

## Table of Contents

Aug 27 <sup>th</sup> : Subjects with Project poke-api-example-with-subjects .....	3
Create a new project poke-api-example-with-subjects.....	3
Open a Git Bash window in your Angular project folder .....	3
Enter the following command .....	3
Open the new project folder in VS Code .....	3
Start the new Project .....	3
In the Project Folder, in a Git Bash window enter .....	3
Open the project in a browser .....	3
Remove all default html in app.component.html.....	4
Will use a module framework in this project.....	4
Create a new module.....	4
In the project folder with a new Git Bash window enter.....	4
Create components under the Poke folder .....	5
In the project folder's Git Bash window enter.....	5
In the project folder's Git Bash window enter.....	5
Components are automatically added to poke.module.ts .....	6
Add export statement to file to allow access to this module's components .....	6
Import the Poke component.....	6
Update the app.module.ts file to import the poke component.....	6
Test the imports.....	7
Add components to app.component.html .....	7
Update Project Files Expanding the Functionality .....	8
Create additional Angular project items.....	8
Create a service.....	8
Update the poke.service.ts file .....	8
Add a Pokemon Type .....	9
Create a source folder called model .....	9
Create a model file pokemon.ts.....	10
Add the following code to pokemon.ts.....	10

Update project files with their final changes.....	11
Update app.component.html .....	11
Replace all the code in app.component.html with .....	11
Update app.component.ts .....	12
Replace all the code in app.component.ts with .....	12
Update app.module.ts .....	13
Replace all the code in app.module.ts with .....	13
Update latest-poke.component.ts .....	14
Replace all the code in latest-poke.component.ts with .....	14
Update latest-poke.component.html .....	15
Replace all the code in latest-poke.component.html with .....	15
Update poke-table.component.ts.....	15
Replace all the code in poke-table.component.ts with .....	15
Update poke-table.component.html .....	16
Replace all the code in poke-table.component.html with .....	16
Add global styling to the project.....	17
Add bulma open source css styling .....	17
Install the bulma library .....	17
Add the following to styles.css.....	17
Styling changes:.....	18
Final Web Page Results .....	19
Appendix: rxjs-observables-and-subjects.md .....	20

## Aug 27<sup>th</sup>: Subjects with Project poke-api-example-with-subjects

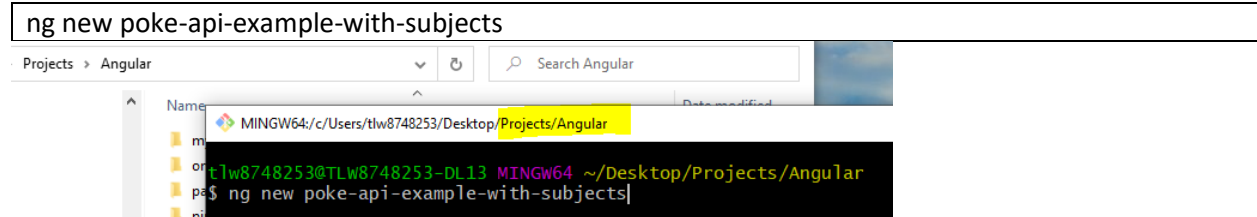
Create a new project poke-api-example-with-subjects

**Note:** this is one of the worst lectures given. He flip flops all over the place. Creates code that does not work, then back pedals to get it to work. This is yet again another reason why there should have been prepared lessons plans instead of just typing on the go.

Open a Git Bash window in your Angular project folder

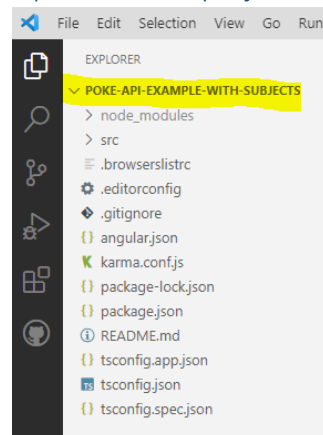
*Enter the following command*

```
ng new poke-api-example-with-subjects
```



Creating through Git Bash will automatically create a Git Hub repository that you can push to your Git Hub repository.

