Digital Research Toolkit for Linguists

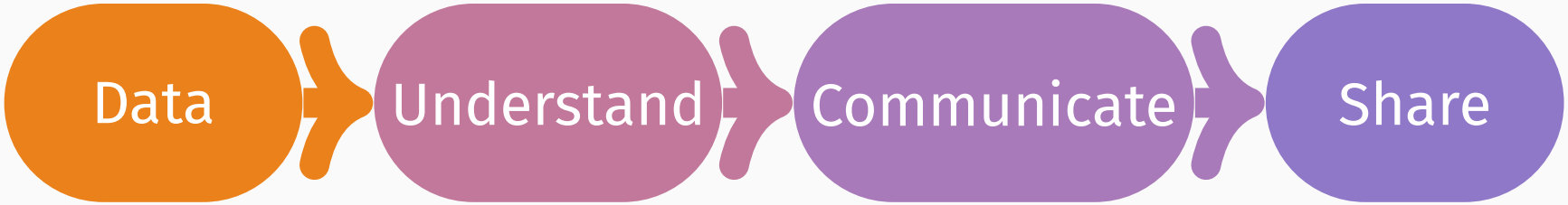
Week 15: Git, GitHub, and SSH

Anna Pryslopska

July 15, 2024

Psycholinguistics and Cognitive Modeling Lab

Questions?



R & RStudio, packages, data types, formats, encoding

import from workspace, assign values, operations, clean, filter, arrange, select, merge, group, summarize, export, visualize

document, research, create clean and beautiful reports

connect, collaborate, backup

Table of contents

SSH

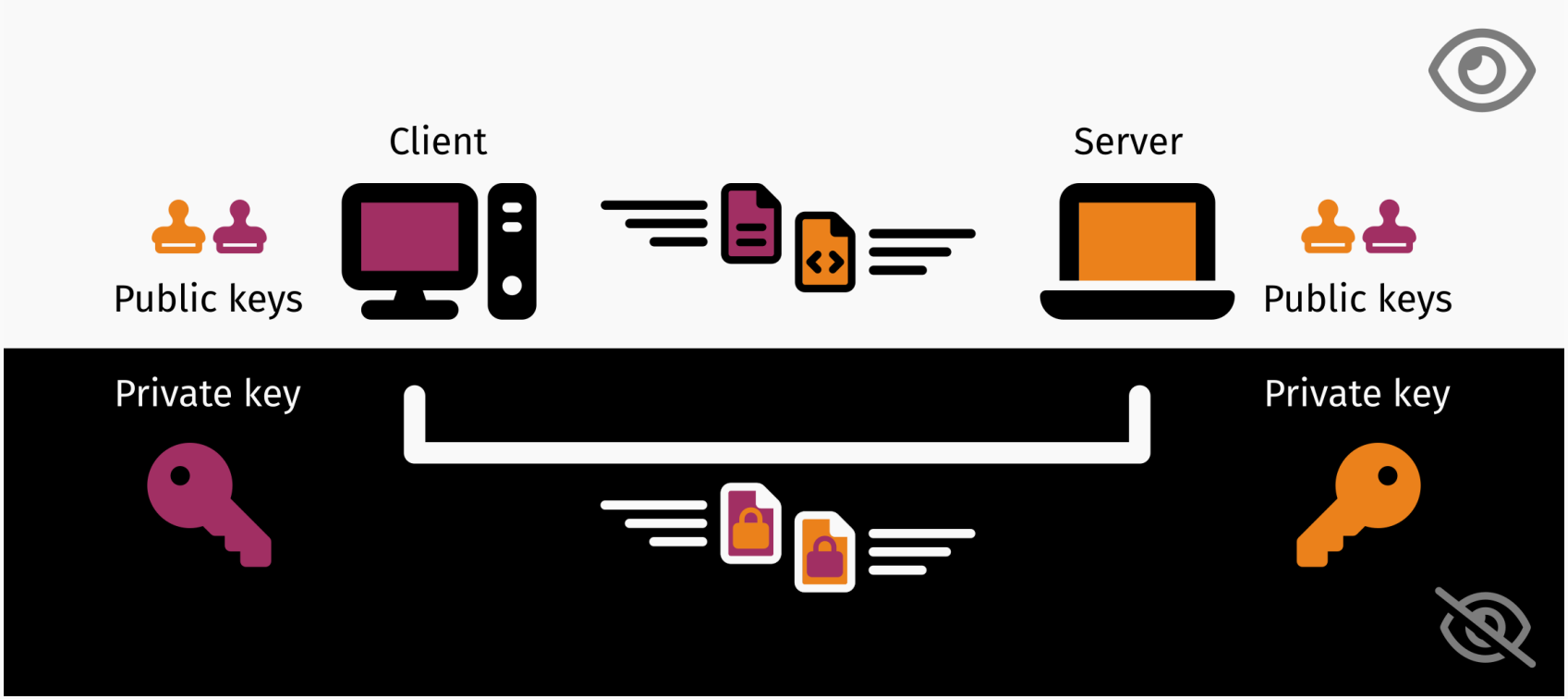
SS Hwæt?

