# Phần 1 React

## Install

### Phần mềm cần sử dụng

Extends trình duyệt chorm

* react-detecter tool : xác định xem page nào dung reactjs
* cài đặt môi trường phát triển Nodejs (phía server) nodejs.org download và cài đặt.
* su dung yarn cung giống nodejs
* IDE có dụng babel, jsx, emmet

## MAIN CONCEPTS

### Khởi tạo PR

<https://reactjs.org/>

* Cai đặt npm install –g create-react-app
* -g là global gọi dc bất kì đâu
* Tạo mới project

npx create-react-app name\_project

cd name\_project

npm start (chạy project)

có thể đổi port mở package.json

"scripts": {

"start": "react-scripts start",

},

Them

"scripts": {

"start": "set port=4200 && react-scripts start",

},

Sau đó chạy lại start

Cấu trúc thư mục

* Public chứa file index.php,
* Src làm việc với thư mục này nhiều
* File package.json khai báo các thông tin và thư viện sử dụng cho pj

Component là phần xây dựng page ví dụ header, menu, footer, …

* Tạo hello word tạo file /src/Demo.js

import React, { Component } from 'react';  
class Demo extends Component {  
 render(){  
 return(  
 <div className="demo">  
 hello  
 </div>  
 );  
 }  
}  
export default Demo;

* Import Demo.js vào src/index.js

import React from 'react';  
import ReactDOM from 'react-dom';  
import './index.css';  
// import App from './App';  
import Demo from './Demo';  
import \* as serviceWorker from './serviceWorker';  
  
ReactDOM.render(<Demo />, document.getElementById('root'));  
  
// If you want your app to work offline and load faster, you can change  
// unregister() to register() below. Note this comes with some pitfalls.  
// Learn more about service workers: https://bit.ly/CRA-PWA  
serviceWorker.unregister();

render ra public/index.html với phần tử có id = root

### Component

Chèn boostrap vào project

* Download và cop vào public
* Gọi vào index.html sử dụng (jquery,scrip, css, …) js gọi ở dưới cùng trong body

%PUBLIC\_URL%/ : khai báo chỉ để thư mục public

* Component là chia nhỏ template để quản lý dễ dàng
* Tạo thư mục /src/component chứa các component con(header, footer, menu, …)
* Tạo dc 1 header component khi gọi trong appcomponent thì header component nằm trong app component và có thể gọi <Header /> hoặc <Header></Header> giống như html
* Tạo teamplate thì App.js là component cha gọi các component con vào (Home. Menu)

import React,{Component} from 'react';  
import './App.css';  
import Home from "./component/Home";  
import Menu from "./component/Menu";  
  
class App extends Component {  
 render()  
 {  
 return (  
 <div className="App">  
 <Menu></Menu>  
 <Home />  
 </div>  
 );  
 }  
 }  
  
 export default App;

* Chú ý phải viết đúng cú pháp thẻ đóng mở của component
* Nên chia nhỏ các component để dễ quản lý

### JSX

* Jsx viết theo kiểu xml và thay thế cho react.createElement().
* Không phải html nhưng viết giống html
* Thuộc tính class -> className
* Để hiển thị ra sử dụng cặp thẻ {a}

Vi du

class App extends Component {  
 render()  
 {

// khai bao bien  
 var a = 5;  
 var b = 7;  
 var name = 'chiendv6948';  
  
 return (  
 <div>  
 <div className="App">  
 <Menu />   
 </div>

// hien thi, thuc hien bien  
 <div>  
 a : {a}  
 <h3>name : {name}</h3>  
 tong {a + b}  
 </div>  
 </div>  
 );  
 }  
}

* Cũng goi theo đối tượng

class App extends Component {  
 render()  
 {  
  
 var product = {  
 id : 1,  
 name : 'ip 7',  
 price : 15000000  
 };  
 return (   
 <div>  
 id : { product.id } <br />  
 name : { product.name } <br />  
 price : { product.price } VND  
 </div>  
  