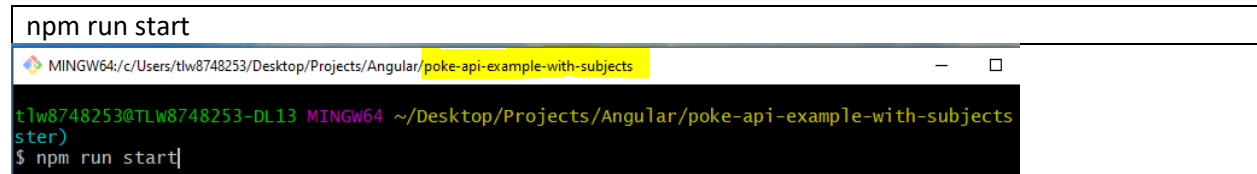
Open the new project folder in VS Code



Start the new Project

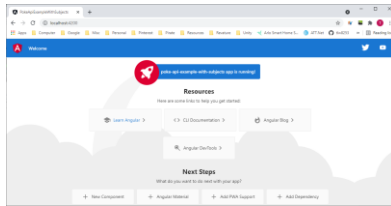
*In the Project Folder, in a Git Bash window enter*

```
npm run start
```

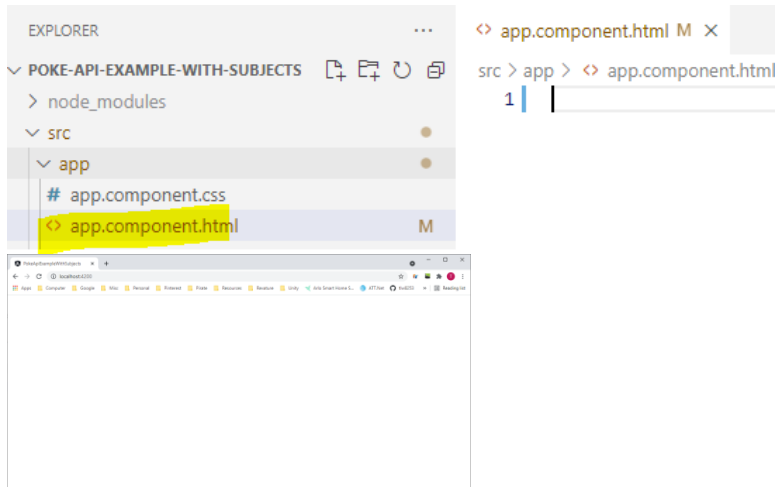


*Open the project in a browser*

```
http://localhost:4200/
```



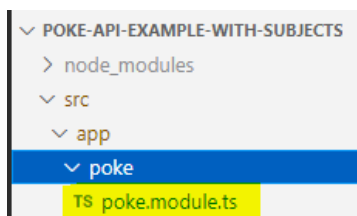
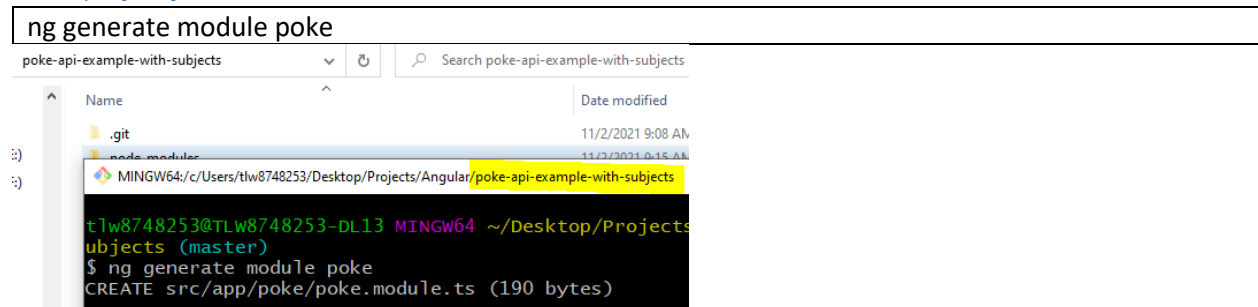
Remove all default html in app.component.html



Will use a module framework in this project

Create a new module

In the project folder with a new Git Bash window enter



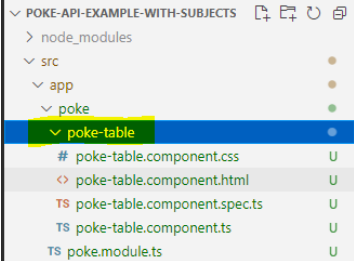
Create components under the Poke folder

*In the project folder's Git Bash window enter*

ng generate component poke/poke-table

```
$ ng generate module poke
CREATE src/app/poke/poke.module.ts (190 bytes)

tlw8748253@TLW8748253-DL13 MINGW64 ~/Desktop/Projects/Angular/poke-api-example-with-subjects (master)
$ ng generate component poke/poke-table
CREATE src/app/poke/poke-table/poke-table.component.html (25 bytes)
CREATE src/app/poke/poke-table/poke-table.component.spec.ts (648 bytes)
CREATE src/app/poke/poke-table/poke-table.component.ts (290 bytes)
CREATE src/app/poke/poke-table/poke-table.component.css (0 bytes)
UPDATE src/app/poke/poke.module.ts (288 bytes)
```

