

GETTING ACQUAINTED WITH MAGORA'S TECH STACK:

ANGULAR IN THE SPOTLIGHT



Angular is a platform for developing mobile and desktop web applications. It came about as a complete rethink of AngularJS, and it was worth it – the framework became much clearer and more flexible and the technology itself is a great step ahead for projects in need of high scalability.

Today I'll tell you about the strengths and weaknesses of Angular, explaining the developer's work in layman's terms.

What is Angular in Essence?

[Angular](#) is a platform that simplifies the assembly of applications on the web. Angular combines the best practices for solving development problems with simple implementation. It offers a highly user-friendly [command-line interface \(CLI\)](#) and console, a convenient graphical interface. Such software solutions are available via browsers on mobile devices and desktops.



Angular can Make Decisions

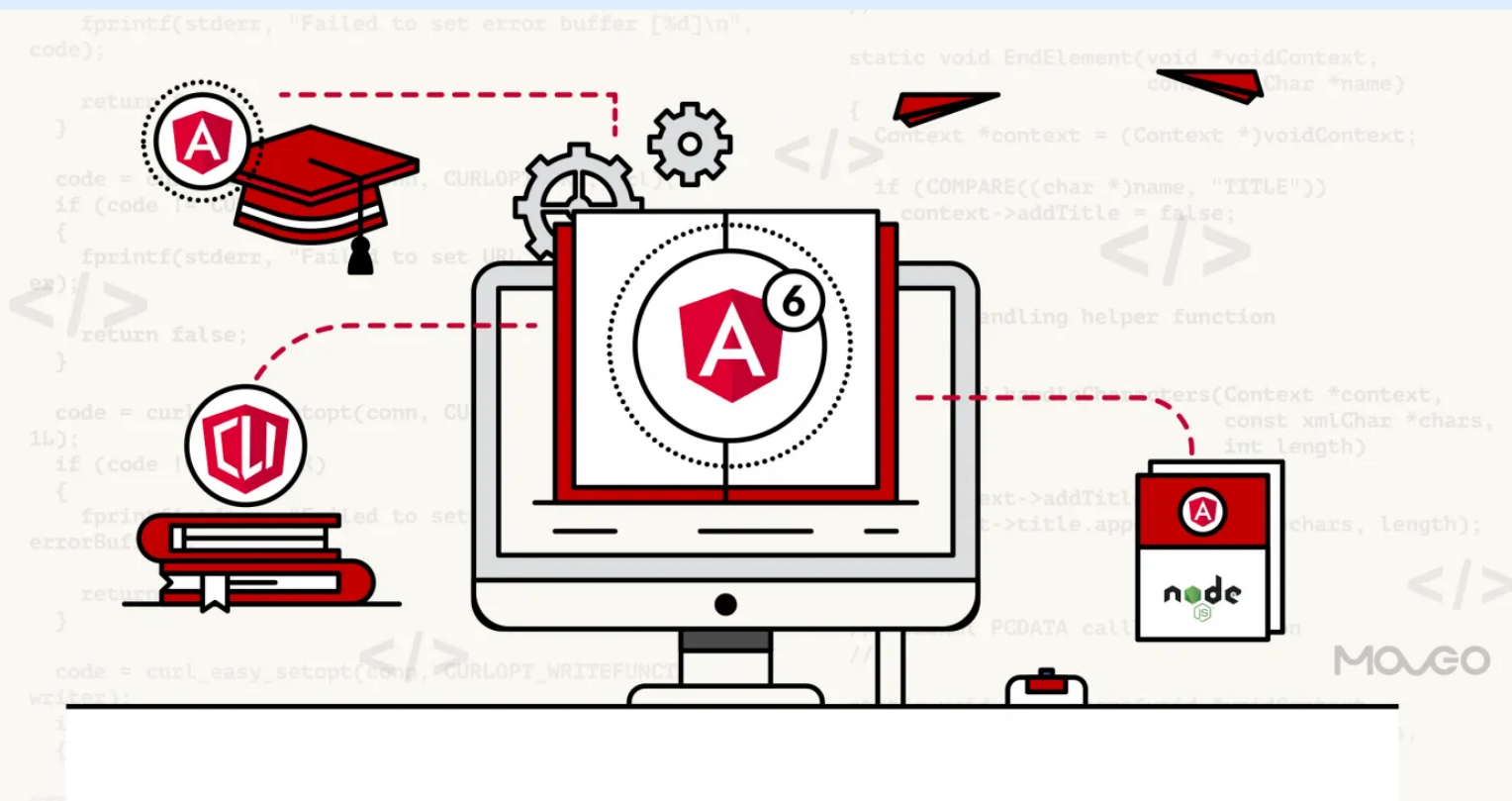
Throughout the development cycle, some of the processes became crucial for the software while changes to the initial functionality were being made. Many factors are well-balanced within Angular, which has a positive impact on the program's operation.

