

# Weex

## 移动端应用架构设计和实战

@凝砺 (宁栗)



促进软件开发领域知识与创新的传播



关注InfoQ官方信息  
及时获取移动大会演讲  
视频信息



[深圳站] 2016年07月15-16日  
咨询热线: 010-89880682

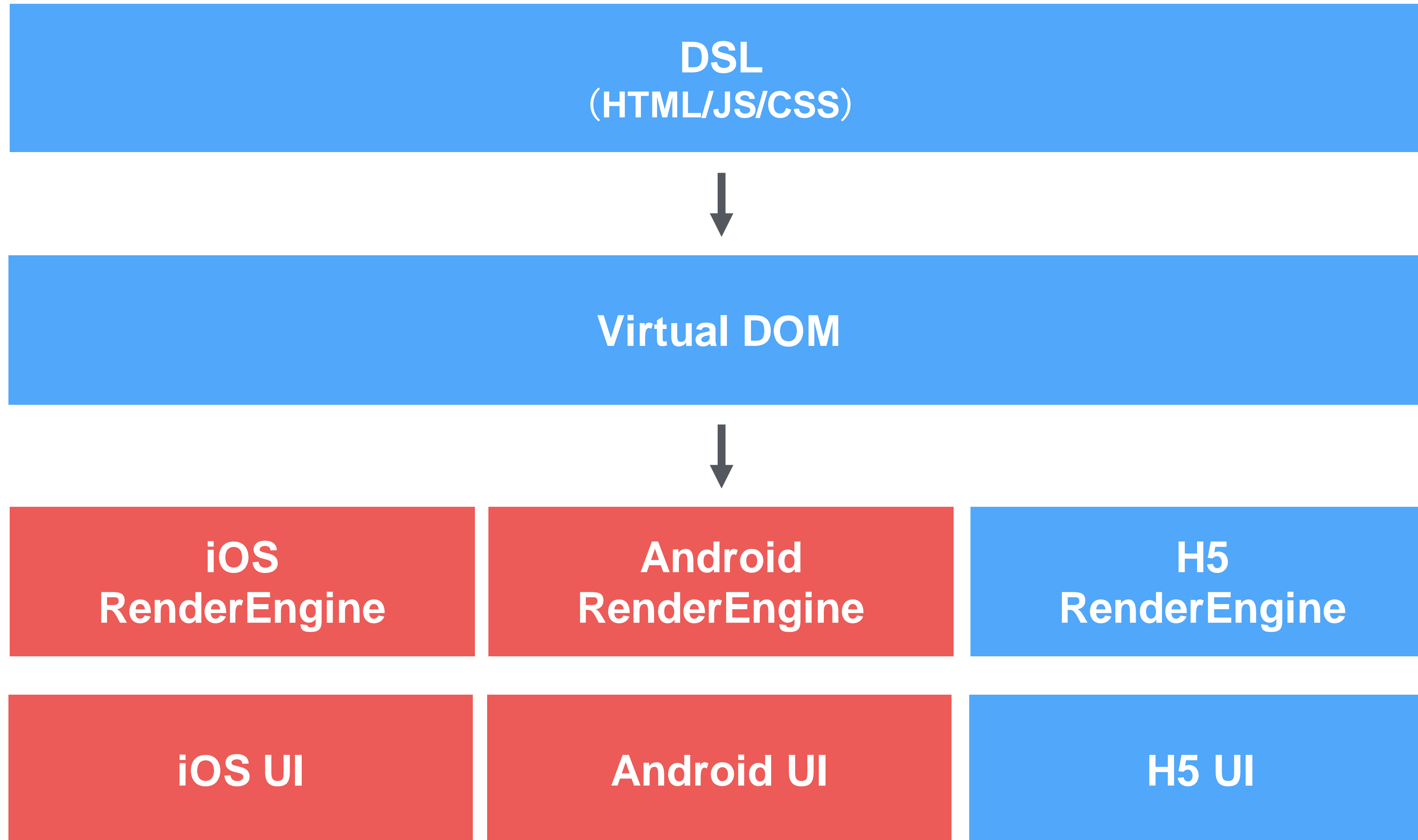


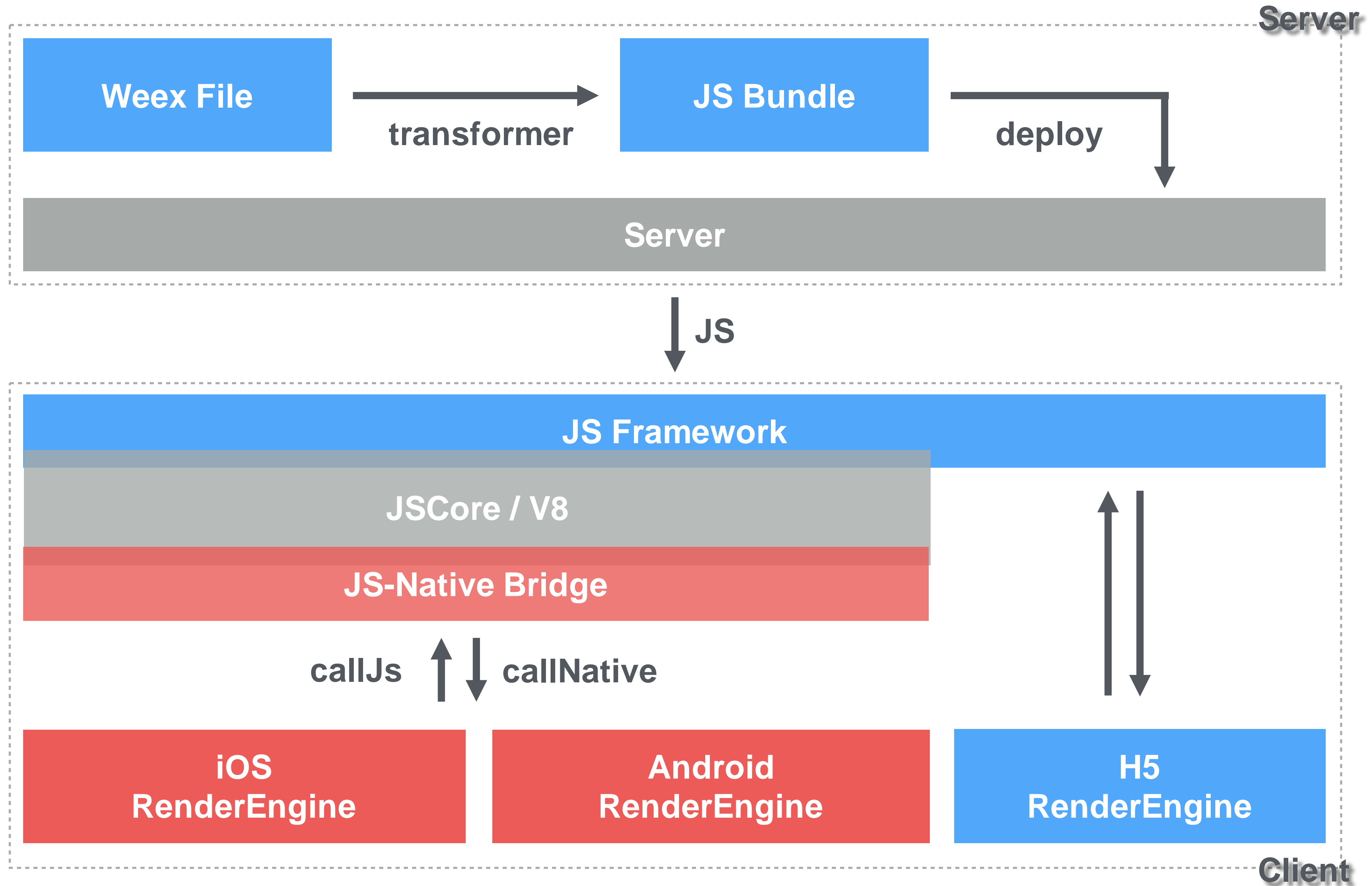
[上海站] 2016年10月20-22日  
咨询热线: 010-64738142

