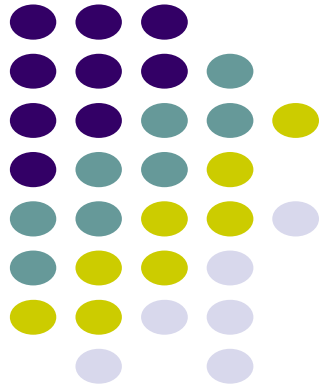
Angular JS

Dr. Arul Xavier V.M



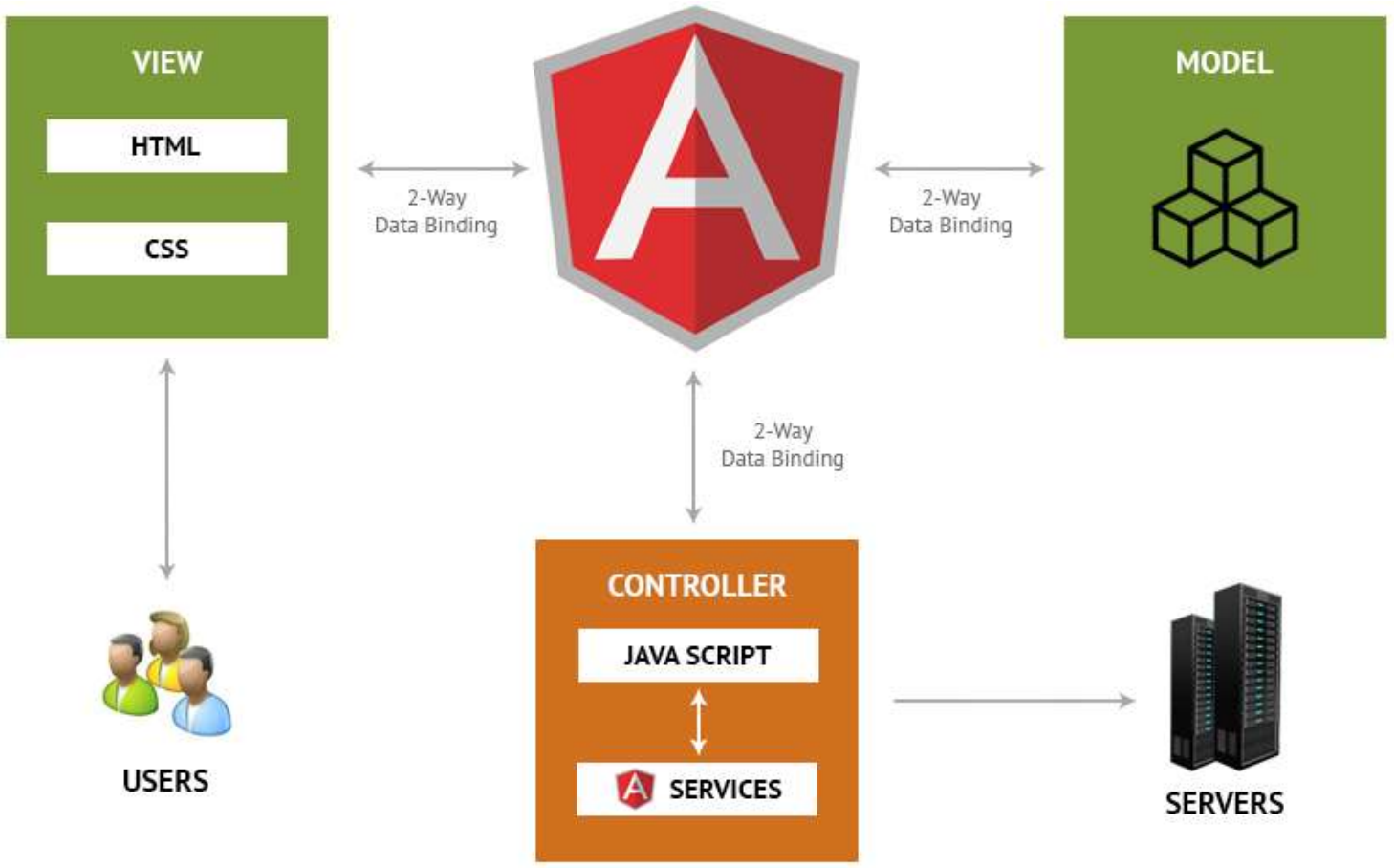
Introducing AngularJS

- AngularJS is a JavaScript Model View Controller(MVC) framework for the Web Application Developments.
- It is based on pure Javascript and HTML.
- AngularJS was created in 2009 by two developers, Misko Hevery and Adam Abrons.