SSH = Secure Shell Protocol

A  is a command-line interpreter that runs in the terminal.

SSH is a network protocol for **operating services securely over an unsecured network**, e.g. remote login and command-line execution (connect and authenticate). (**wikipediassh**)

Client, server, and asymmetric encryption



Public-private key pair

Public key 🔑: identifies that it's you, verifies the client's answer

Private key 🗝️: dictionary of responses to prompts

Key communication is similar to spy speak:

Client <knock knock>
Server Speak. 🔑
Client The significant owl hoots in the night. 🔑
Server Yet many grey lords go sadly to the masterless men. 🔑✔🗝️

Guards Guards by Terry Pratchett

SSH: Create a new key (make SSH key and select this algorithm)

`ssh-keygen -t ed25519` Fingerprint and randomart are **not keys**.

```
anna@AP-UniSTR-Laptop:~/Desktop/Linguistics toolkit course SoSe2024/Big
Project/big_project$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key [redacted]: secret
s
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in secrets
Your public key has been saved in secrets.pub
The key fingerprint is:
SHA256:7J210xr6yuI4Hh9DJHmtIASjBPkPiEOJoZ3v2JXmPHY anna@AP-UniSTR-Laptop
The key's randomart image is:
+--[ED25519 256]--+
|=++..            |
|== +   . .       |
|=oo . + o .      |
|+ o. . B .       |
| . o. + S   .    |
|   +.= o . o o   |
|  . o * E o + .  |
|     ..*.+ . +   |
|     .ooo.+oo    |
+----[SHA256]-----+
```

Show and don’t tell

Show the public key `cat secrets.pub`

(`cat` = concatenate/show, `secrets.pub` = file in hidden folder):

```
Project/big_project$ cat secrets.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIMNVztc+JmZsGosMZr2tmgtP/NVHQyS4bX8b
x0ZUxxA7 anna@AP-UniSTR-Laptop
```

