GitHub Pages Site Creation Tutorial

Using Jekyll and GitHub to Develop a

Site For Your Personal Portfolio

X

Simon McNeely & Aidan Jones

TWR2201

February 26, 2025

Table of Contents

Introduction

Welcome to the tutorial on Creating and Customizing a GitHub Pages Website using Jekyll!

Tutorial Purpose

The purpose of this tutorial is to teach beginners how to create and customize a website using GitHub Pages. This tutorial is designed for individuals with little to no prior experience in web development but who want to learn how to publish and manage a simple website. The tutorial covers essential steps such as setting up a GitHub repository, activating GitHub Pages, installing Jekyll, applying themes, and structuring content with multiple pages and navigation links.

Tutorial Context

GitHub Pages is a free web hosting service that allows users to create and publish static websites directly from a GitHub repository. It is widely used by developers, technical writers, and content creators for hosting project documentation, portfolios, and personal websites. Learning how to use GitHub Pages provides a strong foundation in version-controlled web publishing, making it an essential skill for individuals looking to showcase their work online or manage simple web-based projects efficiently.

Prerequisites

Successful completion of this tutorial requires the following:

- A GitHub account
- Access to the GitHub Desktop application
- A stable internet connection
- Access to an internet browser (ex. Google Chrome, Firefox, etc.)
- Access to a text editor, those pre-installed on a computer included, for example:
  - Notepad++ (Win)
  - TextEdit (Mac)
- Git, Ruby, and Bundler (see section *What you Must Install Before You Begin*)
- A computer with the following specifications, *at minimum*:

Required Computer Specifications

For Windows computers

- Windows 10 (64-bit) or later
- 3 or more GB of available hard drive space

For Mac computers

- macOS 11.0 (Big-Sur) or later
- 3 or more GB of available hard drive space

Tutorial Organization
This tutorial is broken down into two lessons, with each lesson containing two to three topics.

Lesson 1: Setting Up a Blank Site Using GitHub Pages – This lesson guides learners through the process of creating a GitHub Pages site, from setting up a repository and deployment branch to activating GitHub Pages and installing Jekyll.

Lesson 2: Customizing a GitHub Pages Site – This lesson teaches learners how to enhance their GitHub Pages site by applying themes, customizing the main page, adding additional pages, and creating navigation links for better site structure.

It is recommended that readers follow the lessons in order, as each lesson builds on the skills and tools introduced in the previous one. At the end of each lesson, there is a short exercise to reinforce the concepts learned.

The tutorial concludes with a brief summary of the key topics covered and how these skills can be applied to future website projects. The appendix includes additional resources for those who want to explore GitHub Pages and Jekyll in more depth.

Icon Legend

Note: The Note icon in this tutorial provides additional information or context that will help successfully complete the current task

Tip: The Tip icon will alert readers if there is an alternative method or shortcut to complete the current task

Ribbon: The Ribbon icon indicates that you have completed the current lesson and are prepared to continue to the next lesson.

Background
This section will provide the reader with basic information on the command line interface we will be using, a guide on how to read the syntax styling in the tutorial, the required resources, and additional recommended resources.

The Command Line Interface

For this tutorial, we will be using GitBash (for Windows users) or Terminal (for Mac users) as our command line interface. This section will provide the necessary information on these clients to complete this tutorial.

GitBash (Windows Users)

GitBash is the command-line interface that Windows users will be using throughout the duration of this tutorial.

The text in green, purple, and yellow at the top of your GitBash application is called the prompt. It should display information about the system the user is running GitBash on. The prompt should appear like so:

```
craig@SIMON MINGW64 ~
```

When the user changes the directory folder using the `cd` command, the prompt will show the current folder. In this tutorial, we will be changing the directory folder through GitBash. The updated prompt should appear like so:

```
craig@SIMON MINGW64 /C/Users/craig/Documents/GitHub
```

The line following the prompt should start with the dollar sign: `$`. On this line, the user types in commands to run through the GitBash client. In this tutorial, we will be using commands to initialize our website using Jekyll and Bundler. An example of an inputted prompt should appear like so:

```
$ git commit
```

Terminal (Mac Users)

Terminal is the command-line interface that Mac users will be using throughout the duration of this tutorial.

The text at the top of your Terminal application is called the prompt. It should display information about the system the user is running Terminal on. The prompt should appear like so:

```
Aidans-MacBook-Air:~ aidanjones$
```

When the user changes the directory folder using the `cd` command, the prompt will show the current folder. In this tutorial, we will be changing the directory folder through the Terminal. The updated prompt should appear like so:

```
/Users/aidanjones/Documents/GitHub/aidanjones1907.github.io
```

In this tutorial, we will be using commands to initialize our website using Jekyll and Bundler. An example of an inputted prompt should appear like so:

```
git commit
```

How to Read the Syntax Styling In This Tutorial

To make reading this tutorial more straightforward, many tools or actions will be marked by unique syntax styling, including bolding, quotations, font changes, and colour changes. This makes the tools or actions stand out so the reader can understand the material more quickly. The legend for syntax styling goes as follows:

Text that is **bolded** indicates the name of an application, a tool, a button, or an action.

