COMP 431/531: Web Development

Lecture 13: Angular Features

Mack Joyner (mjoyner@rice.edu)

https://www.clear.rice.edu/comp431

Announcements & Reminders

HW #4 (Draft Front-end) due Tue. Oct. 21st at 11:59pm

classroom hw4 repo: https://classroom.github.com/a/hfLD_40C

IC 12 Solution

Transpilation

Modules provide us encapsulation. When *imported* (or *required*) a file is wrapped in an IIFE and provided to the caller as an object with "handles" to the default and optional exported members (functions, variables)

```
2   import particle, { update } from './particle'
3
4   const getLogger = (c, height) => {
5       const log = (msg) => {
6           if (!msg) {
7               log.x = 30
8               log.y = height
9           }
10          const pt = 16
11          c.font = `${pt}px Courier`
12          c.fillStyle = "white"
13          c.fillText(msg, log.x, log.y)
14          log.y = log.y - (4 + pt)
15      }
16      return log
17  }
18
19  const frameUpdate = (cb) => {
20      const rAF = (time) => {
21          requestAnimationFrame(rAF)
22          const diff = ~~(time - (rAF.lastTime || 0)) // ~~ is like floor
23          cb(diff)
24          rAF.lastTime = time
25      }
26      rAF() // go!
27  }
```


Transpilation

- Source-to-source compilation
- Compile next-gen JavaScript to today's JavaScript
- Heavily used prior to 2016 since most browsers didn't support ES2015
- Still used today (e.g. *import*), try it out (https://babeljs.io/repl)

Transpilation and Packing

bundle.js

```
/******/    // Load entry module and return exports
/******/    return __webpack_require__(0);
/******/ })
/************************************************************************
/******/ ([
/* 0 */
/***/ function(module, exports, __webpack_require__) {

    'use strict';

    var _slicedToArray = function () { function sliceIterator(arr, i) {

    var _particle = __webpack_require__(1);

    var _particle2 = _interopRequireDefault(_particle);

    function _interopRequireDefault(obj) { return obj && obj.__esModule

    window.onload = function () {
        var canvas = document.getElementById('app');
        var c = canvas.getContext("2d");
```


Webpack

index.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Physics</title>
  <meta name="viewport" content="width=device-width, initial-scale=1">
</head>

<body>
    <canvas id="app" width="800" height="550"></canvas>
    <script src="bundle.js"></script>
</body>
</html>

```


Angular Features

- How to create standalone classes
- Making AJAX REST calls to a server
  - HttpClient
  - Observables
- Importing the Material module to create a post
- Adding directives to alter DOM layout in template

Create a Angular Class

Create a class in Angular

>> *ng generate class posts/Post*

post.ts

```
1   export class Post {
2     public author:string;
3     public postDate:string;
4     public img:string;
5     public content:string;
6
7     constructor(author:string, postDate:string,
8       this.author = author;
9       this.postDate = postDate;
10      this.img = img;
11      this.content = content;
12    }
13  }
```

Import class in another file

post.component.ts

```
3   import {Post} from "./post";
```


HttpClient

- HttpClient is a library that simplifies http REST calls (e.g *get*)
- Assumes response is JSON (unlike fetch)
- REST call takes either a local path or url
- REST requests should occur in model (service)
- HttpClient REST call returns an Observable

app.config.ts

```
7     export const appConfig: ApplicationConfig = {
8       providers: [provideRouter(routes), provideHttpClient()]
9     };
```

posts.service.ts

```
2   import {HttpClient} from "@angular/common/http";

9   export class PostsService {

11    constructor(private http: HttpClient) { }

17    getPosts(appPosts) {
18      this.http.get("assets/articles.json").subscribe(response => {
        });
      }
```

Non-blocking call, returns observable

Observable

- Observables are objects facilitating message passing
- Observables are comparable to Promises
- HttpClient REST calls return an Observable
- Consumer can *unsubscribe()* to stop listening

app.config.ts

```
7    export const appConfig: ApplicationConfig = {
8      providers: [provideRouter(routes), provideHttpClient()]
9    };
```

posts.service.ts

```
2   import {HttpClient} from "@angular/common/http";

9   export class PostsService {

11    constructor(private http: HttpClient) { }

17    getPosts(appPosts) {
18      this.http.get("assets/articles.json").subscribe(response => {
        });
      }
```

Subscribe to listen for messages from server

Observable Subscribe

- Asynchronous call/event
- May receive multiple values, fetch (Promise) receives single value
- Response by default is JSON
- Response scope is *subscribe* function

app.config.ts

```
7    export const appConfig: ApplicationConfig = {
8      providers: [provideRouter(routes), provideHttpClient()]
9    };
```

posts.service.ts

```
2  import {HttpClient} from "@angular/common/http";

9  export class PostsService {

11    constructor(private http: HttpClient) { }

17    getPosts(appPosts) {
18      this.http.get("assets/articles.json").subscribe(response => {
        });
      }
```

Add additional get argument for alternate type: { responseType: 'text'}

Angular Material <mat-card>

- <mat-card-title> Card title
- <mat-card-subtitle> Card subtitle
- <mat-card-content> Primary card content. Intended for blocks of text
- <img mat-card-image> Card image. Stretches the image to the container width
- <mat-card-actions> Container for buttons at the bottom of the card
- <mat-card-footer> Section anchored to the bottom of the card
- <img mat-card-avatar> An image used as an avatar within the header

Source: https://material.angular.io/components/card/overview

Angular Material <mat-card> Example

- Install Angular Material

  >> ng add @angular/material

- Import MatCardModule (@angular/material/card) in component (twice) for <mat-card>
- Import MatSliderModule (@angular/material/slider) for <mat-slider>
- Built-in support for conditionally showing html elements (@for, @if), no import needed
- Component defines boolean showPosts field

Example

```
@if (showPosts) {
<div class="col-4">
  <mat-card class="mx-3 my-3">
    ...
   </mat-card>
 }
</div>
}
```

https://material.angular.io/components/card/overview

Structural Template Directives

- Alter layout: add, remove, replace DOM elements in template
- @if(selected) { … }

Add div if condition is true

- @for(id of profiles; track id) {<mat-card> … </mat-card>}

One card for each pField

- No cards created for empty profiles array
- Component defines boolean selected and array profiles fields