Headline with the h2 Element

Over the next few challenges, we'll build an HTML5 app that will look something like this:

The h2 element you enter will create an h2 element on the website.

This element tells the browser about the structure of your website. h1 elements are often used for main headings, while h2elements are generally used for subheadings. There are also h3, h4, h5and h6 elements to indicate different and new sections.

Add an h2 tag that says "CatPhotoApp" to create a second HTML element below your "Hello World" h1 element.

Inform with the Paragraph Element

p elements are the preferred element for normal-sized paragraph text on websites. P is short for "paragraph".

You can create a p element like this:

<p>I'm a p tag!</p>

Create a p element below your h2 element, and give it the text "Hello Paragraph".

Uncomment HTML

Commenting is a way that you can leave comments within your code without affecting the code itself.

Commenting is also a convenient way to make code inactive without having to delete it entirely.

You can start a comment with <!-- and end a comment with -->

Uncomment your h1, h2 and pelements.

Comment out HTML

Remember that in order to start a comment, you need to use <!-- and to end a comment, you need to use -->

Here you'll need to end the comment before your h2 element begins.

Comment out your h1 element and your p element, but leave your h2 element uncommented.

Fill in the Blank with Placeholder Text

Web developers traditionally use lorem ipsum text as placeholder text. The 'lorem ipsum' text is randomly scraped from a famous passage by Cicero of Ancient Rome.

Lorem ipsum text has been used as placeholder text by typesetters since the 16th century, and this tradition continues on the web.

Well, 5 centuries is long enough. Since we're building a CatPhotoApp, let's use something called kitty ipsum text.

Delete HTML Elements

Our phone doesn't have much vertical space.

Let's remove the unnecessary elements so we can start building our CatPhotoApp.

Delete your h1 element so we can simplify our view.

Change the Color of Text

Now let's change the color of some of our text.

We can do this by changing the style of your h2 element.

The style that is responsible for the color of an element's text is the "color" style.

Here's how you would set your h2 element's text color to blue:

<h2 style="color: blue">CatPhotoApp</h2>

Change your h2 element's style so that its text color is red.

Use CSS Selectors to Style Elements

With CSS, there are hundreds of CSS properties that you can use to change the way an element looks on your page.

When you entered <h2 style="color: red">CatPhotoApp</h2>, you were giving that individual h2 element an inline style.

That's one way to add style to an element, but a better way is by using CSS, which stands for Cascading Style Sheets.

At the top of your code, create a style element like this:

<style>  
</style>

Inside that style element, you can create a CSS selector for all h2 elements. For example, if you wanted all h2 elements to be red, your style element would look like this:

<style>  
  h2 {color: red;}  
</style>

Note that it's important to have both opening and closing curly braces ({ and }) around each element's style. You also need to make sure your element's style is between the opening and closing style tags. Finally, be sure to add the semicolon to the end of each of your element's styles.

Delete your h2 element's style attribute and instead create a CSS styleelement. Add the necessary CSS to turn all h2 elements blue.

Use a CSS Class to Style an Element

Classes are reusable styles that can be added to HTML elements.

Here's an example CSS class declaration:

<style>  
  .blue-text {  
    color: blue;  
  }  
</style>

You can see that we've created a CSS class called blue-text within the <style> tag.

You can apply a class to an HTML element like this:

<h2 class="blue-text">CatPhotoApp</h2>

Note that in your CSS style element, classes should start with a period. In your HTML elements' class declarations, classes shouldn't start with a period.

Inside your style element, change the h2 selector to .red-text and update the color's value from blue to red.

Give your h2 element the class attribute with a value of 'red-text'.

Style Multiple Elements with a CSS Class

Remember that you can attach classes to HTML elements by using class="your-class-here" within the relevant element's opening tag.

Remember that CSS class selectors require a period at the beginning like this:

.blue-text {  
  color: blue;  
}

But also remember that class declarations don't use a period, like this:

<h2 class="blue-text">CatPhotoApp</h2>

Apply the red-text class to your h2 and p elements.

Change the Font Size of an Element

Font size is controlled by the font-size CSS property, like this:

h1 {  
  font-size: 30px;  
}

Create a second p element after the existing p element with the following kitty ipsum text: Purr jump eat the grass rip the couch scratched sunbathe, shed everywhere rip the couch sleep in the sink fluffy fur catnip scratched.

Inside the same <style> tag that contains your red-text class, create an entry for p elements and set the font-size to 16 pixels (16px).

**Note**  
Due to browser implementation differences, you may need to be at 100% zoom to pass the tests on this challenge.

Also, please do not add a class attribute to your new p element.

Set the Font Family of an Element

You can set an element's font by using the font-family property.

For example, if you wanted to set your h2 element's font to Sans-serif, you would use the following CSS:

h2 {  
  font-family: Sans-serif;  
}

Make all of your p elements use the Monospace font.

Import a Google Font

Now, let's import and apply a Google font (note that if Google is blocked in your country, you will need to skip this challenge).

First, you'll need to make a call to Google to grab the Lobster font and load it into your HTML.

Copy the following code snippet and paste it into the top of your code editor:

<link href="https://fonts.googleapis.com/css?family=Lobster" rel="stylesheet" type="text/css">

Now you can set Lobster as a font-family value on your h2 element.

Apply the font-family of Lobster to your h2 element.

Specify How Fonts Should Degrade

There are several default fonts that are available in all browsers. These include Monospace, Serif and Sans-Serif

When one font isn't available, you can tell the browser to "degrade" to another font.

For example, if you wanted an element to use the Helvetica font, but also degrade to the Sans-Serif font when Helvetica wasn't available, you could use this CSS style:

p {  
  font-family: Helvetica, Sans-Serif;  
}

Now comment out your call to Google Fonts, so that the Lobster font isn't available. Notice how it degrades to the Monospace font.

Add Images to your Website

You can add images to your website by using the img element, and point to a specific image's URL using the src attribute.

