Hyperiondev

TASK

CSS II

Visit our website

Introduction

WELCOME TO THE SECOND CSS TASK!

This task will look at combining the styles considered previously in order to create a slightly more complicated web page which will set you up for the Capstone Project where we will be building a fully functional website.

Get in touch

Connect for support

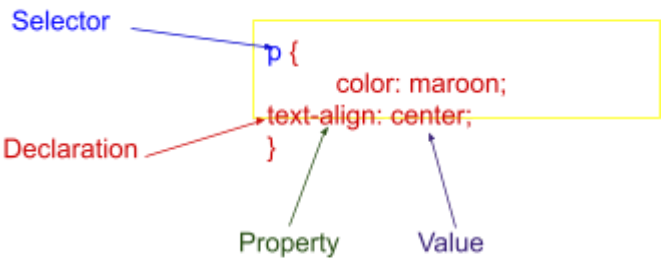
Remember that with our courses, you're not alone! You can contact an expert code reviewer to get support on any aspect of your course.

The best way to get help is to login to Discord at **https://discord.com/invite/hyperdev** where our specialist team is ready to support you.

Our team is happy to offer you support that is tailored to your individual career or education needs. Do not hesitate to ask a question or for additional support!

CSS SELECTORS

As you learned in the previous task, CSS follows the following syntax:
A style sheet consists of a selector and a declaration.



- The **selector** indicates which HTML element you want to style.
- The **declaration** block contains one or more declarations separated by semicolons. A declaration always ends with a semicolon and is surrounded by curly braces.
- Each declaration includes a **property** and a **value**, separated by a colon.

We have thus far only worked with one type of selector. The CSS below should look familiar. Here the selector is always an *element*.

```
p {
   color: red;
   font-family: Arial, Helvetica;
   background-color: blue;
}
body {
   text-align:center;
}

/* selectors for paragraphs and body */
```

However, you can also use *class* and *ID selectors*. A class selector is used when the selector describes the rule for all elements that have a *class attribute with the same name defined*. An ID selector describes the style of an element with a specific id attribute defined.

Open *example.html*. Notice that in this file we have several elements for which either the class or id attribute has been defined. For example, notice how there are several `<a>` elements that have the class attribute set to “moreButton” as shown below.

```
<a href="" class="morebutton">more</a>
```

However, there are also `<a>` elements that do not have a class attribute specified.

```
<li><a href="contact.html">contact</a></li>
```

Therefore, instead of creating a CSS with an element selector ('a') we could create a rule that is specific to a class selector. See an example of this below. When you use a class selector, the name of the class will always be preceded by a dot, '**.**'. The style rule below will cause all elements where the class attribute has been set to 'class="moreButton"' to have bold text.

```
.moreButton {
        font-weight: bold;
    }
```

In a similar manner, you could create a stylesheet that uses ID selectors. ID selectors start with a hash, '#', as is shown in the example below.

```
#mainNav {
        font-family: cursive;
    }
```

The rule above would cause the text of the element where the id attribute equals "mainNav" to be cursive.

Although you can have many elements that have a class attribute with the same value, an ID name must be unique in the document!

NAVIGATION BARS

By this time, you should have experienced using the lists as your navigation bars (the menu where a user can navigate to different pages on your website). You can style these lists using CSS to create navigation bars. For example, below is a bullet-point list of all the pages on the website.

```
<ul>
  <li><a href="index.html">Home</a></li>
  <li><a href="shop.html">Shop</a></li>
  <li><a href="contact.html">Contact Us</a></li>
  <li><a href="about.html">About Us</a></li>
</ul>
```

We can style this into a list with no bullets using **`list-style-type`** below. We also set **`margin`** and **`padding`** to zero to reset any default settings.

```
ul {
  list-style-type: none;
```

```
  margin: 0;
  padding: 0;
}
```

Play around with the other styles in `list-style-type` to see what works best for you.

