

UNIVERSITY

everis

Training

everis

an **NTT DATA** Company

TÉCNICAS DE DESARROLLO DE INTERFACES DE USUARIO (FRONT END) PARA APLICACIONES DE SOFTWARE

Unidad 2: Introducción a la Programación en Angular

TÉCNICAS DE DESARROLLO DE INTERFACES DE USUARIO (FRONT END) PARA APLICACIONES DE SOFTWARE

Unidad 2: Introducción a la Programación en Angular

2.2 Traspaso de Información a Componentes

CONTENIDOS: Traspaso de Información a Componentes

- Enlaces
- Controlador Angular
- Data Binding

OBJETIVO DE APRENDIZAJE

Unidad 2: Introducción a la Programación en Angular 2.2 Traspaso de Información a Componentes

Aplicar técnicas básicas de desarrollo **Front End**, considerando los estándares definidos para el sector.

TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

- La vista tiene acceso al modelo y hay varias formas de mostrar los datos del modelo en la vista.
- Puede usar la directiva **ng-bind**, que vinculará el innerHTML del elemento a la propiedad del modelo especificada

TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

Ingrese su nombre

```
<input type="text"
```

```
  ng-model="nombre">
```

```
<div>Echo: {{ apellido }} </div>
```

```
<div>Echo: {{ ::nombreCompleto }} </div>
```

Enlace bidireccional

Enlace unidireccional

Unión 1 - vez

TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

Enlace bidireccional (ng-model):

- Oyente para el cambio en la entrada configurado automáticamente por Angular actualiza el valor de la propiedad en \$scope.
- La actualización directa del valor de la propiedad se actualiza automáticamente en la interfaz de usuario.

Enlace unidireccional ({{prop}}):

- La actualización directa del valor de la propiedad se actualiza automáticamente en la interfaz de usuario.

Enlace de 1 vez ({{:: prop}}):

- El valor inicializado de prop se actualiza automáticamente en la interfaz de usuario.
- Se elimina el observador de accesorios, por lo que la interfaz de usuario nunca se actualiza.

TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

```
<body>
  <div ng-app="aplicacion" ng-controller="controlador">
    <p ng-bind="nombre"></p>
  </div>
</script>

var variable = angular.module('aplicacion', []);
variable.controller('controlador', function($scope) {
  $scope.nombre = "Pedro";
});
</script>
</body>
```

vinculará el innerHTML del elemento a la propiedad del modelo especificada

ng-bind, vinculará el innerHTML del elemento a la propiedad del modelo especificada

TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

Utilizando las llaves dobles para desplegar el contenido desde el modelo.

```
<body>
<div ng-app="aplicacion" ng-controller="controlador">
| | <p>Nombre: {{nombre}}</p>
</div>

<script>
var variable = angular.module('aplicacion', []);
variable.controller('controlador', function($scope) {
| | $scope.nombre = "Pedro";
});
</script>

</body>
```



Nombre: Pedro

TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

La Directiva ng-model

Utilice la directiva ng-model para vincular datos del modelo a la vista en controles HTML (entrada, selección, área de texto)

```
<body>

<div ng-app="aplicacion" ng-controller="controlador">
|   <input ng-model="nombre">
</div>

<script>
var app = angular.module('aplicacion', []);
app.controller('controlador', function($scope) {
|   $scope.nombre="Pedro";
});
</script>

</body>
```



TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

La Directiva ng-model

- La directiva ng-model proporciona un enlace bidireccional entre el modelo y la vista.
- El enlace de datos en AngularJS es la sincronización entre el modelo y la vista.
- Cuando los datos en el modelo cambian, la vista refleja el cambio y cuando los datos en la vista cambian, el modelo también se actualiza. Esto ocurre de forma inmediata y automática, lo que asegura que el modelo y la vista estén actualizados en todo momento.

TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

Enlace de datos bidireccional

```
<body>
<p>Ingrese su nombre: </p>

<div ng-app="aplicacion" ng-controller="controlador">
  Nombre: <input ng-model="nombre">
  <h1>{{nombre}}</h1>
</div>

<script>
var variable = angular.module('aplicacion', []);
variable.controller('controlador', function($scope) {
  $scope.nombre = "Pedro";
});
</script>
</body>
```



Ingrese su nombre:

Nombre:

Pedro

TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

Controlador Angular

- Las aplicaciones en AngularJS están controladas por controladores.
- Debido a la sincronización inmediata del modelo y la vista, el controlador puede separarse completamente de la vista y simplemente concentrarse en los datos del modelo.
- Gracias al enlace de datos en AngularJS, la vista reflejará cualquier cambio realizado en el controlador.

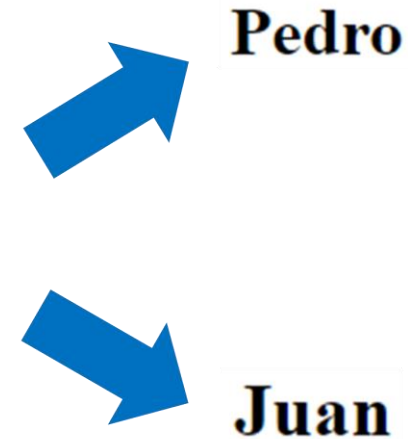
TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

Enlace de datos bidireccional

```
<body>
<div ng-app="aplicacion" ng-controller="controlador">
  <h1 ng-click="cambiarNombre()">{{nombre}}</h1>
</div>

<script>
var variable = angular.module('aplicacion', []);
variable.controller('controlador', function($scope) {
  $scope.nombre = "Pedro";
  $scope.cambiarNombre = function() {
    $scope.nombre = "Juan";
  }
});
</script>
</body>
```



TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

Enlace de datos bidireccional

El método `angular.module` toma 2 argumentos para crear un módulo:

- Nombre del módulo.
- Matriz de dependencias de nombres de módulos.

El método `angular.module` con solo el nombre del módulo recupera el módulo creado previamente.

- Luego, puede declarar componentes, controladores, etc., en él.

El método `module.config` se activa antes que el método `module.run`

Todos los módulos de dependencia se configuran primero.

No importa qué módulos se enumeran primero siempre que las declaraciones del módulo se enumeren antes que las declaraciones de artefactos en ese módulo.

TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

Data Binding

Primero generaremos un componente llamado persona.

```
PS C:\> ng new databinding
Do you want to enforce stricter type checking and stricter bundle budgets in the workspace?
This setting helps improve maintainability and catch bugs ahead of time.
For more information, see https://angular.io/strict Yes
Would you like to add Angular routing? Yes
Which stylesheet format would you like to use? CSS
CREATE databinding/angular.json (3663 bytes)
CREATE databinding/package.json (1201 bytes)
CREATE databinding/README.md (1020 bytes)
CREATE databinding/tsconfig.json (783 bytes)
CREATE databinding/tslint.json (3185 bytes)
CREATE databinding/.editorconfig (274 bytes)
CREATE databinding/.gitignore (631 bytes)
CREATE databinding/.browserslistrc (703 bytes)
CREATE databinding/karma.conf.js (1428 bytes)
CREATE databinding/tsconfig.app.json (287 bytes)
CREATE databinding/tsconfig.spec.json (333 bytes)
CREATE databinding/src/favicon.ico (948 bytes)
CREATE databinding/src/index.html (297 bytes)
CREATE databinding/src/main.ts (372 bytes)
CREATE databinding/src/polyfills.ts (2830 bytes)
CREATE databinding/src/styles.css (80 bytes)
CREATE databinding/src/test.ts (753 bytes)
CREATE databinding/src/assets/.gitkeep (0 bytes)
CREATE databinding/src/environments/environment.prod.ts (51 bytes)
CREATE databinding/src/environments/environment.ts (662 bytes)
CREATE databinding/src/app/app-routing.module.ts (245 bytes)
CREATE databinding/src/app/app.module.ts (393 bytes)
CREATE databinding/src/app/app.component.html (25757 bytes)
CREATE databinding/src/app/app.component.spec.ts (1072 bytes)
CREATE databinding/src/app/app.component.ts (215 bytes)
CREATE databinding/src/app/app.component.css (0 bytes)
CREATE databinding/e2e/protractor.conf.js (904 bytes)
CREATE databinding/e2e/tsconfig.json (274 bytes)
CREATE databinding/e2e/src/app.e2e-spec.ts (662 bytes)
CREATE databinding/e2e/src/app.po.ts (274 bytes)
Installing packages (npm)...
```

