Lab 2 - CSS Basics
The goal of this lab is to help you familiarize with CSS. We will be using minimal HTML boilerplate and focus mainly on CSS. Each section of this instruction corresponds to one <article> within provided HTML, your goal is to add edit CSS for provided content.

TIP: To comment out the line of code in VS code you can use **ctrl + /** (or **cmd + /** on MacOS)
TIP: You can repeat same CSS rule multiple times and only the last defined one will apply, e.g.
In this case the final margin will be set to 30px:
```
.class {
   margin: 20px;
   margin: 10px;
   margin: 30px;
}
```

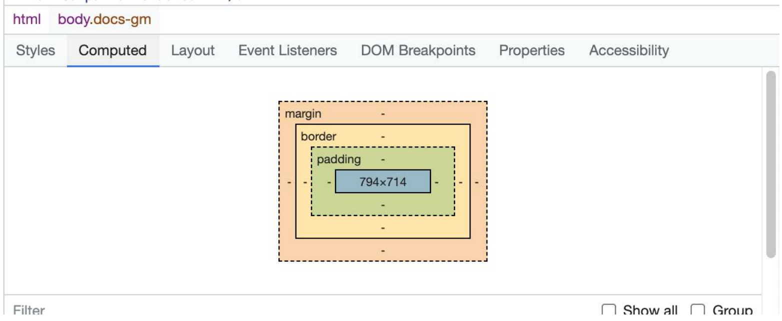
I. Selectors, combinators and pseudo-classes - https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Selectors
1. Selectors
   1.1. Using type selector add margin, padding and border to each <article> on the page, e.g.
```
margin: 20px 0;
padding: 20px;
border: 1px solid black;
```
   1.2. Using class selector change font color of element with class .style-me1 to red
   1.3. Using class selector change font color of element with id #style-me2 to blue
   1.4 Using specific selector change font color to the element that has both class of .style-me1
 and id of #style-me2 to purple.

TIP: Note that in normal situations having duplicate ids is a mistake - there should not be duplicate ids on the same page. We are doing it here only for the simplicity of the exercise.

2. Combinators
   2.1 Using descendant combinator style all the p elements that are descendants of .my-descendants-are-styled. Set their background to red.
   2.2 Using child combinator style all the p elements that are children of .my-children-are-styled. Set their background to red.

II. Box model and common properties
In this section you will be editing CSS rules for the .box class. Use an inspection tool

throughout this exercise - take a look at both Styles and Computed tabs.
1. Set margin and padding on an element with class .box



1.1 Try with different margin and padding syntaxes - 4-value syntax, 2-value syntax and 1-value syntax. Use pixels as units for margin and padding but do not hesitate to try out other types of units.
2. width and height
2.1 Set width and height using different dimension units: px, rem, vh. vw
2.2 Observe how the third box retains height styled via style attribute. Usually vh units are associated with height and vw with width.
2.3 Try resizing the browser window while the height depends on the vw and vh units. What can you observe?
3. border
3.1 Set border on the element, try different border styles and colors
3.2 Make the border 10px thick and observe how thickness of the border moves elements in each direction.
4. box-sizing
4.1 Test different values of box-sizing. How is box-sizing affecting width and height of the element?
5. text-overflow
5.1 Make text overflow outside of the element by setting width and height to smaller values so the text in a second box goes over the box borders.
5.2 Set overflow CSS rule for .box class - test different values of overflow. What does they affect the box?
6. outline
6.1 Set a wide outline (10px) on the .box. Observe that setting the outline does not move elements around like with borders.
7. Border-radius
7.1 Set border-radius on the .box element. Start by trying out different values express in pixels e.g. 20px
7.2 Setting border-radius to 50% together with width=height will make the element look like a circle. If the width is different from height then the element will become an ellipsis. Not that this is just a visual effect of the border and when you inspect the HTML element is still an rectangle.
8. background image
8.1 Set background image using background property. Image to use is located in ./assets/images/img1.png file
8.2 Set background-size to `contain`. Do not hesitate to try other values.
8.3 Set background-repeat to `no-repeat`. Do not hesitate to try other values.

