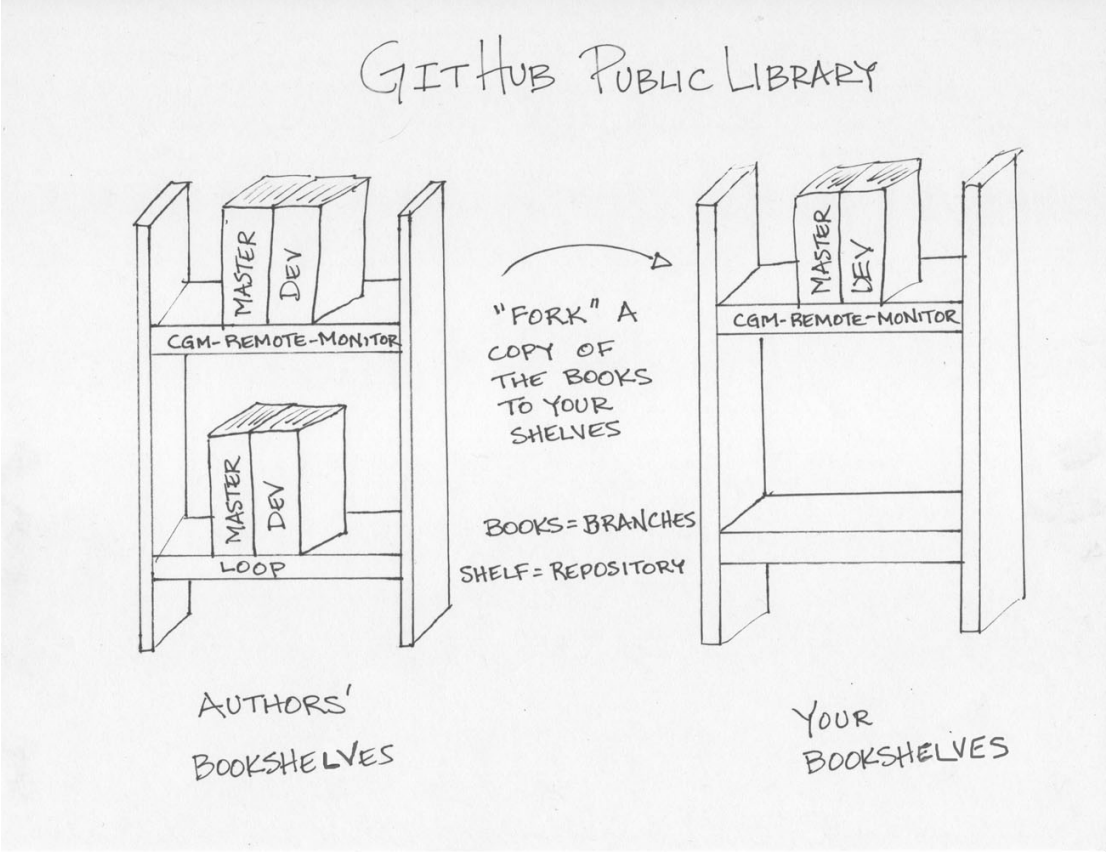
GitHub for New Users

Think of GitHub kind of like a public library. The authors put nice printed books up there to share (let's call the book a **master branch**). Now, sometimes the authors want to add chapters to the printed book. But it's a bit cumbersome to reprint books for every change. Or sometimes they just want to work in pencil for a bit and try new ideas. So they make notations in their own xerox copy of the book (**dev branch**). The books are kept side by side on the same shelf (**repository**).

You come to the library and want to use the books. The library will let you make (**fork**) your very own xerox copy of the books and keep them on your own shelf. When you fork, you get all the copies of the books on the shelf. In other words, you get all the branches of the repository.



If the author adds new notes to the handwritten parts (dev update) or reprints the book (master update), you won't see them in your books on your shelf unless you specifically ask to see them. Usually, you start this process by doing a **Compare** of your book to the author's book. GitHub will tell you what changes the author has made and you can make a **Pull Request** to bring that information over to your book. You then **Merge** those changes into your book. The whole process of comparing, pulling, merging is called **updating your repository**.

Having things in your GitHub repository is great, but we have to build Loop from our local computers. So there's some steps that you can do to make it a little quicker each time you update Loop. You'll be keeping a local copy of your GitHub's Loop branches on your computer and using Commit-Push-Pull in Xcode to save changes across all the various locations. It isn't as hard as it seems…just follow along.

