07-120 Introduction to Software Construction

Fall 2023

Michael Hilton and Mayank Goel

Introductions

Michael Hilton

Associate Teaching Professor

Director of SE Minor/Concentration

Developer for 10 years

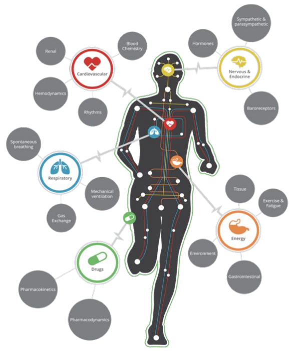
Teach a wide range of classes here at CMU.

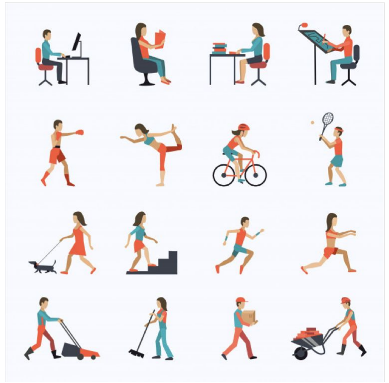
Mayank Goel

Associate Professor (S3D and HCII)

Research and teach: Machine Learning and Sensing

smashlab
SMART SENSING FOR HUMANS





Zeeshan Lakhani (TA)

- PhD Student @ S3D, Studying Programming Langs. (Advised by Frank Pfenning)
- Professional Software Engineer++ for > 11 years
- Currently, work @ Fission on an decentralized, distributed compute engine
- Co-founder and co-organizer of Papers We Love
- 07-120 Office Hours: **Tuesdays ~ 2:30PM - 4PM** @ TCS 223 (zoom available too on-demand)

About this class

07-120 Introduction to Software Construction

Goal of this course: Help students develop the skills to help them become be better at programming

There are things students are supposed to just “pick up”. This class will try to make those hidden skills/processes visible

The class will be highly interactive, and heavily focused participation activities.

No exams, we will have a final project.

This class is an experiment

This is the first time we have offered this class. It is a new concept, so we will experimenting throughout the course

We will be asking you to fill out surveys to help us know how long the homeworks are taking you, and give us feedback so we can adjust the course

Please reach out and talk to us if you have any questions or concerns

Structure of the Course

Most classes will include both a lecture and a collaborative in-class exercise There will be weekly homework assignments for students to practice that week's material.

Evaluation in the course will be approximately as follows:

- Homeworks: 40%
- Final project: 40%
- Participation: 20%

Late Days

For the homeworks, you will have a total of four (4) late days. We do not count weekends or holidays as late days. You may use up to two (2) late days per assignment. To use a late day, you need to to message the course staff in a private message on slack, informing us that you will be using a late day.

Participation

We will not be grading attendance, but most classes will have some participation activity.

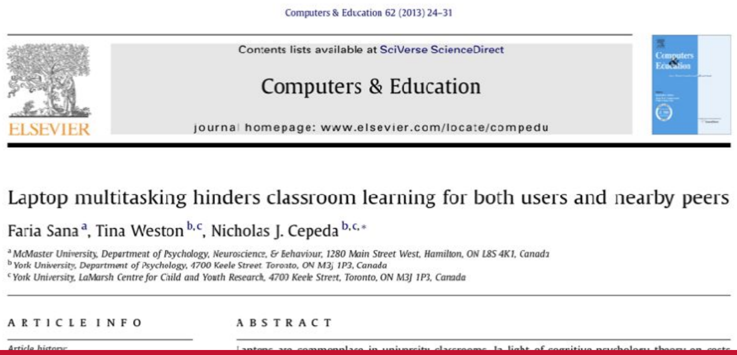
You will not be required to give impromptu speech in front of the class.

You must be present to get participation points.

We will drop your worst two (2) participation points (you get two free misses)

Laptop Usage

"...participants who multitasked on a laptop during a lecture scored lower on a test compared to those who did not multitask, and participants who were in direct view of a multitasking peer scored lower on a test compared to those who were not. The results demonstrate that **multitasking on a laptop poses a significant distraction to both users and fellow students and can be detrimental to comprehension of lecture content.**"


Computers & Education 62 (2013) 24–31

Contents lists available at SciVerse ScienceDirect

Computers & Education

journal homepage: www.elsevier.com/locate/compedu

ELSEVIER

Laptop multitasking hinders classroom learning for both users and nearby peers

Faria Sana[a], Tina Weston[b,c], Nicholas J. Cepeda[b,c,*]

[a] McMaster University, Department of Psychology, Neuroscience, & Behaviour, 1280 Main Street West, Hamilton, ON L8S 4K1, Canada
[b] York University, Department of Psychology, 4700 Keele Street, Toronto, ON M3J 1P3, Canada
[c] York University, LaMarsh Centre for Child and Youth Research, 4700 Keele Street, Toronto, ON M3J 1P3, Canada

ARTICLE INFO

ABSTRACT

Course Logistics

Infrastructure

- Canvas (and Gradescope) homework, grades, other material
- Slack for communication and collaboration.
- Git/Github for coding and turning in work

Logistics:

- Lecture in-person only
- Office Hours in person by default, but on zoom by arraignment.
    - Michael Hilton (TCS 342) 1-2pm Monday
    - Mayank Goel (TCS 340) 10-11am Thursday
    - Zeeshan Lakhani (TCS 223) 2:30-4:00pm Tuesday

If you want to talk to us, **DM/email ALL INSTRUCTORS at once**. Trust me, it's faster.

Version Control

Have you ever seen this?

- Report.txt
- Report_2.txt
- Report_final.txt
- Report_final_2.txt
- Report_final_final.txt
- Report_asjklasdf.txt

Would you like to have save/restore functionality for life?

Version Control

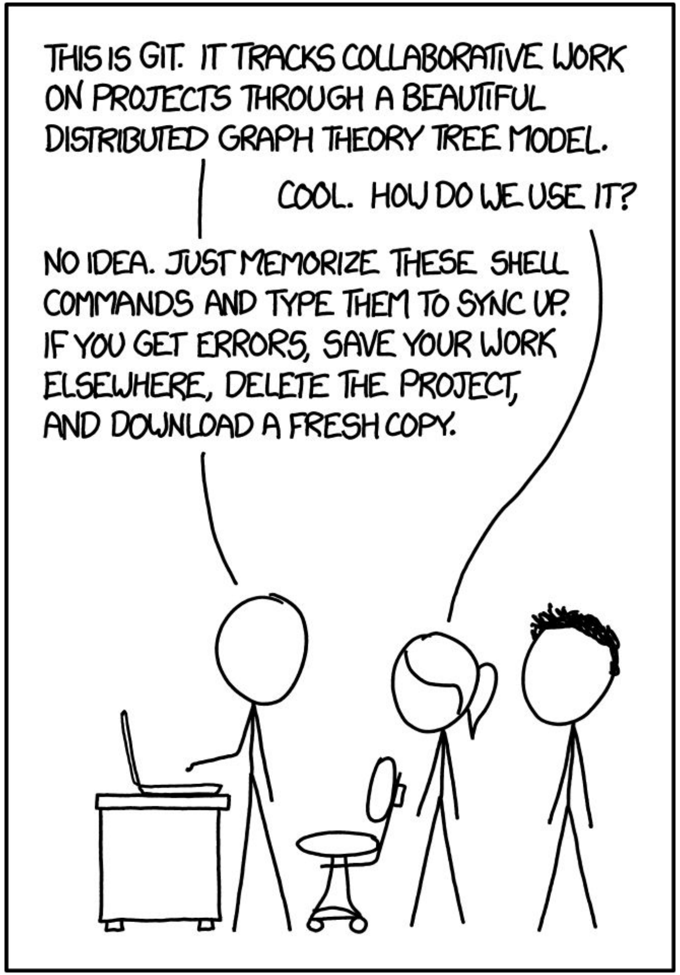
Version Control Software

Version control software is software to track and manage changes to software over time.

The most common VCS system is “git” and it is often used in conjunction with GitHub.

Git was developed by Linus Torvalds for the purposes of supporting the development of Linux

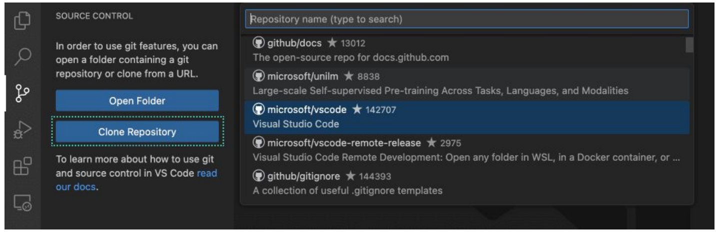
Git Fundamentals


THIS IS GIT. IT TRACKS COLLABORATIVE WORK ON PROJECTS THROUGH A BEAUTIFUL DISTRIBUTED GRAPH THEORY TREE MODEL.
COOL. HOW DO WE USE IT?
NO IDEA. JUST MEMORIZE THESE SHELL COMMANDS AND TYPE THEM TO SYNC UP. IF YOU GET ERRORS, SAVE YOUR WORK ELSEWHERE, DELETE THE PROJECT, AND DOWNLOAD A FRESH COPY.

Clients

Git command line <- I will demo this today. This is the most common (and useful) for power users

Git VSCode plugin <- ok to use, as long as you understand what is going on.



In order to use git features, you can open a folder containing a git repository or clone from a URL.