Angular offers developers the default options for things like:

- **network connectivity;**
- **language selection (this is not a standard feature of Angular, but can be easily integrated) ;**
- **a set of configuration tools.**

These parameters are constantly being tested and adjusted, so the Angular platform is continuously and steadily developing.

Angular was created according to the freedom of choice principle. Depending on the needs of a particular organisation, Angular allows you to change the key parameters or select [any UI Library](#) that suits your task. This concept is epitomised by the [Material Design library](#).



Angular Created for Scale

Angular was created by Google to solve Google-scale problems. This means millions of lines of code, thousands of engineers and a variety of project schedules, requirements and workflow. The platform was designed to enable the creation and management of common code, as well as the sharing of work among the respective roles. Many teams have individual designers, engineers, quality controllers and many other roles. The component model used in Angular was designed to separate these tasks and improve the interaction of a large number of developers. Angular is thus primed for teamwork and complicated projects.

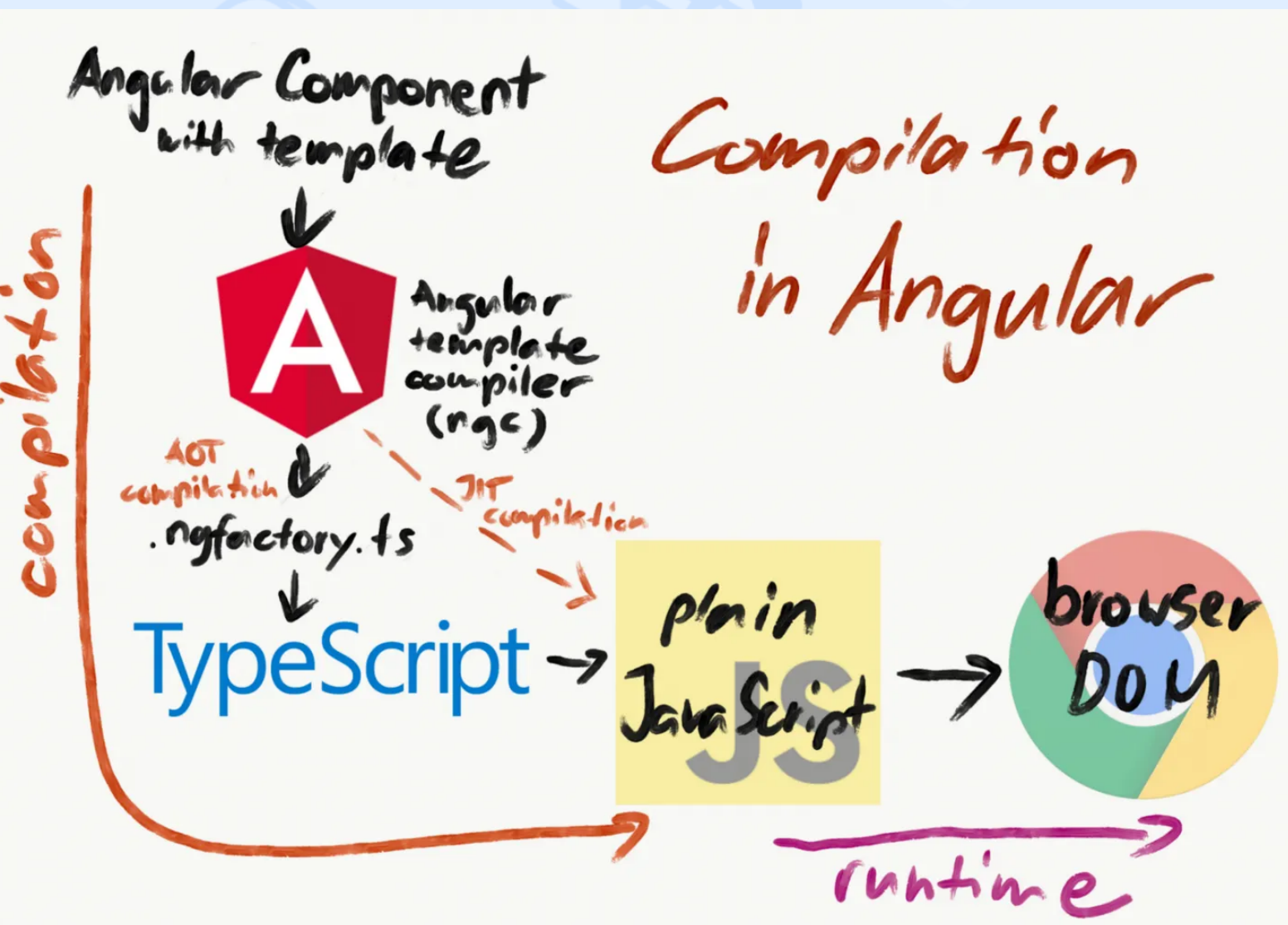
Convenience

Maintenance is the next key concern of many developers. Angular solves this problem in two ways:

1. Angular developers use TypeScript, which allows them to avoid certain types of bugs. It is especially helpful for working in a large team. In addition, TypeScript allows developers who are not familiar with the code base to get involved quickly, as they will be able to familiarise themselves easily with the data types in the app.
2. Angular relies on the convenience of testing. [Dependency injection](#) is the key element of Angular, which makes it easier to write tests. It also includes support for end-to-end testing with [Protractor](#).

Additionally, it prevents some bugs from appearing in the program code, connected with variable types, and helps keep the code safe in unforeseen scenarios. For example, when a variable is null or

undefined, it requires that an extra check be carried out.



Reliability

The Angular team is committed to stable and systematic development and adheres to the concept of public announcement of the release schedule, which allows companies to draw conclusions and make plans for the future about changes in the platform.

Due to Angular's being a Google product, it takes full advantage of the latter's testing infrastructure. Every change is checked for compliance with every Angular project within Google. This means that

