From Zero to Website: Crafting a Professional Academic Website with GitHub Pages and Jekyll

Instructor: Chosen Obih - PhD Student in the School of Plant Sciences, University of Arizona

Overview

Welcome to the workshop on building your own academic website using GitHub Pages and Jekyll! This guide will take you through the process step-by-step, accommodating users on Windows, Ubuntu, and macOS.

Workshop Outline

1. **Introduction and Overview**
2. **Setting Up GitHub and Installing Prerequisites**
3. **Cloning the Academic Pages GitHub Repository**
4. **Editing pages and content**
5. **Adding Content: Publications, CV, and Blog Posts**
6. **Deploying the Website on GitHub Pages**
7. **Customizing Further: Adding Plugins and Features**
8. **Q&A and Troubleshooting**
9. **Wrap-up**

1. Introduction and Overview

- Objective: Develop your personal website.
- Examples: Professional academic websites built with GitHub Pages and Jekyll
- Arun Seetharam
- Chosen Obih

2. Setting Up GitHub and Installing Prerequisites

Step 1: Setting Up Windows Subsystem for linux (WSL). If you a macOS or ubuntu user, skip this step and continue with Step 2.

1. Enable the Windows Subsystem for Linux:
    - Open PowerShell as an Administrator - Press **Win + S**, type PowerShell, right-click on it, and choose Run as Administrator.
    - Run the the command below to enable WSL and Virtual Machine Platform:
    - `wsl --install`
    - This command above will: Enable WSL, Install the necessary components (including the Virtual Machine Platform) and Download and install the default Linux distribution (typically Ubuntu).
    - Retart your computer when prompted
2. Install Ubuntu (if not installed during WSL setup):
    - After restarting, open Microsoft Store.
    - Search for Ubuntu.
    - Choose the version you prefer (e.g., Ubuntu 22.04 LTS, 20.04 LTS) and click Get or Install.
    - Wait for the installation to complete.
3. Set Up Ubuntu:
    - Launch Ubuntu: Open the Start menu, type Ubuntu, and press Enter.
    - Set up your user credentials: The first time you launch Ubuntu, it will prompt you to create a Linux username and password.
    - Enter a username and password (this is independent of your Windows credentials).
    - Your Ubuntu system is now ready to use!
    - Once inside Ubuntu, update the system by running the command below
    - `sudo apt update && sudo apt upgrade -y`
    - Install some essential packages
    - `sudo apt install git python3 build-essential`

Step 2: Set up Git and GitHub

- **What is Git?** Git is a version control system that helps you track changes in your code and collaborate with others.
- **What is GitHub?** GitHub is a platform for hosting and sharing code repositories online.
- **Key Git Commands:** **`git clone`** : Copies a remote repository to your local machine. **`git add`** : Stages changes for the next commit. **`git commit -m "message"`** : Saves changes to your local repository with a message. **`git push`** : Sends your changes to the remote GitHub repository.

1. Go to GitHub and create an account if you don't have one.
2. Download and install **Git**:
    - **Windows**: Git should already be installed in your WSL from the previous step.
    - **macOS**: Install via Homebrew (`brew install git`) or download from Git for macOS.
    - **Ubuntu**: Run `sudo apt update && sudo apt install git` .
3. Configure Git with your GitHub credentials:
    - `git config --global user.name "Your Name"`
    - `git config --global user.email "your.email@example.com"`

Step 3: Install Ruby, Bundle and VC Code

1. Install **Ruby**:
    - **macOS**: Install via Homebrew (`brew install ruby`).
    - **Ubuntu/Windows_WSL**: Run `sudo apt update && sudo apt install ruby-full` .
2. Install **VC Code** (code/text editor):
    - Download from VS Code's website, install, open the app and click on terminal.
    - For Windows WSL users, within the VS Code terminal, select Ubuntu WSL as the terminal choice
3. Install **Bundle** - the Ruby dependency manager by running. `sudo gem install bundler`

Additional step for Windows WSL users

- **Accessing Windows Files from WSL**
- In WSL, your Windows file system is mounted under `/mnt` directory. Each drive (e.g. C:, D:) is mounted as a subdirectory under `/mnt` .
- C Drive: `/mnt/c`
- D Drive: `mnt/d`
- Other drives: `/mnt/e` , `/mnt/f` , etc
- **Steps to navigate to a folder (Windows WSL users only)**
- Within your VS Code WSL terminal
- Type in the command below to change directory to your C drive
- `cd /mnt/c/Users/YourUsername/Documents`

3. Cloning the Academic Pages GitHub Repository

1. Log in to GitHub.

2. Visit this link in a new tab:
    - `https://github.com/academicpages/academicpages.github.io.git`
3. Fork the repo (Remember to rename it to a name you desire - this name will be in your website link)
4. Clone the repository you just forked to your local machine:
    - Open your VS code and click on terminal. Run the command below:
    - `git clone repo https link`
5. Navigate into the repository:
    - `cd your-username.github.io`
6. Open the repo folder in your VC Code

4. Editing pages and content

Step 1: Customize `_config.yml`

1. Open the `_config.yml` file and edit your personal details, such as:
    - **Name**
    - **Email**
    - **Social media links**
2. Run the site locally to test:
    - `bundle install` to install ruby dependencies. If you get errors, delete Gemfile.lock and try again.
    - `sudo bundle exec jekyll serve`
3. Visit `http://localhost:4000` in your browser to preview the site.

5. Adding more Content: Publications, CV, and Blog Posts

To be demonstrated in class

6. Deploying the Website on GitHub Pages

1. Commit your changes:
    - `git add .`
    - `git commit -m "Initial setup of the academic website"`
    - `git push origin main`
2. Go to the repository settings on GitHub, under Pages section, Build and deployment, set the source to the main branch.
3. Wait for a few minutes and access your live site at `https://your-username.github.io`.

7. Customizing Further: Adding Plugins and Features

1. Install plugins such as **Google Analytics** or **social sharing buttons**:
    - Add the plugin to your `Gemfile` and `_config.yml`.
    - Run `bundle install` to install the plugins.

8. Q&A and Troubleshooting

- Common issues: Missing dependencies, configuration errors, and troubleshooting theme display problems.
- Open Q&A session to address participant questions.

9. Wrap-up

- **Recap**: Steps covered during the workshop.
- **Resources**: Git and GitHub
- **PDF Guide**: Download the full workshop guide.

Thank you for participating in the workshop!