An example of this would be:

<img src="https://www.your-image-source.com/your-image.jpg">

All img elements **must** have an alt attribute. The text inside an altattribute is used for screen readers to improve accessibility and is displayed if the image fails to load.

Lets add an alt attribute to our img example above:

<img src="https://www.your-image-source.com/your-image.jpg" alt="Author standing on a beach with two thumbs up. ">

Note that in most cases, img elements are self-closing.

Try it with this image:

https://bit.ly/fcc-relaxing-cat

Size your Images

CSS has a property called width that controls an element's width. Just like with fonts, we'll use px (pixels) to specify the image's width.

For example, if we wanted to create a CSS class called larger-image that gave HTML elements a width of 500 pixels, we'd use:

<style>  
  .larger-image {  
    width: 500px;  
  }  
</style>

Create a class called smaller-image and use it to resize the image so that it's only 100 pixels wide.

**Note**  
Due to browser implementation differences, you may need to be at 100% zoom to pass the tests on this challenge.

Add Borders Around your Elements

CSS borders have properties like style, color and width

For example, if we wanted to create a red, 5 pixel border around an HTML element, we could use this class:

<style>  
  .thin-red-border {  
    border-color: red;  
    border-width: 5px;  
    border-style: solid;  
  }  
</style>

Create a class called thick-green-border that puts a 10-pixel-wide green border with a style of solid around an HTML element, and apply that class to your cat photo.

Remember that you can apply multiple classes to an element by separating each class with a space within its class attribute. For example:

<img class="class1 class2">

Add Rounded Corners with a Border Radius

Your cat photo currently has sharp corners. We can round out those corners with a CSS property called border-radius.

You can specify a border-radius with pixels. Give your cat photo a border-radius of 10px.

Note: this waypoint allows for multiple possible solutions. For example, you may add border-radius to either the .thick-green-border class or .smaller-image class.

Make Circular Images with a Border Radius

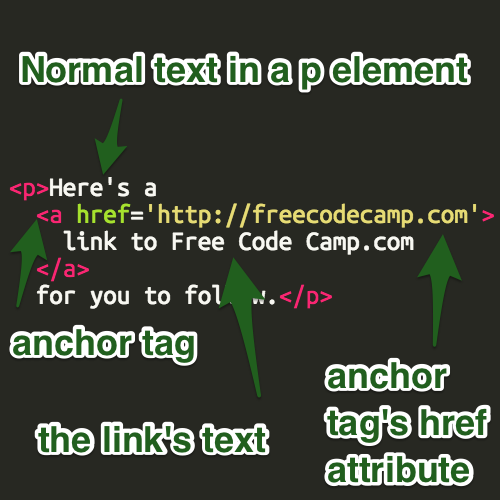
In addition to pixels, you can also specify a border-radius using a percentage.

Give your cat photo a border-radius of 50%.

Link to External Pages with Anchor Elements

a elements, also known as anchor elements, are used to link to content outside of the current page.

Here's a diagram of an a element. In this case, the a element is used in the middle of a paragraph element, which means the link will appear in the middle of a sentence.

**[](https://i.imgur.com/hviuZwe.png)**

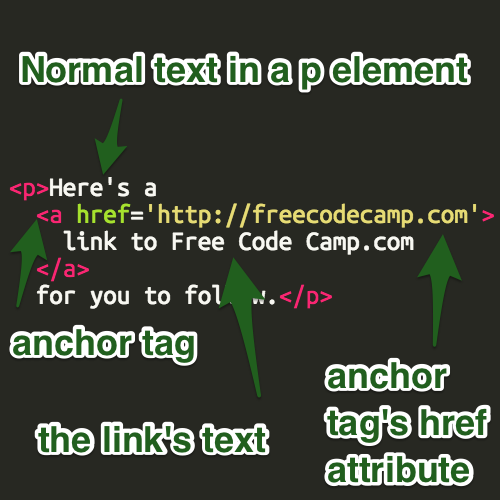
Here's an example:

<p>Here's a <a href="http://freecodecamp.com"> link to Free Code Camp</a> for you to follow.</p>

Create an a element that links to http://freecatphotoapp.com and has "cat photos" as its anchor text.

Nest an Anchor Element within a Paragraph

Again, here's a diagram of an a element for your reference:

**[](https://i.imgur.com/hviuZwe.png)**

Here's an example:

<p>Here's a <a href="https://freecodecamp.com"> link to Free Code Camp</a> for you to follow.</p>

Nesting just means putting one element inside of another element.

Now nest your existing a element within a new p element (just after the existing h2 element) so that the surrounding paragraph says "View more cat photos", but where only "cat photos" is a link, and the rest of the text is plain text.

Make Dead Links using the Hash Symbol

Sometimes you want to add a elements to your website before you know where they will link.

This is also handy when you're changing the behavior of a link using jQuery, which we'll learn about later.

Replace the value of your a element's href attribute with a #, also known as a hash symbol, to turn it into a dead link.

Turn an Image into a Link

You can make elements into links by nesting them within an a element.

Nest your image within an a element. Here's an example:

<a href="#"><img src="https://bit.ly/fcc-running-cats" alt="Three kittens running towards the camera. "></a>

Remember to use # as your a element's href property in order to turn it into a dead link.

Place the existing image element within an anchor element.

Once you've done this, hover over your image with your cursor. Your cursor's normal pointer should become the link clicking pointer. The photo is now a link.

Create a Bulleted Unordered List

HTML has a special element for creating unordered lists, or bullet point-style lists.

Unordered lists start with a <ul> element. Then they contain some number of <li> elements.

For example:

<ul>  
  <li>milk</li>  
  <li>cheese</li>  
</ul>

would create a bullet point-style list of "milk" and "cheese".

Remove the last two p elements and create an unordered list of three things that cats love at the bottom of the page.

Create an Ordered List

HTML has a special element for creating ordered lists, or numbered-style lists.

Ordered lists start with a <ol> element. Then they contain some number of <li> elements.