Text that is encased in "quotations" indicates a file, folder, URL, or a text snippet. This rule does not apply to code snippets.

Text that is written in the font `Courier New` indicates a code snippet or a console command. Make sure to read the instructions carefully so you know what text is included in the command or code snippet.

Text that is coloured red indicates text that you, the reader, can customize. Customizing this text may be mandatory. Make sure to read the steps carefully to know when you are required to change the text marked as customizable.

What You Must Install Before You Begin

This project requires the use of Jekyll and Bundler. To use Jekyll and Bundler on a computer, you need to have Ruby installed. The process for downloading Ruby onto a computer is different for Windows and Mac users. The following section provides download sources and/or instructions for Jekyll, Ruby, and Bundler for both Windows and Mac users.

Windows Users

Windows users will need to download two .exe files, the Ruby Installer, and Git. When installing Git onto a computer, the installer will also install the GitBash client. GitBash will act similarly to the 'Command Prompt' client that comes pre-installed on a Windows computer.

To install Ruby, download the latest version of Ruby Installer from the link below:
https://rubyinstaller.org/

To install Git and GitBash, download the latest version of the 'Standalone Installer' that suits your computer's version from the link below:
https://git-scm.com/downloads/win

Mac Users

Mac users will have to install Ruby by using commands through the 'Terminal' client. When following this tutorial, the Terminal client will be able to respond to Ruby commands.

To install Ruby onto your computer, follow the procedures found in this tutorial:
https://jekyllrb.com/docs/installation/macos/

Recommended Tools

The following section provides options for additional tools that may help with completing this tutorial. None of the following tools are necessary to follow along with this tutorial. However, these tools provide quality-of-life improvements that may help you achieve your goals easier.

Visual Studio Code

Visual Studio Code (VSCode) is a code editor application that has a variety of customization options. Features such as syntax highlighting will allow you to find and edit code snippets much easier. In addition, bug detection tools embedded within VSCode will allow you to catch errors made before compiling the code. Finally, Visual Studio Code allows you to run GitBash from within the VSCode client, which can reduce clutter on your screen.

To install VSCode onto your computer, follow this link:
https://code.visualstudio.com/

BBEdit

BBEdit is a code editor application that is more bare-bones than VSCode if you are looking for a simpler alternative text editor. BBEdit includes support for Markdown and HTML, syntax colouring, text handling, and more.

To install BBEdit onto your computer, follow this link:
https://www.barebones.com/products/bbedit/

Markdown Live Preview

Markdown Live Preview is a web-based markdown compiler. When pushing updates to your site through GitHub, the changes take several minutes to appear on the live site. Therefore, reformatting the pages may take extra time because you may be waiting between each small change. Using an online markdown previewer, you can preview minor changes much quicker, therefore reducing the waiting times by waiting to push until all your changes are complete.

To use Markdown Live Preview, follow this link:
https://markdownlivepreview.com/

Lesson 1: Setting Up a Blank Site Using GitHub Pages

In this lesson, learners will begin the process of setting up a basic website using GitHub Pages—a tool that allows anyone to create and host a website directly from a GitHub repository. This is ideal if you are a beginner web developer looking to create a personal website to showcase your projects. This lesson will guide you through the essential steps to get started.

At the end of this lesson, learners will be able to:

- Create a GitHub Pages site from a GitHub repository
- Install and configure Jekyll, a static site generator, on their GitHub Pages site
- Deploy their site live and make it accessible to the public

Here, learners will create and configure a GitHub repository, set up a deployment branch, and activate GitHub Pages to publish their site. They will then use Git Bash to locate their site and install Jekyll to enable advanced site customization. The lesson concludes with a Multiple-Choice exercise assessing your understanding of the foundational steps required to set up a GitHub Pages site.

By the end of the lesson, learners will have a functional GitHub Pages site ready for further development.

Topic 1: Creating the GitHub Pages Site

Before a website can be published using GitHub Pages, it must be properly set up within a GitHub repository. This topic introduces the fundamental steps needed to establish a GitHub Pages site, ensuring that it is correctly structured for future development and deployment.

