Course wrap-up and Version control

NENS 230: Analysis Techniques in Neuroscience

Outline

1. When to use Matlab
2. More practice
3. Version Control with Git/Github
4. Course recap

Outline

1. **When to use Matlab**
2. More practice
3. Version Control with Git/Github
4. Course recap

When to use Matlab

Matlab strengths:

- Fast to develop code
- Many built-in functions
- Relatively easy syntax
- Fast for matrix (array) operations
- powerful visualization

Matlab weaknesses:

- Slow (compared to C,C++, other compiled languages)
- Licenses are $$$ and annoying. Anyone running your code must have Matlab installed
- Can't develop standalone applications
- Not a general purpose language

Reasons to Choose Matlab

1. Your community/lab uses it by default
   (and you want your code to be useful to others)
   - biosciences (not bioinformatics)
   - mechanical, electrical, aerospace engineering
   - psychology
   - many others
2. You want to prototype something quickly
3. Exploratory data analysis (w/ numerical data)

Alternatives

Scientific Python

- iPython, scipy, numpy, matplotlib
- very popular, free

R

- Optimized for statistics
- Free
- Widely used by bioinformatics/statisticians

Octave

- Built to be a free Matlab alternative
- Can run basic Matlab code (m-files) without any hassle

Julia

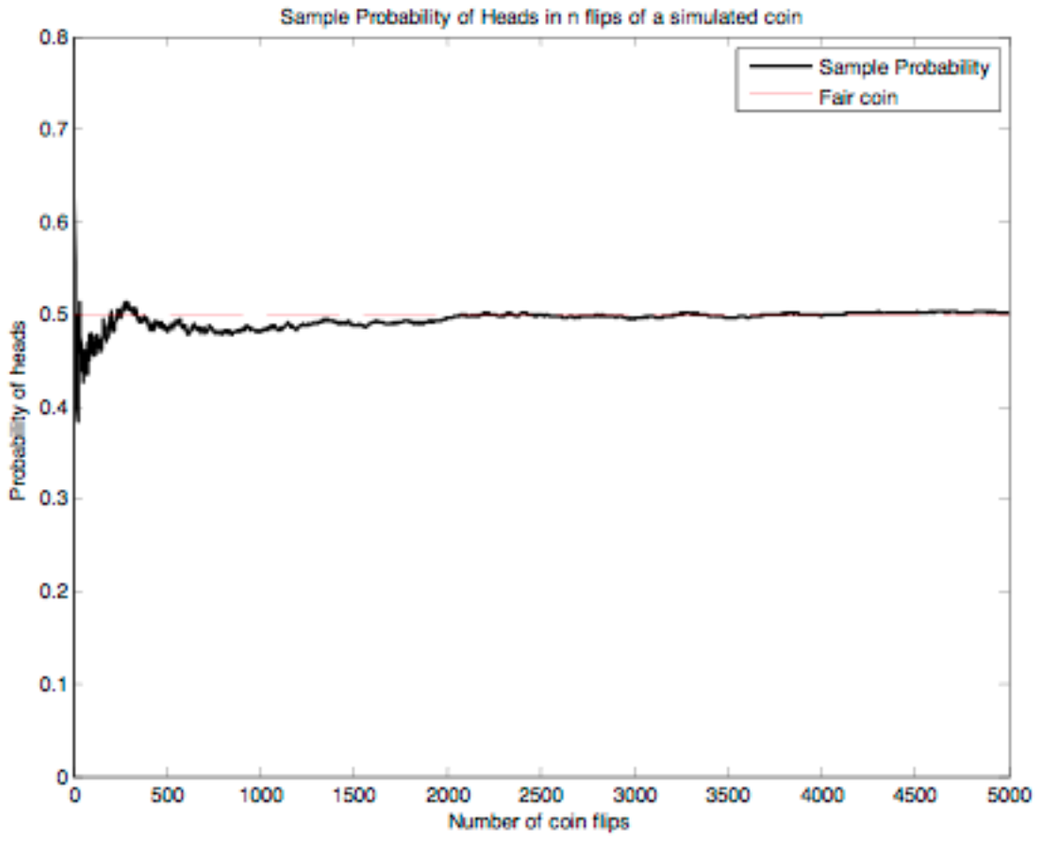
- New project from MIT, free and open source alternative. Fewer compromises (fast, simple, free, general purpose), but still very new.

Outline

1. When to use Matlab
2. **More practice**
3. Version Control with Git/Github
4. Course recap

Choose your own adventure

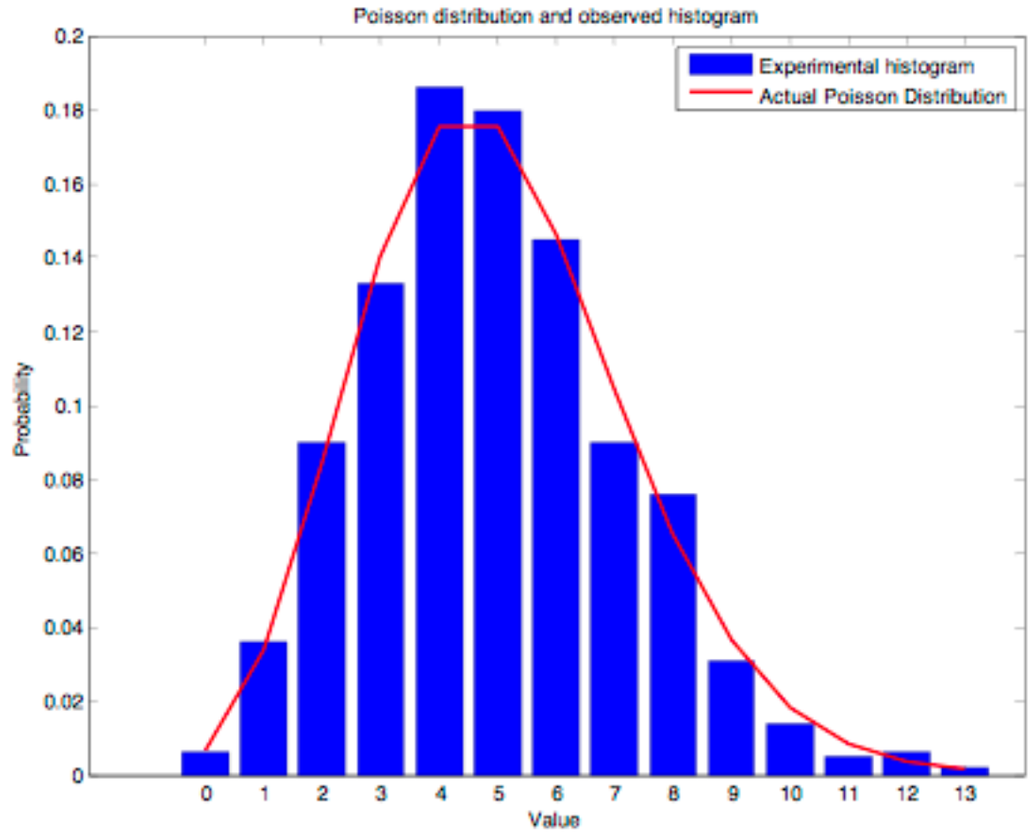
Flipping a coin. Write a script called `coinTest.m` to simulate sequentially flipping a coin 5000 times. Keep track of every time you get 'heads' and plot the running estimate of the probability of getting 'heads' with this coin. Plot this running estimate along with a horizontal line at the expected value of 0.5, as below. This is most easily done without a loop (useful functions: **rand**, **round**, **cumsum**).



http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-094-introduction-to-matlab-january-iap-2010/assignments/

Choose your own adventure

Histogram. Generate 1000 Poisson distributed random numbers with parameter $\lambda = 5$ (**poissrnd**). Get the histogram of the data and normalize the counts so that the histogram sums to 1 (**hist** – the version that returns 2 outputs N and X, **sum**). Plot the normalized histogram (which is now a probability mass function) as a bar graph (**bar**). Hold on and also plot the actual Poisson probability mass function with $\lambda = 5$ as a line (**poisspdf**). You can try doing this with more than 1000 samples from the Poisson distribution to get better agreement between the two.



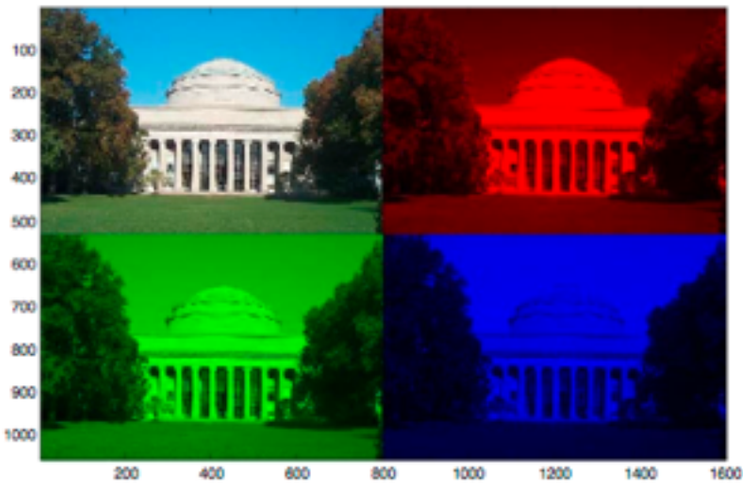
http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-094-introduction-to-matlab-january-iap-2010/assignments/

