Angular JS 2
Router

Created by :
Sangeeta Joshi

Overview

The browser is a familiar model of application navigation:

- Enter a URL in the address bar and the browser navigates to a corresponding page.
- Click links on the page and the browser navigates to a new page.
- Click the browser's back and forward buttons and the browser navigates backward and forward through the history of pages you've seen.

Overview

- The Angular Router ("the router") borrows from this model.
- It can interpret a browser URL as an instruction to navigate to a client-generated view.
- It can pass optional parameters along to the supporting view component that help it decide what specific content to present.

Overview

- You can bind the router to links on a page and it will navigate to the appropriate application view when the user clicks a link.
- You can navigate imperatively when the user clicks a button, selects from a drop box
- And the router logs activity in the browser's history journal so the back and forward buttons work as well.

Component Router: main elements

- • Router - An object that represents the router in the runtime.
- • RouterOutlet—A directive supporting <router-outlet> within your web where the router should render the component.
- • Routes—An array of routes that map URLs to components to be rendered inside the <router-outlet>.
- • RouterLink—A directive supporting the routerLink property in HTML anchor tags.
- • ActivatedRoute—An object that represents the route or routes that are currently active.
- • RouterModule - declares route providers, directives, and functions to pass routes to the root or child modules

A sample routes configuration

- The file app.routing.ts:

```
import { Routes, RouterModule } from '@angular/router';
 import {HomeComponent} from "./home";
 import {ProductDetailComponent} from "./product";

 const routes: Routes = [    {path: '',       component: HomeComponent},    {path: 'product', component: ProductDetailComponent} ];
export const routing = RouterModule.forRoot(routes); // config for the root module
```

If you're configuring routes not for the root module, use RouterModule.forChild(routes)

```
import {Component} from '@angular/core';
@Component({
 selector: 'app',    template: `
  <a [routerLink]="['/']">Home</a>
  <a [routerLink]="['/product']">Product Details</a>

      <router-outlet></router-outlet>    `
 })
 export class AppComponent {}
```

Core Concepts

- <base href>
- add a <base> element to the index.html as the first child in the <head> tag to tell the router how to compose navigation URLs

- If the app folder is the application root, set the href value exactly as:

```
<base href="/">
```

Core Concepts

Router imports:

- The Angular Router is an optional service that presents a particular component view for a given URL.
- It is not part of the Angular core.
- It is in its own library package, @angular/router.
- import { RouterModule, Routes } from '@angular/router';
- src/app/app.module.ts (import)

Core Concepts

Configuration :

- A routed Angular application has one, singleton instance of the Router service.

- When the browser's URL changes, that router looks for a corresponding Route from which it can determine the component to display.

Core Concepts

```
const appRoutes: Routes = [
  { path: 'crisis-center', component: CrisisListComponent },
  { path: 'hero/:id',      component: HeroDetailComponent },
  {
    path: 'heroes',
    component: HeroListComponent,
    data: { title: 'Heroes List' }
  },
  { path: ‘  ',
    redirectTo: '/heroes',
    pathMatch: 'full'
  },
  { path: '**', component: PageNotFoundComponent }
];
```

Core Concepts

```
@NgModule({
  imports: [
    RouterModule.forRoot(appRoutes)
    // other imports here
  ],
  ...
})
export class AppModule { }
```

Core Concepts

- The order of the routes in the configuration matters and this is by design.
- The router uses a first-match wins strategy when matching routes: so more specific routes should be placed above less specific routes.
  - routes with a static path are listed first,
  - followed by an empty path route, that matches the default route.
  - The wildcard route comes last because it matches every URL and should be selected only if no other routes are matched first.

Core Concepts

Router outlet

<router-outlet></router-outlet>

<!-- Routed views go here -->

(when the browser URL is “ /heroes”,

router matches URL to the route path “/heroes”)

&

displays the HeroListComponent

after a ***RouterOutlet*** that is placed in the host view's HTML.

Component Router: main elements

- • Router - An object that represents the router in the runtime.
- • RouterOutlet—A directive supporting <router-outlet> within your web where the router should render the component.
- • Routes—An array of routes that map URLs to components to be rendered inside the <router-outlet>.
- • RouterLink—A directive supporting the routerLink property in HTML anchor tags.
- • ActivatedRoute—An object that represents the route or routes that are currently active.
- • RouterModule - declares route providers, directives, and functions to pass routes to the root or child modules