# Contribute to the ASP.NET Core documentation

This document covers the process for contributing to the articles and code samples that are hosted on the [ASP.NET documentation site](https://docs.microsoft.com/aspnet/). Typo corrections and new articles are welcome contributions.

## How to make a simple correction or suggestion

Articles are stored in the repository as Markdown files. Simple changes to the content of a Markdown file are made in the browser by selecting the \*\*Edit\*\* link in the upper-right corner of the browser window. (In a narrow browser window, expand the \*\*options\*\* bar to see the \*\*Edit\*\* link.) Follow the directions to create a pull request (PR). We will review the PR and accept it or suggest changes.

## How to make a more complex submission

You need a basic understanding of [Git and GitHub.com](https://guides.github.com/activities/hello-world/).

\* Open an [issue](https://github.com/dotnet/AspNetCore.Docs/issues/new) describing what you want to do, such as changing an existing article or creating a new one. We often request an outline for a new topic suggestion. Wait for approval from the team before you invest much time.

\* Fork the [aspnet/Docs](https://github.com/dotnet/AspNetCore.Docs/) repo and create a branch for your changes.

\* Submit a PR to master with your changes.

\* If your PR has the label 'cla-required' assigned, [complete the Contribution License Agreement (CLA)](https://cla.dotnetfoundation.org/).

\* Respond to PR feedback.

For an example where this process led to publication of a new article, see [Issue &num;67](https://github.com/dotnet/docs/issues/67) and [Pull Request &num;798](https://github.com/dotnet/docs/pull/798) in the .NET Docs repository. The new article is [Documenting your code](https://docs.microsoft.com/dotnet/articles/csharp/codedoc).

## Docs Authoring Pack extension in Visual Studio Code

If you use Visual Studio Code to contribute to the ASP.NET documentation, you can boost your productivity by installing the [Docs Authoring Pack](https://marketplace.visualstudio.com/items?itemName=docsmsft.docs-authoring-pack) extension. The extension provides a variety of tools that helps with Markdown linting, code spell checking, and article templates.

## Markdown syntax

Articles are written in [DocFx-flavored Markdown](https://dotnet.github.io/docfx/spec/docfx\_flavored\_markdown.html), which is a superset of [GitHub-flavored Markdown (GFM)](https://guides.github.com/features/mastering-markdown/). For examples of DFM syntax for UI features commonly used in the ASP.NET documentation, see [Metadata and Markdown Template](https://github.com/dotnet/docs/blob/master/styleguide/template.md) in the .NET Docs repo style guide.

## Folder structure conventions

For each Markdown file, a folder for images and a folder for sample code may exist. If the article is [fundamentals/configuration/index.md](https://github.com/dotnet/AspNetCore.Docs/blob/master/aspnetcore/fundamentals/configuration/index.md), the images are in [fundamentals/configuration/index/\\_static](https://github.com/dotnet/AspNetCore.Docs/tree/master/aspnetcore/fundamentals/configuration/index/\_static) and the sample app project files are in [fundamentals/configuration/index/sample](https://github.com/dotnet/AspNetCore.Docs/tree/master/aspnetcore/fundamentals/configuration/index/sample). An image in the \*fundamentals/configuration/index.md\* file is rendered by the following Markdown:

```md

![description of image for alt attribute](configuration/index/\_static/imagename.png)

```

All images should have [alternative (alt) text](https://wikipedia.org/wiki/Alt\_attribute). For advice on specifying alt text, see online resources, such as [WebAIM: Alternative Text](https://webaim.org/techniques/alttext/).

Use lowercase for Markdown file names and image file names.

## Internal links

Internal links should use the `uid` of the target article with an xref link (link text is set to the linked content's title):

```md

<xref:uid\_of\_the\_topic>

```

If the title of the article is unsuitable for link text (for example, a word or phrase in a sentence is the link text), specify the xref link and link text with the following:

```md

[link text](xref:uid\_of\_the\_topic)

```

For more information, see the [DocFX Cross Reference](https://dotnet.github.io/docfx/spec/docfx\_flavored\_markdown.html#cross-reference).

## Images and screenshots

Don't include images with articles, except:

\* In basic onboarding (beginner) tutorials.

\* When an image is needed for clarity.

These restrictions reduce the size of the repository.

As an optional step, ensure that any images and screenshots used in the documentation are compressed, which helps with file size and page load performance. A few popular tools include TinyPNG (using the [TinyPNG website](https://tinypng.com/) or the [TinyPNG API](https://tinypng.com/developers)) or the [Image Optimizer](https://marketplace.visualstudio.com/items?itemName=MadsKristensen.ImageOptimizer) Visual Studio extension.

