warwick C<>ding

Session 5

CSS Layout

Recap from Session 4

- Intro to CSS
- 3 ways of embedding CSS into our HTML pages
- CSS Selectors
- id vs class
- CSS properties

CSS Anatomy

CSS syntax typically follows the “Selector” and “Declaration” of property-value pairs

```
selector {
  property: value;
  property: value;
}
```

Example

```
body {
  color: red;
  background-color: green;
}
```

The correct way to embed CSS

HTML

```
<head>
  <link rel="stylesheet" type="text/css" href="styles.css" >
</head>
```

Lets start!

Block and Inline Elements

- The browser treats a block element as you would when you press enter after a sentence.

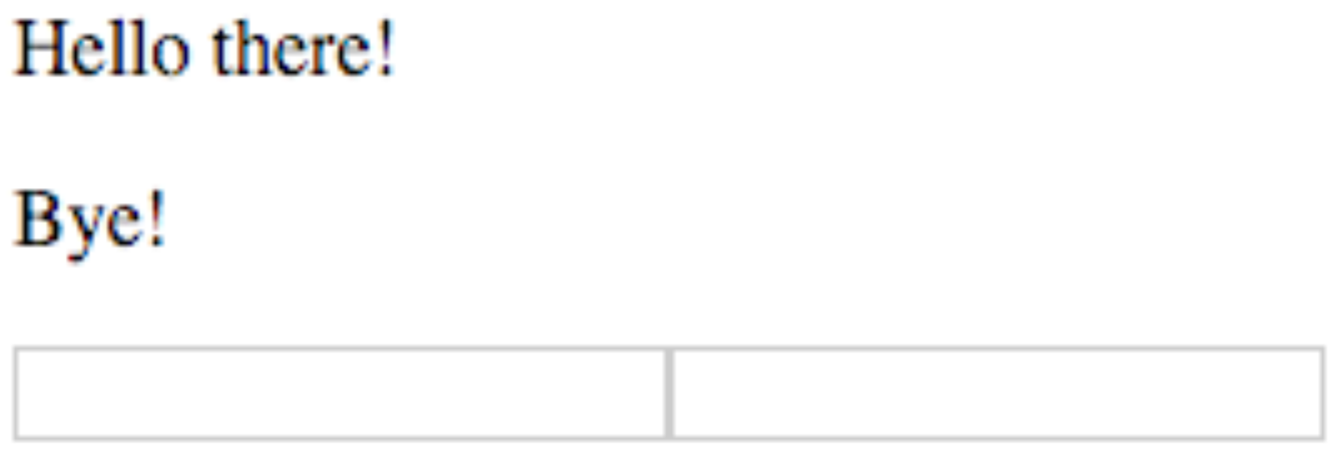
- Inline examples:

  `<input> <a> <img> <br>`

- Block example:

  `<p> <h1> <ul> <li> <table>`

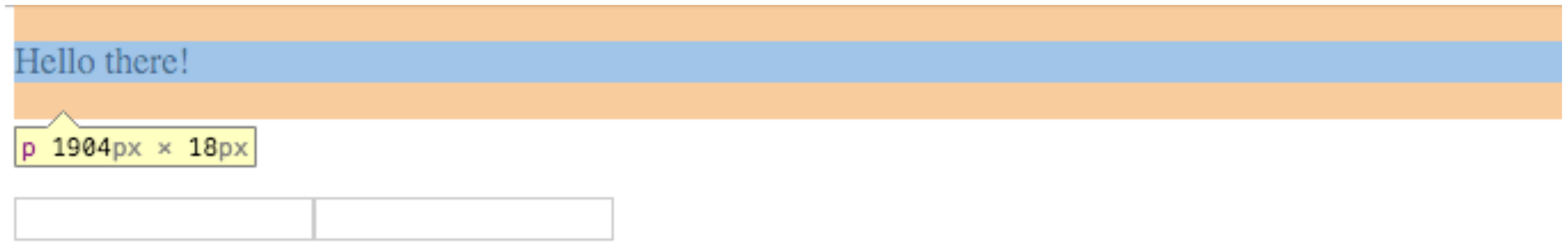
```html
<p>Hello there!</p>
<p>Bye!</p>

<input type="text">
<input type="text">
```



Hello there!

