MAKE SCHOOL

CSS Box Model and Flex

WEB 1.0

Agenda

- Learning Outcomes
- Review: CSS
- Box Model
- More Selectors
- **BREAK**
- Flexbox
- Homework
- Lab Time

Learning Outcomes

By the end of today, you should be able to...

1. **Draw** a picture of the box model
2. **Create** a box of any size
3. **Use** Flex to arrange elements

Review: CSS

Warm-Up (15 minutes)

Quick style this web page about cats!

TIP: Try out Repl.It's "Multiplayer" feature - click on "Share" to give your partner a link!

MAKE SCHOOL

How big is a box?

How big is a box?

All block elements are drawn as a box. The rules for how blocks are drawn is called the:

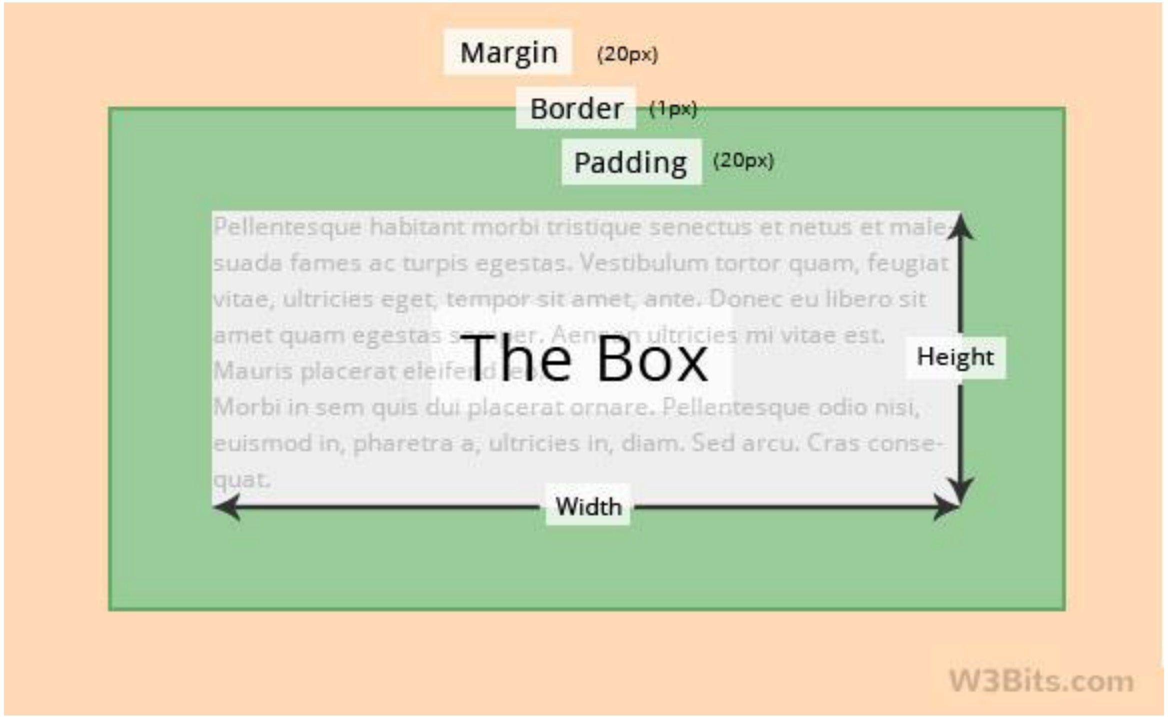
Box Model

The Box Model?

