Eltecon Data Science Course by Emarsys

Introduction

Gábor Kocsis

September 11, 2019

R and RStudio

R and RStudio

- R is a programming language
- RStudio is an IDE for using R
  - IDE: Integrated Development Environment
  - Code editor
  - Built-in automation tools
- R may be used without RStudio, but RStudio cannot be used without R

RStudio projects

- RStudio projects help to divide and organize your work
- Each project has its own working directory and workspace
- You can work in multiple projects at the same time

Relative paths

- Check your working directory with `getwd()`, it might return `"/Users/your_name"`
- If you need, set your working directory with `setwd()`
  - e.g. `setwd("~/eltecon-ds/first_class")`
- When you refer to a file in the working directory, don't need to specify the path
  - `foo <- fread("sales.csv")`
- but, when you refer to a file in another directory, you need to define the path relative to the working directory
  - `foo <- fread("data/sales.csv")`
  - `foo <- fread("../sales.csv")`

Annoying situations

Code calls files that are not available anymore

```
Error in fread("data/raw_data.csv"): File
'data/raw_data.csv' does not exist or is non-readable.
```

Infinite number of versions saved

Don't understand what is happening in the code you wrote last year

```
dt <- fread("data/data.csv")
boo <- foo(dt)
out1 <- doCalculations(boo)
out2 <- doMoreCalculations(boo)
plotFigures(boo)
plotMoreDetailedFigures(boo)
```

Have to change one of your assuptions in your model that was copy pasted throughout the whole project

Working on the same project with others at the same time

Code and data - Guidelines

A good directory structure

- project_directory
  - analysis.R
  - functions.R
  - data
  - figures
  - README.md
  - .gitignore

analysis.R

- Save your analysis script here
- Write comments to separate sections, but do not overuse them
- Use intention revealing names
  - e.g. instead of `ps` use `product_supply`
- Make meaningful distinctions
  - e.g. don't use `cust` and `customers` close to each other
- Use pronounceable names
  - e.g. instead of `purchymd` use `purchase_date`
- Use uppercase letters for constants
  - e.g. `PROPORTION_OF_INCOME_SAVED`, `LEVEL_OF_UTILITY`

functions.R

- Save your functions here
- Functions in long scopes should have short evocative names
  - e.g. `replaceNAWithZero()`
- Functions in small scopes should have long and precise names
  - e.g. `plotPriceElasticityOfGasolineDemand()`

Clean coding

"You shouldn't have to read the body of a function to know what it does. It's name should tell you." - Robert Cecil Martin aka Uncle Bob

data folder

- File names should declare their function
- Always keep a version of the original raw data
- Optionally, use sub-folders in the data folder like
  - raw_data
  - processed_data

