

Angular 7.1 Security and Authentication

CSP, Contexts, Sanitizers, Schema, XSS, CSRF, CORS, XSSI, Authentication Workflow, Web-Authn

There are many design choices web developers make every day that can positively/negatively impact web app security. Security is not a task just to be left to security experts (though having them on the team is certainly a good idea). No, every web developer needs a strong grounding in both web security in general and the security of the web framework they use in particular. This course supplies both for Angular 7.1 app developers. We start with a thorough review of general browser security and then proceed to see how Angular can help in building secure web applications, including exploring in detail how to build authentication.

Angular has a compelling security story, responding well to potential attack vectors. By default an Angular CLI-generated app is very secure. As code is added, security settings can be carefully adjusted as needed.

Authentication is one of the most complex and yet most important aspect of any substantial Angular application. By breaking it into manageable chunks, attendees will appreciate how a well organized authentication workflow should be, and see how to build this inside an Angular application (using [NgRx 7](#) to store the auth token). We also look at authorization and auditing.

Contents of One-Day Training Course	
<p>Target Audience Angular 7.1/TypeScript developers wishing to gain a deeper understanding of how web security in general and security within an Angular app in particular work. Also those who need to implement authentication within an Angular 7.1 app.</p> <p>Prerequisites This is an advanced course and as prerequisites attendees must have an understanding of security fundamentals and general experience of Angular 7.1 development</p>	<p>Overview of Browser Security TLS / SSL Input validation / output encoding Client-side security Open Web Application Security Project Content Security Policy (CSP) CSP feature tour Directives Policy definition Integration with other specifications Strict Transport Security “Defines a mechanism enabling web sites to declare themselves accessible only via secure connections” - RFC Web Cryptography Running cryptographic algorithms inside a browser (AES, RSA, HMAC, SHA, etc.) A W3C recommendation from Jan 2017 Overview of Angular Security Angular’s security best practice XSS CSRF Security contexts & sanitizers “Security risk” marking in doc Cross-Site Scripting (XSS) Idea of malicious code-centric Protecting the DOM Tackling XSS in Angular Sanitizers Types of Angular security contexts Role of sanitizers Custom sanitizers Review of schema for security</p> <p>Cross-Site Request Forgery (XSRF) What browser app needs to prevent XSRF Angular 7.1 HttpClient and XSRF CORS Cross-Origin Request and CORS request A server-side feature / Enabling CORs Cross-Site Script Inclusion (XSSI) JSON APIs and security Preventing execution of JSON responses Dev Tools & Security Review of Chrome Devtools’ Security tab Lighthouse Security audits Angular 7 Authentication Intro Designing auth workflow for Angular applications Login / logout UI & status UI API calls for auth and retrieving token Storing JWT auth token / supplying to APIs AuthGuard for routing Implementing Authentication Important decisions & security implications Public vs. secure routing targets Using NgRx 7 to manage auth token Two-factor authentication (using Twilio) Extending with authorization and auditing Web Authentication API FIDO and W3C have released web-authn Use of web-authn by secure Angular 7 app</p>