USING CSS GRIDS FOR LAYOUT

Extra resource

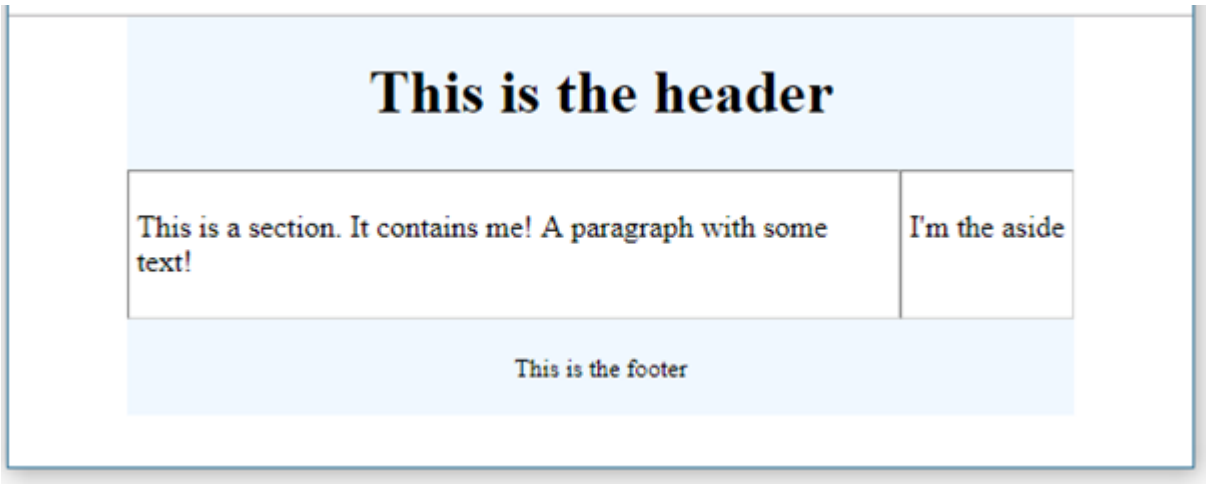
`<header>`, `<footer>`, `<aside>`, etc. elements are new sectioning elements used with HTML5. HTML5 is the latest evolution of the standard that defines HTML. Learn more about HTML5 **here**. Find out more about the `<article>` element and how it compares to `<div>` elements **here**.

Getting the layout of elements correct is one of the trickiest aspects of CSS. You will be required to do some research in this regard to complete this task successfully. Feel free to use whatever resources you like, but **Introduction to CSS layout** and the **CSS layout cookbook** are excellent places to find help.

Assume that we have created the following HTML:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <title>Position example</title>
    <link rel="stylesheet" href="example_style.css" />
</head>
<body>
    <div id="container">
        <header><h1>This is the header</h1></header>
        <section>
            <p>This is a section. It contains me! A paragraph with some text! </p>
        </section>
        <aside><p>I'm the aside</p></aside>
        <footer><p><small>This is the footer</small></p></footer>
    </div>
</body>
</html>
```

And we want this HTML to be displayed in the browser as shown in the image below:



It would be difficult to get this layout using only relative, absolute, static positioning. Instead, we can use a CSS grid template to achieve this layout.

A CSS grid is like a table that is designed to make it easier to position elements on a webpage. The grid usually contains 12 columns.

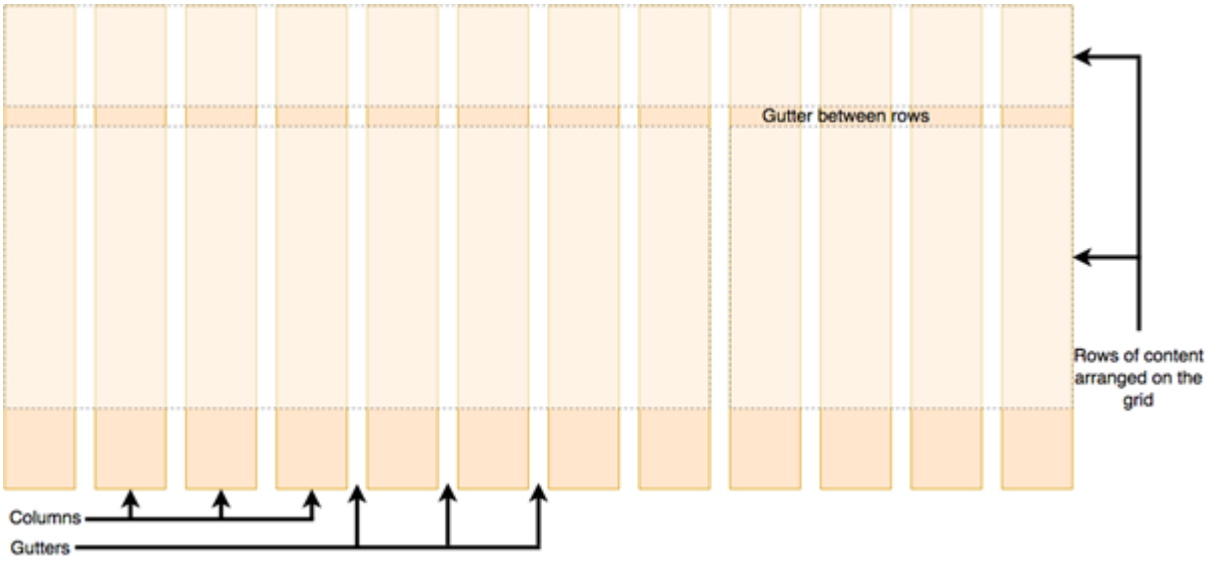


Image source:
https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Grids#A_CSS_Grid_grid_framework)

The position of an element is described in terms of which row it is in and how many columns it takes up. To illustrate, look at the CSS rules below.

