

OSC原创会

年终盛典 2016

Hybrid技术在Flyme的应用实践

BY 杨德模

2016.12.04



About me

OSC 原创会
年终盛典 2016

- 2011.7-2014.6 腾讯 WebQQ、Q+、手Q、Q群
- 2014.7-2015.10 唯品会 特卖会移动平台前端架构
- 2015.11- 魅族科技, Hybrid App架构设计和落地、前端生态和基础设施建设
- <https://github.com/chemdemo>

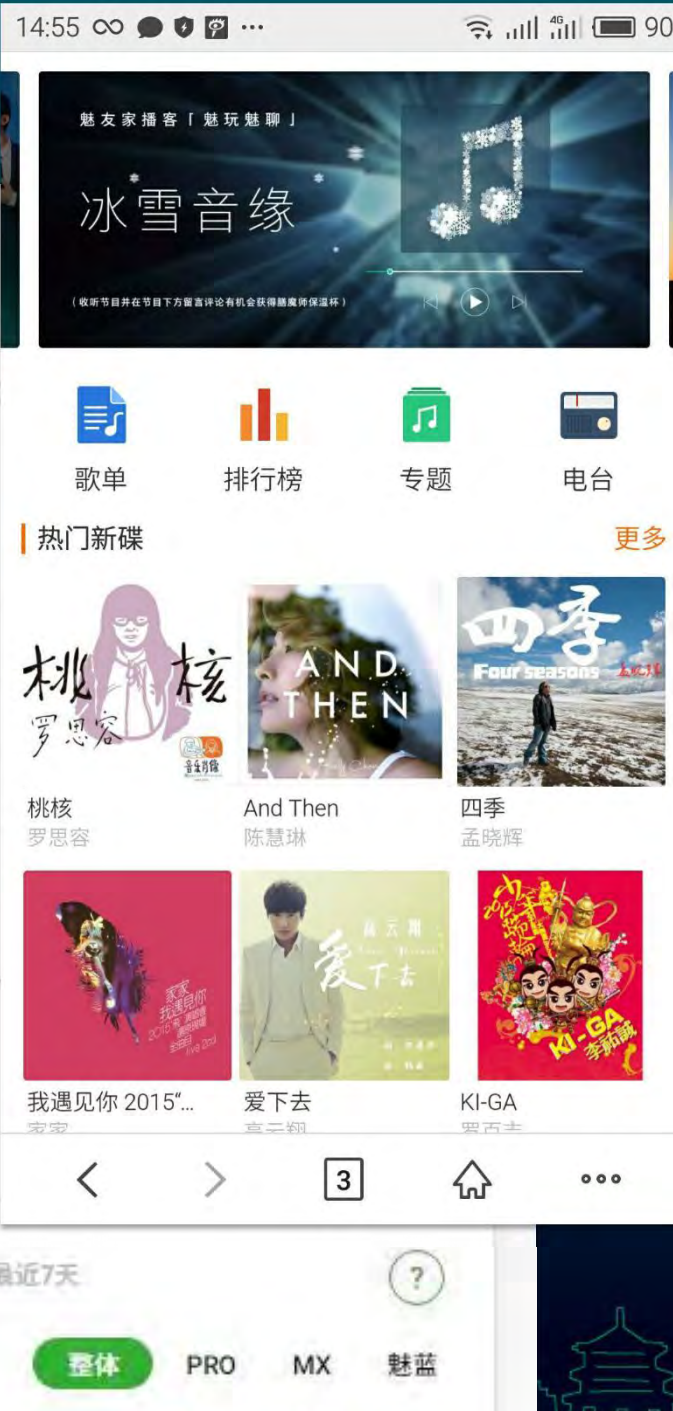
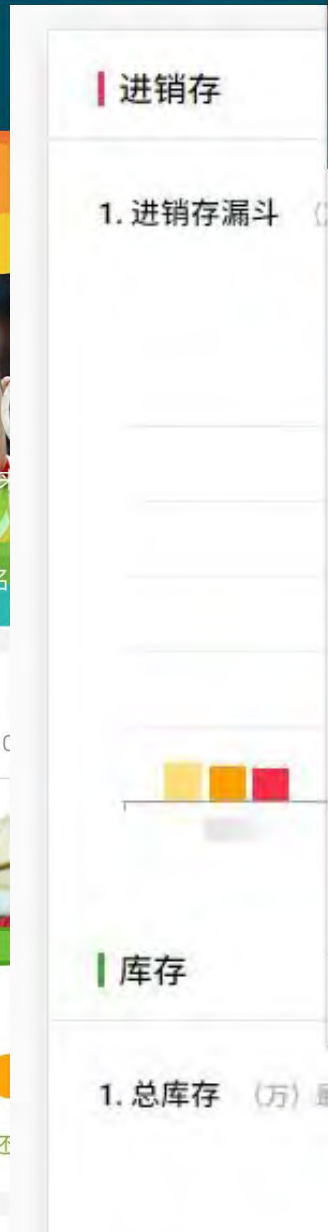
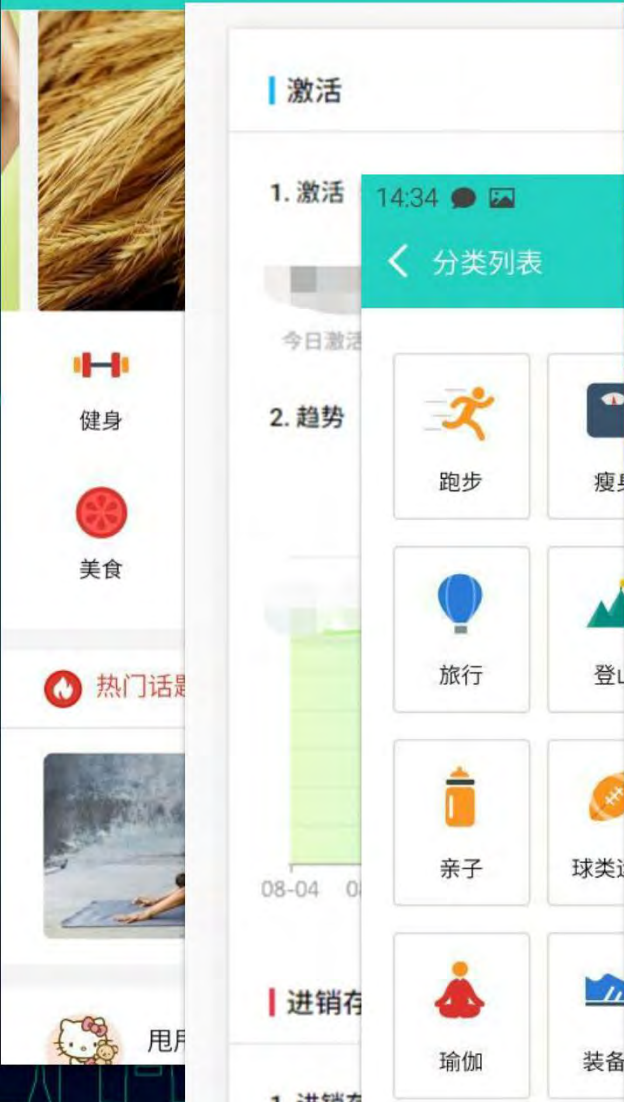
1. Flyme hybrid 原有架构简析