Show and don’t tell

Show the private key `cat secrets`

```
-----BEGIN OPENSSH PRIVATE KEY-----
```

🍑🍒🍆

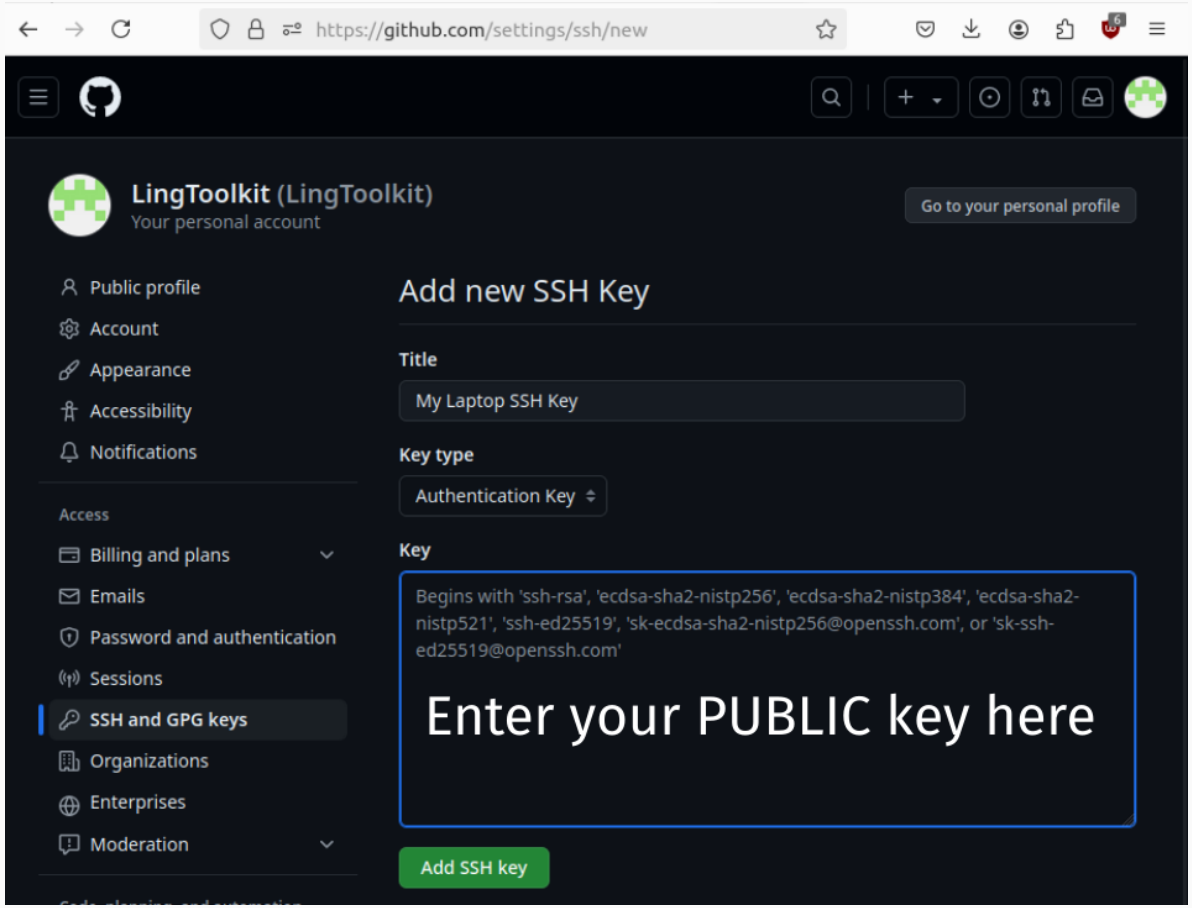
```
-----END OPENSSH PRIVATE KEY-----
```

```
Project/big_project$ cat secrets
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAAAMwAAAAtzc2gtZW
QyNTUxOQAAACDDVc7XPiZmbBqLDGa9rZoLT/zVR0MkuG1/G8dGVMcQOwAAAJiz4fYcs+H2
HAAAAAtzc2gtZWQyNTUxOQAAACDDVc7XPiZmbBqLDGa9rZoLT/zVR0MkuG1/G8dGVMcQOw
AAAEBvwzDNll6J92ETgP8vgYdYesUTBVrfCJtlUIOBHcvkicNVztc+JmZsGosMZr2tmgtP
/NVHQyS4bX8bx0ZUxxA7AAAAFWFubmFAQVAtVW5pU1RSLUxhcHRvcA==
-----END OPENSSH PRIVATE KEY-----
```

Make a new SSH key pair and save it somewhere secret.

```
ssh-keygen -t ed25519
```

Practically, we're using SSH as a substitute for passwords in communicating with GitHub.



Click here for the accompanying video.

