HTML / CSS Worksheet 4

This week's worksheet we will look at:

Flexbox in menus

styling a menu item

flex grow

flex shrink

centring an image on the page

embedding video (primarily from youtube)

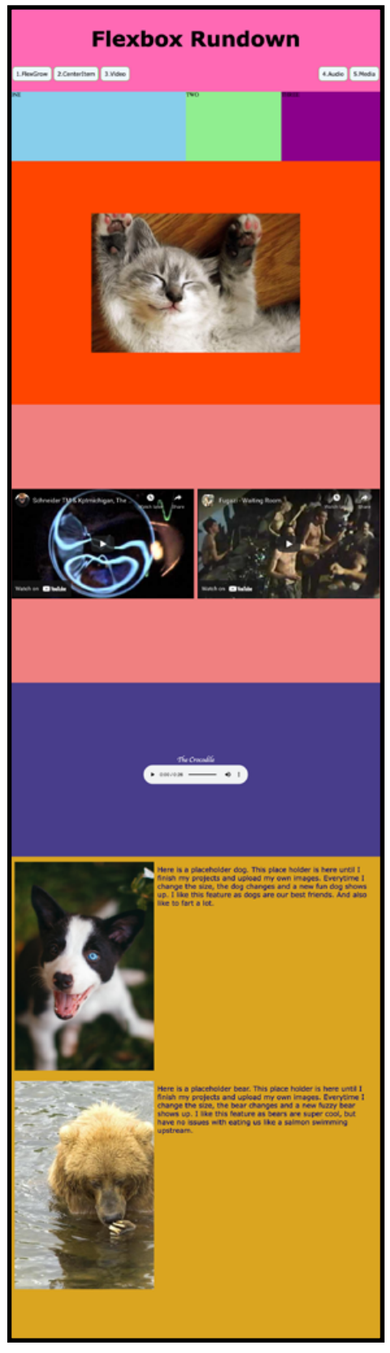
imbedding audio

media items with accompanying text

anchor links

publish on github

****Note** in order to succeed, read the ENTIRE STEP FIRST, THEN code everything by hand or by direction. It's the only way to be sure.**

