React Native

React-Native入门

平安科技移动开发一队　　黄志君

总体介绍

React-Native是神马?

1. 简单来说，react-native就是用javascript代码来调用objective-c / java 的一种技术。

2. react-native最终运行的是ios/andriod的原生态控件。

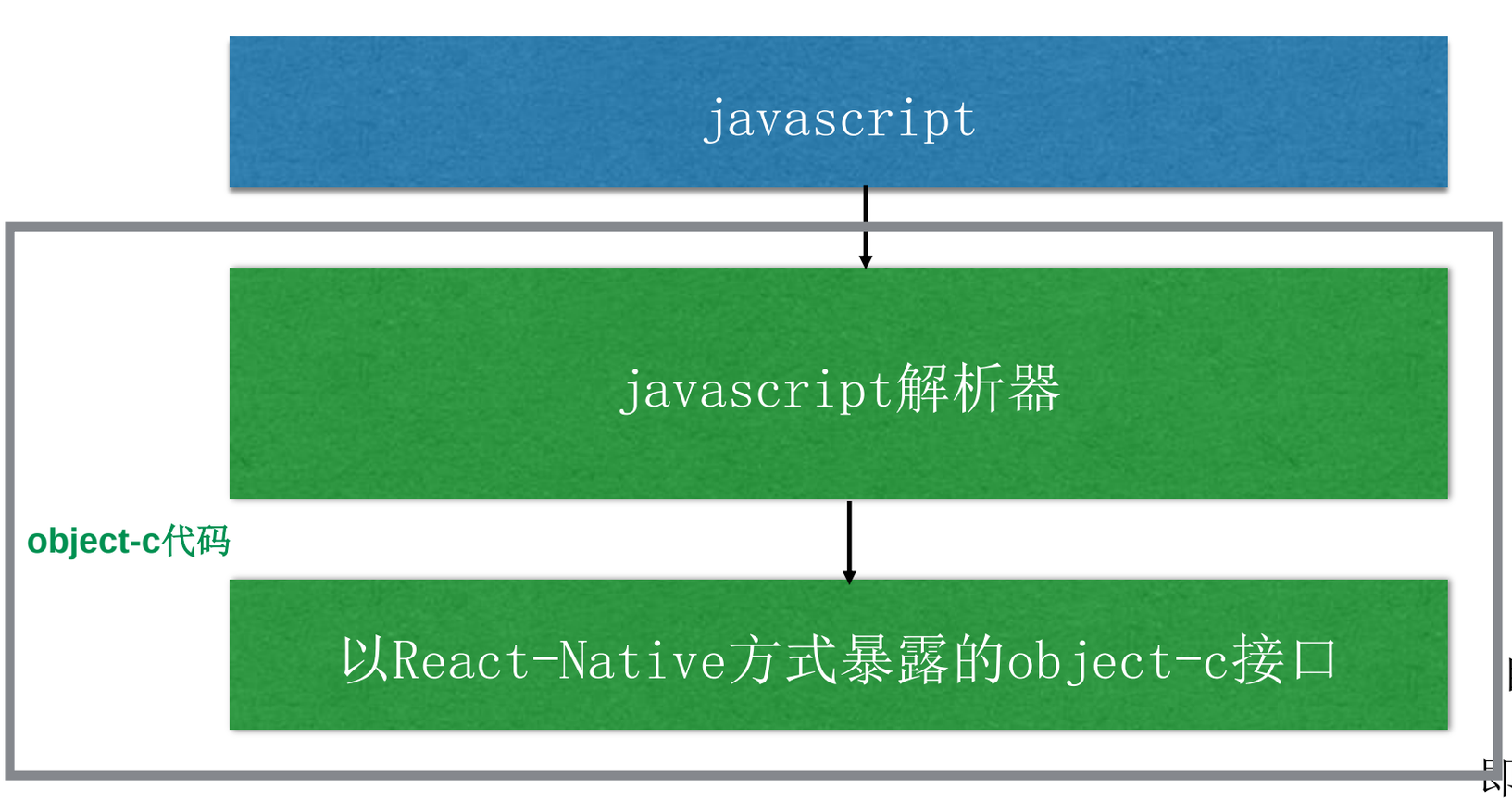
3. react-native 的目的是开发者既有Native的用户体验，又保留React(javascript css)的开发效率。

4.react-native 只作为MVC结构的View展现层。

5.对于IOS开发人员来说，可以也可以理解为 React-Native是一个SDK，一个能用js代码调用IOS代码的SDK。

React-Native是如何实现的呢？

React-Native 通信机制



javascript可以以下载方式替换
实现界面的显示热更新
（前提是调用的类和方法都已实现）

react-native的核心代码

react-native已暴露大部分的标准组件
自定义的组件按react-native格式暴露接口
javascript就可以调用
即可以自己扩展想要的接口给javascript调用

