# Using figures for more than images

Most examples of figures on the web are images with a caption. But that's only one use for the figure element.

In this post we'll look at four different types of figures with captions:

1. Images
2. Code listing
3. Ordered and bulleted listings
4. Tables

We will also look at CSS to create Counters and to place the caption in different locations around the content.

## Warning

The examples in this post use the :has pseudo-class. This is not supported in Firefox according to Caniuse.

Until the Mozilla crew provides support you will have to either not use the property or wrap the code in `@supports` blocks while providing an alternative implementation for Firefox

# What is a figure in HTML?

According to the HTML specification:

> The figure element represents some flow content, optionally with a caption, that is self-contained (like a complete sentence) and is typically referenced as a single unit from the main flow of the document.

A caption can be associated with the `figure` element by inserting a `figcaption`

as the first or the last child of the `figure` element. Browsers will render the first `figcaption` element found in the figure regardless of its location.

# So what can we use figures for?

The four items that we're creating figures for are usually those that would appear in a journal article.

## Images

The basic example is to use an image. This is a barebones example.

```
<`figure`
  <img src="path/to/images/my-image.png" alt="alternate text for image">
  <`figcaption`This is the image caption</`figcaption`
</`figure`
```

## Code Listings

The next example shows how to create figures with code examples. We use the same tags that we'd use when using a syntax highlighter library like Prism.js: `<pre class="language-javascript"><code>` and close them in the reverse order `</code></pre>`.

```
<`figure`
  <pre class="language-javascript"><code>window.CSS.registerProperty({
  name: `--color-${name}`,
  syntax: '<color>',
  inherits: true,
  initialValue: '`${cssColor}`',
}) </code></pre>
  <`figcaption`</`figcaption`
</`figure`
```

If you've installed Prism.js or another syntax highlighter on your page, this should highight the code in your figure but not the caption for the code listing.

# Ordered and bulleted lists

I'm ambivalent about using figures for lists of content. But if we're going to potentially group them together and provide captions, then I guess I'm OK with it.

```
<`figure`
  <ol>
    <li>Do this first</li>
    <li>Do this second</li>
    <li>Do this third</li>
    <li>Do this last</li>
  </ol>
  <`figcaption`List of tasks to do</`figcaption`
</`figure`
```

# Tables

I am not advocating using tables for layout. This is for adding captions to data tables.

We could also use the `caption` as a child of the `table` element itself but I'd rather keep all my captions the same.

Using figures will allow for easy styling with CSS.

```
<table>
  <thead>
    <tr>
      <th scope="col">Valid</th>
      <th scope="col">Invalid</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>"version": 2</td>
      <td>'version': 2</td>
    </tr>
```

```
  </tbody>
</table>
<`figcaption`Valid and invalid examples of JSON code</`figcaption`
</`figure`
```

# Styling figures and captions

We can create different types of figures and we can style them differently based on their type.

## Working with multiple kinds of figures

Each type of figure has its own custom configuration in addition to features we will use when generating text for the captions.

For the sake of brevity I will skip the styling code. You can see in the Codepen at the end of the post.

## Using counters and generated text

Adding numbering to figures is tedious. We could wait until we're done and then add `figure` or `table` or `listing` or whatever name we've assigned to the figure type plus the appropriate number manually... if we later decide to insert or remove new images, then we need to redo all the work.

Instead, we will use the counter function and the content property to annotate the captions without manually adding the type of figure and the number it occupies within the document.

I've chose to use `section` as the root for each chapter of content. It is there that we reset all the counters to make sure they all start from zero.

If we don't want to restart numbering for each separate `section`, we can reset the counters higher up in the cascade, possibly a `main` element or even body.

```
section {
  counter-reset: image-counter code-counter list-counter table-counter;
```

```
}
```

For images, we use `figure:has(> img)` and if the selector matches then we increment the `figure-image` counter.

Then we use the `::before` pseudo element to insert a string containing both text and the value of our counter.

```
figure:has(> img) {
  counter-increment: image-counter;
}

figure:has(img) figcaption::before {
  font-weight: 700;
  content: "Figure " counter(image-counter) ": ";
}
"Figure "
```

While this is convenient, it has a drawback: **The generated content we insert using `::before` is not added to the DOM of the page. This may cause the text not to be accessible to assistive technology tools and applications.**

When working with other figures, the only thing that change are the name of the counter and the css selector that we use.

| Type of table | CSS selector | Counter |
|---|---|---|
| Table | figure:has(> table) | table-counter |
| Code | figure:has(pre) | code-counter; |
| Listings | figure:has(ol)<br>figure:has(ul) | list-counter |

# Controlling the placement of captions

The placement of the caption does matter. If you place the caption before the content, the caption will appear above the content. If you place it below the content, the captions will appear below the content.

If we want to force the captions to be in a given location we can use classes to change the display of the caption to `display: table-caption` and set the location using `caption-side`.

The following example will force all the elements using the class to display the caption before the content regardless of where you place the `figcaption` element in the figure.

```
figure.caption-top {
  display: table;
}

figure.caption-top figcaption {
  display: table-caption;
  caption-side: top;
}
```

# The full set of examples