Harmony IT Solution

learning Hub

Angular

Tahaluf Training Center 2022

Harmony IT Solution

1. Reusable component.
2. Input Decorator.
3. Output Decorator.

Harmony IT Solution

Objective

The Objective of this lecture

- Get to know about the reusable component concept and its purpose for it.
- The Input and output decorator and what the purpose for using these decorators.

Harmony IT Solution

Reusable Component

Overview of Reusable Component

It defines a base structure and behavior that can be used in different contexts/dynamic content.

With every reusable component, we have a parent component.

In the reusable component, this flexible content comes from parent content and ends up in a dedicated slot.

In other words, it is projected down to the parent component.

The Purpose of Reusable Component

- Efficiency: Using the same markup across components, making future changes easy.
- Consistency: Whenever reusable components are updated; they get affected across all of their uses.
- Easy to Test: Testing becomes easier when SRPs are followed.

Harmony IT Solution

Demo of Reusable Component

Harmony IT Solution

Reusable Component Example

Create a new component called CourseCard.

Note: This component will contain the reusable code.

```
PS C:\Users\d.kanaan.ext\Desktop\EduTech> ng g c courseCard
CREATE src/app/course-card/course-card.component.html (26 bytes)
CREATE src/app/course-card/course-card.component.spec.ts (655 bytes)
CREATE src/app/course-card/course-card.component.ts (294 bytes)
CREATE src/app/course-card/course-card.component.css (0 bytes)
UPDATE src/app/app.module.ts (1010 bytes)
```

Harmony IT Solution

Reusable Component Example

Create another component called courses.

```
PS C:\Users\d.kanaan.ext\Desktop\EduTech> ng g c courses
CREATE src/app/courses/courses.component.html (22 bytes)
CREATE src/app/courses/courses.component.spec.ts (633 bytes)
CREATE src/app/courses/courses.component.ts (279 bytes)
CREATE src/app/courses/courses.component.css (0 bytes)
UPDATE src/app/app.module.ts (1096 bytes)
```

Harmony IT Solution

Reusable Component Example

Add the routing for the course component.
In-app-routing.module.ts

```
{
  path:'course',
  component:CoursesComponent
}
```

Reusable Component Example

The course card component template will be the cards for each course in the system.

So instead of repeating the card more than once according to the number of courses in the system, we resort to using the principle of **reusable components**.

Harmony IT Solution

Reusable Component Example

In the courseCard.compoment.html :

```
<div class="">
    <div class="col-md-4 col-sm-4">
    <div class="item">
     <div class="courses-thumb">
     <div class="courses-top">
     <div class="courses-image">
    <img src="{{imagename}}" style="width: 300px;height: 200px;"
    class="img-responsive" alt="">
            </div>
```

Harmony IT Solution

Reusable Component Example

In the courseCard.compoment.html :

```
        <div class="courses-date">
            <span><i class="fa fa-calendar"></i> {{startdate}}</span>
            <span><i class="fa fa-clock-o"></i> {{enddate}}</span>
        </div>
    </div>
    <div class="courses-detail">
        <h3><a href="#">{{coursename}}</a></h3>
    </div>
    </div>
    </div>
    </div>
    </div>
</div>
```

Reusable Component

There is more than one way to send data from parent to child component.

- Using the Input and Output decorator.

Harmony IT Solution

Sending the data as properties in the parent-child component Example :

In course.component.html :

```
<app-navbar></app-navbar>
<app-course-card coursename="Angular"
time="3:00 "
startdate ="1/1/2022" enddate="15/1/2022"
imagename="https://encrypted-tbn0.gstatic.com/images?q=tbn:
ANd9GcRRFhAZzXSieePNJh-4luKVxt2fbjyOs2zdv030JS_
poIL10ICMzF9vfSD23myYxyQOVzQ&usqp=CAU">
</app-course-card>
```

Harmony IT Solution

Sending the data as properties in the parent-child component Example :

In course.component.html:

```
<app-course-card coursename="API"
 time="3:00 " startdate ="15/12/2021"
  enddate="30/12/2021"
  imagename="https://blog.axway.com
/wp-content/uploads/2019/07/GettyImages-
1156783188-2.jpgAPI-Mashup-2.jpg">
</app-course-card>
```

Harmony IT Solution

Input Decorator

Overview of Input Decorator

@Input is a decorator for marking a property as an input.