- 1. What is Weex**
- 2. AppFramework**
- 3. Live demo**
- 4. Q & A**



# What is Weex





```
<template>
  <div class="hello" onclick="clickHandler">
    <text>{{message0}}</text>
    <text>{{message1}}</text>
  </div>
</template>
```

```
<style>
  .hello {
    flex-direction: row;
  }
</style>
```

```
<script>
  module.exports = {
    data: {
      message0: 'Hello',
      message1: ' World.'
    },
    methods: {
      clickHandler: function() {}
    }
  }
</script>
```

DSL

— — — →

— — — →

— — — →

```
module.exports.template = {
  "type": "div",
  "classList": [
    "hello"
  ],
  "events": {
    "click": "clickHandler"
  },
  "children": [
    {"type": "text"...},
    {"type": "text"...}
  ]
};
```

```
module.exports.style = {
  "hello": {
    "flexDirection": "row"
  }
};
```

```
module.exports = {
  data: function() {
    return {
      message0: 'Hello',
      message1: ' World.'
    }
  },
  methods: {
    clickHandler: function() {}
  }
};
```

JS-Bundle

# AppFramework

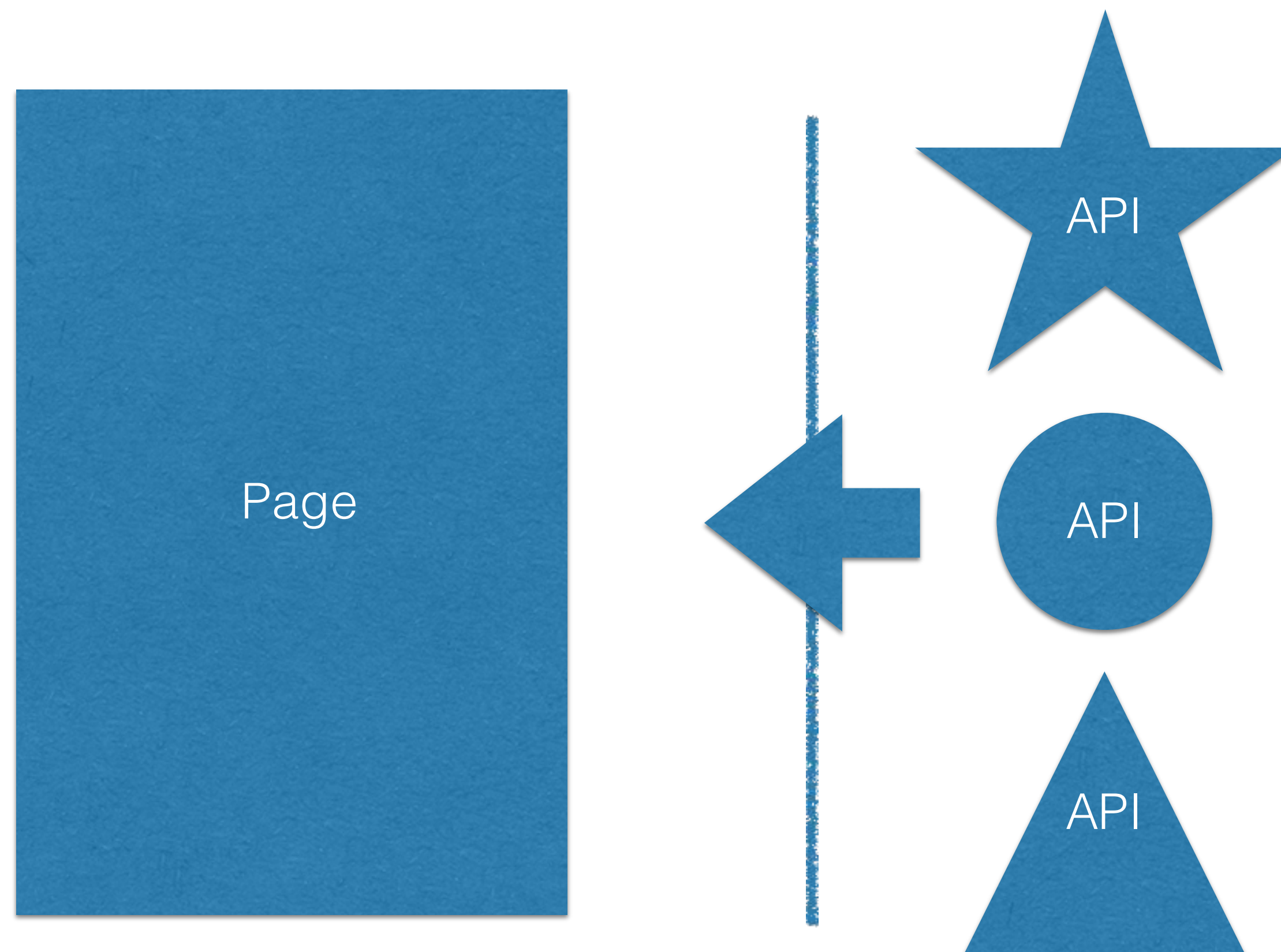


teresaromano  
PHOTOGRAPHY



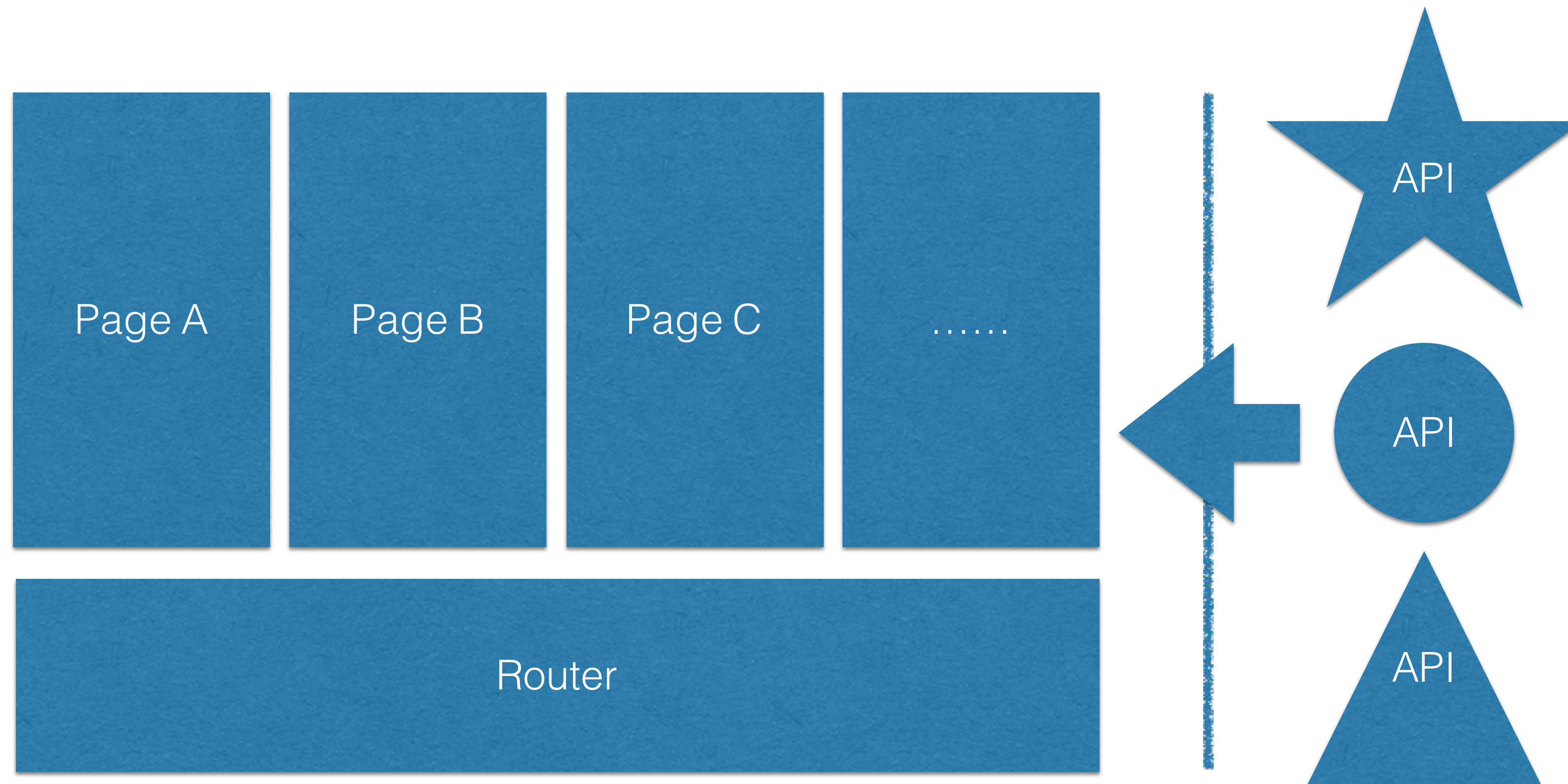
# Weex 移动端研发模型 (I)