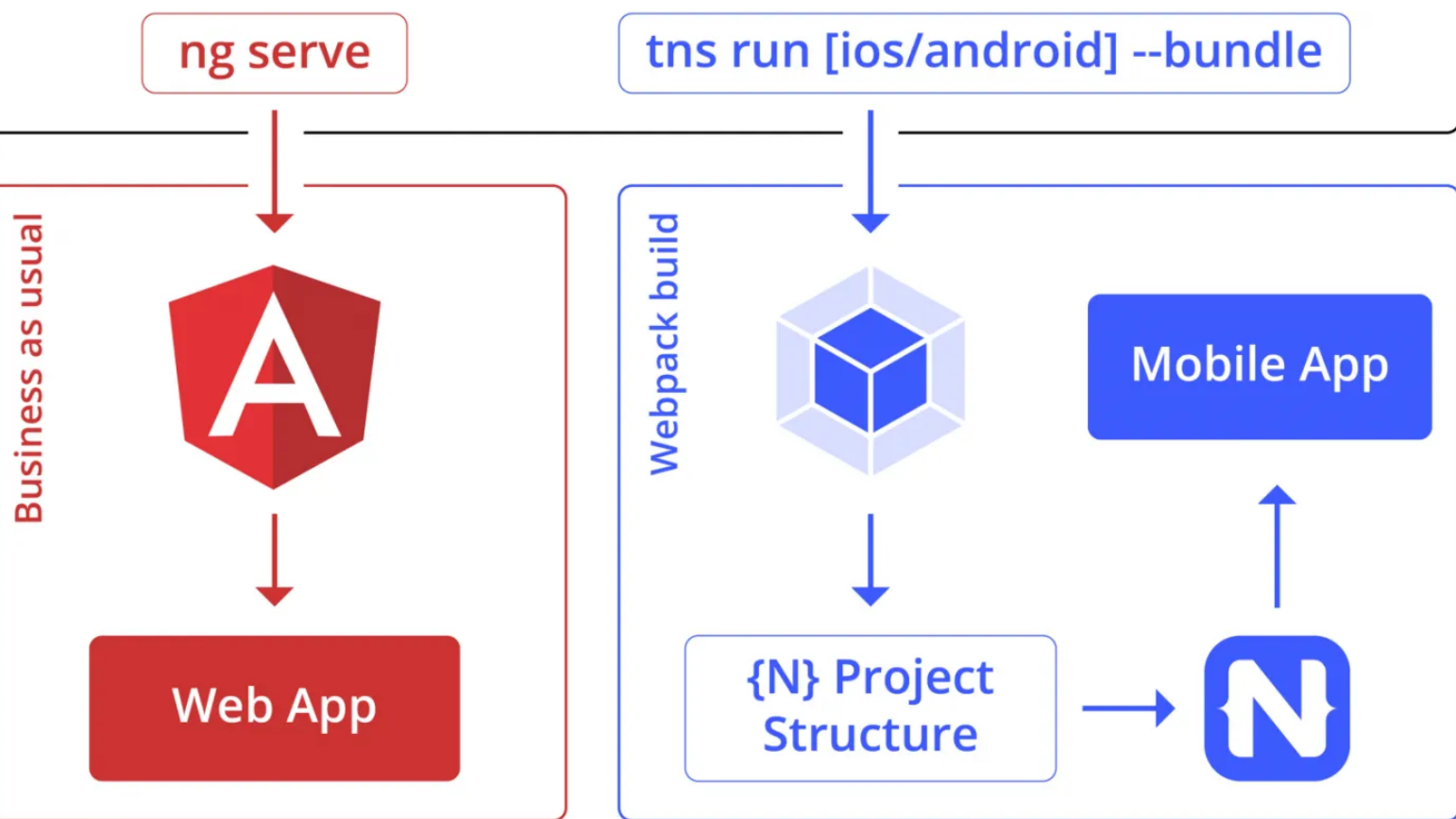
even before any public release, the framework is already being applied to hundreds of projects. This approach reduces the chances of any unintended critical changes or regressions appearing.

What can Angular Provide a Developer with?

I would highlight the following advantages:

- Google, Microsoft support;
- Developer Tools (CLI);
- Unified project structure;
- TypeScript “out-of-the-box” (you can write strongly typed code);
- Reactive programming with RxJS;
- The only framework with Dependency Injection;
- HTML-based templates;
- Cross-browser Shadow DOM (or its emulation);
- Cross-browser support for HTTP, WebSockets, Service Workers;
- No need to customise anything. No more wrappers;
- More modern framework than AngularJS (at the level of [React](#), [Vue](#));
- A sizeable [community](#).

Shared Project



To keep things honest, it's also worth mentioning the disadvantages:

- The entry threshold is higher due to Observable (RxJS), TypeScript, and Dependency Injection;
- For everything to work well and quickly, you'll need to spend time on additional optimisations (it's not super fast by default, but is many times faster than AngularJS and gets faster with each new version);
- In fact, if you're planning to develop a large enterprise app, then you don't have an architecture for managing the state out-of-the-box – you need to add Mobx, MVVM, Redux, CQRS / CQS or another state manager so as not to rack your brains later;
- [Angular Universal](#) has many pitfalls;
- Dynamic component creation is not a trivial task.

In fact, all these disadvantages can be levelled out by the developer's own experience. Everything you need to learn in Angular to develop productive and fast-running apps of any level of complexity is described in the following concepts.

Form Builder

In order to develop truly complex forms, you should know reactive forms, or rather, forget about declarative forms. Here is one of the better [examples](#) (reactive form + validation);

Change Detection

Since Angular uses a two-way data binding model (Model-View-ViewModel) by default, when working with large volumes of such data your apps will run more slowly, so in some cases you will have to care of an appropriate change detection strategy. You can look at various OpenSource projects: PrimeNG, Angular Material, Clarity UI, Angular Bootstrap and others, all of which use ChangeDetection.

Templating

The syntax of templates from the point of view of abstraction hasn't changed too much compared to AngularJS – that is, we can also write conditions, cycles and so on. All you'll need a good understanding of are structural and declarative directives, as well as Input parameters and Output events.



