**Step 1**

Begin with the basic HTML structure. Add a DOCTYPE declaration and html, head, body, and title elements.

Set the language of this page to English. Set the title to Piano

<!Doctype html>

  <html lang="en">

    <head>

<title>Piano</title>

    </head>

    <body>

      </body>

    </html>

# Step 2

Add two meta tags, one to optimize your page for mobile devices, and one to specify an accepted charset for the page.

<meta charset="utf-8"/>

<meta name="viewport" content="width=device-width, initial-scale=1.0">

# Step 3

Time to start working on the piano. Create a div element within your body element with the id set to piano.

<body>

  <div id="piano"></div>

</body>

<div class="keys"></div>

# Step 5

Within your .keys element, add seven div elements. Give them all the class key.

 <div id="piano">

      <div class="keys">

         <div class="key"></div>

         <div class="key"></div>

         <div class="key"></div>

         <div class="key"></div>

         <div class="key"></div>

         <div class="key"></div>

         <div class="key"></div>

      </div>

    </div>

# Step 6

Remember that a class attribute can have multiple values. To separate your white keys from your black keys, you'll add a second class value of black--key. Add this to your second, third, fifth, sixth, and seventh .key elements.

div id="piano">

      <div class="keys">

        <div class="key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

        <div class="key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

      </div>

    </div>

# Step 7

Now copy the set of seven .key elements, and paste two more sets into the .keys div.

<div id="piano">

      <div class="keys">

        <div class="key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

        <div class="key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

        <div class="key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

        <div class="key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

        <div class="key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

        <div class="key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

      </div>

    </div>

# Step 9

Browsers can apply default margin and padding values to specific elements. To make sure your piano looks correct, you need to reset the box model.

Add an html rule selector to your CSS file, and set the box-sizing property to border-box.

html {

  box-sizing: border-box;

}

The ::before selector creates a pseudo-element which is the first child of the selected element, while the ::after selector creates a pseudo-element which is the last child of the selected element. These pseudo-elements are often used to create cosmetic content, which you will see later in this project.

For now, create a CSS selector to target all elements with \*, and include the pseudo-elements with ::before and ::after. Set the box-sizing property to inherit.

# Step 10

Now that you have reset the html box model, you need to pass that on to the elements within as well. To do this, you can set the box-sizing property to inherit, which will tell the targeted elements to use the same value as the parent element.

You will also need to target the pseudo-elements, which are special keywords that follow a selector. The two pseudo-elements you will be using are the ::before and ::after pseudo-elements.

The ::before selector creates a pseudo-element which is the first child of the selected element, while the ::after selector creates a pseudo-element which is the last child of the selected element. These pseudo-elements are often used to create cosmetic content, which you will see later in this project.

For now, create a CSS selector to target all elements with \*, and include the pseudo-elements with ::before and ::after. Set the box-sizing property to inherit.

html {

  box-sizing: border-box;

}

\* ,::before , ::after{

box-sizing:inherit;

}

# Step 11

Now target your #piano element with an id selector. Set background-color property to #00471b, the width property to 992px and the height property to 290px.

#piano{

  background-color:#00471b;

  width:992px;

  height:290px;

}

# Step 12

Set the margin of the #piano to 80px auto.

#piano {

  background-color: #00471b;

  width: 992px;

  height: 290px;

  margin:80px auto;

}

# Step 13

Time to style the keys. Below the #piano rule, target the .keys element with a class selector. Give the new rule a background-color property of #040404, a width property of 949px and a height property of 180px.

.keys{

  background-color:#040404;

  width:949px;

  height:180px;

}

# Step 14

Give the .keys a padding-left of 2px.

.keys {

  background-color: #040404;

  width: 949px;

  height: 180px;

  padding-left:2px;

}

# Step 15

Move the keys into position by adjusting the #piano selector. Set the padding property to 90px 20px 0 20px.

#piano {

  background-color: #00471b;

  width: 992px;

  height: 290px;

  margin: 80px auto;

  padding:90px 20px 0 20px;

}

# Step 16

Time to style the keys themselves. Create a class selector for the .key elements. Set the background-color set to the value #ffffff, the position property to relative, the width property to 41px, and the height property to 175px.

.key{

  position:relative;

  width:41px;

  height:175px;

}

.key {

  background-color: #ffffff;

  position: relative;

  width: 41px;

  height: 175px;

  margin:2px;

  float:left;

}

# Step 18

Now it is time to use the pseudo-selectors you prepared for earlier. To create the black keys, add a new .key.black--key::after selector. This will target the elements with the class key black--key, and select the pseudo-element after these elements in the HTML.

In the new selector, set the background-color to #1d1e22. Also set the content property to "". This will make the pseudo-elements empty.

The content property is used to set or override the content of the element. By default, the pseudo-elements created by the ::before and ::after pseudo-selectors are empty, and the elements will not be rendered to the page. Setting the content property to an empty string "" will ensure the element is rendered to the page while still being empty.

If you would like to experiment, try removing the background-color property and setting different values for the content property, such as "♥". Remember to undo these changes when you are done so the tests pass.

.key.black--key::after{

  content:"";

  background-color:#1d1e22;

}

# Step 19

Give the .key.black--key::after a position property set to absolute and a left property set to -18px.

