Git and GitHub

Juan Carlos Villaseñor-Derbez (JC)

Hands-on: Building Your First Repository

The overall flow:

0. Create a GitHub account (if you don't have one), install git, authenticate (instructions)
1. Create a new repository on GitHub
2. Clone it to your computer
3. Add your project files
4. Make your first commit
5. Push to GitHub

Step 0: Check set up

1. Make sure you have git. In a terminal (Mac) or Git bash (Windows) run:

```
which git
```

```
/usr/bin/git
```

2. Make sure your have introduced yourself to git. To check for your configuration, use this in a terminal:

```
git config -l
```

```
credential.helper=osxkeychain
init.defaultbranch=main
user.email=juancarlos.villader@gmail.com
user.name=jcvdav
core.excludesfile=~/.gitignore
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
core.repositoryformatversion=0
core.filemode=true
core.bare=false
core.logallrefupdates=true
```

```
core.ignorecase=true
core.precomposeunicode=true
remote.origin.url=https://github.com/jcvdav/EVR_628.git
remote.origin.fetch=+refs/heads/*:refs/remotes/origin/*
branch.main.remote=origin
branch.main.merge=refs/heads/main
branch.main.vscode-merge-base=origin/main
pull.rebase=true
branch.dev.remote=origin
branch.dev.merge=refs/heads/dev
branch.test.remote=origin
branch.test.merge=refs/heads/test
```

3. Update {`EVR628tools`}:

```r
remotes::install_github("jcvdav/EVR628tools")
```

Step 1: Create a GitHub Repository

1. Go to github.com and sign in
2. Click the "`+`" icon in the top right
3. Select "*New repository*"
4. Choose a simple name: `hellow_world`
5. Add a description: "*Learning how to use GitHub*"
6. Make it **Public**
7. Check "*Add a README file*"
8. Select gitignore tailored to R
9. Click "*Create repository*"

Step 2: Clone to Your Computer

In RStudio:

1. Top-right corner → New Project
2. Choose "Version Control"
3. Select "Git"
4. Paste your repository URL
5. Choose where to save it locally (Not on iDrive, not on DropBox, not on GoogleDrive…)
6. Click "Create Project"
7. Verify that you now have a `README.md` and `.gitignore` files

Step 3: Set Up Your Project Structure

To create these folders in your repository:

```
environmental-data-project/
  data/
      raw/
      processed/
|      output/
  scripts/
      01_processing/
      02_analysis/
      03_content/
  results/
      img/
      tab/
  docs/
  README.md
```

Use:

```
EVR628tools::create_dirs()
```

Step 4: Create / Modify the global and project .gitignore Files

1. We will use the `usethis::git_vaccinate()`
2. Look at the documentation using `?usethis::git_vaccinate()`
3. Call `usethis::git_vaccinate()`
4. Create a `.gitignore` file to exclude unnecessary files:

```
# R specific
.Rhistory
.RData
.Ruserdata
*.Rproj.user/

# System files
.DS_Store
Thumbs.db
```

Note: You can always add specific files with `git add -f filename`, edit the `.gitignore` file or use the git pane

Step 5: Add Your First "content"

- Create a simple R script in `scripts/content/`
- Choose one of the data sets in `EVR628tools`
- Spend some time reading the documentation
- Visualize it

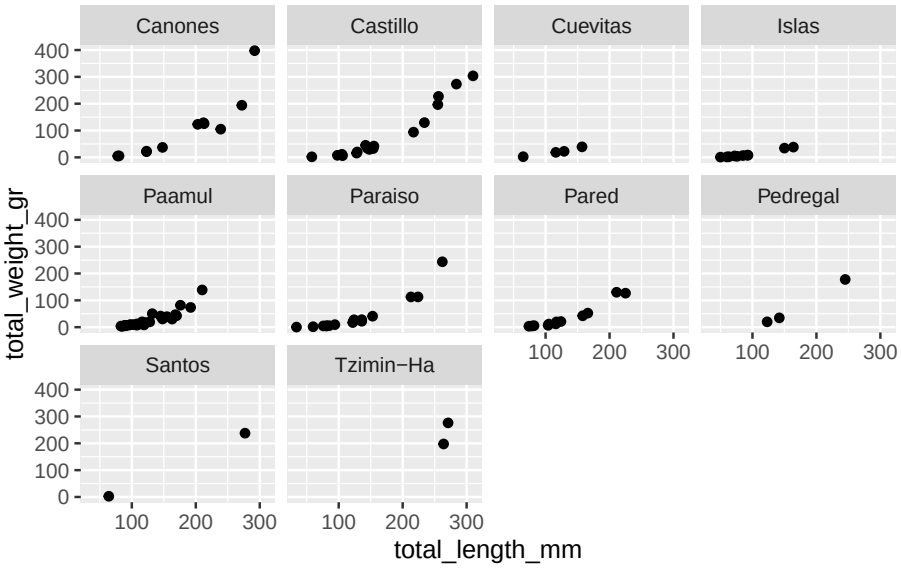
```
# Load packages
library(EVR628tools)
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.2     v tibble    3.3.0
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.1.0
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to beco
```

```
# Lod data
data(data_lionfish)

# Create a simple plot
p <- ggplot(data_lionfish,
            aes(x = total_length_mm, y = total_weight_gr)) +
  geom_point() +
  facet_wrap(~site)

p
```




```
# Save plot
# The above shouldn't work, why?
ggsave(plot = p, filename = "results/figures/first_plot.png")
```

```
Error in `ggsave()`:
! Cannot find directory 'results/figures'.
i Please supply an existing directory or use `create.dir = TRUE`.
```

