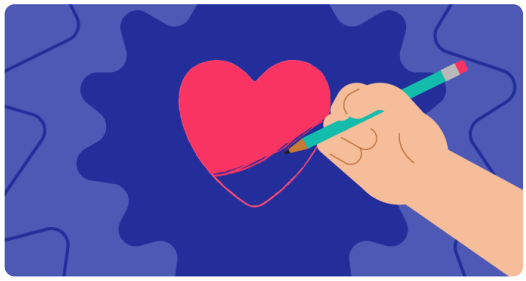
Projects

Anime expressions

Create and style a webpage for an anime drawing tutorial.



Step 1 Introduction

In this project, you will create and style a webpage for an anime drawing tutorial.

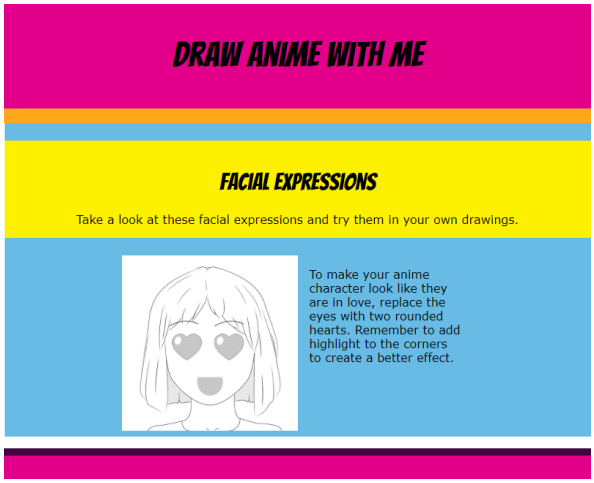
> **Hypertext Markup Language (HTML)** is used to structure a webpage. **Cascading Style Sheets (CSS)** describes exactly how a webpage should look. Without CSS, a webpage would look really boring.

You will:

- Use **HTML** tags to structure a webpage
- Use **CSS** styles to apply layouts, colour palettes, and fonts to your webpage
- Add images and text content to your webpage

> A **web designer** is someone who plans and designs webpages. There are hundreds of millions of active webpages, so if you use HTML with CSS, it can help your webpage attract people's attention.

Tip: Drag the scroll bar down to see the full webpage.



Step 2 Start your webpage

In this step, you will add a header and an introduction to your anime webpage.

In HTML you can type words directly into the code to make the words appear, unformatted, on the webpage.

Try it: Type some words into the left-hand side of the project below. The words you type automatically appear on the webpage to the right.

Tip: If you are using a mobile device, you might have to click on the pencil icon to write your text and then click on the refresh icon to view the output.

- What happens if you type multiple lines of text?
- Would you like to view a webpage that looked like this?

Open the **Anime expressions starter project** (https://trinket.io/html/2d58893157).

Your starter project contains some HTML that you will learn more about throughout the project.

To make your code easier to read, you can collapse the parts of it that you don't need right now.

Click on the small triangle next to line 3 to collapse the `<head>`.

Add a header

Typically, a webpage has three parts. A **header**, the **main** content, and a **footer**.

You can use comments to organise your code and help people to understand the code. Comments are ignored by the web browser.

Find the comment `<!-- The page header code goes here -->`.

I can't find the comment

Have you accidentally collapsed the `<body>` or another section of your webpage?

Click on the ▸ triangle to expand the code.

HTML documents contain **elements** including paragraphs, headings, and images. An element is typically made up of a starting tag, some content, and a closing tag.

A **tag** lets the browser know what kind of element it is. Tags start and end with angle brackets `<>`. The end tag also has a `/`.

Underneath the comment, find the `<header>` and `</header>` tags. Everything you add here appears in your webpage header and is styled as a header.

A `<h1>` tag is used to say that this content is the largest header on the page.

Add `<h1></h1>` **tags** inside your `<header></header>` tags.

Tip: When you add a start tag, the end tag is automatically added so you don't need to type it.

index.html

```
<body>
  <!-- The page header code goes here -->
  <header>
    <h1></h1>
  </header>
```

Tip: It's a good idea to add spaces at the beginning of lines to indent your code. In HTML, you don't need to add in the indents for the code to work, but it does make your code easier to read.

Add the text `Draw anime with me` between the two `<h1>` tags.

index.html

```
<body>
  <!-- The page header code goes here -->
  <header>
    <h1>Draw anime with me</h1>
  </header>
```

Test: Your webpage updates as you type. Some basic styling has been applied to the text because it is inside `<h1>` tags. The header text is bold and large.

