React

React是由Facebook维护的一套框架，并且引用到Instagram

React只是我们熟悉的MVC框架中的V层，只是视图层面的一个框架，所以学起来很简单，只有两个api（createElement，createClass，render）

它与angular相比

比如页面（HTML）是一辆汽车，angular做就是给添加指令，相当于为汽车添加一些金矿，提高它的马力等等，就是在原有的基础上增加一些新的功能，看上去很高大上，但是造成一些性能的消耗。

React则不同，相当于另外创建一辆汽车，这汽车很简单只有四个轱辘，一个座子，所以不需要太多的性能就能开动它。

React是创建一个虚拟的dom，这个dom不是真正页面中dom元素（实际上是一个js对象，用js对象存储dom上的信息，因此比真实的dom小的多，因为它只记录一些必要信息）

操作一个虚拟dom需要的性能远小于操作一个真实的dom需要的性能

在前端开发一个web应用与开发一个网站不同，web应用通常是一个单页面的，因此每次做一些交互会涉及dom的操作，因此无节制的操作dom严重影响了页面的性能，如果我们只操作这些虚拟dom（此时会很好降低性能的消耗），在必要的时候将这些需要展现的视图渲染到页面中会优化这些性能问题（reconciliation）

React在优化上做了两点

一点是渲染时性能的优化（主要是虚拟dom）

第二点是开发时的优化，React的理念有了虚拟dom，因此不要依赖于哪个端，因此它的理念就是一处开发处处使用，如何将虚拟dom渲染到某个端上需要引入不同的插件，调用不同的render方法

