CLASS 1 OF 3 · BASICS

Your Developer Toolkit

Command line, VS Code, Git and GitHub: the tools every developer opens every single working day

```
$ start class-1 _
```

Today's plan: 60 minutes

0–10 min	10–22 min	22–30 min	30–48 min	48–58 min	58–60 min
Install & verify tools	Command line	VS Code	Git & GitHub	Hands-on lab	Assignment

By the end of today you will be able to:

- Open a terminal and move around your folders with commands
- Create a project and open it in VS Code
- Save versions of your work with Git
- Publish your code to GitHub and share the link



Bring every class

A laptop with the tools from today installed. We build on them in Classes 2 and 3.

A developer's day at work

1. **Pick a task**
   A ticket in Jira, Trello or GitHub Issues
2. **Pull code**
   git pull to get the team's latest work
3. **Branch**
   A separate copy for your task
4. **Code & test**
   VS Code and the terminal
5. **Commit & push**
   Save and upload to GitHub
6. **Pull Request**
   Ask a teammate to review
7. **Merge & deploy**
   Your code goes live



ON THE JOB

Junior job posts almost always list Git, GitHub and the command line. Employers expect you to use them from day one, so we start here before writing any real code.

01

Install your tools

Set up the machine once, then use it every day

Install checklist

Tool	Why you need it	Download from	Check it works
VS Code	The code editor you write in	code.visualstudio.com/download	`code --version`
Git	Saves versions of your code	git-scm.com/downloads	`git --version`
Node.js (LTS)	Runs JavaScript outside the browser	nodejs.org/en/download	`node -v`
Python 3	Second language: scripts, back end	python.org/downloads	`python --version`
GitHub account	Stores and shares your code online	github.com/signup	`Log in on the site`

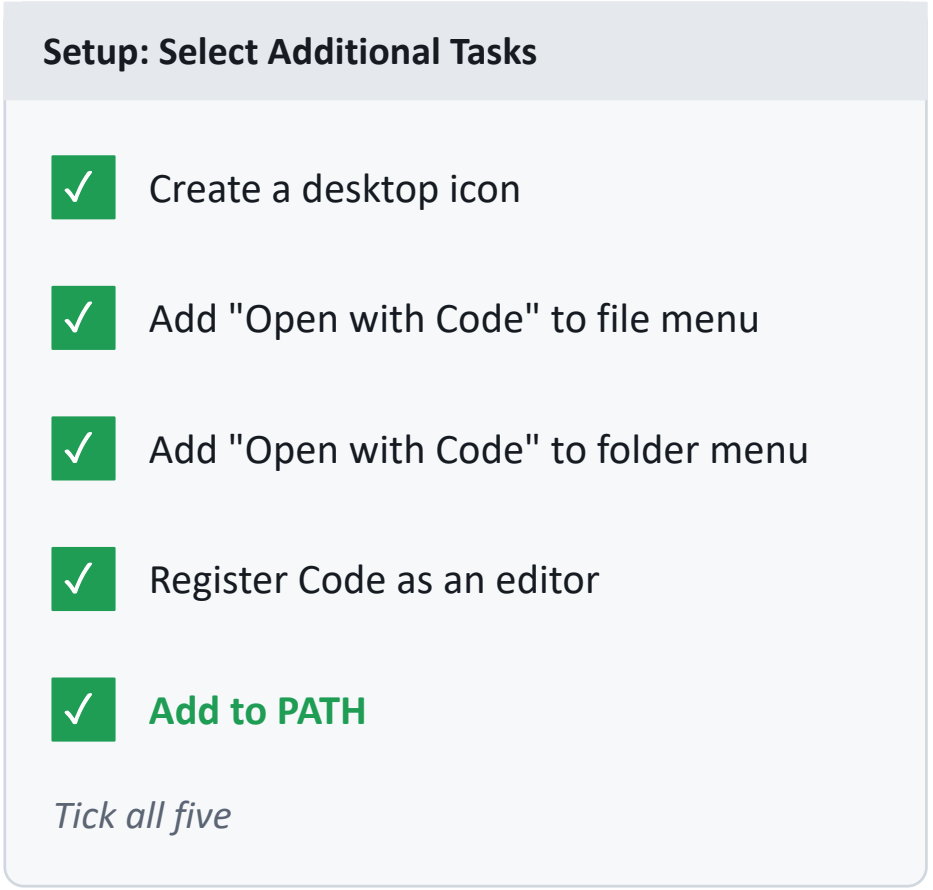


TIP

Always choose the LTS or stable version. LTS means Long Term Support: the version companies run in production.

