**Discussion 2.1 – Angular RouterOutlet**

Patrick Cuauro

Bellevue University

WEB 425-307O Angular with TypeScript

Professor Krasso

August 15, 2023

# Angular RouterOutlet

Angular Router is a powerful JavaScript router built and maintained by the Angular core team that can be installed from the @angular/router package. It provides a complete routing library with the possibility to have multiple router outlets, different path matching strategies, easy access to route parameters and route guards to protect components from unauthorized access.

The Angular router is a core part of the Angular platform. It enables developers to build Single Page Applications with multiple views and allow navigation between these views.

## The Router-Outlet

The **Router-Outlet** is a directive that’s available from the router library where the Router inserts the component that gets matched based on the current browser’s URL. You can add multiple outlets in your Angular application which enables you to implement advanced routing scenarios.

|  |
| --- |
| <router-outlet></router-outlet> |

Any component that gets matched by the Router will render it as a sibling of the Router outlet.

## Routers and Paths

Routers are definitions (objects) comprised from at least a path and a component (or a redirectTo path) attribute. The path refers to the part of the URL that determines a unique view that should be displayed, and component refers to the Angular component that needs to be associated with a path. Based on a route definition that we provide (via a static **RouterModule.forRoot**(routes) method), the Router is able to navigate the user to a specific view.

Each **Route** maps a URL **path** to a component.

That path can be empty which denotes the default path of an application and it’s usually the start of the application.

The Path can take a **wildcard** string(**\*\***). The router will select this route if the requested URL doesn’t match any paths for the defined routes. This can be used for displaying a “Not Fount” view or redirecting to a specific view if no patch is found.

This is an example of a route:

|  |
| --- |
| { path: 'contacts', component: ContactListComponent} |

If this route definition is provided to the Router configuration, the router will render ***ContactLisComponent*** when the browser URL for the web application becomes ***/contacts***.

## Routing Matching Strategies

The Angular Router provides different route matching strategies. The default strategy is simply checking if the current browser’s URL is prefixed with the path.

For example our previous route:

|  |
| --- |
| { path: 'contacts', component: ContactListComponent} |

Could be also written as:

|  |
| --- |
| { path: 'contacts', pathMatch: 'prefix', component: ContactListComponent} |

The **patchMatch** attribute specifies the matching strategy. In this case, it’s **prefix** which is the default.

The second matching strategy is **full**. When it’s specified for a route, the router will check if the path is exactly **equal** to the path of the current browser’s URL:

|  |
| --- |
| { path: 'contacts',pathMatch: 'full', component: ContactListComponent} |

Source:

* A complete guide to routing in Angular:

<https://www.smashingmagazine.com/2018/11/a-complete-guide-to-routing-in-angular/>