Git and GitHub

Version control and git

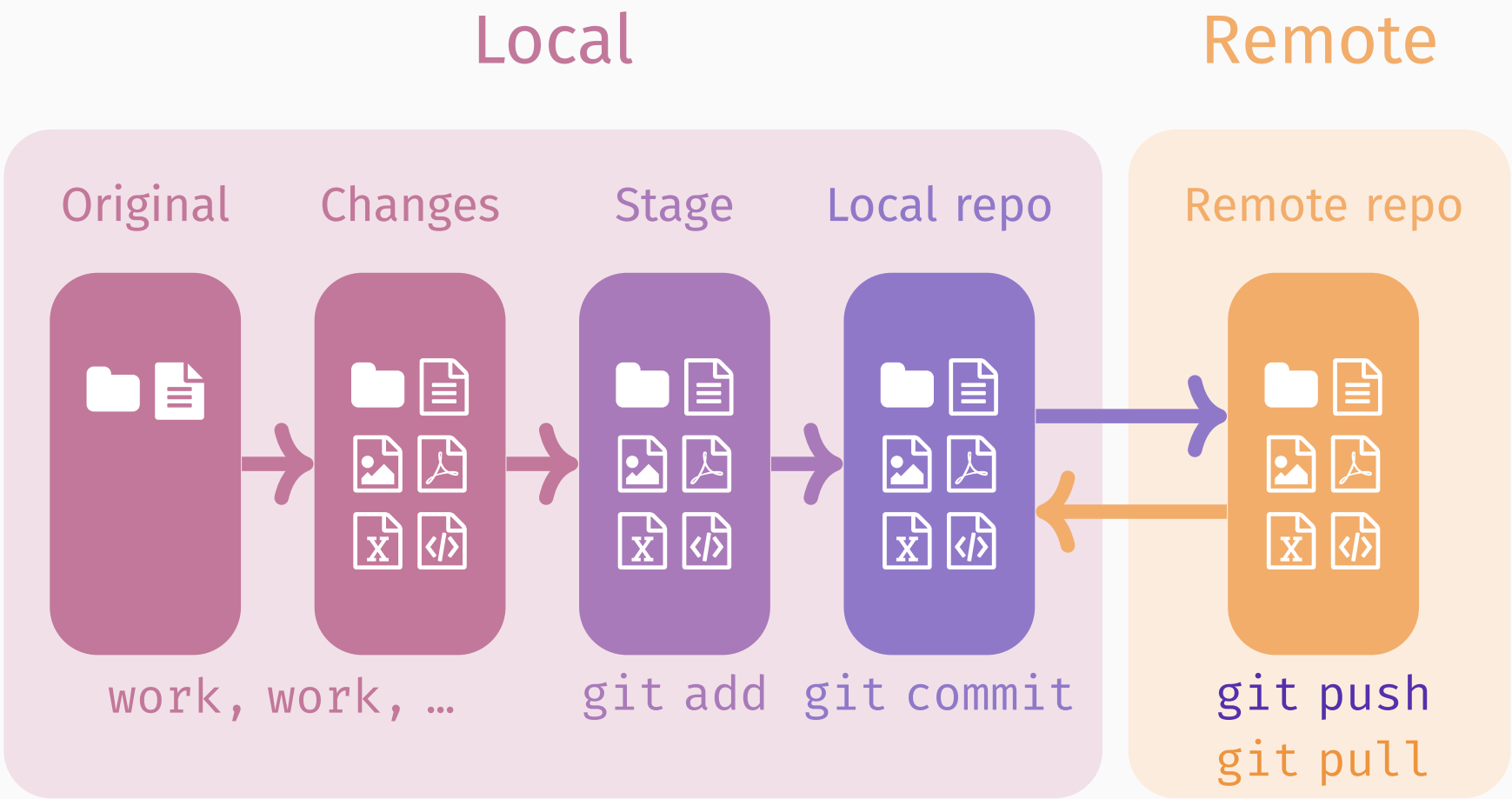
Git vs GitHub

git A tool for tracking changes to your files on **your computer**.

A service for hosting & sharing those files **online** + collaboration tools.

git Git	GitHub
Version control system	Web-based platform
Program	Service that uses Git
Keeps track of file changes over time	Hosts your Git repositories online
Local to you	Remote, makes collaboration possible
Takes snapshots (commits) of a project that you can look at	You push your local repositories + commits to a server
Managing your code and a local copy of someone else's code	Collaboration across multiple remote machines
Offline on your computer	On a server in the cloud
Not social media	LinkedIn for programmers

Workflow


Local
Remote
Original
Changes
Stage
Local repo
Remote repo
work, work, …
git add
git commit
git push
git pull

Glossary

`git clone URL`	make a new copy of a repository.
`git add file.txt`	tell git to track untracked files.
`git commit -m "msg"`	snapshot of tracked files + message
`git status`	what are the changes since the last commit?
`git push`	send changes to a remote repository.
`git pull`	get changes from a remote repository.
`git diff file.txt`	difference in changes between files.
`git log`	show the commits + IDs + messages.

which program (topic) do what more details

Restoring and reverting

Going back in time

```
\documentclass[a4paper,11pt]{book}
\usepackage{import}
\usepackage[backend = biber, style=authoryear]{biblatex}
\usepackage[top=1in]{geometry}

\addbibresource{book.bib}
\usepackage{mypreamble}

\begin{document}


\end{document}
```

original

```
\documentclass[a4paper,11pt]{book}
\usepackage{import}
\usepackage[backend = biber, style=authoryear]{biblatex}
\usepackage[top=1in]{geometry}

\addbibresource{book.bib}
\usepackage{mypreamble}

\begin{document}

\frontmatter
\import{./}{title.tex}

\clearpage
\thispagestyle{empty}

\tableofcontents

\mainmatter

\chapter{Introduction}

This text is taken and slightly edited from \textcite{bielefeld}.

\chapter{First chapter}
\import{chapters/}{chapter1.tex}

\chapter{Second chapter}
\import{chapters/}{chapter2.tex}

\chapter{Third chapter}
\include{chapters/chapter3.tex}

\chapter{Forth chapter}
\printbibliography

\end{document}
```

changed

Going back in time

*You have added FILENAME but **not commited** the changes. Now you...*

... want to unstage this file because it does not fit the current commit.

```
git restore --staged FILENAME
```

