**Components Interaction**

**And EventEmitter**

In app.component.html

<p> Test {{message}} </p>

<app-new-cmp (childEvent)="message=$event" [name1]="userApp"></app-new-cmp>

To Send Data to child component

**<app-new-cmp (childEvent)="message=$event" [name1]="userApp" ></app-new-cmp>**

* userApp is a local variable of app.component.ts
* And Passing to newCmp

Getting Value in newCmp.ts

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

...

@Input('name1') Uname;

And print {{Uname}}

To Send Data to Parent component

**<app-new-cmp (childEvent1)="message=$event" [name1]="userApp" ></app-new-cmp>**

* message is a local variable of app.component.ts
* And Getting from newCmp

Getting Value in newCmp.ts

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

...

@Output() public childEvent1 =new EventEmitter();

…

SendDataToParent(){

console.log('SendDataToParent');

this.childEvent1.emit('Msg From Child');

}

Call from Html

<button (click)="SendDataToParent()">SendDataToParent</button>

Show in App.component.ts

<p> Test {{message}} </p>

<app-new-cmp (childEvent)="message=$event" [name1]="userApp"></app-new-cmp>

**EventEmitter: it’s store data a data asyn and syn to pass or get and it don’t wait for execute code next line it work Parallely**

**Some notification example using Eventemitter**

**Store a data**

**…**

**import { Injectable,EventEmitter } from '@angular/core';**

**import { message } from './class/message'**

**...**

**public newMsgReceive:EventEmitter<message>;**

**...**

**this.newMsgReceive=new EventEmitter()**

**…**

**notifyService(msg:string,type:string){**

**console.log("notifyService");**

**let newmsg=new message(msg,type);**

**this.newMsgReceive.emit(newmsg);**

**}**

**…**

**Create a message class to set data type**

**export class message{**

**constructor(public msg:string, public type:string){}**

**}**

**Getting Data from Eventemitter**

**...**

**import { NoficationSService } from '../service/nofication-s.service';**

**...**

**constructor(public notifyS:NoficationSService) {**

**this.notifyS.newMsgReceive.subscribe(data=>{**

**this.notifyMethod(data);**

**});**

**}**

**…**

**notifyMethod(data){**

**console.log("NoficationSService sdsd",data);**

**setTimeout(()=>{**

**console.log("NoficationSService settimeout");**

**//this.notifyS.notifyService("sdsd","sdsds");**

**},2000);**

**}**

**...**