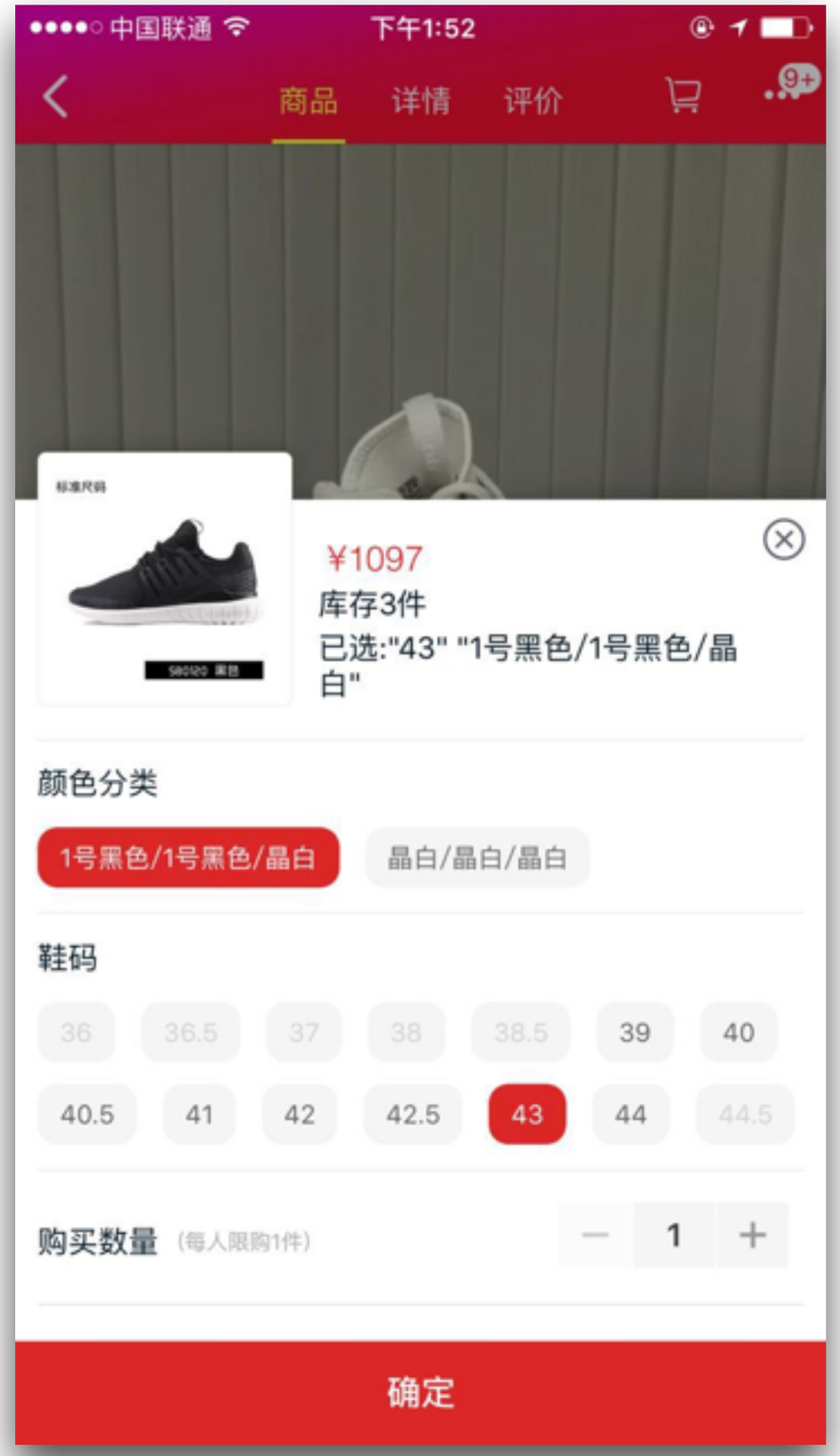
一个页面的前端变迁之路

架构篇

—— 梁冬



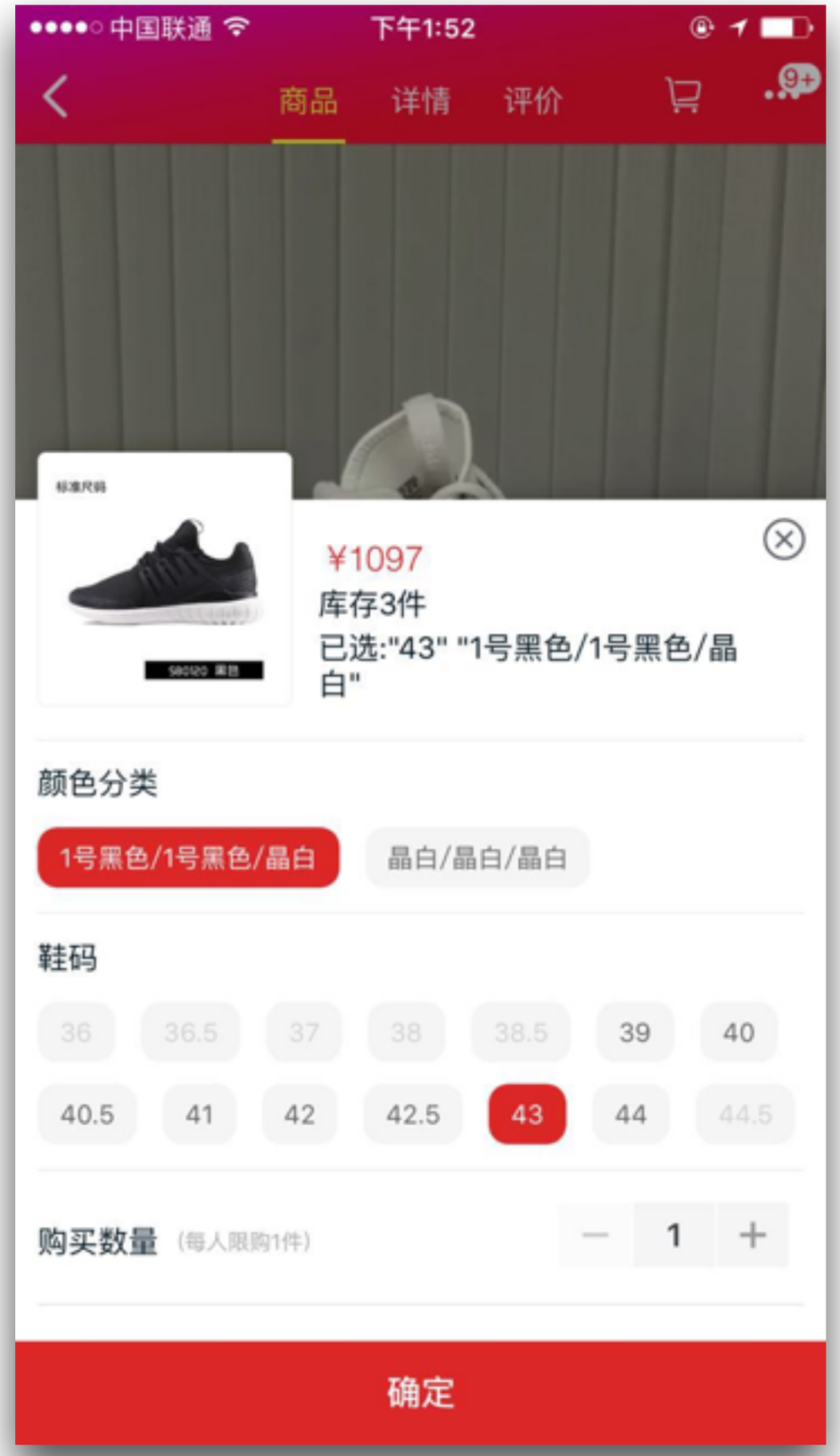
一个商品sku选择页面

1、可以选择颜色和鞋码
2、不同sku的组合会触发信息更改
3、购买数量受若干因素影响

pc, static


淘宝网 Taobao.com
时尚流行服饰
魅力场
1折起
热销精选
五一旅游——看不够的风景！
淘宝集市
品牌商城
二手闲置
全球购
热销商品


YAHOO!
What's New
Check Email
Personalize
Help
Yahoo! Mail
free email for life
Yahoo! Messenger
Yahoo! Auctions
coins, cards, stamps
Search
advanced search
Shopping - Auctions - Yellow Pages - People Search - Maps - Travel - Classifieds - Personals - Games - Chat - Clubs
Mail - Calendar - Messenger - Companion - My Yahoo! - News - Sports - Weather - TV - Stock Quotes - more...
Yahoo! Shopping - Thousands of stores. Millions of products.
Departments
Apparel
Bath/Beauty
Computers
Electronics
Flowers
Food/Drink
Music
Video/DVD
Stores
Sports Authority
Gap
Eddie Bauer
Macy's
Products
Digital cameras
Pokemon
MP3 players
DVD players
Arts & Humanities
Literature, Photography...