It can be used to specify a component property.

Allows data to be passed between parent and child components (property binding).

Component properties should be annotated with the @Input decorator to act as input properties.

Input Decorator Example

Step One :

Start with creating an array to store the properties for each selector.
(Each selector represents an object within this array)

Harmony IT Solution

Input Decorator Example

So in courses.component.ts :

```
courses:any=[{
coursename:'API’,
startdate:'15/12/2021',
enddate:'30/12/2021',
imagename:"https://blog.axway.com/wp-content/uploads/
2019/07/GettyImages-1156783188-2.jpgAPI-Mashup-2.jpg"
 },
```

Harmony IT Solution

Input Decorator Example

```
,{
coursename:"Angular",
startdate:"1/1/2022",
enddate:"15/1/2022",
imagename:"https://encrypted-
tbn0.gstatic.com/images?q=tbn:ANd9GcRRFhAZzXSieePNJh-
4luKVxt2fbjyOs2zdv030JS_poIL10ICMzF9vfSD23myYxyQOVzQ&usqp
=CAU"},
{coursename:"Scrum",
startdate:"18/1/2022",
enddate:"21/1/2022",
imagename:"https://i2.wp.com/iraqtech.io/wp-content/
uploads/Agile-framwork.jpg"}]
```

Harmony IT Solution

Input Decorator Example

Step Two :

Open parent component views (courses.component.html) then pass this variable to the child component instance, which is passed to the parent component.

```
<app-course-card *ngFor="let obj of courses"
[coursename]="obj.coursename“
[startdate] ="obj.startdate"
[enddate]="obj.enddate" [imagename]="obj.imagename">
</app-course-card>
```

Harmony IT Solution

Input Decorator Example

Step Three :

Import the input on the child component (course card component).

So, in the courseCard.component.ts

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

Harmony IT Solution

Input Decorator Example

Step Four :

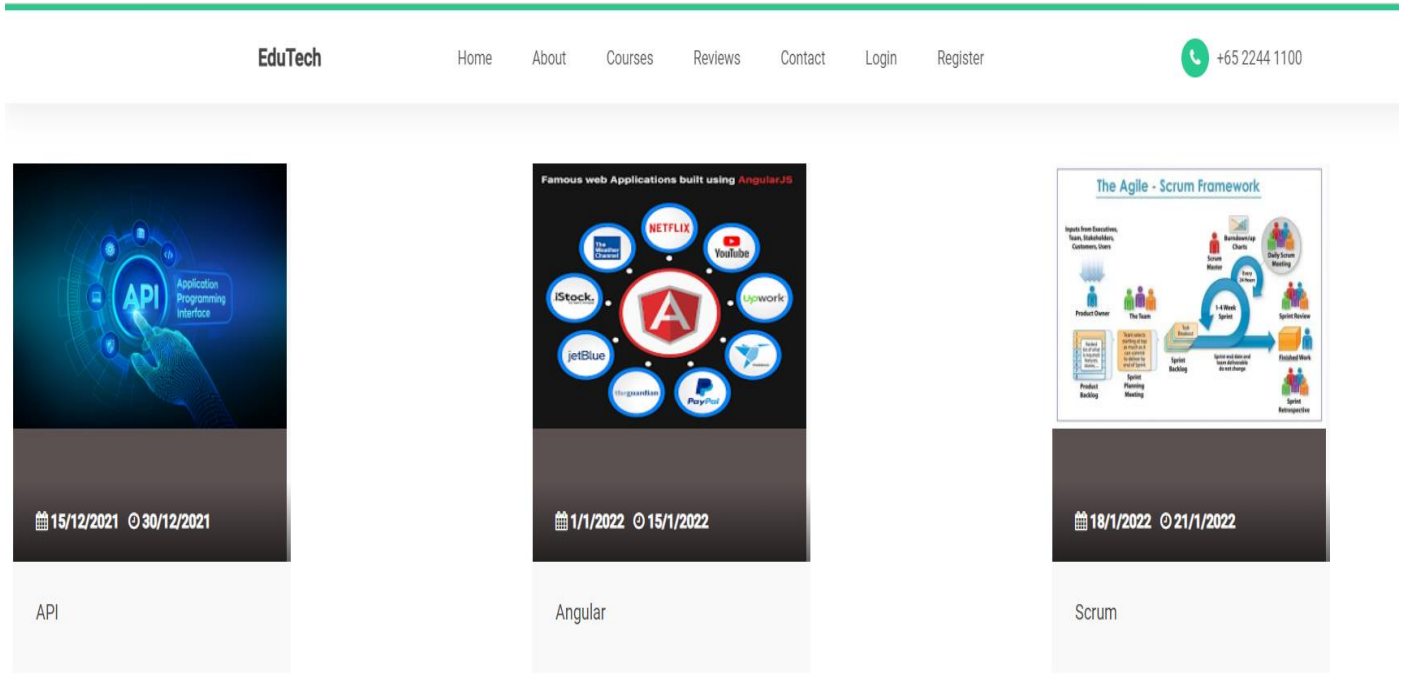
Create variables that are passed from the parent component to the child component.