本套课程主要讨论的是浏览器端，所以需要两个库

React库，主要用来创建虚拟dom

React-dom库，将虚拟dom渲染浏览器上

体验React

要安装两个库

React对象有个createElement方法，在这个方法中我们可以定义一个虚拟dom

第一个参数表示虚拟dom的名称（任意的元素名称：div，a，li等等，还可以是一个组件的名称）

第二个参数表示虚拟dom上一些必要的属性（比如id，classname，title，等等一些必要的属性）

第三个参数表示该虚拟dom中所有子元素（定义文本节点，我们可以直接书写不用createElement）

返回值是虚拟dom

ReactDOM中的render方法

是将虚拟dom渲染到页面中的一个方法

第一个参数：表示虚拟dom

第二个参数：表示真实dom容器元素

第三个参数：是一个回调函数 5.31第二节

**组件**

如果一个虚拟dom复用多次，通常我们将它封装在一个组件中

通常用组件创建一组虚拟dom

创建组件的方法是createClass

参数是一个对象，对象中的属性和方法是对组件的说明

有一个属性叫render方法，是将组件中的虚拟dom输出的，所以我们将虚拟dom定义在render方法中

组件名称通常第一个字母大写

Render

返回值是虚拟dom树

作用域是组件当前对象

通常将组件渲染到页面中要将组件转化成一个虚拟dom，要调用createElement方法转化

JSX语法

解决创建一个虚拟dom成本过大的问题

jsx语法就是让我们书写html元素中标签的方式来定一个虚拟dom，在html中如何书写元素，我们就怎么书写虚拟dom元素

html中定义id是app的div： <div id=”app”></div>

在jsx语法中我们定义一个div： <div id=”app”></div>

但是js不支持这个语法，所以我们需要一些编译的插件或者使用工程化工具编译

工程化工具，我们用babel2插件解析

对于React的jsx语法，我们为了语义化文件，所以我们通常将文件拓展名定义为jsx

首字母大写

嵌套

求值表达式

驼峰命名法

htmlFor和className

插值

在react中插值语法是{}，在大括号中我们可以书写任意的表达式

js内置的方法，定义的方法，运算符，语句等等

Props属性

在html中对于两个相同的div，我给其中一个添加一个类，此时这个div展示的样式就不一样的，因为他具有一个其他div布局的属性，这个属性就叫class

对于完全一致的统一组件那他们展示的样式是完全一致的，如果给他们其中一个添加一些属性，此时这个组件展示的结果就可能会不同了，因此React提出组件属性的概念。

我们可以对组件添加一些属性，让他们展示的结果不一样

在jsx中为组件添加属性跟html中添加属性的方式是一模一样的，只不过React组件可以执行插值（可以将js中数据添加到组件中）

有时候我们没有传递属性数据，此时在属性数据调用方法是不安全的，因此我们要对每一个属性添加默认属性数据

getDefaultProps： 添加默认属性的

注意：一定要有返回值，返回值就是默认属性

样式定义

样式style属性的值只能是对象，不能为style添加字符串的值

font-size这类多单词属性要用驼峰式命名

有css3前缀的属性第一个字母要大写

组件中还是在定义一个虚拟dom中我们都要用对象，并且可以使用变量，可以书写对象字面量，在对象字面量中可以使用表达式

状态state

无状态组件

如果组件被创建并渲染到页面中以后不会再更改，也就是说组件是一成不变的，这类组件，我们只需要在创建之初为其添加一些属性即可完成对样式行为的控制，这类组件以后再也不会改变了，我们称之为无状态组件（stateless组件）（不会与用户产生交互，或者发送异步请求）

