Introduction to Angular

Angular is one of the most popular JavaScript frameworks for building client-side applications. It provides a lot of reusable code like predefined methods, classes, interface etc. which we can use to create dynamic client-side applications.

- Angular is a popular open-source web application framework developed by Google.
- It is designed for building dynamic, single-page applications (SPAs) and is based on TypeScript, a superset of JavaScript.
- Angular provides a comprehensive set of tools and features that simplify the development process and enhance the performance of web applications.

What is Angular?

Definition: Angular is a JavaScript framework which allows us to create single page applications.

- Angular is used for building client-side applications using HTML, CSS and a programming language like JavaScript or TypeScript.
- Angular is not a programming language in itself like JavaScript. Instead, it is a framework which uses a programming language like JavaScript or TypeScript.
- We use Angular for creating single page applications.

What is a Single Page Application?

A Single Page Application (SPA) is a web application that functions within a single web page, providing a smooth and interactive user experience similar to that of a desktop application.

- **Single Page**: SPAs operate within a single HTML page, which is initially loaded from the server.
- **Dynamic Content**: Instead of navigating to different pages, SPAs update the content dynamically without requiring a full page reload.
- **Smooth User Experience**: SPAs use client-side rendering techniques to enhance performance, responsiveness, and interactivity.

- **URL-Based Navigation**: SPAs use the browser's history API to manipulate the URL, enabling bookmarking, back/forward navigation, and direct access to specific application states.
- **RESTful APIs**: SPAs often communicate with back-end servers via RESTful APIs to retrieve and update data asynchronously.

Advantage of single page application

- **Improved Performance**: SPAs load the necessary resources upfront and retrieve data asynchronously, reducing page reloads and providing a faster user experience.
- **Enhanced Responsiveness**: SPAs enable smooth interactions and provide a more fluid user interface by updating specific sections of the page without full refreshes.
- **Native-Like Experience**: SPAs mimic the behavior of native applications, providing users with a familiar and intuitive interface.
- **Code Reusability**: SPAs separate the front-end and back-end logic, allowing for code reuse and easier maintenance.
- **Offline Support**: SPAs can leverage local storage and caching techniques to provide limited offline functionality.
- **Scalability**: SPAs can handle a large number of users while minimizing server load due to client-side rendering and API-driven communication.

Disadvantage of single page application

- **Initial Load Time**: SPAs require loading all necessary resources upfront, resulting in a potentially longer initial load time compared to traditional web applications.
- **SEO Challenges**: Since most content is loaded dynamically, search engine optimization (SEO) can be more complex, requiring server-side rendering or additional techniques.
- **Increased JavaScript Complexity**: SPAs heavily rely on JavaScript frameworks, which may require additional learning and development expertise.
- **Browser Compatibility**: SPAs depend on modern browser capabilities and may not work optimally on older browsers without adequate polyfills.

Why use Angular?

1. **Comprehensive Framework**: Angular is a full-fledged, feature-rich framework that offers a comprehensive solution for building complex web applications. It provides a complete set of tools and features, including templating, data binding, routing, dependency injection, and more. This makes it suitable for large-scale applications where extensive functionality and organization are required.
2. **TypeScript Integration**: Angular is built with TypeScript, a statically-typed superset of JavaScript. TypeScript brings advantages such as enhanced code maintainability, better tooling support, improved error detection during development, and increased productivity for large teams working on the same codebase. The strong typing in Angular reduces the likelihood of runtime errors, leading to more robust applications.
3. **Two-Way Data Binding**: Angular offers powerful two-way data binding, allowing automatic synchronization between the model and the view. This simplifies the process of managing and updating data, making it easier to build dynamic and responsive user interfaces. With two-way data binding, developers can write less code and achieve faster development cycles.
4. **Component-Based Architecture**: Angular follows a component-based architecture, where the application is divided into reusable components. Each component encapsulates its own logic, styles, and templates, promoting code reusability and modularity. This approach enables better code organization, easier maintenance, and facilitates collaboration among team members.
5. **Robustness and Scalability**: Angular provides a solid foundation for building large-scale applications. It includes features like dependency injection, modular architecture, and built-in support for unit testing. These features contribute to code maintainability, testability, and scalability. Angular's modularity allows developers to split the application into smaller, manageable modules, resulting in better performance and reusability.
6. **Angular CLI and Ecosystem**: Angular offers a powerful command-line interface (CLI) tool that automates various development tasks, such as project setup, code generation, testing, and deployment. The Angular ecosystem is rich with a wide range of libraries, tools, and community support, which enhances developer productivity and makes it easier to find solutions to common challenges.

Versions of Angular

The first version of Angular was released in 2010 and it was called AngularJS.

- Angular JS was created as a JavaScript framework for building client applications.
- Soon it gained popularity and it was by far the most popular JavaScript framework available for creating web applications back then.
- But this framework was not designed with the need of today's application in mind. Plus, it was overly complex.
- So, the Angular JS team had to simplify this framework and get rid of all the issues which it had. But it was not a simple task.

In the year 2016, Angular team decided to re-write the original framework from scratch using TypeScript as the programming language. And instead of calling it Angular JS, they simply named it **Angular**.

- Angular is written using a TypeScript programming language.
- Angular was not a simple upgrade to its predecessor, instead it was completely re-written from scratch.
- Angular also fixed all the bugs and issues which AngularJS had.

NOTE: AngularJS & Angular are two completely different frameworks.

- Since the initial release of Angular 2, there have been other versions of Angular which have been released over the years.
- After the release of Angular 2, the Angular team simply adheres to a versioning scheme where a new version of Angular is released every six months.
- That new versions are, however, not the complete rewrite, instead they have some minor changes or few new features.