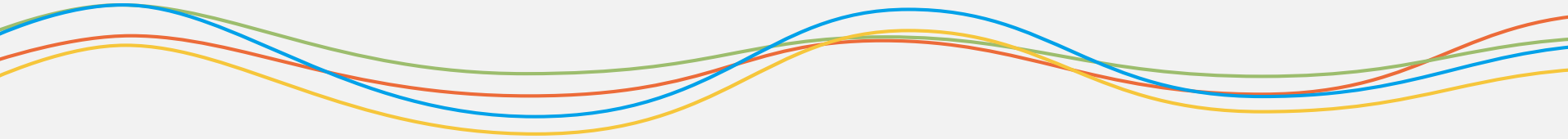
百分点 BAIFENDIAN.COM | 大数据践行者 BIG DATA PRACTITIONER

APMCon

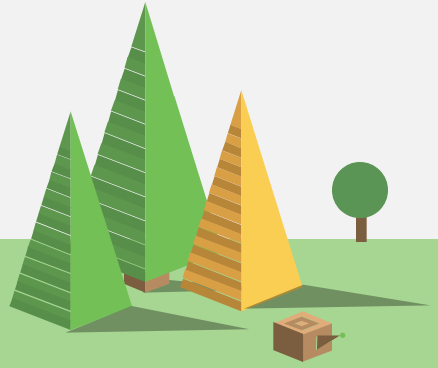
SDK无埋点技术在百分点的探索和实践

百分点集团 唐星

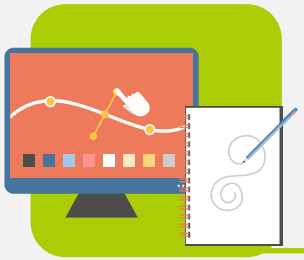


目 录

01 SDK无埋点技术简介

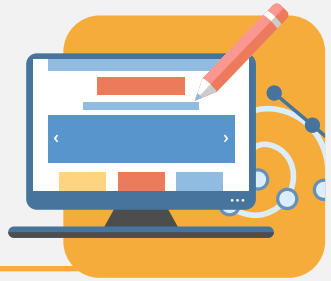


无埋点解决什么问题？



不懂代码的人也可以通过后台配置锚点并实时下发到客户端生效；

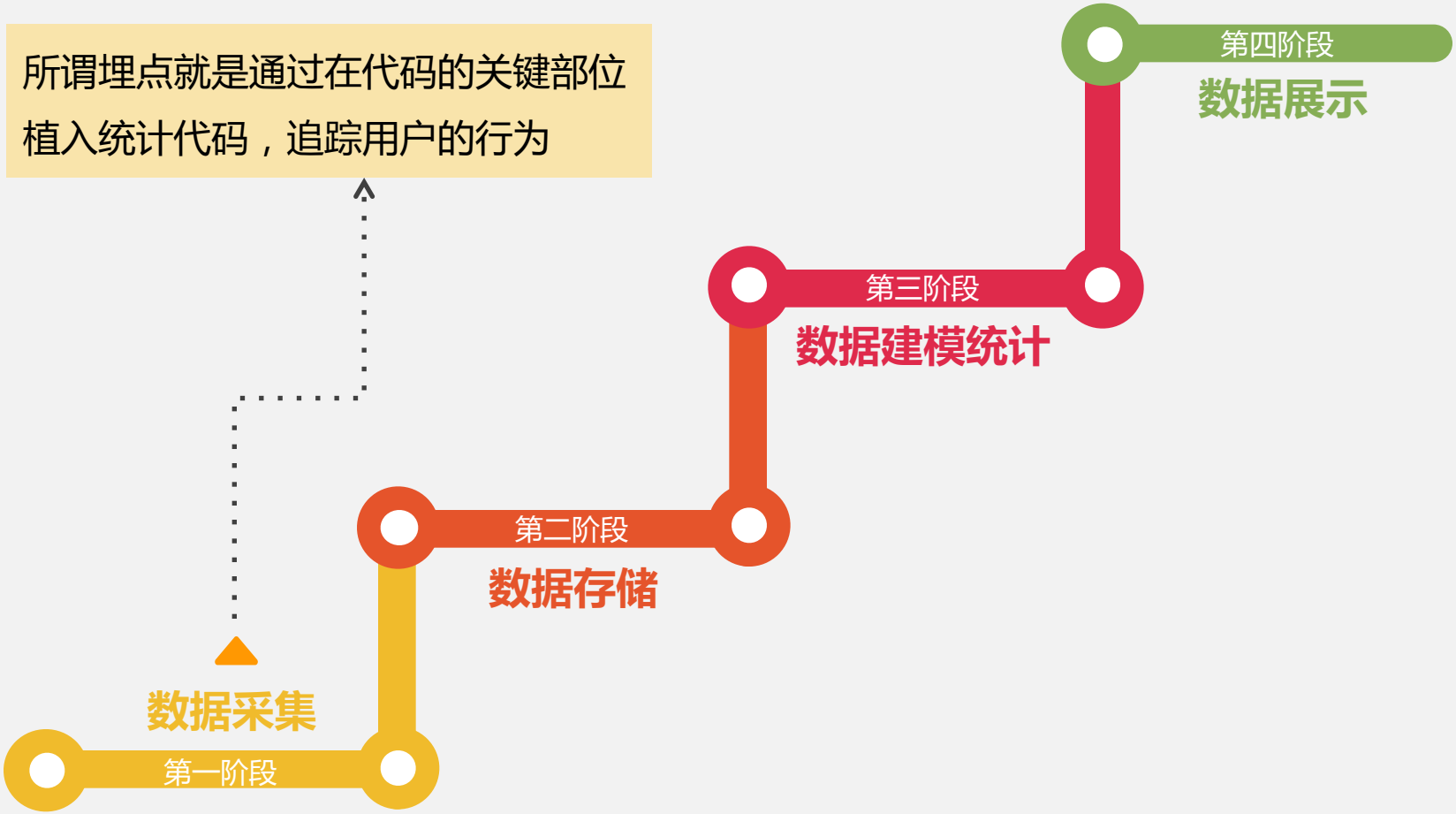
解决代码埋点本身成本过高，可视化操作，更容易上手；把核心代码和配置进行分离；



避免代码写死，需要更新版本才能生效的笨拙方式。变得更为主动灵活高效；

什么是埋点？

所谓埋点就是通过在代码的关键部位植入统计代码，追踪用户的行为



埋点长啥样？

APMCon

一段埋点代码