2. 通用Hybrid App开发体系的建设

- 通讯
- 资源管理
- 性能优化

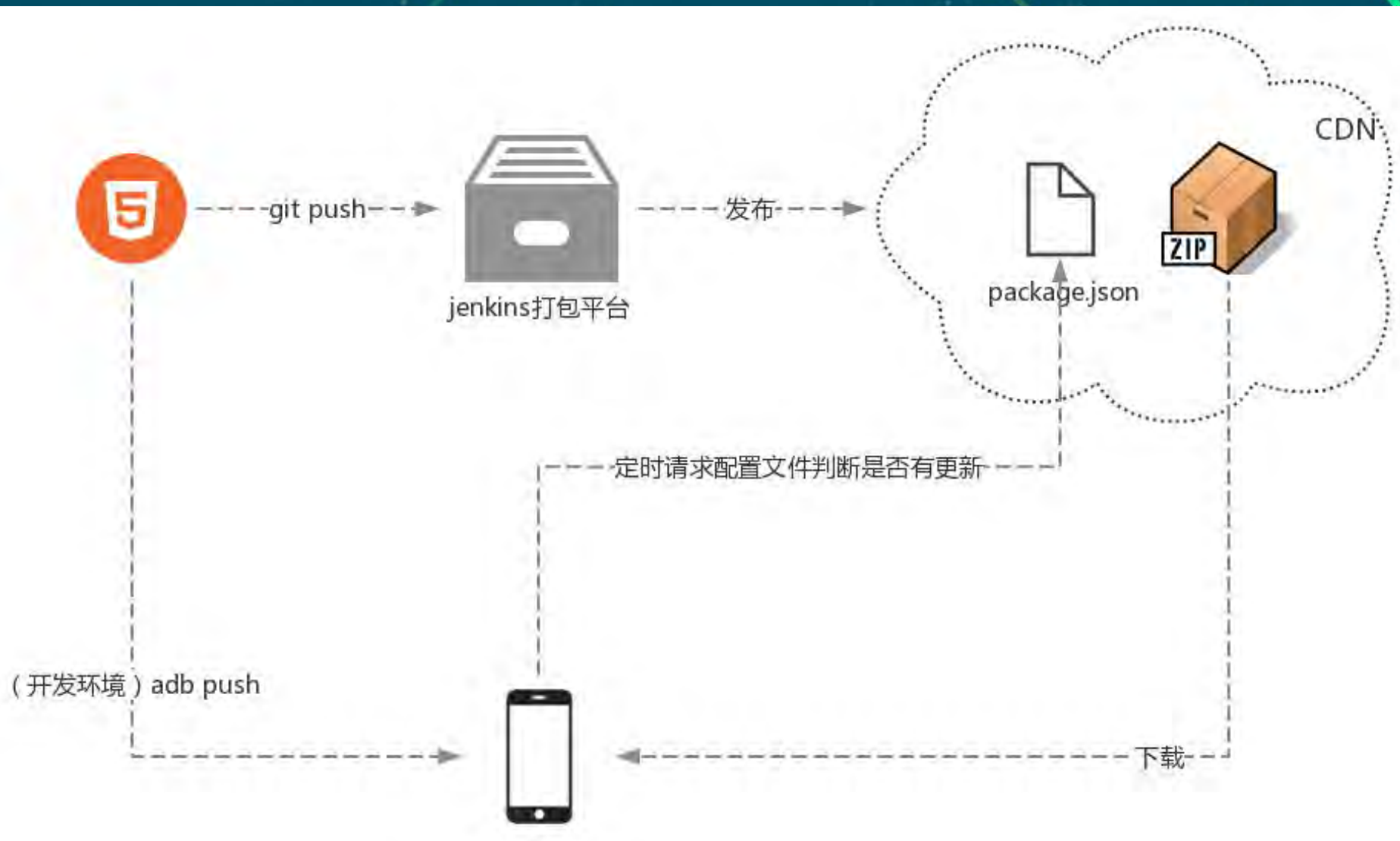
3. 前端工程、组件生态、工具链

Hybrid Apps on FlymeOS



Flyme Hybrid 原有架构流程

OSC 原创会
年终盛典 2016



关于老架构

流程简单

依据配置文件更新，不涉及服务端

代码、功能复用

各团队自己定制webview，公用的功能需要重复开发

不同团队之间，api名称和调用有各具风格，无法公用

接口设计混乱，扩展性差

直接采用Webview.addJavascriptInterface开放API

污染js宿主对象

低版本安卓存在反射漏洞

类JSONP的异步调用请求和响应分离，且无法保证顺序

资源url不一致

H5使用file协议访问，需要安卓处理各种跨域等权限问题

对于分享、H5活动等线上页面需要另外部署，即同一资源个url

H5资源包分发控制能力弱

无法做灰度发布

无法做分包控制

无法做增量更新

做不到实时推送、热更

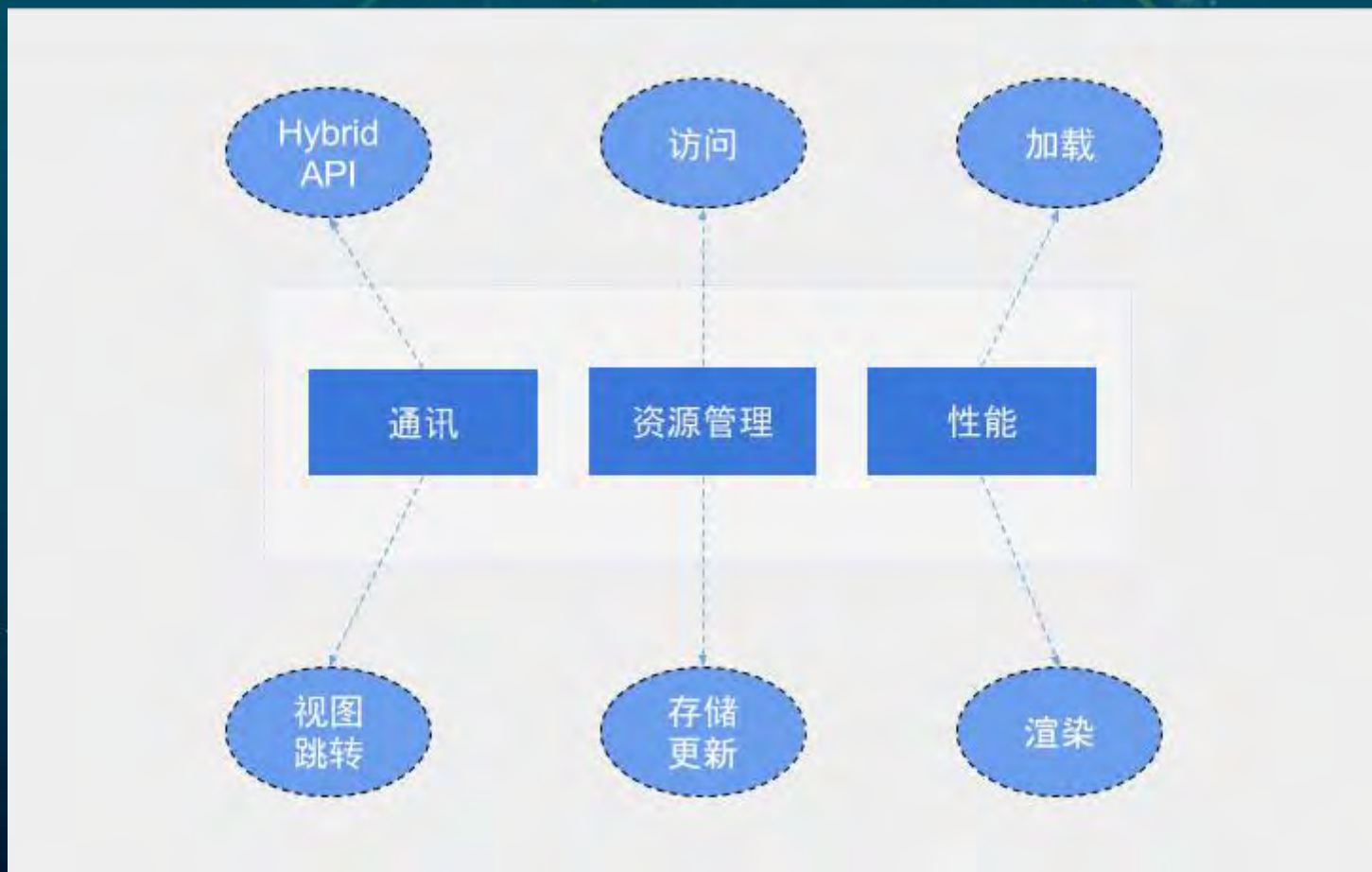
H5打包、发布繁琐

使用jenkins打包静态资源，走java发布渠道发布，流程漫长

问题	解决方案
代码复用	模块化、SDK、前端工程
接口设计	模块化, 优雅设计
访问url不一致	在线和离线均采用唯一url
