Data Binding and Event Handling Lab - Part 2

This lab is a continuation of the previous lab.

**Goals**

* Add event handling
* Display content using structural directives

1. Add Event Handling

In these exercises you add two buttons for event handling. When the user clicks a button, it calls a method in a component. The component updates the field value, and you see the updates in your browser.

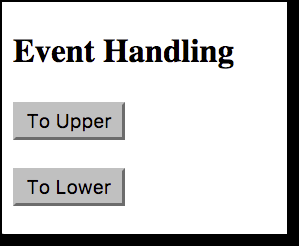
1.1. Add Buttons and Event-Handling Hooks

1. Open **src/app/user/user.component.html**.
2. Add the following code to the **end** of the file:
3. <hr>
4. <h3>Event Handling</h3>
5. <p>
6. <button (click)="convertToUpperCase()">To Upper</button>
7. </p>
8. <p>
9. <button (click)="convertToLowerCase()">To Lower</button>

</p>

* + This code adds buttons labeled **To Upper** and **To Lower** to the view.
  + An event handler is added to each button for the **click** event.
  + When the user clicks the button, the event handler calls a method in the component.
    - If the user clicks **To Upper**, the handler calls the **convertToUpperCase()** method in the **UserComponent** class.
    - If the user clicks **To Lower**, the handler calls the **convertToLowerCase()** method in the **UserComponent** class.

1. Save the file.
2. Move back to your web browser and verify that you see a screen similar to this:



|  |  |
| --- | --- |
|  | The buttons do not work yet. You still need to add event-handling methods. You do that in the next exercise. |

1.2. Add Component Code for Event-Handling Methods

Here you define two event-handling methods in the **UserComponent** class.

1. Open **src/app/user/user.component.ts**.
2. After the **ngOnInit()** method, add these two new methods:
3. convertToUpperCase() {
4. this.firstName = this.firstName.toUpperCase();
5. this.lastName = this.lastName.toUpperCase();
6. }
7. convertToLowerCase() {
8. this.firstName = this.firstName.toLowerCase();
9. this.lastName = this.lastName.toLowerCase();

}

* + These methods are called when the user clicks a button in the view.
  + These methods have the same name as specified in the view.
  + These methods change the **firstName** and **lastName** fields to uppercase and lowercase versions.

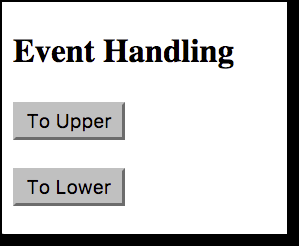
1. Save the file and confirm that your final **user.component.ts** file looks like this:
2. import { Component, OnInit } from '@angular/core';
3. @Component({
4. selector: 'app-user',
5. templateUrl: './user.component.html',
6. styleUrls: ['./user.component.css']
7. })
8. export class UserComponent implements OnInit {
9. firstName : string = "Mary";
10. lastName : string = "Doe";
11. constructor() { }
12. ngOnInit() {
13. }
14. convertToUpperCase() {
15. this.firstName = this.firstName.toUpperCase();
16. this.lastName = this.lastName.toUpperCase();
17. }
18. convertToLowerCase() {
19. this.firstName = this.firstName.toLowerCase();
20. this.lastName = this.lastName.toLowerCase();
21. }

}

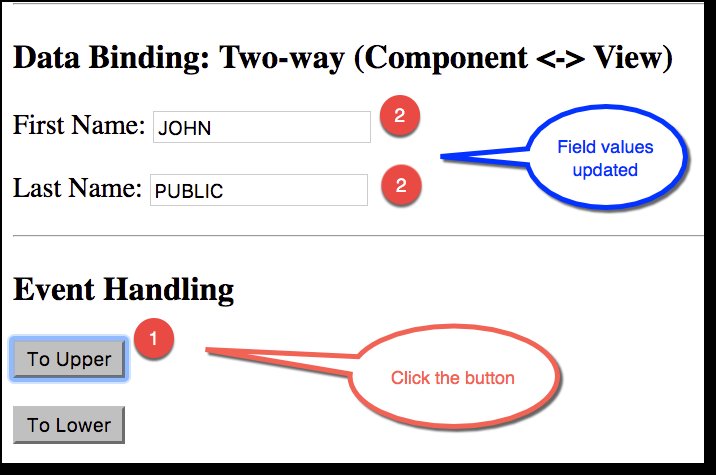
1.3. Test Your App

Now test your app in the browser.

1. Move back to your web browser and confirm that you see a screen similar to this:



1. Click **To Upper**.
2. Confirm that the values are updated in the view:



2. Display Content Using Structural Directives

In these exercises you use the **ngFor** structural directive to display content.

2.1. Define Array Field in the **User** Component

1. Open **src/app/user/user.component.ts**.
2. Move to the line above **constructor()**.
3. Add a new field array:

cities : string[] = ["Philadelphia", "Milan", "Bengaluru", "Tokyo"];

* + This is simply an array of strings. In the next exercise you use the **ngFor** directive to display the cities.

1. Save the file.
   * Expect your final **user.component.ts** file to look like this:
   * import { Component, OnInit } from '@angular/core';
   * @Component({
   * selector: 'app-user',
   * templateUrl: './user.component.html',
   * styleUrls: ['./user.component.css']
   * })
   * export class UserComponent implements OnInit {
   * firstName : string = "John";
   * lastName : string = "Public";
   * cities : string[] = ["Philadelphia", "Milan", "Bengaluru", "Tokyo"];
   * constructor() { }
   * ngOnInit() {
   * }
   * convertToUpperCase() {
   * this.firstName = this.firstName.toUpperCase();
   * this.lastName = this.lastName.toUpperCase();
   * }
   * convertToLowerCase() {
   * this.firstName = this.firstName.toLowerCase();
   * this.lastName = this.lastName.toLowerCase();
   * }

}

2.2. Display Content in the View

1. Open **src/app/user/user.component.html**.
2. Add the following code to the **end** of the file:
3. <hr>
4. <h3>Structural Directives</h3>
5. <ul>
6. <li \*ngFor="let tempCity of cities">
7. {{tempCity}}
8. </li>

</ul>

* + This code builds an unordered list of cities.
  + **\*ngFor** indicates that the **<li>** element and its children constitute a master template.
  + The **ngFor** directive iterates over the component’s **cities** array and renders an instance of this template for each city in the array.
  + The **let tempCity** part of the expression identifies **tempCity** as the template input variable that holds the current city item for each iteration.
  + The current city is displayed using **{{tempCity}}**.

1. Save the file.

2.3. Test Your App

Now test your app in the browser.

* Move back to your web browser and confirm that you see a screen similar to this:

