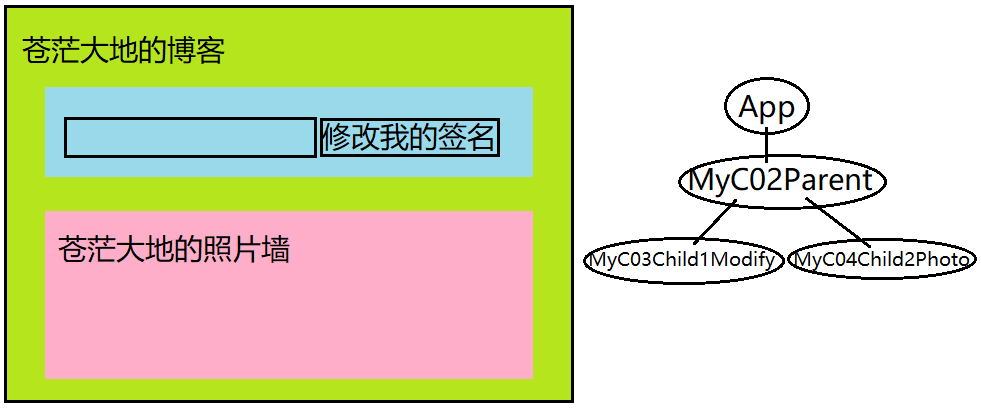
复习：

见思维导图

|  |
| --- |
| Vue.js中父子组件数据传递：Props Down，Events Up  Angular中父子组件数据传递：Props Down，Events Up  React中父子组件数据传递：Props Down，**Props Up** |

1.React中父子组件数据传递



**父=>子：父亲通过子组件的自定义属性，把自己的数据传递下去**

Parent.js：

<Child myNameInChild={this.state.myNameInParent}/>

Child.js：

<h3>{**this.props.**myNameInChild}的照片墙</h3>

**子=>父：父亲通过子组件的自定义属性，把自己的方法传递下去；子组件体内调用此方法，传递实参**

Parent.js：

doModifyMyName=(newName)=>{ }

<Child modifyName={this.doModifyMyName}/>

Child.js：

this.props.modifyName('ABC')

|  |
| --- |
| “Props Up”的原理：  Parent p = new Parent()  p.doModifyMyName = function(newName){ ... }  Child c = new Child()  c.modifyMyName = p.doModifyMyName //父的方法传给子  c.modifyMyName('ABC') |

React中没有直接的兄弟间数据传递机制，只能借助于父组件：兄弟1 => 父组件 => 兄弟2

React中父组件想获得子组件的所有成员，有简便方法：

Parent.js：

constructor(){

super()

this.c3 = React.createRef( ) //创建子组件的引用

}

--------------------------------------

<h3/MyC03 ref={this.c3}> //子组件的引用绑定到某个孩子

--------------------------------------

this.c3.current 就是孩子对象

|  |
| --- |
| Vue.js中父组件获得子组件的引用：  <Child ref="c2" />  this.$refs.c2 |
| Angular中父组件获得子组件的引用：  <Child #c2/>  @ViewChild('c2', {static: true})  private child2 |
| React中父组件获得子组件的引用：  <Child ref={ } /> |

