**@Input Decorator in Angular**

The @Input decorator in Angular is a crucial tool for enabling data flow between components. It allows you to pass data from a parent component to a child component, establishing a clear and efficient communication channel within your application's component hierarchy.

**Key Concepts:**

* **Parent-Child Communication:** The @Input decorator facilitates one-way data flow from the parent component to the child component.
* **Property Binding:** You use property binding in the parent component's template to pass data to the child component's input property.
* **Data Flow:** Changes in the parent component's data will automatically update the child component's input property, reflecting the changes in the child component's view.

**Example:**

**Child Component (child.component.ts)**

TypeScript

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

@Component({

selector: 'app-child',

template: `

<p>Child Component: {{ childData }}</p>

`

})

export class ChildComponent {

@Input() childData: string;

}

**Parent Component (parent.component.ts)**

TypeScript

import { Component } from '@angular/core';

@Component({

selector: 'app-parent',

template: `

<app-child [childData]="parentData"></app-child>

`

})

export class ParentComponent {

parentData: string = 'Hello from Parent!';

}

**Explanation:**

1. **Child Component:**
   * The @Input decorator is used on the childData property in the ChildComponent. This marks childData as an input property that will receive data from the parent component.
2. **Parent Component:**
   * The <app-child> tag represents the child component.
   * [childData]="parentData" uses property binding to pass the value of the parentData property from the parent component to the childData input property of the child component.

**Key Benefits:**

* **Component Reusability:** Child components can be made more reusable by accepting data through input properties.
* **Data Flow Control:** Enforces a clear and predictable data flow within your application.
* **Improved Maintainability:** Makes your application more modular and easier to understand and maintain.

By effectively using the @Input decorator, you can create well-structured and scalable Angular applications with efficient data communication between components.