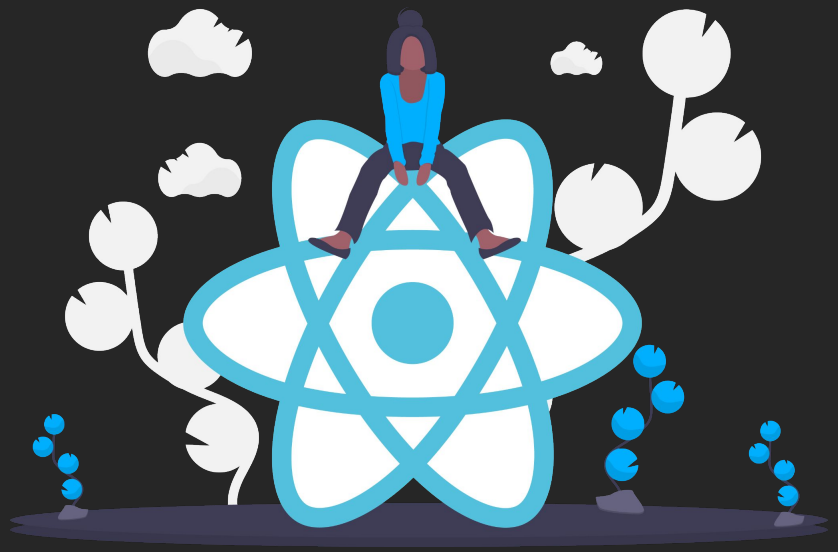
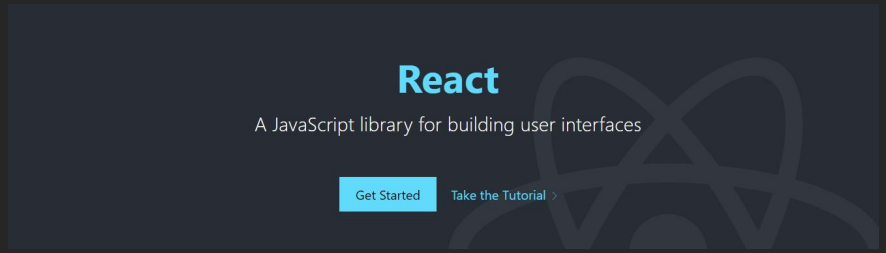


React เบื้องต้น

รู้จักกับ React

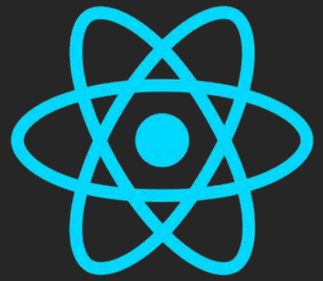


React คือ JavaScript Library ที่ใช้สำหรับสร้าง User Interface (UI) หรือหน้าจอของแอปพลิเคชั่น ถูกพัฒนาขึ้นโดย Facebook

Framework vs Library

Framework

Library



Framework vs Library

Framework

ข้อดี

- มีรูปแบบที่ชัดเจนเป็นแบบแผนเหมาะกับงานที่ทำเป็นทีมต้องเขียนตามรูปแบบที่กำหนดไว้
- มีเครื่องมือทุกอย่างพร้อมให้เราใช้งานได้เลย

ข้อเสีย

- ไม่มีความยืดหยุ่น
- ปรับปรุงแก้ไขการทำงานได้ยาก

Library

ข้อดี

- เลือกเครื่องมือหรือสิ่งที่ต้องการมาใช้งานในระบบของเราได้เลย
- มีความยืดหยุ่นสูง

ข้อเสีย

- ต้องทำทุกอย่างด้วยตนเอง

แนวคิดของ React

React จะใช้นำมาสร้างหน้าเว็บ โดยที่สามารถแบ่งการแสดงผลหน้าเว็บออกเป็นส่วนย่อยหลายๆส่วนได้ โดยที่ไม่ต้องเขียนหน้าเว็บเก็บไว้ในไฟล์เดียว เพื่อให้ง่ายต่อการจัดการ แล้วค่อยนำส่วนย่อยดังกล่าวมาประกอบกันในภายหลัง เราจะเรียกองค์ประกอบที่แบ่งออกเป็นส่วนย่อย ๆ นี้ว่า

“Component”

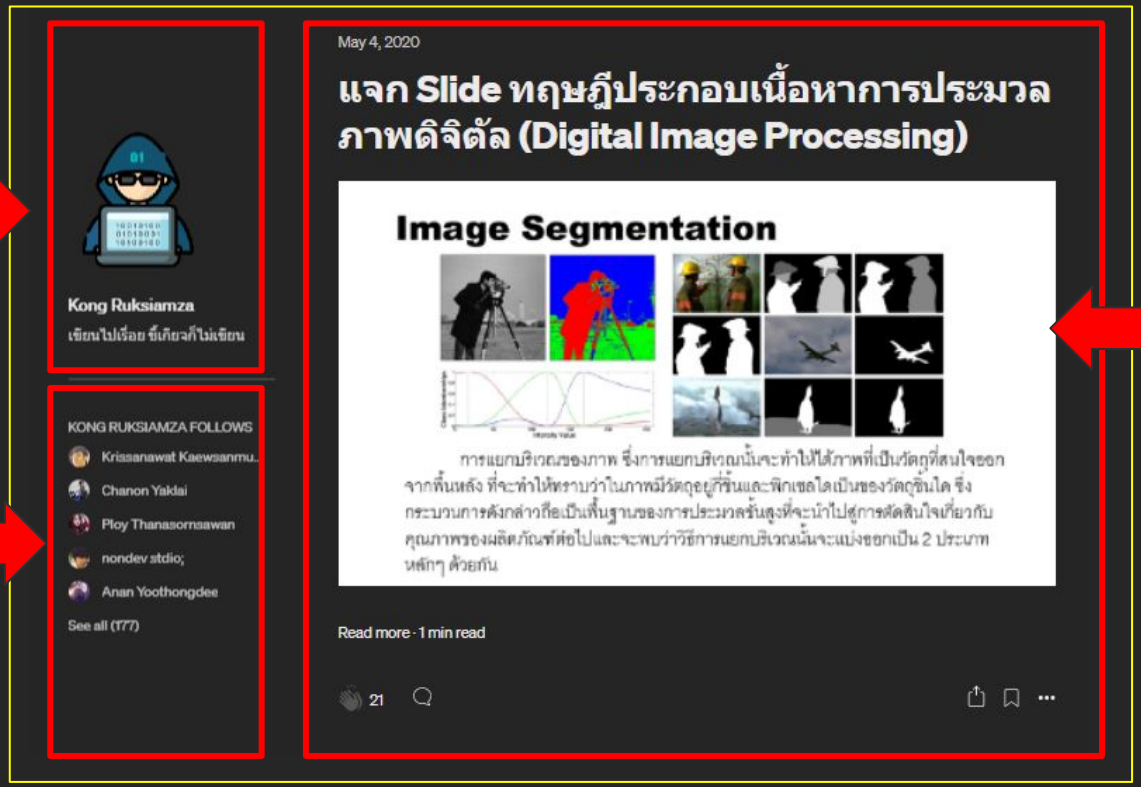
ข้อดีของการสร้าง Component คือ สามารถที่จะออกแบบแล้วนำกลับมาใช้งานในภายหลังได้ โดยที่ไม่ต้องเสียเวลาเขียนใหม่

