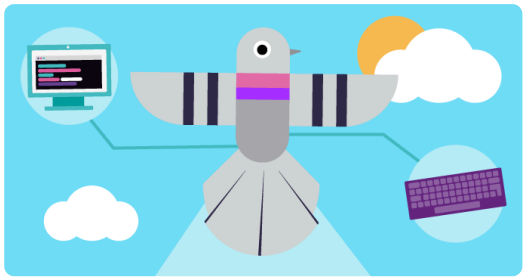
Projects

Bird watch website 2.0

Take your website design skills further!



Step 1 Introduction

Build upon your existing HTML/CSS skills to make a website and gain more control over how it looks.

What you will make

Here is an example of how your website might look after completing these Sushi Cards:

What you will learn

- How to create your own colours using code
- Ways to organise content on your website so that you can apply styles, and to make it friendly for screen readers
- How to use CSS styling to create themes as well as style individual elements
- Controlling the size of elements with different kinds of measurements
- Making things happen when you hover over elements on your page
- Animating elements of your website with CSS
- How to use the developer tools to sneak a peek at the code of any website, and to test out parts of yours

What you will need

Hardware

- A computer capable of accessing **trinket.io** (https://trinket.io)

Software

This project can be completed in a web browser using **trinket.io** (https://trinket.io).

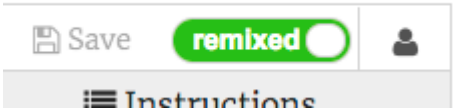
Step 2 Getting set up

- Go to the starter trinket project at **dojo.soy/se-html2-starter** (http://dojo.soy/se-html2-starter). The examples in these Sushi Cards are mainly based on this project.
- If you prefer, you can work with a website that you already made.

I have an account on Trinket

- Click the **Remix** button in the top right-hand corner of the project (if you are not signed in, you will be prompted to do so. Once you've signed in, you will then need to click the **Remix** button again). This creates a copy of the project for you to work with.

It should say **remixed** after you click it:



I don't have an account on Trinket

Even if you don't have an account, you can still work with Trinket.

Saving your work

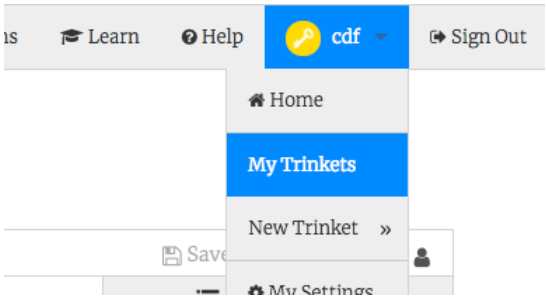
You can save your work by using one of the options in the **Share** menu. You can either download the project or get a link that you can save, for example in a document, or send via email.
Note: each time you make a change to the project, you will get a new link.

How to sign up for an account

If you want to create an account on Trinket, follow the steps below. This will allow you to access your work easily from any computer, and to **remix** projects somebody else has shared with you (meaning save a copy to which you can make changes).

- In browser tab with the starter trinket project, click **Sign Up For Your Free Account**. You will need an email address to sign up.
- Enter your email address and choose a password, or ask somebody to do this for you.

- You can now access all your saved or remixed projects by clicking on your username and going to **My Trinkets**.



Step 3 All the colours!

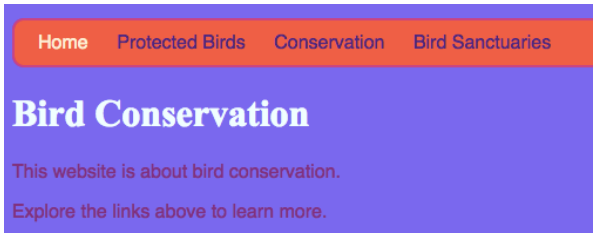
As you have seen before, you can type in many different colour names as words, and the browser will recognise them. But a more common way to set colours is to use something called **hex codes** ('hex' is short for **hexadecimal**, a special way of counting).