H5包分发控制	包动态更新接口+push通道
H5编译、打包、发布	自建包管理平台+前端工具

通用的Hybrid架构

OSC 原创会
年终盛典 2016



设计理论: <https://github.com/chemdemo/chemdemo.github.io/issues/12>

- Android调用H5

```
webview.loadUrl("javascript: alert('hello world')");
```

- Android调用H5: shouldOverrideUrlLoading

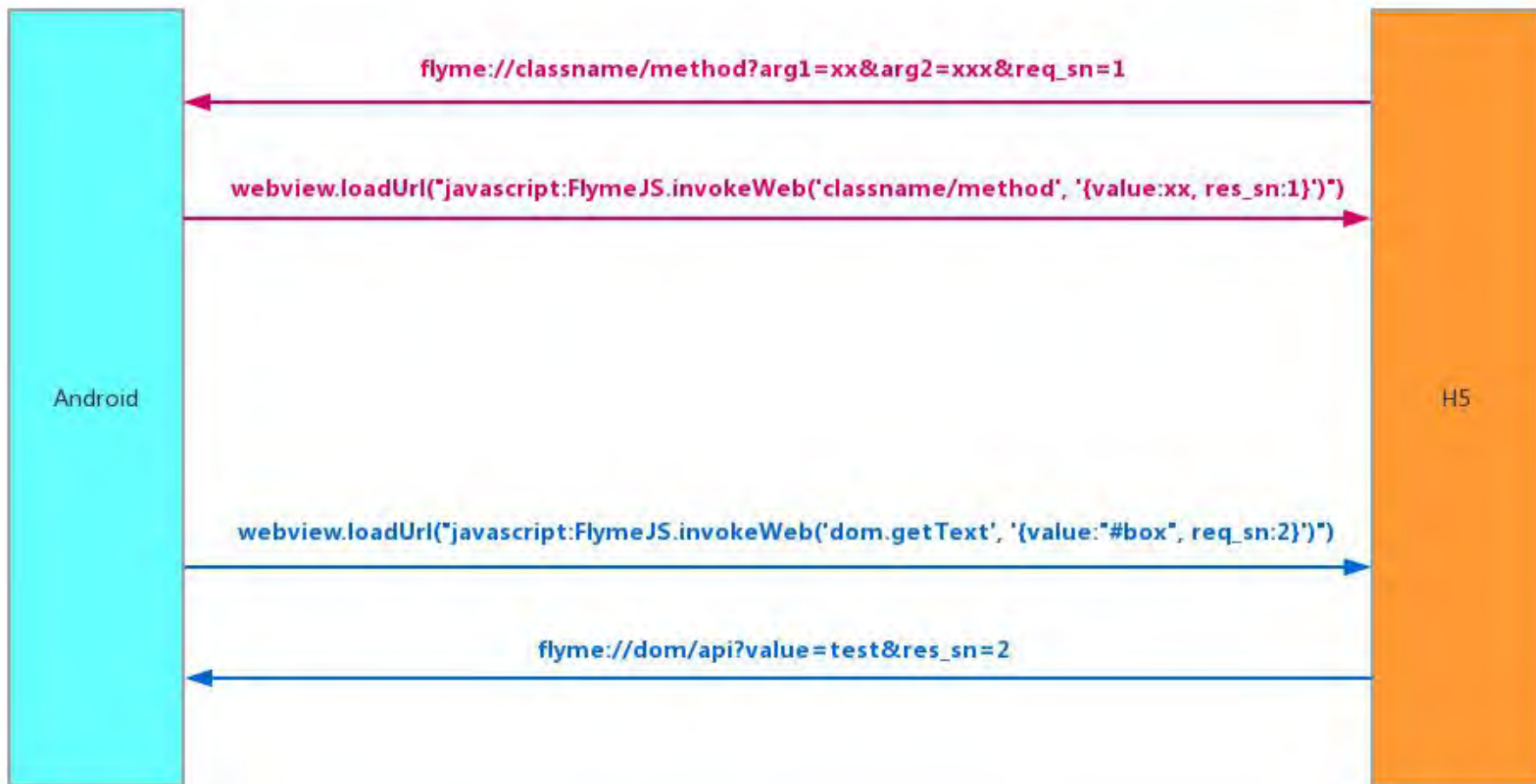
```
@Override
public boolean shouldOverrideUrlLoading(WebView view, String url) {
    // 自定义的schema
    if (url.indexOf("myschema://") != -1) {
        ... // other code
        // 返回true 则表明webview已经“消费”了H5的request事件
        return true;
    }
    // 返回false, webview将用默认的方式处理H5的request事件
    return false;
}
```



