CMPSC 475

Assignment 0: Workflow setup

Due: 12pm, January 13

Description

The purpose of this assignment is to set up the workflow we will be using for project management throughout the semester. This process involves a number of steps that must be completed in order, exactly as specified.

All your projects will be contained in a single *Git Repository* hosted on GitHub. This setup will allow you to access your project from any machine and will also serve as the means by which you submit your project and we give you feedback. In this preliminary assignment you will initialize your local and remote source control repositories, set up your credentials in Xcode, create a *Bot* to verify that your project runs on the server, and learn the basic workflow for developing and submitting projects. Except for entering your credentials (which you should only need to do once), every assignment will involve most of these steps. Understanding them and performing them correctly is the goal of this assignment.

You must be on a Mac running Monterey (macOS 12.1) with Xcode (13) and Git (2.32) installed. Do not use a beta version of Git.

Walk Through

1. If you do not already have a GitHub account, go to github.com and create one. If you have one, log into it now. It's helpful later on to already be signed in to your account.

2. Open the Terminal app and configure Git with your name, email and .gitignore file:

   ```
   git config --global user.name "Your Name Here"
   git config --global user.email youraccessid@psu.edu
   git config --global core.excludesfile ~/.gitignore_global
   ```

   Copy the provided gitignore file (`gitignore_global.txt` in Module 1 on Canvas) file into your home directory. For example, you can use:

   ```
   mv ~/Downloads/gitignore_global.txt ~/.gitignore_global
   ```

   Be sure to include the leading '.' in the destination file name, as above.

3. Link your GitHub account to your classroom repository: In a browser, go to https://classroom.github.com/a/E_33nvnb and authorize GitHub classroom. Then select your name on the roster and accept the assignment CMPSC 475 Spring22. Github will then configure your class repository. You need to refresh your browser to see the result. Click on the repository link.

4. Set up your project repository using the instructions under the "*…or create a new repository on the command line*" heading:

   1. Open up Terminal app and create a new folder in which you will create all your projects.
   2. Use cd to go to this new folder.
   3. Run the Git commands provided by GitHub. All projects for this course should be created in this folder. You'll use the one repository for the entire semester.

5. Start Xcode: Either click on the Launchpad icon in the dock (it looks like 3x3 grid of squares) or tap the Spotlight magnifying class in the upper right corner. Begin typing

"Xcode" and you will narrow the choices with each character you type. When just Xcode is left you can simply hit return. Or you can just ask Siri to open Xcode. The "Welcome to Xcode" window will appear. (Note: you must be running Xcode 13.x)

6. Add your various credentials to Xcode. You can perform this step later.
    1. If you don't already have an Apple ID, go to https://appleid.apple.com/account and create one.
    2. Tap the Xcode icon in the dock to make Xcode the foreground application. (The Xcode menu will now appear in the top left of the screen.)
    3. Select *Xcode>Preferences…* and select the Accounts Tab
    4. Click on the "+" symbol in the lower-left corner and select *Apple ID*.
    5. Enter your Apple ID and password. Click **Add**.
    6. Lastly add your GitHub credentials. Click on the "+" symbol in the lower-left corner and select *GitHub.* You may need to add a personal access token to your Github account with the 'repo' scope. Follow this guide: https://docs.github.com/en/github/authenticating-to-github/creating-a-personal-access-token
    7. If you plan on working on multiple machines export your credentials, again using a selection from the cogwheel. You can then later import them on another machine
7. Create your first project in Xcode.
    1. Select *File>New>Project…* or Key shortcut ⇧⌘N
    2. Select **iOS** in the top bar and then select **App** under the Application section and click *Next*.
    3. Give your project a name such as *Workflow*. The project name will eventually appear under the app icon on the device springboard. Always choose something meaningful (i.e., not "HW 0") and nothing too long (else it will get truncated when it appears on the device). Select Personal Team for Team selection. For the Organization identifier use reverse domain name notation: *edu.psu.AccessID*. (Use your Penn State Access ID.) For Interface, choose *SwiftUI*. For Life Cycle, choose *SwiftUI App*. For Language, choose *Swift*. Make sure the *Use Core Data* box is unchecked and the *Include Tests* box is checked. Click *Next*.
    4. Set the destination to be the folder created in Step 3 above (e.g., CMPSC475). The Source Control option should be grayed out as Xcode detects that this folder is already under source control. Click *Create*. Be sure you perform this step exactly as given.
    5. Build and run your: *Product>Run* (or ⌘R). You may be asked for your password because Developer Tools Access needs to take control of another process for debugging to continue. Once you've done this the iPhone simulator should display a white screen.
8. Add an app icon.
    1. From the File Navigator select Assets.xcassets under the Workflow folder to reveal it in the editor pane. Select the AppIcon from the sidebar.
    2. Create an icon in png format in two sizes - 120x120 and 180x180. Typical names are icon@2x.png and icon@3x.png, designating their resolution. You can use the Preview

