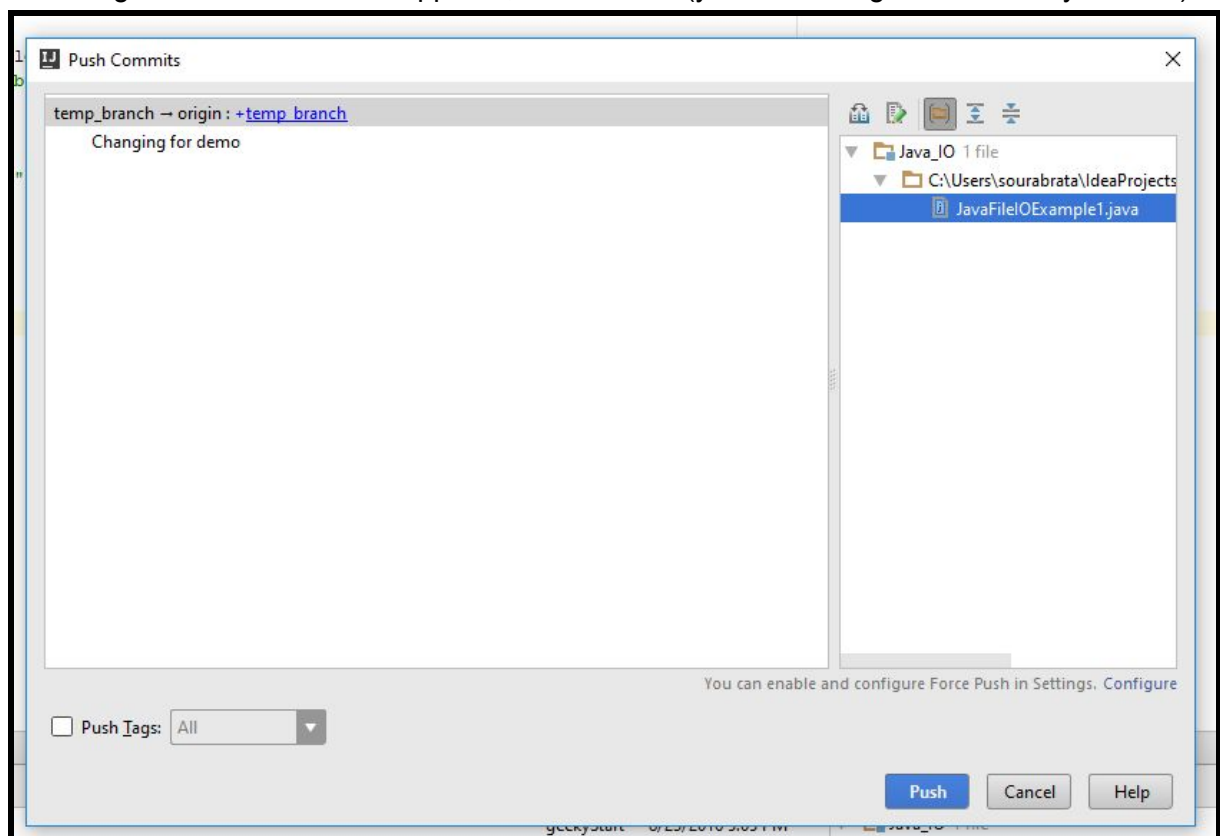


Push changes to a remote branch by creating a remote branch from local branch:

Follow below steps for the same,
Click: git -> Repository -> Push



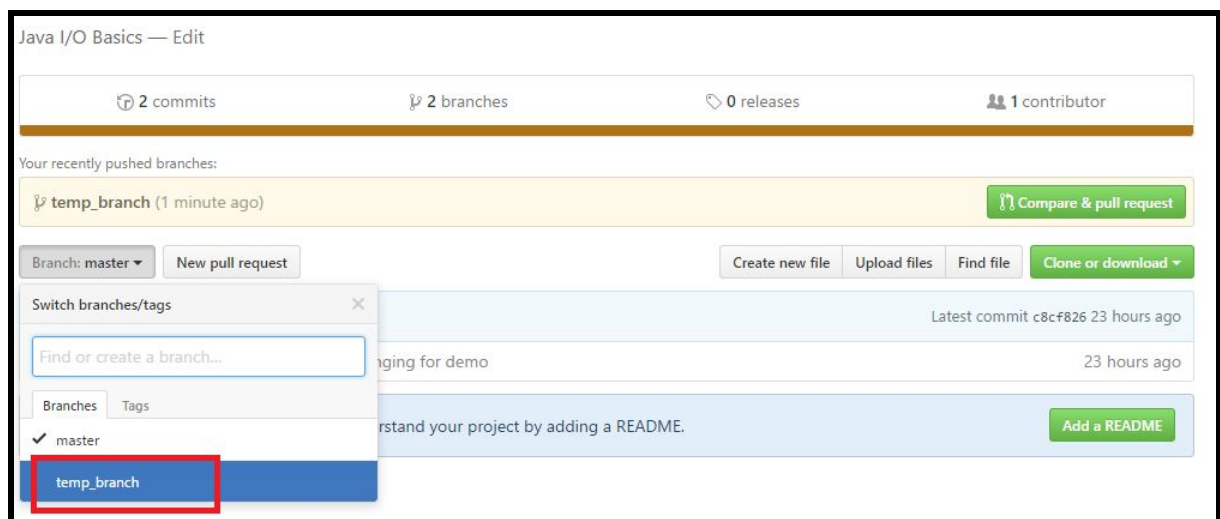
You will get the below window appears, **Click: Push** (you can change the name if you want)