MVC (Model-View-Controller)

- The core concept behind the AngularJS framework is the MVC architectural pattern.
- MVC stands for Model-View-Controller evolved as a way to separate data, logical units and presentation in web application development.
  - The *model* is the data behind the application, usually fetched from the server.
  - The *view* is the UI that the user sees and interacts with. It is dynamic, and generated based on the current model of the application.
  - The *controller* is the business logic and presentation layer, which performs actions such as fetching data, and makes decisions such as how to present the model, which parts of it to display, etc.

Angular JS – Model View Controller



AngularJS Benefits

- AngularJS is a Single Page Application (SPA) Framework.
- An AngularJS application will require fewer lines of code to complete a task than a pure JavaScript.
- AngularJS's declarative nature makes it easier to write and understand applications.
- AngularJS applications can be styled using CSS and HTML independent of their business logic and functionality.
- AngularJS application templates are written in pure HTML

Integration of Angular JS

- It can be added to an HTML page with a <script> tag.
- Angular JS provides set of Directives as HTML attributes.
- Angular JS provides Expressions to bind the data in HTML view page.

Starting Out with AngularJS

- AngularJS Extends HTML
  - AngularJS extends HTML with ng-directives.
  - The ng-app
    - directive used to include AngularJS application in HTML elements.
  - The ng-init
    - directive used to create initial value(model) or calling methods when the HTML content is loaded in the angular JS application.
  - The ng-model
    - directive binds the value of HTML controls (input, select, textarea) to application data.
  - The ng-bind
    - directive binds application data to the HTML view.

Adding Angular JS in HTML

- The angular JS can be included via script tag which just imports the AngularJS library and proves that AngularJS is bootstrapped and working:

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

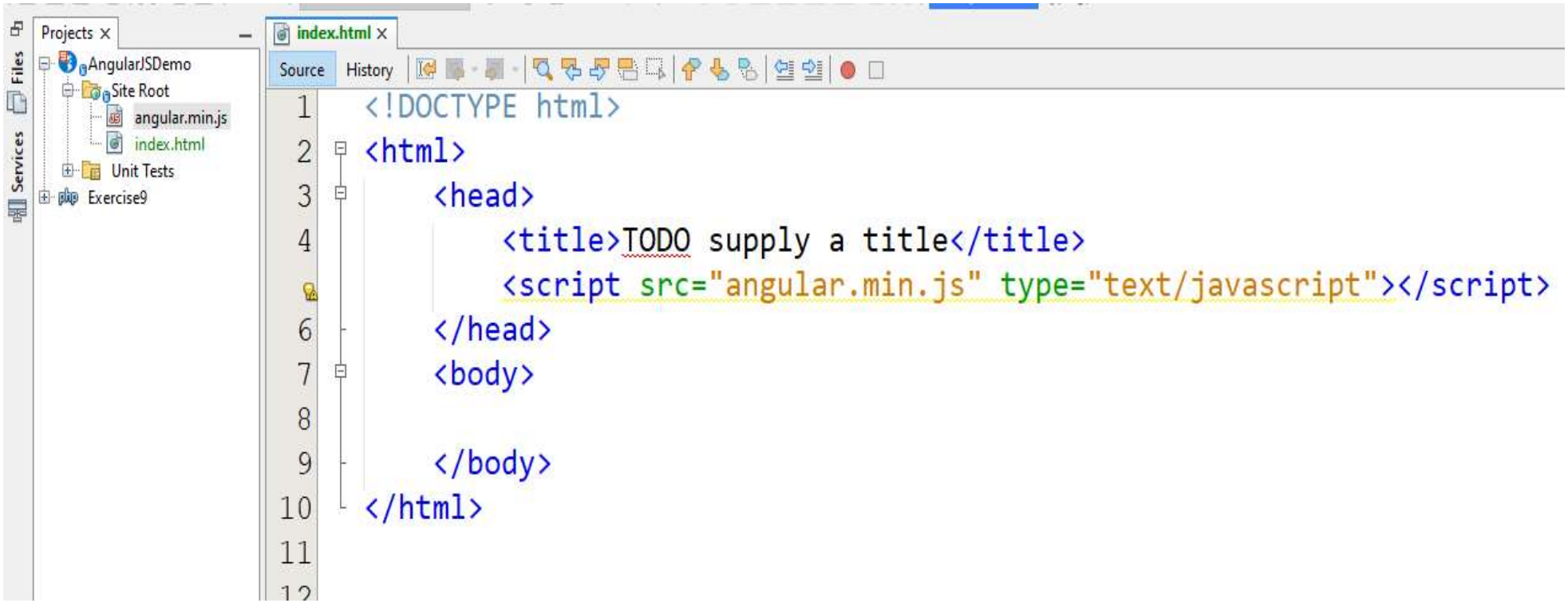
index.html

Source | History

```
1   <!DOCTYPE html>
2   <html>
3       <head>
4           <script src="https://ajax.googleapis.com/ajax/libs
5                   /angularjs/1.6.9/angular.min.js">
6           </script>
        </head>
8       <body>
9
10      </body>
11  </html>
```