## single page + features

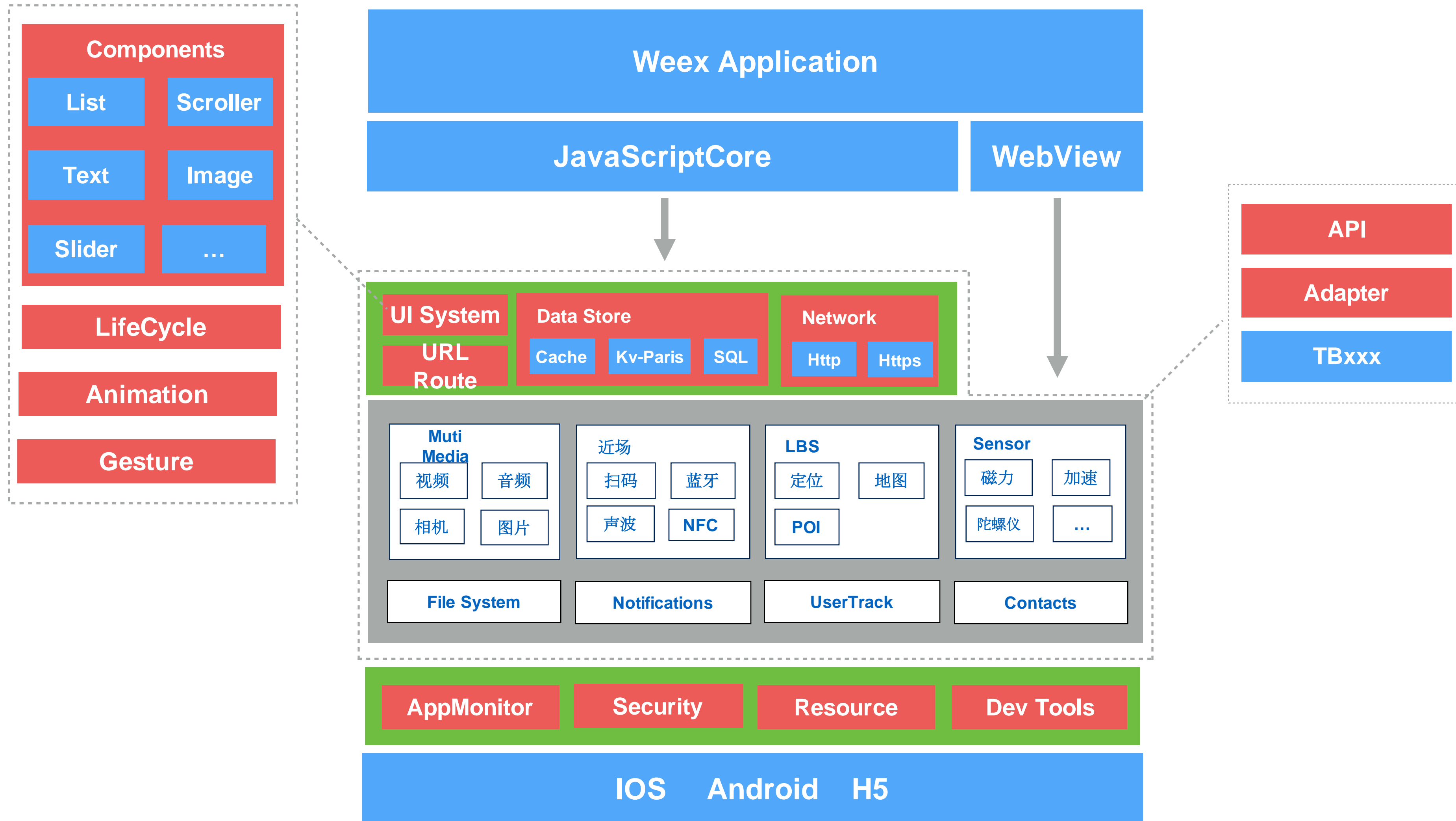


# Weex 移动端研发模型 (II)

## pages + router + features



# 系统架构框图

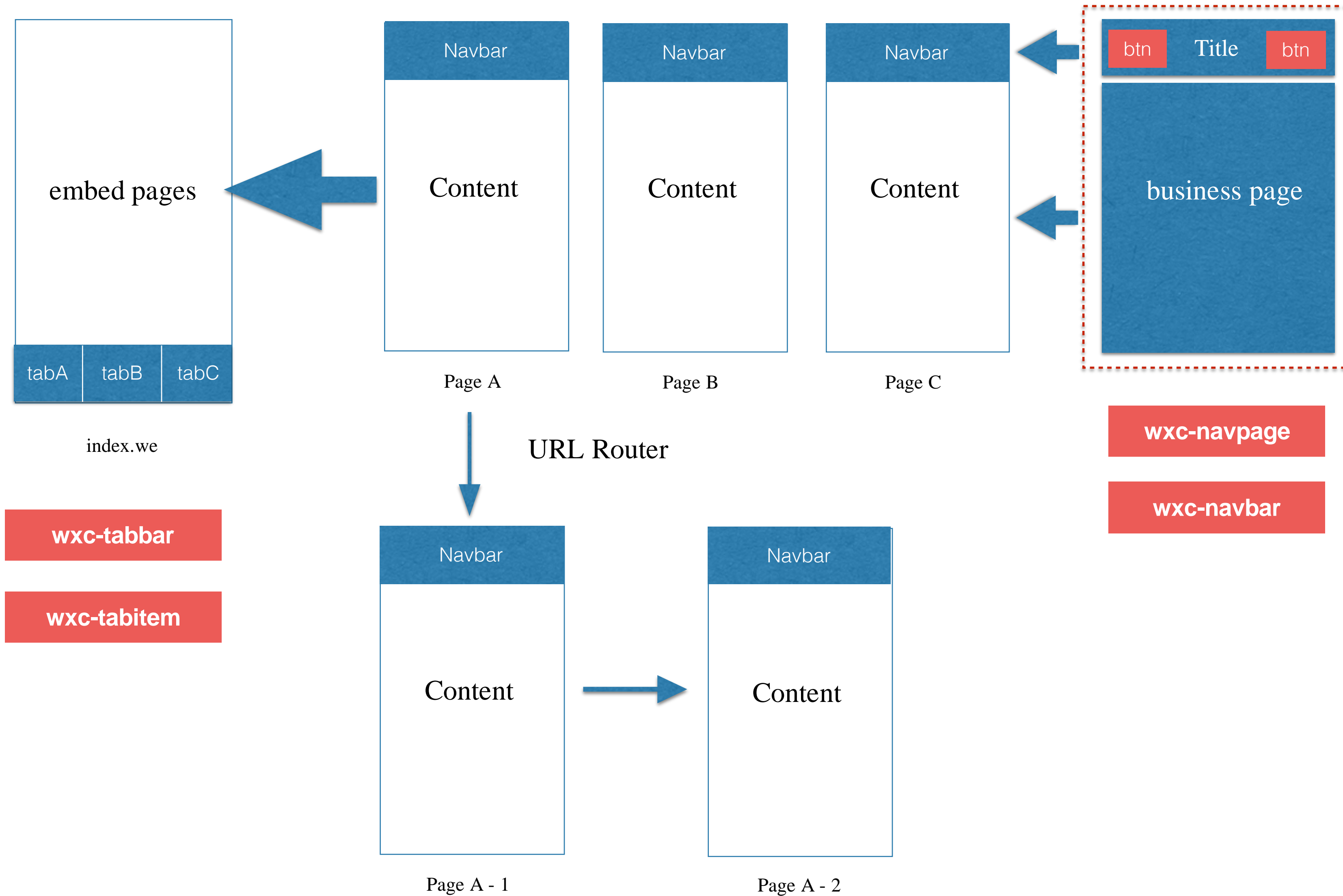


## 六脉神剑

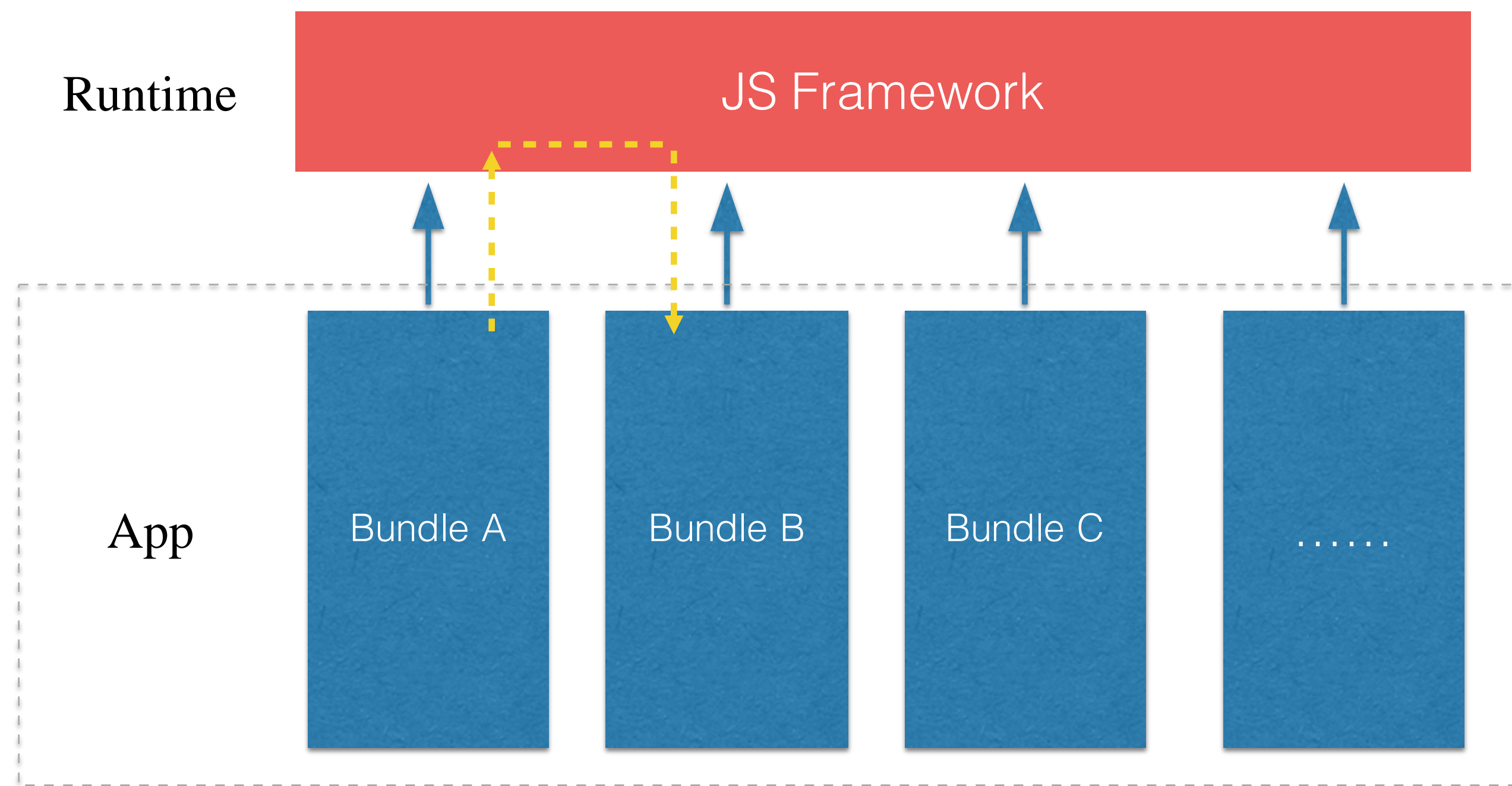
1. 导航体系: TabBar、Navigator、URL Router
2. 生命周期: Application、Page
3. 底层基建: Network、Data Store、Events ...
4. 扩展性: Component、Module、Adapter
5. 性能监控: Memory、Frame Rate、Render Time
6. 开发调试