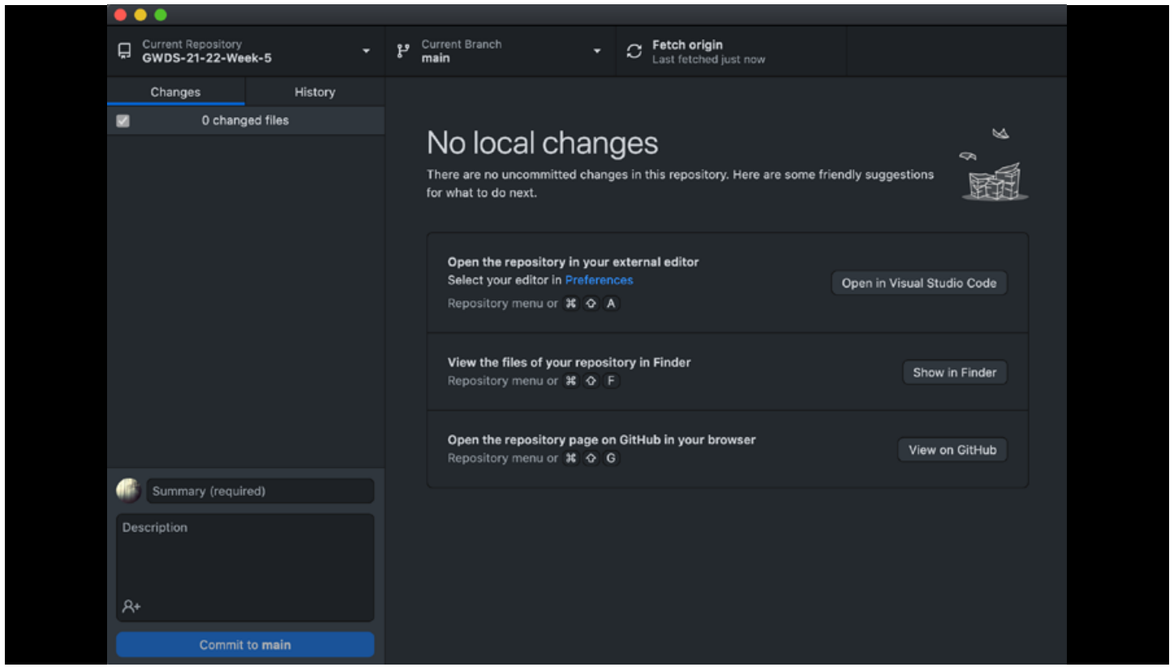


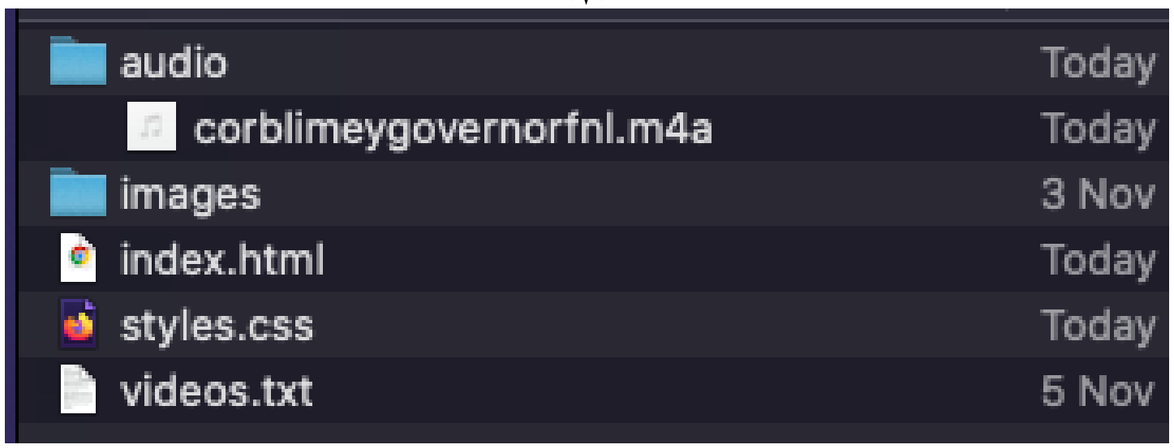
1

And we’re back! Let’s get down to business. Make and publish a new repository, located outside of your previous repositories. You can name it flexbox rundown (I named mine Week 5 but it’s up to you)

2

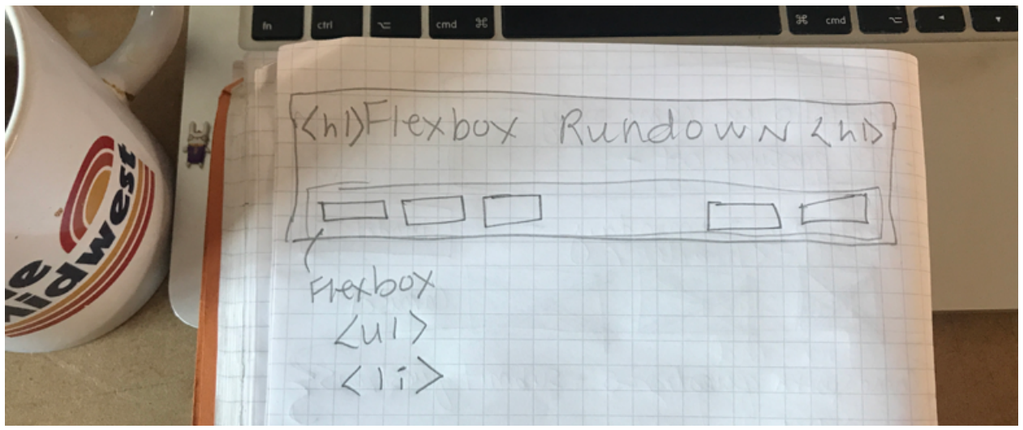
Download files here. (If the link doesn’t work, remember to download this pdf off of GitHub and the links should work).





3

In the files you downloaded you will see that there's a `<header></header>` tag waiting for you that is empty. The header holds what is usually the title and the menu. As always, I made a sketch and labelled it so I know what I am attempting to do (mental model & conceptual model at work).




```
<body>
    <header>
        <h1>Flexbox Rundown</h1>


            <ul>

                    <li>1.FlexGrow</li>
                    <li>2.CenterItem</li>
                    <li>3.Video</li>
                    <li>4.Audio</li>
                    <li>5.Media</li>


            </ul>

    </header>
```

I want a menu that is flexbox so that means it needs a container, and I am going to create my menu using an unordered list (like what we made in the second worksheet). Ultimately, I want three items on the left, and two items on the right.