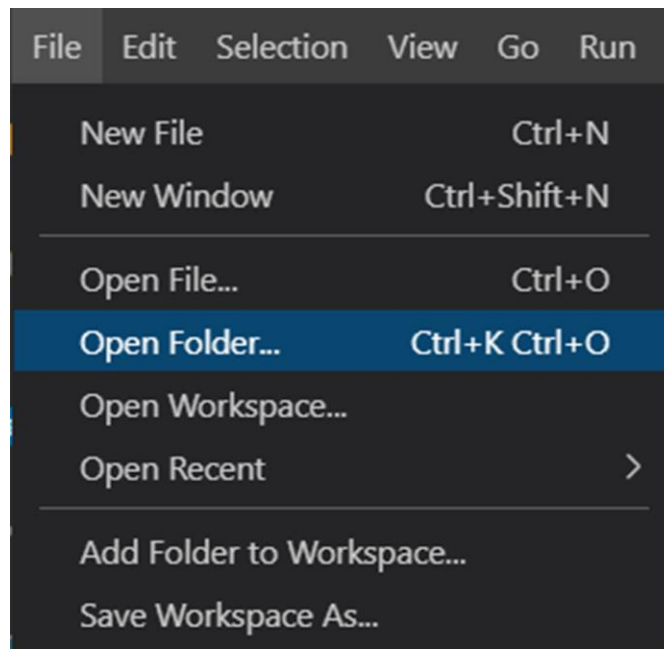
```
Administrador: Windows PowerShell
PS C:\databinding> ng g component persona
CREATE src/app/persona/persona.component.html (22 bytes)
CREATE src/app/persona/persona.component.spec.ts (633 bytes)
CREATE src/app/persona/persona.component.ts (279 bytes)
CREATE src/app/persona/persona.component.css (0 bytes)
UPDATE src/app/app.module.ts (479 bytes)
PS C:\databinding>
```

TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

Data Binding

Creando un nuevo proyecto llamado databinding y abriendo la carpeta.



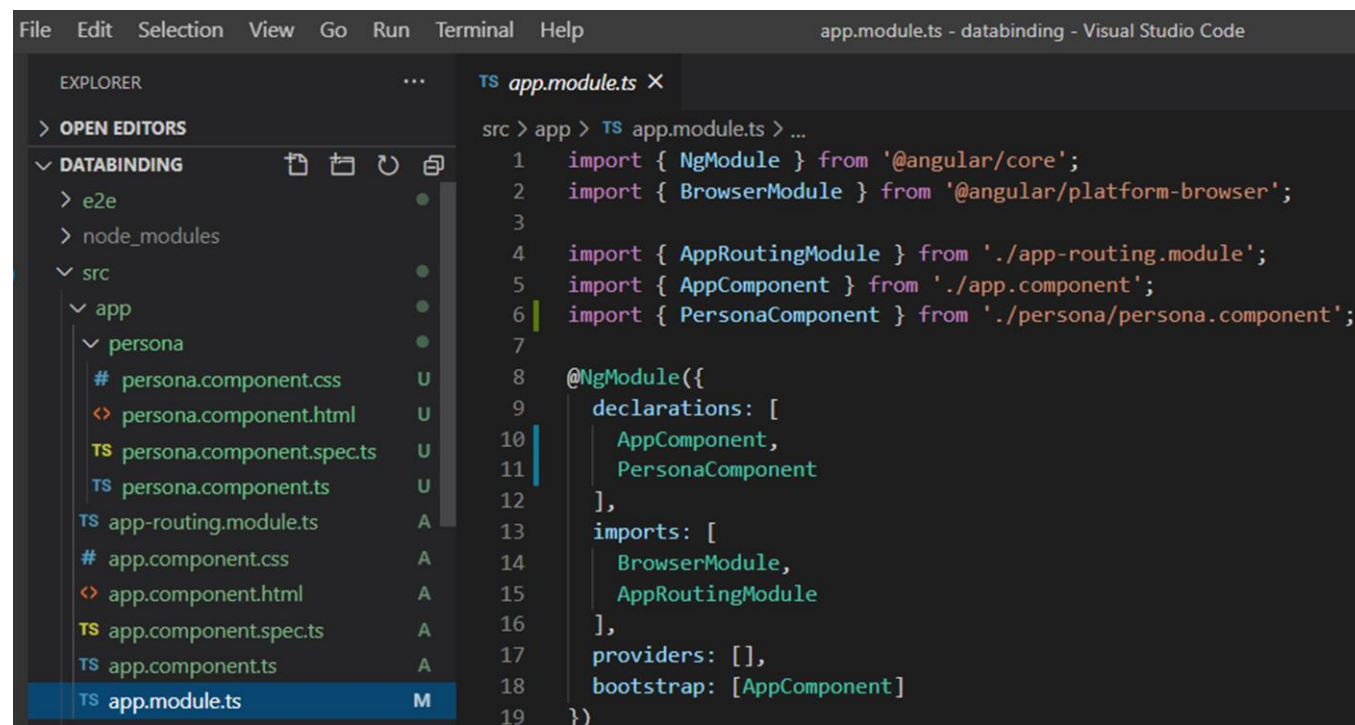
Nombre	Fecha de modificación	Tipo	Tamaño
Archivos de programa	30-01-2021 16:33	Carpeta de archivos	
Archivos de programa (x86)	30-01-2021 17:02	Carpeta de archivos	
databinding	30-01-2021 19:24	Carpeta de archivos	
eclipse	07-10-2020 17:18	Carpeta de archivos	

TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

Data Binding

Fichero app.modules.ts del proyecto databinding.



```
File Edit Selection View Go Run Terminal Help
app.module.ts - databinding - Visual Studio Code

EXPLORER
> OPEN EDITORS
DATABINDING
  > e2e
  > node_modules
  > src
    > app
      > persona
        # persona.component.css
        <> persona.component.html
        TS persona.component.spec.ts
        TS persona.component.ts
      TS app-routing.module.ts
      # app.component.css
      <> app.component.html
      TS app.component.spec.ts
      TS app.component.ts
      TS app.module.ts M

TS app.module.ts X
src > app > TS app.module.ts > ...
1 import { NgModule } from '@angular/core';
2 import { BrowserModule } from '@angular/platform-browser';
3
4 import { AppRoutingModule } from './app-routing.module';
5 import { AppComponent } from './app.component';
6 import { PersonaComponent } from './persona/persona.component';
7
8 @NgModule({
9   declarations: [
10     AppComponent,
11     PersonaComponent
12   ],
13   imports: [
14     BrowserModule,
15     AppRoutingModule
16   ],
17   providers: [],
18   bootstrap: [AppComponent]
19 })
```

TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

Data Binding

Agregando el arreglo personas.

```
EXPLORER
...
> OPEN EDITORS 1 UNSAVED
▼ DATABINDING
  > e2e
  > node_modules
  ▼ src
    ▼ app
      ▼ persona
        # persona.component.css U
        <> persona.component.html U
        TS persona.component.spec.ts U
        TS persona.component.ts U
      TS app-routing.module.ts A
      # app.component.css A
      <> app.component.html A
      TS app.component.spec.ts A

TS persona.component.ts
src > app > persona > TS persona.component.ts > ...
1  import { Component, OnInit } from '@angular/core';
2
3  @Component({
4    selector: 'app-persona',
5    templateUrl: './persona.component.html',
6    styleUrls: ['./persona.component.css']
7  })
8  export class PersonaComponent implements OnInit {
9
10   //Lista de personas
11   personas = ['Pedro', 'Juan', 'Diego', 'Carolina', 'Maria'];
12
13   ngOnInit(): void {
14   }
15
16 }
```

TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

Data Binding

Agregando la etiqueta `<app-persona></app-persona>`

The screenshot shows an IDE with a dark theme. The Explorer panel on the left is expanded to show the 'DATABINDING' section. The file 'app.component.html' is selected and highlighted in blue. The main editor area shows the 'app.component.html' file with the following content:

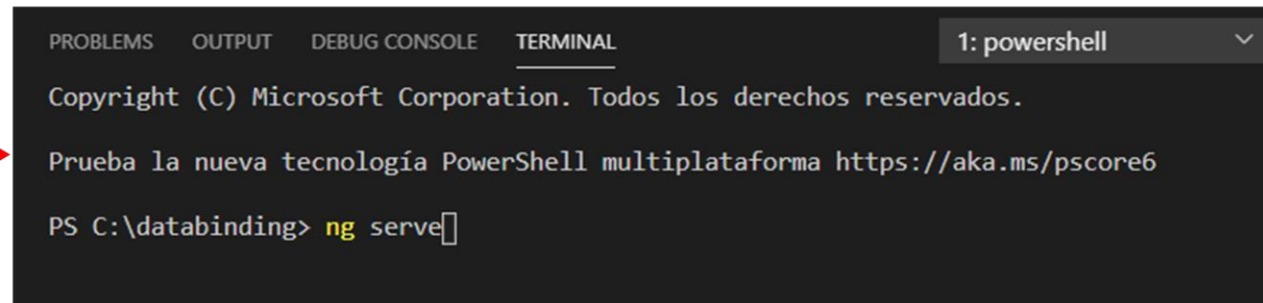
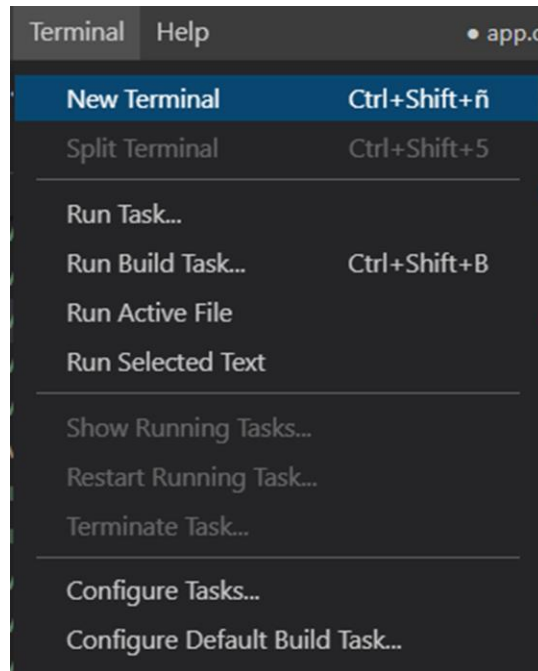
```
src > app > <> app.component.html > ...  
1 | <app-persona></app-persona>  
2 |  
3 |
```

TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

Data Binding

Compilando e iniciando el servidor.



TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

Data Binding
Resultado.



TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

Data Binding

Agregando un botón que activará un método.

```
File Edit Selection View Go Run Terminal Help
• persona.component.html - databinding - Visual Studio Code

EXPLORER
> OPEN EDITORS 1 UNSAVED
▼ DATABINDING
  > e2e
  > node_modules
  ▼ src
    ▼ app
      ▼ persona
        # persona.component.css U
        <> persona.component.html U
        TS persona.component.spec.ts U
        TS persona.component.ts U
      TS app-routing.module.ts A
      # app.component.css A
      <> app.component.html M
      TS app.component.spec.ts A

TS persona.component.ts
<> app.component.html
<> persona.component.html •

src > app > persona > <> persona.component.html > ...
1  <p>persona works!</p>
2
3  <h2>Seleccione una persona</h2>
4  <div>
5      <select>
6          <option *ngFor="let per of personas">{{per}}</option>
7      </select>
8  </div>
9
10 <h2>Unión de eventos</h2>
11 <button (click)="clickMethod($event)">
12     Botón presionado
13 </button>
14
15
16
```

TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

Data Binding

Agregando el método clickMethod().

