


Adding content to your GitHub Pages site using Jekyll

 docs.github.com/en/pages/setting-up-a-github-pages-site-with-jekyll/adding-content-to-your-github-pages-site-using-jekyll

You can add a new page or post to your Jekyll site on GitHub Pages.

GitHub Pages is available in public repositories with GitHub Free and GitHub Free for organizations, and in public and private repositories with GitHub Pro, GitHub Team, GitHub Enterprise Cloud, and GitHub Enterprise Server. For more information, see "[GitHub's products](#)."

People with write permissions for a repository can add content to a GitHub Pages site using Jekyll.

About content in Jekyll sites

Before you can add content to a Jekyll site on GitHub Pages, you must create a Jekyll site. For more information, see "[Creating a GitHub Pages site with Jekyll](#)."

The main types of content for Jekyll sites are pages and posts. A page is for standalone content that isn't associated with a specific date, such as an "About" page. The default Jekyll site contains a file called `about.md`, which renders as a page on your site at `YOUR-SITE-URL/about`. You can edit the contents of that file to personalize your "About" page, and you can use the "About" page as a template to create new pages. For more information, see "[Pages](#)" in the Jekyll documentation.

A post is a blog post. The default Jekyll site contains a directory named `_posts` that contains a default post file. You can edit the contents of that post, and you can use the default post as a template to create new posts. For more information, see "[Posts](#)" in the Jekyll documentation.

Your theme includes default layouts, includes, and stylesheets that will automatically be applied to new pages and posts on your site, but you can override any of these defaults. For more information, see "[About GitHub Pages and Jekyll](#)."

To set variables and metadata, such as a title and layout, for a page or post on your site, you can add YAML front matter to the top of any Markdown or HTML file. For more information, see "[Front Matter](#)" in the Jekyll documentation.

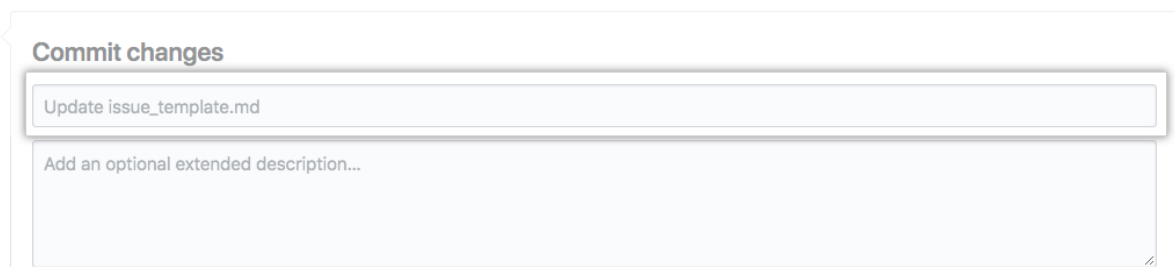
Changes to your site are published automatically when the changes are merged into your site's publishing source. If you want to preview your changes first, you can make the changes locally instead of on GitHub. Then, test your site locally. For more information, see "[Testing your GitHub Pages site locally with Jekyll](#)."

Adding a new page to your site

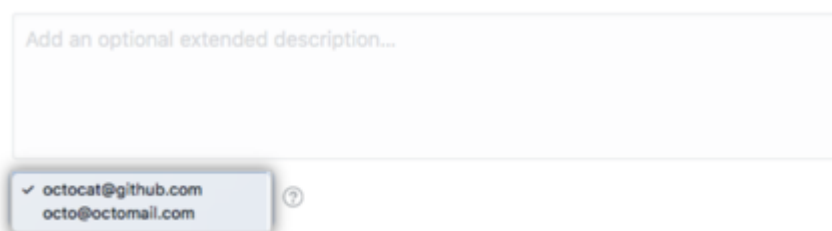
1. On GitHub, navigate to your site's repository.
2. Navigate to the publishing source for your site. For more information about publishing sources, see "[About GitHub Pages](#)."
3. In the root of your publishing source, create a new file for your page called *PAGE-NAME.md*, replacing *PAGE-NAME* with a meaningful filename for the page.
4. Add the following YAML frontmatter to the top of the file, replacing *PAGE TITLE* with the page's title and *URL-PATH* with a path you want for the page's URL. For example, if the base URL of your site is `https://octocat.github.io` and your *URL-PATH* is `/about/contact/`, your page will be located at `https://octocat.github.io/about/contact`.

```
layout: page
title: "PAGE TITLE"
permalink: /URL-PATH/
```