In this topic, learners will:

- Create a Repository on GitHub Desktop to store and manage their website's files
- Create a Deployment Branch to control how and when content is published

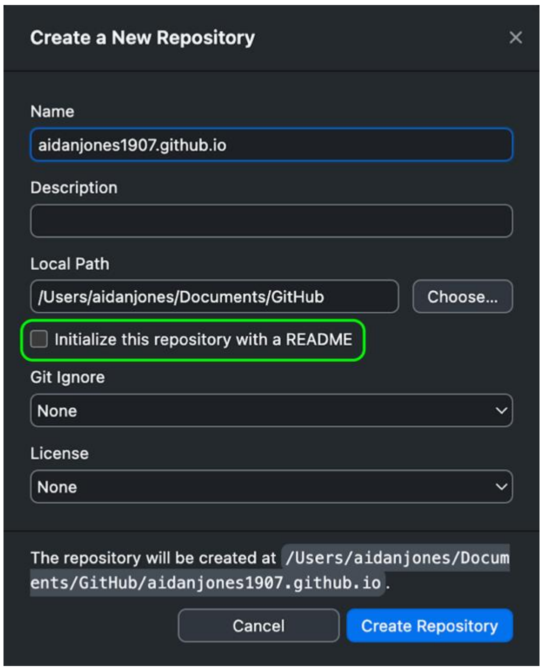
By completing these steps, learners will have a fully prepared GitHub repository that is ready to be activated as a GitHub Pages site.

To Create a Repository on GitHub Desktop

1. Open the **GitHub Desktop** application
2. In the top left corner, under **File**, select **New Repository**
3. In the **Name** text field, enter your GitHub username, followed by ".github.io". The **Name** field should then be populated with "username.github.io"

*Note: This format of repository name is specifically used to create a **User or Organization Site**; other types of sites will require a different naming convention*

4. Ensure that the **Initialize this repository with a README** checkbox is unchecked


Create a New Repository

Name: aidanjones1907.github.io

Description:

Local Path: /Users/aidanjones/Documents/GitHub

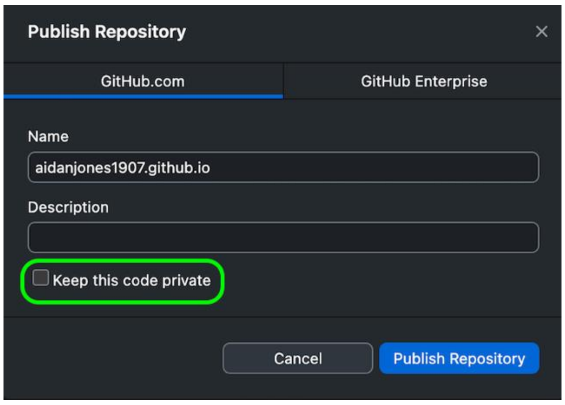
☐ Initialize this repository with a README

Git Ignore: None

License: None

The repository will be created at `/Users/aidanjones/Documents/GitHub/aidanjones1907.github.io`.

5. Select **Create Repository**
6. Select **Publish Repository**
7. In the window that appears, ensure that the **Keep this code private** checkbox is unchecked


Publish Repository

Name: aidanjones1907.github.io

Description:

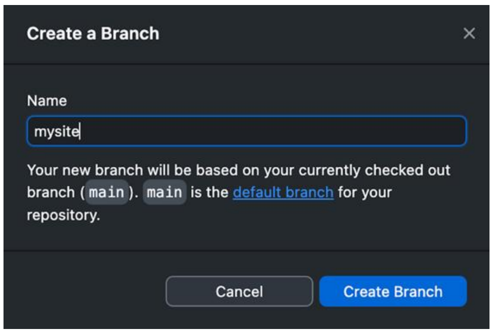
☐ Keep this code private

8. Ensure that the **Name** text field is populated with the "username.github.io" entry from Step 3
9. Select **Publish Repository**

To Create a Deployment Branch

1. Open the **GitHub Desktop** application
2. Ensure that the "username.github.io" repository is currently selected

3. In the top left corner, under **Branch**, select **New Branch**
4. In the **Name** text field, enter a recognizable title, such as "mysite"



Create a Branch

Name

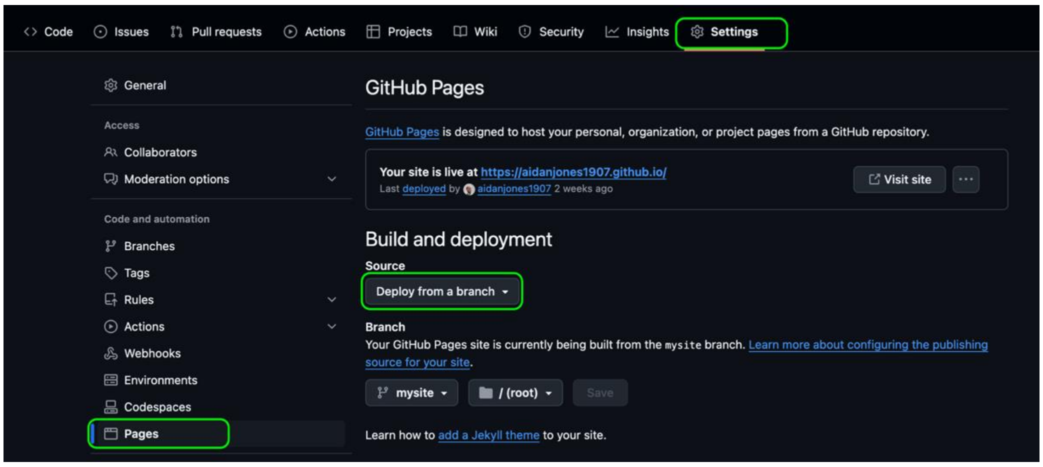
mysite

Your new branch will be based on your currently checked out branch (`main`). `main` is the default branch for your repository.

