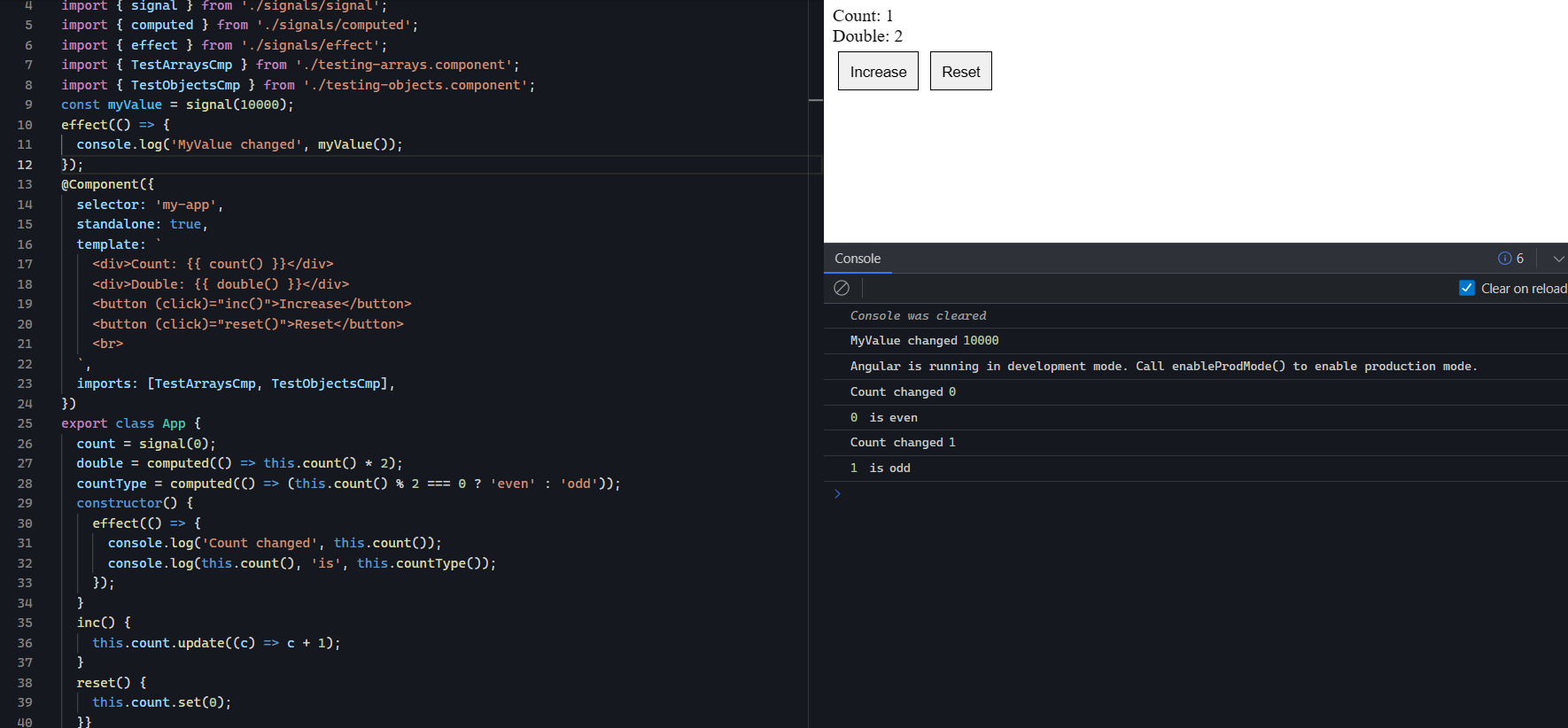
**What's new in Angular 16?**

* With the release of Angular 15, the creators of the Typescript-based web framework established by Google are looking to Angular 16 for enhancements to server-side rendering and runtime performance.
* Angular 16 would investigate hydration and server-side rendering usability enhancements, with non-destructive hydration as a first step.
* It is tentatively scheduled to launch in May 2023.
* The server-side DOM may be reused with this method, which would simply add event listeners and the data structures the Angular runtime requires, rather than rendering everything from scratch.
* Plans call for more research to be done on the constantly changing area of partial hydration and resumeability as a following step.
* Each strategy includes drawbacks, thus Angular's developers want to be well-informed when choosing the best course of action.

