

Unit-5 AngularJS SPA

(Single Page Application)



PATH AHEAD....

Services in AngularJS

Introduction to Single Page application (SPA)

Routing in AngularJS

\$routeProvider in AngularJS

HTML DOM directives

Forms Events and Validations

Services in AngularJS

- ▶ AngularJS supports the concept of Separation of Concerns using services architecture.
- ▶ Services are JavaScript functions or objects, which are responsible to perform only specific tasks. This makes them individual entities which are maintainable and testable.
- ▶ The controllers and filters can call them on requirement basis.
- ▶ Services are normally injected using the dependency injection mechanism of AngularJS.
- ▶ AngularJS provides many inbuilt services. It has about 30 built-in services. For example, \$http, \$route, \$window, \$location, etc.
- ▶ Each service is responsible for a specific task such as the \$http is used to make ajax call to get the server data, the \$route is used to define the routing information, and so on.
- ▶ The inbuilt services are always prefixed with \$ symbol.
- ▶ In AngularJS you can make your own service, or use one of the many built-in services.

Single Page application (SPA)

- ▶ A single-page application (SPA) is a web application or website that interacts with the user by dynamically rewriting the current web page with new data from the web server, instead of the default method of a web browser loading entire new pages.
- ▶ The goal is faster transitions that make the website feel more like a native app.
- ▶ In a SPA, a page refresh never occurs instead, all necessary HTML, JavaScript, and CSS code is either retrieved by the browser with a single page load, or the appropriate resources are dynamically loaded and added to the page as necessary, usually in response to user actions.

Routing in AngularJS

- ▶ Routing in AngularJS is used when the user wants to navigate to different pages in an application but still wants it to be a single page application.
- ▶ AngularJS routes enable the user to create different URLs for different content in an application.
- ▶ The **ngRoute** module helps in accessing different pages of an application without reloading the entire application.
- ▶ **\$routeProvider** is used to configure the routes. It helps to define what page to display when a user clicks a link. It accepts either `when()` or `otherwise()` method.
- ▶ The `ngRoute` must be added as a dependency in the application module as shown below.
- ▶ `const app = angular.module("myApp", ["ngRoute"]);`

Steps to download routing service

Visit the URL : <https://angularjs.org/>

Click on Download AngularJS button.

Click on “Browse additional modules”. It will list all modules.

Click on “[angular-route.min.js](#)”.

Copy all the text and save it in your directory with appropriate name with .js extension.

Role of \$routeProvider in AngularJS

- ▶ Routing allows us create Single Page Applications.
- ▶ To do this, we use ng-view and ng-template directives, and \$routeProvider services.
- ▶ We use \$routeProvider to configure the routes.
- ▶ The config() takes a function which takes the \$routeProvider as parameter and the routing configuration goes inside the function.
- ▶ \$routeProvider is a simple API which accepts either when() or otherwise() method.
- ▶ We need to install the ngRoute module first.
- ▶ Routes can be defined during the application's configuration phase using the \$routeProvider service.

Role of \$routeProvider in AngularJS

```
<script>
  var app = angular.module("myApp", ["ngRoute"]);
  app.config(function($routeProvider) {
    $routeProvider
      .when("/", {
        templateUrl: "main.htm"
      })
      .when("/red", {
        templateUrl: "red.htm"
      })
      .when("/green", {
        templateUrl: "green.htm"
      })
      .when("/blue", {
        templateUrl: "blue.htm"
      });
  });
</script>
```


Steps to implement routing using \$routeProvider:

Reference to angular-route.js (view)

- Use `<script src="angular-route.js"></script>`

Add a dependency to the ngRoute module (module)

- `var module = angular.module("sampleApp", ['ngRoute']);`

configure your \$routeProvider (module)

- Use `when(path, route)` and `otherwise()` methods

Links to your route from within your HTML page. (module)

- `Route 1`

Inclusion of the ng-view directive (view)

- `<div ng-view></div>`

HTML DOM directives

Directive	Description
ng-disabled	It disables a given element.
ng-show	It shows a given element.
ng-hide	It hides a given element.
ng-click	It represents an AangularJS click event.

