Thursday: Custom Css in Angular

In this lesson we'll go through the various ways we can add css styles to our Angular application to make it visually appealing.

There are several ways to add styling to an Angular application. Ideally, there are two major ways of doing this

- Application Level Styles
- Component Level Styles

Application Level Styles

While going through the Angular's folder structure, you may have noticed a file called styles.css. This is where we add style information that is relevant to the entire application.

Inside this file, we can either import an external stylesheet (like we did with the bootstrap stylesheet) or add the HTML class elements directly. However, whenever you decide to add your css styles directly here, you need to be very careful with the naming of the individual styles.

Let's add some application-level styles to our application.

src/styles.css

```
/* You can add global styles to this file, and also import other style files */

@import "~bootstrap/dist/css/bootstrap.css";

* {

box-sizing: border-box;

margin: 0;

padding: 0;

}

body {

font-family: Arial, Helvetica, sans-serif;

line-height: 1.4;

}
```

Component Level Styles

For every Angular component, we create we get a chance to define both the HTML template and the css file associated with that component in the component's metadata.

Note: The styles specified in `@Component` metadata *apply only within the template of that component*.

There are several ways to add styles to a component

- Setting styles in the @component metadata
- Setting external Css files in the @component metadata

- Inline in the template HTML.

Setting styles in the @component metadata

The first option involves adding a styles array property in the component's metadata

For example:

src/app/goal.component.ts

```
.............

@Component({

selector: 'app-goal',

templateUrl: './goal.component.html',

styles: ['h6 {color: red} ']

})

..........
```

This changes the h6 tag to red.
Note: these styles apply *only to this component*. They are *not inherited* by any components nested within the template nor by any content projected into the component.

Setting external Css files in the @component metadata

We can also load styles from external CSS files by adding a `styleUrls` property to a component's `@Component` metadata.

This option is the most used way of using css in angular applications.

src/app/goal.component.ts

```
.............

@Component({

selector: 'app-goal',

templateUrl: './goal.component.html',
styleUrls: ['./goal.component.css']
})

..........
```

src/app/goal.component.css

```
h6 {

   color: red;

}
```

Inline in the template HTML.

We can embed CSS styles directly into the HTML template by putting them inside `<style>` tags.

src/app/goal.component.html

```
<style>
     h6{

          color: red

     }

</style>

<div class="container mt-5">

......
```