*... **completely undo all changes** in the file and restore to a previous commit state:*

```
git checkout FILENAME
```

Restore vs checkout

```
\documentclass[a4paper,11pt]{book}
\usepackage{import}
\usepackage[backend = biber, style=authoryear]{biblatex}
\usepackage[top=1in]{geometry}

\addbibresource{book.bib}
\usepackage{mypreamble}

\begin{document}


\end{document}
```

after **checkout** = original

```
\documentclass[a4paper,11pt]{book}
\usepackage{import}
\usepackage[backend = biber, style=authoryear]{biblatex}
\usepackage[top=1in]{geometry}

\addbibresource{book.bib}
\usepackage{mypreamble}

\begin{document}

\frontmatter
\import{./}{title.tex}

\clearpage
\thispagestyle{empty}

\tableofcontents

\mainmatter

\chapter{Introduction}

This text is taken and slightly edited from \textcite{bielefeld}.

\chapter{First chapter}
\import{chapters/}{chapter1.tex}

\chapter{Second chapter}
\import{chapters/}{chapter2.tex}

\chapter{Third chapter}
\include{chapters/chapter3.tex}

\chapter{Forth chapter}
\printbibliography

\end{document}
```

after **restore** = changed

Going back in time

*You want to undo a change that has been **commited but not pushed***

`git log --oneline`	check the commit ID, e.g. `3add00e`
`git reset ID`	undo **until** the commit (move backwards) but **does not change** the file
`git reset --soft ID`	undo **until** the commit (move backwards) but **does not change the file** and **keeps the staging**
`git reset --hard ID`	same & **changes** the file to earlier version
`git reset --soft HEAD~`	reset the last commit

Going back in time

*You want to undo a change that has been **commited and pushed***

`git revert ID`

undo **only this** change completely
and move forwards as **new commit**

Reset

```
\documentclass[a4paper,11pt]{book}
\usepackage{import}
\usepackage[backend = biber, style=authoryear]{biblatex}
\usepackage[top=1in]{geometry}

\addbibresource{book.bib}
\usepackage{mypreamble}

\begin{document}


\end{document}
```

after `reset --hard` = orginal

```
\documentclass[a4paper,11pt]{book}
\usepackage{import}
\usepackage[backend = biber, style=authoryear]{biblatex}
\usepackage[top=1in]{geometry}

\addbibresource{book.bib}
\usepackage{mypreamble}

\begin{document}

\frontmatter
\import{./}{title.tex}

\clearpage
\thispagestyle{empty}

\tableofcontents

\mainmatter

\chapter{Introduction}

This text is taken and slightly edited from \textcite{bielefeld}.

\chapter{First chapter}
\import{chapters/}{chapter1.tex}

\chapter{Second chapter}
\import{chapters/}{chapter2.tex}

\chapter{Third chapter}
\include{chapters/chapter3.tex}

\chapter{Forth chapter}
\printbibliography

\end{document}
```

after `reset ID` = changed

after `reset --soft` = changed

Restore, checkout, reset, revert

Command	Effect on file	Effect on repo
✅ `restore`	keep changes	unstage
✅ `revert`	delete changes	keep commit, make a new commit
⚠️ `reset`	keep changes	abandon commit, go to earlier commit
⚠️ `reset --soft`	keep changes	abandon commit, go to earlier commit
❗ `reset --hard`	delete changes	abandon commit, go to earlier commit
❗ `checkout`	delete changes	unstage, go to earlier commit

HEAD commit 

`d8c6f90` HEAD Removed everything		newest
`4581f94` Drafted the conclusions	← I want to go back here	
`57ff630` Summarized the results		
`66528ea` Added the experiments		
`a71ea1c` Wrote the introduction		oldest