app on OS X to crop and resize any image. No need to be too creative. A simple "W" will suffice.

- You can use https://appicon.co/ to generate all required image sizes for an icon set.

3. Drag these icons onto appropriate placeholders (iPhone App iOS 7-14). You must always include icons for all appropriate devices and versions of iOS.

9. Commit & push your project to GitHub. You should get in the habit of doing this often.
    1. Select *Source Control>Commit*. Enter a commit message in the text box. In general, the message should be some indication of what you've changed since your last commit. Click the *Commit* button in the lower right corner. All your changes have now been committed *locally*.
    2. Select *Source Control>Push*. You may be asked to enter your name and password for the repository if you have not added your Github credentials. The selected remote should be *origin/main.* Click on the *Push* button. Your latest commit has now been *pushed* to your private GitHub repository. You can also push at the same time as you commit by checking the push box during the commit dialog.
10. Create and Switch to a new branch under Source Control. Initially your project is on the main branch. New branches allow you to create a distinct branch of your project, often used when performing development work. You will use them for all your development work prior to submission.
    1. Select the Source Control Navigator (⌘2).
    2. Create a new branch by clicking the cogwheel in the lower left, selecting *Branch from "main"* and then naming the new branch. *Workflow* would be a good name.
    3. Make a change to your project by editing a Swift file, adding an image, etc. You might need to return to the File Navigator first (⌘1). Commit your changes.
    4. Next you will merge your modified branch back into the main branch. We always do this by first switching back to the main branch and then merging into this branch. Return to the Source Control Navigator, select the main branch and then choose *Checkou*t either by right-clicking (or Control-clicking) on it or clicking on the cogwheel. (Checking out causes Xcode to switch branches). After this, click on Workflow so that it is highlighted blue, but you are still on the main branch. Next use the cogwheel to select *Merge Workflow into main*.
    5. Push your project. All submissions should be pushed from the main branch. This is the only branch I will be looking at.
11. Next you will configure the local Xcode Server and create a bot. Bots are processes run by the Xcode server to perform integrations on the current version of a project in a repository. We'll be using them to verify that your project builds successfully on the server. Follow the instructions on the *Continuous Integration* handout. **Important**: You need access to an account with admin privileges to perform this step. If you are working on a machine with such access (e.g., a machine in a computer lab), you will not be able to perform this step.
12. Congratulations. You've completed the workflow for creating, building, and submitting projects. For all future assignments follow the same workflow:
    1. Checkout or Pull the current version of your repository. (It's always a good idea to do this especially when you are working from multiple machines.)

2. Add a new project to the folder under source control (*File>New>Project…*), but only if you are starting a new app. For some assignments, you'll be extending your project from the previous week. In these cases ***do not*** add a new project.
3. Create a new source control branch off of the main branch. This is where you will do your development work. Never do development work on the main branch.
4. Build and test your project. Get in the habit of committing and pushing your code frequently. I *will* reduce your score if I see only a single commit/push.
5. When you are ready to submit your project merge your branch into main. Be sure to do this by first checking out the main branch and then merging your working branch into main. Delete any old Bot from previous projects and create a new Bot. Commit and push your project to Github and run an integration to ensure your project builds on the server. When I pull your projects I will only pull your main branch.