News & Media
Full Coverage, Newspapers, TV...
Business & Economy
Companies, Finance, Jobs...
Recreation & Sports
Sports, Travel, Autos, Outdoors...
Computers & Internet
Internet, WWW, Software, Games...
Reference
Libraries, Dictionaries, Quotations...
Education
College and University, K-12...
Regional
Countries, Regions, US States...
Entertainment
Cool Links, Movies, Humor, Music...
Science
Animals, Astronomy, Engineering...
In the News
Scores killed in Nigerian riots
Austria's Haider resigns as party leader
Floods trap thousands in Mozambique
more...
Marketplace
Yl Auctions - Peanuts, Pokemon, computers
Free 56K Internet Access
Yahoo! Bill Pay - free 3-month trial
more...
Inside Yahoo!
Yahoo! GeoCities - build your free home page
Play free Fantasy Soccer
Yahoo! Clubs - create your

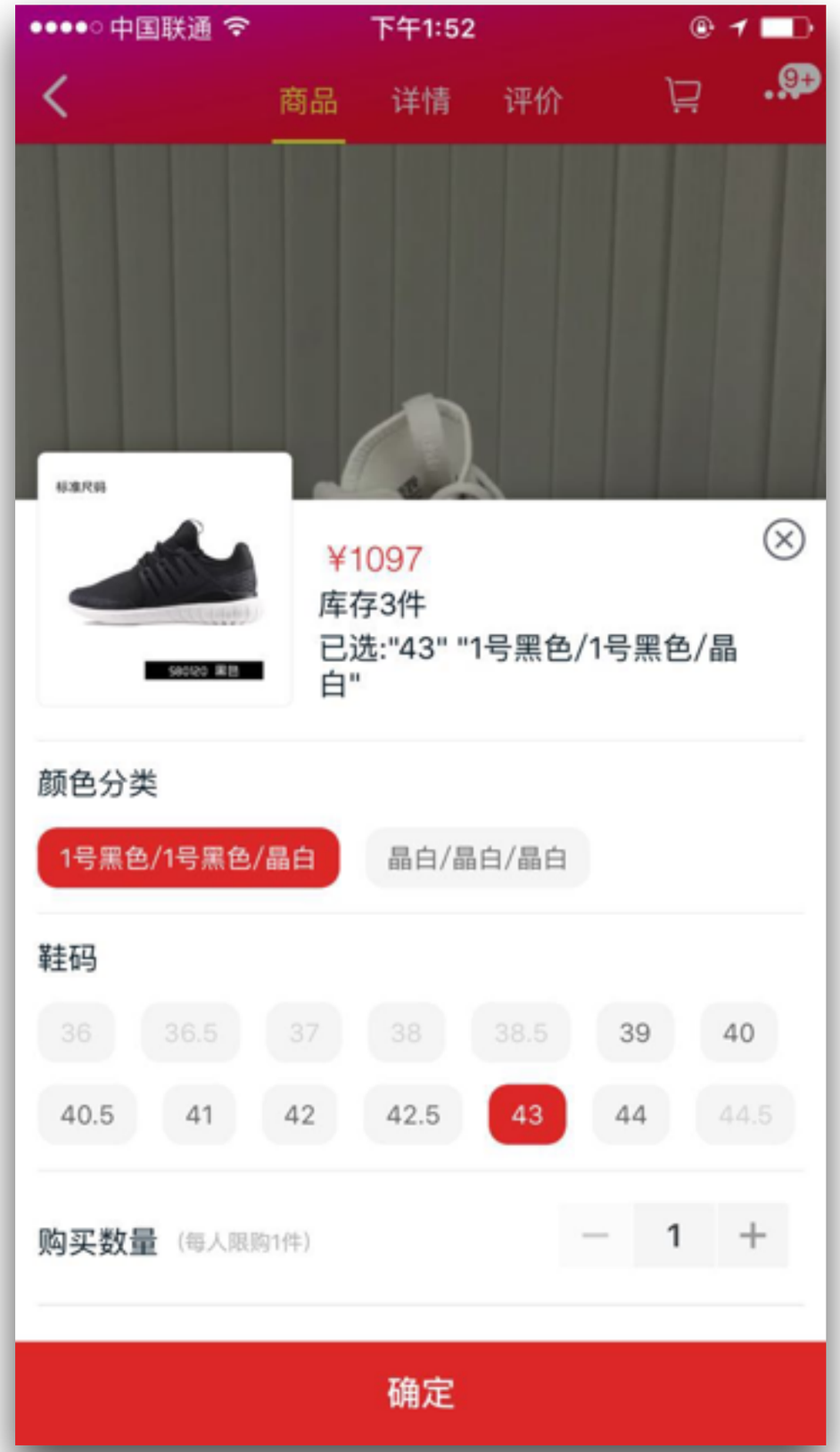



```
<style>
  …
</style>

<body>
  …
  <a href=“url”/>
  …
</body>

<script>
  …
</script>
```