You will get one message saying Push successful,

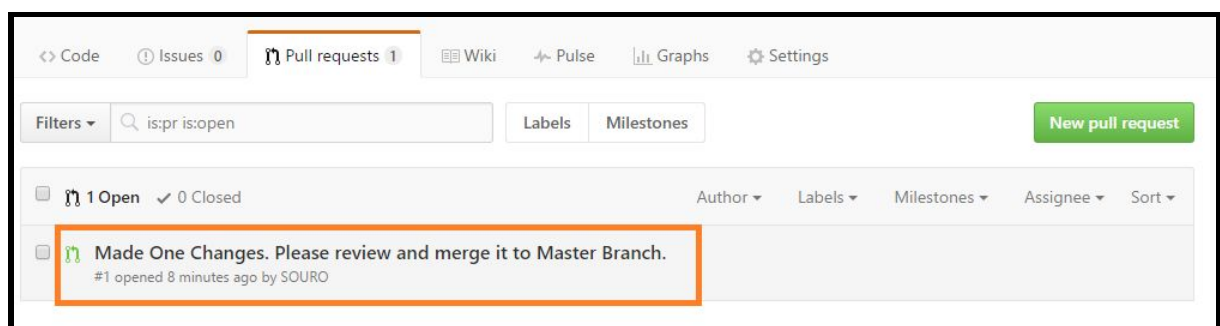


After that go to your Github account, you can see one remote branch got created by name “temp_branch”,



Pull Request to Review your changes:

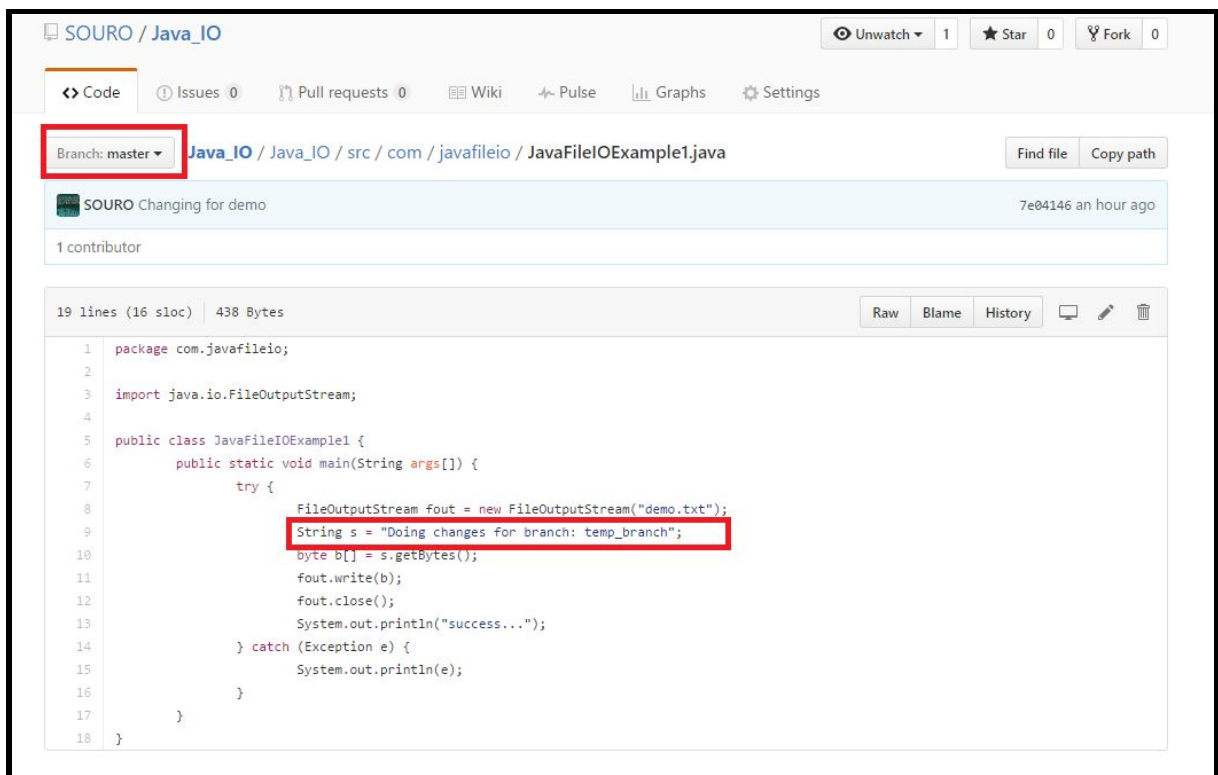
You can create one pull request and ask to review your changes before the concerned person decides to merge your changes to master (or some respective branch), For that, **Click:** New pull request, and put some relevant comment,



If there is no conflicts (between your changes and the master branch's content), it will show you the same message and ask you to merge it,



Once you do that, you can then visit your master branch and you will see that changes are getting reflected in master branch,

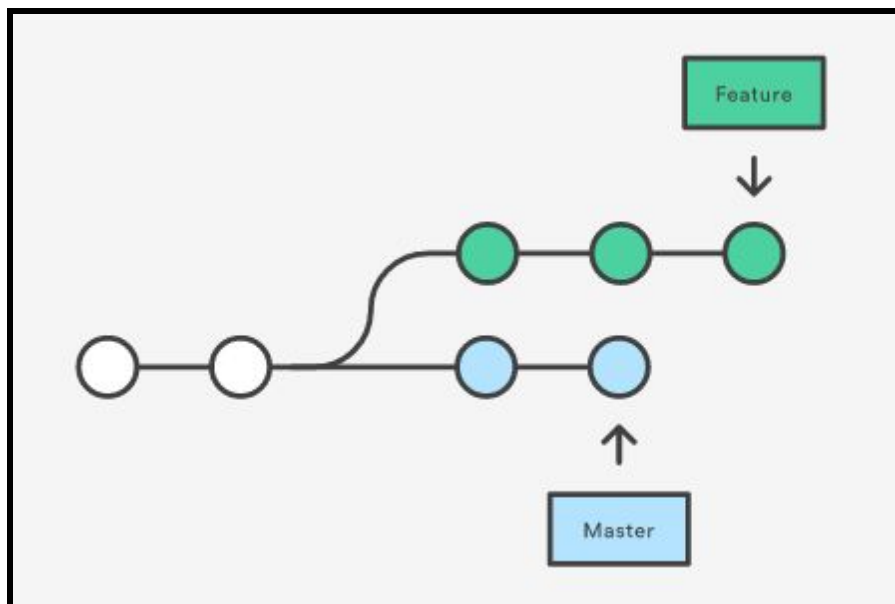


Note: Though you can merge the change automatically(in absence of conflicts), but don't do that if you are not asked for the same, may be it is someone else's task to do so and that person will take the decision to merge or not the same.

Rebase:

Now it may happen that whenever you are are creating one pull request from one of the branch to master, that time one conflict is coming and then you can't automatically merge it. You can deal the same by using rebase technique.

