1. React Native既拥有Native对用户体验，又保留React的开发效率
2. Facevbook于2105.9.15发布，提倡组件化开发
3. Learn once,write everywhere.

(Write once,run anywhere---java)

1. 最先支持ios，通过runtime对ios进行包装，ios7以上
2. 开发环境搭建

1）安装homebrew 2）安装Node.js(brew install node) 3）安装watchman(监控bug及文件变化)

--- brew install watchman 4)安装flow(静态类型检查器，方便找出代码中可能存在对类型错误) --- brew install flow 5)安装React Native ---

(npm install –g react-native-cli)

1. 管理React Native库的版本

1）查看本地版本 react-native –version 或点击项目的package.json文件产看 2）更新本地到最新版本 npm update –g react-native-cli 3）查询版本列表 npm info react-native 4)升级或回退到指定版本 npm install -- save [react-native@0.56](mailto:react-native@0.56).0

7.Flexbox的常用属性

Flexbox布局原理：

7.1)容器属性

a)**flexDirection**: 决定子控件的排向

row：主轴为水平方向，起点在左端

row-reverse：主轴为水平方向，起点在右端

column：（默认值），主轴为垂直方向，起点在上沿

column-reverse：主轴为垂直方向，起点在下沿

b)**justifyContent**:主轴（默认为y轴，当设置了flexDirection为‘row’时，主轴为x轴）对齐方式

flex-start:(默认值),左对齐

flex-end:右对齐

center:中心对齐

space-between:两端对齐,间隔相等

space-around:间隔平局分布



c)**alignItems**:侧轴（默认为x轴）对齐方式

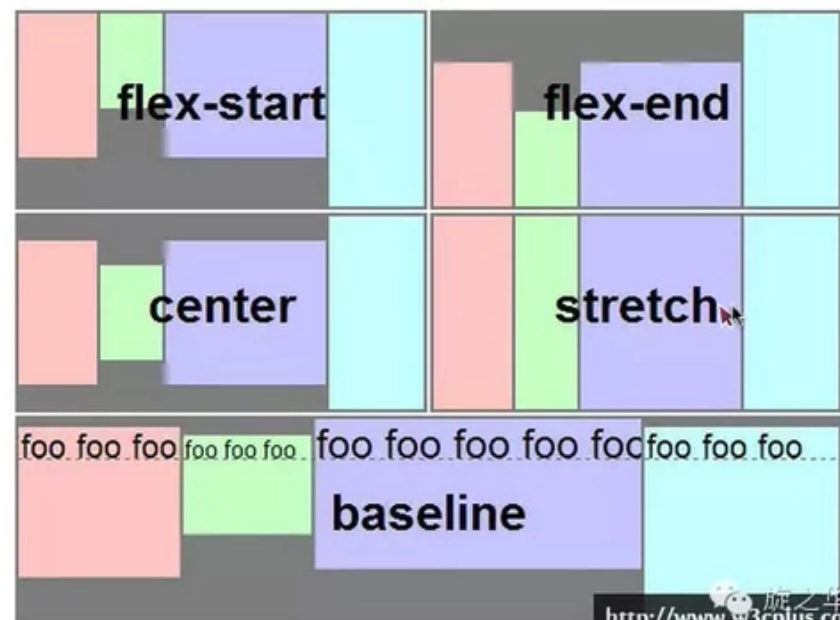
flex-start:起点对齐

flex-end:终点对齐

center:中心对齐

baseline:第一行文字的基线对齐

streth:(默认值)，未设置高度或设置为auto，将占满整个容器的高度



d)**flexWrap**:一条轴线显示不下，如何换行

nowrap:(默认值)，不换行

wrap:换行

7.2)元素属性

a)**flex** 根据父盒子宽高度决定子盒子的宽高度,不设置默认为为控件高度

b)**alignSelf**:单独设置该盒子和其他盒子不一样的顶部对齐方式，可覆盖align-items属性

auto:(默认值)，表示继承父元素align-items,如果没有父元素，等同于strtch

flex-start | flex-end | center | baseline | stretch

8.获取屏幕宽度、高度、分辨率

//定义变量，引入库

var Dimensions = require(‘Dimensions’);

Dimensions.get(‘window’).width | height | scale

9.Image组件加载方式：

1）从项目中加载图片

source={require(“图片路径-相对于当前js文件路径进行查找”)}

2）从App中加载图片（xcode中asset文件夹中，android中drawable文件夹中）

source={{uri:’图片名字’}}

3）从网络加载，一定需要指定图片的尺寸大小

source={{url:https://www.baidu.com/img/logo.png}}

图片属性：

resizeMode:图片的内容模式

Image.resizeMode.stretch 图片被拉伸适应容器的大小，有可能会变形

Image.resizeMode.cover:居中显示，没有被拉伸，超出部分被截断

Image.resizeMode.contain:容器完全容纳图片，等比例拉伸

10.URI、URL、URN

URI：Uniform Resource Identifier统一资源标识符，用来唯一的标识符一个资源，组成：资源的命名机制+存放资源的主机名+资源自身的名称，由路径表示

