Managing a custom domain for your GitHub Pages site

In this article

You can set up or update certain DNS records and your repository settings to point the default domain for your GitHub Pages site to a custom domain.

Mac Windows Linux

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 plans."

People with admin permissions for a repository can configure a custom domain for a GitHub Pages site.

About custom domain configuration

Make sure you add your custom domain to your GitHub Pages site before configuring your custom domain with your DNS provider. Configuring your custom domain with your DNS provider without adding your custom domain to GitHub Enterprise Cloud could result in someone else being able to host a site on one of your subdomains.

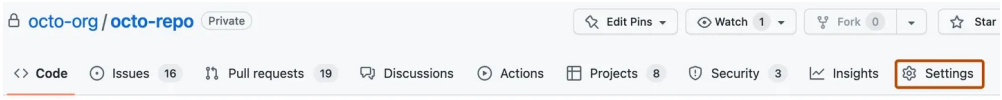
The `dig` command, which can be used to verify correct configuration of DNS records, is not included in Windows. Before you can verify that your DNS records are configured correctly, you must install BIND.

Note: DNS changes can take up to 24 hours to propagate.

Configuring a subdomain

To set up a `www` or custom subdomain, such as `www.example.com` or `blog.example.com`, you must add your domain in the repository settings. After that, configure a CNAME record with your DNS provider.

1. On GitHub Enterprise Cloud, navigate to your site's repository.
2. Under your repository name, click **Settings**. If you cannot see the "Settings" tab, select the ··· dropdown menu, then click **Settings**.



3. In the "Code and automation" section of the sidebar, click **Pages**.
4. Under "Custom domain", type your custom domain, then click **Save**. If you are publishing your site from a branch, this will create a commit that adds a `CNAME` file directly to the root of your source branch. If you are publishing your site with a custom GitHub Actions workflow, no `CNAME` file is created. For more information about your publishing source, see "Configuring a publishing source for your GitHub Pages site."

   > **Note:** If your custom domain is an internationalized domain name, you must enter the Punycode encoded version.
   >
   > For more information on Punycodes, see Internationalized domain name.

5. Navigate to your DNS provider and create a `CNAME` record that points your subdomain to the default domain for your site. For example, if you want to use the subdomain `www.example.com` for your user site, create a `CNAME` record that points `www.example.com` to `<user>.github.io`. If you want to use the subdomain `another.example.com` for your organization site, create a `CNAME` record that points `another.example.com` to `<organization>.github.io`. The `CNAME` record should always point to `<user>.github.io` or `<organization>.github.io`, excluding the repository name. For more information about how to create the correct record, see your DNS provider's documentation. For more information about the default domain for your site, see "About GitHub Pages."

   > **Warning:** We strongly recommend that you do not use wildcard DNS records, such as `*.example.com`. These records put you at an immediate risk of domain takeovers, even if you verify the domain. For example, if you verify `example.com` this prevents someone from using `a.example.com` but they could still take over `b.a.example.com` (which is covered by the wildcard DNS record). For more information, see "Verifying your custom domain for GitHub Pages."

6. Open TerminalTerminalGit Bash.
7. To confirm that your DNS record configured correctly, use the `dig` command, replacing *WWW.EXAMPLE.COM* with your subdomain.

```
$ dig WWW.EXAMPLE.COM +nostats +nocomments +nocmd
> ;WWW.EXAMPLE.COM.                    IN      A
> WWW.EXAMPLE.COM.             3592    IN      CNAME   YOUR-
USERNAME.github.io.
> YOUR-USERNAME.github.io.      43192   IN      CNAME   GITHUB-PAGES-SERVER
.
> GITHUB-PAGES-SERVER .         22      IN      A       192.0.2.1
```

8. If you use a static site generator to build your site locally and push the generated files to GitHub Enterprise Cloud, pull the commit that added the CNAME file to your local repository. For more information, see "Troubleshooting custom domains and

GitHub Pages."

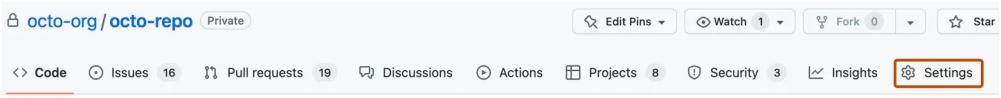
9. Optionally, to enforce HTTPS encryption for your site, select **Enforce HTTPS**. It can take up to 24 hours before this option is available. For more information, see "Securing your GitHub Pages site with HTTPS."