- Take a look at your **style sheet**. That's the file that has `.css` in the name.
- Inside the CSS rules for `body`, set the background colour to the hex code `#7B68EE`:

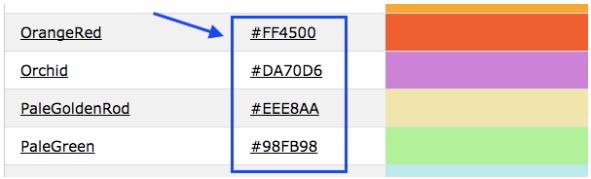
```
background-color: #7B68EE;
```

Note: If you are using a Mac, you can type `#` by press the `alt` and the 3 keys at the same time.

Your website should now have a purple background.



- Not a fan of purple? Go to **this web page** (http://dojo.soy/se-html2-colours) and choose another colour for your style sheet — instead of typing the name of the colour, type in the hex code.



Colour codes allow you to create any colour, even if it's not on any list of colour names.

- Try making up your own colour code. It must start with a `#`. This tells the browser that it is a hex code instead of a colour name. The rest of the code is made up of six characters. They can be any number from **0 to 9** and any letter from **A to F**.

How does it work?

Every colour is made by mixing different amounts of **red**, **green**, and **blue**. You will sometimes see this written down as **RGB**. Each of these colours is represented by two of the six digits in your HEX code. `00` is the minimum, and `FF` is the maximum.

Hexadecimal is a way of counting that makes numbers shorter to write by using the letters A-F as extra digits. The number `255` is written as `FF` in hexadecimal. You don't need to worry about learning to count with hexadecimal numbers. Instead, experiment with different hex codes to get used to using them.

- Here are some basic colours to try out on your website. Try putting in smaller numbers instead of `FF` to see how the shades change.

	R	G	B	Result
#	FF	00	00	Red
#	00	FF	00	Green
#	00	00	FF	Blue
#	FF	FF	00	Yellow
#	FF	00	FF	Magenta
#	00	FF	FF	Cyan
#	FF	8c	00	Dark orange

Mixing the perfect colour can take a lot of experimenting. Luckily, there are plenty of online colour picking tools that help you get the hex code for any colour you want.

- Try out **this colour picker** (http://dojo.soy/se-html2-picker) to choose some hex colour codes to use for the rest of the styles on your website.

Step 4 Organising your page

So far you've used **headings** and **paragraphs** to make your **content** look tidy and easy to read. Let's make it even more organised by grouping things together.

What is content?

Content is all the stuff on your web page, such as text and pictures.

- Go to the `birds.html` file (or one of your own pages if you're not using the example project) and, near the top, just **below** the opening `<main>` tag, type the following on a new line:

```
<main>
  <article>
```

- If your editor automatically added in a closing `</article>` tag for you, delete it.
- At the bottom of the file, just **above** the closing `</main>` tag, add a new line and close the `article` element:

```
</article>
</main>
```

Your `main` element should look something like this now (you might have different content in between the `article` tags of course):

```
<main>
  <article>
    <h1>Birds of conservation concern in Ireland</h1>
    <p>
     There are a number of birds in Ireland whose numbers are in decline.
     Some of those with a high priority for conservation are:
    </p>

    <h2>Barn Owl</h2>
    <p>The barn owl is the most common owl and is found in most parts of the world.
    It has seen a huge decline in Ireland and Europe in recent years.</p>
    <img src="barn-owl-landing.jpg" alt="Barn owl landing on a branch" height="200px" />
    <h2>Curlew</h2>
    <p>The curlew is recognisable by its long curved bill.
    </p>
    <img src="curlew.jpg" width="200px" />
    <p>Curlews use their long bills to search for worms in mud or very soft ground.
    </p>
    <img src="curlew2.jpg" width="200px" />
  </article>
</main>
```

