

## Single responsibility

Apply the [single responsibility principle \(SRP\)](#) to all components, services, and other symbols. This helps make the app cleaner, easier to read and maintain, and more testable.

### Rule of One

#### Style 01-01

Do define one thing, such as a service or component, per file.

Consider limiting files to 400 lines of code.

**Why?** One component per file makes it far easier to read, maintain, and avoid collisions with teams in source control.

**Why?** One component per file avoids hidden bugs that often arise when combining components in a file where they may share variables, create unwanted closures, or unwanted coupling with dependencies.

**Why?** A single component can be the default export for its file which facilitates lazy loading with the router.

The key is to make the code more reusable, easier to read, and less mistake prone.

The following *negative* example defines the AppComponent, bootstraps the app, defines the Hero model object, and loads heroes from the server all in the same file. *Don't do this.*

app/heroes/hero.component.ts

```
1. /* avoid */
2. import { Component, NgModule, OnInit } from '@angular/core';
3. import { BrowserModule } from '@angular/platform-browser';
4. import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';
5.
```



# Codelyzer (comes with Angular CLI)



README.md

npm package 4.5.0

downloads 33M


build passing

build passing

code style prettier

Conventional Commits 1.0.0

gitter join chat



## codelyzer

A set of tslint rules for static code analysis of Angular TypeScript projects.

You can run the static code analyzer over web apps, NativeScript, Ionic, etc.

Vote for your favorite feature [here](#). For more details about the feature request process see [this document](#)

introducing codelyzer