For example:

<ol>  
  <li>Garfield</li>  
  <li>Sylvester</li>  
</ol>

would create a numbered list of "Garfield" and "Sylvester".

Create an ordered list of the top 3 things cats hate the most.

Create a Text Field

Now let's create a web form.

Text inputs are a convenient way to get input from your user.

You can create one like this:

<input type="text">

Note that input elements are self-closing.

Create an input element of type text below your lists.

Add Placeholder Text to a Text Field

Your placeholder text is what appears in your text input before your user has input anything.

You can create placeholder text like so:

<input type="text" placeholder="this is placeholder text">

Set the placeholder value of your text input to "cat photo URL".

Create a Form Element

You can build web forms that actually submit data to a server using nothing more than pure HTML. You can do this by specifying an action on your form element.

For example:

<form action="/url-where-you-want-to-submit-form-data"></form>

Nest your text field in a form element. Add the action="/submit-cat-photo" attribute to this form element.

Add a Submit Button to a Form

Let's add a submit button to your form. Clicking this button will send the data from your form to the URL you specified with your form's action attribute.

Here's an example submit button:

<button type="submit">this button submits the form</button>

Add a submit button to your form element with type submit and "Submit" as its text.

Use HTML5 to Require a Field

You can require specific form fields so that your user will not be able to submit your form until he or she has filled them out.

For example, if you wanted to make a text input field required, you can just add the word required within your input element, you would use: <input type="text" required>

Make your text input a required field, so that your user can't submit the form without completing this field.

Then try to submit the form without inputing any text. See how your HTML5 form notifies you that the field is required?

Note: This field does not work in Safari.

Create a Set of Radio Buttons

You can use radio buttons for questions where you want the user to only give you one answer.

Radio buttons are a type of input

Each of your radio buttons should be nested within its own label element.

All related radio buttons should have the same name attribute.

Here's an example of a radio button:

<label><input type="radio" name="indoor-outdoor"> Indoor</label>

Add a pair of radio buttons to your form. One should have the option of indoor and the other should have the option of outdoor.

Create a Set of Checkboxes

Forms commonly use checkboxes for questions that may have more than one answer.

Checkboxes are a type of input

Each of your checkboxes should be nested within its own label element.

All related checkbox inputs should have the same name attribute.

Here's an example of a checkbox:

<label><input type="checkbox" name="personality"> Loving</label>

Add to your form a set of three checkboxes. Each checkbox should be nested within its own label element. All three should share the name attribute of personality.

Check Radio Buttons and Checkboxes by Default

You can set a checkbox or radio button to be checked by default using the checked attribute.

To do this, just add the word "checked" to the inside of an input element. For example:

<input type="radio" name="test-name" checked>

Set the first of your radio buttons and the first of your checkboxes to both be checked by default.

Nest Many Elements within a Single Div Element

The div element, also known as a division element, is a general purpose container for other elements.

The div element is probably the most commonly used HTML element of all. It's useful for passing the CSS of its own class declarations down to all the elements that it contains.

Just like any other non-self-closing element, you can open a div element with <div> and close it on another line with </div>.

Try putting your opening div tag above your "Things cats love" p element and your closing div tag after your closing ol tag so that both of your lists are within one div.

Nest your "Things cats love" and "Things cats hate" lists all within a single div element.

Give a Background Color to a Div Element

You can set an element's background color with the background-color property.

For example, if you wanted an element's background color to be green, you'd put this within your style element:

.green-background {  
  background-color: green;  
}

Create a class called silver-background with the background-color of silver. Assign this class to your divelement.

Set the ID of an Element

In addition to classes, each HTML element can also have an id attribute.

There are several benefits to using id attributes, and you'll learn more about them once you start using jQuery.

id attributes should be unique. Browsers won't enforce this, but it is a widely agreed upon best practice. So please don't give more than one element the same id attribute.

Here's an example of how you give your h2 element the id of cat-photo-app:

<h2 id="cat-photo-app">

Give your form element the id cat-photo-form.

Use an ID Attribute to Style an Element

One cool thing about id attributes is that, like classes, you can style them using CSS.

Here's an example of how you can take your element with the id attribute of cat-photo-element and give it the background color of green. In your style element:

#cat-photo-element {  
  background-color: green;  
}

Note that inside your style element, you always reference classes by putting a . in front of their names. You always reference ids by putting a # in front of their names.

Try giving your form, which now has the id attribute of cat-photo-form, a green background.

Adjusting the Padding of an Element

Now let's put our Cat Photo App away for a little while and learn more about styling HTML.

You may have already noticed this, but all HTML elements are essentially little rectangles.

Three important properties control the space that surrounds each HTML element: padding, margin, and border.

An element's padding controls the amount of space between the element and its border.

Here, we can see that the green box and the red box are nested within the yellow box. Note that the red box has more padding than the green box.

When you increase the green box's padding, it will increase the distance between the text padding and the border around it.

Change the padding of your green box to match that of your red box.

Adjust the Margin of an Element

An element's margin controls the amount of space between an element's border and surrounding elements.

Here, we can see that the green box and the red box are nested within the yellow box. Note that the red box has more margin than the green box, making it appear smaller.

When you increase the green box's margin, it will increase the distance between its border and surrounding elements.

Change the margin of the green box to match that of the red box.

Add a Negative Margin to an Element

An element's margin controls the amount of space between an element's border and surrounding elements.

If you set an element's margin to a negative value, the element will grow larger.

Try to set the margin to a negative value like the one for the red box.

Change the margin of the green box to -15px, so it fills the entire horizontal width of the yellow box around it.

Add Different Padding to Each Side of an Element

Sometimes you will want to customize an element so that it has different padding on each of its sides.

CSS allows you to control the padding of an element on all four sides with padding-top, padding-right, padding-bottom, and padding-left properties.

Give the green box a padding of 40px on its top and left side, but only 20px on its bottom and right side.

Add Different Margins to Each Side of an Element

Sometimes you will want to customize an element so that it has a different margin on each of its sides.