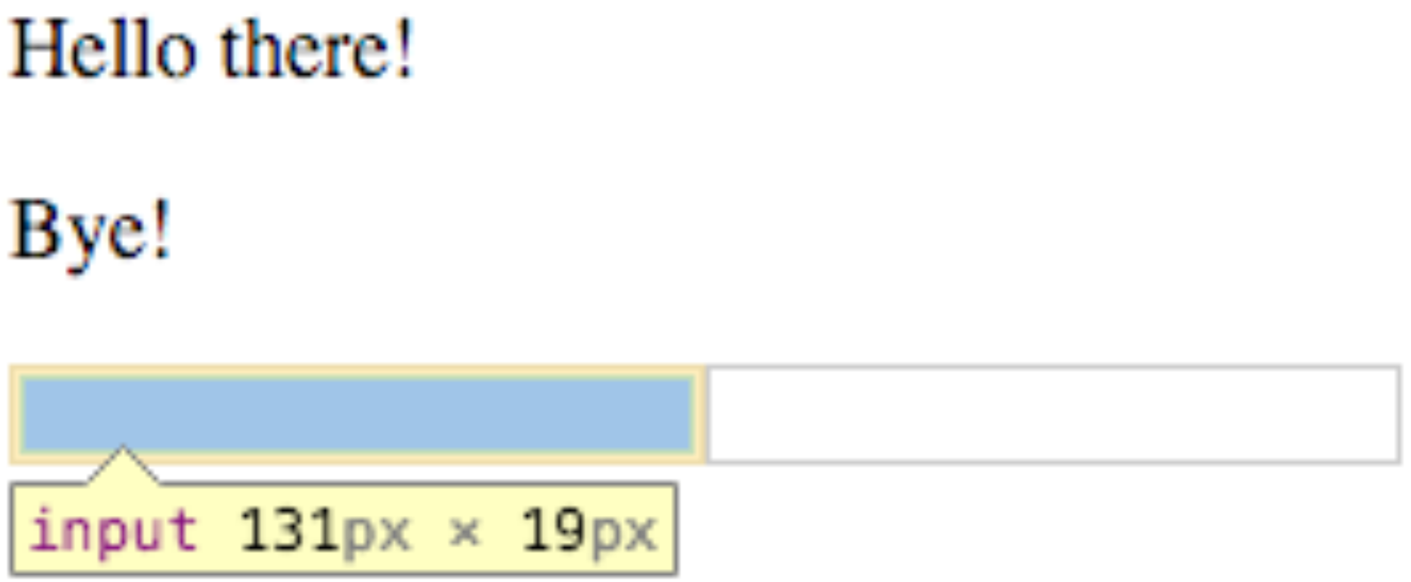
Bye!

If we use the inspect tool in Chrome



Hello there!

p 1904px × 18px



Hello there!

Bye!

input 131px × 19px

Each colour means something when inspected

Live example on CodePen

`display: inline-block`

- This type of display allows the element to behave like a block-level element
- The element will be displayed in line with other elements, and it will not begin on a new line by default

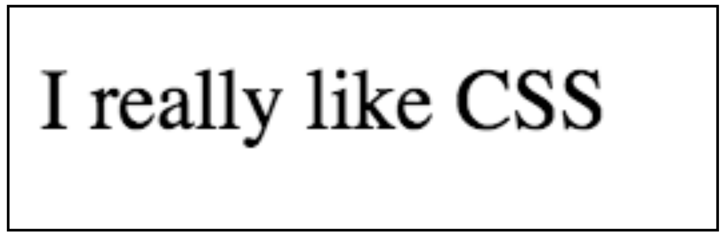
Why group elements together

- In order for our HTML code to be neat, readable and more importantly, re-usable
- We can manipulate similar elements together
- Separate the concerns

