

← Blog

# Simplifying ViewChild and ContentChild in Angular 10.

Reading Time: 5 minutes





Do you feel insecure, anxious and doubtful about ViewChild, ViewChildren, ContentChild and ContentChildren decorators in Angular? No worries!!! I hope you will get clarity simply by reading this Blog.

Essentially, ViewChild, ViewChildren, ContentChild, and ContentChildren, are used for component communication in Angular. Therefore, if a parent component wants access to the child component then it uses these decorators.

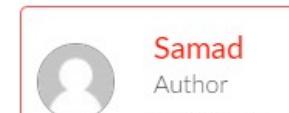
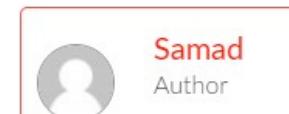
## @ViewChild()

If you want to access the following inside the parent component, use @ViewChild() decorator of Angular.

- DOM Element
- Directive
- Child Component

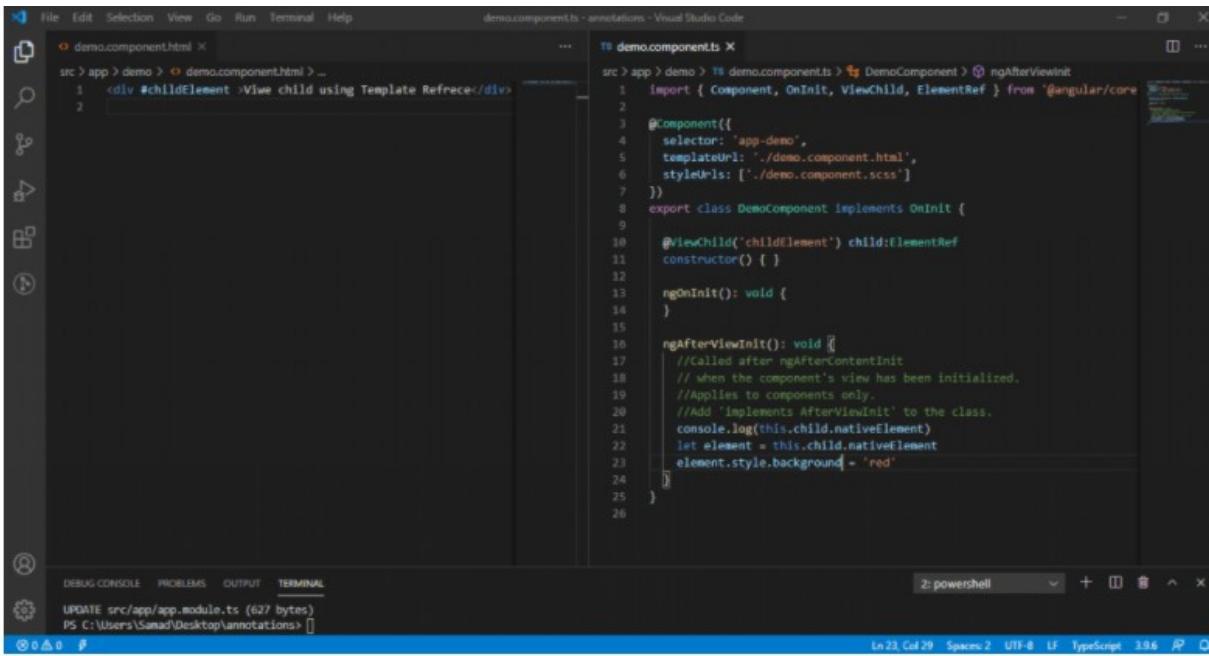
### 1. @ViewChild using DOM Element

First create a component and In this component HTML add Template reference variable like `<input #TemplateName>`. Then in the controller file import dependencies `@ViewChild(), ElementRef, AfterViewInit`. Create an instance of this Template variable. Pass the template variable name in string as an argument to the `@ViewChild()` like below mentioned.