5. Select **Create Branch**
6. Select **Publish Branch**

To Activate a GitHub Pages Site

1. Navigate GitHub.com to the repository of your site, which can be found at "https://github.com/username/username.github.io"
2. In the top banner, select **Settings**
3. On the left side of the page, select **Pages** from the **Code and Automation** menu
4. From the dropdown menu entitled **Source**, found under the **Build and Deployment** menu, ensure that **Deploy from a branch** is selected



GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

Your site is live at https://aidanjones1907.github.io/

Last deployed by aidanjones1907 2 weeks ago

Build and deployment

Source

Deploy from a branch

Branch

Your GitHub Pages site is currently being built from the `mysite` branch. Learn more about configuring the publishing source for your site.

mysite / (root)

Learn how to add a Jekyll theme to your site.

5. Under the **Branch** menu, select the branch you created for your site
6. Select **Save**

Note: A textbox at the top of the page should appear reading "Your site is live at [https://[username].github.io/](https://[username].github.io/). If it does not appear, refresh the page. Visiting this site will currently lead to an Error 404 display page, as the site has currently not been configured yet."

Topic 2: Installing Jekyll to the GitHub Pages Site

Once a GitHub Pages site is set up, it can be enhanced using Jekyll, a static site generator that simplifies website creation and customization. This topic covers the process of activating a GitHub Pages site and integrating Jekyll to enable more advanced web development capabilities.

By activating a GitHub Pages site and configuring Jekyll, learners will gain the ability to customize their site's appearance, manage content more efficiently, and use built-in features such as templates and themes.

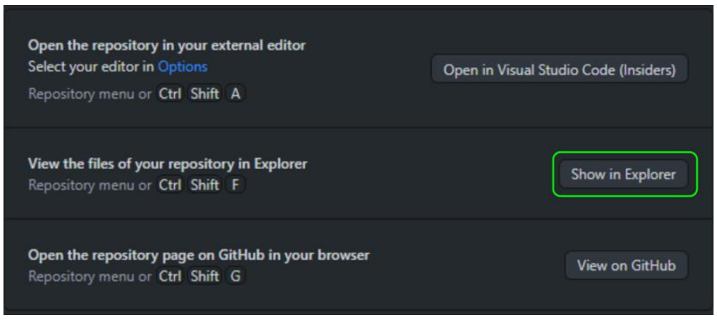
In this topic, learners will:

- Activate a GitHub Pages Site to publish their repository as a live website
- Locate Their Site in Git Bash to verify that it is properly set up
- Activate Their Site with Jekyll to enable advanced customization and content management

By completing these steps, learners will have a fully functional GitHub Pages site powered by Jekyll, providing a solid foundation for further development.

To Locate Your Site in Git Bash

1. Open the **GitHub Desktop** Application
2. Ensure that the "username.github.io" repository is currently selected
3. Select **Show in Explorer**



Open the repository in your external editor
Select your editor in Options
Repository menu or Ctrl Shift A
Open in Visual Studio Code (Insiders)

View the files of your repository in Explorer
Repository menu or Ctrl Shift F
Show in Explorer

Open the repository page on GitHub in your browser
Repository menu or Ctrl Shift G
View on GitHub

*Note: For Mac users, select **Show in Finder***

4. Copy the path of the "username.github.io" repository

*Tip: For Mac users, under **View**, select **Show Path Bar** if not already enabled to more easily copy the path of the repository.*

5. Open the **Git Bash** application

*Note: Mac users will open the **Terminal** application*

6. In the text field, enter `cd` followed by the path of the repository

Note: Ensure that all slash-symbols entered in the pathway are forward slashes (/), not back slashes (\).

7. Press **Enter** on your keyboard.

To Activate Your Site with Jekyll

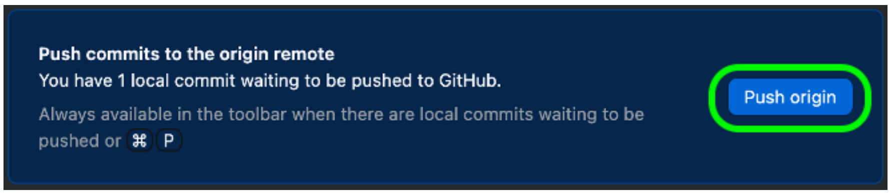
1. With your "username.github.io" repository open in your respective terminal, run the command `jekyll new –skip-bundle .`
2. Open the "username.github.io" repository in the **File Explorer** and locate the file titled "Gemfile"

*Note: For Mac users, open the repository in **Finder**. If the `jekyll new –skip-bundle .` command creates a folder called "–skip-bundle," move the files from the "–skip-bundle" folder into the "username.github.io" folder and then delete the "–skip-bundle" folder.*

