Modulos

https://gustavodohara.com/blogangular/como-agregar-un-modulo-angular-en-tu-aplicacion/

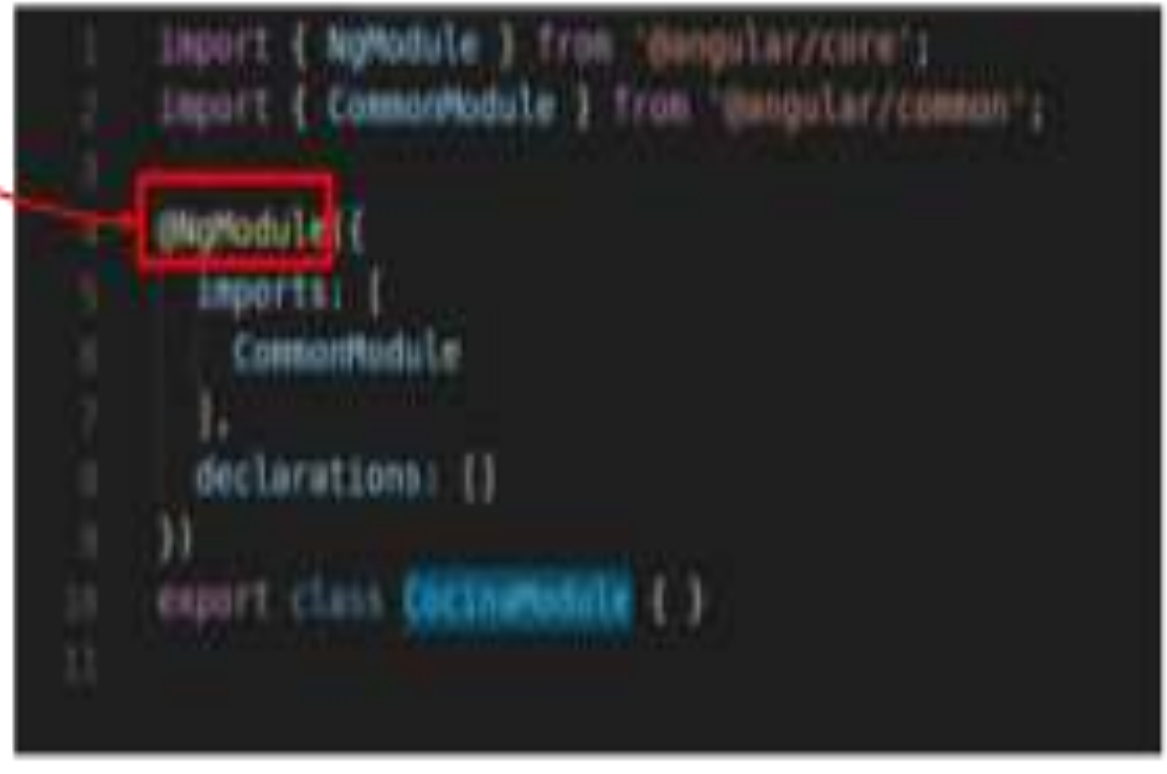
¿Qué es un módulo Angular?

Bueno, para arrancar, un módulo es un conjunto de código y de datos relacionados lógicamente, encapsula (o encierra) los detalles de cómo esta escrito el código; esto es para que aquel que usa el módulo no necesite saber «cómo esta hecho», sino «cómo se usa». El módulo expone una API pública, o sea, hace pública la forma en que deben «usarlo». Finalmente, combinando varios módulos se pueden crear aplicaciones mucho más grandes.

Entonces, ¿qué es un módulo Angular?

Es un módulo con esteroides de Angular, je. Yendo al grano, un módulo Angular es una clase declarada con la palabra «ngModule» precedida por un arroba (@)

Convierte CocinaModule en un módulo Angular



Un módulo Angular te permite re-agrupar tu código, re-ordenarlo en bloques de código con cierta lógica y agregarle funcionalidades de módulos externos (o de otros módulos que hayamos creado).