FIRST TIME BUILDERS

If this is your first time trying to build with Loop, there's an option for building Loop called "direct download". You can still do this. It's quick and easy and doesn't require a GitHub account. Go to the guide if you want to do that option and start from there. Don't look here ☺

However, if you'd like to make your updating life just a little bit easier, you can use Xcode to help save changes back up to your Github account so that you don't have to redo customizations each build. If you'd like that…just follow along with this guide by starting at Fork. (You can skip Compare, Pull Request and Merge as a first time builder because your Loop pull will be current after you fork for the first time.)

FIRST TIME UPDATERS

If you already built Loop through the direct-download method, but never created a GitHub account or forked the repository… Fork with the first time builders above.

If this is not your first time building AND you forked before, skip the Fork section here and go ahead to Compare, Pull Request and Merge. You'll need to update your branches before building so that you have all the shiny new features since you first forked Loop to your Github account.

NOTE to both types of Loop Updaters: When you get to the steps about changing the MAIN_APP_BUNDLE_IDENTIFIER…if you reuse the same one that you used on your iPhone's existing Loop, you won't have to enter in all the basal rates, carb ratios, etc. again. The new Loop app will update the old one and save all the previous Loop settings with it. If you change the MAIN_APP_BUNDLE_IDENTIFIER to something new, Xcode will be installing a brand new Loop app and you will have to redo all the settings entries.

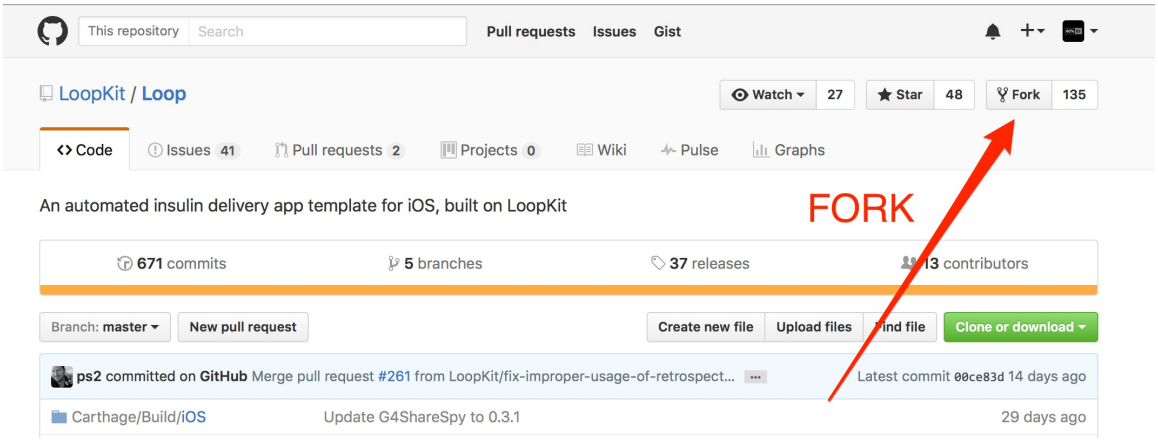
FORK

You'll be forking this repository

Loop
https://github.com/LoopKit/Loop

Create and/or Login to your GitHub account
Go to the link above
Press the fork button

You now have a Loop repository in your account
You can skip the Compare, Pull Request and Merge…move forward to the "How To Use XCode and GitHub together"



COMPARE

If you've had Loop for awhile, you may want to update as new versions are released. To update, you first need to have GitHub compare the changes between your GitHub copies and the author's current versions.

Using the GitHub compare tool sometimes goes in the wrong direction (you'd be trying to replace the author's pages with your pages if you go the wrong direction)…so using direct URLs keeps that from happening.

It's pretty easy to use direct URLs to compare. Copy and paste the URLs below, but replace "yourgithubname" obviously.