Add the first section in your main content

Any main content should be placed between the `<main>` tags. On your webpage, the main content is broken down into **sections**.

Your webpage needs an introduction section. Add `<section>↙section>` tags between the `<main>` tags.

Tip: As you build your webpage, you will add other tags inside your section. Position your cursor between the `<section>` and `↙section>` tag, then press Enter on your keyboard to split the tags across multiple lines.

index.html

```
<!-- The main content for the webpage goes between the main tags -->
<main>
  <section>

  </section>
    <!-- The first drawing and instructions go here -->
```

You are now going to add a subheading within the section that you have just created.

Add the subheading tags `<h2>` between the `<section>` tags.

index.html

```
<!-- The main content for the webpage goes between the main tags -->
<main>
  <section>
    <h2></h2>
  </section>
    <!-- The first drawing and instructions go here -->
```

Now enter the subheading text `Facial expressions` between the `<h2>` tags. Your code should look like this:

index.html

```
<!-- The main content for the webpage goes between the main tags -->
<main>
  <section>
    <h2>Facial expressions</h2>
  </section>
    <!-- The first drawing and instructions go here -->
```

Notice how the text on your webpage is slightly smaller than the big heading above and has bold styling. This is because `<h2>` is a smaller heading than `<h1>`.

You are now going to add a paragraph of text as an introduction to your anime webpage.

Underneath your `<h2>` heading code, add the paragraph `<p>` tags.

index.html

```
<!-- The main content for the webpage goes between the main tags -->
<main>
  <section>
    <h2>Facial expressions</h2>
    <p></p>
  </section>
    <!-- The first drawing and instructions go here -->
```

Between the `<p>` tags, you need to add in this introductory text:

```
Take a look at these facial expressions and try them in your own drawings.
```

Tip: You can highlight the text above and then right-click (tap and hold on mobile) and choose 'Copy'. Then click between the `<p>` tags in your code and then right-click and choose 'Paste'.

Your code should look like this:

index.html

```
<!-- The main content for the webpage goes between the main tags -->
<main>
  <section>
    <h2>Facial expressions</h2>
    <p>Take a look at these facial expressions and try them in your own drawings.</p>
  </section>
    <!-- The first drawing and instructions go here -->
```

Test: You can now see that the text appears under the subheading and uses the default paragraph styling.

Well done! Your page now has a header, a subheading, and an introductory paragraph.

Save your project

Your project is saved automatically. Return to the starter link in the same web browser to see your changes.

I accidentally closed my web project

Click on the **starter project** (https://trinket.io/embed/html/2d58893157) link to open your project. Use the same web browser to see your changes.

If you have a Trinket account

Click the ‘Remix’ button to create a copy of the project in your Trinket account. This means you can save a copy of the trinket on your ‘My Trinkets’ profile.

Step 3 Add a facial expression

In this step, add the first drawing and instruction to your webpage.

First, create a section for each facial expression on the webpage.

Find the comment `<!-- The first drawing and instructions go here -->`.

Add in the `<section>↙section>` tags for your first drawing and instruction content.

index.html

```
33  <main>
34    <section>
35      <h2>Facial expressions</h2>
36      <p>Take a look at these facial expressions and try them in your own drawings.</p>
37    </section>
38
39    <!-- The first drawing and instructions go here -->
40    <section>
41
42    </section>
```

Your starter project contains images to use in this project. To include an image on a webpage, you need to know the filename. First, add an image called `love.png`.

Inside your new section, add an `<img>` tag to display an image. The `src` **attribute** gives the name of the image.

The `<img>` tag doesn't have an end tag.

index.html

```
39  <!-- The first drawing and instructions go here -->
40  <section>
41    <img src="love.png">
42  </section>
```

Test: The `love.png` image appears on your webpage.

Alternative (Alt) text is a description of an image and is important in accessible web design to describe images to people who are unable to see them. The text does not appear on the webpage but it is read aloud by screen readers.

Add the `alt` property to provide alternative text for people who cannot view the image.

You can copy the description of your image and paste it into your code: `The love facial expression.`

index.html

```
<!-- The first drawing and instructions go here -->
<section>
  <img src="love.png" alt="The love facial expression.">
</section>
```

Add a paragraph of text in `<p>↙<p>` tags to describe how to draw the love anime facial expression.

