**React数据传递：**

react数据传递使用两种事物进行传递：

数据传递的载体：承载数据变量---数据传递载体

1、属性（props）在组件外部传入，组件内部通过this.props获得；

2、状态（state）在组件内部设置或更改，组件内部通过this.state获得；

属性表示一个事物与生俱来的特性，并且不容易被修改

状态表示可以被修改的特性

**peops（属性）：**

1、**属性**：是在一个组件被声明的时候（父组件）赋予的，并且**不可以被修改**

其传递方式一般是在组件被调用的时候被传入；

可以传递字符串、对象、数组。（键值对）

2、使用**属性传递数据的目的**：为了实现数据从model到view绑定的单向数据流

3、如何**声明和传递**属性？

i、**key-value**；

声明变量后，在渲染的时候将数据传递到组件内部**<组件名称 key={值}>，**

通过this.props.key获取值

//**声明数据，作为属性传递到组件内部**

var name = "snn";

<div>{this.props.s}</div>

ReactDom.render(<Comp s={name}/>);

ii、**key-value拓展{}、[ ]；**

前两种方法局限性：声明的时候是通过属性key传值到组件内部，只要在外部改变了key

内部无法获取，内部对外部key要求很强

var arr =["img/pic8.jpg","Hello World"];

var \_obj = {img:'img/pic8.jpg',desc:'hahahaha'};

var Comp = React.createClass({

var \_props = this.props.title;

render：function(){

<div>

<div>{\_props.desc}</div>

<img src="{\_props[0]}" />

</div>

}

})

React.render(<Comp title={\_obj} />,document.body)

iii、**<组件名称 {...obj}/>三点展开式**{...对象表达式}，通过**this.props获取**

我们声明一个对象，数组的时候全部传入到组件内部

React.render(<Comp {...\_obj}>,document.body)

iiii:**在组件内部声明，通过react的一个函数声明的**

getDefaultProps：react组件声明周期函数，通过this.props获取

getDefaultProps：function（）{

**//如何声明属性？在该方法里面返回一个对象即可，组件内部都是高内聚的**

return{list:"XXX",src:"cccc.jpg"}

return{service："url"} **//设置组件数据源**

},

**属性传递步骤：**

1、**声明数据**，作为**属性传递到组件内部**

var name = "snn";

2、**声明组件**

<div>{this.props.s}</div>

3、**渲染** 如何将我们声明的数据作为属性传递给组件

ReactDom.render(<Comp s={name}/>);

**state（状态）：**

1、**状态**：表示组件在不同时刻的情况的一种描述，

？可变状态在改变的时候，组件会不会发生连锁反应？

**当状态发生改变时，组件的render方法会进行二次渲染；**

2、核心问题：

状态如何去**初始化**？状态初始化之后，**改变状态**？

**getInitialState方法进行初始化，返回对象表示初始化的状态值；**

**//存储从服务端得到的数据值that.setState({img:\_arr});**

**获取状态：this.state;**

**修改状态：this.setState({key:更新值})**

eg：点击按钮使div透明度慢慢增加

var React = React.createClass({

getInitialState:function(){

return{

opacity:0

}

},

handleClick:function(){

var s = this.steat.opacity;

s+=0.1;

this.setState({opacity:s});

},

render:function(){

var \_s = this.state;

return(

<div>

<div className="styles" style={\_s}></div>

<button onChange={handleClick}>++</button>

</div>

)

}

})

ReactDOM.render(<React/>,document.body);

eg2:输入密码框默认时候是password类型，点击显示（点击按钮上面的内容也发生改变），变成text类型；

var Text = React.crateClass({

getInitialState:function(){

return{

type:"password",

text:"密文",

value:""

}

},

render:function(){

return(

<div>

<input style={{border:'2px solid red',outline:'none'}} type={this.state.type}/>

<button onClick = {this.handleClick}>{this.state.text}</button>

<input type={this.state.type} onChange={this.handleChange} />

<span>New Value:{this.state.value}</span>

<Child name={this.state.value}></Child>

</div>

);

},

handleChange:function(){

var \_value = event.target.value;

this.setState = ({value:\_value});

},

handleClick:function(){

if(this.state.type=="password"){

this.setState({type:'text',text:"明文"})

}else{

this.setState({type:'password',text:"密文"})

}

}

});

//数据由父组件传入子组件

var Child = React.createClass({

render:function(){

var styles = {color:'red'};

return(

<div style={styles}>

test child value:{this.props.name}

</div>

)

}

})

ReactDOM.render(<Text/>,document.body);

**状态使用：**

1、当状态发生变化时：render执行第二次，实现二次渲染

2、jsx关联属性放在状态里面作为变数，通过事件改变参数，页面发生二次渲染，

**二次渲染时，jsx通过{this.state.属性}形式读取最近状态，页面发生刷新**

**单项数据流(model-view)：数据只能从父组件传递到子组件；**

1、**实现数据由父组件传入子组件：**

将一些变化的值保存在父组件的state里面

子组件嵌套在父组件中同样发生二次渲染，子组件也能读取最新的state的值，这样就把

父组件的值传递给子组件，

2、**从子组件传递到父组件：（使用回调函数）**

var Father = React.createClass({

getInitialState:function(){

return{

\_value:""

}

},

render:function(){

var \_this = this;

return(

<div>

<span>父组件得到子组件的值:{this.state.\_value}</span>

<Son name = {function(\_child\_value){

console.log(\_child\_value);

\_this.setState({\_value:\_child\_value});

}}>

</Son>

</div>

)

}

});

var Son = React.createClass({

render:function(){

return(

<div>

<input onChange={this.handleClick}/>

</div>

)

},

handleClick:function(event){

var v = event.target.value;

console.log(v);

this.props.name(v);

}

});

ReactDOM.render(<Father/>,document.body);