Another way to add Angular JS

- Download the angular.min.js file and include using <script> tag.



Getting Started with AngularJS

- This is done through the ng-app directive.
  - This is the first and most important directive that AngularJS has, which denotes the section of HTML that AngularJS controls.
  - Putting it on the **<html>** tag tells AngularJS to control the entire HTML application.
  - We could also put it on the **<body>** or any other element on the page such as ** , <p>** and etc.
  - Any element that is a child of that will be also handled with AngularJS and **anything outside would not be processed**.

Getting Started with AngularJS

ng-app

index.html

Source | History

```
1   <!DOCTYPE html>
2   <html>
3       <head>
4           <script src="https://ajax.googleapis.com/ajax/libs
5                   /angularjs/1.6.9/angular.min.js">
6           </script>
7       </head>
8       <body ng-app="">
9
10      </body>
11  </html>
```

tells AngularJS to control only the body and its child elements

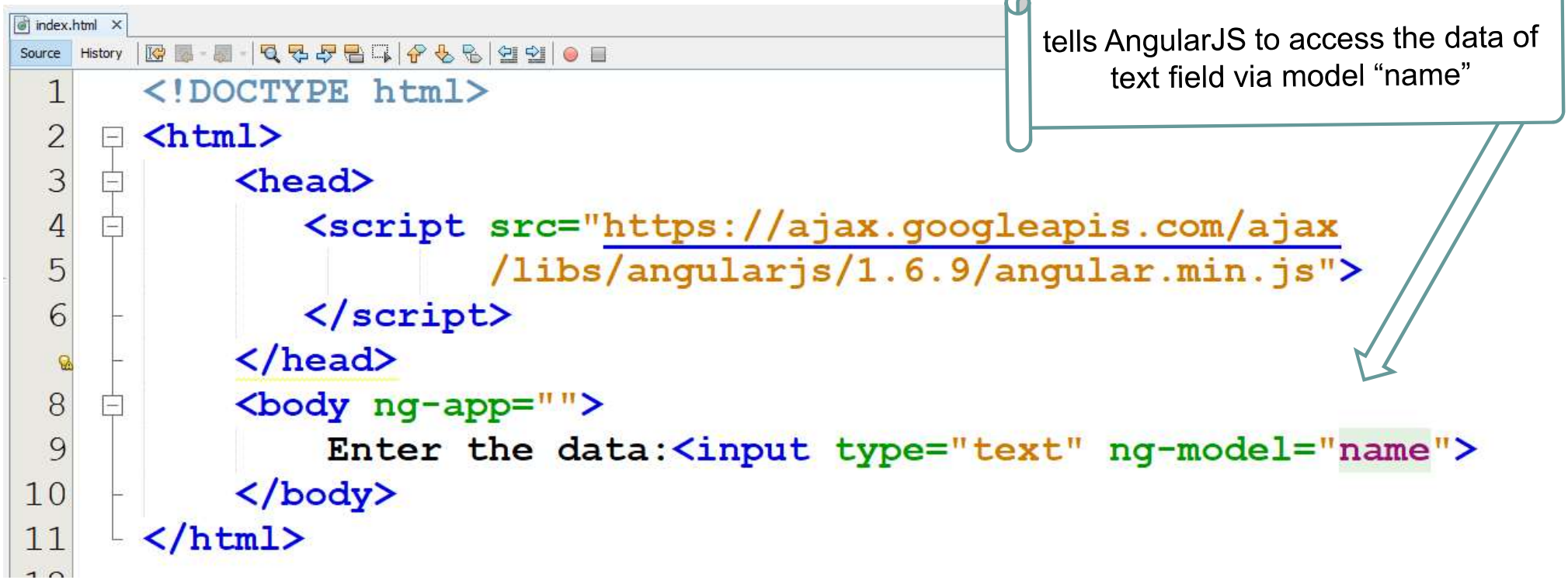
AngularJS ng-init Directive

- We can create initial data model such values, arrays when initiating the application.
- We can also call controller's function using ng-init directive

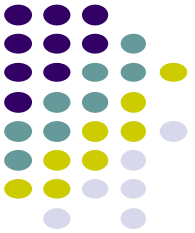
```
<!DOCTYPE html>
<html>
<head>
    <script src="angular.min.js"></script>
</head>
<body ng-app="" ng-init="data=100">
</body>
</html>
```

Creating data model

- The ng-model
  - directive binds the value of HTML controls (input, select, textarea) to application data.