- Now look at the content in your `article` and try to break it up into sections. Put this new pair of tags around each bit: `<section> </section>`. Here's an example of what it might look like:

```
<article>
  <h1>My favourite places to see in Ireland</h1>
  <section>
    <h2>Barn Owl</h2>
    <p>The barn owl is the most common owl and is found in most parts of the world.
    It has seen a huge decline in Ireland and Europe in recent years.</p>
    <img src="barn-owl-landing.jpg" alt="Barn owl landing on a branch" height="200px" />
  </section>
  <section>
    <h2>Curlew</h2>
    <p>The curlew is recognisable by its long curved bill.
    </p>
    <img src="curlew.jpg" width="200px" />
    <p>Curlews use their long bills to search for worms in mud or very soft ground.
    </p>
    <img src="curlew2.jpg" width="200px" />
  </section>
</article>
```

What are the new tags all about?

Think of these new elements as **containers**. They are used to group things together. It's a bit like organising things in boxes and shelves in your home!

This makes your website friendly for screen readers, gives you more control over the layout, and, as you'll see, it allows you to really get creative with the styling.

Anything can go in between the tags. Usually it will be more than one element, but it doesn't have to be. It can be HTML elements of any kind. What you are doing is telling the browser that everything in between these tags belongs together.

Article

The `article` element is a container for a whole piece of content, in this case a set of information about attractions in Ireland. If you have different bits of content that aren't related, you should put each one into its own `article` element instead of putting one set of the tags around the whole lot.

Section

The `section` element lets you divide up related content into smaller chunks and put each chunk into its own container.

Others exist too!

These aren't the only container elements in HTML. If you've ever created a menu and then put it in between `<nav> </nav>` tags, that's another example of a type of container. So are `<main> </main>` and `<header> </header>` — can you think of any more?

Your web page might not look different yet, but once the content has been organised into container tags, you'll be able to do some cool things to it with CSS. Remember, HTML controls how your website is organised, and CSS controls how it looks.

Challenge: organise your website

- Have a go at organising all of the content on your website using the `article` and `section` containers in this way.

I need a hint

Look at the Conservation page of the example project. You'll see that I've added an `article` with a bunch of `section` tags into the file `conservation.html`:

```
<main>
        <article>

        <h1>Conservation efforts</h1>
      <p>
       Various kinds of work are carried out in Ireland in order to protect bird species.
      </p>

      <section>
        <h2>Research and monitoring</h2>
        <p>
          An essential part of bird conservation is monitoring and recording
          information about the species such as their numbers, breeding habits, etc.

        </p>
        <p>
          Scientific research may be carried out to determine whether a species is
          in decline and how to address the problem.
        </p>
      </section>

      <section>
        <h2>Habitat protection</h2>
        <p>
          The destruction of habitat is a serious threat to many birds and
          therefore protecting habitats is crucial to protecting the species.
        </p>

        <p>One example of this is the preservation of wetlands in Ireland.</p>
      </section>

      <section>
        <h2>Control invasive plants and animals</h2>
        <p>
          Mink and rats are a predator that threaten many bird species,
          for example by eating their eggs.
        </p>
        <p>
          Rhododendron is an example of an invasive plant which can very quickly
          take over large areas of countryside, disrupting the biodiversity.
        </p>
      </section>
        </article>
</main>
```

On the next card, you'll design a different theme for each page that's organised into articles and sections!

Step 5 Design some themes

Writing CSS rules for elements such as `section` and `p` is great, but what if you want to make some of them look different from others? On this card you will learn how to apply different sets of style rules to elements of the same type and create a different theme for each page on your website!

- Go to your style sheet file and add the following — be sure to include the dot in front!

```
.topDivider {
  border-top-style: solid;
  border-top-width: 2px;
  border-top-color: #F5FFFA;
  padding-bottom: 10px;
}
```

- Now go to `birds.html` (or the HTML file you're working on if you're using your own project), and add the following **attribute** to each `section` tag:

```
<section class="topDivider">
```

You should see a line appear above each section on the page. Congratulations — you've just used your first **CSS class**!