.key.black--key::after {

  background-color: #1d1e22;

  content: "";

  position:absolute;

  left:-18px;

}

# Step 20

For the .key.black--key::after, set the width to 32px and the height to 100px.

.key.black--key::after {

  background-color: #1d1e22;

  content: "";

  position: absolute;

  left: -18px;

  width:32px;

  height:100px;

}

# Step 21

The piano needs the freeCodeCamp logo to make it official.

Add an img element before your .keys element. Give the img a class of logo, and set the src to https://cdn.freecodecamp.org/platform/universal/fcc\_primary.svg. Give it an alt text of freeCodeCamp Logo.

  <div id="piano">

      <img class="logo" src="https://cdn.freecodecamp.org/platform/universal/fcc\_primary.svg" alt="freeCodeCamp Logo"/>

      <div class="keys

.logo{

  width:200px;

  position:absolute;

  top:23px;

}

# Step 23

The img element needs its parent to have a position set as a point of reference. Set the position of the #piano selector to relative.

#piano {

  background-color: #00471b;

  width: 992px;

  height: 290px;

  margin: 80px auto;

  padding: 90px 20px 0 20px;

  position:relative;

}

# Step 24

To smooth the sharp edges of the piano and keys, start by giving the #piano a border-radius of 10px.

piano {

  background-color: #00471b;

  width: 992px;

  height: 290px;

  margin: 80px auto;

  padding: 90px 20px 0 20px;

  position: relative;

  border-radius:10px;

}

# Step 25

Give the .key selector a border-radius value of 0 0 3px 3px.

.key {

  background-color: #ffffff;

  position: relative;

  width: 41px;

  height: 175px;

  margin: 2px;

  float: left;

  border-radius:0 0 3px 3px;

}

.key.black--key::after {

  background-color: #1d1e22;

  content: "";

  position: absolute;

  left: -18px;

  width: 32px;

  height: 100px;

  border-radius: 0 0 3px 3px;

}

# Step 27

The @media at-rule, also known as a media query, is used to conditionally apply CSS. Media queries are commonly used to apply CSS based on the viewport width using the max-width and min-width properties.

In the below example the padding is applied to the .card class when the viewport is 960px wide and below.

@media (max-width: 960px) {

.card {

padding: 2rem;

}

}

Add a media query that will be applied when the viewport is 768px wide and below.

@media (max-width: 768px) {

    #paino{

      width:358px;

    }

}

# Step 29

Within the @media query, add a .keys selector and set the width to 318px.

@media (max-width: 768px) {

  #piano {

    width: 358px;

  }

  .keys{

    width:318px;

  }

}

# Step 30

Now add a .logo selector to the @media query, and set the width property to 150p

@media (max-width: 768px) {

  #piano {

    width: 358px;

  }

  .keys {

    width: 318px;

  }

  .logo{

    width:150px;

  }

}

# Step 31

You might have noticed the keys collapse when the browser window is smaller than 768px. Set overflow to hidden in the first .keys selector, to take care of this issue. This property will hide any element that is pushed outside the set width value of .keys.

.keys {

  background-color: #040404;

  width: 949px;

  height: 180px;

  padding-left: 2px;

  overflow:hidden;

}

# Step 32

Logical operators can be used to construct more complex media queries. The and logical operator is used to query two media conditions.

For example, a media query that targets a display width between 500px and 1000px would be:

@media (min-width: 500px) and (max-width: 1000px){

}

Add another @media rule to apply if the browser window is wider than 769px but smaller than 1199px.

@media(min-width:769px)and (max-width:1199px){

}

# Step 33

For the new @media rule, set the width of the #piano to 675px and the width of the .keys to 633px.

With that, your piano is complete!

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <title>Piano</title>

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <link rel="stylesheet" href="./styles.css">

  </head>

  <body>

    <div id="piano">

      <img class="logo" src="https://cdn.freecodecamp.org/platform/universal/fcc\_primary.svg" alt="freeCodeCamp Logo" />

      <div class="keys">

        <div class="key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

        <div class="key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

        <div class="key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

        <div class="key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

        <div class="key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

        <div class="key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

        <div class="key black--key"></div>

      </div>

    </div>

  </body>

</html>

CSS

html {

  box-sizing: border-box;

}

\*, \*::before, \*::after {

  box-sizing: inherit;

}

#piano {

  background-color: #00471b;

  width: 992px;

  height: 290px;

  margin: 80px auto;

  padding: 90px 20px 0 20px;

  position: relative;

  border-radius: 10px;

}

.keys {

  background-color: #040404;

  width: 949px;

  height: 180px;

  padding-left: 2px;

  overflow: hidden;

}

.key {

  background-color: #ffffff;

  position: relative;

  width: 41px;

  height: 175px;

  margin: 2px;

  float: left;

  border-radius: 0 0 3px 3px;

}

.key.black--key::after {

  background-color: #1d1e22;

  content: "";

  position: absolute;

  left: -18px;

  width: 32px;

  height: 100px;

  border-radius: 0 0 3px 3px;

}

.logo {

  width: 200px;

  position: absolute;

  top: 23px;

}

@media (max-width: 768px) {

  #piano {

    width: 358px;

  }

  .keys {

    width: 318px;

  }

  .logo {

    width: 150px;

  }

}

@media (max-width: 1199px) and (min-width: 769px) {

#piano{

  width:675px;

}

.keys{

  width:633px;

}

}