Routing

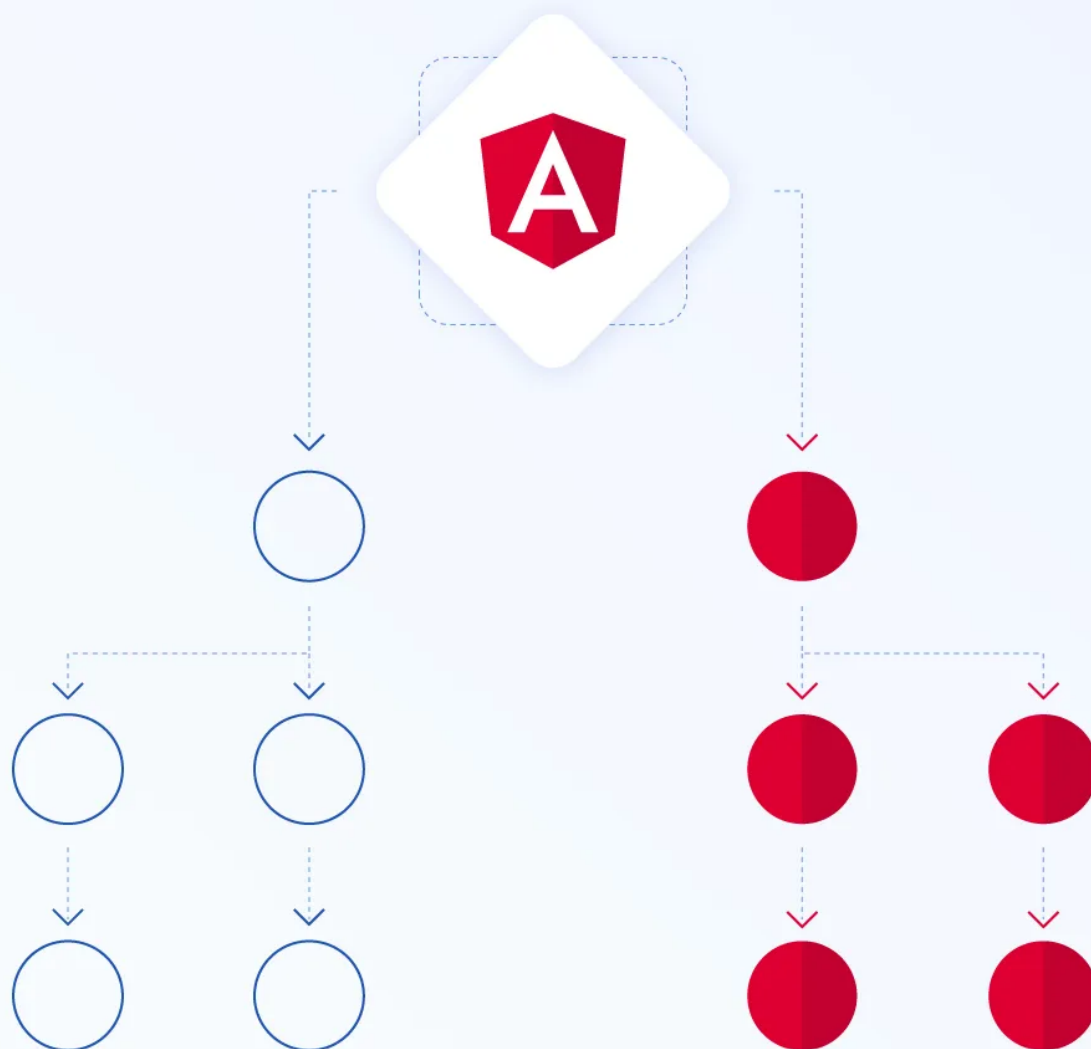
Routing is probably one of the fundamental phenomena of [web app development](#). It's just important to understand that routing, like components, has its own life cycle. If you understand this, you can write really cool apps. It's also worth noting that if you hang a module on any of the routes, as opposed to a component that is responsible for displaying the page along this route, the module will be loaded on the page on demand.

Annotations

This is something many beginners don't know about, yet something worth noting. Annotations, which are used in abundance when writing apps on Angular, are not some kind of rigid TypeScript magic. Annotations are EcmaScript specifications, and when browsers start to support them, they will be natively executed in browser runtime. In fact, annotations are very useful and provide your code with a fairly high level of readability.

