蚂蚁科技

移动分析
使用指南

文档版本：20250731

蚂蚁集团
ANT GROUP

法律声明

蚂蚁集团版权所有©2022，并保留一切权利。

未经蚂蚁集团事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

商标声明

蚂蚁集团 ANT GROUP及其他蚂蚁集团相关的商标均为蚂蚁集团所有。本文档涉及的第三方的注册商标，依法由权利人所有。

免责声明

由于产品版本升级、调整或其他原因，本文档内容有可能变更。蚂蚁集团保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在蚂蚁集团授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过蚂蚁集团授权渠道下载、获取最新版的用户文档。如因文档使用不当造成的直接或间接损失，本公司不承担任何责任。

通用约定

格式	说明	样例
危险	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	危险 重置操作将丢失用户配置数据。
警告	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	警告 重启操作将导致业务中断，恢复业务时间约十分钟。
注意	用于警示信息、补充说明等，是用户必须了解的内容。	注意 权重设置为0，该服务器不会再接受新请求。
说明	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	说明 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击**设置**> **网络**> **设置网络类型**。
粗体	表示按键、菜单、页面名称等UI元素。	在**结果确认**页面，单击**确定**。
`Courier字体`	命令或代码。	执行 `cd /d C:/window` 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	`bae log list --instanceid` *Instance_ID*
[] 或者 [a|b]	表示可选项，至多选择一个。	`ipconfig [-all|-t]`
{} 或者 {a|b}	表示必选项，至多选择一个。	`switch {active|stand}`

目录

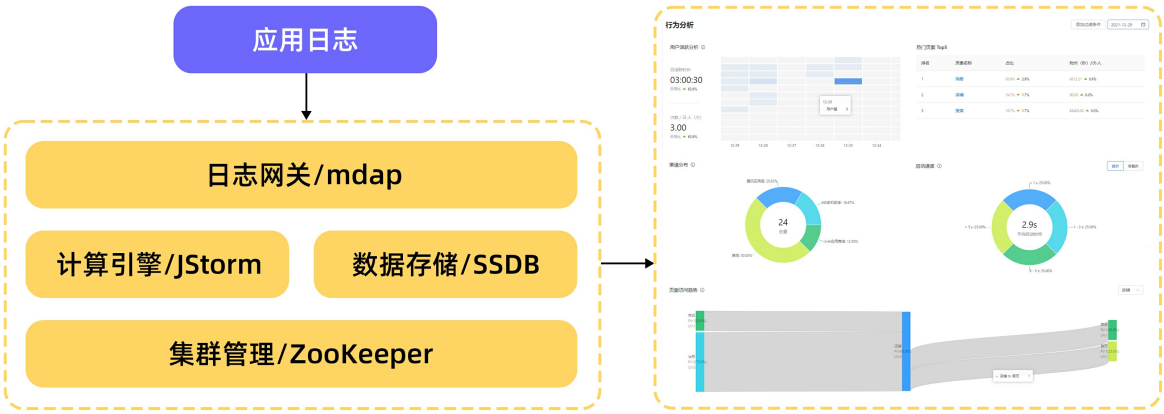
1.移动分析简介

移动分析服务（Mobile Analysis Service，简称 MAS）是 mPaaS 平台的一个核心基础服务组件，是一项方便您轻松地大规模收集、可视化并理解应用程序使用数据的服务。MAS 通过统计和分析客户端流量、性能质量和用户行为等数据，用数据实现产品、运营、推广的决策；通过对闪退的分析，快速定位闪退原因，提高程序稳定性。

组件原理

MAS 涉及的相关组件介绍如下：

- **mdap**：日志采集网关，负责收集客户端埋点日志，收到日志后，直接传输至 JStorm 集群进行计算。
- **JStorm**：实时计算引擎，根据处理规则对日志进行实时解析并将需要的数据存储入库。
- **SSDB**：KV 数据存储层，底层使用 Leveldb，支持单表十亿级记录。
- **ZooKeeper**：集群管理、组件间服务发现。



组件特点

- **极简接入**：引入移动分析组件便可自动搜集用户行为日志、网络日志、异常日志，简单方便。
- **全面分析**：具备用户行为、终端问题、流量、电量、通讯链路、性能等多个目标的分析角度。
- **多维度展现**：可以从终端类型、终端版本、地域、网络类型、厂商机型等多个维度展现和分析移动应用数据。
- **快速定位问题**：闪退、异常日志提供发生错误的接口名称、异常原因、运行环境等信息，帮助开发者快速定位问题。

组件功能

- **用户行为分析**：提供应用使用分析，包括用户报活、用户登录、新增用户等多种指标的统计功能，并支持按照平台、版本、地域、时间的多维度分析对比，方便用户更快速、便捷的了解自身 App 的使用情况。
- **稳定性分析**：提供应用稳定性分析，包括闪退监控、异常监控、性能监控及用户诊断功能，帮助开发人员及时发现、定位问题。
- **问题诊断**：提供应用问题诊断，包括个人用户诊断和诊断日志采集两部分。其中个人用户诊断实时获取用户客户端行为，诊断日志采集通过 push 方式下发指令到客户端传回客户端本地日志。

应用场景

- **通过数据指导业务**：帮助开发、运营人员利用数据进行产品、运营、推广方案的决策。
- **提升用户体验**：快速定位闪退位置，结合热修复功能快速修正 APP 闪退，提升用户体验、增加客户留存率。

2.基本概念

本术语表按拼音首字母对术语进行排序。

埋点

埋点指针对特定用户行为或事件进行捕获、处理和上报的相关技术及其实施过程。在应用中收集一些信息，用来跟踪应用使用的状况，后续用来进一步优化产品或是为运营提供数据支撑，包括访问数、访客数、点击数、停留时长等等。

