THRIVE
Women for Women

THRIVE
Women for Women

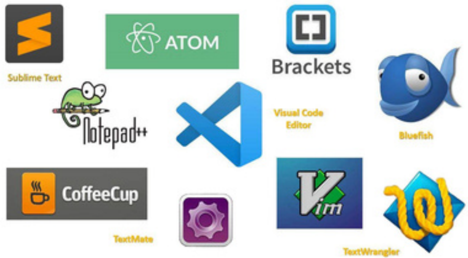
Introduction to VS Code and Version Control (Git & Github)

Lecture Flow

- Introduction to Code Editors
- VS Code(installation, features and shortcuts)
- Introduction to Version Control
- Git(installation, commands, key terms)
- GitHub(account, integration, key features)
- Bringing it all together

THRIVE
Women for Women

Introduction to Code Editors


ATOM
Brackets
Sublime Text
Notepad++
Visual Code Editor
Bluefish
CoffeeCup
TextMate
TextWrangler
vim

Code Editors

- Software programs that enable the user to create and edit text files
- Provide a simple and efficient way to write and edit code in various programming languages
- Unlike word processors, code editors don't have advanced formatting options or layout capabilities

THRIVE
Women for Women

Editors vs IDEs

- Lightweight feature set that can be expanded with extensions
- Consumes fewer system resources
- Simpler, less complex and easier to learn
- Examples include VS Code, Sublime Text,...

- All the functionality of code editors along with lots of additional features and tools
- Consumes more system resources
- Can be harder to learn
- Examples include Visual Studio, IntelliJ IDE, ...

Visual Studio Code(VS Code)

- **Lightweight and flexible code editor** used for writing and editing code across many languages.
- **Highly customizable through extensions**, enabling features like debugging, autocompletion, and version control.
- **Cross-platform and developer-friendly**, available on Windows, macOS, and Linux with an intuitive interface.

Key Features

THRIVE
Women for Women

- Integrated Terminal
- Intellisense
- Version Control
- Cross Platform
- Customizable
- AI Enhanced

Installation

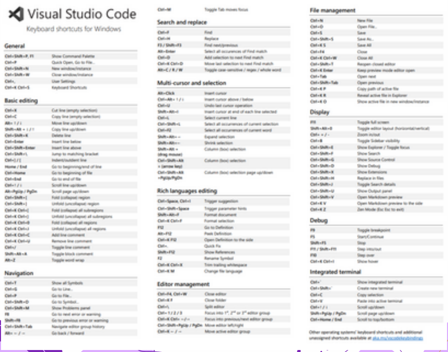
1. **Download VS Code:** Visit https://code.visualstudio.com/download and choose the installer that matches your operating system.
2. **Run the Installer:** Launch the downloaded executable and follow the setup wizard to complete the installation.
3. **Install Supporting Tools:** Install commonly used development tools such as Node.js, Git (we'll discuss this soon), and language runtimes like TypeScript if needed.
4. **Customize Your Environment:** Personalize VS Code with themes, formatters, icon packs, and language-specific extensions.

THRIVE
Women for Women

Basic VS Code Extensions

- **Prettier – Code Formatter** - Auto-formats your JavaScript/TypeScript code
- **ESLint** - Detects and fixes common JS/TS errors
- **Live Server** - Instantly preview frontend changes in the browser
- **Material Icon Theme -** Makes your project files easier to navigate visually
- **One Dark Pro (or any theme you like) -** Comfortable and clean coding interface

THRIVE
Women for Women



Visual Studio Code

Keyboard shortcuts for Windows

General

Ctrl+Shift+P, F1	Show Command Palette
Ctrl+P	Quick Open, Go to File…
Ctrl+Shift+N	New window/instance
Ctrl+Shift+W	Close window/instance
Ctrl+,	User Settings
Ctrl+K Ctrl+S	Keyboard Shortcuts

Basic editing

Ctrl+X	Cut line (empty selection)
Ctrl+C	Copy line (empty selection)
Alt+ ↑ / ↓	Move line up/down
Shift+Alt + ↓ / ↑	Copy line up/down
Ctrl+Shift+K	Delete line
Ctrl+Enter	Insert line below
Ctrl+Shift+Enter	Insert line above
Ctrl+Shift+\	Jump to matching bracket
Ctrl+] / [	Indent/outdent line
Home / End	Go to beginning/end of line
Ctrl+Home	Go to beginning of file
Ctrl+End	Go to end of file
Ctrl+↑ / ↓	Scroll line up/down
Alt+PgUp / PgDn	Scroll page up/down
Ctrl+Shift+[	Fold (collapse) region
Ctrl+Shift+]	Unfold (uncollapse) region
Ctrl+K Ctrl+[	Fold (collapse) all subregions
Ctrl+K Ctrl+]	Unfold (uncollapse) all subregions
Ctrl+K Ctrl+0	Fold (collapse) all regions
Ctrl+K Ctrl+J	Unfold (uncollapse) all regions
Ctrl+K Ctrl+C	Add line comment
Ctrl+K Ctrl+U	Remove line comment
Ctrl+/	Toggle line comment
Shift+Alt+A	Toggle block comment
Alt+Z	Toggle word wrap