Tiene las siguientes ventajas y restricciones con respecto a los módulos Javascript (Para saber mas sobre módulos mira este post):

- En un módulo de Angular sólo se pueden usar *clases declarables* (o sea, un componente, directiva o pipe de Angular). ¿Que pasaría si importáramos algo que no es una

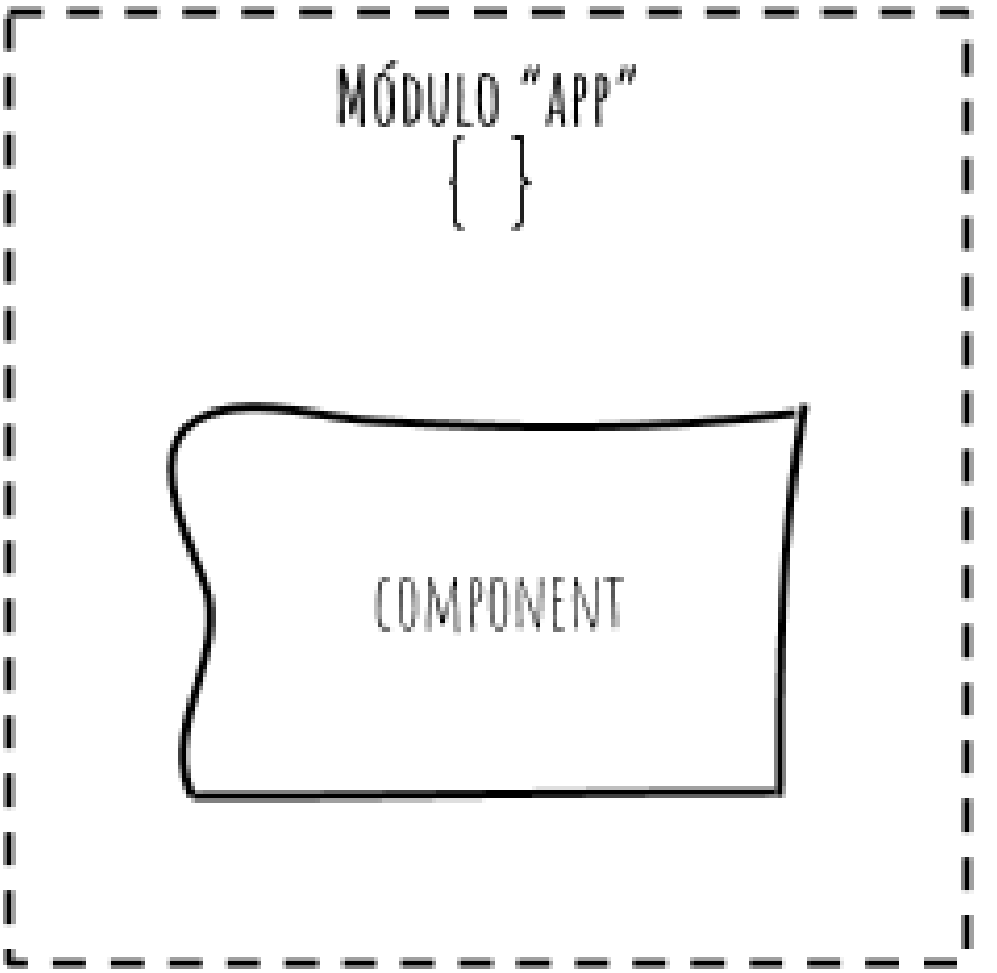
clase declarable? Pues que el código nos arrojaría un error

- En vez de agregar toooodas las clases de tu aplicación en un solo y enorme archivo, las podés agrupar por módulos Angular
- En un módulo Angular, sólo podés exportar (o compartir) con otros módulos Angular, *clases declarables* de tu propiedad o que te importaste (o trajiste de otro módulo)
- Se le puede agregar funcionalidades extra a TODA tu aplicación Angular agregando servicios Angular.