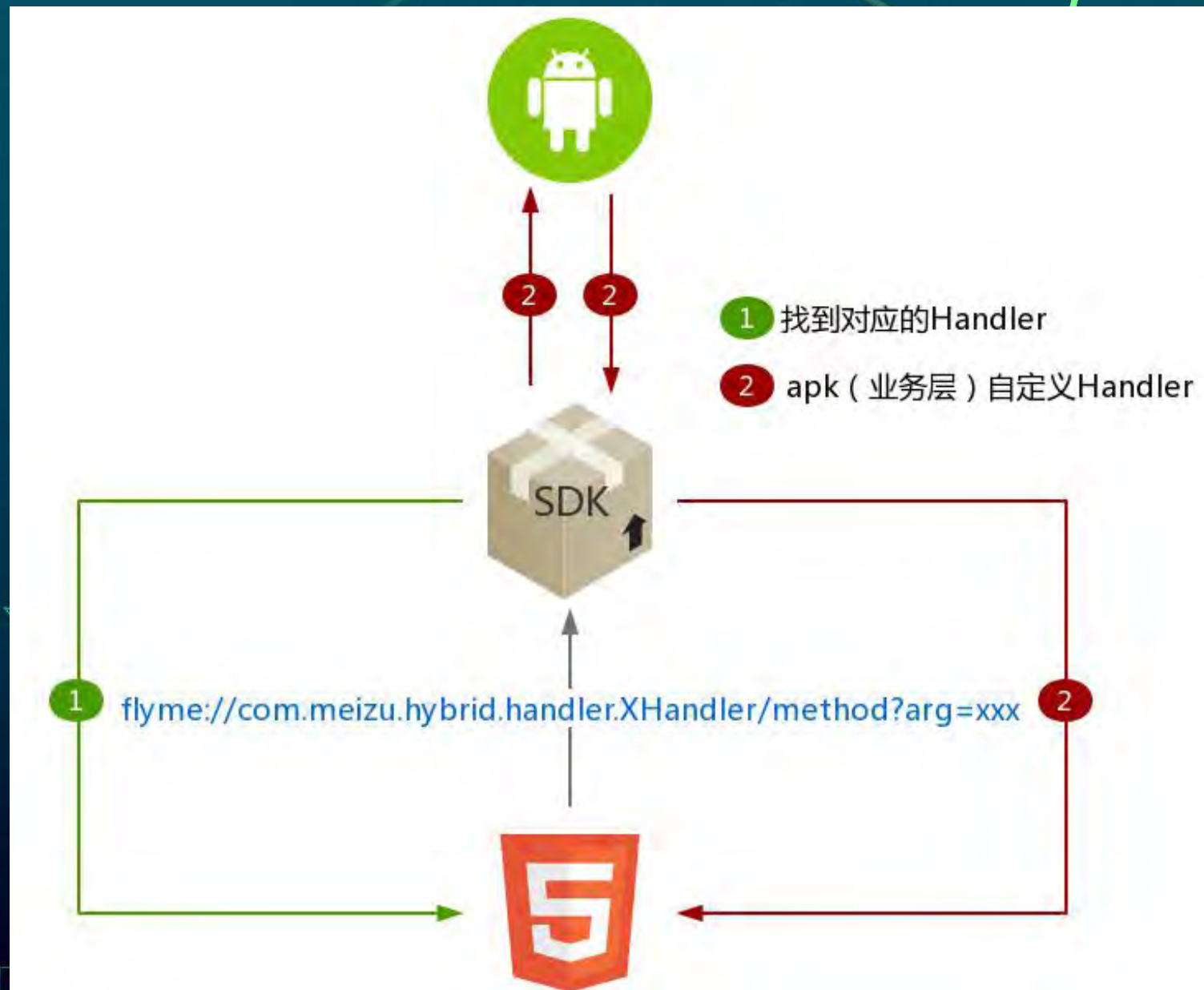
- 在SDK内部封装桥协议约定
- H5的调用封装成以下格式：

```
flyme://[类名]/[方法]?[参数...]
```

- 业务层继承SDK，并根据实际需求扩展Handler
- JS SDK遵循ES6的模块化写法，并提供完善的二次编译打包工具

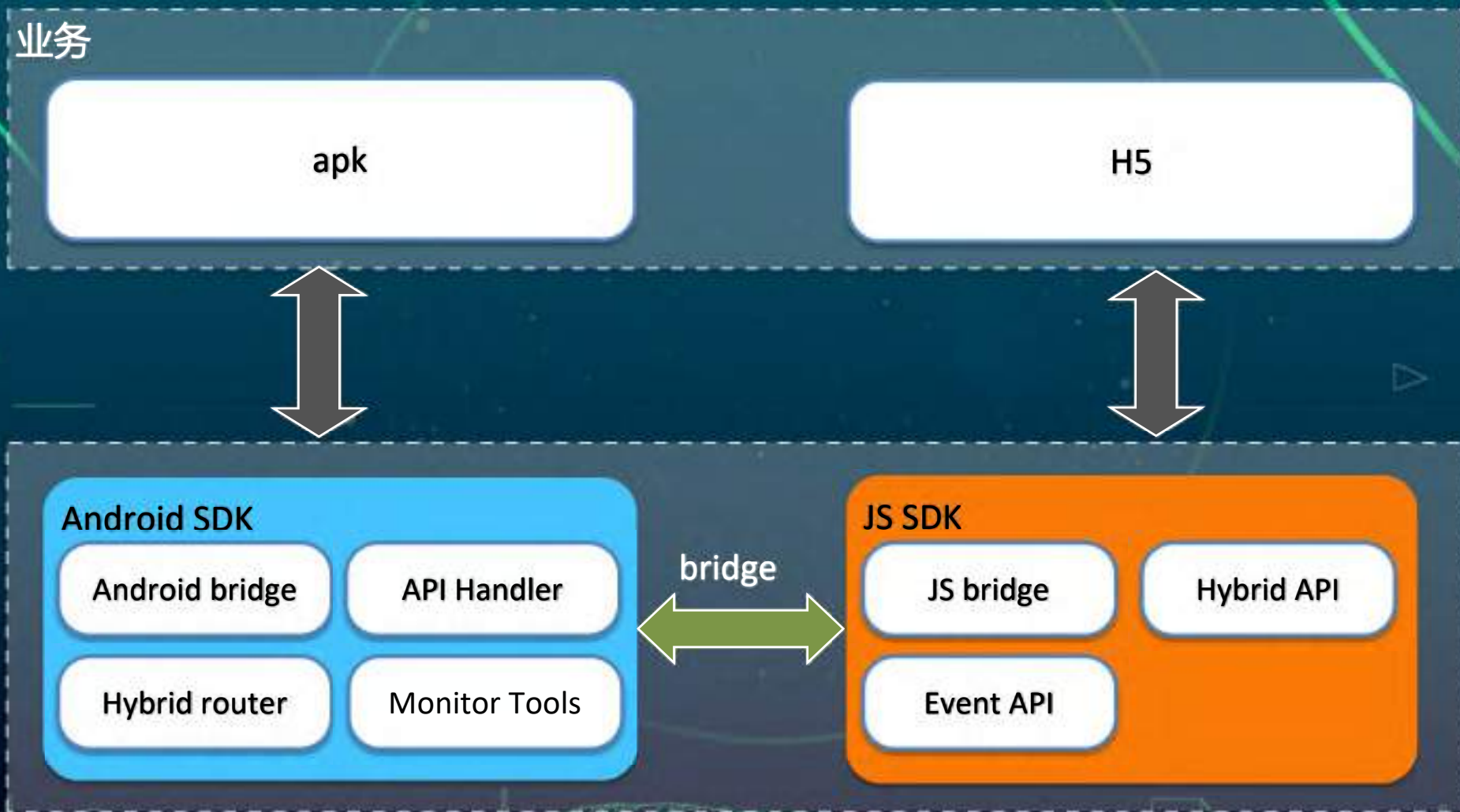


通讯——（回调）调用过程



Hybrid SDK

OSC 原创会
年终盛典 2016



模块化的API设计

OSC 原创会
年终盛典 2016

The image shows a web application interface on the left and a JavaScript object structure on the right. The web application has a list of API modules, each represented by a blue button. The modules are grouped into sections: '获取IMEI', '获取手机SN', '视图相关', '打开music.html', '打开第三方页面 (百度)', '启动账号中心app', 'UI相关', '显示actionBar', '隐藏actionBar', '设置actionBar标题', '设置actionBar副标题', 'toast demo', '显示loading', '隐藏loading', and '对话框'. The JavaScript object structure on the right is a FlymeJS object with various properties and methods, including appName, appVersion, assign, debug, desc, device, deviceName, invokeApp, invokeWeb, isInApp, notifyWeb, off, on, os, ui, util, version, view, web, wrapAPI, and __proto__.

```
Object {debug: 1, web: Object, view: Object, ...}
  appName: (...)
  ▶ get appName: function ()
  appVersion: (...)
  ▶ get appVersion: function ()
  assign: (...)
  ▶ get assign: function ()
  debug: 1
  desc: (...)
  ▶ get desc: function ()
  dev: Object
  device: Object
  deviceName: (...)
  ▶ get deviceName: function ()
  invokeApp: (...)
  ▶ get invokeApp: function ()
  invokeWeb: function invokeWebApi(api, str)
  invokeWebByObj: (...)
  ▶ get invokeWebByObj: function ()
  isInApp: (...)
  ▶ get isInApp: function ()
  notifyWeb: (...)
  ▶ get notifyWeb: function ()
  off: (...)
  ▶ get off: function ()
  on: (...)
  ▶ get on: function ()
  os: (...)
  ▶ get os: function ()
  ui: Object
  util: (...)
  ▶ get util: function ()
  version: (...)
  ▶ get version: function ()
  view: Object
  web: Object
  wrapAPI: (...)
  ▶ get wrapAPI: function ()
  __proto__: Object
```