LOOP MASTER BRANCH
https://github.com/yourgithubname/Loop/compare/master…LoopKit:master

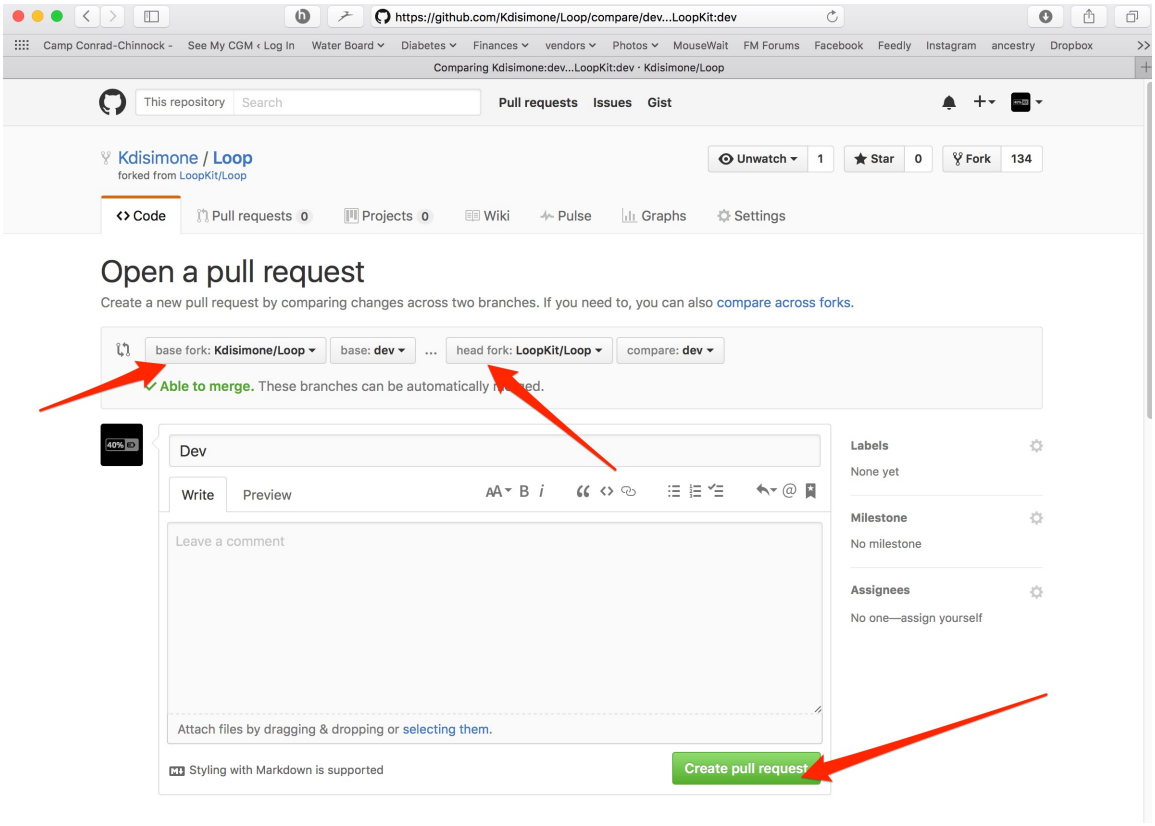
LOOP DEV BRANCH
https://github.com/yourgithubname/Loop/compare/dev…LoopKit:dev

Once you perform a Compare, GitHub is going to offer you to open a Pull Request.

Keep going with that.

PULL REQUEST and MERGE

Make sure your name is in the base fork and the author's name is in the head fork. Check that you are updating the branch (master or dev) that you'd like to. If you used the direct URLs in the previous section, you shouldn't have any problems. Give the pull request a title if it's not already named (doesn't matter what) and then click the green "Create Pull Request"