Install VS Code step by step

1. **Open the download page**
   code.visualstudio.com/download, then click the big Windows button
2. **Run the installer**
   Open VSCodeUserSetup-x64.exe from Downloads and accept the agreement
3. **Tick every box on Select Additional Tasks**
   Especially Add to PATH (see the picture on the right)
4. **Install, then Finish**
   VS Code opens. Pin it to your taskbar
5. **Mac users**
   Download the .zip, drag VS Code to Applications. Then Cmd+Shift+P and run: Shell Command: Install 'code' command in PATH



Install Git step by step

1. **Download**
   git-scm.com/downloads, then Windows, then 64-bit Git for Windows Setup
2. **Run the installer and keep defaults**
   Click Next on most screens, except the three below
3. **Default editor screen**
   Choose: Use Visual Studio Code as Git's default editor
4. **Initial branch name screen**
   Choose: Override the default branch name, and type main
5. **PATH screen**
   Keep: Git from the command line and also from 3rd-party software
6. **Finish**
   You now also have Git Bash, a Linux-style terminal



On a Mac

Open Terminal and type:

```
git --version
```

If Git is missing, macOS offers to install the Command Line Tools. Click Install.

Or with Homebrew: `brew install git`

Install Node.js and Python

Node.js (runs JavaScript)

1. Go to nodejs.org/en/download
2. Pick the version marked LTS, then the Windows Installer (.msi)
3. Click Next and keep all defaults. npm is installed with it
4. Close and reopen your terminal

Python 3

1. Go to python.org/downloads, click Download Python 3.x
2. On the FIRST screen tick **Add python.exe to PATH**
3. Click Install Now
4. If offered, click Disable path length limit

WATCH OUT

Forgot to tick Add to PATH? Run the Python installer again, choose Modify, click Next, tick Add Python to environment variables, then Install.

Verify that everything works

Command Prompt

```
# Open a NEW terminal after installing
C:\Users\student> code --version
1.105.1
C:\Users\student> git --version
git version 2.51.0.windows.1
C:\Users\student> node -v
v22.20.0
C:\Users\student> npm -v
10.9.3
C:\Users\student> python --version
Python 3.13.7
```

Your version numbers may be newer. That is fine.



Seeing this error?

'node' is not recognized as an internal or external command

It means the program is missing or not on PATH.

1. Close every terminal and open a new one
2. Still failing? Restart the computer
3. Still failing? Reinstall and tick Add to PATH

02

The command line

Talk to your computer with text instead of clicks

Why developers live in the terminal

Run code

```
node app.js
python app.py
```

Start apps, scripts and dev servers

Install packages

```
npm install express
pip install requests
```

Add ready-made libraries to a project

Use Git

```
git commit
git push
```

Save and share versions of your code

Manage servers

```
ssh user@server
```

Log in to a remote server and fix it. Servers have no mouse

ON THE JOB

Production servers have no desktop. Deploying an app, reading error logs and restarting a crashed service all happen in a terminal.

How to open a terminal

Command Prompt

Press Win + R, type cmd, press Enter. Or open Start and type cmd

PowerShell

Right-click the Start button and choose Terminal. Or open Start and type PowerShell

Git Bash

Open Start and type Git Bash. Uses the same commands as Mac and Linux

Inside VS Code

Menu Terminal, then New Terminal. Shortcut: Ctrl + ` (the key under Esc)

Mac

Press Cmd + Space, type Terminal, press Enter

Open in a folder

In File Explorer click the address bar, type cmd, press Enter. It opens right there

Reading the prompt: where am I?