CSS allows you to control the margin of an element on all four sides with margin-top, margin-right, margin-bottom, and margin-left properties.

Give the green box a margin of 40px on its top and left side, but only 20px on its bottom and right side.

Use Clockwise Notation to Specify the Padding of an Element

Instead of specifying an element's padding-top, padding-right, padding-bottom, and padding-leftproperties, you can specify them all in one line, like this:

padding: 10px 20px 10px 20px;

These four values work like a clock: top, right, bottom, left, and will produce the exact same result as using the side-specific padding instructions.

Use Clockwise Notation to give the ".green-box" class a padding of 40px on its top and left side, but only 20px on its bottom and right side.

Use Clockwise Notation to Specify the Margin of an Element

Let's try this again, but with margin this time.

Instead of specifying an element's margin-top, margin-right, margin-bottom, and margin-left properties, you can specify them all in one line, like this:

margin: 10px 20px 10px 20px;

These four values work like a clock: top, right, bottom, left, and will produce the exact same result as using the side-specific margin instructions.

Use Clockwise Notation to give the element with the green-box class a margin of 40px on its top and left side, but only 20px on its bottom and right side.

Style the HTML Body Element

Now let's start fresh and talk about CSS inheritance.

Every HTML page has a body element.

We can prove that the body element exists here by giving it a background-color of black.

We can do this by adding the following to our style element:

body {  
  background-color: black;  
}

Use Clockwise Notation to Specify the Margin of an Element

Let's try this again, but with margin this time.

Instead of specifying an element's margin-top, margin-right, margin-bottom, and margin-left properties, you can specify them all in one line, like this:

margin: 10px 20px 10px 20px;

These four values work like a clock: top, right, bottom, left, and will produce the exact same result as using the side-specific margin instructions.

Use Clockwise Notation to give the element with the green-box class a margin of 40px on its top and left side, but only 20px on its bottom and right side.

Style the HTML Body Element

Now let's start fresh and talk about CSS inheritance.

Every HTML page has a body element.

We can prove that the body element exists here by giving it a background-color of black.

We can do this by adding the following to our style element:

body {  
  background-color: black;  
}

Inherit Styles from the Body Element

Now we've proven that every HTML page has a body element, and that its body element can also be styled with CSS.

Remember, you can style your body element just like any other HTML element, and all your other elements will inherit your body element's styles.

First, create a h1 element with the text Hello World

Then, let's give all elements on your page the color of green by adding color: green; to your body element's style declaration.

Finally, give your body element the font-family of Monospace by adding font-family: Monospace; to your body element's style declaration.

Prioritize One Style Over Another

Sometimes your HTML elements will receive multiple styles that conflict with one another.

For example, your h1 element can't be both green and pink at the same time.

Let's see what happens when we create a class that makes text pink, then apply it to an element. Will our class override the body element's color: green; CSS property?

Create a CSS class called pink-text that gives an element the color pink.

Give your h1 element the class of pink-text.

Override Styles in Subsequent CSS

Our "pink-text" class overrode our body element's CSS declaration!

We just proved that our classes will override the body element's CSS. So the next logical question is, what can we do to override our pink-text class?

Create an additional CSS class called blue-text that gives an element the color blue. Make sure it's below your pink-text class declaration.

Apply the blue-text class to your h1 element in addition to your pink-text class, and let's see which one wins.

Applying multiple class attributes to a HTML element is done with a space between them like this:

class="class1 class2"

Note: It doesn't matter which order the classes are listed in the HTML element.

However, the order of the class declarations in the <style> section are what is important. The second declaration will always take precedence over the first. Because .blue-text is declared second, it overrides the attributes of .pink-text

Override Class Declarations by Styling ID Attributes

We just proved that browsers read CSS from top to bottom. That means that, in the event of a conflict, the browser will use whichever CSS declaration came last.

But we're not done yet. There are other ways that you can override CSS. Do you remember id attributes?

Let's override your pink-text and blue-text classes, and make your h1 element orange, by giving the h1element an id and then styling that id.

Give your h1 element the id attribute of orange-text. Remember, id styles look like this:

<h1 id="orange-text">

Leave the blue-text and pink-text classes on your h1 element.

Create a CSS declaration for your orange-text id in your style element. Here's an example of what this looks like:

#brown-text {  
  color: brown;  
}

Note: It doesn't matter whether you declare this css above or below pink-text class, since id attribute will always take precedence.

Override Class Declarations with Inline Styles

So we've proven that id declarations override class declarations, regardless of where they are declared in your styleelement CSS.

There are other ways that you can override CSS. Do you remember inline styles?

Use an in-line style to try to make our h1 element white. Remember, in line styles look like this:

<h1 style="color: green">

Leave the blue-text and pink-text classes on your h1 element.

Override All Other Styles by using Important

Yay! We just proved that in-line styles will override all the CSS declarations in your style element.

But wait. There's one last way to override CSS. This is the most powerful method of all. But before we do it, let's talk about why you would ever want to override CSS.

In many situations, you will use CSS libraries. These may accidentally override your own CSS. So when you absolutely need to be sure that an element has specific CSS, you can use !important

Let's go all the way back to our pink-text class declaration. Remember that our pink-text class was overridden by subsequent class declarations, id declarations, and in-line styles.

Let's add the keyword !important to your pink-text element's color declaration to make 100% sure that your h1element will be pink.

An example of how to do this is:

color: red !important;

Use Hex Code for Specific Colors

Did you know there are other ways to represent colors in CSS? One of these ways is called hexadecimal code, or hex code for short.

We usually use decimals, or base 10 numbers, which use the symbols 0 to 9 for each digit. Hexadecimals (or hex) are base 16 numbers. This means it uses sixteen distinct symbols. Like decimals, the symbols 0-9 represents values zero to nine. Then A,B,C,D,E,F represent values ten to fifteen. Altogether, 0 to F can represent a digit in hexadecimal, giving us 16 total possible values. You can find more information about **[hexadecimal numbers here](https://en.wikipedia.org/wiki/Hexadecimal" \t "_blank)**.