React-Native好处是什么

1.界面热更新，当然前提条件是js调用的object-c接口都存在。

2.对比Html5技术，react-native用的是原生控件，有更好的体验。

3.尽可能地往跨平台靠拢。

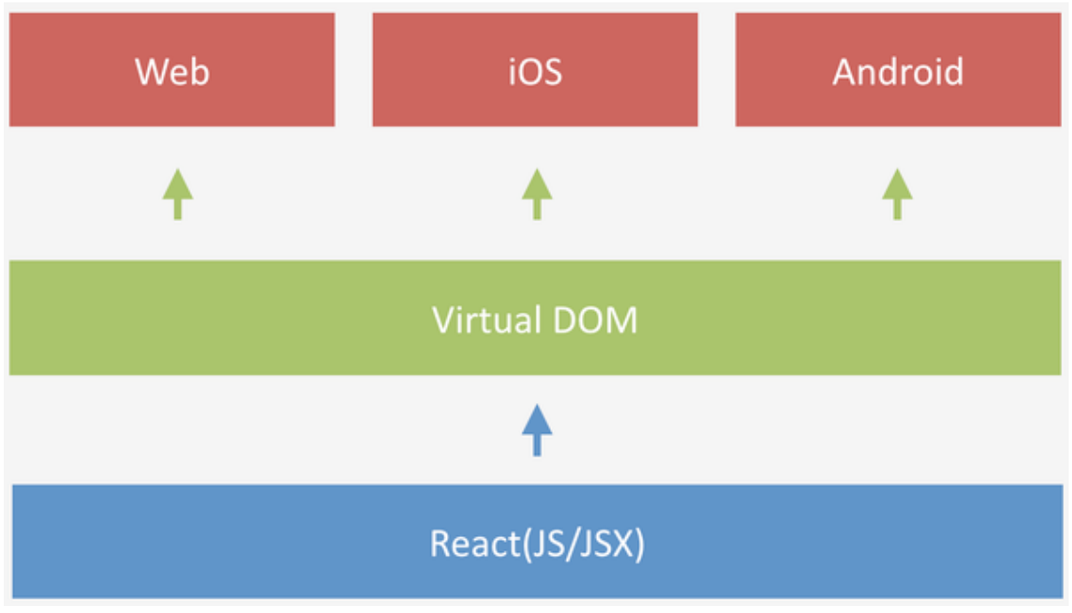
react-native的思想是Learn once, write anywhere，

因为不同Native平台上的用户体验是不同的，

React Native不强求一份原生代码支持多个平台，

所以不提“Write once, run anywhere”（Java），

提出了“Learn once, write anywhere”。



还有什么好处么??

· 支持异步执行
JavaScript应用代码和原生平台之间的所有操作都是异步执行的，原生模块还可以使用额外的线程，从而利用React Native开发出来的应用性能比较高（运行流畅和反应快）。此外，开发者还能够在模拟器或者物理设备上运行应用的同时利用Chrome Developer Tools调试JavaScript代码；

· 触摸操作
React Native实现了一个类似iOS平台下的响应系统，还提供了高级的组件如TouchableHighlight等；

· 引入了Flexbox布局模型和样式
Flexbox布局模型有利于构建常见的UI布局，如stacked和nested boxes布局。React Native还支持常见的Web样式，如fontWeight、font-size等。样式表（StyleSheet）抽象提供了一种优化机制来声明组件 所用到的所有样式和布局；

· 较强的可扩展性
设计React Native主要是为了使得开发者使用常规的原生视图组件扩展和模块就可以开发出一个完整的应用，开发者能够复用已经构建的任何应用或者组件，并且还能够引入自己喜爱的原生Library。

一个IOS开发者学习
React-Native需要的前置知识点

1.Html & Css

http://www.w3school.com.cn/

http://www.runoob.com/

2.Javascript（ECMAScript6新特性 ajax jQuery node.js, CommonJS规范 等）

http://www.runoob.com/

《JavaScript权威指南（第6版）》

《JavaScript高级程序设计》

慕课网的视频教程

3.React （jsx ）

http://reactjs.cn/react/docs/getting-started.html

http://www.ruanyifeng.com/blog/2015/03/react.html

慕课网的视频教程

4.React-Native (Flexbox布局)

http://reactnative.cn/

http://reactnative.cn/docs/getting-started.html

https://github.com/ele828/react-native-guide

http://www.ruanyifeng.com/blog/2015/07/flex-grammar.html

开发环境搭建

React-Native Mac环境开发环境搭建

1.安装Homebrew （osx的包管理工具）:

ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"

2.安装node.js ：

brew install node

3. 安装watchman ：

brew install watchman

4.安装flow ：

brew install flow

5.设置npm淘宝镜像：（也可参考http://npm.taobao.org/）

5.1 创建 .npmrc 文件（在用户主目录下）

5.2 加入以下配置信息：registry = http://registry.npm.taobao.org

6.安装React-Native ：

sudo npm install -g react-native-cli

7.创建React-Native项目HelloWorld:

react-native init HelloWorld3

JavaScript开发IDE

1. WebStrom
2. Atom + Nuclide

React Native 常用语法

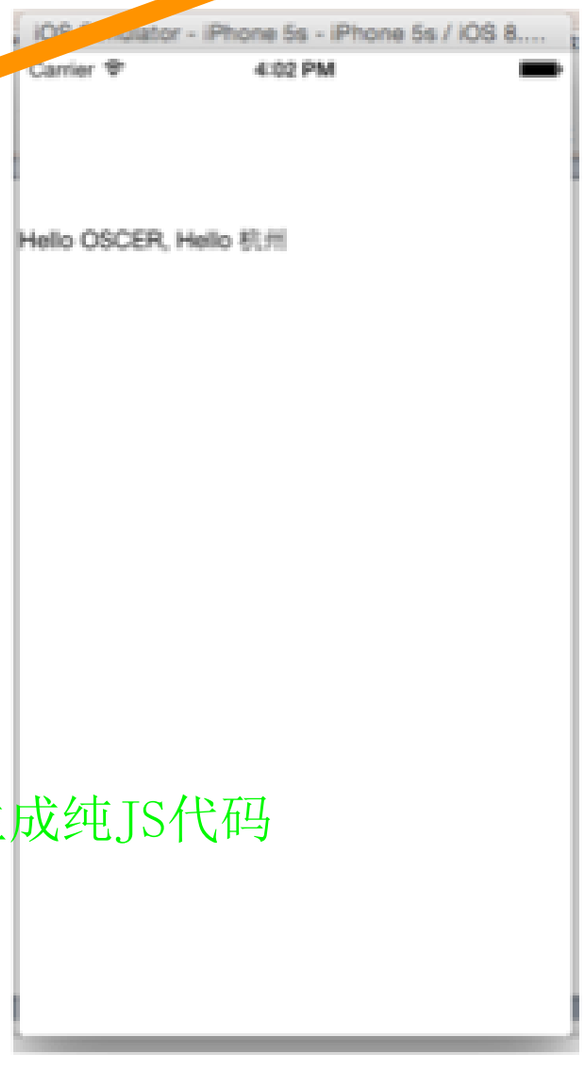
常用语法之初体验

- **Require：引入模块**
- **React.createClass创建组件类 (HZ 可以类比UIViewController的派生类)**
- **render方法渲染视图 (render方法可以类比UIViewController.view)**
- **JSX & XML DOM**
- **AppRegistry注册应用入口**

```
1
2  //引入react-native模块
3  var React = require('react-native');
4  var {
5    StyleSheet,
6    View,
7    Text,
8    AppRegistry,
9  } = React;
10
11 var HZ = React.createClass({
12   render: function(){
13     return(
14       <View style={{marginTop:100}}>
15         <Text>Hello OSCER, Hello 杭州</Text>
16       </View>
17     );
18   }
19
20 });
21
22 //注册react-native应用
23 AppRegistry.registerComponent('hz', () => HZ);
24
```

JSX格式编写视图，经过语法解析后回生成纯JS代码

```
//语法糖形式的写法。
//加载组件代码， 等同于：
var StyleSheet = React. StyleSheet;
var View = React.View;
var Text = React.Text;
var AppRegistry = React. AppRegistry
```



常用语法之CSS规则

```
12  var HZ = React.createClass({
13    render: function(){
14      return(
15        <View style={{padding:30, backgroundColor:'#3BC1FF', flex:1}}>
16          <Text style={styles.text}>Hello OSCER, Hello 杭州</Text>
17        </View>
18      );
19    }
20
21  });
22  //创建样式类
23  var styles = StyleSheet.create({
24    text:{
25      color:'#FFF'
26    }
27  });
```