Four kinds of prompt

```
# Windows Command Prompt
C:\Users\student>
# PowerShell
PS C:\Users\student>
# Git Bash (~ is your home folder)
student@laptop MINGW64 ~ $
# Mac Terminal
student@MacBook ~ %
```

TIP

You type after the > or $ and press Enter to run. Nothing happens until you press Enter.

Term	Meaning
`cwd`	Current working directory: the folder you are in
`C:\Users\student`	Absolute path: the full address from the drive
`Desktop\app`	Relative path: starts from where you are
`.`	This folder
`..`	The parent folder (one level up)
`~`	Your home folder (Bash, PowerShell, Mac)

Essential commands

What you want	Command Prompt (cmd)	PowerShell · Git Bash · Mac
Show where I am	`cd`	`pwd`
List files and folders	`dir`	`ls`
Go into a folder	`cd Desktop`	`cd Desktop`
Go up one level	`cd ..`	`cd ..`
Make a folder	`mkdir my-app`	`mkdir my-app`
Create an empty file	`type nul > app.js`	`touch app.js` (PowerShell: `ni app.js`)
Show a file's contents	`type app.js`	`cat app.js`
Copy a file	`copy a.txt b.txt`	`cp a.txt b.txt`
Move or rename a file	`move a.txt docs`	`mv a.txt docs`
Delete a file	`del app.js`	`rm app.js`
Delete a folder	`rmdir /s my-app`	`rm -r my-app`
Clear the screen	`cls`	`clear`
Open folder in VS Code	`code .`	`code .`

Practical: build a project from the terminal

Command Prompt (cmd)

```
# 1. Go to your Desktop
C:\Users\student> cd Desktop
# 2. Make a folder for the whole course
C:\Users\student\Desktop> mkdir dev-course
C:\Users\student\Desktop> cd dev-course
# 3. Make today's project folder and enter it
...\dev-course> mkdir class1-hello
...\dev-course> cd class1-hello
# 4. Write one line of code into a new file
...\class1-hello> echo console.log("Hello!") > hello.js
# 5. Run it with Node.js
...\class1-hello> node hello.js
Hello!
# 6. Open this folder in VS Code
...\class1-hello> code .
```

What just happened

- **cd** moved you into folders
- **mkdir** created new folders
- **echo … >** wrote text into a new file
- **node** ran your JavaScript file
- **code .** opened the current folder (.) in VS Code
- ...\ is shorthand here for the longer path

Terminal tips that save hours

Tab

Auto-completes names. Type cd Des then press Tab

Up / Down arrows

Scroll through commands you already ran. No retyping

Ctrl + C

Stops a running program, such as a server

cls or clear

Wipes the screen when it gets messy

Spaces in names

Use quotes: cd "My Projects". Better: never use spaces

Paste

Ctrl + V in Windows Terminal and VS Code. Right-click in old cmd

ON THE JOB

Name folders and files in lowercase-with-dashes: my-portfolio, not My Portfolio Final (2). Teams and servers expect it.

03

VS Code

The editor you will write code in all day

A tour of VS Code



1 Activity bar icon	Shortcut
Explorer (files)	Ctrl + Shift + E
Search all files	Ctrl + Shift + F
Source Control (Git)	Ctrl + Shift + G
Run and Debug	Ctrl + Shift + D
Extensions	Ctrl + Shift + X

TIP

Status bar shows the Git branch, errors and the file's language.