- Look at how your web page looks now and compare it to the other pages that have `section` elements. You'll see that only the ones where you added the attribute `class="topDivider"` will have the line on top.

How does it work?

Remember that when you use a CSS **selector** such as `section` or `p` or `nav ul`, the style rules apply to **all** the elements of that type on your website.

With CSS **classes**, you're able to change the style of just **some** of the elements.

Putting a dot in front of your selector makes it into a **class selector**. A class can have any name, so it doesn't have to be the name of a HTML element. For example:

```
.myAwesomeClass {
  /* my cool style rules go here */
}
```

To choose which elements the style rules apply to, you add the `class` **attribute** to those elements in the HTML code: put the name of the class in as the value for the attribute, **without** the dot, like this:

```
class="myAwesomeClass"
```

- Ready to try another class? Add the following CSS code to `styles.css`:

```
.stylishBox {
  background-color: #87CEFA;
  color: #A52A2A;
  border-style: solid;
  border-width: 2px;
  border-color: #F5FFFA;
  border-radius: 10px;
}
```

- Then, on a different page of your website, add the class to some elements there. I'm going to add it to the `section` elements on the Conservation page of my website, like this: `<section class="stylishBox">`.

It looks great, but now my sections are all squashed together.

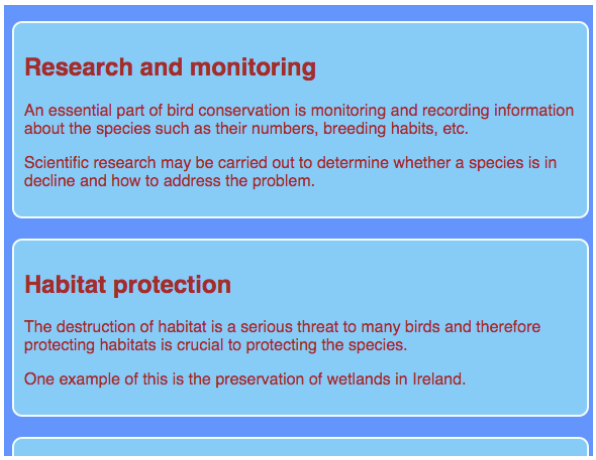
You can apply as many CSS classes to an element as you like. Just write the names of all the classes you want to use inside the `class` attribute (remember, without the dot!), separating them with spaces.

- Let's make another CSS class to give the sections some margin and padding. In the `styles.css` file, create the following CSS class:

```
.someSpacing {
  padding: 10px;
  margin-top: 20px;
}
```

- In your `html` code, add the new class to each of the elements you were working on, like this:

```
<section class="stylishBox someSpacing">
```



So CSS classes let you **choose** which elements to style, and they let you **reuse** the same set of style rules on any elements you want.

- Go to `index.html` and add the `stylishBox` class to the `main` element, or another element on the page. You can remove it again afterwards!

```
<main class="stylishBox">
```

Here's what my home page looks like with the CSS class. I've also added the `topDivider` and `someSpacing` classes to the `img` tag with the picture of the barn owl.

Challenge: make some new classes

- Use CSS **classes** to define a few different picture sizes for your website, for example `.smallPictures` and `.mediumPictures`. Then remove the `width` attribute from each of your `img` elements and add the appropriate class instead.

I need a hint

Here's an `img` tag with a `width` attribute:

```
<img src="barn-owl.jpg" alt="A barn owl" width="100px" />
```

When you remove the `width` attribute and control the size with the CSS class instead, it looks like this:

```
<img src="barn-owl.jpg" class="smallPictures" alt="A barn owl" />
```

By using a CSS class, you can easily change the width of all the pictures at once by changing only one line of code in your style sheet!