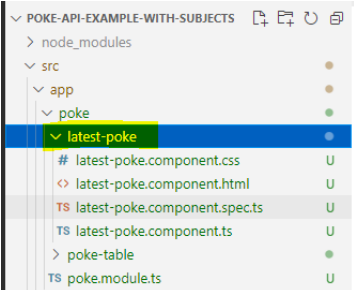


*In the project folder's Git Bash window enter*

ng generate component poke/latest-poke

```
MINGW64/c:/Users/tlw8748253/Desktop/Projects/Angular/poke-api-example-with-subjects

tlw8748253@TLW8748253-DL13 MINGW64 ~/Desktop/Projects/Angular/poke-api-example-with-subjects (master)
$ ng generate component poke/latest-poke
CREATE src/app/poke/latest-poke/latest-poke.component.html (26 bytes)
CREATE src/app/poke/latest-poke/latest-poke.component.spec.ts (655 bytes)
CREATE src/app/poke/latest-poke/latest-poke.component.ts (294 bytes)
CREATE src/app/poke/latest-poke/latest-poke.component.css (0 bytes)
UPDATE src/app/poke/poke.module.ts (388 bytes)
```



*Components are automatically added to poke.module.ts*

```
<> app.component.html M TS poke.module.ts U X
src > app > poke > TS poke.module.ts > ...
1 import { NgModule } from '@angular/core';
2 import { CommonModule } from '@angular/common';
3 import { PokeTableComponent } from '../poke-table/poke-table.component';
4 import { LatestPokeComponent } from '../latest-poke/latest-poke.component';
5
6
7
8 @NgModule({
9   declarations: [
10     PokeTableComponent,
11     LatestPokeComponent
12   ],
13   imports: [
14     CommonModule
15   ]
16 })
17 export class PokeModule { }
18
```

*Add export statement to file to allow access to this module's components*

```
,
exports: [
  PokeTableComponent,
  LatestPokeComponent
]
```

```
src > app > poke > TS poke.module.ts > ...
1 import { NgModule } from '@angular/core';
2 import { CommonModule } from '@angular/common';
3 import { PokeTableComponent } from '../poke-table/poke-table.component';
4 import { LatestPokeComponent } from '../latest-poke/latest-poke.component';
5
6
7
8 @NgModule({
9   declarations: [
10     PokeTableComponent,
11     LatestPokeComponent
12   ],
13   imports: [
14     CommonModule
15   ],
16   exports: [
17     PokeTableComponent,
18     LatestPokeComponent
19   ]
20 })
21 export class PokeModule { }
22
```

Import the Poke component

The new components are not visible in the main app module app.module.ts

*Update the app.module.ts file to import the poke component*

Add PokeModule to import section

```
imports: [
  BrowserModule,
  PokeModule
],
```

This will automatically add an import statement as well as long as you type the import and do not copy and paste.

```

src > app > ts app.module.ts > ...
1  import { NgModule } from '@angular/core';
2  import { BrowserModule } from '@angular/platform-browser';
3
4  import { AppComponent } from './app.component';
5  import { PokeModule } from './poke/poke.module';
6
7  @NgModule({
8    declarations: [
9      AppComponent
10   ],
11   imports: [
12     BrowserModule,
13     PokeModule
14   ],
15   providers: [],
16   bootstrap: [AppComponent]
17 })
18 export class AppModule { }
19

```

Adding the PokeModule gains access to all of the module's components

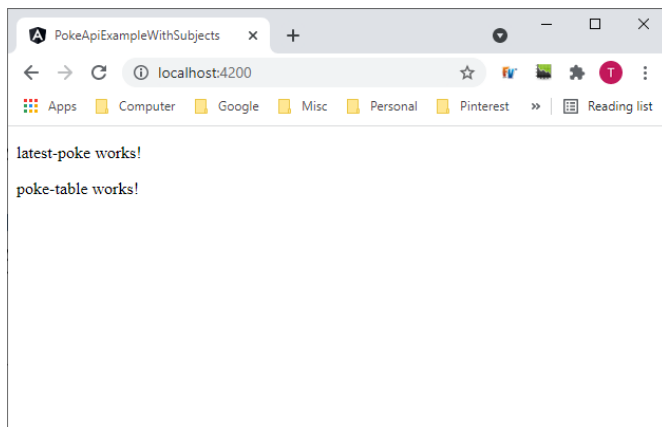
Test the imports

*Add components to app.component.html*

```

<app-latest-poke></app-latest-poke>
<app-poke-table></app-poke-table>

```



This is a basic example of using modules to structure the Angular project encapsulating the modules in related project folders / components.