Version control with Git

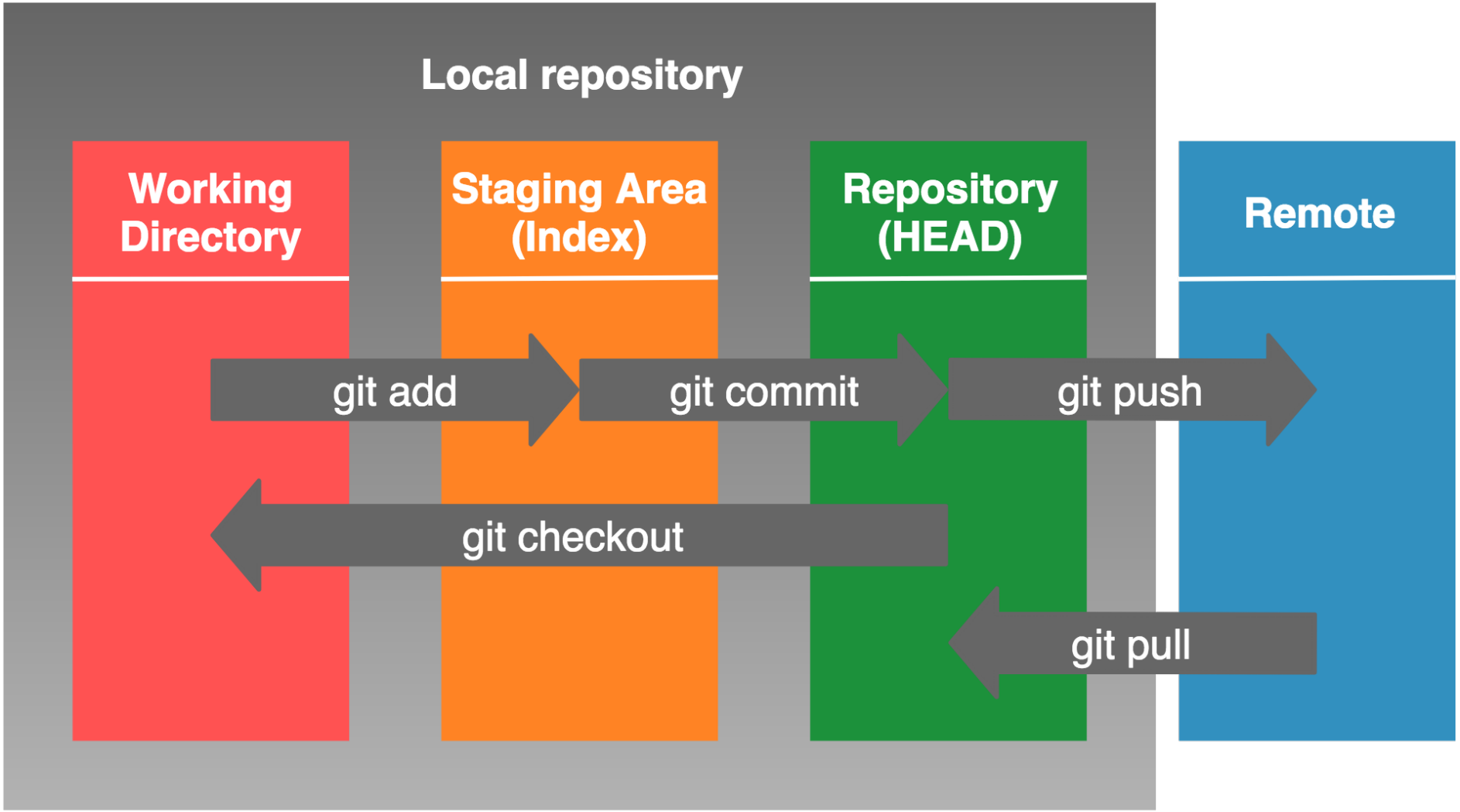
What does git stand for?

- The word “git” is slang for somebody who’s rotten or stupid
- *“I’m an egotistical bastard, so I name all my projects after myself. First Linux, now git.”* - by Linus Torvalds (he wrote the original git version)
- The git README file on github has a longer description