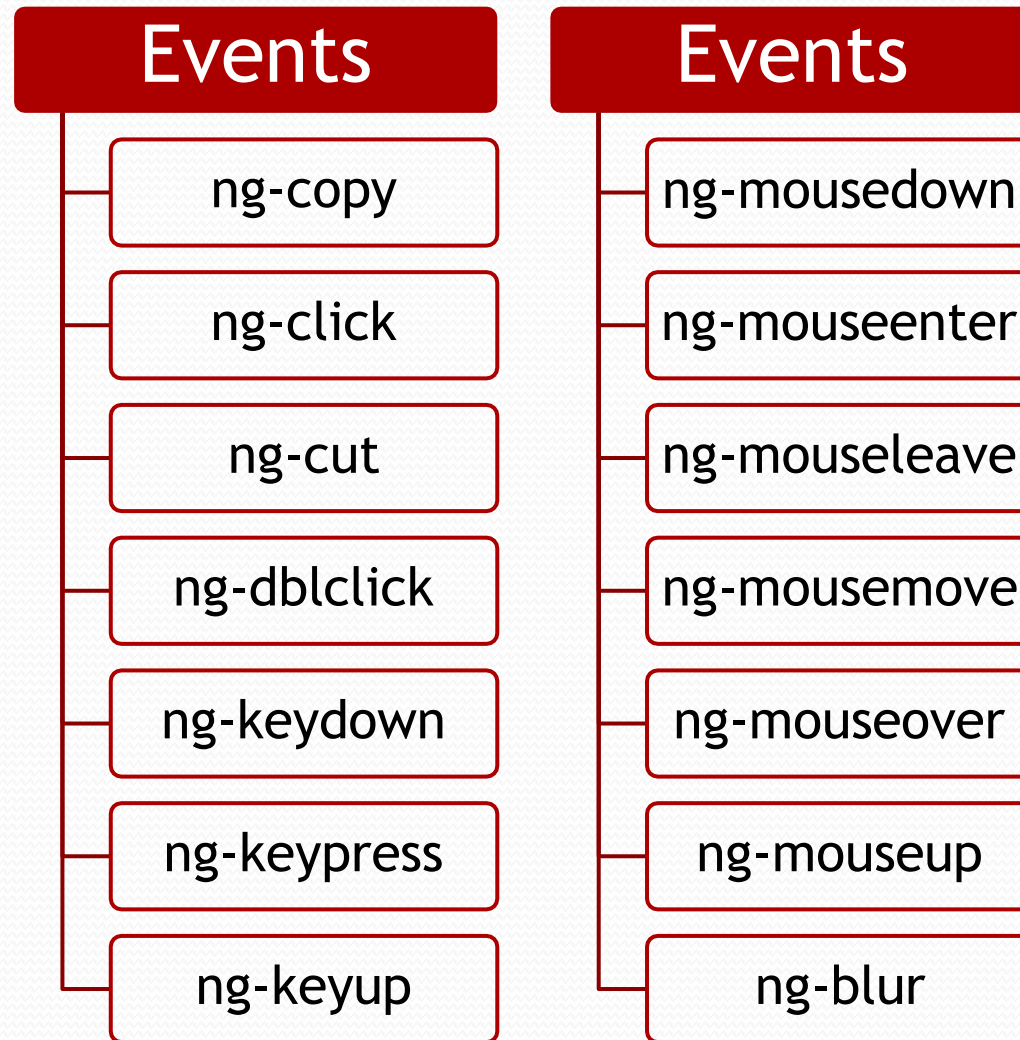
ng-disabled Directive

- ▶ The **ng-disabled Directive** in AngularJS is used to enable or disable HTML elements. If the expression inside the ng-disabled attribute returns true then the form field will be disabled or vice versa. It is usually applied on form field (input, select, button, etc).
- ▶ **Syntax:**
`<element ng-disabled="expression"> Contents... </element>`

Forms Events:

- ▶ AngularJS is incredibly full of events and includes a basic model for how you can add event listeners towards the HTML. It facilitates plenty of events associated with the mouse and keyboard. Most of these events will be put on an HTML Element. If you have written HTML and AngularJS events concurrently, both events can execute, which means an AngularJS event will never overwrite an HTML event.
- ▶ Events in AngularJS can be added using the Directives.