- 数据单一
- 交互单一
- 并没有架构可言

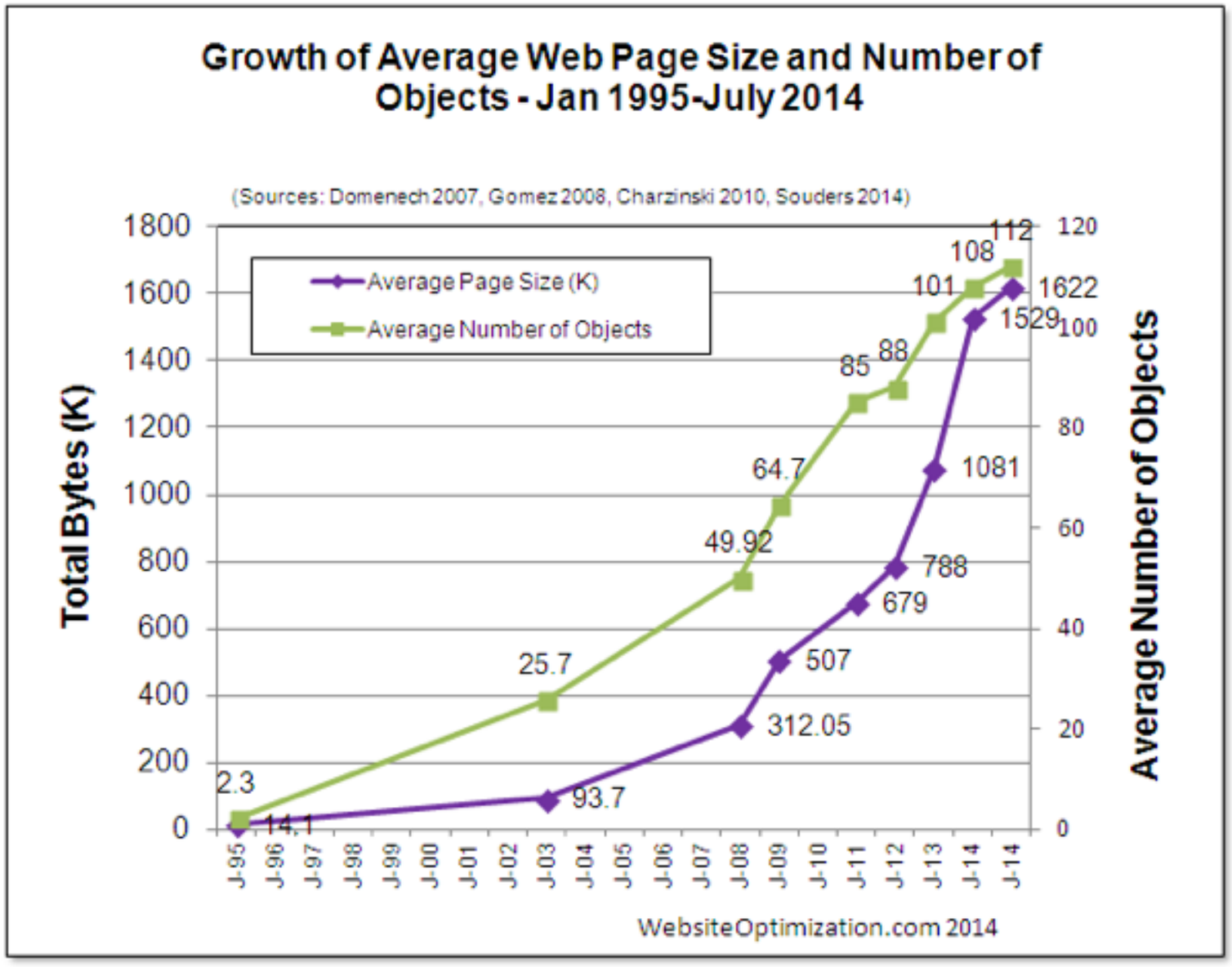
- 搜索引擎友好
- 性能友好
- 前端模块化的始祖

dynamic,
asynchronous




```
<style>
  thousands of styles…
</style>

<body>
  <%
    out.println(“%f”, shoe.getPrice());
  %>
</body>

<script>
  new XMLHttpRequest();
  …
  thousands of code
</script>
```

- 动态数据获取
- 异步按需加载
- 用户友好



- 页面臃肿
- 逻辑复杂
- 加载延时

Best Practices for Speeding Up Your Web Site

- Make JavaScript and CSS External
- Remove Duplicate Scripts
- Minify JavaScript and CSS
- Make Ajax Cacheable
- …

lib, modular


jQuery
write less, do more.
Ext JS
Knockout.
babylon JS
Marionette
ember
Fabric.js
Interactive Framework
MOOTOOLS
handlebars
ROCA
Leaflet
ember
Scala.js
REQUIRE·JS
HTML
GreenSock
Engaging the Internet
Gulp
RIOT
IMMUTABLE
TriNet
Ambitions realized
MIDNIGHT.JS
browserify
SVG.JS
SAMSUNG

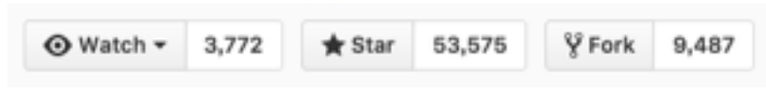

zepto.JS
ANGULARJS
JS
CLJS
JS
NORTHWESTERN UNIVERSITY
knight lab
VIDEOJS
basket.js
express
UNDERSCORE.JS
bacon.js
PixiJS v4
famo.us
Hi there
is.js
Dimas López
chai
MAD LIBS
tc
dojo
jQuery
MOOTOOLS 1.3
BACKBONE.JS
node

