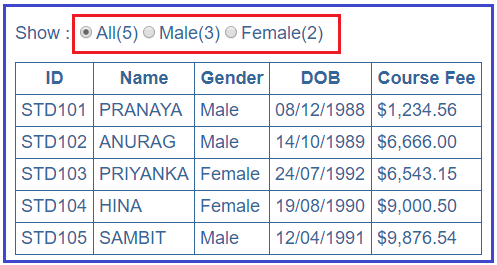
**Angular Component Input Properties**

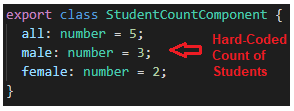
##### **What are Angular Component Input Properties?**

The Angular Component Input Properties are used to pass the data from container component to the nested component

As of now, we have hard-coded the count of students in the **StudentCountComponent**that are displayed next to each radio buttons as shown in the below image.



The following code in the **student-count.component.ts** file does this.



As shown in the above image the values for the 3 properties (all, male, female) are hard-coded. But, here we want the values for these 3 properties (all, male and female) to be passed from the container component i.e. **StudentListComponent**.

##### **How to pass the data from the container component to the nested component?**

We can pass the data from the container component to the nested component using the input property of the angular component. We can easily convert an angular property to an angular input property by using the **@Input** decorator.

In our example, to pass the values for these 3 properties (all, male and female) from the container component i.e. **StudentListComponent**to the nested component i.e. **StudentCountComponent**, we need to decorate the **StudentCountComponent**properties (all, male and female) with **@Input()** decorator.

When we decorate a property with **@Input()**decorator then that property becomes an input property. To use the **@Input()** decorator in your component, first, you need to import it from the **@angular/core**.

So, let’s modify the **StudentCountComponent**as shown below to use the Angular Component **Input**Properties. Open student-count.component.ts file and then copy and paste the following code in it.

**import** **{** Component, Input **}** from '@angular/core';

@Component**({**

selector: 'app-student-count',

templateUrl: './student-count.component.html',

styleUrls: **[**'./student-count.component.css'**]**

**})**

**export** **class** StudentCountComponent **{**

@Input**()**

all: **number**;

@Input**()**

male: **number**;

@Input**()**

female: **number**;

**}**

Here, you can to notice that we have removed the default hard-coded values, as we are going to pass the values for these properties from the Container Component i.e. **StudentListComponent**..

##### **How to Pass data from the parent component to the child component?**

We need to do two modifications in the **StudentListComponent**in order to pass the values from the parent component i.e. **StudentListComponent**to the child component i.e. **StudentCountComponent**.

##### **Modifying student-list.component.ts file:**

So open the student-list.component.ts file and then copy and paste the below code. As you can see in the below code, we have created three methods that are going to return the male students count, female students count and total students count.

**import** **{** Component**}** from '@angular/core';

@Component**({**

selector: 'app-student-list',

templateUrl: './student-list.component.html',

styleUrls: **[**'./student-list.component.css'**]**

**})**

**export** **class** StudentListComponent **{**

students: **any[]** = **[**

**{**

ID: 'std101', FirstName: 'Pranaya', LastName: 'Rout',

DOB: '12/8/1988', Gender: 'Male', CourseFee: 1234.56

**}**,

**{**

ID: 'std102', FirstName: 'Anurag', LastName: 'Mohanty',

DOB: '10/14/1989', Gender: 'Male', CourseFee: 6666.00

**}**,

**{**

ID: 'std103', FirstName: 'Priyanka', LastName: 'Dewangan',

DOB: '7/24/1992', Gender: 'Female', CourseFee: 6543.15

**}**,

**{**

ID: 'std104', FirstName: 'Hina', LastName: 'Sharma',

DOB: '8/19/1990', Gender: 'Female', CourseFee: 9000.50

**}**,

**{**

ID: 'std105', FirstName: 'Sambit', LastName: 'Satapathy',

DOB: '4/12/1991', Gender: 'Male', CourseFee: 9876.54

**}**

**]**;

getTotalStudentsCount**()**: **number** **{**

**return** this.students.length;

**}**

getMaleStudentsCount**()**: **number** **{**

**return** this.students.filter**(**std => std.Gender === 'Male'**)**.length;

**}**

getFemaleStudentsCount**()**: **number** **{**

**return** this.students.filter**(**std => std.Gender === 'Female'**)**.length;

**}**

**}**

The points to remember is that in the filter method we are using the triple equals (===) instead of the double equals (==). The meaning single, double and triple equals in TypeScript are as follows