You can copy the paragraph and paste it into your code: `<p>To make your anime character look like they are in love, replace the eyes with two rounded hearts. Remember to add highlight to the corners to create a better effect.↙<p>`

index.html

```
<!-- The first drawing and instructions go here -->
<section>
  <img src="love.png" alt="The love facial expression.">
  <p>To make your anime character look like they are in love, replace the eyes with two rounded hearts. Remember to add highlight to the corners to create a better effect.</p>
</section>
```

The `<strong>` tag makes important text **bold**.

Add `<strong>` tags around the word 'love':

index.html

```
<!-- The first drawing and instructions go here -->
<section>
  <img src="love.png" alt="The love facial expression.">
  <p>To make your anime character look like they are in <strong>love</strong>, replace the eyes with two rounded hearts. Remember to add highlight to the corners to create a better effect.</p>
</section>
```

Test: The instructions appear below your image and the word **love** is in bold.

Step 4 Style your page

You have used HTML to add tags to your webpage. Now it is time to use CSS to add styles to your page. This step shows you how to change the colours, fonts, and layout on your webpage.

> **Cascading Style Sheets (CSS)** is the language that you use to tell the web browser exactly how your webpage should look, which includes the positioning, colours, and fonts. We call this the style.

Every **rule** in CSS is made up of two parts: the **selector** and the **declaration**.

The **selector** is the part of HTML that you want to style. In this example it is `h1`.

```
h1 {
    color: blue;
    font-size: 12px;
  }
```

The **declaration** is in curly brackets `{}`. It gives instructions of the styles that should be used.

```
h1 {
  color: blue;
  font-size: 12px;
}
```

Link the CSS file

The starter project includes CSS files, which contain a set of useful rules.

Unfold the `<head>` section of your code so that you can view the code inside it.

At the bottom of your `<head></head>` section, there are links to two CSS style sheets that are currently commented out so that they are ignored by the web browser.

Remove the `<!--` and `-->` arrows from the start and end of both lines of link code:

Before

index.html

```
<!-- Include CSS style file -->

  <!-- <link href="style.css" rel="stylesheet" type="text/css" /> -->
  <!-- <link href="candy.css" rel="stylesheet" type="text/css" /> -->
</head>
```

After

index.html

```
<!-- Include CSS style file -->

  <link href="style.css" rel="stylesheet" type="text/css" />
  <link href="candy.css" rel="stylesheet" type="text/css" />
</head>
```

Test: HTML elements have default browser styles that you have seen as you have written your HTML code.

Take a look at your webpage in the right-hand pane. Notice that the styles and layout of your output has now changed.

Tip: To collapse the `<head>` section after you have seen the change, click the arrow next to it.

Click on the `style.css` tab in Trinket to view the code in the CSS file. This CSS file contains all of the CSS for your project. You will find out about some key parts of this CSS file as you create your webpage.

When you add CSS styling to an **element**, it applies that styling to every single element on the page that has the same tag.

Find: Scroll down and find the rule that controls the style of the `<h2>`.

style.css

```
h2 {
  font: var(--title-font); /* Font style stored in the title-font variable */
  text-align: left; /* Align the text */
  padding: 1.5rem; /* Add some space all around the heading */
}
```

This rule states which font should be used, how the text should be aligned, and how much space should be around the header.

At the moment, the `<h2>` heading is aligned to the left.

Change the `text-align` property of the `h2` rule to `center`.

style.css

```
h2 {
  font: var(--title-font); /* Font style stored in the title-font variable */
  text-align: center; /* Align the text */
  padding: 1.5rem; /* Add some space all around the heading */
}
```

Test: Look at your webpage and make sure the 'Facial expressions' text is centred.

Debug: Check the spelling of the word `center`. HTML uses American English spelling.

Step 5 Style with classes

This step shows you how to add classes to customise the styles on your page.

If you want to apply styling to specific elements, you can create a **class** in a CSS file. You can then add a `class=` **attribute** to an element in your HTML code to let the browser know what styling should be applied.

The class styling overrides any element styling that has already been applied. Notice that the changes take place as you add the classes to your code.

Your CSS file has a custom CSS class called `border-bottom`. This class adds a thick, solid-coloured line border to the bottom of any HTML element that uses it.

Go to your `index.html` file and find your `header`.