We'll make our `h1` first and name it
Flexbox Rundown

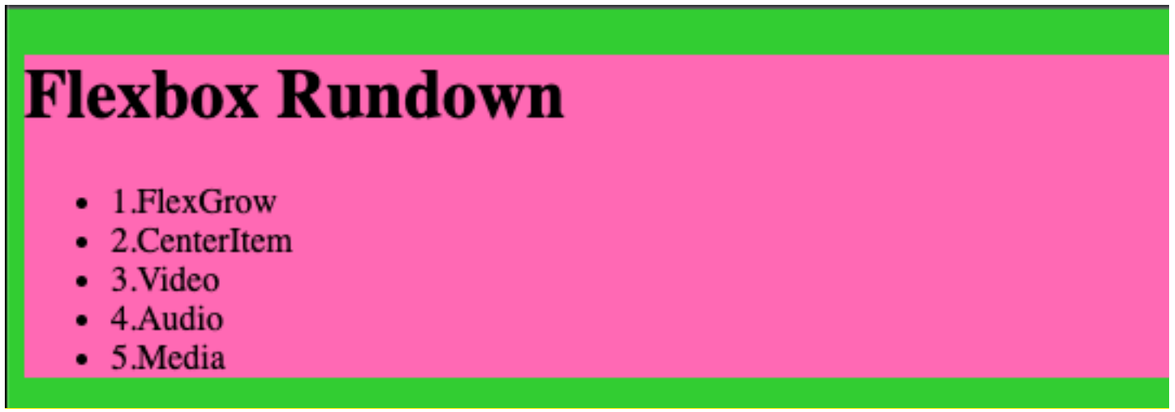
Then we will make an **unordered list** with the following items (just like the image to the right)
1.FlexGrow / 2.CenterItem / 3.Video /
4.Audio / 5.Media. **Save**, ***commit and push***

What do we have here?

Look at your styles.css sheet you downloaded.

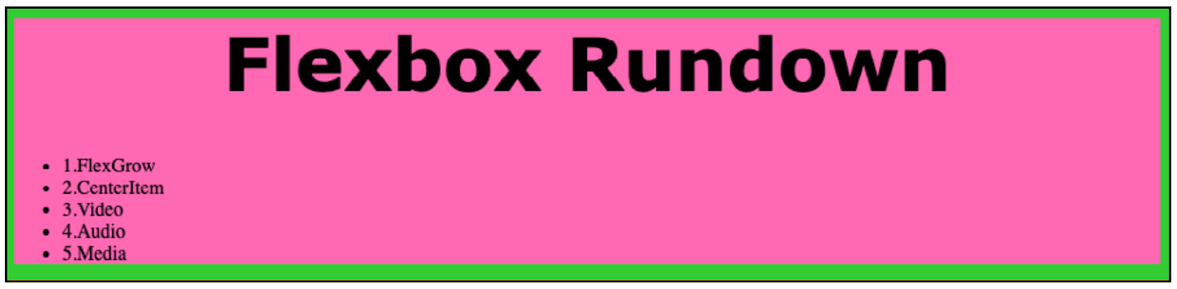
The <body> is lime green. I gave that to you.
Nested inside the <body> is the <header> and that is hot pink. I also gave that to you to see what it looks like all nested inside
It's an unordered list, but I had you put numbers in there. There is a reason for that.

Let's style this to make it look more like a menu



4 First let's get that title looking better
Save, ***commit and push to GitHub***

Now we have changed the font, its size and aligned it center. Does yours look like mine?



Ok, now that unordered list needs some changing. We are going to remove the bullet points, and also make this list a flexbox item. That means it needs a container. So we will go one step at a time.

We will look at our <ul> (unordered list) items and style them.

`list-style-type` allows you to change the style of the bullet points in a list, or in our case, remove them.

`padding` will allow the menu items to have a bit of padding around them. If you want to see how it affects is, place a 1px solid border in your colour choice in the ul style as well to see what is up. The more you know the more you understand.

I want to make our `<li>` (list item(s)) to be flexbox before we continue. Before we style the <li> itself, we must wrap our menu in a container.

We will make a class selector and call it container-navigation, and wrap it around inside our items. **Save**

Now we style it. One step at a time.

Now, **stop**. **Save**, ***commit and push***.

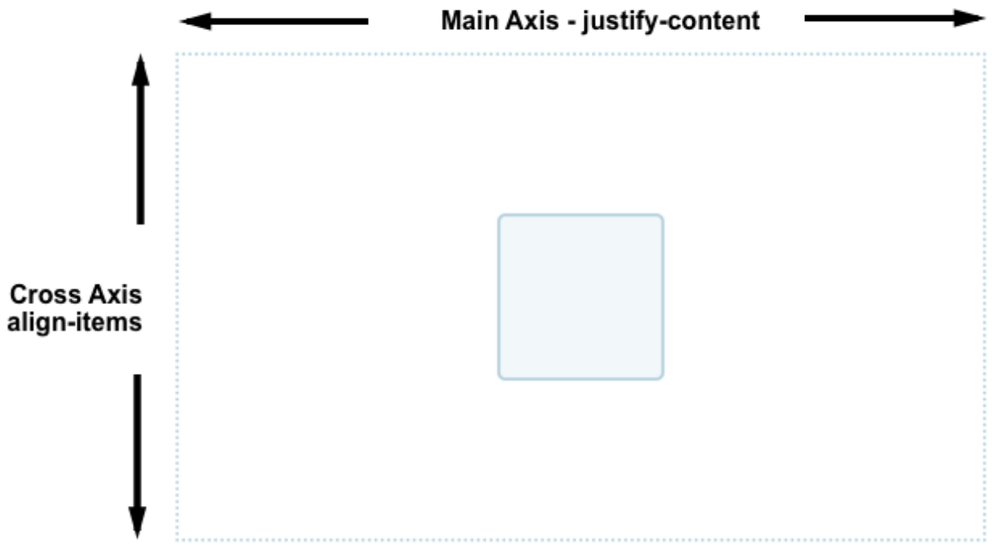
What do you see?

```
ul {
    list-style-type: none;
    padding: 5px;
}
```

```
<ul>
    <div class="container-navigation">
        <li>1.FlexGrow</li>
        <li>2.CenterItem</li>
        <li>3.Video</li>
        <li>4.Audio</li>
        <li>5.Media</li>

    </div>
</ul>
```

```
.container-navigation {
    display: flex;
    flex-wrap: wrap;
}
```