Shortcuts you will use every day

Shortcut	What it does
Ctrl + S	Save the file
Ctrl + `	Show or hide the terminal
Ctrl + P	Open any file by name
Ctrl + Shift + P	Command palette: run any command
Ctrl + /	Comment or uncomment a line
Ctrl + F	Find in this file

Shortcut	What it does
Alt + Up / Down	Move a line up or down
Shift + Alt + Down	Duplicate a line
Ctrl + D	Select the next match (multi-edit)
Shift + Alt + F	Format the whole document
Ctrl + B	Show or hide the sidebar
Ctrl + Z / Ctrl + Y	Undo / redo



Mac: use Cmd instead of Ctrl, and Option instead of Alt

Extensions and settings to set up now

Extension	What it does for you
Prettier	Formats code neatly every time you save
ESLint	Underlines JavaScript mistakes
Python (Microsoft)	Runs, checks and debugs Python
Live Server	Opens HTML with auto-refresh
GitLens	Shows who changed each line and when
Thunder Client	Tests APIs (used in Class 3)
SQLite Viewer	Opens database files (Class 3)

settings.json

```
{
  "editor.formatOnSave": true,
  "files.autoSave": "afterDelay",
  "editor.defaultFormatter":
      "esbenp.prettier-vscode",
  "editor.fontSize": 16
}
```

How to install

Ctrl + Shift + X, search the name, click Install. Check the publisher so you get the official one.

04

Git & GitHub

Save every version of your work and collaborate with a team

Git is not GitHub



Git

- A program on your computer
- Records every change as a commit
- Lets you go back to any old version
- Works offline



GitHub

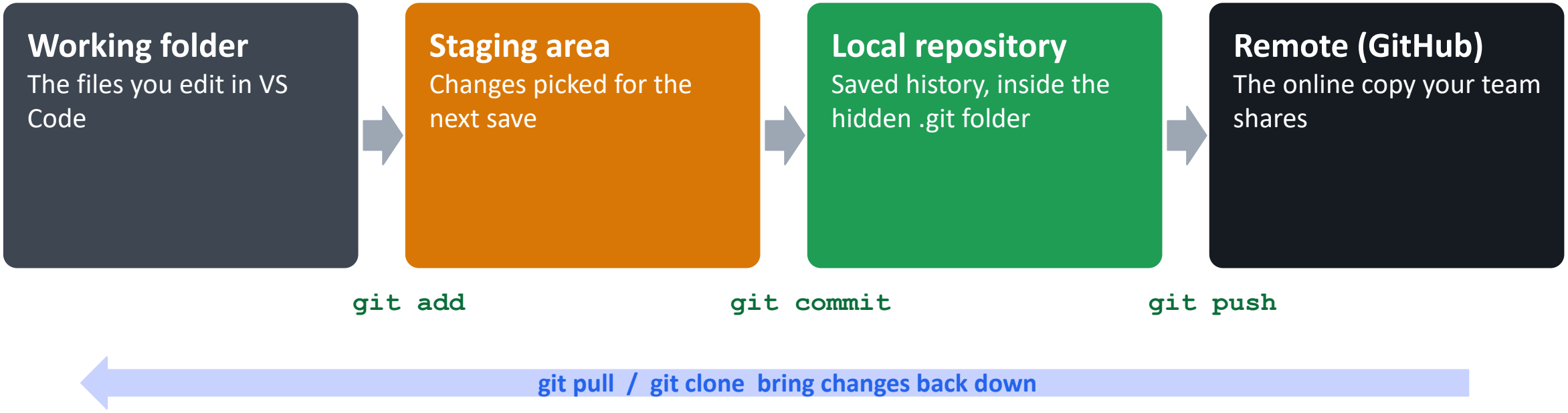
- A website that hosts Git repositories
- Online backup of your code
- Teamwork: Pull Requests and reviews
- Your public portfolio. Similar: GitLab, Bitbucket



ON THE JOB

Your GitHub profile works like a CV. Recruiters open it to see real projects and how often you commit.

How Git works: four places your code lives



Word	Meaning
Repository (repo)	A project folder tracked by Git
Commit	One saved snapshot, with a message and an ID like 3f2a1c9
Branch	A separate line of work, so you do not break main

One-time setup on each computer

Git Bash / any terminal

```
# Tell Git who you are (shown on every commit)
$ git config --global user.name "Grace Okello"
$ git config --global user.email "grace@example.com"

# Call the first branch "main" (GitHub's default)
$ git config --global init.defaultBranch main

# Use VS Code for long commit messages
$ git config --global core.editor "code --wait"

# Check what you set
$ git config --list
```



WATCH OUT

Use the same email as your GitHub account, or your commits will not show on your profile.