# 页面导航



# 页面管理与多实例

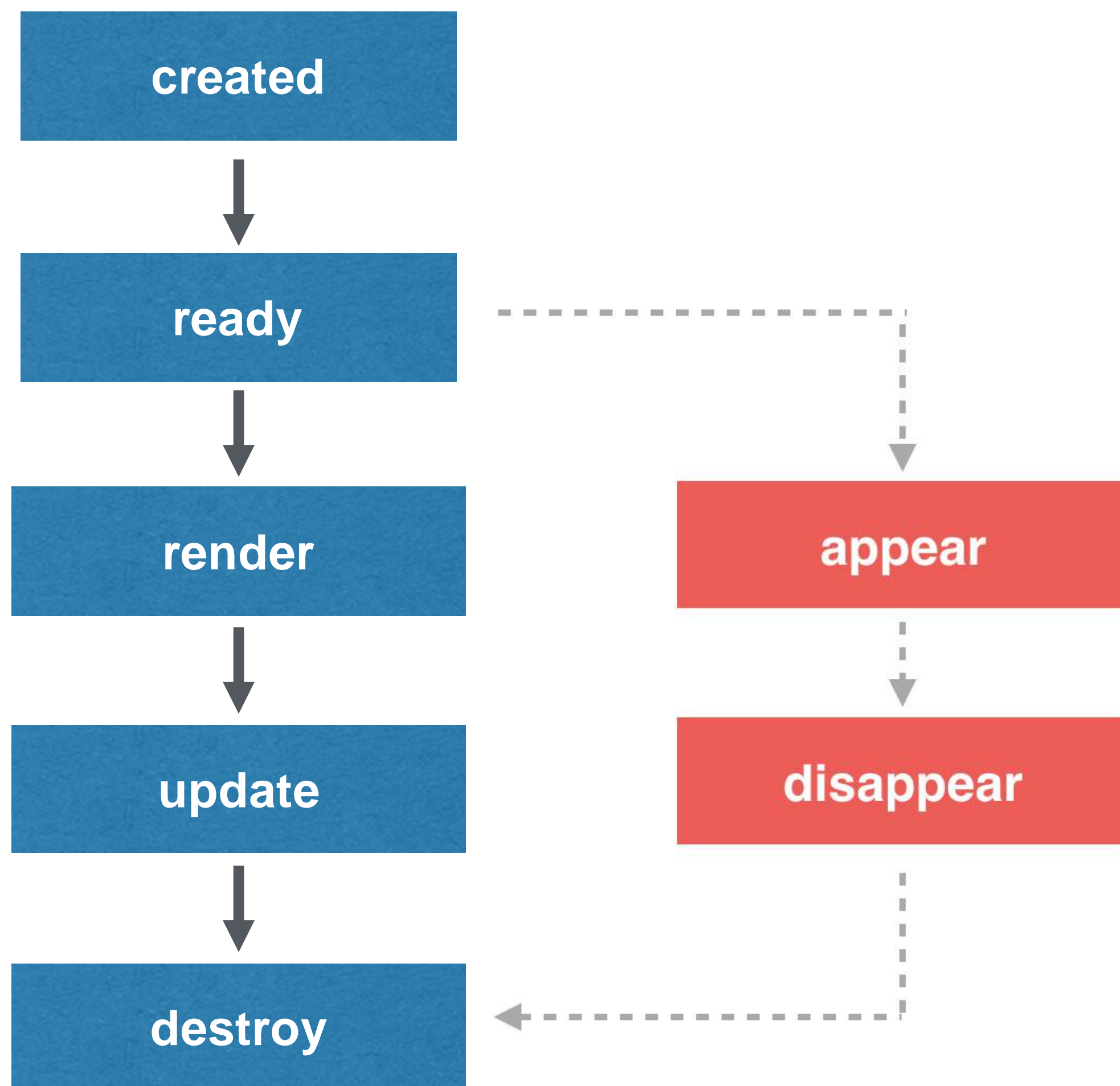


Application = bundles      bundle = instance

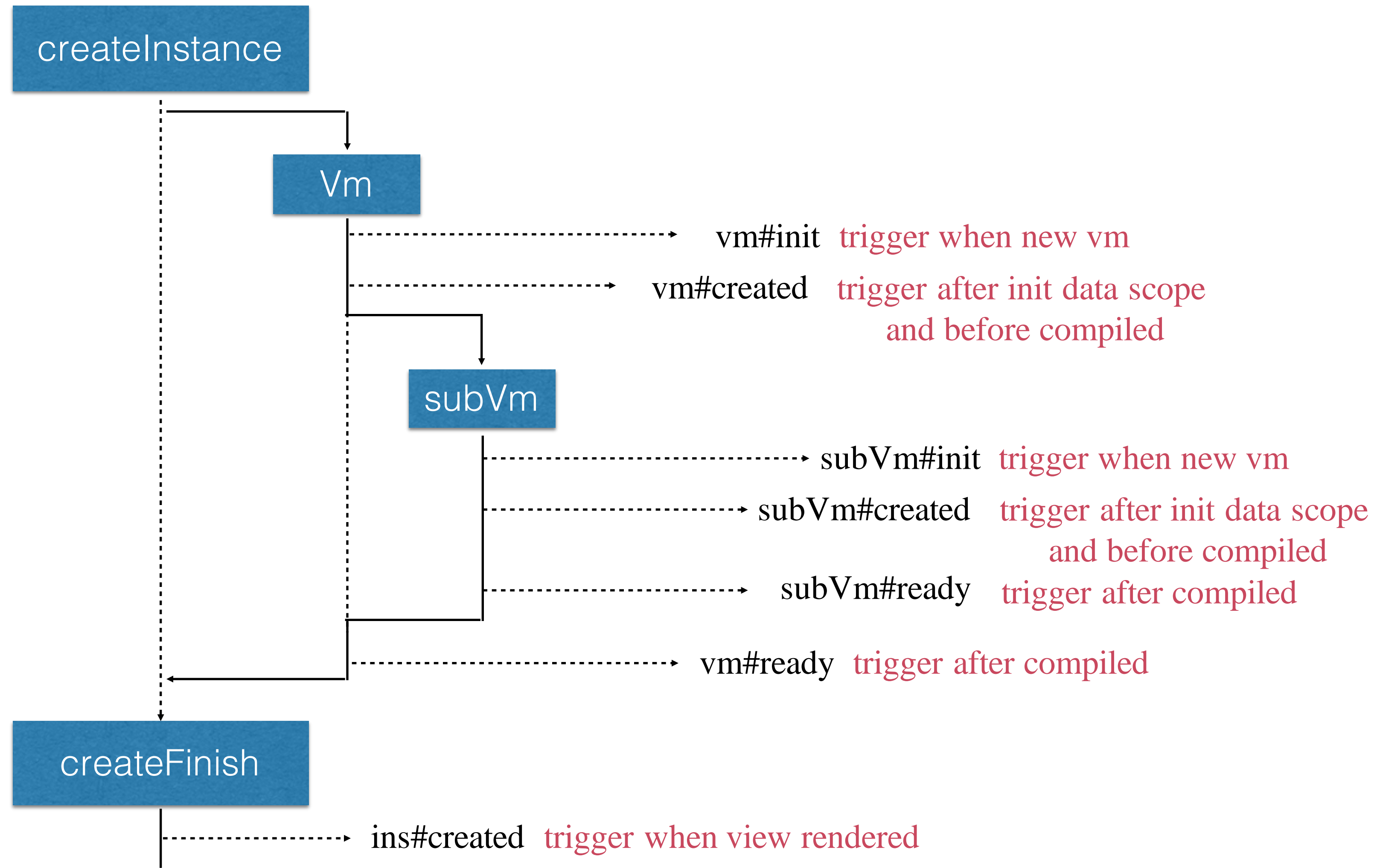
All instances are managed by JS Framework .