The first thing to understand about git rebase is that it solves the same problem as git merge. Both of these commands are designed to integrate changes from one branch into another branch—they just do it in very different ways.



So let's say, you have these branches in remote,

Master
Branch1
Branch2

All are having same content. Say,

Master : Content A
Branch1 : Content A
Branch2: Content A

Now you have changed something in Branch1 locally, committed it and pushed it to remote Branch1 and you create a pull request to master branch and merge the same.

Master : Content B

Branch1 : Content B

Branch2: Content A

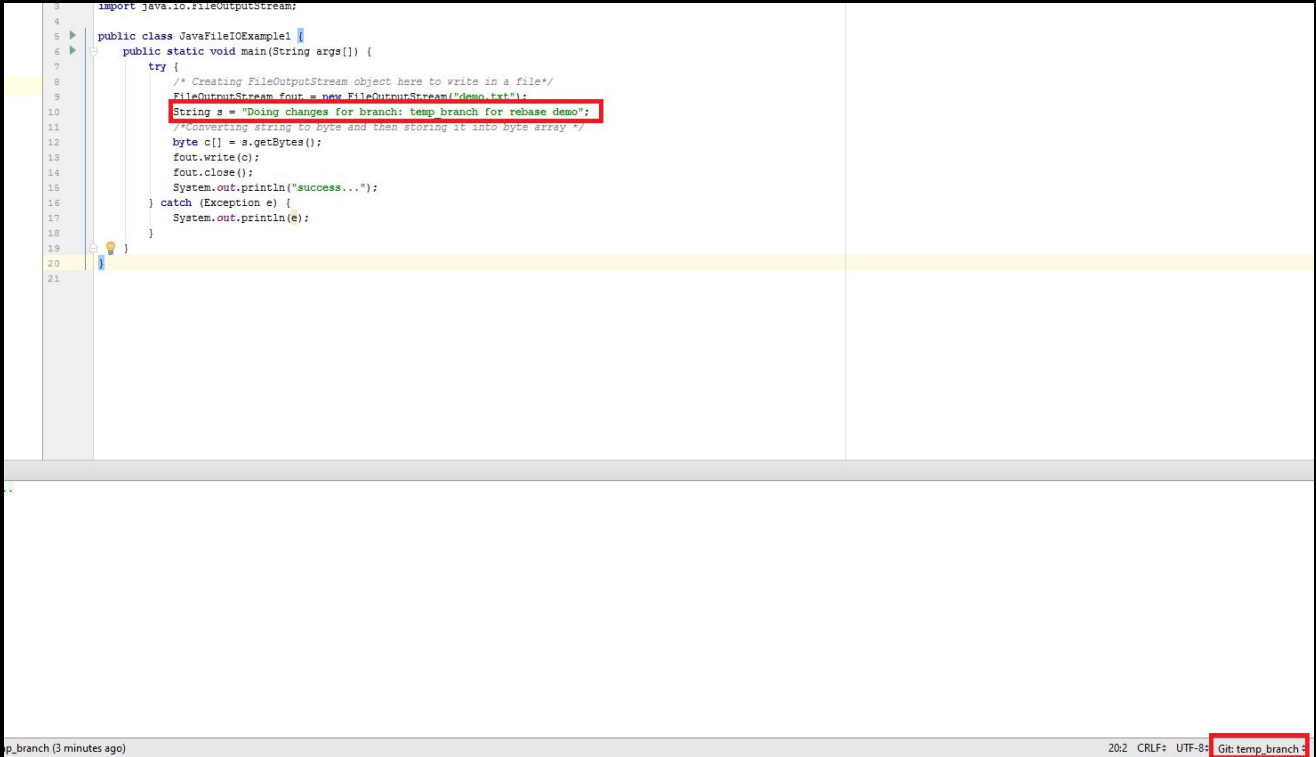
Now you made some changes in Branch2 and then if you create a pull request, then your pull request will be created but it will not be able to merge your changes due to some conflicts.

Because Master is having some changes that Branch2 is not having that and Branch2 is having some changes that Master branch is not having. This is one kind of scenario where conflicts will occur and you can't merge it automatically. There are some other scenarios also present for which conflicts can occur.

Now we can have a look what I have done for the same,

First master, temp_branch, temp_branch2 is having same content.

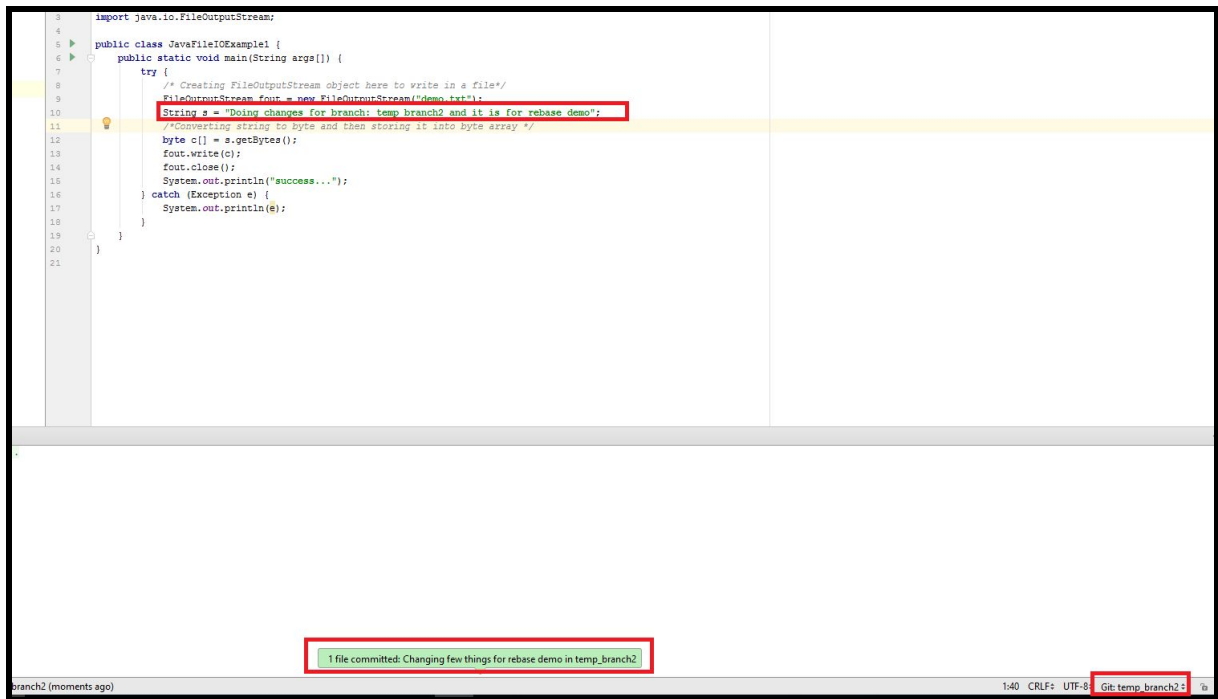
Then I made some changes in temp_branch locally, committed it locally and then push it to remote temp_branch and then create a pull request to master branch and merge it automatically.



```
3 import java.io.FileOutputStream;
4
5 public class JavaFileIOExample1 {
6     public static void main(String args[]) {
7         try {
8             /* Creating FileOutputStream object here to write in a file*/
9             FileOutputStream fout = new FileOutputStream("demo.txt");
10            String s = "Doing changes for branch: temp_branch for rebase demo:";
11            /*Converting string to byte and then storing it into byte array */
12            byte c[] = s.getBytes();
13            fout.write(c);
14            fout.close();
15            System.out.println("success...");
16        } catch (Exception e) {
17            System.out.println(e);
18        }
19    }
20 }
21
```