แนวคิดของ React

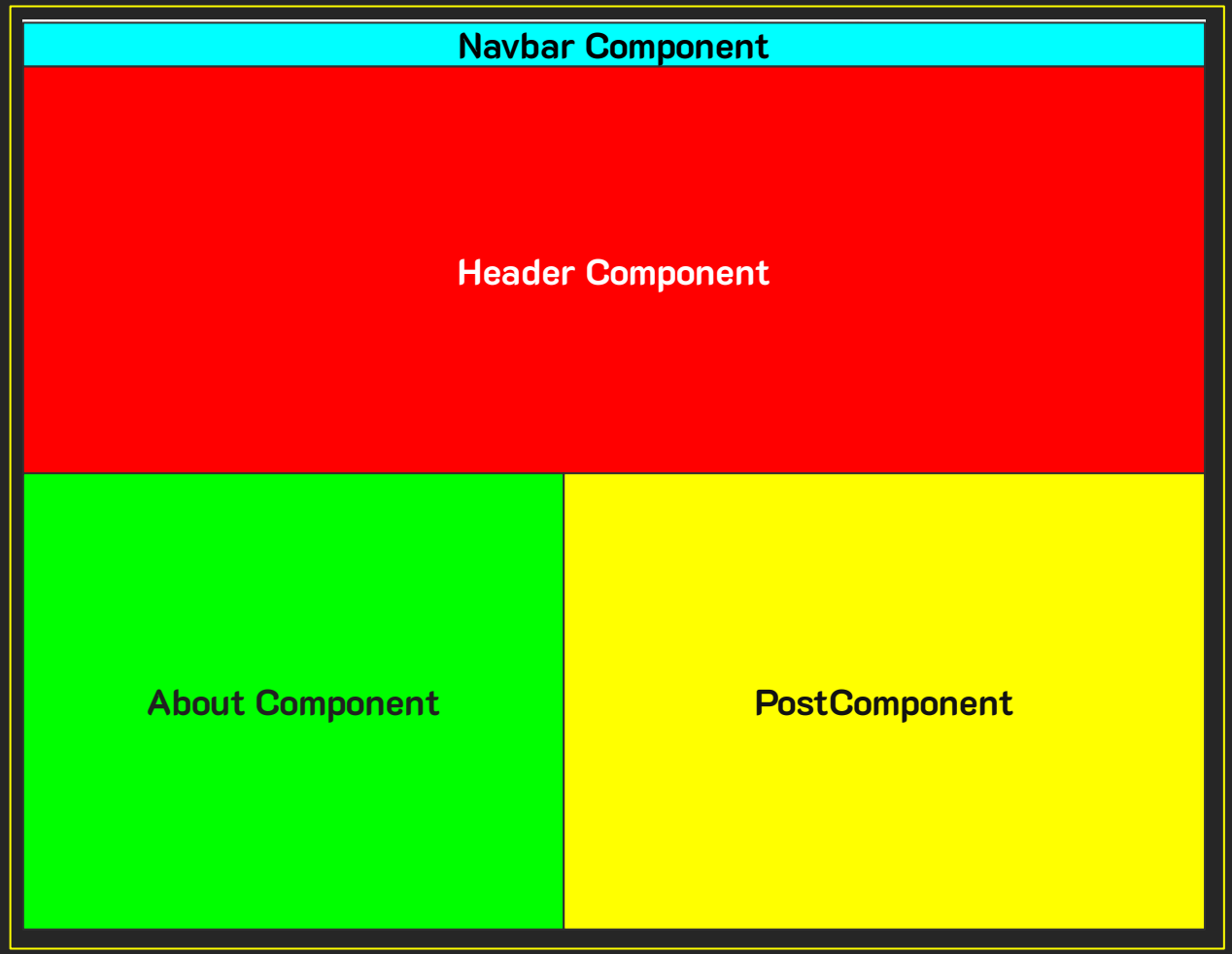
แนวคิดของ React

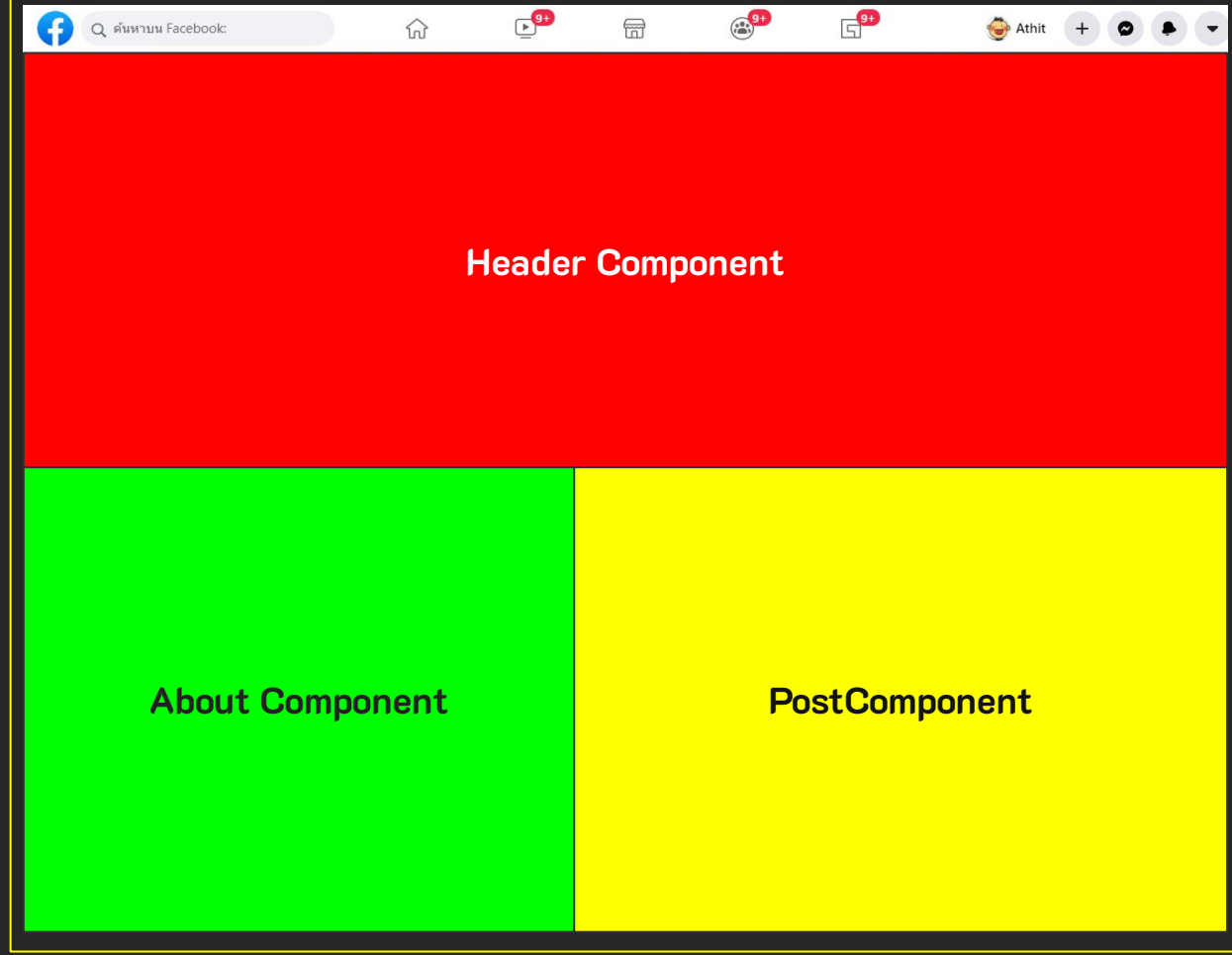
Profile Compoment

Follows Component



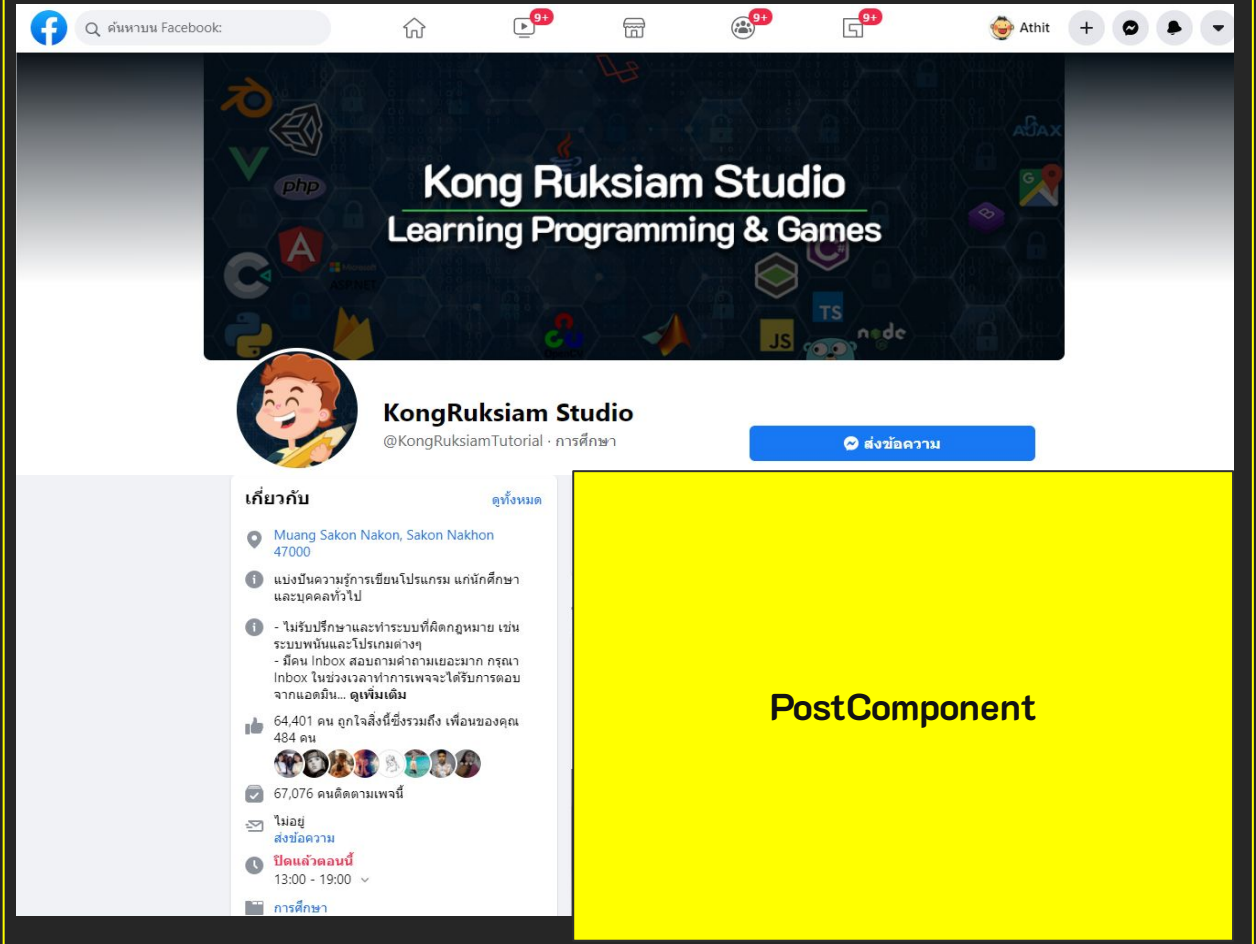
Content Component

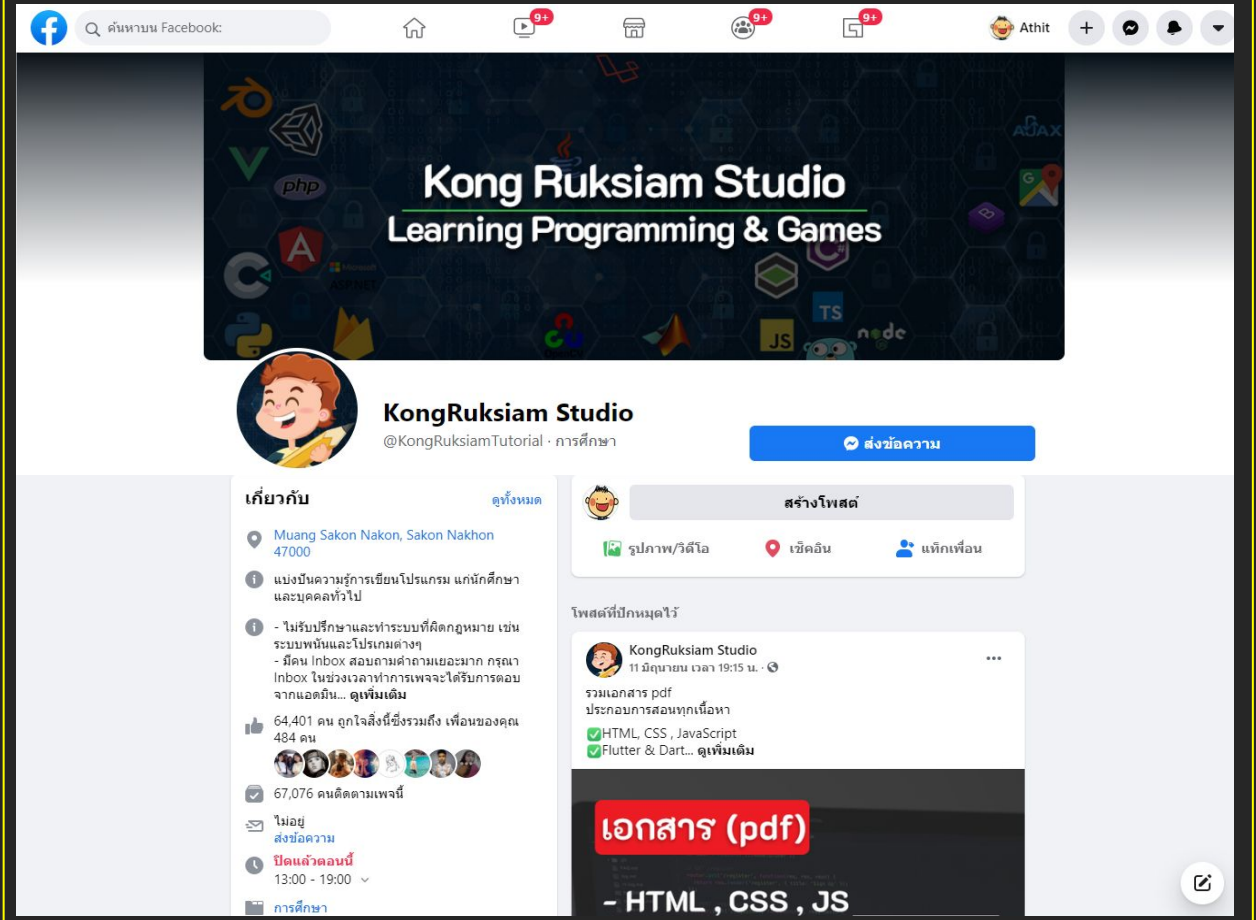

Navbar Component
Header Component
About Component
PostComponent


ค้นหาบน Facebook
9+
9+
9+
Athit
Header Component
About Component
PostComponent

About Component

PostComponent


ค้นหาบน Facebook:
Athit
Kong Ruksiam Studio
Learning Programming & Games
KongRuksiam Studio
@KongRuksiamTutorial · การศึกษา
ส่งข้อความ
เกี่ยวกับ
ดูทั้งหมด
Muang Sakon Nakon, Sakon Nakhon 47000
แบ่งปันความรู้การเขียนโปรแกรม แก่นักศึกษาและบุคคลทั่วไป
- ไม่รับปรึกษาและทำระบบที่ผิดกฎหมาย เช่น ระบบพนันและโปรเกมต่างๆ
- มีคน Inbox สอบถามคำถามเยอะมาก กรุณา Inbox ในช่วงเวลาทำการเพจจะได้รับการตอบจากแอดมิน... ดูเพิ่มเติม
64,401 คน ถูกใจสิ่งนี้ซึ่งรวมถึง เพื่อนของคุณ 484 คน
67,076 คนติดตามเพจนี้
ไม่อยู่
ส่งข้อความ
ปิดแล้วตอนนี้
13:00 - 19:00
การศึกษา
PostComponent


