Angular JS Tutorial for Beginners

Introduction

Sick of confusing documentation and an intimidating AngularJS learning curve? Most of the concepts such as two-way data binding and dependency injection are difficult for many fresher's to comprehend. This tutorial simplifies complex concepts into easy steps and removes the frustration of boilerplate code and repeated debugging. Begin your path to developing dynamic web applications! Click to view our complete **AngularJS course syllabus** and become a skilled developer from a beginner!

Why Students or Freshers Learn Angular JS?

Students or freshers prefer to learn AngularJS to kick-start their front-end web development career due to its:

- **Industry Relevance:** Large corporations continue to support and maintain legacy projects developed using AngularJS.

- **Organized Development:** It introduces the Model-View-Controller (MVC) architectural paradigm, and through it, instills clean, structured coding from the beginning.
- **Time-Saving Concepts:** Two-Way Data Binding and Dependency Injection automatically take care of boilerplate code, increasing productivity.
- **Focus on Testing:** The model is meant for unit testing, an important ability in developing professional, trustworthy applications.
- **Reusable Components:** Learning directives enables the development of reusable HTML components to support effective development of Single Page Applications (SPAs).
- **Career Opportunities:** Being skilled in a well-organized framework such as AngularJS increases employability for different front-end developer positions.

Crack your next interview! Download our guide with the latest **AngularJS interview questions and answers** now!

Step-by-Step Angular JS Tutorial for Beginners

AngularJS is a robust JavaScript framework by Google for building dynamic single-page apps (SPAs). It adds vocabulary to HTML so that you can use HTML as your template language. The major advantage of AngularJS lies in its application of the Model-View-Controller (MVC) pattern and Two-Way Data Binding, which automatically minimizes the boilerplate code needed to integrate data between the Model (JavaScript data) and View (HTML elements).

Installation and Setup

AngularJS has no complicated installation procedures like contemporary frameworks (Angular 2+ or React) since you do not usually use Node.js or a command-line interface to install a version 1 project. You just have to include the JavaScript library file in your HTML.

Step 1: Create Project Structure

Create a new directory for your project and include the following two files:

File Name	Purpose
index.html	The main application view and entry point.
app.js	The JavaScript file for your AngularJS application logic (module, controller).

Step 2: Include the AngularJS Library

You may include AngularJS with a Content Delivery Network (CDN), which is the easiest way for beginners.

Add the following code in your index.html file:

```
<!DOCTYPE html>

<html lang=”en”>

<head>

    <meta charset=”UTF-8">

    <title>AngularJS Beginner Tutorial</title>

</head>

<body>
```

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

    <script src=”app.js”></script>

</body>

</html>
```

Core Concepts: Directives, Modules, and Controllers

AngularJS operates by introducing special attributes, referred to as directives, to HTML elements.

Step 3: Define the Application (ng-app)

The ng-app directive is fundamental; it establishes the root element of an AngularJS application and instructs AngularJS to assume control of that part of the HTML.

Adjust your file to incorporate the directive, referencing it to a name you will utilize within your JavaScript file:

```
<body ng-app=”myApp”>

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

    <script src=”app.js”></script>

</body>
```

Step 4: Build the Module (The Application Container)

The Module is where your application components (controllers, services, filters) get registered.

Define the module named myApp in your app.js file: (matching to the ng-app value)

```
// app.js

// 1. Create the AngularJS module named 'myApp'

// The empty array [] is for listing module dependencies (we have none for now)

var app = angular.module('myApp', []);
```

Step 5: Build the Controller (Application Logic)

The Controller is a JavaScript function that contains the application logic and data. It manages the data flow to the view (HTML).

Define a controller named MyController in your app.js file:

```
// app.js

var app = angular.module('myApp', []);

// 2. Register a Controller to the module

// $scope is a core object passed to the controller

// It is the data model that binds the controller to the view

app.controller('MyController', function($scope) {

    // Define a property on the $scope object

    $scope.greeting = "Hello from the AngularJS Controller!";

    $scope.user = {
```

```
        name: “Alice”,

        role: “Frontend Developer”

    };

    // Define a function on the $scope object

    $scope.updateRole = function() {

        $scope.user.role = “Senior Developer”;

    };

});
```

Linking Controller and View

Step 6: Connect the Controller with the View (ng-controller)

Use the ng-controller directive on your index.html to associate the HTML section with the MyController that you’ve created.