Add `class="border-bottom"` after the word `header` in your `header` tag.

index.html

```
27  <body>
28    <!-- The page header code goes here -->
29    <header class="border-bottom">
30      <h1>Draw anime with me</h1>
31    </header>
```

Add the `border-top` class to your `footer` code to apply a thick border to the top of your footer.

index.html

```
55  <!-- Webpage footer -->
56    <footer class="border-top">
```

The `primary` class sets a contrasting background and text colour for most of the main content. The `secondary` and `tertiary` classes set additional colour combinations that look good with the colours in the `primary` class.

Add the `secondary` class to your `footer` code to apply a different colour background to your footer.

index.html

```
55  <!-- Webpage footer -->
56    <footer class="border-top secondary">
```

Add `class="primary"` to `<main>`.

index.html

```
<!-- The main content for the webpage goes between the main tags -->
<main class="primary">
```

Add `secondary` to `<header>`.

index.html

```
<!-- The page header code goes here -->
<header class="border-bottom secondary">
```

The `xcenter` class in your CSS file aligns items horizontally across the page.

Add `class="tertiary"` to the **first** `<section>` element.

Also, add `class="xcenter"` to the `<p>` in the same section.

index.html

```
<!-- The main content for the webpage goes between the main tags -->
<main class="primary">
  <section class="tertiary">
    <h2>Facial expressions</h2>
    <p class="xcenter">Take a look at these facial expressions and try them in your own drawings.</p>
  </section>
```

Webpages can be viewed on many different devices and should be **responsive** to each device. This means that if a user views your page on a mobile phone, it should respond to a smaller screen and if they view it on a desktop PC, it should respond to a larger screen.

CSS can change the layout on a webpage, as well being used to change colours, fonts, and borders.

Find the **second** `<section>`.

Add `class="wrap"` to the `<section>` tag.

index.html

```
<!-- The first drawing and instructions go here -->
<section class="wrap">
  <img src="love.png" alt="The love facial expression.">
  <p>To make your anime character look like they are in love, replace the eyes with two rounded hearts. You can add three more hearts inside for a fun effect.</p>
</section>
```

You can also add coloured borders in different styles to HTML elements. The `dashed-border` class in the style file creates a dashed border.

Add the `dashed-border` class to the `<img>`.

index.html

```
<!-- The first drawing and instructions go here -->
<section class="wrap">
  <img class="dashed-border" src="love.png" alt="The love facial expression.">
  <p>To make your anime character look like they are in love, replace the eyes with two rounded hearts. You can add three more hearts inside for a fun effect.</p>
</section>
```

You can make the corners of an element rounded with the `rounded` class.

Add the `rounded` class to the `<img>`.

index.html

```
<!-- The first drawing and instructions go here -->
<section class="wrap">
  <img class="dashed-border rounded" src="love.png" alt="The love facial expression.">
  <p>To make your anime character look like they are in love, replace the eyes with two rounded hearts. You can add three more hearts inside for a fun effect.</p>
</section>
```

Test: Drag the bar between the text editor and your webpage to make the webpage narrower.

The text should move below the image. This is the layout for users who view the webpage on a mobile phone.

Drag the bar back after you test it, so you can see the image and text side-by-side.

Step 6 Colours and fonts

In this step, you can try out different colour palette and font choices.

Now that you have started to add custom classes to your code, you may have noticed that colour has been added to the page. In CSS, you can use **variables** to create a colour palette for your webpage.

CSS variables start with two dashes: `--primary`.

Colours are specified using hexadecimal notation (hex) and begin with '#'. There are lots of websites where you can find hex colours to use.

Go to your `candy.css` file. This file sets the colour variables for the candy colour palette.

In the candy colour palette, the `--primary` variable is set to `#ebeaeb`, a pale grey.

candy.css

```
/* Candy colour palette and fonts */

:root {
  --primary: #ebeaeb;
  --onprimary: #625d61;
  --secondary: #f5bdd5;
  --onsecondary: #1d3d58;
  --tertiary: #b5a9b2;
  --ontertiary: #422215;
  --page: #ffffff;
  --onpage: #000000;
  --detail: #e697b9;
  --detail2: #415a89;
```

Tip: CSS uses different comment markers to HTML. Multiline comments start with `/*` and end with `*/`. The browser ignores code that is inside the comment markers. `/* Candy colour palette and fonts */` is an example of a CSS comment.

You can also use variables for fonts. The `--header-font` is set to `3rem 'Fredoka One', cursive;`

`3rem` means that this font should be three times the normal font size.

`'Fredoka One', cursive` means that the browser should use the 'Fredoka One' font if it can. If this font isn't available, the browser should use the **fallback font**, which is `cursive`.

Fonts for the web

Web designers carefully consider the font styles for their website.

