React Native

Learn once, write anywhere

What is the  React?

"A JAVASCRIPT LIBRARY FOR BUILDING USER INTERFACES"

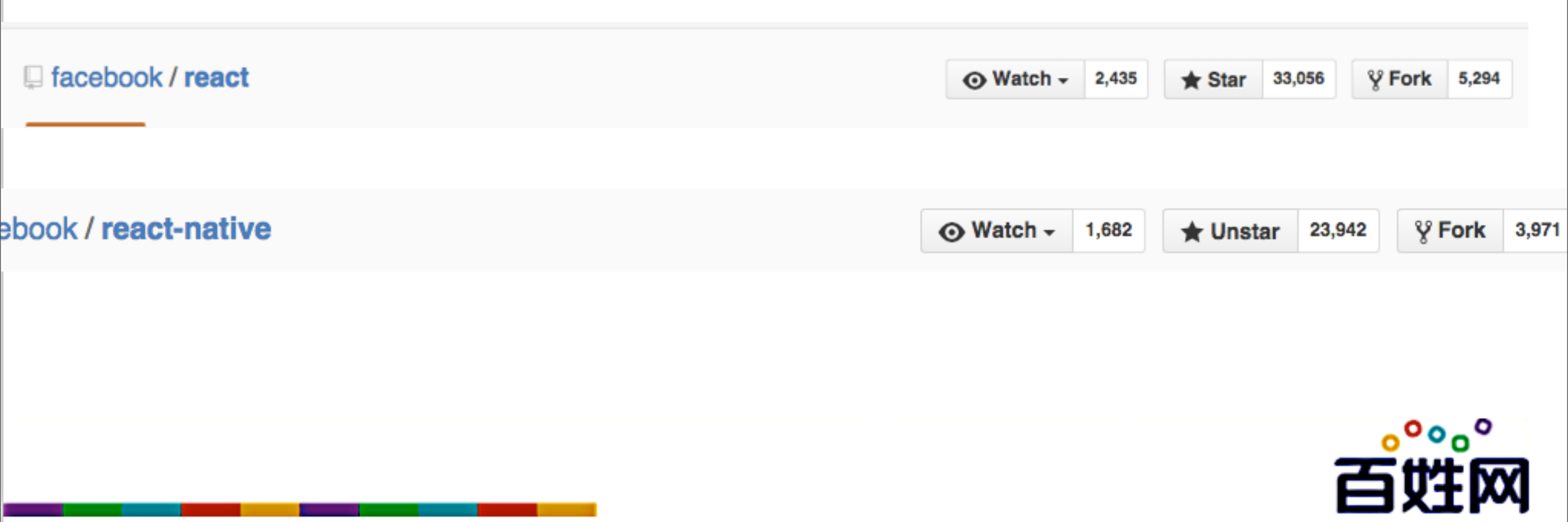
“一个用来构建 **用户界面** 的 javascript 库”。

Why niu

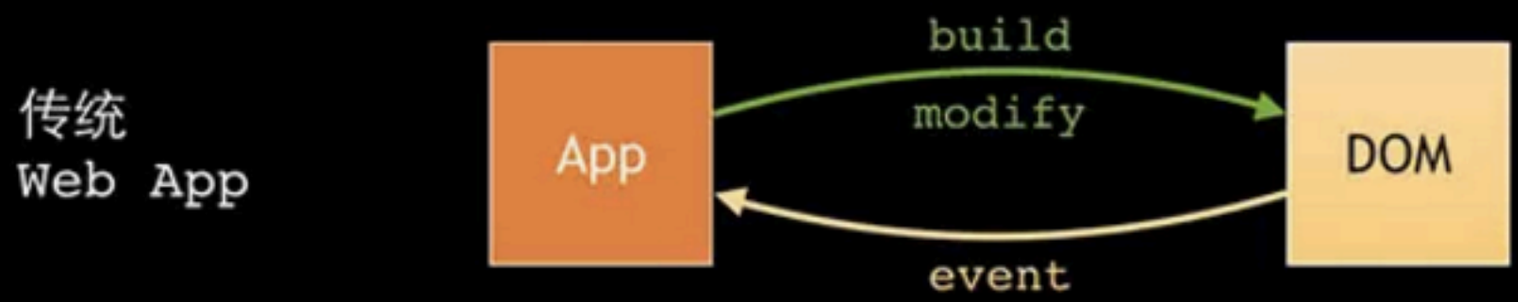
- 极速的渲染性能
- 组件互相独立，关系隔离，可复用
- 跨平台
- Facebook和Instagram帮我们做了充足的测试
  - Facebook MAU 13.9亿，移动端 7.45亿
  - Instagram MAU 3亿