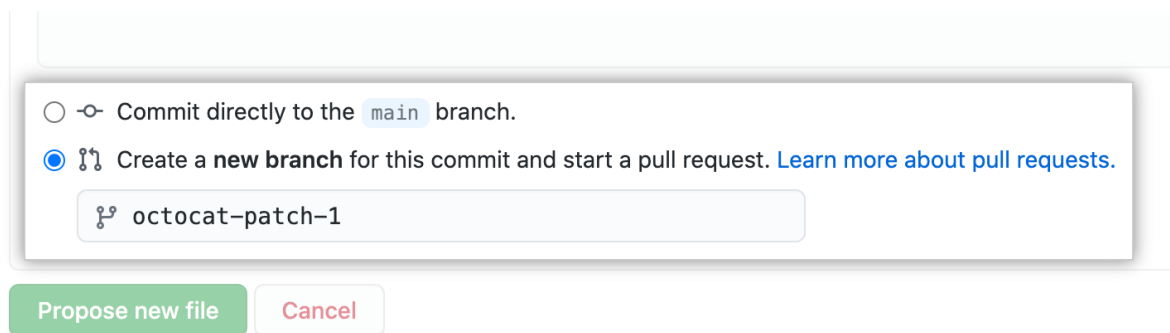
5. Below the frontmatter, add content for your page.
6. At the bottom of the page, type a short, meaningful commit message that describes the change you made to the file. You can attribute the commit to more than one author in the commit message. For more information, see "[Creating a commit with multiple co-authors](#)."



7. If you have more than one email address associated with your GitHub account, click the email address drop-down menu and select the email address to use as the Git author email address. Only verified email addresses appear in this drop-down menu. If you enabled email address privacy, then `<username>@users.noreply.github.com` is the default commit author email address. For more information, see "[Setting your commit email address](#)."

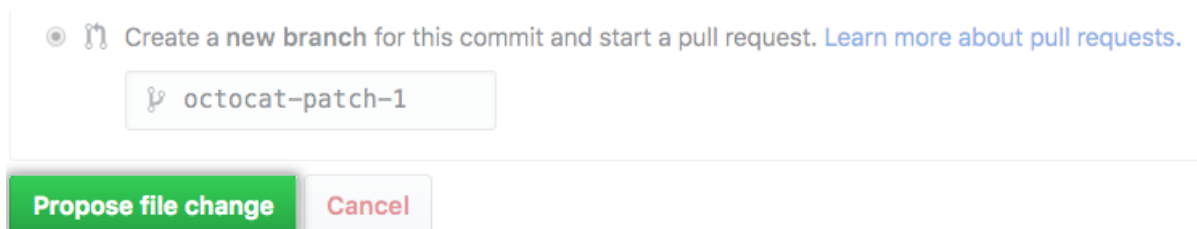


8. Below the commit message fields, decide whether to add your commit to the current branch or to a new branch. If your current branch is the default branch, you should choose to create a new branch for your commit and then create a pull request. For more information, see "[Creating a new pull request](#)."