Our vertical list should now be horizontal. Now it's your turn to do some flexboxing.



Flexbox works on a main axis and a cross axis. It is declared with either row or column. It also has a row-reverse and column-reverse which is helpful if you are working in a language that reads right to left (RTL). We will look at all these. It is up to you to save, and then look at your local browser to see what each flex item does.

At the end, you'll end up here, RTL.
Save, ***commit and push***.


```css
.container-navigation {
    display: flex;
    flex-wrap: wrap;
    flex-direction: row;
    justify-content: space-evenly;
}
```

```css
.container-navigation {
    display: flex;
    flex-wrap: wrap;
    flex-direction: row;
    justify-content: flex-end;
}
```

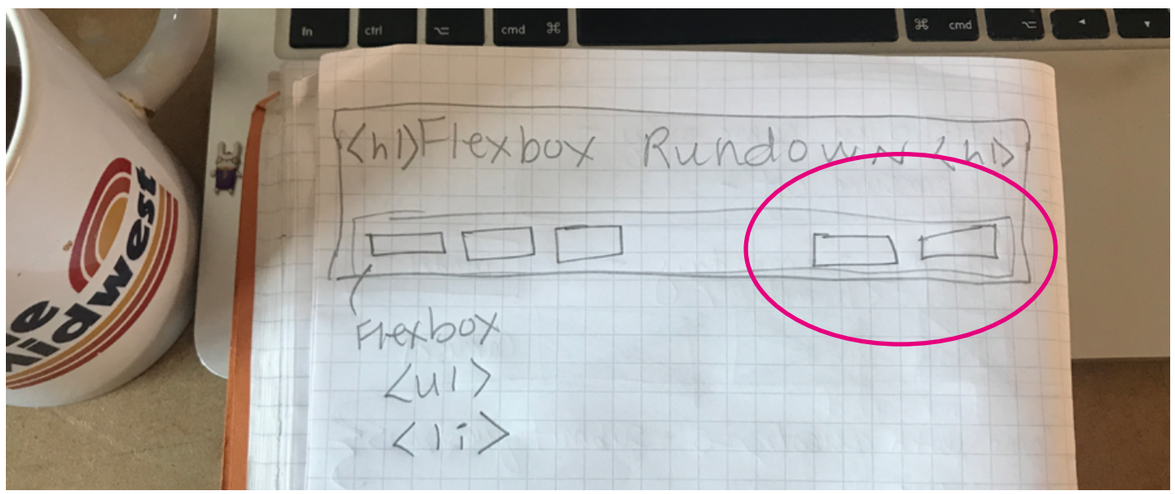
```css
.container-navigation {
    display: flex;
    flex-wrap: wrap;
    flex-direction: row;
    justify-content: center;
}
```

```css
.container-navigation {
    display: flex;
    flex-wrap: wrap;
    flex-direction: row-reverse;
    justify-content: space-between;
}
```

We will do the same with the column. Then land where we want to finish before we push two items to the right. You can learn much more about how these axis items flex here.

When you finish you will look like this:

Now we have made our menu horizontal, we need to push two of the boxes to the right and keep three to the left.



I will be working with margins. This is a part of the **box model** and something you can read more about here. I want to make the margin go automatically to the left, pushing the other two items to the right.

```
.container-navigation {
    display: flex;
    flex-wrap: wrap;
    flex-direction: column;
    align-items:flex-end;
}
```

```
.container-navigation {
    display: flex;
    flex-wrap: wrap;
    flex-direction: column;
    align-items:flex-start;
}
```

```
.container-navigation {
    display: flex;
    flex-wrap: wrap;
    flex-direction: column;
    align-items:center;
}
```

```
.container-navigation {
    display: flex;
    flex-wrap: wrap;
    flex-direction: column-reverse;
    align-items:center;
}
```

```
.container-navigation {
    display: flex;
    flex-wrap: wrap;
    flex-direction: row;
    justify-content: flex-start;
```

Create a class selector that will go inside my <li> tag, like this in the first image.