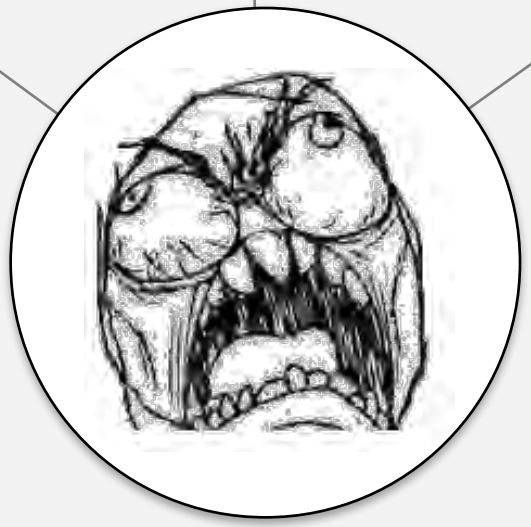
```
btnComment.setOnClickListener(new OnClickListener() {

        @Override
        public void onClick(View arg0) {
            // have params
            Map<String, Object> params = new HashMap<String, Object>();
            params.put("uid", "301358");
            params.put("iid", "1024958");
            params.put("comm", "非常好用");
            params.put("score", 10248);
            //调用sdk的api发送自定义事件
            BfdAgent.onEvent(mContext, "MComment", params);
        }
    });
```

