In-Class Problem Set: Reproducible Visualization Workflow (R + GitHub)

Goal. Extend the code from the lecture slides to produce a small, reproducible workflow: load the provided dataset, create multiple figures using explicit mappings, and submit your work through GitHub.

What to submit (in your GitHub repo).

- A script file: `scripts/lab.R`
- A short write-up: `outputs/writeup.md`
- Saved figures: at least 4 image files in `figures/`
- (Optional but recommended) a log file: `outputs/log.txt`

Rules.

- Work inside an **R Project**.
- Use a **sequential, hard-coded workflow** (no user-defined functions).
- You may consult notes and documentation. If you use any external code, cite it in your write-up.

Questions

1. **Create an R Project (proof required).**
   (a) Create an R Project for this course on your computer.
   (b) **Proof:** In your `outputs/writeup.md`, include:
      - the output of `getwd()` run from inside the project, and
      - a screenshot showing the `.Rproj` file in your project folder *or* the RStudio Project name visible in the RStudio window.
2. **Load the provided dataset from the `data/` folder.**
   (a) Confirm the dataset file exists in `data/`. (Do not manually move it.)
   (b) Write code in `scripts/lab.R` to load it into R as an object named `vdem`.

```
# PSEUDOCODE:
# 1) read the .rds file from the data/ folder
# 2) store it as vdem

vdem <- readRDS("data/vdem.rds")
```

   (c) **Proof:** In `outputs/writeup.md`, include:
      - the dimensions of `vdem` (rows × columns), and
      - the first 3 column names.

```
# PSEUDOCODE for the proof:
dim(vdem)
names(vdem)[1:3]
```

3. **Work with fixed variables (no selection required).**
   (a) For all plots in this assignment, use the following variables from the dataset:
      - Access to justice for women: `v2clacjstw`
      - Access to justice for men: `v2clacjstm`
      - Freedom from political killings: `v2clkill`
      - Freedom from torture: `v2cltort`
   (b) Treat *all four variables as continuous.* You do **not** need to identify or justify variable types.
   (c) Your baseline plot will use the following mapping (note: no trailing comma):

```
# PSEUDOCODE:
# x = justice for women
# y = justice for men
# points = one row per observation

library(ggplot2)

p0 <- ggplot(vdem, aes(x = v2clacjstw, y = v2clacjstm)) +
  geom_point()

p0
```

   (d) Then create a version that uses **both** color and size:

```
# PSEUDOCODE:
# color = freedom from political killings
# size  = freedom from torture

p1 <- ggplot(vdem, aes(
  x = v2clacjstw,
  y = v2clacjstm,
  color = v2clkill,
  size  = v2cltort
)) +
  geom_point()

p1
```

   (e) **Proof:** In `outputs/writeup.md`, include:
      - confirmation that all four variables exist in the dataset (show `names(vdem)` output or a short snippet), and
      - their reported class from `str()`.

```
# PSEUDOCODE for the proof:
names(vdem)
str(vdem[, c("v2clacjstw","v2clacjstm","v2clkill","v2cltort")])
```

Pseudocode guide (follow this order in `scripts/lab.R`):

   (a) Load required libraries (`ggplot2`, optionally `tidyverse`).
   (b) Read the dataset from `data/` into `vdem`.
   (c) Confirm required variables exist (`names(vdem)`; `str(...)`).

(d) Make the baseline plot (x = `v2clacjstw`, y = `v2clacjstm`).
(e) Make the enhanced plot (add color = `v2clkill`, size = `v2cltort`).
(f) Save plots with `ggsave()` into `figures/`.

4. **Create a reproducible folder structure + (optional) logging.**
   (a) Ensure these folders exist in your project:
   - `scripts/`
   - `outputs/`
   - `figures/`
   - `logs/` *(optional but recommended)*

   (b) In `scripts/lab.R`, add code that creates any missing directories (without errors).

```
# PSEUDOCODE:
# if a folder does not exist, create it
dir.create("scripts", showWarnings = FALSE)
dir.create("outputs", showWarnings = FALSE)
dir.create("figures", showWarnings = FALSE)
dir.create("logs", showWarnings = FALSE)
```

   (c) **Proof:** In `outputs/writeup.md`, include `list.files()` output showing the folders.

```
# PSEUDOCODE for proof:
list.files()
```

   (d) **Optional (challenge):** Create a simple log file `outputs/log.txt` that records:
   - the current date/time,
   - the dataset filename loaded,
   - and the names of the four required variables.

5. **Make three plot extensions + comment on them.**
   Using your baseline plot as the starting point, create **three** distinct extensions (three separate figures). Each figure must include a caption in your write-up that explains:
   - what variables are mapped to what visual properties,
   - what comparison is easiest to make,
   - and one default choice you are accepting (or changing) and why.

   Your three extensions must come from different categories below (choose any three):
   (a) **Add an annotation layer:** add a title + axis labels.
   (b) **Handle overplotting:** use transparency (`alpha`) and briefly explain why.
   (c) **Scale adjustment:** adjust the color scale and/or size scale and explain the effect.

   Saving requirement: Save each plot to `figures/` using `ggsave()` (do not rely on screenshots). Name files clearly (e.g., `figures/plot1.png`, `figures/plot2.png`, `figures/plot3.png`).

```
# PSEUDOCODE for saving:
ggsave("figures/plot_baseline.png", plot = p0, width = 7, height = 5)
ggsave("figures/plot_color_size.png", plot = p1, width = 7, height = 5)
```

6. **If you finish early:**
   - Add a short "limitations" note (1–2 sentences) about what the plot cannot show.
   - Add a deliberately "bad" version and write 3 bullets on why it misleads.

7. **GitHub requirement:** Commit and push your work.
   Proof: In `outputs/writeup.md`, include:
   - the output of `git status` *after* committing (showing a clean working tree), and

- either a screenshot of your GitHub repo showing the latest commit *or* the commit hash and message.

```
# PSEUDOCODE (terminal):
git add .
git commit -m "finish lab"
git status
git push
```

Checklist (before you leave)

- `scripts/lab.R` exists and runs top-to-bottom
- `outputs/writeup.md` exists and includes required proofs
- At least 4 figures saved in `figures/`
- Work is pushed to GitHub