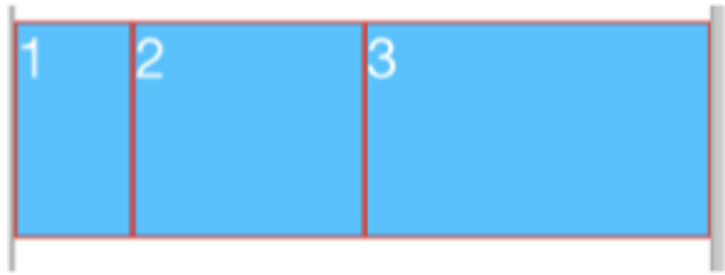
- 内联样式
- 类样式

- 内联样式：第一个{ }是js语法，相当于模板EJS引擎的<%%>
- 内联样式：第二个{ }是js对象或者JSX语法的组件
- 类样式： StyleSheet.create创建样式，可以引用text对象
- 样式驼峰命名

常用语法之Flex布局

```
var HZ = React.createClass({
  render: function(){
    return(
      <View style={{marginTop:30, flex:1}}>
        <Text style={styles.text}>Hello OSCER, Hello 杭州</Text>
        <View style={{height:100, flexDirection:'row', backgroundColor:'#3BC1FF',}}>
          <View style={[{flex:1}, styles.border]}>
            <Text style={styles.text}>1</Text>
          </View>
          <View style={[{flex:2}, styles.border]}>
            <Text style={styles.text}>2</Text>
          </View>
          <View style={[{flex:3}, styles.border]}>
            <Text style={styles.text}>3</Text>
          </View>
        </View>
      </View>
    );
  }

});
```



- flex：比例1＋2+3
- flexDirection: row or column
- alignItems：水平居中
- justifyContent: 垂直居中

```
  <View style={{marginTop:20,height:60,borderWidth:1,
    borderColor:'#000', alignItems:'center', justifyContent:'center'}}>
    <View style={{width:30, height:30, borderWidth:1,borderColor:'red'}}></View>
  </View>
</View>
```

常用语法-组件化&自定义组件

```
var Item = React.createClass({
  render:function(){
    return (
      <View>
        <Image style={{width:120,height:60, resizeMode:Image.resizeMode.contain}}
               source={{uri:'http://www.oschina.net/img/logo_s2.png'}}></Image>
        <Text>开源中国</Text>
      </View>
    );
  }
});
module.exports = Item;

var HZ = React.createClass({
  render: function(){
    return(
      <View style={{padding:30, flex:1}}>
        <Item></Item>
        <Item></Item>
        <Item></Item>
      </View>
    );
  }

});
```

- module.exports
- 组件引用require
- 组件得好处？ & 颗粒化
- 复杂组件如何做，有需要哪些要素？--下一张ppt

常用语法-数据状态引用—props & state & ref

```
var Item = React.createClass({
  getInitialState:function(){
    return {
      name:'开源中国开源世界'
    };
  },
  componentWillMount: function(){
    console.log('2');
  },
  render:function(){
    console.log('3');
    return (
      <View>
        <Text style={{color:'red'}}>{this.props.num}</Text>
        <Image style={{width:120,height:60, resizeMode:Image.resizeMode.contain}}
               source={{uri:'http://www.oschina.net/img/logo_s2.png'}}></Image>
        <Text>{this.state.name}</Text>
      </View>
    );
  },
});

module.exports = Item;
var HZ = React.createClass({
  render: function(){
    return(
      <View style={{padding:30, flex:1}}>
        <Item num="1"></Item>
        <Item num="2"></Item>
        <Item num="3"></Item>
      </View>
    );
  }
});
```