Navigation

Ctrl+T	Show all Symbols
Ctrl+G	Go to Line…
Ctrl+P	Go to File…
Ctrl+Shift+O	Go to Symbol…
Ctrl+Shift+M	Show Problems panel
F8	Go to next error or warning
Shift+F8	Go to previous error or warning
Ctrl+Shift+Tab	Navigate editor group history
Alt+ ← / →	Go back / forward
Ctrl+M	Toggle Tab moves focus

Search and replace

Ctrl+F	Find
Ctrl+H	Replace
F3 / Shift+F3	Find next/previous
Alt+Enter	Select all occurences of Find match
Ctrl+D	Add selection to next Find match
Ctrl+K Ctrl+D	Move last selection to next Find match
Alt+C / R / W	Toggle case-sensitive / regex / whole word

Multi-cursor and selection

Alt+Click	Insert cursor
Ctrl+Alt+ ↑ / ↓	Insert cursor above / below
Ctrl+U	Undo last cursor operation
Shift+Alt+I	Insert cursor at end of each line selected
Ctrl+L	Select current line
Ctrl+Shift+L	Select all occurrences of current selection
Ctrl+F2	Select all occurrences of current word
Shift+Alt+ →	Expand selection
Shift+Alt+ ←	Shrink selection
Shift+Alt + (drag mouse)	Column (box) selection
Ctrl+Shift+Alt + (arrow key)	Column (box) selection
Ctrl+Shift+Alt +PgUp/PgDn	Column (box) selection page up/down

Rich languages editing

Ctrl+Space, Ctrl+I	Trigger suggestion
Ctrl+Shift+Space	Trigger parameter hints
Shift+Alt+F	Format document
Ctrl+K Ctrl+F	Format selection
F12	Go to Definition
Alt+F12	Peek Definition
Ctrl+K F12	Open Definition to the side
Ctrl+.	Quick Fix
Shift+F12	Show References
F2	Rename Symbol
Ctrl+K Ctrl+X	Trim trailing whitespace
Ctrl+K M	Change file language

Editor management

Ctrl+F4, Ctrl+W	Close editor
Ctrl+K F	Close folder
Ctrl+\	Split editor
Ctrl+ 1 / 2 / 3	Focus into 1st, 2nd or 3rd editor group
Ctrl+K Ctrl+ ←/→	Focus into previous/next editor group
Ctrl+Shift+PgUp / PgDn	Move editor left/right
Ctrl+K ← / →	Move active editor group

File management

Ctrl+N	New File
Ctrl+O	Open File…
Ctrl+S	Save
Ctrl+Shift+S	Save As…
Ctrl+K S	Save All
Ctrl+F4	Close
Ctrl+K Ctrl+W	Close All
Ctrl+Shift+T	Reopen closed editor
Ctrl+K Enter	Keep preview mode editor open
Ctrl+Tab	Open next
Ctrl+Shift+Tab	Open previous
Ctrl+K P	Copy path of active file
Ctrl+K R	Reveal active file in Explorer
Ctrl+K O	Show active file in new window/instance

Display

F11	Toggle full screen
Shift+Alt+0	Toggle editor layout (horizontal/vertical)
Ctrl+ = / -	Zoom in/out
Ctrl+B	Toggle Sidebar visibility
Ctrl+Shift+E	Show Explorer / Toggle focus
Ctrl+Shift+F	Show Search
Ctrl+Shift+G	Show Source Control
Ctrl+Shift+D	Show Debug
Ctrl+Shift+X	Show Extensions
Ctrl+Shift+H	Replace in files
Ctrl+Shift+J	Toggle Search details
Ctrl+Shift+U	Show Output panel
Ctrl+Shift+V	Open Markdown preview
Ctrl+K V	Open Markdown preview to the side
Ctrl+K Z	Zen Mode (Esc Esc to exit)

Debug

F9	Toggle breakpoint
F5	Start/Continue
Shift+F5	Stop
F11 / Shift+F11	Step into/out
F10	Step over
Ctrl+K Ctrl+I	Show hover

Integrated terminal

Ctrl+`	Show integrated terminal
Ctrl+Shift+`	Create new terminal
Ctrl+C	Copy selection
Ctrl+V	Paste into active terminal
Ctrl+↑ / ↓	Scroll up/down
Shift+PgUp / PgDn	Scroll page up/down
Ctrl+Home / End	Scroll to top/bottom

Other operating systems' keyboard shortcuts and additional unassigned shortcuts available at aka.ms/vscodekeybindings

THRIVE
Women for Women

Introduction to Version Control

Version control is a system that records changes to a file or set of files over time so that you can recall specific versions later.

This allows software developers to see the entire history of who changed what at any given time — and roll back from the current version to an earlier version if they need to.

Git

- Git is a **distributed version control system** used to track changes in source code during a software development
- Tracks changes in source code over time, enabling version comparison and rollback.
- Supports collaboration by allowing multiple people to work in parallel without overwriting each other's changes.
- Enables offline work, with commits and history available locally, syncing changes when connected.