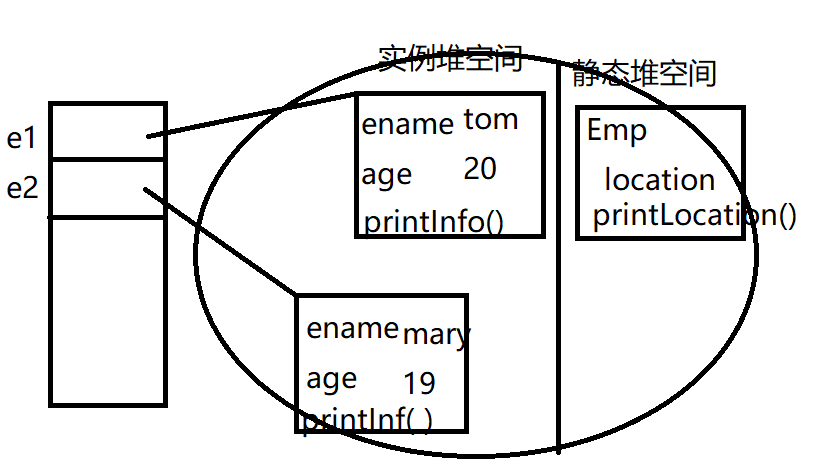
2.理论知识补充：class中的静态(static)成员

类(class)中的成员分为两大类：

①实例成员：实例属性、实例方法

②静态成员：静态属性、静态方法

概念：如果某个class的每一个实例都具备的成员，且值各不相同（例如员工的姓名、年龄），应该声明为实例成员。 如果某个class的所有实例都共用的成员，值在内存中只需保存一份（例如达内所有员工的所在地），应该声明为静态成员。



注意：一般情况下，class内部，静态属性通过静态方法来访问，实例成员通过实例方法来访问。class外部，实例成员需要通过对象的引用来访问；静态成员通过类名来访问，例如：Emp.location、 Emp.printLocation( )

**强化记忆：三大框架中，组件加载完成和即将卸载对应的钩子函数**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Vue.js** | **Angular** | **React** |
| 组件加载完成 | mounted(){ } | ngOnInit(){ } | componentDidMount(){ } |
| 组件即将卸载 | beforeDestroy(){ } | ngOnDestroy(){ } | componentWillUnmount(){ } |

3.面试题：React中组件的声明周期钩子函数 —— 比较凌乱

提示：React不同版本中生命周期钩子函数各不相同！React16.3和之前版本中的都不相同！

React组件的生命周期钩子函数分为三组：

**1)首次渲染相关函数**

①contstructor()

②getDerivedStateFromProps() 用于将this.props转为this.state

③render()

④componentDidMount() 用于初始化组件中的数据，如异步获取服务器端数据

**2)二次渲染相关函数(props属性更改、setState状态修改等)**

①getDerivedStateFromProps() 需要返回转换得到state对象或null

②shouldComponentUpdate() 需要返回true或false

③render()

④componentDidUpdate()

**3)组件卸载相关函数**

①componentWillUnmount() 用于销毁组件创建的长期存在数据，如定时器

4.复习：前端技术中创建移动App的技术有哪些？

①原生开发： Android下用Java/Kotlin，iOS下用OC/Swift

②H5/WebView：使用浏览器内核打开特定的页面 —— Vue.js/Mint-UI

③混编开发：H5/WebView+Cordova/PhoneGap —— Angular/Ionic

④JS Bridge：代码使用JS，使用桥转换，运行时是Java或OC —— React/RN

⑤Flutter：使用全新的Dart语言，直接在手机GPU上绘图

5.RN概述

中文网：https://reactnative.cn/

ReactNative：使用JS语法编写移动App应用，RN会把JS转换为底层Java或OC，最终运行于手机——完全不依赖于浏览器或者WebView组件！

搭建RN应用开发环境：

①下载并安装全局脚手架工具

npm i -g react-native-cli

②运行脚手架工具，创建出空白项目

react-native init 项目名

③下载并安装原生Android/iOS开发软件，创建客户端安装程序

安卓开发需要：Windows + Android Studio + Android SDK

iOS开发需要：Mac OS + XCode

具体过程参考：https://reactnative.cn/docs/getting-started/

最后在项目根目录下得到：android\app\build\outputs\apk\debug\app-debug.apk文件——AndroidApp的安装程序，需要把此APK文件安装到真实手机或者模拟器手机中

④进入空白项目，运行其中的开发服务器——PC机目前是服务器

cd 项目目录

npm start

⑤使用真实手机/模拟器访问测试——Android系统是客户端

Windows服务器的IP地址:8081

|  |
| --- |
| 查看Windows服务器的IP地址：  cmd > ipconfig |

课后任务：

①总结三大框架中的组件生命周期钩子函数

②总结三大框架中父子组件间数据传递的方法

③使用自己的Windows当RN服务器，真实安卓手机或者安卓模拟器(如夜神)当RN客户端，搭建RN开发环境，见到Welcome界面就算成功 —— 中间可能碰到各种错误，试着重新启动、重新安装一下