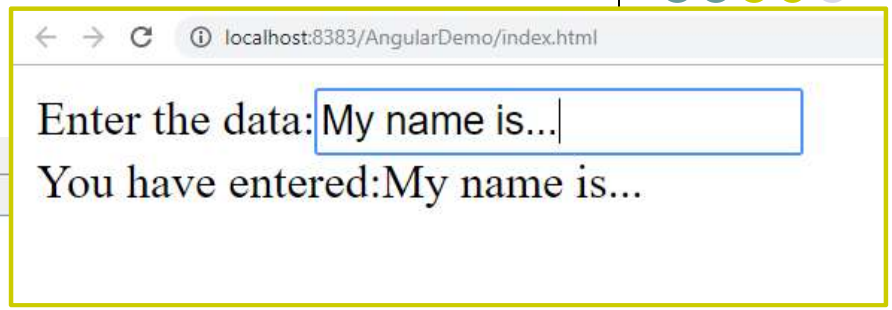
Binding data model to HTML view



- Binding data to HTML view can be done in two ways
  - Using double curly expression
    - {{model_name}}
  - Using ng-bind directive
    -

Binding data model to HTML view

- Using double curly expression



index.html

Source History

```
1   <!DOCTYPE html>
2   <html>
3       <head>
4           <script src="https://ajax.googleapis.com/ajax
5                   /libs/angularjs/1.6.9/angular.min.js">
6           </script>
7       </head>
8       <body ng-app="">
9           Enter the data:<input type="text" ng-model="name">
10          <br>
11          You have entered:{{name}}
12      </body>
13  </html>
14
```