事件

事件用于记录用户在 App 内的一个动作。您可以在任意动作（如按钮点击）触发时，埋入一个自定义事件。

事件 ID

用于唯一标识一个事件。事件是 App 全局的，因此，同一个 mPaaS 应用中事件 ID 必须唯一。

属性

一个事件包含一些信息，如触发事件的用户 ID、App 版本、设备型号等，这些信息即为属性。移动分析平台预置了一些常用属性；此外，您可以根据实际情况自定义属性。

属性 ID

用于唯一标识一个属性。属性是 App 全局的，因此，同一个 mPaaS 应用中属性 ID 必须唯一。

事件分析

事件及其属性信息会以日志的形式先存储在本地客户端，然后上报至移动分析服务器。在控制台完成相关配置和操作后，您可以查看事件分析报表。

3.快速开始

本文介绍移动分析的使用流程，以及如何启动运行 App 并查看运行后生成的分析报表。

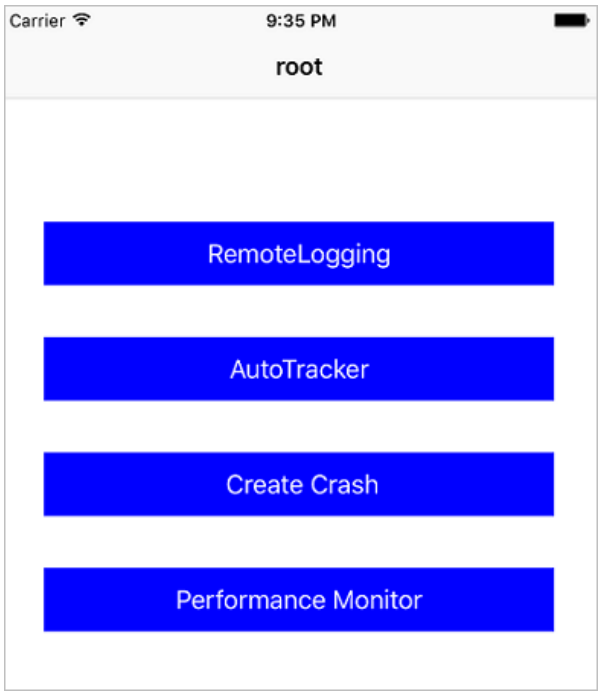
使用流程

1. 在 mPaaS 控制台中创建您的应用。
2. 客户端应用使用 mPaaS 中的埋点 SDK 进行埋点，请参考文档 接入 Android SDK 或 接入 iOS SDK。
3. 在 mPaaS 控制台中，点击左侧导航栏中的 **移动分析** 查看统计数据。
4. 如果 iOS App 需要调试符号反解服务，则需要通过 mPaaS 控制台的 **移动分析** > **性能分析** > **iOS 符号表管理** 页面上传 dSYM 符号表文件。上传符号表文件后，符号反解会在闪退日志上传后实时进行（延迟在分钟级）。如不上传符号表，则只展示原始闪退日志内容。

启动运行 App

直接运行 App。在 App 中点击按钮、切换页面、操作发生闪退时，都会产生一条埋点日志并同步上传至客户端上报日志的服务器。这样，可在移动分析控制台中查看埋点数据的分析报表。