初始化this.state对象

Item的props属性在父组件渲染时候传入

- **props属性，扩展内部视图和数据model**
- **state状态**
- ref引用

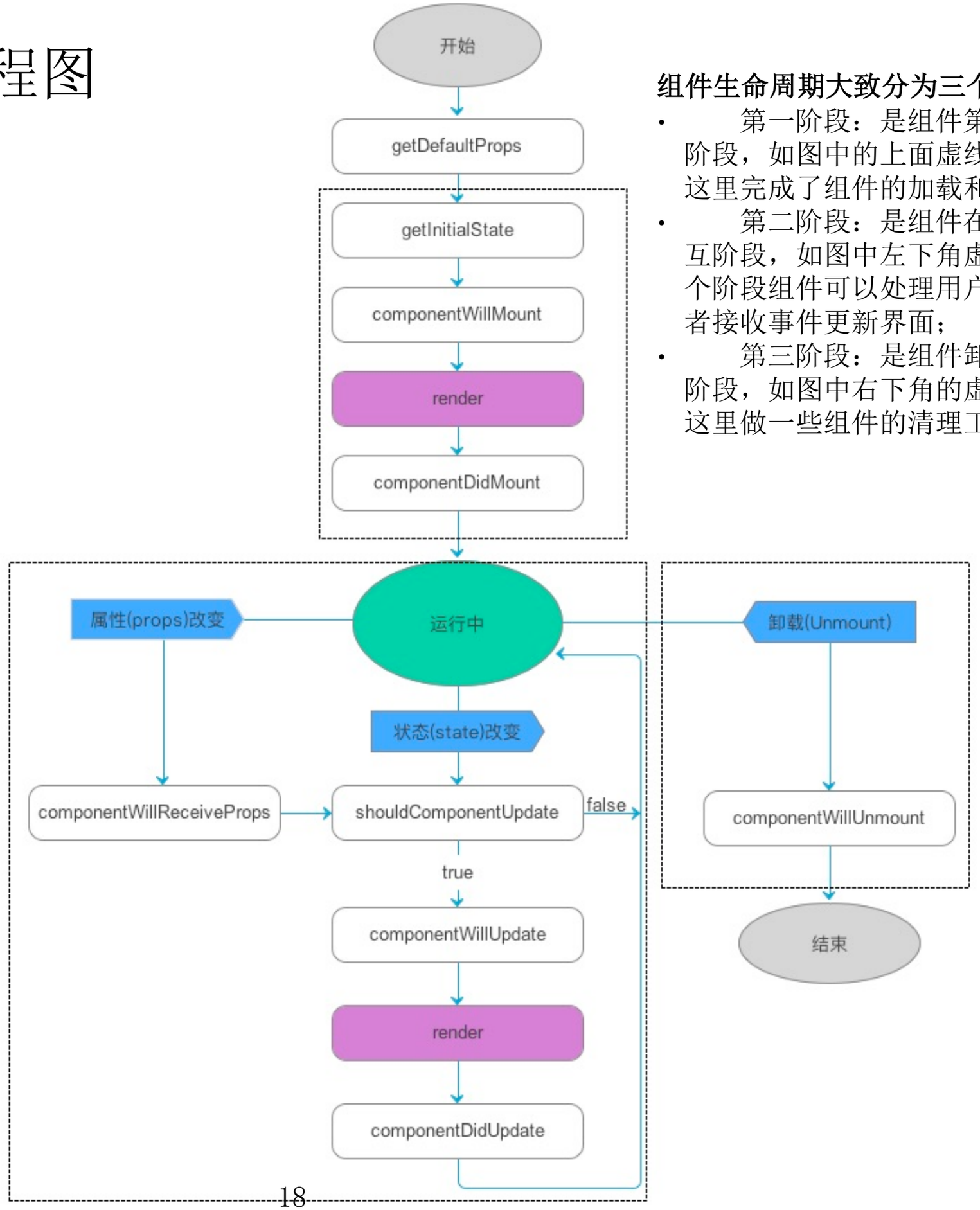
常用语法-组件生命周期

```
var Item = React.createClass({
  getInitialState:function(){
    console.log('1');
    return null;
  },
  componentWillMount: function(){
    console.log('2');
  },
  render:function(){
    console.log('3');
    return (
      <View>
        <Image style={{width:120,height:60, resizeMode:Image.resizeMode.contain}}
               source={{uri:'http://www.oschina.net/img/logo_s2.png'}}></Image>
        <Text>开源中国</Text>
      </View>
    );
  },
  componentDidMount: function(){
    console.log('4');
  },
  componentWillUnmount: function(){
    console.log('5');
  }
});
```

- getInitialState
- componentWillMount
- Render
- componentDidMount

RN组件生命周期流程图

生命周期	调用次数	能否使用setSate()
getDefaultProps	1(全局调用一次)	否
getInitialState	1	否
componentWillMount	1	是
render	>1	否
componentDidMount	1	是
componentWillReceiveProps	>=0	是
shouldComponentUpdate	>=0	否
componentWillUpdate	>=0	否
componentDidUpdate	>=0	否
componentWillUnmount	1	否



组件生命周期大致分为三个阶段：

- 第一阶段：是组件第一次绘制阶段，如图中的上面虚线框内，在这里完成了组件的加载和初始化；
- 第二阶段：是组件在运行和交互阶段，如图中左下角虚线框，这个阶段组件可以处理用户交互，或者接收事件更新界面；
- 第三阶段：是组件卸载消亡的阶段，如图中右下角的虚线框中，这里做一些组件的清理工作。

DEMO演示

HelloWorld工程讲解

1.代码结构。

2.React-Native javascript代码分析。

2.1CommonJS规范:

CommonJS定义的模块分为:{模块引用(require)} {模块定义(exports)} {模块标识(module)}

2.2 界面刷新触发方式 this.setState(). 刷新本类的控件和它挂载的子控件.

