

# Web Components/Angular Elements/Microfrontends

## Custom Elements, Shadow DOM, NgElement, createCustomElement, App Shell, Architecture

Full stack developers have been successfully using microservices server-side for a few years and now they would like to apply that architecture client-side, an approach known as microfrontends. In some ways microfrontends are similar to microservices: a clear need to break up a monolith application, allow different parts of a large app to evolve and be deployed at their own pace, perhaps using distinct foundational technologies. In other ways, they are different: microservices can run separately in data centers on a (e.g. Kubernetes) container cluster, whereas we wish a set of microfrontends to run isolated in a web browser

and yet appear to end-users to be a single integrated interactive application, with some shared capabilities. In this specialist course we first review W3C Web Components - a set of standards that allow components to be produced and consumed by different web frameworks (or different versions of the same framework). Then we look at Angular Elements - new to Angular 6 - which allows the construction of web components using Angular. Then we explore microfrontends - what are they, how to best build them using Angular Elements, how to host them (app shell) & how to design large-scale web applications using them.

<b>Contents of One-Day Training Course</b>	
<p><b>Target Audience</b> Experienced Angular developers working on large Angular projects who wish to compose them out of microfrontends.</p> <p><b>Prerequisites</b> Full stack developers with good all-round experience of Angular 6.1/7.</p> <p>Awareness of role of containers and microservices for server-side development highly relevant.</p>	<p><b>Overview</b> What are we trying to achieve? Dividing an app into dynamic components * Standards - W3C Web Components * Implementation - Angular Elements * Design approach - Microfrontends</p> <p><b>W3C Custom Elements</b> Create your own HTML elements Attribute, properties, events New CustomElementRegistry How we can use them for microfrontends</p> <p><b>W3C Shadow DOM</b> Shadow tree and light tree Angular <a href="#">ViewEncapsulation.ShadowDOM</a> (note: <a href="#">Native is deprecated</a>)</p> <p><b>Support Features: Slots, HTML Templates/Custom Events</b> Relevant additional HTML/DOM features that modern browsers support</p> <p><b>Composability in Angular</b> Angular dynamic components Lazy loading &amp; extensibility Specialist use of NgModules</p> <p><b>Angular Elements Intro</b> NgElement, ngBootstrap() createCustomElement() Schematics</p> <p><b>Advanced Angular Elements</b> Build process Managing element lifecycle / evolution Loading a library containing elements Review of <a href="#">Angular Elements' source tree</a> Importance of Render3/Ivy [Angular 7]</p> <p><b>Microfrontend Architecture</b> What microservices brings to server app How best to apply idea client-side Microfrontend = Large (somewhat contained) slice of interactive app Critical to make group of them appear as an integrated app</p> <p><b>Building Microfrontends</b> Structuring microfrontends Source layout Expected classes and interfaces Design patterns</p> <p><b>App Shell</b> Microfrontends have to live within a shell Custom app shell hosts microfrontends What capabilities it could offer: notification – recently used-quick links-app discovery</p> <p><b>Microfrontends and ..</b> Routing Internationalization Security Styling Sharing widgets</p> <p><b>Browser Containers</b> Future idea: containers in the browser <a href="#">containers = namespaces + cgroups</a> How best to bring idea to browser [polyfill] Hosting microfrontend in browser container</p> <p><b>Project</b> Bringing together everything covered in this course, we conclude with a review of a substantial project that uses microfrontends, based on Angular Elements.</p>