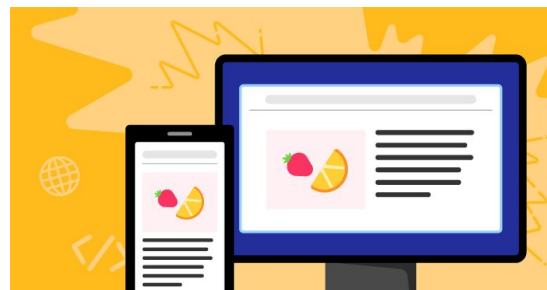


Build a web page

Introduction

What will you make:

In this project you will use the skills you have developed so far to create a webpage about something you want to share.



The **World Wide Web** or **web** is a vast collection of connected webpages. The web allows people to view content from all over the world and create their own webpages to share with others. Can you imagine what life was like before the web?

You will:

- Build a webpage to share information with others
- Create a webpage using multiple sections with a variety of HTML elements
- Use and customise or add CSS classes to style your page
- Type this rpf.io/favourite-thing on a new browser tab to explore the webpage.

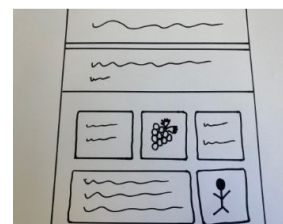


Project brief:

- Create a webpage about something you enjoy and want to share with others—this could be tips for pet owners, your favourite jokes, a game guide, useful advice, or a list of helpful websites.
- Your webpage should have different sections with a mix of content such as text, images, emojis, headings, quotes, lists, or links.
- Use a colour scheme, fonts, and CSS styles that look good together, and ensure the page is easy to read by using clear fonts and good colour contrast. You can also add your own CSS classes, include fun elements like animations or flip cards.
- Share your page with the Raspberry Pi community to inspire others.







1. [Your topic](#)




In this step you will decide what your webpage is about.



What do you want to share?

Think about the purpose of your webpage.
It could be:

-  An information page about a topic you are interested in
-  A collection of links to the best webpages on a topic
-  Information about a celebration or an event from your culture
-  Your favourite (polite!) jokes
-  A game walkthrough or other tutorial
-  Information about a pet or an animal

-  Some advice that you have found helpful
-  Tourist information about a place you have visited or would like to visit
-  Information about a medical condition or disability that you think more people should be aware of

Tip: Choose a topic you already know well, so you can focus on building your webpage instead of spending time researching.

Who is it for?

Who are you making your webpage for? You may find it helpful to think about the types of visitors you want to make your page for.

- How old will your visitors be?
- How much will they know about your topic?
- Is the page for other experts in your topic or is it for beginners?

Tip: If you make a page about a topic you know lots about, make sure you explain it clearly to other people.

Step 1:

Take out a pencil and an A4 sheet/paper/notebook and sketch out your webpage layout.

Step 2:

Open the starter project rpf.io/webpage-starter. The code editor will open in another browser tab.

Step 3:

Change the `<title>` element text to the title of your page.

The `<title>` tag is shown in the title bar of your web browser and should be relevant to your page. An example of a `<title>` tag is 'Awesome jokes about dinosaurs'.

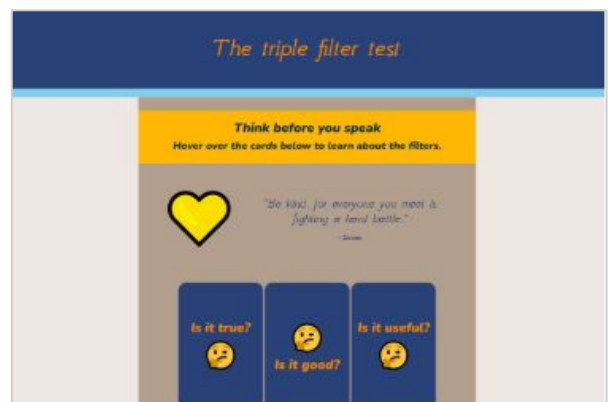
```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-
    <meta http-equiv="X-UA-Compatible" content="
    <!-- Title shown in web browsers-->
    <title>My website title</title>
```

2. Build and test

Now it's time to build your webpage.

Tip: Plan your project so you can create a basic webpage in the time you have available and then upgrade the project if you have time left.

Tip: Remember to test your project each time you add something. It is much easier to find and fix bugs before you make more changes.



Let's start creating your webpage—here's a quick reminder of the valuable skills you've built.

2.1 Colour palette

2.1.1 Colour palette variables explained

The **default.css** style sheet includes a palette of colour variables:

:root means that these variables are used for the whole page.

```
:root {  
  --primary: #bccad0;  
  --onprimary: #4f4e4e;  
  --secondary: #495054;  
  --onsecondary: #ffffff;  
  --tertiary: #747474;  
  --ontertiary: #ffffff;  
  --page: #ffffff;  
  --onpage: #000000;  
  --detail: #9ba8ae;  
  --detail2: #000000;  
}
```

The colour palette variables:

- **page** and **onpage**: contrasting colours used for the background of your page.
- **primary**, **secondary**, and **tertiary**: these can be used whenever you want different coloured **<section>** or **<div>** elements.
- **onprimary**, **onsecondary**, and **ontertiary**: used for text to contrast with the primary, secondary, and tertiary colours.
- **detail** and **detail2**: colours that can be used to add coloured highlights. **detail2** is used for **** text and borders.

The **page** colour scheme is used outside of the **<main>** content.

The **primary** colour scheme is used for content inside **<main>** unless you use a different class.

Use or personalise starter project colour palettes.

2.1.2 Starter project colour palettes

Step 1:

The starter project contains 20 colour palette CSS files.

The starter project is set up to use **default.css**, which is a greyscale colour palette.



Step 2:

Find: In the **<head></head>** element of **index.html**, find the line of code that links to **default.css**.

index.html

```
!-- Include CSS style file -->  
  
  <link href="style.css" rel="stylesheet" type="text/css" />  
  <link href="animation.css" rel="stylesheet" type="text/css" />  
  <link href="default.css" rel="stylesheet" type="text/css" />  
</head>
```

Step 3:

Change the filename in the link to use the CSS filename of the colour palette you want to use.

index.html

```
Include CSS style file -->

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

You can get a list of all the included colour palettes and their filenames here rpf.io/colour-palettes under starter project colour palette section.

2.1.3 Create your own custom colour palette

The starter project uses the **default.css** file to assign colours to the variables. The existing colours use a greyscale scheme.



Choose: Change the colour codes in **default.css** to the colours you would like to use in your webpage. Your webpage colours update as you change the colour codes.

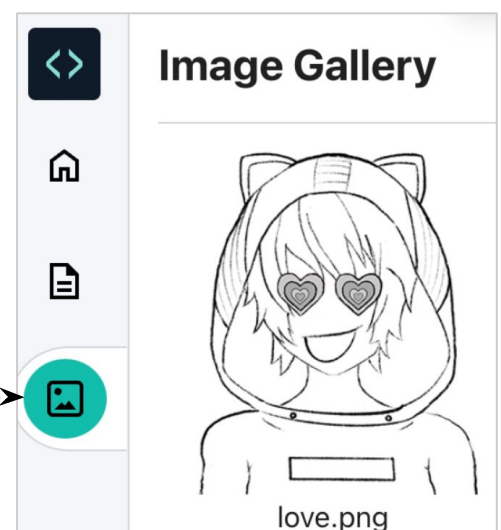
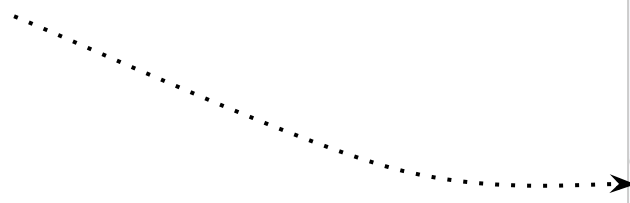
```
:root {
  --primary: #bccad0;
  --onprimary: #4f4e4e;
  --secondary: #495054;
  --onsecondary: #ffffff;
  --tertiary: #747474;
  --ontertiary: #ffffff;
  --page: #ffffff;
  --onpage: #000000;
  --detail: #9ba8ae;
  --detail2: #000000;
```

Create a colour palette from an image

2.1.4 Choosing an image from the library

Step 1:

Click on the 'Image gallery' icon.



Step 2:

Scroll through the image library and choose the image file you would like to use in your webpage.

Step 3:

Add your image to the `<main></main>` in `index.html` so that it appears on your webpage.

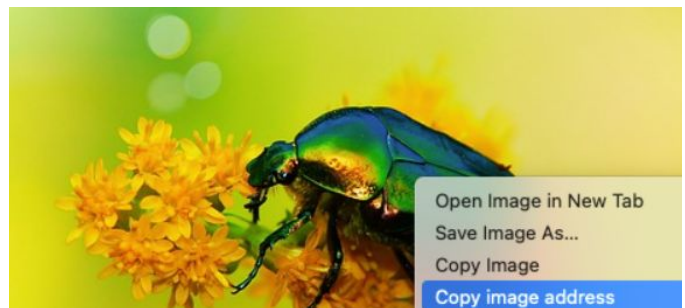
index.html

```
<!-- The main content for the webpage goes between the main  
<main>  
  Lorem ipsum dolor sit amet.  
    