The Box Model has the following properties:

- Width
- Height
- Padding - *How much space between content & border?*
- Border
- Margin - *How much space between border & neighboring elements?*

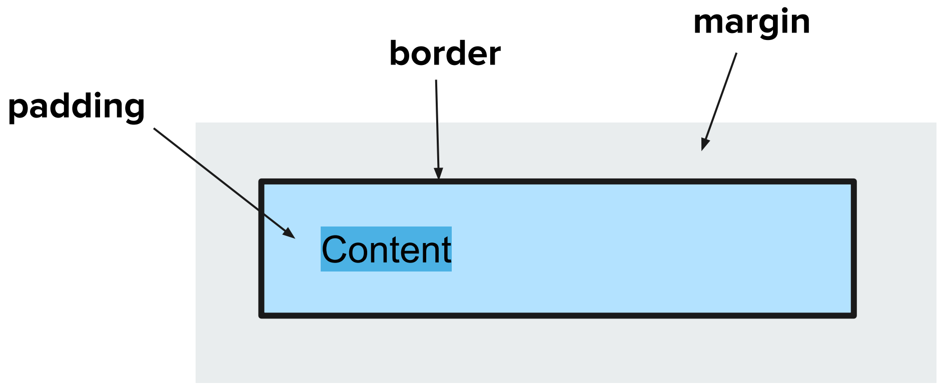
The Box Model


Margin (20px)
Border (1px)
Padding (20px)
Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Vestibulum tortor quam, feugiat vitae, ultricies eget, tempor sit amet, ante. Donec eu libero sit amet quam egestas semper. Aenean ultricies mi vitae est. Mauris placerat eleifend leo.
Morbi in sem quis dui placerat ornare. Pellentesque odio nisi, euismod in, pharetra a, ultricies in, diam. Sed arcu. Cras consequat.
The Box
Height
Width
W3Bits.com

The Box Model

CSS is written as a series of style rules. Each style rule starts with a **selector** (to specify which elements we are modifying), and a list of **properties** we want to modify.

```
h1 {
  width: 200px;
  height: 40px;
  border-width: 3px;
  background-color: blue;
}
```



The Box Model (10 minutes)

Try it for yourself

https://make-school-courses.github.io/WEB-1.0-Web-Foundations/#/box-model.html

New Selectors

Select by Tag Name

A selector can select all elements with a particular tag name.

index.html

```html
<p>
   This is a <em>paragraph</em>.
</p>


<p>
   This is another paragraph.
</p>
```

style.css

```css
p {
   font-family: Helvetica;
}
```

Here, we're specifying that **all paragraphs** should use the **Helvetica font**.

Select by Class

MAKE SCHOOL

We can also select by **class name**:

index.html

```
<span class="error">
   The password you entered is
incorrect.
</span>
```

style.css

```
.error {
   color: red;
   font-weight: bold;
}
```

We use a class when there may be **multiple elements** that belong to the class.

Select by Id

We can also select by **id**:

index.html

```html
<section id="about-me">
   <h1>About Me</h1>
   <p>...</p>
</span>
```

style.css

```css
#about-me {
   margin-top: 10px;
   border: 1px dashed black;
}
```

Only one element on the page can have a particular id! In other words, ids must be unique.

Select by Id

You can also specify which element you're selecting for, in addition to the class/id.

index.html

```html
<section id="about-me">
   <h1>About Me</h1>
   <p>...</p>
</span>
```

style.css

```css
span#about-me {
   margin-top: 10px;
   border: 1px dashed black;
}
```

This means: Select for any span elements that have the class "about-me"

Combining Selectors

We can select for an element that is a **descendent** of another element:

index.html

```html
<section id="about-me">
   <h1>About Me</h1>
   <p>...</p>
</section>
```

style.css

```css
#about-me h1 {
   font-size: 1.2em;
}
```

In this case, we're selecting for the **h1** element that is **inside of** the section with id "about-me".

Check for Understanding

How would we select for all images inside of p elements?

index.html

```
<p>
   Here are some photos...
   <img id="my-photo1">
   <img id="my-photo2">
</p>
```

style.css

```
{
   width: 250px;
}
```

Check for Understanding

MAKE SCHOOL

How would we select for all inputs inside of span elements with class error?

index.html

```html
<span class="error">
   Please enter your name.
   <input type="text" name="firstname">
   <input type="text" name="lastname">
</span>
```

style.css

```css
                 {
   border: 1px solid red;
}
```

Break - 10 min

“Take a 10 minute break and wrap a tag around everything you see.”

CSS Diner (20 minutes)

Break out into groups of 2 and play the CSS Diner game.