In CSS, we can use 6 hexadecimal digits to represent colors, two each for the red (R), green (G), and blue (B) components. For example, #000000 is black and is also the lowest possible value. You can find more information about the **[RGB color system here](https://en.wikipedia.org/wiki/RGB_color_model" \t "_blank)**.

Replace the word black in our body element's background-color with its hex code representation, #000000.

Use Hex Code to Mix Colors

To review, hex codes use 6 hexadecimal digits to represent colors, two each for red (R), green (G), and blue (B) components.

From these three pure colors (red, green, and blue), we can vary the amounts of each to create over 16 million other colors!

For example, orange is pure red, mixed with some green, and no blue. In hex code, this translates to being #FFA500.

The digit 0 is the lowest number in hex code, and represents a complete absence of color.

The digit F is the highest number in hex code, and represents the maximum possible brightness.

Replace the color words in our style element with their correct hex codes.

|  |  |
| --- | --- |
| **Color** | **Hex Code** |
| Dodger Blue | #2998E4 |
| Green | #00FF00 |
| Orange | #FFA500 |
| Red | #FF0000 |

Use Abbreviated Hex Code

Many people feel overwhelmed by the possibilities of more than 16 million colors. And it's difficult to remember hex code. Fortunately, you can shorten it.

For example, red's hex code #FF0000 can be shortened to #F00. This shortened form gives one digit for red, one digit for green, and one digit for blue.

This reduces the total number of possible colors to around 4,000. But browsers will interpret #FF0000 and #F00 as exactly the same color.

Go ahead, try using the abbreviated hex codes to color the correct elements.

|  |  |
| --- | --- |
| **Color** | **Short Hex Code** |
| Cyan | #0FF |
| Green | #0F0 |
| Red | #F00 |
| Fuchsia | #F0F |

Use RGB values to Color Elements

Another way you can represent colors in CSS is by using RGB values.

The RGB value for black looks like this:

rgb(0, 0, 0)

The RGB value for white looks like this:

rgb(255, 255, 255)

Instead of using six hexadecimal digits like you do with hex code, with RGB you specify the brightness of each color with a number between 0 and 255.

If you do the math, the two digits for one color equal 16 times 16, which gives us 256 total values. So RGB, which starts counting from zero, has the exact same number of possible values as hex code.

Let's replace the hex code in our body element's background color with the RGB value for black: rgb(0, 0, 0)

Use RGB to Mix Colors

Just like with hex code, you can mix colors in RGB by using combinations of different values.

Replace the color words in our style element with their correct RGB values.

|  |  |
| --- | --- |
| **Color** | **RGB** |
| Blue | rgb(0, 0, 255) |
| Red | rgb(255, 0, 0) |
| Orchid | rgb(218, 112, 214) |
| Sienna | rgb(160, 82, 45) |

Use Responsive Design with Bootstrap Fluid Containers

In the HTML5 and CSS section of FreeCodeCamp we built a Cat Photo App. Now let's go back to it. This time, we'll style it using the popular Bootstrap responsive CSS framework.

Bootstrap will figure out how wide your screen is and respond by resizing your HTML elements - hence the name Responsive Design.

With responsive design, there is no need to design a mobile version of your website. It will look good on devices with screens of any width.

You can add Bootstrap to any app by adding the following code to the top of your HTML:

<link rel="stylesheet" href="//maxcdn.bootstrapcdn.com/bootstrap/3.3.1/css/bootstrap.min.css"/>

In this case, we've already added it for you to this page behind the scenes.

To get started, we should nest all of our HTML in a div element with the class container-fluid.

Make Images Mobile Responsive

First, add a new image below the existing one. Set its src attribute to https://bit.ly/fcc-running-cats.

It would be great if this image could be exactly the width of our phone's screen.

Fortunately, with Bootstrap, all we need to do is add the img-responsive class to your image. Do this, and the image should perfectly fit the width of your page.

Center Text with Bootstrap

Now that we're using Bootstrap, we can center our heading element to make it look better. All we need to do is add the class text-center to our h2 element.

Remember that you can add several classes to the same element by separating each of them with a space, like this:

<h2 class="red-text text-center">your text</h2>

Create a Bootstrap Button

Bootstrap has its own styles for button elements, which look much better than the plain HTML ones.

Create a new button element below your large kitten photo. Give it the class btn and the text of "Like".

Create a Block Element Bootstrap Button

Normally, your button elements with a class of btn are only as wide as the text that they contain. For example:

<button class="btn">Submit</button>

This button would only be as wide as the word "Submit".

Submit

By making them block elements with the additional class of btn-block, your button will stretch to fill your page's entire horizontal space and any elements following it will flow onto a "new line" below the block.

<button class="btn btn-block">Submit</button>

This button would take up 100% of the available width.

Taste the Bootstrap Button Color Rainbow

The btn-primary class is the main color you'll use in your app. It is useful for highlighting actions you want your user to take.

Add Bootstrap's btn-primary class to your button.

Note that this button will still need the btn and btn-block classes.

Call out Optional Actions with Button Info

Bootstrap comes with several pre-defined colors for buttons. The btn-info class is used to call attention to optional actions that the user can take.

Create a new block-level Bootstrap button below your "Like" button with the text "Info", and add Bootstrap's btn-info and btn-block classes to it.

Note that these buttons still need the btn and btn-block classes.

Warn your Users of a Dangerous Action

Bootstrap comes with several pre-defined colors for buttons. The btn-danger class is the button color you'll use to notify users that the button performs a destructive action, such as deleting a cat photo.