 );  
 }  
}

* Sử dụng câu điều kiện để hiển thị

Vd sử dung thuộc tính status

class App extends Component {  
 render()  
 {  
   
 var product = {  
 id : 1,  
 name : 'ip 7',  
 price : 15000000,  
 status : false  
 };  
 return (  
 <div >  
 <div className="App">  
 <Menu />   
 </div>  
 <div className='ml-30'>  
 id : { product.id } <br />  
 name : { product.name } <br />  
 price : { product.price } VND <br />  
 status : { product.status ? 'Active' : 'Pending'}  
 </div>  
 </div>  
 );  
 }  
}  
  
export default App;

cách khác viết hàm để sử dụng trạng thái status

class App extends Component {

showstatus(product){  
 if (product.status){

return <h3>  
 <div className='ml-30'>  
 status : { product.status ? 'Active' : 'Pending'}  
 </div>  
 </h3>  
 }  
}

render()  
 {  
   
 var product = {  
 id : 1,  
 name : 'ip 7',  
 price : 15000000,  
 status : false  
 };  
 return (  
 <div >  
 <div className="App">  
 <Menu />   
 </div>  
 <div className='ml-30'>  
 id : { product.id } <br />  
 name : { product.name } <br />  
 price : { product.price } VND <br />  
 status : { this.showstatus(product) }

</div>  
 </div>  
 );  
 }  
}  
  
export default App;

* Hiển thị danh sách đối tượng

class App extends Component {  
 render()  
 {  
 var user = [  
 {  
 id : 1,  
 name : 'aa1',  
 aeg : 12  
 },  
 {  
 id : 2,  
 name : 'aa2',  
 aeg : 12  
 },  
 {  
 id : 3,  
 name : 'aa3',  
 aeg : 12  
 }  
 ];  
 // duyệt mảng (sd vòng lặp trả về từng đối tượng trong mảng)  
 var element = user.map((user, index) => {  
 // key co the dung user.id

// nếu báo lỗi  
 return <div key={index}>  
 <h2>  
 name : {user.name}  
 </h2>  
 </div>  
 });  
 return (  
 <div >  
 <div className="App">  
 <Menu />   
 </div>  
 <div>  
 {element}  
 </div>  
 </div>  
 );  
 }  
}  
  
export default App;

note sử dụng key là index hay id cho đối tượng

### Props

* Là đối tượng của 1 component
* Truyền data từ cha cho con, theo dạng key = ‘value’ (k đặt key = chidren) value nhận vài có kiểu data là chuỗi. bỏ dấu { } để truyền đúng kiểu data
* Nhận lại thông qua từ khóa this.props.keyVidu:

Class cha

import React,{Component} from 'react';  
import './App.css';  
import Menu from "./component/Menu";  
import Productprops from "./component/Productprops";  
  
class App extends Component {  
 render()  
 {  
 return (  
 <div >  
 <div className="App">  
 <Menu />  
 </div>  
 <div className='container'>  
 <div className='row'>  
 <div className='row'>  
 <div className="col-xs-12">

{/\* truyền data cho thăng con \*/}  
 <Productprops   
 name = "Ip 6"  
 price = '12000'  
 />  
 </div>  
 </div>  
 </div>  
 </div>  
 </div>  
 );  
 }  
}  
  
export default App;

class con

import React,{Component} from 'react';  
  
class Productprops extends Component {  
 render()  
 {  
 return (  
 <div>  
 <div class="col-xs-4 col-sm-4 col-md-4 col-lg-4">  
 <div class="thumbnail">  
 <img data-src="#" alt="" />  
 <div class="caption">  
 {/\*gọi data từ thằng cha với thuộc tính tương ứng (name) this.props.name \*/}  
 <h3>{ this.props.name }</h3>  
 <p>  
 { this.props.price }  
 </p>  
 <p>  
 <a href="#" class="btn btn-primary">Add Cart</a>  
 </p>  
 </div>  
 </div>  
 </div>  
 </div>  
 );  
 }  
}  
  
export default Productprops;

