RouterLink

A basic example of using the RouterLink directive can look like this:

```
<a routerLink="/red-pill/neo">Go!</a>
```

The different url segments can also be passed in an array like this:

```
<a [routerLink]="['/', 'red-pill', 'neo']">Go!</a>
```

```
<a routerLink="/path">
<a [routerLink]="[ '/path', routeParam ]">
<a [routerLink]="[ '/path', { matrixParam: 'value' } ]">
<a [routerLink]="[ '/path' ]" [queryParams]="{ page: 1 }">
<a [routerLink]="[ '/path' ]" fragment="anchor">
```

on site http://localhost:4200/candidate/jobs

```
 <a [routerLink]="['/','dashboard']" >
 <a [routerLink]="['','dashboard']" >
http://localhost:4200/dashboard
a [routerLink]="['dashboard']" >
http://localhost:4200/candidate/dashboard
```

```
[routerLink]="['item.id']"
```

http://localhost:4200/candidate/jobs/item.id

```
 [routerLink]="[item.id]"
```

http://localhost:4200/candidate/jobs/15

There are two methods available on Angular's Router class to navigate imperatively in your component classes: Router.navigate and Router.navigateByUrl. Both methods return a promise that resolve to true if the navigation is successful, null if there's no navigation, false if the navigation fails or is completely rejected if there's an error.
You pass-in an array of url segments to **Router.navigate** or a string to **Router.navigateByUrl**, just like you would using the RouterLink directive.

To use either method, you'll first want to make sure that the Router class is injected into your component class:

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

```
import { Router } from '@angular/router';

@Component({
  // ...
})
export class AppComponent {

  constructor(private router: Router) {}

  // ...
}
```
Copy

Router.navigate

Here's a basic example using the navigate method:

```
goPlaces() {
  this.router.navigate(['/', 'red-pill']);
}
```
Copy
And here's an example demonstrating how navigate is thenable:

```
goPlaces() {
  this.router.navigate(['/', 'red-pill']).then(nav => {
    console.log(nav); // true if navigation is successful
  }, err => {
    console.log(err) // when there's an error
  });
}
```
Copy

Router.navigateByUrl

Router.navigateByUrl is basically the same as Router.navigate, except that a string is passed-in instead of url segments and the navigation should be absolute and start with a /:

```
goPlaces() {
  this.router.navigateByUrl('/red-pill;x=white-rabbit/neo');
```

```
}
```

Using Query Parameters with `Router.navigate`

If you are navigating to the route imperatively using `Router.navigate`, you will pass in query parameters with `queryParams`.

In our example, if we want to route visitors to the products with the list ordered by popularity, it would look like this:

```
goProducts() {
  this.router.navigate(['/products'], { queryParams: { order: 'popular' } });
}
```
Copy

This will result in a URL that looks like this:

```
http://localhost:4200/products?order=popular
```
You can also provide multiple query parameters. In our example, if we want to route visitors to the products with the list ordered by popularity and filtered with an expensive price range, it would look like this:

```
goProducts() {
  this.router.navigate(['/products'], { queryParams: { order: 'popular', 'price-range': 'not-cheap' } });
}
```
Copy

This will result in a URL that looks like this:

```
http://localhost:4200/products?order=popular&price-range=not-cheap
```
Now, you have an understanding of how `queryParams` can be used to set query parameters.

Access

```
 SearchJobs(jobTitle: string) {

   this.router.navigate(['/candidate/jobs'], { queryParams: { jobTitle: jobTitle
} });


 }
```

```
 this.activatedRoute.queryParams
     .subscribe(params => {
       console.log(params);

       console.log(params.jobTitle);
     }
     );
```

