Layouts with Flexbox

Revision

So far we've relied on the display properties `block` and `inline` for layout.

In this session we are going to look at another display property called **Flexbox**, which is a more **powerful** tool for creating **complex** and **responsive** HTML layouts.

Flexbox

💡 Flexbox

We can use the `flex` property to **modify** the **layout** of **elements** on our page.

Flexbox gives a container the ability to **alter** the **width** and/or **height** of the items inside it.

This makes it **useful** for implementing **responsive design** as elements can change size dynamically to best fill the space across different sized screens.

To use Flex we use the display property on the container around items that we want to change size:

```
.container {
    display: flex;
}
```

👉 Flexbox

Open `introduction-to-flexbox` in your code editor.

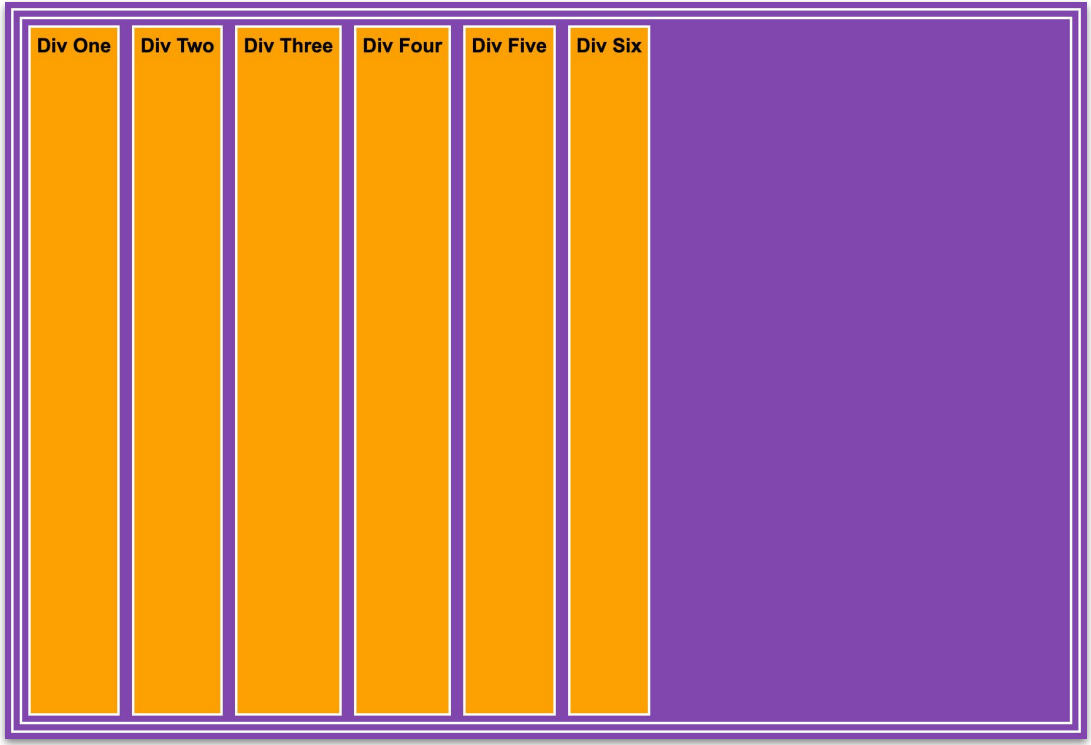
Open `index.html` in a web browser.

👉 Flexbox

Add `flex` to the `body`:

```
body {
    display: flex;
}
```

✅ Refresh your browser, it should look pretty similar to the screenshot on the right.