Step 6 Individual style

Let's jazz up the home page a bit! With another kind of CSS selector, you can apply a unique set of CSS rules to just **one specific element**.

- Go to `index.html` and find a paragraph (`p`) element, or add one in if you don't have any. Add the following **attribute** to the tag:

```
<p id="myCoolText">
    This website is about bird conservation.
</p>
```

The `id` is a name you give a particular element to **identify** it. No two elements on a page should ever have the same `id`!

- Now go to your style sheet and add the following code:

```
#myCoolText {
    color: #003366;
    border: 2px ridge #ccffff;
    padding: 15px;
    text-align: center;
}
```

Your text should look like this now:

A selector with a `#` in front of it is used to apply CSS rules to one specific element on your website. You specify the element with the help of the name that you assigned the element's `id` attribute.

- Let's do one for the `body` of the home page. Go to `index.html` and add an `id` to the `body` tag.

```
<body id="frontPage">
```

- In the style sheet, add the following CSS rules:

```
#frontPage {
    background: #48D1CC;
    background: linear-gradient(#fea3aa, #f8b88b, #faf884, #baed91, #baed91, #b2cefe, #f2a2e8, #fea3aa);
}
```

You should get something that looks like this:

You just used a **gradient**! That's the name given to the effect where one colour fades into another. Note: The first `background` property above the gradient one determines a default colour for browsers that don't support gradients.

If you typed the code perfectly and you didn't get the lovely rainbow effect above, it could be that your browser doesn't support gradients.

You can make lots of different effects with gradients. If you want to learn more, go **here** (http://dojo.soy/se-html2-gradients).

Challenge: style some more elements

- Try giving another element an `id` and styling that element using the ID selector with a `#` as above. How about making one picture have a `border-radius` of `100%` so that it's fully rounded? Any other pictures on the website will stay the same as they are.

I need a hint

To define style rules for a specific element, you use the `#` symbol, and the name that you gave the element as its `id`.

```
#owly {
  border-radius: 100%;
}
```

Note: the name you type in front of the CSS rules should **exactly** match the name you put in the element's `id` attribute.

Step 7 See the code on other websites!

Note: To complete this step, you need use one of these web browsers: Chrome, Firefox, or Internet Explorer/Edge. If you don't have access to one of them, you can just continue on to the next card.

On this card you'll learn how to sneak a peek at the code of any website using the **inspector tool**, and you'll also find out how to make some changes that only you can see!

- Before you start, make sure your project is saved. Then refresh your website by clicking the refresh icon in your browser.
- On your web page (the actual page, not the code) highlight the text with the border that you added on the previous card, then right-click on it and select the option **Inspect** from the menu that appears. (The option might be called 'Inspect Element' or similar, depending on what browser you are using. If you're having trouble finding a menu option, just ask someone at your Dojo for help.)

A whole new box will appear in your web browser with lots of tabs and code: the **developer tools**, or **dev tools** for short. Here you can see the code for the thing you clicked on, as well as the code for the whole page!

Inspecting the HTML code

- Look for the tab that shows you the HTML code for the page (it might be called 'Elements' or 'Inspector'). The code should look pretty much the same as how you typed it in your HTML file! You can click the little triangles on the right-hand side to expand code that is hidden.
- Double-click on the text in between the tags. You should be able to edit it now! Type something in and press `Enter`.
- Do you see the text update on your website? Note: only you can see these changes.
- Now **reload** the page and watch what happens. Your changes should disappear!
- In the top left-hand corner of the dev tools box, click the icon that looks like a tiny rectangle with an arrow. Now you can move your cursor over the web page, and the HTML inspector will show you the code describing it.

Inspecting the CSS code

- Let's have a look at the CSS code next. Look for the **Styles** tab in the developer tools (it might be called 'Style Editor' or similar). You should see a bunch of CSS rules, including the ones you created for that paragraph, `#myCoolText`.
- In the `#myCoolText` rules, click on the value next to the `color` property. Try typing in a different value. Watch the text on your web page change colour straight away!