* Nhận nội dung bên trong thẻ this.pros.chidren

Dùng children nhận nội dung bên thằng cha

<Productprops  
 price = '12000'  
>  
 Ip 6s plus 16gb  
</Productprops>

Class con sử dung

<h3>{ this.props.children }</h3>

Lúc này gọi dc nội dung trong thẻ từ class cha nên thuộc tính của thẻ ở lớp cha k dc đặt thuộc tính là children

* Sử dụng vong lặp

Sử dụng map() để hiển thị danh sách

class App extends Component {  
 render()  
 {  
 var product = [  
 {  
 id : '1',  
 name : 'ip6s',  
 price : 12000000,  
 status : true  
 },  
 {  
 id : '2',  
 name : '6s plus',  
 price : 14500000,  
 status : true  
 },  
 {  
 id : '3',  
 name : 'ip6s',  
 price : 14000000,  
 status : false  
 },  
 ];  
 let elements = product.map((product,index) => {  
 return <Productprops  
 key = {product.index}  
 price = { product.price }  
 >  
 { product.name }  
 </Productprops>  
 });  
 return (  
 <div >  
 <div className="App">  
 <Menu />  
 </div>  
 <div className='container'>  
 <div className='row'>  
 <div className='row'>  
 <div className="col-xs-12">  
 {elements}  
 </div>  
 </div>  
 </div>  
 </div>  
 </div>  
 );  
 }  
}

* Sử dụng điều kiện hiển thị

let elements = product.map((product,index) => {  
 let result='';  
 if (product.status){  
 result = <Productprops  
 key = {product.index}  
 price = { product.price }  
 >  
 { product.name }  
 </Productprops>  
 }  
 return result;  
});

### Handling event

* Sử dụng ES6
* Gọi thông qua arrow function
  + {() => this.conclick(param)}

Vd

class Productprops extends Component {  
 onAddToCart(text){  
 alert(text);  
 console.log(text);  
 }  
 render()  
 {  
 return (  
 <div>  
 <div className="col-xs-4 col-sm-4 col-md-4 col-lg-4">  
 <div className="thumbnail">  
 <img data-src="#" alt="" />  
 <div className="caption">  
 {/\*gọi data từ thằng cha với thuộc tính tương ứng (name) this.props.name \*/}  
 <h3>{ this.props.children }</h3>  
 <p>  
 { this.props.price }  
 </p>  
 <p>  
 <a href="#" className="btn btn-primary" onClick={() =>this.onAddToCart('mua thanh coong')}>Add Cart</a>  
 </p>  
 </div>  
 </div>  
 </div>  
 </div>  
 );  
 }  
}  
  
export default Productprops;

sử dung arrow function k hthi alert khi f5

Có thể truyền vào children

* Gọi và sdung

props c1 tạo contructor có tham số và super(props).

class Productprops extends Component {  
 //tạo constructor và supper(props)  
 // đối tượng là this.ten\_function = this.ten\_function.bin(this)  
 constructor(props){  
 super(props);  
 // console.log(props);  
 this.onAddToCart = this.onAddToCart.bind(this);  
 }  
 //gọi đên thuộc tính cần sd  
 onAddToCart(){  
 alert(this.props.children);  
 }  
 render()  
 {  
 return (  
 <div>  
 <div className="col-xs-4 col-sm-4 col-md-4 col-lg-4">  
 <div className="thumbnail">  
 <img data-src="#" alt="" />  
 <div className="caption">  
 {/\*gọi data từ thằng cha với thuộc tính tương ứng (name) this.props.name \*/}  
 <h3>{ this.props.children }</h3>  
 <p>  
 { this.props.price }  
 </p>  
 <p>  
 {/\*gọi function k co param\*/}  
 <a href="#" className="btn btn-primary" onClick={this.onAddToCart}>Add Cart</a>  
 </p>  
 </div>  
 </div>  
 </div>  
 </div>  
 );  
 }  
}  
  
export default Productprops;

* **Cách 2 k cần bind**

Gọi theo arrow function

onAddToCart2 = ()=>{  
 alert(this.props.children);  
}

cách này giảm thiểu khai báo vì k cần bind và sử dụng construct

hoặc sử dụng

<button className='btn btn-warning' onClick={() => this.onClick2("ssa")}>Click me2!</button