Reset HEAD

```
git reset --hard 4581f94
```

← Disappeared without a trace

`4581f94` HEAD Drafted the conclusions — newest

`57ff630` Summarized the results

`66528ea` Added the experiments

`a71ea1c` Wrote the introduction — oldest

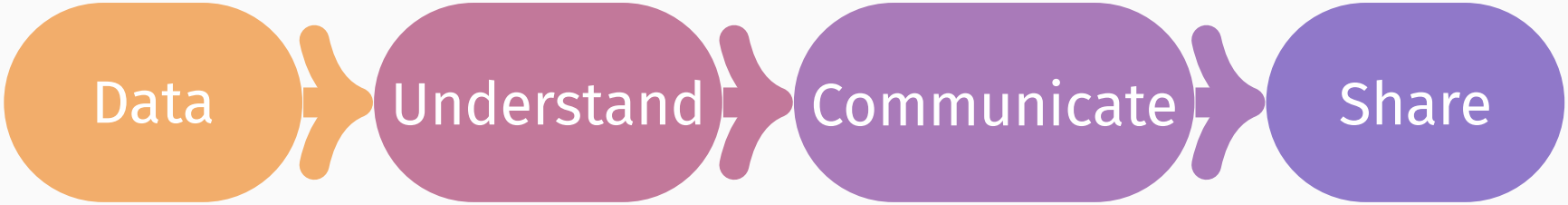
Revert HEAD 

```
git revert HEAD -m "Added everything back"
```

`g4c5t4g` HEAD Added everything back	newest
`d8c6f90` Removed everything	← old HEAD
`4581f94` Drafted the conclusions	
`57ff630` Summarized the results	
`66528ea` Added the experiments	
`a71ea1c` Wrote the introduction	oldest

Course wrap-up

Workflow



import,
clean,
transform,
export,
visualize,
model

document,
create
clean and
beautiful
reports

connect,
collaborate,
backup,
replicate

Data and understanding data

- ✔ R and RStudio IDE
- ✔ Packages (installing and loading)
- ✔ Working directory
- ✔ Directories and file hierarchy
- ✔ Scripts
- ✔ Assignment
- ✔ Scripts
- ✔ Data types
- ✔ Encoding
- ✔ Reading in data

Understanding data 1/3

- ✔ Inspecting data
- ✔ Renaming
- ✔ Selecting
- ✔ Dealing with missing data
- ✔ Coding basics
- ✔ Logic
- ✔ Filtering
- ✔ Arranging
- ✔ Pipes

Understanding data 2/3

- ✔ Tidy code
- ✔ Power of names
- ✔ Joining data frames
- ✔ If...else
- ✔ Grouping
- ✔ Summarizing
- ✔ Getting help
- ✔ Data visualization goals
- ✔ Accessibility and WCOG
- ✔ Plot types and choice of visualization
- ✔ Plotting in R with `ggplot2` and `esquisse`

Understanding data 3/3

- ✔ R programming basics and RStudio IDE
- ✔ Write scripts
- ✔ File encoding, variable naming, and tidy code with pipes
- ✔ Install and load packages
- ✔ Import/export data from/to the working directory
- ✔ Save and remove objects in the environment
- ✔ Preprocess raw data (filtering, renaming, arranging, mutating, selecting, if else)
- ✔ Make sense of data (merging, grouping, summarizing)
- ✔ Print and visualize the results
- ✔ Find help

Communicating results 1/2

- ✔ Documentation and why it's important
- ✔ Markdown and Pandoc
- ✔ Creating documentation with Quarto
- ✔ Knitting to PDF from R
- ✔ LaTeX document structure
- ✔ Scientific document structure
- ✔ Basic LaTeX commands
- ✔ Scientific document structure
- ✔ Typography in LaTeX
- ✔ Creating tables
- ✔ Including plots
- ✔ Cross-referencing and hyperlinking
- ✔ LaTeX for linguists

Communicating results 2/2

- ✔ Large project management
- ✔ Including custom styles
- ✔ Including PDF files
- ✔ Basic Bib(La)TeX
- ✔ Citation types
- ✔ Bibliography styles
- ✔ Reference managers
- ✔ Literature research
- ✔ Command line, terminal, console, and shell
- ✔ Render/run LaTeX, R, and Quarto via CLI
- ✔ GREP
- ✔ Most common command line commands

Sharing and collaborating

- ✔ Text editors and their uses
- ✔ Version control
- ✔ Git basics
- ✔ Git restoring and reverting
- ✔ GitHub
- ✔ SSH