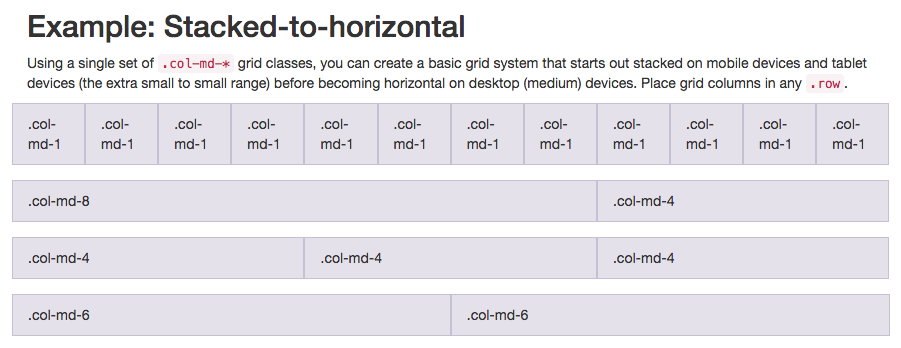
Create a button with the text "Delete" and give it the class btn-danger.

Note that these buttons still need the btn and btn-block classes.

Use the Bootstrap Grid to Put Elements Side By Side

Bootstrap uses a responsive grid system, which makes it easy to put elements into rows and specify each element's relative width. Most of Bootstrap's classes can be applied to a div element.

Here's a diagram of how Bootstrap's 12-column grid layout works:

**[](https://i.imgur.com/FaYuui8.png)**

Note that in this illustration, the col-md-\* class is being used. Here, md means medium, and \* is a number specifying how many columns wide the element should be. In this case, the column width of an element on a medium-sized screen, such as a laptop, is being specified.

In the Cat Photo App that we're building, we'll use col-xs-\*, where xs means extra small (like an extra-small mobile phone screen), and \* is the number of columns specifying how many columns wide the element should be.

Put the Like, Info and Delete buttons side-by-side by nesting all three of them within one <div class="row"> element, then each of them within a <div class="col-xs-4"> element.

The row class is applied to a div, and the buttons themselves can be nested within it.

Ditch Custom CSS for Bootstrap

We can clean up our code and make our Cat Photo App look more conventional by using Bootstrap's built-in styles instead of the custom styles we created earlier.

Don't worry - there will be plenty of time to customize our CSS later.

Delete the .red-text, p, and .smaller-image CSS declarations from your style element so that the only declarations left in your style element are h2 and thick-green-border.

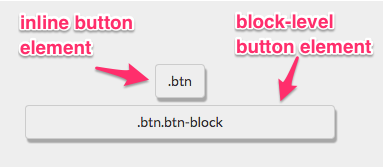
Then delete the p element that contains a dead link. Then remove the red-text class from your h2 element and replace it with the text-primary Bootstrap class.

Finally, remove the "smaller-image" class from your first img element and replace it with the img-responsive class.

Use Spans for Inline Elements

You can use spans to create inline elements. Remember when we used the btn-block class to make the button fill the entire row?

This image illustrates the difference between inline elements and block-level elements:

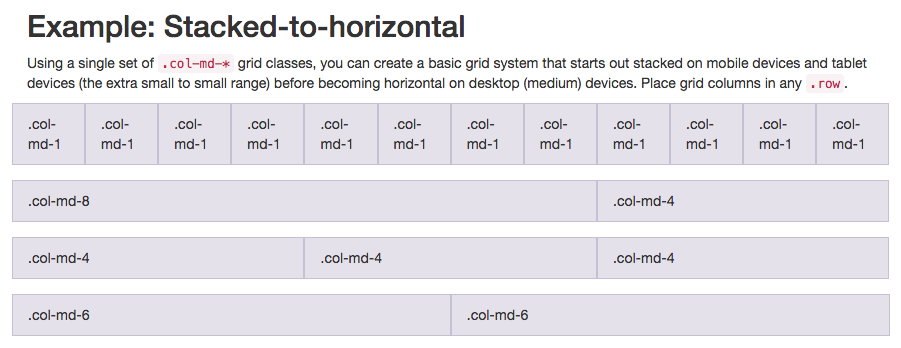
**[](https://i.imgur.com/O32cDWE.png)**

Create a Custom Heading

We will make a simple heading for our Cat Photo App by putting the title and relaxing cat image in the same row.

Remember, Bootstrap uses a responsive grid system, which makes it easy to put elements into rows and specify each element's relative width. Most of Bootstrap's classes can be applied to a div element.

Here's a diagram of how Bootstrap's 12-column grid layout works:

**[](https://i.imgur.com/FaYuui8.png)**

Note that in this illustration, the col-md-\* class is being used. Here, md means medium, and \* is a number specifying how many columns wide the element should be. In this case, the column width of an element on a medium-sized screen, such as a laptop, is being specified.

In the Cat Photo App that we're building, we'll use col-xs-\*, where xs means extra small (like an extra-small mobile phone screen), and \* is the number of columns specifying how many columns wide the element should be.

Nest your first image and your h2 element within a single <div class="row"> element. Nest your h2 element within a <div class="col-xs-8"> and your image in a <div class="col-xs-4"> so that they are on the same line.

Notice how the image is now just the right size to fit along the text?

Add Font Awesome Icons to our Buttons

Font Awesome is a convenient library of icons. These icons are vector graphics, stored in the .svg file format. These icons are treated just like fonts. You can specify their size using pixels, and they will assume the font size of their parent HTML elements.

You can add Font Awesome to any app just by including it by adding the following code to the top of your HTML:

<link rel="stylesheet" href="//maxcdn.bootstrapcdn.com/font-awesome/4.5.0/css/font-awesome.min.css"/>

In this case, we've already added it for you to this page behind the scenes.

The i element was originally used to make other elements italic, but is now commonly used for icons. You add the Font Awesome classes to the i element to turn it into an icon, for example:

<i class="fa fa-info-circle"></i>

Use Font Awesome to add a thumbs-up icon to your like button by giving it an i element with the classes fa and fa-thumbs-up.

Add Font Awesome Icons to all of our Buttons

Font Awesome is a convenient library of icons. These icons are vector graphics, stored in the .svg file format. These icons are treated just like fonts. You can specify their size using pixels, and they will assume the font size of their parent HTML elements.