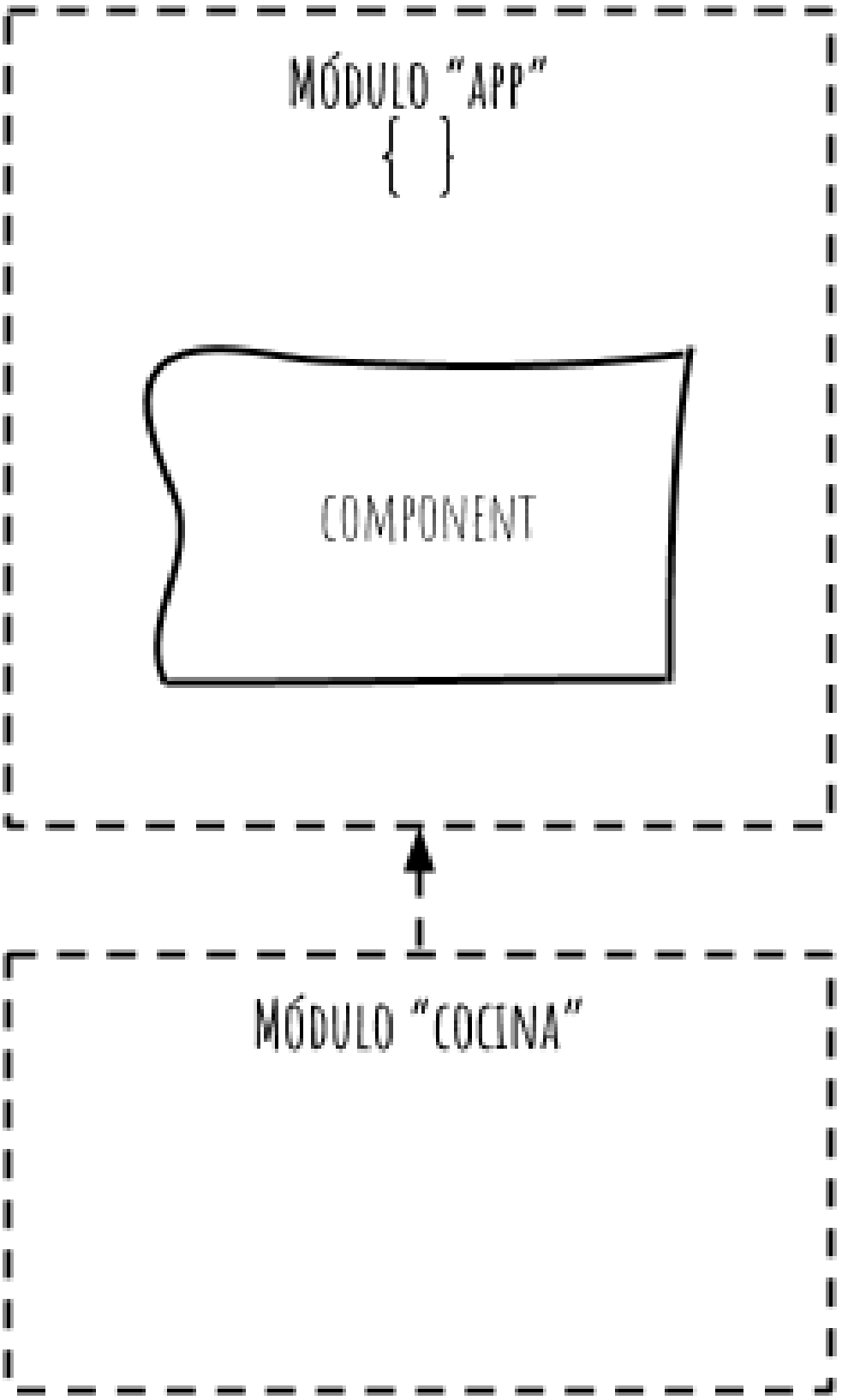
Una breve explicación

Lo primero que vamos a hacer es crear un proyecto de cero usando Angular-cli (si no sabés cómo hacerlo, mirá este post).
Cuando creamos un proyecto de esta forma ya se nos crea un módulo Angular, llamado «*App*» y, dentro de ese módulo, un componente llamado «a*pp*»