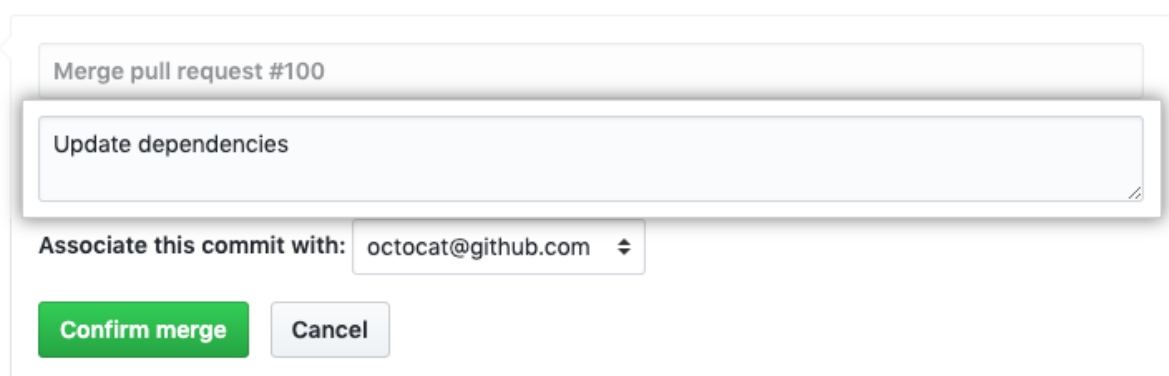
The screenshot shows a GitHub commit dialog. At the top, there's a text input field for the commit message. Below it, there are two radio button options: "Commit directly to the `main` branch." (which is unselected) and "Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)" (which is selected). Below the second option, there is a text input field containing the branch name `octocat-patch-1`. At the bottom, there are two buttons: "Propose new file" (in green) and "Cancel" (in red).

9. Click **Propose file change**.



This screenshot is similar to the previous one, showing the same GitHub commit dialog. The "Create a new branch" option is still selected, and the branch name `octocat-patch-1` is in the input field. In this version, the "Propose file change" button is highlighted in green, indicating it is the next step to click.

10. Create a pull request for your proposed changes.
11. In the "Pull Requests" list, click the pull request you would like to merge.
12. Click **Merge pull request**. For more information, see "[Merging a pull request](#)."
13. If prompted, type a commit message, or accept the default message.



The screenshot shows a GitHub dialog for merging a pull request. At the top, it says "Merge pull request #100". Below that is a text input field with the placeholder text "Update dependencies". Underneath is a section labeled "Associate this commit with:" followed by a dropdown menu showing the email `octocat@github.com`. At the bottom, there are two buttons: "Confirm merge" (in green) and "Cancel" (in red).

14. Click **Confirm merge**.
15. Optionally, delete the branch. For more information, see "[Creating and deleting branches within your repository](#)."

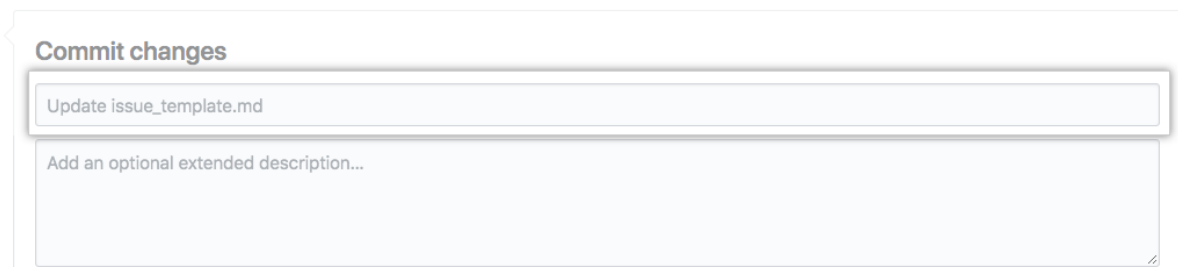
Adding a new post to your site

1. On GitHub, navigate to your site's repository.

2. Navigate to the publishing source for your site. For more information about publishing sources, see "[About GitHub Pages](#)."
3. Navigate to the `_posts` directory.
4. Create a new file called `YYYY-MM-DD-NAME-OF-POST.md`, replacing `YYYY-MM-DD` with the date of your post and `NAME-OF-POST` with the name of your post.
5. Add the following YAML frontmatter to the top of the file, replacing `POST TITLE` with the post's title, `YYYY-MM-DD hh:mm:ss -0000` with the date and time for the post, and `CATEGORY-1` and `CATEGORY-2` with as many categories you want for your post.