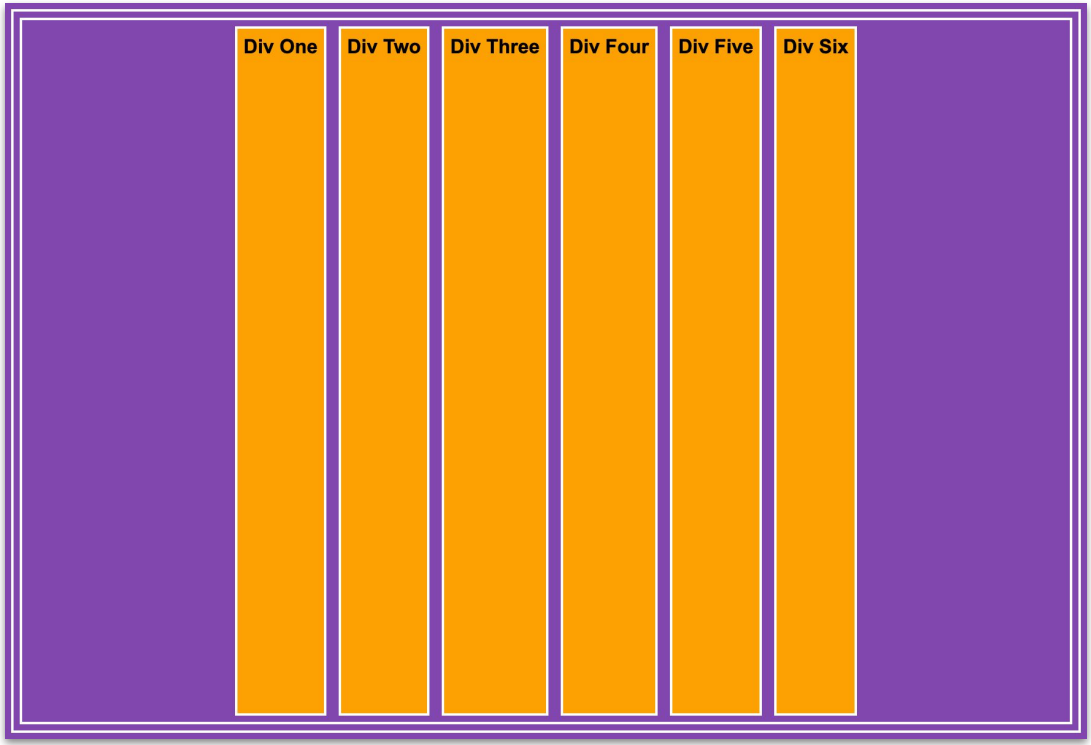
👉 Flexbox

Add `flex` to the `body`:

```
body {
    display: flex;
    justify-content: center;
}
```

Try the different values for `justify-content`:

`flex-start`	`space-between`
`flex-end`	`space-around`
`center`	`space-evenly`



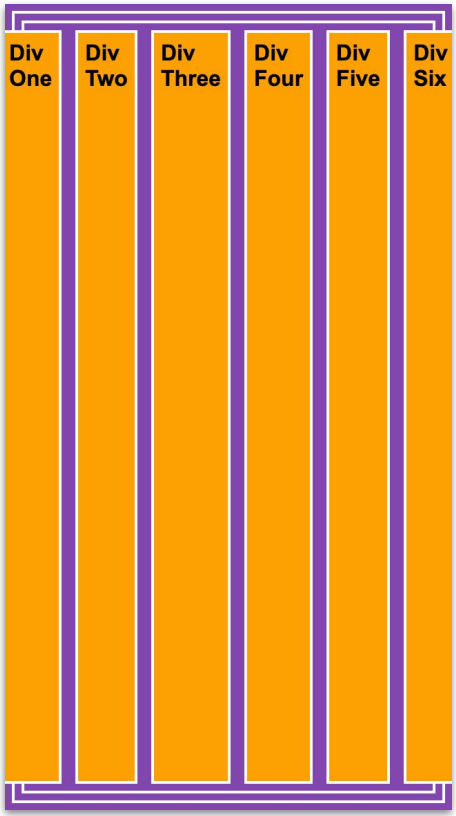
✅ Refresh your browser and see what changed for each!

👉 Flexbox

When you make the screen really small? You might notice that the divs are overflowing the body box.