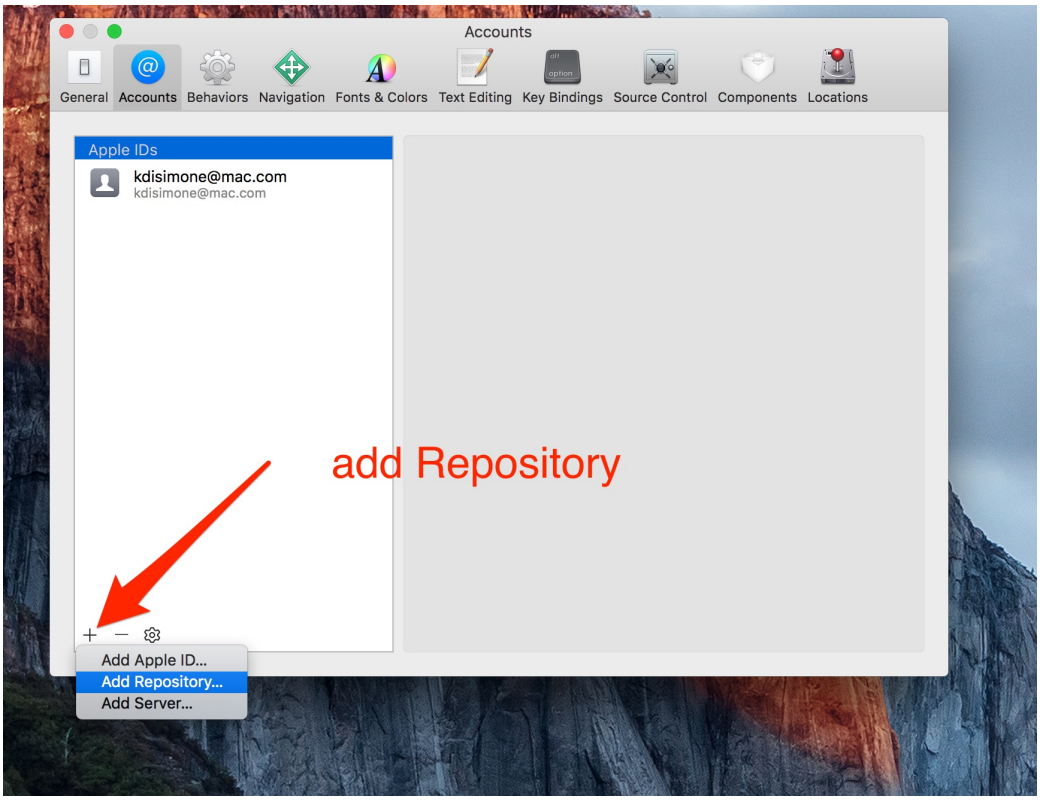


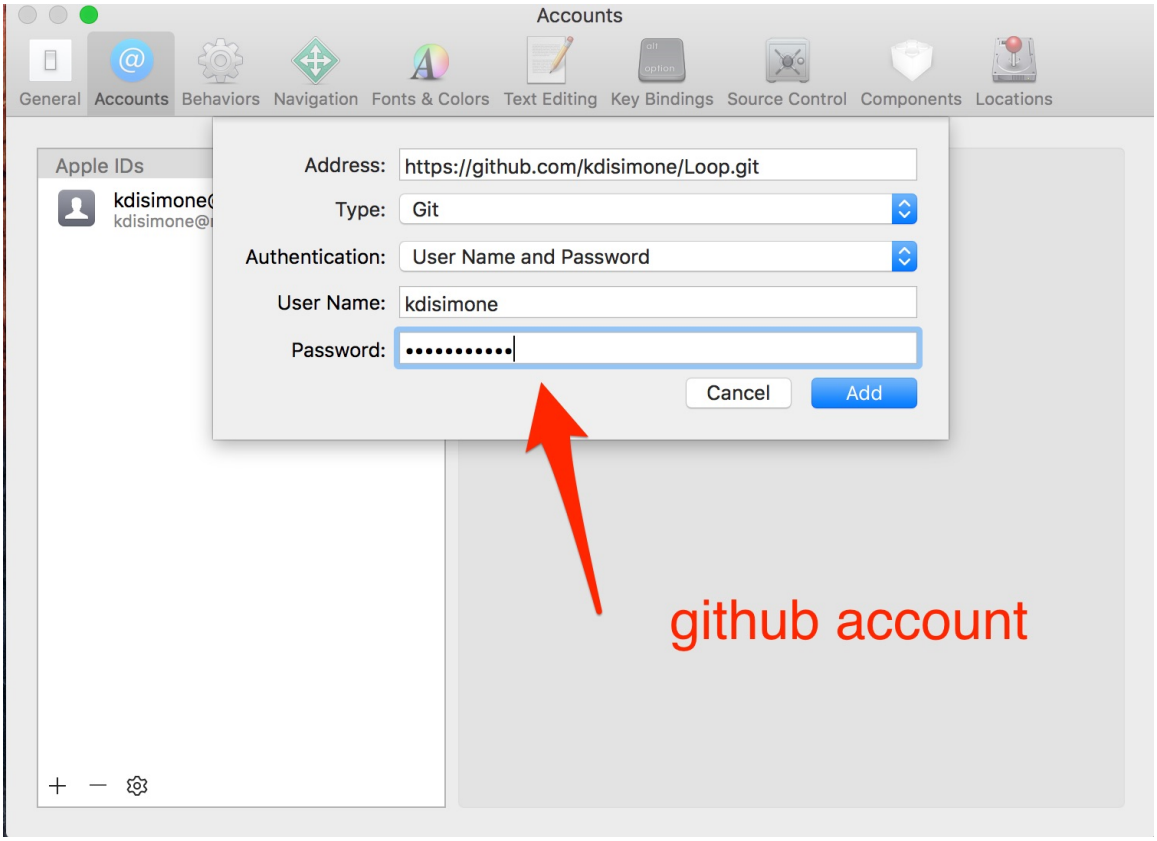
Scroll down and click the green "Merge Pull Request". Then click the green "Confirm Merge". There you just updated your selected branch on your GitHub repo.