</main>
```

2.1.5 Get colours from an image

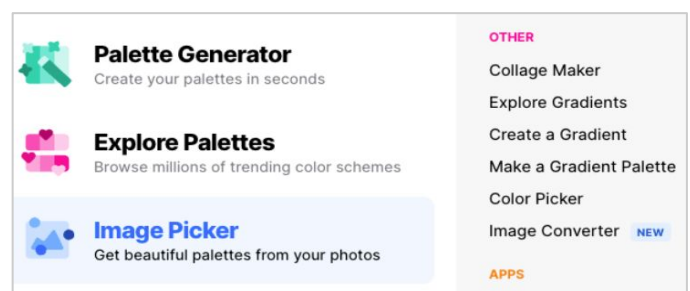
Step 1:

Right-click on an image in your webpage and select **Copy image address** or **Copy image link**.



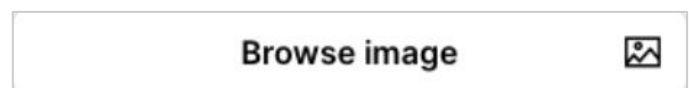
Step 2:

In a new browser window, go to rpf.io/coolers Select the 'Tools' menu then 'Image picker'..



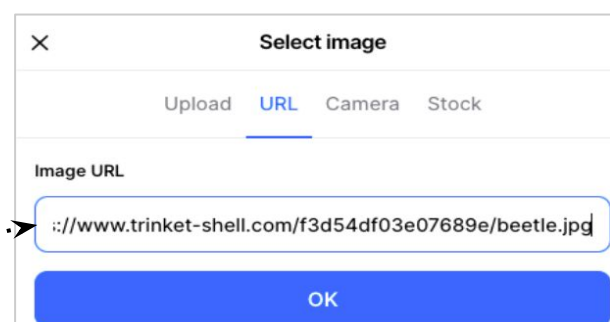
Step 3:

Click on the 'Browse image' button.



Step 4:

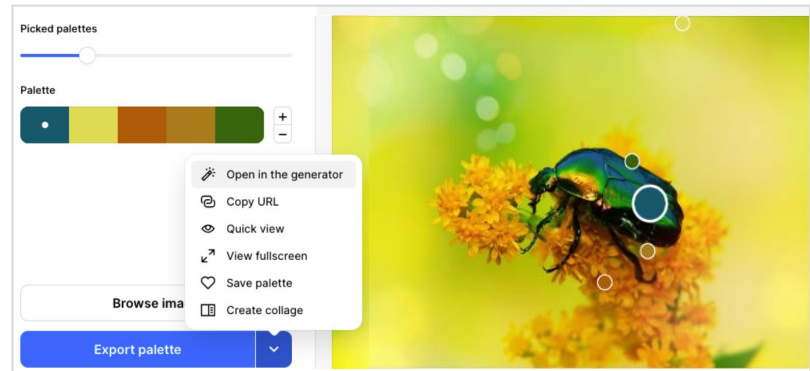
Click on 'URL' then paste the copied image address into the 'Image URL' box. Click 'OK'.



Sample palettes are created from your image. You can use the 'Picked palettes' slider to select which colour scheme you want to use.

Step 5:

When you are happy with the palette, click on the drop-down arrow of the 'Export palette' button and select 'Open in the generator'.



The final palette is shown. The coded letters and numbers are the hex codes for your chosen colours.



Step 6:

Update the variable values in your `default.css` file to use these new colours.

```

4  :root {
5    --primary: #08586B;
6    --onprimary: #4f4e4e;
7    --secondary: #E0DB54;
8    --onsecondary: #ffffff;
9    --tertiary: #AF5C08;
10   --ontertiary: #ffffff;
11   --page: #ffffff;
12   --onpage: #000000;
13   --detail: #AB7C1C;
14   --detail2: #38640D;
15 }

```

Customise colour palettes for style and accessibility.

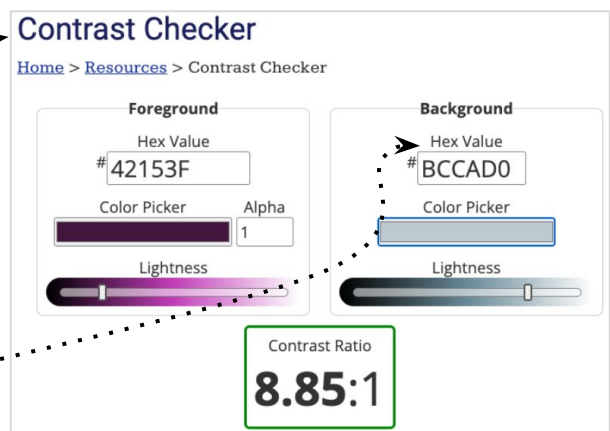
2.1.6 Find a contrasting colour

Step 1:

Open the Contrast checker website
rpf.io/contrast-checker.

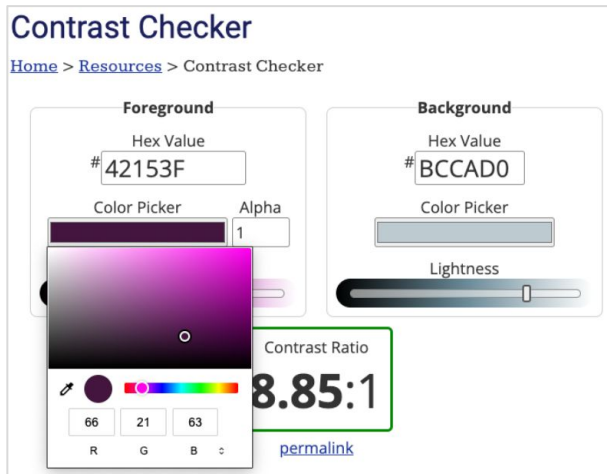
Step 2:

Under the 'Background' heading, enter the hex code for your background colour. `primary`, `secondary`, and `tertiary` are all background colours.



Step 3:

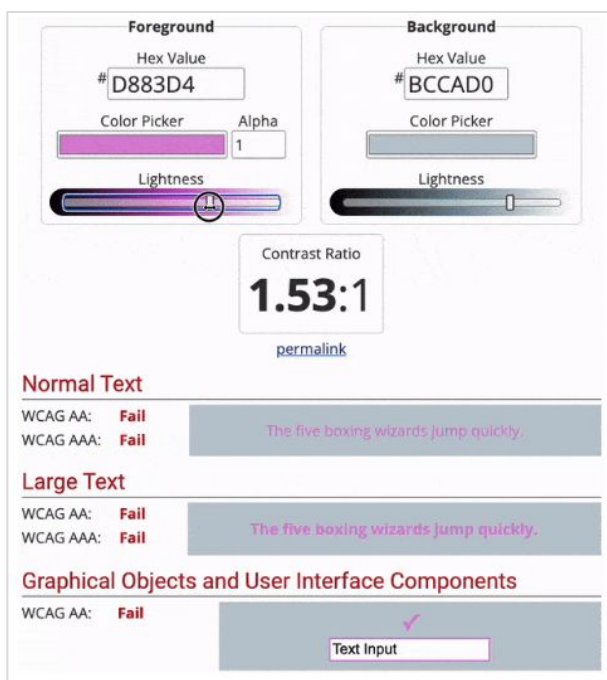
Choose: To the left of the 'Background' section is a 'Foreground' section. Use the colour picker to pick a colour that you think works well with your background colour.



Notice that you must choose a contrasting colour. If your background colour is dark, then use a light foreground colour.

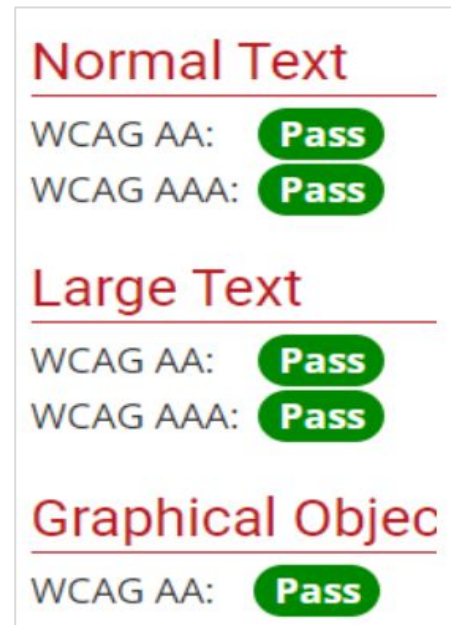
Step 5:

If the status for your colour is not showing as 'Pass', then you can slide the 'Lightness' bar until they do.



Step 4:

Check that you have chosen a contrasting colour by looking at the statuses below. They should all be 'Pass'.



Step 5:

Copy and paste the hex code for the contrasting colours into your colour palette. You need to find contrasting colours for:

- onprimary
- onsecondary
- ontertiary

2.1.7 Add extra colour variables

Step 1:

You can create additional variables in `default.css` to store more colours.

default.css

