Weex 在饿了么的实践经验

ChenYong(题叶)

饿了么, 大前端

很多小伙伴在使用 Weex

页面的更新在 App 当中偏多，

嵌入在 WebView 当中

饿了么前端场景

大量的 WebView

Vue 开发者多于 React 开发者(Element UI)

店铺页面, 活动页面

“移动端业务中大多追求页面的轻量与流畅，

HTML5 的性能瓶颈为人诟病已久，最为明显的问

题是加载性能。”

React Native?

“蜂鸟配送”等 App 有使用, 积累经验

React Native 的列表占用内存过大，没有复用机制

React Native 不同版本之间的 breaking change 太多, 后期维护成本过高，并不适用于饿了么这个体量的 App

Better ListView - FlatList

Browse files

Summary:
We really need a better list view - so here it is!

Main changes from existing `ListView`:

* Items are "virtualized" to limit memory - that is, items outside of the render window are unmounted and their memory is reclaimed. This means that instance state is not preserved when items scroll out of the render window.
* No `DataSource` - just a simple `data` prop of shape `Array<any>`. By default, they are expected to be of the shape `{key: string}` but a custom `rowExtractor` function can be provided for different shapes, e.g. graphql data where you want to map `id` to `key`. Note the underlying `VirtualizedList` is much more flexible.
* Fancy `scrollTo` functionality: `scrollToEnd`, `scrollToIndex`, and `scrollToItem` in addition to the normal `scrollToOffset`.
* Built-in pull to refresh support - set set the `onRefresh` and `refreshing` props.
* Rendering additional rows is usually done with low priority, after any interactions/animations complete, unless we're about to run out of rendered content. This should help apps feel more responsive.
* Component props replace render functions, e.g. `ItemComponent: ReactClass<{item: Item, index: number}>` replaces `renderRow: (...) => React.Element<*>`
* Supports dynamic items automatically by using `onLayout`, or `getItemLayout` can be provided for a perf boost and smoother `scrollToIndex` and scroll bar behavior.
* Visibility callback replaced with more powerful viewability callback and works in vertical and horizontal mode on at least Android and iOS, but probably other platforms as well. Extra power comes from the `viewablePercentThreshold` that lets the client decide when an item should be considered viewable.

Demo:

https://www.facebook.com/groups/576288835853049/permalink/753923058089625/

Reviewed By: yungsters

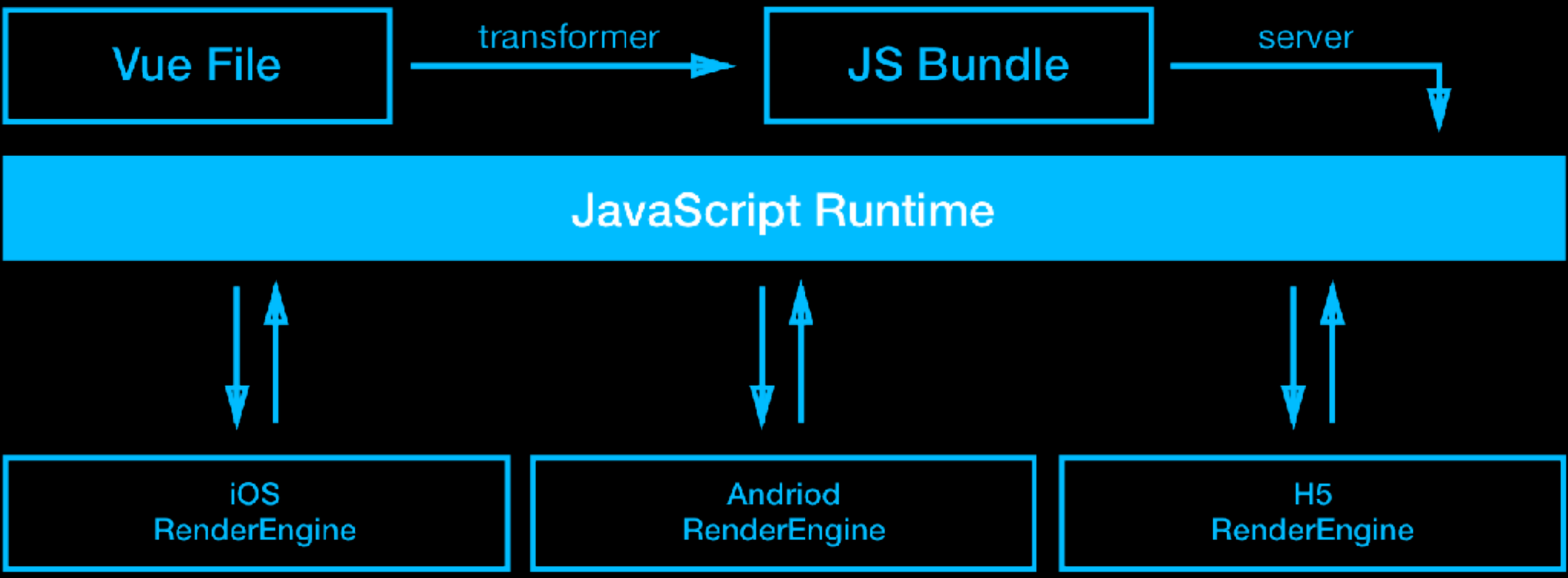
Differential Revision: D4412469

fbshipit-source-id: e2d891490bf76fe14df49294ecddf78a58adcf23

master (#1) v0.44.0-rc.0 ··· latest

sahrens committed with facebook-github-bot on Feb 5 1 parent 57daad9 commit a3457486e39dc752799b1103ebe606224a8e8d32

Weex


Vue File
transformer
JS Bundle
server
JavaScript Runtime
iOS
RenderEngine
Andriod
RenderEngine
H5
RenderEngine

6 lines (5 sloc) | 96 Bytes

```
1  <template>
2    <div>
3      <text style="font-size: 100px;">Hello World.</text>
4    </div>
5  </template>
```

```
11
12  var text = document.createElement('text', {
13    attr: { value: 'Hello World' },
14    classStyle: { fontSize: 48 }
15  })
16
17  body.appendChild(image)
18  body.appendChild(text)
19  document.documentElement.appendChild(body)
20
```

Framework

Vue(Weex, default)

Rax(React)

Legacy(旧版本 .we)

Vanilla(纯 js 写法)

通过 Webpack loader 完成打包,

早期只有 .we 目前作为 Legacy 模式

一套构建高性能、可扩展的

原生应用跨平台开发方案

轻量: 体积小、语法简单、易于掌握

可扩展: 可以横向扩展和定制原生组件和功能

高性能: 对加载时间和资源占用深度优化，给你更好的体验

开发 Weex 页面

Weex 试验页面

交互少

访问量大

基于 Vue

具体工作

业务的实现：投入了一个人，基于 Vue 版本，3 天左右

Weex 的报错和性能监控：前端和 App 方配合，大概两周

Weex 的降级策略：这个是我们和 App 方讨论后得出降级方案，主要由 App 方实现

从立项到上线大概花了三周的时间

发现页的 Weex 版本性能非常优异(12 月底上线)

计划逐步接入更多页面


先先贝

Weex 开发体验

基于 Web 技术, 学习门槛低

仅有 Flexbox 布局, text 无法嵌套, 难以实现长文当中样式的混合

No Cookies. 只能用 storage 来完成对应信息的存储

三端一致性不完全, 特别是 HTML5 支持不完善

CSS 和 Web 当中存在差异

DevTools 的成熟度不够. 热替换不够强大

相对 Web 而言组件丰富度不够

Object.free 对类库带来一些限制, 以及 ES5 引擎

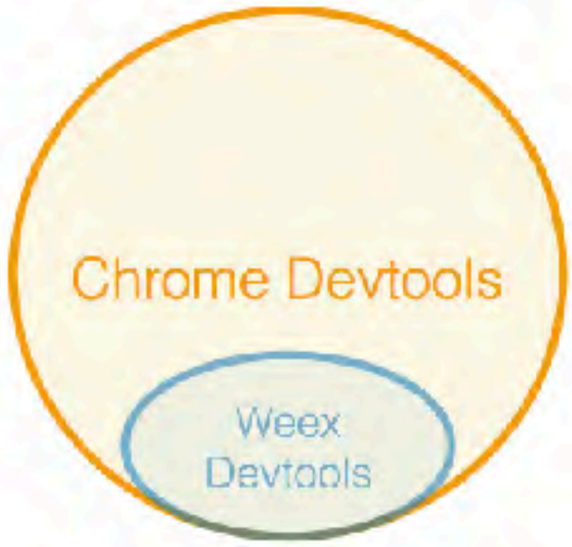
...

