**1、安装create-react-app脚手架工具**

npm install -g create-react-app

1. **通过create-react-app新建react项目**

npx create-react-app my-app

cd my-app

yarn start / npm start

# 2-3 工程目录文件简介

**yarn.lock**

安装包版本号，没事别动即可

**README.md**

项目说明文件，可用markdown语法自己编写项目介绍

**package.json**

node的包文件，命令，脚手架自动生成的文件

**.gitignore**

使用git管理代码时，有些文件不想上传到git仓库，就把文件放这里

**node-modules**

依赖第三方的包，没事不要动

**public – facicon.ico**

网页title上的图标

**public – index.html**

项目的首页模板

**public – manifest.json**

与ServiceWorker.js有关，离线时的快捷方式目录之类的。。。

**src - 项目所有源代码文件夹**

**index.js**

整个程序运行的入口文件

**ServiceWorker.js**

Web离线应用解决方案

**app.test.js**

自动化测试文件

# 2-4 React中的组件

组件就是页面的一部分。

**import** React, { *Component* } **from 'react'**; // 结构赋值

// 等价于

**import** React **from 'react'**;

const Component = React.Component

ReactDOM.render(<**App** />, ***document***.getElementById(**'root'**));

ReactDOM第三方模块，render方法能把一个组件挂载到某个DOM节点下

<**App** /> - jsx语法

在react中如果使用jsx语法，必须引入react

# 2-5 React中最基础的JSX语法

在JSX语法中，如果我们要使用自己创建的组件，直接使用标签名即可

<App />

注意：组件开头必须大写

# 3-1 使用React编写Todolist功能

在JSX语法里，要求每个组件render函数返回的内容，外层必须包裹元素

在最外层不想多出一个DIV元素时，可以使用fragment占位符

**Fragment 占位符（需引入）**



# 3-2 React中的响应式设计思想和事件绑定

JSX语法 参数外加大括号 { value }

<**input value=**{ **this**.**state**.**inputValue** }/>

react事件绑定 onChange

原生 onchange

绑定事件时，要用 **.bind(this)** 对函数的作用域进行变更

<**input  
 value=**{ **this**.**state**.**inputValue** }  
 **onChange=**{ **this**.handleInputChange.bind(**this**) }  
/>

改变数据的值

handleInputChange(e) {  
 **this**.setState({  
 **inputValue**: e.tartget.**value** })  
}

state 负责存储组件里的数据

constructor(props) {  
 **super**(props) *// 固定写法* **this**.**state** = {  
 **inputValue**: **''**,  
 **list**: []  
 }  
}

setState 改变数据项里的内容

**this**.setState({  
 **inputValue**: e.tartget.**value**})

# 3-3 实现TodoLsit 新增删除功能

...

**展开运算符**

在ES6中用...来表示展开运算符，它可以将数组方法或者对象进行展开。先来看一个例子它是如何使用的。

const arr1 = [1, 2, 3];

const arr2 = [...arr1, 10, 20, 30];

这样，arr2 就变成了[1, 2, 3, 10, 20, 30];

handleItemDelet(index) {  
 *// immutable  
 // state 不允许我们做任何的改变* **const** list = [...**this**.**state**.**list**]  
 list.splice(index, 1)  
 **this**.setState({  
 **list**: list  
 })  
}

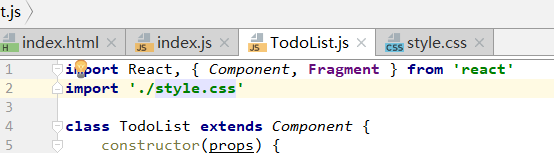
# 3-4 JSX语法细节补充

在jsx里写注释：将注释内容放入大括号内

{ */\*注释\*/* }

直接引入css文件

**import './style.css'**



定义元素的css的类名**用className**

<**input className="input"**/>

不转义内容中的标签 <div><div/>之类的html标签

**dangerouslySetInnerHTML=**{{**\_\_html**: item}}

label 扩大点击区域标签

实现点击label文字，鼠标在input框中

<**label htmlFor="insertArea"**>输入内容</**label**>  
<**input  
 id="insertArea"  
 className="input"  
 value=**{ **this**.**state**.**inputValue** }  
 **onChange=**{ **this**.handleInputChange.bind(**this**) }  
/>

# 3-5 拆分组件与组件之间的传值