选择一个前端库的维度

Scale

Functionality

Community



Stats

77,360 downloads in the last day

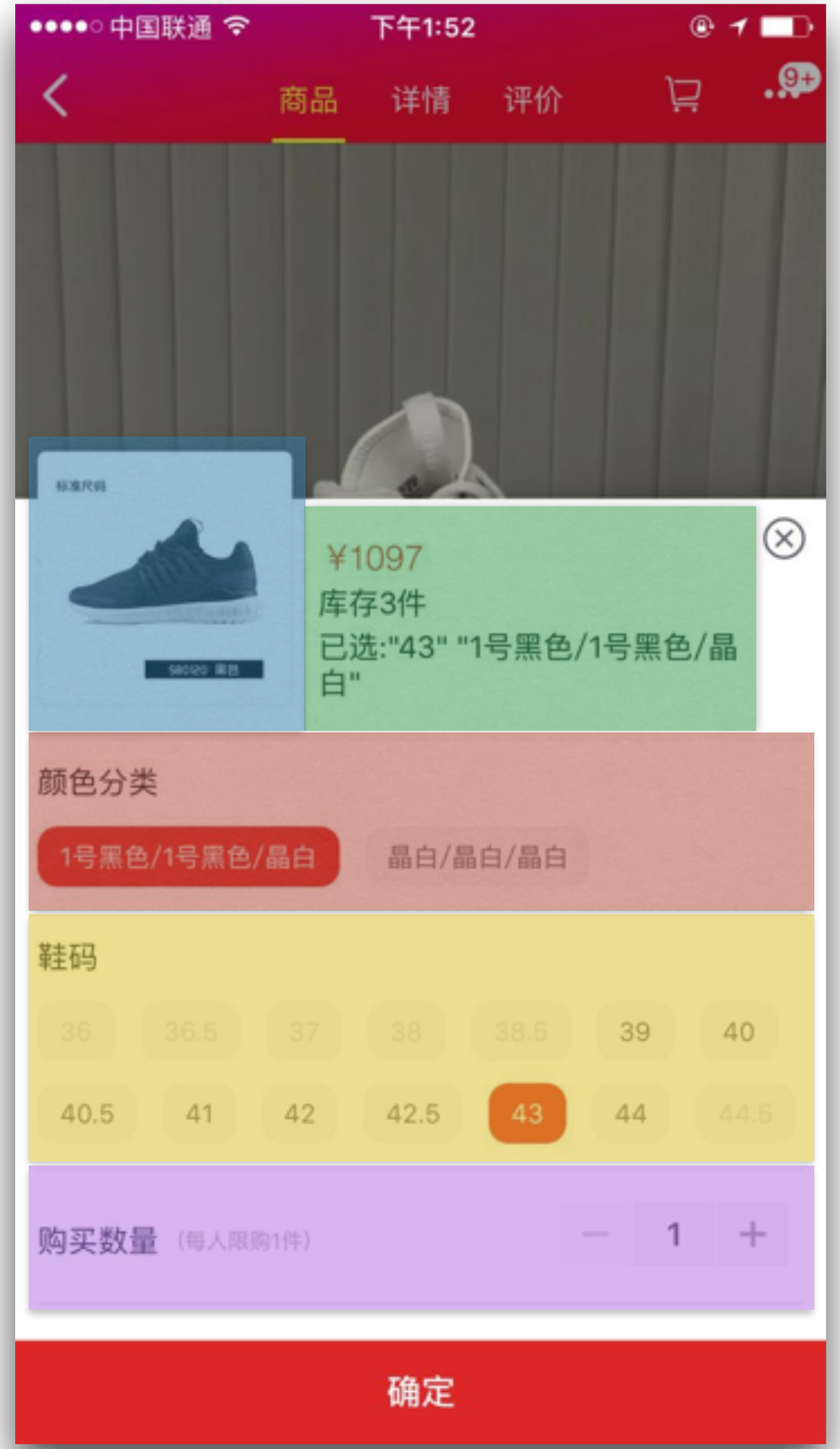
428,972 downloads in the last week

1,892,360 downloads in the last month

346 open issues on GitHub

33 open pull requests on GitHub

前端库的蓬勃发展为前端架构的定型创造了可能。
所有的架构几乎都离不开模块的组织。
所有的模块化几乎都离不开合理的模块管理和加载机制。




```
// we got moduleA.js/css/html
// we got moduleB.js/css/html
// we got moduleC.js/css/html

<link href="moduleA/B/C.css"/>
<body>
  partial moduleA/B/C.html
</body>
<script src="moduleA/B/C.js"/>
<script>
  var Page = {
    init: function() {
      use moduleA/B/C todo…
      // write other code here
    };
  };
  Page.init();
</script>
```

如何管理和加载前端的js模块?

CMD规范

```
define(function(require, exports, module) {
    var a = require('./a')
    a.doSomething()
    // 此处略去 100 行
    var b = require('./b') // 依赖可以就近书写
    b.doSomething()
    // ...
})
```

Sea.js

AMD规范

```
define(['./a', './b'], function(a, b) { // 依赖一开始就写好
    a.doSomething()
    // 此处略去 100 行
    b.doSomething()
    ...
})
```

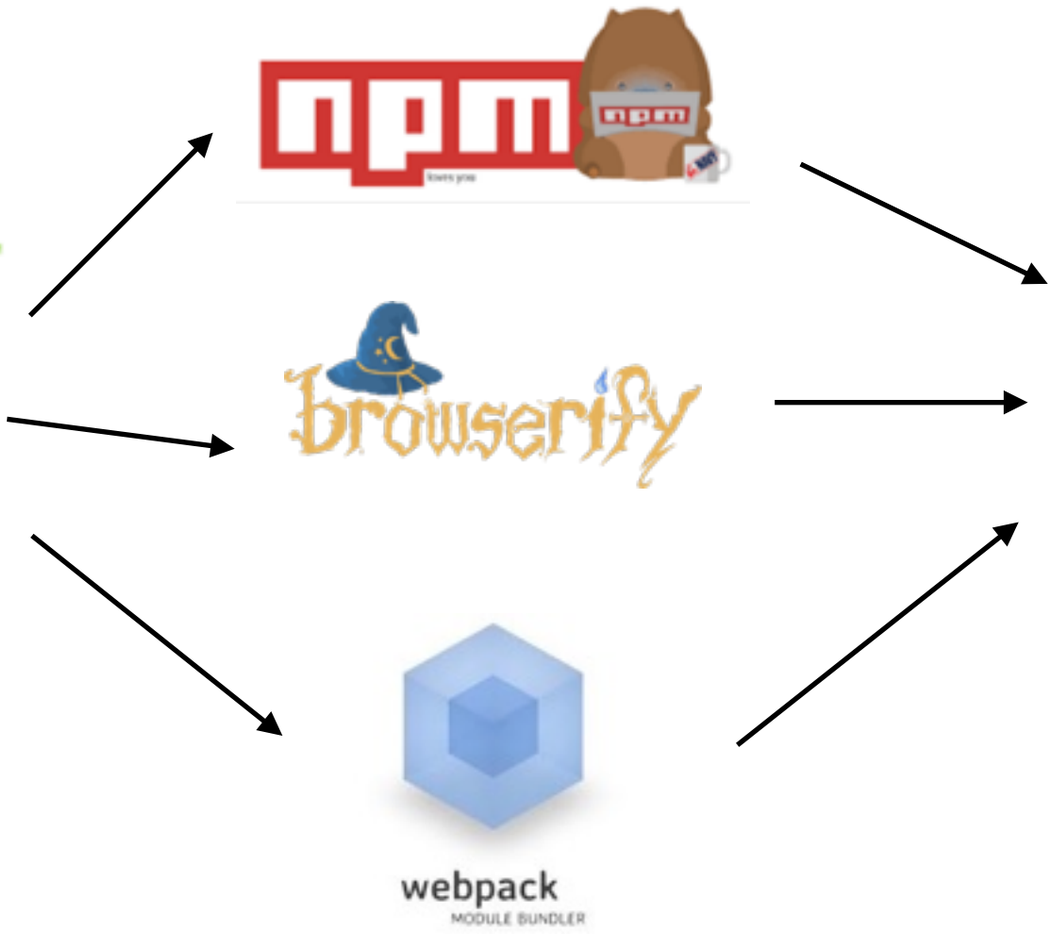
REQUIRE.JS

KISSY

如何管理和加载前端的js模块？（最流行的玩法）

Commonjs规范

```
var a = require('./a.js');
module.exports = {
    foo: function() {
        a.doSomething();
        …
    }
};
```