3. In the "Gemfile" file, enter # at the beginning of the line of text that reads `gem "jekyll"`
4. In a new line, copy and paste `"gem github-pages", "~> 232", group: :jekyll_plugins` into the file
5. Save and close the "Gemfile" file
6. In your respective terminal, run the command `bundle install`
7. In the "username.github.io" folder, select and open the file named ".gitignore" in a text editor of your choice

*Note: For Mac users, this file is hidden by default. Press **Command** + **Shift** + **.** to make the file visible.*

8. In a new line, copy and paste `Gemfile.lock` into the file and save the file
9. In GitHub Desktop, select **Push Origin**


Push commits to the origin remote
You have 1 local commit waiting to be pushed to GitHub.
Always available in the toolbar when there are local commits waiting to be pushed or ⌘ P

Exercise 1: Multiple Choice

This exercise is designed to assess your understanding of the foundational steps required to set up a GitHub Pages site, including creating a repository, setting up a deployment branch, activating GitHub Pages, and installing Jekyll. Successfully setting up a GitHub Pages site is crucial for publishing and managing web content. By mastering these steps, you will be able to host and maintain your own website efficiently.

Instructions: Choose the correct answer from the choices below. There is only one correct per question. Correct answers to the questions can be found in the Appendix.

What is the correct format for naming a GitHub Pages repository?

A) mywebsite.github.com
B) username.github.io
C) github.io/username
D) pages.github.io

Which command is used to install Jekyll on your GitHub Pages site?

A) `jekyll install`
B) `jekyll new -skip-bundle .`
C) `bundle install`
D) `git push origin`

After publishing your repository, where can you activate GitHub Pages?

A) In the **Issues** tab of the repository
B) In the **Settings** tab under **Pages**
C) In the **GitHub Desktop** application
D) In **Git Bash** using a terminal command

Which command is used in Git Bash to navigate to your GitHub Pages repository?

A) `cd /GitHubPages`
B) `git init`
C) `cd repository-path`
D) `git clone repository-url`

Congratulations, you have now completed Lesson 1: Setting Up a Blank Site Using GitHub Pages!

You should now successfully be able to create a GitHub Pages site from a GitHub repository, install and configure Jekyll on your GitHub Pages site, and deploy a site live and make it accessible to the public.

Lesson 2: Customizing a GitHub Pages Site

In this lesson, learners will enhance the appearance and functionality of their GitHub Pages site by customizing its design and structure. This is ideal if you have set up a basic website, but you want it to look more polished and include multiple pages for different content. This lesson will guide you through the process of selecting a theme, modifying the main page, and adding new pages to improve your site's presentation and usability.

At the end of this lesson, learners will be able to:

- Select and apply a theme to change the visual design of their GitHub Pages site
- Customize the main page by overriding the default content
- Create and link additional pages to expand their site's structure

The lesson concludes with a Multiple-Choice exercise assessing your understanding of how to customize a GitHub Pages site, including selecting and applying themes, modifying the main page, and expanding the site with additional pages and links.

By the end of the lesson, learners will have a well-structured and visually appealing website that serves as a strong foundation for further development.

Topic 1: Adjusting the Appearance of a GitHub Pages Site

This topic introduces learners to the process of customizing the appearance of their GitHub Pages site by selecting and applying a theme, as well as modifying the main page's content. These steps help create a polished, professional-looking website. By learning to apply themes and modify the main page, learners will be able to create a site that aligns with their vision and needs.

In this topic, learners will:

- Select a Theme for a GitHub Pages Site to choose a design that fits their style and purpose
- Add a Theme to a GitHub Pages Site to apply the selected theme and update the site's appearance
- Override the Main Page of a GitHub Pages Site to customize the homepage with unique content

By completing these steps, learners will have a visually enhanced GitHub Pages site that is both professional and personalized.

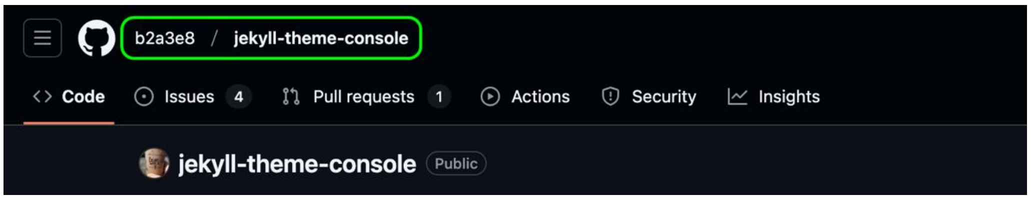
To Select a Theme for a GitHub Pages Site

1. In your browser of choice, navigate to "https://github.com/topics/jekyll-theme"

2. Select a theme that you'd like to adapt your GitHub Pages site to

Note: For the sake of this tutorial, a basic temporary theme can be used such as "jekyll-theme-console" created by "b2a3e8"