Remember, updating one branch does NOT update the other branches in the repository. You need to do this for each branch specifically and separately.

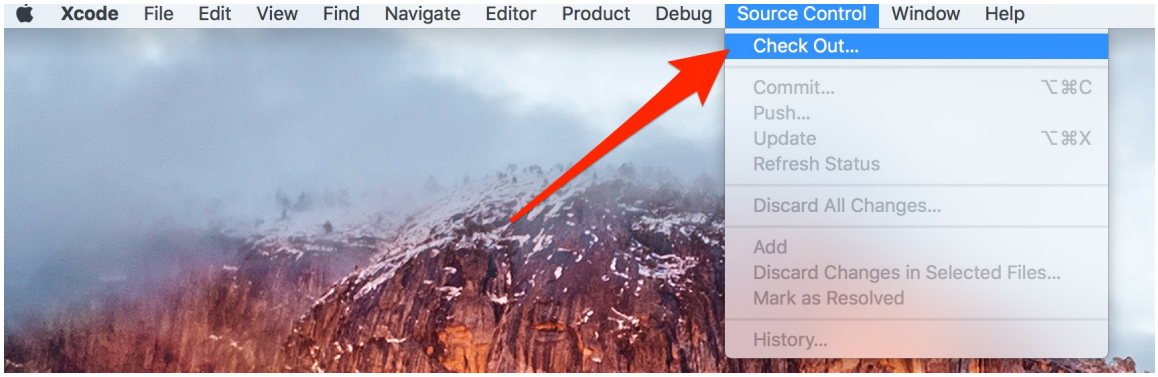
How to use XCode and GitHub together

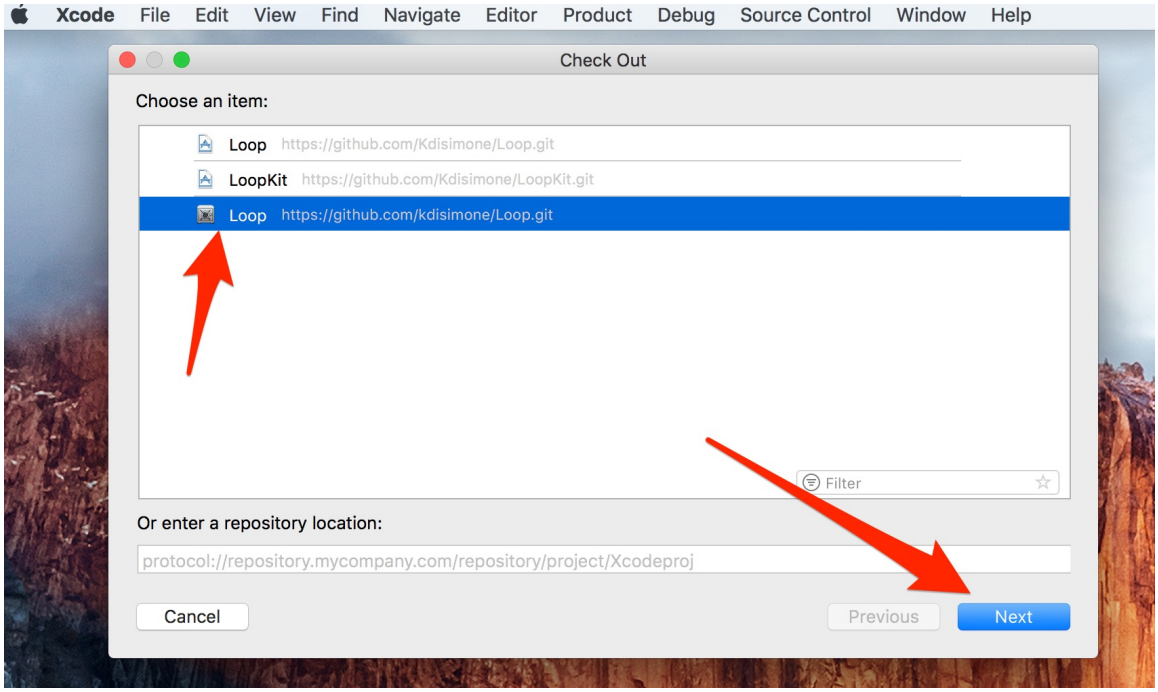
1. Add your updated Loop repository to XCode using XCode preferences