Span vs Div

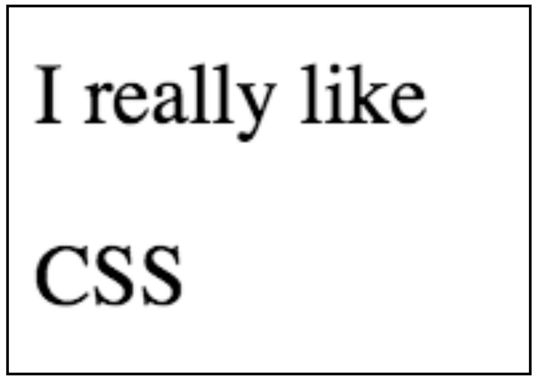
- Span creates an inline element

```
<p>I really like <span>CSS</span></p>
```


I really like CSS

- Div creates a block element

```
<p>I really like <div>CSS</div></p>
```


I really like

CSS

Width & Height

- The width and height properties can be used to resize block level elements and `<img>` elements

```
.profile-pic {
  width: 100px;
  height: 25px;
}
```

Live dimensions example on CodePen

overflow

- The overflow property states what would happen if our content is bigger that the specified dimensions.
- The default is **visible**, i.e flows outside
- **hidden** hides/masks the inner content
- **auto** brings up a scroll bar

```
div {
    width: 100px;
    height: 25px;
    overflow: ???;
  }
```

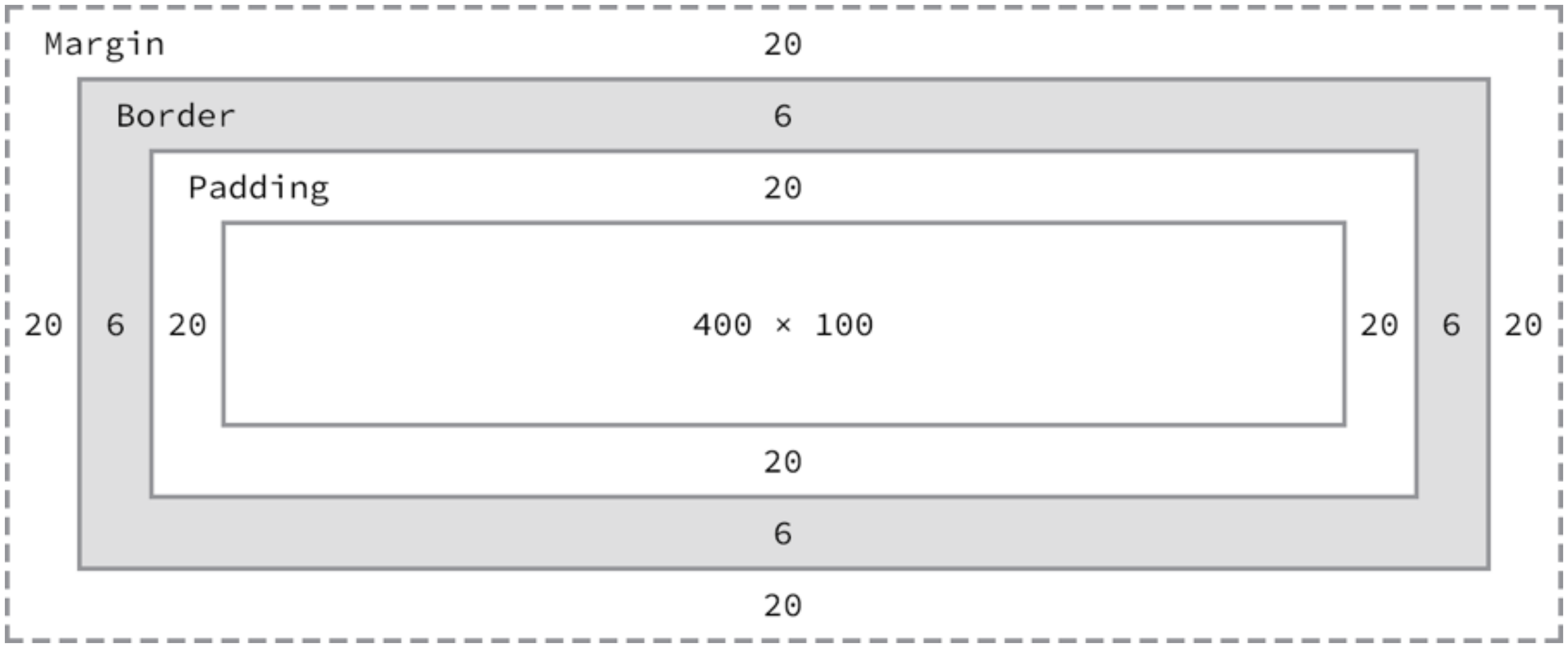
```
img {
   width: 100px;
   height: 25px;
}
```