In the coursCard.component.ts

```
@Input() coursename :string|undefined
@Input() startdate :string|undefined
@Input() enddate :string|undefined
@Input() teachercourse :string|undefined
@Input() imagename :string|undefined
```

Harmony IT Solution

The Example Result



Harmony IT Solution

Output Decorator

Overview of Output Decorator

In the @Output decorator, data is passed from the child to the parent component.

This annotation binds a property of type Angular EventEmitter.

Use it in components with the @Output directive to emit custom events synchronously or asynchronously.

Harmony IT Solution

Output Decorator Example

Create an instance of the eventEmmiter class In the child component to emit the output event in the parent component.

```
@Output() opneProfile=new EventEmitter();
```

Harmony IT Solution

Exercises

Generate a new component called profile and give it a routine.

Exercises

Create an output event that redirects to the profile component when the user clicks on the course image.

Harmony IT Solution

Exercises _ Solution

In the child views (courseCard.component.html)
Add the click event in the image tag

```
<img (click)="showPorfile()" src="{{imagename}}"
style="width: 300px;height: 200px;"
class="img-responsive" alt="">
```

Exercises _ Solution

Then implement the show profile function in the courseCard.component.ts

```
showPorfile(){
this.router.navigate(['profile']); }
```

Note: Don’t forget to define an instance of the router class in the constructor.

Harmony IT Solution

Exercises _ Solution _ 2

Using the output decorator to pass the data from the child to the parent component.

The show profile function will emit the open profile that is passed to the parent component (course component).

Exercises _ Solution _ 2

Step One

Import the output decorator and the Event Emitter in the child component (course card component).

In courseCard.component.ts

```
import { Component, EventEmitter, Input, OnInit, Output }
 from '@angular/core';
```

Harmony IT Solution

Exercises _ Solution _ 2

Step Two

Create an instance of Event Emitter in the child component.

```
@Output() opneProfile=new EventEmitter();
```

Harmony IT Solution

Exercises _ Solution _ 2

Step Three

In the child component views (courseCard.component.html)

```
<img (click)=goToProfile()" src="{{imagename}}" style="width
200px;"
class="img-responsive" alt="">
```

Harmony IT Solution

Exercises _ Solution _ 2

Step Three

In the courseCard.component.ts

```
goToProfile(){
  this.opneProfile.emit();
}
```

Harmony IT Solution

Exercises _ Solution _ 2

In the courses.component.html add an output event called show profile.

```
app-course-card *ngFor="let obj of courses"
  [coursename]="obj.coursename"
  [startdate] ="obj.startdate"
  [enddate]="obj.enddate"
  [imagename]="obj.imagename" (click)="showPorfile()"></app-course-card>
```

Harmony IT Solution

References

[1] Angular, “Angular,” *Angular.io*, 2019. https://angular.io/

[2] “Complete Angular Tutorial For Beginners,” *TekTutorialsHub*. https://www.tektutorialshub.com/angular-tutorial/

[3]“npm | build amazing things,” *Npmjs.com*, 2019. https://www.npmjs.com/

[4]“Angular Tutorial for Beginners | Simplilearn,” *Simplilearn.com*. https://www.simplilearn.com/tutorials/angular-tutorial (accessed Aug. 19, 2022).

Harmony IT Solution