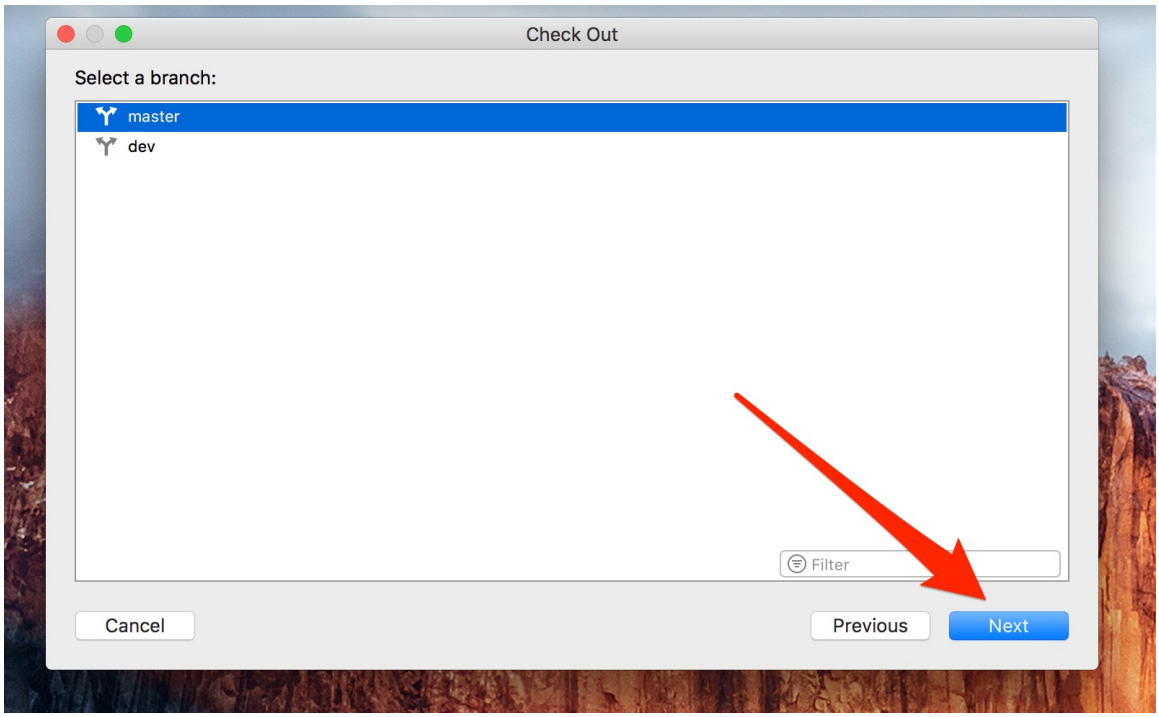




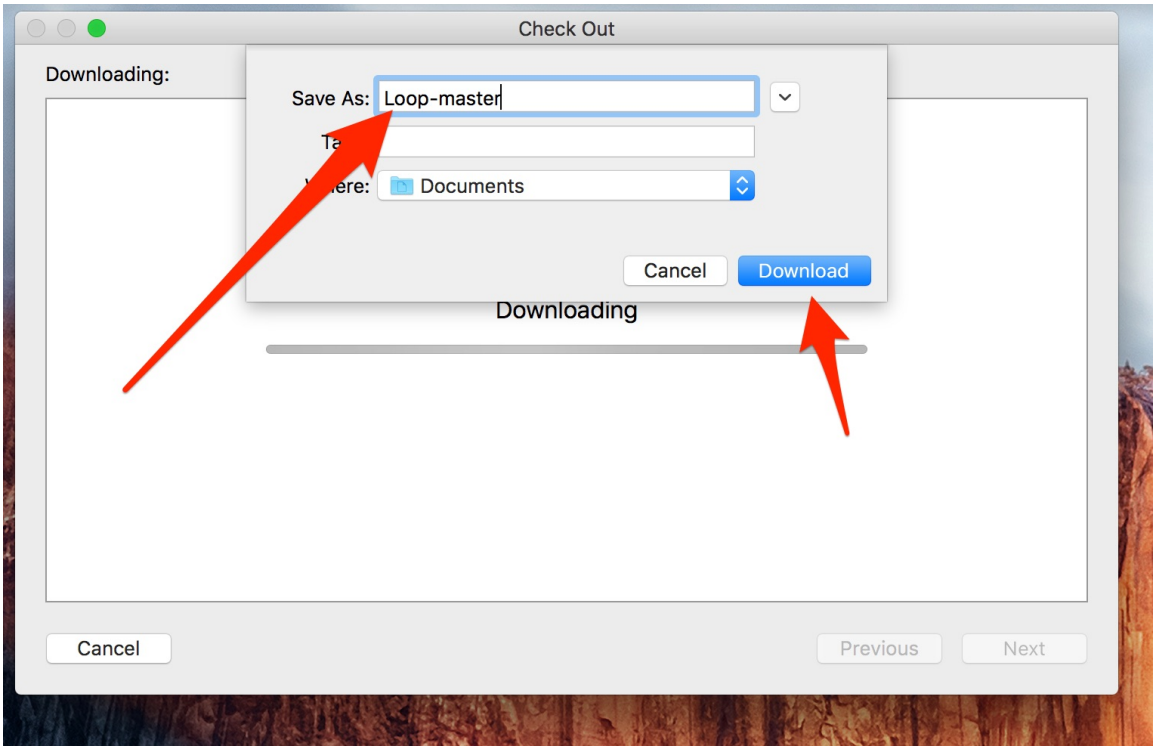
2. Use Source Control to download a local copy of the branch you'd like to build from



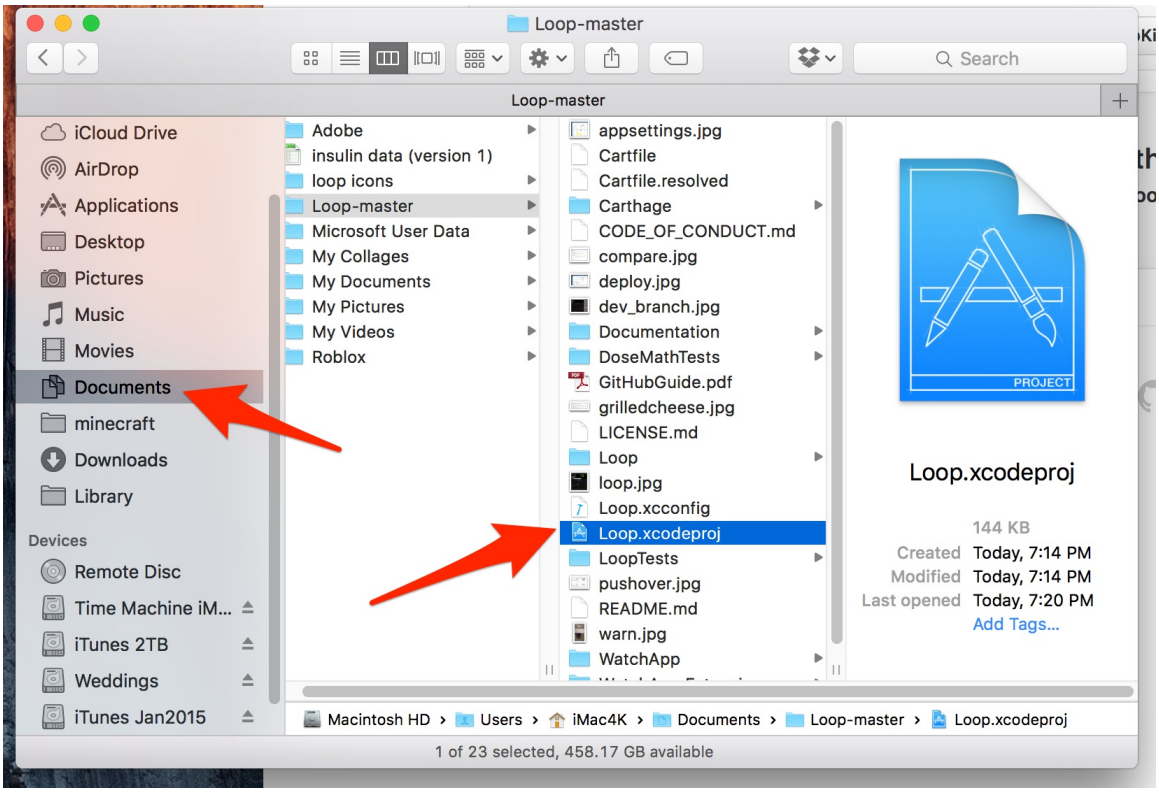




3. Rename the folder to something that helps you tell the difference between dev and master branches on your local computer, because you will likely build from both at different times. Download the folder and look where it is being saved (in this case to “documents” folder)