埋点过程中的痛点

需要开发写代码、熟悉api、容易错埋漏埋；

扩展性灵活性差，增加功能就得再次埋点

修改需要重新编译打包，更新代价大



无埋点又是什么噱头

APMCon

Codeless

开发人员只需要集成
几行初始化代码即可

热更新

修改、添加方便，无需重新
编译打包，效率高，成本低

可视化

可视化埋点操作，运营
和实施人员就可以完成

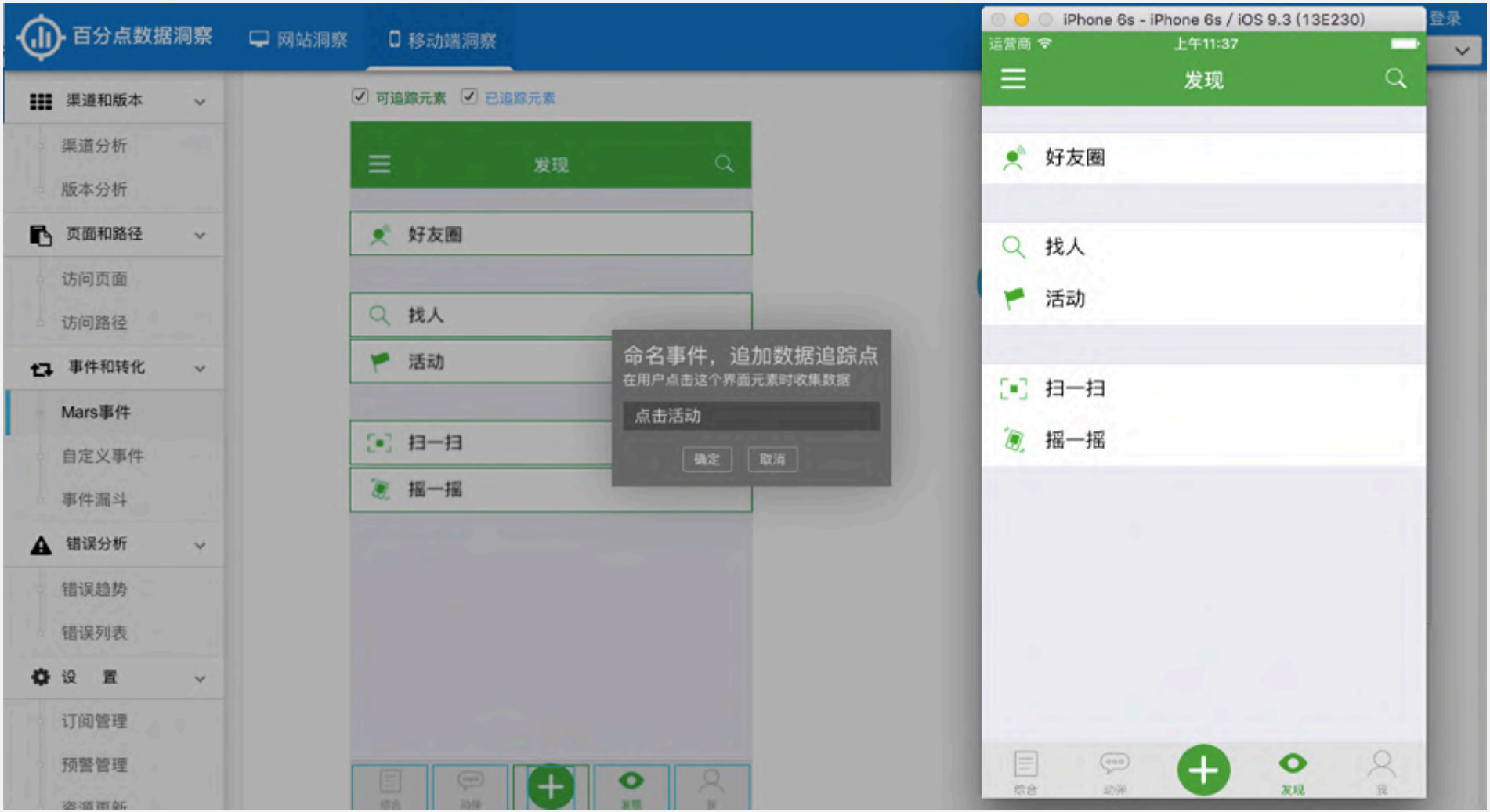
配置即代码

灵活、可扩展
埋点功能由配置控制

无埋点长啥样？


百分点数据洞察
网站洞察
移动端洞察
配置中心
欢迎您：
admin
marstest
版本分析
页面和路径
访问页面
访问路径
事件和转化
Mars事件
自定义事件
事件漏斗
错误分析
错误趋势
错误列表
设　置
订阅管理
预警管理
资源更新
Mars事件管理
1，摇一摇，连接一台设备
操作方法
1，摇一摇，连接一台设备
2，平台自动获取连接设备当前界面
3，选择界面中的元素来追加事件
4，事件生效，开始获取数据
注意事项
在设备上打开您的应用，在刚打开的前10秒内摇动设备5次
请在WIFI下使用本功能
在设备上打开您的应用
在刚打开的前10秒内 摇动设备5次
需应用后台进程关闭状态下初次打开
尝试连接中...

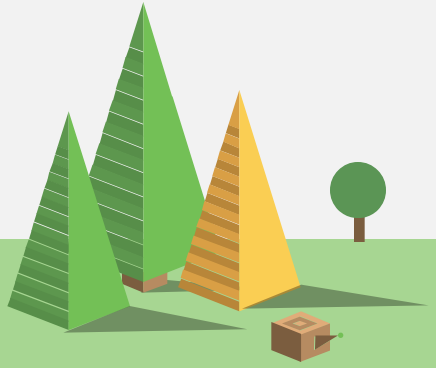
无埋点长啥样？


百分点数据洞察
网站洞察
移动端洞察
登录
渠道和版本
渠道分析
版本分析
页面和路径
访问页面
访问路径
事件和转化
Mars事件
自定义事件
事件漏斗
错误分析
错误趋势
错误列表
设　置
订阅管理
预警管理
可追踪元素
已追踪元素
发现
好友圈
找人
活动
扫一扫
摇一摇
命名事件，追加数据追踪点
在用户点击这个界面元素时收集数据
点击活动
确定
取消
iPhone 6s - iPhone 6s / iOS 9.3 (13E230)
运营商
上午11:37
发现
好友圈
找人
活动
扫一扫
摇一摇
综合
动弹
发现
我

