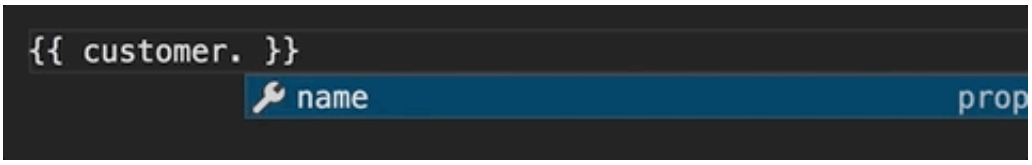
Angular Language Service

The Angular Language Service is a way to get completions, errors, hints, and navigation inside your Angular templates whether they are external in an HTML file or embedded in annotations/decorators in a string. The Angular Language Service autodetects that you are opening an Angular file, reads your `tsconfig.json` file, finds all the templates you have in your application, and then provides language services for any templates that you open.

Autocompletion

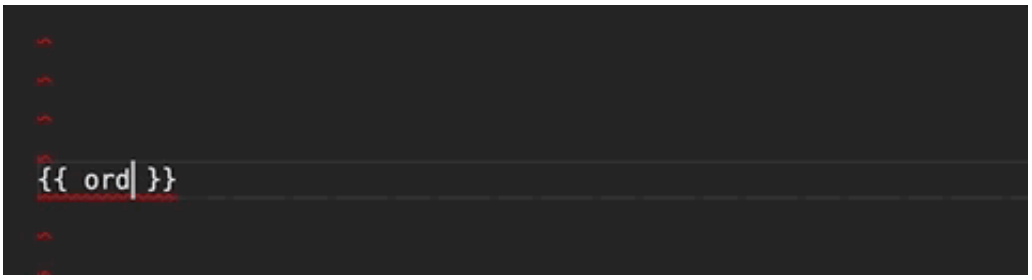
Autocompletion can speed up your development time by providing you with contextual possibilities and hints as you type. This example shows autocomplete in an interpolation. As you type it out, you can hit tab to complete.



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

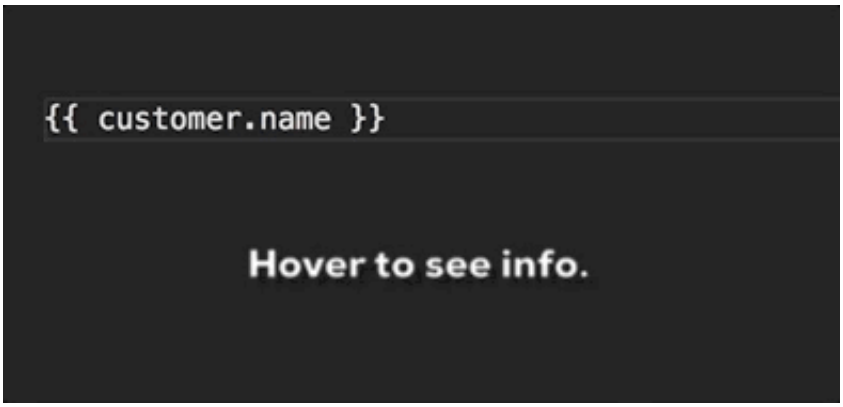
Error checking

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



Navigation

Navigation allows you to hover to see where a component, directive, module, etc. is from and then click and press F12 to go directly to its definition.



Angular Language Service in your editor

Angular Language Service is currently available for Visual Studio Code and WebStorm.

Visual Studio Code

In Visual Studio Code, install Angular Language Service from the store, which is accessible from the bottom icon on the left menu pane. You can also use the VS Quick Open (⌘+P) to search for the extension. When you've opened it, enter the following command:

```
ext install ng-template
```

Then click the install button to install the Angular Language Service.

WebStorm

In webstorm, you have to install the language service as a dev dependency. When Angular sees this dev dependency, it provides the language service inside of WebStorm. Webstorm then gives you colorization inside the template and autocomplete in addition to the Angular Language Service.

Here's the dev dependency you need to have in `package.json`:

```
devDependencies {
    "@angular/language-service": "^4.0.0"
}
```

Then in the terminal window at the root of your project, install the `devDependencies` with `npm` or `yarn`:

```
npm install
```

OR

```
yarn
```

OR

```
yarn install
```

Sublime Text

In Sublime Text, you first need an extension to allow Typescript. Install the latest version of typescript in a local `node_modules` directory:

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

Then install the Angular Language Service in the same location:
`sh npm install --save-dev @angular/language-service`

Starting with TypeScript 2.3, TypeScript has a language service plugin model that the language service can use.

Next, in your user preferences (`Cmd+,` or `Ctrl+,`), add:

```
"typescript-tsdk": "<path to your folder>/node_modules/typescript/lib"
```

Installing in your project

You can also install Angular Language Service in your project with the following `npm` command:

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

Additionally, add the following to the `"compilerOptions"` section of your project's `tsconfig.json` .

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

Note that this only provides diagnostics and completions in `.ts` files. You need a custom sublime plugin (or

modifications to the current plugin) for completions in HTML files.

How the Language Service works

When you use an editor with a language service, there's an editor process which starts a separate language process/service to which it speaks through an RPC. Any time you type inside of the editor, it sends information to the other process to track the state of your project. When you trigger a completion list within a template, the editor process first parses the template into an HTML AST, or abstract syntax tree. Then the Angular compiler interprets what module the template is part of, the scope you're in, and the component selector. Then it figures out where in the template AST your cursor is. When it determines the context, it can then determine what the children can be.

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 and asks the TypeScript Language Service what the members of data are. TypeScript then returns the list of possibilities.

For more in-depth information, see the Angular Language Service API

More on Information

For more information, see Chuck Jazdzewski's presentation on the Angular Language Service from ng-conf 2017.