



# Angular

## 3.2. Property Binding

# 1. Property binding

- Podemos establecer relación entre elementos HTML y el código TypeScript.
- Vamos a generar una aplicación nueva y sobre esa aplicación vamos a introducir un botón sobre el que haremos el property binding.

# 1. Property binding

- Introducimos el comando (desde la terminal)

**ng new app3 --standalone=false**

```
Angular: ng new app3 --standalone=false
? Which stylesheet format would you like to use? CSS
? Do you want to enable Server-Side Rendering (SSR) and Static Site Generation (SSG/Prerendering)? No
CREATE app3/angular.json (2835 bytes)
CREATE app3/package.json (1035 bytes)
CREATE app3/README.md (1058 bytes)
CREATE app3/tsconfig.json (903 bytes)
CREATE app3/.editorconfig (274 bytes)
CREATE app3/.gitignore (548 bytes)
CREATE app3/tsconfig.app.json (263 bytes)
CREATE app3/tsconfig.spec.json (273 bytes)
CREATE app3/.vscode/extensions.json (130 bytes)
CREATE app3/.vscode/launch.json (470 bytes)
CREATE app3/.vscode/tasks.json (938 bytes)
CREATE app3/src/main.ts (214 bytes)
CREATE app3/src/favicon.ico (15086 bytes)
```

# 1. Property binding

- Instalamos bootstrap (para más sencillez en código CSS)

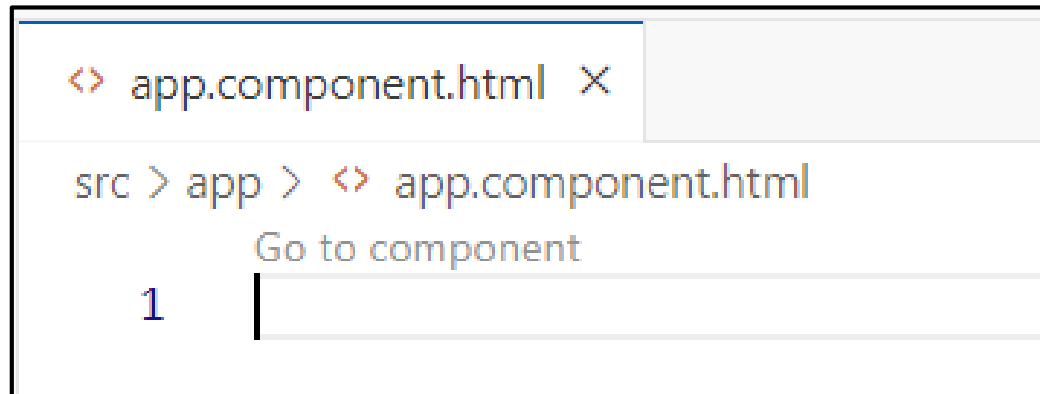
```
Angular/app3>npm i --save bootstrap
added 2 packages, and audited 988 packages in 4s
121 packages are looking for funding
  run `npm fund` for details
4 moderate severity vulnerabilities
To address all issues (including breaking changes), run:
  npm audit fix --force
Run `npm audit` for details.
```

# 1. Property binding

```
1  "styles": [  
2    "src/styles.css",  
3    "node_modules/bootstrap/dist/css/bootstrap.min.css"  
4  ],  
5  "scripts": [  
6    "node_modules/bootstrap/dist/js/bootstrap.min.js"  
7  ]  
8}
```

# 1. Property binding

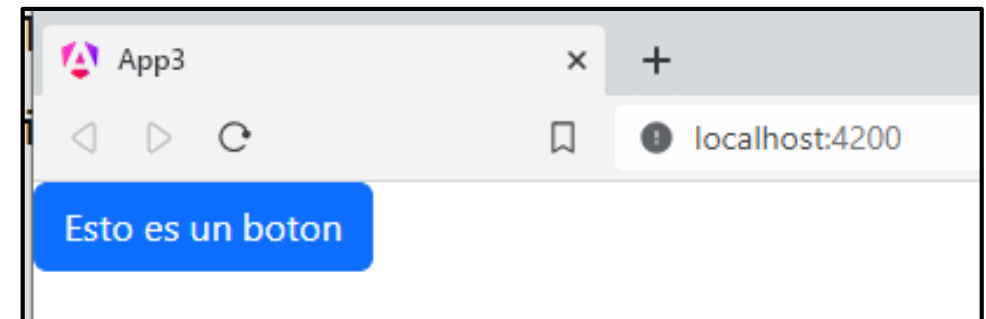
- Borramos el contenido de app.component.html



# 1. Property binding

- Vamos a crear un botón, y vamos a modificar las propiedades del botón desde TS.
- En la parte HTML del componente añadimos el botón