Tendríamos algo así:



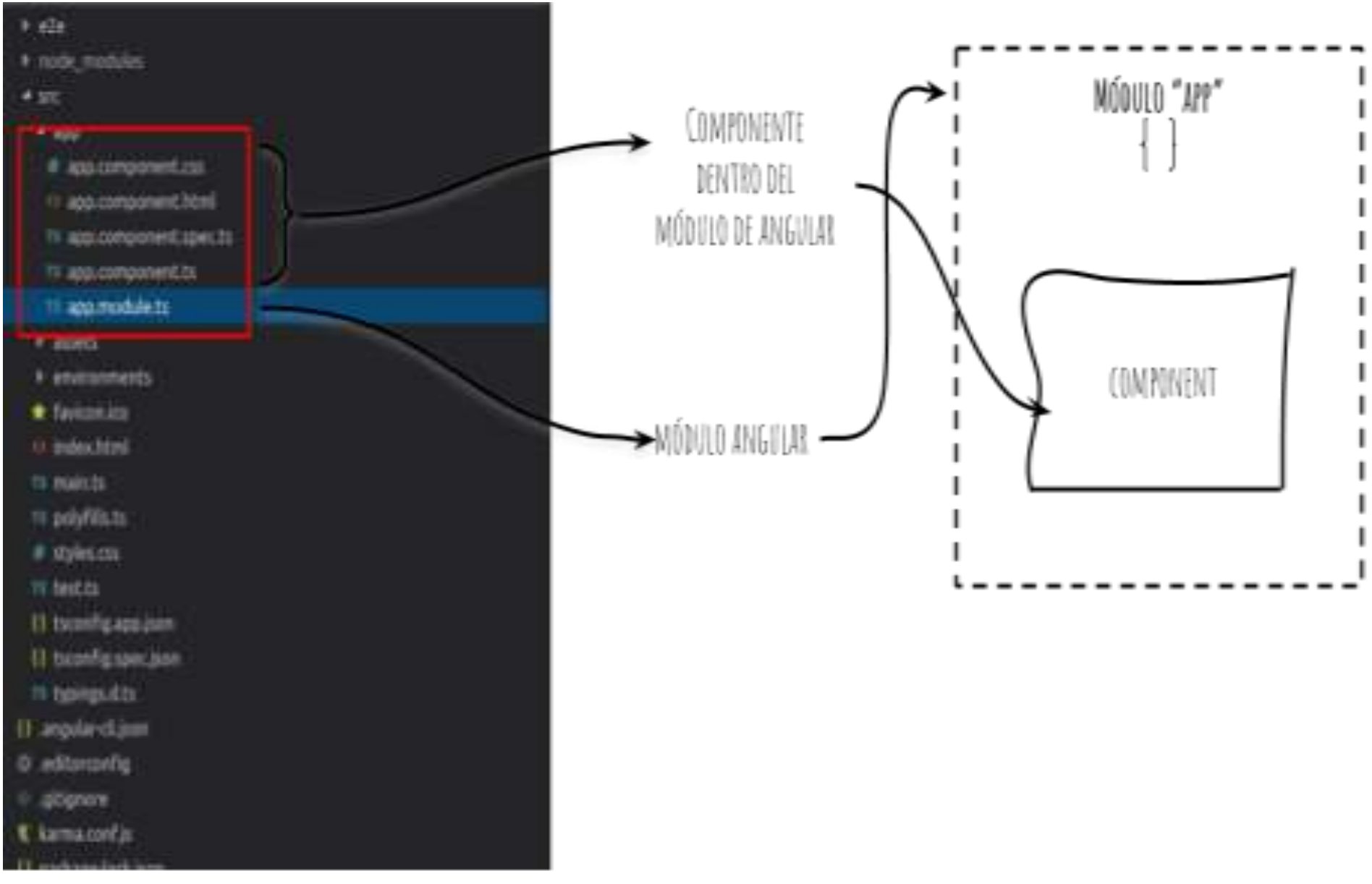
La idea es agregarle un segundo módulo Angular, que dependa del módulo que ya existe «a*pp*«. Lo vamos a llamar «Cocina»