What --global means

Apply this setting to every project on this computer. You run these once, not per project.

Your first repository, step by step

#	Command	What it does
1	`cd dev-course/class1-hello`	Go into the project folder (from Desktop)
2	`git init`	Turns the folder into a repository (creates a hidden .git folder)
3	`git status`	Shows what changed. Red = changed but not staged
4	`git add hello.js`	Stages one file. Use git add . for all files. Now green
5	`git commit -m "Add hello script"`	Saves a snapshot of the staged files with a message
6	`git log --oneline`	Shows the history: one line per commit



The rhythm you will repeat thousands of times: edit → git add → git commit

What it looks like in the terminal

Git Bash: class1-hello

```
$ git init
Initialized empty Git repository in .../class1-hello/.git/
$ git status
On branch main
Untracked files:
        hello.js
$ git add .
$ git commit -m "Add hello script"
[main (root-commit) 3f2a1c9] Add hello script
 1 file changed, 1 insertion(+)
$ git log --oneline
3f2a1c9 (HEAD -> main) Add hello script
```

How to read it

- **Untracked**: Git sees a new file but does not track it yet
- **root-commit**: the very first commit
- **3f2a1c9**: the commit's short ID
- **HEAD -> main**: you are on the main branch, at this commit

Publish to GitHub

1. **Create the repo on GitHub**
   Click + (top right), then New repository. Name it class1-hello

2. **Leave it empty**
   Do not tick Add a README. Click Create repository

3. **Copy the URL GitHub shows**
   https://github.com/your-name/class1-hello.git

4. **Run the 3 commands on the right**
   A browser opens the first time: sign in and approve

5. **Refresh the GitHub page**
   Your code and commits are online

Git Bash: class1-hello

```
$ git remote add origin https://github.com/
      your-name/class1-hello.git
$ git branch -M main
$ git push -u origin main
```

Part	Meaning
`remote add origin`	Link this folder to GitHub under the nickname origin
`branch -M main`	Make sure the branch is called main
`push -u origin main`	Upload. -u remembers it, so next time just git push

Git commands cheat sheet

Command	What it does	When you use it
`git clone <url>`	Downloads a repo from GitHub to your computer	First day on a project
`git pull`	Downloads your teammates' new commits	Start of every day
`git switch -c feature/login`	Creates a new branch and moves onto it	Start of a new task
`git status`	Shows changed, staged and new files	All the time
`git diff`	Shows the exact lines you changed	Before staging
`git add .`	Stages all changes	Before committing
`git commit -m "msg"`	Saves a snapshot with a message	After each small step
`git push`	Uploads your commits to GitHub	End of a task or day
`git log --oneline`	Shows the commit history	Checking what happened
`git branch`	Lists branches; * marks the current one	Before switching
`git switch main`	Moves to an existing branch	Changing tasks
`git restore app.js`	Throws away uncommitted changes to a file	Undoing a mistake

The team workflow companies use

A normal working day

```
# Morning: get the team's latest code
$ git switch main
$ git pull
# Start the task on its own branch
$ git switch -c feature/signup-form
# ...write code and test it...
$ git add .
$ git commit -m "feat: validate signup form"
$ git push -u origin feature/signup-form
# On GitHub: open a Pull Request
```

Good commit messages

Prefix	Use for
`feat:`	A new feature
`fix:`	A bug fix
`docs:`	README or comments
`refactor:`	Cleaner code, same behaviour
`chore:`	Setup, packages, config



Bad: "update", "stuff", "final2"

Good: "fix: show error when email is taken"

Pull Requests and code review

1. **Push branch**
   git push -u origin feature/x
2. **Open PR**
   Title, description, screenshots
3. **Review**
   Teammates comment on lines
4. **Fix & push**
   The PR updates by itself
5. **Approved**
   Checks and tests pass
6. **Merge**
   Branch joins main. Delete it

ON THE JOB

In a team you never push straight to main. Most companies block it: every change goes through a PR.