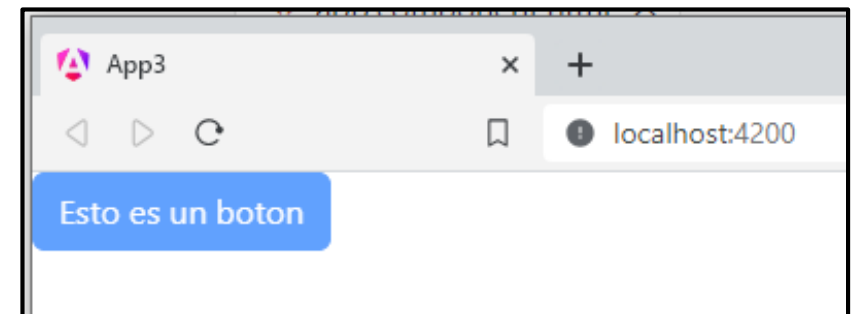
```
<> app.component.html ×  
src > app > <> app.component.html > button.btn.btn-primary  
Go to component  
1 <button class="btn btn-primary">Esto es un boton</button>
```



# 1. Property binding

- Podríamos deshabilitar este botón con el modificador **disabled**

```
<> app.component.html x
src > app > <> app.component.html > button.btn.btn-primary
Go to component
1 <button class="btn btn-primary" disabled="disabled ">Esto es un
  boton</button>
```





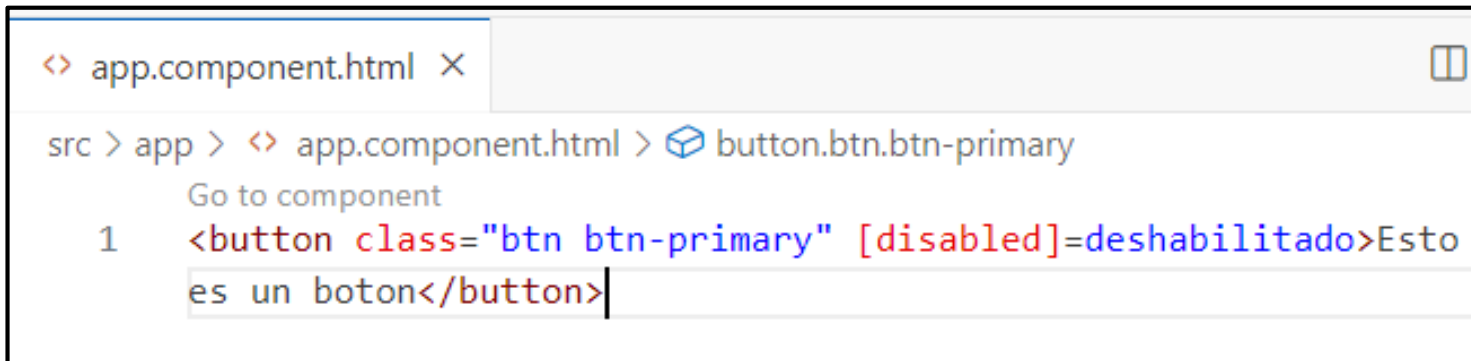
# 1. Property binding

- Podemos hacer que ese botón dependa de una condición en TypeScript.
- Podemos establecer un atributo deshabilitado que estará a **true**

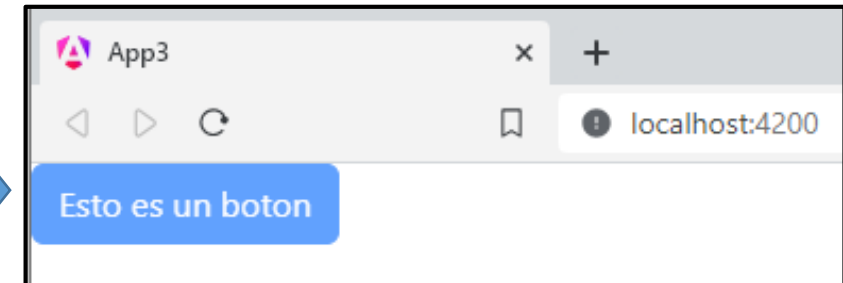
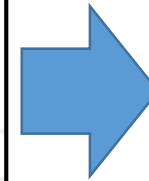
```
export class AppComponent {  
  title = 'app3';  
  
  deshabilitado=true;  
}
```

# 1. Property binding

- Podemos ahora vincular la propiedad del html con [] al código TS.



```
<> app.component.html ×  
src > app > <> app.component.html > button.btn.btn-primary  
Go to component  
1 <button class="btn btn-primary" [disabled]=deshabilitado>Esto  
  es un boton</button>
```



# 1. Property binding

- Si cambiamos a false, el botón vuelve a estar activo

```
export class AppComponent {  
  title = 'app3';  
  
  deshabilitado=false;  
}
```