To learn more about how to use git and source control in VS Code read our docs.

github/docs ★ 13012
The open-source repo for docs.github.com

microsoft/unilm ★ 8838
Large-scale Self-supervised Pre-training Across Tasks, Languages, and Modalities

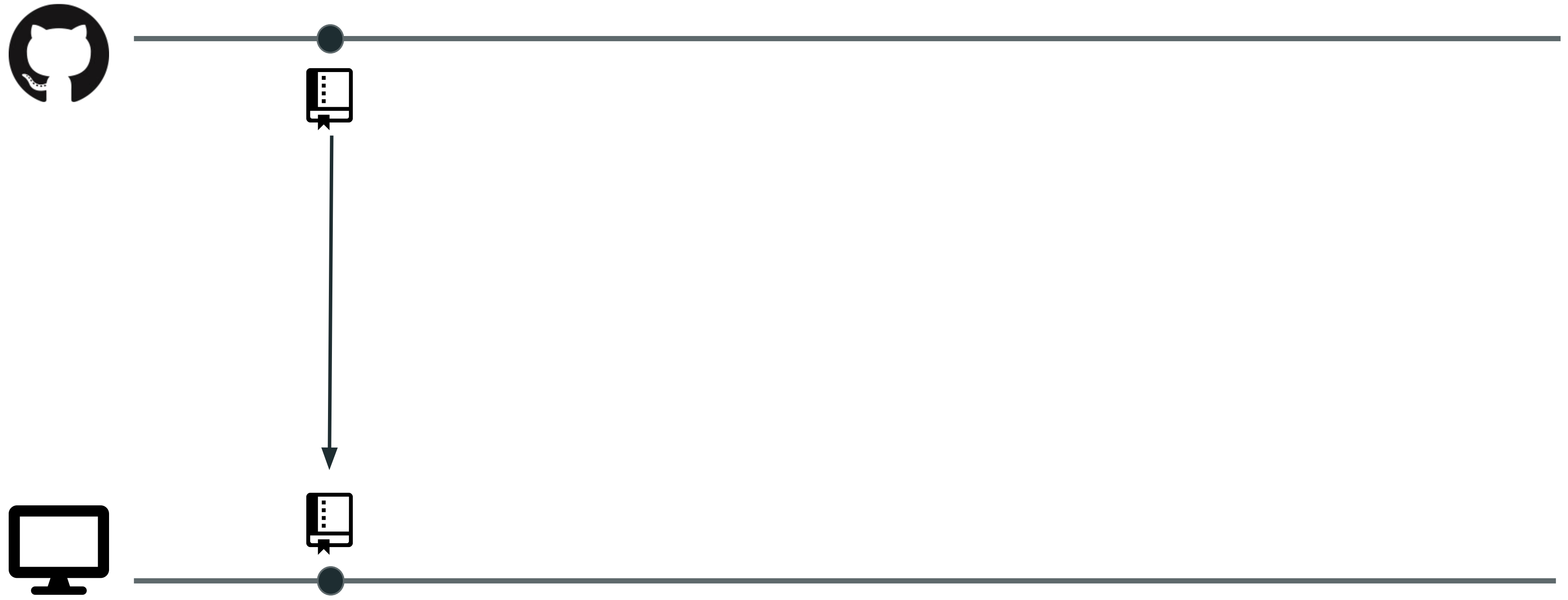
microsoft/vscode ★ 142707
Visual Studio Code

microsoft/vscode-remote-release ★ 2975
Visual Studio Code Remote Development: Open any folder in WSL, in a Docker container, or ...

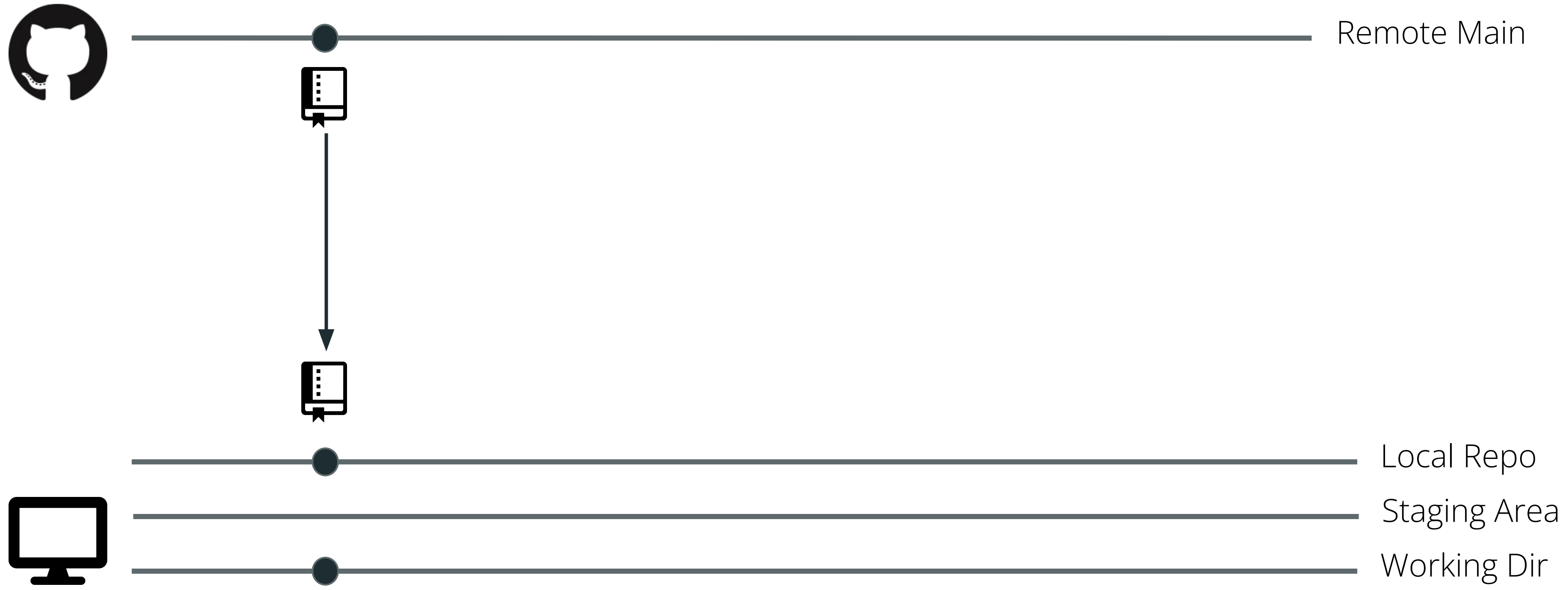
github/gitignore ★ 144393
A collection of useful .gitignore templates

GitHub Desktop: DO NOT USE!! THERE BE DRAGONS!!