3.React-Native object-c代码简单分析。

3.1 AppDelegate.m查看加载React-Native 界面方式。

3.2 React-Native 核心objective-c 代码库浏览。

4.React-Native 已实现的JS组件和接口简单浏览,跑demo。

http://reactnative.cn/docs/getting-started.html

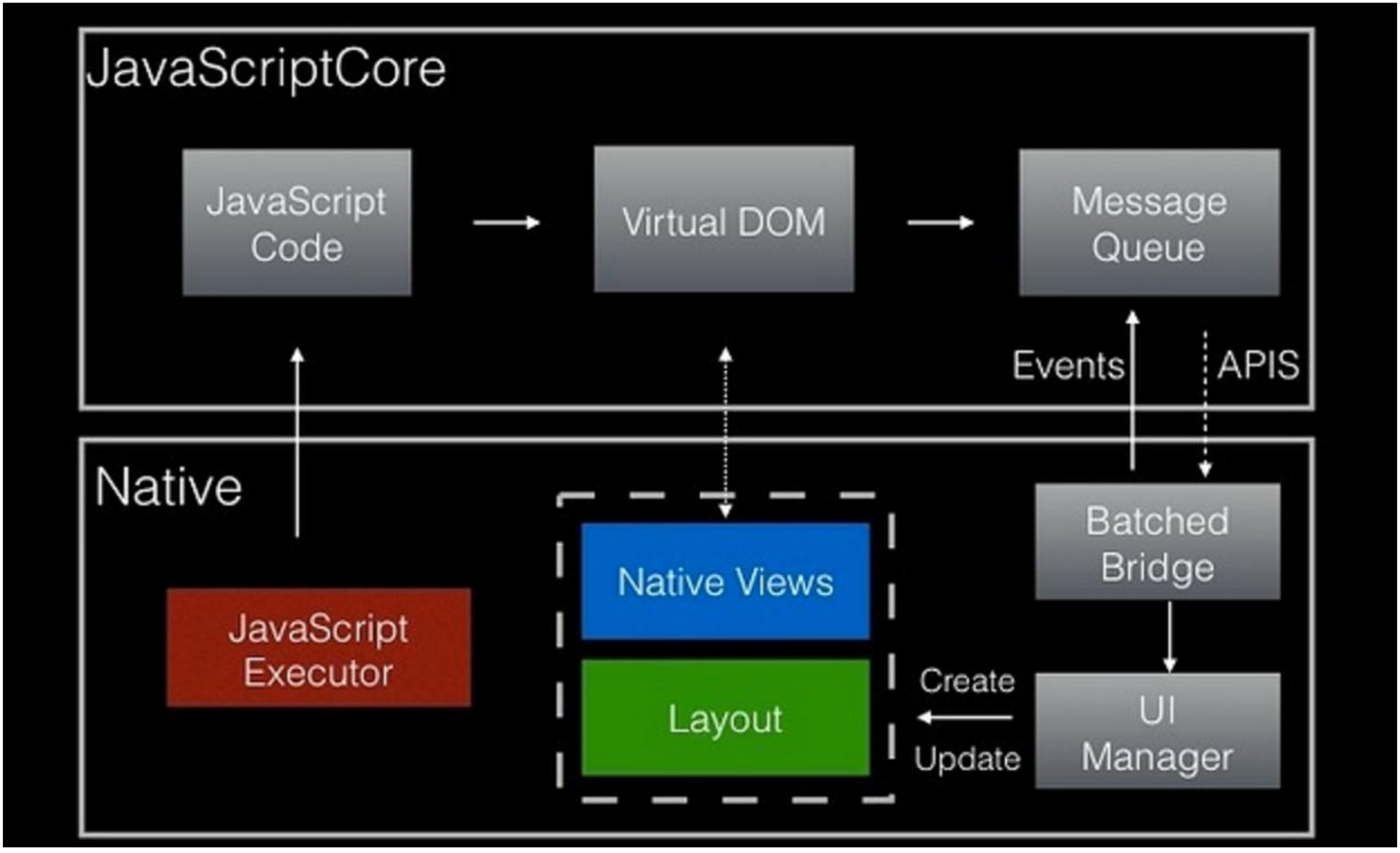
Javascript调试

1.在模拟器上 control+D,显示菜单，点击 “Debug in Chrome”