## Update Project Files Expanding the Functionality

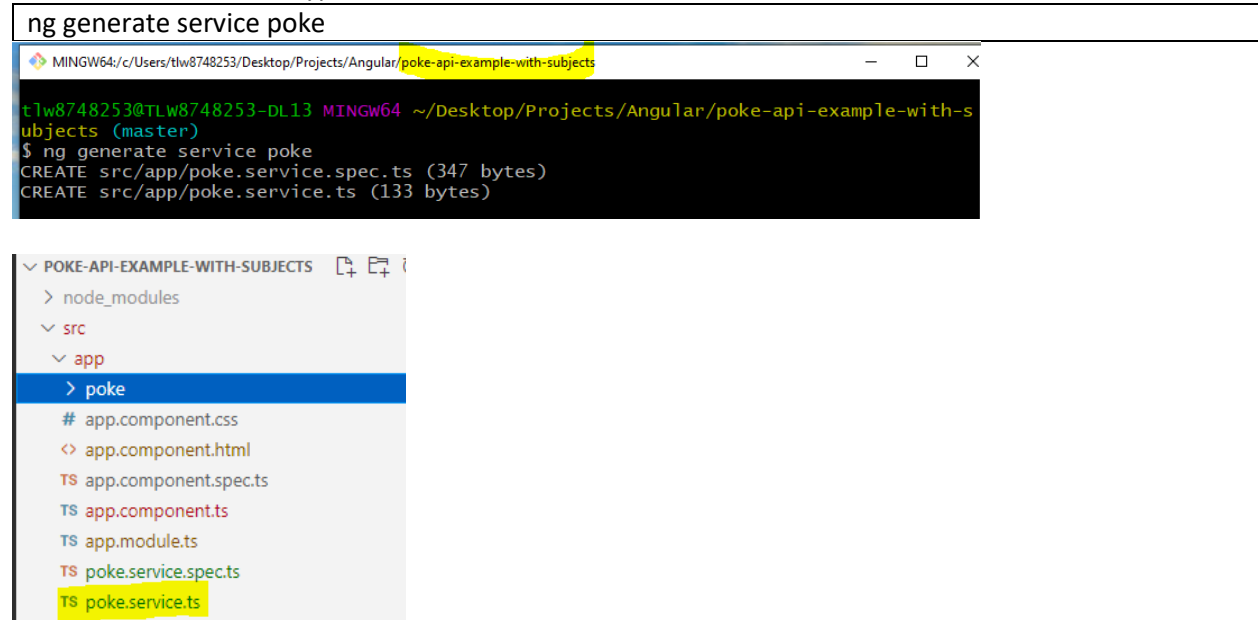
From this point forward we are going to complete file updates to the final project. This is due to the confusing way this lecture and project is being presented in the recording. You can try and follow the lecture, best of luck.

## Create additional Angular project items

### Create a service

In the Git Bash window type

```
ng generate service poke
```



```
tlw8748253@tlw8748253-DL13 MINGW64 ~/Desktop/Projects/Angular/poke-api-example-with-s
ubjects (master)
$ ng generate service poke
CREATE src/app/poke.service.spec.ts (347 bytes)
CREATE src/app/poke.service.ts (133 bytes)
```

POKE-API-EXAMPLE-WITH-SUBJECTS

- node\_modules
- src
  - app
    - poke
      - app.component.css
      - app.component.html
      - app.component.spec.ts
      - app.component.ts
      - app.module.ts
      - poke.service.spec.ts
      - poke.service.ts

### Update the `poke.service.ts` file

Replace the code in the file with the following

```
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import { Observable } from 'rxjs';
import { Pokemon } from '../model/pokemon';
import { Subject } from 'rxjs';

@Injectable({
  providedIn: 'root'
})
export class PokeService {

  private subject: Subject<Pokemon>

  constructor(private http: HttpClient) {
    this.subject = new Subject();
  }
}
```



```

getPokemonById(id: number): void {
  this.http.get<Pokemon>(`https://pokeapi.co/api/v2/pokemon/${id}`).subscribe((pokemon) => {
    this.subject.next(pokemon); // Here we are publishing the pokemon to the subject
    // Any subscriber to the subject will be able to receive this pokemon data
  });
}

getSubject(): Subject<Pokemon> {
  return this.subject;
}
}

