



Angular

3.5. Referencias locales

1. Caso de partida

- Partimos de un formulario sencillo, realizado con two way data binding en el que tenemos un campo nombre y otro apellido de tipo string y un botón para agregar.
- Al hacer clic sobre el botón creará un objeto y lo mostraremos con console.log

1. Caso de partida

app.module.ts

```
import { AppComponent } from './app.component';  
import { FormsModule } from '@angular/forms';
```

```
@NgModule({  
  declarations: [  
    AppComponent  
  ],  
  imports: [  
    BrowserModule,  
    AppRoutingModule,  
    FormsModule  
  ],  
})
```

1. Caso de partida

app.component.html

```
<div class="container">

  <label class="col-md-2 control-label">Nombre</label>
  <input type="text" name="nombre" id="nombre" [(ngModel)]
    ="nombre" class="form-control">

  <label class="col-md-2 control-label">Email</label>
  <input type="email" name="email" id="email" [(ngModel)]="email"
    class="form-control">

  <button (click)="crearPersona()" class="btn btn-primary">Crear
    Persona</button>

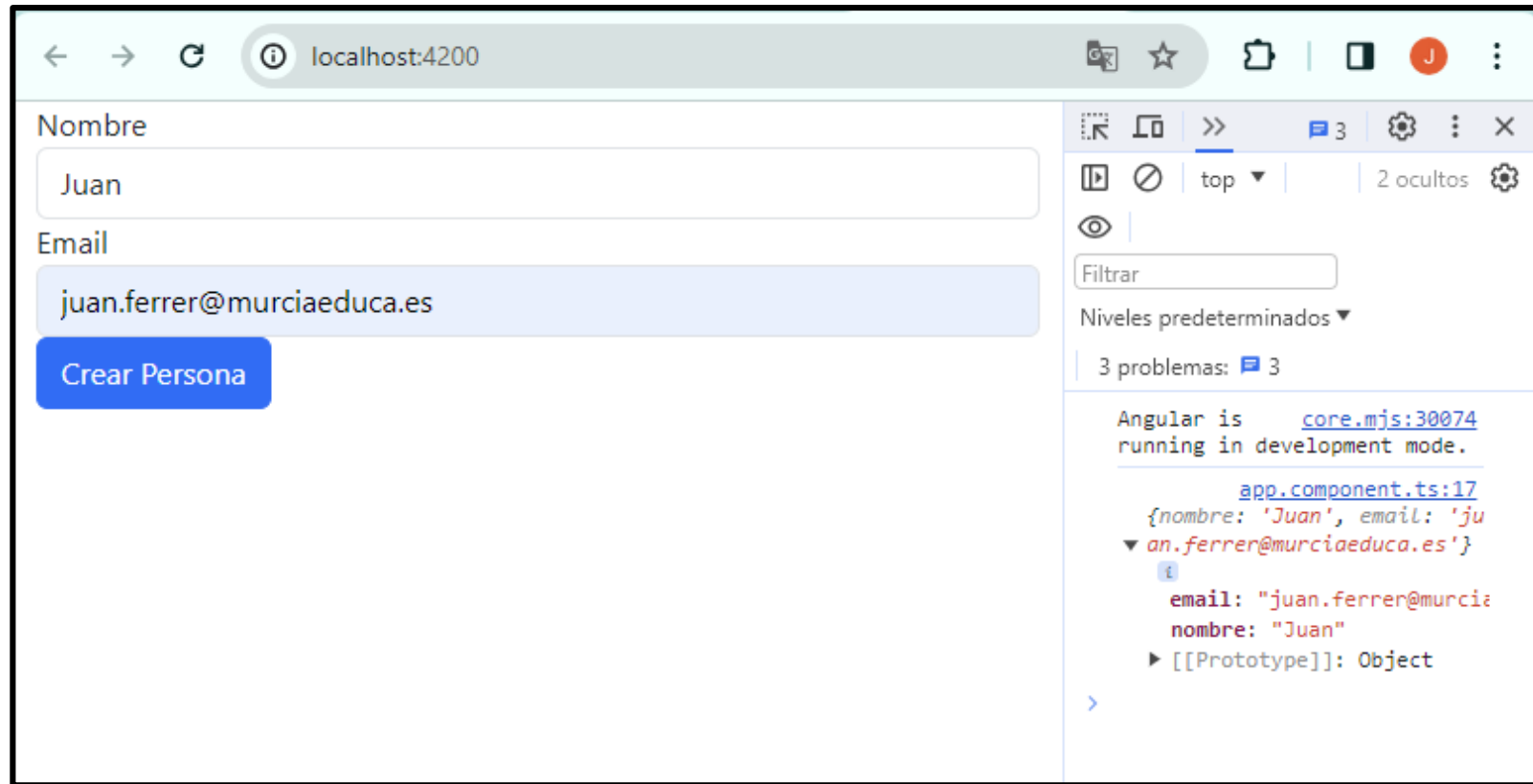
</div>
```

1. Caso de partida

app.component.ts

```
export class AppComponent {  
  nombre="";  
  email="";  
  
  crearPersona() {  
    const obj={  
      nombre:this.nombre,  
      email:this.email  
    }  
    console.log(obj);  
  }  
}
```

1. Caso de partida



2. Modificando el componente

- Si los datos de los campos input solo los necesitamos para procesarlos en un método y no interactuamos más podemos evitar cargar FormsModule en app.module.ts y el uso de two way data binding.
- Para ello eliminamos la etiqueta ngModule y la sustituimos por un identificador del campo con #

