**Week 2 Discussion: Git collaboration continued**

In discussions last week, we practiced working in branches in *our own project without collaborators*, then submitting pull requests (PRs) to ourselves to incorporate our branches into the main branch.

This week, we’ll **work with a partner on a single version-controlled project, pushing changes in branches and submitting PRs that our partner will merge into main**.

**Allison & Casey DEMO (first).**

Our demo follows the instructions below, but we recommend just listening / watching us work through it.

**STEP 1. Assign partners (1 & 2)**

In breakout rooms, Allison & Casey will assign partners (we will specify who in each pair is Partner 1 & Partner 2). Partner 2, add your github username in the chat so your partner can add you as a collaborator (next step).

**STEP 2. Partner 1 makes version-controlled R Project**

Partner 1:

* Create a new GitHub repo called **git-collab** (with a ReadMe)
* Clone to create a version-controlled R Project
* Create a new .Rmd
* Delete everything below the first code chunk
* In setup code chunk, attach the **tidyverse** and **palmerpenguins** packages
* Save the .Rmd as **our\_report.Rmd** in the project root
* Stage, commit, pull, push all updates back to GitHub
* Add Partner 2 as a collaborator (Settings > Manage Access)

**STEP 3: Partner 2 accept invite & clone**

Partner 2:

* Accept the invitation (you should receive an email invitation)
* Clone the repo to get it locally in RStudio

**STEP 4: Both partners create a branch to work in**

* Both partners: create a new git branch named yourinitials-branch (e.g. mine would be **ah-branch** and Casey’s would be **co-branch**)

**STEP 5: Partner 1 edit, push changes & PR, Partner 2 review & merge**

* + **Partner 1: In your branch,** add a new code chunk to the .Rmd in which you write code to make a scatterplot of penguin body mass (body\_mass\_g) versus flipper length (flipper\_length\_mm):

ggplot(data = penguins, aes(x = body\_mass\_g,

y = flipper\_length\_mm)) +

geom\_point()

* Stage, commit, pull, then push updates to your branch.
* In GitHub, submit a pull request.

* + **Partner 2: Review PR & merge**

In GitHub, you may get an auto-announcement of a PR. Or, you should see a ‘1’ next to ‘Pull Requests’ indicating there is one PR open. Review the additions that Partner 1 is asking you to pull into the main branch, then merge.

**STEP 6: Both partners git merge origin**

Both partners: In the Terminal, type command & press Enter:

git pull

git merge origin

This command will merge the main branch (which will cause no changes to Partner 1’s .Rmd, but Partner 2’s .Rmd should be updated with the penguin graph that has been merged into main from Partner 1’s PR).

**STEP 6: Partner 2 pushes changes & PR, Partner 1 review & merge**

* + **Partner 2:** add a new code chunk to the .Rmd in which you write code code to make a subset of the penguins dataset, called **adelie**, that only contains observations for Adélie penguins:

adelie <- penguins %>%

filter(species == “Adelie”)

* Stage, commit, pull, then push updates to your branch.
* In GitHub, submit a pull request
  + **Partner 1: Review PR & merge**

In GitHub, you may get an auto-announcement of a PR. Or, you should see a ‘1’ next to ‘Pull Requests’ indicating there is one PR open. Review the additions that Partner 2 is asking you to pull into the main branch, then merge.

**STEP 7: Both partners git merge origin**

Both partners: In the Terminal, type command & press Enter:

git pull

git merge origin

This command will merge the main branch (which will cause no changes to Partner 2’s .Rmd, but Partner 1’s .Rmd should be updated with the filtering step that has been merged into main from Partner 2’s PR).