Subject Exercise

0. Start by creating a new application → ng new subjectTraining. Then run the command 'ng serve -o' to open the application.

1. Use the following API to get your data http://angular.at/api/flight?from=F

2. Import ReactiveFormsModule and HttpClientModule into app.module.ts

```
[..]
import { ReactiveFormsModule } from '@angular/forms';
import { HttpClientModule } from '@angular/common/http';
import { FlightsComponent } from './flights/flights.component';

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

3. Create a flight.model.ts interface to represent flights data

```
export interface Flight {
  id: number;
  from: string;
  to: string;
  date: string;
  delayed: boolean;
}
```

4. Create a new Flights Component under src> app → ng g c employeeList

5. Inject HttpClient into the constructor

```
constructor(private http: HttpClient) {}
```

6. Create a new subject called flight
7. Using the subject variable created, create an observable to retrieve the data from http://www.angular.at/api/flight?from=F, using the 'http' declaration you have made earlier.

```
flightSubject = new Subject();

 flightSubject$: Observable<Flight[]> = this.flightSubject.asObservable()
  .pipe(
   exhaustMap((resp) => {
     const url = 'http://www.angular.at/api/flight?from=F';
     const headers = new HttpHeaders().set('Accept', 'application/json');
     return this.http.get<Flight[]>(url, { headers });
   })
 );
```

8. Display the result as a table in flights.component.html

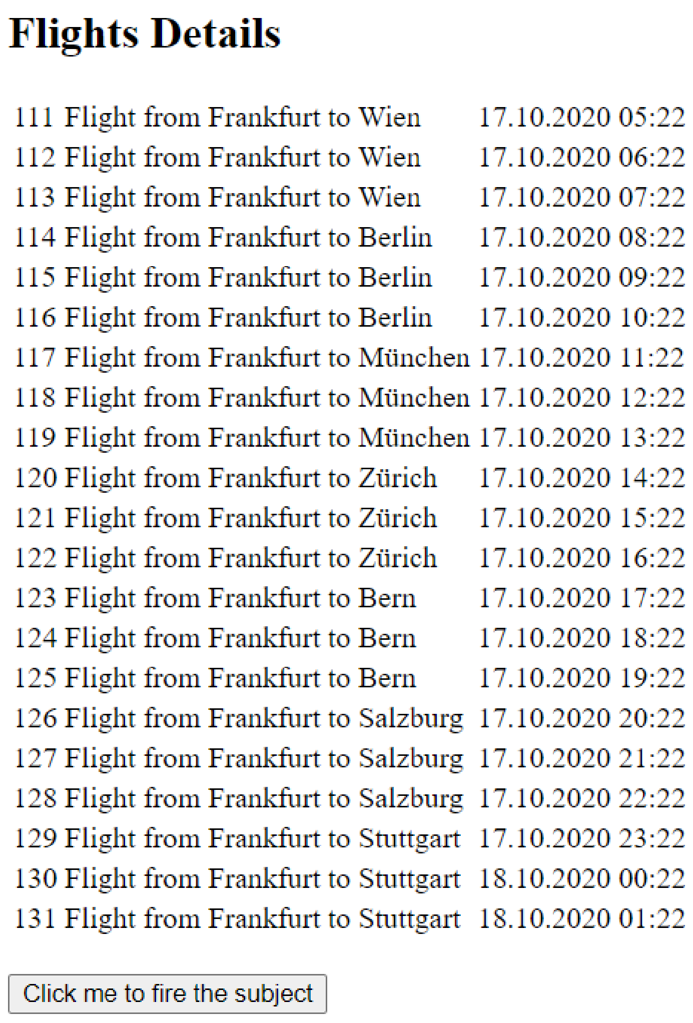
```
<h2 class="title">Flights Details</h2>

<table class="table table-striped">
  <tr *ngFor="let f of flights$ | async">
    <td>{{ f.id }}</td>
    <td>Flight from {{ f.from }} to {{ f.to }}</td>
    <td *ngIf="f.delayed">Delayed</td>
    <td>{{ f.date | date: "dd.MM.yyyy HH:mm" }}</td>
  </tr>
</table>
<br />
```

9. Notice that you are not able to see the results of the flight details, and from Inspect> Network, there no outgoing API call. You would need to call the .next() method to trigger the subject.

```
  clickAction(): any {
    this.flightSubject.next();
  }
```

```
<button (click)="clickAction()">Click me to fire the subject
</button>
```

Result



Flights Details

111	Flight from Frankfurt to Wien	17.10.2020 05:22
112	Flight from Frankfurt to Wien	17.10.2020 06:22
113	Flight from Frankfurt to Wien	17.10.2020 07:22
114	Flight from Frankfurt to Berlin	17.10.2020 08:22
115	Flight from Frankfurt to Berlin	17.10.2020 09:22
116	Flight from Frankfurt to Berlin	17.10.2020 10:22
117	Flight from Frankfurt to München	17.10.2020 11:22
118	Flight from Frankfurt to München	17.10.2020 12:22
119	Flight from Frankfurt to München	17.10.2020 13:22
120	Flight from Frankfurt to Zürich	17.10.2020 14:22
121	Flight from Frankfurt to Zürich	17.10.2020 15:22
122	Flight from Frankfurt to Zürich	17.10.2020 16:22
123	Flight from Frankfurt to Bern	17.10.2020 17:22
124	Flight from Frankfurt to Bern	17.10.2020 18:22
125	Flight from Frankfurt to Bern	17.10.2020 19:22
126	Flight from Frankfurt to Salzburg	17.10.2020 20:22
127	Flight from Frankfurt to Salzburg	17.10.2020 21:22
128	Flight from Frankfurt to Salzburg	17.10.2020 22:22
129	Flight from Frankfurt to Stuttgart	17.10.2020 23:22
130	Flight from Frankfurt to Stuttgart	18.10.2020 00:22
131	Flight from Frankfurt to Stuttgart	18.10.2020 01:22