Angular 10|9|8 Nested Routing with Multiple RouterOutlet using loadChildren having own Router Modules

```
import { NgModule, Component } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';
import {SignInComponent} from './homeModule/home-components/sign-in/sign-
in.component'

.... and others
import { AuthGuardService as AuthGuard  } from './Services/auth-guard.service';
const routes: Routes = [
  { path: '', component: CarouselComponent },
  { path: 'sign-in/:name', component: SignInComponent },
  { path: 'home',component: CarouselComponent },
  { path: 'register/:name',component: RegisterComponent },
  { path: 'recruiter',canActivate:[AuthGuard],
  loadChildren: () => import(`./recruiterModule/recruiter-module.module`)
```

```
  .then(r => r.RecruiterModuleModule) },
  { path: '**', component: PageNotFoundComponent }
];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

```
import { NgModule } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';

import {ShowJobsComponent} from './recruiter-components/show-jobs/show-
jobs.component'
.... and others

import {AdminGuardService as AdminGuard } from '../Services/admin-guard.service';

const routes: Routes = [
  { path: '', component: RecruiterHomeComponent, children: [
    {path: 'createJobs', component: JobsCreateComponent},
    {path: 'showJobs', component: ShowJobsComponent},
    {path: 'editJobs/:id', component: JobEditComponent},
    { path: 'createCompany', component: CompanyCreateComponent},
    { path: 'companyProfile', component: CompanyProfileComponent},
    { path: 'companyEdit', canActivate: [AdminGuard], component: CompanyEditCompo
nent}
  ]}
];
@NgModule({
  imports: [RouterModule.forChild(routes)],
  exports: [RouterModule]
})
export class RecruiterModuleRoutingModule { }
```

Importing Routing Modules

AppModule

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

import { AppRoutingModule } from './app-routing.module';
```

```
import { AppComponent } from './app.component';
import {HomeModuleModule} from './homeModule/home-module.module'
import {RecruiterModuleModule} from './recruiterModule/recruiter-module.module'
import { HttpClientModule } from '@angular/common/http';


@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    FormsModule,
    HomeModuleModule,
    HttpClientModule,
    RecruiterModuleModule

  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

```
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';
import { FormsModule }   from '@angular/forms';

import { RecruiterModuleRoutingModule } from './recruiter-module-
routing.module';
import {JobsCreateComponent} from './recruiter-components/jobs-create/jobs-
create.component'
.... and others

@NgModule({
  declarations: [
    JobsCreateComponent,
    CompanyCreateComponent,
    RecruiterHomeComponent,
```

```
    FooterComponent,
    HeaderComponent,
    MainContentComponent,
    CompanyProfileComponent,
    CompanyEditComponent,
    JobEditComponent,
    ShowJobsComponent

  ],
  imports: [
    CommonModule,FormsModule,
    RecruiterModuleRoutingModule,SharedModule
  ],
  exports: [
    JobsCreateComponent,
    CompanyCreateComponent,
    RecruiterHomeComponent,
    FooterComponent,
    HeaderComponent,
    MainContentComponent

  ]
})
export class RecruiterModuleModule { }
```

An **export** what you put is the **exports** property of the @NgModule decorator. It enables an **Angular** module to expose some of its components/directives/pipes to the other modules in the applications. Without it, the components/directives/pipes defined in a module could only be used in that module

```
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';
import { FormsModule }   from '@angular/forms';

import { HomeHeaderComponent } from '../homeModule/home-
components/header/header.component'
import { CarouselComponent } from '../homeModule/home-
components/carousel/carousel.component'
.... and others
import { AppRoutingModule } from '../app-routing.module';
import {SharedModule} from '../SharedModule/shared-module.module'
@NgModule({
```

```
  declarations: [
    HomeHeaderComponent,
    CarouselComponent,
    HomeComponent,
    RegisterComponent,
    SignInComponent


  ],
  imports: [
    CommonModule,FormsModule,AppRoutingModule,SharedModule
  ],
  exports: [
    HomeHeaderComponent,
    CarouselComponent,
    HomeComponent,
    RegisterComponent,
    SignInComponent

  ]
})
export class HomeModuleModule { }
```

Sekoj module koj ima router-outlet treba da ima svoja routing configuracija i da se importiraat konfiguraciite vo modulot

```
$ ng g m leaves --routing //generating module and routing
```