示例如下：



查看分析报表

要查看分析报表，您需要先登录移动分析控制台：

1. 登录 mPaaS 控制台，并选择应用。
2. 在左侧导航栏中单击 **移动分析** 菜单，可看到基础分析、性能分析、组件使用分析等功能菜单。

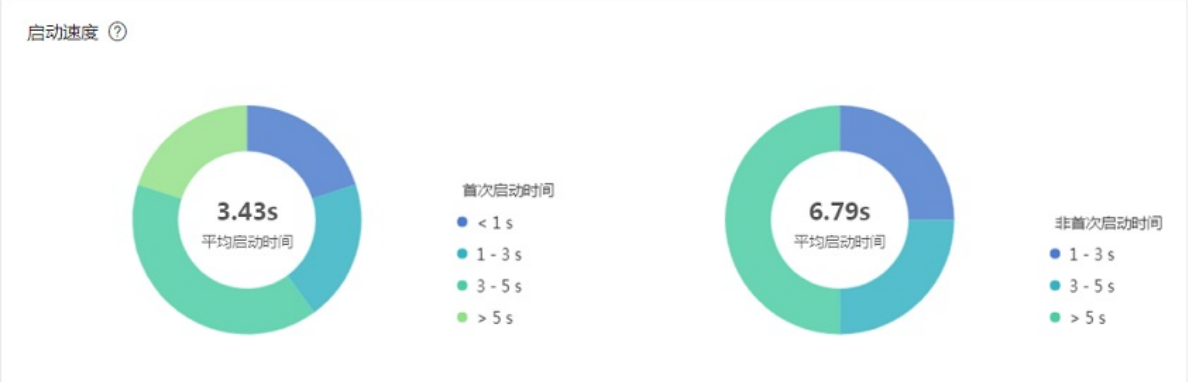
移动分析报表示例如下：

- 应用启动次数、活跃用户数、活跃账号数

  可在 **数据概览** > **实时大盘** 中查看应用启动次数、活跃用户数、活跃账号数。

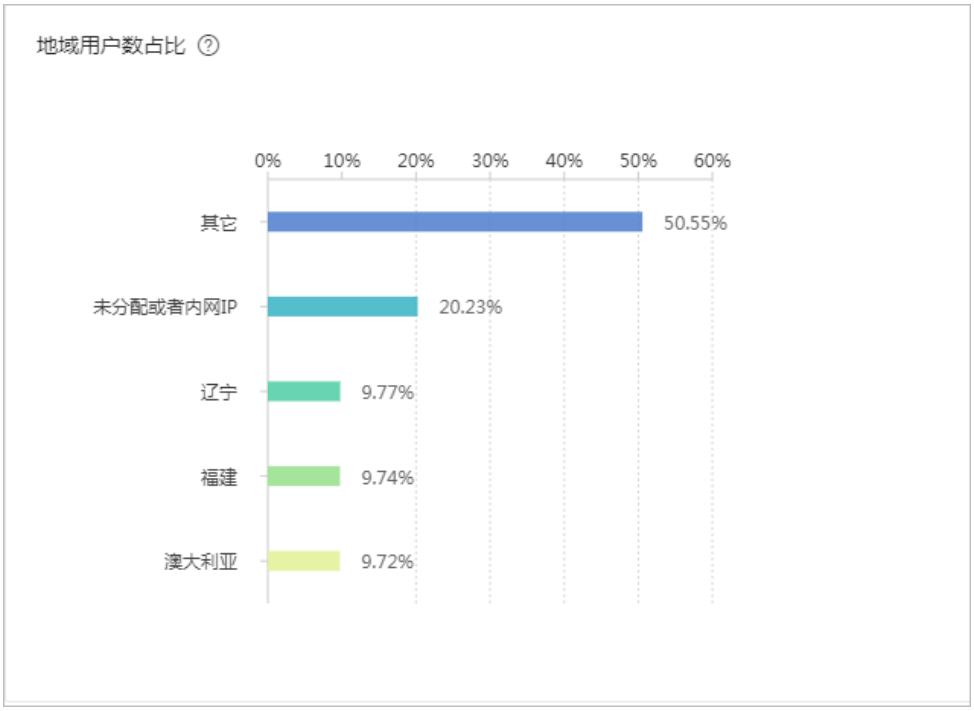
- 启动速度

  可在 **基础分析** > **行为分析** 中查看应用启动速度。



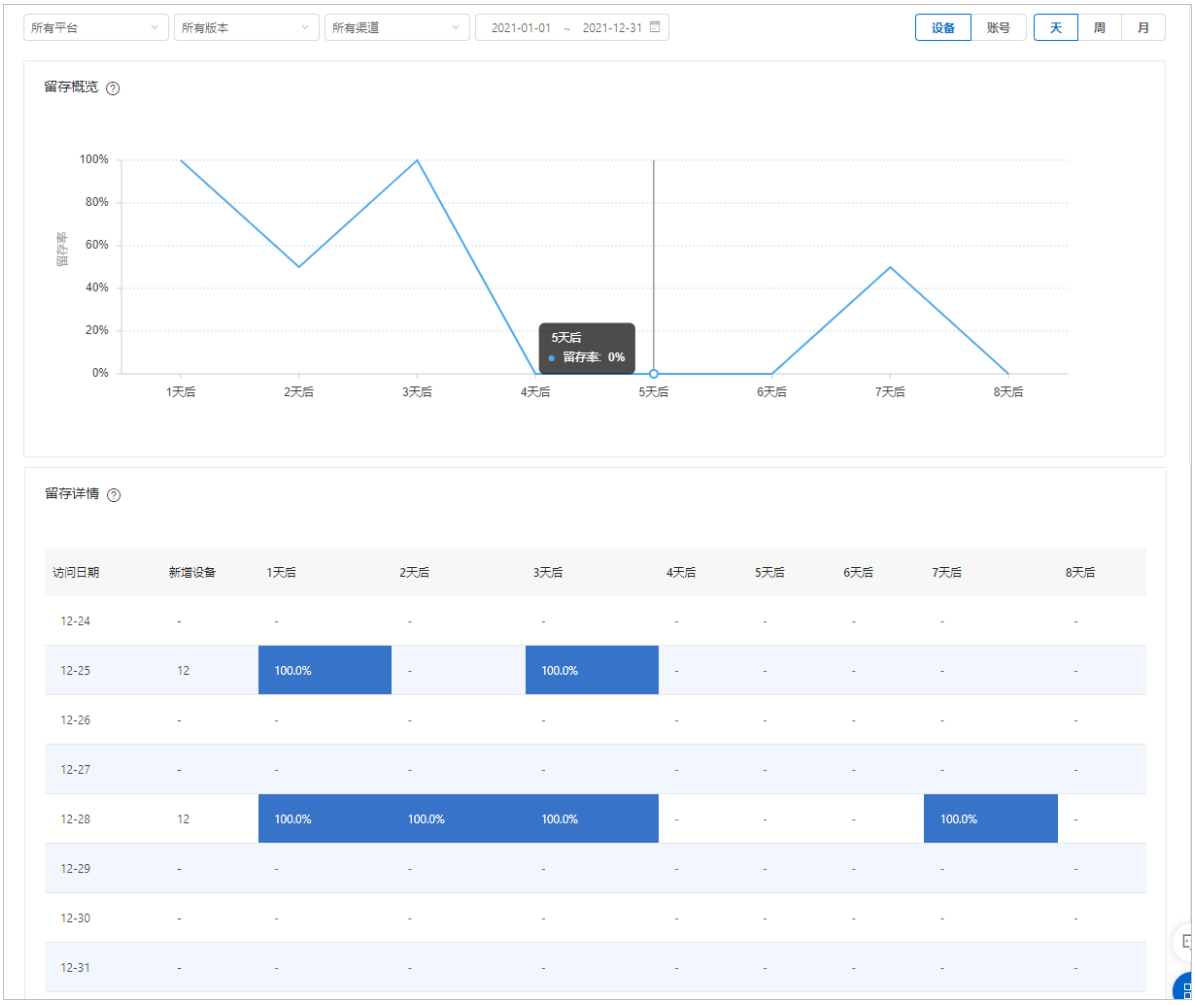
- 地域用户数占比

  可在 **基础分析** > **行为分析** 中查看地域用户数占比。



- 留存分析

  可在 **基础分析** > **留存分析** 中查看留存分析报告。

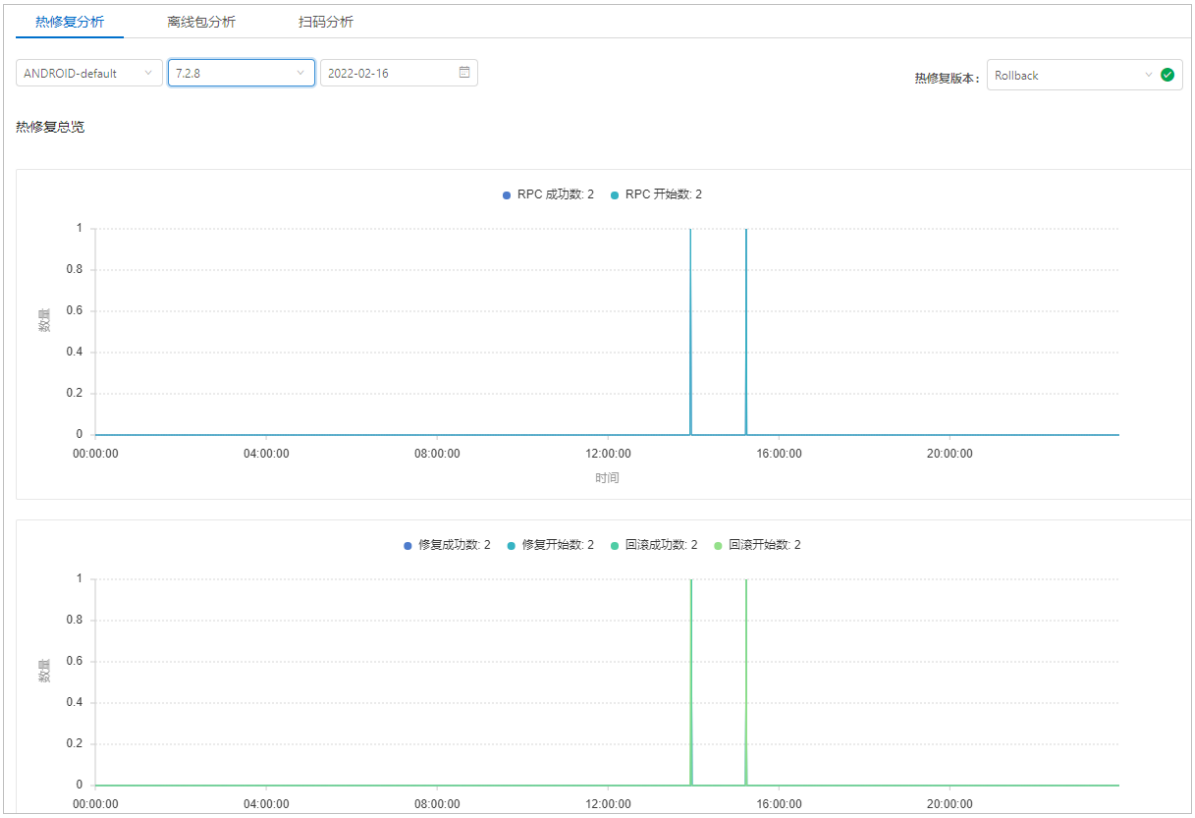


- 闪退分析

  可在 **性能分析** > **闪退报告** 中查看闪退报告。

- 热修复分析

  可在 **组件使用分析** > **热修复分析** 中查看热修复报告。


热修复分析
离线包分析
扫码分析
ANDROID-default
7.2.8
2022-02-16
热修复版本：
Rollback
热修复总览
RPC 成功数: 2
RPC 开始数: 2
数量
时间
00:00:00
04:00:00
08:00:00
12:00:00
16:00:00
20:00:00
修复成功数: 2
修复开始数: 2
回滚成功数: 2
回滚开始数: 2

4.接入 Android

4.1. 快速开始

移动分析依赖客户端 SDK 来进行埋点，收集用户行为以及 App 性能等相关数据生成日志并上报到服务端。根据 mPaaS 客户端与服务端协定的埋点数据格式，服务端从客户端上传的埋点日志中提取有效数据，从而实现对客户端各项指标的监控分析。