Simplifying ViewChild and ContentChild in Angular 10





```
File Edit Selection View Go Run Terminal Help demo.component.ts - annotations - Visual Studio Code
demo.component.html X ...
src > app > demo > demo.component.html > ...
1 <div #childElement>View child using Template Referec</div>
2
...
3 import { Component, OnInit, ViewChild, ElementRef } from '@angular/core'
4
5 @Component({
6   selector: 'app-demo',
7   templateUrl: './demo.component.html',
8   styleUrls: ['./demo.component.scss']
9 })
10 export class DemoComponent implements OnInit {
11
12   @ViewChild('childElement') child: ElementRef;
13
14   ngOnInit(): void {
15
16     ngAfterViewInit(): void {
17       //Called after ngAfterContentInit
18       // when the component's view has been initialized.
19       //Applies to components only.
20       //Add 'implements AfterViewInit' to the class.
21       console.log(this.child.nativeElement)
22       let element = this.child.nativeElement
23       element.style.backgroundColor = 'red'
24     }
25   }
26 }
```

DEBUG CONSOLE PROBLEMS OUTPUT TERMINAL  
UPDATE src/app/app.module.ts (627 bytes)  
PS C:\Users\Samad\Desktop\annotations> []

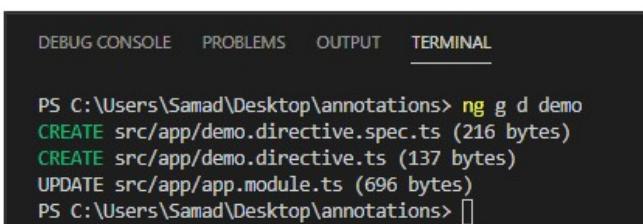
03 Dec, 2020

Simplifying ViewChild and ContentChild in Angular 10

Here childElement is the template variable and in the controller file we are changing the color of the DOM element.

## 2. @ViewChild() using Directive:-

In this method, first, we need to create one directive.



```
DEBUG CONSOLE PROBLEMS OUTPUT TERMINAL
PS C:\Users\Samad\Desktop\annotations> ng g d demo
CREATE src/app/demo.directive.spec.ts (216 bytes)
CREATE src/app/demo.directive.ts (137 bytes)
UPDATE src/app/app.module.ts (696 bytes)
PS C:\Users\Samad\Desktop\annotations> []
```

Samad

Author

03 Dec, 2020

Simplifying ViewChild and ContentChild in Angular 10

In this directive, we need to import dependencies like ElementRef and AfterViewInit and create one method in this directive that changes the color of the text.

```
ts demo.component.ts  demo.component.html  ts demo.directive.ts X
src > app > ts demo.directive.ts > DemoDirective
1 import { Directive, ElementRef } from '@angular/core';
2
3 @Directive({
4   selector: '[appDemo]'
5 })
6 export class DemoDirective {
7
8   constructor(private eleRef: ElementRef) { }
9   ngAfterViewInit(): void {
10     //Called after ngAfterContentInit
11     // when the component's view has been initialized. Applies to components only.
12     //Add 'implements AfterViewInit' to the class.
13     this.eleRef.nativeElement.style.color = 'red'
14   }
15
16   changeColor(color){
17     this.eleRef.nativeElement.style.color = color
18   }
19 }
20
```



Simplifying ViewChild  
and ContentChild in  
Angular 10

Pass the directive selector into the parent component like this, Here appDemo is the directive name.

HTML:

```
ts demo.component.ts  demo.component.html X  ts demo.directive.ts
src > app > demo > demo.component.html > ...
1 <div>Viwe child using Directive</div>
2 <p appDemo>change my color</p>
3 <div>
4   <input type="radio" name="rd" (click)="changeDirectiveColor('green')">Green
5   <input type="radio" name="rd" (click)="changeDirectiveColor('yellow')">Yellow
6   <input type="radio" name="rd" (click)="changeDirectiveColor('blue')">Blue
7 </div>
8
```