```

4  :root {
5    --primary: #08586B;
6    --onprimary: #4f4e4e;
7    --secondary: #E0DB54;
8    --onsecondary: #ffffff;
9    --tertiary: #AF5C08;
10   --ontertiary: #ffffff;
11   --page: #ffffff;
12   --onpage: #000000;
13   --detail: #AB7C1C;
14   --detail2: #38640D;
15   --highlight: #DC9110;
16   --onhighlight: #443C35;
17 }
```

Step 2:

You can create a class in `style.css` that uses your new colours.

style.css

```

8  .secondary {
9    background: var(--secondary);
10   color: var(--onsecondary);
11 }
12
13 .tertiary {
14   background: var(--tertiary);
15   color: var(--ontertiary);
16 }
17
18 .highlight {
19   background: var(--highlight);
20   color: var(--onhighlight);
21 }
```

Step 3:

Your new class can then be used in your `index.html` file at any time so that an element can use your new colour pairing.

index.html

```
<section class="highlight">
```

Use your colour palettes to set text and background colours.

2.1.8 Use primary, secondary, and tertiary colour classes

Step 1:

Use the `primary`, `secondary` or `tertiary` class to set the background and font colours. These classes use the colours set in the font palette at the `index.html`

style.css



```

<section class="wrap">
  <div class="primary">
    <p>Lorem ipsum</p>
  </div>
  <div class="secondary">
    <p>Lorem ipsum</p>
  </div>
  <div class="tertiary">
    <p>Lorem ipsum</p>
  </div>
</section>
```

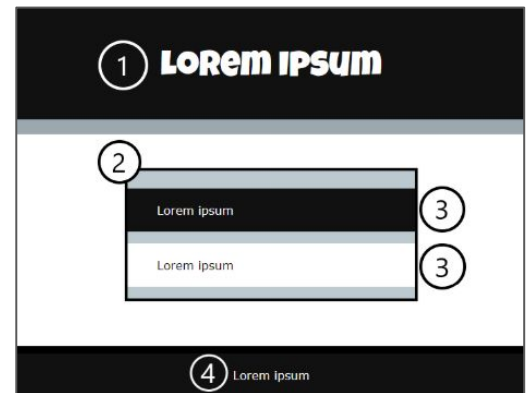

2.2 Structure

2.2.1 Overall page structure

There are four essential parts to a webpage that can be used to structure the layout.

These are:

1. The header `<header>`
2. The main content `<main>`
3. A section of content `<section>`
4. The footer `<footer>`



The **header** typically contains the title for the webpage.

The **main** is where you add all of the main content for your webpage.

A **section** is used to divide the content of the main part of your webpage into different areas. This is useful because different styling can be applied to each section.

The **footer** can be used for a fun greeting or for essential information like a contact or link to a feedback form. It is typically very small.

2.2.2 Add a full width section with title and text

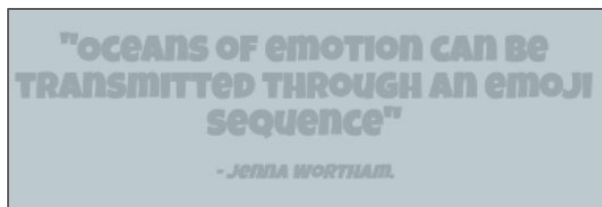
The section will take up the maximum width allowed, as set by the styling of the `<main></main>` tags.

index.html

```
<section class="wrap">
  <h2>Section title</h2>
  <p>Section text.</p>
</section>
```

2.2.3 Add a quote

```
<section class="wrap">
  <blockquote>
    <p>"Oceans of emotion can be transmitted through an emoji sequence"</p>
    <cite>- Jenna Wortham.</cite>
  </blockquote>
</section>
```



Websites can be viewed on many different devices and should be **responsive** to each device. This means that if a user views your website 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.

2.2.4 Using rem for sizes

Where possible, **rem** should be used to state the size of an element in a HTML webpage.

This is because **rem** works with the default size that the user's browser has specified. **1rem** is typically equivalent to 16px but a user might decide to set their **rem** to a larger or smaller size based on their needs and preferences.

When you use **3rem** for the size, you are stating that the element is three times the size of the default size.

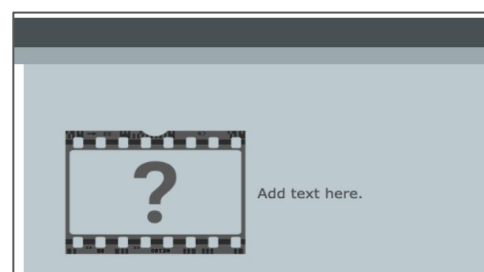
```
.bigfont {  
  font-size: 3rem;  
}
```

Using **rem** is great because it allows your webpage to be responsive to the needs of your user. CSS can also be used to responsively change the layout of the webpage.

CSS can also be used to responsively change the layout of the webpage.

2.2.5 Add side by side image and text

```
<section class="wrap">  
    
  <div>  
    <p>Add text here.</p>  
  </div>  
</section>
```



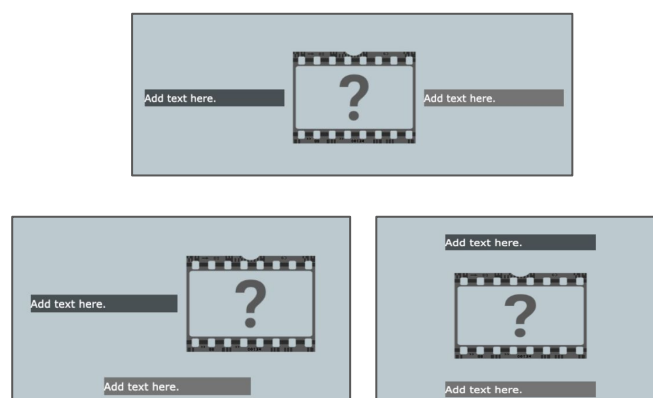
You can swap the order of the **** and **<div>** elements if you want the text to come first.

2.2.6 Add a wrapped section with regular width elements

Step 1:

Add or remove **<div>** and **** elements as needed. These elements will wrap if there is not enough space.

