Sending Data from Child to Parent:

Child Component (child.component.ts):

In the child component, you define an `EventEmitter` with the `@Output` decorator. This emitter is used to send data from the child component to the parent component.

```
import { Component, Output, EventEmitter } from '@angular/core';

@Component({
  selector: 'app-child',
  template: `
    <button (click)="sendDataToParent()">Send Data to Parent</button>
  `,
})
export class ChildComponent {
  @Output() sendData = new EventEmitter<string>();

  sendDataToParent() {
    // Emit an event with the data to be sent to the parent
    this.sendData.emit('Data from child to parent');
  }
}
```

Parent Component (parent.component.ts):

In the parent component, you bind to the child component's `sendData` event using `(sendData)="receiveDataFromChild($event)"`. The method `receiveDataFromChild` is then called whenever the child component emits the event.

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

@Component({
  selector: 'app-parent',
  template: `
    <app-child (sendData)="receiveDataFromChild($event)"></app-child>
    <p>Data from child: {{ dataFromChild }}</p>
  `,
})
export class ParentComponent {
  dataFromChild: string;

  // Method to handle the received data from the child component
  receiveDataFromChild(data: string) {
    this.dataFromChild = data;
  }
}
```

Sending Data from Parent to Child:

Child Component (child.component.ts):

In the child component, you use the `@Input` decorator to define a property (`dataFromParent`) that can receive data from the parent component.

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

@Component({
  selector: 'app-child',
  template: `
    <p>Data from parent: {{ dataFromParent }}</p>
  `,
})
export class ChildComponent {
  @Input() dataFromParent: string;
}
```

Parent Component (parent.component.ts):

In the parent component, you bind the property `dataFromParent` on the child component using `[dataFromParent]="dataToChild"`. This allows you to pass data from the parent to the child.

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

@Component({
  selector: 'app-parent',
  template: `
    <app-child [dataFromParent]="dataToChild"></app-child>
    <button (click)="sendDataToChild()">Send Data to Child</button>
  `,
})
export class ParentComponent {
  dataToChild = 'Data from parent to child';

  // Method to update the data sent to the child component
  sendDataToChild() {
    this.dataToChild = 'Updated data from parent';
  }
}
```

In summary:

- The `@Output` decorator in the child component allows you to emit events to the parent component.
- The `@Input` decorator in the child component allows you to receive data from the parent component.

These mechanisms facilitate communication between parent and child components in Angular, enabling you to build more modular and reusable components.