Angular Routing

In a single-page app, you change what the user sees by showing or hiding portions of the display that correspond to particular components, rather than going out to the server to get a new page. As users perform application tasks, they need to move between the different views that you have defined.

To handle the navigation from one view to the next, you use the Angular `Router`. The `Router` enables navigation by interpreting a browser URL as an instruction to change the view.

To explore a sample app featuring the router's primary features, see the .

Prerequisites

Before creating a route, you should be familiar with the following:

- Basics of components
- Basics of templates
- An Angular app—you can generate a basic Angular application using the Angular CLI.

Learn about Angular routing

Common routing tasks

```
<p>Learn how to implement many of the common tasks associated with Angular routing.</p>
<p class="card-footer">Common routing tasks</p>
```

Single-page applications (SPAs) routing tutorial

```
<p>A tutorial that covers patterns associated with Angular routing.</p>
<p class="card-footer">Routing SPA tutorial</p>
```

Tour of Heroes expanded routing tutorial

```
<p>Add more routing features to the Tour of Heroes tutorial.</p>
<p class="card-footer">Routing Tour of Heroes</p>
```

Creating custom route matches tutorial

```
<p>A tutorial that covers how to use custom matching strategy patterns with Angular routing.
<p class="card-footer">Custom route matches tutorial</p>
```

Router reference

```
<p>Describes some core router API concepts.</p>
<p class="card-footer">Router reference</p>
```