### Refs

* Lấy giá trị từ các ô input, ….
* Cú pháp ref =’key’
* Lấy giá trị this.refs.key.value
* import React,{Component} from 'react';   
  import Menu from "./component/Menu";  
  import Productprops from "./component/Productprops";  
    
  class App extends Component {  
   constructor(props){  
   super(props);  
   this.onAddProduct = this.onAddProduct.bind(this);  
    
   }  
   onAddProduct()  
   {  
   console.log(this.refs.name.value);  
   }  
   render()  
   {  
   var product = [  
   {  
   id : '1',  
   name : 'ip6s',  
   price : 12000000,  
   status : true  
   },  
   {  
   id : '2',  
   name : '6s plus',  
   price : 14500000,  
   status : true  
   },  
   {  
   id : '3',  
   name : 'ip6s',  
   price : 14000000,  
   status : false  
   },  
   ];  
   let elements = product.map((product,index) => {  
   return <Productprops  
   key = {index}  
   price = { product.price }  
   >  
   { product.name }  
   </Productprops>  
   });  
   return (  
   <div >  
   <div className="App">  
   <Menu />  
   </div>  
   <div className='container'>  
   <div className='row'>  
   <div className='row'>  
   <div className="row">  
   <div className="col-xs-12 col-sm-12 col-md-12 col-lg-12"> <div className="panel panel-default">  
   <div className="form-group">  
   <label>Name</label>  
   <input type="text" name="" id="input" className="form-control" ref="name" />  
   <button type="button" name="" id="sub" onClick={this.onAddProduct}>Save</button>  
   </div>  
   </div>  
   </div>  
   </div>  
   <div className="col-xs-12">  
   {elements}  
   </div>  
   </div>  
   </div>  
   </div>  
   </div>  
   );  
   }  
  }  
    
  export default App;

### State

* là trạng thái component
* khai báo những giá trị cần lưu trữ riêng của component
* tạo state tại contructor gọi this.state ({key:value, key1:value1, …})
* gọi state this.state.key
* thay đổi state this.setState({key:value, key1:value1, …})
* khi setState dc gọi thì hàm render dc gọi

class App extends Component {  
 // viết trong constructor  
 constructor(props){  
 super(props);  
 this.state = {  
 product: [  
 {  
 id: '1',  
 name: 'ip6s',  
 price: 12000000,  
 status: true  
 },  
 {  
 id: '2',  
 name: '6s plus',  
 price: 14500000,  
 status: true  
 },  
 {  
 id: '3',  
 name: 'ip6s',  
 price: 14000000,  
 status: false  
 },  
 ],  
 isActive : true  
 }  
 }  
  
 render()  
 {  
 //gọi state vào  
 let elements = this.state.product.map((product,index) => {  
 return <tr key={ product.id }>  
 <td>{ index }</td>  
 <td>{ product.name }</td>  
 <td>  
 <span className="label label-success">{product.name}</span>  
 </td>  
 </tr>  
 });  
 return (  
 <div >  
 <div className="App">  
 <Menu />  
 </div>  
 <div className='container'>  
 <div className='row'>  
 <div className='row'>  
 <div className="row">  
 <div className="col-xs-12 col-sm-12 col-md-12 col-lg-12">  
 <div className="panel panel-danger">  
 <div className="panel-danger">  
 State  
 </div>  
 </div>  
  
 </div>  
 </div>  
 <table className='table table-bordered table-hover'>  
 <tbody>  
 <tr>  
 <td>stt</td>  
 <td>ten</td>  
 <td>giá</td>  
 </tr>  
 { elements }  
 </tbody>  
 </table>  
  