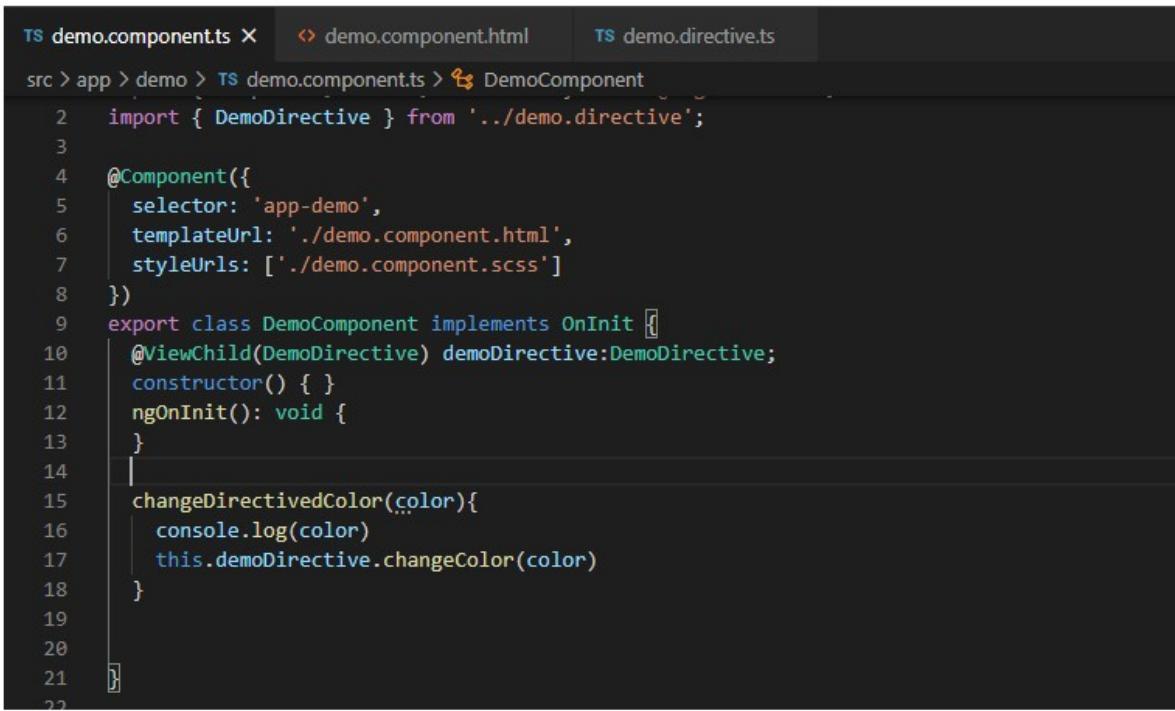
Simplifying ViewChild  
and ContentChild in  
Angular 10

Ts:

- In the parent component controller file, we need to import @ViewChild decorator and

pass the directive name as an argument to the @ViewChild() and then access the method which we created in the directive. In parent HTML we need to pass the directive selector as an attribute.

- Here demoDirective is an instance of DemoDirective and we are passing text color through changeDirectiveColor() method.



The screenshot shows a code editor with three tabs: demo.component.ts, demo.component.html, and demo.directive.ts. The demo.component.ts file contains the following code:

```
src > app > demo > TS demo.component.ts > DemoComponent
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'app-demo',
5   templateUrl: './demo.component.html',
6   styleUrls: ['./demo.component.scss']
7 })
8 export class DemoComponent implements OnInit {
9   @ViewChild(DemoDirective) demoDirective:DemoDirective;
10  constructor() { }
11  ngOnInit(): void {
12  }
13
14  changeDirectiveColor(color){
15    console.log(color)
16    this.demoDirective.changeColor(color)
17  }
18
19
20
21 }
22
```



### 3. @ViewChild() using Child Component:

Create parent and child component in the below-mentioned format.

Parent component:



The terminal window shows the command "ng g c parent" being run, followed by two "CREATE" messages indicating the creation of parent.component.html and parent.component.spec.ts files.