8.4 Set background-position `to center`. Do not hesitate to try other values.

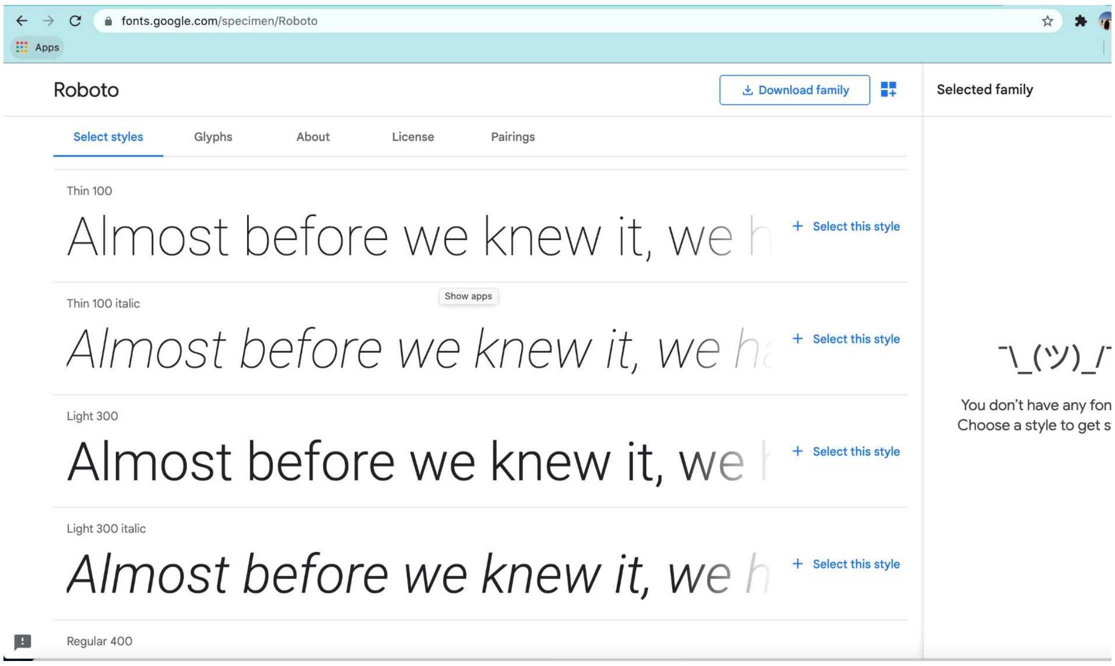
III. Positioning
In this classes you will be editing
1. Absolute
1.1 Set absolute position on .position-absolute class
2. relative and absolute
2.1 Set relative position on .position-relative class
2.2 Try different values for left, right, top, bottom on .absolute class. Start by not setting left and right or top and bottom at the same time, e.g. set top and left. What can you observe?
2.3 How is absolute within relative different from absolute that is outside of relative?
2.4 Set high z-index value on .position-absolute class. How does it affect the element?

TIP: You can use ctrl+f (cmd+f on MacOs) to search for specific content on the page.

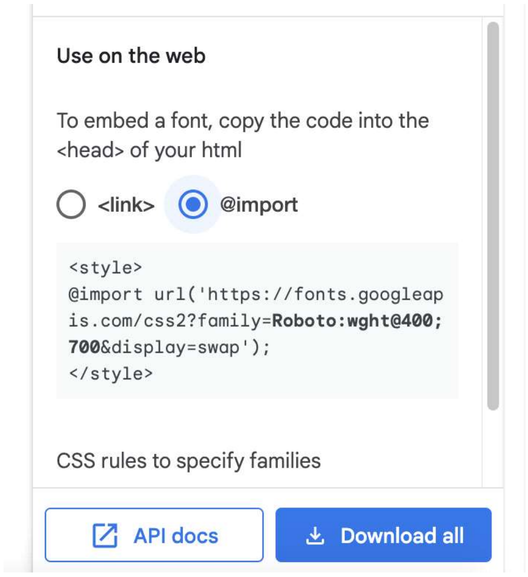
3. sticky
2.1 Set sticky position on .position-sticky class and scroll the page to see it work.
2.2 Note that we placed an HTML element with .position-sticky class outside of the article. This is because sticky positioned elements can be affected only by scrolling on their direct parents.

IV. Styling text
In this section we will focus on styling text. Search for the answers on the Internet.
1. Color - set font color using `color` CSS rule on .text-style class
2. Font-size - set font-size CSS rule on .text-style class
3. Line-height - set line-height CSS rule on .text-style class
4. font-weight - set font-weight CSS rule on .text-style class
4.1 Try both, keyword and number values, e.g. `bold` and `800`
5. Importing fonts and font-family
5.1 Visit https://fonts.google.com/
5.2 find fonts that you like
5.3 Click select style on various flavours of the same font. Selecting font styles is used to save the bandwidth for the user, if we do not intend to use certain styles of the font then we should not select them.

Select the styles. Especially consider the font-weight you set before. `Regular 400` on the list corresponds to the font used when no font-weight is used at all.