Choose your own adventure

Image processing. Write a function to display a color image, as well as its red, green, and blue layers separately. The function declaration should be `im=displayRGB(filename)`. `filename` should be the name of the image (make the function work for *.jpg images only). `im` should be the final image returned as a matrix. To test the function, you should put a jpg file into the same directory as the function and run it with the filename (include the extension, for example `im=displayRGB('testImage.jpg')`). You can use any picture you like, from your files or off the internet. Useful functions: **imread**, **meshgrid**, **interp2**, **uint8**, **image**, **axis equal**, **axis tight**.

a. To make the program work efficiently with all image sizes, first interpolate each color layer of the original image so that the larger dimension ends up with 800 pixels. The smaller dimension should be appropriately scaled so that the length:width ratio stays the same. Use **interp2** with cubic interpolation to resample the image.

b. Create a composite image that is 2 times as tall as the original, and 2 times as wide. Place the original image in the top left, the red layer in the top right, the green layer in the bottom left, and the blue layer in the bottom right parts of this composite image. The function should return the composite image matrix in case you want to save it as a jpg again (before displaying or returning, convert the values to unsigned 8-bit integers using **uint8**). *Hint:* To get just a single color layer, all you have to do is set the other two layers to zero. For example if `X` is an MxNx3 image, then `X(:,:,2)=0; X(:,:,3)=0;` will retain just the red layer. Include your code and the final image in your homework writeup. It should look something like this:

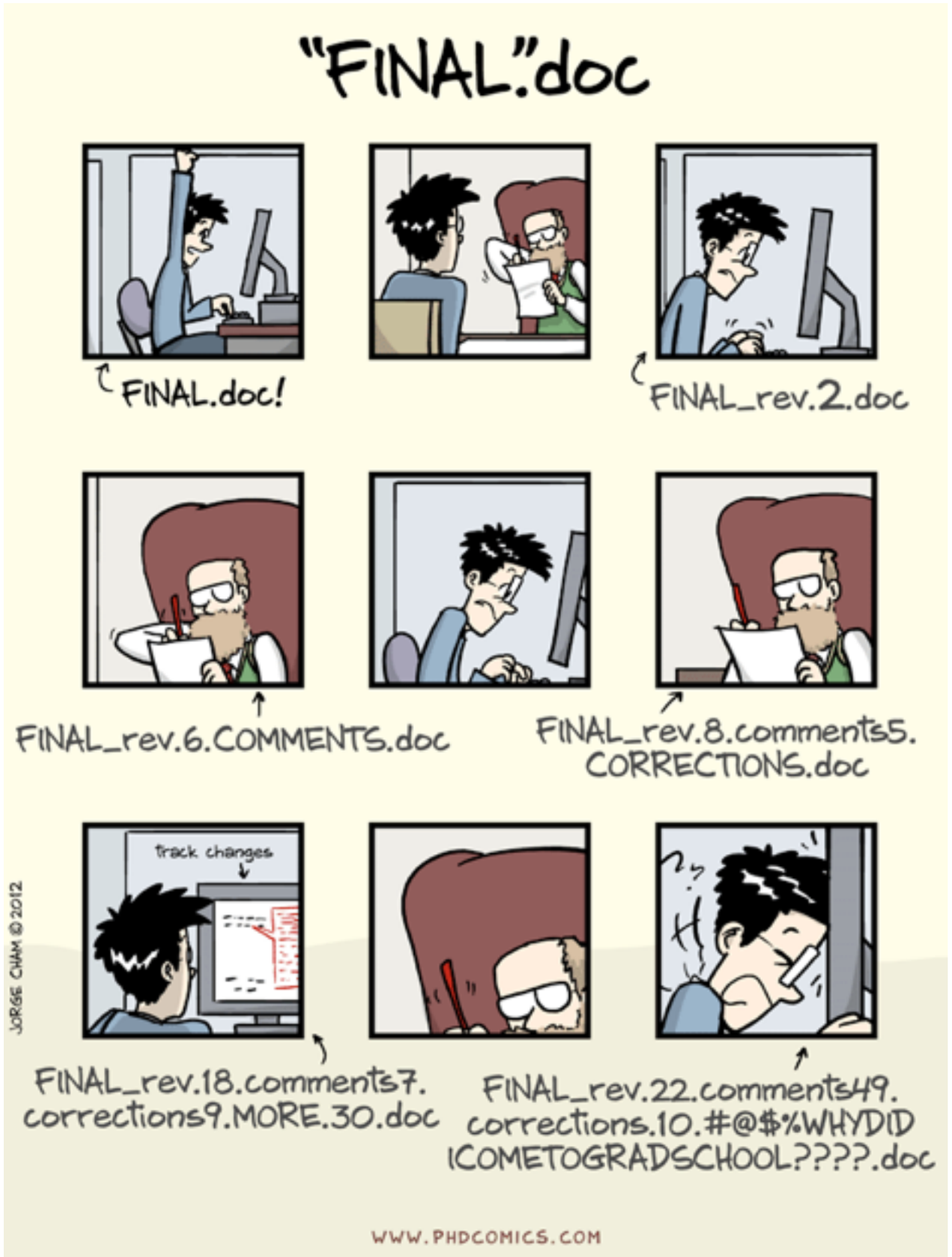


http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-094-introduction-to-matlab-january-iap-2010/assignments/

Outline

1. When to use Matlab
2. More practice
3. **Version Control with Git/Github**
4. Course recap

Version Control



Sound familiar?

There has to be a better way!

“Piled Higher and Deeper” by Jorge Cham

Version Control with 

- What is version control?
  - Method of keeping track of different versions of text files. Useful for:
    - iterative development, experimental code
    - sharing code
- What problems does it solve? Have you ever:
  - Made a change to code, realized it was a mistake and wanted to revert back?
  - Wanted to share your code, or let other people work on your code?
  - Lost code or had a backup that was too old?
  - Had to maintain multiple versions of some code?
  - Wanted to experiment with a new feature without interfering with working code?
  - Wanted to see the difference between two (or more) versions of your code?
  - Wanted to prove that a particular change broke or fixed a piece of code?
  - Wanted to review the history of some code?
  - Wanted to submit a change to someone else's code?

http://stackoverflow.com/questions/1408450/why-should-i-use-version-control

Version Control Software

Several options available

- Git, SVN, Mercurial

Git is de-facto standard and ubiquitous

Often used with Github (online service to host Git repositories

Far more to cover than we have time for. This is worth teaching yourself!

Several useful resources:

- Very simple intro: http://try.github.io
- More detailed: http://git-scm.com/book

git terminology

- Code is stored in a ***repository*** (‘repo’)
- ***Working Tree*** **-** the directory containing your current code
- ***Commit*** **-** When you want to save a snapshot of your project or are finished with some analysis/new feature.
- ***HEAD*** - The name of the currently checked-out commit

starting git

- Can use the command line or a GUI tool (eg: github app)
- Start a git repository:

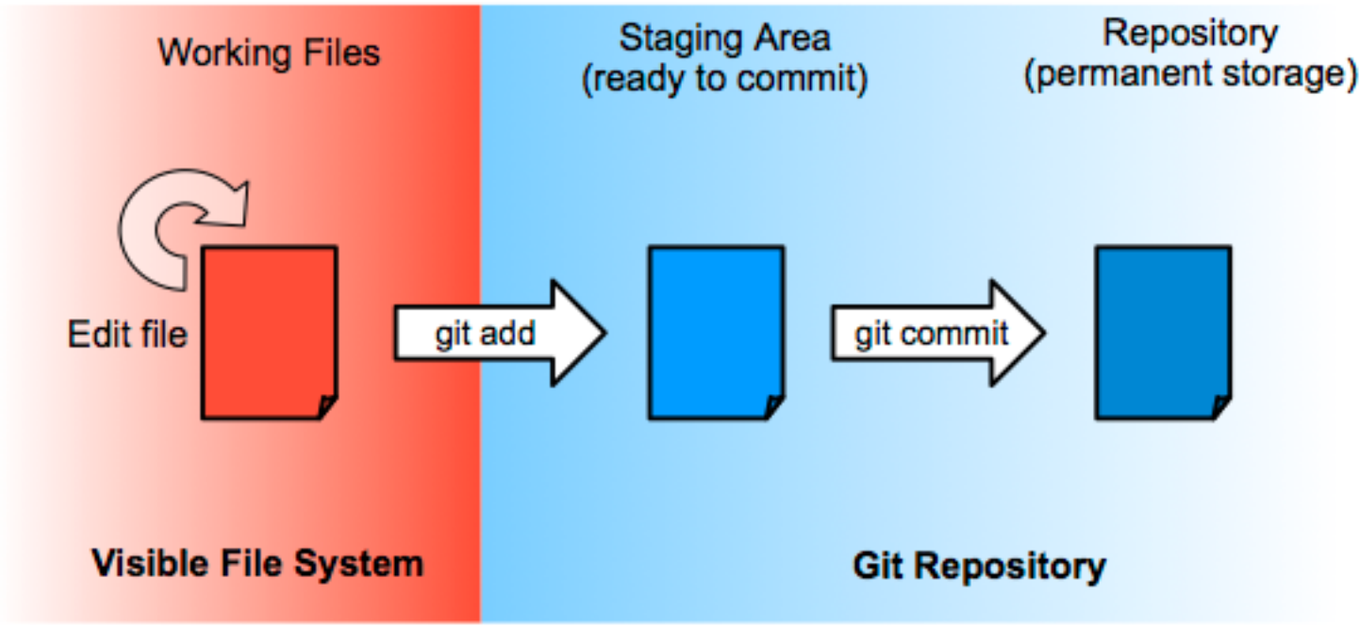
>>> cd /project/folder

>>> git init

>>> git add ./*.m (or specific files)