MÓDULO "APP"
{ }
COMPONENT
MÓDULO "COCINA"

¡Manos a la Obra!

Lo primero que vamos a hacer es crear un proyecto de cero usando Angular-cli.

ng new mi-casa-inteligente


e2e
node_modules
src
app
app.component.css
app.component.html
app.component.spec.ts
app.component.ts
app.module.ts
assets
environments
favicon.ico
index.html
main.ts
polyfills.ts
styles.css
test.ts
tsconfig.app.json
tsconfig.spec.json
typings.d.ts
angular-cli.json
editorconfig
gitignore
karma.conf.js
COMPONENTE DENTRO DEL MÓDULO DE ANGULAR
MÓDULO ANGULAR
MÓDULO "APP"
{ }
COMPONENT

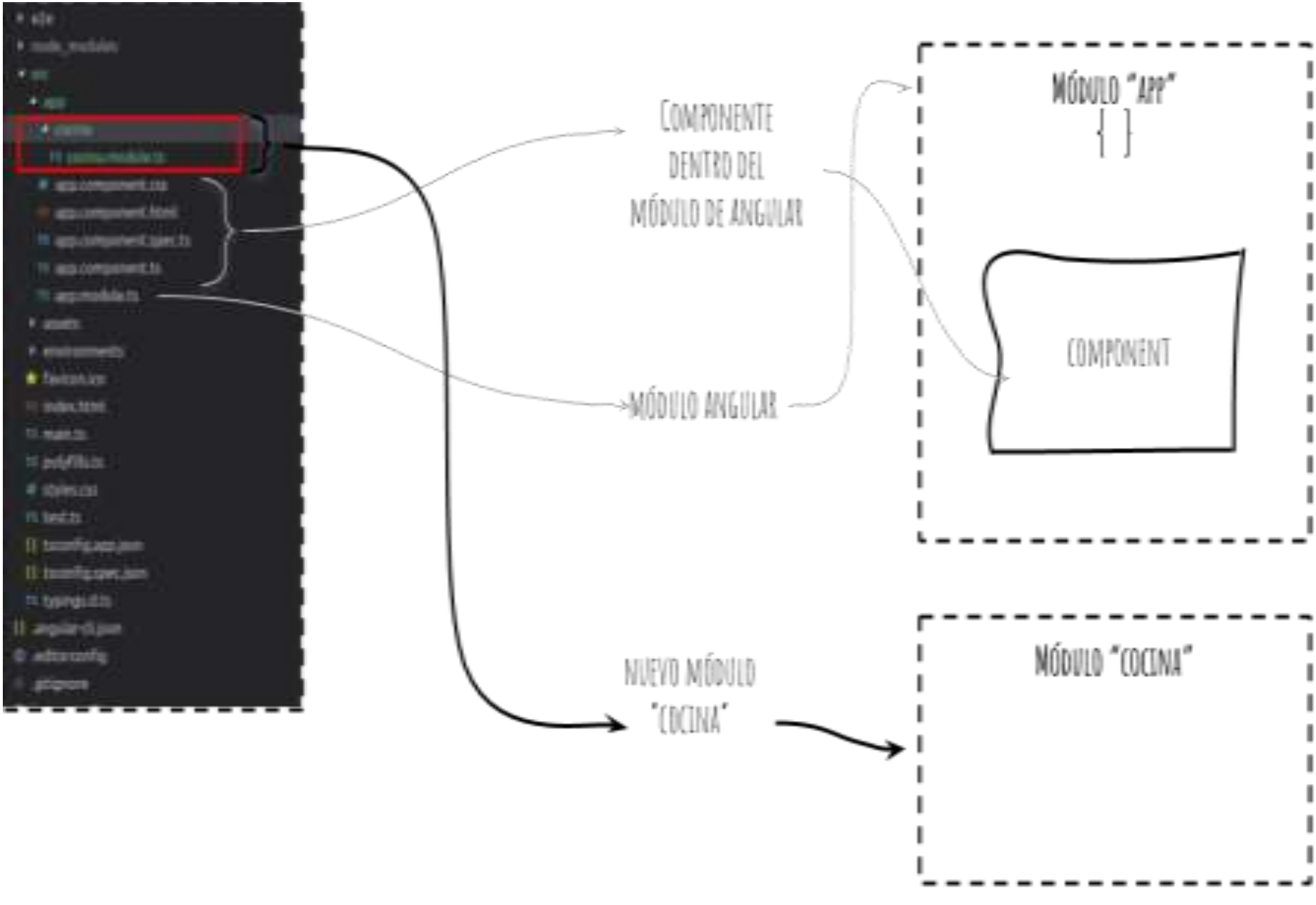
Como mencionamos antes, se nos crea un módulo «app» y dentro, su componente «app» (el componente sería la página web)