本文介绍如何快速将 MAS 组件接入到 Android 客户端。目前，MAS 组件支持 **原生 AAR 接入** 和 **组件化接入** 两种接入方式。

完整的接入过程分为以下六步：

1. 添加 SDK
2. 初始化 mPaaS
3. 添加配置
4. 添加日志
5. 查看本地日志
6. 上报日志

前置条件

- 若采用原生 AAR 方式接入，需要先 将 mPaaS 添加到项目。
- 若采用组件化方式接入，需要先完成 组件化接入流程。

添加 SDK

原生 AAR 方式

参考 管理组件依赖（原生 AAR）AAR 组件管理，通过 **组件管理（AAR）** 在工程中安装 **日志（LOGGING）**组件。

组件化方式

在 Portal 和 Bundle 工程中通过 **组件管理** 安装 **日志（LOGGING）** 组件。更多信息，参考 接入流程添加组件依赖。

初始化 mPaaS

原生 AAR 接入

如果您使用原生 AAR 接入方式，则需要初始化 mPaaS。

在 `Application` 对象中添加以下代码：

```
public class MyApplication extends Application {

    @Override
    public void onCreate() {
        super.onCreate();
        // mPaaS 初始化
        MP.init(this);
    }
}
```

详情请参考：初始化 mPaaS。

组件化接入

使用组件化接入方式，mPaaS 框架会自动初始化，您无需操作。

添加配置

上传日志

上传日志需要访问网络，请在 `AndroidManifest` 中声明以下权限。

```
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.INTERNET" />
```

日志诊断

如果您需要使用 日志诊断 功能，请在 `AndroidManifest` 中声明以下权限，并在 Android 6.0+ 设备上运行时动态申请该权限。

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
```

诊断日志会保存到设备的 SD 卡上，如果未申请该权限，可能无法获取诊断日志。

设置渠道号

如果您需要在控制台中区分不同渠道的 apk 的数据，您可以为 apk 设置渠道号。

在工程的 `assets` 目录下创建 `channel.config` 文件，修改 `channel_id` 的值即可。

```
# 标识当前的发布渠道
channel_id=alipay
```

如果未创建 `channel.config` 文件，渠道号默认为 `mpaas_default`。

获取 IMEI / IMSI

在低于 Android 10 的系统上，已获得相关权限的情况下默认会获取设备的 IMEI 和 IMSI，如果您需要完全禁止获取这类信息的行为，请在 `AndroidManifest` 中添加以下配置：

```
<meta-data
    android:name="imei.switch"
    android:value="off" />
```

> **说明**
>
> 仅在 10.2.3.6 及以上基线生效，添加配置后，移动分析、消息推送、数据同步都将不再获取设备的 IMEI 和 IMSI。

添加日志

SDK 接入完毕后，添加以下日志：

- 报活日志
- 性能日志
- 闪退日志
- 自定义事件日志
- 自动化日志

查看本地日志

查看本地日志 了解本地日志信息。

上报日志

将客户端本地文件中的日志同步到日志服务器。参见 上报日志 说明文档了解相关操作。

4.2. 添加日志

4.2.1. 报活日志

本文介绍如何添加报活日志。

报活日志分为两类：

- **设备报活**：用于统计应用的装机量。
- **用户报活**：用于统计应用的用户量。

您可以在移动分析控制台的 **数据概览** 页面中查看活跃用户、新增用户、活跃账号等指标。

设备报活埋点

mPaaS 框架会自动上报，开发者无需处理。如需关闭框架的自动上报，可通过在 `AndroidManifest.xml` 中添加以下配置（基线 10.1.68.30 及以上版本支持）实现：

```
<meta-data
    android:name="report.launch.switch"
    android:value="off" />
```

关闭自动上报后，可根据需求选择时机上报，代码如下：

```
MPLogger.reportClientLaunch();
```

应用从后台回到前台时，若距离上次设备报活超过上报间隔时间（默认为 30 分钟），将再次报活。

您可以自定义上报间隔时间（单位为毫秒），代码如下：

```
MPLogger.setReportClientLaunchInterval(long interval);
```

若在上报间隔时间内多次调用设备报活，仅首次有效。

用户报活埋点

用户报活的代码如下：

```
MPLogger.reportUserLogin(String userId);
```

`userId` 是您应用登录系统中用户的标识，您可以在用户登录成功后或在其他成功获取 `userId` 的情况下调用用户报活接口。

调用用户报活接口后，`userId` 就成功设置在代码中（即成功设置 `userId`）。除此之外，您还可以调用下方设置用户 ID 的方法，单独设置 `userId`。

设置用户 ID 的代码如下：

```
MPLogger.setUserId(String userId);
```

设置 `userId` 后，可便于在 mPaaS 控制台进行 实时发布 > 白名单 发布时使用该 `userId`。

当用户退出登录时，请调用 `MPLogger.setUserId(null)` 清空 `userId`，以确保相关数据的准确性。

4.2.2. 性能日志

本文介绍如何为移动分析添加性能日志。

移动分析接入 Android 的性能日志包括：

- 启动速度日志
- 卡顿日志
- 卡死日志

您可以在 **mPaaS 控制台 > 移动分析 > 基础分析** 中查看启动速度指标；在 **mPaaS 控制台 > 移动分析 > 性能分析** 中查看卡顿、卡死报告。

启动速度埋点

应用启动时长 = 调用该方法的时刻 - 应用开始启动的时刻。

推荐在首页 `Activity` 的 `onCreate()` 方法中调用如下方法启动速度埋点。

```
MPLogger.reportLaunchTime(Context context);
```

卡顿埋点

卡顿的定义为 Android 主线程超过 **2.25 秒** 仍未执行完一个方法。卡顿阈值因 APK 包类型而异：

- APK 为 debug 包时，卡顿阈值为 0.75 秒，以便调试时快速发现潜在的卡顿问题。
- APK 为 release 包时，卡顿阈值为 2.25 秒。

开启卡顿监控

方式一

界面需要继承 mPaaS 提供的 `BaseActivity`、`BaseFragmentActivity` 或 `BaseAppCompatActivity` 类，凡是继承了这些类的界面都会自动监控卡顿。

方式二

> **重要**
> 该方式仅在基线 10.2.3.50 及以上版本支持。

在 `Activity` 的生命周期方法中手动调用相关接口，例如：

```
import android.app.Activity;
import android.app.Application;
import android.os.Bundle;

import com.mpaas.mas.adapter.api.MPLogger;

public class MPLifecycle implements Application.ActivityLifecycleCallbacks {

    private int mVisibleActivityCount = 0;
    private boolean isBackground = false;

    @Override
    public void onActivityCreated(Activity activity, Bundle bundle) {

    }

    @Override
    public void onActivityStarted(Activity activity) {
        mVisibleActivityCount++;
        if (isBackground) {
            isBackground = false;
            // 应用回到前台时调用
            MPLogger.monitorAppForeground();
        }
    }

    @Override
    public void onActivityResumed(Activity activity) {
        // 更新 Activity 上下文
        MPLogger.monitorActivityResumed(activity);
    }

    @Override
    public void onActivityPaused(Activity activity) {
    }

    @Override
    public void onActivityStopped(Activity activity) {
        mVisibleActivityCount--;
        if (mVisibleActivityCount <= 0) {
            isBackground = true;
            // 应用退到后台时调用
            MPLogger.monitorAppBackground();
        }
    }

    @Override
    public void onActivitySaveInstanceState(Activity activity, Bundle bundle) {
    }