- 灵活的库选型
- 按需选择模块
- 资源共享
- 自动合并

自从使用了库，进行前端模块化开发之后……

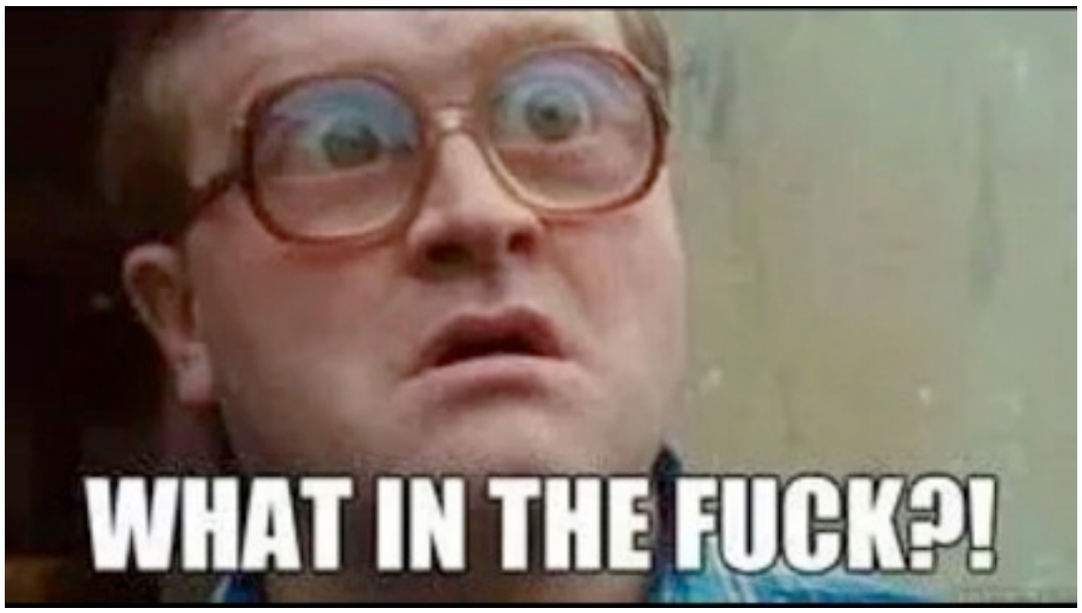
- 结构清晰，可维护
- 专注于业务逻辑
- 可复用

- 我们并没有关注每个模块内部的逻辑复杂性
- 我们并没有关注模块间的依赖程度
- 所谓可维护，只是个相对概念……

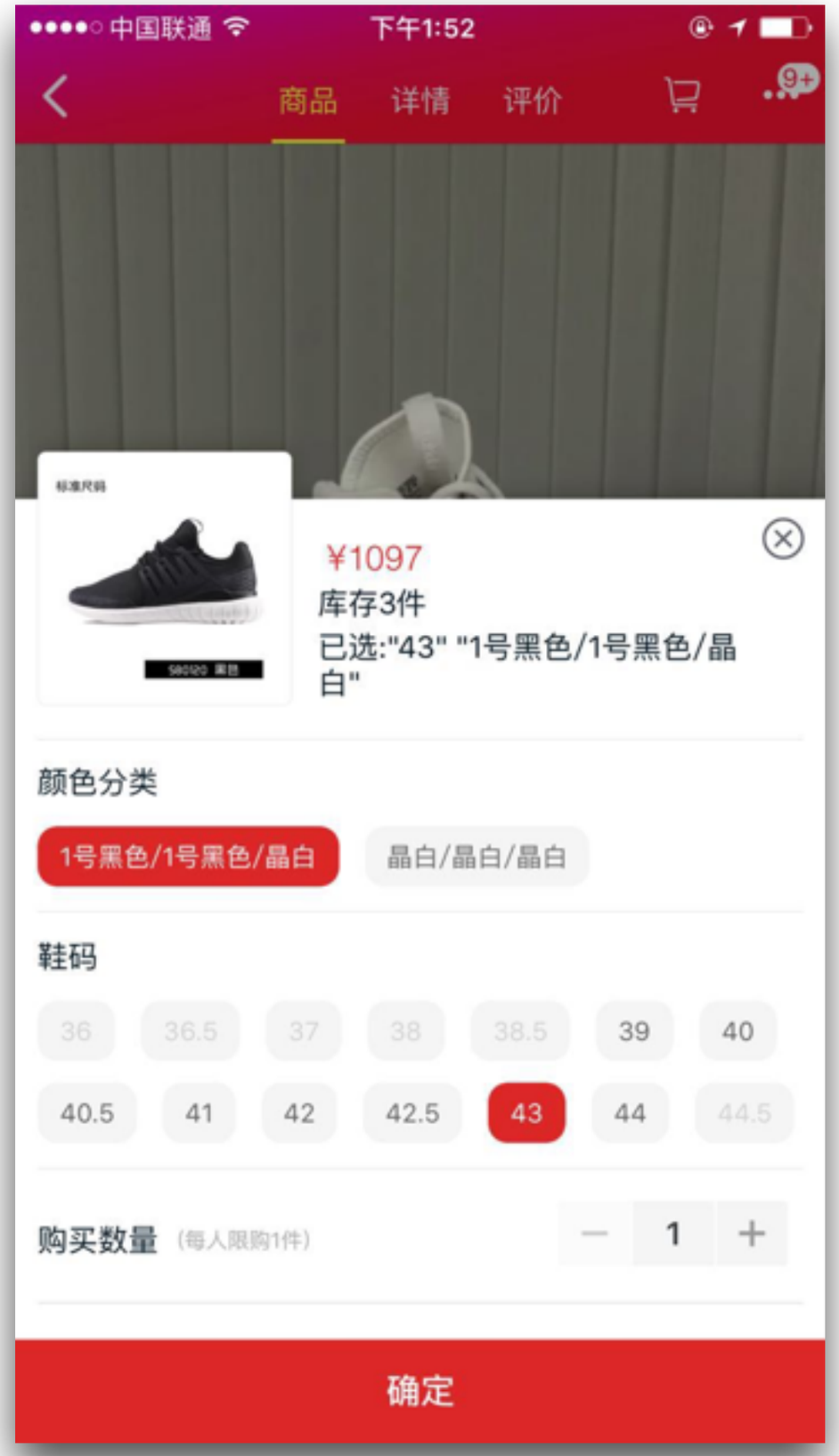
MV*,

effective

前端MVC，MVVM，MDV.....