Make sure to practice good **Pair Programming**:

- The **Driver** shares their screen & types in code
- The **Navigator** tells their driver what to type

Flex Box

Flexbox

Flexbox (or just **Flex**) is a tool you can use to arrange things on the screen.

In the days before Flexbox, it was really hard to arrange elements horizontally on the screen. You had to use `table`s (with rows and columns), which was really annoying.

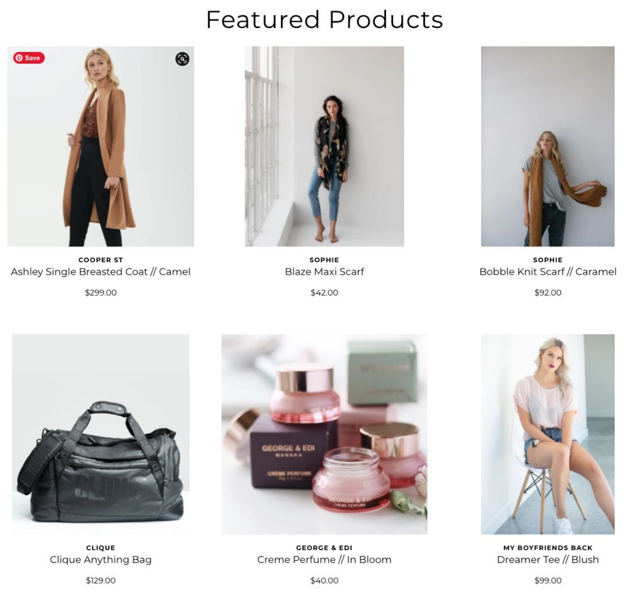
Now, it's considered to be bad practice to use `table`s to arrange elements (they're only used for tabulating data). So instead, we use Flexbox.

An Example

Let's say we are building an online store, and we want to arrange the items in the store into rows and columns.

We want to have the same amount of space in between the items. When the user's screen size shrinks, some items should be pushed down to the next row.

How do we do that?



An Example

Check out this Repl.It to see how we might accomplish this with only a few lines of code. (The "items" are just represented as black squares, for now.)

Flex

Here `display: flex` is applied to the `ul` element.

The `li` tags are arranged inside of their parent.

```css
ul {
     display: flex;
}
```

```html
<ul>
     <li>Apples</li>
     <li>Oranges</li>
     <li>Pears</li>
</ul>
```

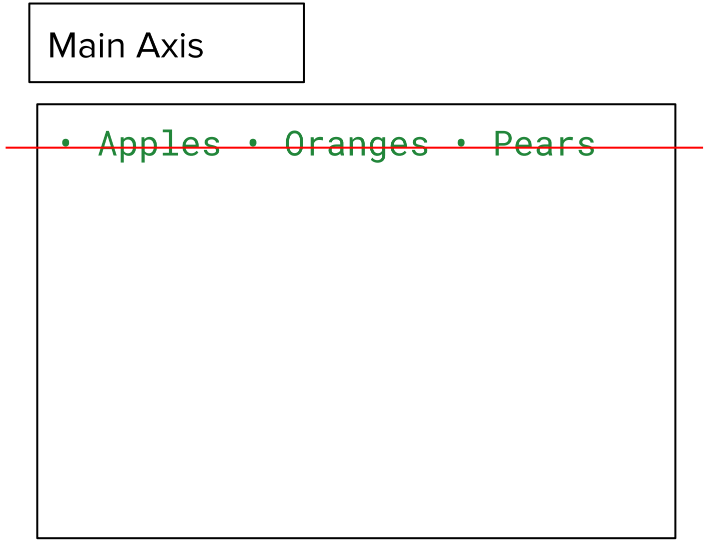
```
• Apples • Oranges • Pears
```

Flex

The main axis: **Flex arranges elements on an axis**

```css
ul {
    display: flex;
}
```

```html
<ul>
    <li>Apples</li>
    <li>Oranges</li>
    <li>Pears</li>
</ul>
```