The **three** most common categories of font are:

- Library fonts
- Web safe fonts
- Fallback fonts

Library fonts are typically imported from a third party library such as **Google Fonts** (https://fonts.google.com/). Companies will sometimes pay a fee to use a font as part of their website branding.

Web safe fonts are standard fonts that should be available through any web browser. However, you can never be 100% sure that this is the case. Here is a list of web safe fonts:

- Arial
- Verdana
- Helvetica
- Tahoma
- Trebuchet MS
- Times New Roman
- Georgia
- Garamond
- Courier New
- Brush Script MT

Fallback fonts are generic font families that are used to match the styling that the web designer would like to use. The main font families are:

- Serif - a font style typically used in print publishing, letters have tiny decorative edges called 'serifs'
- Sans-serif - a clean, screen readable font without the decorative edges
- Monospace - a font where each character uses the same width of space
- Cursive - a handwriting font
- Fantasy - a decorative font typically used for big headings

If a fallback font isn't listed then the web browser will use the browsers default font which is typically Times New Roman.

Find the variables that set the fonts for your webpage.

candy.css

```
--body-font: 1rem 'Verdana', sans-serif;
  --header-font: 3rem 'Fredoka One', cursive;
  --title-font: 2rem 'Fredoka One', cursive;
  --quote-font: lighter 1.5rem 'Chewy', cursive;
}
```

The `primary` colours are designed to be used the most in the main content of the page, followed by the `secondary` and then `tertiary` colours. This means that you can easily design new colour palettes and switch between them.

The starter project also includes a vivid colour palette file called `vivid.css`.

Find the `vivid.css` file.

Notice that the colour and font variables have the same name, but that the colours and fonts used are different in this colour palette.

vivid.css

```
/* Vivid colour palette and fonts */

:root {
  --primary: #68bbe5;
  --onprimary: #000000;
  --secondary: #e2008a;
  --onsecondary: #000000;
  --tertiary: #fdf100;
  --ontertiary: #000000;
  --page: #ffffff;
  --onpage: #000000;
  --detail: #ffa71a;
  --detail2: #41063c;

  --body-font: 1rem Verdana, sans-serif;
  --header-font: lighter 3rem "Bangers", cursive;
  --title-font: lighter 2rem "Bangers", cursive;
  --quote-font: lighter 1.5rem 'Chewy', cursive;
}
```

Go to `index.html` and change the CSS link code to link to the `vivid.css` file:

index.html

```
<!-- Include CSS style file -->

<link href="style.css" rel="stylesheet" type="text/css" />
<link href="vivid.css" rel="stylesheet" type="text/css" />
```

Test: Make sure your webpage now uses the brighter colours and different fonts.

Upgrade your project

If you have time, you can add more steps to the tutorial and customise the style to your preferences.

At the moment your tutorial only has one step! The starter project includes images for additional steps.

There are images called `happy.png` and `annoyed.png`.

You can use these descriptions or write your own:

- **Happy**: 'A happy expression in anime uses two thick curved lines for the eyes and a wide mouth to create a happy, laughing face.'
- **Annoyed**: 'An annoyed facial expression can be made by pointing the eyebrows down towards the centre. The eyes use a thick line with a semicircle underneath. The mouth is slightly curved.'

Create a new `<section>` for each new tutorial step. Look at the section you have already created to remind yourself of the HTML tags and CSS classes you need to use.

You can also change:

- The colour palette to `default` or `candy`.
- The colours in the colour palette CSS file. **Tip:** You can find **new hex colours** (https://rpf.io/colours) to use.
- Your webpage to use colour names, such as 'teal' and 'coral'.
- The size of the fonts used. `1rem` is the normal font size. `3rem` is three times as big.
- The border. Try different border effects in the `dashed-border` style or switch to the `solid-border` class.

Completed project

You can view the **completed project here** (https://trinket.io/html/b2ccbccbef).

Save your project

What next?

If you are following the **Intro to web** (https://projects.raspberrypi.org/en/pathways/web-intro) path, you can move on to the **Top five emojis** (https://projects.raspberrypi.org/en/projects/top-5-emoji-list) project. In this project, make a list of your favourite five emojis, with use of animation effects.

Published by **Raspberry Pi Foundation** (https://www.raspberrypi.org) under a **Creative Commons license** (https://creativecommons.org/licenses/by-sa/4.0/).
View project & license on GitHub (https://github.com/RaspberryPiLearning/anime-expressions)