### What is a URL

URL stands for **Uniform Resource Locator** and in its simplest definition it’s a web address that references a web page.

**What is Angular Routing**

Routing allows you to move from one part of the application to another part or one View to another View.

In Angular, Routing is handled by the Angular Router Module.

## Introducing the Angular Routing

In [Angular](https://www.simplilearn.com/tutorials/angular-tutorial/what-is-angular), routing plays a vital role since it is essentially used to create Single Page Applications. These applications are loaded just once, and new content is added dynamically. Applications like Google, Facebook, Twitter, and Gmail are a few examples of SPA. The best advantage of SPA is that they provide an excellent user experience and you don’t have to wait for pages to load, and by extension, this makes the SPA fast and gives a desktop-like feel.

**Angular Router**

The Router is a separate module in Angular. It is in its own library package, [**@angular/router**](https://angular.io/api/router). The Angular Router provides the necessary service providers and directives for navigating through application views.

Using Angular Router you can

* Navigate to a specific view by typing a URL in the address bar
* Pass optional parameters ([query parameters](https://www.tektutorialshub.com/angular/angular-passing-optional-query-parameters-to-route/)) to the View
* Bind the clickable elements to the View and load the view when the user performs application tasks
* Handles back and forward buttons of the browser
* Allows you to load the view dynamically
* Protect the routes from unauthorized users using [Route Guards](https://www.tektutorialshub.com/angular/angular-route-guards/)

### Components of Angular Router

### Router

An **Angular Router** is a service ([Angular Router API](https://angular.io/api/router/Router)) that enables navigation from one component to the next component as users perform application tasks like clicking on menus links, and buttons, or clicking on the back/forward button on the browser. We can access the router object and use its methods like navigate() or navigateByUrl(), to navigate to a route

**Route**

Route tells the Angular Router which view to display when a user clicks a link or pastes a URL into the browser address bar. Every Route consists of a path and a component it is mapped to. The Router object parses and builds the final URL using the Route

Each route can have the following properties:

* path is a string that specifies the path of the route.
* [pathMatch](https://angular.io/api/router/Route#pathMatch) is a string that specifies the matching strategy. It can take prefix (default) or [full](https://angular.io/api/core/Version#full).
* component is a component type that specifies the component that should be mapped to the route.
* [redirectTo](https://angular.io/api/router/Route#redirectTo) is the URL fragment which you will be redirected to if a route is matched.

**Routes**

Routes is an array of Route objects our application supports

**RouterOutlet**

The outerOutlet is a directive (<router-outlet>) that serves as a placeholder, where the Router should display the view

**RouterLink**

The RouterLink is a directive that binds the HTML element to a Route. Clicking on the HTML element, which is bound to a RouterLink, will result in navigation to the Route. The RouterLink may contain parameters to be passed to the route’s component.

**RouterLinkActive**

RouterLinkActive is a directive for adding or removing classes from an HTML element that is bound to a RouterLink. Using this directive, we can toggle CSS classes for active RouterLinks based on the current RouterState

**ActivatedRoute**

The ActivatedRoute is an object that represents the currently activated route associated with the loaded Component.

**RouterState**

The current state of the router includes a tree of the currently activated routes together with convenience methods for traversing the route tree.

**RouteLink Parameters array**

The Parameters or arguments to the Route. It is an array that you can bind to RouterLink directive or pass it as an argument to the Router.navigate method.