

TypeScript/Node 11/Angular 7 Toolchain

Understand, Organize, Build, Write, Document, Test, Optimize, Verify, Debug, Package, Explore, Deploy

Lots of things have to happen correctly in order to efficiently transform source code on developers' computers into commercial digital products in use by customers around the world.

After writing the source, the code has to be transpiled, unit tested, documented for developers, stored in repositories, optimized, debugged, the performance monitored, the source style has to be verified, libraries and components packaged, minified & applications deployed. For TypeScript developers, a toolchain is needed with appropriate tools for each of these tasks.

The TypeScript/Node/Angular toolchain often gets less attention than the language/runtimes themselves but it is vital for highly productive developers to have a deep understanding of the toolchain and what it can offer. As projects get larger and there is pressure to deliver updates in shorter time frames but with higher quality expectations, successfully leveraging the TypeScript/Node/Angular toolchain and all its rich capabilities makes all the difference.

This course explores a range of useful developer tools that when used together achieves just that.

Contents of One-Day Training Course	
<p>Target Audience Software engineers who wish to more fully explore the toolchain options available for TypeScript projects for Node and/or Angular.</p> <p>Prerequisites Experience of TypeScript programming for Node and/or Angular.</p>	<p>Toolchain Tour Range of tools needs for high-productivity high-quality TypeScript/Node/Angular programming Node 11's Inspector Protocol</p> <p>TSC – TypeScript Transpiler Tsc and its options More advanced uses for configuration file</p> <p>TypeScript Language Service Written in TypeScript, an external process supplying a range of language services How to call from non-TypeScript clients</p> <p>Visual Studio Code Code editor for various languages Use of Visual Studio Code for TypeScript programming, both Node and Angular</p> <p>TypeDoc Code Documentation TypeDoc auto-generating documentation Professional finish</p> <p>TSLint, etc. Palantir's TSLint is used to make sure code complies with styling/maintainability</p> <p>Angular Package Format The Angular Package Format is a well-specified layout used for Angular packages Third-party libraries should follow layout</p> <p>Angular Augury In-depth debugging of Angular content Chrome integration (v1.2x supports ng)</p> <p>WTF - Web Tracing Framework Low-level collection of precise high-volume tracing data Visualizing results</p> <p>ts-node Node-based execution environment for TypeScript code</p> <p>Gulp.ts Gulp is a very popular task runner We explore how to write gulp files in TypeScript (requires ts-node) Defining the build process using gulp.ts Handling the build & source directory tree</p> <p>Webpack 4 Role of module bundler Dependency graph and bundles</p> <p>NPM & Yarn Creating packages using Node Package manager (NPM) and Yarn Publishing to NPM Adding extra features to package (e.g. CI)</p> <p>Jasmine & Karma What to test Test syntax in Jasmine Executing tests using karma Karma.config.js</p> <p>Protractor End-to-end testing of angular apps Writing tests for entire integrated app Automated collection of results</p> <p>Real-world Toolchain Usage Exploring the toolchain setup for the main Angular project itself – a very large open source TypeScript that is widely deployed</p> <p>Project Correctly setting up toolchain usage for our own enterprise projects</p>