# Mobile Application Development

LECTURE 10

# Passing Data Between Components

**Creating Component inside an Ionic Page**

- Lets, first create a component from our existing code. Here's studentspage.page.html.

- Since we might use <ion-item> somewhere else, we'll create a component of it. Reusability.

```
<ion-header>
  <ion-toolbar>
    <ion-buttons slot="start">
      <ion-menu-button></ion-menu-button>
    </ion-buttons>
    <ion-title>Students List</ion-title>
  </ion-toolbar>
</ion-header>

<ion-content>

  <ion-card *ngFor="let item of students">
    <ion-item (click)="changeUrl(item)">
      <ion-card-content>
        <ion-label>{{ item?.name }} | {{ item?.id.substring(0,4) }}
        </ion-label>
      </ion-card-content>
    </ion-item>
  </ion-card>
```

# Creating Component inside an Ionic Page

- Lets generate a component. I typed the name as studentslist/studentslistitem because I don't want it in App folder but in studentslist folder.

```
C:\Users\Alamgir\Desktop\MAD-Workbooks\mad-workbooks (lecture10 -> origin)
λ ionic g
? What would you like to generate? component
? Name/path of component: studentslist/studentslistitem
> ng.cmd generate component studentslist/studentslistitem
CREATE src/app/studentslist/studentslistitem/studentslistitem.component.html (35 bytes)
CREATE src/app/studentslist/studentslistitem/studentslistitem.component.spec.ts (796 bytes)
CREATE src/app/studentslist/studentslistitem/studentslistitem.component.ts (308 bytes)
CREATE src/app/studentslist/studentslistitem/studentslistitem.component.scss (0 bytes)
[OK] Generated component!
```

# Creating Component inside an Ionic Page

- We move the relevant code to the new component files.

src > app > studentslist > studentslistitem > studentslistitem.component.html > ion-item

```
1  <ion-item (click)="changeUrl(item)">
2    <ion-card-content>
3      <ion-label>{{ item?.name }} | {{ item?.id.substring(0,4) }}
4      </ion-label>
5    </ion-card-content>
6  </ion-item>
```

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

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

  constructor(private router: Router) { }

  ngOnInit() {}

  changeUrl(user) {
    const id = user.id.substring(0, 4);
    const url = `studentslist/${id}`;

    this.router.navigateByUrl(url);

    // or

    // this.router.navigate([url]);
  }
}
```

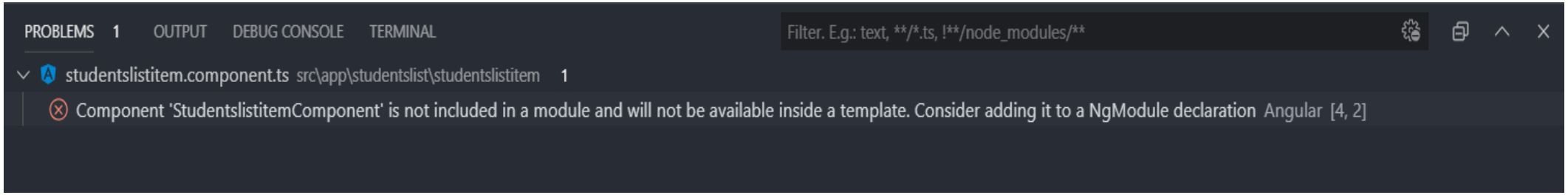
# Creating Component inside an Ionic Page

- Here's how the code would look like. But we have a few problems. We created a component but did not add to any module. Remember, every component belongs to some module.