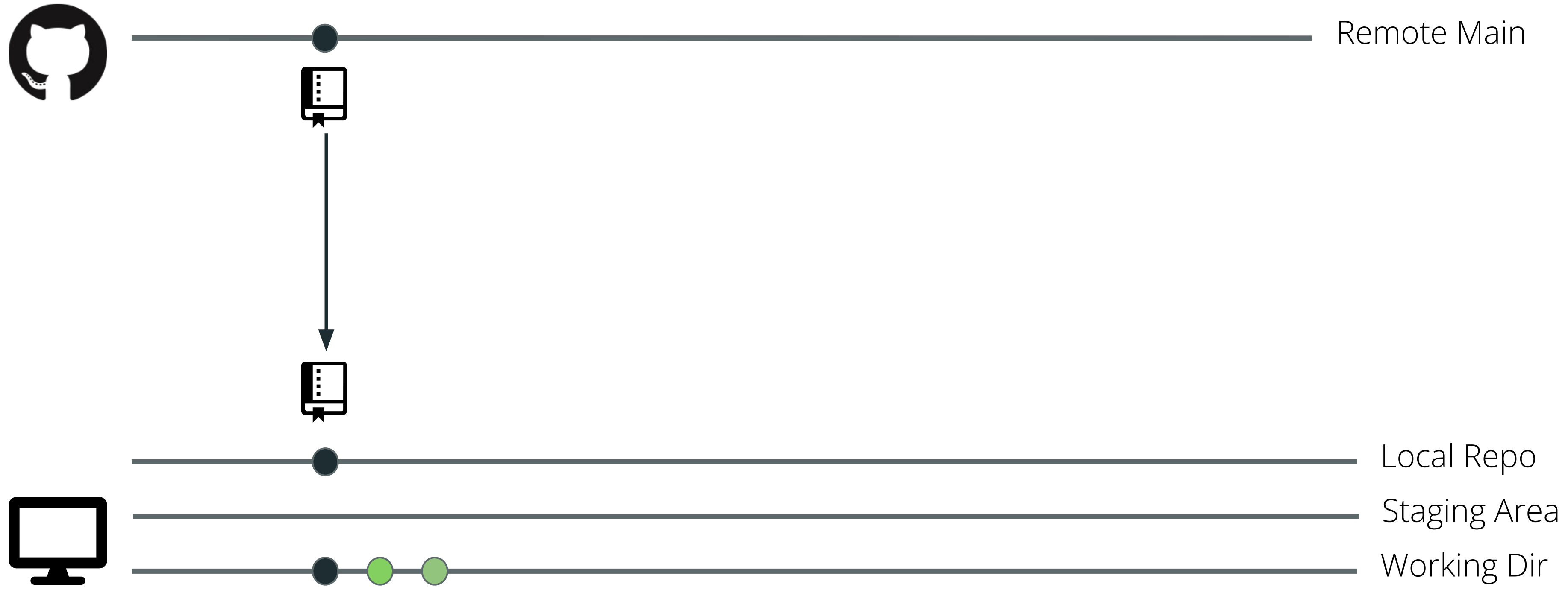
> git clone REPO



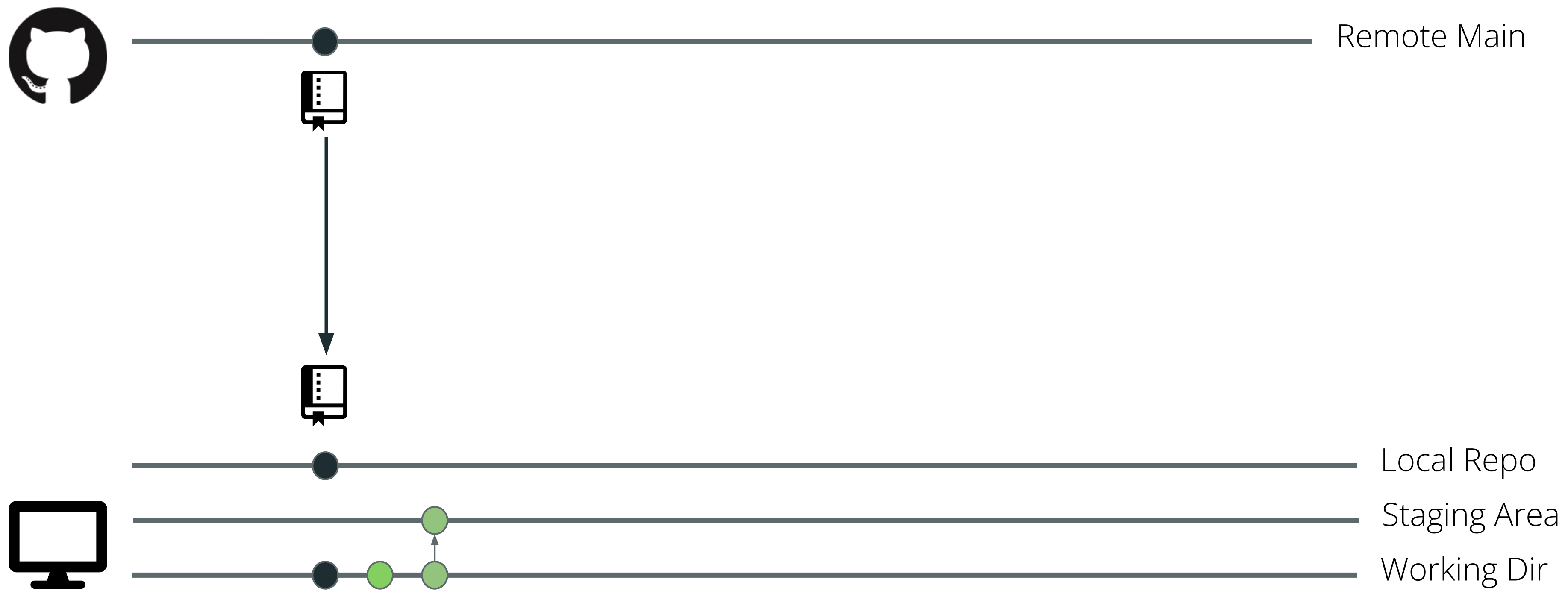
> git clone REPO



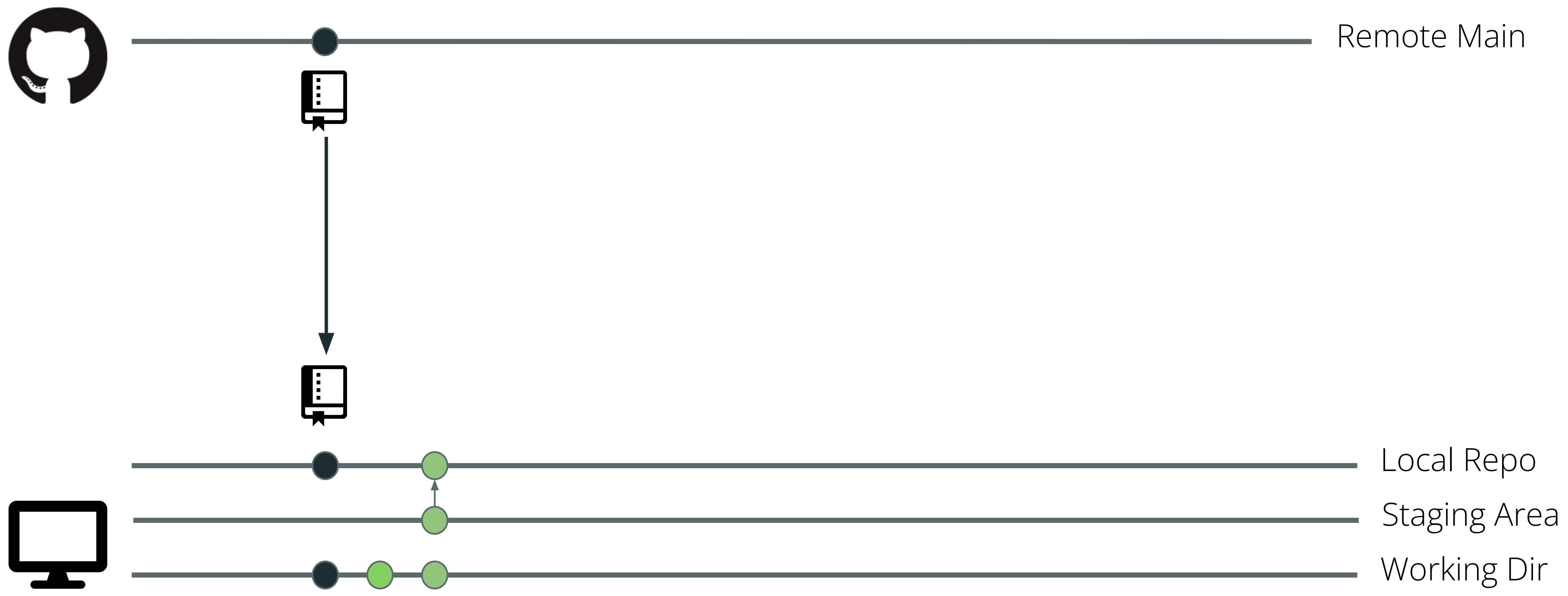
Edit file in working directory



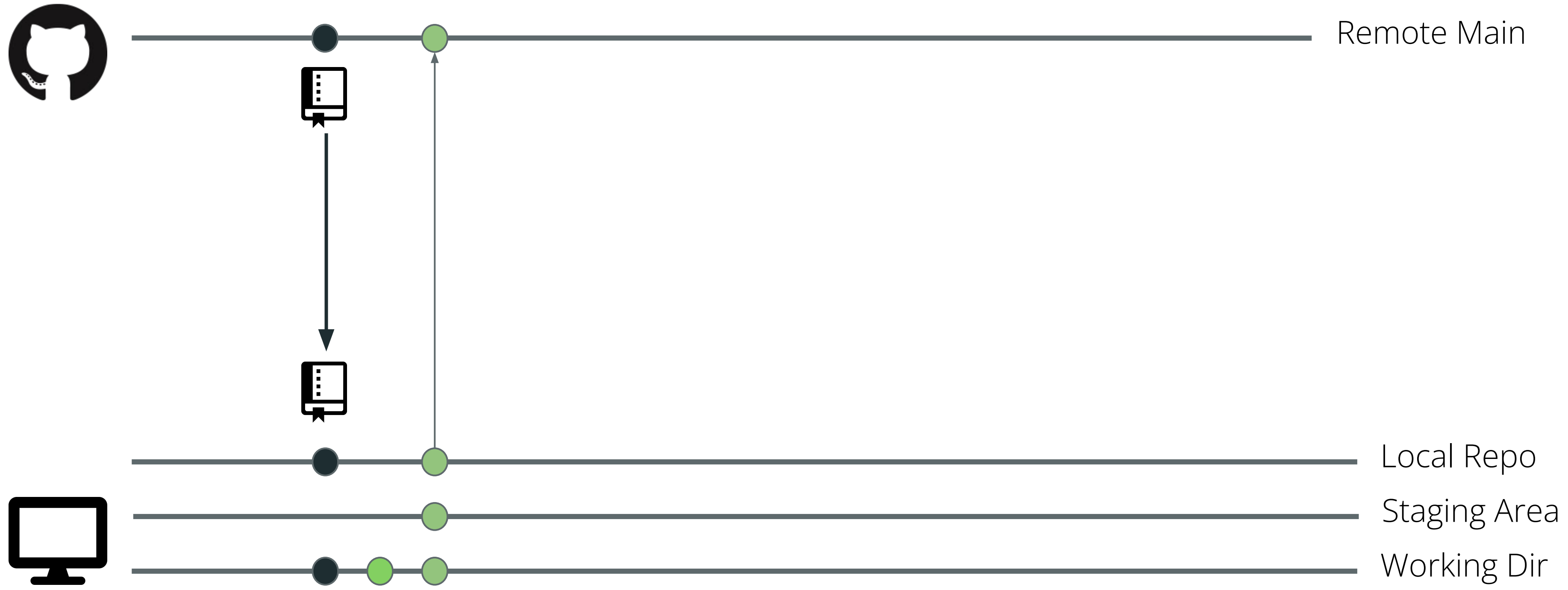
> git add File



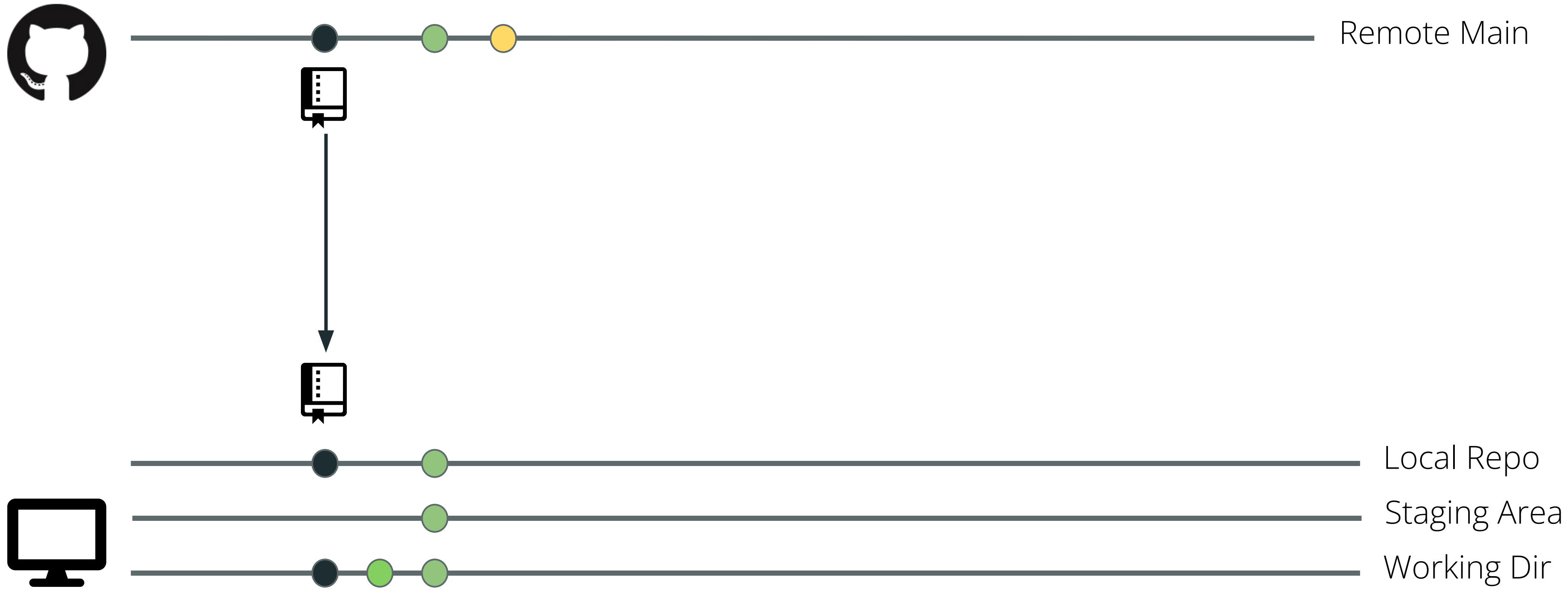
> git commit -m “added green edit”