Configuring an apex domain

To set up an apex domain, such as `example.com`, you must configure a custom domain in your repository settings and at least one `ALIAS`, `ANAME`, or `A` record with your DNS provider.

If you are using an apex domain as your custom domain, we recommend also setting up a `www` subdomain. If you configure the correct records for each domain type through your DNS provider, GitHub Pages will automatically create redirects between the domains. For example, if you configure `www.example.com` as the custom domain for your site, and you have GitHub Pages DNS records set up for the apex and `www` domains, then `example.com` will redirect to `www.example.com`. Note that automatic redirects only apply to the `www` subdomain. Automatic redirects do not apply to any other subdomains, such as `blog`. For more information, see "Configuring a subdomain."

1. On GitHub Enterprise Cloud, navigate to your site's repository.
2. Under your repository name, click **Settings**. If you cannot see the "Settings" tab, select the ••• dropdown menu, then click **Settings**.



3. In the "Code and automation" section of the sidebar, click **Pages**.
4. Under "Custom domain", type your custom domain, then click **Save**. If you are publishing your site from a branch, this will create a commit that adds a `CNAME` file directly to the root of your source branch. If you are publishing your site with a custom GitHub Actions workflow, no `CNAME` file is created. For more information about your publishing source, see "Configuring a publishing source for your GitHub Pages site."
5. Navigate to your DNS provider and create either an `ALIAS`, `ANAME`, or `A` record. You can also create `AAAA` records for IPv6 support. If you're implementing IPv6 support, we highly recommend using an `A` record in addition to your `AAAA` record, due to slow adoption of IPv6 globally. For more information about how to create the correct record, see your DNS provider's documentation.
   - To create an `ALIAS` or `ANAME` record, point your apex domain to the default domain for your site. For more information about the default domain for your site, see "About GitHub Pages."
   - To create `A` records, point your apex domain to the IP addresses for GitHub Pages.

     ```
     185.199.108.153
     185.199.109.153
     185.199.110.153
     185.199.111.153
     ```

   - To create `AAAA` records, point your apex domain to the IP addresses for GitHub Pages.

```
2606:50c0:8000::153
2606:50c0:8001::153
2606:50c0:8002::153
2606:50c0:8003::153
```

Warning: We strongly recommend that you do not use wildcard DNS records, such as `*.example.com`. These records put you at an immediate risk of domain takeovers, even if you verify the domain. For example, if you verify `example.com` this prevents someone from using `a.example.com` but they could still take over `b.a.example.com` (which is covered by the wildcard DNS record). For more information, see "Verifying your custom domain for GitHub Pages."

6. Open TerminalTerminalGit Bash.
7. To confirm that your DNS record configured correctly, use the `dig` command, replacing *EXAMPLE.COM* with your apex domain. Confirm that the results match the IP addresses for GitHub Pages above.
   - For `A` records:

     ```
     $ dig EXAMPLE.COM +noall +answer -t A
     > EXAMPLE.COM    3600    IN A     185.199.108.153
     > EXAMPLE.COM    3600    IN A     185.199.109.153
     > EXAMPLE.COM    3600    IN A     185.199.110.153
     > EXAMPLE.COM    3600    IN A     185.199.111.153
     ```

   - For `AAAA` records:

     ```
     $ dig EXAMPLE.COM +noall +answer -t AAAA
     > EXAMPLE.COM     3600    IN AAAA     2606:50c0:8000::153
     > EXAMPLE.COM     3600    IN AAAA     2606:50c0:8001::153
     > EXAMPLE.COM     3600    IN AAAA     2606:50c0:8002::153
     > EXAMPLE.COM     3600    IN AAAA     2606:50c0:8003::153
     ```

     Remember to also check your `A` record.

8. If you use a static site generator to build your site locally and push the generated files to GitHub Enterprise Cloud, pull the commit that added the CNAME file to your local repository. For more information, see "Troubleshooting custom domains and GitHub Pages."
9. Optionally, to enforce HTTPS encryption for your site, select **Enforce HTTPS**. It can take up to 24 hours before this option is available. For more information, see "Securing your GitHub Pages site with HTTPS."

Configuring an apex domain and the `www` subdomain variant

When using an apex domain, we recommend configuring your GitHub Pages site to host content at both the apex domain and that domain's `www` subdomain variant.

To set up a `www` subdomain alongside the apex domain, you must first configure an apex domain by creating an `ALIAS`, `ANAME`, or `A` record with your DNS provider. For more information, see "Configuring an apex domain."