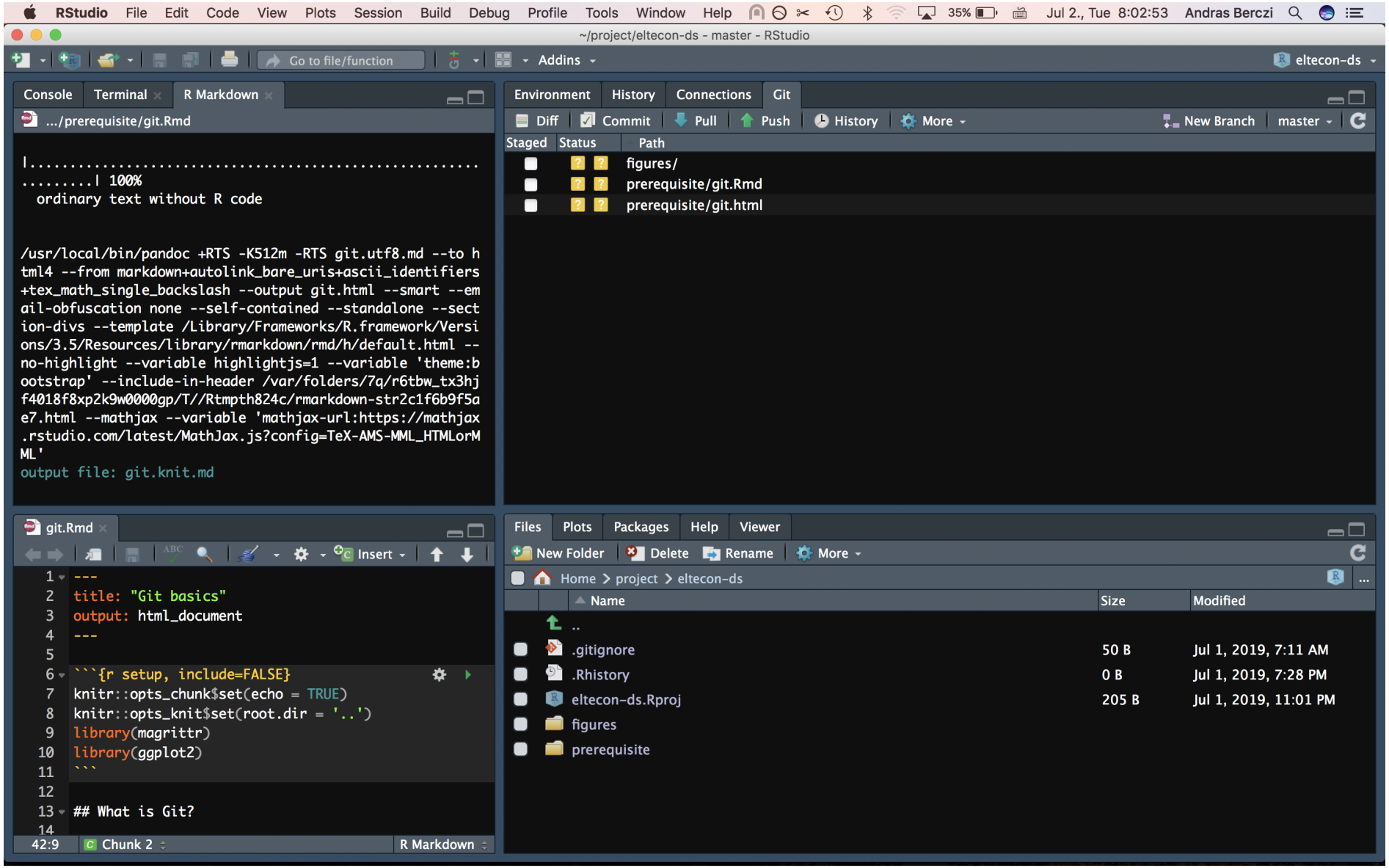
Git is a version control system

- Store code and data under version control
- Benefits
  - Track different versions of the same file
  - Keep only things you currently need
  - Collaborate easily
- Git in RStudio
- Git in command line