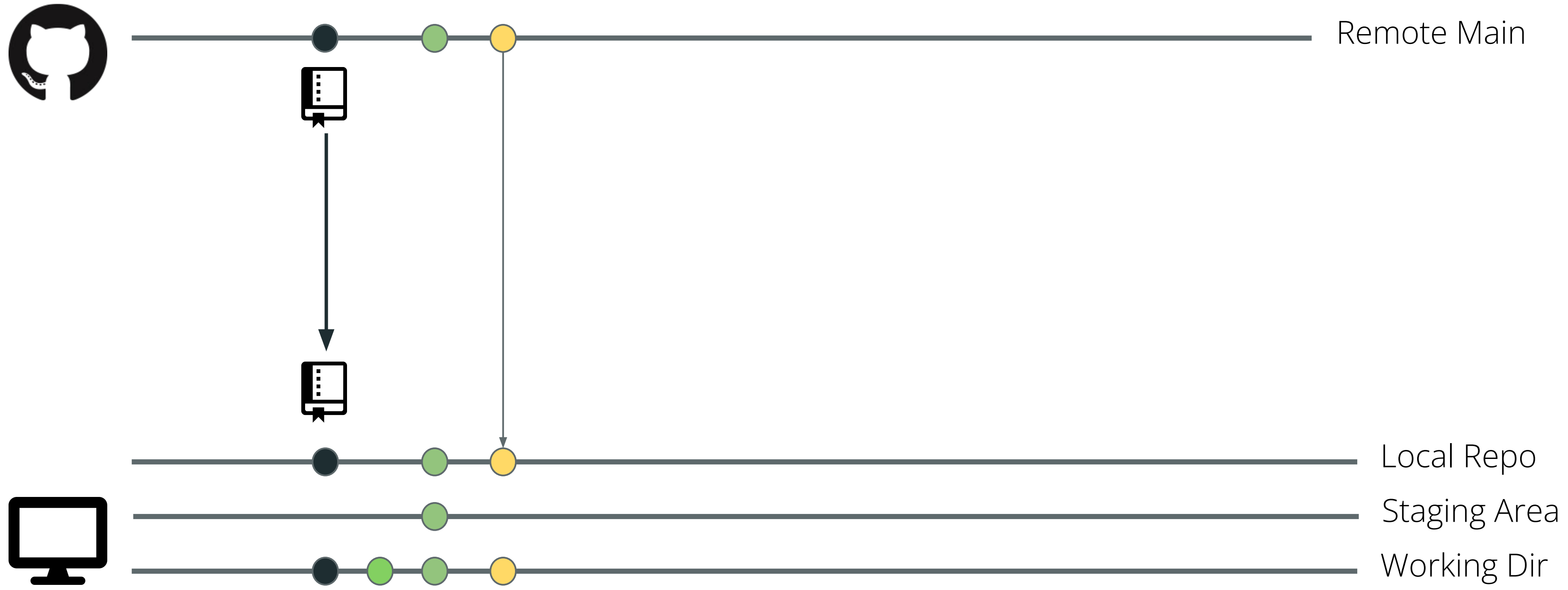
> git push origin main



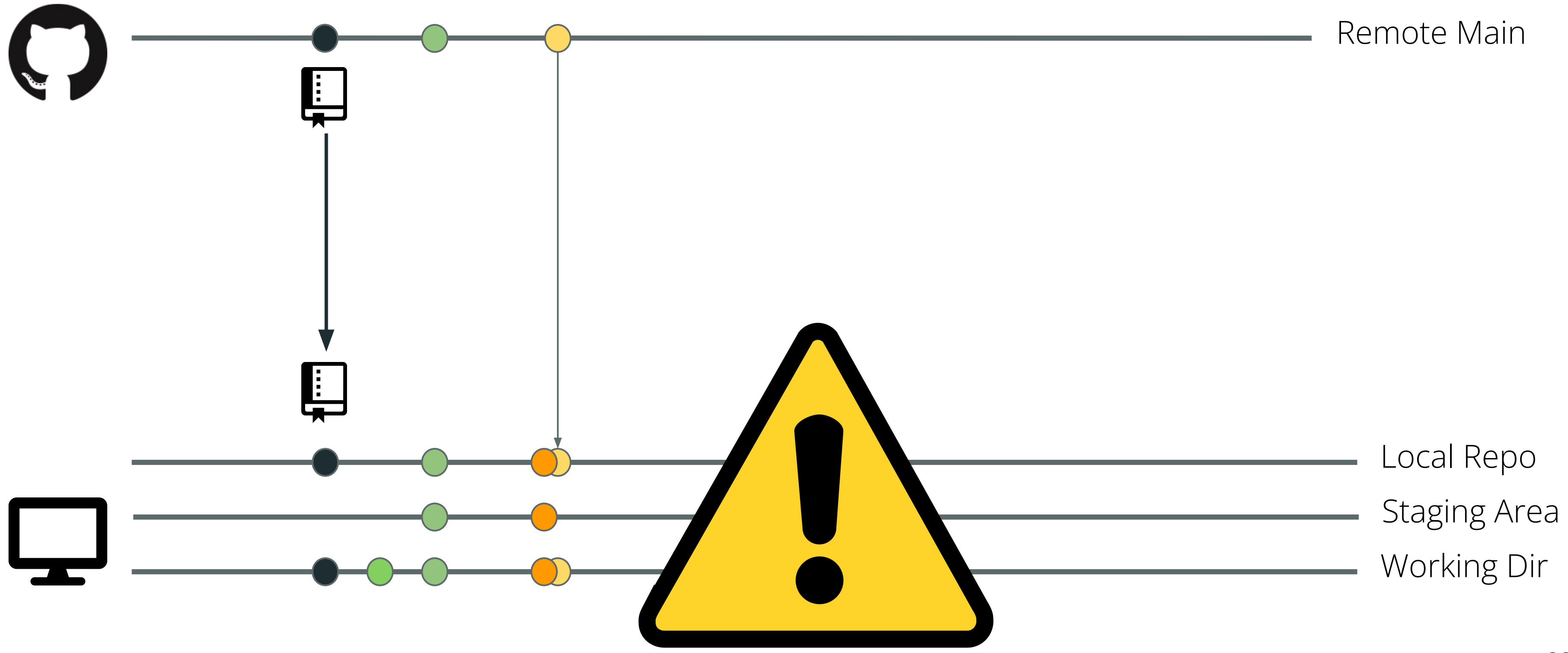
> git fetch (check for changes on remote)



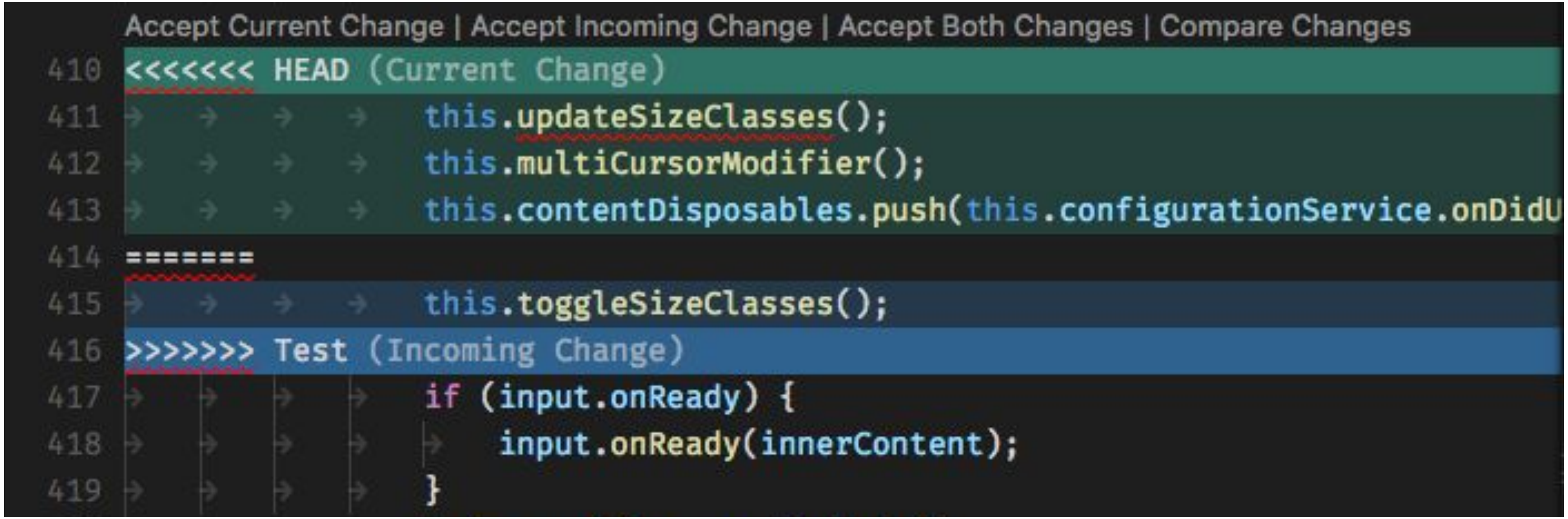
> git pull



> git pull can lead to Merge Conflicts

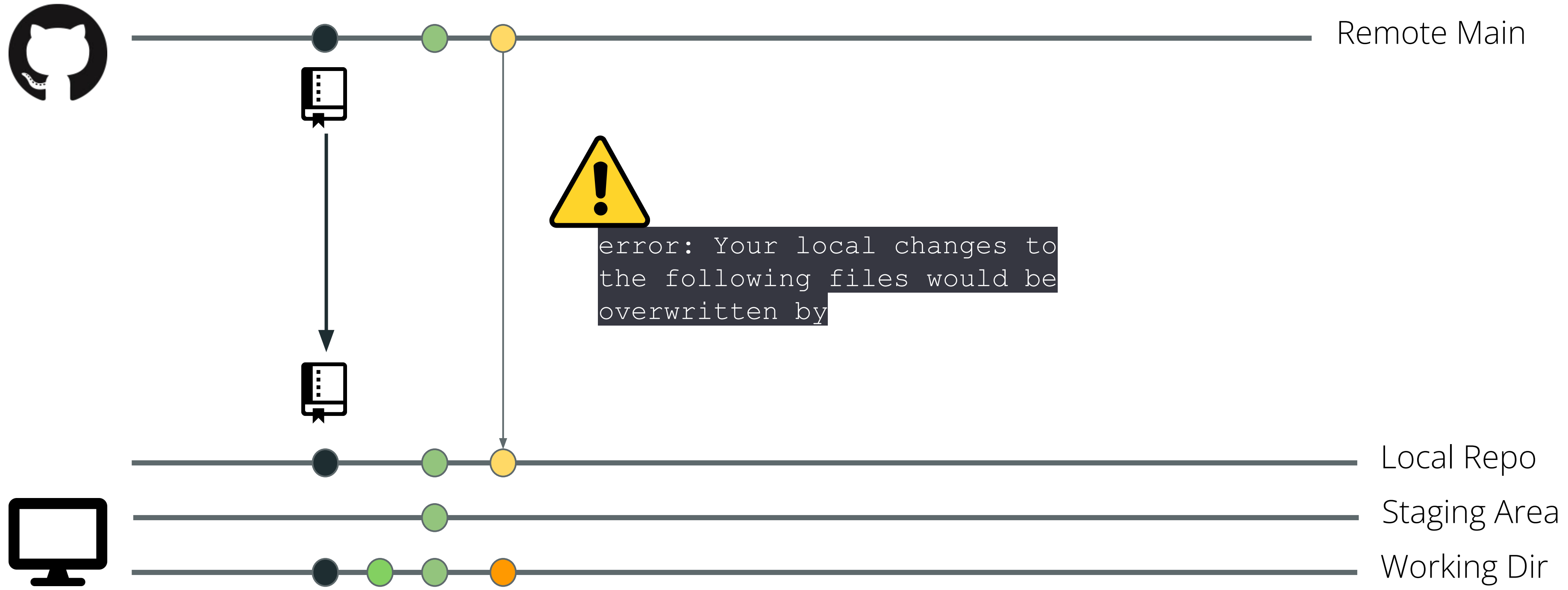


Merge Conflict (in VS Code)


Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes

```
<<<<<<< HEAD (Current Change)
				this.updateSizeClasses();
				this.multiCursorModifier();
				this.contentDisposables.push(this.configurationService.onDidU
=======
				this.toggleSizeClasses();
>>>>>>> Test (Incoming Change)
				if (input.onReady) {
					input.onReady(innerContent);
				}
```

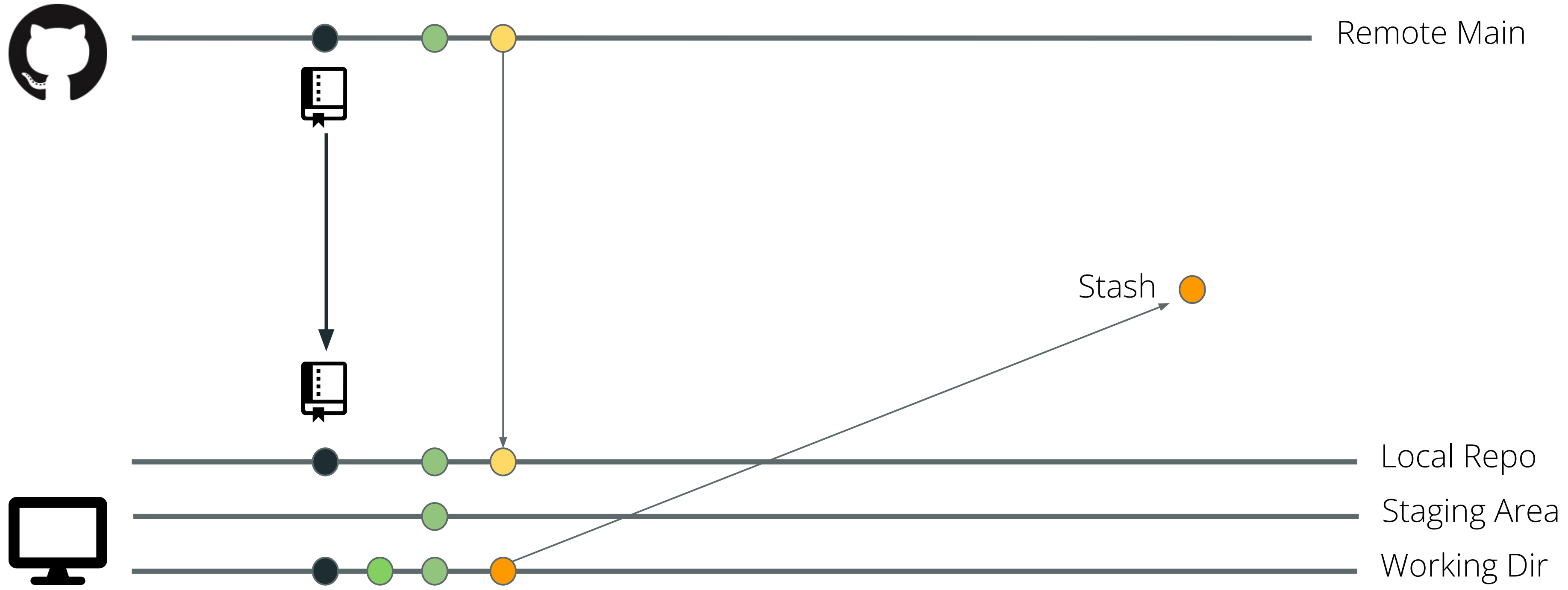
> git pull



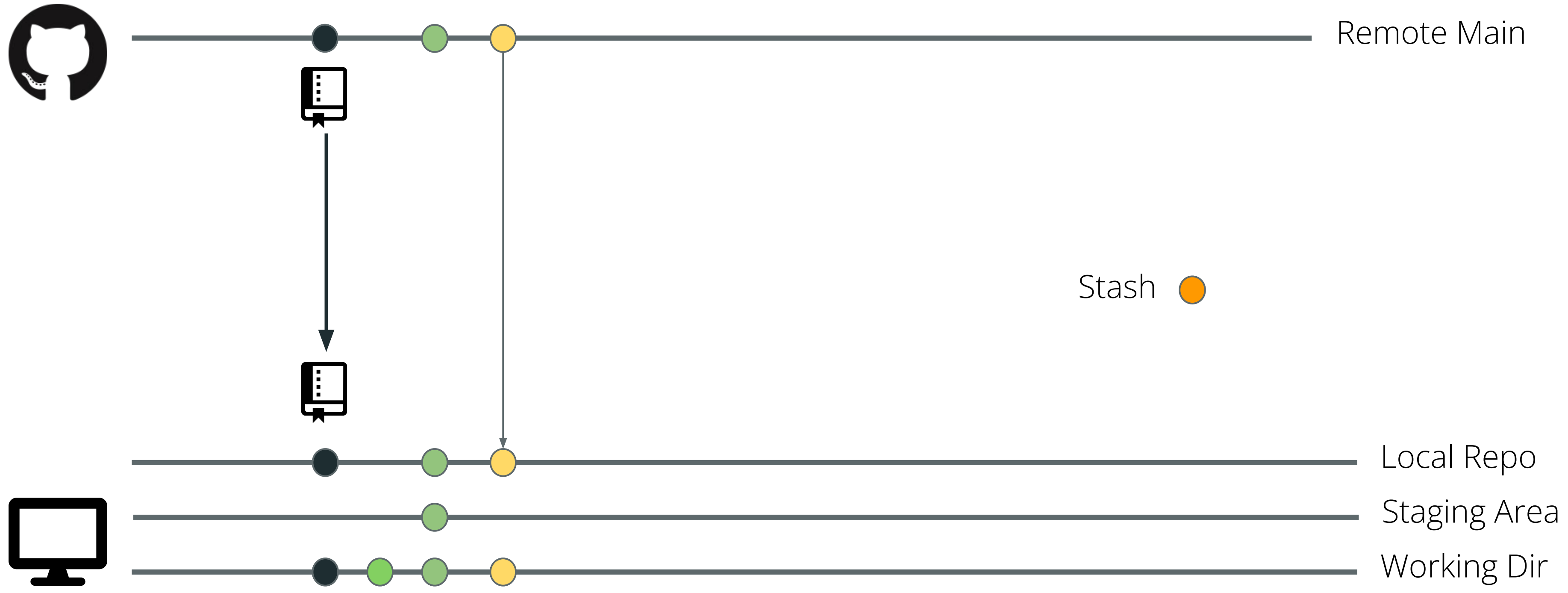
```
error: Your local changes to
the following files would be
overwritten by
```

