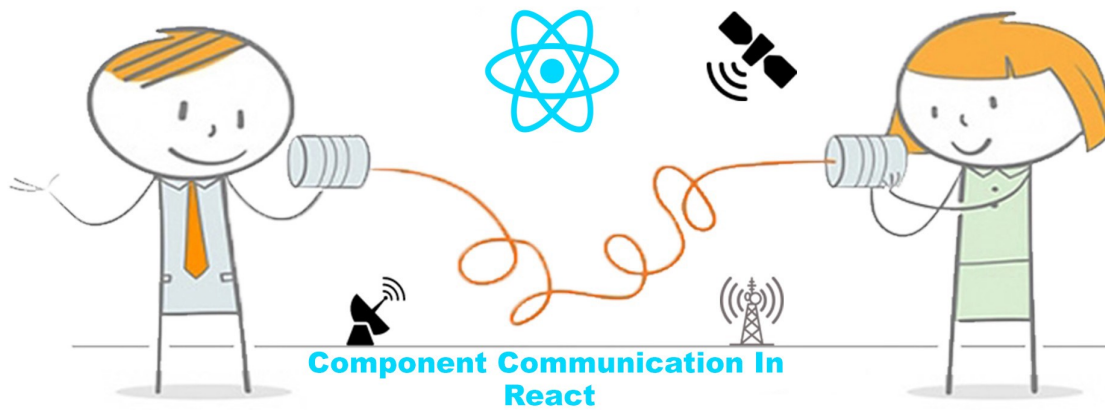


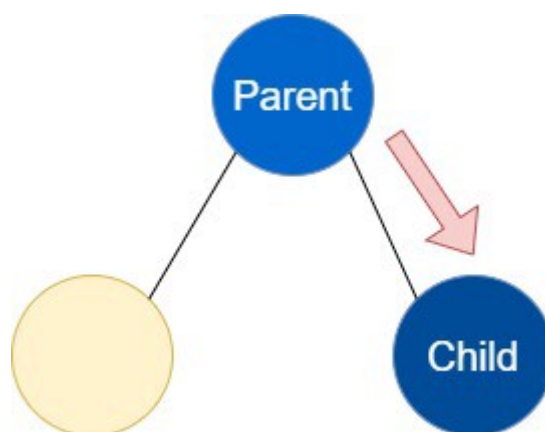
Interview Questions



Q1. How do React Components Communicate with each other?
(Google)

Answer:

1. Parent to Child:



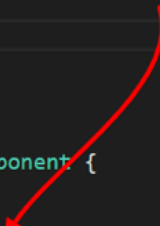
Using Props:

In this type of communication, a parent passes the data to the child by adding an extra attribute in the child component declaration.

```
import React, { Component } from 'react'

export default class ParentComponentProps extends Component {
  constructor() {
    super();
    this.state = {name: 'Mark'};
  }
  render() {
    return (
      <div>
        <ChildComponent name="Sumeet" /> {/* For Static Value */}
        {/* Or */}
        <ChildComponent name={this.state.name} /> {/* For Dynamic Value */}
      </div>
    )
  }
}

class ChildComponent extends Component {
  render() {
    return (
      <h1>Hi {this.props.name}</h1>
    )
  }
}
```

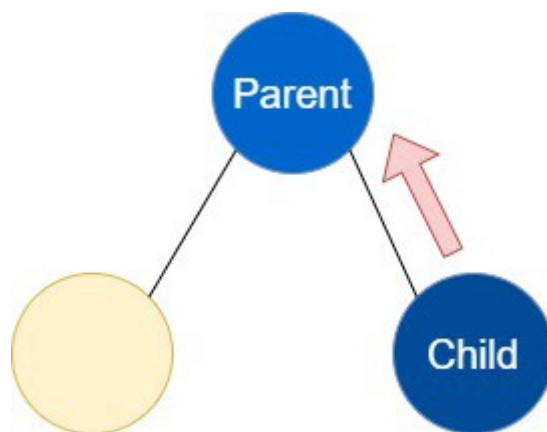


Consider the above example, we have added an extra attribute called “**name**” in the child component declaration. This attribute can be accessed inside the child component via **props**.

Props provide **one-way communication** from a parent to a child. Any changes in the values of an attribute/argument in the prop will directly reflect in the child.

But what if you want to call a method inside the **child component** when the value is changed.

2. Child to Parent



Data from a child can be passed to the parent using a callback. This can be achieved by using the following steps.

- Create a callback method in parent and pass it to the child using props.
- Child can call this method using “this.props.[yourCallbackName]” from child and pass data as argument.

```
export default class UsingCallback extends Component {
  constructor() {
    super();
    this.state = {name: 'Sumeet'};
  }
  setName = (name) => {
    this.setState({name});
  }
  render() {
    return (
      <div>
        <h1>Hi {this.state.name}!</h1>
        <ChildComponent setName={this.setName}/>
      </div>
    )
  }
}

class ChildComponent extends Component {
  sendParent = () => {
    this.props.setName('John');
  }
  render() {
    return(
      <button onClick={this.sendParent}>Click Me</button>
    )
  }
}
```



The diagram illustrates the flow of data and function passing between two React components. A green arrow originates from the `setName` method in the `UsingCallback` parent component and points to the `setName={this.setName}` prop in the `<ChildComponent>` JSX element. A yellow arrow originates from the `sendParent` method in the `ChildComponent` and points to the `this.props.setName('John')` line, demonstrating how the child calls the parent's method via props.

In the above example, we have created a method called “setName” and passing it as props to the child. In the Child component on click of a button we are calling parent method with argument.