其实我们只关心：数据和视图解耦、模块间解耦、
降低代码复杂度、代码的可维护性

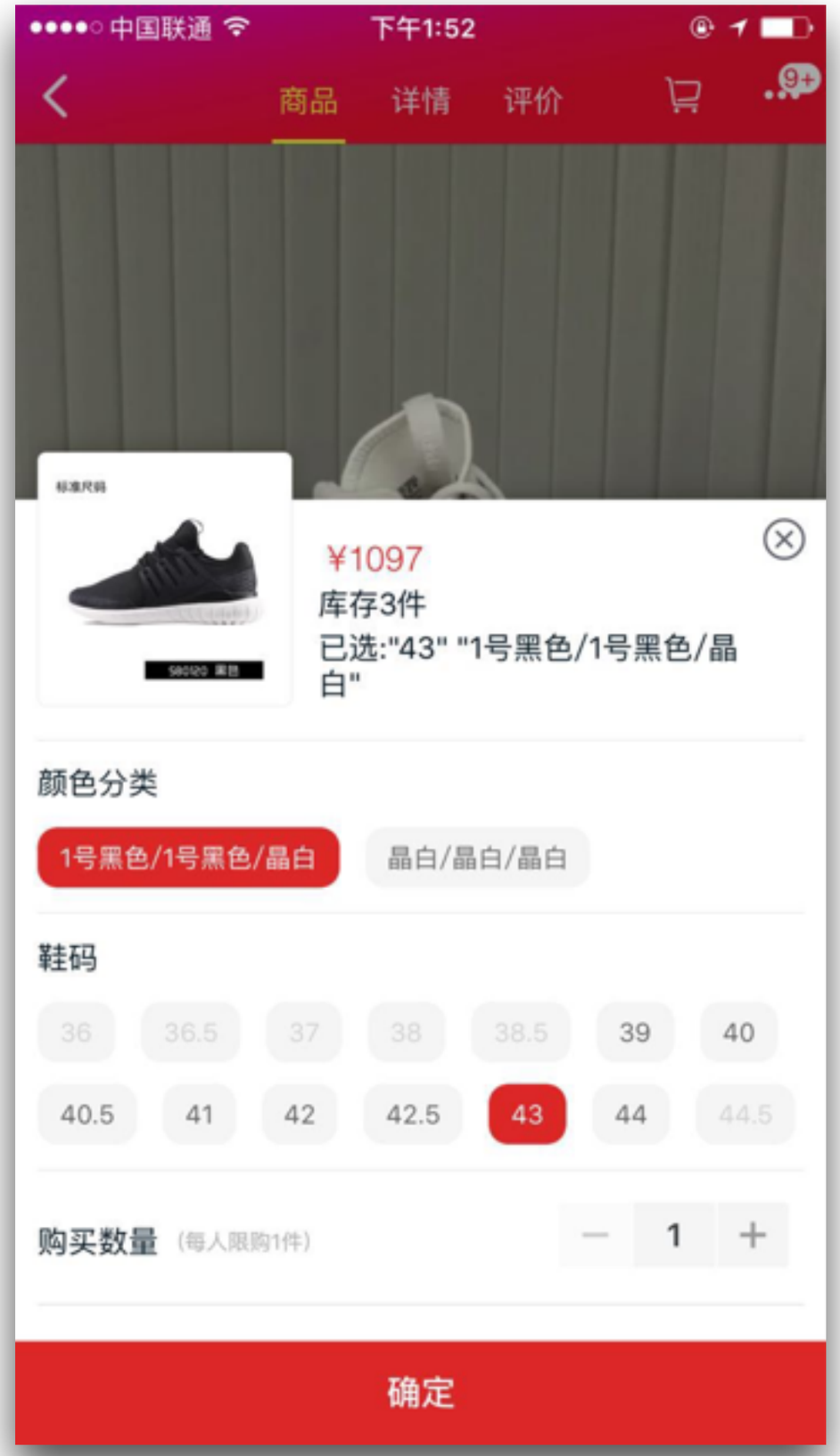



```
<div id="picker">
  <span id="price"></span>
  <btn id="btn1" data-clr="1"/>
  <btn id="btn2" data-clr="2"/>
</div>

// 千辛万苦找来的js文件
$('#btn1').click(fn() {
  $.ajax({
    success: fn(data) {
      $('#price').html(data.price);
    }
  })
});
```

如果页面里有10000个div呢?

如有有10000个btn都绑定了事件呢?

如果有10000个地方需要更新price呢?




```
<picker ng-controller=“Ctrl”>
  <span>{{price | filter}}</span>
  <btn ng-click=“getPrice(1)”/>
  <btn ng-click=“getPrice(2)”/>
</picker>

// js
app.controll(‘Ctrl’, fn($scope) {
  $scope.getPrice = fn() {
    $.ajax({
      success: fn(data) {
        $scope.price = data.price
      }
    })
  }
})
```