3. Select the name of the theme to navigate to the theme's page
4. Record the author's username and theme name of your theme of choice



To Add a Theme to a GitHub Pages Site

1. Using the **File Explorer**, open the "username.github.io" repository folder

*Note: For Mac users, open the repository in **Finder.***

2. Open the file titled "_config.yml"
3. Navigate to the line starting with `remote_theme:`

Note: If there is no line of text that contains `remote_theme:` add the text line anywhere in the document on a line by itself

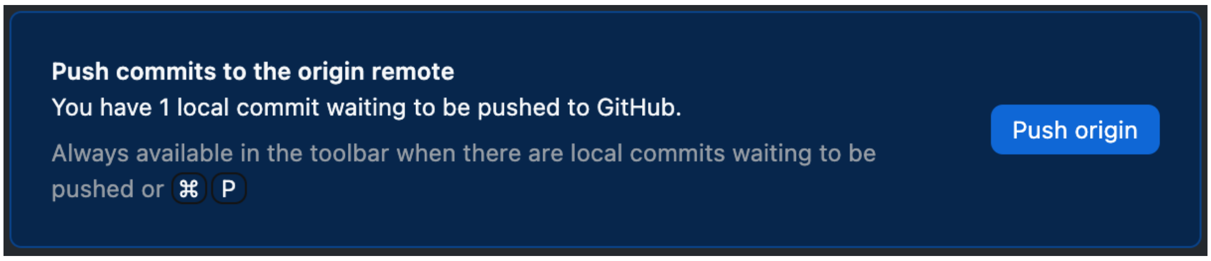
4. Replace all text on the line that follows `remote_theme:` with the `author's username` and `theme name` you recorded in Step 3, between two quotations marks (`""`) and separated by a single forward slash (`/`)

Note: Done correctly, this should look like `remote_theme: "author's username/theme name"`

Note: Some themes may require users to enter in a version number at the end of the line; it is best practice to check the repository's instructions for installing a remote theme before installation.

5. Navigate to the line starting with `url:`
6. Between the two quotation marks on the line that follows `url:` enter `https://username.github.io/`
7. Save and close the "_config.yml" file

8. In GitHub Desktop, select **Push Origin**


Push commits to the origin remote
You have 1 local commit waiting to be pushed to GitHub.
Always available in the toolbar when there are local commits waiting to be pushed or ⌘ P

Note: Commit summary text is arbitrary and will not affect the final product.

To Override the Main Page of a GitHub Pages Site

1. Using the **File Explorer**, open the "username.github.io" repository folder

*Note: For Mac users, open the repository in **Finder.***

2. Locate and open the file titled "index.md" in the text editor of your choice

Note: If only a file titled "index.html" is present, delete the file and create a file titled "index.md"

Note: Files that end in ".md" are sometimes also expressed as ending in ".markdown". No functional difference exists between the two.

3. At the top of the "index.md" file, paste the following code on five separate lines:

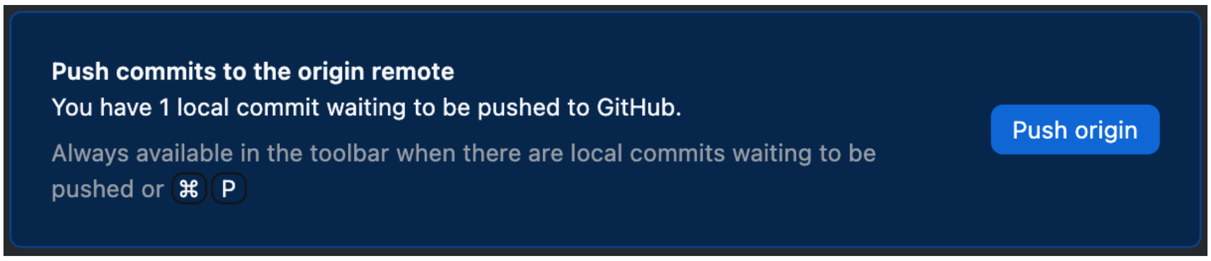
```
—
layout: page
title: /home
permalink: /
—
```

4. Customize the "index.md" file to your preferences using Markdown syntax

Note: For the purposes of this tutorial, basic filler text will suffice and can be edited more thoroughly at a later time.

5. Save and close the "index.md" file

6. In GitHub Desktop, select **Push Origin**



Push commits to the origin remote
You have 1 local commit waiting to be pushed to GitHub.
Always available in the toolbar when there are local commits waiting to be pushed or ⌘ P

Note: Commit summary text is arbitrary and will not affect the final product.

Topic 2: Expanding a GitHub Pages Site

This topic focuses on expanding a GitHub Pages site by adding additional pages and creating hyperlinks to connect them, transforming a basic landing page into a structured, multi-page website.

Adding additional pages allows users to separate content into different sections, making information more accessible and improving navigation. Creating hyperlinks between these pages ensures that visitors can easily explore the site without confusion.