El siguiente paso es agregar el módulo «Cocina», esto lo hacemos con el siguiente comando:

ng g module cocina

nota: (este comando hay que hacerlo dentro del directorio «*mi-casa-inteligente*» que se crea cuando hacemos ng new mi-casa-inteligente)

Esto nos crea un directorio «cocina» junto con su módulo «cocina.module.ts»


Componente
dentro del
módulo de angular
Módulo angular
Módulo "app"
{ }
component
Nuevo módulo
"cocina"
Módulo "cocina"

Ahora veamos qué tiene el archivo «cocina.module.ts» que se creó en el paso anterior

```
1  import { NgModule } from '@angular/core';
2  import { CommonModule } from '@angular/common';
3
4  @NgModule({
5    imports: [
6      CommonModule
7    ],
8    declarations: []
9  })
10 export class CocinaModule { }
11
```

Primero, vemos que el módulo Angular es una clase, como mencionamos antes, llamada «CocinaModule»

Después, algo interesante, la clase «CocinaModule» es «decorada» con la palabra NgModule (o sea, se agrega «@» seguido de NgModule). Esto le indica a Angular que ésta clase «CocinaModule«, es un módulo Angular.

Por último, vemos que en el módulo Angular se importa el módulo «CommonModule» y que, para usar tanto los módulos «NgModule» como «CommonModule» (que son propias de Angular), hay que importarlos usando la

forma de TypeScript (para más detalle, ver el siguiente post)