```
<ion-card *ngFor="let item of students">
  <app-studentslistitem></app-studentslistitem>
  <!-- <ion-item (click)="changeUrl(item)">
    <ion-card-content>
      <ion-label>{{ item?.name }} | {{ item?.id.substring(0,4) }}
      </ion-label>
    </ion-card-content>
  </ion-item> -->
</ion-card>
```

# Creating Component inside an Ionic Page

- So we will import it in the module file of studentslist. VSCode is also telling us about the problem




- Now we go to the studentslist.module.ts file and add the component into the declarations

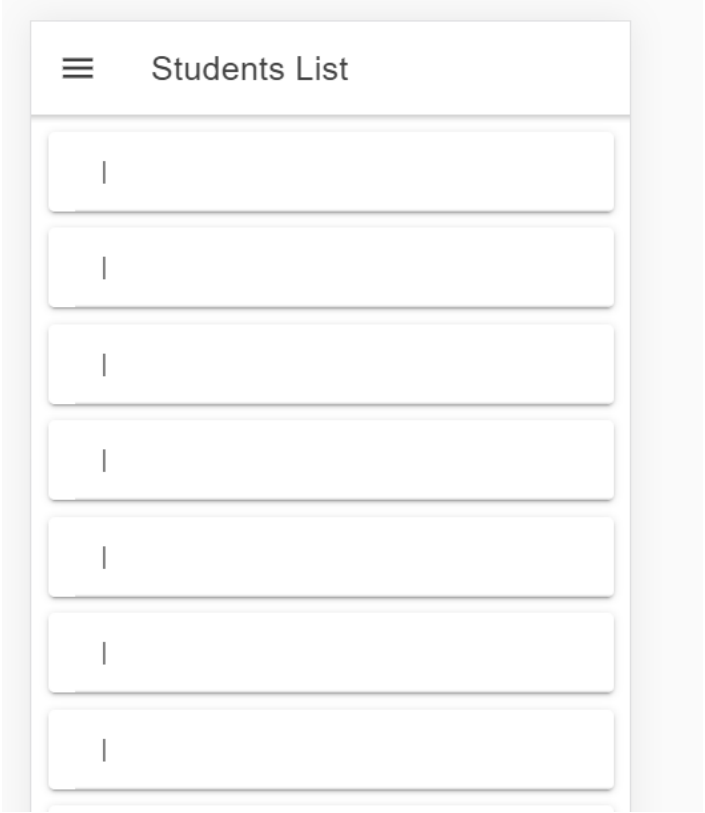
# Creating Component inside an Ionic Page

- Just copy the class name and past it inside declarations, you will see a yellow icon, click on it and it will auto-import the file.

```
import { FormsModule } from '@angular/forms';
import { Routes, RouterModule } from '@angular/router';

import { IonicModule } from '@ionic/angular';

import { StudentslistPage } from './studentslist.page';
import { StudentslistitemComponent } from './studentslistitem/studentslistitem.component';

const routes: Routes = [
  {
    path: '',
    component: StudentslistPage
  }
];

Unsaved changes (cannot determine recent change or authors)
@NgModule({
  imports: [
    CommonModule,
    FormsModule,
    IonicModule,
    RouterModule.forChild(routes)
  ],
  declarations: [StudentslistPage,StudentslistitemComponent]
})
export class StudentslistPageModule {}
```

# Creating Component inside an Ionic Page

- Now our app is working but its not displaying anything.
- Why? Because we have to tell it what to display.
- It has no idea what **item** variable is.
- So we use **@Input()** property binding, to pass item to the component.



# Passing Data Between Components

- Now we pass via Input item variable to the component. But the linter is telling us that the component doesnot know what **item** is.

```
<ion-card *ngFor="let item of students">
  <app-studentslistitem [item]="item"></app-studentslistitem>
  <!-- <ion-item (click)="changeUrl(item)">
    <ion-card-content>
      <ion-label>{{ item?.name }} | {{ item?.id.substring(0,4) }}
      </ion-label>
    </ion-card-content>
  </ion-item> -->
</ion-card>
```

# Passing Data Between Components

- We create @Input() and let it know that its will be passed down from a top/higher component.

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

import { Router } from '@angular/router';

@Component({
  selector: 'app-studentslistitem',
  templateUrl: './studentslistitem.component.html',
  styleUrls: ['./studentslistitem.component.scss']
})
export class StudentslistitemComponent implements OnInit {
  constructor(private router: Router) {}
  @Input() item;
  ngOnInit() {}
```

# Passing Data Between Components

- Now we run our application and its working the same as before.

- **Parent, Child and Sibling Components**

- Parent component are basically higher/upper component for a inner component. In our example,

**app-studentslist** is the parent component to **app-studentslistitem** while it itself is child Component

```
<ion-card *ngFor="let item of students">
  <app-studentslistitem [item]="item"></app-studentslistitem>

</ion-card>
```