# 生命周期



# 生命周期（实例）

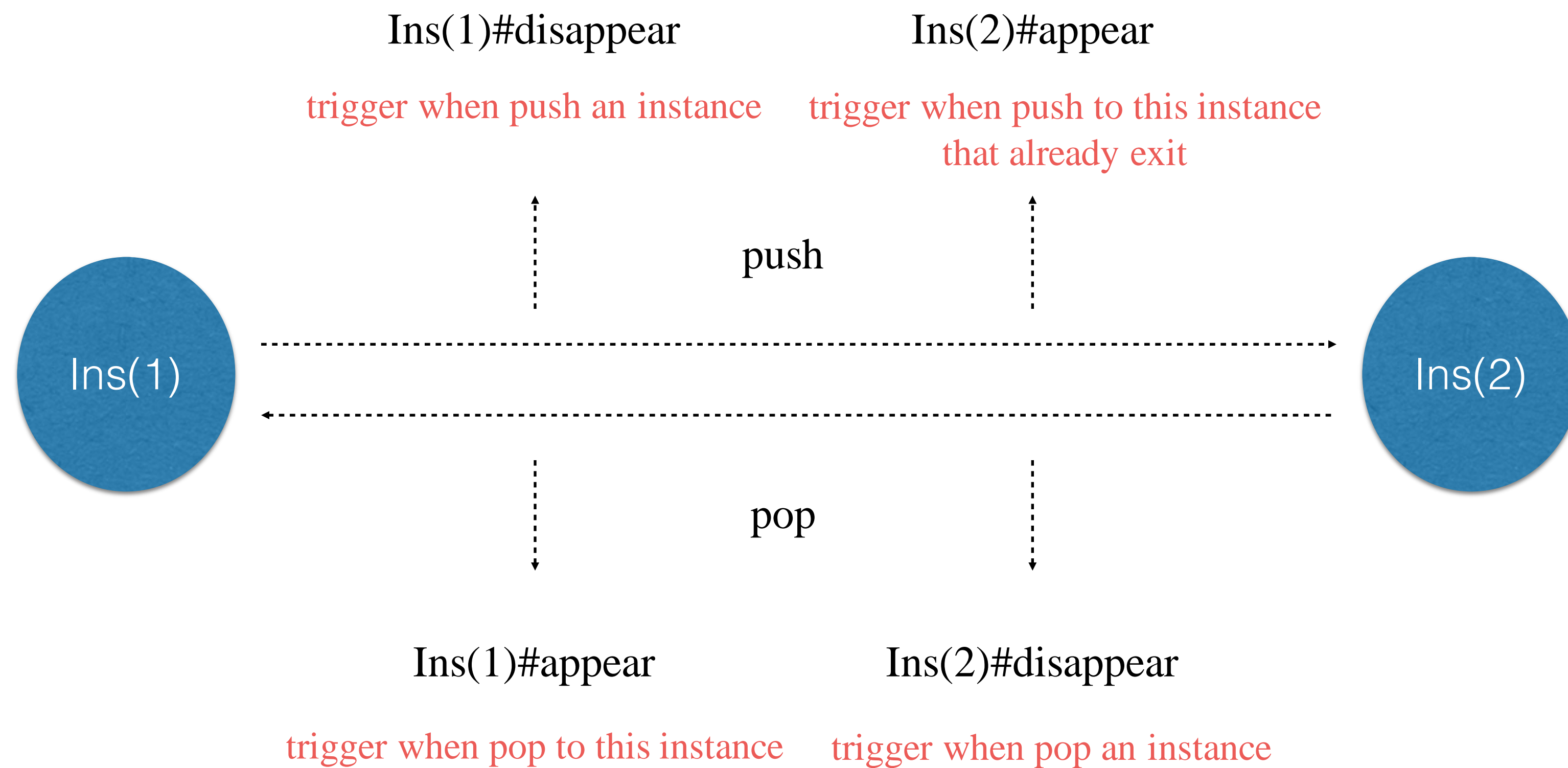


创建实例流程





# 生命周期（导航）



# 扩展性

1. Component

2. 公共组件

3. Module

4. Adapter 适配层



```

@implementation WXImageComponent

- (instancetype)initWithRef:(NSString *)ref type:(NSString *)type styles:(NSDictionary *)styles attributes:(NSDictionary *)attributes events:(NSArray *)events weexInstance:(WXSDKInstance *)weexInstance
{
    if (self = [super initWithRef:ref type:type styles:styles attributes:attributes events:events weexInstance:weexInstance]) {
        _imageSrc = [WXConvert NSString:attributes[@"src"]];
        _resizeMode = [WXConvert UIViewContentMode:attributes[@"resize"]];
        _imageQuality = [WXConvert WXImageQuality:styles[@"quality"]];
        _imageSharp = [WXConvert WXImageSharp:styles[@"sharpen"]];
    }

    return self;
}

- (UIView *)loadView
{
    return [[WXImageView alloc] init];
}

- (void)viewDidLoad
{
    UIImageView *imageView = (UIImageView *)self.view;
    imageView.contentMode = _resizeMode;
    imageView.userInteractionEnabled = YES;
    imageView.clipsToBounds = YES;
    imageView.exclusiveTouch = YES;
}

- (void)updateStyles:(NSDictionary *)styles
{
    if (styles[@"quality"]) {
        _imageQuality = [WXConvert WXImageQuality:styles[@"quality"]];
    }
}

- (void)updateAttributes:(NSDictionary *)attributes
{
    if (attributes[@"src"]) {
        _imageSrc = [WXConvert NSString:attributes[@"src"]];
    }
    if (attributes[@"resize"]) {
        _resizeMode = [WXConvert UIViewContentMode:attributes[@"resize"]];
        self.view.contentMode = _resizeMode;
    }
}

```

## LifeCycle Of Component

**loadView:** Creates the view that the component manages.

**viewDidLoad:** Called after the component's view is loaded and set.

**updateStyles:** Called when component's style are updated.

**updateAttributes:** Called when component's attributes are updated.

**<image style="your-custom-style" src="image-remote-source" resize="contain/cover/stretch"></image>**



```
//wxc-tabitem.we
<template>
  <div class="container" style="background-color: {{backgroundColor}}"
    onclick="onclickitem">
    <image class="top-line" src = "http://gtms03.alicdn.
      com/tps/i3/TB1mdsiMpXXXXXpXXXXNw4JIXXX-640-4.png"></image>
    <image class="tab-icon" src = {{icon}}></image>
    <text class="tab-text" style="color: {{titleColor}}">{{title}}</text>
  </div>
</template>
```

```
//wxc-tabbar.we
<template>
  <div class="wrapper">
    <embed class="content" style="visibility:{{visibility}}" repeat={{tabItems}}
      src={{src}} type="weex"></embed>
    <div class="tabbar" append = "tree">
      <wxc-tabitem repeat={{tabItems}} index={{index}} icon={{icon}}
        title={{title}} title-color={{titleColor}}></wxc-tabItem>
    </div>
  </div>
</template>
```



```
<template>
  <div style="flex-direction: column;">
    <wxc-tabbar tab-items = {{tabItems}}></wxc-tabbar>
  </div>
</template>

<script>
  require('weex-components');
  module.exports = {
    data: {
      dir: 'examples',
      tabItems: [
        {
          index: 0,
          title: 'tab1',
          titleColor: '#000000',
          icon: '',
          image: 'http://gtms01.alicdn.com/tps/i1/TB1qw.hMpXXXagXXX9t7RGVXX-46-46.png',
          selectedImage: 'http://gtms04.alicdn.com/tps/i4/TB16jjPMpXXXazXVXX9t7RGVXX-46-46.png',
          src: 'component/tabbar/tabbar-item.js?itemId=tab1',
          visibility: 'visible',
        },
      ],
    },
  }
}
```



## 如何选择

	Component	公共组件
开发成本	前端+客户端	前端
动态性	依赖客户端升级 (iOS, Android)	动态部署
定制型	弱	强
交互性	可支持复杂交互行为	简单交互行为
使用场景	常用基础组件或与业务相关长期稳定的组件	个性化定制组件

Tips: 公共组件的可塑性取决于WeexSDK提供的基础能力, 包括手势和动画等。

欢迎各位开发者通过github提交  
您的作品!!!