Keep Changes

> git stash

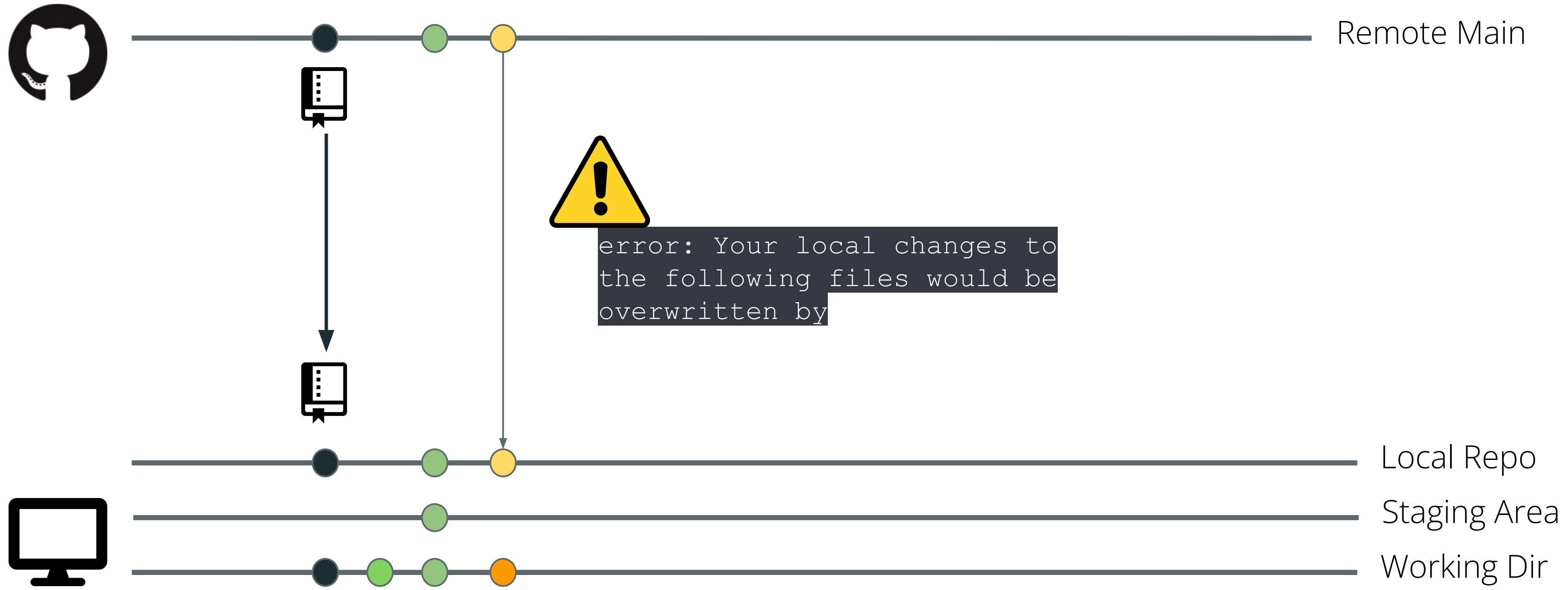


> git pull



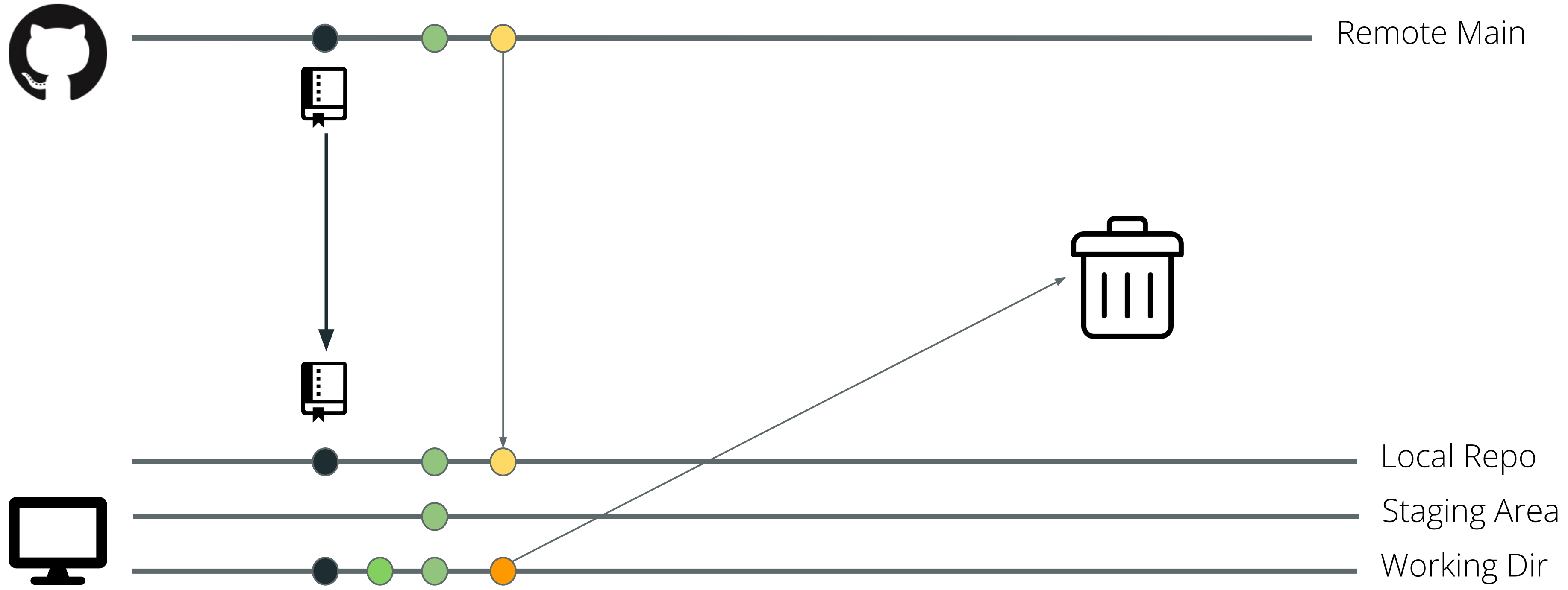
Discard Changes

> git pull

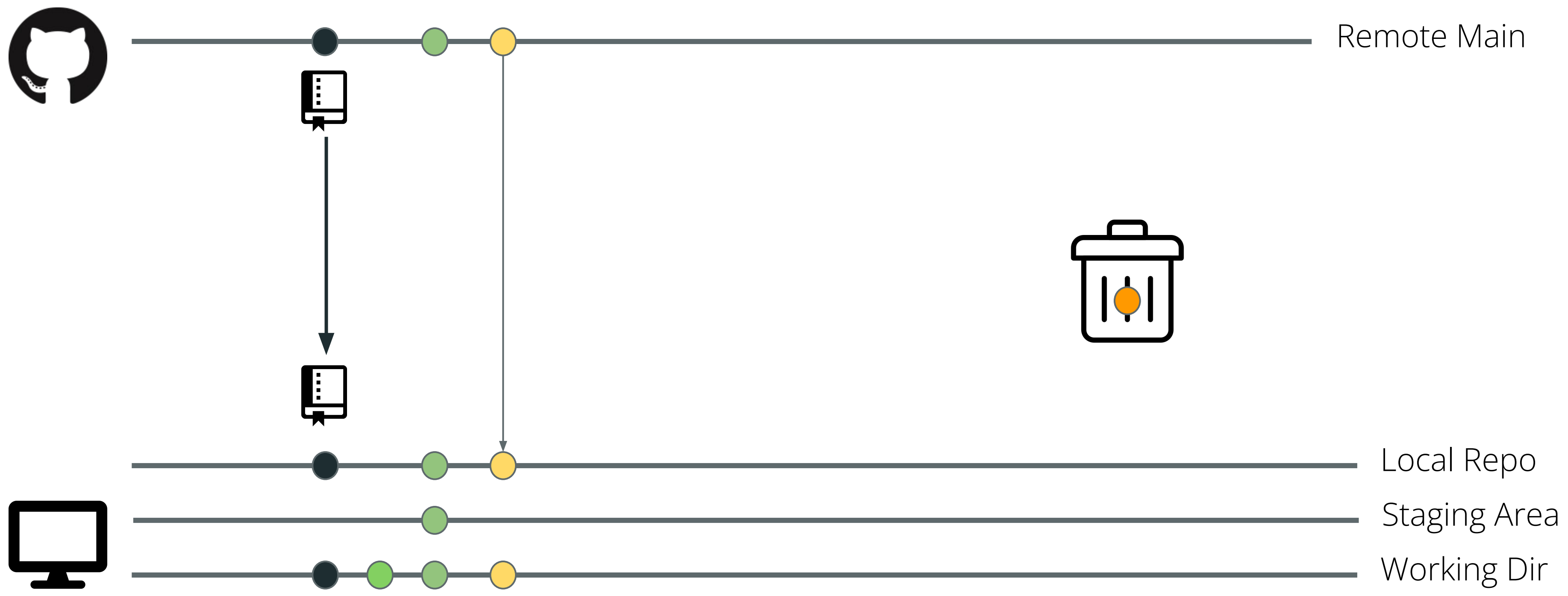


```
error: Your local changes to
the following files would be
overwritten by
```

> git reset –hard



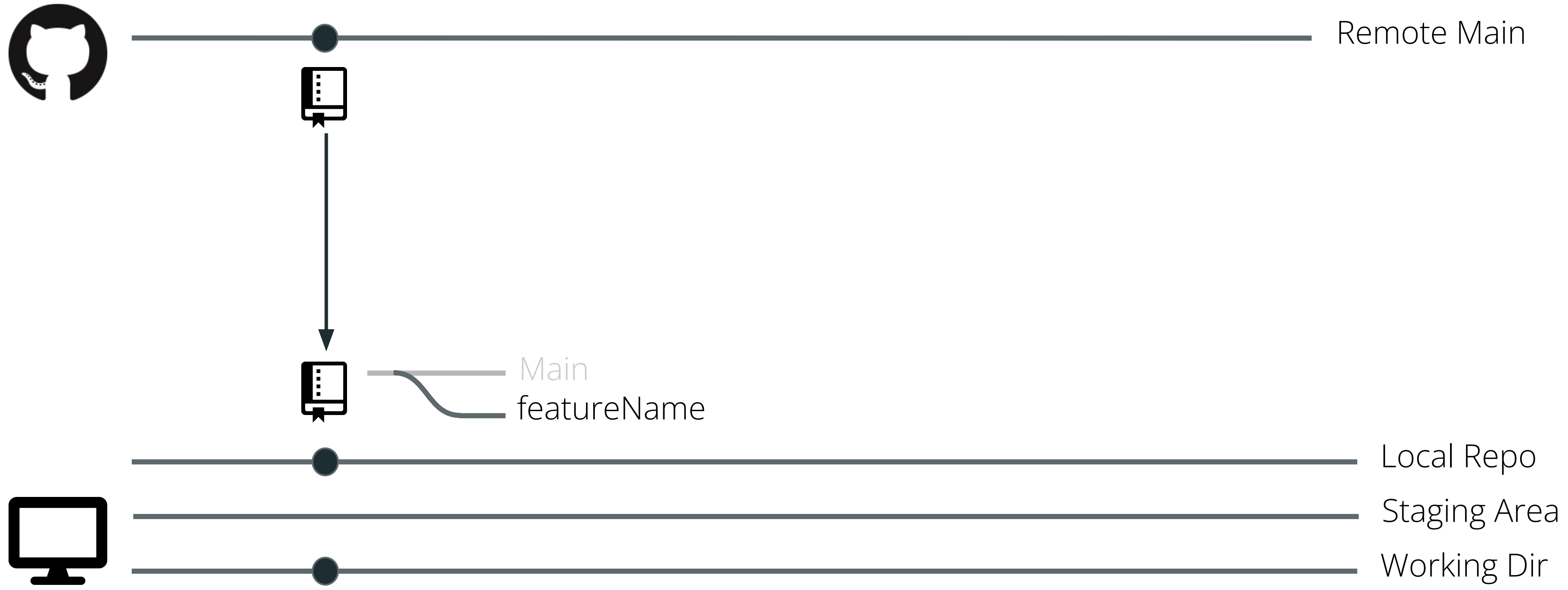
> git pull



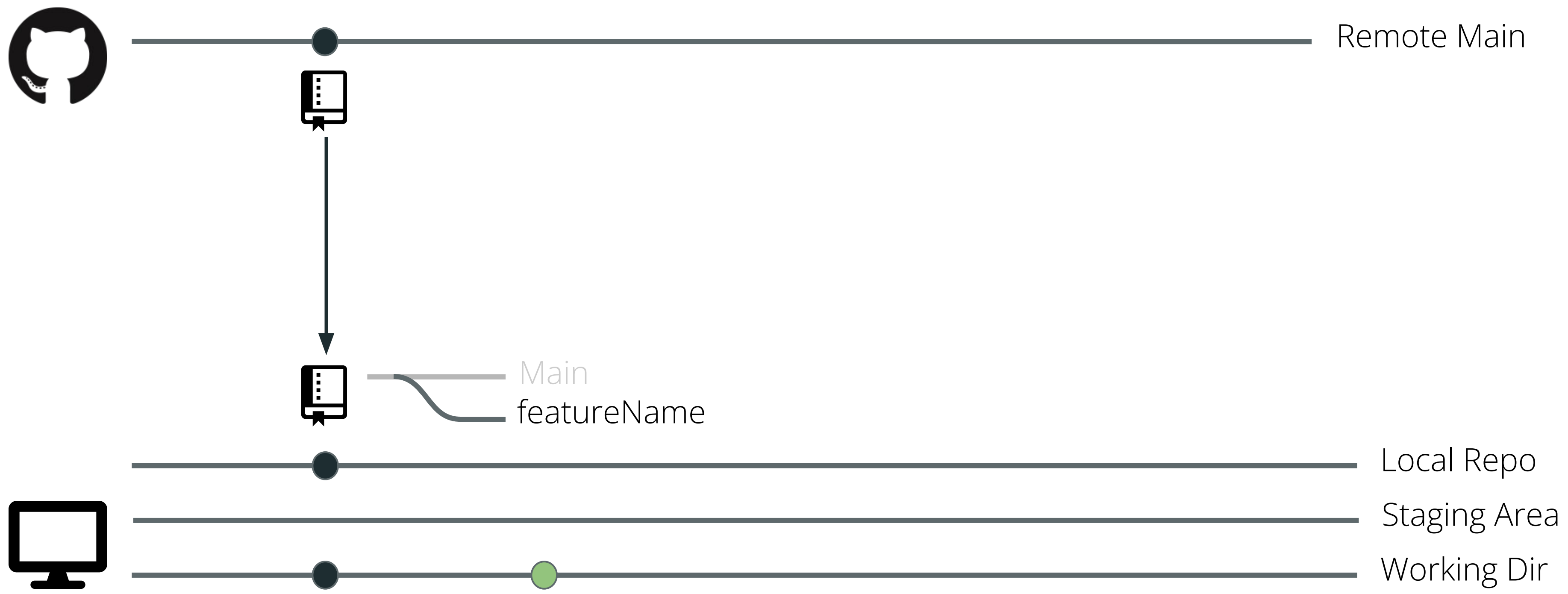
What if you want to work on multiple things at once?

