

Angular Libraries via ng-packagr in CLI v6

Vancouver Angular Meetup - May 11, 2018

Libraries and NPM

- Not an alternative to NPM (or is it?)
- Easier to share and maintain?

Generate library

- `ng generate library my-lib`
 - Goes into `projects/my-lib` folder.
- `ng build my-lib`
- `ng build my-lib --prod`
 - Goes into `dist/my-lib` folder.

Structure

└ projects	
└ my-lib	
└ src	
└ lib	
TS my-lib.component.spec.ts	U
TS my-lib.component.ts	U
TS my-lib.module.ts	U
TS my-lib.service.spec.ts	U
TS my-lib.service.ts	U
TS public_api.ts	U
TS test.ts	U
K karma.conf.js	U
{} ng-package.json	U
{} ng-package.prod.json	U
{} package.json	U
{} tsconfig.lib.json	U
{} tsconfig.spec.json	U
{} tslint.json	U

Import your library

- Angular 6 is required to import
- `import { something } from 'library-name';`

More experimental



API / @angular/core

defineInjectable

FUNCTION

EXPERIMENTAL

Construct an `InjectableDef` which defines how a token is constructed by the DI system, and in which injectors (if any) it will be available.

```
defineInjectable<T>(opts: {  
  providedIn?: Type<any> | 'root' | 'any' | null;  
  factory: () => T;  
}): never
```

Parameters

opts Type: `{ providedIn?: Type | 'root' | 'any' | null; factory: () => T; }`.

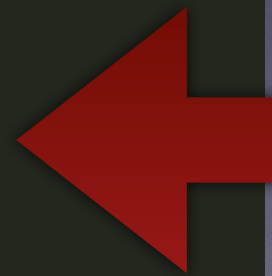
Returns

never

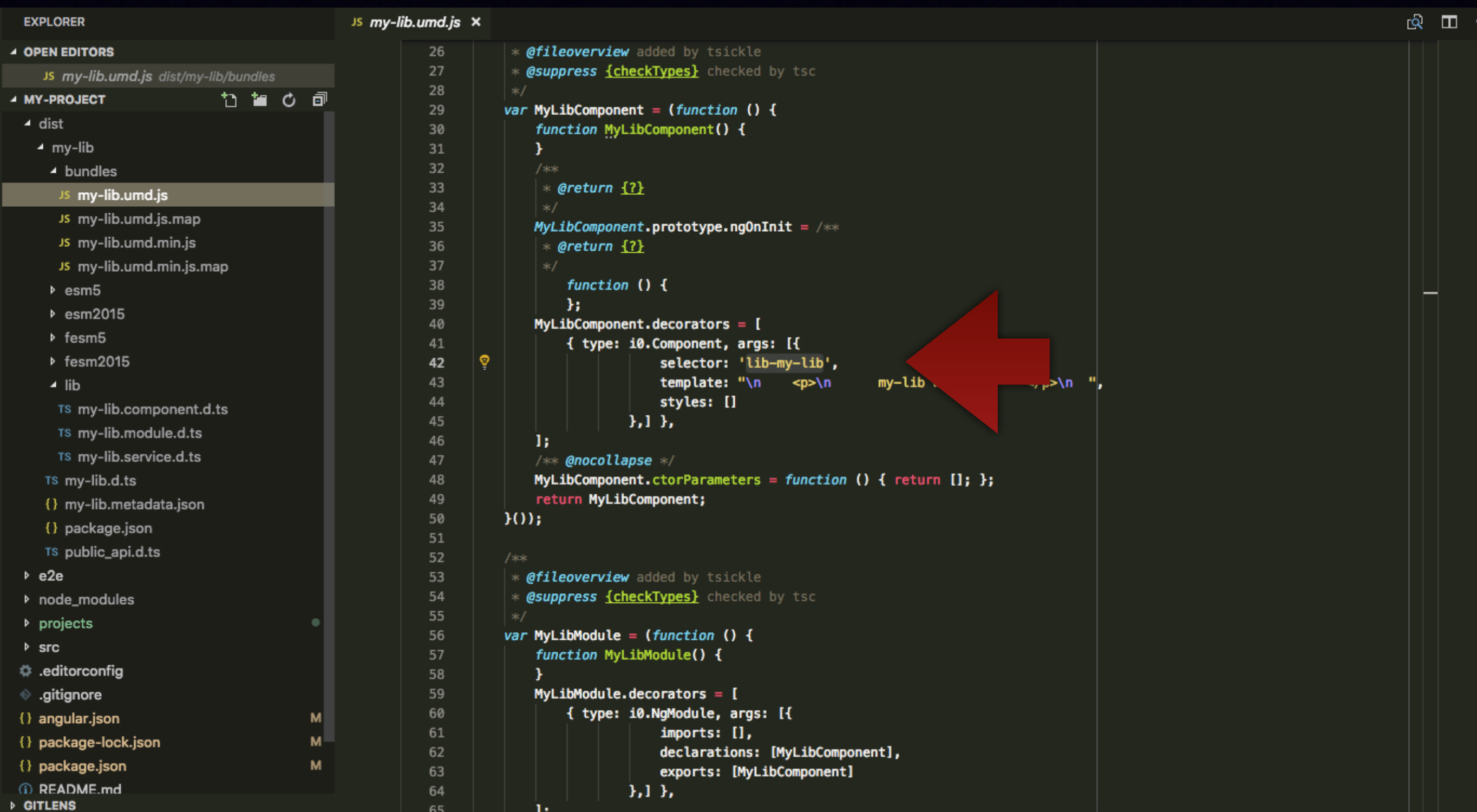
Using CLI build

```
{ } tsconfig.json x
You, a few seconds ago | 1 author (You)
1 {
2   "compileOnSave": false,
3   "compilerOptions": {
4     "baseUrl": "./",
5     "outDir": "./dist/out-tsc",
6     "sourceMap": true,
7     "declaration": false,
8     "moduleResolution": "node",
9     "emitDecoratorMetadata": true,
10    "experimentalDecorators": true,
11    "target": "es5",
12    "typeRoots": [
13      "node_modules/@types"
14    ],
15    "lib": [
16      "es2017",
17      "dom"
18    ],
19    "paths": {
20      "my-lib": [
21        "dist/my-lib"
22      ]
23    }
24  }
25 }
```