ค้นหาบน Facebook:
Athit
Kong Ruksiam Studio
Learning Programming & Games
KongRuksiam Studio
@KongRuksiamTutorial · การศึกษา
ส่งข้อความ
เกี่ยวกับ
ดูทั้งหมด
Muang Sakon Nakon, Sakon Nakhon 47000
แบ่งปันความรู้การเขียนโปรแกรม แก่นักศึกษาและบุคคลทั่วไป
- ไม่รับปรึกษาและทำระบบที่ผิดกฎหมาย เช่น ระบบพนันและโปรเกมต่างๆ
- มีคน Inbox สอบถามคำถามเยอะมาก กรุณา Inbox ในช่วงเวลาทำการเพจจะได้รับการตอบจากแอดมิน... ดูเพิ่มเติม
64,401 คน ถูกใจสิ่งนี้ซึ่งรวมถึง เพื่อนของคุณ 484 คน
67,076 คนติดตามเพจนี้
ไม่อยู่
ส่งข้อความ
ปิดแล้วตอนนี้
13:00 - 19:00
การศึกษา
สร้างโพสต์
รูปภาพ/วิดีโอ
เช็คอิน
แท็กเพื่อน
โพสต์ที่ปักหมุดไว้
KongRuksiam Studio
11 มิถุนายน เวลา 19:15 น. ·
รวมเอกสาร pdf
ประกอบการสอนทุกเนื้อหา
✅HTML, CSS , JavaScript
✅Flutter & Dart... ดูเพิ่มเติม
เอกสาร (pdf)
- HTML , CSS , JS

แนวคิดของ React

สำหรับการสร้าง Component นั้นก็จะมีรูปแบบการเขียนคล้ายๆกับ HTML เพียงแต่ว่า React จะใช้ส่วนที่เรียกว่า JSX ในการเขียนซึ่งมีความคล้ายคลึงกับ HTML มาก โดยการเขียน JSX นั้น คือการเขียนในไฟล์ JavaScript ไม่ใช่ในไฟล์ HTML ซึ่งจะลงรายละเอียดในหัวข้อของ Component

Imperative Programming

เป็นรูปแบบการเขียนโปรแกรมแบบเดิมที่ใช้ใน JavaScript ในการบอกว่าเว็บของเราต้องการอยากจะสร้างอะไร แล้วให้กระทำอะไร เช่น

```
const btnEl = document.createElement('button')
btnEl.innerHTML = 'Send Data'
document.body.appendChild(btn)
```

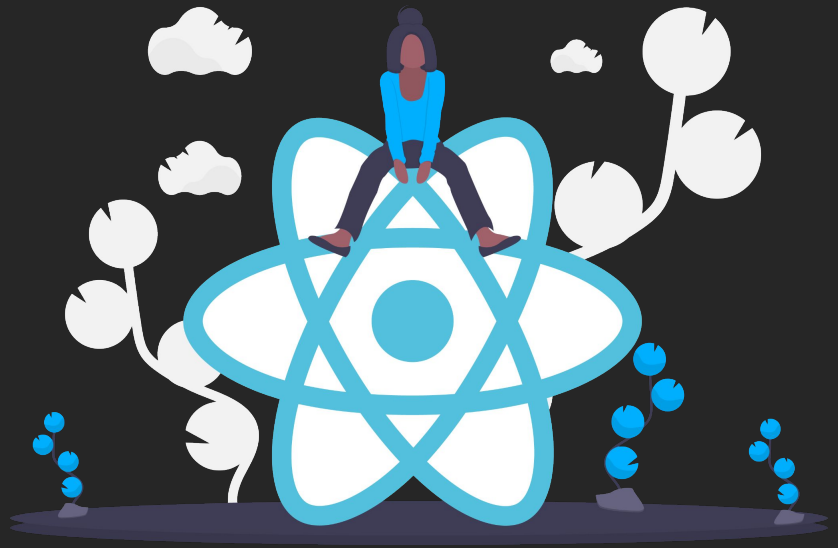
Declarative Programming

เป็นรูปแบบการเขียนโปรแกรมที่ได้รับความนิยมในปัจจุบันเนื่องจากทำให้โค้ดมีความกระชับอ่านแล้วเข้าใจง่าย มีความหมายที่ตรงตัว

```
<Button>Send Data</Button>
```

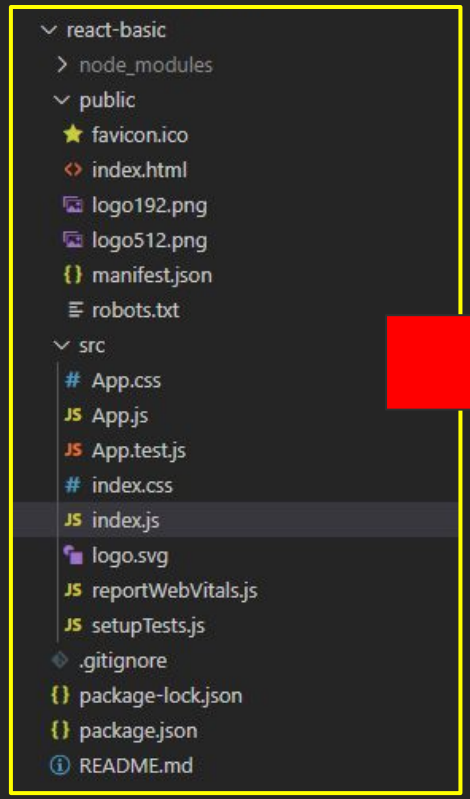
แนวคิดของ React

- Component คือ ส่วนประกอบต่างๆที่ถูกนำมาประกอบรวมกันเป็นหน้าเว็บ (คล้ายๆ สร้าง tag ขึ้นมาใช้เอง เช่น <KongRuksiam/> เป็นต้น)
- State คือ ข้อมูลที่อยู่ภายใน Component แต่ละตัว
- Props คือ ข้อมูลที่ถูกส่งจาก Component (แนวคิดมาจากการกำหนด Attribute หรือ Properties ใน HTML)



เตรียมเครื่องมือให้พร้อม ก่อนลุย React !!!

โครงสร้างโปรเจค



- **node_modules** - เก็บโมดูลหรือไลบรารี่ที่ทำงานภายในโปรเจคของเรา
- **public** - เก็บไฟล์ public ต่างๆ เช่น รูปภาพ และ index.html เป็นต้น
- **src** - สำหรับเก็บ Component หรือโครงสร้างของแอพพลิเคชั่น
- **package.json** - เก็บข้อมูลต่างๆรวมถึง pakcage ที่จะใช้ทำงานภายในโปรเจค

โครงสร้างโปรเจค

/Public/Index.html

ไฟล์ที่ใช้สำหรับการแสดงผลข้อมูลบนเบราเซอร์ จะรันผ่านไฟล์ index.html สิ่งที่เราสนใจใน index.html ก็คือ คำสั่งส่วนที่อยู่ใน <body> ตรงส่วนของ

```
<body>
  <noscript>You need to enable JavaScript to run this app.</noscript>
  <div id="root"></div>
  <!--
    This HTML file is a template.
    If you open it directly in the browser, you will see an empty page.

    You can add webfonts, meta tags, or analytics to this file.
    The build step will place the bundled scripts into the <body> tag.

    To begin the development, run `npm start` or `yarn start`.
    To create a production bundle, use `npm run build` or `yarn build`.
  -->
</body>
```

โครงสร้างโปรเจค

/Src/App.js

เป็นไฟล์หลักในการรันแอพพลิเคชั่นขึ้นมา โดยนำส่วนประกอบต่างๆมาประกอบกันแล้วนำไปแสดงผลในบราวเซอร์

index.html App.js

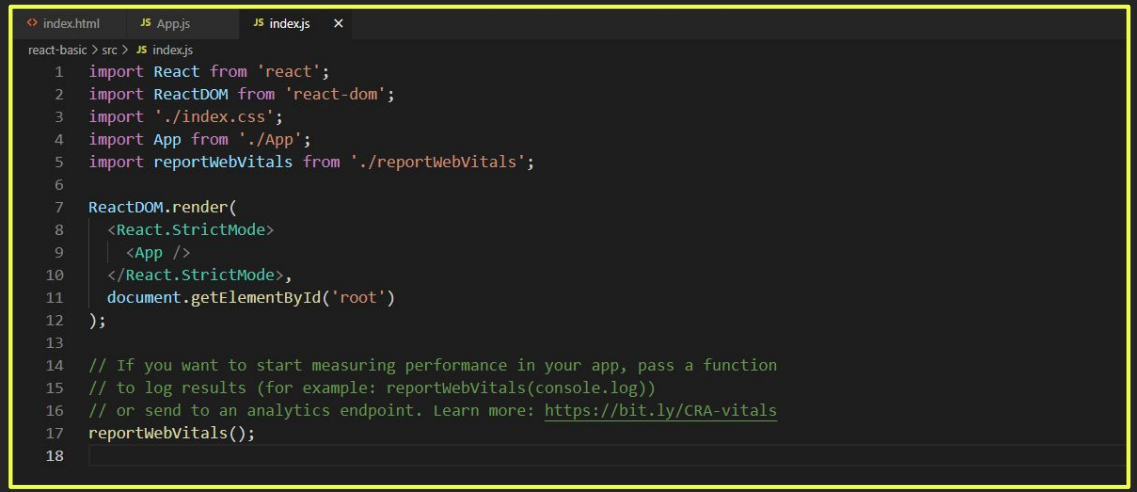
eact-basic > src > App.js > ...

```
1  import logo from './logo.svg';
2  import './App.css';
3
4  function App() {
5    return (
6      <div className="App">
7        <header className="App-header">
8          <img src={logo} className="App-logo" alt="logo" />
9          <p>
10           Edit <code>src/App.js</code> and save to reload.
11         </p>
12         <a
13           className="App-link"
14           href="https://reactjs.org"
15           target="_blank"
16           rel="noopener noreferrer"
17         >
18           Learn React
19         </a>
20       </header>
21     </div>
22   );
23 }
24
25 export default App;
```