Note: you can also click the coloured square to change the colour using a colour picker tool.

- Click in the space after the colour. A new line starts, where you can type more CSS. Type the following and press `Enter`:

```
background-color: #660066;
```

You should see the background change on that piece of text.

How does it work?

When you change website code using the developer tools, you are **temporarily** changing what it looks like **in your browser**. You aren't actually changing the files that make up the website.

When you refresh the page, you are loading up the website again from its files (on the internet or on your computer). That's why your changes disappear.

Now that you know that, you can have some fun messing with the code on other websites!

- Try using these tools to look at the code on another website. You can even make changes if you like! Remember, only you can see the changes you make, and everything will reset when you refresh the page.

Step 8 Automatically adjust the size

Up until now you've been using **pixels** to set the size of things, e.g. `10px`. On this card you will learn about other measurements you can use.

- Go to `index.html` and find the `img` element with the picture of the barn owl, or find another `img` tag on your website.
- Delete the `width` attribute if it's there, and give the element an `id` if it doesn't already have one.

```
<img src="barn-owl.jpg" id="owly" alt="A barn owl" />
```

- In your CSS file, define the `width` property for your picture as shown below (you might need to create the CSS block with the `id` selector if you haven't already done so on a previous card).

```
#owly {
  width: 50%;
  border-radius: 100%;
}
```

Note: 50% (50 percent) is **half**.

- Try resizing your browser window and watch what happens to the picture.

You should see that the picture gets bigger and smaller when you make the window bigger and smaller. That is because it is taking up 50% of the width of the **main** element (which is roughly the width of the page).

How does it work?

When you set the size of something in pixels, you are setting an exact size and it doesn't change. This is called an **absolute** measurement.

Another way to set the size of things is using **relative** measurements, so that size depends on how big elements are compared to each other. Then, whenever one thing changes size, everything else will automatically change size as well to keep the same **proportions**.

When you're using **relative** measurements, it's important to know what the **parent** of your element is. The parent is the thing that your element is inside of, and that's what the measurement will be in relation to. For example, the parent of the image above is the `article` element, because the `img` element is in between the `<article></article>` tags.

If you set the `width` of an element to `100%`, that will make it be the same width as the parent container it's in.

- Experiment with different numbers in front of the `%`.

Step 9 Animation

Did you know you can use CSS to make things move around? You'll learn how on this card!

- Before you get started, make sure you have a picture on your website with an `id` and a corresponding CSS block which sets the `width` to `100px`. I'm going with the picture of the barn owl from before, and my CSS block looks like this:

```
#owly {
    border-radius: 100%;
    width: 100px;
}
```

- Go to the bottom of your CSS file and add the following code:

```
@keyframes myFirstAnimation {
    from {
        width: 100px;
    }
    to {
        width: 300px;
    }
}
```

This code creates an animation called `myFirstAnimation` that you can add to any element on your website. What do you think it will do?

- Find your CSS rules for the picture and add the following three properties:

```
animation-name: myFirstAnimation;
animation-duration: 2s;
animation-iteration-count: 1;
```

- Now watch what happens on your web page! Try different values for `animation-iteration-count` to see what it does.

- Let's try another animation! Add the following code to the end of your CSS file:

```
@keyframes rainbowGlow {
    0% {
        color: #FFD700;
    }
    50% {
        color: #663399;
    }
    100% {
        color: #FFD700;
    }
}
```

- Now find the `#myCoolText` CSS rules from earlier and add in the animation code:

```
#myCoolText {
    color: #003366;
    border: 2px ridge #ccffff;
    padding: 15px;
    text-align: center;
    animation-name: rainbowGlow;
    animation-duration: 1.5s;
    animation-iteration-count: 1;
}
```

When you use **percentage values** instead of `from` and `to`, you're able to set in-between values as well as just start and end values. You can set as many in-between values as you like using different percentage values from `0` all the way up to `100`.