与 HTML 的标签有许多不同
与 Native XML 中有许多相似的标签

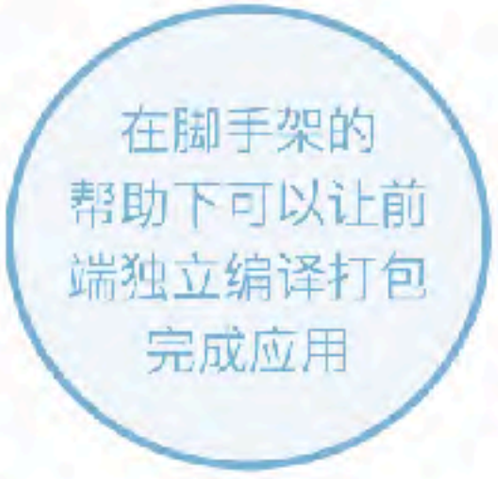
只支持 Flex 布局

目前只支持比较少的 CSS

默认只支持 Storage
相当于 HTML5 中的 LocalStorage



相当于 Chrome Devtools
但提供的功能有限



独立开发的成本较低

Weex 页面性能

启动步骤

初始化 Weex Runtime 和 Framework 代码

js 引擎初始化页面实例

由 Weex SDK 渲染页面

```
{
  "JSLibInitTime": "157",
  "JSTemplateSize": "65",
  "SDKInitInvokeTime": "7",
  "SDKInitTime": "231",
  "communicateTime": "146",
  "networkTime": "0",
  "screenRenderTime": "184",
  "totalTime": "184",
  "fromCache": true
}
```

SDKInitInvokeTime 也是 SDK 初始化方法调用时间， 初始化时是异步进行的， 所以 SDKInitTime 会比较大

SDKInitTime 是 SDK 启动初始化时间, 应该是一次性的开销， 不是每个 Weex instance 的开销, 第一次启动执行

communicateTime 是 Weex 实例的创建时间， 现在的用法每次新建页面都会创建新的实例

screenRenderTime 是首屏渲染时间, totalTime 是完整渲染时间

networkTime 网络消耗时间(自己实现)

fromCache 代码是否来自缓存(自己实现)

数据	communicateTime	networkTime	totalTime	communicateTime + totalTime
平均值	227.38	546.59	286.84	513.29
中位数	210	372	254	468
up90 平均值	201.42	325.19	246.35	450.53
up90 中位数	197	323	241	441
up90 数值	348	973	435	781

Android: 网络330~ms(?), 渲染450~ms

数据	communicateTime	networkTime	totalTime	communicateTime + totalTime
平均值	74.31	355.08	110.07	184.35
中位数	64	127	86	148
up90 平均值	63.69	190.16	91.65	157.04
up90 中位数	59	73	80	139
up90 数值	134	714	190	320

iOS: 网络190~ms, 渲染160~ms

降级方案

App 是否开启 Weex SDK：大部分 App 都有在线的配置管理，这个是 App 本身的配置管理功能做的开关。

一个是某个页面的 Weex 版本是否开启，当 Weex 不开启的时候的降级 Web 页面。

为了保证不影响用户的使用，我们采用的灰度策略如下：

在支持 Weex 的 App 灰度发版之后，我们在业务低峰期的时候开启了这个开关。

通过我们接入的监控平台，我们可以及时对 Weex 页面的错误进行监控，如果出现大规模报错，则会立即把支持 Weex 的开关关闭。

开关配置文件 sitemap

```
{
  "url": "https://www.example.com/discover/index.html",
  "weex-url": "https://www.example.com/discover/weex.js",
  "weex-enabled": true,
  "weex-minimal-version": "0.8.0"
}
```

url：Webview 使用的 Web 页面的地址，当 weex-enabled 为 false 的时候，会使用这个地址打开一个 WebView

weex-url: Weex 资源所在的位置，正常情况下使用此 URL 下载 Weex 使用的 JavaScript 文件

weex-enabled: boolean 类型，默认为 false，当此属性为 true 的时候，才会使用 weex-url 加载一个 WeexView

weex-minimal-version: 字符串类型，代表了加载 weex-url 需要使用的 Weex 的最低版本，此属性必填，如果不填则 weex-enabled 不生效。当 App 中的 Weex SDK 版本比这个版本低的时候，则会 fallback 到 webview 的形式

“Weex 的版本兼容性优异，我们从 0.8.0 升级到 0.10.0，还没有出现需要降级的情况"

通过升级 Webpack loader 升级来使用新的语法

- // {“framwork”: “Vue”}

  兼容 .we 写法的文件

总结

技术	学习成本	弱交互性能	强交互实现	版本间兼容性
HTML5	低	一般	能实现, 性能差	好(大量 Polyfill 可用)
React Native	中	好	能实现, 性能好	差(Breaking changes 多)
Weex	低	好	一般, 部分拖动相关效果无法实现	好