Use Font Awesome to add an info-circle icon to your info button and a trash icon to your delete button.

Responsively Style Radio Buttons

You can use Bootstrap's col-xs-\* classes on form elements, too! This way, our radio buttons will be evenly spread out across the page, regardless of how wide the screen resolution is.

Nest all of your radio buttons within a <div class="row"> element. Then nest each of them within a <div class="col-xs-6"> element.

Responsively Style Checkboxes

You can use Bootstrap's col-xs-\* classes on form elements, too! This way, our checkboxes will be evenly spread out across the page, regardless of how wide the screen resolution is.

Nest all your checkboxes in a <div class="row"> element. Then nest each of them in a <div class="col-xs-4"> element.

Style Text Inputs as Form Controls

You can add the fa-paper-plane Font Awesome icon by adding <i class="fa fa-paper-plane"></i> within your submit button element.

Give your form's text input field a class of form-control. Give your form's submit button the classes btn btn-primary. Also give this button the Font Awesome icon of fa-paper-plane.

Line up Form Elements Responsively with Bootstrap

Now let's get your form input and your submission button on the same line. We'll do this the same way we have previously: by using a div element with the class row, and other div elements within it using the col-xs-\* class.

Nest both your form's text input and submit button within a div with the class row. Nest your form's text input within a div with the class of col-xs-7. Nest your form's submit button in a div with the class col-xs-5.

This is the last challenge we'll do for our Cat Photo App for now. We hope you've enjoyed learning Font Awesome, Bootstrap, and responsive design!

Create a Bootstrap Headline

Now let's build something from scratch to practice our HTML, CSS and Bootstrap skills.

We'll build a jQuery playground, which we'll soon put to use in our jQuery challenges.

To start with, create an h3 element, with the text jQuery Playground.

Color your h3 element with the text-primary Bootstrap class, and center it with the text-center Bootstrap class.

House our page within a Bootstrap Container Fluid Div

Now let's make sure all the content on your page is mobile-responsive.

Let's nest your h3 element within a div element with the class container-fluid.

Create a Bootstrap Row

Now we'll create a Bootstrap row for our inline elements.

Create a div element below the h3 tag, with a class of row.

Split your Bootstrap Row

Now that we have a Bootstrap Row, let's split it into two columns to house our elements.

Create two div elements within your row, both with the class col-xs-6.

Create Bootstrap Wells

Bootstrap has a class called well that can create a visual sense of depth for your columns.

Nest one div element with the class well within each of your col-xs-6 div elements.

Add Elements within your Bootstrap Wells

Now we're several div elements deep on each column of our row. This is as deep as we'll need to go. Now we can add our button elements.

Nest three button elements within each of your well div elements.

Apply the Default Bootstrap Button Style

Bootstrap has another button class called btn-default.

Apply both the btn and btn-default classes to each of your button elements.

Create a Class to Target with jQuery Selectors

Not every class needs to have corresponding CSS. Sometimes we create classes just for the purpose of selecting these elements more easily using jQuery.

Give each of your button elements the class target.

Add ID Attributes to Bootstrap Elements

Recall that in addition to class attributes, you can give each of your elements an idattribute.

Each id must be unique to a specific element and used only once per page.

Let's give a unique id to each of our divelements of class well.

Remember that you can give an element an id like this:

<div class="well" id="center-well">

Give the well on the left the id of left-well. Give the well on the right the id of right-well.

Label Bootstrap Wells

For the sake of clarity, let's label both of our wells with their ids.

Above your left-well, inside its col-xs-6 div element, add a h4 element with the text #left-well.

Above your right-well, inside its col-xs-6 div element, add a h4 element with the text #right-well.

Give Each Element a Unique ID

We will also want to be able to use jQuery to target each button by its unique id.

Give each of your buttons a unique id, starting with target1 and ending with target6.

Make sure that target1 to target3 are in #left-well, and target4 to target6 are in #right-well.

Label Bootstrap Buttons

Just like we labeled our wells, we want to label our buttons.

Give each of your button elements text that corresponds to its id's selector.

Use Comments to Clarify Code

When we start using jQuery, we will modify HTML elements without needing to actually change them in HTML.

Let's make sure that everyone knows they shouldn't actually modify any of this code directly.

Remember that you can start a comment with <!-- and end a comment with -->

Add a comment at the top of your HTML that says Only change code above this line.

Learn how Script Tags and Document Ready Work

Now we're ready to learn jQuery, the most popular JavaScript tool of all time. Don't worry about JavaScript itself - we will cover it soon.

Before we can start using jQuery, we need to add some things to our HTML.

First, add a script element at the top of your page. Be sure to close it on the following line.

Your browser will run any JavaScript inside a script element, including jQuery.

Inside your script element, add this code: $(document).ready(function() { to your script. Then close it on the following line (still inside your script element) with: });

We'll learn more about functions later. The important thing to know is that code you put inside this function will run as soon as your browser has loaded your page.

This is important because without your document ready function, your code may run before your HTML is rendered, which would cause bugs.

Target HTML Elements with Selectors Using jQuery

Now we have a document ready function.

Now let's write our first jQuery statement. All jQuery functions start with a $, usually referred to as a dollar sign operator, or as bling.

jQuery often selects an HTML element with a selector, then does something to that element.

For example, let's make all of your button elements bounce. Just add this code inside your document ready function:

$("button").addClass("animated bounce");

Note that we've already included both the jQuery library and the Animate.css library in the background so that you can use them in the editor. So you are using jQuery to apply the Animate.css bounce class to your button elements.

Additionally make sure to use $("button").addClass("animated bounce"); instead of $('button').addClass("animated bounce"); since single-quote selectors will not pass our tests.

Target Elements by Class Using jQuery

You see how we made all of your button elements bounce? We selected them with $("button"), then we added some CSS classes to them with .addClass("animated bounce");.

You just used jQuery's .addClass() function, which allows you to add classes to elements.

First, let's target your div elements with the class well by using the $(".well") selector.