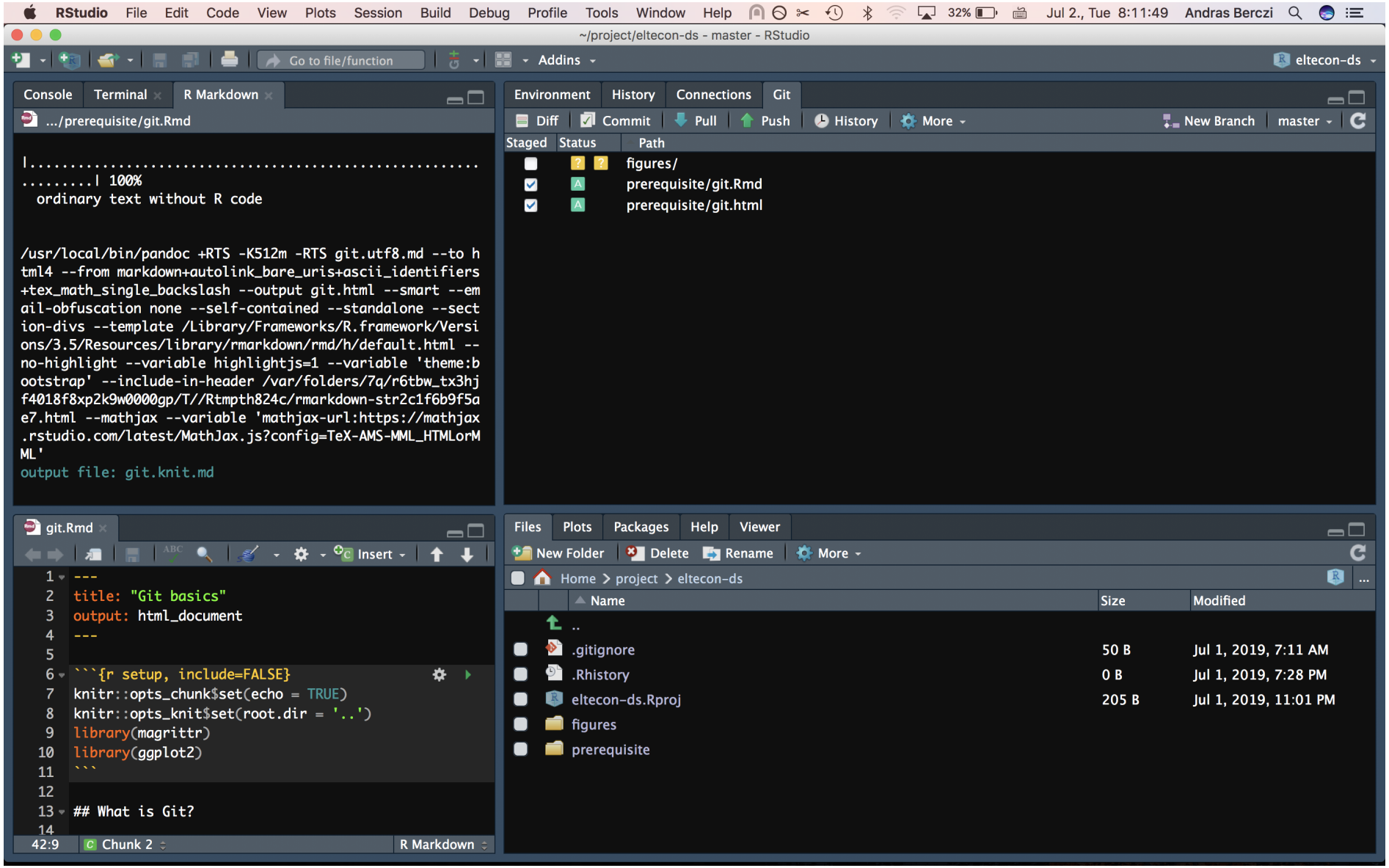
Git stages


Local repository
Working Directory
Staging Area (Index)
Repository (HEAD)
Remote
git add
git commit
git push
git checkout
git pull

