1. Introduction
   1. Routing overview
2. Routing Basics
   1. Setting up, Base path, importing routes
   2. Activating routes
   3. Directives:
      1. RouterLink
      2. RouterLinkActive
      3. RouterOutlet
3. Routing Features

|  |
| --- |
| constructor(private router: Router);        }); |
| this.router.navigate(['/login']); |
| <a class="nav-link" [routerLink]="['/products']" routerLinkActive="active" [routerLinkActiveOptions]="{ exact: true }">Product List</a> |

1. Route Parameters
   1. Parameters

|  |
| --- |
| { path: '', redirectTo: 'info', pathMatch: 'full' },  { path: 'info', component: ProductEditInfoComponent },  { path: 'tags', component: ProductEditTagsComponent }, |

* 1. ActivatedRoute Service:
     1. Url Segments
     2. Route parameters
     3. Query Parameters
     4. Resolve data...
  2. Reading Parameter through snapshot approach

const id = +this.route.snapshot.paramMap.get('id');

* 1. Reading Route through observable

    this.route.paramMap.subscribe(

      params => {

        const id = +params.get('id');

        this.getProduct(id);

      }

    )

* 1. Defining optional parameter

Required:

{ path: 'products/:name/:code/:startDate/:endDate',

component: ProductEditInfoComponent },

optional:

 <a class="nav-link" [routerLink]="['/products', {name: productName, code: productCode, startDate: availabiltityStart, endDate: availabilityEnd]"></a>

* 1. Defining query parameter

         <a [routerLink]="['/products', product.id]"

            [queryParams]="{filterBy: listFilter, showImage: showImage}">

                {{ product.productName }}

              </a>

1. Prefetching Data using Route Resolve
   1. Prevents display of a partial page
   2. Reuse code
   3. Improves flow when an error occurred

Reading Resolved Data:

* 1. Snapshot
  2. Observables

|  |
| --- |
| //    \*\*\*\*Snapshot way      const resolvedData: ProductResolved = this.route.snapshot.data['resolvedData'];      this.errorMessage = resolvedData.error;      this.onProductRetrieved(resolvedData.product);      // \*\*\* Through subscirber \*/      this.route.data.subscribe(data => {        const resolvedData: ProductResolved = data['resolvedData'];        this.errorMessage = resolvedData.error;        this.onProductRetrieved(resolvedData.product);      }); |

1. Child Routes:

|  |
| --- |
| children: [            { path: '', redirectTo: 'info', pathMatch: 'full' },            { path: 'info', component: ProductEditInfoComponent },            { path: 'tags', component: ProductEditTagsComponent },          ], |

1. Grouping and Component-less Routing:
   1. Better organization
   2. Share resolvers and guards
   3. Lazy loading
2. Styling, Animating and watching Routes
   1. BrowserAnimationModule
   2. Routing Events
      1. NavigationStart
      2. RoutesRecognized
      3. NavigationEnd
      4. NavigationCancel
      5. NavigationError
         1. EnableTracing: true
3. Secondary Routes:
4. Route Guards
   1. CanActivate
   2. CanActivateChild
   3. CanDeativate
   4. canLoad
5. Lazy Loading
   1. Requirements:
      1. Use a feature module
      2. Routes grouped under a single parent
      3. Not implemented in another module