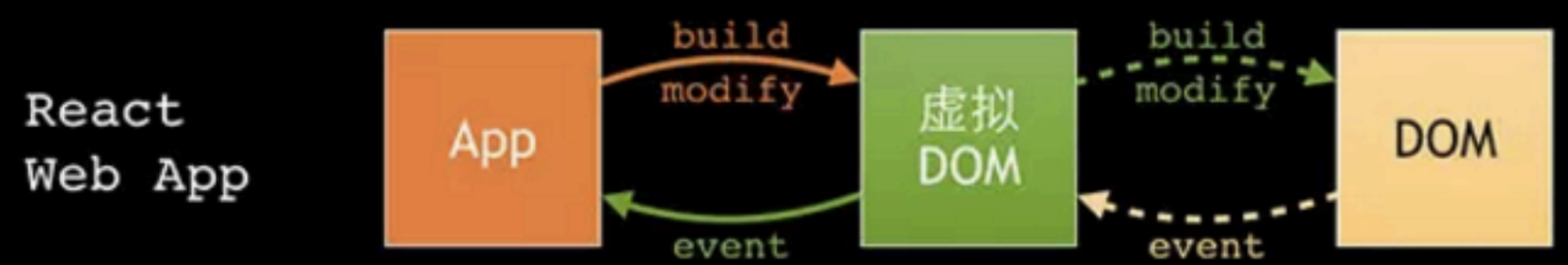
Apps using React Native

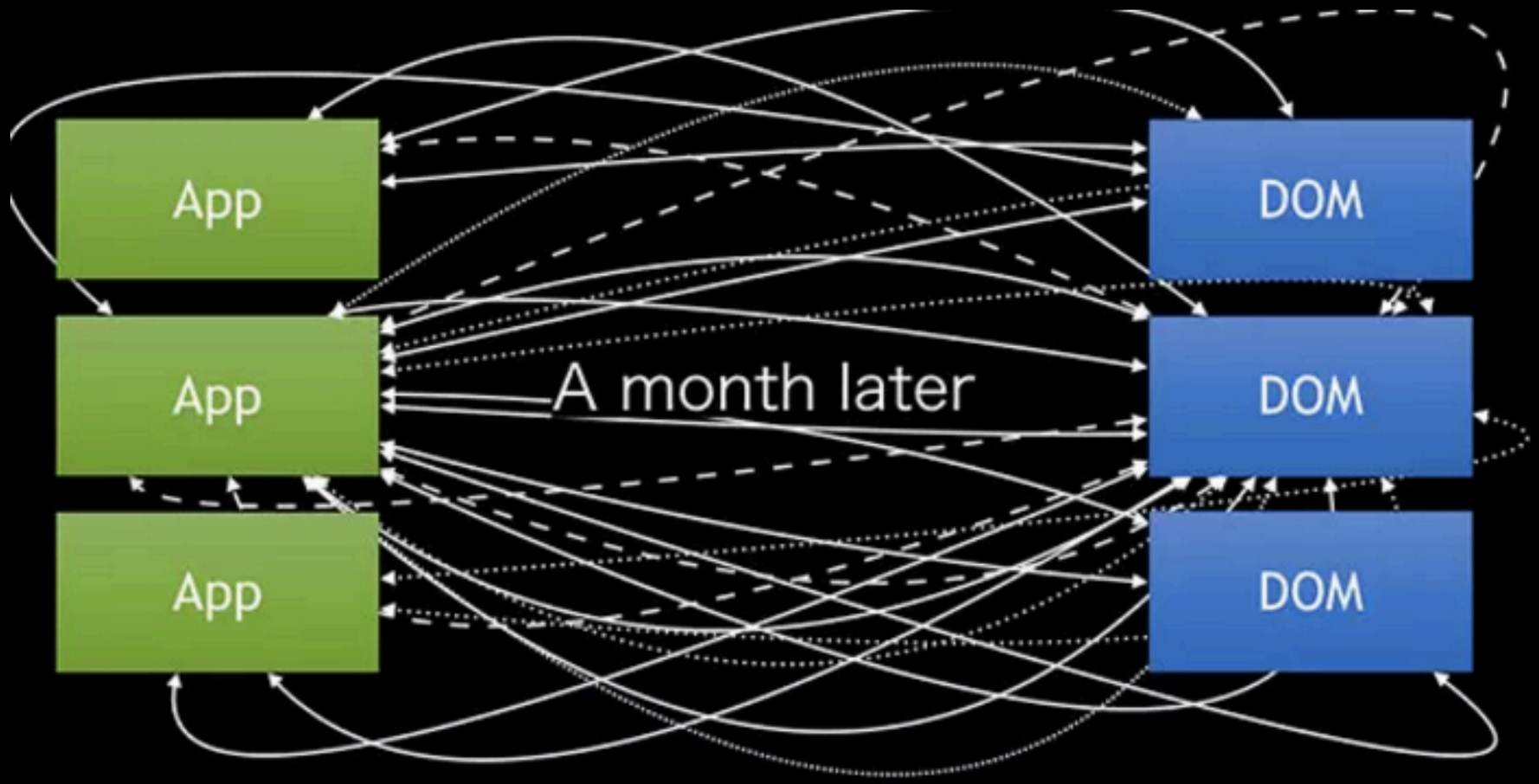
https://facebook.github.io/react-native/showcase.html