**Signals**

* Inspired by Solid.js, signals are a new method of controlling state changes in Angular apps. Signals are functions that can be updated by calling them with a new value (set()), and they return a value (get()).
* Signals may also be dependent on one another, resulting in a reactive value graph that updates itself automatically whenever a dependency changes.
* RxJS observables, which are still available in Angular v16, can be combined with signals to construct robust and expressive data flows.
* We can establish the default value and change it as necessary with the help of signals.
* Example of Signals.



**References** :

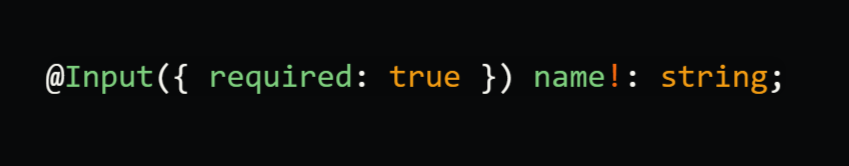
* <https://youtu.be/oGDgjIV7jrs>
* [Sarah Drasner on Twitter: "What if I told you Angular is about to have a renaissance? 🎆 Angular just released a new Signals reactivity primitive and it's just the first step! I made a little explainer about what it is: https://t.co/I5EL6zS9O9 https://t.co/CM1ZDc4R5w" / Twitter](https://twitter.com/sarah_edo/status/1628065696247857152)

# Zone.js (Zoneless) and Reactivity Model

* The reconsideration of the reactivity model and the making of Zone.js optional in Angular v16 are two of the most eagerly awaited developments.
* A package called Zone.js uses browser API monkey-patches to detect changes and start change detection in Angular applications.
* While doing so makes Angular simple to use and create, it also increases the framework's overhead and complexity.
* Zone.js will be optional in Angular v16, allowing developers to choose to manage reactivity using RxJS or signals instead.

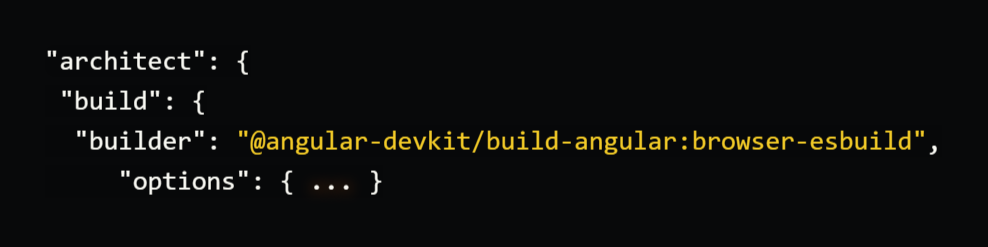
# Required Input

* The community eagerly expected the opportunity to make specific inputs mandatory.
* There have been a number of workarounds employed up until now to accomplish this:
* If the variable was not declared, an error would be raised in the NgOnInit Lifecycle, and our component's selector would be changed to include the various required inputs.
* Both of these ideas had benefits, but they also had drawbacks.
* Making an input mandatory will be a straightforward configuration object that is supplied to the input annotation's metadata starting with version 16.



