

# How to Resize Images in HTML

To resize an image in HTML, use the `width` and `height` attributes of the `<img>` tag. You can also use various CSS properties to resize images.

---

Here's an image at its original size:



You should be seeing this image at its original size, unless your device is narrow and has resized it.

Below is the same image after it's been resized using [HTML](#) (using the `width` and `height` attributes of the `<img>` tag):

---

```

```

```
<p>This image is resized using the
'width' and 'height' attributes of the
'img' tag.</p>
```



T

'h



You can play with the dimensions and click Run to see how it affects the end result.

## Resizing with CSS

It's usually better to use [CSS](#) to resize images in HTML. You can do this with the [height](#) and [width](#) properties.

```
<!DOCTYPE html>
<title>Example</title>
<style>
.cats {
  width: 200px;
  height: 121px;
}
</style>


<p>This image is resized using the CSS
'width' and 'height' properties.</p>
```

T  
ai

In my stylesheet, I create a [class](#) called `cats` and I then reference that in the `<img>` tag. By doing this, only images with `class="cats"` will set to this size. You can use whatever class

name you like. Just be sure that the `<img>` tag references the correct class.

Also, in this example I used `px` to indicate the image size in pixels. You can use any other valid `<length>`, `<percentage>`, or various other values as required.



## Resizing Considerations

Resizing images with HTML/CSS should only be done if necessary. It is usually better to use an image that is the correct size in the first place than to resize it with HTML. This is because resizing it with HTML doesn't reduce the file size — the full file still has to be downloaded before it can be resized.

Therefore, if possible, resize the image to the correct dimensions first using an image editor (such as [GIMP](#)) before uploading it to your website/blog.

## Resize an Image Dynamically

You can use the CSS `max-width` property to resize the image dynamically.

The following example displays quite a large image in a small viewport. To see it in different sized viewports click [Editor](#) and [Preview](#). You can also resize your window to see the image shrink and expand.

```
<!DOCTYPE html>
<title>Example</title>
<style>
.cats {
  max-width: 100%;
}
</style>


<p>This image is resized dynamically
using the CSS 'max-width' property.</p>
```



T  
C

By omitting any width/height declarations and **only** using `max-width: 100%;`, the image will be displayed at 100% of the size of its container, but no larger. If the image is larger than its container, the image will shrink to fit. However, if the image is smaller than its container, it will be displayed at its true size (i.e. it won't increase in size to fit the container).

This technique can be handy when designing responsively for the image/web page to be displayed across multiple sized devices.

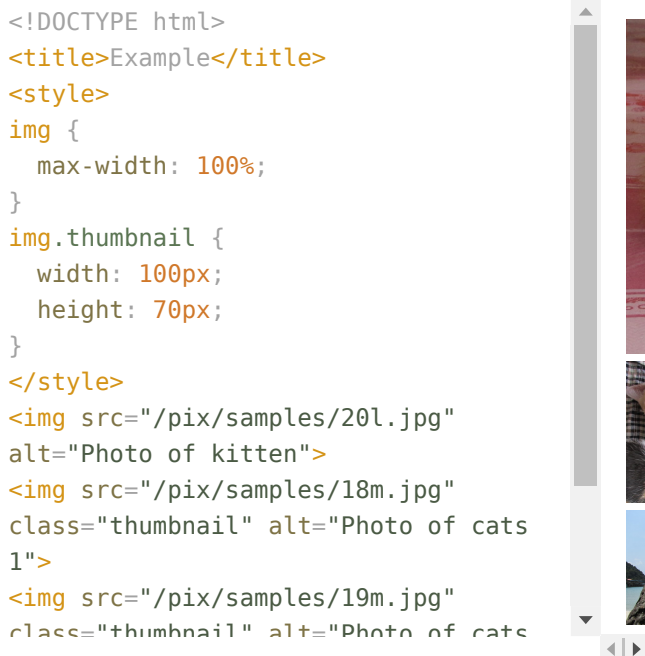
Read on for a more advanced method for responsively resizing images (using the `<picture>` element).

## Resize ALL Images

The previous CSS examples put the styles into a class, so that we can apply it selectively to certain images. You can also apply styles to *all* images if you like.