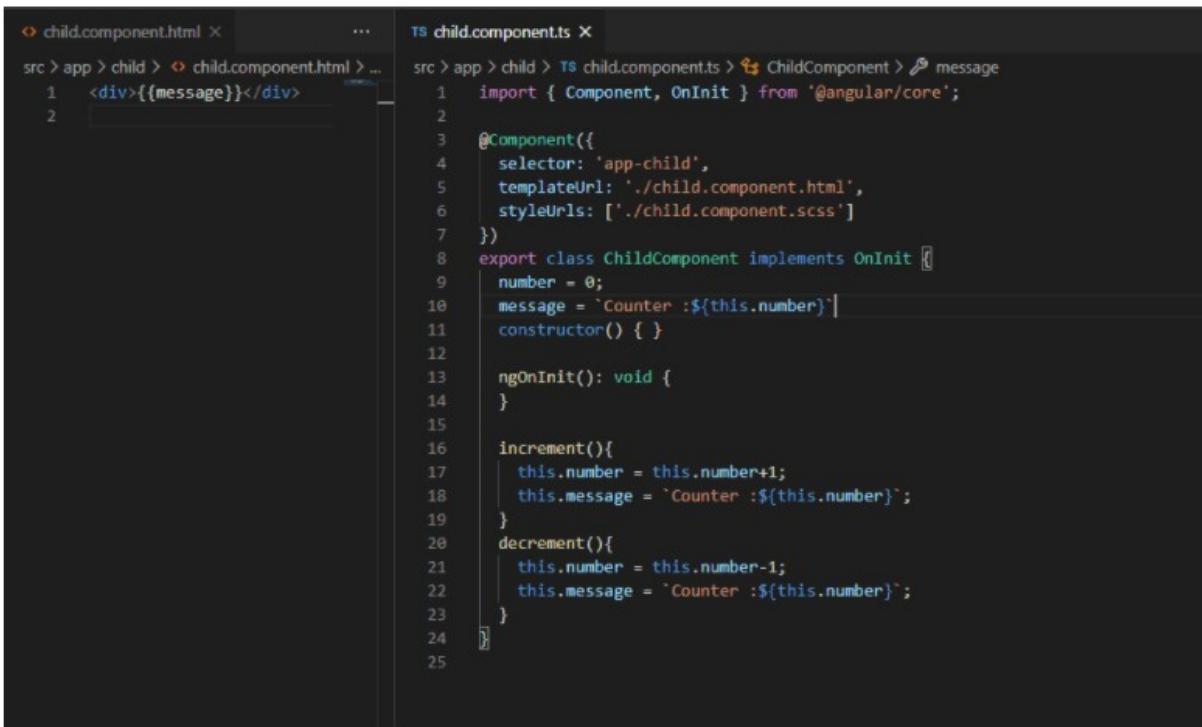
```
PS C:\Users\Samad\Desktop\annotations> ng g c parent
CREATE src/app/parent/parent.component.html (21 bytes)
CREATE src/app/parent/parent.component.spec.ts (628 bytes)
```

```
CREATE src/app/parent/parent.component.ts (226 bytes)
CREATE src/app/parent/parent.component.scss (0 bytes)
UPDATE src/app/app.module.ts (475 bytes)
PS C:\Users\Samad\Desktop\annotations>
PS C:\Users\Samad\Desktop\annotations> █
```

Child component:

```
PS C:\Users\Samad\Desktop\annotations> ng g c child
CREATE src/app/child/child.component.html (28 bytes)
CREATE src/app/child/child.component.spec.ts (621 bytes)
CREATE src/app/child/child.component.ts (272 bytes)
CREATE src/app/child/child.component.scss (0 bytes)
UPDATE src/app/app.module.ts (553 bytes)
PS C:\Users\Samad\Desktop\annotations> █
```

Create methods for increment and decrement in the child component to call in the parent component.



The screenshot shows a code editor with two files open. On the left is `child.component.html`, which contains a single line of HTML: `<div>{{message}}</div>`. On the right is `child.component.ts`, which contains the following TypeScript code:

```
src > app > child > child.component.html > ...
1  <div>{{message}}</div>
2
3  ts child.component.ts ×
src > app > child > ts child.component.ts > ChildComponent > message
1  import { Component, OnInit } from '@angular/core';
2
3  @Component({
4    selector: 'app-child',
5    templateUrl: './child.component.html',
6    styleUrls: ['./child.component.scss']
7  })
8  export class ChildComponent implements OnInit {
9    number = 0;
10   message = `Counter :${this.number}`;
11   constructor() { }
12
13   ngOnInit(): void {
14   }
15
16   increment(){
17     this.number = this.number+1;
18     this.message = `Counter :${this.number}`;
19   }
20   decrement(){
21     this.number = this.number-1;
22     this.message = `Counter :${this.number}`;
23   }
24 }
25
```



Simplifying ViewChild  
and ContentChild in  
Angular 10

Import the child component and pass the child component class name as an argument to

the `@ViewChild` decorator and Call child component methods into parent component as mentioned in below.