2.模拟器直接刷新界面control+R。

React-Native开发模式

React-Native原理



从　关注分支　到　关注状态

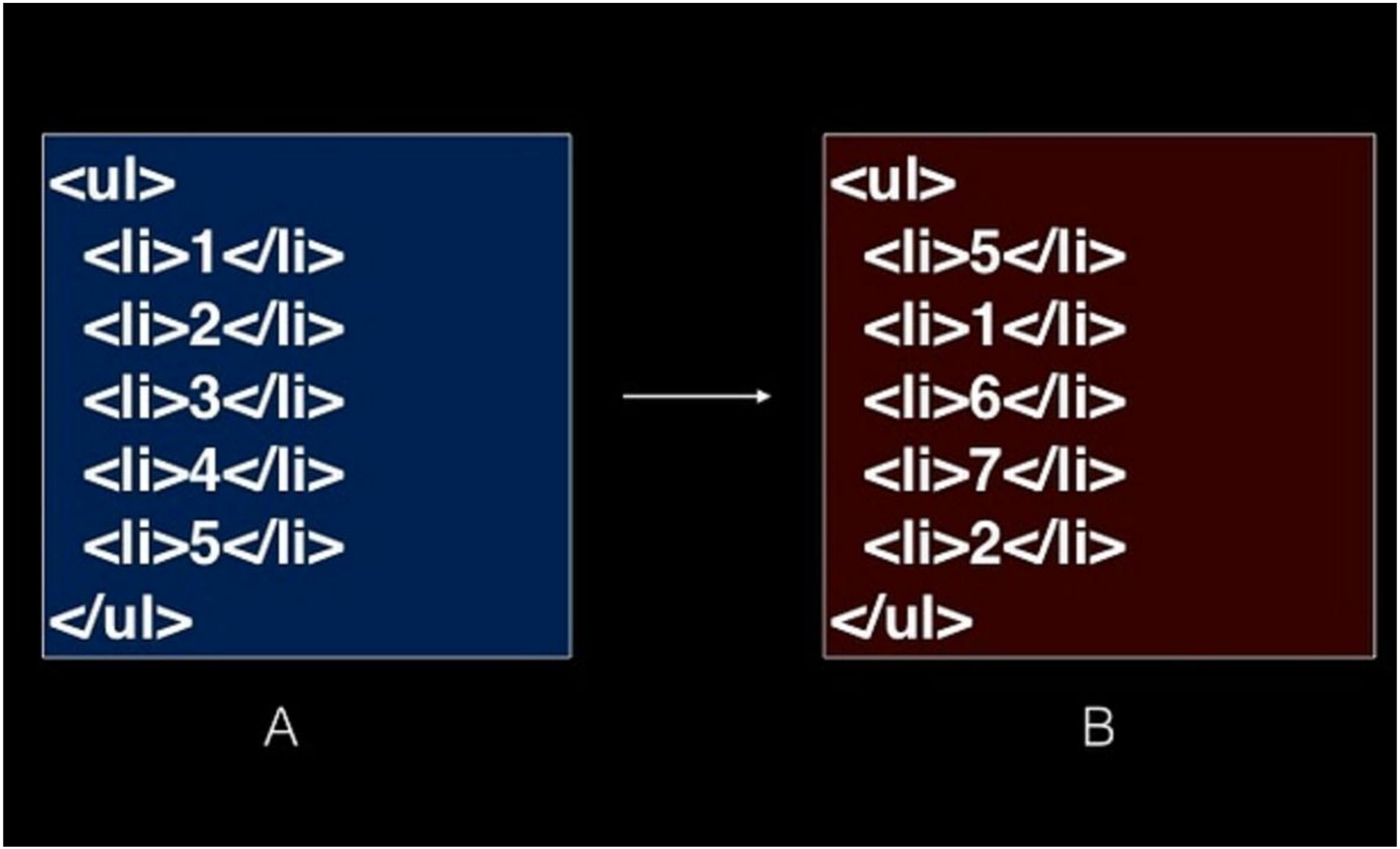
```
if (dialog.isShow()) {   // 分支1
  dialog.hide()
}

if (dialog.isHide()) {    // 分支2
  dialog.show()
}
```