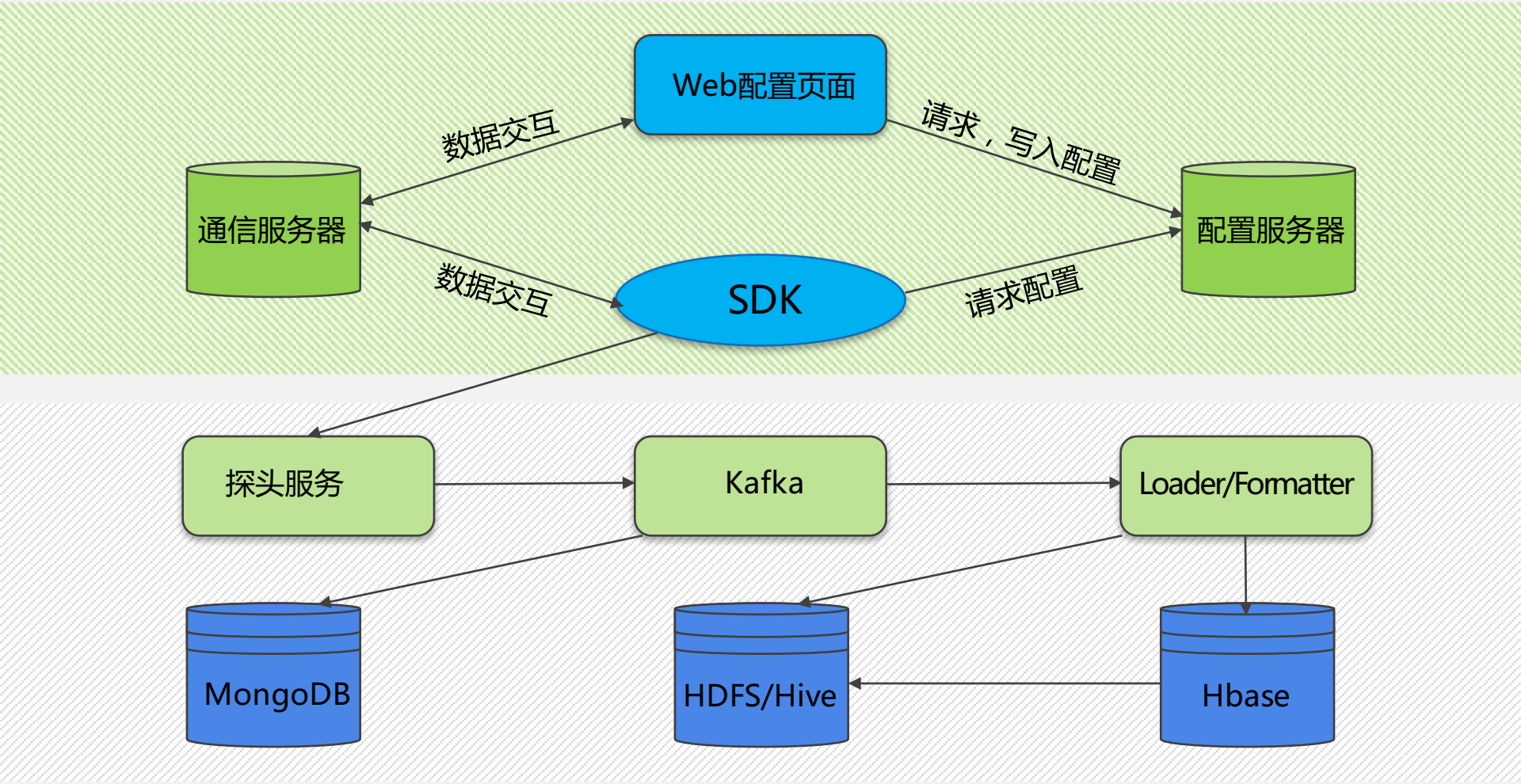
埋点vs无埋点

- 无埋点解决了埋点的部分痛点
- 无埋点是一种自然的进化，方便客户集成，操作可视化
- 无埋点也有自己的一些问题，例如无法传递一些复杂的信息，只能捕获一些简单的点击和文字变化事件行为
- 两者不是对立的，相辅相成，各有各的使用场景

02 SDK无埋点技术如何实现



整体架构图


Web配置页面
数据交互
请求，写入配置
通信服务器
配置服务器
数据交互
SDK
请求配置
探头服务
Kafka
Loader/Formatter
MongoDB
HDFS/Hive
Hbase

传输协议

基于TCP层的长连接

因为SDK和配置后台都需要和通信服务器进行双向交互。这种情况用http协议就无法满足双向交互和实时反馈的需求；而且http轮询也耗费性能；所以最终SDK跟服务端使用socket建立长连接，配置Web端跟服务器使用websocket建立长连接