HTML:

```
▷ parent.component.html X
src > app > parent > ▷ parent.component.html > ⌂ div
1   <h1>ViewChild using child Component</h1>
2   <app-child></app-child>
3
4   <div>
5     <button (click)="incrementChildCount()">Increment</button>
6     <button (click)="decrementChildCount()">Decrement</button>
7   </div>
```



TS:

```
ts parent.component.ts X  ▷ parent.component.html
src > app > parent > ts parent.component.ts > ParentComponent > constructor
1  import { Component, OnInit, QueryList, ViewChildren } from '@angular/core';
2  import { ChildComponent } from '../child/child.component';
3
4  @Component({
5    selector: 'app-parent',
6    templateUrl: './parent.component.html',
7    styleUrls: ['./parent.component.scss']
8  })
9  export class ParentComponent implements OnInit {
10    @ViewChildren(ChildComponent) childComponent:QueryList<ChildComponent>;
11    constructor() { }
12    ngOnInit(): void {
13    }
14
15    ngAfterViewInit(): void {
16      //Called after ngAfterContentInit when the component's view has been initialized.Applies to components only.
17      //Add 'implements AfterViewInit' to the class.
18      console.log(this.childComponent.toArray())
19    }
20
21    incrementChildCount(id){
22      this.childComponent.toArray()[id].increment()
23    }
24    decrementChildCount(id){
25      this.childComponent.toArray()[id].decrement()
26    }
27  }
28 }
```



## @ViewChildren()

and ContentChild in  
Angular 10

- If the parent component has more than one same child component then we are using @ViewChildren decorator. Use to get the QueryList of child Component from the view DOM.
- Any time a child element is added, removed, or moved, the query list will be updated, and the changes observable of the query list will emit a new value. View queries are set before the ngAfterViewInit callback is called.

Parent component:

HTML: In parent component two same child components are there. If we need to access both we need to use @ViewChildren decorator.

The screenshot shows a code editor with two tabs: 'parent.component.ts' and 'parent.component.html'. The 'parent.component.html' tab is active, displaying the following HTML code:

```
src > app > parent > parent.component.html > div
1   <h1>ViewChild using child Component</h1>
2   <app-child></app-child>
3   <app-child></app-child>
4   <div>
5     <button (click)="incrementChildCount('0')">Increment</button>
6     <button (click)="decrementChildCount('1')">Decrement</button>
7 </div>
```



TS:

To access both child components in the parent component We need to convert the childComponent instance as an array like below mentioned this.childComponent.toArray()  
Otherwise it will always trigger the first child component.

The screenshot shows a code editor with two tabs: 'parent.component.ts' and 'parent.component.html'. The 'parent.component.ts' tab is active, showing the following TypeScript code:

```
ts parent.component.ts < parent.component.html
```

```
src > app > parent > ts parent.component.ts > ParentComponent > constructor
1 import { Component, OnInit, QueryList, ViewChildren } from '@angular/core';
2 import { ChildComponent } from '../child/child.component';
3
4 @Component({
5   selector: 'app-parent',
6   templateUrl: './parent.component.html',
7   styleUrls: ['./parent.component.scss']
8 })
9 export class ParentComponent implements OnInit {
10   @ViewChildren(ChildComponent) childComponent:QueryList<ChildComponent>;
11   constructor() []
12   ngOnInit(): void {
13
14 }
15
16 ngAfterViewInit(): void {
17   //Called after ngAfterContentInit when the component's view has been initialized.Applies to components only.
18   //Add 'implements AfterViewInit' to the class.
19   console.log(this.childComponent.toArray())
20 }
21
22 incrementChildCount(id){
23   | this.childComponent.toArray()[id].increment()
24 }
25 decrementChildCount(id){
26   | this.childComponent.toArray()[id].decrement()
27 }
28 }
```



**Samad**

Author

03 Dec, 2020

Simplifying ViewChild  
and ContentChild in  
Angular 10

## @ContentChild():

Use to get the first element or the directive matching the selector from the content DOM.

If the content DOM changes, and a new child matches the selector, the property will be updated.

Content queries are set before the ngAfterContentInit callback is called.

Does not retrieve elements or directives that are in other components' templates, since a component's template is always a black box to its ancestors.

We are creating student details by using child components and we are accessing or updating child component content using @ContentChild() decorator,@ContentChild always triggers first child component content.If we need to add content inside any component we need to use <ng-content> element.