Save, ***commit and push.***

We should be here.

Great. Now I want my links to look more like simple buttons. In the css file you downloaded, I gave you an empty li class. We want to,

- change the font to Verdana
- make the padding 10px on all sides
- give it a margin of 3px on the top and bottom
- give it a border that is lime green
- curve the corners of the box (with border-radius)
- give it a very light blue background

Save, ***commit and push***

```
.push {
    margin-left: auto;
}
```

```
<ul>
    <div class="container-navigation">
        <li>1.FlexGrow</li>
        <li>2.CenterItem</li>
        <li>3.Video</li>
        <li class="push">4.Audio</li>
        <li>5.Media</li>

    </div>
</ul>
```

```
li {
    font-family: Verdana, Geneva,
    Tahoma, sans-serif;
    padding: 10px;
    margin: 0px 3px;
    border: 1px solid limegreen;
    border-radius: 10px;
    background-color: aliceblue;

}
```

Flexbox Rundown

1.FlexGrow | 2.CenterItem | 3.Video | 4.Audio | 5.Media

Flexbox Rundown

1.FlexGrow | 2.CenterItem | 3.Video | 4.Audio | 5.Media

5

Moving along. This next section is about flex-grow and flex-shrink. This determines how much a box is allowed to grow and shrink within a container. I advise strongly you read more about it here. Will you use it this semester? Maybe.

Basically If you have three boxes or more, how do they all behave with each other? The first number is how much the box is allowed to grow, the second is how much it is allowed to shrink and the third value is either set to auto, or a specific pixel size. To be honest, this is the least well explained and a bit of a wibbly wobbly concept to get your head around. Here is an article that describes it better than I ever could at the moment.

Each box is set to `flex: 1 1 auto;`

Head to `.box2` and change the values to:
`2 1 auto.` Then **Save**, ***commit and push.***

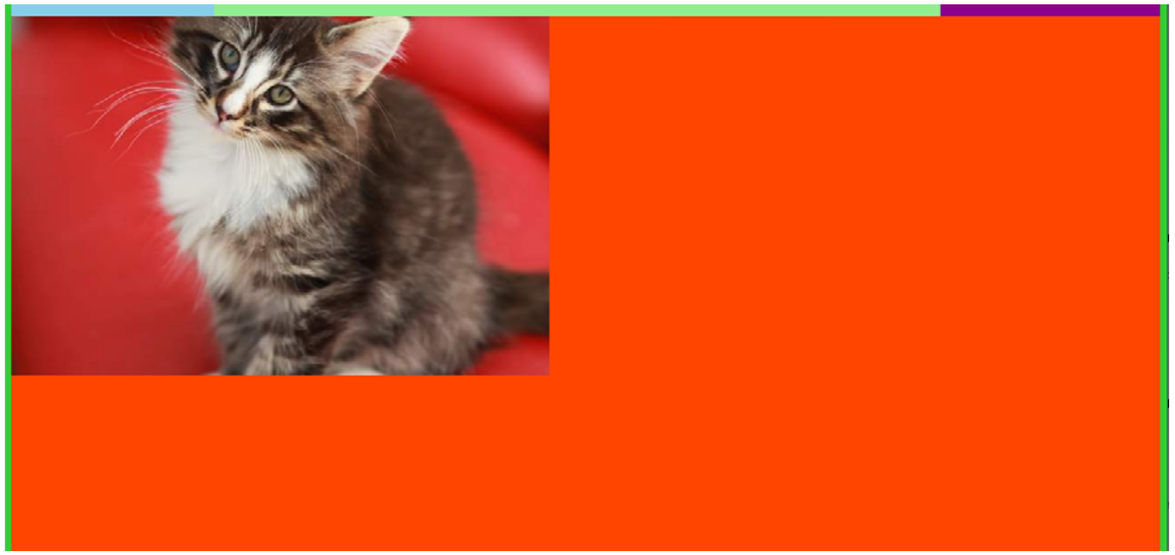
Refresh and see what happens.
Now back at .box2 change the values to:
`2 -1 auto.` Then **Save**, ***commit and push.***

Refresh and see what happens. You can try different combinations, and how they work with the three boxes together. Can you think of a reason you would use them.

```
.box1 {
    background-color: skyblue;

    height: 200px;
    flex: 1 1 auto;

}

.box2 {

    background-color: lightgreen;

    height: 200px;
    flex: 1 1 auto;
}

.box3 {

    background-color: darkmagenta;

    height: 200px;
    flex: 1 1 auto;

}
```

6

The next thing we see in our webpage is a kitten. It's a placekitten to be exact. This is a web based link that acts as a placeholder image you can specify by size. If we look at section-two, you will see the image import to placekitten.com then the dimensions of the image. 600 being width and 400 being height. Change around these two values and see how the image changes. Then land back somewhere around the original numbers, or something similar either landscape or portrait.