Android嵌入SDK

H5引入flymejs

调用

H5调用Native接口

Native调用H5接口

事件

H5引入flymejs

flymejs兼容ES6

- ES6方式

```
import FlymeJS from 'flymejs'
```

- (CommonJS)

```
var FlymeJS = require('flymejs')
```

- AMD(RequireJS)

```
require(['flymejs'], function(FlymeJS) {})
```

- script标签

```
<script src="flymejs.js"></script>
```

```
<script>
```

```
</script>
```

调用

H5调用Native接口

```
// js code
// 获取手机网络状态
FlymeJS.device.getNetwork(function(v) {
    console.log(v)
})
```

Native调用H5接口

协议

扩展FlymeJS (H5)

快速开始 (Android)

Hybrid高级配置 (Android)

公共Handler (Android)

自定义handler (Android)

公共Event (Android)

自定义事件 (Android)

离线访问 (Android)

AbsHybridSourceManager类

HybridWebViewClient回调

D(RequireJS)以及原生js方式引用

```
flymejs'
```

```
flymejs.js');
```

```
function(FlymeJS) {})
```

```
</script>
```


H5作为数据接收方

OSC源创会
年终盛典 2016

未来未
来自客
的数据



其他按键

置等

状态变更、
等



反向通知——事件

OSC 原创会
年终盛典 2016

H5 listen

- keypress
- UI status change
- data change
- device status change

android notify

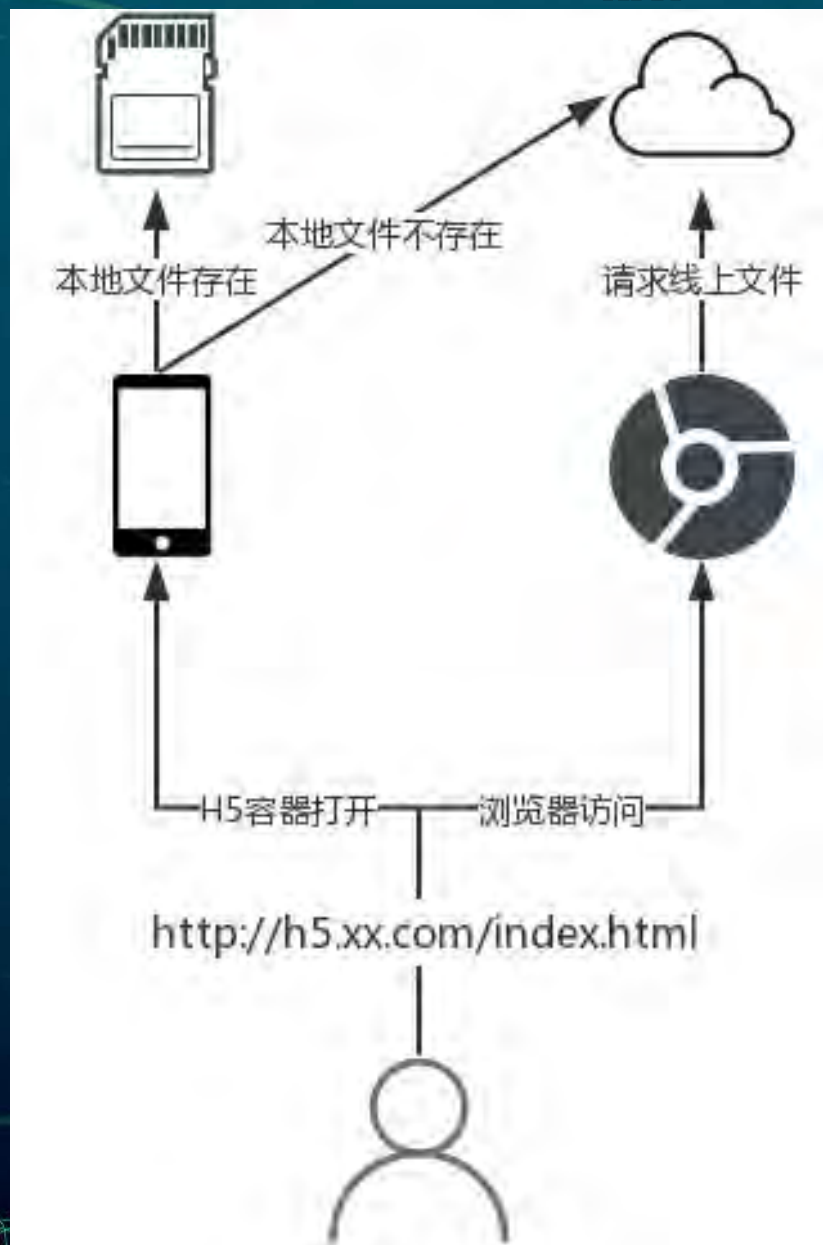
H5 callback

```
// H5拦截mback按键
FlymeJS.on('mback', function mbackHandler(data) {
    console.info(data)
})

// 客户端notify
Webview.loadUrl("javascript:
    FlymeJS.notify('com.meizu.hybrid.event.MBack',
    JSON.stringify({value: 'xxx'}))
")
```

H5资源访问

- 相同的资源使用唯一url定位
- 离线文件不存在或过期则走线上
- 基本覆盖所有H5应用场景



- H5资源文件打包路径不受约束，根据正则规则将线上url匹配离线文件
- 客户端实现路径重定向、重写等功能

```
"routes": [  
  {  
    "regex": "//h5.meizu.com/(.+)",  
    "result": "//h5.meizu.com/$1",  
    "rule": "replace"  
  },  
  {  
    "regex": "//h(\\d+).meizu.com/(.+)",  
    "result": "//h5.meizu.com/$2",  
    "rule": "override"  
  }  
]
```

几种Web缓存方案对比

方案	原理	优点	缺点
Web缓存	HTTP头	纯Web方案, 简单	伪离线 无网不可用
manifest	HTML5新接口	纯Web方案, 简单	更新有坑
离线包	请求拦截+本地替换	无网可用, 资源易于控制	需要后台、客户端等资源和人力的配合
Cache API	W3C新草案	纯web方案, 可实现更细粒度地定制资源	更新依赖Service Worker, 两者皆不成熟, webview兼容问题

