Object-fit and background-size

One very frustrating thing when working with images on the web is when you're trying to fit an image into a container.

If the image has different dimensions and aspect ratio than the container, the image will be resized to fit, loosing potentially important portions of the image or stretching it to the point it becomes blury and hard to find the details on it.

This post will discuss three attributes that may help solve this type of problems: `object-fit`, `background-size` and `aspect-ratio`.

Background

Before we start there are a coouple of background items to talk about.

Replaced Elements

In the context of CSS, replaced elements are elements contents are not affected by the current document's styles.

We can control the position of a replaced element (like using float to control the position of an image or video in relation to surrounding elements) but we can't control the content of the replaced element. However, you can style the fallback for replaced elements as shown in Styling broken Images by Ire Aderinokun

These are some common replaced elements:

- <iframe>
- <video>
- <embed>
- <img>
- <input type="image">

object-fit

The `object-fit` property sets how the content of a replaced element, such as an `<img>` or `<video>`, should be resized to fit its container.

This will address the problem of seeing a small part of the image when the image is too large to fit the container.

The `object-fit` property takes one of the following values:

`contain` : The replaced content is scaled to maintain its aspect ratio while fitting within the element's content box. The entire object is made to fill the box, while preserving its aspect ratio, so the object will be "letterboxed" if its aspect ratio does not match the aspect ratio of the box.

`cover` : The replaced content is sized to maintain its aspect ratio while filling the parent's entire content box. If the replaced content is larger than the parent's content box it will be clipped to fit. : Clipping the image may drop parts of the image from the viewport.

`fill` (**default**) : The replaced content is sized to completely fill the paren't element's content box. If the object's is smaller than the content box, then the object will be stretched to fit. : Stretching an image may cause it to pixelate and make it hard to distinguish details on the image.

none : The replaced content is not resized.

`scale-down` : The content is sized as if none or `contain` were specified, whichever would result in a smaller concrete object size.

background-size

The `background-size` property controls the size of the element's background image. It does the same thing for background images than `object-fit` does for replaced elements.

Spaces not covered by a background image are filled with the element's `background-color`, and the background color will be visible behind background images that have transparency/translucency.

The `background-size` property is specified in one of the following ways:

- Using the `contain` or `cover` keyword values
- Using a `width` value only, in which case the height defaults to `auto`
- Using both a `width` and a `height` value, in which case the first sets the width and the second sets the height
- Each value can be a `length`, a `percentage`, or `auto`

To specify the size of multiple background images, separate the value for each one with a comma.

The possible values for background-size are:

`contain` : Scales the image as large as possible within its container without cropping or stretching the image. If the container is larger than the image, this will result in image tiling, unless the background-repeat property is set to no-repeat.

`cover` : Scales the image as large as possible to fill the container, stretching the image if necessary. If the proportions of the image differ from the element, it is cropped either vertically or horizontally so that no empty space remains.

`auto` : Scales the background image in the corresponding direction such that its intrinsic proportions are maintained.

`<length>` : Stretches the image in the corresponding dimension to the specified length. Negative values are not allowed.

`<percentage>` : Stretches the image in the corresponding dimension to the specified percentage of the background positioning area. : The background positioning area is determined by the value of `background-origin` (by default, the padding box). However, if the background's `background-attachment` value is fixed, the positioning area is instead the entire viewport. Negative values are not allowed.

Check MDN's background-size for more information.

aspect-ratio

The `aspect-ratio` property sets a preferred aspect ratio for the box, which will be used in the calculation of auto sizes and some other layout functions.

The possible values are:

`auto` : Replaced elements with an intrinsic aspect ratio use that aspect ratio, otherwise the box has no preferred aspect ratio. Size calculations involving intrinsic aspect ratio always work with the content box dimensions.

`<ratio>` : The box's preferred aspect ratio is the specified ratio of width / height : If height and the preceding slash character are omitted, height defaults to 1 : Size calculations involving preferred aspect ratio work with the dimensions of the box specified by box-sizing

Giving replaced elements fixed dimensions with attributes

In the early days, we used to add explicit `width` and `height` attributes to images just to make sure browsers would render them correctly.

Over time we stopped adding the attributes and relied on CSS and the browser to size the image for us.

That's why I found it interesting to see this snippet discussed when talking about aspect ratio and browser's built-in stylesheets:

```
img, input[type="image"], video, embed, iframe, marquee, object, table {
  aspect-ratio: attr(width) / attr(height);
}
```

It appears we do need explicit `width` and `height` attributes to make sure that browsers will render the image with the correct aspect ratio.

Adding dimension attributes will also help better render the page and eliminate layout shift by telling the browser how much space to allocate for the image in the layout phase before it renders the image.

Barry Pollard goes into much more detail about this in Setting Height And Width On Images Is Important Again

Links

- A Deep Dive Into object-fit And background-size In CSS
- Let’s Learn About Aspect Ratio In CSS
- Setting Height And Width On Images Is Important Again
- Designing An Aspect Ratio Unit For CSS
- MDN
    - object-fit
    - background-size
    - aspect-ratio
    - Mapping the width and height attributes of media container elements to their aspect-ratio
- Replaced Elements
    - HTML replaced and void elements
    - Replaced elements