```

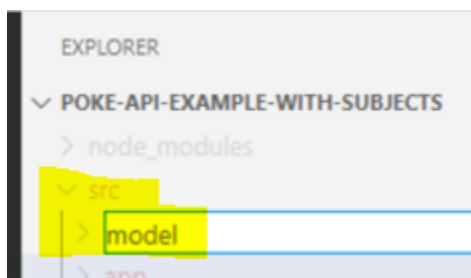
```

src > app > TS pokeService.ts > ...
1 import { Injectable } from '@angular/core';
2 import { HttpClient } from '@angular/common/http';
3 import { Observable } from 'rxjs';
4 import { Pokemon } from '../model/pokemon';
5 import { Subject } from 'rxjs';
6
7 @Injectable({
8   providedIn: 'root'
9 })
10 export class PokeService {
11
12   private subject: Subject<Pokemon>
13
14   constructor(private http: HttpClient) {
15     this.subject = new Subject();
16   }
17
18   getPokemonById(id: number): void {
19     this.http.get<Pokemon>(`https://pokeapi.co/api/v2/pokemon/${id}`).subscribe((pokemon) => {
20       this.subject.next(pokemon); // Here we are publishing the pokemon to the subject
21       // Any subscriber to the subject will be able to receive this pokemon data
22     });
23   }
24
25   getSubject(): Subject<Pokemon> {
26     return this.subject;
27   }
28
29 }
30

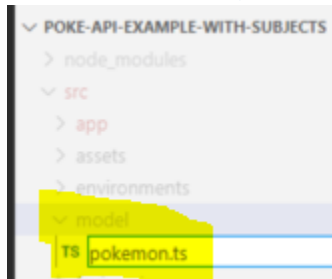
```

### Add a Pokemon Type

Create a source folder called model



Create a model file pokemon.ts



Add the following code to pokemon.ts

This information is to receive information from <https://pokeapi.co/>

```
export interface Pokemon {  
  id: number,  
  name: string,  
  height: number,  
  weight: number,  
  base_experience: number,  
  sprites: { front_default: string }  
}
```

```
src > model > TS pokemon.ts > ...  
1  export interface Pokemon {  
2    id: number,  
3    name: string,  
4    height: number,  
5    weight: number,  
6    base_experience: number,  
7    sprites: { front_default: string }  
8  }
```

Update project files with their final changes

Update app.component.html

Replace all the code in app.component.html with

```
<div class="container">
  <div>
    <label>Pokemon ID:</label>
    <input [(ngModel)]="pokemonIdInput" type="number"/>
  </div>
  <div>
    <button (click)="onGetPokemonClick()">Get Pokemon</button>
  </div>
</div>

<section class="section">
  <div class="container">
    <div class="columns">
      <div class="column is-6">
        <app-latest-poke></app-latest-poke>
      </div>
      <div class="column is-6">
        <app-poke-table></app-poke-table>
      </div>
    </div>
  </div>
</section>
```

The line `<input [(ngModel)]="pokemonIdInput" type="number"/>` is two way data binding.

```
src > app > < app.component.html >...
1  <div class="container">
2    <div>
3      <label>Pokemon ID:</label>
4      <input [(ngModel)]="pokemonIdInput" type="number"/>
5    </div>
6    <div>
7      <button (click)="onGetPokemonClick()">Get Pokemon</button>
8    </div>
9  </div>
10
11 <section class="section">
12   <div class="container">
13     <div class="columns">
14       <div class="column is-6">
15         <app-latest-poke></app-latest-poke>
16       </div>
17       <div class="column is-6">
18         <app-poke-table></app-poke-table>
19       </div>
20     </div>
21   </div>
22 </section>
```

Update app.component.ts

Replace all the code in app.component.ts with

```
import { Component } from '@angular/core';
import { PokeService } from './poke.service';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {

  pokemonIdInput: number = 0;

  constructor(private pokeService: PokeService) {}

  onGetPokemonClick() {
    this.pokeService.getPokemonById(this.pokemonIdInput);
  }

}
```

```
src > app > TS app.component.ts > ...
1 | import { Component } from '@angular/core';
2 | import { PokeService } from './poke.service';
3 |
4 | @Component({
5 |   selector: 'app-root',
6 |   templateUrl: './app.component.html',
7 |   styleUrls: ['./app.component.css']
8 | })
9 | export class AppComponent {
10 |
11 |   pokemonIdInput: number = 0;
12 |
13 |   constructor(private pokeService: PokeService) {}
14 |
15 |   onGetPokemonClick() {
16 |     this.pokeService.getPokemonById(this.pokemonIdInput);
17 |   }
18 |
19 | }
```

