PART 3

DATA BINDING

DATA BINDING

➤ So far what we have done are only static content

➤ Communication between your typescript code & template (html)

➤ Fetch data from a server, do some calculation then display

➤ To output data:

  ➤ String interpolation {{ data }}

  ➤ Property binding [property]="data"

➤ React to (User) Events

  ➤ Event binding (event)="expression"

  ➤ Two way data binding [(ngModel)] = "data"

STRING INTERPOLATION

```
<> app.component.html    TS servers.component.ts  ✕    <> servers.component.html

1    import { Component, OnInit } from '@angular/core';
2
3    @Component({
4      selector: 'app-servers',
5      templateUrl: './servers.component.html',
6      styleUrls: ['./servers.component.css']
7    })
8    export class ServersComponent implements OnInit {
9
10     name: string = "James";
11
12     constructor() { }
13
14     ngOnInit() {
15     }
16
17
18
19   }
20
```

```html
<div class="container">
  <p class="sample">
    {{ name }}
  </p>
</div>
```

PROPERTY BINDING

```html
<div class="container">
<p class="sample">
  {{ name }} - {{ serverId }}
</p>
</div>
<div class="container">
  <button class="btn btn-primary" [disabled]="!allowNewServer">Add Server</button>
  <p [innerText]="allowNewServer"></p>
</div>
```

```
name: string = "James";
serverId = 10;
allowNewServer = false;

constructor() {
  setTimeout(() => {
    this.allowNewServer = true;
  } ,2000)
}

ngOnInit() {
}
```

EVENT BINDING

```html
<div class="container">
<p class="sample">
  {{ name }} – {{ serverId }}
</p>
</div>
<div class="container">
  <button class="btn btn-primary" [disabled]="!allowNewServer" (click)="onCreateServer()">Add Server</button>
  <!-- <p [innerText]="allowNewServer"></p> -->
  <p>{{ serverCreationStatus }}</p>
</div>
```

EVENT BINDING

```typescript
import { Component, OnInit } from '@angular/core';
import { setTimeout } from 'timers';

@Component({
  selector: 'app-servers',
  templateUrl: './servers.component.html',
  styleUrls: ['./servers.component.css']
})
export class ServersComponent implements OnInit {

  name: string = "James";
  serverId = 10;
  allowNewServer = false;
  serverCreationStatus = "No server was created!";

  constructor() {
    setTimeout(() => {
      this.allowNewServer = true;
    } ,2000)
  }

  ngOnInit() {
  }

  onCreateServer() {
    this.serverCreationStatus = "Server was created!";
  }

}
```

```typescript
@Component({
  selector: 'app-servers',
  templateUrl: './servers.component.html',
  styleUrls: ['./servers.component.css']
})
export class ServersComponent implements OnInit {

  name: string = "James";
  serverId = 10;
  allowNewServer = false;
  serverCreationStatus = "No server was created!";
  serverName = '';

  constructor() {
    setTimeout(() => {
      this.allowNewServer = true;
    }, 2000)
  }

  ngOnInit() {
  }

  onCreateServer() {
    this.serverCreationStatus = "Server Created!";
  }

  onUpdateServerName(event: any) {
    console.log(event);
    this.serverName = event.target.value;
  }

}
```

MORE ON PASSING

```html
<div class="container">
<label for="">{{ serverName }}</label>
<input type="text" class="form-control" (input)="onUpdateServerName($event)">
<p class="sample">
  {{ name }} - {{ serverId }}
</p>
</div>
<div class="container">
  <button class="btn btn-primary" [disabled]="!allowNewServer" (click)="onCreateServer()">Add Server</button>
  <!-- <p [innerText]="allowNewServer"></p> -->
  <p>{{ serverCreationStatus }}</p>
</div>
```

EXERCISES

➤ Create an angular app

➤ That shows string Interpolation

➤ That shows property binding

➤ That shows event binding using click

➤ Use bootstrap design

➤ Add a input field that updates a property using two way data binding using ngModel from previous example

DIRECTIVES

➤ Instructions in the Dom

➤ Components are sample of this

NGIF TO OUTPUT DATA

➤ Directive based on booleans or expression

```html
<div class="container">
<label for="">{{ serverName }}</label>
<input type="text" class="form-control" (input)="onUpdateServerName($event)">
<p class="sample">
  {{ name }} - {{ serverId }}
</p>
</div>
<div class="container">
  <button class="btn btn-primary" [disabled]="!allowNewServer" (click)="onCreateServer()">Add Server</button>
  <!-- <p [innerText]="allowNewServer"></p> -->
  <p>{{ serverCreationStatus }}</p>
  <p *ngIf="serverCreated">Server was created, server name is {{ serverName }}</p>
</div>
```

MORE ON NGIF

```typescript
export class ServersComponent implements OnInit {

  name: string = "James";
  serverId = 10;
  allowNewServer = false;
  serverCreationStatus = "No server was created!";
  serverName = '';
  serverCreated = false;

  constructor() {
    setTimeout(() => {
      this.allowNewServer = true;
    }, 2000)
  }


  ngOnInit() {
  }


  onCreateServer() {
    this.serverCreated = true;
    this.serverCreationStatus = "Server Created!";
  }
```

OTHER DIRECTIVES

➤ ngIf & Else

➤ ngStyle

➤ ngClass

➤ ngFor

END OF PART 3