- 托管H5离线包，前端只需要轻量的包管理
- CI、测试、灰度、发布一体化
- 版本管理、增量对比
- 本地资源包更新检查接口
- 静默更新（服务端主动推送）

待发包测试

OSC 原创会
年终盛典 2016

首页 / 提交测试 / 10002

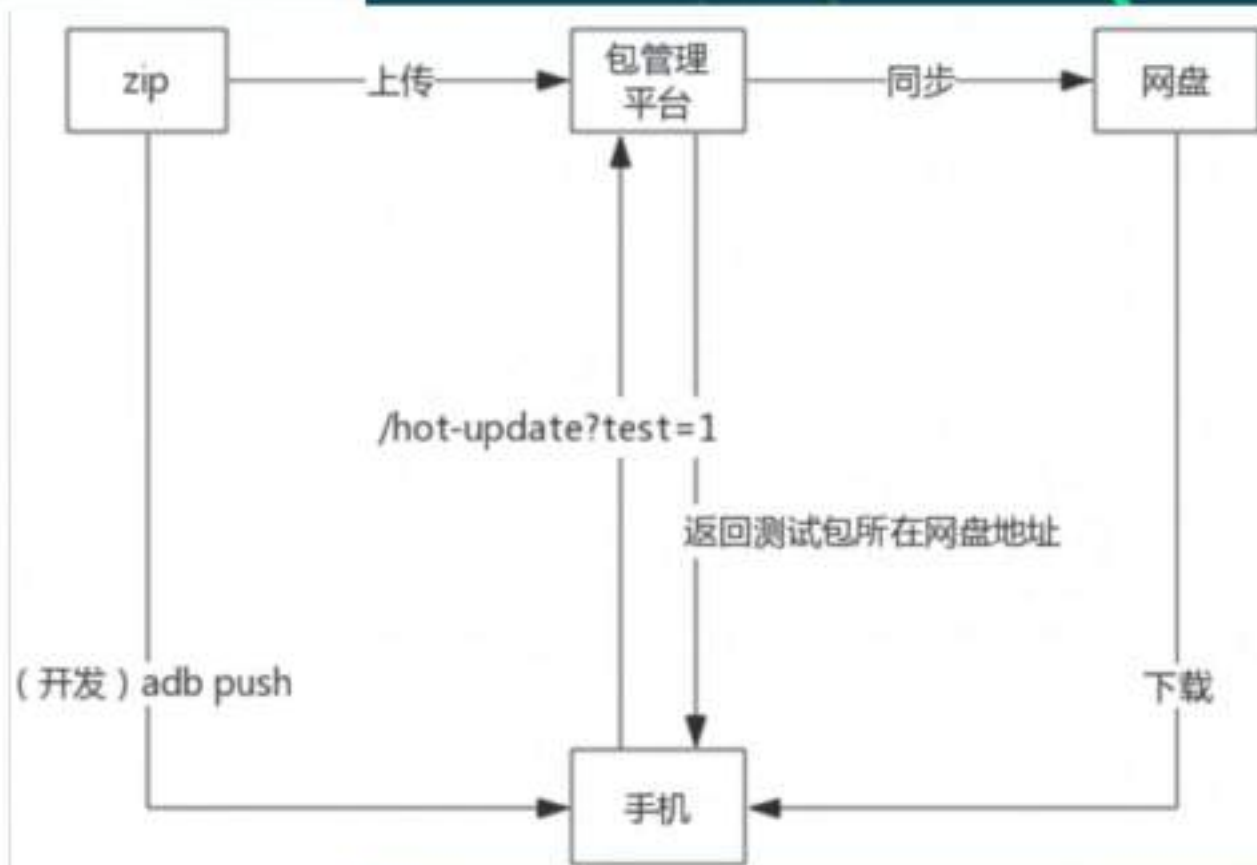
项目: 用户中心
状态: 测试中
在线版本: 1.0.2
测试版本: 1.0.2
测试包地址: [ucdemo.zip](#)

文件列表

[h5.meizu.com/uc/a.html](#)
[h5.meizu.com/uc/b.html](#)
[package.json](#)
[uc-res.meizu.com/uc/css/a.min.css](#)
[uc-res.meizu.com/uc/css/b.min.css](#)
[uc-res.meizu.com/uc/css/common.min.css](#)
[uc-res.meizu.com/uc/img/webpack.png](#)
[uc-res.meizu.com/uc/js/a.min.js](#)
[uc-res.meizu.com/uc/js/b.min.js](#)
[uc-res.meizu.com/uc/js/common.min.js](#)
[uc-res.meizu.com/uc/js/vendors.min.js](#)

↑ 重新上传包

☁ 同步到CDN



离线包分发控制

OSC源创会
年终盛典 2016

多种灰度方式，更精确地控制离线资源更新范围

显示 / 发布 / 10022

项目:	魅族健康
状态:	同步中
在线版本:	1.0.84
待发布版本:	1.0.85
待发包地址:	xc_v1.0.86.zip
待发增量包地址:	xc_patch_v1.0.84-v1.0.85.zip

灰度发布 全量发布

请选择灰度方式

百分比 IMEI白名单灰度 PUSH推送（静默更新） 选择地域

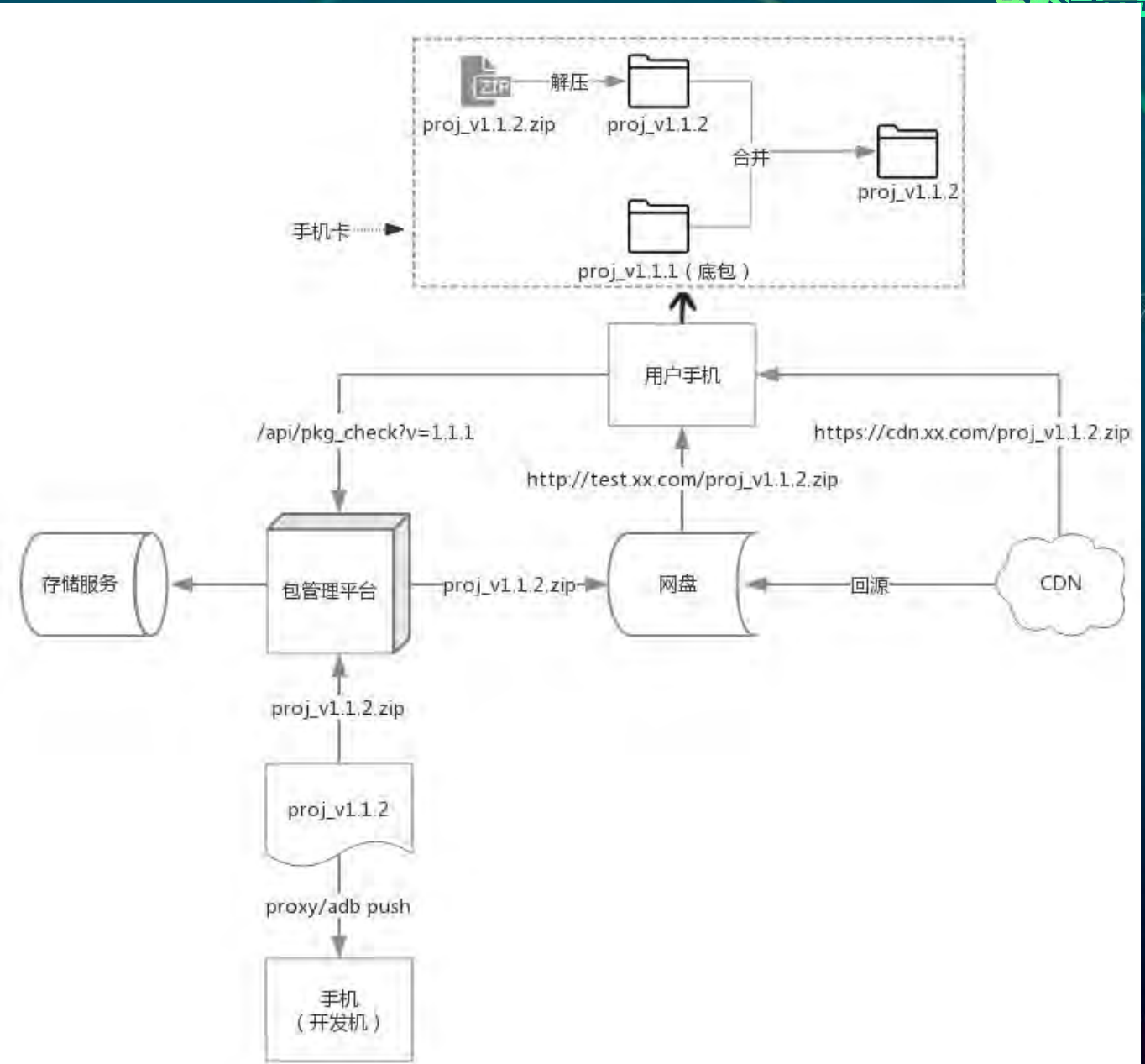
[修改项目灰度配置](#)

请务必确保zip包已经在CDN生效!