```
body {
    width: 80%;
    margin:auto;
}
#container {
    display:grid;
}
header {
    grid-column: 1/13;
    grid-row: 1;
    background-color: aliceblue;
    text-align: center;
}
section{
    grid-column:1/9;
    grid-row:2;
    padding:4px;
    border: 1px inset lightgrey;
}
aside {
    grid-column: 9/13;
    grid-row: 2;
    padding: 4px;
    border: 1px inset lightgrey;
}
footer{
    grid-column: 1/13;
    grid-row: 3;
    background-color:aliceblue;
    text-align: center;
}
```

We want the `<header>` element to span across the full width of the whole container/grid. Thus, we specify that we want it to start at the first line (1) of the grid and end at the last line (13) of the grid. To put the `<header>` element to be at the top of the grid, put it in the first row of the grid.

Place the `<section>` element in the second row of the grid. The element is eight columns wide, so the section element starts at the first line of the grid and ends at the 9th line.

Take note:

As a software developer, it is obviously important to be able to write code/markup that is correct and runs to produce the desired outcome. As a professional web developer, more than this is needed though. You also want to be sure that your code is written in such a way that it is easy to debug, update, and maintain by yourself or by teams of developers. Readability is therefore very important. Your code/markup should be easy to read.

In this regard, note the following about the values you assign to class and id attributes: You can name a class whatever you like but it is best practice to use meaningful names. For example, “moreButton” is a good example of a name for a class because it is descriptive. “a1” would not be a good name for a class because it is not descriptive.

Also remember to use tools like **Beautify** to improve the readability of your CSS.

CASCADE: SPECIFICITY

Remember that the word “Cascade” in the term “Cascading Style Sheets” basically has to do with the order in which rules are applied.

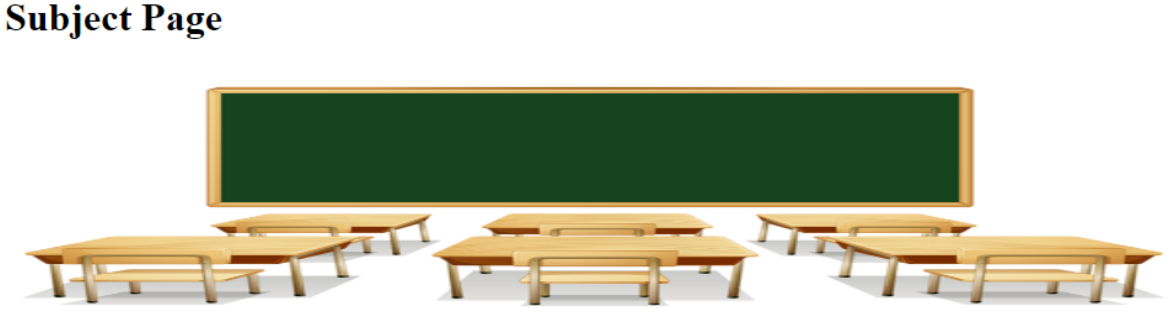