```
<body ng-app=”myApp”>

    <div ng-controller=”MyController”>

        <h2>{{ greeting }}</h2>

    </div>

    </body>
```

When you browse index.html, you should see: “Hello from the AngularJS Controller!”

Step 7: Display Data with Expressions

AngularJS utilizes expressions, wrapped in double curly braces {{ }}, to present data from $scope to the HTML.

Modify the div on your index.html:

* *

<h2>Data Display with Expressions</h2>

* Message: **{{ greeting }}** *

* User Name: **{{ user.name }}** *

* User Role: **{{ user.role }}** *

* *

Two-Way Data Binding (ng-model)

Two-Way Data Binding is probably AngularJS's strongest feature. The ng-model directive connects the input element's value directly to a property on the $scope. Any alteration in the input field will update the $scope property in real time, and any alteration to the $scope property will update the input field and all connected expressions in real time.

Step 8: Implement Two-Way Binding

Add an input element to your index.html and scope it to a new $scope property named nameInput.

In index.html:

* *

<h3>Two-Way Data Binding (ng-model)</h3>

<label for="name">Enter a name:</label>

<input type="text" ng-model="nameInput" id="name" placeholder="Type here…">

* Hello, **{{ nameInput }}**! *

* *

Event Handling and Conditional Logic

Step 9: Handle Events (ng-click)

* *

<h3>Event Handling (ng-click)</h3>

* Current Role: **{{ user.role }}** *

<button ng-click="updateRole()">Promote User</button>

* *

Clicking the "Promote User" button will call the function in MyController, updating the displayed role to "Senior Developer."

Step 10: Conditional Display (ng-show / ng-if)

You can conditionally show or hide HTML elements with directives such as ng-show and ng-if.

- **ng-show/ng-hide:** Toggles the CSS display: none; attribute. The element is still in the DOM.

- **ng-if:** Fully removes or adds the element to/from the DOM (better for advanced structures).

Add a button and toggle a message with a boolean property on the $scope.

In app.js, set the property to initialize it:

```
app.controller('MyController', function($scope) {
    // … other properties …
    $scope.showSecret = false;
    $scope.toggleSecret = function() {
        $scope.showSecret = !$scope.showSecret; // Inverts the value
    };
});
```

In index.html:

```
<div ng-controller="MyController">
    <h3>Conditional Logic (ng-show)</h3>
    <button ng-click="toggleSecret()">
        {{ showSecret ? 'Hide' : 'Show' }} Secret Message
    </button>
    <p ng-show="showSecret">**Shh! This is the secret AngularJS message!**</p>
```

```
</div>
```

Looping with ng-repeat

The ng-repeat directive is employed for iterating over collections (arrays) and providing respective HTML elements for each item. This is important for showing lists of data.

Step 11: Establish Data for Repetition

Add an array of objects to your $scope in app.js:

```
app.controller('MyController', function($scope) {
    // … existing code …
    $scope.tasks = [
        { name: "Setup AngularJS", completed: true },
        { name: "Create Controller", completed: true },
        { name: "Implement ng-model", completed: false },
        { name: "Complete Tutorial", completed: false }
    ];
});
```

Step 12: Loop and Show Data

Employ an unordered list and use ng-repeat on the list item in your index.html.

```
<div ng-controller="MyController">
    <h3>Looping (ng-repeat)</h3>
```

```
    <h4>To-Do List:</h4>

    <ul>

        <li ng-repeat="task in tasks">

            **{{ $index + 1 }}.** {{ task.name }}

            <span ng-if="task.completed" style="color: green;">(Done)</span>

            <span ng-if="!task.completed" style="color: red;">(Pending)</span>

        </li>

    </ul>

</div>
```

The $index variable is a special attribute offered by ng-repeat that returns the current item's index in the array.

Filtering and Sorting

AngularJS filters shape data for presentation and can be applied directly in expressions with the pipe symbol (|).

Step 13: Search Filtering

You can filter the list of tasks by including a search box bound by ng-model.

In index.html, include an input field and update the ng-repeat expression:

```
<div ng-controller="MyController">

    <h3>Filtering Data</h3>
```

```
    <label for="search">Search Tasks:</label>

    <input type="text" ng-model="search.name" id="search" placeholder="Filter by name…">

    <ul>

        <li ng-repeat="task in tasks | filter: search">

            {{ task.name }}

            <span ng-if="task.completed" style="color: green;">(Done)</span>

        </li>

    </ul>

    <p>Showing **{{ (tasks | filter: search).length }}** of **{{ tasks.length }}** tasks.</p>

</div>
```

Now, when you enter a word in the search box, the list will automatically narrow down to display only those tasks whose name includes the entered word.

Step 14: Sorting Data

The orderBy filter enables you to sort the members of a collection according to a given property.

Update the ng-repeat yet again to sort the filtered list by name:

```
<div ng-controller="MyController">

    <h3>Sorting Data</h3>
```

```
    <p>Current Sort: **Name** | **Reversed: {{ reverseSort ? 'Yes' : 'No' }}**</p>
    <button ng-click="reverseSort = !reverseSort">Toggle Sort Direction</button>
    <ul>
        <li ng-repeat="task in tasks | filter: search | orderBy: 'name': reverseSort">
        </li>
    </ul>
</div>
<script>
    // In app.js (or inline for simplicity)
    // Add this property to your MyController's $scope:
    $scope.reverseSort = false;
</script>
```

Summary of Core Angular JS Directives and Concepts

Concept	Directive/Syntax	Purpose
Application Root	ng-app="myModule"	Bootstraps and initializes the AngularJS application.
Controller Logic	ng-controller="MyController"	Attaches a JavaScript controller function

		(logic/data) to an HTML element.
Data Display	{{ expression }}	Displays the value of a scope variable or expression in the view.
Two-Way Binding	ng-model=”myVariable”	Synchronizes data between the view (input element) and the model ($scope).
Event Handling	ng-click=”myFunction()”	Calls a function defined on the $scope when the element is clicked.
Conditional Display	ng-show=”boolean” / ng-if=”boolean”	Conditionally shows/hides or adds/removes an element from the DOM.
Looping/Iteration	ng-repeat=”item in array”	Renders an element once for each item in a collection.
Data Formatting	`{{ data	filterName }}`

Learn further with our **Angular JS Challenges and Solutions**.

Real Time Examples for Angular JS Tutorial for Learners

Here are some real-time examples for AngularJS tutorial for beginners:

Real-Time Form Display and Validation (Two-Way Data Binding)

- **Objective:** Display user input and validation messages in real-time while they are typing.
- **Core Concepts:** ng-model, {{ expression }}, and ng-show.
- **Mechanism:** An <input> field uses ng-model to bind to a $scope variable (e.g., user.email). An {{ expression }} elsewhere on the page displays the email instantly (real-time display). A <p> element with ng-show checks a validation flag (e.g., myForm.email.$invalid) and shows an error message instantly when the input is invalid. This demonstrates how the View and Model stay synchronized.

Dynamic Search Filtering (List Management)

- **Objective:** Filter a list of items instantly based on what the user inputs in a search field.
- **Core Concepts:** ng-repeat and the filter filter.
- **Mechanism:** A list of items (e.g., products, contacts) is rendered using ng-repeat=”item in items | filter: searchText”. An <input> field is bound to the searchText model using ng-model. As the user types, the searchText model updates, and the filter automatically processes the entire list, redrawing the view in real-time without requiring any manual DOM manipulation or JavaScript looping logic.

Live Button State and Text Toggle (Event and Conditional Logic)

- **Objective:** Update a button’s text and appearance right after an action, presenting the new status.
- **Core Concepts:** ng-click, ng-bind (or {{ }}), and ng-disabled.
- **Mechanism:** A button uses ng-click=”toggleStatus()” to flip a boolean on the $scope (e.g., isSubscribed). The button’s text is dynamically set using ng-bind or {{ isSubscribed ? ‘Unsubscribe’ : ‘Subscribe’ }}. Additionally, the button is disabled after clicking using ng-disabled=”isSubmitting”, and a separate message

is shown using ng-show=”isSubmitting”. This provides immediate feedback to the user on application state changes.

Explore more **angular js project ideas** in this link.

FAQs About Angular JS Tutorial for Beginners

1. Is Angular JS easy to learn?

AngularJS (Angular 1) is moderately difficult to learn once you have good JavaScript basics. Its ideas such as Two-Way Data Binding and directives make initial work easier, but its scope and lifecycle can prove difficult for beginners who have no experience.

2. How to use Angular JS in HTML?

You apply it by including the AngularJS library through a <script> tag, followed by the ng-app directive (to init the app) and ng-controller directive to the HTML element. Data is rendered through expressions such as {{ data }}.

3. Can I learn Angular in 3 days?

No, acquiring Angular (the new framework, not AngularJS) in 3 days is extremely unrealistic. It normally takes 2-3 months of serious effort to get the main concepts such as TypeScript, components, modules, RxJS, and dependency injection.

4. Is Netflix built with Angular?

No, Netflix is largely developed using React for the user interface. Although the new Angular framework is solid, Netflix opted for React due to its component-based approach and ecosystem for development.

5. Is Angular JS in demand?

Yes, the new Angular (version 2+) is extremely in demand, particularly for enterprise-level large applications and high-level dashboards.

6. Is Angular still used in 2025?

Yes, current Angular (2+) remains strongly applicable and in demand in 2025, especially among large corporations and firms that appreciate its rigid, opinionated architecture, scalability, and native TypeScript support.

7. Is Angular front or backend?

Angular is a front-end framework. It is used solely for creating the user interface (UI) and client-side logic that executes in the browser of the user, interacting with a standalone backend API for data.

8. Does Amazon use React or Angular?

Both React and Angular are utilized by Amazon in various areas of its massive ecosystem, as well as Vue.js. Its technology stack is varied, but React is reportedly used extensively in much of its front-end web applications.

9. Which big company uses Angular?

Many large organizations employ Angular, including most famously Google (for apps such as Google Cloud and the Google Marketing Platform), Microsoft (for certain aspects of Xbox), PayPal, and Deutsche Bank.

10. What is the salary of Angular developer in TCS?

The **Angular JS Developer Salary in India** especially in companies like TCS (Tata Consultancy Services) depends dramatically by experience (YOE), but a general range for a mid-grade developer is generally ₹4,00,000 to more than ₹10,00,000 or more per year.

Conclusion

You have now stepped into the world of AngularJS successfully. Having learned the basic concepts, initializing using ng-app, logic using ng-controller, display using, and two-way binding using ng-model through this Angular JS tutorial for beginners, you know how to build dynamic data-driven front-end applications in a reduced amount of code.To remain up to date and secure, take our **Angular JS Course in Chennai** today and develop enterprise-class applications with TypeScript and the newest tools!