In Angular 17.3 and later, you can leverage **Signal Outputs** for a more streamlined and reactive approach to child-to-parent communication.

**Key Concepts:**

* **Signal Outputs:** A specialized type of Signal that allows a child component to emit data to its parent.
* **Reactivity:** Changes to the Signal Output value automatically trigger updates in the parent component, thanks to the reactive nature of Signals.

**How to Use Signal Outputs:**

1. **Declare the Signal Output in the Child Component:**

TypeScript

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

@Component({

// ...

})

export class ChildComponent {

myOutput = output<string>();

// ...

}

* + output<string>() creates a Signal Output that will emit string values.

1. **Update the Signal Output in the Child Component:**

TypeScript

// In ChildComponent's method

this.myOutput.set('Data from Child');

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

TypeScript

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

@Component({

// ...

})

export class ParentComponent {

childData = input(this.childComponent.myOutput);

// ...

}

* + input(this.childComponent.myOutput) binds the myOutput Signal from the child component to the childData Signal in the parent component.

1. **Use the Received Data in the Parent Component:**

HTML

<p>Data from Child: {{ childData() }}</p>

**Benefits of Signal Outputs:**

* **Improved Reactivity:** Automatic updates in the parent component when the child's Signal Output changes.
* **Enhanced Performance:** Can lead to more efficient change detection.
* **Conciseness:** Simpler syntax compared to traditional @Output() with EventEmitter.
* **Better Integration with Signals:** Seamlessly integrates with the Signals API for a more consistent and reactive experience.

**Note:**

* Signal Outputs are a relatively new feature.
* The API and behavior might evolve in future versions of Angular.

By utilizing Signal Outputs, you can create more efficient, reactive, and maintainable Angular applications with improved communication between components.