 </div>  
 </div>  
 </div>  
 </div>  
 );  
 }  
}  
  
export default App;

sửa isactive vào gọi vào

onSetState(){  
 if (this.state.isActive == true) {  
 this.setState(  
 {  
 isActive:false  
 });  
 }  
 else{  
 this.setState({  
 isActive:true  
 });  
 }  
}

gọi

<table className='table table-bordered table-hover'>  
 <tbody>  
 <tr>  
 <td>stt</td>  
 <td>ten</td>  
 <td>giá</td>  
 </tr>  
 { elements }  
 </tbody>  
</table>  
<button type="button" className="btn btn-default" onClick = {this.onSetState}>  
 Active : { this.state.isActive === true ? "true":'false'}  
</button>

So sánh props va state

|  |  |
| --- | --- |
| **Props** | **State** |
| Nhận dữ liệu từ bên ngoài | Nhận dữ liệu nội bộ |
| Không thể thay đổi giá trị | Có thể thay đổi giá trị |
|  | Phạm vị private trong component |

### project1:

B1: tạo pj

B2 : tạo component

Tách các component chuẩn

Nên tách mỗi chức năng 1 component, tách càng nhỏ càng tốt

* tạo chức năng colorpicker

class App extends Component{  
 constructor(props){  
 super(props);  
 this.state={  
 color : 'red',  
 fontsize : 15  
 }  
 this.onSetColor=this.onSetColor.bind(this);  
 }  
  
 //lấy data từ thằng con gán prams  
 // set lại color dc lấy từ thằng con  
 onSetColor(params){  
 this.setState({  
 color: params  
 });  
 }  
 render(){  
 return(  
 <div className="container">  
 <div className="row">  
 {/\*onReceiveColor nhan data tu thang con sang thang cha\*/}  
 <ColorPicker color = {this.state.color} onReceiveColor={this.onSetColor}/>  
 <div className="col-xs-6 col-sm-6 col-md-6 col-lg-6">  
 <Setting />  
 <Reset />  
 </div>  
 <div className="col-xs-12 col-sm-12 col-md-12 col-lg-12">  
 <Result color={this.state.color} />  
 </div>  
 </div>  
 </div>  
 );  
 }  
}  
export default App;

compornent con

class ColorPicker extends Component{  
 constructor(props){  
 super(props);  
 this.state={  
 colors : [  
 "red",  
 "green",  
 "blue",  
 "#ccc"  
 ]  
 }  
 }  
  
 // hien thi color  
 showColor(color){  
 console.log(this.props.color);  
 return {  
 backgroundColor : color,  
 }  
 }  
 // trả data cho thằng cha App (onReceiveColor)  
 setActiveColor(color){  
 // console.log(color);  
 this.props.onReceiveColor(color);  
 }  
 render(){  
 var elements = this.state.colors.map((color,index)=>{  
 return <span  
 key={index} className={this.props.color === color ? 'active color' : 'color'}  
 style={this.showColor(color)}  
 onClick={() =>this.setActiveColor(color)}  
 >  
 </span>  
 });  
 return (  
 <div className="col-xs-6 col-sm-6 col-md-6 col-lg-6">  
 <div className="panel panel-primary">  
 <div className="panel-heading">  
 Color Picker  
 </div>  
 <div className="panel-body">  
 {elements}  
 </div>  
 <hr/>  
 </div>  
 </div>  
 );  
 }  
   
}  
  
export default ColorPicker;

result

class Result extends Component{  
 setStyle(){  
 return{  
 color : this.props.color,  
 borderColor : this.props.color  
 };  
 }  
 render(){  
 return (  
 <div className="col-xs-12 col-sm-12 col-md-12 col-lg-12 ">  
 <p>Color: red - frontsize : 15px</p>  
 <div id="content" className='result' style={this.setStyle()}>  
 <span >  
 Setting  
 </span>  
 </div>  
 </div>  
 );  
 }  
   
}

.result{  
 border: solid 1px;  
 border-color: red;  
 padding: 10px;  
}  
  
.color{  
 float: left;  
 width: 40px;  
 height: 40px;  
 margin-left: 5px;  
}  
.active{  
 border: solid 5px;  
  
