



CMP73010
Managing Software Development

Workshop 3
Distributed Version Management

Contents

DISTRIBUTED VERSION MANAGEMENT	3
Objectives.....	3
Overview	3
Review	3
A simple project example.....	4
GitHub in an IDE environment	4

Distributed Version Management

Objectives

On completion of this topic you should be able to:

1. Use GitHub to fork, change, and request pulls on GitHub
2. Use Visual Studio GitHub extension to fork, clone and modify a Visual Studio project on GitHub.
3. Use Visual Studio to commit changes, push to GitHub and request pull of you changes.

Overview

Distributed version management, as you have learned in the study guide, is managing system component versions across the Internet. In this workshop, we will look at GitHub, a popular version management system used in public domain systems and in private development projects. It is not the only product available but we will look at how it works in some detail, e.g. the Australian company Atlassian has several commercial products in this area (atlassian.com).

Some of the activities in this workshop will require you to create an account on GitHub. Since we will be using the public version you should be careful not to disclose personal data. For example, do not disclose your student ID in the system or in documents saved to the system.

Review

To start will review the version distributed version management terminology you have seen in the study guide. GitHub uses slightly different terminology but it should be easy to follow. The following table is a summary.

Study guide/Textbook	GitHub
Repository	Repository Usually called a “Repo”
Mainline	Master – also consider the main branch
Branch	Branch – in GitHub a “fork” operation creates a branch
Merge	Merge – this is more general, e.g. a fork can be merged with the master or any other branch
	Commit – save changes to your branch. Separate commits form a codeline
Push	Push – copy repository from user’s machine to the GitHub repository
Pull	Pull – take another branch and merge with your branch. Any branch may pull any branch. Pull request – a developer requests the master owner to merge with the master (mainline)

At this stage if you have not read the study guide or textbook you should do it to help with the reminder of this workshop.

A simple project example

We will now create a simple project in one account and go through the pull request and merge process. This is explained in the GitHub hello world tutorial. Note that there are other tutorials on the GitHub website that walk you through other more complicated processes. Make sure you read and understand the documentation in this tutorial.

In the on-line session accompanying this workshop your instructor will cover the same process except that the changes and pull request will be done in a separate GitHub account. This is the more usual distributed use of the system.



Activity 3-1: Example1: Acceptance tests

1. Sign up to GitHub and do the GitHub “Hello world” tutorial:

<https://guides.github.com/activities/hello-world/>

2. In your account, fork the *barrywilks/CMP73010-Simple* project. Change the file and do a pull request (push operation). Check your list of pull requests to see how GitHub records them (note that your request will not be pulled).
3. Go to the following GitHub project:

<https://github.com/Scanate/EthList>

This is a crowdsourced document on the Ethereum blockchain system. This is use of GitHub to allow the public to maintain a html document.

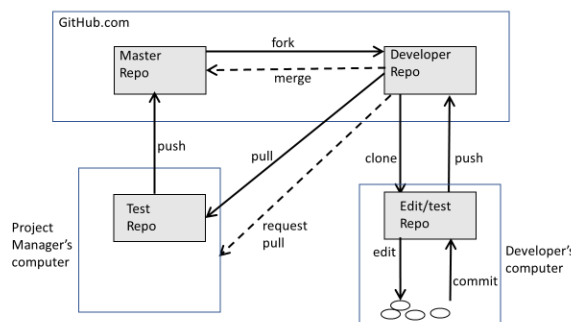
- a. Click on the “Contributors” link to see how real people have been involved in pull requests
- b. If you are lucky, you may see an active pull request. Have a look at it. The owner of the master branch must decide if it should be merged.

GitHub in an IDE environment

In this section, we will look at a simple Visual Studio Project which whose master repository is hosted on GitHub. We have a simple Visual studio project. This is different than the documentation projects in the previous sections because we will need to compile the source code before we can execute it. The following steps are optional! If you feel you do not want to do this “hands-on” you will be able to see the process in the recorded workshop.