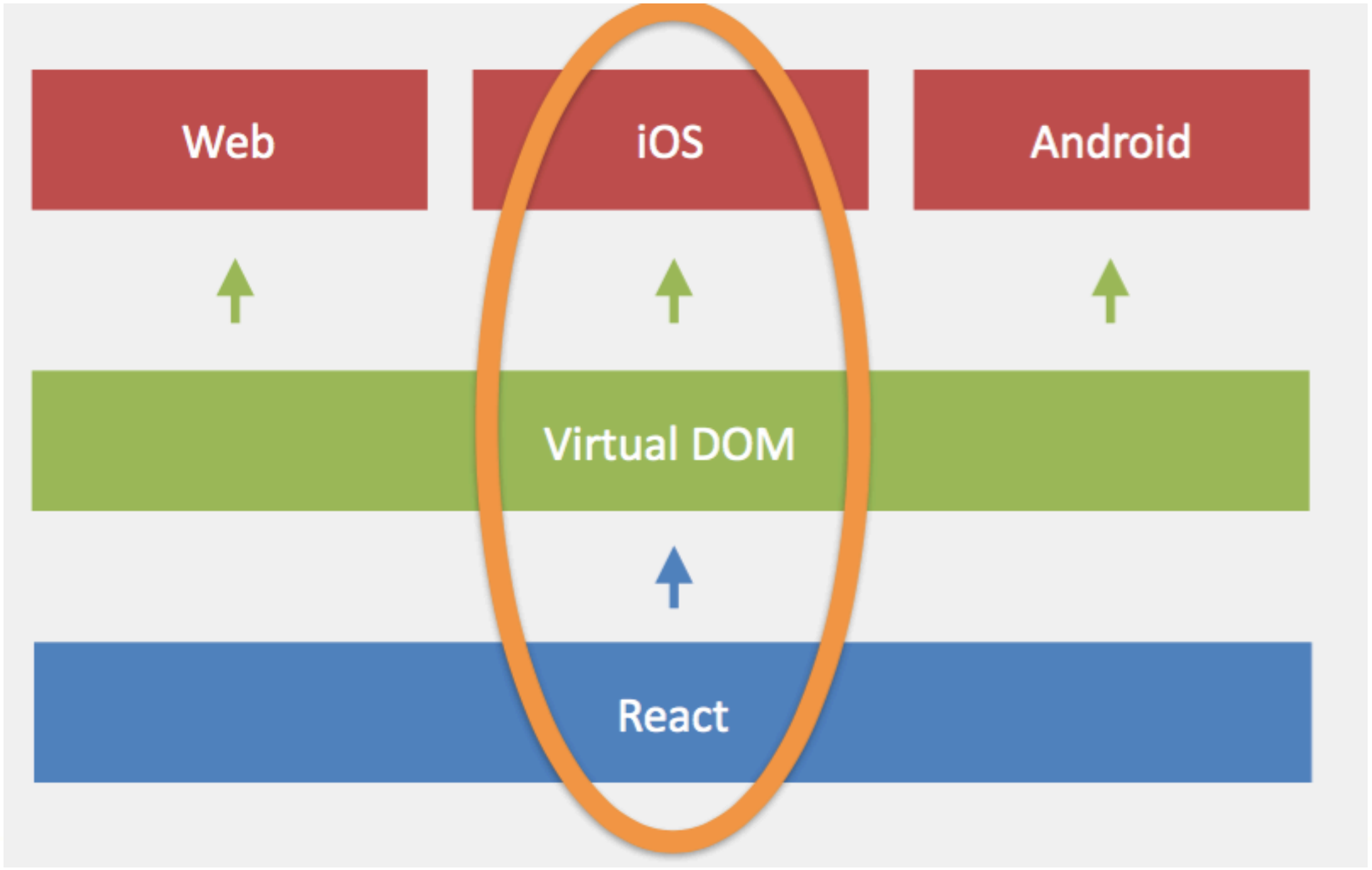
Virtual Dom


传统
Web App
App
build
modify
event
DOM

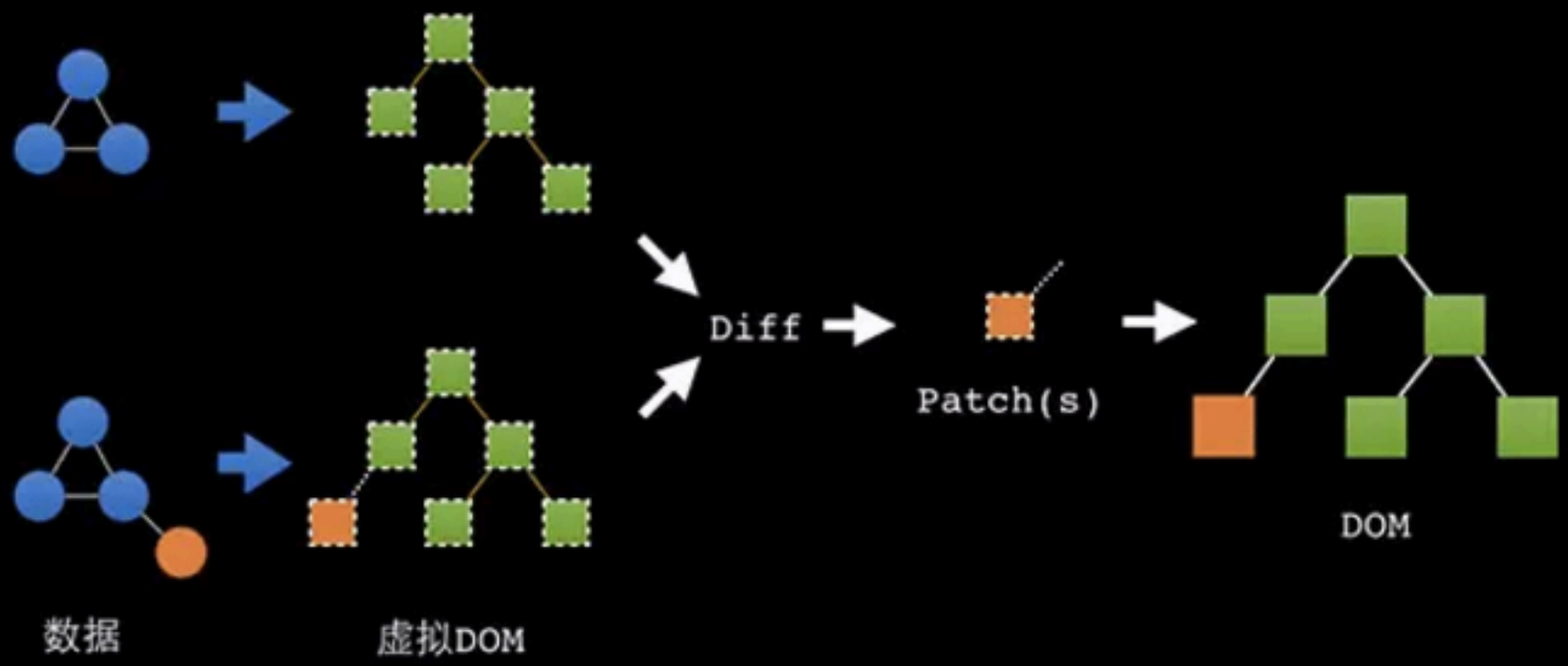

React
Web App
App
build
modify
event
虚拟
DOM
build
modify
event
DOM


App
App
App
DOM
DOM
DOM
A month later

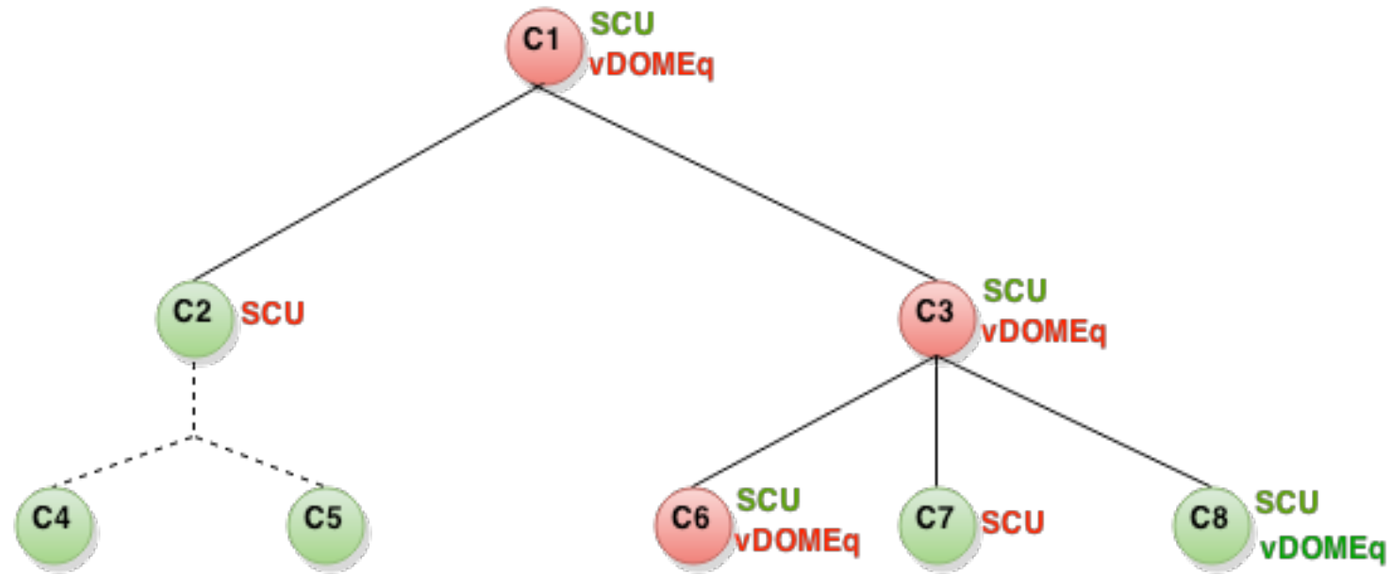
浏览器的工作原理

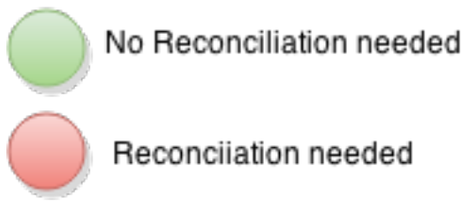


Advanced Performance



shouldComponentUpdate





SCU
SCU
shouldComponentUpdate?

vDOMEq
vDOMEq
are virtual DOMs equivalent?

Q：RN和React.js是一个东西吗?

A：RN和React.js共用一些抽象层，但具体有很多差异，且目标平台不同：
RN目前只能开发iOS/Android App，而React.js用于开发web页面。

Q：RN所支持的最低iOS和Android版本?

A：Android >= 4.1 (API 16)

iOS >= 7.0

Q：RN和cordova/phonegap是一个东西吗?

A：不一样。RN不是一个webview（但包含了webview组件），不能直接复用web页面代码。
RN的性能接近原生，超过cordova/phonegap。

组件!

组件列表

- ActivityIndicatorIOS
- DatePickerIOS
- DrawerLayoutAndroid
- Image
- ListView
- MapView
- Navigator
- Modal
- NavigatorIOS
- PickerIOS
- ProgressBarAndroid
- ProgressViewIOS
- PullToRefreshViewAndroid
- ScrollView
- SegmentedControlIIOS
- SliderIOS
- SwitchAndroid
- SwitchIOS
- TabBarIOS
- TabBarIOS.Item
- Text
- TextInput
- ToolbarAndroid
- TouchableHighlight
- TouchableNativeFeedback
- TouchableOpacity
- TouchableWithoutFeedback
- View
- ViewPagerAndroid
- WebView

component = data + jsx

```
var Hello = React.createClass({
  propTypes: {
    name: React.PropTypes.String.isRequired
  },

  getInitialState: function () {
    return {
      name: this.props.name
    };
  },

  render: function () {
    return (
      <ul>
        <li>hello {this.state.name}!</li>
      </ul>
    );
  }
});
```

props
属性，数据源

state
来自props

JSX
通过state控制DOM结构变化

state与props

不要改变prop值

props 主要作用是提供数据来源，可以简单的理解为 props 就是构造函数的参数。 state 唯一的作用是控制组件的表现，用来存放会随着交互变化状态，比如开关状态等。