}

* Phần 2 chuc nang tang giam và reset

### Form

class App extends Component{  
 constructor(props){  
 super(props);  
 this.state = {  
 txtname : 'aa',  
 txtpassword : '',  
 txtdesc : 'ádasd',  
 sltGender : 1,  
 rdlang : 'vi',  
 chkbStatus : false  
 };  
 this.onHandleChange = this.onHandleChange.bind(this);  
 this.onSubmitform = this.onSubmitform.bind(this);  
 }  
  
 //Lay data tu form nhiều thuộc tính tru checkbox  
 onHandleChange(event){  
 //name và value lay từ target  
 var target = event.target;  
 // lấy name của form(txtuser, txtpassword)  
 var name = target.name;  
 // lấy giá trị của form của các thuoc tính, checkbox thì lấy data bằng taget.checked  
 // var value = target.value;  
 var value = target.type === 'checkbox' ? target.checked:target.value;  
   
 // gọi state  
 this.setState({  
 [name] : value  
 })  
 }  
  
 onSubmitform(event){  
 // khi submit form k load lại trang  
 event.preventDefault();  
 console.log(this.state);  
 }  
  
 render(){  
 return (  
 <div className="container-fluid">  
 <div className="row">  
 <div className="col-xs-8 col-sm-8 col-md-8 col-lg-8">  
 <div className="panel panel-primary">  
 <div className="panel-heading">  
 <h3 className="panel-title">Form</h3>  
 </div>  
 <div className="panel-body">  
 <form onSubmit={this.onSubmitform} >  
 {/\*value thường đi theo onChange\*/}  
 <div className="form-group">  
 <label >User name</label>  
 <input type="text"  
 className="form-control"  
 placeholder="User"  
 name='txtname'  
 onChange={ this.onHandleChange }  
 value={this.state.txtname}  
 />  
 </div>  
 <div className="form-group">  
 <label >Password</label>  
 <input type="password"  
 className="form-control"  
 placeholder="Password"  
 name='txtpassword'  
 onChange={ this.onHandleChange }  
 value={this.state.txtpassword}  
 />  
 </div>  
 <div className="form-group">  
 <label>Password</label>  
 <textarea className="form-control"  
 rows='5'  
 name='txtdesc'  
 onChange={ this.onHandleChange }  
 value={this.state.txtdesc}  
 />  
 </div>  
  
 {/\*the <option> k can thuoc tinh selected vi  
 the select co thuoc tinh value goi den state nên xác định giá trị của option  
 \*/}  
 <div className="form-group">  
 <label>Gene</label>  
 <select name="sltGender"  
 id="input"  
 className="form-control"  
 required="required"  
 value={this.state.sltGender}  
 onChange={ this.onHandleChange }  
 >  
 <option value={0}>nu</option>  
 <option value={1}>nam</option>  
 </select>  
 </div>  
 {/\*thuoc tinh checked xdinh xem giá trị ban đâu để select \*/}  
 <div className="form-group">  
 <label>Ngôn Ngữ</label>  
 <br/>  
 <label>  
 <input  
 type="radio"  
 name="rdlang"  
 value='en'  
 onChange={ this.onHandleChange }  
 checked={this.state.rdlang === 'vi'}  
 />  
 Anh  
 </label>  
 <br/>  
 <label>  
 <input  
 type="radio"  
 name="rdlang"  
 value='vi'  
 onChange={ this.onHandleChange }  
 checked={this.state.rdlang === 'vi'}  
 />  
 Viet  
 </label>  
 </div>  
 {/\*truong hop dac biet\*/}  
 <div className="checkbox">  
 <label>  
 <input  
 type="checkbox"  
 value={true}  
 name={'chkbStatus'}  
 onChange={this.onHandleChange}  
 checked={this.state.chkbStatus === true}  
 />  
 Status  
 </label>  
 </div>  
   
  
 <button type="submit" className="btn btn-primary">Submit</button> &nbsp;  
 <button type="submit" className="btn btn-default">Delete</button>  
 </form>  
 </div>  
 </div>  
 </div>  
 </div>  
 </div>  
 );  
 };  
}  
  
export default App;

### Pj2- quản lý them sửa xóa công việc

# Phần 2 Redux

# Phần 3 Router

# Phần 4 Connect API