URL：Uniform Resource Locator统一资源定位符，是Internet上用来描述资源的字符串，组成：协议+存有资源的主机IP地址(有时包括端口)+资源的具体地址

RUN：Universal Resource Name统一资源名称，通过名字来标识资源

11.TextInput常见属性

1）value 默认值

3）keyboardType键盘类型,

eg: keyboardType={‘number-pad’}

4）multiline:true为多行输入，false:单行输入（默认）

eg: multiline={true}

5）password: true为密码输入框，默认false,

注意：当多行输入时，密码输入框失效

6）placeholder:占位文字

7）placeholderTextColor:占位文字颜色

8）clearButtonMode:’never’,’while-editing’,

’unless-editing’,’always’

9)editable:是否可编辑

1）onChangeText函数：监听用户输入的值

2）onBlur函数：当文本输入是模糊当，调用回调函数

11.Touchable组件

11.1不透明触摸 TouchableOpacity:当点击按下的时候，该组件的透明度会降低

<TouchableOpacity activeOpacity={0.5}>

<Text>666</Text>

</TouchableOpacity>

11.2 常见的触摸事件

onPress={()=>this.click()} 点击

onPressIn={()=>this.click()} 按下

onPressOut={()=>this.click()} 抬起

onLongPress={()=>this.click()} 长按

iOS和android平台选择

Platform.select({

ios: ‘在苹果手机显示的内容’,

android: ‘在安卓手机显示的内容’,

});

//注意要导入Platform控件

自定义控件输出：module.exports = 自定义类名

导入使用：var A = require(‘./类名’)；

使用：<类名 />

初始化类写法

ES5

Var 类名 = React.createClass({….})

ES6

Class 类名 extends Component{….}

React Native组件生命周期:

包括 实例化阶段、存在阶段和销毁阶段

实例化阶段函数：

1.getDefaultProps:初始化一些默认而且不可以修改的属性，初始化后，再次使用该组件不会再调用该函数

2.getInitialState: 初始化组件的一些状态，会多次调用，可以将控制控件的状态的一些变量放在这里初始化，比如控件的文字，通过this.state来取值，通过this.setSteate({}) 来修改state值。（**注意：**一旦调用了this.setState方法，组件一定会调用render方法，对组件进行再次渲染，不过，如果React框架会自动根据DOM的状态判断是否需要真正的渲染）

3.componentWillMount相当于oc的viewwillappear，在组件将要被加载到视图之前调用

4.render是一个组件中必须有的方法，本质上是一个函数，并返回jsx或其他组件来构成DOM，此外，在render函数中，只可通过this.state和this.props来访问之前函数中初始化的数据

5.componentDidMount在调用render方法后，组件加载成功并被成功渲染出来后，所要执行的后续操作，一般在该函数中处理网络请求、异步操作等

ScrollView组件

1.两个要点：

a)Scrollview必须有一个确定的高度才能正常工作。通常有两种做法：第一种：直接给scrollview进行设置高度（不建议）第二种：scorllviw中不要加 flex：1

b)scrollview内部的其他响应者尚无法阻止scrollview本身成为响应者

2.常用属性

contentContainerStyle:

horizontal: 为true时，子视图会水平排成一行，默认垂直

keyboardDismissMode 拖拽视图时，是否隐藏键盘 none：（默认值）不隐藏， on-drag：拖拽开始时候隐藏

interactive:

keyboardShouldPersistTaps: false(默认值):点击其他地方，键盘隐藏，true:不消失

onScroll : function 在滚动过程中调用，每帧最多调用一次此回调函数，调用频率可以用scrollEventThottle控制

refreshControl 下拉刷新功能

showHorizontalScrollIndicator 是否显示水平滚动条

showVerticalScrollIndicator 是否显示垂直方向滚动条

alwaysBounceHorizontal 水平方向弹簧效果

获取真实DOM节点

在React Native中，组件并不是真实的DOM节点，而是存在于内存之中的一种数据结构，叫做虚拟DOM，只有当它插入文档后，才会变成真实的DOM

根据React的设计，所有DOM变动，都先在虚拟DOM上发生，然后将实际发生变动的部分，反应在真实DOM上，这种算法叫做DOM diff算法，可以极大提高网页的性能

给组件设置 ref=“标记名称”，通过 this.refs.标记名称获取对应的DOM

ES5和ES6的差异化：es5和es6都是对ecmascript规范的补充，es5已经大规模使用了，es6目前可能在个别平台存在浏览器兼容性问题。

区别1:创建组件

es5中使用React.createClass()

es6中必须继承React.component

区别2:组件的属性props

es5中通过函数getDefaultProps(){return{name:value}}

es6中通过类名.defaultProps={name:value}(可以表示static定义在class内，也可以定义在class外)

区别3:组件的状态state

es5中通过函数getInitialState(){return{name:value}}

es6通过构造器

constructor(props){

super(props);

this.state={name:value}

}

引入类库步骤：

1.终端进入RN项目路径，输入

npm i react-timer-mixin(类库名) --save

2.在使用地方引入

var TimerMixin = require(“react-timer-mixin”);

3.注册

mixins:[TimerMixin]