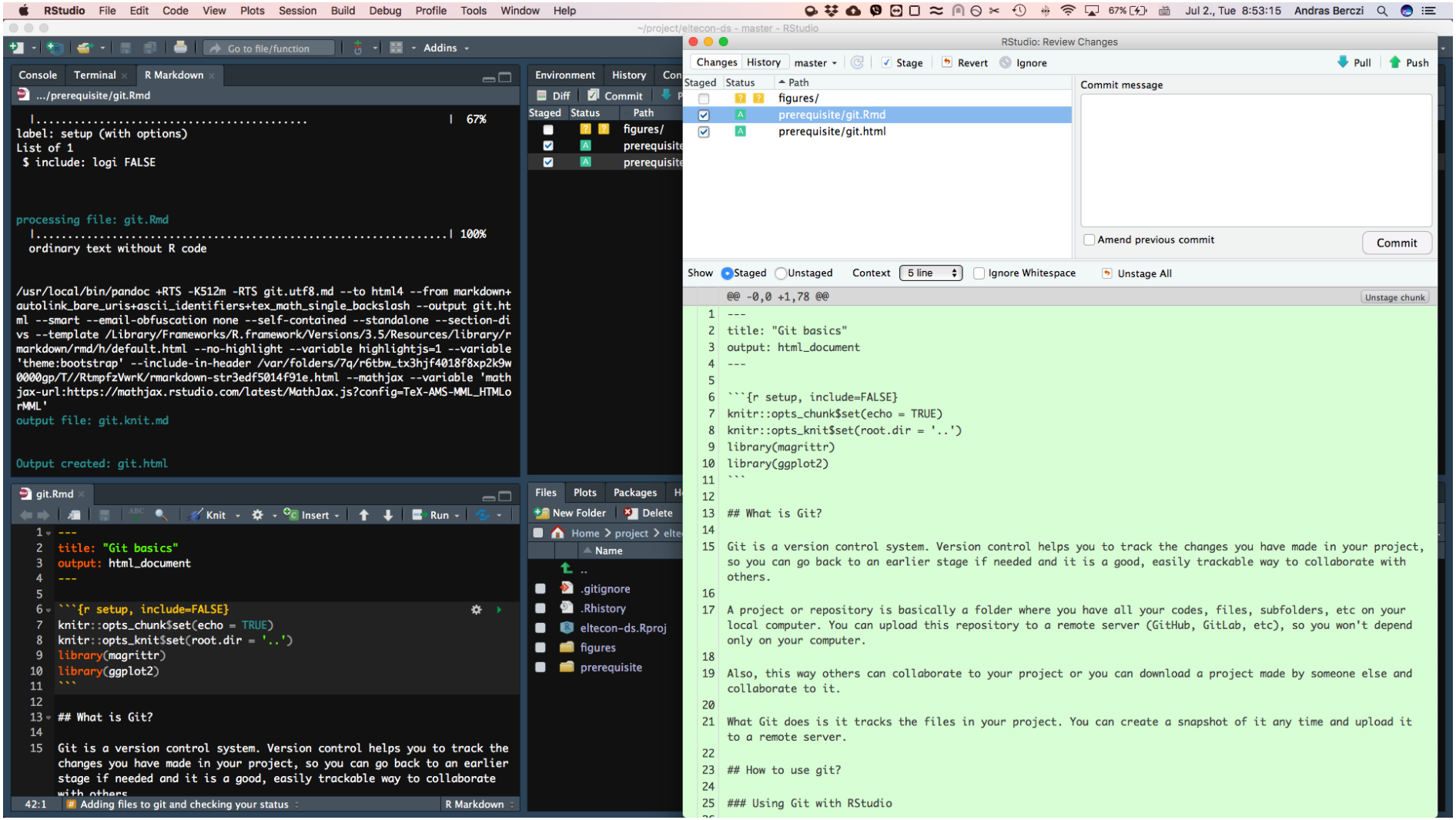
Git status in Rstudio



Git add in Rstudio



Git commit in Rstudio



Git commands

- git clone
- git status
- git add .
- git diff
- git commit -m “Commit message”
- git push
- git pull