Note that, just like with CSS declarations, you type a . before the class's name.

Then use jQuery's .addClass() function to add the classes animated and shake.

For example, you could make all the elements with the class text-primary shake by adding the following to your document ready function:

$(".text-primary").addClass("animated shake");

Target Elements by ID Using jQuery

You can also target elements by their id attributes.

First target your button element with the id target3 by using the $("#target3") selector.

Note that, just like with CSS declarations, you type a # before the id's name.

Then use jQuery's .addClass() function to add the classes animated and fadeOut.

Here's how you'd make the button element with the id target6 fade out:

$("#target6").addClass("animated fadeOut").

Delete your jQuery Functions

These animations were cool at first, but now they're getting kind of distracting.

Delete all three of these jQuery functions from your document ready function, but leave your document ready function itself intact.

Target the same element with multiple jQuery Selectors

Now you know three ways of targeting elements: by type: $("button"), by class: $(".btn"), and by id $("#target1").

Although it is possible to add multiple classes in a single .addClass() call, let's add them to the same element in *three separate ways*.

Using .addClass(), add only one class at a time to the same element, three different ways:

Add the animated class to all elements with type button.

Add the shake class to all the buttons with class .btn.

Add the btn-primary class to the button with id #target1.

Remove Classes from an element with jQuery

In the same way you can add classes to an element with jQuery's addClass() function, you can remove them with jQuery's removeClass() function.

Here's how you would do this for a specific button:

$("#target2").removeClass("btn-default");

Let's remove the btn-default class from all of our button elements.

Change the CSS of an Element Using jQuery

We can also change the CSS of an HTML element directly with jQuery.

jQuery has a function called .css() that allows you to change the CSS of an element.

Here's how we would change its color to blue:

$("#target1").css("color", "blue");

This is slightly different from a normal CSS declaration, because the CSS property and its value are in quotes, and separated with a comma instead of a colon.

Delete your jQuery selectors, leaving an empty document ready function.

Select target1 and change its color to red.

Disable an Element Using jQuery

You can also change the non-CSS properties of HTML elements with jQuery. For example, you can disable buttons.

When you disable a button, it will become grayed-out and can no longer be clicked.

jQuery has a function called .prop() that allows you to adjust the properties of elements.

Here's how you would disable all buttons:

$("button").prop("disabled", true);

Disable only the target1 button.

Change Text Inside an Element Using jQuery

Using jQuery, you can change the text between the start and end tags of an element. You can even change HTML markup.

jQuery has a function called .html() that lets you add HTML tags and text within an element. Any content previously within the element will be completely replaced with the content you provide using this function.

Here's how you would rewrite and emphasize the text of our heading:

$("h3").html("<em>jQuery Playground</em>");

jQuery also has a similar function called .text() that only alters text without adding tags. In other words, this function will not evaluate any HTML tags passed to it, but will instead treat it as the text you want to replace the existing content with.

Change the button with id target4 by emphasizing its text.

Use appendTo to Move Elements with jQuery

Now let's try moving elements from one divto another.

jQuery has a function called appendTo()that allows you to select HTML elements and append them to another element.

For example, if we wanted to move target4from our right well to our left well, we would use:

$("#target4").appendTo("#left-well");

Move your target2 element from your left-well to your right-well.

Remove an Element Using jQuery

Now let's remove an HTML element from your page using jQuery.

jQuery has a function called .remove() that will remove an HTML element entirely

Remove element target4 from the page by using the .remove() function.

Clone an Element Using jQuery

In addition to moving elements, you can also copy them from one place to another.

jQuery has a function called clone() that makes a copy of an element.

For example, if we wanted to copy target2from our left-well to our right-well, we would use:

$("#target2").clone().appendTo("#right-well");

Did you notice this involves sticking two jQuery functions together? This is called function chaining and it's a convenient way to get things done with jQuery.

Target the Parent of an Element Using jQuery

Every HTML element has a parentelement from which it inheritsproperties.

For example, your jQuery Playgroundh3 element has the parent element of <div class="container-fluid">, which itself has the parent body.

jQuery has a function called parent()that allows you to access the parent of whichever element you've selected.

Here's an example of how you would use the parent() function if you wanted to give the parent element of the left-wellelement a background color of blue:

$("#left-well").parent().css("background-color", "blue")

Give the parent of the #target1 element a background-color of red.

Target the Children of an Element Using jQuery

Many HTML elements have childrenwhich inherit their properties from their parent HTML elements.

For example, every HTML element is a child of your body element, and your "jQuery Playground" h3 element is a child of your <div class="container-fluid">element.

jQuery has a function called children()that allows you to access the children of whichever element you've selected.

Here's an example of how you would use the children() function to give the children of your left-well element the color of blue:

$("#left-well").children().css("color", "blue")

Give all the children of your #right-wellelement a color of orange.

Target a Specific Child of an Element Using jQuery

You've seen why id attributes are so convenient for targeting with jQuery selectors. But you won't always have such neat ids to work with.

Fortunately, jQuery has some other tricks for targeting the right elements.

jQuery uses CSS Selectors to target elements. target:nth-child(n) css selector allows you to select all the nth elements with the target class or element type.

Here's how you would give the third element in each well the bounce class:

$(".target:nth-child(3)").addClass("animated bounce");

Make the second child in each of your well elements bounce. You must target the children of element with the target class

Target Even Numbered Elements Using jQuery

You can also target all the even-numbered elements.

Here's how you would target all the odd-numbered elements with class target and give them classes:

$(".target:odd").addClass("animated shake");

Note that jQuery is zero-indexed, meaning that, counter-intuitively, :odd selects the second element, fourth element, and so on.

Try selecting all the even-numbered elements and giving them the classes of animated and shake.

Use jQuery to Modify the Entire Page

We're done playing with our jQuery playground. Let's tear it down!

jQuery can target the body element as well.

Here's how we would make the entire body fade out: $("body").addClass("animated fadeOut");

But let's do something more dramatic. Add the classes animated and hinge to your body element.