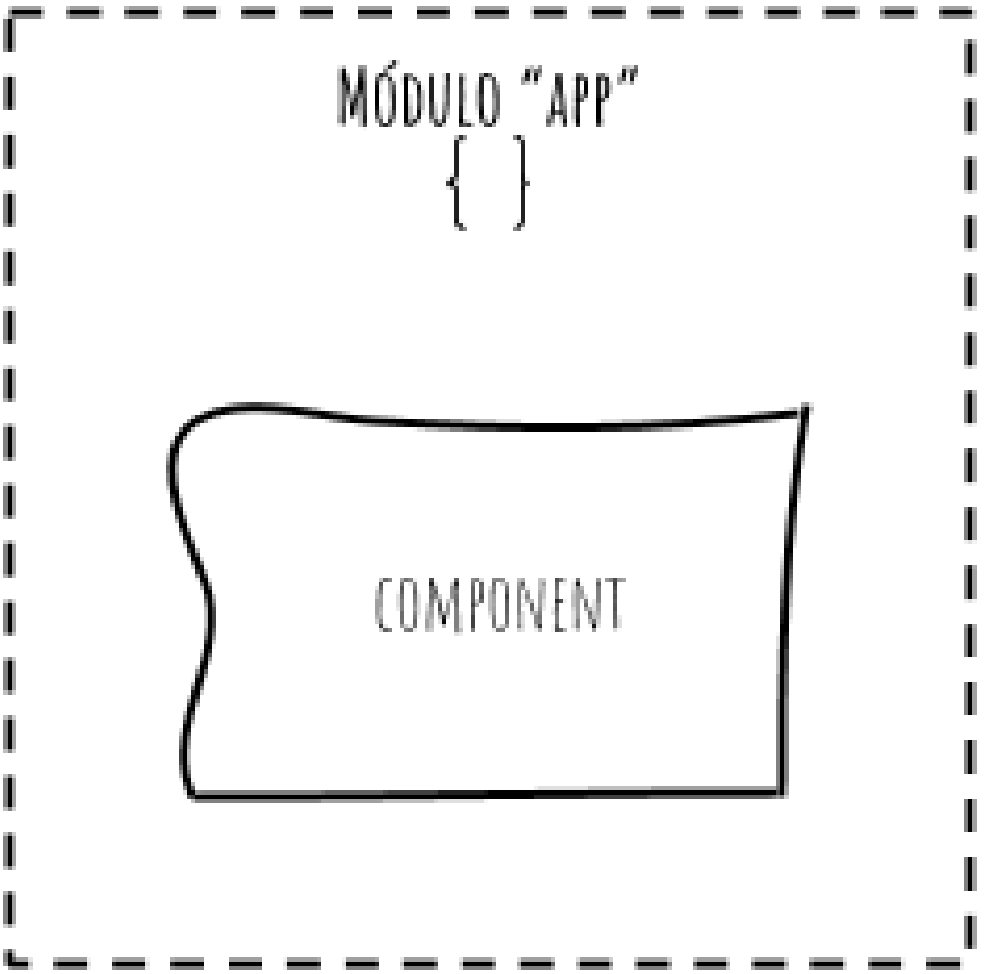
Imports de módulos de Angular

Imports de TypeScript (porque TypeScript también modulariza a su manera)