    @Override
    public void onActivityDestroyed(Activity activity) {
    }

}
```

当 APK 为 debug 包时，卡顿监控为全量统计；APK 为 release 包时，卡顿监控为采样统计，采样率为 10%。

卡死埋点

卡死即 Android 系统的 ANR，通常情况下指主线程无响应时间 **超过 5 秒**。

要开启卡死监控，详情请参考上文卡顿埋点中的 开启卡顿监控。

4.2.3. 闪退日志

闪退日志用于统计应用的闪退情况。您可以在 mPaaS 控制台的 **移动分析** > **性能分析** 中查看闪退报告。

框架会自动捕获闪退信息并上传日志到服务端，无需处理。

4.2.4. 自定义事件日志

根据业务需求，您可以实现自定义埋点，用于分析用户的行为。

在接入客户端后，您还需要在移动分析控制台的 **自定义分析** > **自定义配置** 中配置相关属性和事件，然后才能在 **自定义分析** > **事件分析** 中查看相关数据。详情参见 配置自定义分析。

埋点代码如下：

```
MPLogger.event(String logId, String bizType, Map<String, String> params);
```

- `logId`：埋点 ID，不能为空，对应控制台新建事件中的事件 ID。
- `bizType`：业务类型，可以为空，默认为 `userbehavior`。业务类型中不能包含下划线 `_`。
  - 拥有相同的 `bizType` 的自定义日志会存储在一个名为 `时间戳_包名-进程_bizType` 的本地文件中。
  - 关于日志文件的命名规范，请参考 查看本地日志 中的日志文件名。
- `params`：扩展参数，可以为空。`params` 的 key 对应控制台新建属性中的属性 ID，value 为该属性的值。

您还可以使用以下重载方法：

```
MPLogger.event(String logId);

MPLogger.event(String logId, String bizType);

MPLogger.event(String logId, Map<String, String> params);
```

4.2.5. 自动化日志

自动化日志用于记录页面切换事件。您可以借此分析应用各功能或运营页面的 PV 和 UV 等数据。

初始化埋点

调用下面方法，初始化自动化日志埋点。

```
MPLogger.enableAutoLog();
```

- 对于 Portal&Bundle 工程，推荐在 `MockLauncherActivityAgent` 的 `postInit` 方法中调用。
- 对于原生 AAR 工程，推荐在 `Application` 的 `onCreate` 方法中调用，且需在调用 mPaaS 框架初始化方法之后。

配置 Activity

Activity 从 `onResume` 开始到 `onPause` 结束，会记录一次页面打开的 PV，页面标识为 Activity 的类名。

- 继承自 mPaaS 框架的 `BaseActivity`、`BaseFragmentActivity` 或 `BaseAppCompatActivity` 的 Activity 可自动记录。
- 若不继承 mPaaS 基类，可在 `BaseActivity` 中添加监听生命周期代码：

```
public class BaseActivity extends Activity {

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    MPTracker.onActivityCreate(this);
}

@Override
public void onWindowFocusChanged(boolean hasFocus) {
    super.onWindowFocusChanged(hasFocus);
    MPTracker.onActivityWindowFocusChanged(this, hasFocus);
}

@Override
protected void onResume() {
    super.onResume();
    MPTracker.onActivityResume(this);
}

@Override
protected void onPause() {
    super.onPause();
    MPTracker.onActivityPause(this);
}

@Override
protected void onDestroy() {
    super.onDestroy();
    MPTracker.onActivityDestroy(this);
}
}
```

配置 Fragment

- 使用 mPaaS 提供的 `com.mpaas.mas.adapter.api.BaseFragment`，直接继承即可。
- 使用官方提供的 `support-v4` 库中的 `Fragment`，需让 `BaseFragment` 实现 `TrackPageConfig` 接口，并添加监听生命周期代码：

```
public class BaseFragment extends Fragment implements TrackPageConfig {

/**
 * 页面标识，一般使用类名
 * 不传会导致控制台页面分析中不显示
 */
@Override
public String getPageSpmId() {
    return this.getClass().getName();
}

@Override
public Map<String, String> getExtParam() {
    return null;
}

@Override
public boolean isTrackPage() {
    return true;
}

@Override
public void onResume() {
    super.onResume();
    MPTracker.onFragmentResume(this);
}

@Override
public void onPause() {
    super.onPause();
    MPTracker.onFragmentPause(this);
}

@Override
public void onHiddenChanged(boolean hidden) {
    super.onHiddenChanged(hidden);
    MPTracker.onFragmentHiddenChanged(this, hidden);
}

@Override
public void setUserVisibleHint(boolean isVisibleToUser) {
    super.setUserVisibleHint(isVisibleToUser);
    MPTracker.onFragmentSetUserVisibleHint(this, isVisibleToUser);
}

@Override
public void onDestroy() {
    super.onDestroy();
    MPTracker.onFragmentDestroy(this);
}
}
```

添加自定义参数

在 10.1.68.44 及以上基线版本中，可通过以下方法在自动化日志中添加自定义参数。

```
MPLogger.addAutoLogCustomParam("test_key1", "test_value1");
MPLogger.addAutoLogCustomParam("test_key2", "test_value2");

Map<String, String> params = new HashMap<>();
params.put("test_key3", "test_value3");
params.put("test_key4", "test_value4");
MPLogger.addAutoLogCustomParams(params);
```

4.3. 查看本地日志

通过本文，您将能够快速在本地找到想要查看的日志。

日志保存路径

日志保存路径取决于 `assets/channel.config` 中的 `release_type` 字段：

- 若该字段值为 dev、test 或 testpre ，则为 **线下模式**；若为 release，则为 **线上模式**。
- 若无该文件或未填写 `release_type` 字段：
  - apk 为 debug 包时为 **线下模式**。
  - apk 为 release 包时为 **线上模式**。

保存路径：

- **线上模式** 的日志保存路径为：`/data/data/[PackageName]/files/mdap`，上传后即删除。
- **线下模式** 的日志保存路径为：`/sdcard/Android/data/[PackageName]/files/mdap`，上传开始时会备份至 `/sdcard/Android/data/[PackageName]/files/mdap/upload`，备份时会增加时间戳为文件名前缀。

日志文件名

文件命名规则为：(时间戳_)进程名-业务码。其中业务码（bizType）如下：

- 设备报活日志/用户报活日志：`mPaaSAliveAndroid`。
- 应用启动速度日志：`mPaaSLaunchAndroid`。
- 闪退日志：`mPaaSCrashAndroid`。
- 卡顿日志：`mPaaSLAGAndroid`。
- 卡死日志：`mPaaSANRAndroid`。
- 自定义事件日志：埋点时传入的 `bizType`，不传则默认为 `UserBehaviorAndroid`。
- 自动化日志：`mPaaSAutomationAndroid`。

日志内容格式

- 设备报活日志：请参考 日志模型 > 行为埋点。其中 **字段 16**（埋点 ID） 为 `reportActive`。
- 用户报活日志：请参考 日志模型 > 行为埋点。其中 **字段 16**（埋点 ID） 为 `login`。
- 应用启动速度日志：请参考 日志模型 > 性能埋点 。其中 **字段 12**（性能 ID） 为 `performance`，**字段 13**（性能类别） 为 `time_startup`。
- 闪退日志：请参考 日志模型 > 闪退埋点。
- 卡顿日志：请参考 日志模型 > 卡顿埋点。
- 卡死日志：请参考 日志模型 > 卡死埋点。
- 自定义事件日志：请参考 自定义事件埋点 > 行为埋点。其中 **字段 16**（事件 ID） 对应埋点时传入的 `logId`。

- 自动化日志：请参考 日志模型 > 自动化埋点。其中 **字段 09**（行为 ID） 为 `auto_openPage` ，表示页面自动化。

下一步

上报日志

4.4. 上报日志

写到客户端本地文件中的日志，会通过以下方式同步到日志服务器。

- 自动上报：满足一定条件后自动上报。
- 日志开关上报：在自动上报的基础上，通过服务端下发的开关值，修改自动上报触发的条件。
- 手动上报：调用上报日志接口强制上传。

自动上报

触发日志自动上报的条件如下：

- 本地缓存的日志满一定条数会自动触发上报，具体规则为：
  - 报活、启动速度、卡顿、卡死、闪退日志实时上报。
  - 自定义事件日志、自动化日志满 50 条时上报。
- 应用程序压后台时会触发一次上报。

日志开关上报

在上述默认触发日志上报的条件下，您还可以通过控制台的 **移动分析** > **日志管理** > **配置上传开关** > **埋点配置** 页面动态控制日志上报的时机。

其中：

- **上报开关**：只有打开此开关，此日志和开关配置才生效。
- **网络**：选择 **全网环境** 或 **仅在 WiFi** 环境下上报。
- **业务码**：与客户端埋点时的业务码（bizType）对应。常见的业务码请参见 查看本地日志。
- **日志上报条数**：本地文件中此类型的日志到达条数触发日志上报。
- **日志上报比率**：按用户维度设置日志上报的比率，采用千分制，如 1000 表示全部用户都上报。
- **最低上报等级**：每条日志写入时都会设置一个等级，小于等于此设置值的日志上报。如设置最低上报等级为 2 时，则等级为 1 和 2 的日志会上报，而等级为 3 的日志不上报。

更多信息，请参见 配置日志上报开关。

手动上报

通过以下方式手动上报：

