Lab Routes

Exercise 1: Activate routing

Currently, we've done pretty much everything inside of AppComponent. In this exercise, we'll make the AppComponent responsible for being the container where all the views are loaded. All its content will be moved to a new HomeComponent.

1. In `src/app`, copy `app.component.ts` and name the file `home.component.ts`. Inside the file, name the component HomeComponent, set the `templateUrl` property to `home.components.ts` and remove the `selector` and `styleUrls` properties.

2. Remove the content from AppComponent, leaving it an empty component:

```
import { Component } from '@angular/core';

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

}
```

3. In `src/app`, create a new file `home.component.html`. Place the content from `app.component.html` inside this file, leaving `app.component.html` empty.

4. In `app.component.html`, define a `<router-outlet></router-outlet>` element where views will be dynamically inserted.

5. Declare the HomeComponent as part of your module inside of `app.module.ts`:

```
import { HomeComponent } from './home.component';

// ...

@NgModule({
    declarations: [..., HomeComponent],
    imports: [...],
    providers: [...],
    bootstrap: [...]
})
export class AppModule { }
```

6. Time to define routes. Create an `app.routes.ts` and define the first route.

```
import { Route } from '@angular/router';
import { HomeComponent } from './home.component';

export const routes: Route[] = [
    { path: '', component: HomeComponent }
];
```

7. Activate routing by going go to *app.module.ts* and using the `RouterModule` with the defined routes.

```
import { RouterModule } from '@angular/router';
import { HomeComponent } from './home.component';
import { routes } from './app.routes';

// ...

@NgModule({
    declarations: [..., HomeComponent],
    imports: [..., RouterModule.forRoot(routes)],
    providers: [...],
    bootstrap: [...]
})
export class AppModule { }
```

Our application should now work as it did before! With the added benefit that we can now start adding more pages.

Exercise 2: The invitation page

In this exercise, we will be creating an invitation page. Here we can:

- Describe what we'll be celebrating (the event).
- Enter three possible dates to host the event.
- Select who we want to invite.

These steps should largely resemble the steps of the previous exercise.

1. In *src/app/*, create an `InviteComponent` by creating a `.ts` and `.html` file.
2. In the `.ts` file, define a class with `@Component()` applied. Again, you don't need to supply a `selector` for this component.
3. In the HTML, enter some arbitrary text for the time being. Let's get the page working first.
4. Declare the component as part of the module.
5. In *app.routes.ts*, add a `/invite` route to this component.
6. In *app.component.html*, it might be nice to include some hyperlinks to the two pages:

Your page should now be working. Now you can fill in the rest. On your invite page, create a form with input fields for the event, the possible dates en the people to invite to your fun party. When pressing the submit button, simply a `console.log()` of all the form values is enough.

Some hints:

- For selecting the contacts, you'll probably use something like checkboxes that have to be generated depending on the number of contacts. If you're using model-driven forms,

you'll have to use `formArrayName` and `formGroupName`: https://scotch.io/tutorials/how-to-build-nested-model-driven-forms-in-angular-2