Binding data model to HTML view

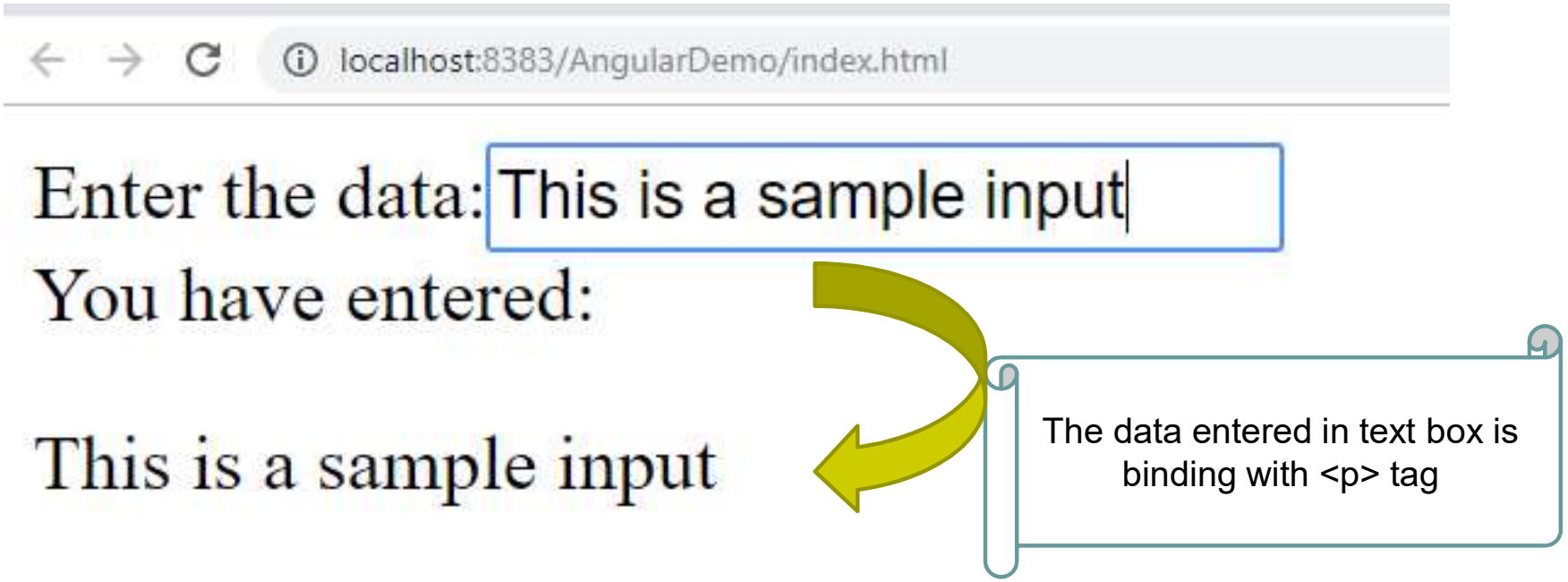
- Using ng-bind directive

index.html

Source History

```
1   <!DOCTYPE html>
2   <html>
3       <head>
4           <script src="https://ajax.googleapis.com/ajax
5                   /libs/angularjs/1.6.9/angular.min.js">
6           </script>
        </head>
8       <body ng-app="">
9           Enter the data:<input type="text" ng-model="name">
10          <br>
11          You have entered: <p ng-bind="name"></p>
12      </body>
13  </html>
```

Binding data model to HTML view - Sample Output



AngularJS Modules

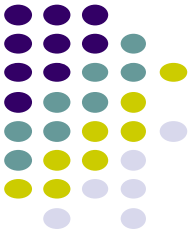
- Modules are AngularJS's way of packaging relevant code under a single name.
- An AngularJS module defines an application.
- The module is a container for the different parts of an application.
- The module is a container for the application Controllers.
  - Controllers always belong to a module.

AngularJS Modules

- In addition to being a container for related JavaScript, the module is also what AngularJS uses to bootstrap an application.
  - What that means is that we can tell AngularJS what module to load as the main entry point for the application by passing the module name to the ng-app directive.
  - The ng-app directive takes an optional argument, which is the name of the module to load during bootstrapping.

Creating an Angular JS module

- A module is created by using the AngularJS function angular.module

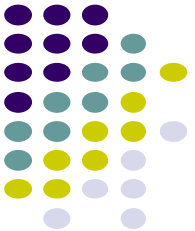


```
<!DOCTYPE html>
<html>
<head>
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
    <script>
        var app = angular.module("myapp",[]);
    </script>
</head>
<body ng-app="myapp">

</body>
</html>
```

It creates the module, first argument “name of the module” and second argument is an array of additional libraries.

Tells angular JS to control <body> contents using the newly created module

Creating First Controller



- Controllers in AngularJS used to create business logic, the JavaScript functions that perform or trigger the majority of our UI-oriented work.

Creating First Controller

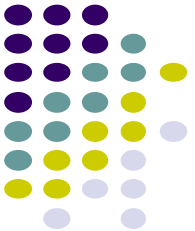
```
<!DOCTYPE html>
<html>
<head>
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <script src="angular.min.js"></script>
    <script>
        var app = angular.module("myapp",[]);
        app.controller('mycontroller',function(){
            //logic goes here
        })
    </script>
</head>
<body ng-app="myapp" ng-controller="mycontroller">

</body>
</html>
```

Creating data in Controller

- $scope object can be used to create data inside controller.
- Later, this data can be binded in HTML view elements.

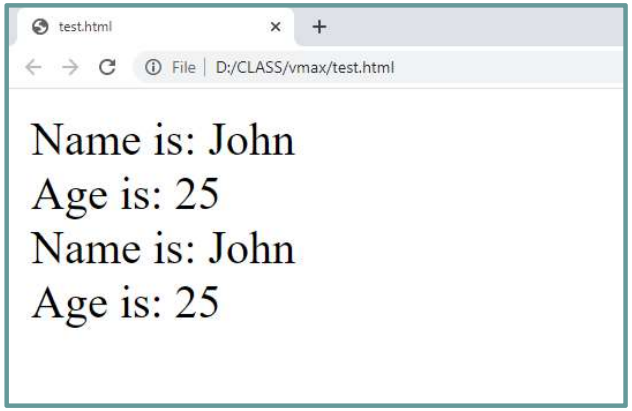
```
<!DOCTYPE html>
<html>
<head>
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <script src="angular.min.js"></script>
    <script>
        var app = angular.module("myapp",[]);
        app.controller('mycontroller',function($scope){
            $scope.name = "John";
            $scope.age = 25;
        })
    </script>
</head>
<body ng-app="myapp" ng-controller="mycontroller">
</body>
</html>
```

Binding data from controller to HTML view



- Once we create a controller variable, you can bind the data using the variable names.