```
<section class="wrap">  
  <div class="secondary">  
    <p>Add text here.</p>  
  </div>  
    
  <div class="tertiary">  
    <p>Add text here.</p>  
  </div>  
</section>
```

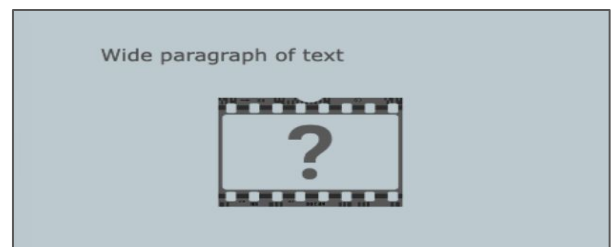


Step 2:

Use the **primary**, **secondary**, and **tertiary** styles to control the background and font colour for paragraphs of text.

2.2.7 Add a wrapped section with wide and narrow elements

```
<section class="wrap">
  <div class="wide">
    <p>Wide paragraph of text</p>
  </div>
  
</section>
```



2.2.8 Add large text tiles

Step 1:

You can use heading tags (`<h1>`, `<h2>`, and `<h3>`) to create large text headings in coloured tiles

Step 2:

Use the `tile` class to make sure your tiles are all the same height.

```
<main class="page">
  <section class="wrap">
    <div class="primary tile">
      <h1>LOREM</h1>
    </div>

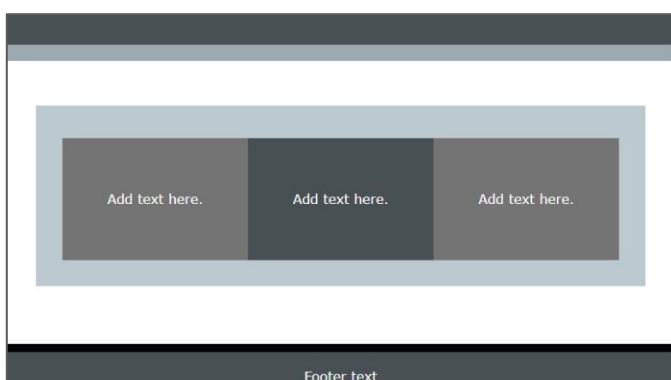
    <div class="secondary tile">
      <h2>LOREM<br>IPSUM</h2>
    </div>

    <div class="tertiary tile">
      <h3>Lorem<br>ipsum<br>dolor</h3>
    </div>
  </section>
</main>
```



Tip: You can adjust the `height` of the `tile` class in `style.css`.

2.2.9 Add wrapped tiles with text, text, text

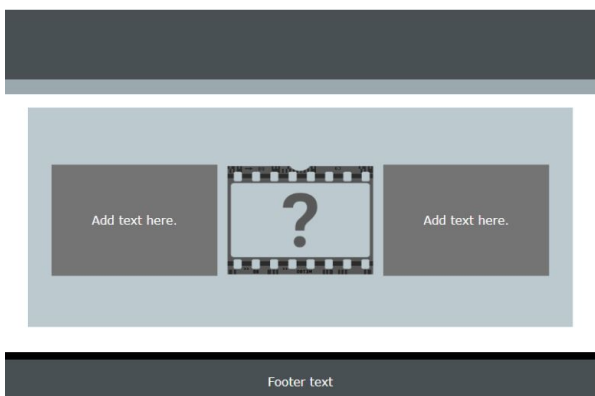


The code example creates three tiles of equal height. The text within the tile is centred on the x-axis and y-axis.

- `xcenter` places the text in the centre horizontally
- `ycenter` places the text in the centre vertically
- `tile` sets a fixed height for the `div` content

```
<section class="wrap">
  <div class="tertiary xcenter ycenter tile">
    <p>Add text here.</p>
  </div>
  <div class="secondary xcenter ycenter tile">
    <p>Add text here.</p>
  </div>
  <div class="tertiary xcenter ycenter tile">
    <p>Add text here.</p>
  </div>
</section>
```

2.2.10 Add wrapped tiles with text, image, text



The code example creates three tiles. The first tile contains text, the second tile is an image, and the third tile contains text. The two text blocks are of equal height using the **tile** class.

- **Xcenter** places the text in the centre horizontally
- **ycenter** places the text in the centre vertically
- **tile** sets a fixed height for the **div** content

```
<section class="wrap">
  <div class="tertiary xcenter ycenter tile">
    <p>Add text here.</p>
  </div>
  
  <div class="tertiary xcenter ycenter tile">
    <p>Add text here.</p>
  </div>
</section>
```

If you need to adjust the height of the text boxes, then you can change the CSS code.

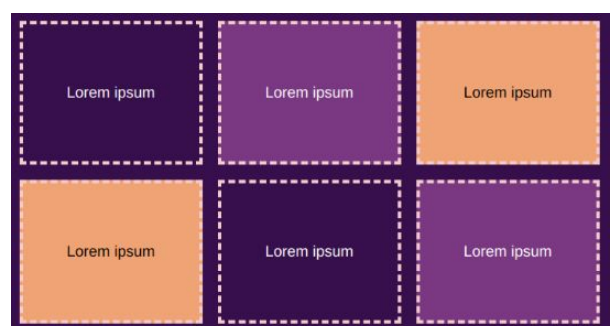
```
.tile {
  height: 9.4rem;
}
```

2.2.11 Add a gap between wrapped elements

Step 1:

You can use the **gap** property on the **wrap** class in **style.css** to create a horizontal and/or vertical gap between wrapped items. This can be useful when you use borders or shadows.

Note: The **gap** property is not supported by older web browsers.



Step 2:

Align and space your content to improve the look of your webpage.

```
/* Styles just for the .wrap class */

.wrap {
  /* Make content wrap over multiple rows */
  display: flex;
  flex-wrap: wrap;
  justify-content: center;
  align-items: center;
  box-sizing: border-box;
  gap: 1rem 1rem; /* horizontal and vertical gap */
}
```

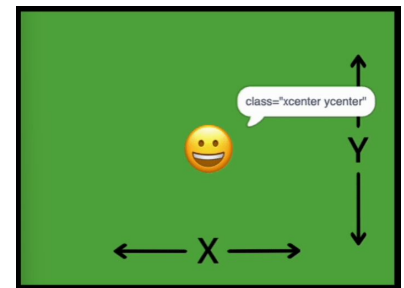
2.2.12 X and y centring

The **xcenter** class in your **style.css** file aligns items to the centre horizontally.

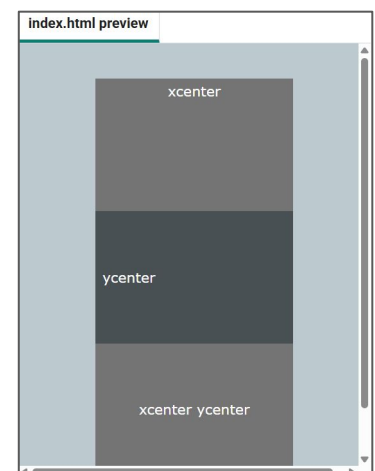
The **ycenter** class in your **style.css** file aligns items to the centre vertically.

Step 1:

Apply both the **xcenter** and **ycenter** classes, you align items to the centre both horizontally and vertically.

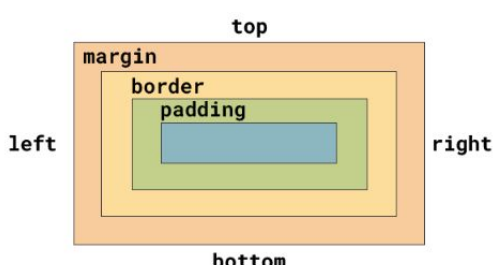


```
<section class="wrap">
  <div class="xcenter tile tertiary">
    <p>Lorem ipsum</p>
  </div>
  <div class="ycenter tile secondary">
    <p>Lorem ipsum</p>
  </div>
  <div class="xcenter ycenter tile tertiary">
    <p>Lorem ipsum</p>
  </div>
</section>
```



2.2.13 Padding and margins

The diagram below shows the **box model**. Web designers use this model to see which values they need to adjust to make their margins, padding, and borders the right size.



The **margin** property is the outermost area of the element.

The **border** nests inside the **margin**.

The **padding** nests inside the **border**.

The space in the centre shows the content within the element. The code below shows the settings for the **margin** and **padding** properties.

Step 1:

The code shows the settings for the **margin** and **padding** properties.