取消

确定

离线包更新流程



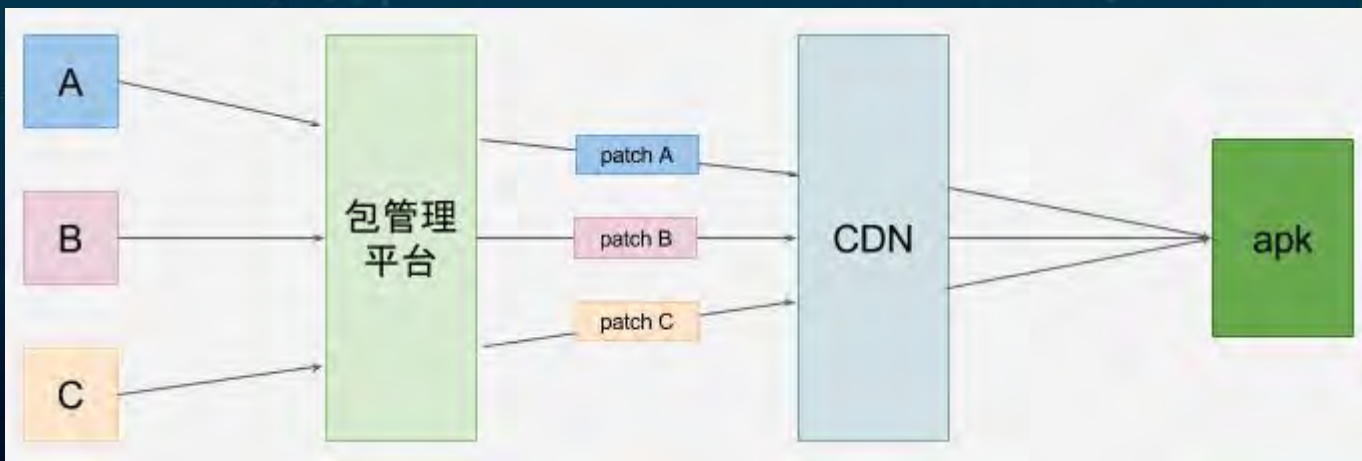
离线包瘦身——增量包

OSC 原创会
年终盛典 2016

- 上传测试包时同时自动生成增量包
- 更新时同时返回增量和全量包地址
- 增量包在客户端进行hash校验防篡改
- 文件级别的增量diff



- 同一个apk内的H5按功能分为多个模块
- 模块之间的更新频率不同
- 生成离线包时只打包改动的模块进行更新
- 在客户端合并多包



ucdemo_v1.0.2

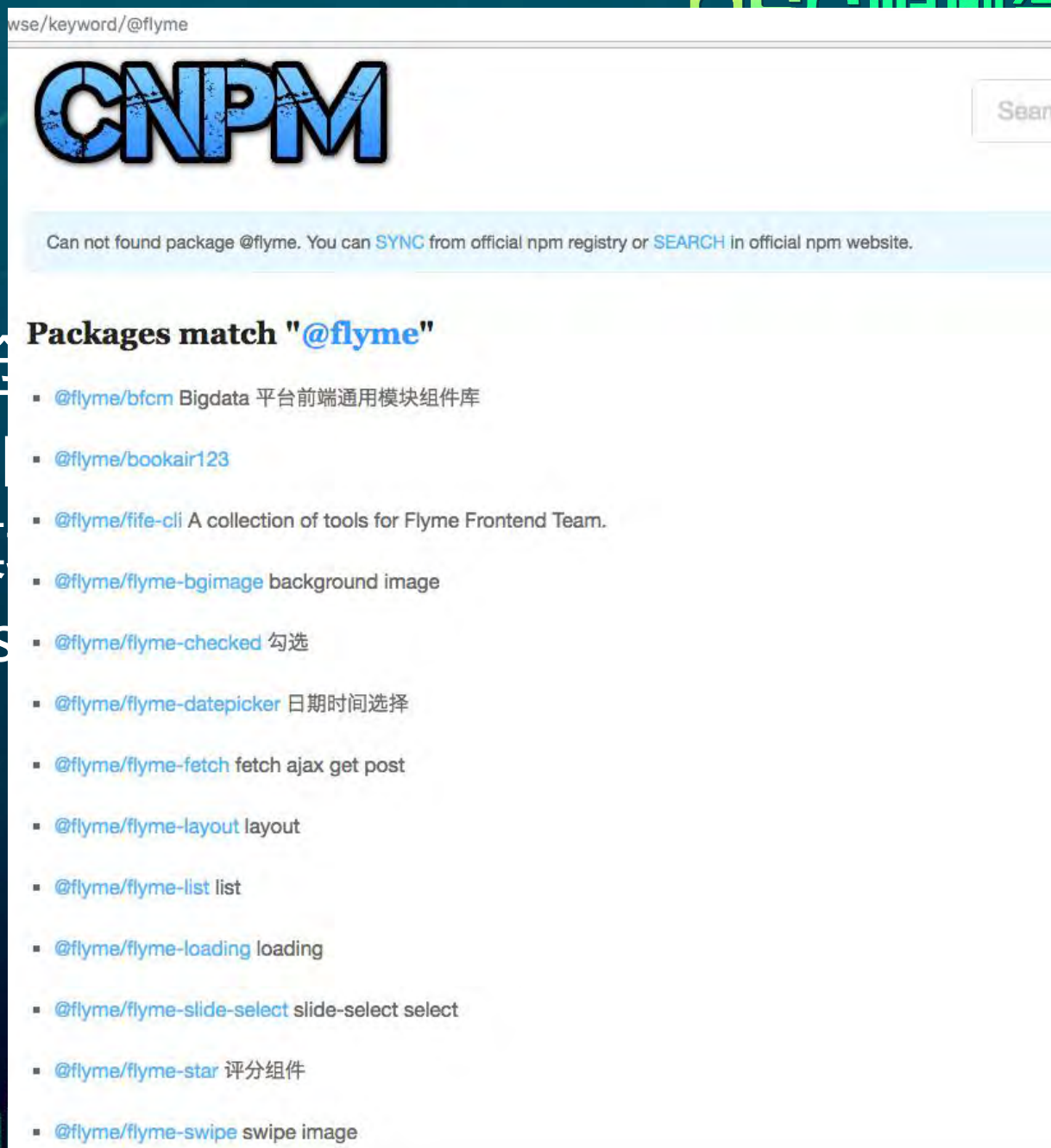
名称

- h5.meizu.com
 - uc
 - a.html
 - b.html
 - package.json
 - uc-res.meizu.com
 - uc
 - css
 - img
 - js

```
{  
  "apk": "com.meizu.usercenter", // apk名  
  "id": 10002, // 包id  
  "moduleId": 0, // apk内的模块序号  
  "version": "1.0.2", // 包版本号  
  "routes": [ // H5资源访问规则  
    {  
      "regex": "//h5.meizu.com/(.+)",  
      "result": "//h5.meizu.com/$1",  
      "rule": "replace"  
    },  
    {  
      "regex": "//h(\\d+).meizu.com/(.+)",  
      "result": "//h5.meizu.com/$2",  
      "rule": "override"  
    },  
    {  
      "regex": "//abc.meizu.com/xxx/([^.]+)?\\.jsp?(.*)",  
      "result": "//abc.meizu.com/pages/$1.html?$2",  
      "rule": "replace"  
    }  
  ]  
}
```