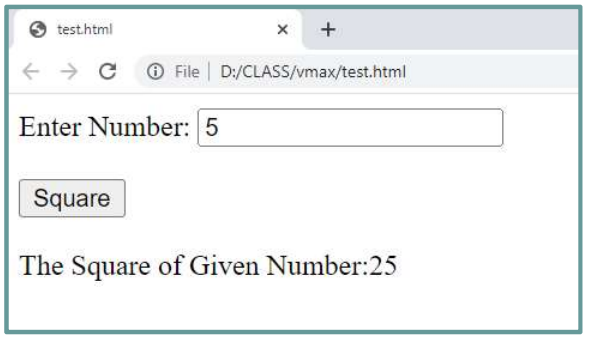
```
<!DOCTYPE html>
<html>
<head>
    <script src="angular.min.js"></script>
    <script>
        var app = angular.module("myapp",[]);
        app.controller('mycontroller',function($scope){
            $scope.name = "John";
            $scope.age = 25;
        })
    </script>
</head>
<body ng-app="myapp" ng-controller="mycontroller">
    <div>Name is: {{name}}</div>
    <div>Age is: {{age}}</div>
    <div>Name is: <span ng-bind="name"></span></div>
    <div>Age is: <span ng-bind="age"></span></div>
</body>
</html>
```



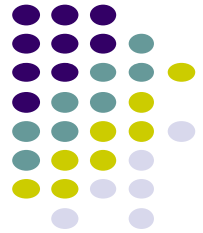
Adding methods inside Controller

- User defined function can be included in controller using $scope object.
- The user defined function can be called using ng-click directive

```
<html>
<head>
    <script src="angular.min.js"></script>
    <script>
        var app = angular.module("myapp",[]);
        app.controller('mycontroller',function($scope){
            $scope.findSquare = function(){
                $scope.result = $scope.data ** 2;
            }
        })
    </script>
</head>
<body ng-app="myapp" ng-controller="mycontroller">
    Enter Number: <input type="text" ng-model="data"><br><br>
    <button ng-click="findSquare()">Square</button><br><br>
    <div>The Square of Given Number:{{result}}</div>
</body>
</html>
```