Branches

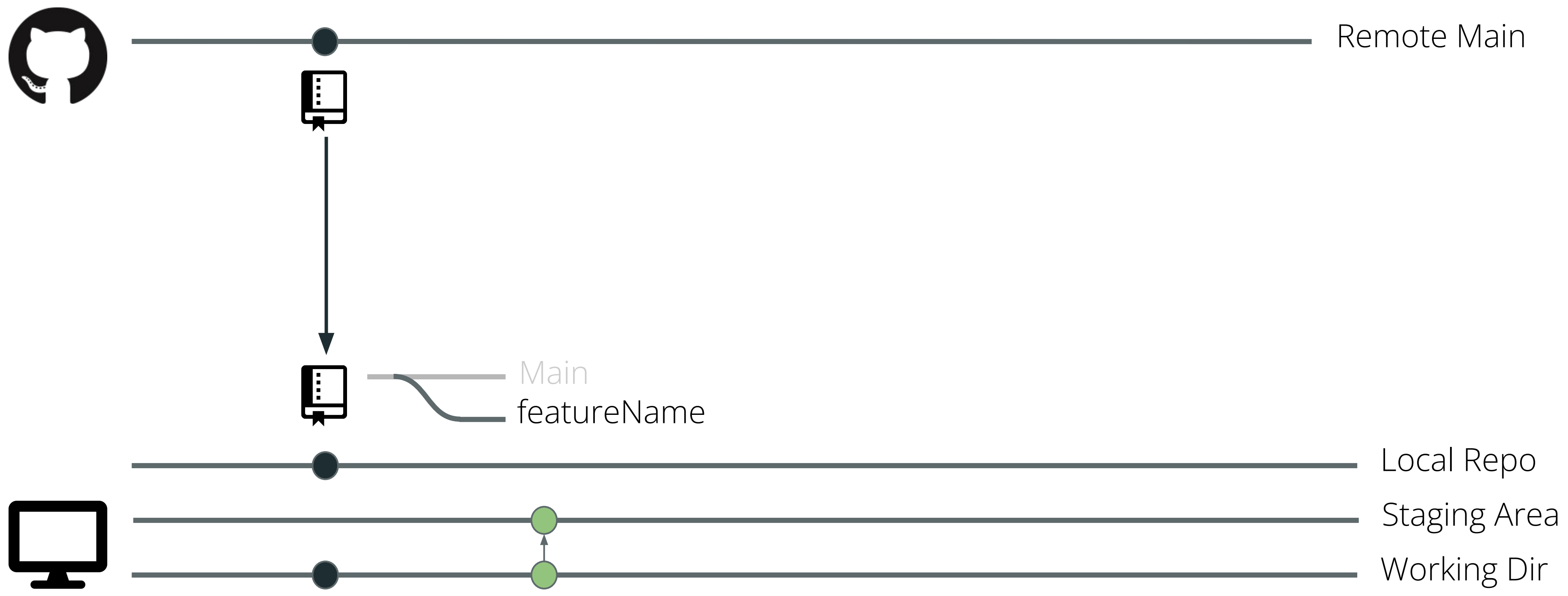
> git checkout -b featureName



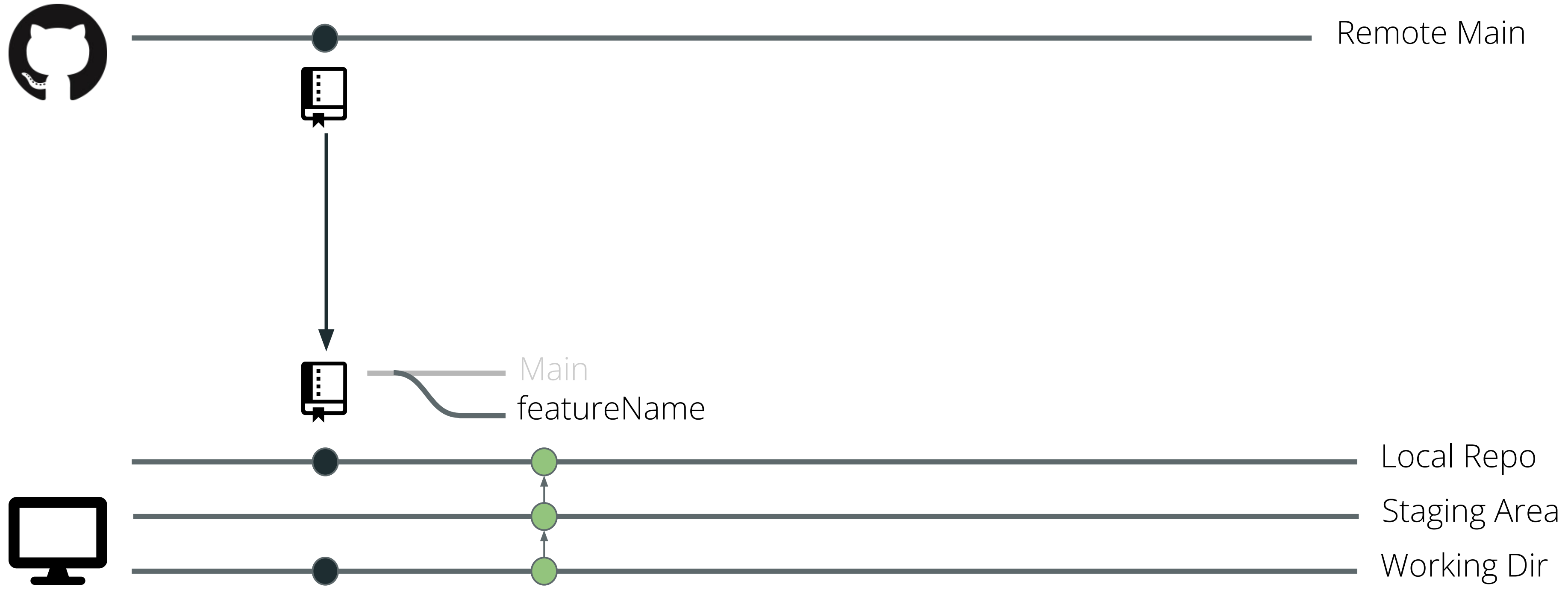
edit file(s) in working directory



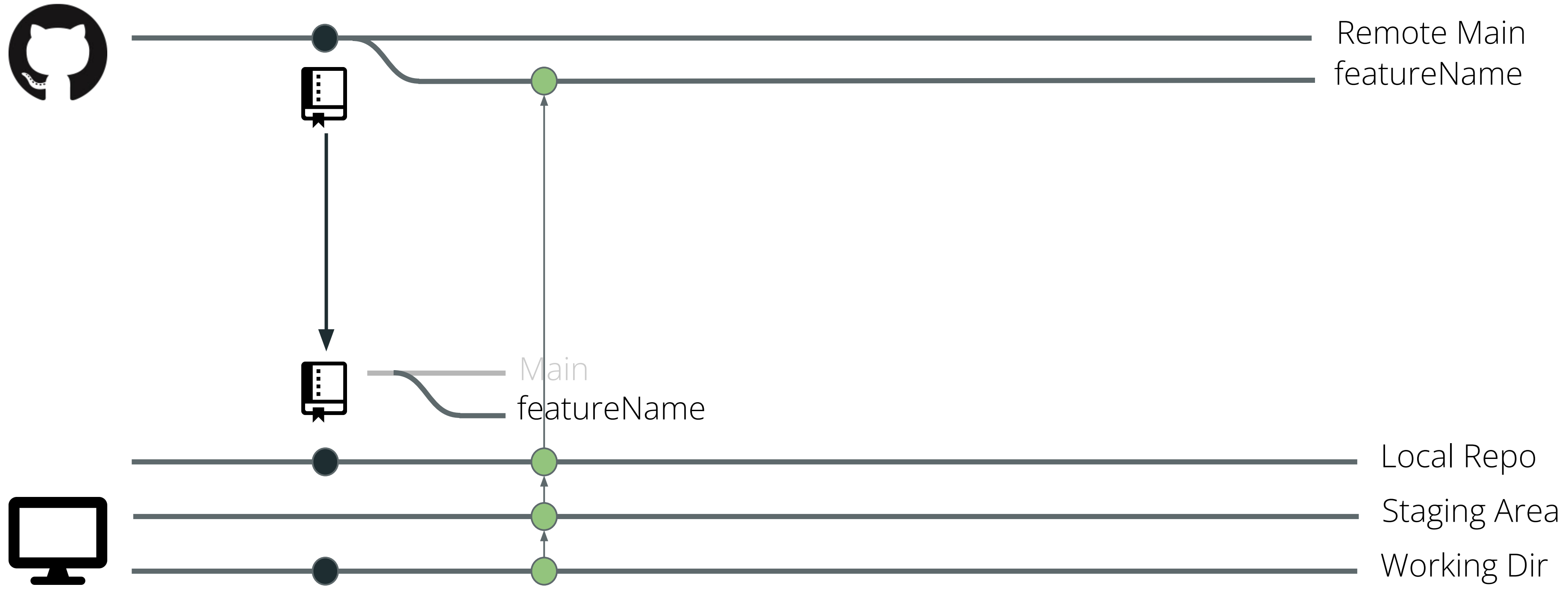
> git add File


Remote Main
Main
featureName
Local Repo
Staging Area
Working Dir

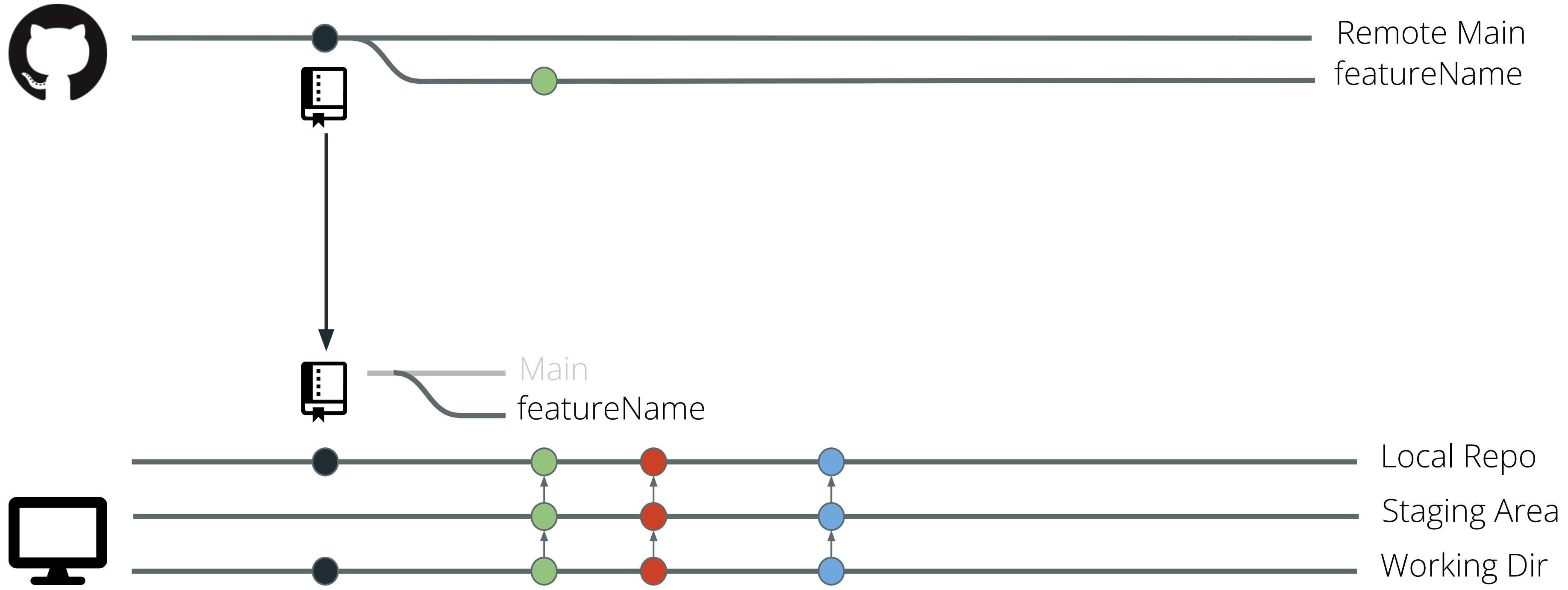
> git commit -m “added green edit”