- Change the value of `animation-iteration-count` to `infinite`. See if you can guess what will happen before you test it!
- Try out different values for `animation-duration` to speed up or slow down your animation.
- One final trick! Add this animation code:

```
@keyframes slide {
    0% {
        background-position-x: 0;
    }
    100% {
        background-position-x: 600vw;
    }
}
```

- Now find the `#frontPage` CSS rules you wrote earlier and change them to:

```
#frontPage {
    background: repeating-linear-gradient(-45deg, red 0%, yellow 7.14%, lime 14.28%, cyan 21.42%, cyan 28.56%, blue 35.7%, magenta 42.84%, red 50%);
    background-size: 600vw 600vw;
    animation: slide 10s infinite linear forwards;
}
```

Don't worry about understanding all of the code above… just sit back and enjoy!!

To learn about more things you can do with animation, visit **this web page** (http://dojo.soy/se-css-animation). Have fun!

On the next card you'll learn how to make cool things happen when you hover the mouse cursor over things!

Step 10 Add hover effects

You can make your website more **interactive** by making cool stuff happen when you hover over things with the mouse cursor!

- Find your CSS rules for the `img` elements, or create some if you don't have any. Add in a border, and then add a new block of rules right below:

```
img {
  border: 2px solid White;
}
img:hover {
  border: 2px dashed Navy;
}
```

You've just used a special type of CSS block called a **pseudo-class**.

How does it work?

A **pseudo-class** is a bit different from a **class** that you create yourself. You can recognise it by the `:`.

Pseudo-classes come built in to HTML elements: you can add `:hover` style rules to any element, class, or `id` selector in your style sheet without needing to add anything extra in your HTML code.

- What do you think will happen? Check what pages on your website have pictures on them (add a picture if there aren't any!), then move your cursor over a picture to find out!

- Let's use this new `:hover` pseudo-class together with a CSS class to make links glow when you hover over them! Add a link to your web page and include an attribute to specify the class name. Remember, links are defined using the `<a>` tag, like so:

```
<p>
  <a class="niceLinks" href="https://en.wikipedia.org/wiki/Bird_conservation">Click here</a>
    to read about bird conservation on Wikipedia.
</p>
```

- Add the following code to your style sheet, then run your code to see your lovely links in action.

```
.niceLinks {
  text-decoration: none;
  color: #FFFAF0;
}
.niceLinks:hover {
  color: #00FF7F;
}
```

- Why not add the attribute `class="niceLinks"` to all of the links in your menu bar as well?

You can combine all of these tricks with animations too!

- Find the CSS block for the picture of the barn owl again (or whatever picture you were working on earlier). Add the following code to your style sheet file:

```
#owly {
  border-radius: 100%;
  width: 100px;
}
#owly:hover {
  animation-name: rollOver;
  animation-duration: 1s;
  animation-iteration-count: 1;
}
@keyframes rollOver {
  0% {
    transform: rotate(0deg);
  }
  100% {
    transform: rotate(-360deg);
  }
}
```

- Can you guess what will happen?
- Move the cursor over the picture to find out!

Challenge: make glowing rainbow links

- Can you use the `rainbowGlow` animation from the previous card to make the links in your menu keep changing colours when the cursor is hovering over them?

I need a hint

You can add `hover` effects directly to the `nav` menu like this:

```
nav ul li a:hover {
  animation-name: rainbowGlow;
  animation-duration: 1.5s;
  animation-iteration-count: infinite;
}
```

Or, if you want to make other links on your website flash rainbow colours too, you can add the animation to the `.niceLinks` class instead, like this:

```
.niceLinks:hover {
  color: #00BFFF;
  animation-name: rainbowGlow;
  animation-duration: 1.5s;
  animation-iteration-count: infinite;
}
```