## Update app.module.ts

Replace all the code in app.module.ts with

```
import { HttpClientModule } from '@angular/common/http';
import { NgModule } from '@angular/core';
import { FormsModule } from '@angular/forms';
import { BrowserModule } from '@angular/platform-browser';

import { AppComponent } from './app.component';
import { PokeModule } from './poke/poke.module';

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    HttpClientModule,
    FormsModule,
    PokeModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

```
src > app > TS app.module.ts >
1  import { HttpClientModule } from '@angular/common/http';
2  import { NgModule } from '@angular/core';
3  import { FormsModule } from '@angular/forms';
4  import { BrowserModule } from '@angular/platform-browser';
5
6  import { AppComponent } from './app.component';
7  import { PokeModule } from './poke/poke.module';
8
9  @NgModule({
10   declarations: [
11     AppComponent
12   ],
13   imports: [
14     BrowserModule,
15     HttpClientModule,
16     FormsModule,
17     PokeModule
18   ],
19   providers: [],
20   bootstrap: [AppComponent]
21 })
22 export class AppModule { }
23
```

Update latest-poke.component.ts

*Replace all the code in latest-poke.component.ts with*

```
import { Component, OnInit } from '@angular/core';
import { PokeService } from 'src/app/poke.service';
import { Pokemon } from 'src/model/pokemon';

@Component({
  selector: 'app-latest-poke',
  templateUrl: './latest-poke.component.html',
  styleUrls: ['./latest-poke.component.css']
})
export class LatestPokeComponent implements OnInit {

  latestPokemon!: Pokemon;

  constructor(private pokeService: PokeService) {}

  ngOnInit(): void {
    const subject = this.pokeService.getSubject();

    subject.subscribe((pokemon) => {
      this.latestPokemon = pokemon;
    });
  }
}
```

```
src > app > poke > latest-poke > TS latest-poke.component.ts > ...
1  import { Component, OnInit } from '@angular/core';
2  import { PokeService } from 'src/app/poke.service';
3  import { Pokemon } from 'src/model/pokemon';
4
5  @Component({
6    selector: 'app-latest-poke',
7    templateUrl: './latest-poke.component.html',
8    styleUrls: ['./latest-poke.component.css']
9  })
10 export class LatestPokeComponent implements OnInit {
11
12     latestPokemon!: Pokemon;
13
14     constructor(private pokeService: PokeService) {}
15
16     ngOnInit(): void {
17         const subject = this.pokeService.getSubject();
18
19         subject.subscribe((pokemon) => {
20             this.latestPokemon = pokemon;
21         });
22     }
23
24 }
25 |
```

## Update latest-poke.component.html

Replace all the code in latest-poke.component.html with

```
<h1>Latest Pokemon Added:</h1>
<div>
  <ul>
    <li>{{ latestPokemon.id }}</li>
    <li>{{ latestPokemon.name }}</li>
    <li>{{ latestPokemon.height }}</li>
    <li>{{ latestPokemon.weight }}</li>
    <li>{{ latestPokemon.base_experience }}</li>
    <li><img [src]="latestPokemon.sprites.front_default" [alt]="image of ' + latestPokemon.name"></li>
  </ul>
</div>
```

```
src > app > poke > latest-poke > <> latest-poke.component.html > ...
1  <h1>Latest Pokemon Added:</h1>
2  <div>
3    <ul>
4      <li>{{ latestPokemon.id }}</li>
5      <li>{{ latestPokemon.name }}</li>
6      <li>{{ latestPokemon.height }}</li>
7      <li>{{ latestPokemon.weight }}</li>
8      <li>{{ latestPokemon.base_experience }}</li>
9      <li><img [src]="latestPokemon.sprites.front_default" [alt]="image of ' + latestPokemon.name"></li>
10   </ul>
11 </div>
..
```

## Update poke-table.component.ts

Replace all the code in poke-table.component.ts with

```
import { Component, OnInit } from '@angular/core';
import { PokeService } from 'src/app/poke.service';
import { Pokemon } from 'src/model/pokemon';

