Now we will pass data from child component to Parent component using @Output decorator

We will first write code to select the radio button.

 selectedButtonValue:string='All'

and in radiobutton we will change the value according to selection

<span>Filter</span> &nbsp;&nbsp;

<input

  type="radio"

  name="filter"

  value="All"

  [(ngModel)]="selectedButtonValue"

/>

All coursess({{ all }})&nbsp; &nbsp;

<input

  type="radio"

  name="filter"

  value="Available"

  [(ngModel)]="selectedButtonValue"

/>

Available({{ available }})&nbsp; &nbsp;

<input

  type="radio"

  name="filter"

  value="Notavailable"

  [(ngModel)]="selectedButtonValue"

/>

NotAvailable({{ notavailable }})<br />

Now we will create custom event in Child class ;

Step 1:Create property and it’s type as eventemitter and create object and use @Output annonations.

@Output()

  filterradiobuttonSelctionChange: EventEmitter<string>= new EventEmitter <string>

Step 2: Create a method which will raise the event.

onRadioButtonSelectionChange() {

    this.filterradiobuttonSelctionChange.emit(this.selectedButtonValue)

  }

This method will set the value.

Now we have to call this event on radio button

 (change)="onRadioButtonSelectionChange()".

<span>Filter</span> &nbsp;&nbsp;

<input

  type="radio"

  name="filter"

  value="All"

  [(ngModel)]="selectedButtonValue"

**(change)="onRadioButtonSelectionChange()"**

/>

All coursess({{ all }})&nbsp; &nbsp;

<input

  type="radio"

  name="filter"

  value="Available"

  [(ngModel)]="selectedButtonValue"

**(change)="onRadioButtonSelectionChange()"**

/>

Available({{ available }})&nbsp; &nbsp;

<input

  type="radio"

  name="filter"

  value="Notavailable"

  [(ngModel)]="selectedButtonValue"

**(change)="onRadioButtonSelectionChange()"**

/>

NotAvailable({{ notavailable }})<br />

Now we will create a variable in parent component to hold the value of selected radiobutton.

 productCountRadioButton: string = 'All';

a method is also created to hold the record

onFilterRadioButtonChange(dat: string) {

    this.productCountRadioButton = dat;

  }

Step: Final on

<app-filter

  [all]="getAll()"

  [available]="getAvailable()"

  [notavailable]="getNotAvailable()"

**(filterradiobuttonSelctionChange)="onFilterRadioButtonChange($event)"**

></app-filter>

We will now use ng-container as \*ngFor and \*ngif command cannot be used in single line.

 \*ngIf="

      productCountRadioButton === 'All' || productCountRadioButton === p.details