Another important rule to remember is that *the more specific a rule is, the higher its precedence*. For example, in a stylesheet that uses element selectors, class selectors and ID selectors, *element selectors are the least specific* (because they could match the most elements in a page) whereas ID selectors are the most specific. Therefore, ID selectors will be applied over class selectors and element selectors. See *example2.html* to clarify this concept further.

WHAT WILL WE BE DOING?

In this task, we are going to style a page which advertises different subjects to take at a high-school level. We recommend using external CSS wherever possible, but you may use inline and internal as needed. All of the required HTML elements have been provided for you, but feel free to add more if you so wish!

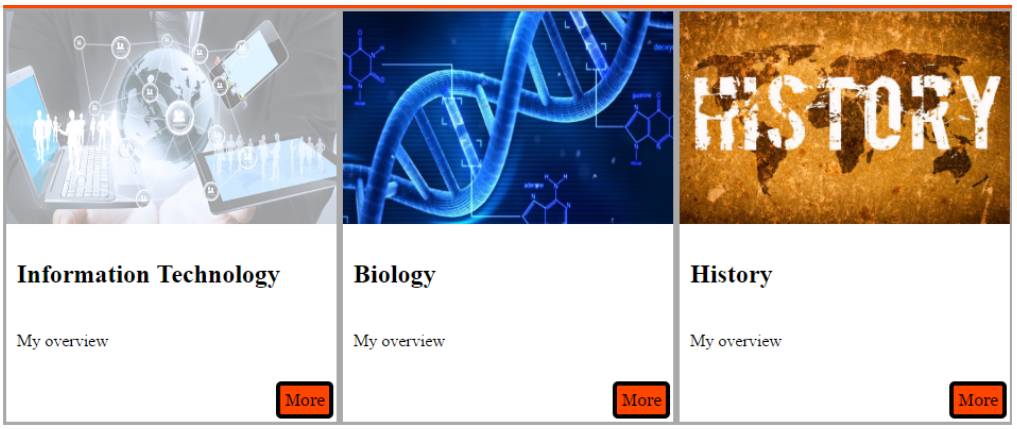
REQUIREMENTS

There will be a main heading and an image at the top of the page, before the navigation bar, as shown below:

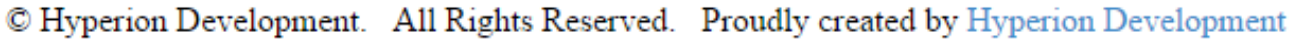


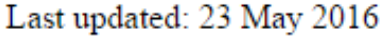
The webpage will have a navigation bar which will have links to the homepage, one per subject, and a contact page. These do not need to be functional, but the supposed links should still exist.

We will offer three subjects to choose from, namely *Information Technology*, *Biology*, and *History*. These subjects will be encapsulated in *article* elements. Each article will consist of a heading (the name of the subject), a picture relating to the subject, and a small paragraph which gives a brief overview of the subject. There will be a details button which will react when a mouse hovers over it, but will not provide extra functionality.

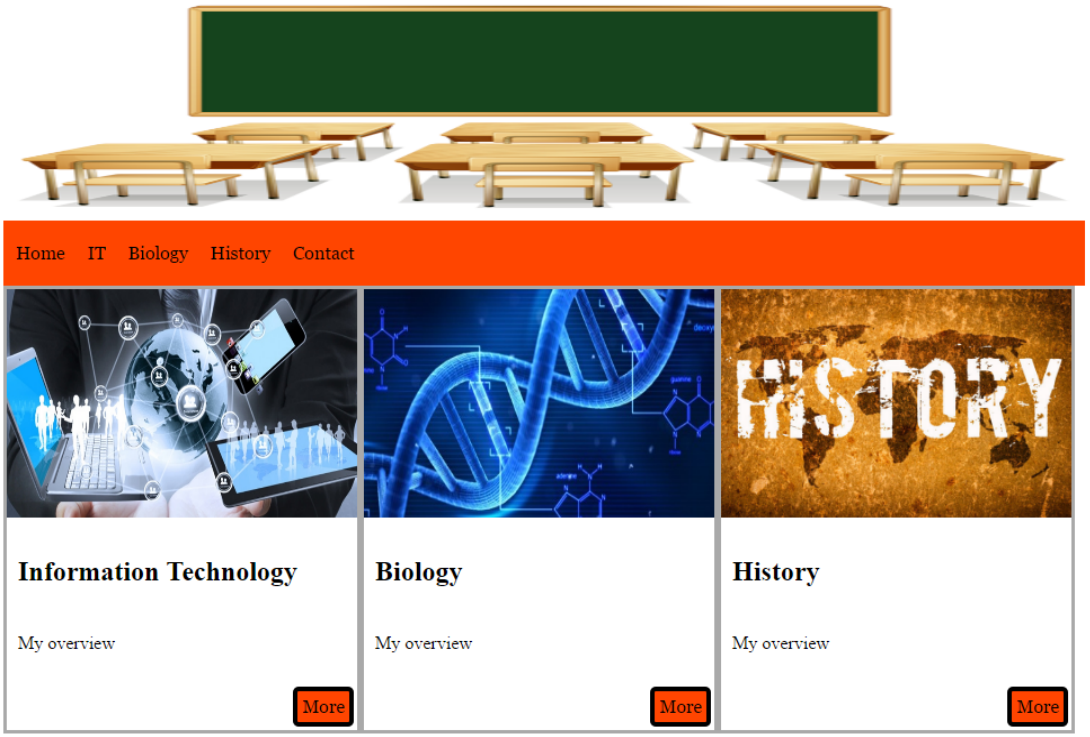


In the footer element, there will be all the extra information which is usually contained in web pages.





You have been provided with all of the necessary HTML and images (already linked). After applying the various CSS rules outlined in the instructions, your webpage should look something like the one below:



Open the HTML file to view what the page looks like without CSS having been applied!

Take note:

Here are two more cool CSS tricks:

1. If you would like to apply a certain style rule to an element when it is in a certain state (e.g. if you hover over it) you can do this as shown below:

```
/* Any button over which the user's pointer is hovering */
button:hover {
  color: blue;
}
```

In this example, 'hover' is a pseudo-class (a keyword that describes the state of the selector). To see a list of pseudo-classes, see **here**.

2. If you would like to apply a certain rule to an element that is a descendant (child) of another element, you can do so as shown below:

```
li li {
  list-style-type: circle;
}
```

The rule above will make all list items that are descendants of other list items have a circle as a bullet point. To see more about using a descendant combinator, see **here**.

SPOT CHECK 1

Let's see what you can remember from this section.

1. What is the difference between an element selector, an ID selector, and a class selector? What is the hierarchy among these?

2. In terms of the grid layout, how would we write the CSS to get a heading 1 to start at column 2, span to column 11 and sit in row 1?

Instructions

Open **example.html** in VSCode and read through the comments before attempting these tasks.

Compulsory Task

Follow these steps:

- Create a CSS file and link it in your HTML file (**task.html**).

The Heading and Navigation

- Centre your whole page, using 50% as your width. (Hint: look at margin types).
- Set your main picture (of the classroom) to span across the screen, and make it align with the rest of your page.
- Make your unordered list have no bullets, be aligned in the middle, and have a different font (of your choice). (Hint: look at **list-style-type**). Give them no padding.
- Set the background colour of your navigation to “orangered”, and give it 3px padding. (See additional reading ‘CSS notes for professionals’ Chapter 9: Padding)
- When links on the page are hovered over, the text should change to white. (Hint: use a pseudo-class)
- All links should have no text-decoration and be black.

The Articles

- Each article should be floated left, have a width of 32.4%, a height of auto, and a border which is dark grey and solid. (See additional reading ‘CSS notes for professionals’ chapter 10 and 14)

- The articles should have an additional property of display:inline-block.
- The paragraphs in the articles should be justified and have a padding of 10 px. (Hint: look at html children).
- The pictures should take up the full width of the articles and have a height of 200px. This will distort the images slightly, but don't worry about this.
- When the pictures are hovered over, the opacity should be reduced to half. (See additional reading 'CSS notes for professionals' Chapter 19: Opacity)
- The "more" buttons should look the same (in terms of colours) as they do in the picture of the example web page, entitled **Subject page**, on page 10, earlier in this task. This is left to you to work out how to do.
- These buttons should be floated right, given padding and a border, as well as a margin of 3px.
- When these buttons are hovered over, the background colour should change to black and the text to orangered.

The Footer

- The text should be centred.

If you are having any difficulties, please feel free to contact our specialist team **on Discord** for support.

Completed the task(s)?

Ask an expert to review your work!

Review work

Things to look out for:

1. Make sure that you have installed and set up all programs correctly. You have set up **Dropbox** correctly if you are reading this, but **Visual Studio Code** may not be installed correctly.

Rate us

Share your thoughts

HyperionDev strives to provide internationally-excellent course content that helps you achieve your learning outcomes.

Think that the content of this task, or this course as a whole, can be improved, or think we've done a good job?

Click here to share your thoughts anonymously.

SPOT CHECK 1 ANSWERS:

1. An element selector is used in CSS to add style elements using an element's tag (e.g. `h1` or `p`). A class selector is created with a dot (`.`) and can be added as an attribute to an element in the HTML file. What's special about a class is that it can be used repeatedly over multiple elements. An ID selector is created with a hash (`#`) and can be added as an attribute to an element in the HTML file. Each element can only have one ID and each page can only have one element with that ID. Element selectors are executed first, then ID selectors, then class selectors.
2.

```
#container {
      display:grid;
}

h1 {
      grid-column: 2/11;
    grid-row: 1;
}
```