传输的数据使用json格式，数据量小而且便于解析

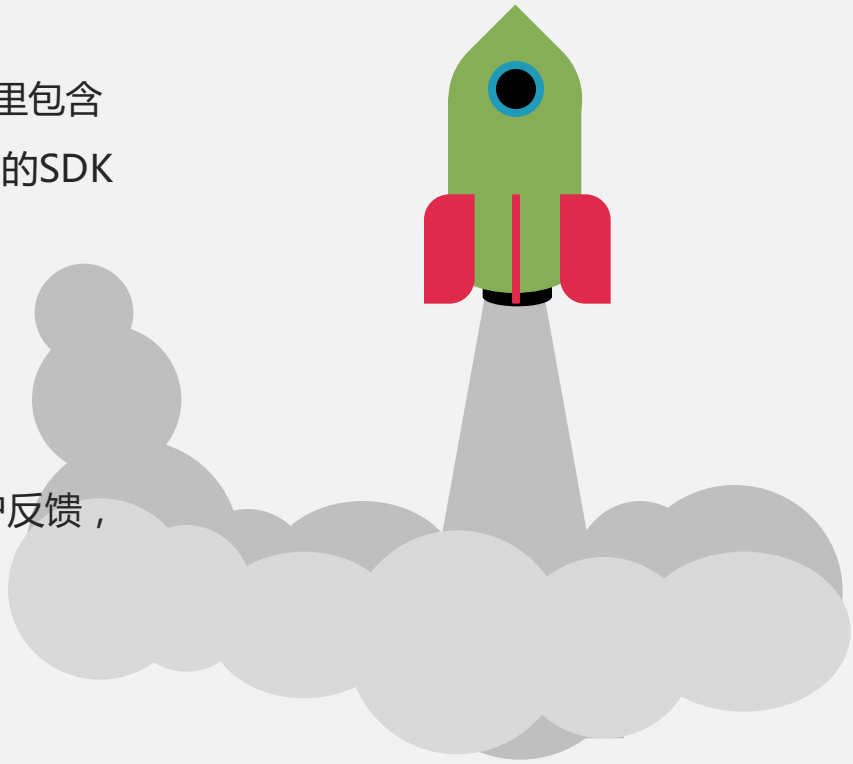
传输的过程中跟服务端约定秘钥，使用对称加密算法对内容进行加密，同时也采用压缩算法尽可能减小传输的数据量

如何触发无埋点配置

SDK是内嵌在App内，没有任何和用户交互的界面

通过第三方程序触发？

例如通过手机扫描一个二维码，然后二维码里包含
一个特殊的UrlSchemal和参数来触发app内的SDK



通过手势摇一摇触发

发通过手势摇一摇来触发，结合震动给用户反馈，
这种不需要界面交互，刚好可以满足需求
最终我们选择的也是这种方式

app界面如何传输到配置页面

APMCon

通过一些远程桌面协议例如VNC协议来进行传输？

VNC协议的优势在于传递速度快，但是协议比较复杂，而且需要在SDK内实现VNC的服务端，不太现实，舍弃这种方案

这种方案相对比较简单，易于实现。只需要把截图数据进行base64之后传递给服务端即可，主要是关注截图和传输的优化即可

SDK截取页面屏幕，每隔一段时间传递给服务端

屏幕截图的关键代码

Android关键代码：

```
View view = activity.getWindow().getDecorView();
view.setDrawingCacheEnabled(true);
view.buildDrawingCache();
if (view.getDrawingCache() == null) {
    return null;
}
Bitmap b1 = Bitmap.createBitmap(view.getDrawingCache());
```

iOS关键代码：

```
// 创建一个bitmap的context
// 并把它设置成为当前正在使用的context
UIGraphicsBeginImageContext(view.bounds.size);
CGContextRef currnetContext = UIGraphicsGetCurrentContext();
[view.layer renderInContext:currnetContext];
// 从当前context中创建一个改变大小后的图片
UIImage* image = UIGraphicsGetImageFromCurrentImageContext();
// 使当前的context出堆栈
UIGraphicsEndImageContext();
return image;
```

控件信息获取及ID生成技术

首先是安卓

1

获取时机

主要是监控所有activity的生命周期回调，在onResume里触发，主要是实现了系统的ActivityLifecycleCallbacks接口

2

获取控件的哪些信息

因为后台配置页面要在截图上绘制每个组件的坐标并框选出来，
所以需要获取控件的坐标x，y还有width和height，同时还要获取控件是否可见以及透明度等有用的信息