```
main {  
  background: var(--primary); /* Colour the background */  
  color: var(--onprimary); /* Colour the text */  
  margin: 0 auto; /* Centre if the browser is really wide */  
  min-width: 25rem; /* Don't let the content get too narrow */  
  max-width: 70rem; /* Don't let the content get too wide */  
  padding: 0;  
  padding-top: 0.5rem; /* Padding at the top */  
  margin-bottom: 1em; /* Gap before the footer */  
}
```

Step 2:

You can also specify which side of the content you wish to add margins, padding, and borders to.

```
main {  
  background: var(--primary); /* Colour the background */  
  color: var(--onprimary); /* Colour the text */  
  margin: 0 auto; /* Centre if the browser is really wide */  
  min-width: 25rem; /* Don't let the content get too narrow */  
  max-width: 70rem; /* Don't let the content get too wide */  
  padding: 0;  
  padding-top: 0.5rem; /* Padding at the top */  
  margin-bottom: 1em; /* Gap before the footer */  
}
```

2.3 Fonts and text elements

Choose fonts to use on your webpage.

2.3.1 Use fonts from the starter project

The included fonts are:

Spirax
Titan One
HANALEI FILL
MONOTON
Miltonian
Flavors
LUCKIEST GUY
Ranchers
FASTER ONE
Lobster
EATER
Finger Paint

Tip: If you have used one of the CSS colour palettes included in the starter project, a matching font has already been assigned. You can replace this font if you would like to use a different one.

Step 1:

Go to **default.css** or the filename of your chosen colour palette (example **festival.css**).

Step 2:

Replace the existing **header-font**, **title-font**, and **quote-font** with the name of your chosen font.

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

Step 3:

You can also add your font variables to existing classes or use them when you make your own classes.

```
.bigfont {  
    font-size: 3rem;  
    font: var(--header-font);  
}
```

2.3.2 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

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 uses the browser's default font, which is typically Times New Roman. Fantasy: a decorative font typically used for big headings

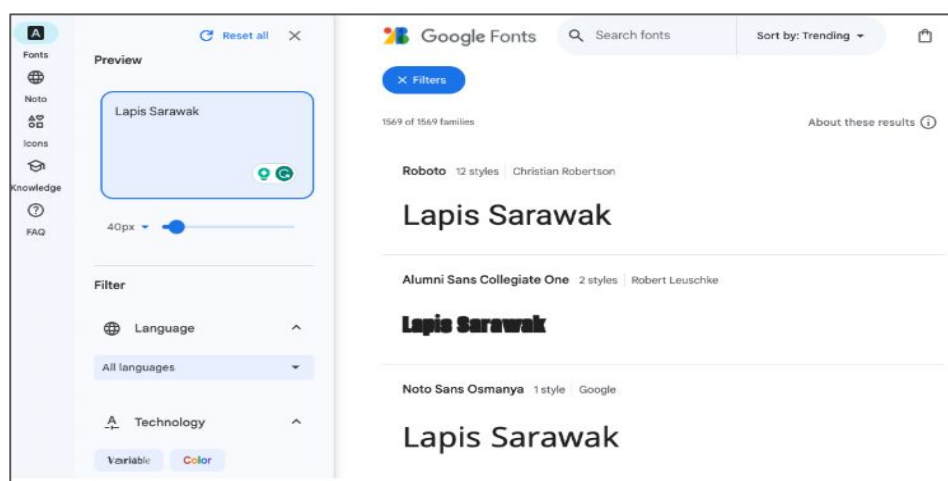
2.3.3 Add Google fonts

Google has a free online font library with over 1600 fonts to choose from. The site allows you to browse different fonts using your own sample text to help you find the right font. Google Fonts then gives you the **HTML** and **CSS** that you need to **link** or **import** the font into your website.

Find a font

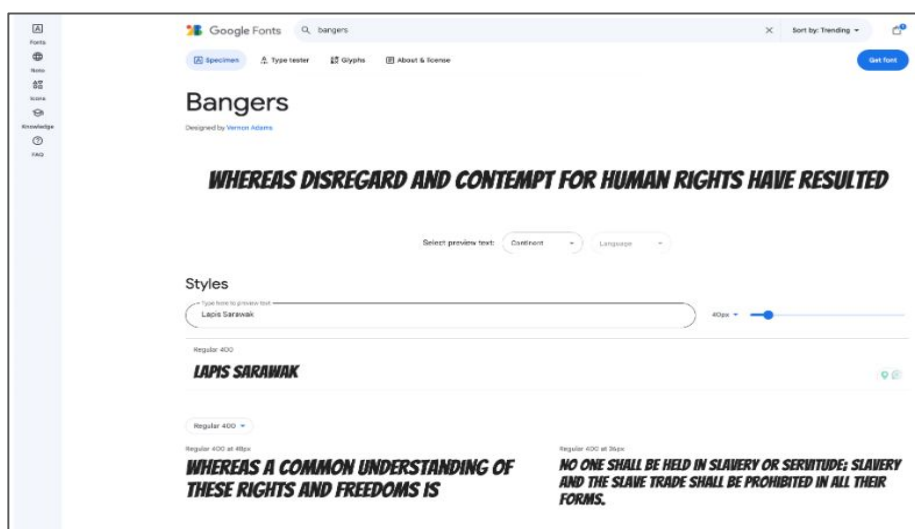
Step 1:

Open fonts.google.com and type some sample text in the **Preview** box.



Step 2:

There are many different search filters to use. You can search by language, or different font properties. Scroll down until you find a font you like. If you know the name of the font you want to use, type it in the main search box.



Notice that you can now see an example of the Bangers font that has been applied to the sample text.

Choose your font size

This example shows 40px.

Get the embed code

Step 1:

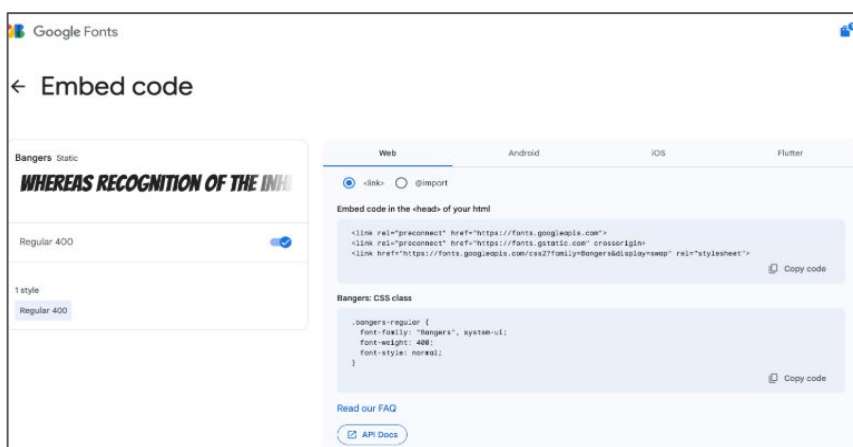
Click the 'Get font' button at the top right. You will see your selected fonts.



Step 2:

Click the 'Get embed code' button. You can either use the `<link>` method or the `<import>` method.

Link method



Step 3:

Copy and paste the HTML code inside the `<head>` tags in your HTML document.



You need to add in the correct CSS so that the web browser knows when to use this font.

Step 4:

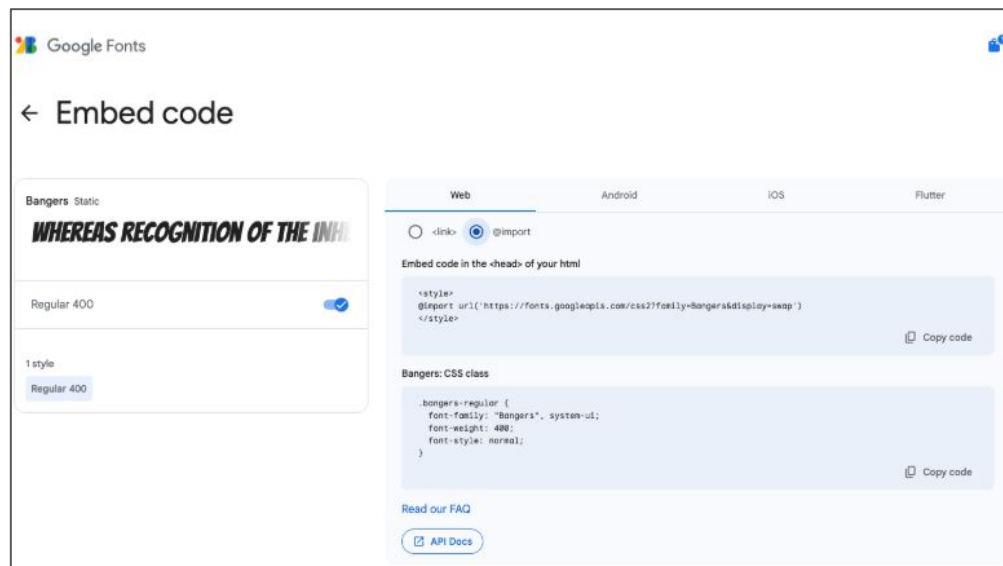
Go to your `default.css` file and find the font variables (it may also be the file containing the colour palette you have chosen, for example `fiesta.css`).

Step 5:

Add (or replace) the code for your chosen font. In our example, we use `Bangers, cursive;`.