4. Open the Loop.xcodeproj file from your folder

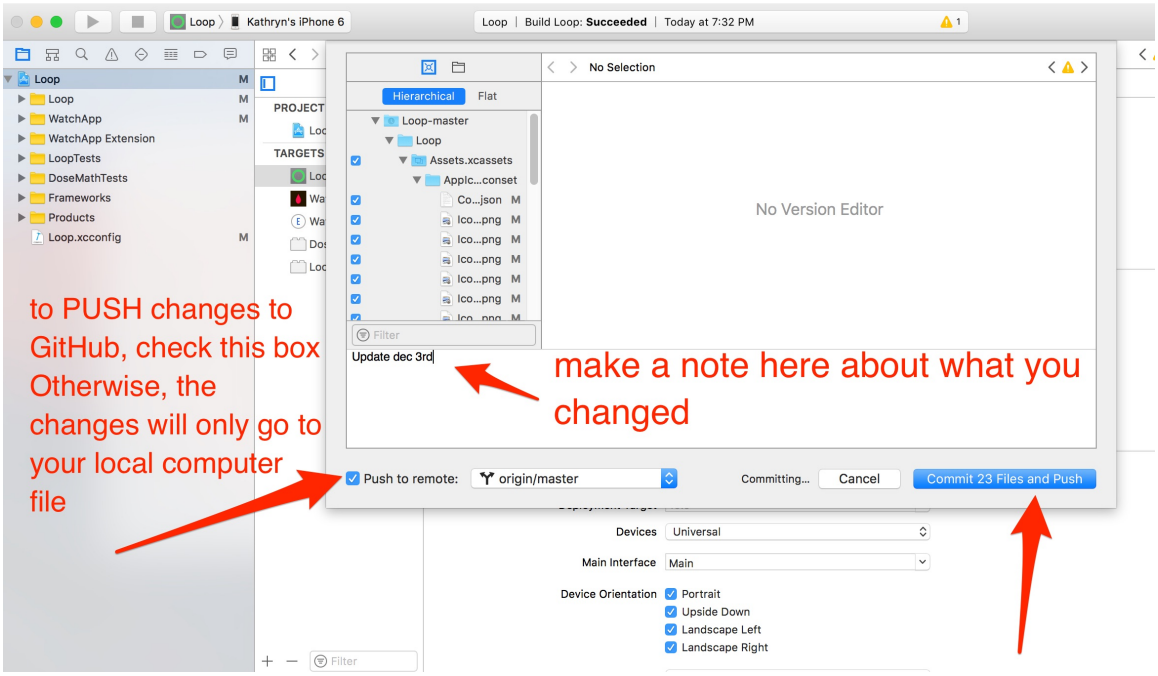


BUILD just like the guide starting in Section 4.2

1. **Change the MAIN_APP_BUNDLE_INDENTIFIER from com.loudnate**
2. **Make customizations**
3. **Sign all three targets**
4. **Build onto iphone**

And now that you're done building, we are going to Commit and Push changes. "Commit" saves the changes onto your local folder on your computer
"Push" saves the changes up into your GitHub repository

5. Go to Xcode menu at top of screen, choose Source Control, and choose Commit from the dropdown menu. Make a note about the changes, click the "push to remote" checkbox, and then the blue button to "commit and push"



Bravo! Now you have a repo in GitHub AND your local computer that have all your customizations and MAIN_APP_BUNDLE_IDENTIFIER saved.

FUTURE UPDATES

When you want to do future updates for Loop, simply go back to Compare, Pull Request and Merge section of this document. Perform those actions to update your GitHub repository.

Then go to the old Loop folder you downloaded onto your computer (remember it was in the Documents folder in this example?). Double click on the Loop.xcodeproj file again. When Xcode opens, choose Source Control from the top menu, and then choose Pull from the dropdown menu. Select the branch you'd like to build from (ideally the same branch's folder that you opened the Loop.xcodeproj from on your computer)

This will now Pull the updated parts from your newly updated GitHub repo into your local folder. All of your customizations will be good to go because they were saved, and you should just be able to build straight away. Your MAIN_APP_BUNDLE_INDENTIFIER and signing teams should be saved, but you can double check if you want. ☺