# 如何组织页面代码

1. 从整体进行组件化分离
2. 定义每个组件
3. 把组件有机组合起来



```
@implementation WXEventModule

@synthesize weexInstance;

WX_EXPORT_METHOD(@selector(openURL:))

- (void)openURL:(NSString *)url
{
    NSString *newURL = url;
    if ([url hasPrefix:@"//"]) {
        newURL = [NSString stringWithFormat:@"http:%@", url];
    } else if (![url hasPrefix:@"http"]) {
        // relative path
        newURL = [NSURL URLWithString:url relativeToURL:weexInstance.scriptURL].absoluteString;
    }

    UIViewController *controller = [[WXDemoViewController alloc] init];
    ((WXDemoViewController *)controller).url = [NSURL URLWithString:newURL];

    [[weexInstance.viewController navigationController] pushViewController:controller animated:YES];
}

@end
```

Weex后续会陆续提供一些基础性module,如定位服务、调用相册和拍照等, 偏重实际业务的module扩展还有赖于开发者自定义。

欢迎各位开发者通过github提交  
您的作品!!!





```
[WXSDKEngine initWithSDKEnvironment];

[WXSDKEngine registerHandler:[WXImgLoaderDefaultImpl new] withProtocol:@protocol(WXImgLoaderProtocol)];
[WXSDKEngine registerHandler:[WXEventModule new] withProtocol:@protocol(WXEventModuleProtocol)];

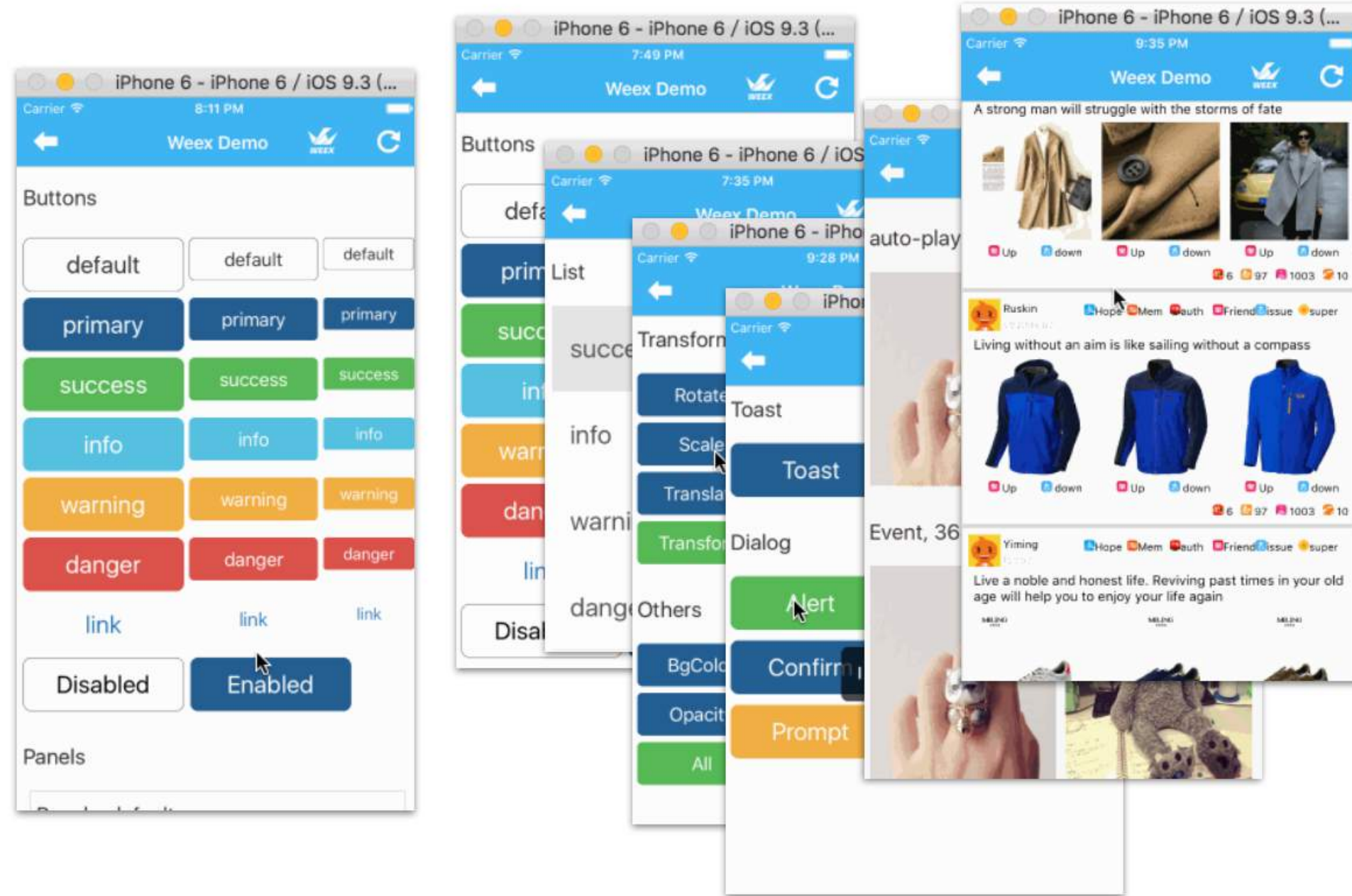
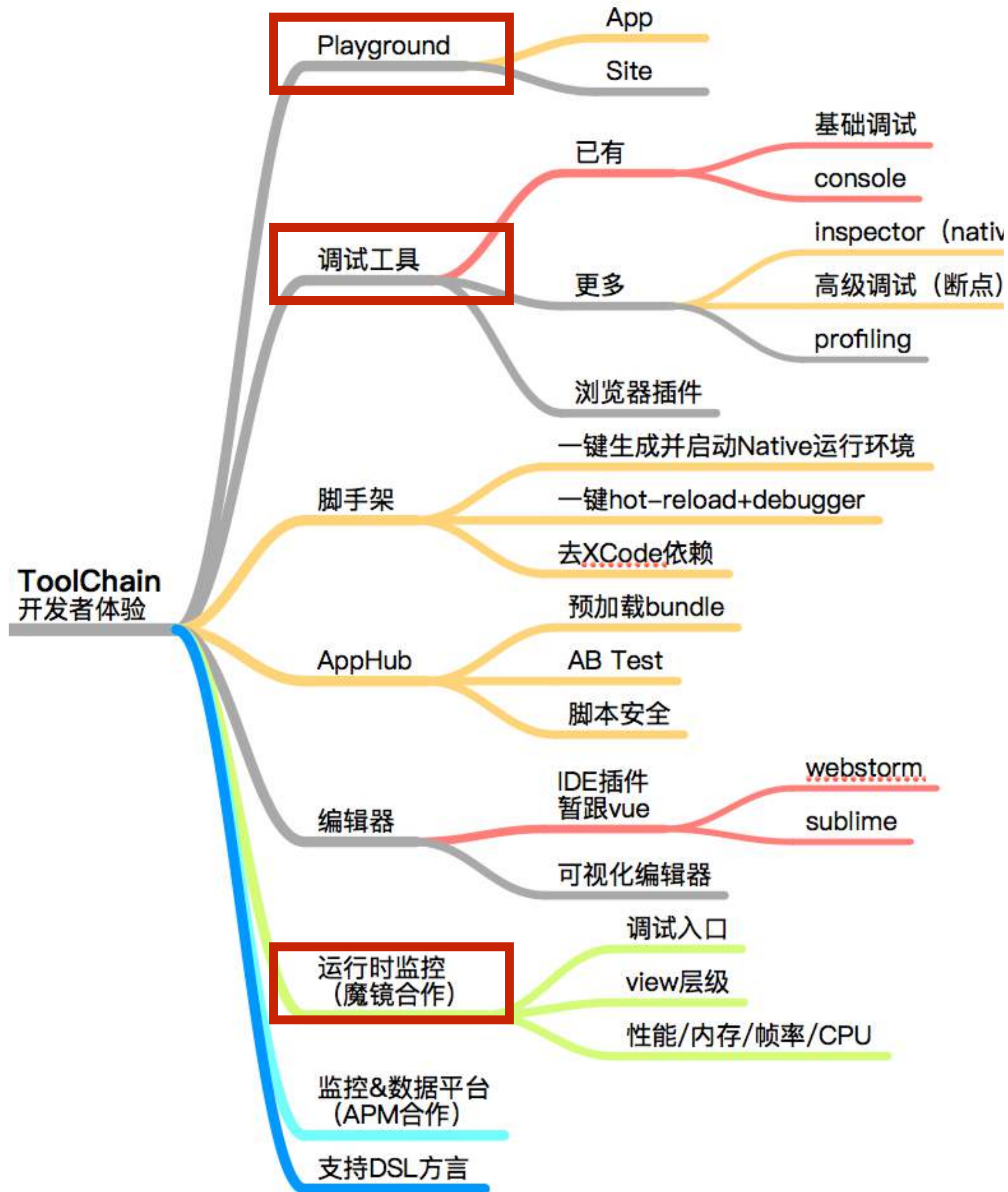
#pragma mark -
#pragma mark WXImgLoaderProtocol

- (id<WXImageOperationProtocol>)downloadImageWithURL:(NSString *)url imageFrame:(CGRect)imageFrame userInfo:
(NSDictionary *)userInfo completed:(void(^)(UIImage *image, NSError *error, BOOL finished))completedBlock
{
    if ([url hasPrefix:@"//"]) {
        url = [@"http:" stringByAppendingString:url];
    }
    return (id<WXImageOperationProtocol>)[[SDWebImageManager sharedManager] downloadImageWithURL:[NSURL
        URLWithString:url] options:0 progress:^(NSInteger receivedSize, NSInteger expectedSize) {
    } completed:^(UIImage *image, NSError *error, SDImageCacheType cacheType, BOOL finished, NSURL *imageURL)
        {
            if (completedBlock) {
                completedBlock(image, error, finished);
            }
        }
    ]];
}
```