Observables

It is Observables, in fact, will form the EcmaScript specification, and all this will be natively supported in browsers. From the point of view of theory, if you expand the concept, Observer is a behavioural design pattern also known as Dependents. It creates a mechanism for the class, allowing you to receive an instance of an object of this class by notifying other objects about changes to their state, thereby observing them.



Shadow DOM

Shadow DOM is a tool for creating a separate DOM tree inside an element that is not visible from the outside without applying special methods. This is the W3C specification. Roughly speaking, this is a convenient way to create isolated and reusable web components. Technically, if browsers already supported many of the concepts that Angular uses, we wouldn't need transpilers and other build systems – everything we wrote on Angular would work natively.

Angular Popularity

As shown by Google Trends, Angular and React are two of the most popular web technologies in 2018 – as well as, according to analysts' [forecasts](#), for 2019-2020.

Today, developers with the requisite knowledge and ability to develop apps on Angular are highly valued.

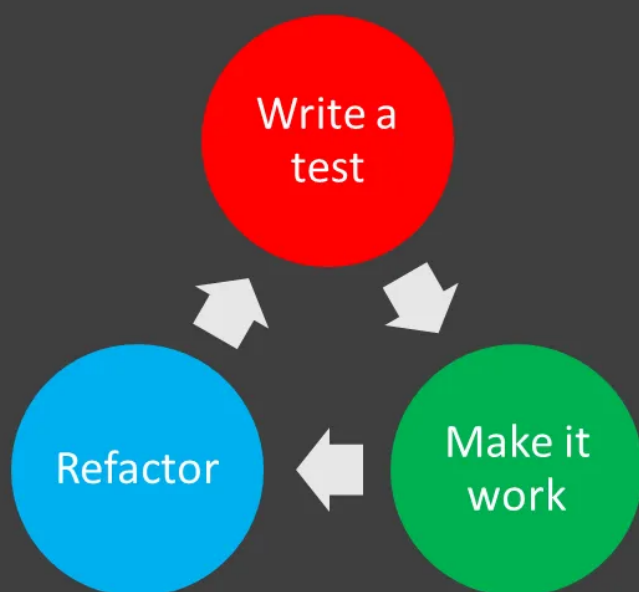
Websites and web apps developed with Angular

There are many major websites built using Angular. Some of them are as follows:

- [PayPal](#)
- [The Guardian](#)
- [Netflix](#)

We also make good use of Angular in our work. Here are some examples of our projects:

- [Vileda Sales tool](#)
- [StarBuffet app](#)
- [SwipedOn](#)



In general, Angular is a handy development tool for solving various programming tasks, such as:

- **Developing single-page apps (SPAs).** The browser doesn't reload the entire page but only fragments of it, while the user interacts with the app (presses buttons, links etc.). Angular has a Router to navigate inside the app. Inside the component template, using the `<router-outlet>` tag, you can select one or more areas to display different parts of the page. For example, in a Twitter application, you can select the middle part of the page under `<router-outlet>`, and if the user clicks the Notifications menu, another component will be shown in this area.
- **Braking an app down into modules.** An app should have one root module with minimal functionality, and features can be implemented in separate modules, for example, Billing, Shipping, etc. Moreover, these modules can be "lazy loaded" to minimize the size of the app page. For example, the main page of the app doesn't load the Billing module until the user clicks the "Pay" button. And, you can easily make such features automatically load in the background after the main page is already loaded and the user has started working with the app.
- **Creating forms.** It's hard to imagine a web app where forms are not needed. Well, at least you need to log in. Behind each form is an object with a data model that stores all values entered by the user. Angular offers two APIs for forms: template-driven and reactive. The one that is template-driven allows you to encode forms without bothering to create a model in TypeScript. Just sprinkle HTML-forms with several directives, and Angular will create a model. For more advanced work with forms, use the reactive Forms API.



ANGULAR



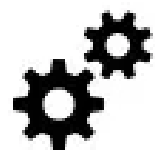
Write Less
Code



Two-Way
Data Binding



Power of
MVVM



Eases
Testing

Looking for the professional Angular developers on the market?

Fortunately, at Magora we have senior developers who have perfectly mastered in Angular and can solve any task using this advanced technology.

So, if you have any - we are at your service.