โครงสร้างโปรเจค

/Src/index.js

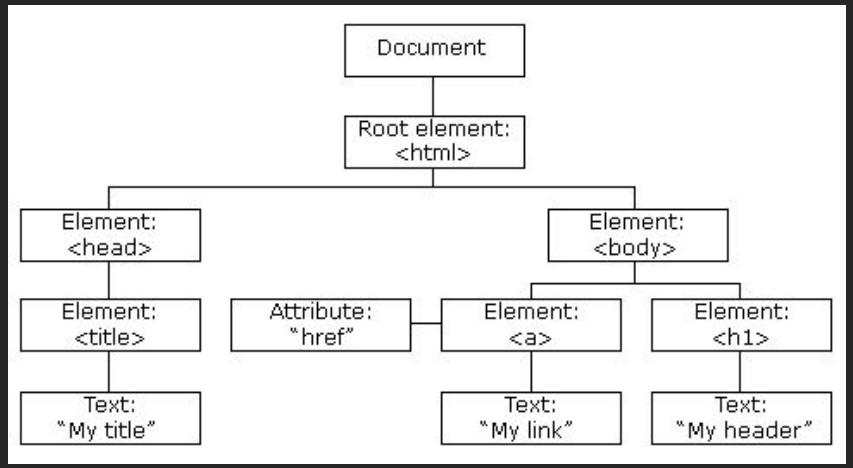
เป็นไฟล์สำหรับเชื่อมการทำงานระหว่าง App.js และ Index.html



ReactDOM Render HTML

เมื่อหน้าเว็บโหลดเสร็จเรียบร้อย Web Browser มันจะสร้าง DOM ของหน้านั้นขึ้นมา
โดยมอง HTML เป็นโครงสร้างต้นไม้ (ก้อน Object) หรือ**เรียกว่า DOM (Document Object Model)**

```
<html>
<head>
        <title>My title</title>
</head>
 <body>
<a href="#">My link</a>
<h1>My header</h1>
</body>
</html>
```



Tag ต่าง ๆ ใน HTML จะเรียกว่า Element

ReactDOM Render HTML

ReactDOM เป็นไลบรารี่เหมือนกับ React ทำหน้าที่เฉพาะในการจัดการกับ DOM และใช้เฉพาะกับ React เท่านั้น

คำสั่ง ReactDOM.render() จะทำการสร้าง DOM (Virtual DOM) ที่มีลักษณะของโครงสร้างต้นไม้ แล้วนำโครงสร้างดังกล่าวใส่ลงไปยัง DOM จริงๆ (Real DOM) ซึ่งเป็นวิธีการในการ Render JSX ออกมาแสดงผลทางหน้าจอ ยกตัวอย่าง เช่น

```
ReactDOM.render(<p>Hello World</p>, document.getElementById('root'));
```

ReactDOM Render HTML

```
ReactDOM.render(<p>Hello World</p>, document.getElementById('root'));
```

หรือ

```
const data = (
    <div>
        <h1>KongRuksiam</h1>
    </div>
);
ReactDOM.render(data, document.getElementById('root'));
```

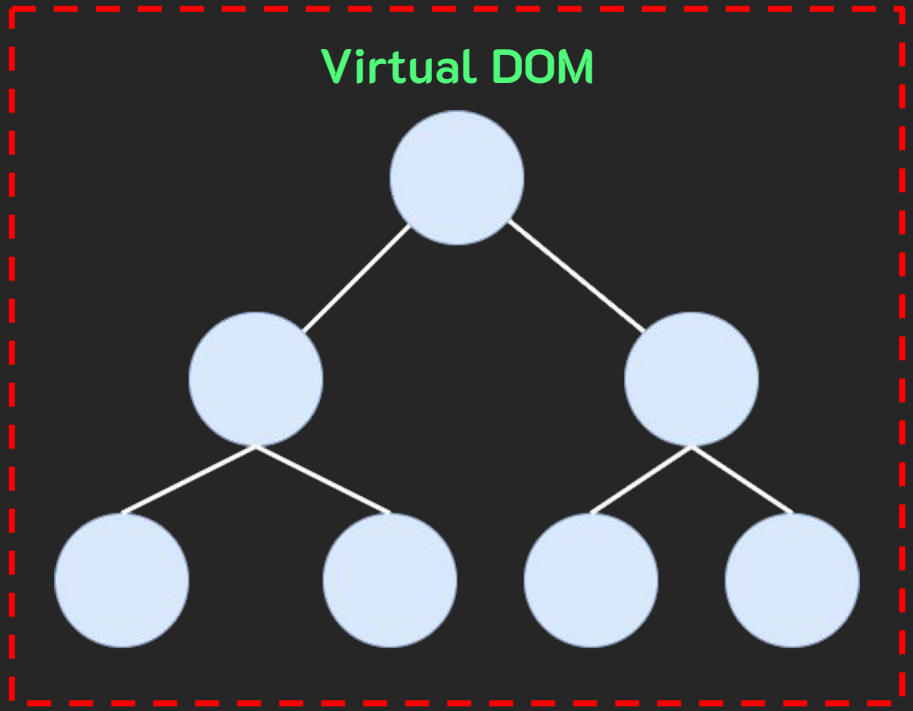
Virtual DOM

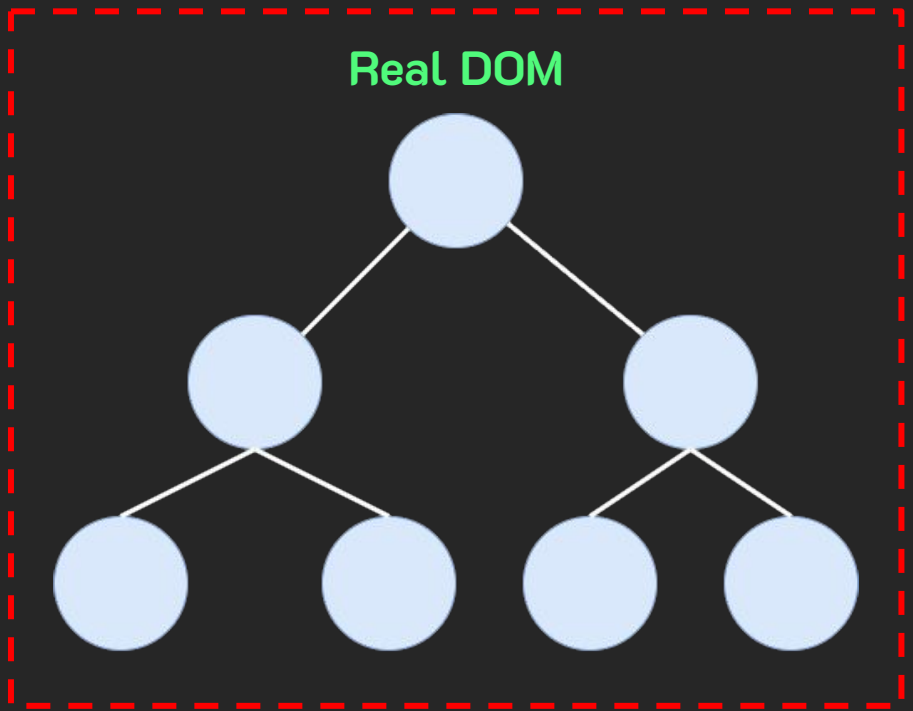
การทำงานของ Virtual DOM

Virtual DOM มีลักษะคล้ายๆกับ DOM ใน HTML โดย Virtual DOM จะทำการคัดลอก DOM จริง (Real DOM) มาเก็บลง Memory ถ้ามีการเปลี่ยนแปลงเกิดขึ้นที่ Component ก็จะอัพเดตเฉพาะ Component ที่เปลี่ยนแปลงเท่านั้น โดยไม่จำเป็นต้องอัพเดต DOM ใหม่หมดทั้งหน้า ทำให้เกิดการทำงานได้อย่างรวดเร็ว