overflow-x & overflow-y

- Using overflow-x and overflow-y, we can manually set the properties for the horizontal and vertical axes

- Maybe we want to have a scroll motion only for the vertical axis.

The Box Model



- The box model tells us that every element is a **rectangular box!**

- Each element may have the following properties:

  - width

  - height

  - padding

  - borders

  - margins

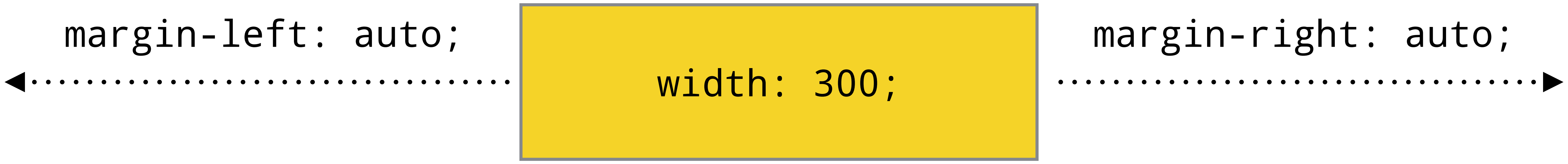
Margins

- This is the area that separates the box from the other boxes
- We can declare/change all the margins or individually

```
img {
    margin: 10px;
  }

img {
    margin: 10px 15px 20px 10px;
  }

img {
    margin-top: 10px;
  }
```

Auto-margin

- Auto margin is sometimes used to centre elements
- The said element has to have a **width**



Border

- The border property styles the edge around the box and is specified in the following order:

- Thickness, style, colour.

```
img {
    border: 2px solid red;
  }
```

Padding

- The padding is the "whitespace" between the border and the content

```
img {
    padding: 10px;
  }

img {
    padding: 10px 15px 20px 10px;
  }

img {
    padding-bottom: 10px;
  }
```

Positioning Content

- The position property is used to specify a positioning scheme for an element. The default is "static" which puts the element in the <u>**normal**</u> flow.

- In normal flow, **inline** boxes flow from left to right, wrapping to next line when needed.

- In normal flow, **block** boxes flow from top to bottom, making a new line after every box.

`position: relative`

- The **relative** value means that this is a new position relative to the original one
- To use **relative**, we have to add more properties, such as: `top`, `right`, `bottom`, `left`.

`position: absolute`

- Unlike, **relative,** this will take the element out of the normal flow & position with respect to the window.
- Window here means the whole <body>

Float

- The float property is used to float boxes on the sides of other boxes, allowing other content to flow around it. First used to wrap text around images.

- You can float non-image elements, as long as you specify a width for them first.

Clear

- The clear property can be used to specify that an element should *not* wrap around floated elements above it.