@Component({
  selector: 'app-poke-table',
  templateUrl: './poke-table.component.html',
  styleUrls: ['./poke-table.component.css']
})
export class PokeTableComponent implements OnInit {

  pokemonArr: Pokemon[] = [];

  constructor(private pokeService: PokeService) { }

  ngOnInit(): void {
    const subject = this.pokeService.getSubject();

    subject.subscribe((pokemon) => {
      this.pokemonArr.push(pokemon);
    });
  }
}
```

```
}  
  
}
```

```
src > app > poke > poke-table > Ts poke-table.component.ts > ...  
1 import { Component, OnInit } from '@angular/core';  
2 import { PokeService } from 'src/app/poke.service';  
3 import { Pokemon } from 'src/model/pokemon';  
4  
5 @Component({  
6   selector: 'app-poke-table',  
7   templateUrl: './poke-table.component.html',  
8   styleUrls: ['./poke-table.component.css']  
9 })  
10 export class PokeTableComponent implements OnInit {  
11  
12   pokemonArr: Pokemon[] = [];  
13  
14   constructor(private pokeService: PokeService) { }  
15  
16   ngOnInit(): void {  
17     const subject = this.pokeService.getSubject();  
18  
19     subject.subscribe((pokemon) => {  
20       this.pokemonArr.push(pokemon);  
21     });  
22   }  
23  
24 }
```

## Update poke-table.component.html

*Replace all the code in poke-table.component.html with*

```
<h1>Pokemon Table:</h1>  
  
<table>  
  <thead>  
    <tr>  
      <th>id</th>  
      <th>name</th>  
      <th>height</th>  
      <th>weight</th>  
      <th>base experience</th>  
      <th>image</th>  
    </tr>  
  </thead>  
  <tbody>  
    <ng-container *ngFor="let pokemon of pokemonArr; let i = index;">  
      <tr>  
        <td>{{ pokemon.id }}</td>  
        <td>{{ pokemon.name }}</td>  
        <td>{{ pokemon.height }}</td>  
        <td>{{ pokemon.weight }}</td>  
        <td>{{ pokemon.base_experience }}</td>  
        <td><img [src]="pokemon.sprites.front_default" [alt]="'image of ' + pokemon.name"></td>
```



```

    </tr>
  </ng-container>
</tbody>
</table>

```

```

src > app > poke > poke-table > < poke-table.component.html > ...
1  <h1>Pokemon Table:</h1>
2
3  <table>
4    <thead>
5      <tr>
6        <th>id</th>
7        <th>name</th>
8        <th>height</th>
9        <th>weight</th>
10       <th>base experience</th>
11       <th>image</th>
12     </tr>
13   </thead>
14   <tbody>
15     <ng-container *ngFor="let pokemon of pokemonArr; let i = index;">
16       <tr>
17         <td>{{ pokemon.id }}</td>
18         <td>{{ pokemon.name }}</td>
19         <td>{{ pokemon.height }}</td>
20         <td>{{ pokemon.weight }}</td>
21         <td>{{ pokemon.base_experience }}</td>
22         <td><img [src]="pokemon.sprites.front_default" [alt]='image of ' + pokemon.name"></td>
23       </tr>
24     </ng-container>
25   </tbody>
26 </table>

```

Add global styling to the project

Add bulma open source css styling

<https://bulma.io/>

*Install the bulma library*

In the Git Bash window type

```
npm install bulma
```

```
MINGW64: c:/Users/tlw8748253/Desktop/Projects/Angular/poke-api-example-with-subjects
```

```

tlw8748253@TLW8748253-DL13 MINGW64 ~/Desktop/Projects/A
ubjects (master)
$ npm install bulma

```

*Add the following to styles.css*

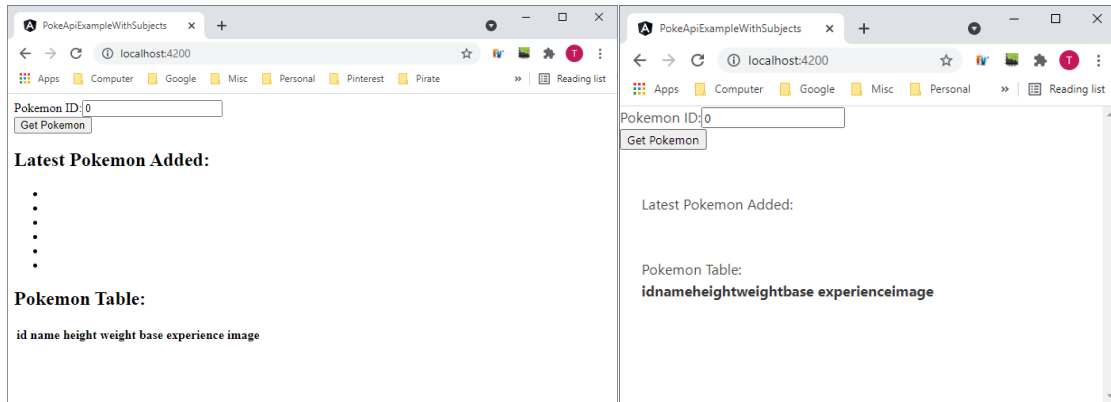
```
@import 'bulma/css/bulma.min.css';
```