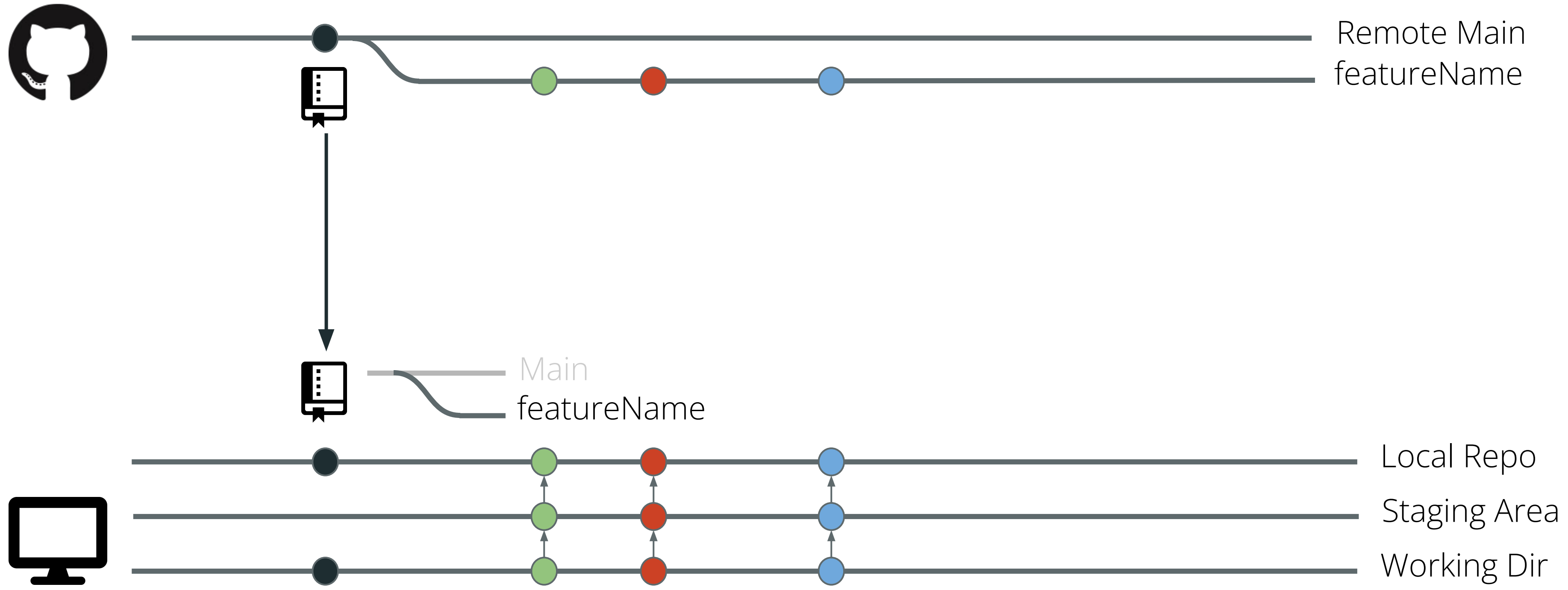
> git push origin featureName



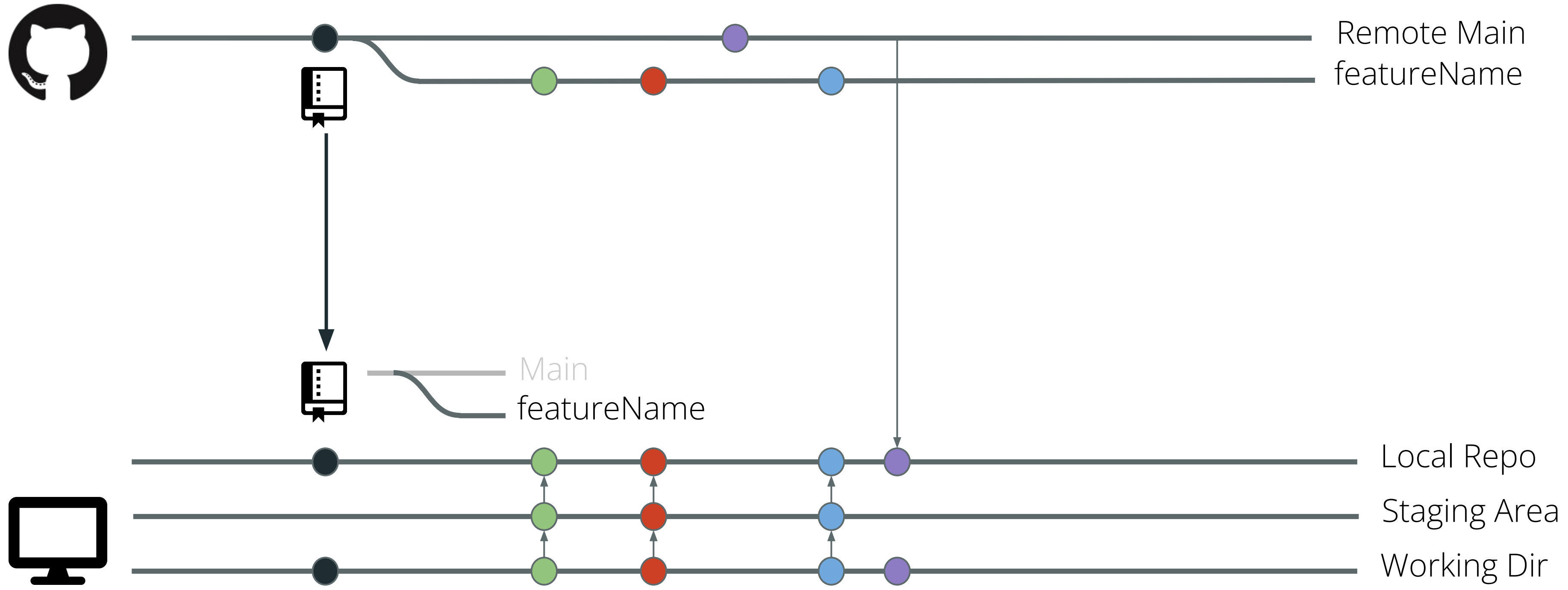
Continue work



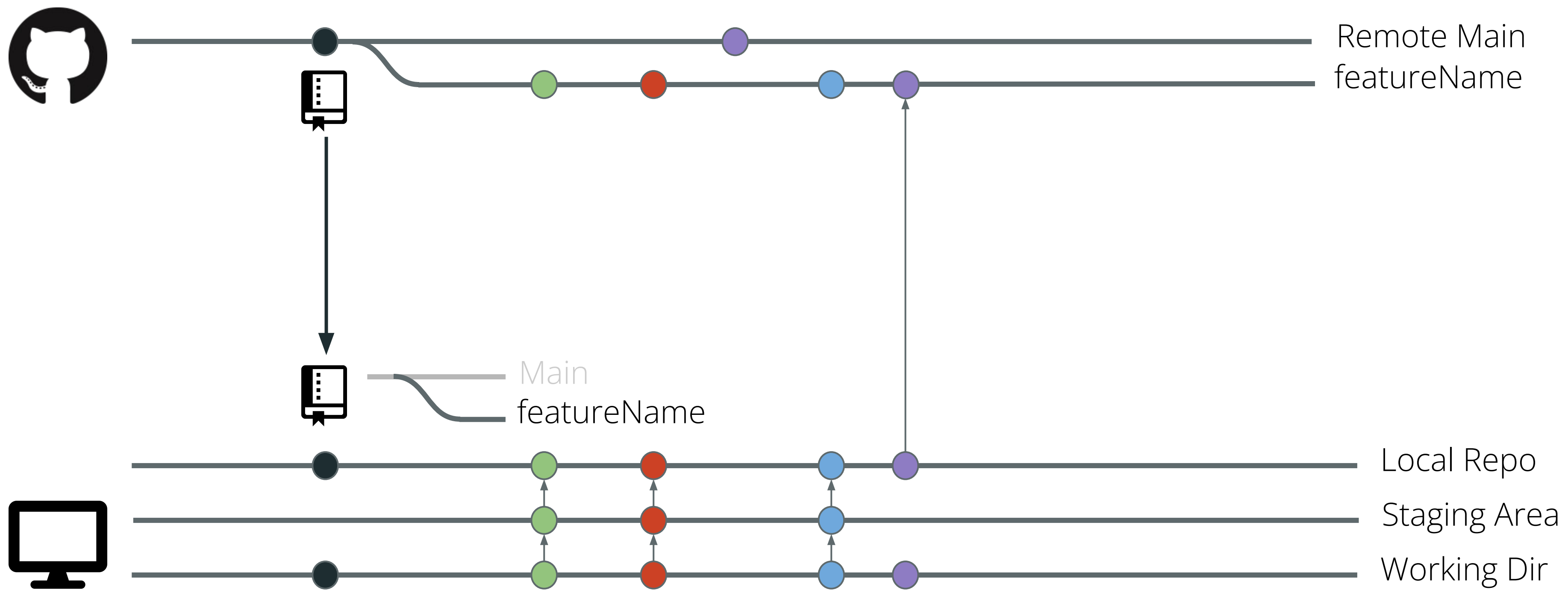
> git push remote featureName



> git pull origin main //Resolve any conflicts



> git push featureName



Example