3

如何生成控件的唯一ID

主要是根据一个控件不变的属性组合来生成，例如控件的类名、tag，当前activity的包名，控件位于整个view tree的路径等信息组合起来然后做一个md5

控件信息获取及ID生成技术

APMCon

首先是安卓

1

获取时机

主要是监控所有activity的生命周期回调，在onResume里触发，主要是实现了系统的ActivityLifecycleCallbacks接口

```
@SuppressLint("NewApi")
public class AnalyticsActivityLifeCycle implements ActivityLifecycleCallbacks {
    @Override
    public void onActivityResumed(Activity activity) {
        // TODO 保存当前activiti对象
        AnalyticsControl control = AnalyticsControl.getInstance();
        control.mHandler.sendEmptyMessage(AnalyticsControl.NORMAL);
        control.activity = activity;
    }
```

控件信息获取及ID生成技术

APMCon

首先是安卓

2

获取控件的哪些信息

因为后台配置页面要在截图上绘制每个组件的坐标并框选出来，
所以需要获取控件的坐标x，y还有width和height，同时还要获取控件是否可见以及透明度等有用的信息

```
viewType.element_width = viewchild.getWidth();
viewType.element_height = viewchild.getHeight();
viewType.element_visible = viewchild.isShown();
viewType.element_alpha = viewchild.getAlpha();

viewType.element_path.addAll(path);
viewType.element_index.addAll(indexList);
viewType.element_path.add(path.size(), viewchild.getClass().getSimpleName());
```

控件信息获取及ID生成技术

APMCon

首先是安卓

3 **如何生成控件的唯一ID**

主要是根据一个控件不变的属性组合来生成，例如控件的类名、tag，当前activity的包名，控件位于整个view tree的路径等信息组合起来然后做一个md5

```
// 根据view的属性(id,tag,className,contentDescription)等信息产生id
private String getId(View view, List<String> pathList,
        List<Integer> indexList) {
    StringBuffer buffer = new StringBuffer();
    buffer.append(view.getId() + "|");
    buffer.append(activity.getClass().getName() + "|");
    // buffer.append(view.getTag() + "|");
    buffer.append(view.getClass().getName() + "|");
    StringBuffer newBuffer = new StringBuffer();
    for (int i = 0; i < pathList.size(); i++) {
        if (i < indexList.size()) {
            newBuffer.append(pathList.get(i) + indexList.get(i) + "|");
        } else {
            newBuffer.append(pathList.get(i) + "|");
        }
    }
    buffer.append(newBuffer.toString() + "|");
    buffer.append(view.getContentDescription() == null ? "" : view
            .getContentDescription());
    return ScreemShotUtils.stringToMD5(buffer.toString());// md5加密
}
```

控件信息获取及ID生成技术

然后是 ios

1

获取时机

主要是hook所有的viewcontroller的viewDidAppear函数，利用ios的运行时使用method swizzling技术拦截系统函数，替换为自己的函数实现

2

获取控件的哪些信息

这块和android的差不多，主要是控件的坐标x，y还有width和height，同时还要获取控件是否可见以及透明度等有用的信息

3

如何生成控件的唯一ID

主要是根据一个控件不变的属性组合来生成，例如控件的类名，当前viewcontroller的类名，控件的点击事件名selctor，控件位于整个view tree的路径等信息组合起来然后做一个md5

控件信息获取及ID生成技术

APMCon

然后是 ios

1

获取时机

主要是hook所有的viewcontroller的viewDidAppear函数，利用ios的运行时使用method swizzling技术拦截系统函数，替换为自己的函数实现

```
if (![BFDToolsForClass getMethodClass:class_getName(class) methodName:@"viewDidAppear:"]) {

    [BFDToolsForClass addClassMethodClass:class_getName(class) method:@"viewDidAppear:"withArgumentsDestypes:
        BFDClassArgumentsOrReturnValueTypeVoid withReturnValueDestypes:BFDClassArgumentsOrReturnValueTypeObjectC methodIMP:(IMP)
        replaceMethodViewDidAppearIMP];
}else{
    [BFDToolsForClass addClassMethodClass:class_getName(class) method:@"replaceDidAppear"  withArgumentsDestypes:
        BFDClassArgumentsOrReturnValueTypeVoid withReturnValueDestypes:BFDClassArgumentsOrReturnValueTypeObjectC methodIMP:(IMP)
        replaceMethodViewDidAppearIMP];
    [BFDToolsForClass replaceInstanceMethodClass:class_getName(class) method:@"replaceDidAppear" withDesClass:class_getName(class)
        method:@"viewDidAppear:" ];
}
```