```

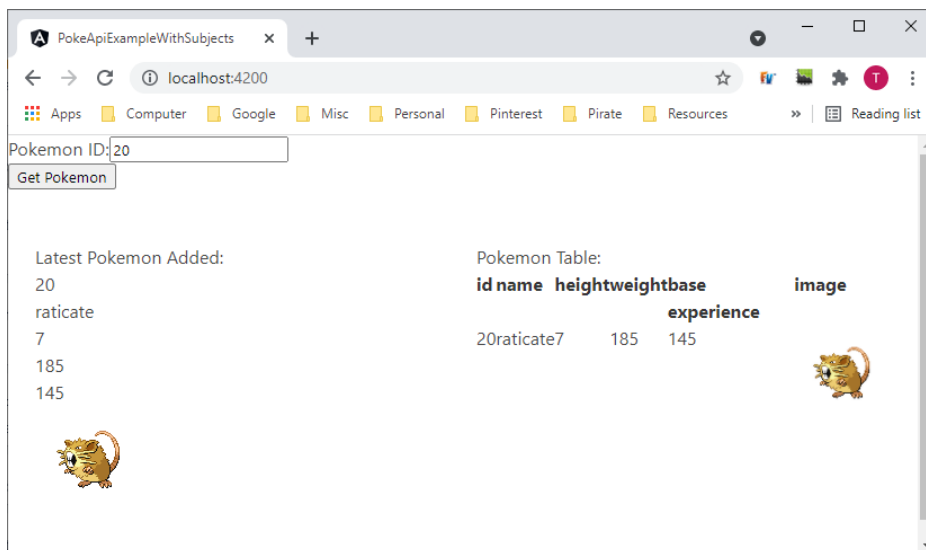
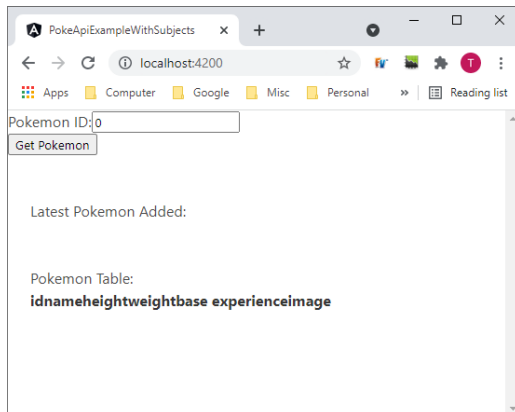
src > # styles.css
1  /* You can add global styles to this file, and also import other style files */
2  @import 'bulma/css/bulma.min.css';
3

```

### Styling changes:



## Final Web Page Results



Addition information on subjects is found in the notes in the appendix:

[Appendix: rxjs-observables-and-subjects.md](#)

## Appendix: rxjs-observables-and-subjects.md

### # RxJS Observables and Subjects

#### ## Observables

Observables are from the RxJS library, which the developers of Angular simply decided to utilize within the Angular framework. Therefore this preference of the Angular developers is what led to the fact that we deal with Observables instead of Promises when it comes to the HttpClient (unlike with fetch).

- Similar to promises, observables require some callback function for the data to be passed into once the data is made available
  - Where they differ is in the fact that observables can have multiple values instead of one, OVER TIME
- You can have multiple subscribers to an observable, while you can only have a single subscriber to a promise
- Observables can be cancelled, Promises cannot
- Overall, however, promises and observables are pretty similar
  - You can actually convert from an observable to a Promise
  - But, this should only be done if you are expecting only a single value from the Observable (such as when you utilize HttpClient)
  - Because HttpClient will send a single request, and receive a single response

#### ## Subjects

- Subjects follow the pattern of pub-sub (publisher - subscriber)
- We can publish values to a subject (using .next(<value>))
- Any subscriber to the subject will then receive that "update"
  - An analogy would be a weather channel where we publish weather updates to all viewers and listeners