I want this image to be dead center in my section. Because this is the only element inside the section, **the section becomes the container**. I'm no fortune teller but this might come in handy in other modules and classes. Just sayin'. Let's get to it.

```
<div class="section-two">

    <img src="http://placekitten.com/
    600/400">

</div>
```

Head to your `.section-two` which is already partially populated, and under the height add the following (then **save**, ***commit and push***):

```
.section-two {
    background-color: orangered;
    height: 700px;
    display: flex;
    align-items: center;
    justify-content: center;
}
```

First we declare the section a flexbox, then we align items (column) to center, and justify content (row) to center and this image lands right in the center.

Look like this? If so, cool. If not, don’t worry, retrace your steps.

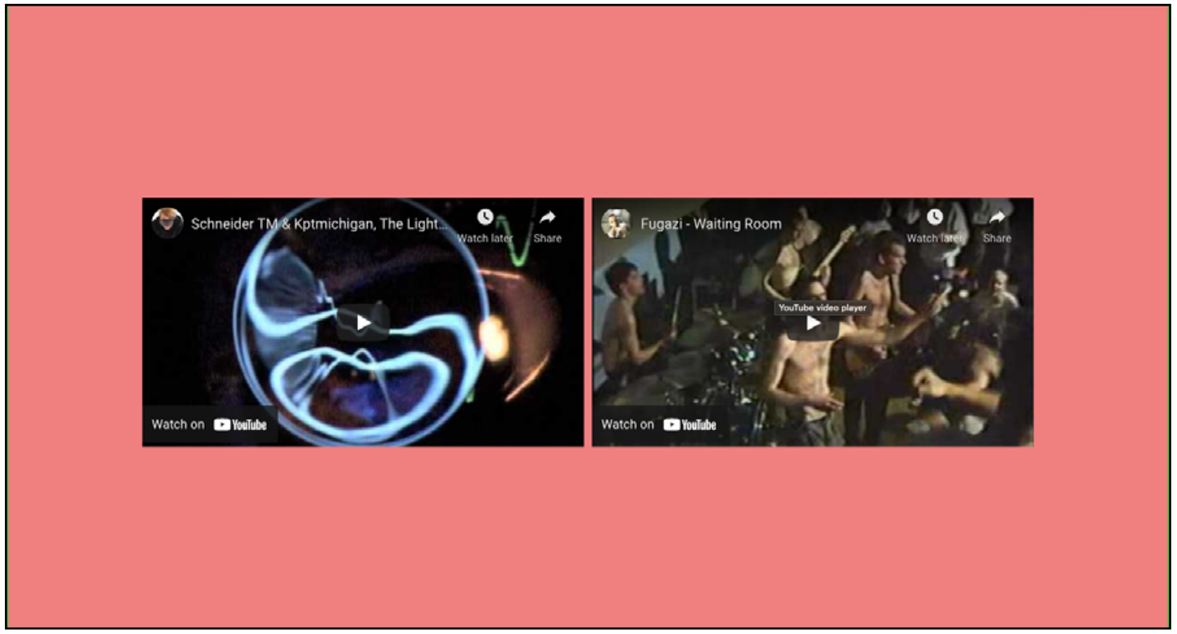
7

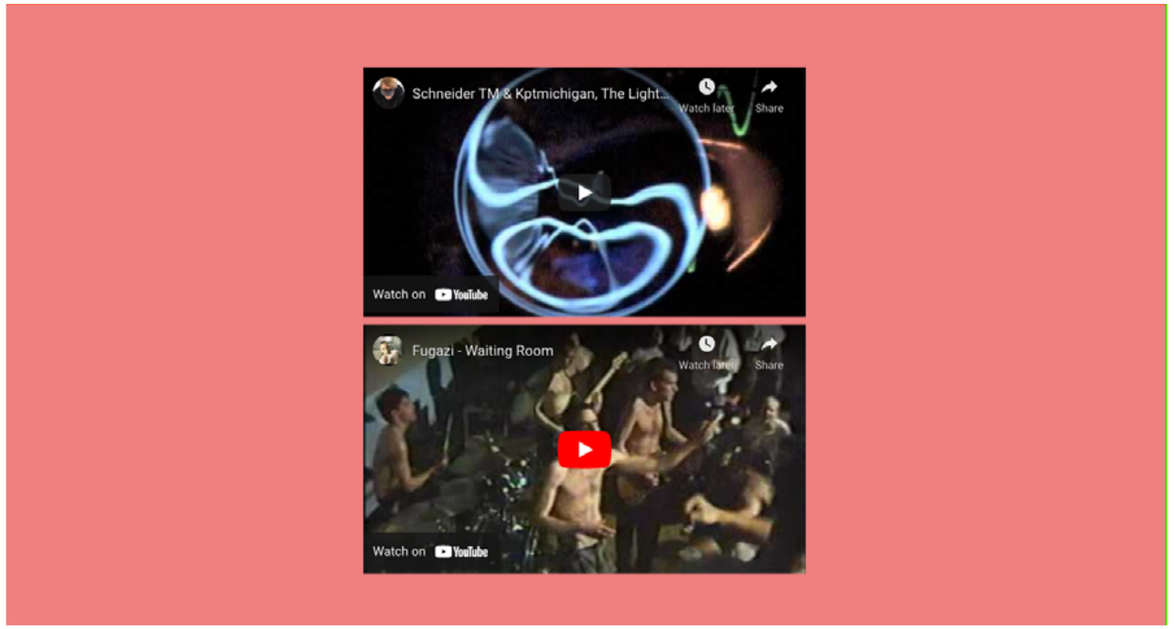
Now we are going to import two youtube videos. Head to the .txt file I provided in the down loads and place that code inside `<section-three>` in your html.