temp_branch (3 minutes ago) 20:2 CRLF UTF-8 Git: temp_branch

Now that master and temp_branch is having same content and temp_branch2 is having old content. I made some changes into temp_branch2, committed it locally, push it to remote temp_branch2,



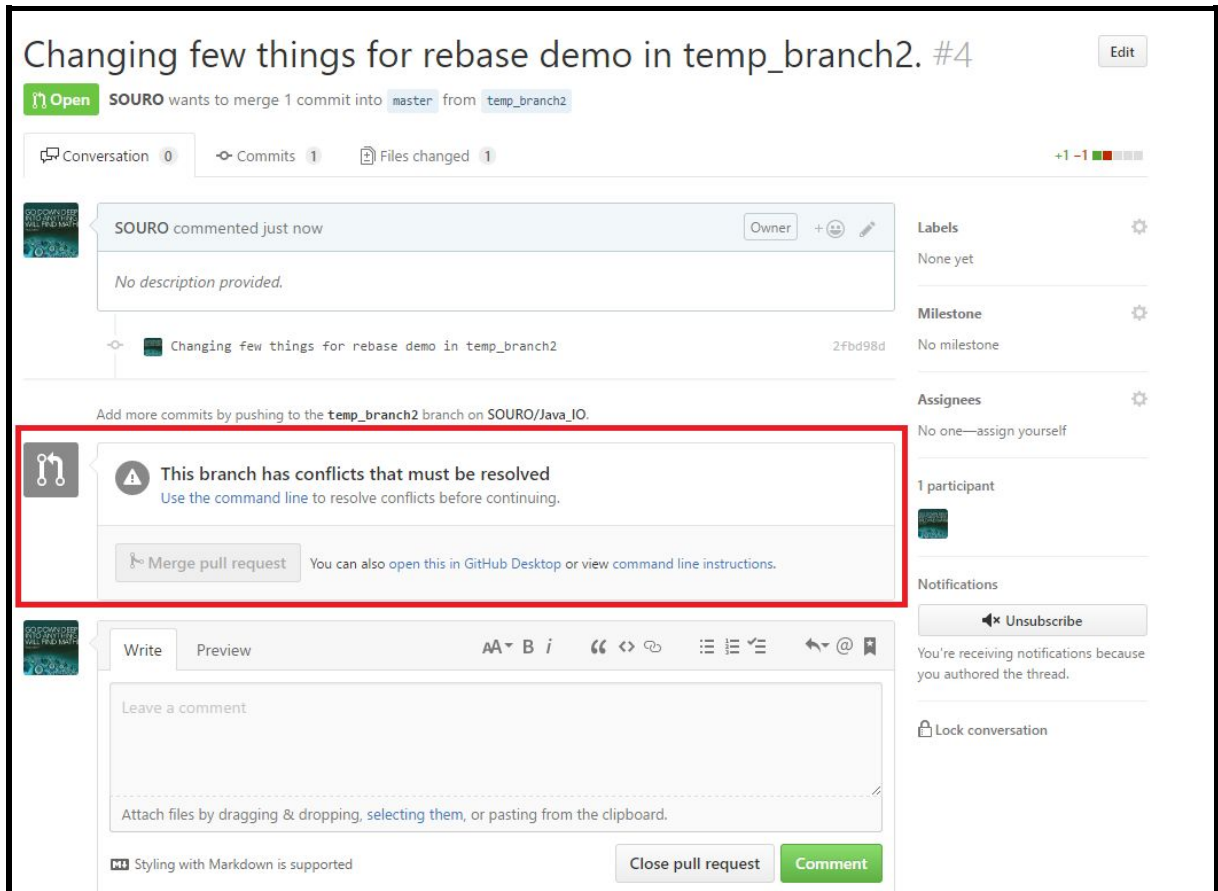
The screenshot shows an IDE with a Java file named `JavaFileIOExample1.java`. The code is as follows:

```
1 import java.io.FileOutputStream;
2
3 public class JavaFileIOExample1 {
4     public static void main(String args[]) {
5         try {
6             /* Creating FileOutputStream object here to write in a file*/
7             FileOutputStream fout = new FileOutputStream("demo.txt");
8             String s = "Doing changes for branch: temp_branch2 and it is for rebase demo";
9             /*Converting string to byte and then storing it into byte array */
10            byte c[] = s.getBytes();
11            fout.write(c);
12            fout.close();
13            System.out.println("success...");
14        } catch (Exception e) {
15            System.out.println(e);
16        }
17    }
18 }
19
20
21
```