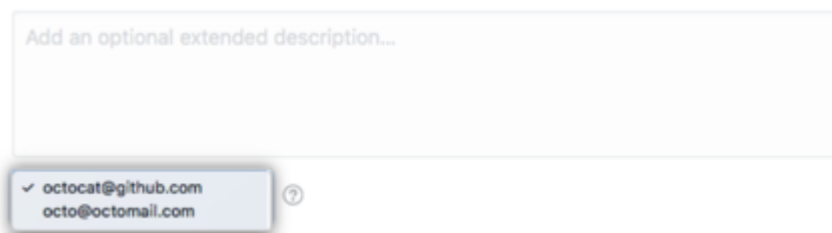
```
layout: post
title: "POST TITLE"
date: YYYY-MM-DD hh:mm:ss -0000
categories: CATEGORY-1 CATEGORY-2
```

6. Below the frontmatter, add content for your post.
7. At the bottom of the page, type a short, meaningful commit message that describes the change you made to the file. You can attribute the commit to more than one author in the commit message. For more information, see "[Creating a commit with multiple co-authors](#)."



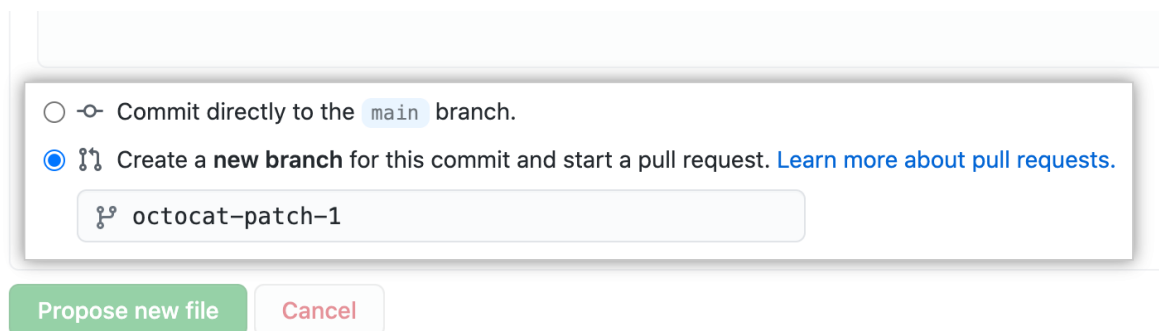
The screenshot shows the GitHub commit interface. At the top, it says "Commit changes". Below that, there is a text input field containing "Update issue_template.md". Underneath the input field is a larger text area with the placeholder text "Add an optional extended description...".

8. If you have more than one email address associated with your GitHub account, click the email address drop-down menu and select the email address to use as the Git author email address. Only verified email addresses appear in this drop-down menu. If you enabled email address privacy, then `<username>@users.noreply.github.com` is the default commit author email address. For more information, see "[Setting your commit email address](#)."



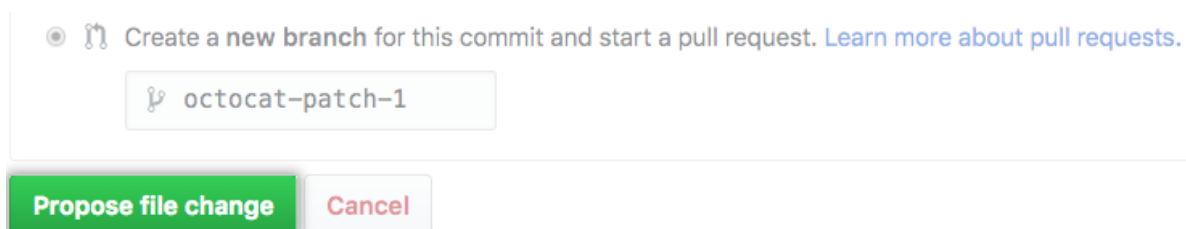
The screenshot shows the email address selection dropdown menu. It is positioned below the "Add an optional extended description..." text area. The dropdown menu is open, showing two email addresses: "octocat@github.com" (which is selected and has a checkmark) and "octo@octomail.com". A question mark icon is visible to the right of the dropdown menu.

