Nested Code Fences in Markdown

By **Susam Pal** on 19 Jan 2026

Today, we will meet a spiky-haired nerd named Corey Dumm, who normally lives within Markdown code fences. We will get to know him a bit, smile with him when his fences hold and weep quietly when misfortune strikes.

One of the caveats of the Markdown universe is the wide variety of Markdown implementations available. In these parallel universes, the rules of Markdown rendering differ subtly. In this post, we will focus only on the CommonMark specification. Since GitHub Flavoured Markdown (GFM) is a strict superset of CommonMark, whatever we discuss here applies equally well to both CommonMark and GFM.

Contents

Basic Code Fences

Corey had a knack for working with computers ever since he was a kid.

````
Corey at his computer:

```
(o_o)--.|[_]|
```
````

Everything was perfect in Corey's world. The CommonMark renderer would convert the Markdown above to the following HTML:

> Corey at his computer:

```
 (o_o)--.|[_]|
```

▶ View HTML

At this point, all was well. Corey grew quickly. Before long, he had a head full of spiky hair. Then the fences began to matter.

```
Corey, all grown up:

```
 ```
(o_o)--.|[_]|
```
```

Let us see how this renders. I must warn you that during the Markdown-to-HTML translation, Corey loses his hair. Some viewers may find the following scene disturbing. Viewer discretion is advised. Here is the rendered HTML:

Corey, all grown up:

(o_o)--.|[_]|

▶ View HTML

Corey's hair is gone! What a catastrophic accident! Corey is alright, though. He is still quite afraid of Markdown fences, but otherwise well and bouncing back. Why did this happen? The second set of triple backticks immediately ends the fenced code block started by the first set of triple backticks. As a result, Corey's smiley face ends up outside the fenced code block. The triple backticks that were once Corey's hair are now woven into the fabric of the surrounding HTML. Fortunately, CommonMark offers a few ways to avoid such accidents.

## Fancy Code Fences

In CommonMark, there are two main ways to include triple backticks within fenced code blocks. First, we can use tildes as the code fence:
```

```
Corey, all grown up:

~~~
 ```
(o_o)--.|[_]|
~~~
```

In fact, a code fence need not consist of exactly three backticks or tildes. Any number of backticks or tildes is allowed, as long as that number is at least three. The following is therefore equivalent:

```
Corey, all grown up:

~~~~~
 ```
(o_o)--.|[_]|
~~~~~
```

And so is this:

```
Corey, all grown up:

`````
 ```
(o_o)--.|[_]|
`````
```

All three examples render like this:

> Corey, all grown up:
>
> ```
>  ```
> (o_o)--.|[_]|
> ```

▶ View HTML

No hair is lost in translation.

Basic Code Spans
```

A similar problem arises with inline code spans. Most Markdown users know to use backticks to delimit inline code spans. For example:

```
An old picture of Corey at his computer: `(o_o)--.|[_]|`
```

This produces the following output:

> An old picture of Corey at his computer: `(o_o)--.|[_]|`

▶ View HTML

However, what do we do when we need to put Corey's dear friend Becky Trace within an inline code span? Becky has short, straight hair tucked neatly on either side of her face. Here's a picture of her:

```
`(o_o)`
```

I believe you can already see the difficulty here. Inline code spans use backticks as delimiters. So when we put Becky within a code span, the first backtick in Becky's face would terminate the code span immediately and then the rest of Becky would lie outside it. CommonMark offers solutions for this kind of situation as well.

Fancy Code Spans

An inline code span delimiter need not consist of exactly one backtick. It can consist of any number of backticks. So `` `foo` `` and ``` ``foo`` ``` produce identical HTML. There is another important but less well-known detail. When the text inside an inline code span begins and ends with spaces, one space is removed from each end before rendering. So `` `foo` `` and `` ` foo ` `` are equivalent. Therefore, when we need to put backticks within an inline code span, we can start the code span using multiple backticks and a space. For example:

```
Meet Corey's friend Becky Trace: `` `(o_o)` ``
```

Here is the rendered output:

> Meet Corey's friend Becky Trace: `` `(o_o)` ``

▶ View HTML

Becky has her hair intact too. We have avoided the mishap that once caused great distress to Corey. That, my friends, is how backticks survive nesting in Markdown.

Specification

Before I finish this post, let us take a look at the CommonMark specification to see where these details are defined. The excerpts quoted below are taken from CommonMark Spec Version 0.30, which is by now over four years old.

From section 4.5 Fenced Code Blocks:

> A code fence is a sequence of at least three consecutive backtick characters (`) or tildes (~). (Tildes and backticks cannot be mixed.)

> The content of the code block consists of all subsequent lines, until a closing code fence of the same type as the code block began with (backticks or tildes), and with at least as many backticks or tildes as the opening code fence.

From section 6.1 Code Spans:

> A backtick string is a string of one or more backtick characters (`) that is neither preceded nor followed by a backtick.
>
> A code span begins with a backtick string and ends with a backtick string of equal length. The contents of the code span are the characters between these two backtick strings, normalized in the following ways:
>
> - First, line endings are converted to spaces.
> - If the resulting string both begins *and* ends with a space character, but does not consist entirely of space characters, a single space character is removed from the front and back. This allows you to include code that begins or ends with backtick characters, which must be separated by whitespace from the opening or closing backtick strings.

I hope these little nuggets of Markdown trivialities will one day prove useful in your own Markdown misfortunes.

Comments | #web | #technology