- 语义性更强
- 数据与视图分离
- 双向绑定
- 依赖注入

适用于数据驱动型的应用，
或后台管理系统居多。

专注于双向绑定，更轻量，
MVVM架构下的移动端首选。

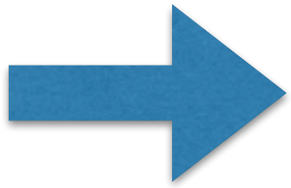
web components

web components

- 封装
- 隔绝

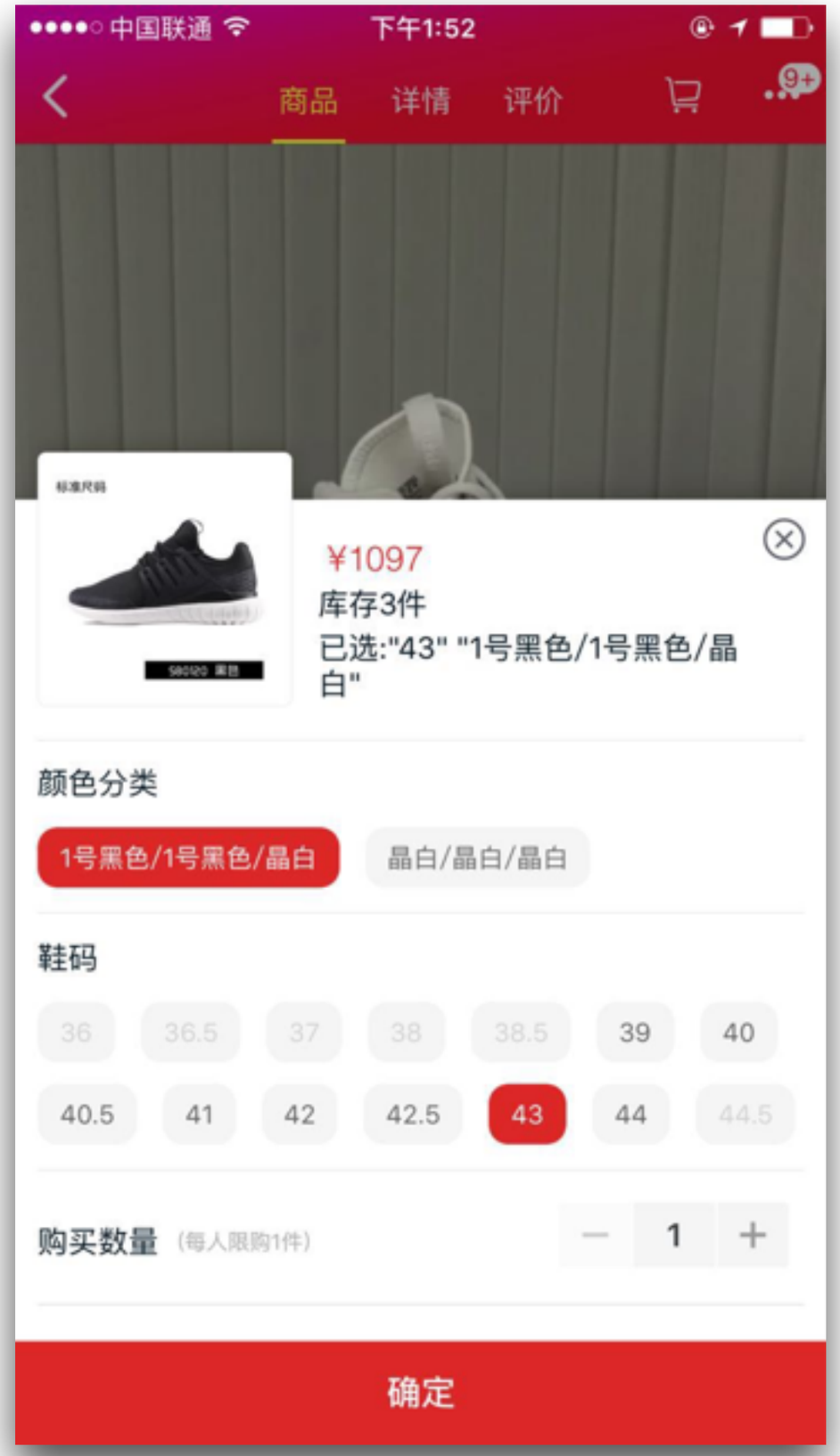
virtual dom

- 高效的re-render机制



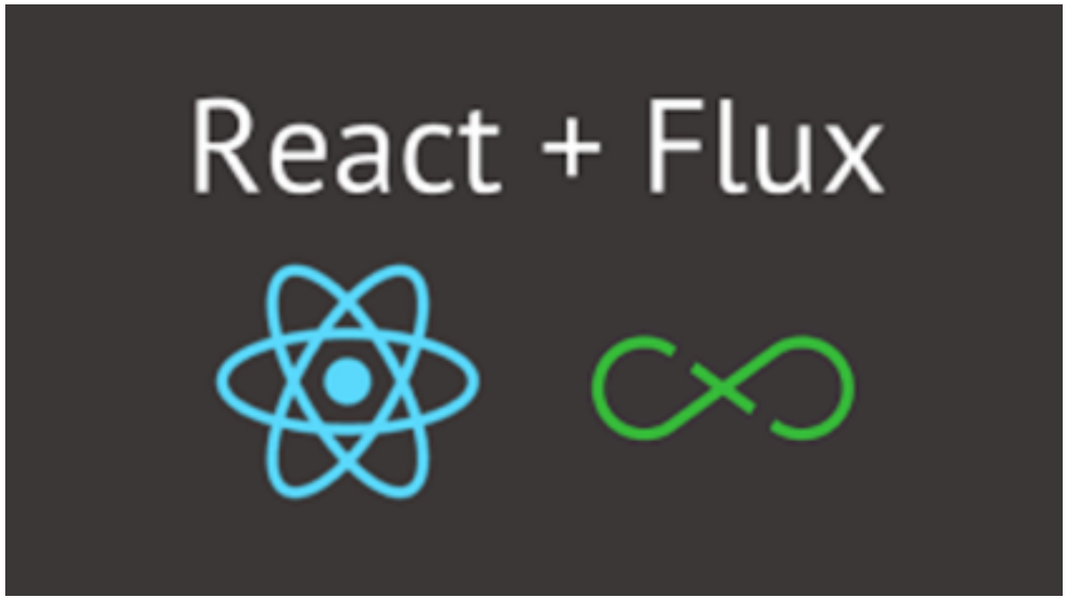
hybrid

- 细粒度的模块
- 移动端开发友好

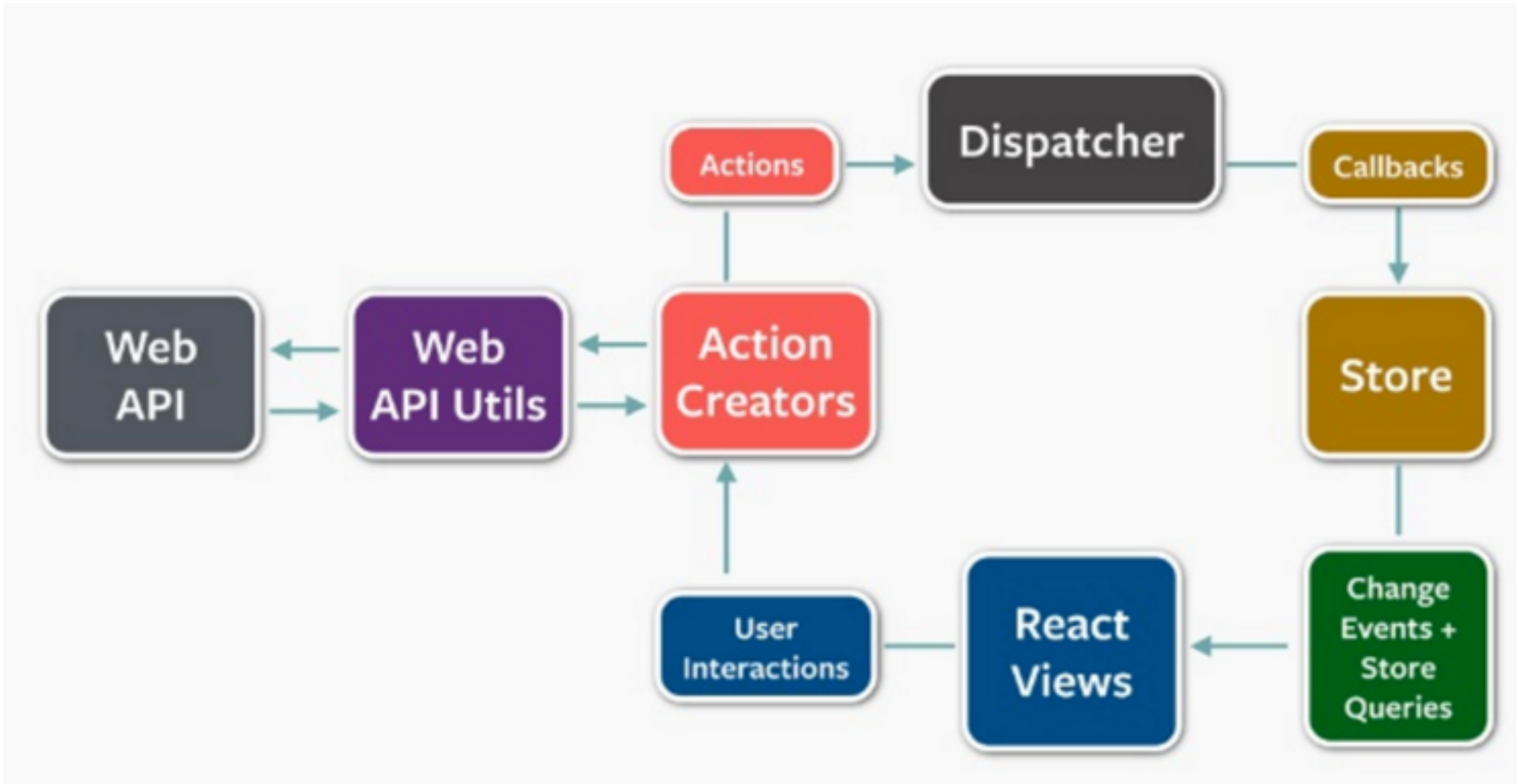



```
class PricePanel extends React.Component {
  constructor(props) {
    super(props);
    this.state = {price: 0};
  }

  componentDidMount() {
    this.setState({price: 1});
  }

  render() {
    return (
      // jsx语法
      <div>¥{this.state.price}</div>
    );
  }
}
```