The screenshot shows the Visual Studio Code interface with the Explorer on the left and the Code Editor on the right. The Explorer shows the project structure with the file `persona.component.ts` selected. The Code Editor shows the implementation of the `PersonaComponent` class in `persona.component.ts`. The class is decorated with `@Component` and implements `OnInit`. It has a `selector` of `'app-persona'`, a `templateUrl` of `'./persona.component.html'`, and `styleUrls` of `['./persona.component.css']`. The `ngOnInit()` method is implemented as `void`. The `clickMethod()` is implemented as follows:

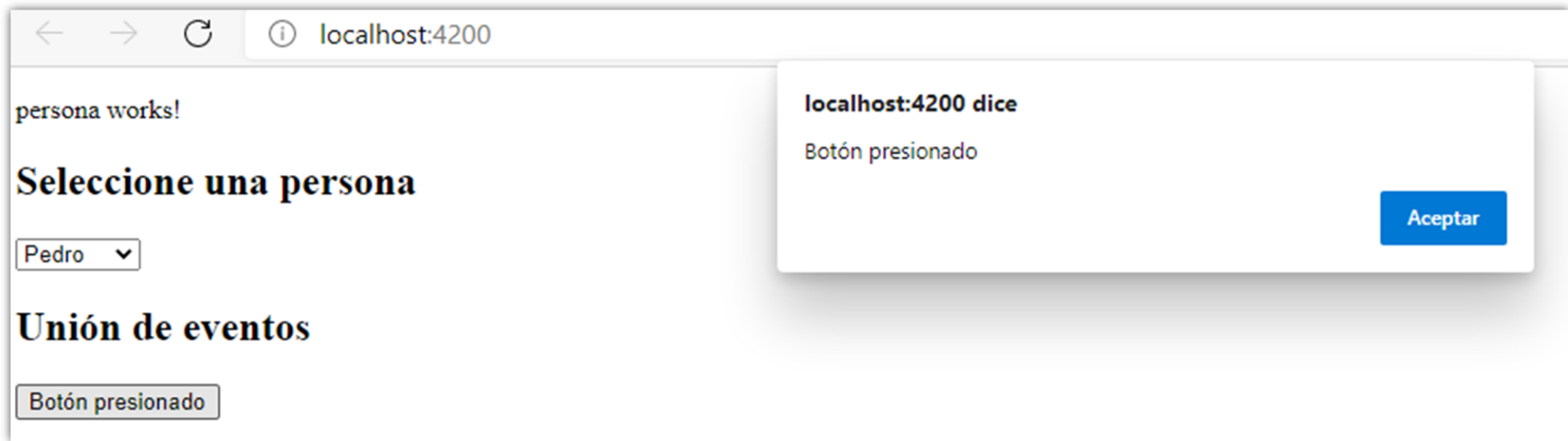
```
1 import { Component, OnInit } from '@angular/core';
2
3 @Component({
4   selector: 'app-persona',
5   templateUrl: './persona.component.html',
6   styleUrls: ['./persona.component.css']
7 })
8 export class PersonaComponent implements OnInit {
9
10   //Lista de personas
11   personas = ['Pedro', 'Juan', 'Diego', 'Carolina', 'Maria'];
12
13   ngOnInit(): void {
14   }
15
16   clickMethod() {
17     alert("Botón presionado");
18   }
19 }
20
```

TRASPASO DE INFORMACIÓN A COMPONENTES:

Traspaso de información a componentes.

Data Binding

Al presionar el botón se muestra el mensaje en pantalla.



BIBLIOGRAFÍA

Referencias bibliográficas

- Williamson, K. (2015). Learning AngularJS: A Guide to AngularJS Development.
- Freeman, A. (2014). Pro AngularJS (Expert's Voice in Web Development).
- Dayley, B. y Dayley, B. (2015). AngularJS, JavaScript, and jQuery All in One, Sams Teach Yourself.
- Freeman, A. (2020). Pro Angular 9: Build Powerful and Dynamic Web Apps.
- Seshadri, S. y Green, B. (2014). AngularJS: Up and Running: Enhanced Productivity with Structured Web Apps.
- Lerner, A. (2013). ng-book - The Complete Book on AngularJS
- Waikar, M. (2015). Data-oriented Development with AngularJS
- Ruebbelke, L. (2015). AngularJS in Action



an **NTT DATA** Company