↓

```
<Dialog show={this.state.show} />  // 状态1
```

从　关注过程　到　关注数据



1
2
3
4
5

5
1
6
7
2

A
B

从　关注过程　到　关注数据

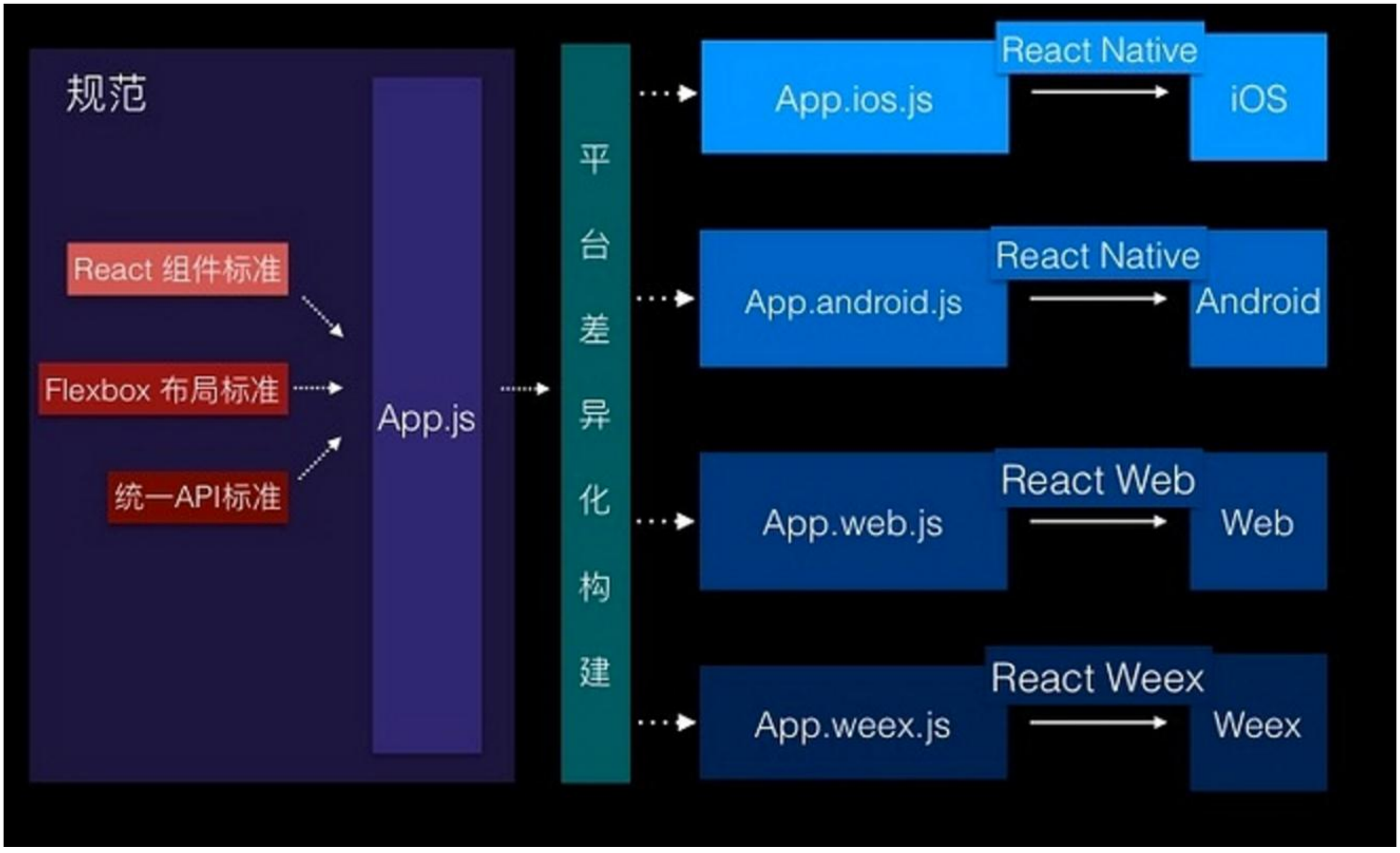
```
append child 1
append child 2
append child 3
append child 4
append child 5
```

```
getInitialState() {
  return {
    list: [1, 2, 3, 4, 5]
  }
}
render() {
  var list = this.state.list;
  return (<ul>
    {list.map(function(li, index){
        return <li key={index}>{li}</li>
    })}
    </ul>)
}
```

```
append child 6
append child 7
remove child 3
remove child 4
move child 5
move child 1
move child 2
```

```
var newData = { list: [5, 1, 6, 7, 2] }
this.setState( newData)
```

设计原则：面向标准，解耦实现



坑点& 总结

1.React-Native只有Native （IOS andriod）实现，andriod还不够完整。

2.注意要熟悉 Javascript 的ECMAScript5规范和ECMAScript6规范的差别。
(http://bbs.reactnative.cn/topic/15/react-react-native-%E7%9A%84es5-es6%E5%86%99%E6%B3%95%E5%AF%B9%E7%85%A7%E8%A1%A8)

谢谢大家！