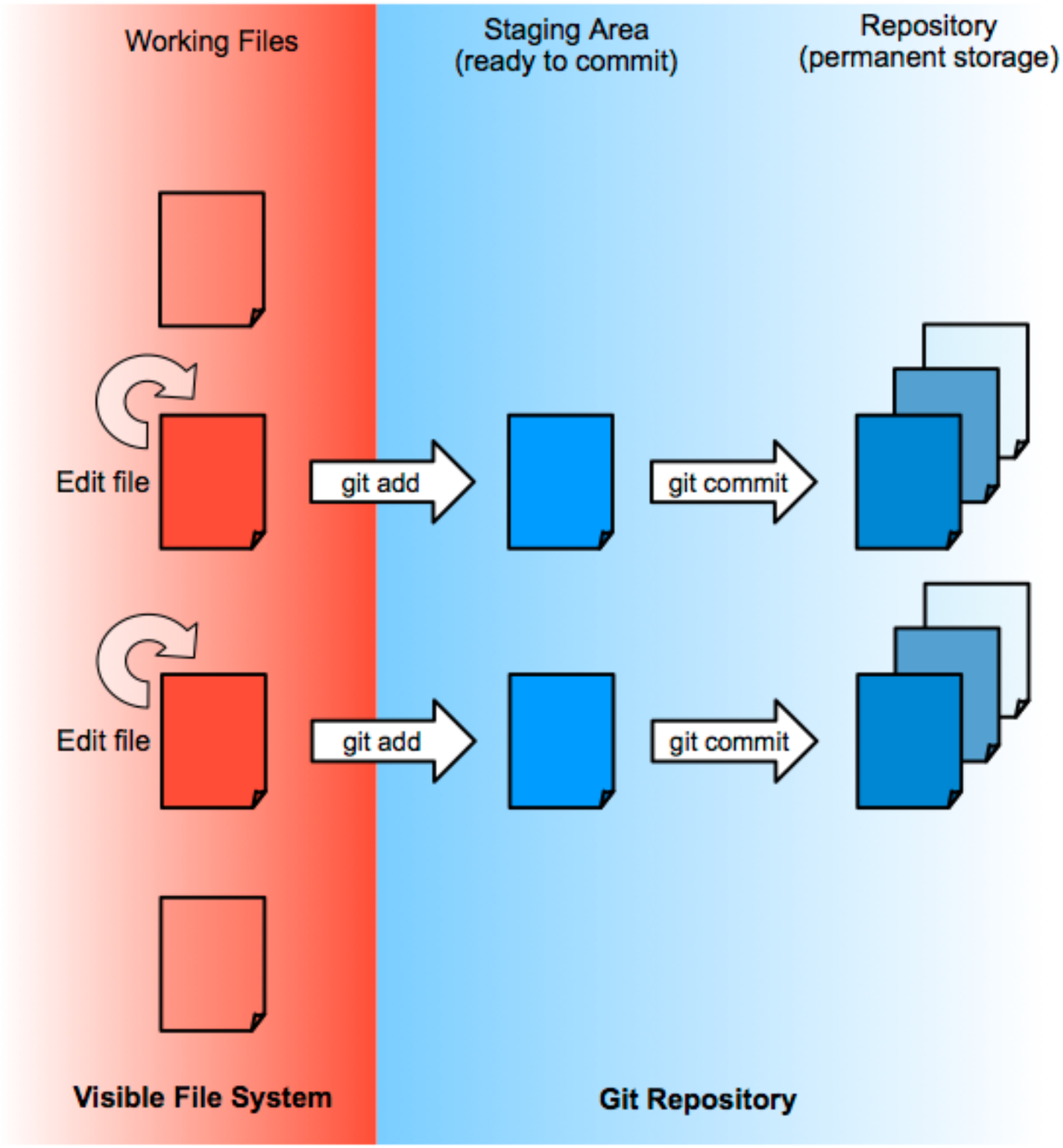
>>> git commit

Git basics



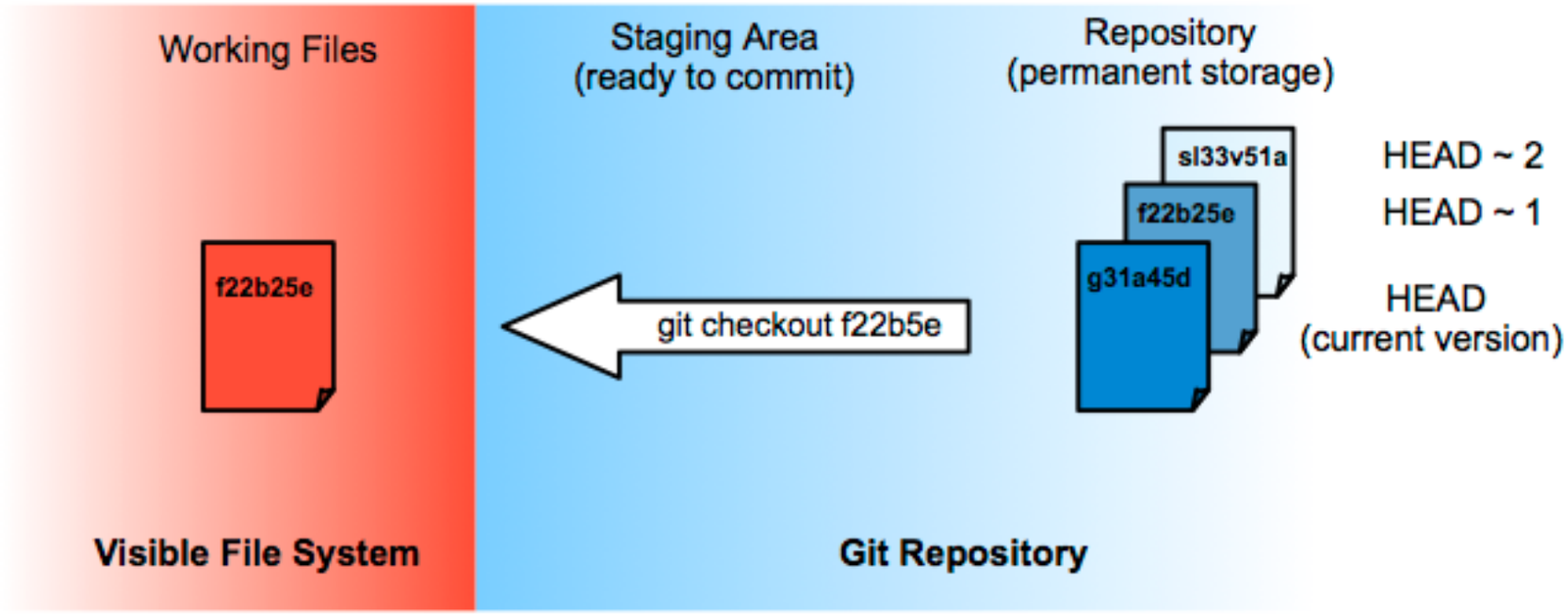
http://software-carpentry.org/v5/novice/git/01-backup.html

Git basics



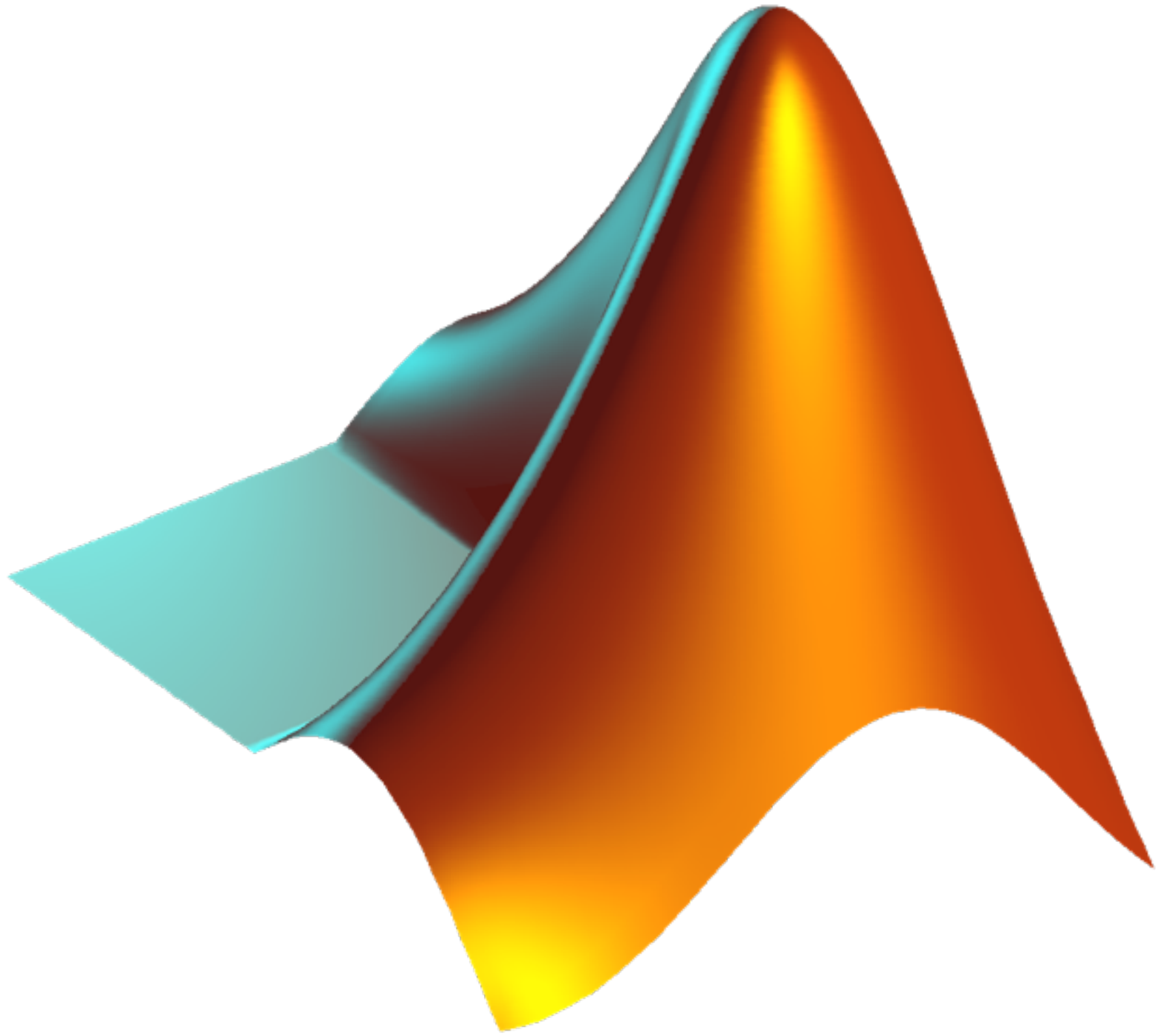
http://software-carpentry.org/v5/novice/git/01-backup.html