Tips: 开发者可依据协议约定, 自行实现协议。通过注册机制, 新的实现类会动态替换已有的实现。



# 开发体验



- Tutorial
- Syntax

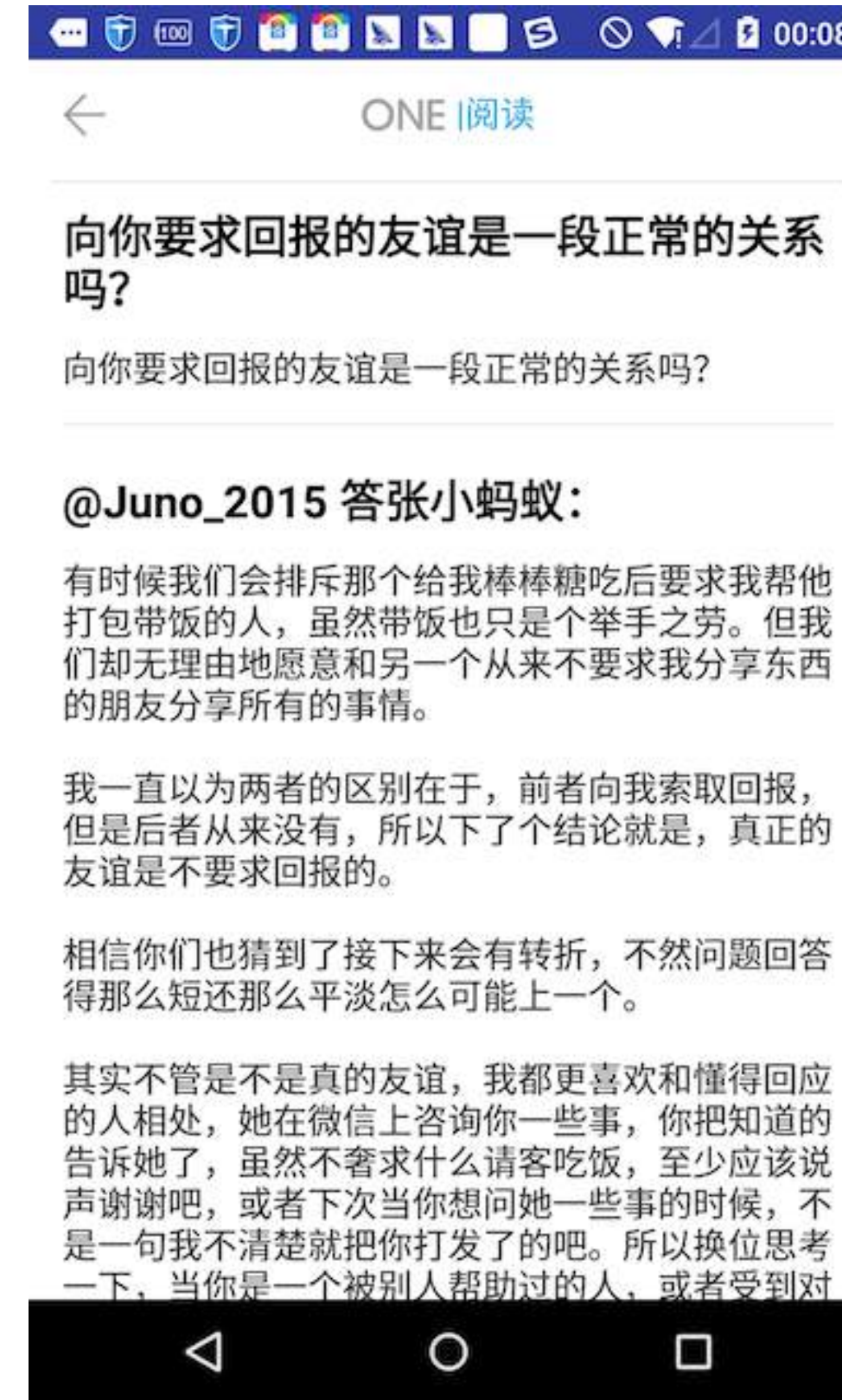
## References

- Data
- Style
- Event
- Display
- Render
- Component
- Find
- Component
- Page
- How-tos
- Manage
- Main
- Requ
- Trans

## Service & Tools

- Bootstrap
- Component
- Instance
- Built-in
  - CLI
  - Transformer
  - Playground App





引用于 [https://github.com/dodola/WeexOne?utm\\_source=tuicool&utm\\_medium=referral](https://github.com/dodola/WeexOne?utm_source=tuicool&utm_medium=referral)



*Live Demo*

MCSS  
MARINE & FAMILY SERVICES