**DOM Manipulation:**

**Reference URLs:**

[Angular - ViewChild](https://angular.io/api/core/ViewChild)

[Understanding the ViewChild and ViewChildren decorators in Angular 10 - LogRocket Blog](https://blog.logrocket.com/understanding-the-viewchild-and-viewchildren-decorators-in-angular-10/)

**ViewChild (Decorator):**

Property decorator that configures a view query. The change detector looks for the first element or the directive matching the selector in the view DOM. If the view DOM changes, and a new child matches the selector, the property is updated.

## **ViewChildren**

The@ViewChildren decorator works similarly to the @ViewChild decorator but instead of configuring a single view query, it gets a query list. From the component view DOM, it retrieves a [QueryList](https://angular.io/api/core/QueryList" \t "_blank) of child elements. This list is updated when any changes are made to the child elements.

A view query is a requested reference to a child element within a [component view](https://angular.io/guide/glossary#view) which contains metadata of the element. The scope of these decorators is limited to the component view and its embedded child views. These decorators are especially helpful in instances where being able to access and modify elements within the view in conventional ways is not possible.

For example, if a library ships with a component or directive with a public non-input or non-output property you’d like to change, these decorators would allow you to access and modify them. These decorators are also helpful in exposing providers configured in child components to inject dependencies ( like services, configuration values, etc. ) that the main component may not have access to.

## **View queries and AfterViewInit lifecycle hook**

The AfterViewInit lifecycle hook is called when the component view and its child views are completely initialized. So for immediate modifications or assignments, the best place to access view queries would be in the ngAfterViewInit callback because the view queries are already resolved and set. Trying to access them before ngAfterViewInit responds may generate undefined values. However, the @ViewChild decorator provides a static property that can be set to resolve a view query before change detection runs. We’ll cover how to use this property below.

**Template Reference Variables:**

**Reference URL:** [Template Reference Variable in AngularJS - GeeksforGeeks](https://www.geeksforgeeks.org/template-reference-variable-in-angularjs/#:~:text=Template%20Reference%20Variable%20in%20angular,Directives)

**Template Reference Variable**in angular is used to access all the properties of any element inside DOM. It can also be a reference to an Angular component or directive or a web component.

Template Reference Variable can refer to the following –

* DOM element
* Directives
* Angular Component
* Web Component

**Content Projection:**

**Reference URL:** [Ng-Content & Content Projection in Angular - TekTutorialsHub](https://www.tektutorialshub.com/angular/ng-content-content-projection-in-angular/)

[@Input](https://www.tektutorialshub.com/angular/angular-input-output-eventemitter/) decorator will pass data from the parent component to the child component. But it is only limited to data. We cannot use that technique to pass the content which includes the HTML elements, CSS, etc to the child component. To do that we have to make use of content projection.

Content projection is a way to pass the HTML content from the parent component to the child component. The child component will display the template in a designated spot. We use the ng-content element to designate a spot in the template of the child component. The ng-content also allows us to create multiple slots using the selector attribute. The parent can send different content to each slot.

## What is ng-content

The ng-content tag acts as a placeholder for inserting external or dynamic content. The Parent component passes the external content to the child component. When Angular parses the template, it inserts the external content where ng-content appears in the child component’s template

We can use content projection to create a reusable component. The components that have similar logic & layout and can be used in many places in the application.

## Multiple Projections using ng-content

The ng-content is very powerful. It allows us to create multiple slots in the template. Each slot must define a selector. You can think this as a multiple arguments to the component.

In the parent component we can create different contents and each of those contents can be projected into any of those slots depending on their selector. To implement this, we make use of the **ng-content Select attribute**. The select attribute is a CSS Selector