Git basics



http://software-carpentry.org/v5/novice/git/01-backup.html

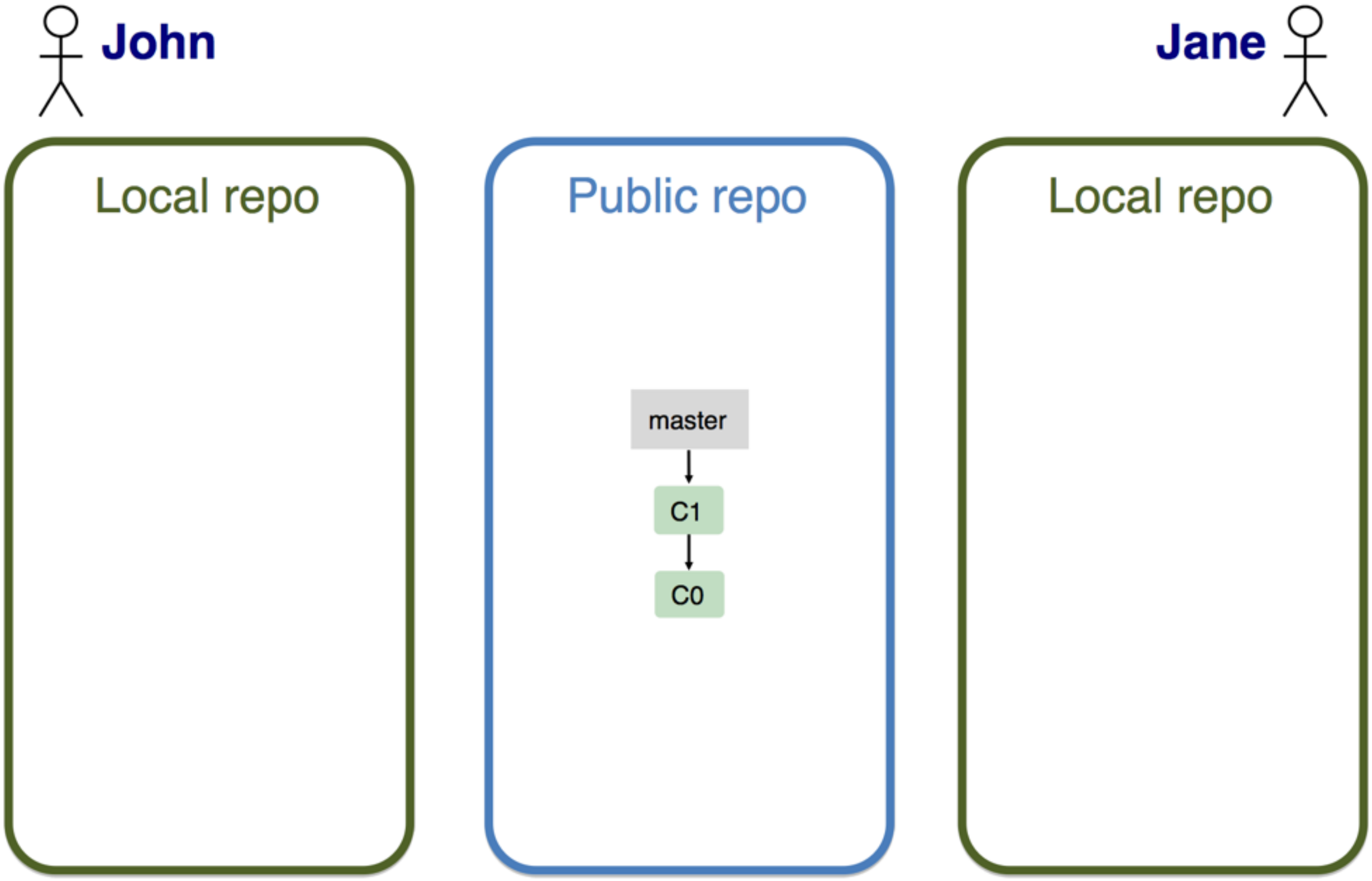
Demo: git



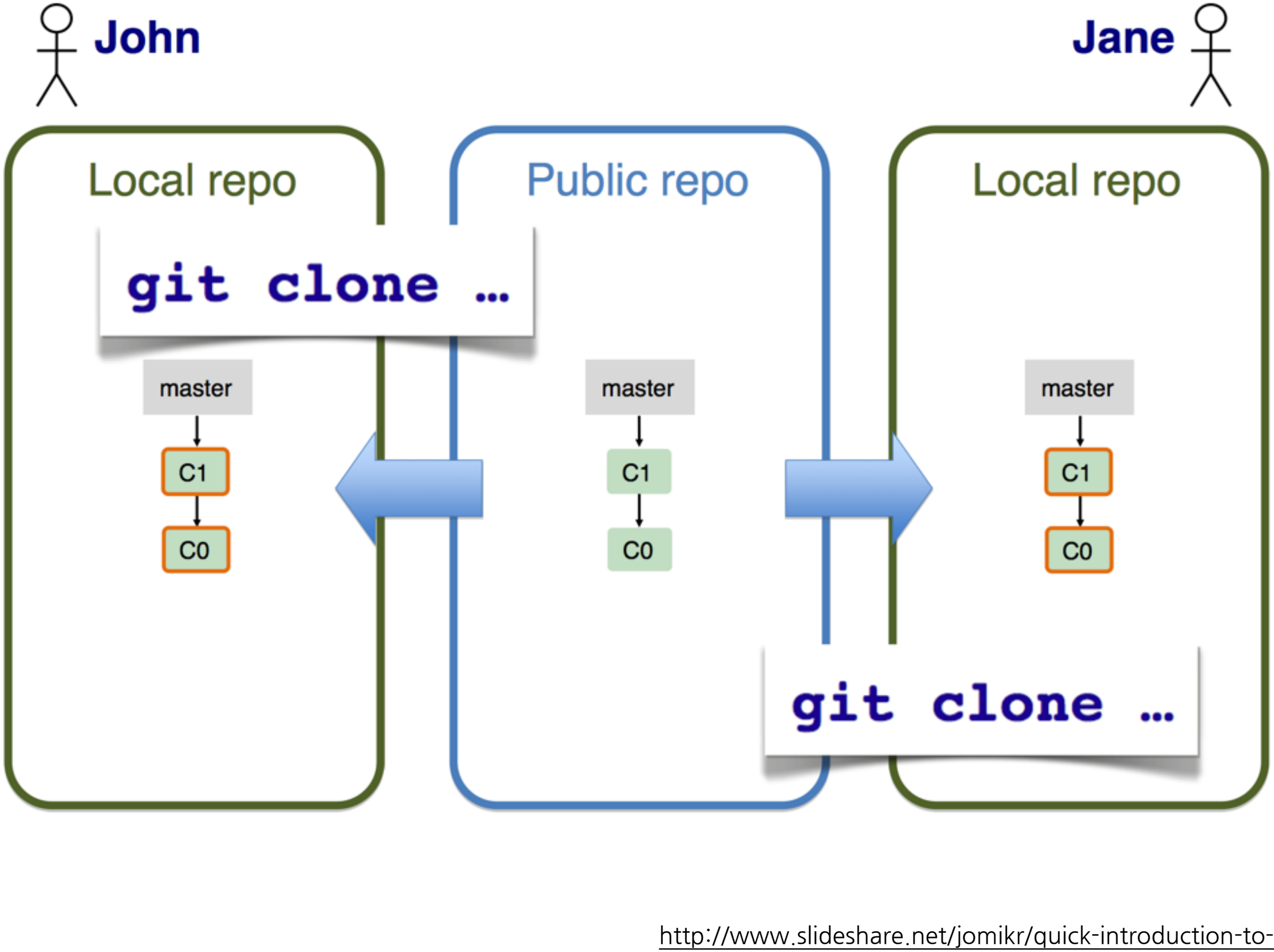
Collaborating with git

- How can you share your local repository with others?