In this topic, learners will:

- Create Additional Pages of a GitHub Pages Site to organize content into separate sections
- Create a Hyperlink to a Page of a GitHub Pages Site to connect pages and improve site navigation

By completing these steps, learners will develop a well-structured website with clear navigation, making it easier for visitors to find and interact with their content.

To Create Additional Pages of a GitHub Pages Site

1. Using the **File Explorer**, open the "username.github.io" repository folder

*Note: For Mac users, open the repository in **Finder.***

2. Create a new folder titled "pages"
3. In the "pages" folder, create a new file titled "newpage.md"
4. Open the "newpage.md" file in a text editor of your choice

5. At the top of the "newpage.md" file, paste the following code on five separate lines:

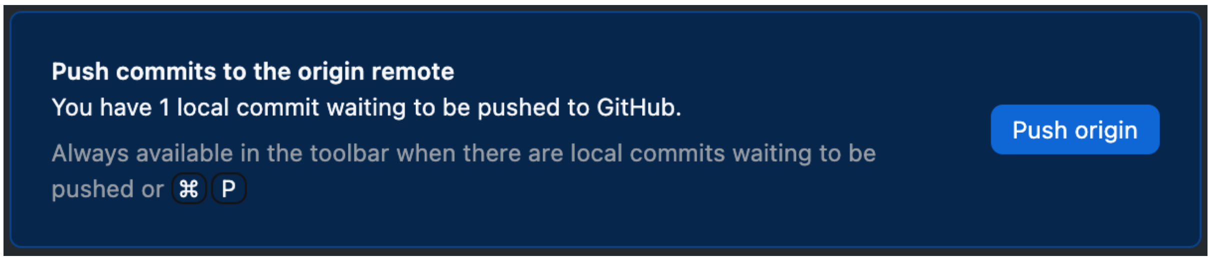
```
–
layout: page
title: New Page
permalink: /newpage/
–
```

Note: For the purpose of this tutorial, the text above is being used for `title` *and* `permalink` *parameters. When creating a new page, you can name these as you please so long as the characters are alphanumerical. The* `permalink` *parameter must not contain any spaces.*

6. Customize the "newpage.md" file to your preferences using Markdown syntax

Note: For the purposes of this tutorial, basic filler text will suffice and can be edited more thoroughly at a later time.

7. Save and close the "newpage.md" file
8. In GitHub Desktop, select **Push Origin**



Push commits to the origin remote
You have 1 local commit waiting to be pushed to GitHub.
Always available in the toolbar when there are local commits waiting to be pushed or ⌘ P

Note: Commit summary text is arbitrary and will not affect the final product.

To Create a Hyperlink to a Page of a GitHub Pages Site

1. Open the Markdown file of the page you would like to create a link to in the text editor of your choice

Note: For the sake of this tutorial, if you do not yet have a page to link to, a link can be created to the "index.md" file

2. On a new line in the file of choice, enter `[][]` with no spaces between the brackets
3. Between the first set of brackets, enter the text that you would like to appear on the page of your site, such as `Open this page`
4. Between the second set of brackets, enter a tag for your page, such as `new page`.
   Note: Done correctly, *this should look like* `[Open this page][new page]`.

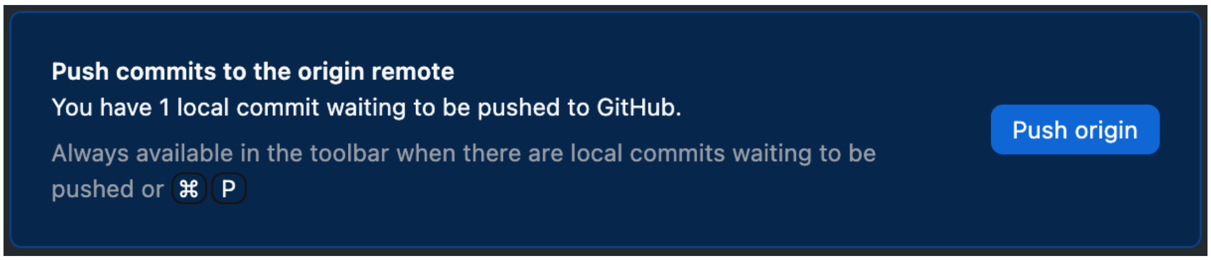
5. On another new line below the previous line, enter the tag name in square brackets, followed by a colon

Note: Done correctly, this should look like `[new page]`*:*

6. Add a space after the colon
7. On the same line, after the space, enter the hyperlink to the page, which will be formatted as `https://www.username.github.io/newpage/`

Note: For the purposes of this tutorial, we are attaching `newpage/` *to the end of our hyperlink. If you named the* `permalink` *parameter at the top of your page file something else, you would use that name.*

8. Save and close the Markdown file
9. In GitHub Desktop, select **Push Origin**



Push commits to the origin remote
You have 1 local commit waiting to be pushed to GitHub.
Always available in the toolbar when there are local commits waiting to be pushed or ⌘ P

Note: Commit summary text is arbitrary and will not affect the final product.