## Code snippets

Articles frequently contain code snippets to illustrate points. DFM allows you to copy code into the Markdown file or refer to a separate code file. We prefer to use separate code files whenever possible to minimize the chance of errors in the code. The code files are stored in the repo using the folder structure described earlier for sample projects.

The following examples illustrate [DFM code snippet syntax](https://dotnet.github.io/docfx/spec/docfx\_flavored\_markdown.html#code-snippet) for use in a \*configuration/index.md\* file.

To render an entire code file as a snippet:

```md

[!code-csharp[](configuration/index/sample/Program.cs)]

```

To render a portion of a file as a snippet by using line numbers:

```md

[!code-csharp[](configuration/index/sample/Program.cs?range=1-10,20,30,40-50]

[!code-html[](configuration/index/sample/Views/Home/Index.cshtml?range=1-10,20,30,40-50]

```

For C# snippets, reference a [C# region](https://docs.microsoft.com/dotnet/csharp/language-reference/preprocessor-directives/preprocessor-region). Whenever possible, use regions rather than line numbers because line numbers in a code file tend to change and become out of sync with line number references in Markdown. C# regions can be nested. If referencing the outer region, the inner `#region` and `#endregion` directives aren't rendered in a snippet.

To render a C# region named "snippet\_Example":

```md

[!code-csharp[](configuration/index/sample/Program.cs?name=snippet\_Example)]

```

To highlight selected lines in a rendered snippet (usually renders as yellow background color):

```md

[!code-csharp[](configuration/index/sample/Program.cs?name=snippet\_Example&highlight=1-3,10,20-25)]

[!code-csharp[](configuration/index/sample/Program.cs?range=10-20&highlight=1-3]

[!code-html[](configuration/index/sample/Views/Home/Index.cshtml?range=10-20&highlight=1-3]

[!code-javascript[](configuration/index/sample/UsingOptionsSample.csproj?range=10-20&highlight=1-3]

```

## Test changes with DocFX

Test your changes with the [DocFX command-line tool](https://dotnet.github.io/docfx/tutorial/docfx\_getting\_started.html#2-use-docfx-as-a-command-line-tool), which creates a locally hosted version of the site. DocFX doesn't render style and site extensions created for docs.microsoft.com.

DocFX requires:

\* .NET Framework on Windows.

\* Mono for Linux or macOS.

### Windows instructions

\* Download and unzip \*docfx.zip\* from [DocFX releases](https://github.com/dotnet/docfx/releases).

\* Add DocFX to your PATH.

\* In a command shell, navigate to the folder that contains the \*docfx.json\* file (\*aspnet\* for ASP.NET content or \*aspnetcore\* for ASP.NET Core content) and run the following command:

```console

docfx --serve

```

\* In a browser, navigate to `http://localhost:8080/group1-dest/`.

### Mono instructions

\* Install Mono via Homebrew:

```console

brew install mono

```

\* Download the [latest version of DocFX](https://github.com/dotnet/docfx/releases).

\* Extract the archive to \*$HOME/bin/docfx\*.

\* Create a pair of aliases for \*\*docfx\*\* in a bash shell. The first alias is used to build the documentation. The second alias is used to build and serve the documentation.

```console

alias docfx='mono $HOME/bin/docfx/docfx.exe'

alias docfx-serve='mono $HOME/bin/docfx/docfx.exe --serve'

```

\* In a command shell, navigate to the folder that contains the \*docfx.json\* file (\*aspnet\* for ASP.NET content or \*aspnetcore\* for ASP.NET Core content) and run the following command to build and serve the docs via its alias:

```console

docfx-serve

```

\* In a browser, navigate to `http://localhost:8080/group1-dest/`.

## Voice and tone

Our goal is to write documentation that is easily understandable by the widest possible audience. To that end, we established guidelines for writing style that we ask our contributors to follow. For more information, see [Voice and tone guidelines](https://github.com/dotnet/docs/blob/master/styleguide/voice-tone.md) in the .NET repo.

## Microsoft Writing Style Guide

The [Microsoft Writing Style Guide](https://docs.microsoft.com/style-guide/welcome/) provides writing style and terminology guidance for all forms of technology communication, including the ASP.NET Core documentation.

## Redirects

If you delete an article, change its file name, or move it to a different folder, create a redirect so that people who bookmarked the article don't receive a \*404 Not Found\* error. Add redirects to the [master redirect file](https://github.com/dotnet/AspNetCore.Docs/blob/master/.openpublishing.redirection.json).