***html***

*opening and closing tags are <h1> , </h1>*

*paragraph element :p //p elements are the preferred element for paragraph text on websites. p is short for "paragraph".*

**Uncomment HTML**

Commenting is a way that you can leave comments for other developers within your code without affecting the resulting output that is displayed to the end user.

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

Comments in HTML start with <!-- and end with a -->

<!-- -->

**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.

**Introduction to HTML5 Elements**

HTML5 introduces more descriptive HTML tags. These include main, header, footer, nav, video, article, section and others.

These tags give a descriptive structure to your HTML, make your HTML easier to read, and help with Search Engine Optimization (SEO) and accessibility. The main HTML5 tag helps search engines and other developers find the main content of your page.

Example usage, a main element with two child elements nested inside it:

<main>

<h1>Hello World</h1>

<p>Hello Paragraph</p>

</main>

**Note:** Many of the new HTML5 tags and their benefits are covered in the Applied Accessibility section.

Adding image

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.freecatphotoapp.com/your-image.jpg">

Note that img elements are self-closing.

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

**Note:** If the image is purely decorative, using an empty alt attribute is a best practice.

Ideally the alt attribute should not contain special characters unless needed.

Let's add an alt attribute to our img example above:

<img src="https://www.freecatphotoapp.com/your-image.jpg" alt="A bu

**Link to External Pages with Anchor Elements**

You can use a (*anchor*) elements to link to content outside of your web page.

a elements need a destination web address called an href attribute. They also need anchor text. Here's an example:

<a href="https://www.freecodecamp.org">this links to freecodecamp.org</a>

Then your browser will display the text this links to freecodecamp.org as a link you can click. And that link will take you to the web address https://www.freecodecamp.org.

//external links

**Link to Internal Sections of a Page with Anchor Elements**

a (*anchor*) elements can also be used to create internal links to jump to different sections within a webpage.

To create an internal link, you assign a link's href attribute to a hash symbol # plus the value of the id attribute for the element that you want to internally link to, usually further down the page. You then need to add the same id attribute to the element you are linking to. An id is an attribute that uniquely describes an element.

Below is an example of an internal anchor link and its target element:

<a href="#contacts-header">Contacts</a>

...

<h2 id="contacts-header">Contacts</h2>

When users click the Contacts link, they'll be taken to the section of the webpage with the **Contacts** header element.

**Nest an Anchor Element within a Paragraph**

You can nest links within other text elements.

<p>

Here's a <a target="\_blank" href="https://www.freecodecamp.org"> link to www.freecodecamp.org</a> for you to follow.

</p>

Let's break down the example. Normal text is wrapped in the p element:

<p> Here's a ... for you to follow. </p>

Next is the *anchor* element <a> (which requires a closing tag </a>):

<a> ... </a>

target is an anchor tag attribute that specifies where to open the link. The value \_blank specifies to open the link in a new tab. The href is an anchor tag attribute that contains the URL address of the link:

<a href="https://www.freecodecamp.org" target="\_blank"> ... </a>

The text, link to www.freecodecamp.org, within the a element is called *anchor text*, and will display the link to click:

<a href=" ... " target="...">link to freecodecamp.org</a>

The final output of the example will look like this:

Here's a [link to www.freecodecamp.org](https://www.freecodecamp.org/) for you to follow.

**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 JavaScript, which we'll learn about later.

The current value of the href attribute is a link that points to "https://www.freecatphotoapp.com". Replace the href attribute value with a #, also known as a hash symbol, to create a dead link.

For example: href="#"

**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://www.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.

**odered list**

**<ol>**

**<li> ......<li>**

**</ol>**

**create a text field**

**<input type="text">**

**Add Placeholder Text to a Text Field**

Placeholder text is what is displayed in your input element before your user has inputted anything.

You can create placeholder text like so:

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

**Note:** Remember that input elements are self-closing.

**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>

**Create a Set of Radio Buttons**

You can use *radio buttons* for questions where you want the user to only give you one answer out of multiple options.

Radio buttons are a type of input.

Each of your radio buttons can be nested within its own label element. By wrapping an input element inside of a label element it will automatically associate the radio button input with the label element surrounding it.

All related radio buttons should have the same name attribute to create a radio button group. By creating a radio group, selecting any single radio button will automatically deselect the other buttons within the same group ensuring only one answer is provided by the user.

Here's an example of a radio button:

<label>

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

</label>

It is considered best practice to set a for attribute on the label element, with a value that matches the value of the id attribute of the input element. This allows assistive technologies to create a linked relationship between the label and the related input element. For example:

<input id="indoor" type="radio" name="indoor-outdoor">

<label for="indoor">Indoor</label>

We can also nest the input element within the label tags:

<label for="indoor">

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

</label>

**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 can be nested within its own label element. By wrapping an input element inside of a label element it will automatically associate the checkbox input with the label element surrounding it.

All related checkbox inputs should have the same name attribute.

It is considered best practice to explicitly define the relationship between a checkbox input and its corresponding label by setting the for attribute on the label element to match the id attribute of the associated input element.

Here's an example of a checkbox:

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

**Use the value attribute with Radio Buttons and Checkboxes**

When a form gets submitted, the data is sent to the server and includes entries for the options selected. Inputs of type radio and checkbox report their values from the value attribute.

For example:

<label for="indoor">

<input id="indoor" value="indoor" type="radio" name="indoor-outdoor">Indoor

</label>

<label for="outdoor">

<input id="outdoor" value="outdoor" type="radio" name="indoor-outdoor">Outdoor

</label>

Here, you have two radio inputs. When the user submits the form with the indoor option selected, the form data will include the line: indoor-outdoor=indoor. This is from the name and value attributes of the "indoor" input.

If you omit the value attribute, the submitted form data uses the default value, which is on. In this scenario, if the user clicked the "indoor" option and submitted the form, the resulting form data would be indoor-outdoor=on, which is not useful. So the value attribute needs to be set to something to identify the option.

**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>

**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.

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

**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 property that is responsible for the color of an element's text is the color style property.

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

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

**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, class names start with a period. In your HTML elements' class attribute, the class name does not include the period.

**Style Multiple Elements with a CSS Class**

Classes allow you to use the same CSS styles on multiple HTML elements. You can see this by applying your red-text class to the first p element.

**Change the Font Size of an Element**

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

h1 {

font-size: 30px;

}

**Set the Font Family of an Element**

You can set which font an element should use, 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;

}

**Specify How Fonts Should Degrade**

There are several default fonts that are available in all browsers. These generic font families 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 degrade to the sans-serif font when Helvetica isn't available, you will specify it as follows:

p {

font-family: Helvetica, sans-serif;

}

Generic font family names are not case-sensitive. Also, they do not need quotes because they are CSS keywords.

**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. This class should add a 10px, solid, green border around an HTML element. Apply the class to your cat photo.

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

<img class="class1 class2">

**Add Rounded Corners with 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 challenge allows for multiple possible solutions. For example, you may add border-radius to either the .thick-green-border class or the .smaller-image class.

**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 div element.

**Use an id Attribute to Style an Element**

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

However, an id is not reusable and should only be applied to one element. An id also has a higher specificity (importance) than a class so if both are applied to the same element and have conflicting styles, the styles of the id will be applied.

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.