JSX 做的事情就是根据 state 和 props 中的值，结合一些视图层面的逻辑，输出对应的 DOM 结构。

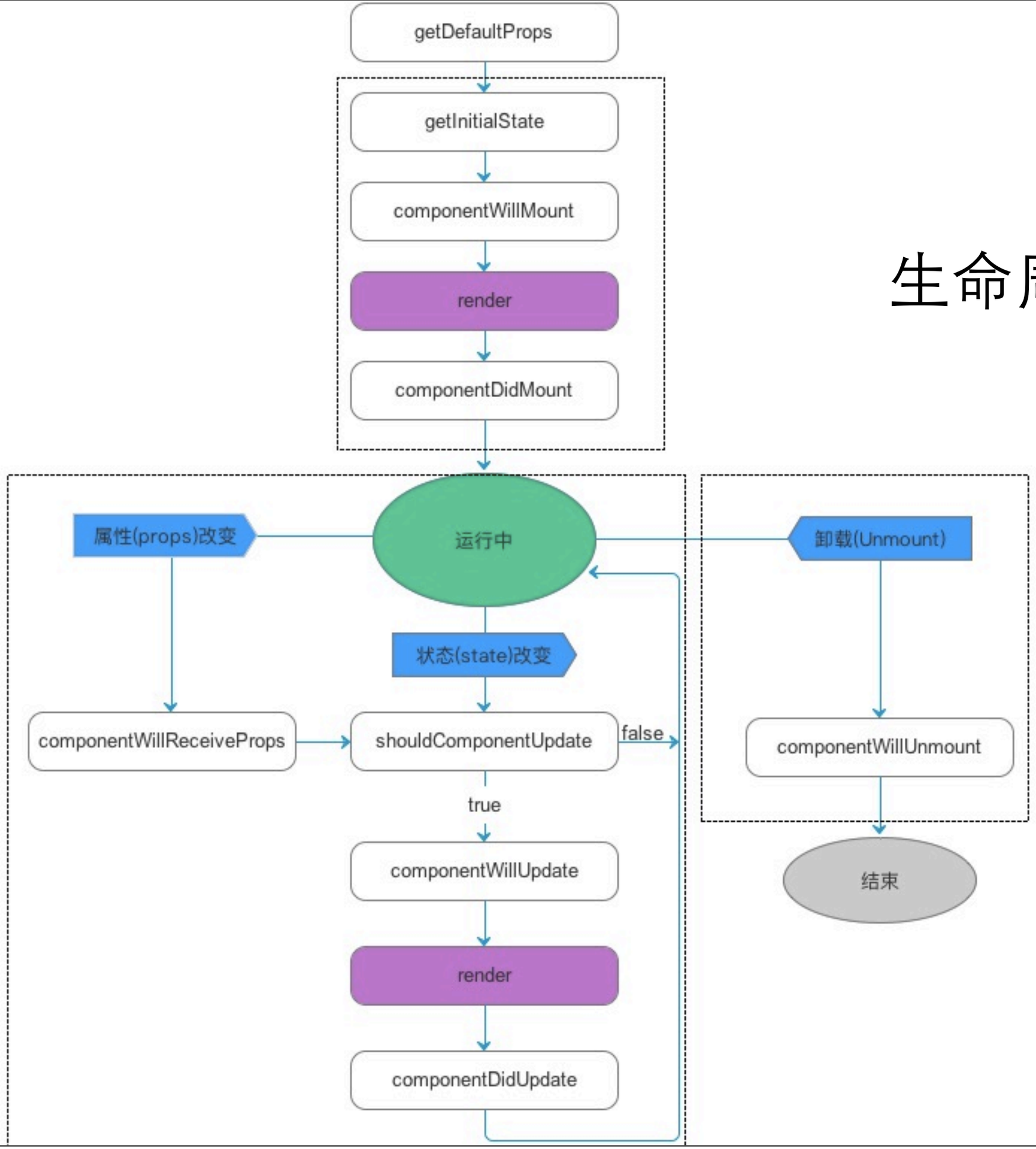
传递prop

State Machines

状态机

15年12月18日 星

生命周期

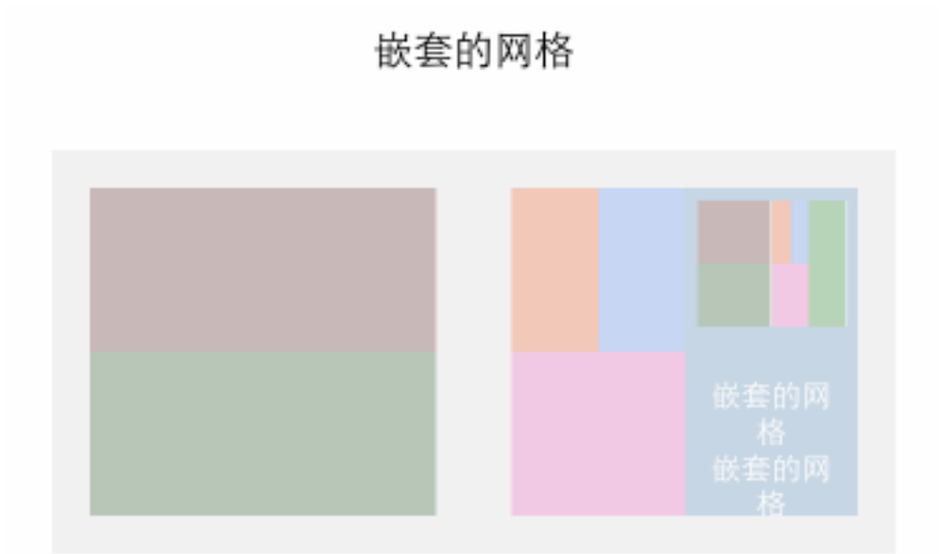

getDefaultProps
getInitialState
componentWillMount
render
componentDidMount
属性(props)改变
运行中
卸载(Unmount)
状态(state)改变
componentWillReceiveProps
shouldComponentUpdate
false
componentWillUnmount
true
componentWillUpdate
render
componentDidUpdate
结束