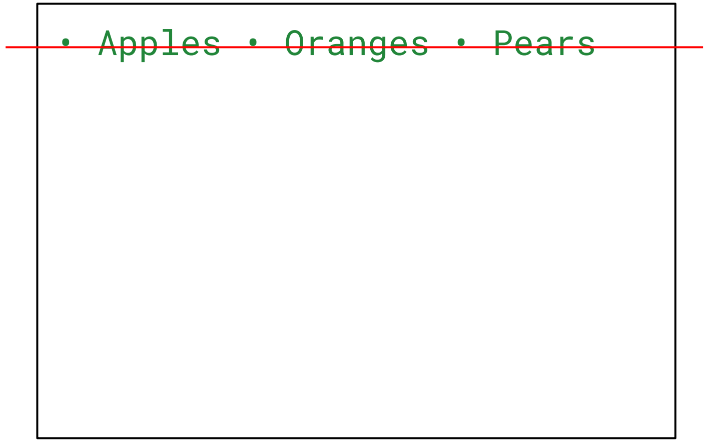
Flex

The axis can be horizontal - **row** - this is the default

```
ul {
  display: flex;
  flex-direction: row;
}
```

```
<ul>
     <li>Apples</li>
     <li>Oranges</li>
     <li>Pears</li>
</ul>
```

flex-direction: **row**



Flex

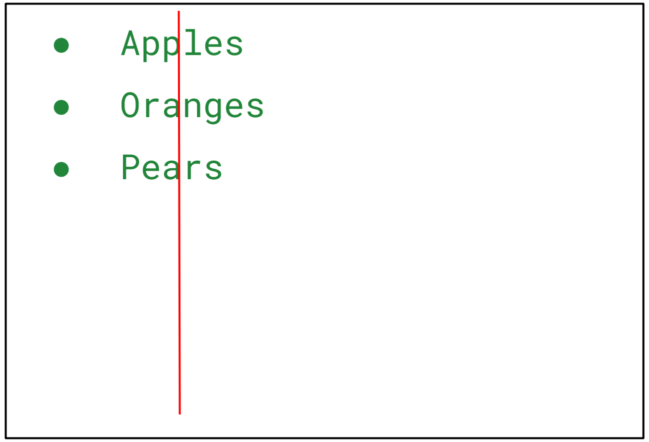
MAKE SCHOOL

The axis can be vertical - **column**

```css
ul {
  display: flex;
  flex-direction: column;
}
```

```html
<ul>
     <li>Apples</li>
     <li>Oranges</li>
     <li>Pears</li>
</ul>
```

flex-direction: **column**

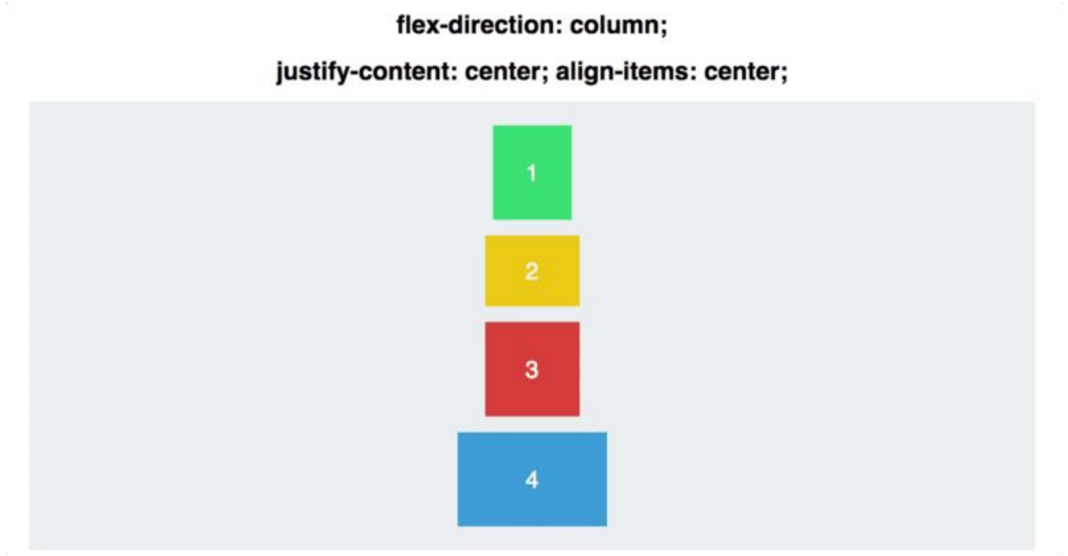


- Apples
- Oranges
- Pears

Flex

Flex-direction can be **row** or **column**

```
ul {
  display: flex;
  flex-direction: column | row;
}
```