***ถ้าใช้ (DOM แบบปกติจะ Refresh ทั้งหน้าเพื่ออัพเดตหน้าเว็บที่เปลี่ยนแปลงไป)

Virtual DOM

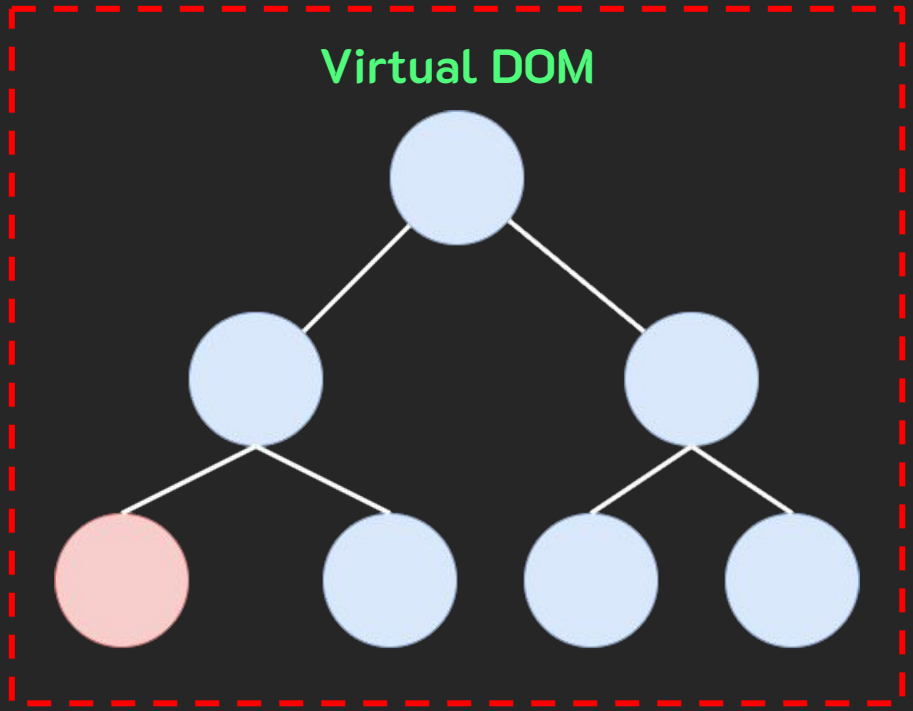


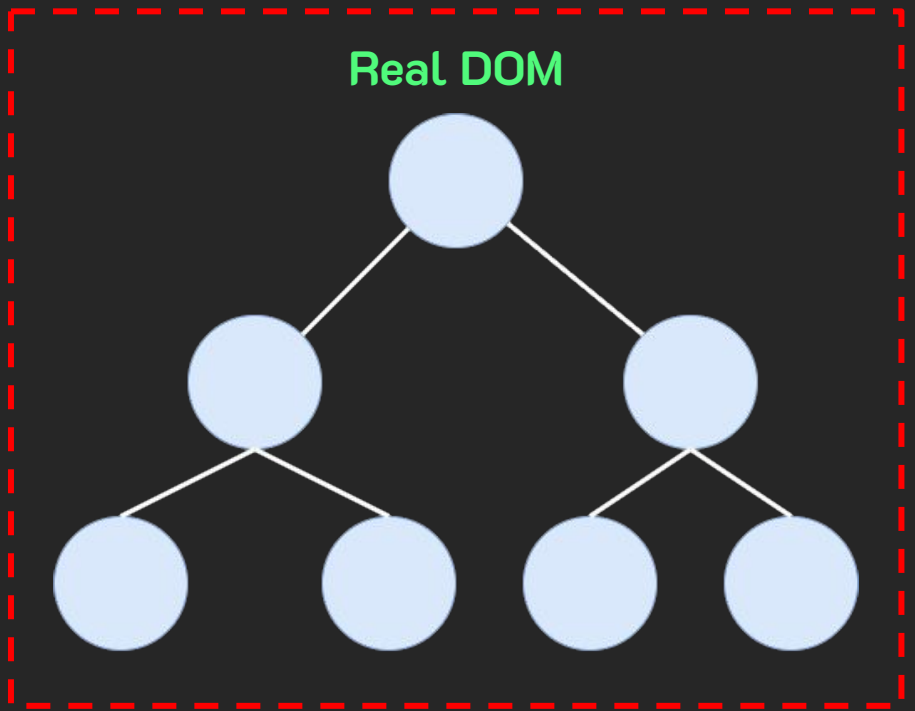


https://www.youtube.com/c/KongRuksiamOfficial/

https://www.facebook.com/KongRuksiamTutorial/

Virtual DOM

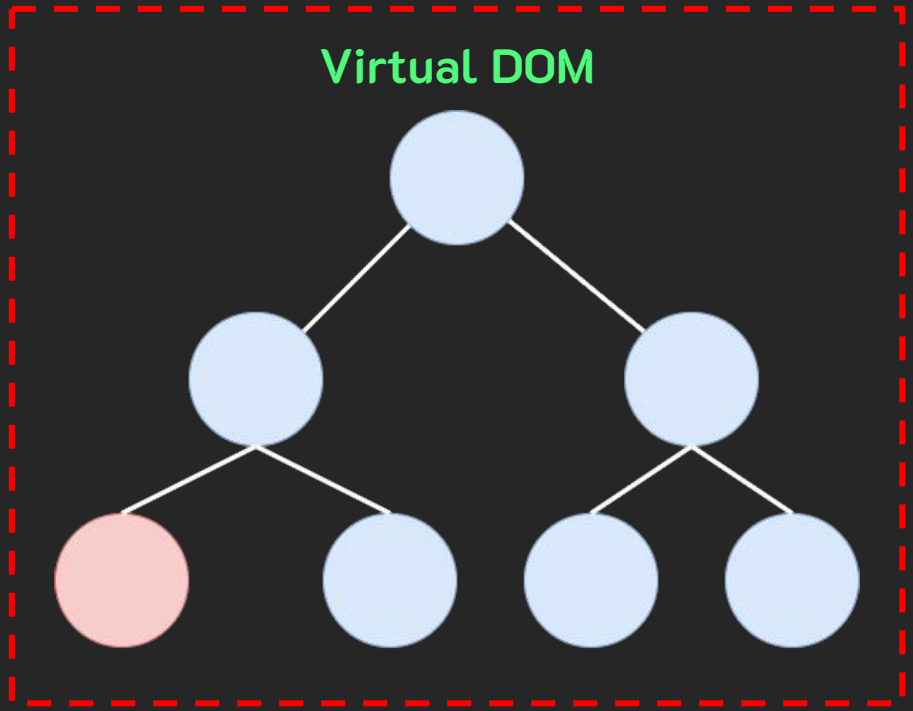


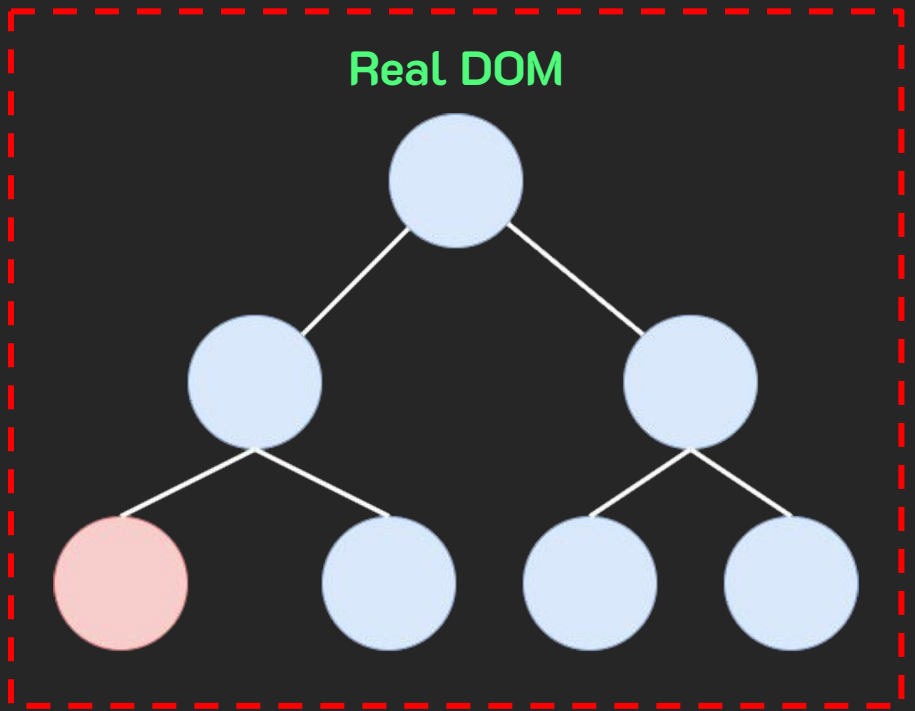


https://www.youtube.com/c/KongRuksiamOfficial/

https://www.facebook.com/KongRuksiamTutorial/

Virtual DOM

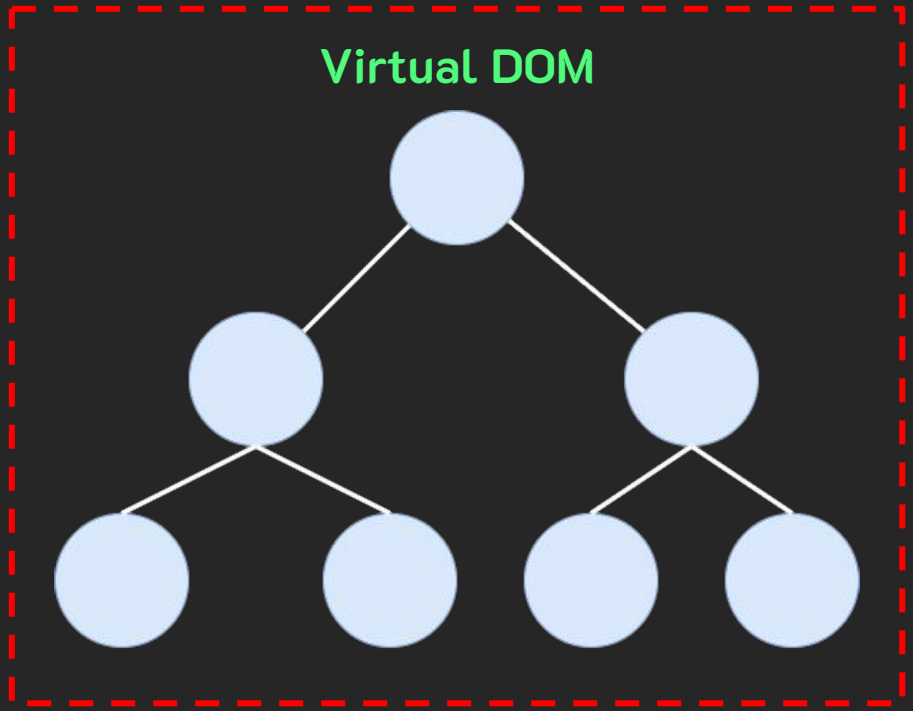


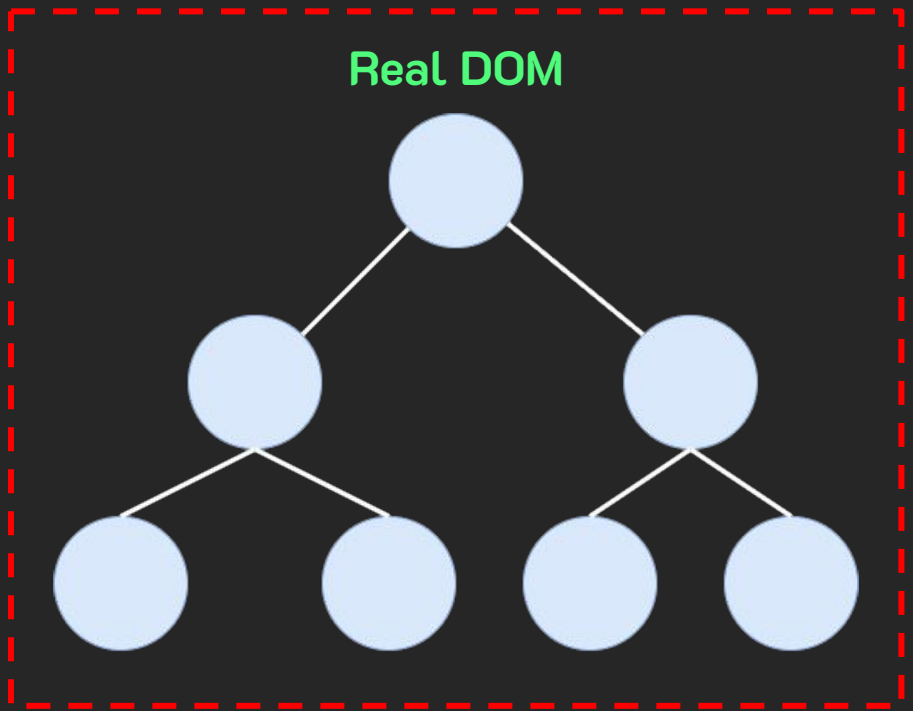


https://www.youtube.com/c/KongRuksiamOfficial/

https://www.facebook.com/KongRuksiamTutorial/

Virtual DOM





https://www.youtube.com/c/KongRuksiamOfficial/

https://www.facebook.com/KongRuksiamTutorial/

การสร้าง Component

ใน React การจะสร้างหน้าเว็บขึ้นมาได้นั้นจะใช้สิ่งที่เรียกว่า Component โดย Component คือ ส่วนย่อยแต่ละส่วนของหน้าเว็บที่นำมาประกอบกัน โดยจะเขียนด้วย ภาษา JavaScript เพื่อทำหน้าตาแต่ละส่วนของเว็บหรือส่วนย่อยนั้นๆ มาแทนที่จะเขียนใน HTML แทน

แต่ใน React จะไม่เขียน HTML ใน HTML แต่จะเขียนใน JavaScript แทน ซึ่งจะใช้ส่วนที่เรียกว่า JSX (JavaScript XML) ที่ทำให้ใส่ HTML เข้าไปเขียนใน JavaScript ได้ ดังนั้น React คือ การสร้างหน้าเว็บด้วยภาษา JavaScript ที่มี HTML แทรกอยู่ ก็คือ JSX นั่นเอง !