you can also
put it in
node_modules
as a hack



my-lib.umd.js

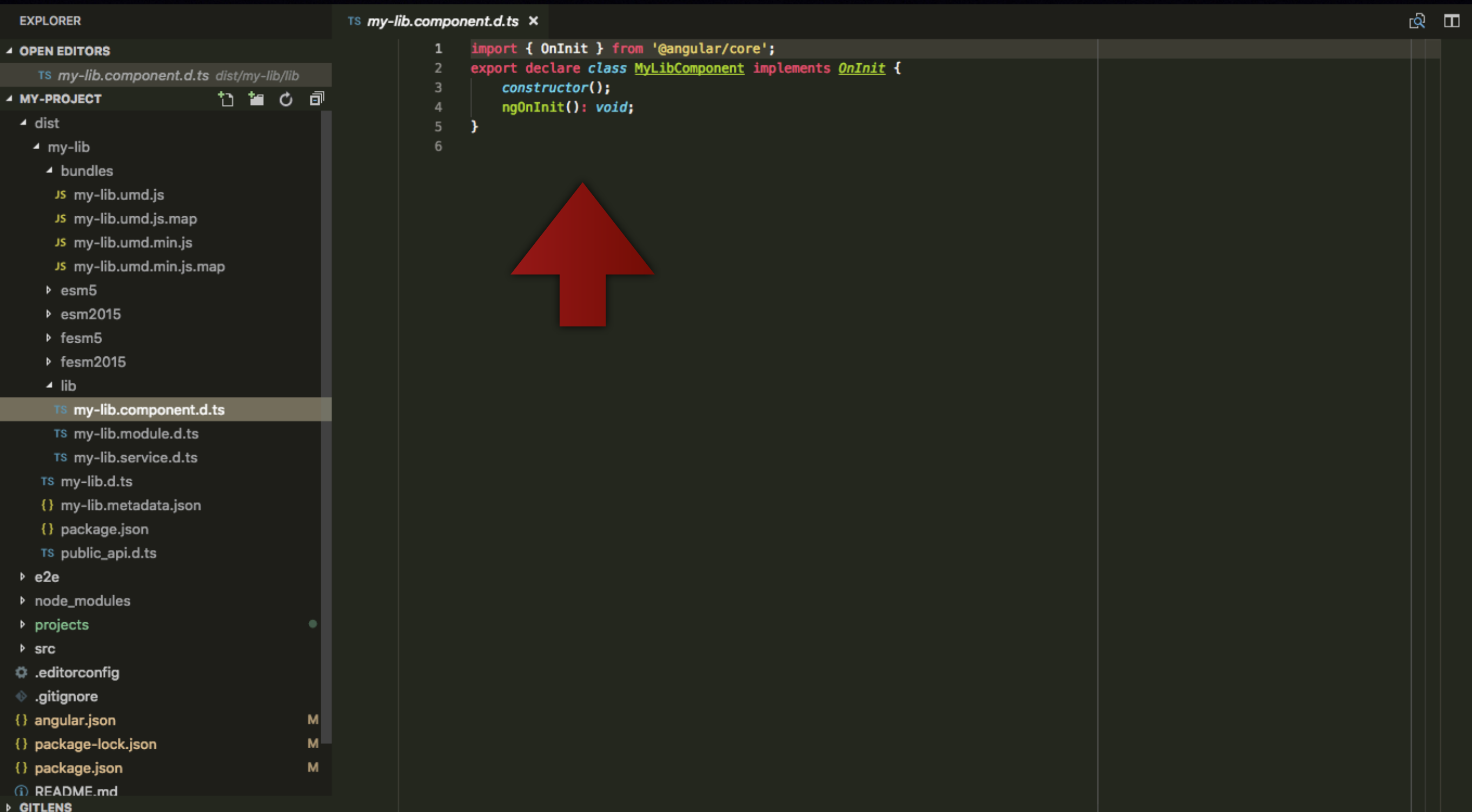


The screenshot shows the Visual Studio Code editor with the file `my-lib.umd.js` open. The Explorer sidebar on the left shows the project structure, including the `dist/my-lib/bundles` directory where the file is located. The code in the editor is as follows:

```
26  * @fileoverview added by tsickle
27  * @suppress {checkTypes} checked by tsc
28  */
29  var MyLibComponent = (function () {
30    function MyLibComponent() {
31    }
32    /**
33     * @return {?}
34     */
35    MyLibComponent.prototype.ngOnInit = /**
36     * @return {?}
37     */
38    function () {
39    };
40    MyLibComponent.decorators = [
41      { type: i0.Component, args: [{
42        selector: 'lib-my-lib',
43        template: "\n    <p>\n    my-lib\n    </p>\n    ",
44        styles: []
45      },,], },
46    ];
47    /** @nocollapse */
48    MyLibComponent.ctorParameters = function () { return []; };
49    return MyLibComponent;
50  }());
51
52  /**
53   * @fileoverview added by tsickle
54   * @suppress {checkTypes} checked by tsc
55   */
56  var MyLibModule = (function () {
57    function MyLibModule() {
58    }
59    MyLibModule.decorators = [
60      { type: i0 NgModule, args: [{
61        imports: [],
62        declarations: [MyLibComponent],
