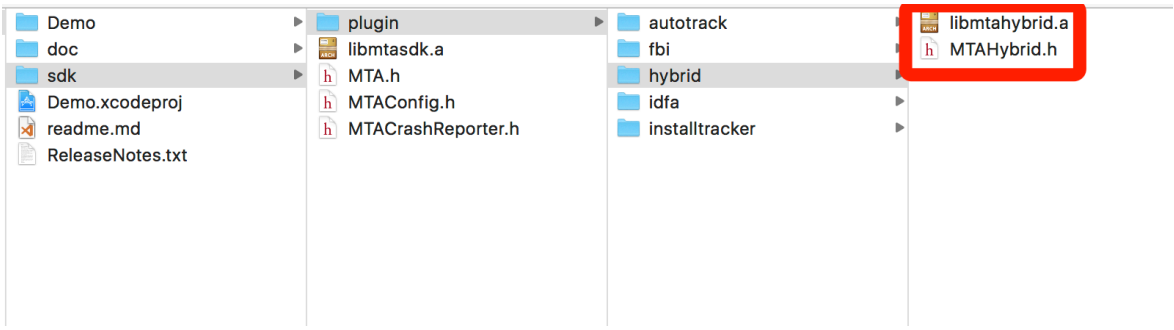
MTA APP H5 混合统计使用说明

使用了混合统计功能以后，在 APP 内加载的 html 页面也能通过 Native 的方式上报页面访问事件和自定义kv事件。使用混合统计功能时，需要在 iOS，Android 以及 HTML 端同时接入对应的 SDK。

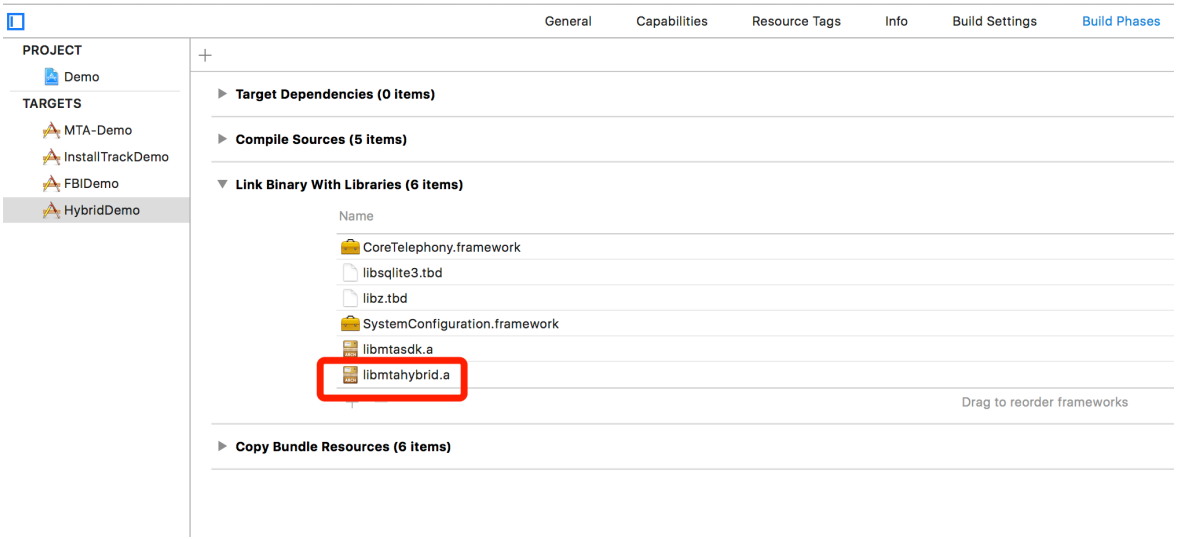
接入方法

Native 端(iOS)

- 参考MTA接入文档接入MTA。
- 在 MTA iOS SDK 包中的 sdk/plugin/hybrid 目录下找到 libmtahybrid.a 静态库和 MTAHybrid.h 头文件。



- 将静态库文件连接至工程中。



添加 Native 代码 (具体例子可以参考demo)

UIWebView

- 在 UIWebView 的 delegate 中添加以下代码

```
- (BOOL)webView:(UIWebView *)webView
    shouldStartLoadWithRequest:(NSURLRequest *)request
    navigationType:(UIWebViewNavigationType)navigationType {
    // 处理MTA混合统计请求的代码
    if ([MTAHybrid handleRequest:request
        fromWebView:webView]) {
        return NO;
    }

    // 原有的代码
    return YES;
}
```

- 在 UIWebView 被隐藏，或者从父 view 中移除时，调用

```
+ (void)stopWebView:(UIWebView *)webView;
```

方法

- 在 UIWebView 重新被显示，或者重新添加到父 view 上时， 调用

```
+ (void)restartWebView:(UIWebView *)webView;
```

