**Ways to call one component in another component in Angular 10**

*Different ways you can include one angular component in another*

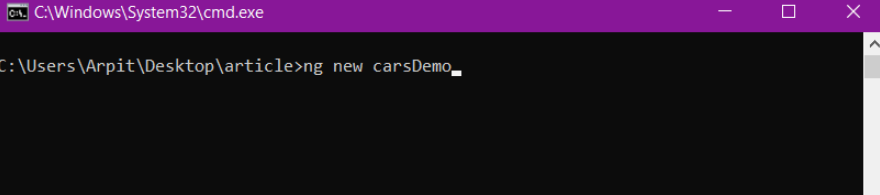
The one of the advantages provided by angular is to divide a page in separate components. This makes managing page content and business logic easy, with business logic in there own component. Also when you need the same functionality at various places based on condition, reusable components helps like a charm.

To use your child component in the main page (which is also a component) , it is required to call/include in the parent component. It can done in the 3 ways.

Here I will quickly walk you over steps to create angular application and components then we will move to ways to call component.

**1. Create angular project**

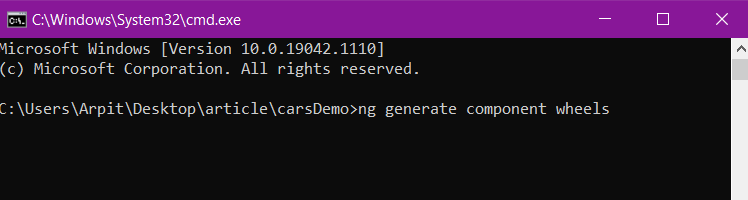
Run the following command in the command prompt “*ng new carsDemo”*to create project.



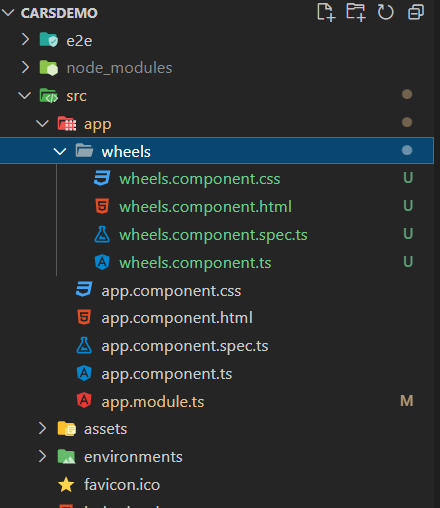
Then navigate to the project by running “*cd carsDemo*” in cmd.

**2. Creating component**

Components can be created using angular cli or manually. Use command *“ng generate wheels”*or *“ng g c wheels”*to generate a child component.



The project folder will look similar to this —



Folder structure tree opened in VS code

***Now lets see the ways to call wheels component(child) in app component(parent).***

The wheels component looks like-

import { Component, OnInit } from '@angular/core';@Component({  
selector: 'app-wheels',templateUrl: './wheels.component.html',  
styleUrls: ['./wheels.component.css']  
})export class WheelsComponent implements OnInit {constructor() { }  
ngOnInit(): void {  
}  
}

**The different ways to call child component in parent are**

**1. Using selector as HTML tag**

This is widely used way to include child component in other components and you might be already aware of this way.

In this selector of child component is used in the parent component i.e. app component in this case, as normal HTML tag.

<app-wheels></app-wheels>

**2. Using selector as Class**

Yes , It possible to include child component by using selector as class name. This way will require some changes in selector metadata of the component.ts file of child.

A “. ”(dot) is needed to be add to use selector as class name. It is similar to the way we define CSS class, e.g. *“.btn-group”*

@Component({  
**selector: '.app-wheels',**  
templateUrl: './wheels.component.html',  
styleUrls: ['./wheels.component.css'],  
})export class WheelsComponent implements OnInit {  
constructor() {}ngOnInit(): void {}  
}

Now the child component can be included in the parent using class name like this.

<div class="app-wheels"></div>

**3. Using selector as Directive**

Directive in angular makes complex task achieve like a charm, whether it is built in angular directive*(\*ngIf , \*ngFor, ngClass)* or custom directive. You can call child component in the parent component using the selector as directive on HTML element.

*Ways to call component by class and directive works same as selector as tag. It just depends on the use case required.*

To use selector as directive, change is required in the selector of the child component. Selector name need to be closed in square bracket, just like in case of property binding.

@Component({  
**selector: '[app-wheels]',**  
templateUrl: './wheels.component.html',  
styleUrls: ['./wheels.component.css'],  
})export class WheelsComponent implements OnInit {  
constructor() {}ngOnInit(): void {}  
}

Now the child component can be included in the parent using directive like this.

<div app-wheels></div>

*In case you are using*“Input” *decorator to pass some data from parent to child and the alias or name of input property is same as name of selector, then calling child component and data passing can be done using single directive only.*

I have setup the component to take value using “*Input*”decorator and logged it on page load.

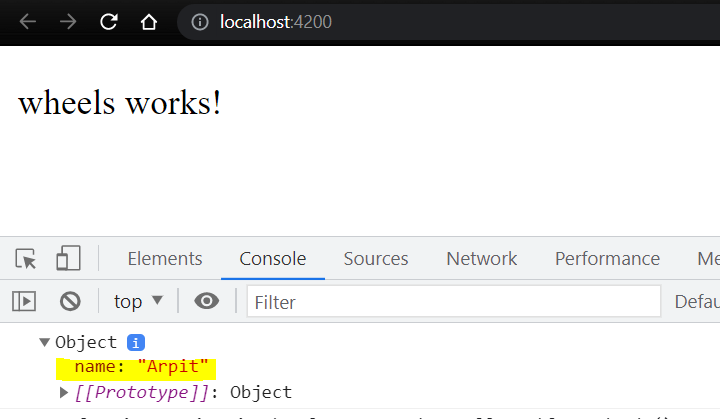
The changes in child component are—

@Component({  
selector: '[app-wheels]',  
templateUrl: './wheels.component.html',  
styleUrls: ['./wheels.component.css'],  
})export class WheelsComponent implements OnInit {  
**@Input('app-wheels') inData: any;**  
constructor() {}ngOnInit(): void {  
**console.log(this.inData);**  
}  
}

**Passing Data —** To pass data along with component calling using same directive we need to use property binding on the same directive.

*//app.component.html*  
<div **[app-wheels]="sendData"**></div>*//app.component.ts*  
sendData = {  
name: 'Arpit',  
};

This will do the trick , Output will be —



Output of the served project in chrome

*Thank you !*