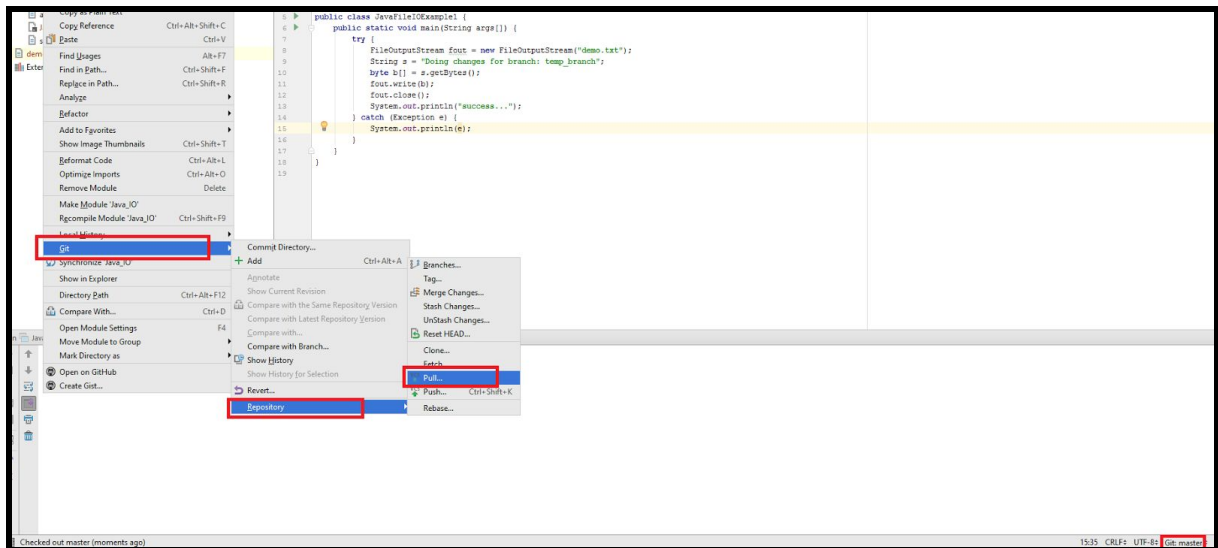
Below the code editor, a green box highlights the commit message: "1 file committed: Changing few things for rebase demo in temp_branch2".

At the bottom of the IDE, the status bar shows "branch2 (moments ago)" on the left, "1:40 CRLF UTF-8" in the center, and "Git: temp_branch2" on the right.

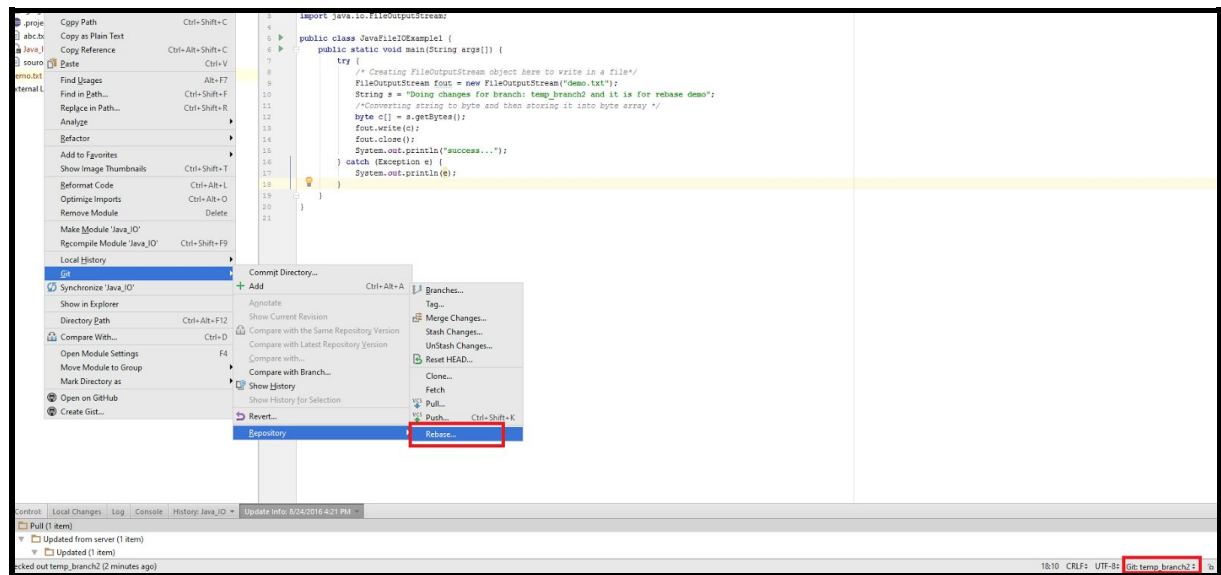
Now create a pull request to master branch but you can see I am not able to merge it automatically due to conflicts that occurs because of above stated scenario.



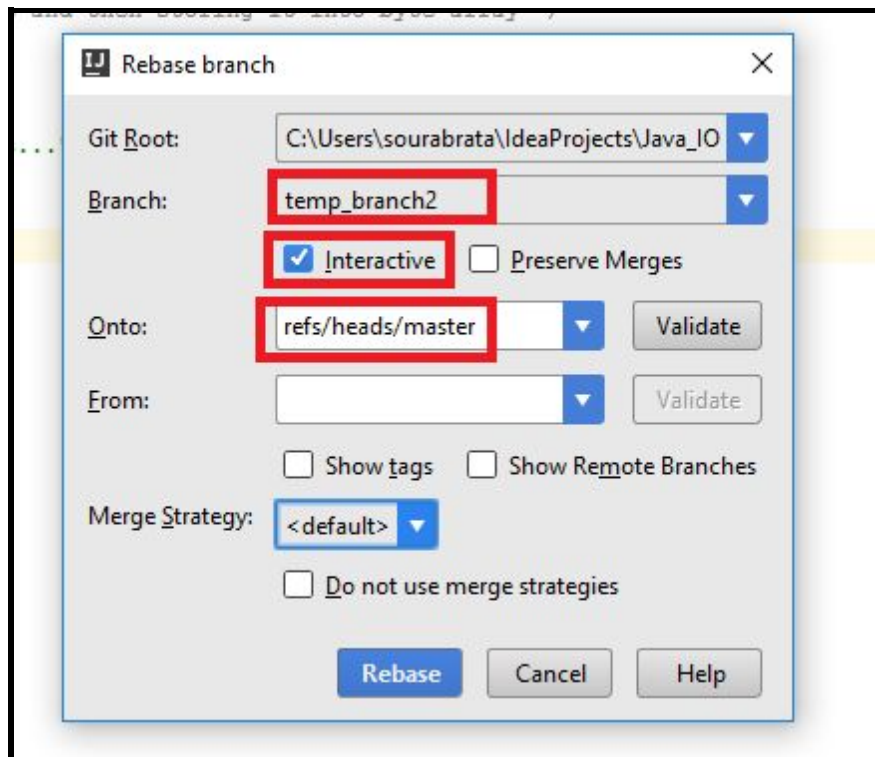
Now switch to master branch locally and take the latest code from remote master branch by doing pull,