รูปแบบการสร้าง Component

สามารถสร้างได้ 2 รูปแบบ

- Class Component
- Functional Component

โดยทั้งคู่จะ Return HTML ออกมาและเขียน JSX ด้านในส่วนของการ Return

Functional Component

สร้างแบบ Component ที่ง่ายที่สุด คือ เป็นรูปแบบฟังก์ชั่นโดยสร้างฟังก์ชันธรรมดาๆ และ Return HTML ออกมา โดย กำหนดให้ตัวอักษรตัวแรกของชื่อฟังก์ชั่นเป็นตัวพิมพ์ใหญ่เสมอ เช่น

```
function HelloComponent() {
    return <h1>สวัสดี React </h1>;
}
ReactDOM.render(<HelloComponent />, document.getElementById('root'));
```

Class Component

สร้าง Class ที่ extends มาจาก React.Component และ Return HTML ออกมา แต่ถ้าสร้างเป็นแบบ Class จะมีความสามารถในการใช้งานมากกว่าแบบ Functional

```
class HelloComponent extends React.Component {
  render() {
    return <h1>สวัสดี React</h1>;
  }
}
ReactDOM.render(<HelloComponent />, document.getElementById('root'));
```

External Component

การสร้าง Component ไว้เป็นไฟล์ด้านนอกที่มีนามสกุล .js จากนั้นก็นำเข้ามาทำงานในหน้าเว็บ

```
function HelloComponent(){
    return <h1>สวัสดี Component แบบ External</h1>
}
export default HelloComponent
```

React JSX

รูปแบบการเขียน JSX

- สามารถเขียนใน / section / article / Fragment / <> ก็ได้

และต้องมีการกำหนด Tag เปิด - ปิด ทุกครั้ง

- การใส่ Class Style ที่เป็น Attribute ใน JSX จะมีการกำหนด className แทน class เนื่องจากคำว่า class เป็น keyword

React JSX

รูปแบบการเขียนแบบ div

```
function HelloComponent(){
        return (
                <div>
                        <div><h3>สวัสดี React </h3></div>
                </div>
        );
}
```

React JSX

รูปแบบการเขียนแบบ section / article

```
function HelloComponent(){
        return (
                <section>
                        <article><h3>สวัสดี React </h3></article>
                </section>
        );
}
```

React JSX

รูปแบบการเขียนแบบ React.Fragment

```
function HelloComponent(){
        return (
                <React.Fragment>
                        <div><h3>สวัสดี React </h3></div>
                </React.Fragment>
        );
}
```

React JSX

รูปแบบการเขียนแบบ Fragment แบบ <>

```
function HelloComponent(){
        return (
                <>
                        <div><h3>สวัสดี React </h3></div>
                </>
        );
}
```

React Props

Props (Properties) คือ ตัวแปรที่สามารถส่งเข้าไปใน Components ได้ ผ่านการกำหนด Attribute ส่งผลให้ Component แต่ละตัวสามารถรับค่าจากด้านนอกเข้าไปทำงานได้

React Props

<ชื่อคอมโพเนน ชื่อพร็อพเพอร์ตี้ =ค่าที่กำหนดให้พร็อพเพอร์ตี้/>

Keys

Keys เป็นพร็อพเพอร์ตี้ที่อยู่ใน JSX โดย Keys จะมีค่าไม่ซ้ำกัน กำหนดขึ้นเพื่อให้ React นำไปเช็คว่ามี Component ใดบ้างที่เปลี่ยนแปลงค่าไปในการ Render หน้าเว็บ

React PropsType (Validation)

เป็นการประกาศรูปแบบหรือชนิดของ Props ที่ส่งเข้าไปทำงานใน Component เช่น กำหนดชนิดข้อมูลของ Props หรือ บังคับให้ต้องกำหนดค่า Props ทุกครั้งที่มีการเรียกใช้งาน Component เป็นต้น

React PropsType (Validation)

ติดตั้ง prop-types

npm install prop-types

React PropsType (Validation)

การใช้งาน

```
ชื่อคอมโพเนน.propsTypes = {
    ชื่อพร็อพเพอร์ตี้ : รูปแบบของพร็อพเพอร์ตี้
}
```

State

State คือ ตัวแปรที่เก็บข้อมูลภายใน Component คล้ายๆกับ Props

แต่การใช้งาน Props นั้น ข้อมูลจะไม่สามารถเปลี่ยนแปลงค่าได้

แต่ State สามารถทำได้

ฉะนั้น ถ้าต้องการให้ข้อมูลภายในแอพสามารถเปลี่ยนแปลงค่าได้ ในระหว่างรันแอพก็จะใช้ State ซึ่งรูปแบบเดิมจะเขียนภายใน Class Component

Stateless vs Stateful

- Stateless คือ State ที่ไม่มีการเปลี่ยนแปลงค่า
- Stateful คือ State ที่มีการเปลี่ยนแปลงค่า

React Hook

Hook เป็นฟีเจอร์ที่มีอยู่ใน React เวอร์ชั่น 16.8 เป็นต้นไป ใช้สำหรับจัดการเกี่ยวกับ State หรือฟีเจอร์ต่างๆที่อยู่ภายใน React โดยที่ไม่ต้องเขียนใน Class Component แต่จะใน Functional Component แทน

การใช้งาน React Hook

1. เขียนใน Functional Component
2. เขียนในส่วนของ Top-Level ของ Function
   (อย่าเขียนใน Condition , Loop , Nested Function นอกจากจะเขียน Custom Hook)

ตัวอย่าง React Hook

1. useState
2. useEffect
3. useReducer
4. useContext
5. useMemo
6. useCallBack

useState

import {useState} from ‘react’

[ชื่อ State , ฟังก์ชั่นที่ใช้เปลี่ยนแปลงข้อมูลใน State] = useState(ค่าเริ่มต้นของ State)

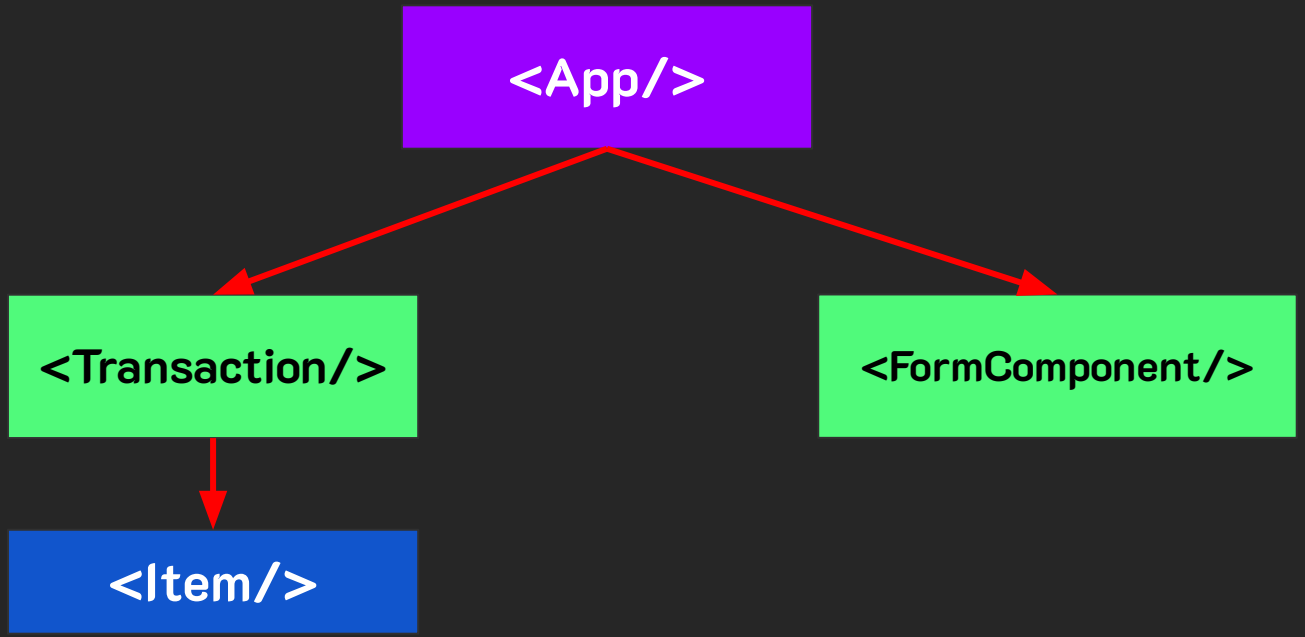
ตัวอย่าง

const [name,setName] = useState (‘kongruksiam’)

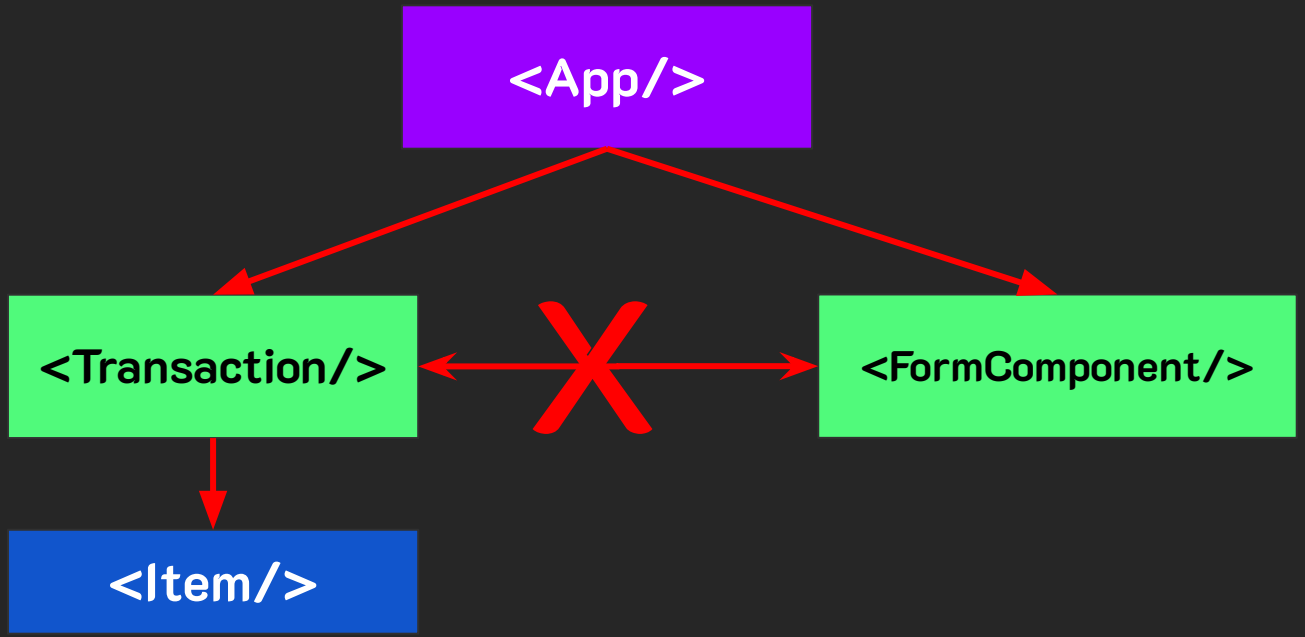
const [age,setAge] = useState(30)

