Angular Language Service

Contents ›

The Angular Language Service provides code editors with a way to get completions, errors, hints, and navigation inside Angular templates. It works with external templates in separate HTML files, and also with in-line templates.

Features

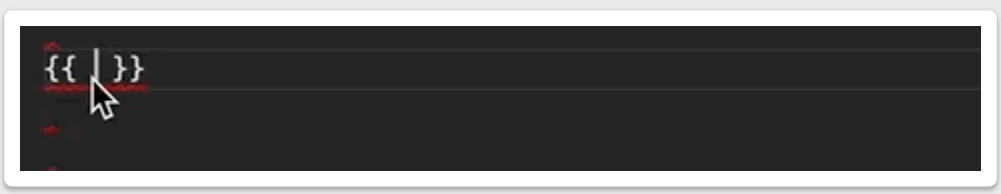
Your editor autodetects that you are opening an Angular file. It then uses the Angular Language Service to read your `tsconfig.json` file, find all the templates you have in your application, and then provide language services for any templates that you open.

Language services include:

- Completions lists
- AOT Diagnostic messages
- Quick info
- Go to definition

Autocompletion

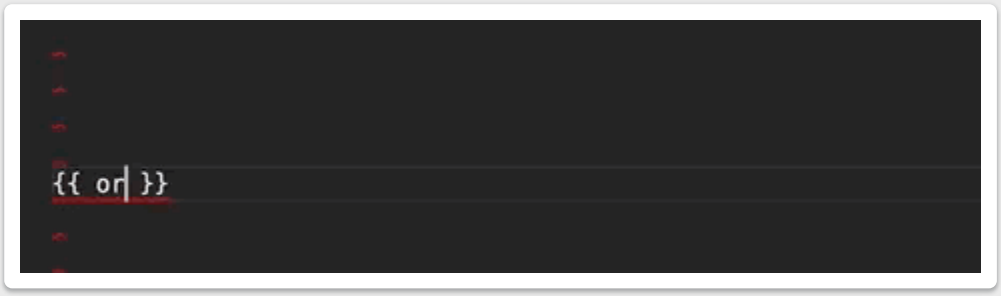
Autocompletion can speed up your development time by providing you with contextual possibilities and hints as you type. This example shows autocomplete in an interpolation. As you type it out, you can hit tab to complete.



There are also completions within elements. Any elements you have as a component selector will show up in the completion list.

Error checking

The Angular Language Service can forewarn you of mistakes in your code. In this example, Angular doesn't know what `orders` is or where it comes from.



Quick info and navigation

The quick-info feature allows you to hover to see where components, directives, modules, and so on come from. You can then click "Go to definition" or press F12 to go directly to the definition.

Angular Language Service in your editor

Angular Language Service is currently available as an extension for Visual Studio Code, WebStorm, and Sublime Text.

Visual Studio Code

In Visual Studio Code, install the extension from the Extensions: Marketplace. You can open the marketplace from the editor using the Extensions icon on the left menu pane, or use VS Quick Open (⌘+P on Mac, CTRL+P on Windows) and type "? ext".

In the marketplace, search for Angular Language Service extension, and click the **Install** button.

WebStorm

In WebStorm, you must install the language service package as a project dependency.

1. Add the following to your `devDependencies` in your project's `package.json`

package.json

```
devDependencies {
  "@angular/language-service": "^6.0.0"
}
```

2. In the terminal window at the root of your project, install the `devDependencies` with `npm` or `yarn`:

```
npm install
```

OR

```
yarn
```

OR

```
yarn install
```

When Angular sees this dev dependency, it provides the language service in the WebStorm environment. WebStorm then gives you colorization inside the template and autocomplete in addition to the Angular Language Service.

Sublime Text

In Sublime Text, the Language Service supports only in-line templates when installed as a plug-in. You need a custom Sublime plug-in (or modifications to the current plug-in) for completions in HTML files.

To use the Language Service for in-line templates, you must first add an extension to allow TypeScript, then install the Angular Language Service plug-in. Starting with TypeScript 2.3, TypeScript has a plug-in model that the language service can use.

1. Install the latest version of TypeScript in a local `node_modules` directory:

```
npm install --save-dev typescript
```

2. Install the Angular Language Service package in the same location:

```
npm install --save-dev @angular/language-service
```

3. Once the package is installed, add the following to the `"compilerOptions"` section of your project's `tsconfig.json`.

tsconfig.json

```
```

```
"plugins": [
    {"name": "@angular/language-service"}
]
```

4. In your editor's user preferences (`Cmd+,` or `Ctrl+,`), add the following:

Sublime Text user preferences

```
"typescript-tsdk": "/node_modules/typescript/lib"
```

This allows the Angular Language Service to provide diagnostics and completions in `.ts` files.

How the Language Service works

When you use an editor with a language service, the editor starts a separate language-service process and communicates with it through an RPC, using the Language Server Protocol. When you type into the editor, the editor sends information to the language-service process to track the state of your project.

When you trigger a completion list within a template, the editor first parses the template into an HTML abstract syntax tree (AST). The Angular compiler interprets that tree to determine the context: which module the template is part of, the current scope, the component selector, and where your cursor is in the template AST. It can then determine the symbols that could potentially be at that position..

It's a little more involved if you are in an interpolation. If you have an interpolation of `{{data.---}}` inside a `div` and need the completion list after `data.---`, the compiler can't use the HTML AST to find the answer. The HTML AST can only tell the compiler that there is some text with the characters "`{{data.---}}`". That's when the template parser produces an expression AST, which resides within the template AST. The Angular Language Services then looks at `data.---` within its context, asks the TypeScript Language Service what the members of `data` are, and returns the list of possibilities.

More information

- For more in-depth information on the implementation, see the Angular Language Service API.
- For more on the design considerations and intentions, see design documentation here.
- See also Chuck Jazdzewski's presentation on the Angular Language Service from ng-conf 2017.

Version 10.0.10-local+sha.84d1ba792b.