Switch to temp_branch2 for which conflict was happening, and do the rebase,



Provide **onto** field's value to remote master branch,



You can select any of the following options,

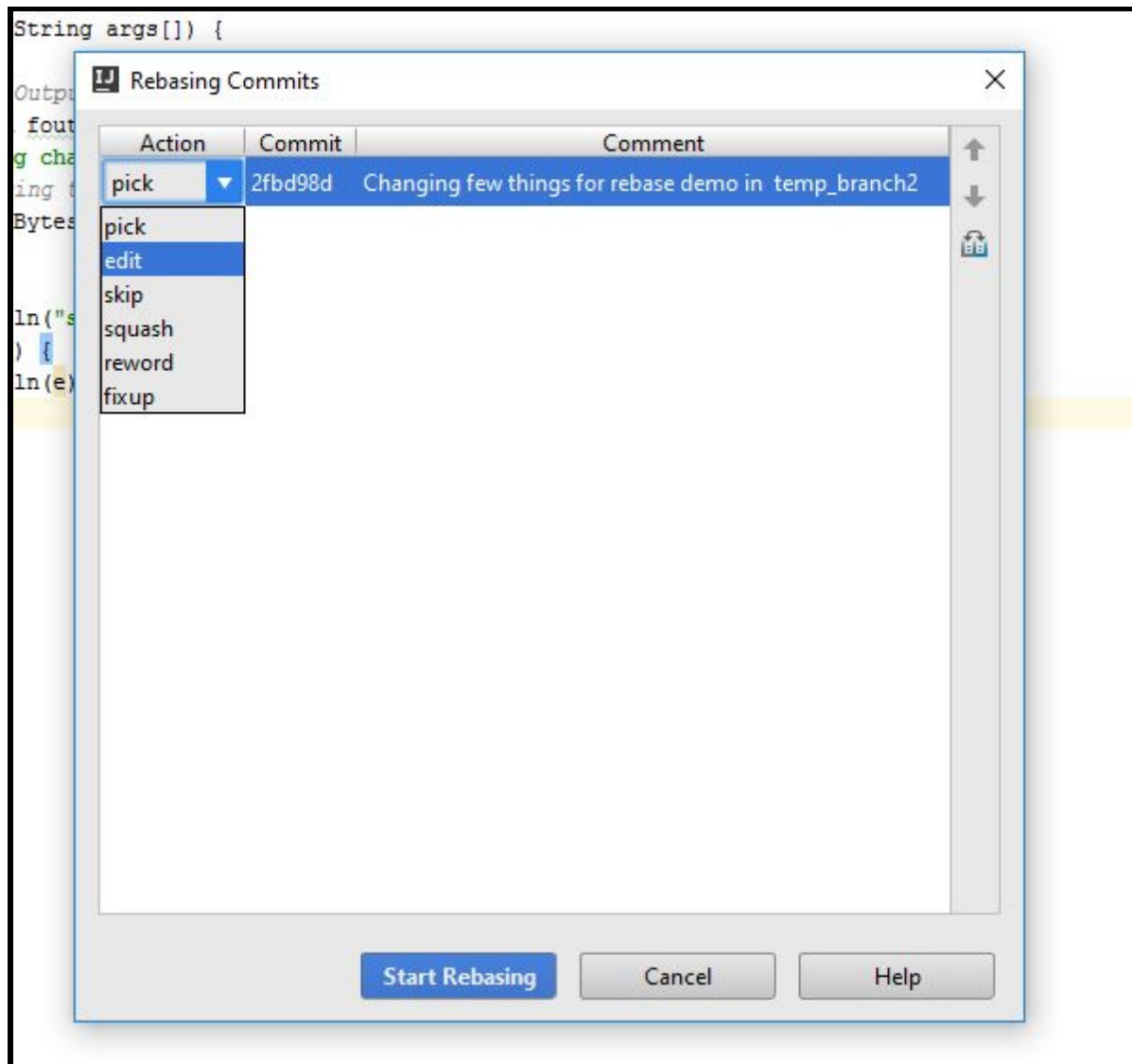
To apply a commit as is, select the **Pick** option.

To update a commit before applying, select the **Edit** option.