Installing Git

1. Go to https://git-scm.com/download and choose your operating system.
2. Open the downloaded file and follow the setup wizard.
3. When prompted, keep the recommended/default options (no changes needed).
4. Finish & Verify Installation. Open Terminal / Command Prompt and type: “git --version” If a version number appears, git is installed successfully

GitHub

- A website and cloud-based service that helps developers store and manage their code, as well as track and control changes to their code
- It makes it a lot easier for individuals and teams to use Git for version control and collaboration
- Without GitHub, using Git generally requires a bit more technical savvy and use of the command line

Git vs GitHub

- A tool for tracking changes in code
- Runs locally on your computer
- Lets you commit, undo, and manage history of you project
- Can be used without internet connection

- A platform for storing Got repositories online
- Runs in the cloud(website/service)
- Helps you share, colaborate and review code with others
- Requires Internet to access remote repositories

Repository

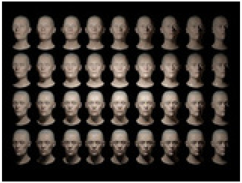
- A repository is a directory or folder that contains files and metadata for a project under version control.
    I. Local Repository: A Git repository on a local machine.
    II. Remote Repository: A Git repository on a remote server, such as GitHub

Syncing: Keeping the local and remote repositories up-to-date with each other.

Cloning

- Cloning a repository on GitHub allows you to make a copy of the repository on your local machine.
- This is useful if you want to work on the code locally, make changes, and then push those changes back to the GitHub repository.
- You can a clone a repository using this command:

```
git clone <URL>
```



Staging

The staging area (also called the index) is an intermediate space where changes are gathered before they are committed.

The staging area allows you to:

- Selectively choose changes to commit.
- Break down large changes into smaller, logical commits.
- Review your work before finalizing it in the repository.

Commiting

A commit is basically a snapshot of your project at a particular point in time.

A commit is a way of saving changes to your local Git repository.

Commits are like checkpoints, allowing you to track changes to your project over time.

Best Practices

- Commit frequently
- Only stage what changes you’re confident with
- Write descriptive commit messages
- Use atomic commits (let it represent a single contained change)
- Use simple present tense
  - Example : “Update Admin Dashboard”

Push

- The Git Push command is used to upload commits from your local repository to a remote repository (e.g., GitHub, G).
- Once pushed, other developers can access these changes and contribute by pulling them into their local repositories.
- Before pushing, you must commit your changes locally

Pull

- Git pull is a command which is used to fetch and integrate the changes which are present in the remote repository to the local repository.
- It is used to update the local branch with changes from the remote repository branch.
- This is useful when others have made changes to the remote repository that you want to incorporate into your local repository.

Branch

- A branch is a separate line of development that allows you to work on a feature or fix without affecting the main codebase.
- It allows multiple developers to work on the same codebase without conflicting change

How Git Branches Work?

- Main Branch: This is the primary branch where the stable and production-ready code resides.
- Creating a Feature Branch: When you want to work on a new feature or a bug fix, you create a new branch.
- Making Commits: As you develop, you make changes and record them as commits.
- Merging: Once you are satisfied with your work in the feature branch, you merge it back into the main branch. This integrates your changes into the stable version of the project, making them part of the overall codebase

Merge

- A merge is the process of combining changes from one branch into another branch.
- It is used to integrate the changes made in one branch into the main codebase and vice versa.

Pull Request

- A Pull Request is a feature in GitHub that allows users to propose changes to a repository and collaborate on those changes with other users.
- It is often used in open-source projects where multiple people may be contributing to the same codebase.

Basic Commands

- Initialize a new local repository [git init]
- Create a local copy of a remote repository (git clone <URL>)
- Check the status of your project (which files changed) (git status)
- Stage changes you intend to save (git add <file> or git add .)
- Commit the staged changes with a meaningful message (git commit -m "Your message")

Basic Commands

- Upload your local commits to the remote repository (git push)
- Download and integrate remote changes into your local repo (git pull)
- View the commit history and what changed (git log)
- Create and switch to a new branch to work safely (git checkout -b <branch>)
- Merge changes from one branch into another (git merge <branch>)

Exercise

1. Create a Repository on GitHub

- Go to GitHub and click New Repository.
- Name it my-first-repo (or any name).
- Check “Add a README” and create the repository.

2. Clone the Repository Locally

- Copy the repository URL.
- Open VS Code or Terminal and run:

```
git clone <URL>
```

- Open the cloned folder

3. Make a Change

- Create a new file called hello.txt.
- Add the text, e.g., “This is my first session at Thrive. It’s awesome”

4. Stage and Commit your changes

- stage the file

```
git add .
```

- commit with a message

```
git commit -m “Add hello.txt”
```

5. Push Changes to GitHub

- Upload your commit:

```
git push
```

- Refresh your GitHub repository to see the changes online.

6. Create a branch

- make a new branch for experimenting

```
git checkout -b thrive-branch
```

- Make a change, commit, and push it to the new branch.

THRIVE
Women for Women

Q/A

THRIVE
Women for Women

Thank You!