Communication between Events in LWC

LWC dispatches standard DOM events. Components can also create and dispatch custom events.

We use events to communicate up in the containment hierarchy. Use custom events to pass data from child to parent component.

The DOM events system is a programming design pattern that includes these elements.

1. An event name, called a type
2. A configuration to initialize the event
3. A JavaScript object that emits the event

In Lightning web components, CustomEvent provides a more consistent experience across browsers, including Internet Explorer. It requires no setup or boilerplate, and it allows you to pass any kind of data via the **detail** property, which makes it flexible.

To communicate down the component containment hierarchy, pass properties to the child via HTML attributes, or call its public methods. To communicate between components, use Lightning message service or a publish-subcribe utility.

# Ways to Communicate between components

There are 3 ways in which two components can communicate between each other and exchange data.

1. Communication using a public method -> Used to communicate from Parent to child. When we need to send data from Parent to child we use this approach.
2. Communication using a custom event -> Used to communicate from child to Parent. When you need to pass data from child to Parent we need to use custom events.
3. Use Lightning Message Service(LMS) or Pub-Sub Model to communicate between events which are not in the same containment hierarchy. Components which are not related to each other, we can use LMS or PubSub to communicate b/w them.

## **Parent to Child Communication in LWC**

When we want to send data from Parent component to child component then we use Parent to Child communication.

This done using a **@api** decorator by making a public property in child component which will hold the value passed from Parent.

On the Parent side, we need to include the child component name using namespace and kebab case notation along with the property name.

Public Properties are read only in the child component. We cannot modify the data received from Parent.

If the value is changed in the Parent Component then it will be automatically passed to the child component.

For ex consider the below code: -

<template for:each={meetingRoomsInfo} for:item="room">

<li key={room.roomName}>

<c-meeting-room meeting-room-info={room}></c-meeting-room>

</li>

</template>

The highlighted Code above is the way of including child component in the Parent Component.

# **SLOTS**

When we want to pass additional markup from Parent to Child we use Slots.

If we want to pass only data then use public properties(@api).

Slots are of 2 types:

1. Names slots 🡪 In this slot is having a name. And there can be multiple names slots in a component.
2. Un-named slots 🡪 Only one unnamed slot is allowed in LWC. If we try to make more than one unnamed slot then all the markup will go inside a single slot. So, its better to use only one unnamed slot in a component.

Ex:

**Parent Component Markup:**

<c-meeting-room meeting-room-info={room}>

<p slot="addtionalMarkup">Hello World</p> 🡪 Named Slot

<p>UnNamed Slot</p> 🡪 Unnamed Slot

</c-meeting-room>

Code written between the child component tag will be sent a slot.

**Child Component Markup:**

<slot name="addtionalMarkup"></slot>

<slot></slot>

# **Calling a Child’s Public method from Parent**

We can call a child’s public method from Parent and get a value in return.

**Parent.Js Code snippet:**

const childComponent = this.template.querySelector('c-child-component');

const returnedValue = childComponent.selectCheckbox(this.value);

this.msg = returnedValue;

The highlighted method(selectCheckbox) is a child’s public method which we are calling from parent and passing a value as a parameter.