```
MPLogger.uploadAll();
```

5.接入 iOS

5.1. 接入流程介绍

移动分析依赖客户端 SDK 来进行埋点，收集用户行为及 App 性能相关数据，生成日志并上报到服务端。根据 mPaaS 客户端与 mPaaS 服务端协定的埋点数据格式，服务端从客户端上传的埋点日志中提取有效数据，从而实现对客户端各项指标的监控分析。

您可以基于已有工程使用 CocoaPods 接入移动分析 SDK 到客户端。接入流程如下：

1. 通用接入流程

   为了使用 mPaaS 组件，您首先需要参考 基于已有工程且使用 CocoaPods 接入 完成通用接入流程。

2. 接入移动分析组件

   i. 使用 CocoaPods 添加 SDK。

   ii. 参考对应版本的 SDK 使用文档，使用 SDK。包括配置工程和添加各类埋点日志。

   iii. 查看本地日志。

   iv. 上报日志。

5.2. 添加 SDK

本文介绍如何将移动分析组件接入到 iOS 客户端。您可以基于已有工程使用 CocoaPods 接入移动分析 SDK 到客户端。

前置条件

您已接入工程到 mPaaS。更多信息，请参见 基于已有工程且使用 CocoaPods 接入。

添加 SDK

使用 cocoapods-mPaaS 插件添加移动分析 SDK。

操作步骤如下：

1. 在 Podfile 文件中，使用 `mPaaS_pod "mPaaS_Log"` 添加移动分析组件依赖。
2. 执行 `pod install` 即可完成接入。

后续操作

参考对应版本的 SDK 使用文档，使用 SDK。

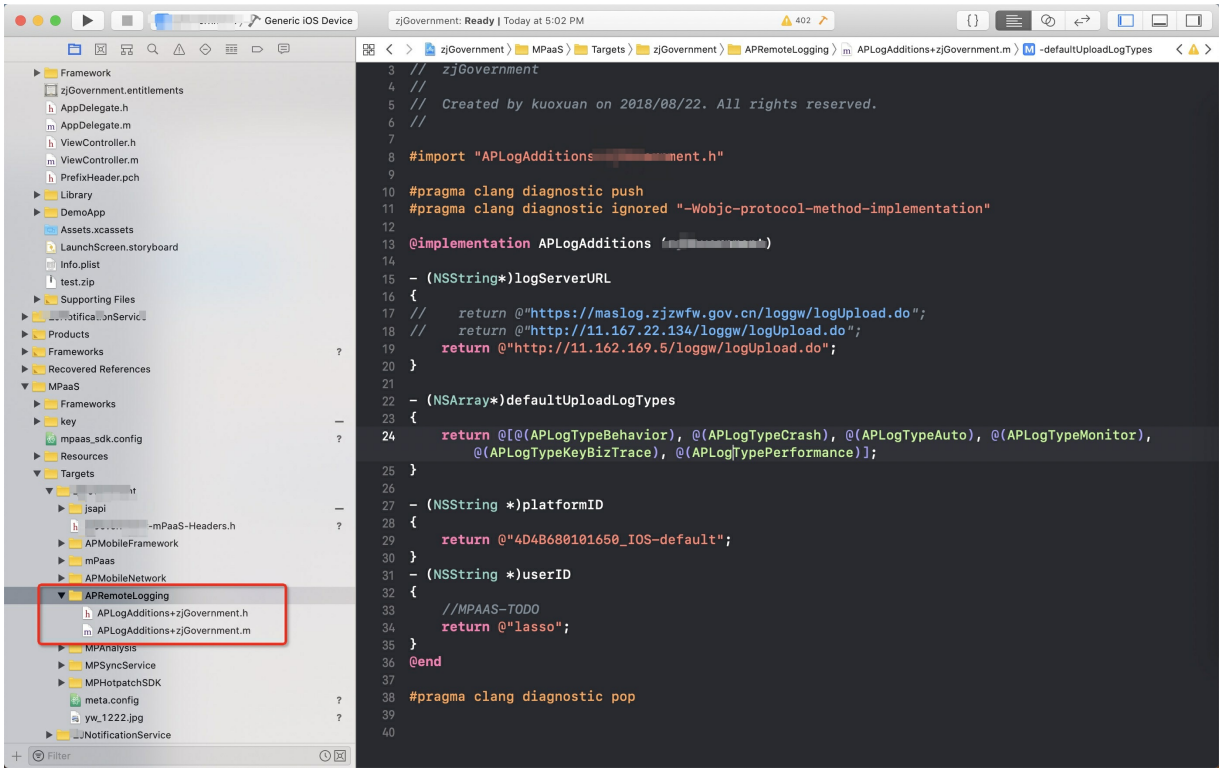
5.3. 使用 SDK

5.3.1. 配置工程

添加 SDK 之后，需要对工程进行一些配置。

旧版本升级后配置

在 10.1.32 及以后版本中，无需再添加 `APRemoteLogging` 类的 Category 文件，中间层会实现包装从 `meta.config` 中读取。所以升级版本后请检查工程中是否存在旧版本配置，如果有请移除。下图为应从新版本中移除的 `APRemoteLogging` 类的 Category 文件。



配置用户 ID

在工程配置用户 ID。埋点日志中默认会记录用户 ID。您可以在 `MPaaSInterface` 的 Category 文件中配置用户 ID：