布局篇

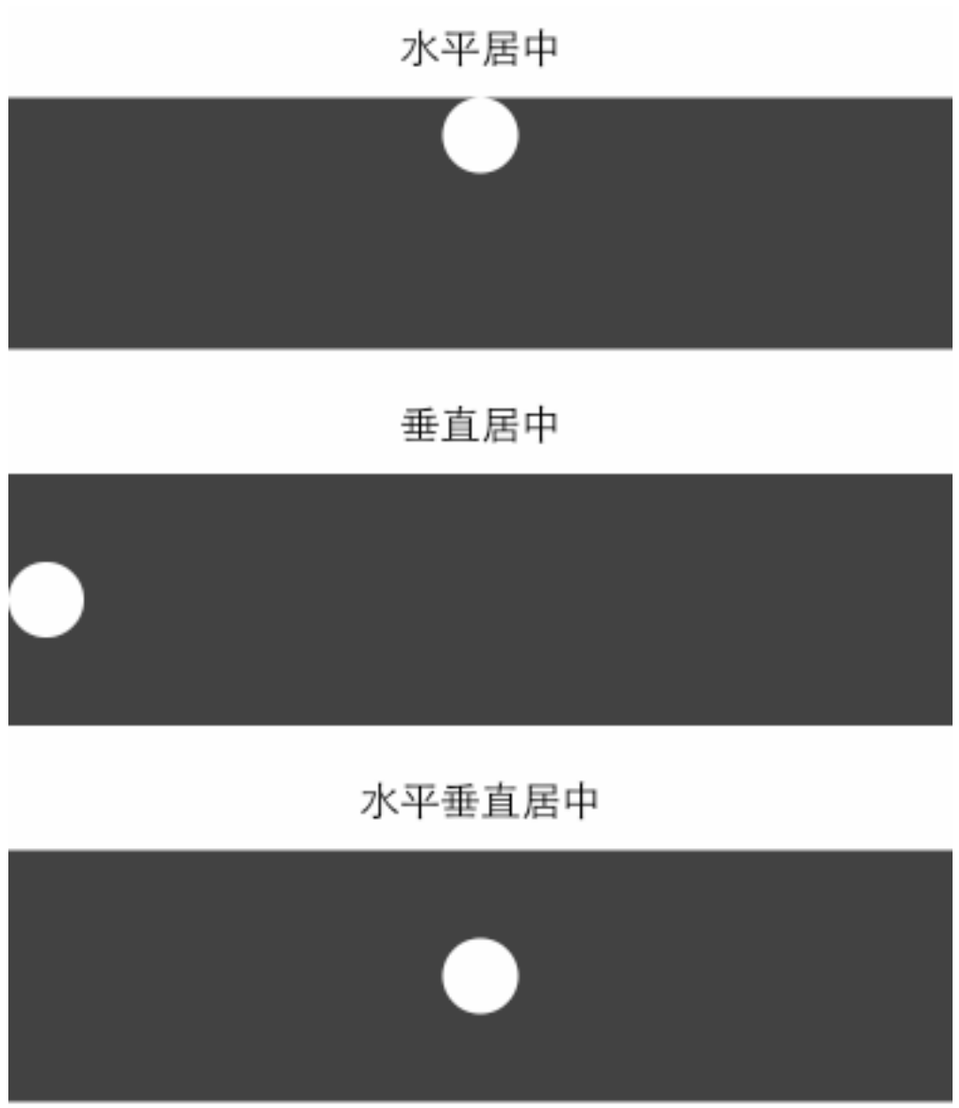
宽度单位和像素密度

flexbox

水平垂直居中



`alignItems` 和 `justifyContent`



android framelayout

JSX

语法进阶

```
function render() {
  return <div className="foo" />;
}
```

becomes

```
"use strict";

var _ref = React.createElement("div", { className: "foo" });

function render() {
  return _ref;
}
```

and

```
var Foo = require('Foo');
function createComponent(text) {
  return function render() {
    return <Foo>{text}</Foo>;
  };
}
```

becomes

```
"use strict";

var Foo = require("Foo");
function createComponent(text) {
  var _ref = React.createElement(Foo, null, text);

  return function render() {
    return _ref;
  };
}
```

写一个Project

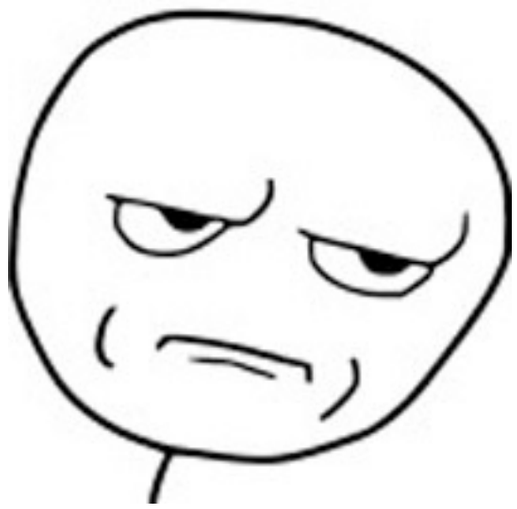
——以android举例

50%的人死在了配置上。。 T_T

这其中，一半死在了翻不了墙，翻得了太慢的卡了两天没动静

native开发死在配react上

react开发死在配native上



百姓网

最小安装子集

necessary

homebrew
nvm→node
watchman
jdk
android sdk
simulator
gradle

choice

flow
android studio
webstorm
atom+nuclide
virtualbox+
genymotion

其他推荐插件

除了 Nuclide 之外，还可以根据你的需求安装其他的一些插件，这里推荐一些插件：

- react：React 的语法补全和智能重排；
- react-snippets：React 的代码段；
- highlight-selected：高亮当前双击选中的标记；
- jshint：检查 JavaScript 的语法，支持 JSX （需要在插件设置中开启 Support Linting JSX）；
- emmet：用 emmet （Zen Coding）方式快速编写页面；
- save-session：让 Atom 记住上一次打开的会话；
- browser-plus：在 Atom 中内嵌一个浏览器窗口，方便页面调试（其实 Atom 本身就是一个浏览器）；
- minimap：如果你对 Sublime Text 的 minimap 念念不忘；
- atomic-emacs：Emacs 键盘布局，适合 Emacs 用户使用；
- vim-mode：Vim 键盘布局，适合 Vim 用户使用。

reference

http://react-native.cn/docs/getting-started.html#content
http://www.bingjie.me/2015/11/25/react-native-android.html