The following diagram can be used as a reference:

GitHub/Visual Studio Terminology



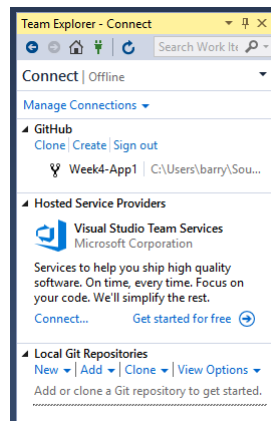
The following steps fork a Visual studio project so you can modify it with Visual Studio. The final step does a push request.


Step 1: install the GitHub extension for Visual studio. this can be achieved using the following steps:

- Select “Tools/Extensions and updates”
- Click the “Online” menu (left hand side)
- In the search box (top right) type “GitHub”
- Select and install “GitHub extension for Visual Studio”
- The install process will require you to exit visual studio. It will take a little while to download and install the extension.
- Restart Visual studio

Step 2: Enter your GitHub credentials so Visual Studio will sign on for you (a credentials screen may pop up if you attempt any other operation). This may happen after the next step.

- Select the menu “Team/Manage Connections...”
- The team explorer window will appear in your IDE similar to the following.




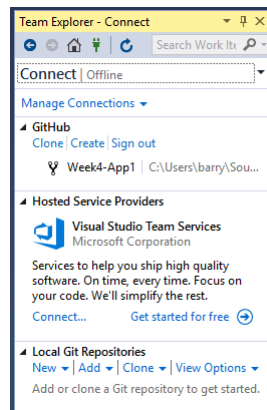
- Click on the home button () and a list of options will appear:
- Click on the “Settings” button and then click “Global Settings”. enter your GitHub account details

Step 3: Fork the test project barrywilks/week4-App1 in your GitHub account.

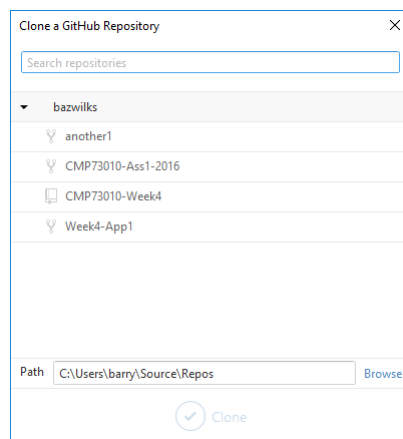
- Using your browser, sign onto GitHub (yes, you must sign on in the browser *and* in Visual Studio)
- Search for the repository barrywilks/week4-App1
- Fork this repository.

Step 4: Back in Visual Studio, clone the project so it is moved to your workstation

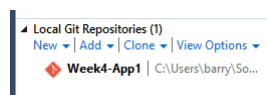
- In the Team Explorer window, click on the Manage Connections button (). Your team explorer will look like the following. This has all the options opened. The “GitHub” option shows your projects in you remote GitHub account. The “Hosted Services” are the Microsoft Team Services which is a Microsoft product that we are not using. The “Local Git Resources” are empty, but this is where we will be cloning the repository from your GitHub account.



- Click on your forked project in the link in the “GitHub” area and then click the “Clone” link. If necessary, you may need to click on the small triangle next to the label to expand the display. This will display a dialog box:




- You may have one GitHub project or you may have many as shown above. Select the project called week4-App1 forked in step 3.
- You next need to specify where to save the project on your computer. In the same dialog box this is the text field under the list of your GitHub projects. If the project already exists then you will need a new location (as in the above picture). However, if this is your first cloned project you can save it in the default location.
- Click the “clone” button in the dialog box to start the operation. It will take a little while to copy the files from your GitHub account
- Your local project will appear in the “Local Git Repositories”:

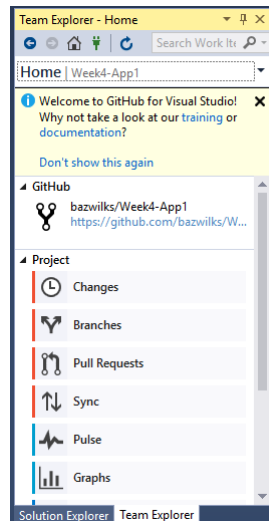


Step 5: Make changes and test them

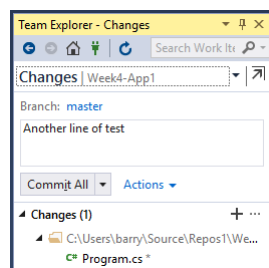
- Switch to the normal Solution Explorer window (using tab at bottom of Team Explorer window. You should recognize this as the window you used for C# programming.
- Double click on the “.sln” file to open the project as a Visual Studio C# project.
- Click on the **Program.cs** file to open the program. You should be able to read the code
- Run the program to check the output.
- Add an extra line of output by inserting a `Console.WriteLine(“some stuff”)` call
- Run to make sure it works

Step 6: Commit your changes – This remembers the state of the code on your local machine,

- Switch to Team Explorer by clicking the tab at the bottom of the Solution Explorer currently displayed.
- Click on the home button () if necessary to display the new list of option available:




- Click the “Changes” button and enter a description of the change, say “Another line of text”. The dialog will appear similar to the following.

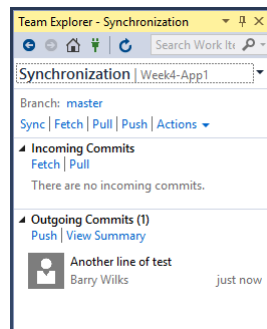


Notice how all of the files that have changed since the last commit are listed to help you verify the changes.

- Click on “Commit All” button to complete the commit operation. Note that the button will not be clickable unless you change something in Step 5.
- A dialog box will appear for you to confirm the commit operation.


Step 7: Push your changes to your GitHub repository. “Push” means copy a committed version from your local repository to your GitHub repository.

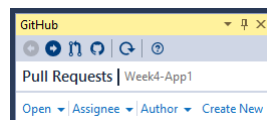
- Go back to the home menu, e.g. home button ()
- Click on the “Sync” option. There will be a dialog like the following. Notice how your commits are listed at the bottom of the dialog.



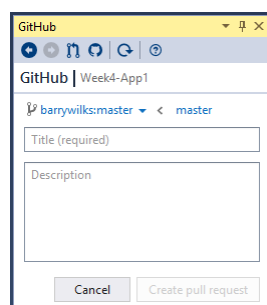
- Click on the commit you wish to push to your GitHub account.
- Click on the “push” link above the list of commits.
- There will be a delay and a progress bar displayed as the files are copied to GitHub.

Step 8: Do a “Pull Request” so master owner can consider your changes currently on GitHub. Note that you can do a Pull Request on your GitHub account or with Visual studio. You can also view other people’s Pull Requests, leave comments, discuss, etc.

- Go back to the home menu, e.g. home button ()
- Click on the “Pull Request” button. A dialog appears:

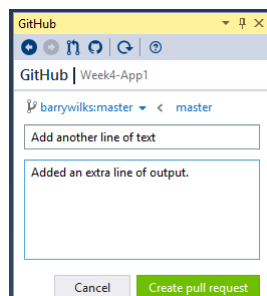


- Click on the “Create New” link. A dialog similar to the following will appear (it will contain your information)

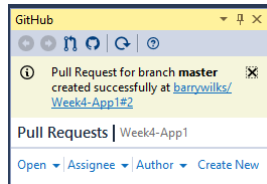


Notice how the master project is shown with a small arrow from your version on GitHub. In large projects, there will be several options for the source and the destination.

- Now enter a title for your proposal and some detailed description, e.g.



- Click on “Crete pull request”. After a short while the status of the request is returned, e.g.



Activity 2-2: Visual Studio interaction with GitHub

1. Fork the GitHub project into your GitHub account as shown above.
2. Clone the repository onto your local machine
3. Run the program to make sure it works.
4. Make a change to the program, e.g. as above.
5. Commit your changes (this is still your local version)
6. Push a working repository onto your GitHub account
7. Do a pull request to request a merge of your changes. (your changes will not be merged but the on-line tutorial will merge some changes so you can see from the master repository user how this works.)

This process is not assessed in this unit. However, the general strategy of distributed version management in tools like GitHub is required. If you decide to do the above exercise it will help you understand the process better than just reading/viewing it.