Almost before we knew it, we h



Use on the web

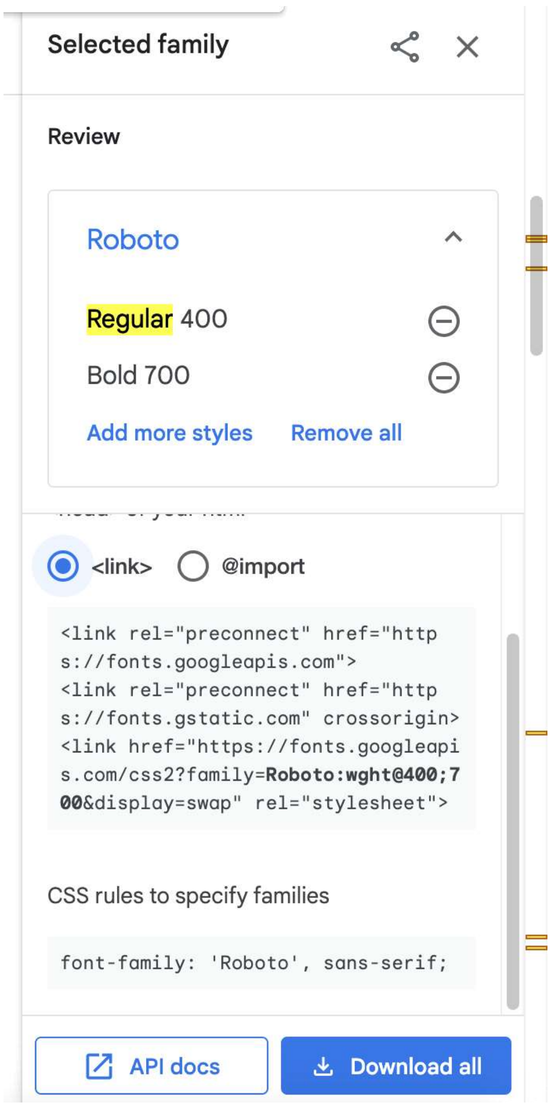
To embed a font, copy the code into the <head> of your html

<link> [✓ @import]

```
<style>
@import url('https://fonts.googleap
is.com/css2?family=Roboto:wght@400;
700&display=swap');
</style>
```

CSS rules to specify families

Select @import from `Use on the web` section and copy @import tag
Copy entire @import declaration and paste it into the <style>



```
<link rel="preconnect" href="http
s://fonts.googleapis.com">
<link rel="preconnect" href="http
s://fonts.gstatic.com" crossorigin>
<link href="https://fonts.googleapi
s.com/css2?family=Roboto:wght@400;7
00&display=swap" rel="stylesheet">
```

```
font-family: 'Roboto', sans-serif;
```

Copy font-family CSS rule into your .style-text class. sans-serif part of that rule activates in the situation where the previous font could not be loaded, e.g. we misspelled ‘Roboto’ or there is no internet connection. In that case, the browser will load the default sans-serif font.

TIP: Have a look at difference between serif and sans-serif fonts
TIP: font-family Roboto can be inferred looking at ?family=Roboto part of @import declaration

6. text-align - set text-align CSS rule on .text-style class
7. letter-spacing - set letter-spacing CSS rule on .text-style class
8. word-spacing - set word-spacing CSS rule on .text-style class
9. text-transform - set text-transform CSS rule on .text-style class

10. text-decoration - set text-decoration CSS rule on .text-style class

V. Styling links

In this section we will be styling links in different states:

- a, a:link - link in its untouched form
- a:visited - link that has been visited
- a:hover - link that is hovered with a cursor
- a:active - link that has been "hovered" using keyboard
- a:focus - link that has been clicked but the cursor is still not released or the browser is navigating to another webpage

Note that links come with a lot of styling from the browser. It is common to reset all those styles before developing CSS to simplify the styling process.

1. a, a:link, a:visited

  1.1 Style link in its basic form, set font color (using `color` CSS property)

  1.2 Consider using color, font-size, font-weight, font-decoration and border for your styles and feel free to use any other styles.

2. a:hover, :active, a:focus

3. Many websites do not apply special styling to a:visited links. Extract a:visited link and apply a style using CSS for it.

  2.1. If you do not create style for a:visited the styles will leak from `a` and `a:link` CSS rules

VI. calc()

In this section we will perform simple calculations using calc.

1. Try setting `width: 100vw;` on .calc-style class. The element should overflow outside of the screen.

1.1 Set `width: calc(100vw - 100px);` on .calc-style class. Now the element occupies 100px less than the entire width of the screen.

1.3 While in this example we could just use `width: 100%`, Sometimes you will stumble upon scenarios where the `width: 100%` solution does not work due to other CSS properties. calc() is not commonly used and should be avoided. At the same time a lot of CSS frameworks use calc() internally so when we are using CSS frameworks we usually do not need to use calc().