Exercise 2: Multiple Choice

This exercise will help you evaluate your understanding of how to customize a GitHub Pages site, including selecting and applying themes, modifying the main page, and expanding the site with additional pages and links. A well-structured and visually appealing website improves user experience and readability. Understanding how to modify a GitHub Pages site ensures that your content is both engaging and easy to navigate.

Instructions: Choose the correct answer from the choices below. There is only one correct per question. Correct answers to the questions can be found in the Appendix.

What is the purpose of adding a theme to a GitHub Pages site?

A) To increase the site's loading speed
B) To provide a consistent design and layout
C) To make the repository private
D) To prevent others from viewing the site

Which file must be modified to apply a remote theme to a GitHub Pages site?

A) “index.html”
B) “_config.yml”
C) “theme.css”
D) “style.md”

How can you override the main page of a GitHub Pages site?

A) By renaming the repository
B) By modifying the “_config.yml” file
C) By editing or creating an “index.md” file
D) By selecting a different branch in the repository

What is the correct way to create an additional page in a GitHub Pages site?

A) Add a new Markdown file with a permalink in the repository
B) Modify the existing “_config.yml” file
C) Create a new repository for each page
D) Use Git Bash to generate new HTML files automatically

Congratulations, you have now completed Lesson 2: Customizing a GitHub Pages Site!

You should now successfully be able to select and apply a theme to change the visual design of your GitHub Pages site, customize the main page by overriding the default content, and create and link additional pages to expand your site’s structure.

Tutorial Summary

You have successfully completed the tutorial on how to create and customize a website using GitHub Pages. The skills you have acquired in this tutorial are essential for publishing and managing static websites and can be applied to various web development and documentation projects.

After completing this tutorial, you should now be able to:

- Set up a GitHub repository and activate a GitHub Pages site
- Install and configure Jekyll for enhanced site customization
- Apply and modify themes to adjust the appearance of your site
- Create additional pages and link to them for better site navigation

You should also be familiar with using the following tools and features:

- GitHub Desktop for repository management
- Git Bash for command-line navigation and site configuration
- Jekyll for generating and managing static web pages
- Markdown for structuring content on your site

The skills you have learned in this tutorial are valuable for technical communicators who need to create and maintain online documentation, portfolios, or project sites. Understanding GitHub Pages provides a strong foundation in version-controlled web publishing, a skill that is highly relevant to the field of technical writing.

GitHub Pages and Jekyll offer many more customization options beyond what was covered in this tutorial. If you would like to explore additional features, check out the resources listed in the Appendix.

Congratulations on completing the tutorial! Good luck with all future GitHub Pages projects.

Appendix

The following appendices will provide answers to the multiple-choice questions from the two exercises and additional resources for customizing your Jekyll site.

Multiple Choice Answers

The following sections include answers to the multiple-choice questions from Lesson 1: Setting Up a Blank GitHub Pages Site and Lesson 2: Customizing a GitHub Pages Site.

Exercise 1: Multiple Choice

What is the correct format for naming a GitHub Pages repository?

B) username.github.io

Which command is used to install Jekyll on your GitHub Pages site?

B) `jekyll new –skip-bundle .`

After publishing your repository, where can you activate GitHub Pages?

B) In the **Settings** tab under **Pages**

Which command is used in Git Bash to navigate to your GitHub Pages repository?

A) `cd /GitHubPages`

Exercise 2: Multiple Choice

What is the purpose of adding a theme to a GitHub Pages site?

B) To provide a consistent design and layout

Which file must be modified to apply a remote theme to a GitHub Pages site?

B) "_config.yml"

How can you override the main page of a GitHub Pages site?

C) By editing or creating an "index.md" file

What is the correct way to create an additional page in a GitHub Pages site?

A) Add a new Markdown file with a permalink in the repository

Additional Markdown Resources

This appendix will provide links to resources that will help you customise your Jekyll site. Jekyll uses the kramdown markdown processor, so some of the default markdown options may or may not be available. The following links will provide reference material so you can understand the features available on Jekyll sites.

This link provides a list of markdown features that Jekyll supports:

https://www.markdownguide.org/tools/jekyll/

This link provides documentation for basic markdown syntax that may be useful for your site:

https://www.markdownguide.org/basic-syntax/

This link provides documentation for additional markdown syntax that may be useful for your site:

https://www.markdownguide.org/extended-syntax/

This link provides introductory tutorials for HTML syntax. Markdown files in Jekyll allow for embedded HTML, so this may be useful if you are interested in further customization:

https://www.w3schools.com/html/html_intro.asp

Think link provides access to a short youtube course on GitHub Pages and Jekyll if you are interested in in-depth information on the topic:

https://youtube.com/playlist?list=PL3MdArvT5LVdkeEZ6x6feSMRjYjoMVEUZ&si=DsmXQKSZ1wOh98G_