从原生入口

MainActivity→index.android.js

AppDelegate→index.ios.js

4. 与native交互

4.1 原生模块

4.1.1

这里官网上有一个错误。这个错误可能是底层改过导致。

要注意的事项：

1> //.addPackage(new CustomReactPackage())

这件事情也是蛮奇怪的，不知道底层如何实现的。

2> //return Collections.emptyList();

Toast非常有用。它不仅是UI界面与用户交互的提示，对于我们调试代码，获取输出也非常方便。虽然我们可以使用`console.log(msg)`来输出信息，但这样我们还得切换到控制台看，或者在chrome里看，这是非常不方便的。

我们将toast的show函数轻微地封装一下，就非常方便了。

这也从侧面说明了JS的函数都是运行在主线程的，方便渲染。

4.1.2

多线程

如果你在底层封装好了一个Native网络通信Api等，你必须确保自己不运行在 MainThread 里。原生模块不应对自己被调用时所处的线程做任何假设，然后我们用回调函数来实现它们的交互。

回调的使用

还记得android里超级好用的 AsyncTask 么？在iOS里类似于 GCD 。

随时通知JavaScript

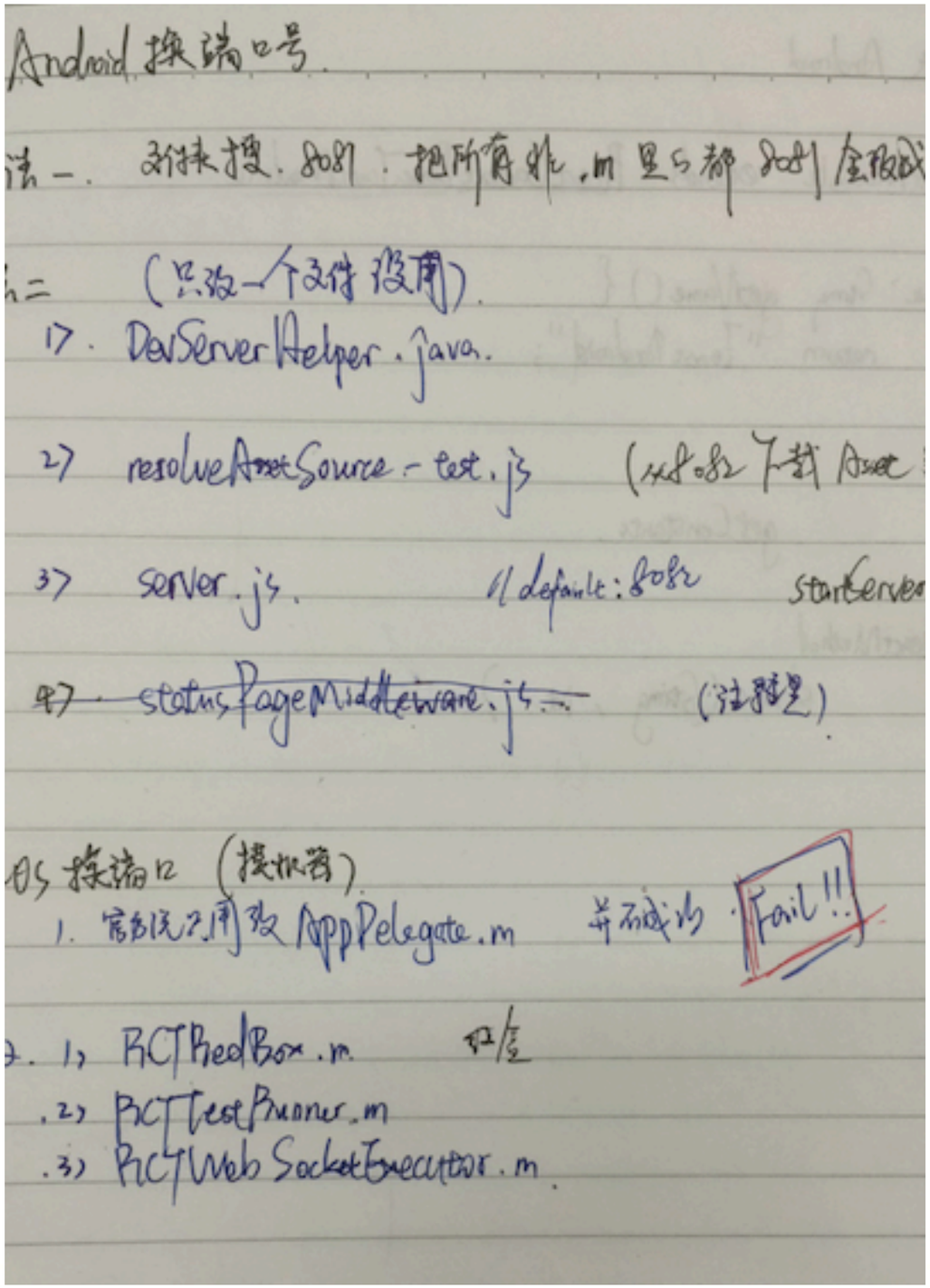
让我们来举个例子，比如一个网络断开事件。

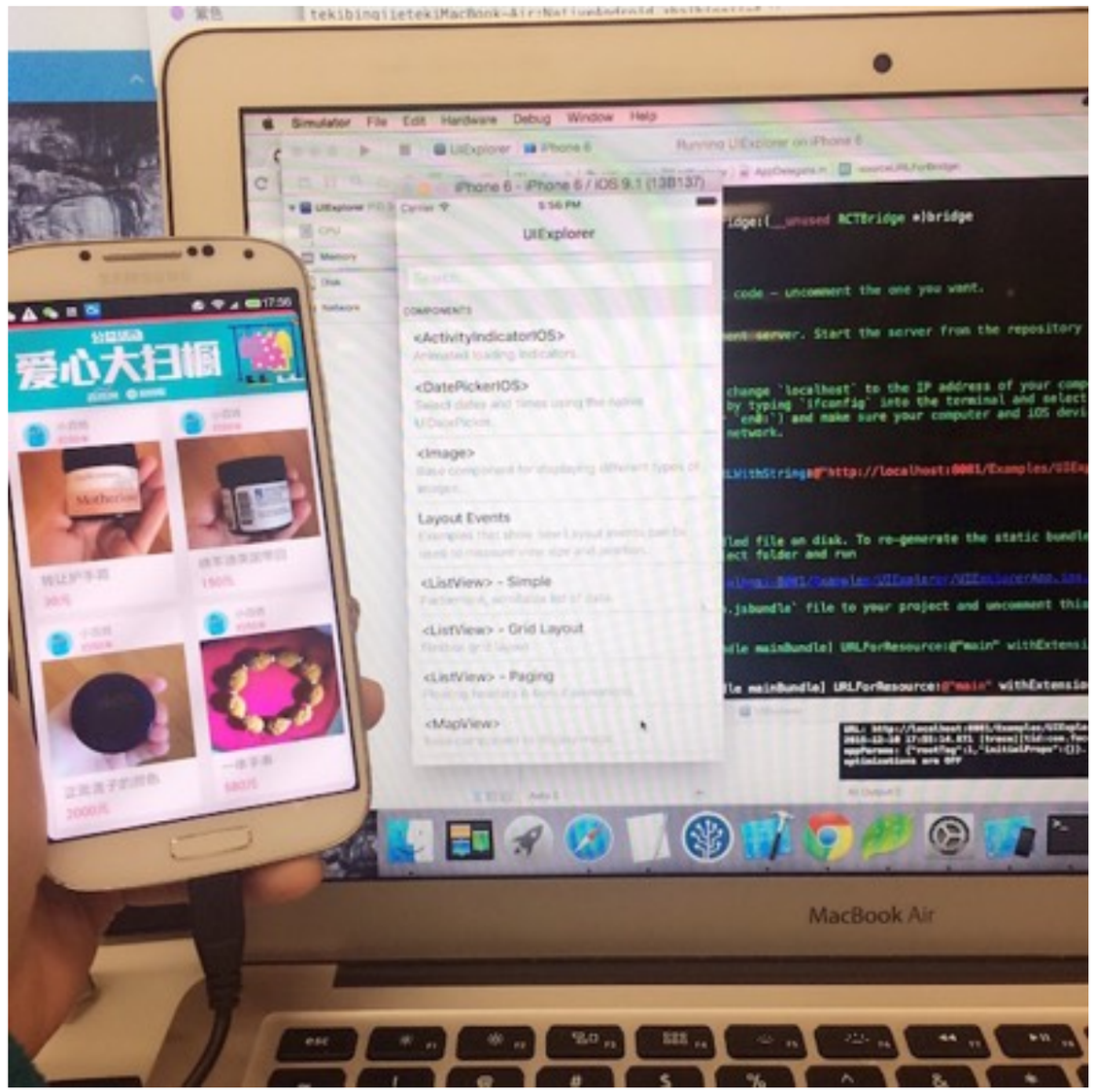
首先在AndroidManifest.xml 取得 "CHANGE_WIFI_STATE" 权限。然后在网络断开时弹出一个通知框，两个选项分别是“切换省流量版”和“继续使用”。 点击切换省流量版

这里面涉及到方案的选取，我们究竟把这个通知框放在 native 中，还是JS 中。

4.2 原生组件

5. 改端口


Android 换端口号.
法一. 全搜 8081 . 把所有的 .m 里与都 8081 全改成
法二 (只改一个文件夹即可).
1. DevServerHelper.java.
2) resolveAssetSource - test.js
3) server.js. // default: 8082 startServer
4) statusPageMiddleware.js (注释掉)
的换端口 (模拟器)
1. 官方说不用改 AppDelegate.m 并成功 Fail!!
2. 1) RCTRedBox.m 红屏
2) RCTTestRunner.m
3) RCTWebSocketExecutor.m


爱心大扫帧
UIExplorer
MacBook Air

Demo

发布

```
public class MainActivity extends Activity {