方法

WKWebView

- 在 WKWebView 的 navigationDelegate 中添加以下代码

```
- (void)webView:(WKWebView *)webView
    decidePolicyForNavigationAction:(WKNavigationAction *)navigationAction
    decisionHandler:(void (^)(WKNavigationActionPolicy))decisionHandler {
    // 处理MTA混合统计请求的代码
    if ([MTAHybrid handleAction:navigationAction
        fromWKWebView:webView]) {
        decisionHandler(WKNavigationActionPolicyCancel);
        return;
    }

    // 原有的代码
    decisionHandler(WKNavigationActionPolicyAllow);
}
```

- 在 WKWebView 被隐藏，或者从父 view 中移除时，调用

```
+ (void)stopWKWebView:(WKWebView *)wkWebView;
```

方法

- 在 WKWebView 重新被显示，或者重新添加到父 view 上时， 调用

```
+ (void)restartWKWebView:(WKWebView *)wkWebView;
```

方法

Native 端(Android)

Hybrid 统计是在原生统计基础上进行的，在开始之前，请确保已按照MTA官网，正常接入MTA Android SDK配置和初始化流程。

初始化Hybrid模块

在Application或MainActivity的onCreate初始化MTA基础统计接口后，需要额外调用以下接口，初始化Hybrid模块，开发者可根据是否使用与原生App一致的Appkey来决定灵活使用哪个初始化接口。

```
/**
* 初始化Hybrid模块，默认使用原生App的appkey、渠道等配置信息
* @param context 上下文对象
*/
public static void init(Context context);

/**
* 初始化Hybrid模块，使用指定的appkey和渠道信息
* @param context 上下文对象
* @param appkey Hybrid统计所使用的Appkey
* @param channel Hybrid统计所使用的渠道
*/
public static void init(Context context, String appkey, String channel);
```

示例

```
public class MyApp extends Application{
    public static Application application = null;
    @Override
    public void onCreate() {
        super.onCreate();
        // 其它代码

        // 使用默认Appkey初始化Hybrid模块
        StatHybridHandler.init(this);
    }
}
```

配置WebView

在需要使用Hybrid统计的WebView组件，调用以下方法进行初始化。

初始化WebSettings

```
/**
* 初始化WebSettings
* @param webSettings 待初始化的webSettings
*/
public static void initWebSettings(WebSettings webSettings)
```

示例

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    webView = (WebView) findViewById(R.id.webview);
    webSettings.setJavaScriptEnabled(true);
    StatHybridHandler.initWebSettings(webSettings);
    webView.setWebViewClient(new MyWebViewClient());
}
```

配置WebViewClient

Natiive使用拦截MTA专用的url跳转方式与H5交互，因此，需在WebViewClient的shouldOverrideUrlLoading方法调用SDK接口，进行url拦截。

```
/**
* 拦截MTA专用url跳转
* @param webView 当前WebView
* @param url 当前url
*/
public static void handleWebViewUrl(WebView webView, String url);
```

示例

```
public class MyWebViewClient extends WebViewClient {
    @Override
    public boolean shouldOverrideUrlLoading(WebView view, String url) {
        super.shouldOverrideUrlLoading(view, url);
        if(StatHybridHandler.handleWebViewUrl(view, url)){
            return true;
        }
        return true;
    }
}
```

HTML 端

页面统计

需要统计app webview的基础访问、点击事件时，请在webview里加入以下js 代码

```
<script src="//pingjs.qq.com/mta/app_link_h5_stats.js" name="MtaLinkH5">
</script>
```

- 注意：
  - 后续的方法上报都必须保证已加载以上js sdk

手动上报页面访问统计

访问页面时，上报页面访问情况

```
MtaLinkH5.pageBasicStats({
  'title': '必填-每页要求不重复'
});
```

- 注意：
  - 确定联动分析js sdk已载入，并且设置了title名称
  - title为必填项目，并且每页的title都要求不重复，重复影响统计

设置登陆帐号

用于设置用户登陆帐号信息

```
MtaLinkH5.setLoginUin(uin);
```

- 注意：
  - 确定联动分析js sdk已载入
  - uin 为设置的用户帐号，string类型

自定义事件

用于页面自定义事件埋点上报

```
MtaLinkH5.eventStats(event_id, param_json);
```

- 注意：
  - 确定联动分析js sdk已载入
  - event_id 为事件id,在事件中添加后拷贝过来
  - param_json 为事件参数以及事件参数值，每个参数对应一个参数值，为json格式

例子：

```
<button onclick="MtaLinkH5.eventStats('test_event')">事件-不带参数</button>
<button onclick="MtaLinkH5.eventStats('test_event', {'param1':'value1'})">事件-单个参数</button>
<button onclick="MtaLinkH5.eventStats('test_event', {'param1':'value1','param2':'value2'})">事件-多个参数-参数建议最多不超过5个</button>
```