# Passing Data Between Components

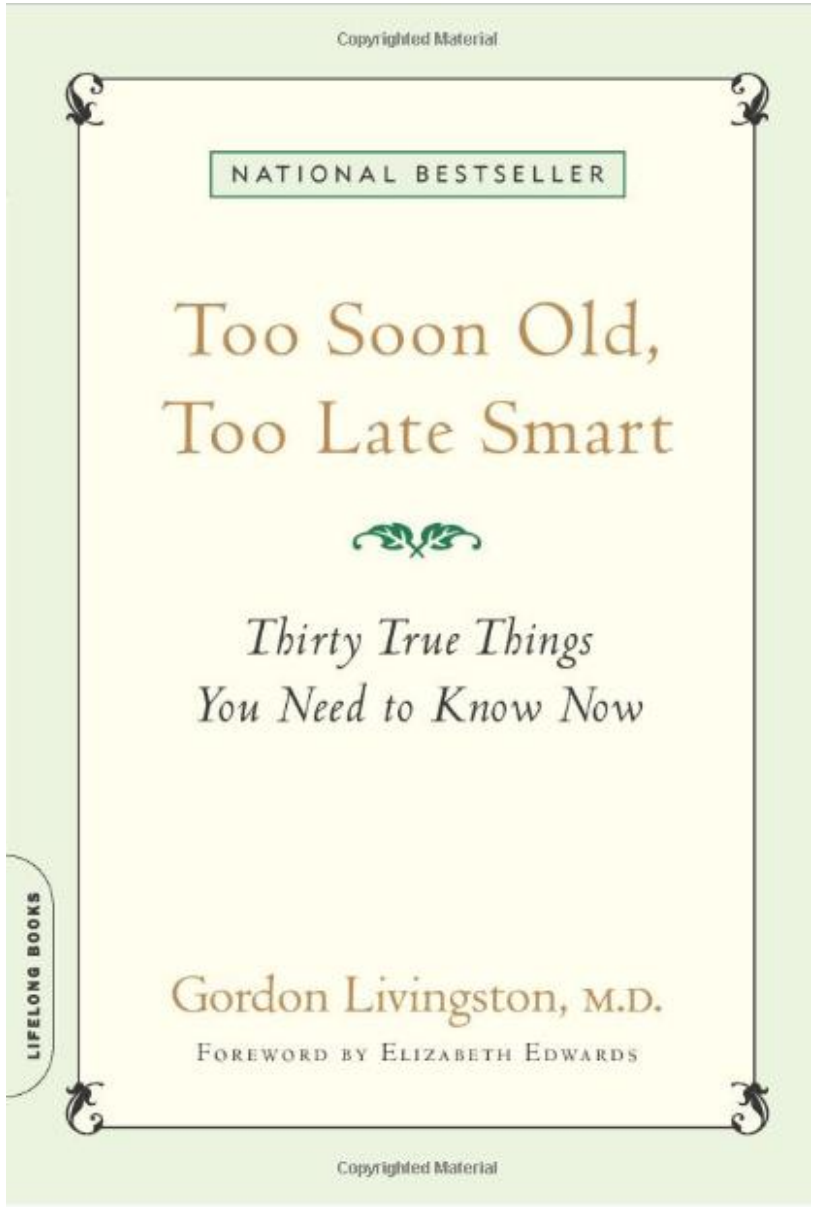
- Sibling components are basically on the same level. Siblings mean brother/sisters. They're the same level.

- Here, app-studentslistitem and app-my-custom-component are sibling components because they're at the same level.

```
<ion-card *ngFor="let item of students">
  <app-studentslistitem [item]="item"></app-studentslistitem>

</ion-card>

<app-my-custom-component> </app-my-custom-component>
```

*“Only bad things happen quickly.*

*Virtually all the happiness-producing processes in our lives take time, usually a long time: learning new things, changing old behaviors, building satisfying relationships, raising children. This is why patience and determination are among life’s primary virtues.”*



# Passing Data Between Components

- There are three major ways to pass / share data between components.

1. Using Input(), Output() event emitters.

2. Using a parents shared state.

3. Using an Angular Service.

# Passing Data Between Components

**Using Input(), Output() event emitters**

This is ideal for parent, child components. However, once the number of components increase, this becomes hard.

**Using a parents shared state**

We declare the variable in the parent component and keep passing it down via Input. Can get complicated very quickly.

# Passing Data Between Components

**Using Services**

This is mostly the desired approach when using Input(), Output() becomes complicated.

**Using Store/NgRx**

This is an option for large applications. Basically all the state is stored in one global store. We will not cover this in this course.

# Passing Data Between Components

**Review:**

Parent to Child: Sharing Data via Input.

Child to Parent: Sharing Data via Output.

When passing data between components that lack a direct connection, such as siblings, grandchildren, etc, you should you a shared service. We will use rxjs alongside Services (BehaviorSubject).