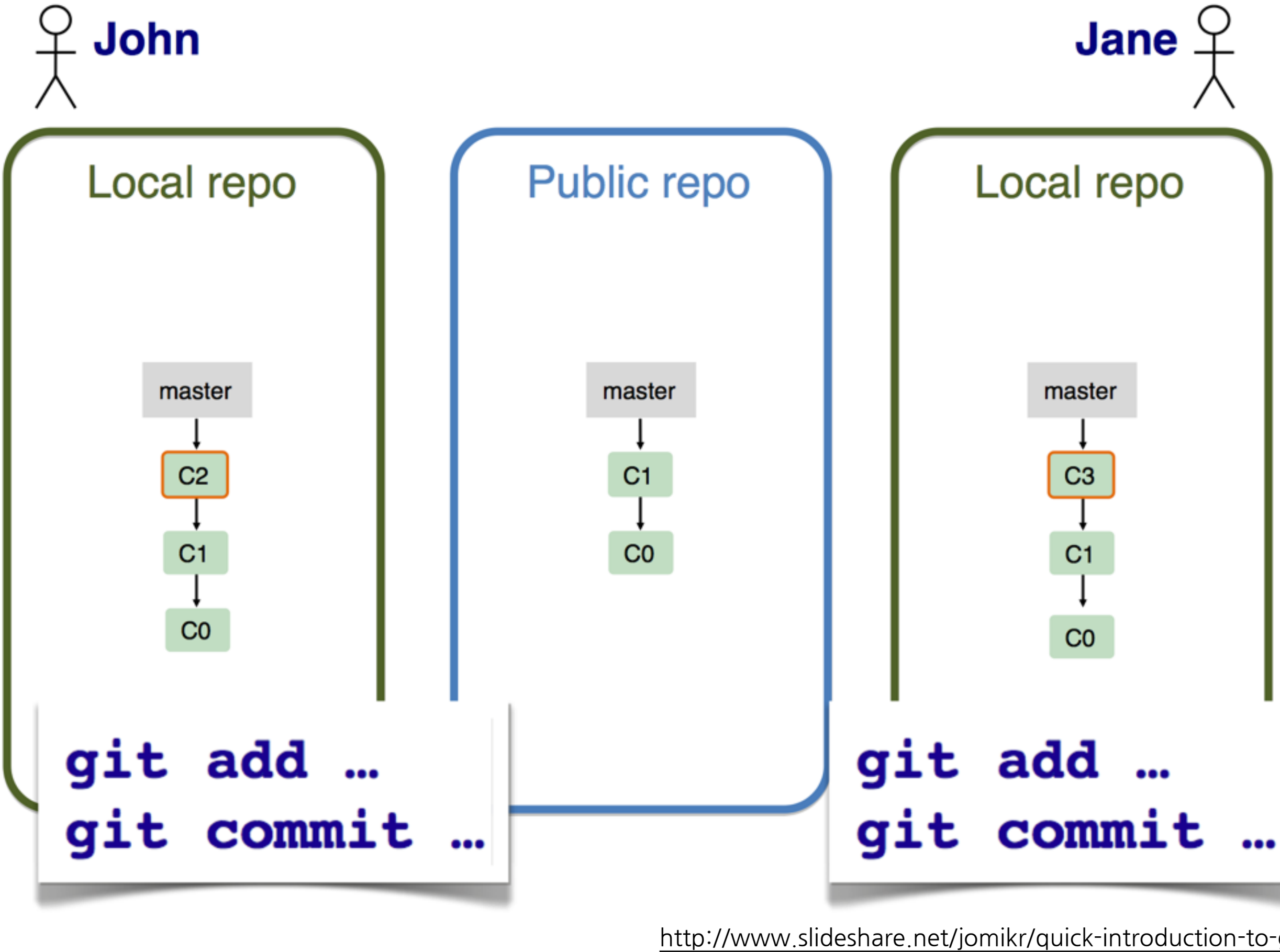
- Two new git commands: push and pull



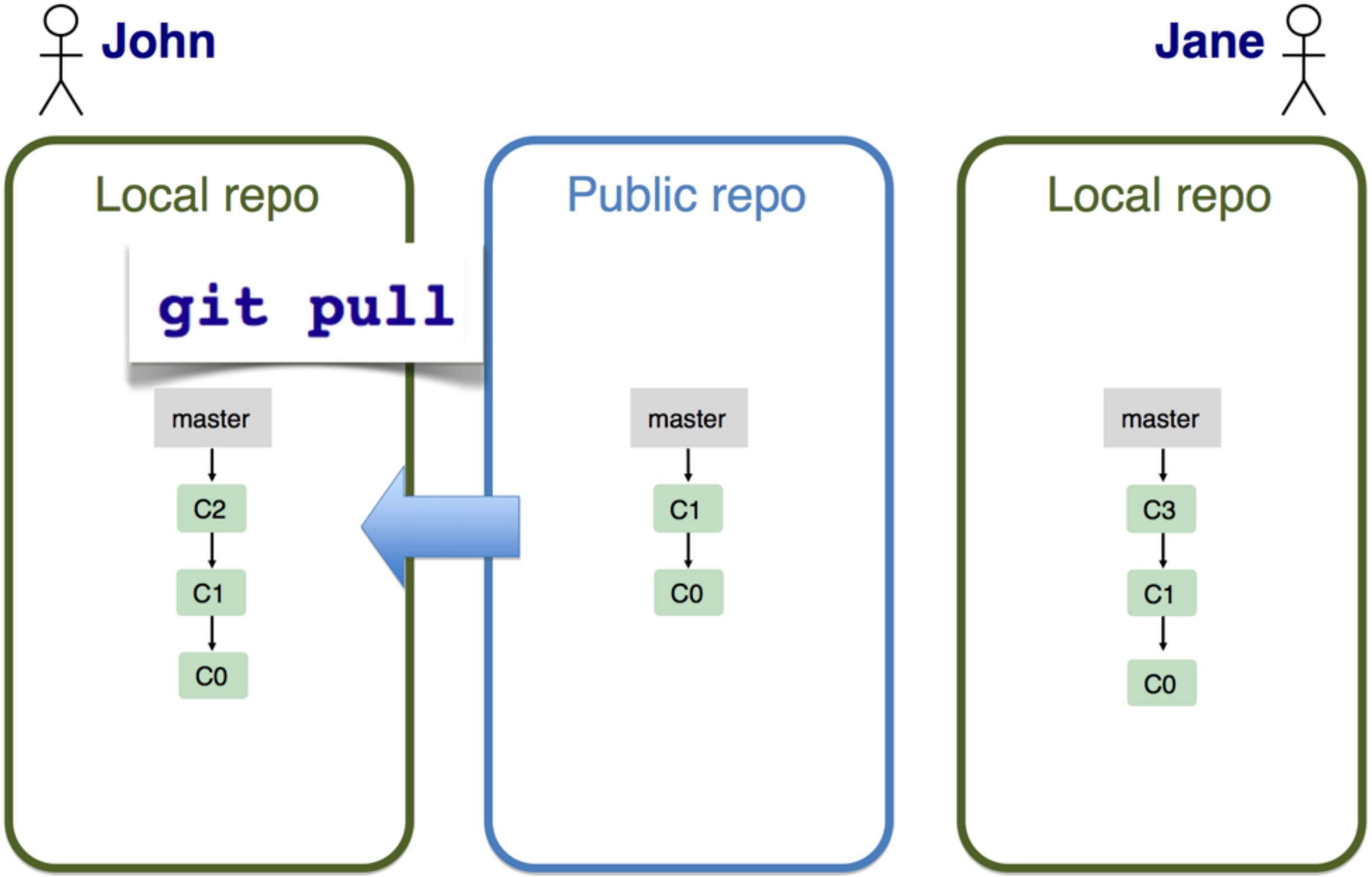
http://www.slideshare.net/jomikr/quick-introduction-to-git



http://www.slideshare.net/jomikr/quick-introduction-to-git

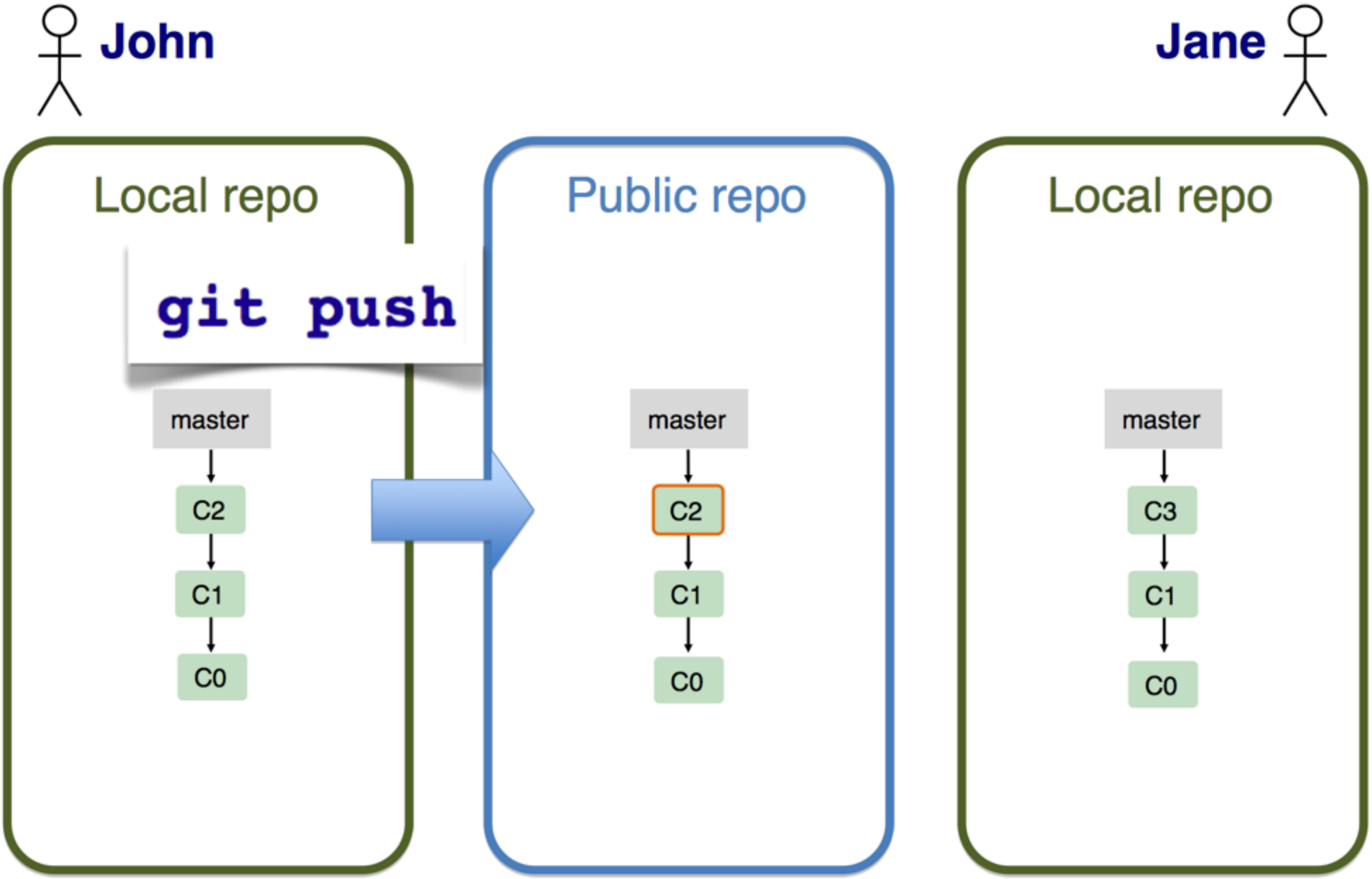


http://www.slideshare.net/jomikr/quick-introduction-to-git

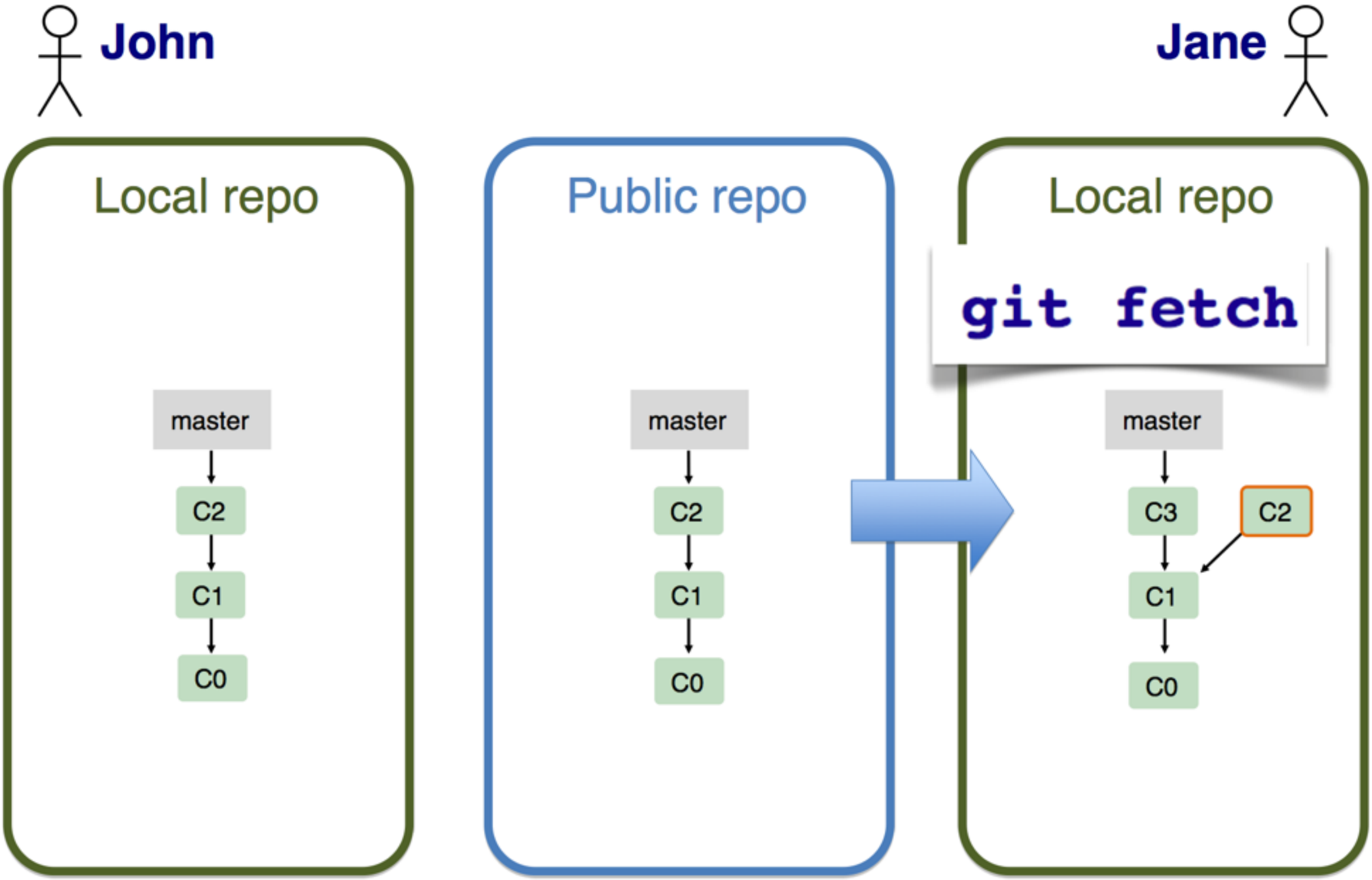


(nothing new to pull)

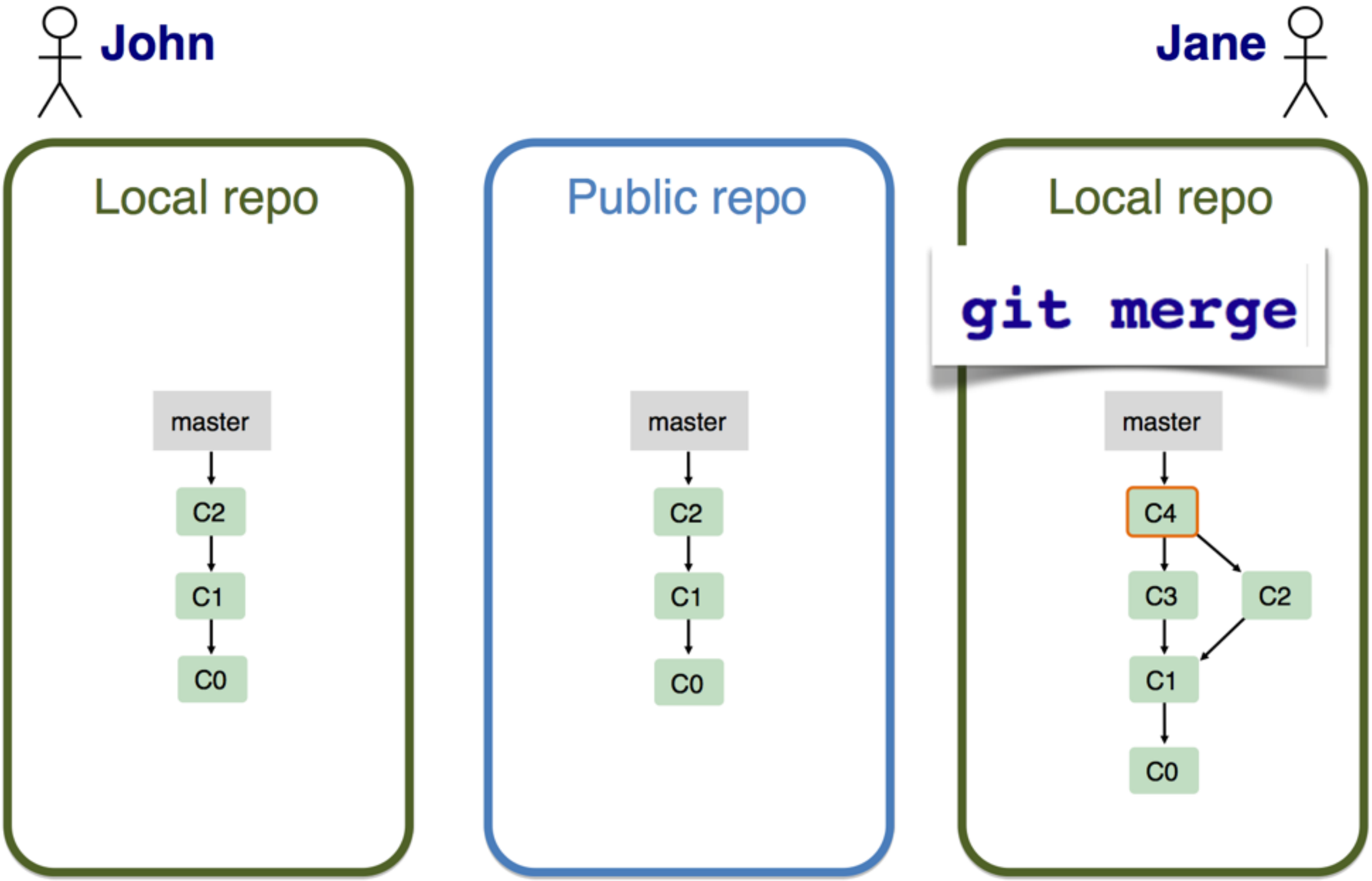
http://www.slideshare.net/jomikr/quick-introduction-to-git



