```
return (
    <div>
        <Item name1 = {response[0].itemName}></Item>
    </div>
);
}
```

Index of object

⇒ But what happens when we do this

```
<Item name1 = {response[0].itemName} I am Rishabh </Item>
```

This will not be visible.

To make this visible, we need to add this in item.js file.

```
<div>
    <p className="Rishabh">{itemName}</p>
    {props.children}
</div>
```

* REACT Events

Handeling events with React elements is very similar to handling events on DOM elements.

React has the same events as HTML: click, change, mouseover etc. (All the events start with on like onClick, etc)

→ React events are written in camelCase syntax:

onClick instead of onclick.

→ React event handlers are written inside curly braces:

onClick={shoot} instead of onclick="shoot()".

Q. What happens when we use onClick={shoot()} prop instead of onClick={shoot} prop?

In this case the function shoot() will be called and it will call function even if we ~~no~~ did'nt clicked on button.

## * Hooks

A hook is a special function that lets you "hook into" React features. For example, useState is a hook that lets you add React state to function components.

## * USESTATE

The React useState Hook allows us to track state in a function component.

This hook is used for storing variables that are part of your application's

state and will change as the user interacts with your website.

The state change happens per component instance basis.

State is used when we want to manage changing data in an application.

Ex. we need to import useState first.

```
import {useState} from "react";

function FavColor() {

    const [color, setColor] = useState("");
```

~~Existing~~ color is our current state

It is the function that is used to update our state

HW useState is a asynchronous hook, it will wait for the component to finish its cycle, re-render, and then it will update the state.

Q. Why const is used and still the value changes?

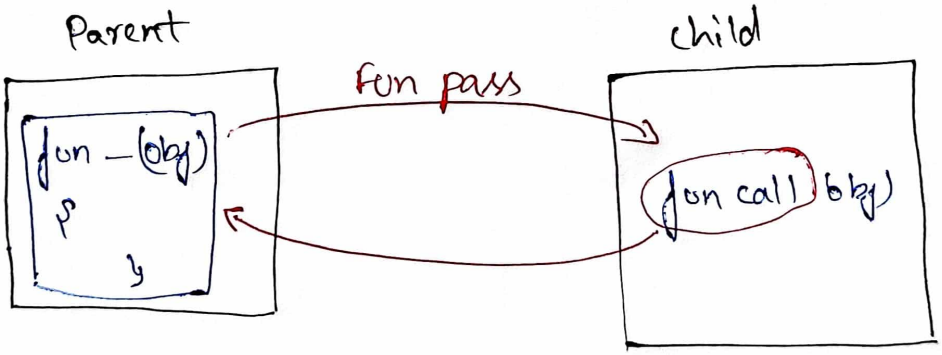
When a component is rendered, the function is executed again, creating a new 'color' variable, which has nothing to do with the previous variable.

* ~~onClick~~ onChange

This event in REACT detects when the value of an input element changes.

* Lifting State Up

Sharing state is accomplished by moving it up to the closest common ancestor (parent) of the components that need it.




Step 1: Function will pbe passed from parent toe child.

Step 2: Function will be called in child component with input parameter.

Step 3: Value will be accessed.