After you **save**, ***commit and push***, I want you to center it inside the section, like in step 6. Then I want you to designate it as a column using flexbox. So it looks like the picture to the right. I have given you all the tools in the previous steps in this worksheet to do this, so now I want you to give it a try.

If you put the two videos back to a row (like in the first screenshot, can you make the images wrap? We have looked at that before as well, how can you make that happen as well?

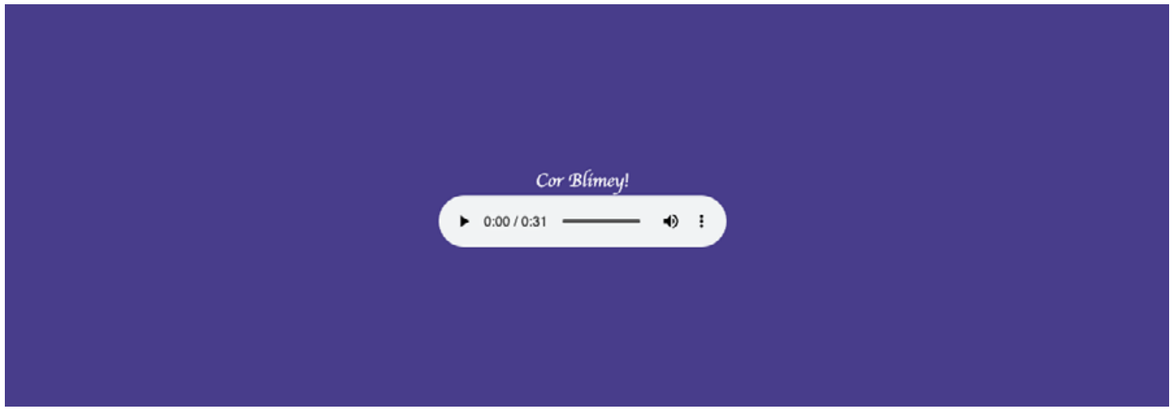
Save, ***commit and push.***





8

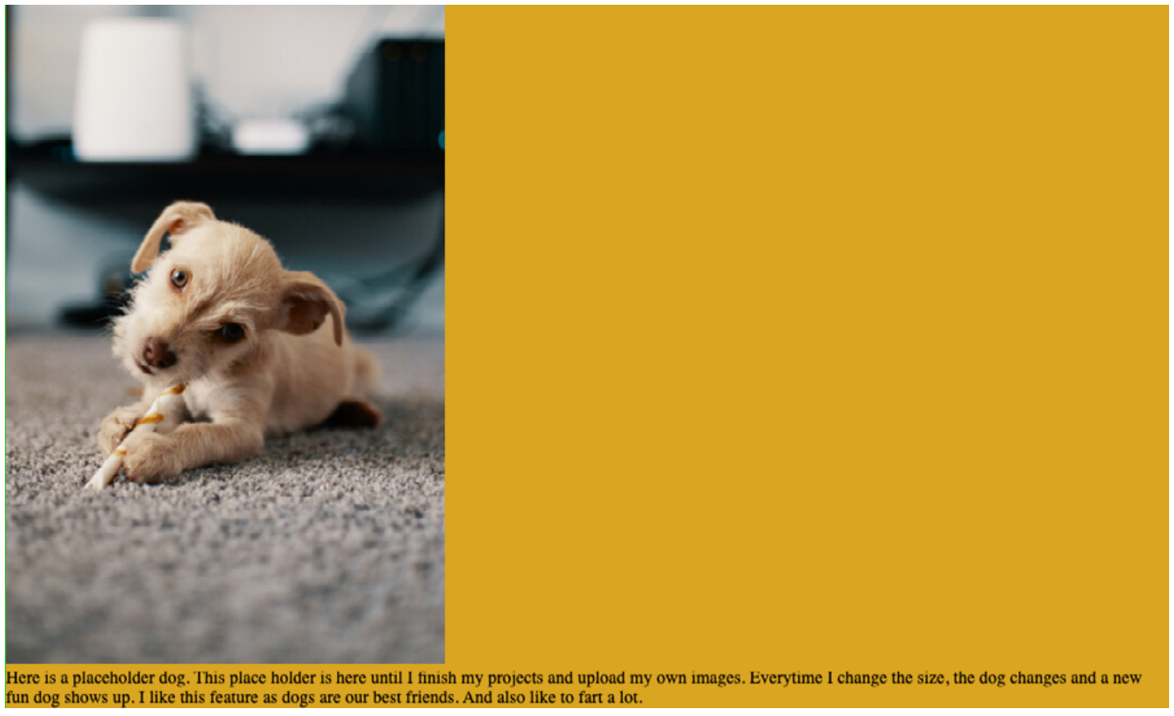
In Section 4 you see an audio file and a title. Center that like we did in the last two sections for practice, and style the title provided for you. (hint: the title is `<figcaption>`