控件信息获取及ID生成技术

然后是 ios

2

获取控件的哪些信息

这块和android的差不多，主要是控件的坐标x，y还有width和height，同时还要获取控件是否可见以及透明度等有用的信息

```
+(void)viewNameInfoSave:(UIView*)objc andview:(UIViewController*)vc  method:(NSString * )method  viewId:(NSString *)viewId
{
    NSMutableArray *array = [NSMutableArray arrayWithArray:[BfDPropertyData shareInstaceData].listArray];

    NSString *element_path = [NSString stringWithFormat:@"/%@/%@",[[objc class] superclass],[objc class]];
    NSMutableDictionary *mDic = [NSMutableDictionary dictionary];
    CGRect rect =[BFDToolsForView ConvertingAbsoluteRect:objc];

    [mDic setObject:viewId forKey:@"element_id"];
    [mDic setObject:STR_APDING(@"%.f", objc.alpha) forKey:@"element_alpha"];
    [mDic setValue:STR_APDING(@"%.f", rect.origin.x) forKey:@"element_x"];
    [mDic setValue:STR_APDING(@"%.f", rect.origin.y) forKey:@"element_y"];
    [mDic setValue:STR_APDING(@"%.f", rect.size.width) forKey:@"element_width"];
    [mDic setValue:STR_APDING(@"%.f", rect.size.height) forKey:@"element_height"];
    [mDic setObject:STR_APDING(@"%@", (objc.opaque||!objc.hidden)?@"YES":@"NO") forKey:@"element_visible"];

    [mDic setObject:element_path forKey:@"element_path"];
```

控件信息获取及ID生成技术

APMCon

然后是 ios

3

如何生成控件的唯一ID

主要是根据一个控件不变的属性组合来生成，例如控件的类名，
当前viewcontroller的类名，控件的点击事件名selctor，
控件位于整个view tree的路径等信息组合起来然后做一个md5

```
+(NSString *)idWithControllerName:(NSString *)controllerName viewName:(NSString *)viewName
                        EventName:(SEL)sel indexForView:(NSInteger)index {


    NSString * s=NSStringFromSelector(sel);

    NSString * ControllerId=MD5(IS_NULL(controllerName));
    NSString * viewId=MD5(STR_APDING(@"%@|%@",ControllerId, IS_NULL(viewName)));
    NSString * eventId=MD5(STR_APDING(@"%@|%@|%@|%ld",ControllerId,viewId,IS_NULL(s),(long)index));

    return eventId;
}
```

动态绑定控件事件 (Android)

只要设置了代理就可以对具备Accessibility能力的view实现点击事件的响应。View.AccessibilityDelegate具体说明请参考Android的开发者文档。

```
for (int i = 0; i < size; i++) {
    View view = list.get(i).view;
    AccessibilityDelegate delegate = getViewAccessibilityDelegate(view);
    if (delegate == null) {
    } else if (delegate instanceof TriggerAccessibilityDelegate) {
        newDelegate = (TriggerAccessibilityDelegate) delegate;
    } else {
        map.put(view, delegate);
    }
    view.setAccessibilityDelegate(newDelegate);
}
```

主要是可利用Android中View的sendAccessibilityEvent(int eventType)
方法来实现事件触发的；

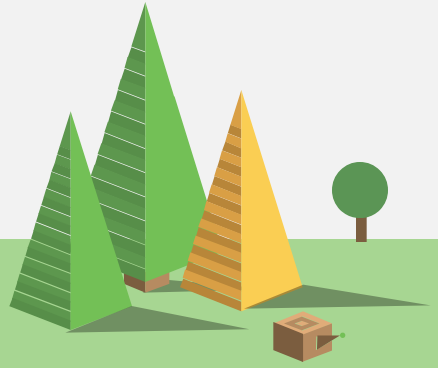
```
class TriggerAccessibilityDelegate extends View.AccessibilityDelegate {

    @Override
    public void sendAccessibilityEvent(View host, int eventType) {
        // 监听点击事件，判断是否是配置的控件
        super.sendAccessibilityEvent(host, eventType);
        if (map.containsKey(host) && map.get(host) != null) {
            View.AccessibilityDelegate delegate = map.get(host);
            if (!delegate.getClass().getName()
                    .contains("com.tendcloud.tenddata")) {
                map.get(host).sendAccessibilityEvent(host, eventType);
            }
        }
```

动态绑定控件事件(ios)