63        exports: [MyLibComponent]
64      },,], },
65    ];
```

A red arrow points to the `template` property in the `MyLibComponent` decorator on line 42, which is set to `"\n <p>\n my-lib\n </p>\n "`.

my-lib.component.d.ts



The image shows a screenshot of the Visual Studio Code editor interface. On the left, the Explorer sidebar displays the project structure under 'MY-PROJECT'. The 'lib' directory is expanded, showing several TypeScript declaration files, with 'my-lib.component.d.ts' selected and highlighted. The main editor area displays the content of 'my-lib.component.d.ts', which defines a class 'MyLibComponent' that implements the 'OnInit' interface from '@angular/core'. The code includes an import statement, an export declaration, a constructor, and an 'ngOnInit' method. A large red arrow is superimposed on the editor, pointing upwards towards the 'export declare' line. The Explorer sidebar also shows other files like 'my-lib.module.d.ts', 'my-lib.service.d.ts', 'my-lib.d.ts', 'my-lib.metadata.json', 'package.json', 'public_api.d.ts', 'e2e', 'node_modules', 'projects', 'src', '.editorconfig', '.gitignore', 'angular.json', 'package-lock.json', and 'package.json'.

```
1 import { OnInit } from '@angular/core';
2 export declare class MyLibComponent implements OnInit {
3     constructor();
4     ngOnInit(): void;
5 }
6
```