**受控组件**

受控组件：

<input value={this.state.inputvalue} onChange={(e)=>{this.setState({

inputvalue: e.target.value

})}}>

非受控组件：<input>

受控组件即通过state控制元素的值

一个简单的状态跟新

//构造函数初始状态

constructor(props){

super(props);

this.state = {

AccountIDL:"",

}

}

//控件更新状态

<input value={this.state.AccountIDL} onChange={(event)=> this.setState({AccountIDL: event.target.value})} type="text" />

.........

**file input 标签**

文件标签比较特殊，以下示例显示了如何通过 ref 来访问提交处理程序中的文件：

class FileInput extends React.Component {

constructor(props) {

super(props);

this.handleSubmit = this.handleSubmit.bind(this);

}

handleSubmit(event) {

event.preventDefault();

alert(this.fileInput.files[0].name);

}

render() {

return (

<input type="file" ref={ input => { this.fileInput = input; } } />

);

}

}

**受控 Input 组件的 null 值**

在 受控组件 上指定值 value 的值为null可防止用户更改输入，除非您希望如此。

**受控组件的替代方案**

有时使用受控组件有些乏味，因为你需要为每一个可更改的数据提供事件处理器，并通过 React 组件管理所有输入状态。当你将已经存在的代码转换为 React 时，或将 React 应用程序与非 React 库集成时，这可能变得特别烦人。在这些情况下，您可能需要使用[不受控的组件](http://www.css88.com/react/docs/uncontrolled-components.html)，用于实现输入表单的替代技术。