In Angular 17 and later, you can use **Signal Inputs** to receive data from parent components using Signals. This approach leverages the power of Signals for more efficient and reactive data flow.

**Key Concepts:**

* **Signals:** A wrapper around a value that notifies consumers (like templates or other signals) when that value changes.
* **Signal Inputs:** A special type of Signal that receives data from a parent component. They are read-only, meaning you cannot directly modify their value within the child component.

**How to Use Signal Inputs:**

1. **Define the Signal Input:**

TypeScript

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

@Component({

// ...

})

export class ChildComponent {

myInput = input<string>();

}

* + input<string>() creates a Signal Input that expects a string value from the parent component.

1. **Bind the Signal Input in the Parent Component:**

TypeScript

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

@Component({

// ...

})

export class ParentComponent {

parentData = signal('Hello from Parent!');

// ...

}

1. **Use the Signal Input in the Child Component:**

TypeScript

// In ChildComponent's template

<p>Child Component: {{ myInput() }}</p>

* + The myInput() function accesses the current value of the Signal Input.

**Benefits of Signal Inputs:**

* **Improved Reactivity:** Changes in the parent component's data will automatically trigger updates in the child component, thanks to the reactive nature of Signals.
* **Enhanced Performance:** Can lead to more efficient change detection compared to traditional @Input() with ngOnChanges.
* **Cleaner Code:** Provides a more concise and declarative way to handle component interactions.

**Example:**

TypeScript

// ChildComponent

<p>Child Component: {{ myInput() }}</p>

// ParentComponent

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

**Note:**

* Signal Inputs are still under development in Angular.
* The API and behavior might evolve in future versions.

By using Signal Inputs, you can create more efficient, reactive, and maintainable Angular applications with improved data flow between components.