**Tutorial 9**

***Cloning a Repository***

In this tutorial we will explore *cloning.* In a previous tutorial we saw how we might have a local repository and want to *pull* changes someone else has made from the remote repository. However, suppose we don’t have any code locally, no repository and we want to grab all the code from a branch of a remote repository. The solution is to *clone* the remote repository. The *clone* command pulls all the code from a branch of the remote and creates a local repository. Then, you can do the normal things: modify, add, commit, push.

We will first create a branch on GitHub, then clone that branch locally, modify it locally, the push back to the branch.

This tutorial continues from Tutorial 8.

**Steps to Complete**

1. Login to GitHub if necessary and display your “test” repository.
2. Create a new branch with the name, “new\_feature” as shown on the right.
3. Note that the active branch is now *new\_feature*. Click the drop-down and it will display all the branches and indicate the active branch with a checkmark as shown on the right.
4. Edit *foo.txt* on the *new\_feature* branch by changing the first line to: “Monday-Friday”, type a commit title, and commit.
5. Use Cmder (or Window Explorer) and create a new (empty, not inside an existing repository) folder named, *gitex\_clone* and navigate into it.

**λ**  mkdir gitex\_clone

**λ**  cd gitex\_clone

1. Clone the *new\_feature* branch in the *test* repository. Substitute your username.

**λ**  git clone -b new\_feature https://github.com/YOUR\_USER\_NAME/test.git

Note:

* This command creates copy (clone) of the remote *new\_feature* branch in a Git repository in a local folder named *test.* Why it names it *test*, instead of *new\_feature,* I don’t know. However, as we see shortly, *Cmder* labels it as the *new\_feature* branch.
* You might wonder why didn’t use “origin”, our alias for the remote repository. For example, why was the command not:

**λ**  git clone -b new\_feature origin

The answer is that the folder *gitex\_clone*, where we issued the clone command, is not a repository. Thus, there is no config file. And, related, you can’t define “origin” for the same reason.

1. Examine the contents of the *gitex\_clone* folder. The result is shown on the right.

**λ**  dir

1. Navigate into the *test* folder and examine the contents. The result is shown on the right (your display will not show *bar.txt*, only *foo.txt*).

**λ**  cd test

**λ**  dir

1. Add this text to *foo.txt,* “Change from local clone”

**λ**  notepad foo.txt

1. Stage and commit

**λ**  git commit -a -m "More enhancements"

1. Now we are ready to push this back to the remote repository; however, we want to push to the *new\_feature* branch. Actually, we don’t have to do anything special because when we cloned, Git created a *remote tracking branch.* Theclone created a reference to the remote branch in the config file. Let’s see these items before by displaying the config file.



**λ**  git config -l

Notice the last few lines that define *origin* and the reference to the remote branch, as shown on the right.

**NOTE: the top arrow, “Definition of origin” in the figure is incorrect (as it is using SSH and we are using HTTPS). Yours will say:**

remote.origin.url=https://github.com/YOUR\_USER\_NAME/test.git

1. Push the changes to the remote repository.

**λ**  git push -u origin new\_feature

Remember, the *new\_feature* reference in the command above is referring to the name of the **local branch**. The *push* command uses the config file to know which branch to push to on the remote. In general, you can push to any remote branch with this syntax:

**λ**  git push -u origin local\_branch\_name:remote\_branch\_name

1. Go to GitHub and verify that *foo.txt* in the *new\_feature* branch shows the change, as shown on the right.
2. Switch to the *master branch.* Verify that *foo.txt* in the *master* branch has not been modified.
3. **Do the following:**
4. Make a screen shot similar to the one shown on the right. Make sure it shows your User ID and your name in the text.
5. Place the image in the *HW VCS* document in the appropriate place.
6. The image should easily readable without zooming in or out.

The next tutorial will pick up exactly where we finished here. Make a backup before proceeding if desired.