2. Modificando el componente

app.component.html

```
Go to component
<div class="container">

  <label class="col-md-2 control-label">Nombre</label>
  <input type="text" name="nombre" id="nombre" #nombre
  class="form-control">

  <label class="col-md-2 control-label">Email</label>
  <input type="email" name="email" id="email" #email
  class="form-control">

  <button (click)="crearPersona(nombre,email)" class="btn
  btn-primary">Crear Persona</button>

</div>
```


2. Modificando el componente

app.component.ts

```
export class AppComponent {  
  crearPersona(nombreDOM:HTMLInputElement,emailDOM:HTMLInputElement) {  
    const obj={  
      nombre:nombreDOM.value,  
      email:emailDOM.value  
    }  
    console.log(obj);  
  }  
}
```

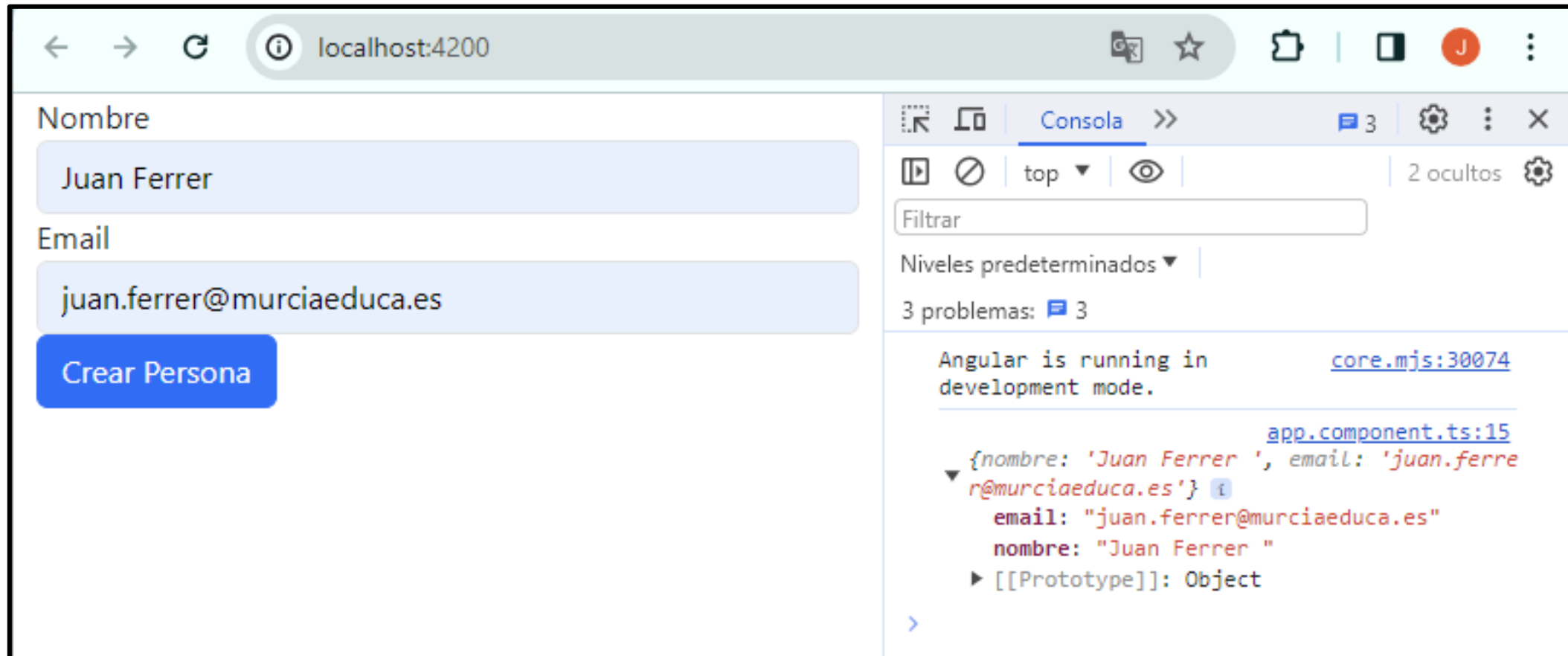
2. Modificando el componente

app.module.ts

```
import { AppComponent } from './app.component';

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
  ],
})
```

2. Modificando el componente



The screenshot displays a web browser window at `localhost:4200` and the Angular DevTools console. The browser window shows a form with two input fields: "Nombre" (Name) containing "Juan Ferrer" and "Email" containing "juan.ferrer@murciaeduca.es". Below the inputs is a blue button labeled "Crear Persona". The Angular DevTools console is open, showing the "Consola" tab. It displays a message: "Angular is running in development mode." followed by a log entry from `app.component.ts:15`. The log entry is an object: `{nombre: 'Juan Ferrer ', email: 'juan.ferrer@murciaeduca.es'}`. The console also shows "3 problemas" (3 problems) and "2 ocultos" (2 hidden).

Nombre

Juan Ferrer

Email

juan.ferrer@murciaeduca.es

Crear Persona

Consola

3

Filtrar

Niveles predeterminados

3 problemas: 3

Angular is running in development mode. [core.mjs:30074](#)

[app.component.ts:15](#)

```
{nombre: 'Juan Ferrer ', email: 'juan.ferrer@murciaeduca.es'}
```

email: "juan.ferrer@murciaeduca.es"

nombre: "Juan Ferrer "

[[Prototype]]: Object