Como dijimos antes, un módulo de Angular, es una clase

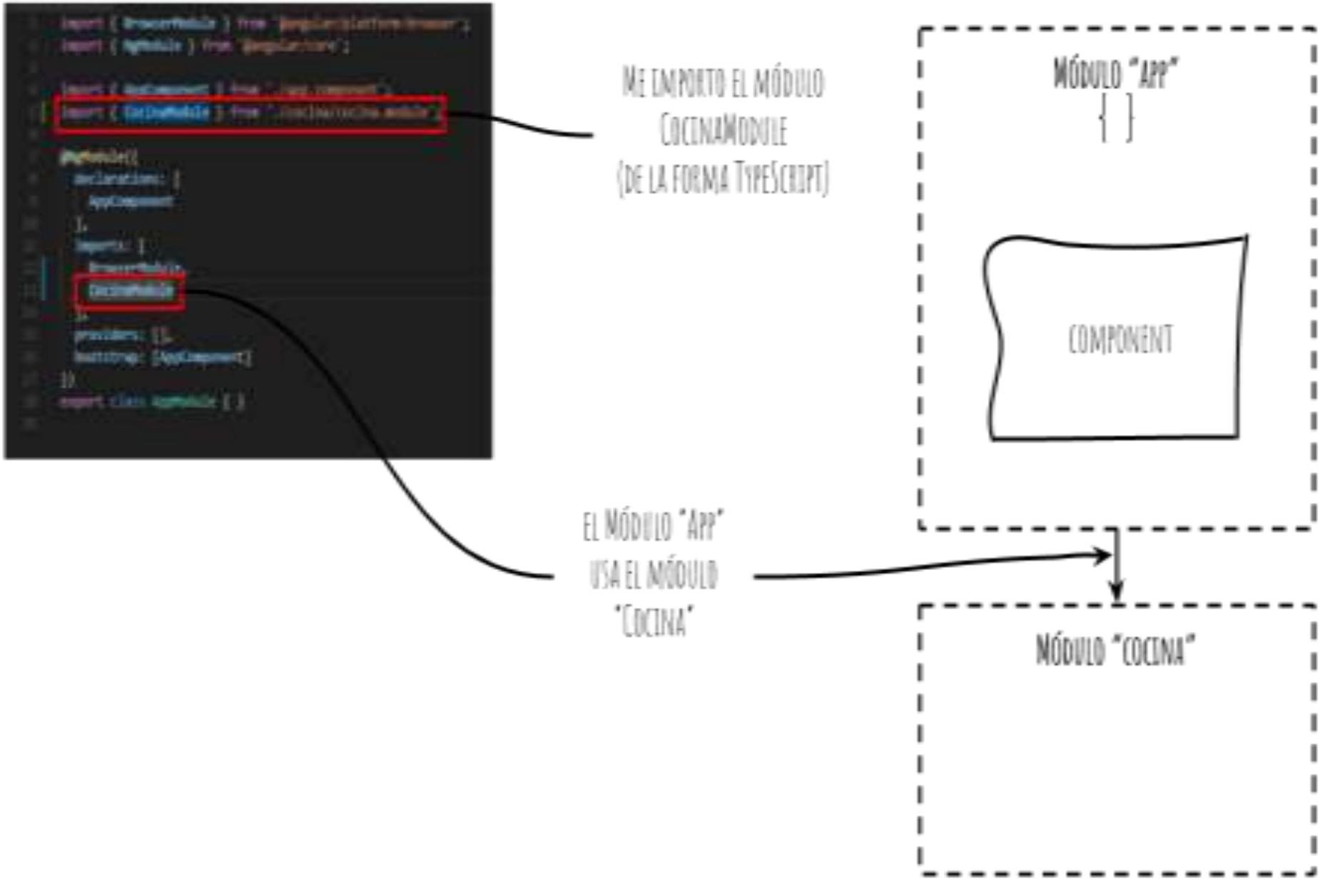
Y ahora, ¿cómo uno la cocina con el resto de la casa?

¡Acá falta algo! ¡la cocina no está dentro de la casa! Efectivamente, el módulo «*Cocina*» todavía no está linkeado a la aplicación, sólo está creado el módulo, pero volando en el aire. Sería algo así:


MÓDULO "APP"
{ }
COMPONENT


MÓDULO "COCINA"

Ahora, lo que hacemos es agregar el módulo «*Cocina*» al módulo «*app*» (en el archivo «*app.module.ts*» del proyecto que se creó con el comando «*ng new mi-casa-inteligente*«). Para ésto, a la lista de «*imports*«, que está dentro de nuestro módulo Angular, se agrega el módulo «*CocinaModule*«, que creamos en el paso anterior


Me importo el módulo
CocinaModule
(de la forma TypeScript)
Módulo "app"
{ }
component
El módulo "App"
usa el módulo
"Cocina"
Módulo "cocina"

Y así conectamos los dos módulos y ¡ampliamos nuestra casa!

Conclusiones

Si bien hay muchas cosas mas que se pueden hacer con los módulos de Angular, esto sería lo necesario para arrancar. En otros posts, explicaré el resto de las opciones para hacer nuestra «casa» todavía mas inteligente ;=)