Parent component:



**Samad**

Author

03 Dec, 2020

Simplifying ViewChild  
and ContentChild in  
Angular 10

HTML:

```

    ⚑ parent.component.html X  TS parent.component.ts  TS subjects.component.ts  ⚑ subjects.component.html  ⚑ child.com
src > app > parent > ⚑ parent.component.html > ...
1   <h1>ContentChild usage</h1>
2   <ng-container *ngFor="let student of students">
3     <app-child [student]='student'>
4       <div>Subjects:
5         <ng-container *ngFor="let subject of student.subjects">
6           <app-subjects [subjects]='subject'></app-subjects>
7         </ng-container>
8       </div>
9     </app-child>
10    </ng-container>
11

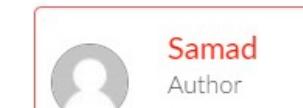
```

TS:

```

    ⚑ parent.component.html  TS parent.components.ts X  TS subjects.component.ts  ⚑ subjects.component.html  ⚑ child.component.html
src > app > parent > TS parent.components.ts > ↗ ParentComponent > ↘ students
1   import { Component, OnInit } from '@angular/core';
2
3   @Component({
4     selector: 'app-parent',
5     templateUrl: './parent.component.html',
6     styleUrls: ['./parent.component.scss']
7   })
8   export class ParentComponent implements OnInit {
9     subjects = [
10       {
11         e1:"Analog Electronics",
12         e2:"Digital Electronics",
13         e3:"VLSI"
14       }
15     ]
16     students=[
17       {
18         "studentId": 1,
19         "name": "Samad",
20         "age": 22,
21         "subjects":this.subjects
22       },
23       {
24         "studentId": 2,
25         "name": "Varma",
26         "age": 21,
27         "subjects":this.subjects
28       },
29     ]

```



```
30     "studentId": 3,
31     "name": "Mahendra",
32     "age": 21,
33     "subjects":this.subjects
34   ],
35 }
36
37   constructor() { }
38   ngOnInit(): void {}
39 }
40 }
```

03 Dec, 2020

Simplifying ViewChild  
and ContentChild in  
Angular 10

Child component:

HTML:

```
parent.html      TS parent.component.ts      <> child.component.html X      TS child.component.ts
src > app > child > <> child.component.html > ...
1   <div>
2     <div>Name:{{student.name}}</div>
3     <div>Age:{{student.age}}</div>
4     <ng-content></ng-content>
5   </div>
6   |
```

TS:

```
parent.html      TS parent.component.ts      <> child.component.html      TS child.component.ts X      TS subjects.co
src > app > child > TS child.component.ts > ChildComponent
2   import { SubjectsComponent } from '../subjects/subjects.component';
3
4   @Component({
5     selector: 'app-child',
6     templateUrl: './child.component.html',
7     styleUrls: ['./child.component.scss']
8   })
9   export class ChildComponent implements OnInit {
10     @ContentChild(SubjectsComponent) subjectsComponent:SubjectsComponent
11
12     @Input() student;
13     constructor() { }
14
15     ngOnInit(): void {
16       |
17     }
18   }
```



Samad

Author

03 Dec, 2020

Simplifying ViewChild  
and ContentChild in  
Angular 10

```
16    }
17    ngAfterContentInit(): void {
18      //Called after ngOnInit
19      //when the component's or directive's content has been initialized.
20      //Add 'implements AfterContentInit' to the class.
21      this.subjectsComponent.subjects.e1 = "Communications"
22    }
23  }
24 }
25 
```

Sub child component:

HTML:

```
parent.html   TS parent.component.ts   <> child.component.html   TS child.
src > app > subjects > <> subjects.component.html > ...
1   <div>
2     <div>Engineering 1:{{subjects.e1}}</div>
3     <div>Engineering 2{{subjects.e2}}</div>
4     <div>Engineering 3{{subjects.e3}}</div>
5   </div>
6 
```

TS:

```
parent.html   TS parent.component.ts   <> child.component.html   TS child.component.ts
src > app > subjects > TS subjects.component.ts > SubjectsComponent > ngOnInit
1   import { Component, OnInit, Input } from '@angular/core';
2
3   @Component({
4     selector: 'app-subjects',
5     templateUrl: './subjects.component.html',
6     styleUrls: ['./subjects.component.scss']
7   })
8   export class SubjectsComponent implements OnInit {
9     @Input() subjects
10    constructor() { }
11
12    ngOnInit(): void [
13    ]
14 
```