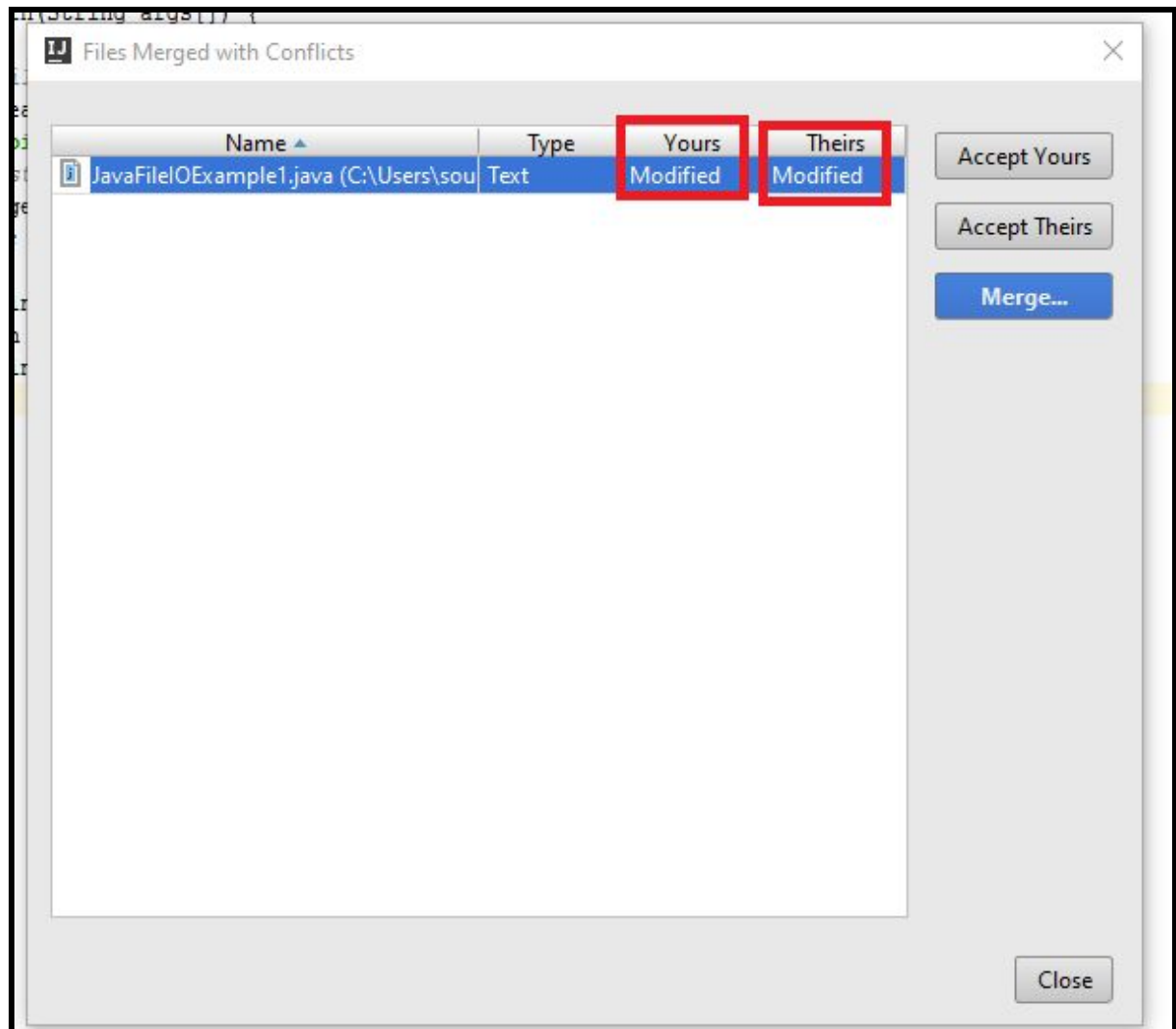
To ignore a commit, select the **Skip** option.

To combine a commit with the previous commit, select the **Squash** option.

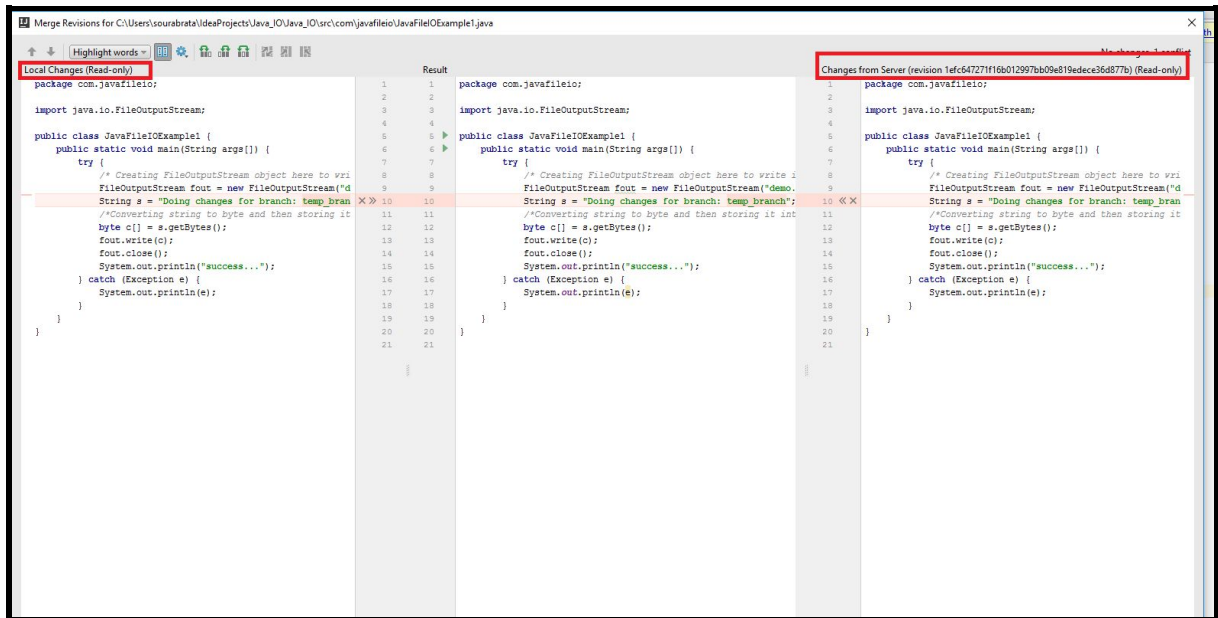
Do the needful as per your requirement. Here i have selected **Edit** option.