Let's make them wrap rather than overflowing:

```
body {
    display: flex;
    justify-content: center;
    flex-wrap: wrap;
}
```



✅ Refresh your browser and see what changed!

👉 Flexbox

We can also align the content vertically:

```
body {
    display: flex;
    justify-content: center;
    flex-wrap: wrap;
    align-content: center;
}
```

Try the different values for `align-content`:

`flex-start`	`stretch`
`flex-end`	`space-between`
`center`	`space-around`

✅ Refresh your browser and see what changed for each!

👉 Flexbox

So far we have applied flex properties to the parent element. We can also apply properties per child element:

```
body {
      display: flex;
      justify-content: center;
      flex-wrap: wrap;
      align-content: center;
}

.demo-div {
      flex-grow: 1;
}
```

The unitless value represents a proportion and dictates what amount of available space inside the parent the element should take up.

Since every div has flex-grow set to 1, the remaining space in the parent is distributed equally to all children.

✅ Refresh your browser and see what changed!

👉 Flexbox

We could also add a different `flex-grow` value to one div to make it larger than the others:

```
.demo-div {
     flex-grow: 1;
}

#div-6 {
    flex-grow: 2;
}
```

Since we have set `div-6` to have a value of `2`, it tries to take up twice as much space as the others.

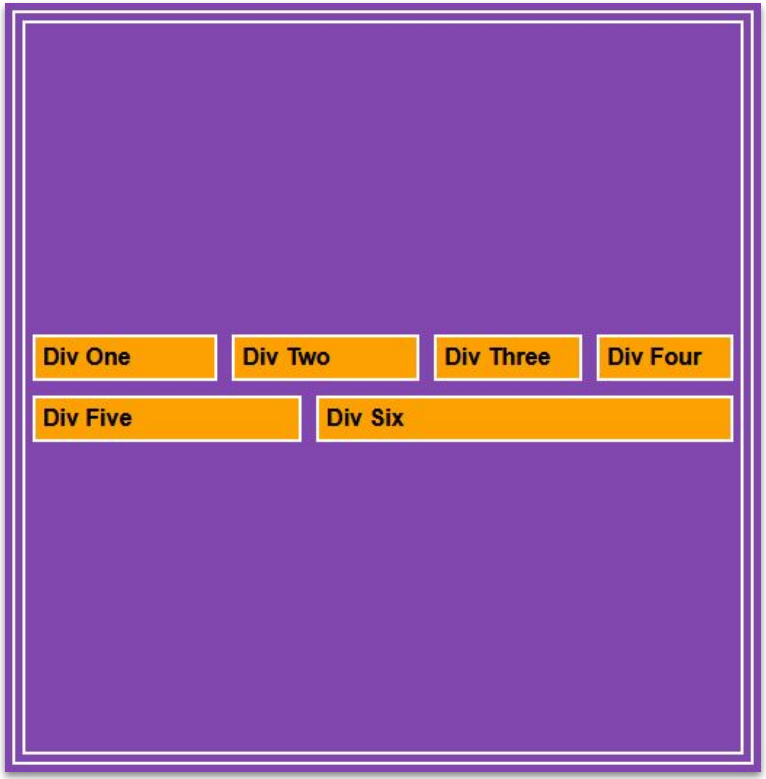
✅ Refresh your browser and see what changed!

👉 Flexbox

We could also choose to manually define the width of each div instead:

```
.demo-div {
      flex-grow: 1;
}
#div-6 {
      flex-grow: 2;
}

#div-1 {
    flex-basis: 20%;
}
#div-2 {
    flex-basis: 100px;
}
```



✅ Refresh your browser and see what changed!

👉 Flexbox

We can even change the order of the divs:

```
#div-6 {
      flex-grow: 2;
      order: 1;
}

#div-1 {
      flex-basis: 20%;
      order: 2;
}

#div-2 {
      flex-basis: 100px;
      order: 3;
}
```

✅ Refresh your browser and see what changed!

Why are divs 1, 2 and 3 still in the same positions?