Simplifying ViewChild  
and ContentChild in  
Angular 10



Simplifying ViewChild  
and ContentChild in  
Angular 10

```
15  }
16
```

## @ContentChildren():-

- To access or update all child components content we are using @ContentChildren decorator.
- Use to get the QueryList of elements or directives from the content DOM. Any time a child element is added, removed, or moved, the query list will be updated, and the changes observable of the query list will emit a new value.
- Content queries are set before the ngAfterContentInit callback is called.
- Does not retrieve elements or directives that are in other components' templates, since a component's template is always a black box to its ancestors.

Just we need to change the child component like this importing @ContentChildrent,QueryList.

Child component:

TS:

```
parent.component.html      parent.component.ts      child.component.html      child.component.ts      subjects.component.ts
src > app > child > ts child.component.ts > ChildComponent > subjectsList
1 import { Component, OnInit, Input, AfterContentInit, ContentChildren, QueryList } from '@angular/core';
2 import { SubjectsComponent } from '../subjects/subjects.component';
3
4 @Component({
5   selector: 'app-child',
6   templateUrl: './child.component.html',
7   styleUrls: ['./child.component.scss']
8 })
9 export class ChildComponent implements OnInit,AfterContentInit {
10   @ContentChildren(SubjectsComponent) subjectsList:QueryList<SubjectsComponent>
11   @Input() student;
12   constructor() { }
13
14   ngOnInit(): void { }
```



```
14     }
15 }
16 ngAfterContentInit(): void {
17     //Called after ngOnInit.
18     //when the component's or directive's content has been initialized.
19     //Add 'implements AfterContentInit' to the class.
20     // this.subjects.subjects.e1 = "Hello"
21     this.subjectsList.toArray().forEach((component)=>{
22         component.subjects.e1 = "Digital communication"
23     })
24 }
25
26 }
27 }
```

Well,

In this blog, we discussed @ViewChild,@ViewChildren,@ContentChild, and @ContentChildren decorators in Angular. These are excellent utilities for querying child elements in views. For a parent to child component communication, these are very useful and interesting decorators. Still, if you have any questions please feel free to ask questions in the comment box and I will write another [blog](#) answering those. Thank you!!!

Share



Leave a Comment



Samad

Author

03 Dec, 2020

---

Simplifying ViewChild  
and ContentChild in  
Angular 10

Name

Email

[Leave a Comment](#)

## Recent Articles •



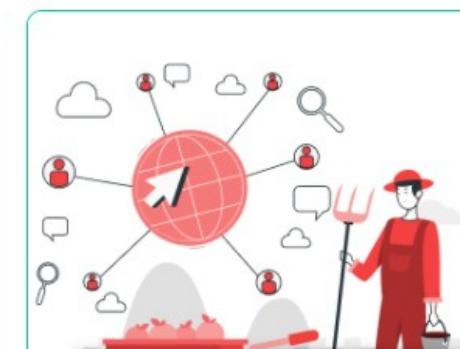
**Tips for B2B Design and Development**

- Sarath  
03 Dec, 2020



**Accessibility in design:  
How to get started?**

- Anurag  
03 Dec, 2020



**Key Benefits of IoT in  
Agriculture & Smart  
Farming**

- Sushree  
03 Dec, 2020

We can also make your ideas sticky. What's your idea?

If you are still not convinced, we have duct tape & glue!



Name

Email

Company Location (Country)

Mobile Number (Optional)

[Let's Connect →](#)



India

Vasista Bhavan, 3rd Floor,  
A.P.H.B Colony, Gachibowli,  
Hyderabad, Telangana.



India

10th floor, RMZ Latitude Commercial,  
Bellary Rd, Hebbal,  
Bengaluru.



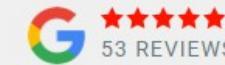
USA

Divami Inc 951 Mariners Island Blvd,  
Suite 300 PMB3005,  
San Mateo, CA 94404.



Phone

+91 (40) 6733 7033  
+1 (408) 634 8266



Email

connect@divami.com  
hr@divami.com



© Copyrights 2021 Divami Design Labs | [Privacy Policy](#)