```
@implementation MPaaSInterface (Portal)
- (NSString *)userId
{
    return @"the-user-id";
}
@end
```

后续操作

接下来，可以添加以下类型的日志：

- 报活日志
- 性能日志
- 闪退日志
- 自定义事件日志
- 自动化日志

5.3.2. 添加报活日志

报活日志用于统计应用的装机量和用户量，您可以在移动分析控制台的 **数据概览** 页面中查看新增用户、活跃用户、活跃账号等指标。

支持基于 mPaaS 框架和原生工程进行日志埋点。

基于 mPaaS 框架

框架会在应用启动后自动记录一次报活，并同步到应用分析控制台，您无需处理。

基于原生工程

SDK 封装了报活接口，推荐您在 `didFinishLauncher` 中调用，通过引用头文件 `<MPMasAdapter/MPAnalysisHelper.h>` 来实现接口调用。

```
- (void)reportActive
{
    // 用户报活
    [[MPAnalysisHelper sharedInstance] writeLogForReportActive];
}
```

5.3.3. 添加性能日志

性能日志用于统计应用的启动速度、卡顿与卡死等情况。您可以在移动分析控制台的 **基础分析** 页面查看启动速度指标；在 **性能分析** 页面中查看卡顿与卡死报告。

支持基于 mPaaS 框架和原生工程进行日志埋点。

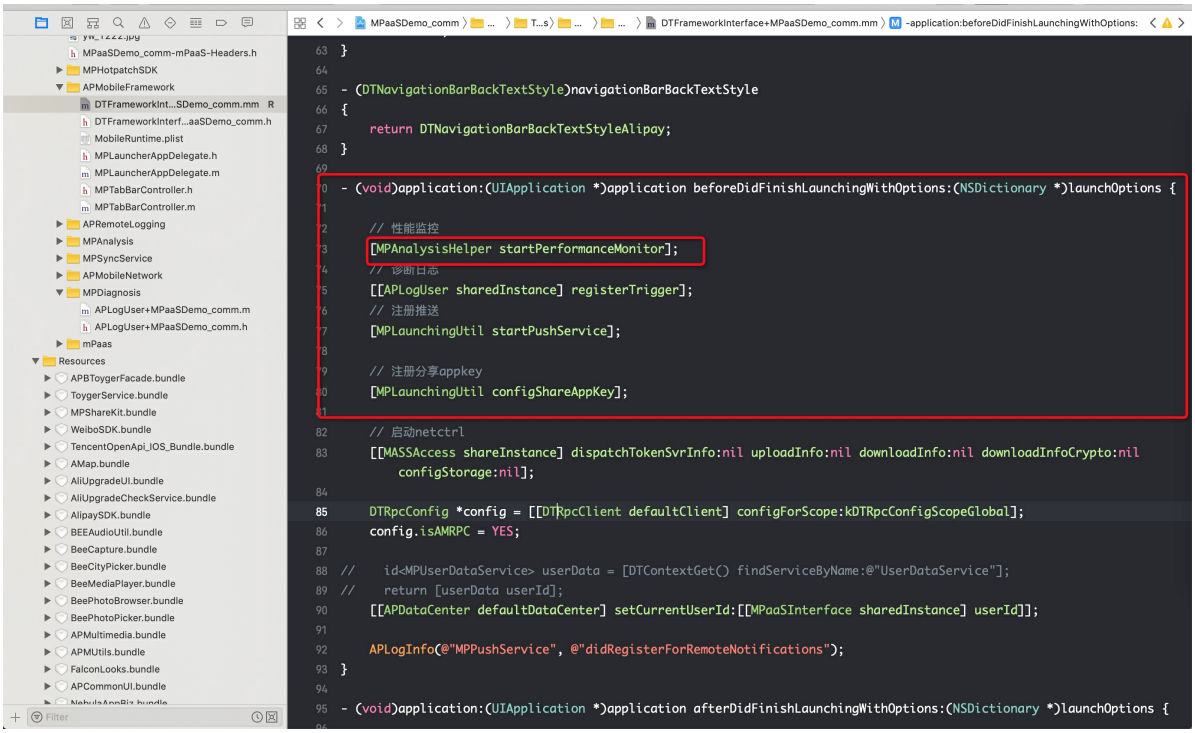
基于 mPaaS 框架

1. 卡顿监控默认对 10% 的设备开启，可通过下面这个接口设置卡顿开启率。

   `[MPAnalysisHelper setLagMonitorPercent: 100]; // 100%` 监控，需要在 `startPerformanceMonitor` 调用之前设置

   卡顿监控只有在真机上并且非 Xcode 调试状态下是打开的。

2. 在启动时调用 `[MPAnalysisHelper startPerformanceMonitor]` ，推荐在 `-(void)application:(UIApplication *)application beforeDidFinishLaunchingWithOptions:(NSDictionary *)launchOptions` 方法中调用。



基于原生工程

SDK 封装了性能监控接口，推荐您在 AppDelegate 的 `- (BOOL)application:(UIApplication )application didFinishLaunchingWithOptions:(NSDictionary )launchOptions` 方法中调用 `[PerformanceHelper performanceMonitor]` 。

```
    #import "PerformanceHelper.h"
    #import <MPAnalysis/MPAnalysisHelper.h>

    static NSTimeInterval __start_timestamp = 0;

    @implementation PerformanceHelper

    + (void)load
    {
        __start_timestamp = CFAbsoluteTimeGetCurrent();
    }

    + (void)performanceMonitor
    {
        //start performance monitor

       [MPAnalysisHelper setLagMonitorPercent: 100]; // 100% 监控，需要在
startPerformanceMonitor 调用之前设置
        [MPAnalysisHelper startPerformanceMonitor];

        //record the time interval used for the app startup
        NSTimeInterval time = CFAbsoluteTimeGetCurrent() - __start_timestamp;
        [[MPAnalysisHelper sharedInstance] writeLogForStartupWithTime:time];

    }

    @end
```

卡顿监控只有在真机上并且非 Xcode 调试状态下是打开的。

自定义设置性能监控阈值

当默认的性能监控阈值无法满足您的需求时，可以自定义设置相关阈值。

设置卡顿阈值

```
#引入头文件
#import <MPMasAdapter/MPAnalysisHelper.h>

/**
 设置主线程卡顿监控的阈值，单位秒，可不设置，默认值为2秒
 */
+ (void)setLagTimeThreshold:(NSUInteger)threshold;

/**
 设置卡顿检测的间隔时长， 建议 lagTimeThreshold / lagCheckInterval 等于整数
 */
+ (void)setLagCheckInterval:(NSTimeInterval)interval;
```

设置卡死阈值

```
#import <MPMasAdapter/MPMasSettings.h>

#创建 MPMasSettings 分类，重写下面方法做自定义


/**
 获取卡死时长阈值，需自定义时在Category中重写，建议 anrTimeThreshold / anrCheckInterval 等于整数
 */
- (NSUInteger)anrTimeThreshold
{
    return 自定义的时长;
}


/**
 获取卡死检测间隔时长，需自定义时在Category中重写，建议 anrTimeThreshold / anrCheckInterval 等于整数
 */
- (NSTimeInterval)anrCheckInterval
{
    return 自定义的检测间隔时长;
}
```

设置启动卡死时长阈值