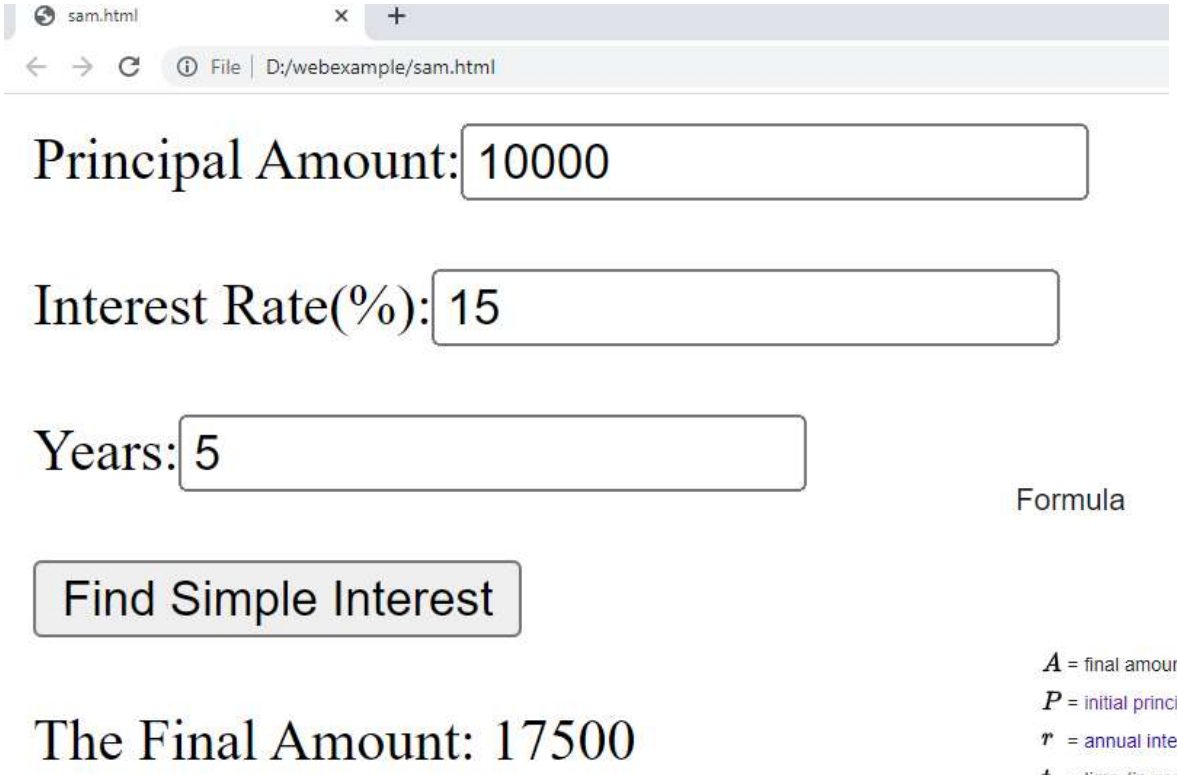
ng-click directive example



- ng-click directive can be used to trigger function from HTML button element.

```
<script>
    var app = angular.module("myapp",[]);
    app.controller("mycontrol",function($scope){
        $scope.findSI = function(){
            $scope.output = $scope.p * (1+($scope.r/100)*$scope.t);
        }
    })
</script>

<body ng-app="myapp" ng-controller="mycontrol">
    Principal Amount:<input type="text" ng-model="p"><br><br>
    Interest Rate(%):<input type="text" ng-model="r"><br><br>
    Years:<input type="text" ng-model="t"><br><br>
    <button ng-click="findSI()">Find Simple Interest</button><br><br>
    The Final Amount: <span ng-bind="output"></span>
</body>
```