```
--body-font: 1.1rem Verdana, sans-serif;  
--header-font: lighter 3rem 'Bangers', cursive;  
--title-font: lighter 2rem 'Bangers', cursive;  
--quote-font: lighter 1.5rem 'Bangers', cursive;
```

Import method



Step 1:

Open **style.css**.

Step 2:

Copy and paste the import code at the top. Make sure you add a semicolon ; to the end of the line when you add it to your CSS file.

```
@import url('https://fonts.googleapis.com/css2?family=Bangers&display=swap');
```

Step 3:

Open **default.css** and find the font variables (it may also be the file containing the colour palette you have chosen, for example **fiesta.css**).

Step 4:

Add (or replace) the code for your chosen font. In our example, we use **Bangers, cursive**;

```
--body-font: 1.1rem Verdana, sans-serif;  
--header-font: lighter 3rem 'Bangers', cursive;  
--title-font: lighter 2rem 'Bangers', cursive;  
--quote-font: lighter 1.5rem 'Bangers', cursive;
```


Usage of placeholder text while you work on your layout.

2.3.4 Add 'lorem ipsum' placeholder text

It is traditional to use nonsense Latin text as a placeholder on a webpage so that you can see what a layout looks like.

```
<p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Atque, officia libero! Quam  
perspiciatis necessitatibus repellat vel doloremque possimus ullam molestiae modi.  
Exercitationem nostrum, quibusdam enim velit iusto unde delectus molestiae?</p>
```

Insert text elements.

2.3.5 Add an ordered list with numbered items

The `` element creates an ordered list. Each `` list item inside an ordered list is numbered.



```
<section class="xcenter">  
  <ol>  
    <li>Lorem</li>  
    <li>Ipsum</li>  
    <li>Dolor</li>  
  </ol>  
</section>
```

B. Add an unordered list

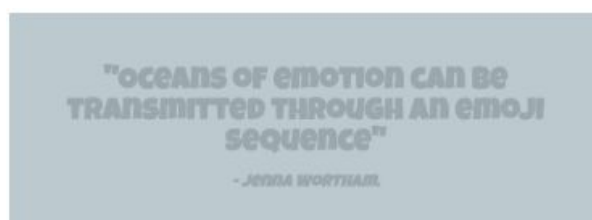
The `` element creates an unordered list. Each `` list item inside an unordered list has a bullet point.



```
<section class="xcenter">  
  <ul>  
    <li>Lorem</li>  
    <li>Ipsum</li>  
    <li>Dolor</li>  
  </ul>  
</section>
```

2.3.6 Add a quote

```
<section class="wrap">  
  <blockquote>  
    <p>"Oceans of emotion can be transmitted through an emoji sequence"</p>  
    <cite>- Jenna Wortham.</cite>  
  </blockquote>  
</section>
```



Style your text

2.3.7 Add paragraphs, headers, or large text

A `<h1>` tag is used to say that this content is the largest header on the page. The next header tag is `<h2>` used for a lower level title. To add paragraph text, use the `<p>` tag:

```
<header class="border-bottom secondary">
  <h1>Lorem ipsum</h1>
</header>

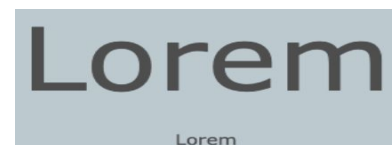
<main>
  <section>
    <h2>Lorem ipsum dolor.</h2>
    <p>Lorem ipsum dolor sit amet.</p>
  </section>
</main>
```



Tip: The starter project has custom styles in the `style.css` file to set the fonts used by the `<h1>`, `<h2>`, and `<p>` elements so they match the project fonts palette.

You can also use the `bigfont` and `hugefont` custom classes included in the starter project.

```
<p class="hugefont">Lorem</p>
<p class="bigfont">Lorem</p>
```



2.3.8 Centre text horizontally and vertically

Some items are automatically centred because of the style for their element.



Step 1:

use the `xcenter` and `ycenter` classes to centre other elements horizontally and vertically.

```
<section class="wrap">
  <div class="tertiary xcenter tile">
    <p>Lorem ipsum</p>
  </div>
  <div class="secondary ycenter tile">
    <p>Lorem ipsum</p>
  </div>
  <div class="tertiary xcenter ycenter tile">
    <p>Lorem ipsum</p>
  </div>
</section>
```

2.3.9 Use bold and italic text

Step 1:

Use the `` and `` tags to emphasise text. `` is used for important text that should be bold. `` is used for emphasised text that should have an italic (slanted) font.

Lorem ipsum **Lorem ipsum** *Lorem ipsum* ***Lorem ipsum***

```
<section>
  <p>Lorem ipsum <strong>Lorem ipsum</strong> <em>Lorem
</section>
```

Tip: You can combine `` and `` tags to create text that is bold and italic

2.4 Images

Insert an image from the starter project library.

2.4.1 Choose an image from the library

The starter file has a library of useful images. Click on the 'View and Add images' icon.

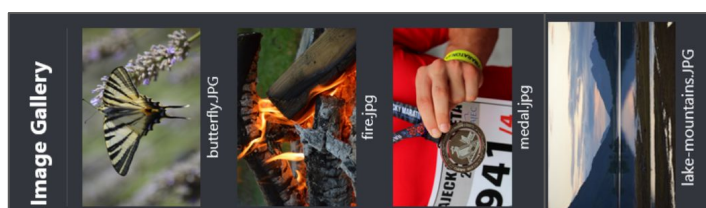


Step 1:

Scroll through the image library and make a note of the file name of an image you would like to use in your webpage.

Step 2:

Add your image to the `<main></main>` in `index.html` so that it appears on your webpage.



```
<!-- The main content for the webpage goes between the ma
<main>
  Lorem ipsum dolor sit amet.
  
</main>
```

2.4.2 Add alt text to images

Alternative (alt) text is a description of an image. Alt text is an important part of accessible web design as it describes 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.

Step 3:

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

```

```



Use an emoji as an image.

2.4.3 Choose an emoji

There are over 3,000 emojis available for you to use. They cover a wide variety of emotions, themes, and activities.

Tip: Emojis look slightly different on different devices, so someone else may not see exactly the same image as you. Some emojis are not supported on some devices and these appear as a square instead.

The emoji keyboard

Your device might have an emoji keyboard that you can use to select an emoji.

- **Mobile or tablet:** Press the emoji icon (this is typically a smiley face)
- **Windows:** Windows key + Full stop
- **Mac:** CTRL + CMD + Space
- **Linux:** Ctrl + Alt + E

Emojipedia

[Click to choose](#) some of the most popular emojis that you could use in your project. You can copy them from here.

If you can't find what you are looking for in the examples above, then you can visit [emojipedia](#) and explore a wider variety of emojis there.

2.4.4 Add a huge emoji

Anywhere you have an `` element, you can replace it with a huge emoji.

```
<p class="narrow, hugefont">  
  🍁  
</p>
```

2.4.5 Add a background image.

You can add a background image to appear behind other elements on your webpage.

Step 1:

Find this style declaration in `style.css`



You could also try uncommenting the other properties.

Step 2:

Remove the `/*` and `*/` comment markers and replace `name.jpg` with the name of your background image.