- React 本身只涉及UI层，如果搭建大型应用，必须搭配一个前端框架。也就是说，你至少要学两样东西，才能基本满足需要：React + 前端框架。
- Facebook官方使用的是 Flux 框架。**Flux 是一种架构思想，专门解决软件的结构问题**。它跟MVC 架构是同一类东西，但是更加简单和清晰。



flux的单向数据流动模型



1. 用户访问 View
2. View 发出用户的 Action
3. Dispatcher 收到 Action，要求 Store 进行相应的更新
4. Store 更新后，发出一个"change"事件
5. View 收到"change"事件后，更新页面

write one run anywhere


weex

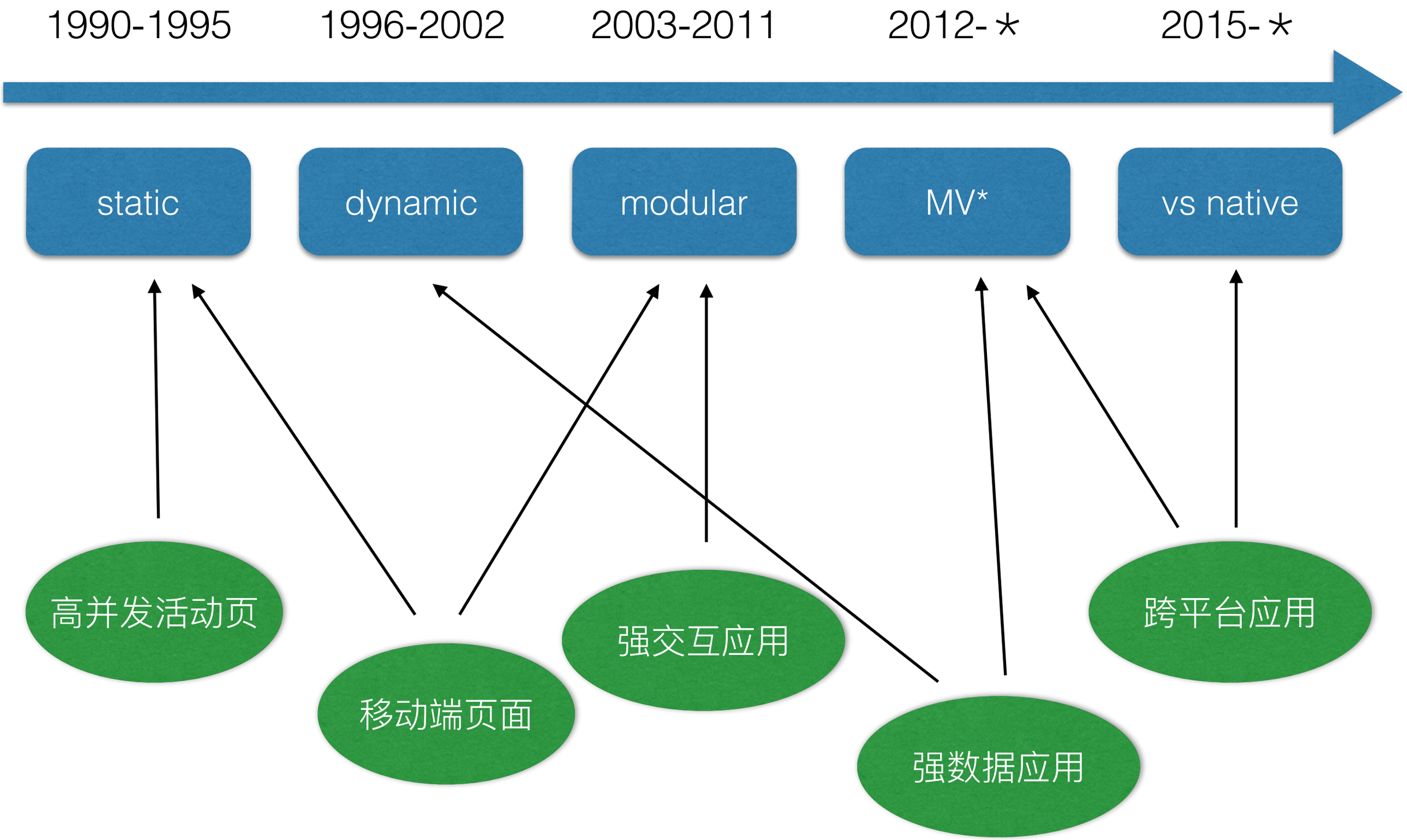

React Native


微信
小程序

```
<template>
  <container>
    <text style="font-size: {{size}}”>
     {{title}}
     </text>
  </container>
</template>
<script>
  module.exports = {
    data: {
      size: 48,
      title: 'Alibaba Weex Team'
    }
  }
</script>
<style>
  .wrapper {width: 600;}
  .title {width: 400; height: 50;}
  .highlight {color: #ff0000;}
</style>
```


```
import React, { Component } from 'react';
import { AppRegistry, StyleSheet, Text, View } from 'react-native';

class LotsOfStyles extends Component {
  render() {
    return (
      <View>
        <Text style={styles.red}>just red</Text>
      </View>
    );
  }
}
const styles = StyleSheet.create({
  red: {
    color: 'blue',
    fontWeight: 'bold',
    fontSize: 30,
  }
});

AppRegistry.registerComponent('LotsOfStyles', () => LotsOfStyles);
```


```
<!--index.wxml-->
<view class="container">
  <view class="userinfo">
    <text class="user-info">{{userinfo}}</text>
  </view>
</view>

 //index.js
 //获取应用实例
 var app = getApp()
 Page({
   data: {
     motto: 'Hello World',
     userInfo: {}
   },
   onLoad: function () {
       …
   }
 })

/**index.wxss**/
.userinfo {
  display: flex;
  flex-direction: column;
  align-items: center;
}
```


1990-1995
1996-2002
2003-2011
2012-*
2015-*
static
dynamic
modular
MV*
vs native
高并发活动页
移动端页面
强交互应用
强数据应用
跨平台应用

thx!