http://www.slideshare.net/jomikr/quick-introduction-to-git

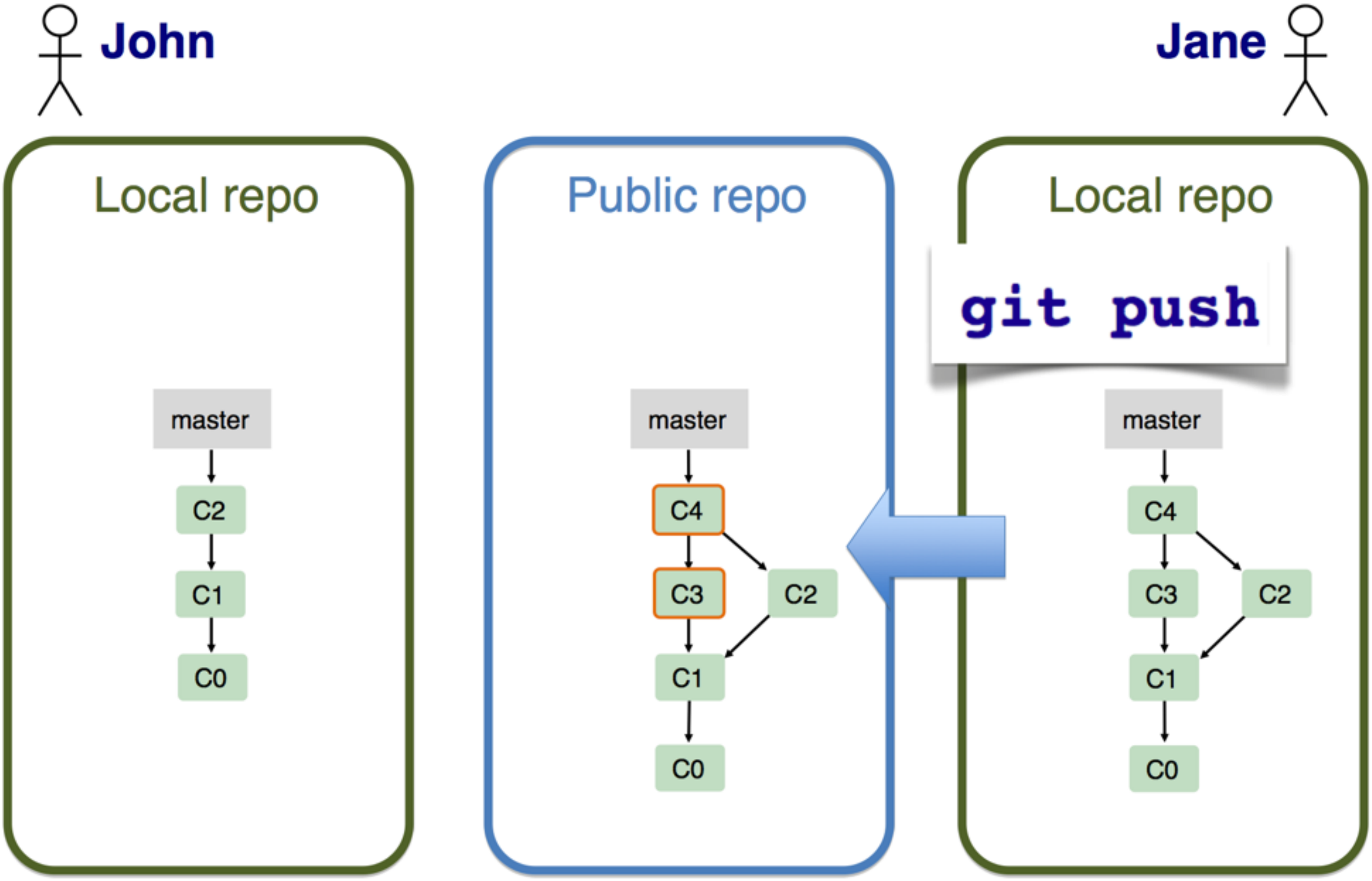


http://www.slideshare.net/jomikr/quick-introduction-to-git

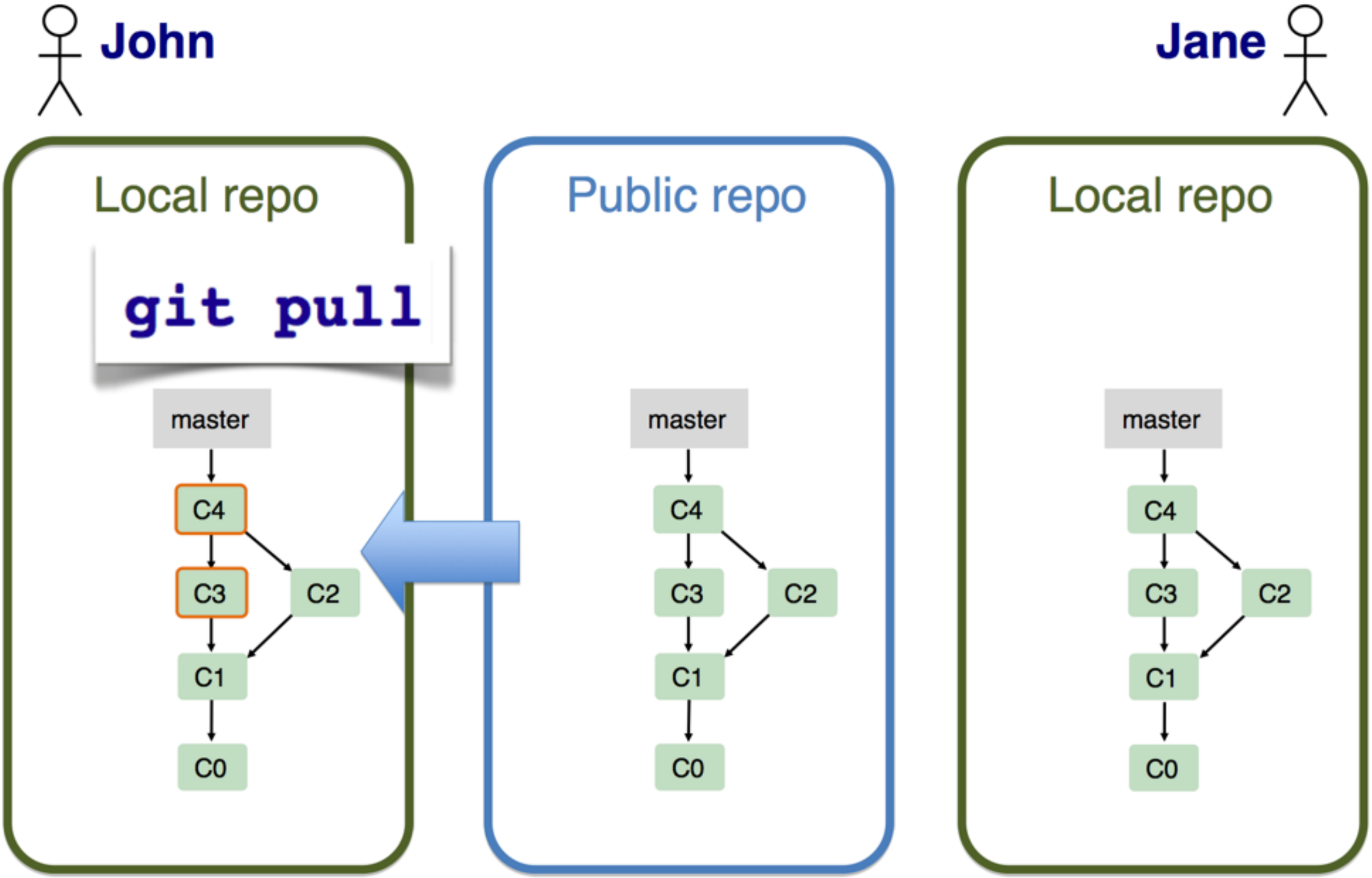


NB: **git pull** = fetch + merge

http://www.slideshare.net/jomikr/quick-introduction-to-git

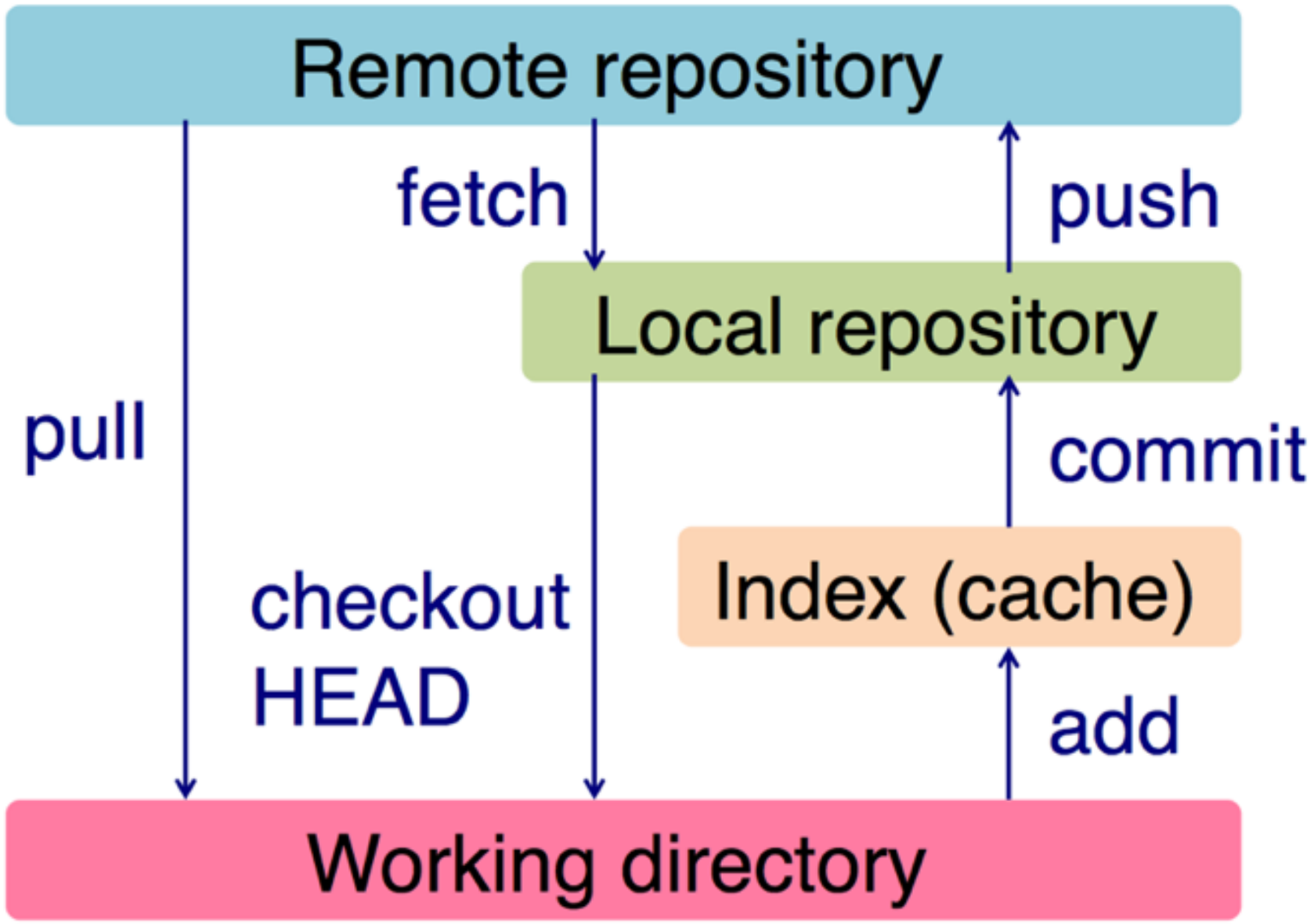


http://www.slideshare.net/jomikr/quick-introduction-to-git



http://www.slideshare.net/jomikr/quick-introduction-to-git

git working model

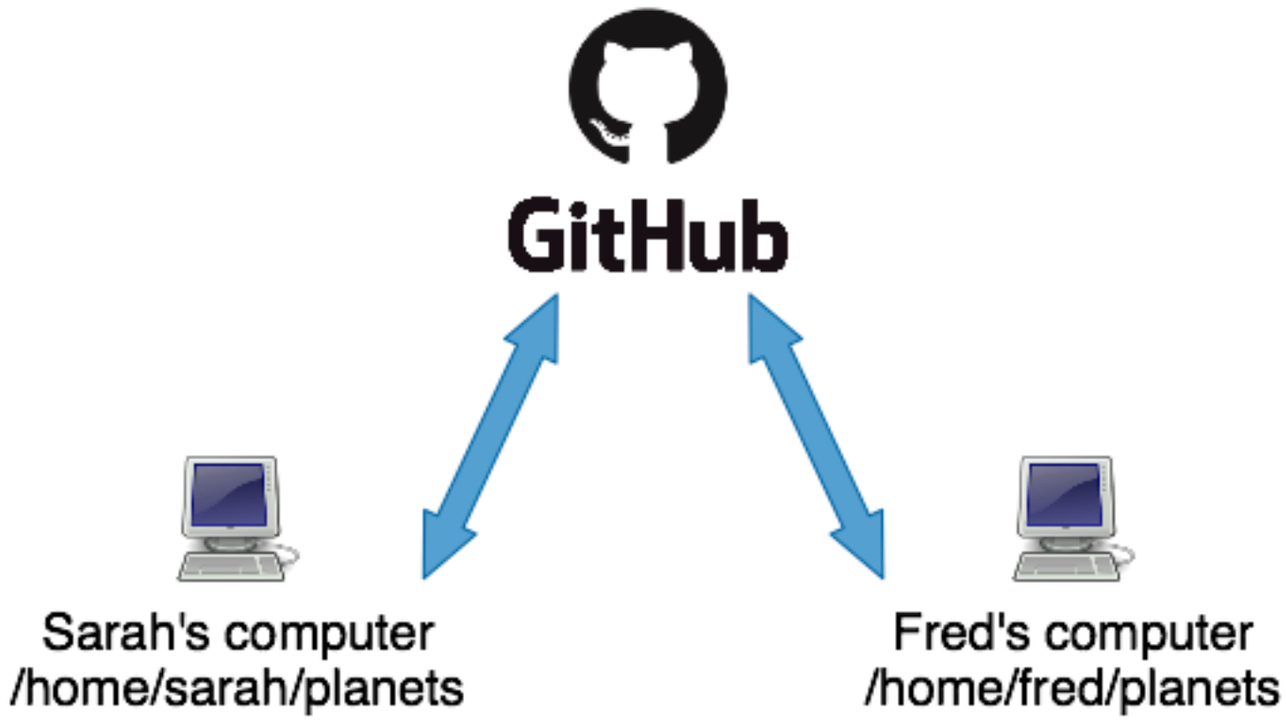