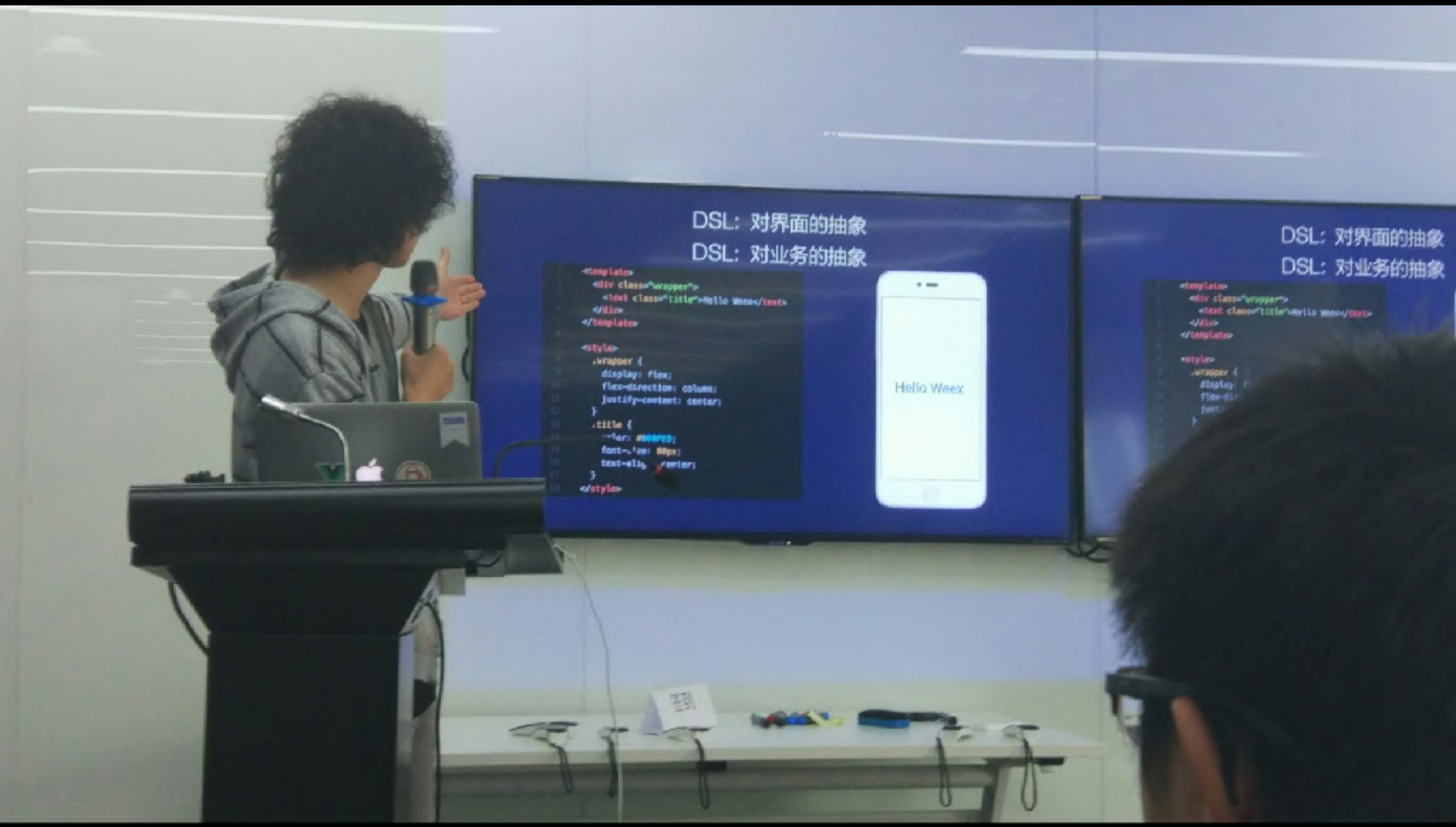
使用 React Native 需要熟悉 React 全家桶，这个学习成本比 Vue 高不少

关于 React Native 的 Breaking Changes，可以通过 Google 搜索 "React Native 放弃”，可以搜到大量的文章

Github 上有一个用 React Native 高仿 Eleme App 的实现，大部分效果都能实现；基于我们对 Weex 的理解，Weex 实现拖动部分交互非常困难，甚至目前的版本不可能实现

“从我们的实践上看，Weex 运行稳定，性能优异，在做好监控降级机制之后，可以放心的投入到生产”

社区交流


DSL：对界面的抽象
DSL：对业务的抽象
Hello Weex
DSL：对界面的抽象
DSL：对业务的抽象


WEEX Conf
WEEX

知乎专栏: Weex 入坑指南

快速开始 Weex 之旅

手把手编译 Playground

Native App 的运行与构建

Debug 调试是一门手艺活

@饿了么前端