Step 6: Update Your README

Create a comprehensive README.md

```
# My first repo

## Description

Analysis of environmental data for EVR 628 course.

## Project Structure

- `scripts/`: R scripts for analysis
- `results/`: Output figures and tables

## Author

[Your Name and email?]
```

Step 7: Stage-Commit-Push

1. Stage, commit your R script
2. Push
3. Stage, commit your README
4. Stage, commit your plot
5. Push

Now lets colaborate

1) Simple collaboration

- Find a partner
- Decide who will be partner A and who will be partner B

Parter A

1. Go to settings -> collaborators -> Add collaborator
2. Add partner B's username and send an invitation
3. Wait until B tells you they are done.

... wait ...

4. Pull

Parter B

1. Give partner A your username
2. Check your email
3. Go to the repository
4. Repeat the cloning process:

- You will have a new project
- Use a different location (I suggest desktop)

5. Make one change to partner A's figure, and update the exported image
6. Stage the files, then commit, and push
7. Tell partner A you are done

Changes to make by B:

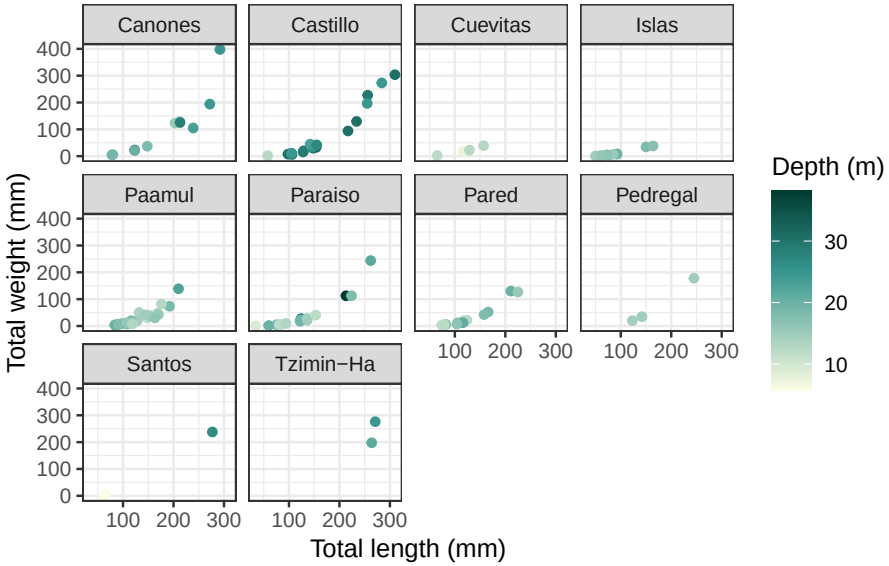
```
# Example: environmental_data_analysis.R
library(EVR628tools)
library(tidyverse)

data(data_lionfish)

# Create a simple plot
p <- ggplot(data_lionfish,
```

```
            aes(x = total_length_mm, y = total_weight_gr,
                ### B adds color by depth
                color = depth_m)) +
  geom_point() +
  facet_wrap(~site) +
  ### B modifies default colors
  scale_color_gradientn(colors = palette_IPCC(var = "prec", type = "seq")) +
  ### B adds labels
  labs(x = "Total length (mm)",
       y = "Total weight (mm)",
       color = "Depth (m)") +
  ### B adds theme
  theme_bw()

p
```



- Now switch who is A and who is B and repeat

2) Branching and pull requesting

> **!** Important
>
> Partner **A** works on Partner **B's** project, and *vice versa*

0. **Make sure you are in your partner's site**
1. In your Git pane, click on "New Branch"
2. Create a new branch, I suggest you call it `dev_your_name`
3. Make a change to the figure's code, then stage and commit
4. Export the figure, then stage -> commit -> push
5. Go to the project's repo on GitHub.com
6. Make sure your new branch and changes are there
7. Go to the Pull Requests tab
8. Start a new pull request (from `dev_your_name` into `main`)
9. Explain what the pull request does and submit it
10. Your partner will now get an email
11. Partner can incorporate pull request as needed