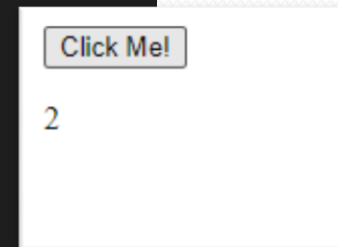
Forms Events:



Forms Events:

```
<!DOCTYPE html>
<html>
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>

<body>
  <div ng-app="myApp" ng-controller="myCtrl">
    <button ng-click="count = count + 1">Click Me!</button>
    <p>{{ count }}</p>
  </div>
  <script>
    var app = angular.module('myApp', []);
    app.controller('myCtrl', function ($scope) {
      $scope.count = 0;
    });
  </script>
</body>
</html>
```



Form Validation:

- ▶ AngularJS offers client-side form validation.
- ▶ AngularJS monitors the state of the form and input fields (input, textarea, select), and lets you notify the user about the current state.
- ▶ AngularJS also holds information about whether they have been touched, or modified, or not.
- ▶ You can use standard HTML5 attributes to validate input, or you can make your own validation functions.
- ▶ Client-side validation cannot alone secure user input. Server side validation is also necessary.
- ▶ AngularJS supports different form states and input field states to provide validations.
- ▶ These states can be used to show meaningful messages to the user.
- ▶ AngularJS also provides validation directives as well as validation CSS classes.

Form Validation:

Form **input fields** have the following **states**. These all are the properties of the input field which can be either true or false.

\$untouched

- It shows that field has not been touched yet.

\$touched

- It shows that field has been touched.

\$pristine

- It represents that the field has not been modified yet.

\$dirty

- It illustrates that the field has been modified.

\$invalid

- It specifies that the field content is not valid.

\$valid

- It specifies that the field content is valid.

Form Validation:

Forms have the following **states**. These all are the properties of the input field which can be either true or false.

\$pristine

- It represents that the fields has not been modified yet.

\$dirty

- It illustrates that one or more fields has been modified.

\$invalid

- It specifies that the form content is not valid.

\$valid

- It specifies that the form content is valid.

\$submitted

- It specifies that the form is submitted.

Form Validation:

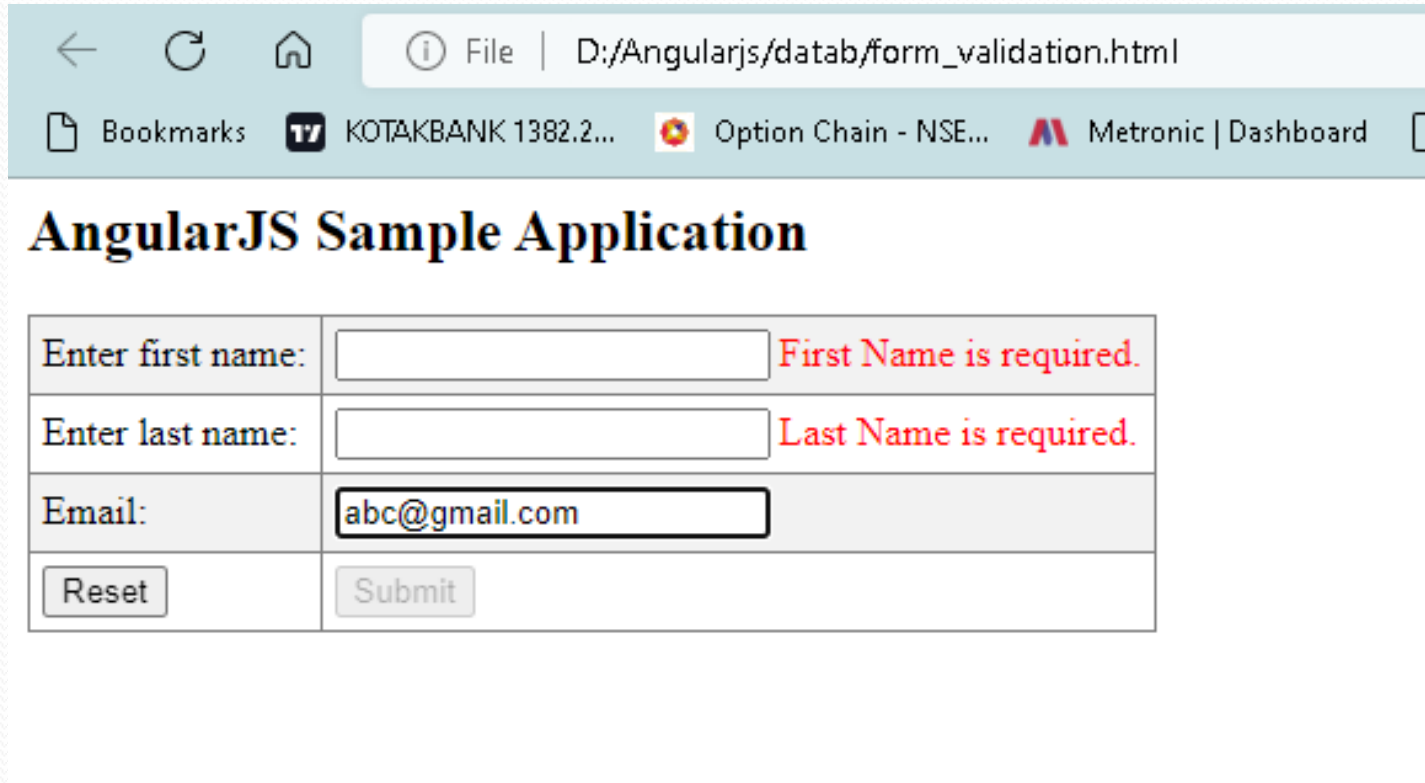
- ▶ AngularJS includes the following **validation directives**

Directive	Description
ng-required	Sets required attribute on an input field.
ng-minlength	Sets minlength attribute on an input field.
ng-maxlength	Sets maxlength attribute on an input field. Setting the attribute to a negative or non-numeric value, allows view values of any length.
ng-pattern	Sets pattern validation error key if the ngModel value does not match the specified RegEx expression.

Form Validation:

CSS Class	Description
ng-valid	Angular will set this CSS class if the input field is valid without errors.
ng-invalid	Angular will set this CSS class if the input does not pass validations.
ng-pristine	Angular will set this CSS class if a user has not interacted with the control yet.
ng-dirty	Angular will set this CSS class if the value of form field has been changed.
ng-touched	Angular will set this CSS class if a user tabbed out from the input control.
ng-untouched	Angular will set this CSS class if a user has not tabbed out from the input control.
ng-submitted	Angular will set this CSS class if the form has been submitted.

Form Validation:



The screenshot shows a web browser window with the address bar displaying "D:/Angularjs/datab/form_validation.html". The browser's bookmark bar includes "Bookmarks", "KOTAKBANK 1382.2...", "Option Chain - NSE...", and "Metronic | Dashboard". The main content area is titled "AngularJS Sample Application" and contains a form with the following fields and validation messages:

Enter first name:	<input type="text"/>	First Name is required.
Enter last name:	<input type="text"/>	Last Name is required.
Email:	<input type="text" value="abc@gmail.com"/>	
<input type="button" value="Reset"/>	<input type="button" value="Submit"/>	