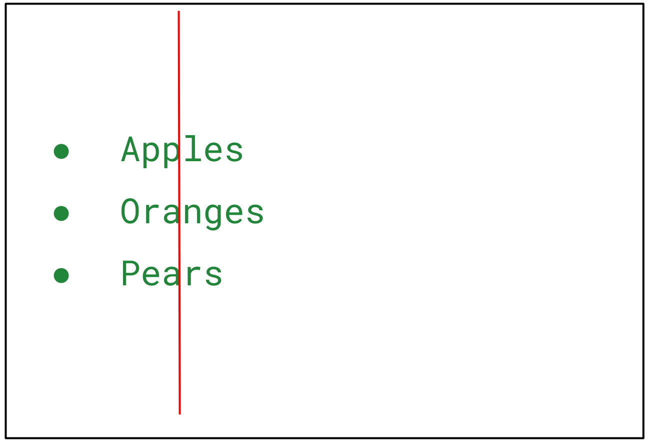
Flex

Use **justify-content** to arrange elements on the **main axis**.

```
ul {
  display: flex;
  flex-direction: column;
  justify-content: center;
}
```

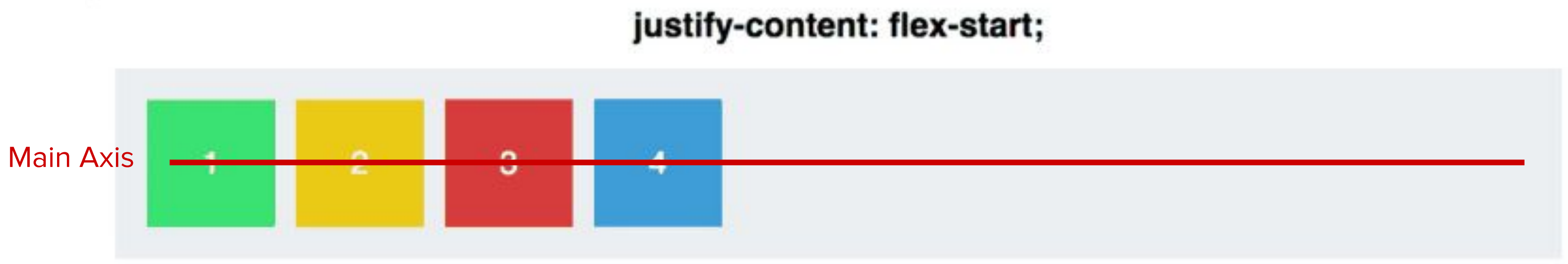
```
<ul>
     <li>Apples</li>
     <li>Oranges</li>
     <li>Pears</li>
</ul>
```

justify-content: **center**



Flex

```
ul {
  display: flex;
  flex-direction: row;
  Justify-content: flex-start | center | flex-end | space-between | space-around;
}
```



CSS Challenges

Flex

Solve these challenge problems then move on to the homework

Play a Game with Flex

Flex

Try one of these games to practice your flex box skills

Tower defense: http://www.flexboxdefense.com

Flex box Froggy: https://flexboxfroggy.com

MAKE SCHOOL

Lab Time

Homework

This week's assignments:

- CSS Challenges Part 1 - Due tonight at midnight
- CSS Challenges Part 2 - Due next Tuesday
- Portfolio Part 2: Styles - Due next Tuesday

Lab Rules

Modify your Zoom name (hover over your photo and click the 3 dots) to put a "1" before your name if you'd like to work alone, "2" if you'd like to work with a partner.

Stay in the main Zoom room if you’d like to stay for more Q&A, homework help, etc.

Go to your individual breakout room if you’d prefer to work with a partner or have quiet time!