จะได้ Array ที่ Destructuring จาก useState

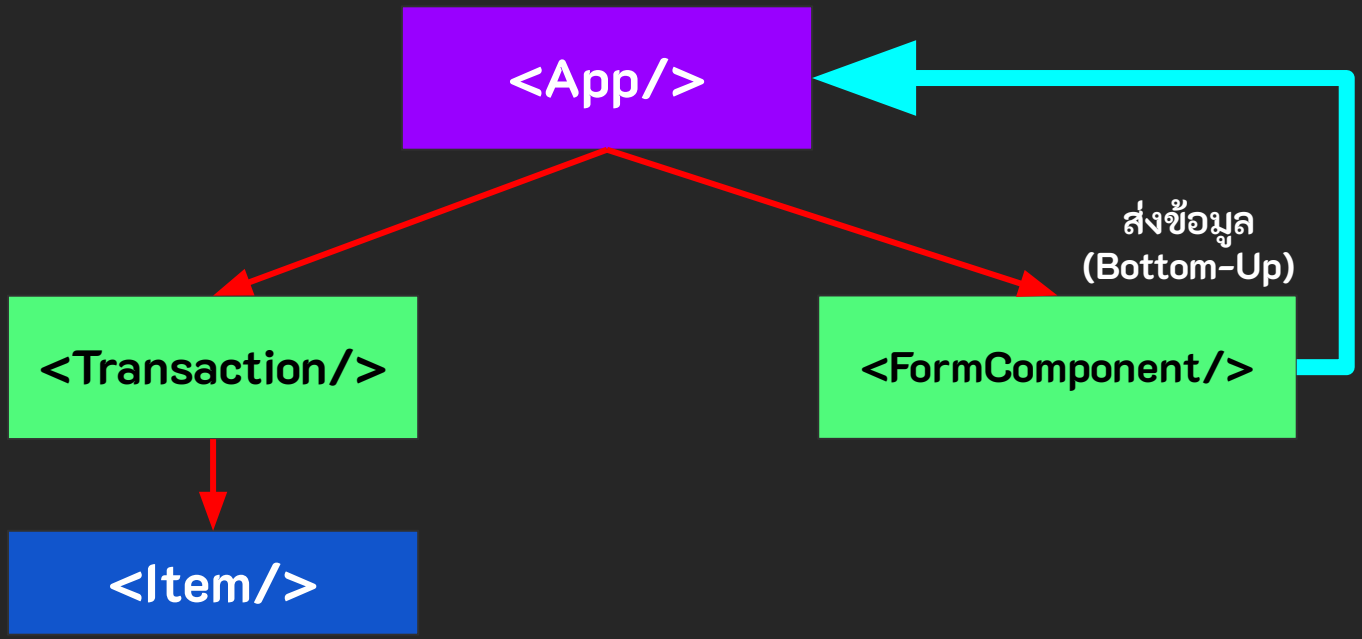
Children to Parent (Bottom-Up)



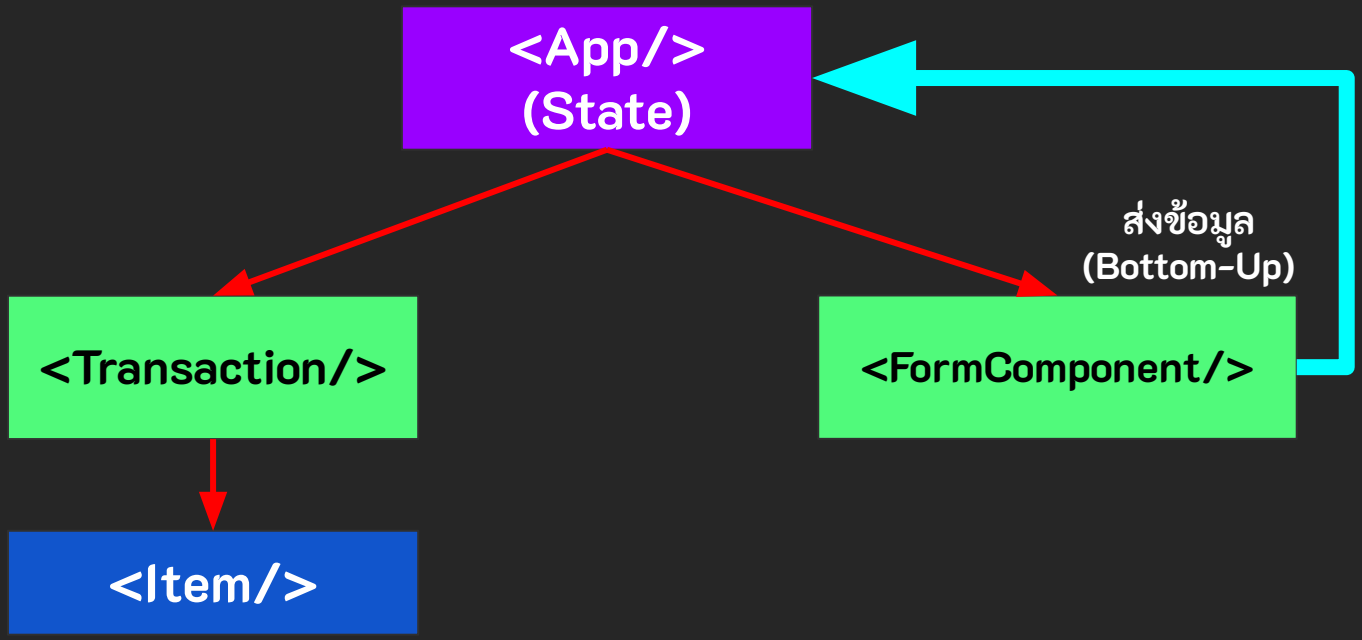
Children to Parent (Bottom-Up)



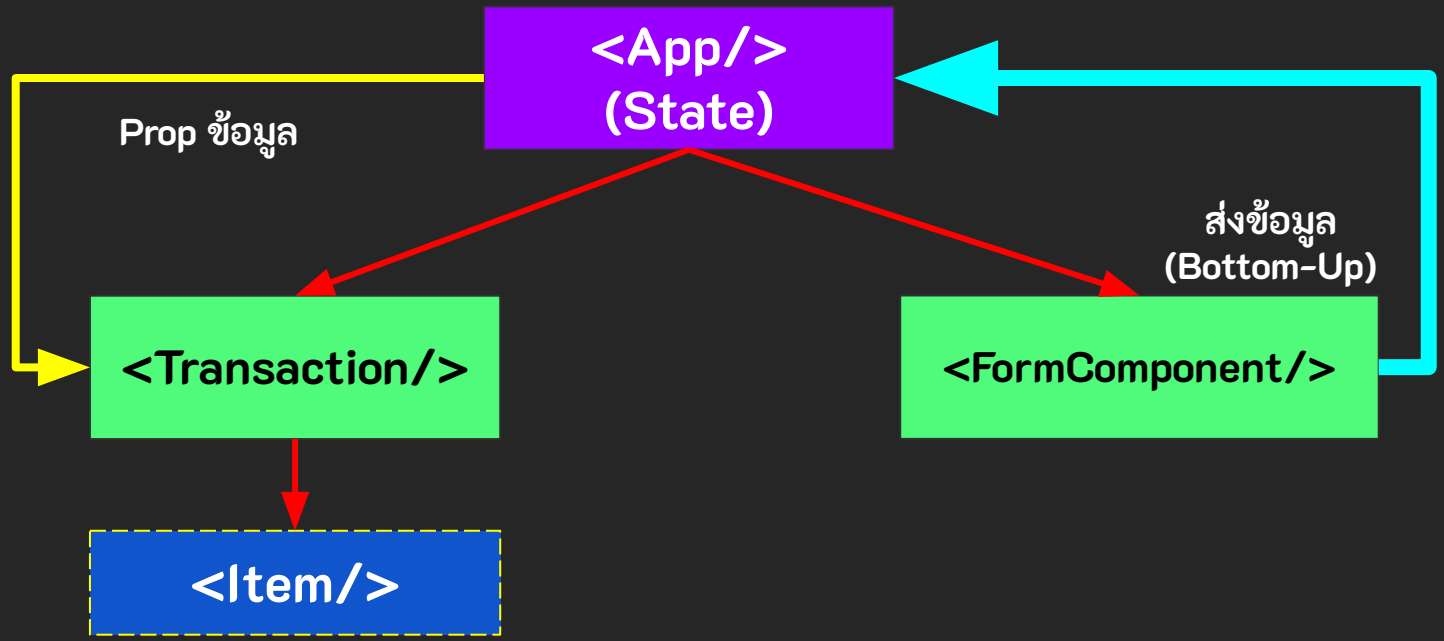
Children to Parent (Bottom-Up)



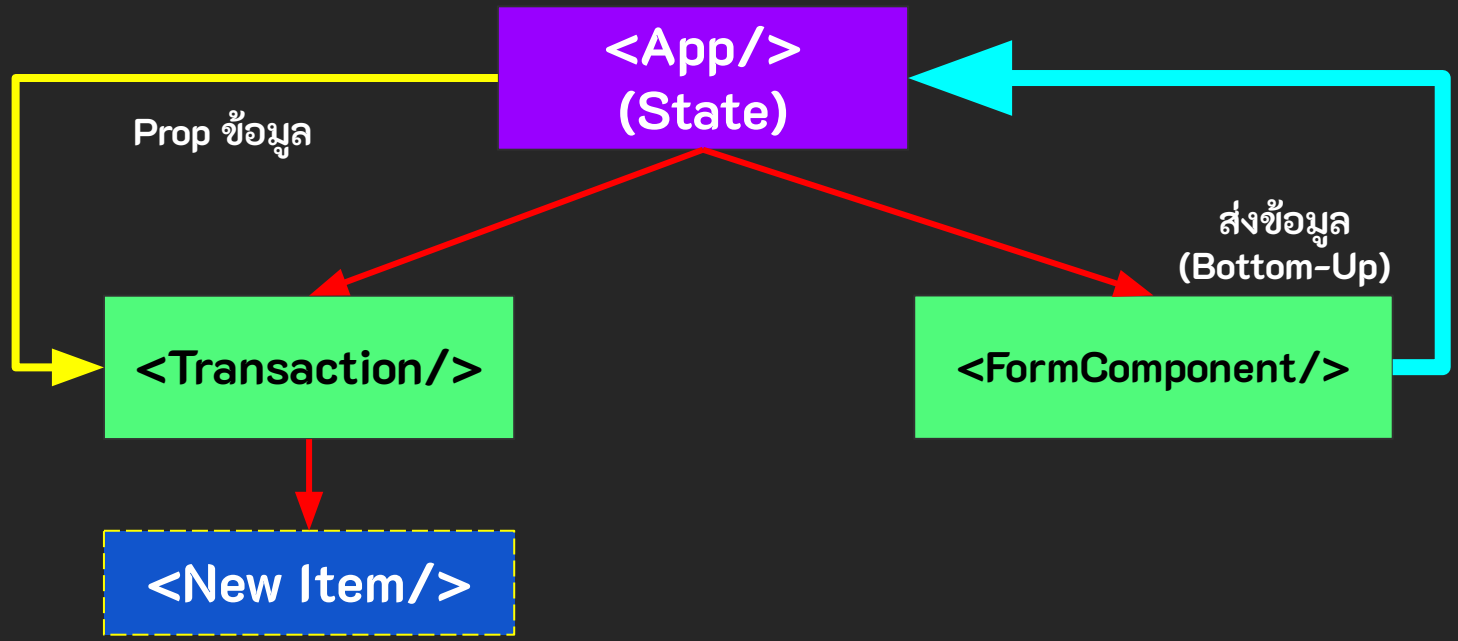
Children to Parent (Bottom-Up)



Children to Parent (Bottom-Up)



Children to Parent (Bottom-Up)



useEffect

** Effect คือ ผลกระทบ

** useEffect คือ ผลกระทบที่เกิดขึ้นภายใน Component ใช้เพื่อต้องการทราบว่าเกิดการอัพเดตหรือเปลี่ยนแปลงอะไรขึ้นบ้างภายใน Component จนส่งผลให้ Component เกิดการ Render ใหม่

