Angular 5 Training Course

Exercise H-hike

Setup

- Rebuild the project in **useful/hike**

```
npm install
ng serve --open
```

- A service reads data from **backpacks.json** and logs it to the browser console.

Display the data on screen

- Define a property in the main component that will contain the data read from the service;

```
packs;

draw( packs ) {
    this.packs = packs;
}
```

- Display the array in the template

```
{{ packs | json }}
```

Create a component for each pack

- Create a new component.

```
ng generate component pack --dry-run
ng generate component pack
```

- Iterate over instances of the new pack component within the ngFor.

```
<app-pack *ngFor="let p of packs"></app-pack>
```

Inputs in the pack component

- Define an Input in the pack component.

```
import { Input } from '@angular/core';
@Input() pack : Pack;
```

- Pass each pack as an **Input** into the pack component.

```
<app-pack *ngFor="let p of packs" [pack]="p"></app-pack>
```

- Test that the pack data is passed in by adding debugging to the ngOnInit lifecycle hook.
- *Note we should test the input property bindings in ngOnInit not the constructor.*

```
constructor() {}
ngOnInit() {
    console.log(this.pack);
}
```

- Change the pack template to display one pack.
- Note how the relative path for the image is constructed.

```
<section class="pack">
    <img src="{{'assets/packs/' + pack.image}}">
    <p>{{ pack.name }}</p>
    <p>{{ pack.description }}</p>
    <p>{{ pack.price }}</p>
    <p>{{ pack.code }}</p>
</section>
```

- Refactor the main template to use a **Flexbox** for layout.
- *This uses the .flex-packs rule defined in the main CSS file.*

```
<section class="flex-packs">
    <app-pack *ngFor="let p of packs" [pack]="p"></app-pack>
</section>
```

Define a custom Backpack type.

- Typescript allows us to define a custom type.
- Define an interface in file **app/pack/type.pack.ts**

```
interface BackPack{
    name: string;
    image: string;
    description: string;
    price: number;
    code: string;
}

export class Pack implements BackPack {
    constructor(
        public name: string,
        public image: string,
        public description: string,
        public price: number,
        public code: string
    ) {}
}
```

- Import the new Pack type into **src/app/app.component.ts**

```
import { Pack } from './pack/type.pack';
```

- Using Typescript define an array of type Pack.

```
packs:Pack[];
```

- Use this type in the service.

```
import { Pack } from '../pack/type.pack';
```

getData(path) { return this.http.get<Pack[]>(path); }

Using Promises with HTTP

- Compare the alternative Promise and Observable syntax for HTTP calls.

```
    getData(path) {
        return this.http.get<Pack[]>( path )
    }

    getDataWithPromise( path ) {
        return this.http.get<Pack[]>( path ).toPromise();
    }
```

- Code in the main component:

```
    ngOnInit() {
            this.getPacks();
            // this.getPacksWithPromise();
        }

getPacks() {

    this.ds.getData( this.path )

    .subscribe(
        data => this.packs = data,
        e => this.error( e )
     )

}

getPacksWithPromise() {

     this.ds.getDataWithPromise( this.path )
     .then(
         data => this.packs = data,
         e => this.error( e )
      );
}

error( e ) {
    // HttpErrorResponse
    console.log( e.status, e.statusText );
}
```