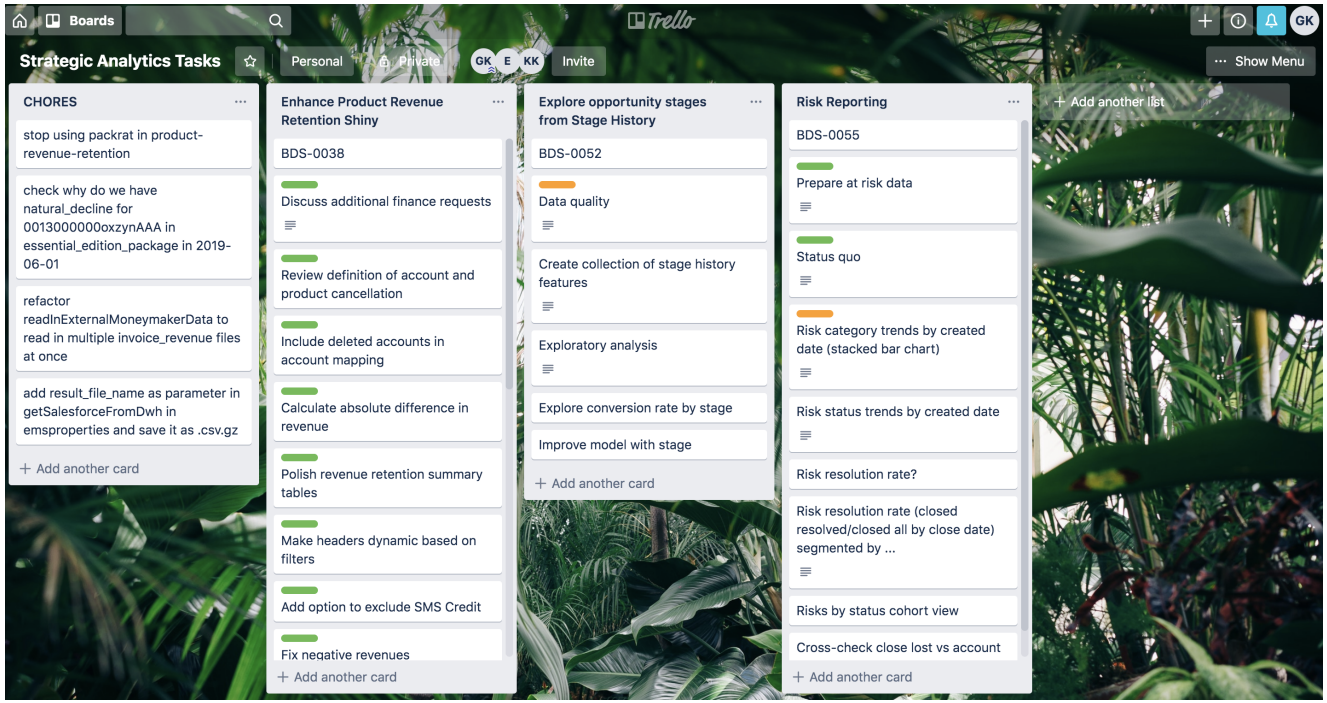
Git tips

- Commit early, commit often
- Use short but descriptive commit messages
- Keep your commits as small as possible, but keep your changes in each commit related
- Run the whole code you modified before pushing it to the repository
- Include files in .gitignore you don't want to track

Task management

Task management

- Manage your tasks with a task management system
- We use Trello and Google Spreadsheets



Summary

Summary

- R and RStudio
  - RStudio projects
  - Relative paths
- Annoying situations
- Code and data - Guidelines
  - A good directory structure
  - analysis.R
  - functions.R
- Version control with Git
- Task management

Sources

Sources

- https://www.r-bloggers.com/structuring-r-projects/
- https://support.rstudio.com/hc/en-us/articles/200526207-Using-Projects
- https://web.stanford.edu/~gentzkow/research/CodeAndData.pdf
- http://www.informit.com/articles/article.aspx?p=1323426
- https://dzone.com/articles/naming-conventions-from-uncle-bobs-clean-code-phil
- https://www.quora.com/What-does-git-stand-for