A good PR description says

What changed, why, how to test it, and a screenshot if the screen changed.

Merge conflicts: don't panic

fees.js: conflict

```
<<<<<<< HEAD
const fee = amount * 0.02;
=======
const fee = amount * 0.015;
>>>>>>> feature/new-fee
```

- **HEAD**: the version on your current branch
- **Below =======**: the incoming version from the other branch

1. **Open the file in VS Code**
   Conflicts are highlighted in colour
2. **Choose a version**
   Click Accept Current, Accept Incoming or Accept Both. Or edit by hand
3. **Remove all markers**
   No <<<<<<<, ======= or >>>>>>> left
4. **Stage and commit**
   git add fees.js, then git commit
5. **Prevent it**
   Pull often, keep branches short

.gitignore: what never goes on GitHub

.gitignore

```
# Packages: reinstall with npm install
node_modules/
venv/
# Secrets: passwords and API keys
.env
# System and editor files
.DS_Store
Thumbs.db
# Build output and logs
dist/
*.log
```

WATCH OUT

Pushed a password by mistake? Deleting the file is not enough: it stays in the history. Change the password or key immediately.

Ready-made templates

github.com/github/gitignore has a .gitignore for Node, Python, Laravel and more. GitHub can also add one when you create a repo.

Hands-on lab: 10 minutes

1. **In the terminal:** `cd Desktop/dev-course`
2. **Create and open the folder**
   mkdir class1-lab, then cd class1-lab, then code .
3. **Create about.md**
   Your name, why you are learning to code, three goals
4. **Save it to Git:** `git init, git add ., git commit`
5. **Create class1-lab on GitHub and push it**
6. **Add a line, commit and push again**
   Check the commit history on GitHub
7. **Post your repo link in the class group**

about.md

```
# About Me
My name is Grace.

## Why I code
I want to build apps.

## My 3 goals
1. Build a website
2. Get an internship
3. Publish an app
```



TIP

Ctrl + Shift + V previews Markdown in VS Code.

Assignment 1: due before Class 2

A. Terminal only

Without the mouse, create a folder portfolio containing index.html, README.md, a css folder with style.css and a js folder with app.js

B. Git

At least 5 commits with clear messages. Create branch feature/about, add an About section to index.html, then merge it into main

C. GitHub

Push to a public repo. README explains what the project is and how to open it

D. Bonus

Finish the first 4 levels of learngitbranching.js.org and screenshot the result

Target structure

```
portfolio/
├── index.html
├── README.md
├── css/
│   └── style.css
└── js/
    └── app.js
```

Submit

Post your GitHub repo link in the class group.

Documentation and practice

Resource	Use it for	Link
Pro Git book (free)	The complete Git reference	git-scm.com/book/en/v2
GitHub Docs	Repos, branches, Pull Requests	docs.github.com/en/get-started
Learn Git Branching	Interactive Git game	learngitbranching.js.org
GitHub Skills	Short hands-on courses inside GitHub	skills.github.com
VS Code Tips and Tricks	Editor shortcuts and features	code.visualstudio.com/docs/getstarted/tips-and-tricks
The Missing Semester (MIT)	Command line and shell, free videos	missing.csail.mit.edu
gitignore templates	Ready-made .gitignore files	github.com/github/gitignore
roadmap.sh	What to learn next, step by step	roadmap.sh

Recap and next class

Tools installed

VS Code, Git, Node.js, Python, GitHub

Terminal

cd, ls/dir, mkdir, code . and running files

VS Code

Layout, shortcuts, extensions, format on save

Git & GitHub

init, add, commit, push, branches, Pull Requests

NEXT: CLASS 2

Programming Fundamentals

Variables, conditions, loops, functions, data structures and OOP in JavaScript and Python.

Bring: your laptop with Node and Python working, and your Assignment 1 link.