โดยสาเหตุหลักๆที่ Component Render ใหม่จะมาจากการเปลี่ยนแปลงค่าที่อยู่ภายใน Props และ State นั่นเอง

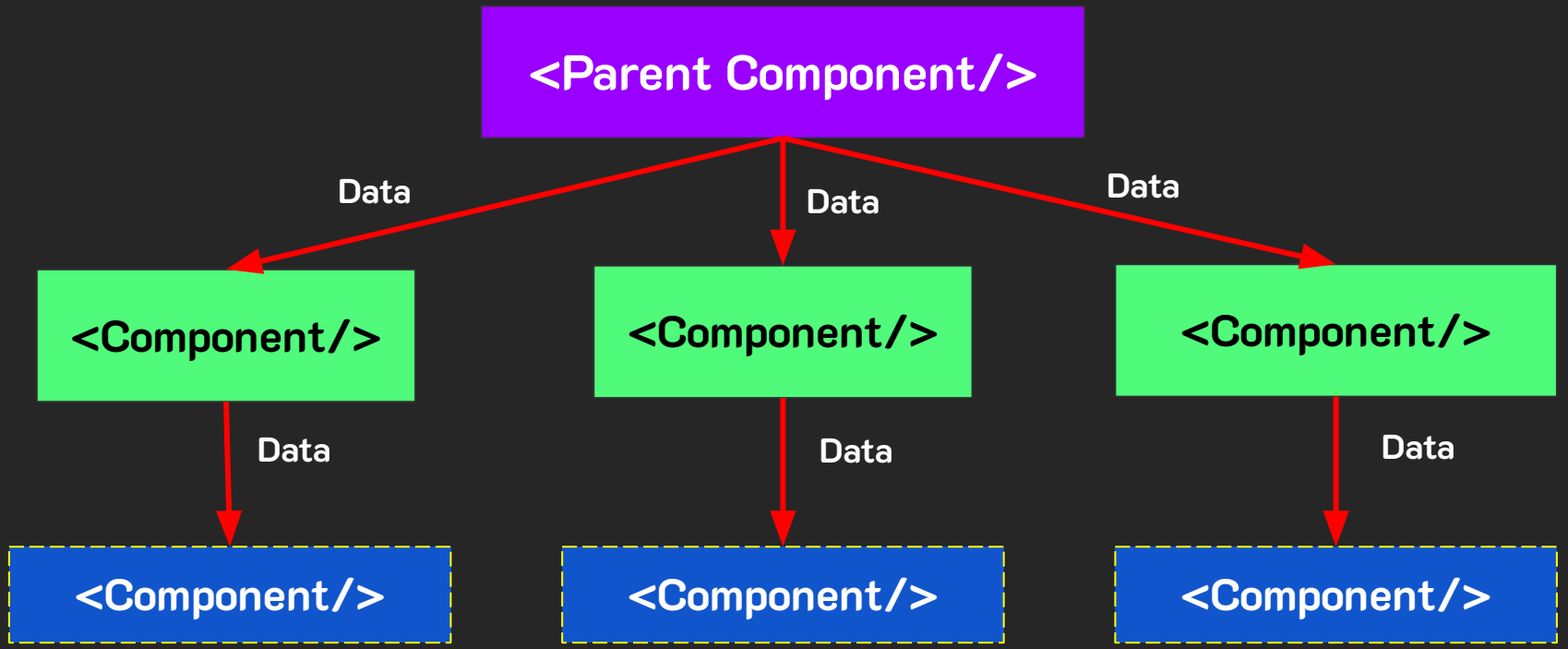
useEffect

การใช้งาน useEffect จึงนำมาใช้งานเพื่อตรวจสอบการเปลี่ยนแปลงที่เกิดขึ้นภายใน Application ของเราว่ามีข้อมูลส่วนใดบ้างที่เปลี่ยนแปลงไปจากค่าหนึ่งไปสู่อีกค่าหนึ่ง จนส่งผลให้ Render Component ใหม่อีกครั้ง

Context API (Global State)

ในกรณีที่ต้องการอยากให้แอพของเรามีข้อมูลกลางและอยากให้มีการแชร์ข้อมูลเกิดขึ้นใน Component จะทำอย่างไร เช่น ต้องการให้ทุกๆ Component สามารถเข้าถึงข้อมูลชุดเดียวกันได้ โดยที่ไม่ต้องใช้รูปแบบส่งข้อมูลจาก Component แม่ไปยัง Component ลูก (Props)

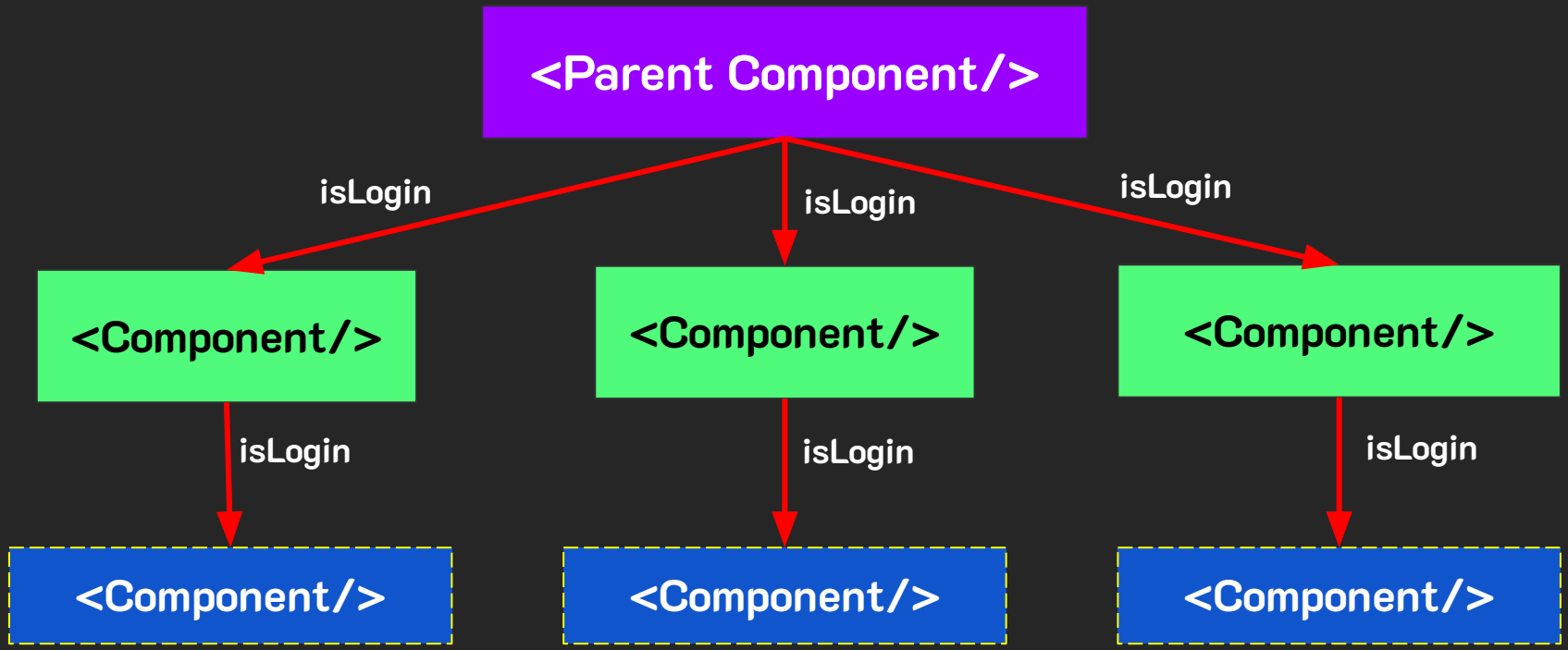
ContextAPI (Global State)



https://www.youtube.com/c/KongRuksiamOfficial/

https://www.facebook.com/KongRuksiamTutorial/

ContextAPI (Global State)

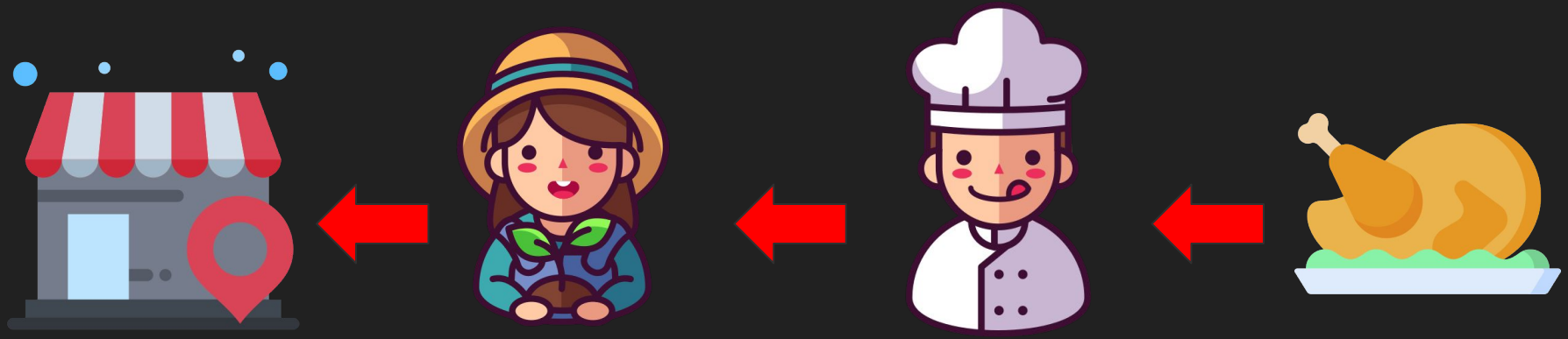


ContextAPI (Global State)

Context API มีองค์ประกอบ 2 ส่วน คือ

- Provider (Parent) ดูแลและจัดการข้อมูลแล้วนำไปส่งให้ Consumer
- Consumer (Children) นำข้อมูลที่ได้จาก Provider ไปสร้างหรือแสดงผลใน Component

แผนภาพการทำงาน



เขียนป้ายบอกเมนูอาหาร (Component)

แม่ค้า (Consumer)

พ่อครัว (Provider)

เมนูอาหาร (ข้อมูล)

useReducer

**** เป็นการจัดการ State ในรูปแบบของ Redux

โดยทั่วไป State สามารถอ่านค่าได้อย่างเดียว ถ้าต้องการอยากจะเปลี่ยนแปลงค่า State จะใช้ useReducer โดยการกำหนดรูปแบบการกระทำที่เกิดขึ้นกับ State ของเราผ่านส่วนที่เรียกว่า Action

React Router

การพัฒนาแอพด้วย React ประกอบไปด้วยการแสดงผลมากกว่า 1 หน้าจอ การที่จะทำให้ผู้ใช้งานไปยังส่วนการแสดงผลต่างๆที่เกิดขึ้นจากการใช้งาน Component ในแอพ เราจะใช้ส่วนที่เรียกว่า

Route (การสร้างเส้นทาง) !!!

React Router

ติดตั้ง npm install react-router-dom

1.สร้าง Router/ Route เพื่อกำหนดเส้นทางการเข้าถึง Component ของการกำหนด Path

2.สร้าง Switch ในการเปลี่ยนเส้นทางการเข้าถึง Component

3.กำหนด Link เพื่อเชื่อมโยง Path กับ Component