Parent Child and Child Parent Communication

When multiple Lightning web components compose an app, we often want those components to share information. How we communicate from one component to another depends on whether and how the components are related. A component inside another component creates a parent-child relationship. How a parent communicates with a child is different from how a child communicates with its parent.

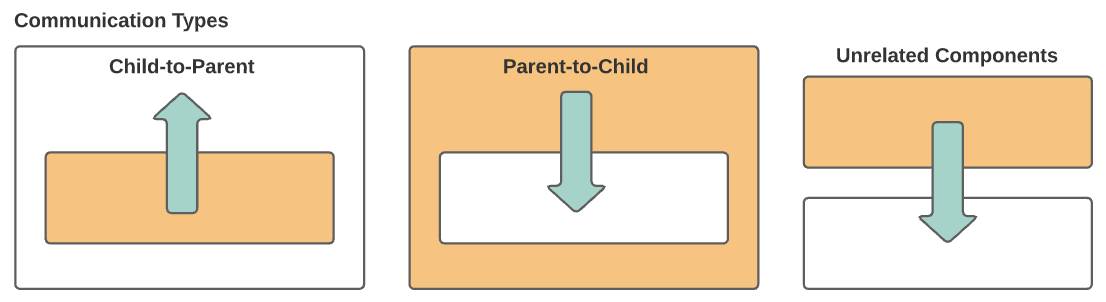
Why we go for Parent and Child?

We create small reusable components separately and install them inside another component. This concept is called as component composition.

1. No need to re write the code again.
2. Easy to call other components.
3. Reduce the code size

Build Lightning web components that communicate with one another.

* Communicate from a child component to its parent component.
* Communicate from a parent component to a child component.
* Communicate from a component to an unrelated component.



Parent to Child

To enable communication from a parent component to a child component, the child exposes a property or function to make it public. Then the parent can update the child's public property or call the child's public function. Primitive data types, objects and arrays

* Public Property -The @api decorator in the child component exposes a property, making it public, so that the parent component can update it.

Syntax:

@api childProperty;

//Passing this in the parent html

child-property

* Call a Public Function-The method should be make public @api.Using this.template.queryselector we can call the child component functions from parent components

Syntax:

@api childMethod ()

{

alert (‘Hi I am child method’);

}

* Public Properties with getter-setter - These are the component attributes with @api get and set functions. The parent component can set the data in a similar way it sets for public properties. The difference is that you can process data sent to/from the parent component. The actual data is stored in a different private attribute from the component. This approach gives you flexibility so that you can alter the data received from parents in the child components.

Syntax

personObject ;

@api

get counter() {

return this.personObject;

}

set counter(value) {

this. personObject = value;

}

Child to Parent

A child component dispatches a custom event that triggers an update in the parent component.

1.Calling parent method using simple event

2.Event with data

3.Event bubbling

1.Calling parent method using simple event

1.Create Custom Event

2.Dispatch Event

Syntax:

const event = new CustomEvent(‘eventName’,

details:{ value : ‘Message to pass to parent’

})

this.dispatchEvent(event);

Parent and Child in html

childComponent -> if this our file name

parentComponent.html

<template>

…….

…….

<c-child-component></c-child-component>

</template>