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

Goal. Practice a simple, reproducible visualization workflow in R: load a dataset, make a small number of plots using fixed variables, save your output, and push your work to 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 3 image files in `figures/`

Rules.

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

Questions

1. **Create an R Project (proof required).**
   (a) Create an R Project for this course.
   (b) **Proof:** In `outputs/writeup.md`, include:
   - the output of `getwd()`, and
   - a screenshot showing the project name in RStudio.
2. **Load the dataset.**
   (a) Confirm the dataset file exists in `data/`.
   (b) Load the dataset into R as an object named `vdem`.

```
# Step 1: load the dataset
vdem <- readRDS("data/vdem.rds")
```

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

```
# Proof code
dim(vdem)
names(vdem)[1:3]
```

3. **Variables used in all plots.**
   For all plots in this assignment, use the following variables:
   - `v2clacjstw` (access to justice for women)
   - `v2clacjstm` (access to justice for men)
   - `v2clkill` (freedom from political killings)
   - `v2cltort` (freedom from torture)

Treat all variables as continuous.

Proof: In `outputs/writeup.md`, show that these variables exist and report their class.

```
names(vdem)
str(vdem[, c("v2clacjstw","v2clacjstm","v2clkill","v2cltort")])
```

4. **Create the baseline plot.**
   Create a scatterplot with:
   - x-axis: `v2clacjstw`
   - y-axis: `v2clacjstm`

```
library(ggplot2)

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

p0
```

Save the plot to `figures/`.

```
ggsave("figures/plot_baseline.png", plot = p0)
```

5. **Create two plot extensions (your design choices).**
   Starting from the baseline plot, create **two** additional figures. For each extension, you will make a **design choice** about how to encode extra information.

Extension 1 (your choice: add ONE encoding).
Create a version of the baseline plot where you add **one** of the following:
- map one variable to **color**, *or*
- map one variable to **size**.

You must use one of these variables as your added information: `v2clkill` or `v2cltort`.

```
# PSEUDOCODE:
# Choose ONE:
# - color = <your choice: v2clkill OR v2cltort>
# OR
# - size  = <your choice: v2clkill OR v2cltort>

p1 <- ggplot(vdem, aes(
  x = v2clacjstw,
  y = v2clacjstm,
  <your choice goes here>
)) +
  geom_point()

p1
```

Save this plot.

```
ggsave("figures/plot_extension1.png", plot = p1)
```

Extension 2 (your choice: add a SECOND encoding).
Create another version where you add a **second** encoding, so your plot uses **both** color and size. Use the remaining variable that you did *not* use in Extension 1.

```
# PSEUDOCODE:
# If Extension 1 used:
#   color = v2clkill
# then Extension 2 should add:
#   size  = v2cltort
#
# If Extension 1 used:
#   size  = v2cltort
# then Extension 2 should add:
#   color = v2clkill

p2 <- ggplot(vdem, aes(
  x = v2clacjstw,
  y = v2clacjstm,
  <your two choices go here>
)) +
  geom_point()

p2
```

Save this plot.

```
ggsave("figures/plot_extension2.png", plot = p2)
```

6. **Brief written interpretation.**
   In `outputs/writeup.md`, write **3 short bullets**:
   - One thing you can see in the baseline plot.
   - One thing your Extension 1 makes easier to see.
   - One thing your Extension 2 makes easier (or harder) to see.
   - What argument is being conveyed by these plots. What are some ways the argument can be improved?
7. **GitHub submission (proof required).**
   Commit and push your work.

```
git add .
git commit -m "Finish in-class visualization lab"
git push
```

Proof: In `outputs/writeup.md`, include:
- the output of `git status` after committing,
- the commit hash or a screenshot of the GitHub repo.

Checklist (before you leave)

- `scripts/lab.R` runs top-to-bottom without errors

- `outputs/writeup.md` includes all required proofs
- Three figures saved in `figures/`
- Changes pushed to GitHub