**Vite as Development Server**

* With the release of Angular 14, a new Javascript Bundler is now available for use: EsBuild
* The build time could be cut by roughly 40% thanks to this new Bundler, which has the potential to be quite quick. The main issue is that this new functionality and efficiency improvement could only be utilised for builds rather than development (dev server).
* Esbuild can now be utilised during development in the upcoming version of Angular thanks to Vite.
* Update the builder as follows in angular.json to enable this feature:



# Non-Destructive Hydration

* By affixing JavaScript behaviour and event listeners, the process of transforming server-side produced HTML material into a fully interactive and useful web page on the client-side is referred to as hydration in web development.
* This increases SEO and accessibility while decreasing the time-to-interactive (TTI).
* Hydration has long been a feature of frameworks like React or Next.js, but implementing it in Angular was challenging.
* Hydration will be supported out of the box in Angular 16, which will speed up and improve SSR applications.

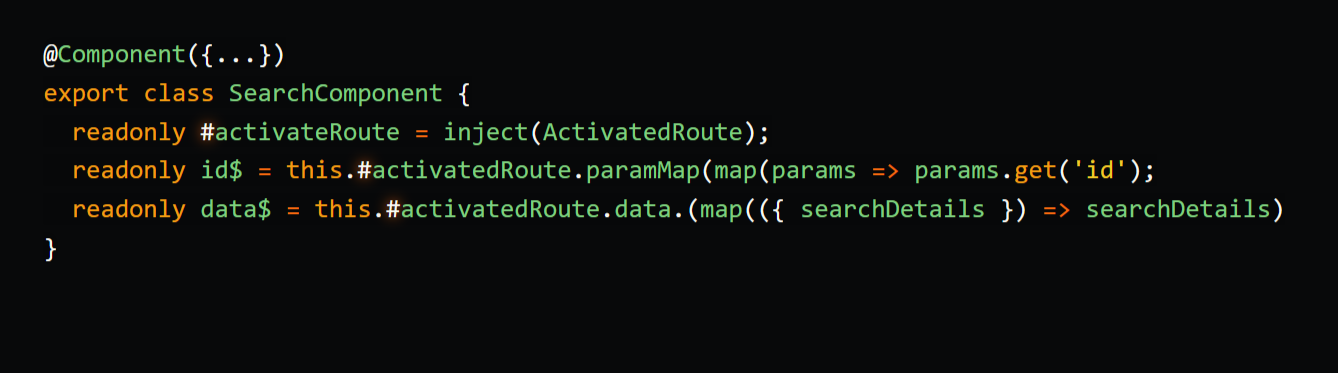
# New DestroyRef injector

* DestroyRef, a new resource made available with Angular v16, enables the registration of destroy callbacks for a particular lifecycle scope.
* Components, directives, pipelines, embedded views, and instances of EnvironmentInjector are all compatible with this capability.
* This usage is really basic.



# Automatic route params mapping

Prior to Angular 16, the ActivatedRoute service had to be explicitly injected in order to obtain not just the url parameter but also any query parameters or associated data.

****

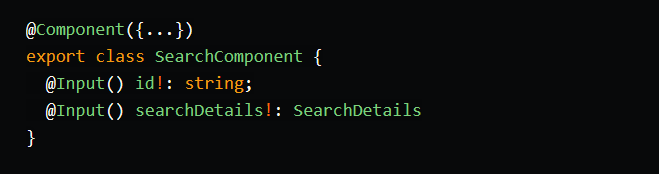
* With Angular 16, the various route parameters can be directly bound to component inputs, eliminating the need to inject the ActivatedRoute service to obtain them.
* The RouterModule settings contain the ability to activate this functionality for an application that makes use of the module system.



* This is a function that should be called by a standalone programme.

****

* The component is greatly simplified after the functionality has been engaged.

****

**Conclusion**

* Without a doubt, Angular 16 introduces a few interesting new features.
* Some of them, like signals or vite like dev server, are still in the experimental stage.
* Whatever the case, these new features will undoubtedly alter the way we code our Angular applications by reducing boilerplate, making them even more optimised, and by making it easier to integrate cutting-edge technologies like vitest or dramatist.
* Version 16 of Angular has not yet been made available.
* This article's descriptions of some api may still change.
* However, this offers you a general notion of what to anticipate from Angular's upcoming version.