http://www.slideshare.net/jomikr/quick-introduction-to-git

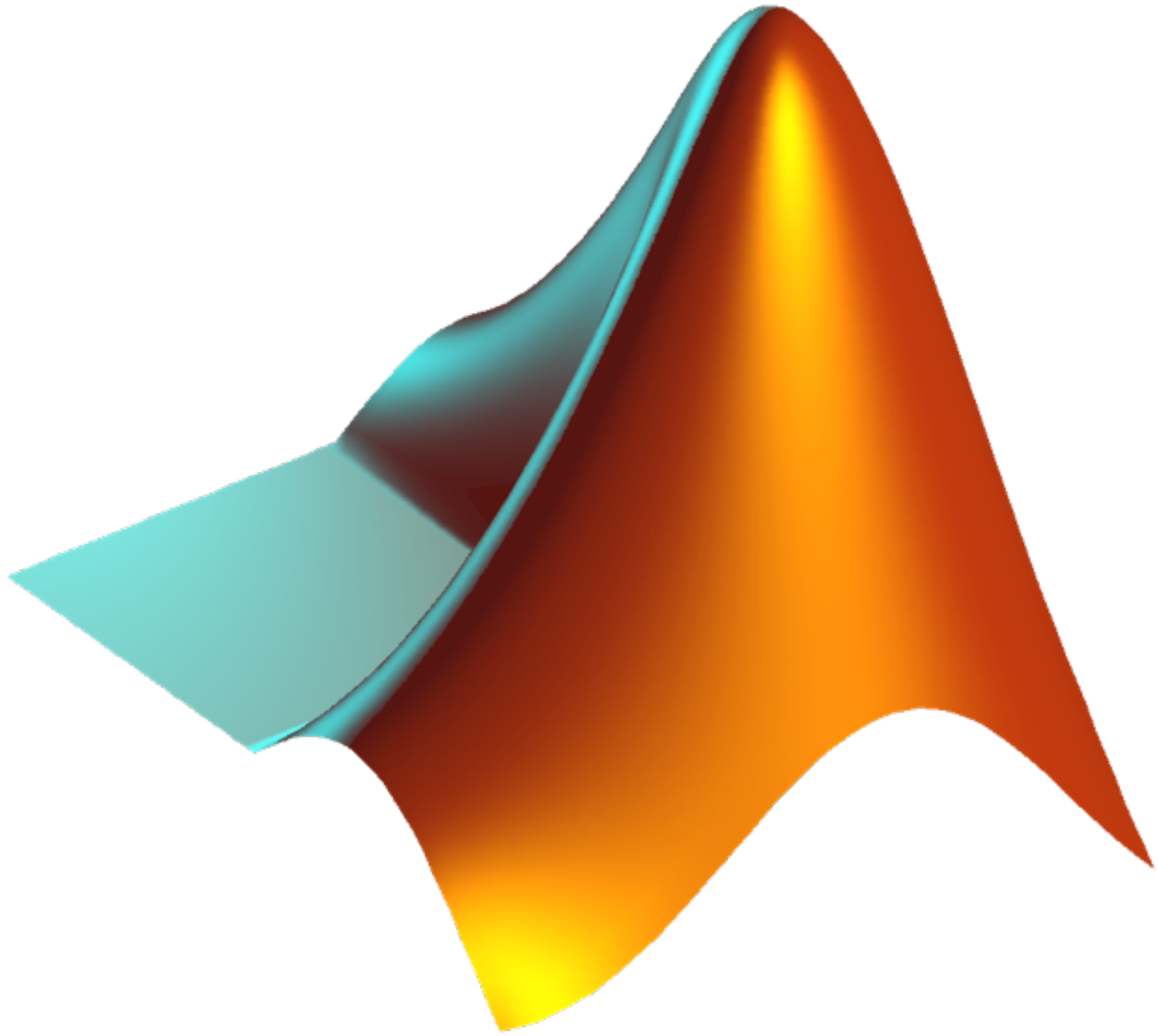
Pushing changes

- after commit, push changes to remote repository

>>> git push



Demo: Github



Outline

1. When to use Matlab
2. More practice
3. Version Control with Git/Github
4. **Course recap**

What we've learned

1. Programming basics
2. Loading, manipulating, slicing data
3. Data visualization
4. Control flow: for loops, if statements
5. Writing good code: comments, spacing, style
6. Statistics, signal processing, image processing
7. Version control