After you configure the apex domain, you must configure a CNAME record with your DNS provider.

1. Navigate to your DNS provider and create a `CNAME` record that points `www.example.com` to the default domain for your site: `<user>.github.io` or `<organization>.github.io`. Do not include the repository name. For more information about how to create the correct record, see your DNS provider's documentation. For more information about the default domain for your site, see "About GitHub Pages."

2. To confirm that your DNS record configured correctly, use the `dig` command, replacing *WWW.EXAMPLE.COM* with your `www` subdomain variant.

```
$ dig WWW.EXAMPLE.COM +nostats +nocomments +nocmd
> ;WWW.EXAMPLE.COM                    IN      A
> WWW.EXAMPLE.COM.              3592    IN      CNAME   YOUR-
USERNAME.github.io.
> YOUR-USERNAME.github.io.      43192   IN      CNAME   GITHUB-PAGES-SERVER.
> GITHUB-PAGES-SERVER.          22      IN      A       192.0.2.1
```

DNS records for your custom domain

If you are familiar with the process of configuring your domain for a GitHub Pages site, you can use the table below to find the DNS values for your specific scenario and the DNS record types that your DNS provider supports. For more information, including how to configure your GitHub Pages site on GitHub Enterprise Cloud and how to verify the configuration using the `dig` command, refer to the sections above.

To configure an apex domain, you only need to pick a single DNS record type from the table below. To configure an apex domain and `www` subdomain (for example, `example.com` and `www.example.com`), configure the apex domain and then the subdomain. For more information, see "Configuring an apex domain and the `www` subdomain variant."

Warning: We strongly recommend that you do not use wildcard DNS records, such as `*.example.com`. These records put you at an immediate risk of domain takeovers, even if you verify the domain. For example, if you verify `example.com` this prevents someone from using `a.example.com` but they could still take over `b.a.example.com` (which is covered by the wildcard DNS record). For more information, see "Verifying your custom domain for GitHub Pages."

Scenario	DNS record type	DNS record name	DNS record value(s)

Apex domain (`example.com`)	`A`	`@`	`185.199.108.153`
			`185.199.109.153`
			`185.199.110.153`
			`185.199.111.153`

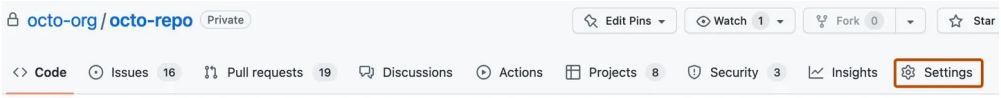
Apex domain (`example.com`)	`AAAA`	`@`	`2606:50c0:8000::153`
			`2606:50c0:8001::153`
			`2606:50c0:8002::153`
			`2606:50c0:8003::153`

Apex domain (`example.com`)	`ALIAS` or `ANAME`	`@`	`USERNAME.github.io` or `ORGANIZATION.github.io`
Subdomain (`www.example.com`, `blog.example.com`)	`CNAME`	`SUBDOMAIN.example.com.`	`USERNAME.github.io` or `ORGANIZATION.github.io`

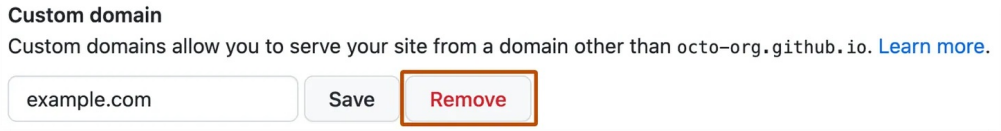
Removing a custom domain

If you get an error about a custom domain being taken, you may need to remove the custom domain from another repository.

1. On GitHub Enterprise Cloud, navigate to your site's repository.
2. Under your repository name, click **Settings**. If you cannot see the "Settings" tab, select the ··· dropdown menu, then click **Settings**.



3. In the "Code and automation" section of the sidebar, click **Pages**.
4. Under "Custom domain," click **Remove**.



Securing your custom domain

If your GitHub Pages site is disabled but has a custom domain set up, it is at risk of a domain takeover. Having a custom domain configured with your DNS provider while your site is disabled could result in someone else hosting a site on one of your subdomains.

Verifying your custom domain prevents other GitHub users from using your domain with their repositories. If your domain is not verified, and your GitHub Pages site is disabled, you should immediately update or remove your DNS records with your DNS provider. For more information, see "Verifying your custom domain for GitHub Pages."

Further reading

- "Troubleshooting custom domains and GitHub Pages"