    private static final String JSBUNDLE_FILE = "ReactNativeDevBundle.js";

    private static void copyFile(InputStream in, OutputStream out) throws IOException {
        byte[] buffer = new byte[1024];
        int read;
        while((read = in.read(buffer)) != -1){
            out.write(buffer, 0, read);
        }
    }
    private void prepareJSBundle() {
        File targetFile = new File(getFilesDir(), JSBUNDLE_FILE);
        if (!targetFile.exists()) {
            try {
                OutputStream out = new FileOutputStream(targetFile);
                InputStream in = getAssets().open(JSBUNDLE_FILE);

                copyFile(in, out);
                out.close();
                in.close();
            } catch (FileNotFoundException e) {
                e.printStackTrace();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        prepareJSBundle();
        ...
    }
}
```

在线更新

第一次打开 React Native 应用的时候，会连接 JS Server 下载一个 `ReactNativeDevBundle.js` 文件，然后放到应用数据的 files 目录下，就能运行这个 JS 文件了

只要通过网络下载并替换这个 JS 文件，就可以实现 APP 的更新。不需要下载 APK 包更新，也不需要市场发布，只要后台上线一个新的 JS，客户端就能立即更新了。这就绕过了应用市场，解决了应用更新困难的问题，修复 BUG 一秒上线，新 Feature 一秒到达用户

在RN中，
JS工程师解决的问题是:
[将基本组件拼装成可用的React组件]
Native工程师解决的问题是:
[提供核心组件，足够的扩展性、灵活性和性能]

评估的方面

Thanks

百姓网致力于提供连接个人和个人生活需求的本地信息平台。

15年12月18日 星

Q&A