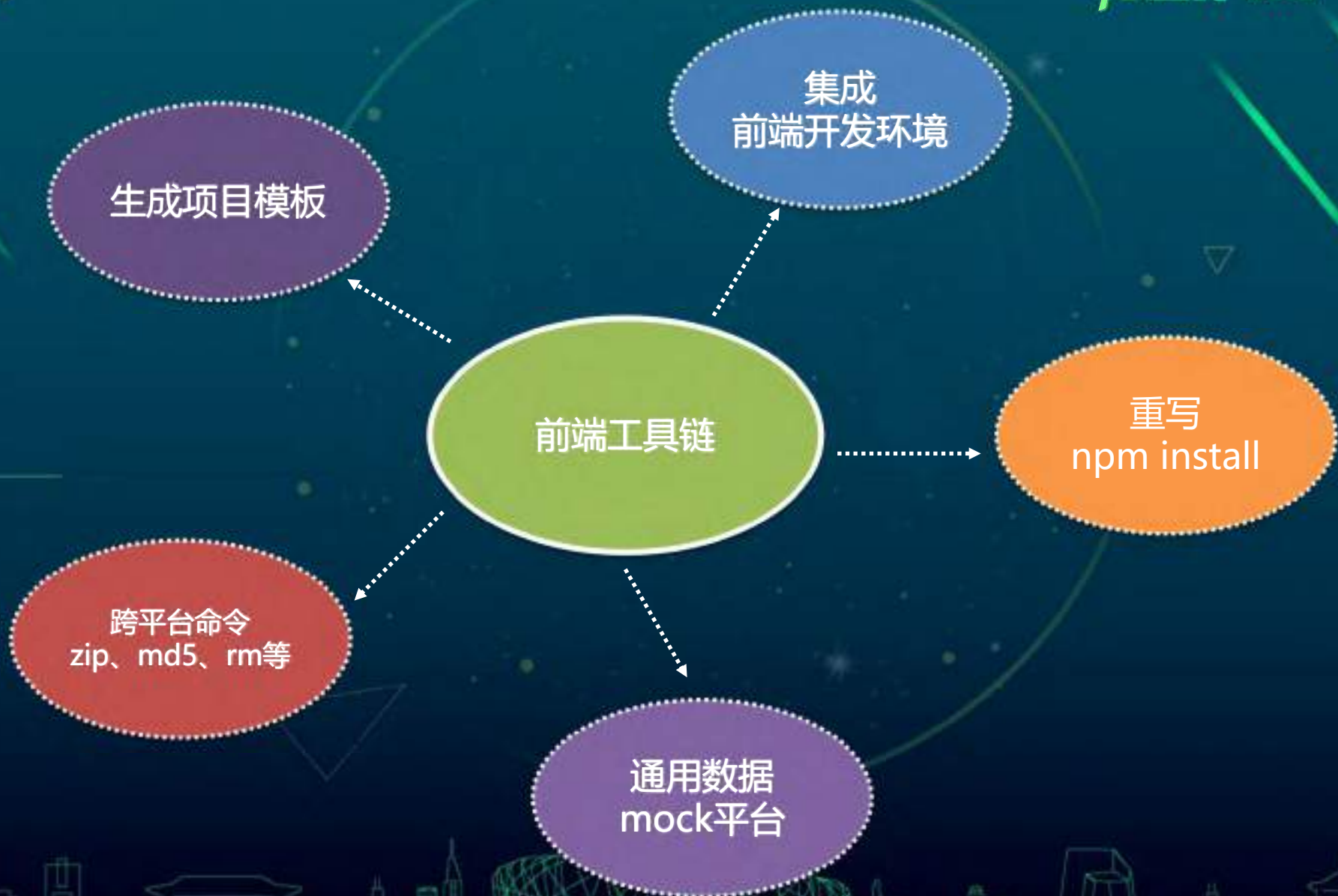

前端组件生态

- 私有npm仓库
- 基于React
- 遵循ES6模
- 组件publish



The screenshot shows the CNPM search interface. At the top, the URL is 'wse/keyword/@flyme'. The CNPM logo is prominently displayed. Below the logo, a message states: 'Can not found package @flyme. You can SYNC from official npm registry or SEARCH in official npm website.' The main section is titled 'Packages match "@flyme"' and lists several packages:

- [@flyme/bfcm](#) Bigdata 平台前端通用模块组件库
- [@flyme/bookair123](#)
- [@flyme/fife-cli](#) A collection of tools for Flyme Frontend Team.
- [@flyme/flyme-bgimage](#) background image
- [@flyme/flyme-checked](#) 勾选
- [@flyme/flyme-datepicker](#) 日期时间选择
- [@flyme/flyme-fetch](#) fetch ajax get post
- [@flyme/flyme-layout](#) layout
- [@flyme/flyme-list](#) list
- [@flyme/flyme-loading](#) loading
- [@flyme/flyme-slide-select](#) slide-select select
- [@flyme/flyme-star](#) 评分组件
- [@flyme/flyme-swipe](#) swipe image



逐渐完善的工具链

OSC 原创会
年终盛典 2016

Your Projects Starred Projects Explore Projects

Filter by name...

Last updated

New Project

V	fife / venice-api H5静态资源包更新API	★ 0
V	fife / venice-web Webapp publish platform	★ 0
H	fife / hybrid-doc http://h5.meizu.com/hybrid/	★ 0
C	fife / component-slotmachine slot machine component.	★ 0
F	fife / flymejs JavaScript layer for FlymeOS hybrid framework.	★ 3
T	fife / template-webapp 基于React.js & webpack的webapp模板	★ 0
T	fife / template-component React component generater template build with webpack.	★ 0
C	fife / component-geetest slider verification drag verification component for flyme	★ 0
F	fife / fife-cli A collection of tools for FIFE.	★ 0
F	fife / flymeui 参考客户端flymeui的web组件	★ 0
T	fife / template-flymeui 开发flymeui组件的模板	★ 0
C	fife / component-android Android与js交互公共组件封装	★ 0

Flyme Hybrid 体系

OSC 原创会
年终盛典 2016



(类) Hybrid开发方案对比

方案	原理	优点	缺点	适用范围
ReactNative/ Weex	js书写 原生代码运行	高性能、跨平台	接入门槛高	大公司、 创业团队
Hybrid	定制且易扩展的 webview	方案通用、 稳定、技术细节 容易把控	技术实现不通用 开发成本高 性能不够好	大中型团队
Ionic/Cordova	高度定制的 webview	丰富的api和完 善的工具、插件, 纯web方案	高度定制导致扩展性差, 性能差, 依赖官方后台 实现热更	个人开发者

谢谢!