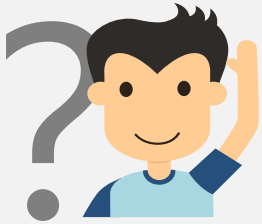
ios主要遍历当前界面所有的控件，并根据配置信息筛选需要绑定的控件，然后通过method swizzling 的技术替换控件的点击事件来达到绑定控件的事件；

```
NSMutableString * methodTmp=[NSMutableString stringWithFormat:@"replaceMethodIMPbtnOrgestrureEvent%@",method];


methodName =[methodTmp stringByReplacingOccurrencesOfString:@":" withString:@""];

Method method_m = getMethodFormInstanceWithMethodName(object_getClassName(target), method);

NSArray * array =[BFDToolsForClass getMethodArgumentType:method_m];

BOOL isGesture= [BFDToolsForClass getMethodClass:object_getClassName(target) methodName:methodName];
Method method_s =  class_getClassMethod([target class], NSSelectorFromString(methodName));

[[NSUserDefaults standardUserDefaults] setValue:@"Value" forKey:[BFDUnit BFDMD5HashStr:@"replaceMethod"]];

if (!isGesture&&![BFDToolsForClass getMethodClass:object_getClassName(target) methodName:methodName]&&method_s==nil) {

    if (array.count==3) {
        [BFDToolsForClass addClassMethodClass:(char *) object_getClassName(target) method:methodName  withArgumentsDestypes:
            BFDClassArgumentsOrReturnValueTypeVoid withReturnValueDestypes:BFDClassArgumentsOrReturnValueTypeObjectC methodIMP:(IMP)
            replaceMethodIMPArgumentOnebtnEvent];

        [BFDToolsForClass replaceInstanceMethodClass:(char *) object_getClassName(target) method:methodName withDesClass:(char *)
            object_getClassName(target) method:method];
    }else if(array.count==4){
        [BFDToolsForClass addClassMethodClass:(char *) object_getClassName(target) method:methodName  withArgumentsDestypes:
            BFDClassArgumentsOrReturnValueTypeVoid withReturnValueDestypes:BFDClassArgumentsOrReturnValueTypeObjectC methodIMP:(IMP)
            replaceMethodIMPArgumentTwobtnEvent];

        [BFDToolsForClass replaceInstanceMethodClass:(char *) object_getClassName(target) method:methodName withDesClass:(char *)
            object_getClassName(target) method:method];
    }
}
```

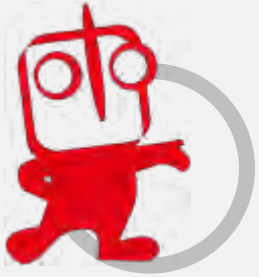
03 实践中遇到的坑和解决方法



长连接断开的问题



问题：链路如果长时间没有数据通信就会断开
并且服务端和客户端也没有及时的回调通知



解决方法：这是使用长连接经常遇到的一个问题，处理方法就是要有一个简单的心跳机制，来保活链路，防止因长时间没有数据流动被防火墙或者网关杀掉

摇一摇遇到的问题

问题：摇一摇触发成功如何给用户反馈？以及用户有时候运动中的摇一摇误触发怎么解决？

解决方法：摇一摇触发成功连上服务器后可以用震动的形式给用户反馈，以反馈给用户，用户就不用一直摇来摇去；
防止误触发主要是通过服务端返回一个消息通知成功后才给用户以震动的提示，然后开始传输截屏和控件的信息到服务器；这样就不会走着走着手机突然就震动了；

问题：ios如何能在所有界面都能监听的摇一摇事件

解决方法：这个主要是利用ios的运行时的一些技术，拦截到系统的rootViewController，然后动态的给注入一个摇一摇的监听方法，这样就能在所有的界面监听摇一摇的事件了。

界面传输效率性能的优化

1. 截图使用jpeg格式进行最大的压缩比，然后图片质量选择0.6这样既不影响清晰度，也能够最大限度的减少数据传输量，进而提高传输的效率

2. 屏幕截图的传输频率，默认是每隔5秒传输一次，但这样的话手机操作时无法实时传送到服务端以及web管理段，这给用户一种延时的感觉，这块我们主要是针对界面切换和列表以及滚动这些事件做了主动触发，这样用户就能感觉到管理页面跟手机几乎是同步的

3. 有些时候屏幕界面是没有变化，所以这时候发送截图没有必要，所以我们队截图做一个md5保存，然后下次发送屏幕截图时先对比一下两次截图的md5值是否一样，如果一样就不发送截图

控件ID重复导致错误和重复发送

问题：通用的ID生成规则针对某些特殊情况的控件会产生相同的ID，例如同样类名的button，并且位于相同的父view上，而且程序逻辑控制每次动态add其中一个，这样他们的view tree上的path也一样；导致生成的ID重复；进而影响界面绘制和事件统计

解决方案：因为SDK能拿到的控件信息有限，所以当这些通用规则都失效的情况下，只能通过服务端对这些控件进行特殊的配置，进行定制，例如button里的text可能不一样，所以可以用这项信息添进去

SDK技术发展趋势

自动化

尽量让第三方app集成起来方便，尽量做到让第三方开发者调用最少的代码就能集成SDK，如果能够做到全自动化集成是最好的；

可视化

尽量把一些简单和重复的操作可视化，这样可以把开发者解放出来交给运营人员或者实施人员来处理业务逻辑；

动态更新

SDK的更新成本还是挺高的，由于移动端开发的一些限制问题，app的动态更新一直是个难题尤其是ios下，目前只能通过脚本语言实现一些多变模块的动态更新，SDK也要朝着这个方向尽量做到核心模块稳定，其他模块慢慢的可配置化和脚本语言化。

THANK YOU