9. Below the commit message fields, decide whether to add your commit to the current branch or to a new branch. If your current branch is the default branch, you should choose to create a new branch for your commit and then create a pull request. For more information, see "[Creating a new pull request](#)."



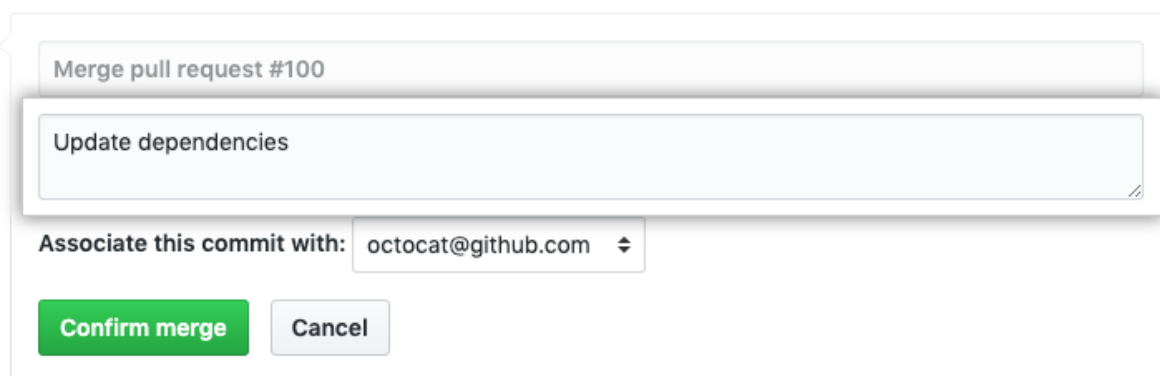
The screenshot shows a GitHub commit dialog. At the top, there's a text input field for the commit message. Below it, there are two radio button options: "Commit directly to the `main` branch." (which is unselected) and "Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)" (which is selected). Below the second option, there is a text input field containing the branch name `octocat-patch-1`. At the bottom, there are two buttons: "Propose new file" (in green) and "Cancel" (in light gray).

10. Click **Propose file change**.



This screenshot is similar to the previous one, showing the same GitHub commit dialog. The "Create a new branch" option is still selected, and the branch name `octocat-patch-1` is in the input field. In this version, the "Propose file change" button is highlighted with a green border, indicating it should be clicked.

11. Create a pull request for your proposed changes.
12. In the "Pull Requests" list, click the pull request you would like to merge.
13. Click **Merge pull request**. For more information, see "[Merging a pull request](#)."
14. If prompted, type a commit message, or accept the default message.



The screenshot shows the "Merge pull request #100" dialog in GitHub. It has a title bar that says "Merge pull request #100". Below the title bar is a text input field with the placeholder text "Update dependencies". Underneath that, it says "Associate this commit with:" followed by a dropdown menu showing the email address `octocat@github.com`. At the bottom, there are two buttons: "Confirm merge" (in green) and "Cancel" (in light gray).

15. Click **Confirm merge**.
16. Optionally, delete the branch. For more information, see "[Creating and deleting branches within your repository](#)."

Your post should now be up on your site! If the base URL of your site is `https://octocat.github.io`, then your new post will be located at `https://octocat.github.io/YYYY/MM/DD/TITLE.html`.

Next steps

You can add a Jekyll theme to your GitHub Pages site to customize the look and feel of your site. For more information, see "[Adding a theme to your GitHub Pages site using Jekyll](#)."