When you are done, **save**, ***commit and push.***

9.

Almost done. Now In section-five you will see a <media-container> (container should give you a hint) with another place holder image. This time it is from placedog.com, and place-bear.com. These are similar to the placekitten and are handy when you are figuring out what sizes you need for your website, or if you already know your sizes, but your content (projects aren't finished and you need a for position only image to put in its place. I have provided some copy for you. Now we are going to style them so the image is on the left and the copy is on the right using flexbox.



First we look at our .media container. We want it to be the flexbox container with the content as flex-start (remember the menu) flex-wrap: wrap, which means it will wrap around itself when it gets too narrow, and some padding to keep it tidy. **Save**. Where are we. Not much has changed yet.

```
.media-container {
    display: flex;
    justify-content: flex-start;
    flex-wrap: wrap;
    padding: 15px 25px 15px 15px;

}
```

Next, we will look at .media-container .mediac-ontent which is a style for these two selectors to-gether. Now have a look, here's that flex: 1; which means that media content and media container has a **flex-grow of 1**. (What happens when you change that number?)

```
.media-container .mediacontent {
    flex: 1;
    padding: 10px;
}
```

Finally, I want to just style .mediacontent a bit. change the font family and colour and size.

Flip the order now. Learn how to do it here.
This technique might be useful in layouts.

Save, ***commit and push***.

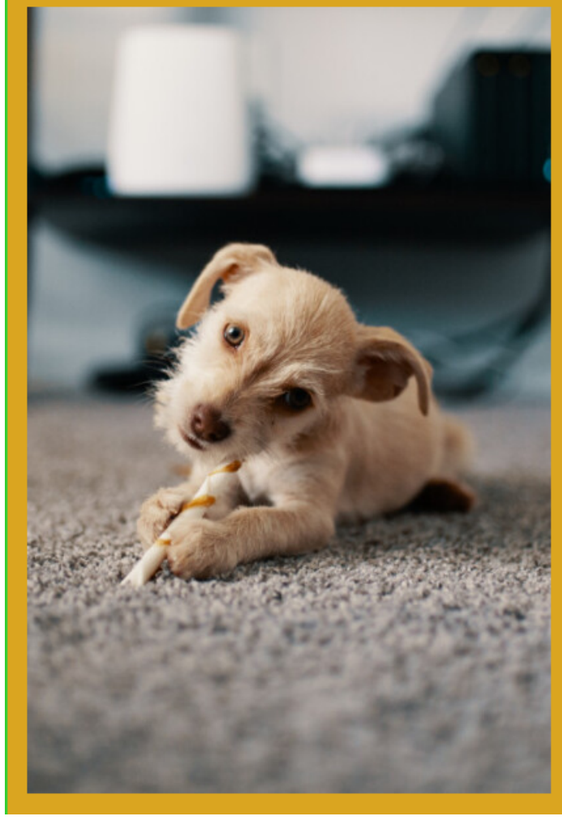
```
.mediacontent {
    font-family: Verdana, Geneva,
    Tahoma, sans-serif;
    color: navy;
    font-size: 20px;


}
```

10

And that's almost a wrap. Now, dear reader, it is your turn to build your own lightsabre, so to speak. Our menu items are labeled in such a way that they would make great navigation buttons by turning them into anchor links on the same page. Now, I want you to figure out how to do that. About 98% of learning how to code is looking it up and figuring it out. That's what I want you to do. Make your top menu items Anchor links to our different sections we created today. You can start reading about it here. Good luck!

Don't forget to **save**, ***commit and push***!

Wanna be extra? Try practice publishing this page on github!



Here is a placeholder dog. This place holder is here until I finish my projects and upload my own images. Everytime I change the size, the dog changes and a new fun dog shows up. I like this feature as dogs are our best friends. And also like to fart a lot.