有状态组件

如果组件创建后根据用户的不同交互产生不同的行为（样式），这一类组件我们称之为有状态组件，组件处于哪种状态是由用户决定的。

组件内部通常会维护一个状态，这个变量叫state，跟props一样，我们都可以更改他们

但是我们说props是在组件创建时传递的属性值不同而决定的，因为以后通常是更改不了的（除了子组件）

state通常是在产生交互时候改变的，因此它的改变永远伴随着一个交互

每次state，props的改变都会执行一次render方法来重新渲染组件，组件是否更改根据虚拟dom有没有改变决定

state与props一样也是可以设置默认状态，state设置默认状态通过

getInitialState设置的，设置方式同props一样通过return {}来设置

在组件内部改变状态，我们通过setState方法来改变

setState方法调用一定会触发render方法

交互通常是由事件触发的，因此我们会定义事件

定义事件

在react定义一个事件跟html中为元素绑定事件是一模一样的。是通过on+事件名称定义的

事件都有回调函数，通常将事件定义在组件内部，通过this来添加事件回调函数

作用域是组件实例化对象

复习

createElement 创建一个虚拟dom的方法，有React对象提供

第一个参数表示dom元素名称，它的值可以是html中元素的tagName，例如div，p,h1等等，也可以是组件的名称

第二个参数表示虚拟dom的属性

从第三个参数开始表示的虚拟dom的子元素

Render 将虚拟dom渲染到浏览器中，ReactDOM提供的

第一个参数表示虚拟dom（如果是组件，必须转换成虚拟DOM）

第二个参数表示真实的dom容器

createClass 创建组件的方法

参数是一个对象，对象中我们定义组件相关的方法属性

Render，将组件中的虚拟dom输出

JSX语法 为了简化开发的，是对创建虚拟dom简化的

内容：像创建html中dom元素一样创建一个虚拟dom，创建方式一模一样

特殊属性

Class 我们要写成className

For 我们要写成htmlfor

插值 向虚拟dom中动态的添加数据（可以插入元素的内容中，也可以插入在元素的属性中）

语法{}

{}中我们可以使用js中的任何方法，组件实例化对象中的方法，语句，运算符等等

Props属性

属性是为了实现组件创建时的差异化，通过对创建的组件添加属性来实现对组件创建的结果改变

在组件内部可以通过this.props来获取组件的属性数据

getDefaultProps

作用域 作用域是组件构造函数

参数 空的

返回值 默认的属性，必须要有返回值

Style 虚拟dom中特殊的一个属性，不能使用行内样式的属性，只能插入一个js的变量

属性名称 是特殊的

如果是多单词属性，fon-size:我们要写成驼峰式的命名fontSize

如果是css3的属性： transation，我们要写成WebkitTransition，第一个字母要大写

State状态 通常组件内部维护的一个数据，通常在组件发生交互时候发生改变  
 State和props一样在状态改变的时候，都会触发render

Props是在组件创建时候提供的，往往是不会更改的

State在组件发生交互的时候改变，它是组件自身提供的

getInitialState 设置默认的状态

作用域是组件实例化对象

返回值 表示默认的状态 必须要有返回值，是个对象

setState 是用来改变状态的，通过this.setState调用

他的参数就是一个对象，这个对象中每个属性就是将要改变的状态

事件 跟html中绑定事件的方式是一样的 onClick，他的回调函数，往往我们要写在组件内部

回调函数

作用域 组件实例化对象

第一个参数 表示事件对象

组件生命周期

React在设计组件时候，将组件看成是一个有生命的物体，让它具有一个周期，组件的周期分成三大周期

创建期 在组件创建的时候，进入该阶段

存在期 在组件已经创建，内部的状态属性发生改变的时候，会进入该阶段

销毁期 在组件从页面中删除，进入该阶段

创建期

在创建的时候，组件会进入五个阶段

第一个阶段 获取默认的属性getDefaultProps

组件还没有创建出来我们还看不到组件内部的对象属性

第二个阶段 获取默认的状态 getInitialState

组件内部数据已经创建出来，我们可以访问到组件实例化对象了

在该阶段已经解析了属性，因此我们可以将属性的值传递给状态（很常用的为状态添加数据的方式）

在组件外部只能为组件添加属性数据，而不能添加状态数据

因为处理状态的阶段在处理属性的阶段后面，因此可以在状态处理方法中处理属性的数据

第三个阶段 组件将要被创建 componentWillMount

指的是组件的虚拟dom

因为我们的状态属性已经设置完成了，因此我们可以在这个阶段处理状态数据以及属性数据

第四个阶段 组件渲染输出虚拟dom render

一定要通过return返回虚拟dom树 （必须有返回值，返回值必须是虚拟dom）

所有的dom都要在一个父dom中

第五个阶段 组件已经构建完成 componentDidMount

组件已经创建完成虚拟dom，在这个阶段我们可以操作dom元素

操作的结果有时直接反映在页面中

这五个阶段所有方法的参数都是空的，并且除了第一个阶段方法的作用域是构造函数，其他的作用域都是组件的实例化对象

ReactDOM.findDOMNode 根据虚拟dom获取真实的dom元素

参数是this，就可以访问组件实例化对象了

eg：更改字体颜色 ReactDOM.findDOMNode(this).style.color=’red’

子组件

React中虚拟dom跟html中真实的dom是一样的，真实的dom可以设置属性添加事件，虚拟dom也可以添加属性，设置事件，真实的dom我们可以给他添加一些子元素，所以虚拟dom我们也可以给他添加子元素，所以他们的行为表现是一致的。组件也可以看做是一个虚拟dom，所以组件中也可以使用其他组件，添加的这个组件就叫做子组件。

<Nav>

<Title></Title>

</Nav>

例如上面的组件

我们说Title组件就是Nav的子组件，

Nav组件就是Title组件的父组件

既然是组件无论是Nav还是Title，我们都可以为他们添加属性，那么在一个组件中，嵌入另外一个子组件，当为其添加属性的时候，这个如果是父组件中的数据，那此时就是父组件向子组件中传递数据，父组件中数据常用的是属性和状态，所以为子组件添加的数据就有两种，

一种是传递的属性，一种是传递的状态

当传递属性的时候，<Title data={this.props.data}></Title>,

这种传递的数据的特征：通常来说只能传递一次（除非这个父组件又作为其他组件的子组件发生了属性改变）

当传递状态的时候，<Title data={this.state.data}></Title>

这种传递数据的特征是：通常父组件中状态改变就会让子组件发生一次属性的改变，会触发子组件进入存在期，并引发子组件存在期的五个阶段方法的调用

当传递的是自定义数据，<Title data=‘124’></Title>

特征：data属性永远无法改变

存在期

在存在期有五个阶段

第一个阶段表示组件将要接收新的属性 componentWillReceiveProps

第二个阶段表示组件是否应该更新 shouldComponentUpdate

第三个阶段表示组件将要被更新 componentWillUpdate

第四个阶段表示组件更新输出虚拟DOM Render

第五个阶段表示组件已经更新完成 componentDidUpdate