Simple Interest Calculator



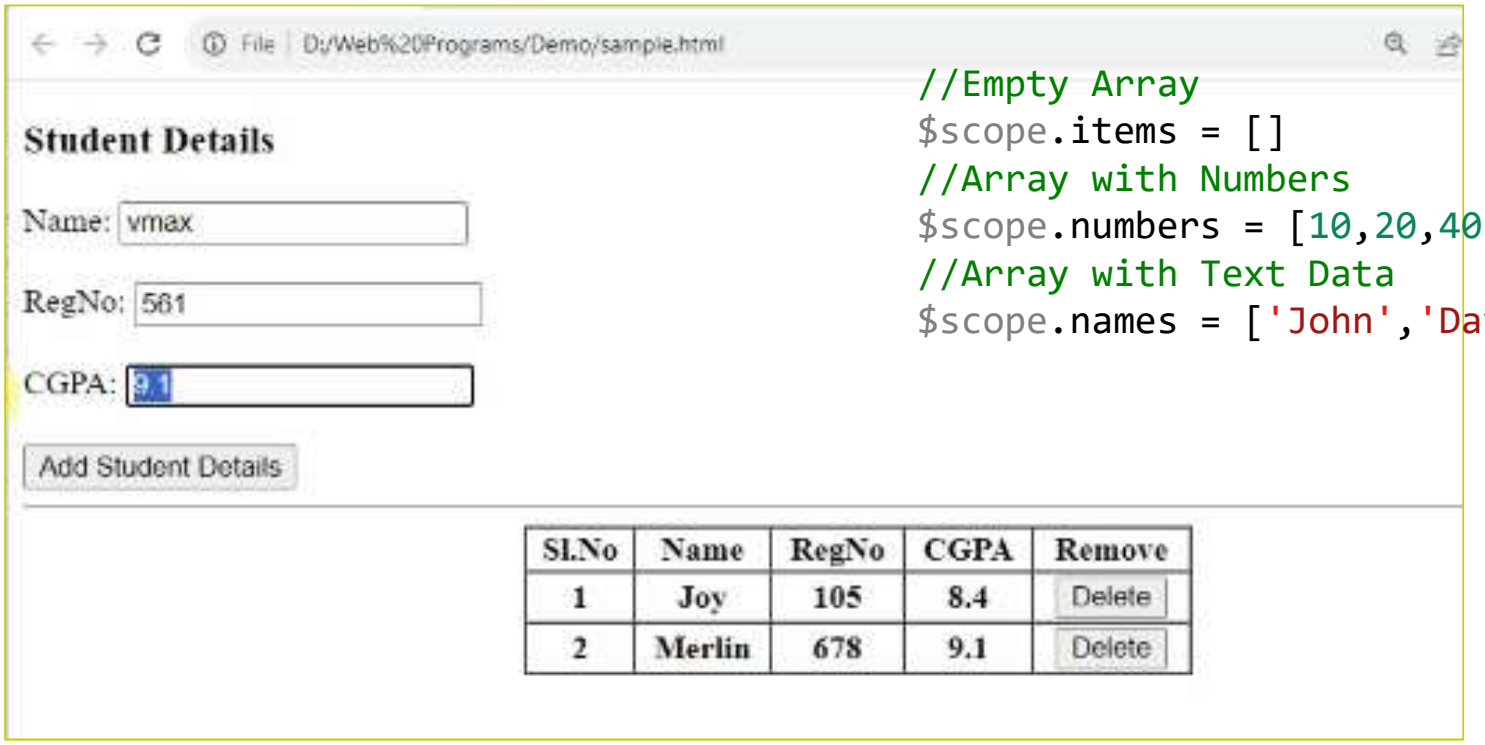
Formula

$$A = P(1 + rt)$$

A = final amount
P = initial principal balance
r = annual interest rate
t = time (in years)

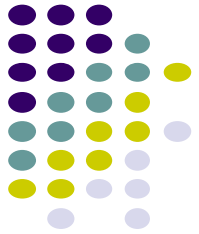
Working with Arrays in Angular JS

Collection of data is represented as arrays in angular JS.
Array elements can be represented using square brackets [ ]




```
//Empty Array
$scope.items = []
//Array with Numbers
$scope.numbers = [10,20,40,59,79]
//Array with Text Data
$scope.names = ['John','David','Arul','Vmax']
```

Creating Arrays in Angular JS

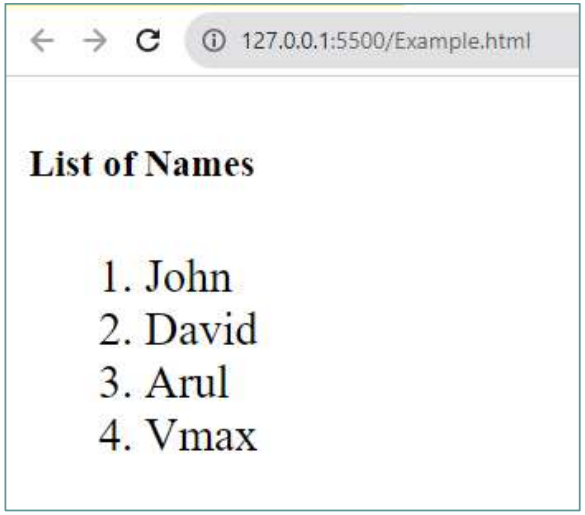


```
<!DOCTYPE html>
<html lang="en">
<head>
    <script src="angular.min.js"></script>
    <script>
        var app = angular.module("MyApp",[])
        app.controller("MyController",function($scope){
            //Empty Array
            $scope.items = []

            //Array with Numbers
            $scope.numbers = [10,20,40,59,79]

            //Array with Text Data
            $scope.names = ['John','David','Arul','Vmax']
        })
    </script>
</head>
<body ng-app="MyApp" ng-controller="MyController"> </body>
</html>
```

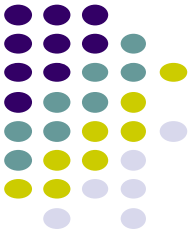
Binding Array Elements to HTML View

- ng-repeat directives used to iterate over an array and display them in the HTML view. ng-repeat="eachVar in arrayVar"

```
<!DOCTYPE html>
<html lang="en">
<head>
    <script src="angular.min.js"></script>
    <script>
        var app = angular.module("MyApp",[])
        app.controller("MyController",function($scope){
            $scope.names = ['John','David','Arul','Vmax']
        })
    </script>
</head>
<body ng-app="MyApp" ng-controller="MyController">
    <h5>List of Names</h5>
    <ol>
        <li ng-repeat="item in names">
            {{item}}
        </li>
    </ol>
</body>
</html>
```



ng-repeat with track by ID



- track by $index can be used to retrieve array elements with $index when array contains duplicates items.

```
<script>
    var app = angular.module("myapp",[]);
    app.controller("mycontrol",function($scope){
       $scope.data = [1,2,2,3,4,4,5];
    })
</script>

<body ng-app="myapp" ng-controller="mycontrol">
    <p ng-repeat="x in data track by $index">
         {{"Data: " + x + " Index:" + $index }}
    </p>
</body>
```

$index in ng-repeat

- The ng-repeat directive also exposes some helper variables which allows us to gain some insight into the current element.
  - $index -> gives us the index or position of the item in the array.

```
<script>
    var app = angular.module("MyApp",[])
    app.controller("MyController",function($scope){
        $scope.names = ['John','David','Arul','Vmax']
    })
</script>

<ol>
     <li ng-repeat="item in names">
         Index:{{$index}}
         Data:{{item}}
     </li>
 </ol>
```