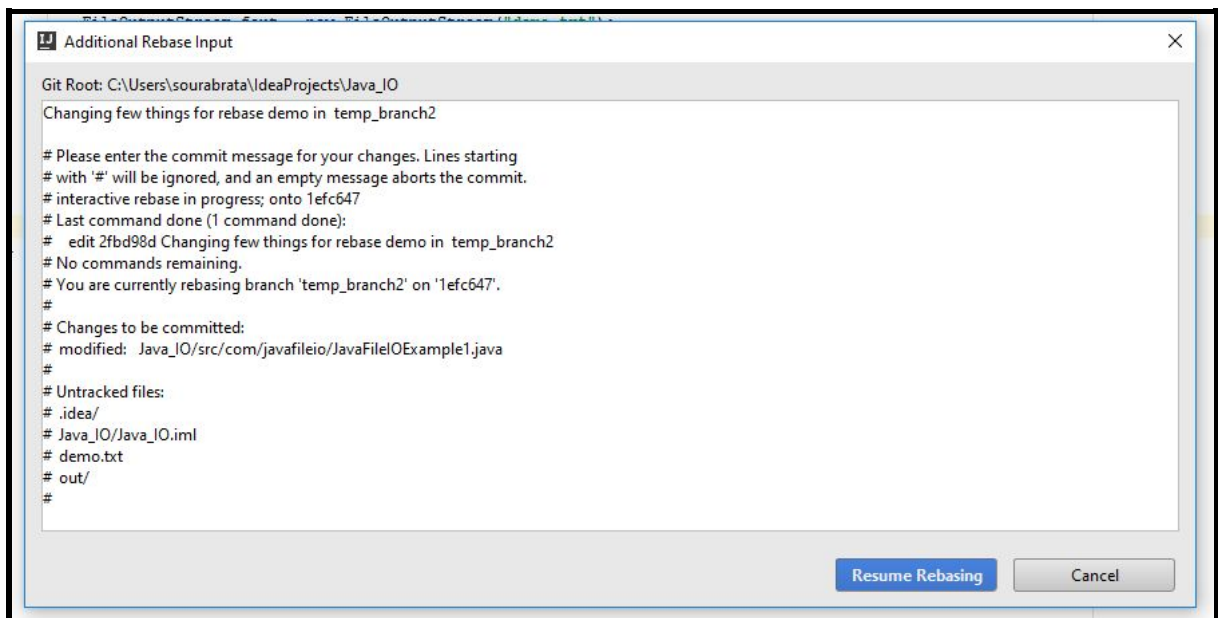
Double click on that blue highlighted row to solve the conflicts manually,



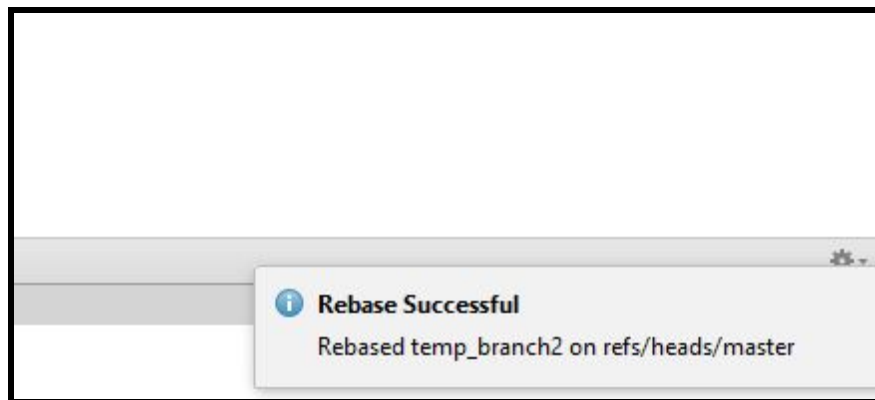
Solve the conflict in the below provided editor as per your requirement,



Resume rebase process,

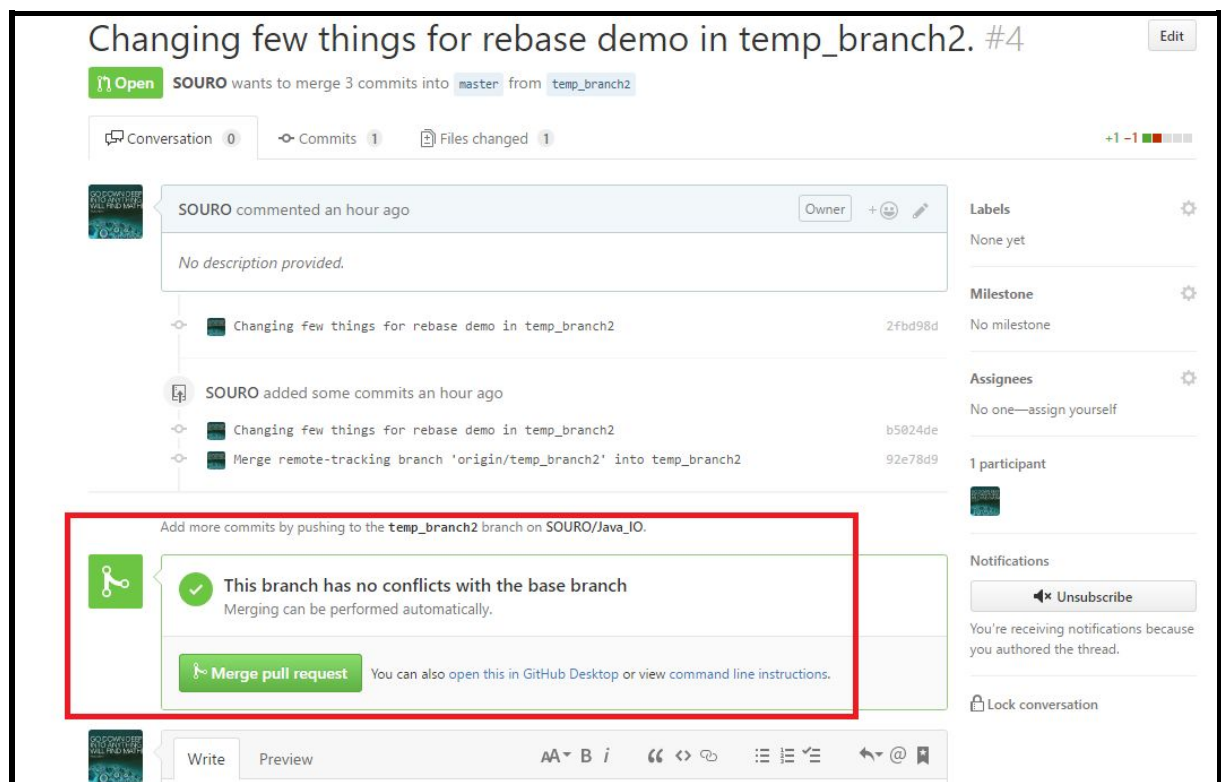


On completion you will get one successful message,



After that push your changes to remote.

Now go to your remote github account, you can see that you are now able to merge the pull request as conflict is already resolved,



SQUASH

Multiple commits in a single commit. so that it can be helpful for the reviewer to track your changes. To Help this situation, Github provides one feature call Squash commit.

In IntelliJ while you are doing rebase then from action select “squash” that will help you to do multiple commits in single.