1. = Assign a value
2. == Compare two values
3. === Compare two values and their types

##### **Modify student-list.component.html file:**

Open student-list.component.html file and then copy and paste the following code in it. As you can see in the above HTML, within the **<app-student-count>** directive we are using the property binding to bind the properties (all, male, female) of the nested component i.e. **StudentCountComponent** with the 3 methods in the container component i.e. **StudentListComponent**.

**<app-student-count** [all]="getTotalStudentsCount()"

[male]="getMaleStudentsCount()"

[female]="getFemaleStudentsCount()"**></app-student-count>**

**<br/><br/>**

**<table>**

**<thead>**

**<tr>**

**<th>**ID**</th>**

**<th>**Name**</th>**

**<th>**Gender**</th>**

**<th>**DOB**</th>**

**<th>**Course Fee**</th>**

**</tr>**

**</thead>**

**<tbody>**

**<tr** \*ngFor='let student of students'**>**

**<td>**{{student.ID | uppercase}}**</td>**

**<td>**{{student.FirstName | uppercase}}**</td>**

**<td>**{{student.Gender}}**</td>**

**<td>**{{student.DOB | date:'dd/MM/y'}}**</td>**

**<td>**{{student.CourseFee | currency:'USD':true:'1.2-2'}}**</td>**

**</tr>**

**<tr** \*ngIf="!students || students.length==0"**>**

**<td** colspan="10"**>**

No Students to display

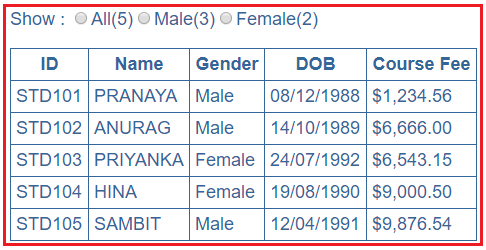
**</td>**

**</tr>**

**</tbody>**

**</table>**

That’s it. We have done with our part 2 implementation. So save all the changes and run the application and you will see the correct count of students next to each radio button as shown in the image below.



To confirm whether it’s working or not, let’s add the following student object to the student’s array in the **StudentListComponent** as shown below.

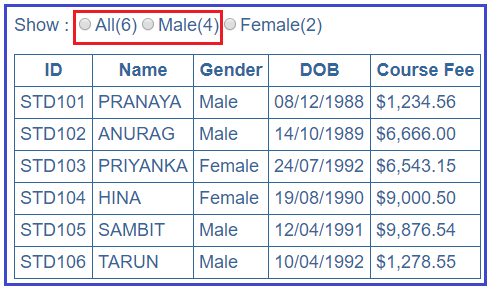
{

ID: 'std106', FirstName: 'Tarun', LastName: 'Mallick',

DOB: '4/10/1992', Gender: 'Male', CourseFee: 1278.55

}

Now, save all the changes and then reload the web page and you will notice that All count and the Male count is increased by 1 as expected as shown in the image below.



At the moment when we click on the radio buttons, then nothing is happening. So in the next article we will discuss [**how to pass the data from the child component to the parent component**](https://dotnettutorials.net/lesson/angular-component-output-properties/)i.e. when a radio button’s checked event is raised in the child component i.e. **StudentCountComponent**, we want to know about it in the parent component i.e. **StudentListComponent** so that we can react to them and then decide which students to show in the table depending on the selection of the radio button.