```
/* add a background image to body */  
  
body {  
  background-image: url('mybackground.png'); /* Uncommen  
  /*background-repeat: repeat;*/ /* Make the image repe  
  /*background-size: cover;*/ /* Make the image cover t  
}
```

Tip: You don't need to update the HTML because this style applies directly to the `<body>` tag. You could create a class to apply a background image to specific elements.

2.4.6 Add a transparency effect

Step 1:

Use the **transparent** class to make an element partially transparent so you can see the content behind it.

```
<div class="transparent">
  <p>Add text here</p>
</div>
```

```
/* Add a transparent effect */
.transparent {
  opacity: 0.95;
}
```

Tip: Adjust the **opacity** value for the **transparent** class in **style.css**: **0** is completely transparent and **1** is not at all transparent.

Style your images with rounded corners or borders.

2.4.7 Add rounded corners

Step 1:

To make the corners of an element rounded, you can use the **rounded** class.

```
<div class="rounded">
  <p>Add text here.</p>
</div>
```

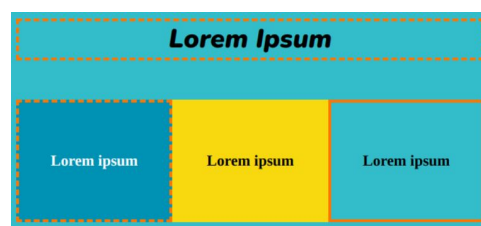
```
.rounded {
  border-radius: 1rem;
}
```

Tip: You can adjust the **border-radius** of the **rounded** class in **style.css**.

2.4.8 Add solid or dashed borders

Step 1:

Use the **solid-border** or **dashed-border** class to add a solid or dashed border around a **<section>** or **<div>**. The borders use the **detail2** colour.



```
<section>
  <h2 class="xcenter dashed-border">Lorem Ipsum</h2>
</section>

<section class="wrap">
  <div class="secondary dashed-border xcenter ycenter tile">
    <h3>Lorem ipsum</h3>
  </div>
  <div class="tertiary xcenter ycenter tile">
    <h3>Lorem ipsum</h3>
  </div>
  <div class="primary solid-border xcenter ycenter tile">
    <h3>Lorem ipsum</h3>
  </div>
</section>
```

Tip: You can adjust the **border** values for the **solid-border** and **dashed-border** classes in **style.css**.

2.4.9 Add a shadow effect

Step 1:

You can use the **shadow** CSS class to add a drop shadow effect to HTML elements, such as **<section>**, **<div>**, ****, and **<blockquote>**. The text 'Lorem ipsum dolor sit amet.' on a pink background with a green drop shadow. This example adds a shadow effect to a **<blockquote>** element.

```
<main class="page">
  <section class="wrap">
    <blockquote class="secondary shadow"><p>Lorem ipsum
  </section>
</main>
```

Step 2:

You can adjust the properties of the **shadow** class in **style.css** to create different shadow effects.

```
.shadow {
  box-shadow: 5px 5px 3px 0px #888888; /* right and bot
  /*box-shadow: 5px 5px 4px 2px var(--detail);*/
}
```

Tip: Try to add colour to your shadows. Use your detail colours **var(--detail)** or **var(--detail2)** to create coloured shadow effects.

2.5 Animations

2.5.1 Create a flip cards.

Flip cards help people interact with a webpage. They also allow you to provide additional information in a small space or hide information that shouldn't be immediately visible. Some common examples of animated flip cards are revision cards, business cards, social media contact cards, and product cards.

A flip card has four main parts:

- The card itself
 - The flip control
 - The front card face
 - The back card face.

Step 1:

Add this code to **index.html** to position your card on the webpage:

```
<div class="card"> <!-- The card -->
  <div class="card-content"> <!-- To control the flip -->
    <div class="card-face"> <!-- The front card face -->
      <!-- Add your front card content here including images -->
    </div>

    <div class="card-face flipme"> <!-- The back card -->
      <!-- Add your back card content here including images -->
    </div>
  </div>
</div>
```

The flip card uses four CSS classes:

- **card**: sets the size of the card
- **card-content**: controls the flip timing, perspective, and effect
- **Card-face**: hides the face when it is flipped to the back
- **flipme**: flips the second card face 180 degrees on the y-axis so it is flipped in the opposite direction to the front card face

The **card-content** class rotates 180 degrees on the y-axis when hovered over, which means the two card faces switch position

Customise the card faces

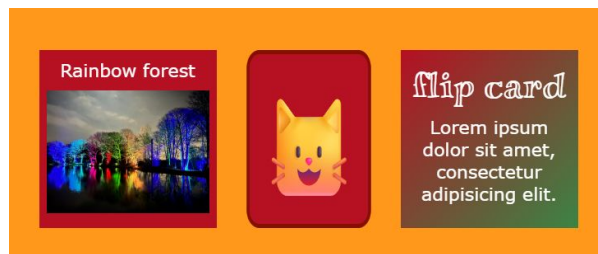
You can add text (including emojis), images, quotes, or lists to your card faces.

You can apply one or more style classes, such as:

- **Rounded**
- **xcenter, ycenter**
- **gradient1, gradient2**
- **dashed-border, solid-border**
- **primary, secondary, tertiary**

An example on how to do this is shown below:

```
<div class="card">
  <div class="card-content">
    <div class="card-face gradient1 rounded shadow">
      
    </div>
    <div class="card-face flipme gradient2 rounded ycenter shadow">
      <h2>Lapis Sarawak</h2>
      <p>A cake baked in layers to make colourful patterns.</p>
    </div>
  </div>
</div>
```



Use animations to change the appearance or position of elements on your page over time.

2.5.2 Use starter project animations

CSS animations are used to grab people's attention and make a website engaging without slowing down how long it takes to load a webpage. The animations work by changing one or more properties of an element over a period of time. Examples of animations are hover effects, loading images, text animation, particle effects, and animated images.

You can use HTML and CSS to create animations that change elements on a webpage.

A CSS **@keyframes** rule can be set to change over time. You can change colour, position, size, rotation, and many other properties.

@keyframes control how the element should look when a percentage of the running animation is complete.

The starter project contains an **animation.css** file with custom-made animations that can be used to bring your content to life.

The starter animations are:

- **spinme**
- **bounceme**
- **scaleme**
- **rollmeleft**
- **rollmeright**
- **movemeleft**
- **Movemeright**

Step 1:

Add the animation as a class to the object you want to animate.

```
<section class="xcenter">
  <h2 class="scaleme">Animations</h2>
</section>

<section class="xcenter">
  <p class="bigfont rollmeleft">🐶</p>
  <p class="bigfont bounceme">🐶</p>
  <p class="bigfont rollmeright">🐶</p>
</section>

<section class="xcenter">
  <p class="bigfont movemeleft">🐶</p>
  <p class="bigfont spinme">🐶</p>
  <p class="bigfont movemeright">🐶</p>
</section>
```

2.5.3 Customise starter animations

Each of the starter animations in the **animation.css** file has an **animation** property group that states how the animation should run:

The **animation** line in the **spinme** example is broken up into:

- **rotate-center**: the name of the animation
- **linear**: the animation timing (linear is the same playing speed throughout, other examples are **ease**, **ease-in**, and **ease-out**)
- **8s**: the duration of the animation in seconds
- **2**: the number of times the animation should run (can be **infinite** for continuous running)

```
.spinme {
  animation: rotate-center linear 8s 2;
  display: inline-block;
}
```

Step 1:

Change any of these values to alter the animation. Another way to customise the animation is to adjust the **@keyframes** rule. **@keyframes** control how the element should look when a percentage of the running animation is complete.

Step 2:

In the **rotate-center** animation used by **spinme**, the animation rotates the object from 0 degrees at the start of the animation (**0%**), to 360 degrees at the end of the animation (**100%**).

```
@keyframes rotate-center {
  /* The spin me animation code */
  0% { /* Rotate from 0 to 360 degrees */
    transform: rotate(0);
  }
  100% {
    transform: rotate(360deg);
  }
}
```

Step 3:

Animations can have specific styles applied at other percentage points during the animation run. For example, the **scale** animation has specified points at 0%, 20%, 40%, 60%, and 80%.

```
@keyframes scale {
  /* The scale animation code */
  0% {
    transform: scale(1, 1);
  }
  20% {
    transform: scale(1.1, 1.1);
  }
  40% {
    transform: scale(1.2, 1.2);
  }
  60% {
    transform: scale(1.1, 1.1);
  }
  80% {
    transform: scale(1, 1);
  }
}
```

Step 4:

The animation can have more than one style changed at each point. For example, the **bounce** animation changes the size and y-coordinate to create a realistic bounce effect.

```
@keyframes bounce {
  /* The bounce animation code */
  0% {
    transform: scale(1, 1) translateY(0); /* Sta
  }
  10% {
    transform: scale(1.1, 0.9) translateY(0); /*
  }
  30% {
    transform: scale(1, 1) translateY(-6rem); /*
  }
  50% {
    transform: scale(1, 1) translateY(0); /* Mov
  }
}
```

Step 4:

You can change colour, position, size, rotation, and many more properties if you change the **@keyframes** code.

2.6 More style

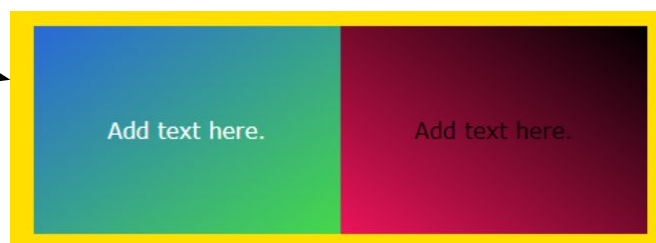
Add more style to any elements on your webpage.

2.6.1 Add rounded corners- refer page number-23.

2.6.2 Add solid or dashed borders- refer page number-23

2.6.3 Add a gradient

The **gradient1** and **gradient2** styles provide different gradient effects



Tip: To change the colours and direction of the gradient, adjust the **background-image** values for the **gradient1** and **gradient2** classes in **style.css**.

```
<div class="gradient1">
  <p>Add text here</p>
</div>
<div class="gradient2">
  <p>Add text here</p>
</div>
```

```
.gradient1 {
  background-image: linear-gradient(
    to bottom right,
    var(--secondary),
    var(--detail)
  );
  color: var(--onsecondary);
}

.gradient2 {
  background-image: linear-gradient(
    to top right,
    var(--tertiary),
    var(--detail)
  );
  color: var(--ontertiary);
}
```

2.6.4 Animate part of a header or the paragraph text

Step 1:

The `` tag can be used to change part of your text without starting a new line. You can use the `` tag to add an animation to part of the text.

Step 1:

You can also use `` to change the colour or strength of text.

```
<h1><span class="movemeleft">L</span>orem ipsum</h1>
```

```
<p>Lorem ipsum dolor sit amet, liberoconsectetur adipisicing elit. <span class="tertiary scaleme">Atque</span> officia! </p>
```



2.6.5 Add a shadow effect

Step 1:

You can use the `shadow` CSS class to add a drop shadow effect to HTML elements, such as `<section>`, `<div>`, ``, and `<blockquote>`.

The text 'Lorem ipsum dolor sit amet.' on a pink background with a drop shadow. This example adds a shadow effect to a `<blockquote>` element.

```
<main class="page">
  <section class="wrap">
    <blockquote class="secondary shadow"><p>Lorem ipsum
  </section>
</main>
```

Step 2: You can adjust the properties of the `shadow` class in `style.css` to create different shadow effects.

```
.shadow {
  box-shadow: 5px 5px 3px 0px #888888; /*box-shadow: 5px 5px 4px 2px var(--de
}
```

The text 'Lorem ipsum dolor sit amet.' on a pink background with a green drop shadow.

Tip: Try to add colour to your shadows. Use your detail colours `var(--detail)` or `var(--detail2)` to create coloured shadow effects.

Step 3: You can add new CSS classes whenever you want to create a new style. Make sure you give the style a sensible name.

Try to make sure your style is reusable and just contains properties that you want to use together.

You can create your own class to make a new style

2.6.6. Add a CSS class

This `photo` class creates a printed photo style that can be applied to an image.

```
<section>
  
</section>
```



```
.photo {
  border: 1px solid #D0D0D0; /* Add a solid border */
  width: 14rem;
  height: 15rem;
  background: #ffffff;
  padding-top: 1rem;
  padding-left: 1rem;
  padding-right: 1rem;
  padding-bottom: 3rem;
  box-shadow: 8px 8px 10px 4px #888888; /* Add a box shadow */
  transform: rotate(3deg);
}
```

Test: Show someone else your project and get their feedback. Do you want to make any changes to your webpage?

Tip: Now is a good time to add comments to your code to make it easier to debug and upgrade. Comments also help other people read your code.

2.6.7 Comment your HTML and CSS code

Comments help to explain the code. They are written in different ways for HTML and CSS.

A comment in a HTML file starts `<!--` and ends `-->`.

A comment in a CSS file starts `/*` and ends `*/`.

index.html

style.css

```
/* This is a CSS comment */
```

2.7 Debug

You might find some bugs in your HTML or CSS that you need to fix. Here are some common bugs.

My webpage isn't displaying correctly

2.7.1 Incorrect tags or properties

Step 1:

Carefully check the spelling of your HTML tags, attributes, and classes. Incorrect tags can mean that the tag text gets displayed on your webpage instead of controlling the layout. This example incorrectly uses 'image' instead of 'img'. **** is the correct HTML tag.

```
<image class="bordered-box" src="happy.png" alt="An outline of an anime style girl with a happy facial expression."/>
```

It is also incorrect to have spaces in tags, so the example is incorrect.

2.7.2 Mismatched tags or brackets

Step 1:

Make sure that tags with an open and close tag such as **<div>** and **</div>** are correctly matched and nested.

This HTML is incorrect because there is no closing **</div>** tag.

```
< h1>Lorem ipsum</h1>
```

```
<section>  
<div><p>Lorem ipsum</p>  
</section>
```

```
<div><p>Lorem ipsum</div></p>
```

Tip: If your HTML is incorrect, then sometimes a web browser will work out what you meant. You should still make sure your HTML is correct as incorrect HTML might cause problems later. Incorrect HTML also makes it difficult for screen readers to understand your page.

Further debug questions

2.7.3. My image isn't displayed

Step 1:

Check carefully that the name of your image in the **src** property matches the file name. Make sure the use of capital letters is the same. For example, 'myimage.png' is **not** the same as 'myimage.PNG'.

This HTML does not display an image saved as 'happy.PNG':

```

```

2.7.4 My font isn't displayed

When you want to use a new Google Font, you need to make sure you have:

- Added the `<link>` from Google Fonts to the `<head>` section of your webpage
- Updated the font variables in `default.css` or the style sheet that you are using for your colour and font palette

This example adds the 'Bangers' font and uses it for headers and titles, but not for the body:

Index.html

```
<!-- Import fonts from Google -->
<link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link href="https://fonts.googleapis.com/css2?family=Bangers&display=swap" rel="stylesheet">
```

default.css

```
--body-font: 1.1rem Verdana, sans-serif;
--header-font: lighter 3rem 'Bangers', cursive;
--title-font: lighter 2rem 'Bangers', cursive;
```

2.7.5 My link to a webpage doesn't work

Step 1:

Remember that the `<a>` (anchor) tag is used to create a link to another webpage (not the 'link' tag, which is used to link to resources such as fonts).

Step 2:

Also check that you have the correct web address (URL) for the `href` property.

The part of a web address after the domain name (such as 'projects.raspberrypi.org') is case sensitive, so you need to make sure the capital letters match.

This example uses correct HTML to link to a webpage that opens in a new browser tab:

```
<a href="https://projects.raspberrypi.org/en/raspberrypi/web-intro" target="_blank">Make a
webpage like this!</a>
```

You might find a bug not listed here. Can you figure out how to fix the bug?

We love hearing about your bugs and how you fixed them. Use the **Send feedback** button at the bottom of this page if you found a different bug in your project.

2.8 Share

Now is the time to share your webpage so that it can receive visitors.

Note: Remember to check that you haven't included any personal information in your webpage.

Why not show your project to one of your friends?

Why not invite your friends to create a project? Let them know how you had fun.