In fact, you can combine it, so that all images are styled a certain way, and then also be specific about certain images.

Here's an example of what that could look like:



This sets all `<img>` elements to a maximum width of 100%.

However, it also sets a class for thumbnails (called `thumbnail`) which explicitly sets their width and height. Therefore, images with that class will be sized using those explicit dimensions instead.

You might notice that I've prefixed the class name with `img.` in the stylesheet. What I mean is, I've used `img.thumbnail` in the CSS. By doing this, I'm specifying that only `<img>` elements that use the `thumbnail` class will have those styles applied. This prevents you from inadvertently applying the class to the wrong element in the event that another element uses a class of the same name.

For example, if a `<video>` element had `class="thumbnail"`, the above class won't be applied to it. You would need to use something like `video.thumbnail` in your stylesheet for video thumbnails.

However, this is entirely optional. Feel free to change `img.thumbnail` to just `.thumbnail` if you want the class

applied to all elements that use that class.

## Background Images

You can also resize background images using CSS. In particular, you can use the `background-size` property to resize background images.

Here's an example:



```
<!DOCTYPE html>
<title>Example</title>
<style>
div.bubbles {
  width: 80vw;
  height: 80vh;
  padding: 10px;
  background-image:
url(/pix/samples/bubble2.gif);
  border: 1px solid black;
  background-size: contain;
}
</style>
<div class="bubbles">
This element has a background image.
The CSS uses the 'background-size'
property to resize the background
image
```

In this example, I use the `contain` keyword to specify the size of the background image. You can also supply any `<length>` or `<percentage>` value, the `cover` keyword, or `auto` (which specifies that the image size is automatically determined using the intrinsic width and/or height of the image).

Further to this, all CSS properties are able to use the `initial`, `inherit`, and `unset` values.

When working with background images, you also have the option of using the `background` property as a shorthand to specify the background image URL, its size, the background color, etc, all in one go. I encourage you to look at that page, because there are several other things you can do with regards to resizing/cropping/repeating background images, etc.

## The `<picture>` Element and Responsive Design

One of the more recent additions to HTML is the `<picture>` element.

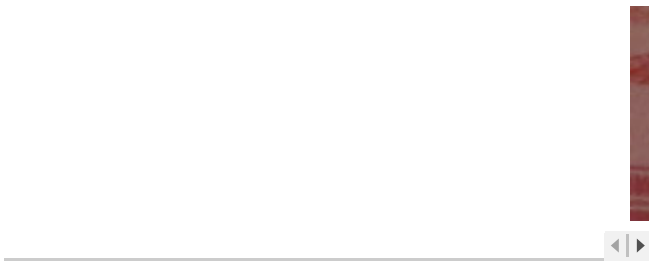
This element allows you to load a different image, depending on the user's screen pixel density, viewport size, image format, and other factors.

Here's an example:

```
<picture>
  <source srcset="/pix/samples/20l.jpg"
media="(min-width: 800px)">
  <source srcset="/pix/samples/20m.jpg"
media="(min-width: 600px)">
  
</picture>
```







Click the two orientation buttons at the top right of the editor to toggle the two images. Those two buttons change the orientation of the editor/preview pane, and therefore the width of the viewport.

Alternatively, try clicking the **Preview** button, then resizing your browser window to see the same effect.

Doing this should switch the image from a close-up of the kitten's face, to a larger version that includes the kitten's whole body.

I don't know what device/size screen you'll be viewing this on, so you may need to resize your screen or reorient your device to see the effect.

Alternatively you can adjust the code to use a more suitable width for your device (i.e. change `600px` to a different value).


The `<picture>` element contains multiple `<source>` elements to determine possible source images for the `<img>` element, depending on

factors such as screen pixel density, viewport size, image format, etc.

When using this element, you're not actually *resizing* the image as such.

What you're doing is selecting the appropriately sized image for a given situation.

The good thing about this method is that it allows you to do things like, display a cropped version of the image for smaller viewports, while displaying an uncropped version for larger ones.

 Quackit on Facebook

© Copyright 2000 - 2022 Quackit.com

---

 [Privacy and cookie settings](#)

---

Managed by Google. Complies with IAB TCF. CMP ID: 300