```
#import <MPMasAdapter/MPMasSettings.h>

#创建 MPMasSettings 分类，重写下面方法做自定义


/**
 获取启动卡死时间阈值，需自定义时在Category中重写
 */
- (NSUInteger)startupAnrTimeThreshold
{
    return 自定义的时长;
}
```

5.3.4. 添加闪退日志

闪退（Crash）日志用于统计应用的闪退情况。您可以在移动分析控制台的 **性能分析** 页面中查看闪退报告。

支持基于 mPaaS 框架和原生工程进行日志埋点。

基于 mPaaS 框架

在接入了框架（工程中有 `APMobileFramework` 库）的情况下，闪退上报模块会自动捕获闪退日志并上传到服务端，您只需在集成 SDK 后确认闪退监控的开关为打开即可。为保证闪退日志能及时上报，推荐您在 `main` 函数中开启 Crash 监控。

```
#import <MPMasAdapter/MPMasAdapter.h>

[MPAnalysisHelper enableCrashReporterService];
```

基于原生工程

在未接入框架（工程中无 `APMobileFramework` 库）的情况下，需要您在启动时打开闪退监控开关，并且在启动后上报闪退日志。

1. 在 main 方法中开启 Crash 监控。

```
#import <MPMasAdapter/MPMasAdapter.h>

[MPAnalysisHelper enableCrashReporterService];
```

2. 在启动流程的 `didFinishLaunchingWithOptions` 方法中上报 Crash 日志。

```
#import <MPMasAdapter/MPMasAdapter.h>

[[MPAnalysisHelper sharedInstance] writeLogForCrashReporter];
```

容灾开关

默认情况下，当连续发生四次闪退时会触发容灾处理，将会清理 `Documents` 目录下的文件，以避免因脏数据导致的闪退问题。在 10.1.60 及以上版本，您可以手动调用以下接口开启或关闭容灾处理。

```
#import <MPMasAdapter/MPAnalysisHelper.h>
/**
 * 开启/关闭闪退容灾处理，默认开启
 */
+ (void)enableDisasterRecovery:(BOOL)enable;
```

注意事项

- 只有真机运行程序的闪退日志才会被捕获并上传到日志服务器。如您需要调试闪退监控，请断开 Xcode，且请勿使用模拟器。
- 为保证闪退日志中的 version 与 product version 一致，务必在项目的 `info.plist` 中将 bundle version 与 product version 设置为相同的版本号。

5.3.5. 添加自定义事件日志

自定义事件日志记录按钮、链接点击等操作，可在 App 内任意动作触发时埋入，用于自定义事件分析和漏斗分析等功能。根据业务需求，您可以通过自定义事件埋点实现用户行为分析。

在接入客户端后，您还需要在移动分析控制台的 **自定义分析 > 自定义配置** 中配置相关属性和事件，然后才能在 **自定义分析 > 事件分析** 中查看相关数据。

埋点接口

自定义事件埋点接口定义在 `MPMasAdapter` 的 `MPRemoteLoggingInterface` 类中。接口定义如下：

```
/**
 *  行为埋点接口。客户端版本、用户 ID、设备 ID、操作系统版本、网络类型、设备类型、软件版本会自动填充，不需要业务埋点。
 *  @param bizType  可选，业务类型，默认为 User_behavior_iOS，建议业务方填写业务标识
 *  @param eventId  必填， 埋点 ID
 *  @param extParam 可选，扩展参数，业务自己根据需要填充。元素是字典，字典内容可自定义，字典会被转换成key-value的字符串计入日志。
 */
+ (void)writeLogWithBizType:(NSString *)bizType
                    eventId:(NSString *)eventId
                   extParam:(NSDictionary *)extParam;
```

参数说明

参数	说明
bizType	可选，默认为 `User_behavior_iOS` ，建议业务方填写业务标识。
eventId	埋点 ID，对应控制台新建事件中的事件 ID。
extParam	扩展参数，字典中的 key 对应控制台新建属性中的属性 ID，value 的类型决定属性对应的数据类型。

代码示例

```
[MPRemoteLoggingInterface writeLogWithBizType:@"customBiz" eventId:@"customEvent" extParam:@{@"key":@"v"}];
```

5.3.6. 添加自动化日志

页面自动埋点自动记录页面的打开、来源、停留时长等信息。用于分析页面 PV、UV、来源去向等指标。

埋点

您无需手动进行日志埋点，原因如下：

- 自动化日志是一种特殊的行为埋点。
- 在运行时利用 Objective-C 方法交换技术替换系统方法，同时插入监控方法。
- 当用户触屏或是系统通知等方式触发监控方法时，监控方法会主动调用行为日志的埋点接口进行埋点日志的上报。

规则

在 `UIViewController` 的生命周期中，从 `viewDidAppear` 开始为页面打开，到 `viewWillDisappear` 为页面结束，记录当前页面打开一次，页面 PV 加一。

添加自定义参数

可通过下述方法为页面自动化埋点添加自定义的参数。自定义参数会出现在埋点扩展字段4（custParm=123^cus2=myinfo 的格式）。

```
#import <MPMasAdapter/MPRemoteLoggingInterface.h>

/**
  设置自动化埋点的自定义的扩展参数
 */
+ (void)setAutoRemoteLogExtendParam:(NSDictionary *)extParam;
```

设置自动化点击埋点开关

```
#import <MPMasAdapter/MPMasSettings.h>

#创建 MPMasSettings 分类，重写下面方法做自定义


/**
 是否开启自动化点击埋点，默认开启，需关闭时返回 NO
 */
+ (BOOL)enableAutoClick
{
 return NO;
}
```

5.4. 查看本地日志

调用日志接口写的日志会先写入到本地应用的沙盒文件中，触发日志上报逻辑后，再上传到日志服务器。

本地日志格式

10.1.60 & 10.1.68 版本

- 写入本地的日志在沙盒的 **Library** > **atrack** > **logs** 文件夹下。该文件夹仅存放还未上报的日志，已经上报的日志不再保存。
- 日志文件命名规则为 `业务码.log` 。根据写入日志时传入的业务码（bizType）参数对日志进行分类，同类型的日志会写入相同文件。目前，几种常见的埋点类型有：
  - `autotrack` ：自动化埋点。
  - `behavior` ：行为埋点（包括报活埋点和自定义事件埋点等）。自定义事件埋点中，您可以通过 `bizType` 参数自定义业务码；更多信息，请参见 自定义事件埋点。
  - `performance` ：性能埋点（包括应用启动速度埋点等）。

10.1.32 版本

> **重要**
>
> **自 2020 年 6 月 28 日起，mPaaS 停止维护 10.1.32 基线。请使用 10.1.68 或 10.1.60 系列基线。可以参考 mPaaS 10.1.68 升级指南 或 mPaaS 10.1.60 升级指南 进行基线版本升级。**

- 写入本地的日志在沙盒的 `Library > log` 文件夹下。
- 日志文件命名规则为 `业务码.时间戳.log` 。根据写入日志时传入的业务码（bizType）参数对日志进行分类，同类型的日志会写入相同文件。目前，几种常见的埋点类型有：
  - `autotrack` ：自动化埋点。
  - `behavior` ：行为埋点（包括报活埋点和自定义事件埋点等）。自定义事件埋点中，您可以通过 `bizType` 参数自定义业务码；更多信息，请参见 自定义事件埋点。
  - `crash` ：异常埋点。
  - `performance` ：性能埋点（包括应用启动速度埋点等）。
- 程序每次冷启动都会重新生成一个 `.log` 的日志文件。

埋点日志格式

- 打开一个日志文件，可以看到每个日志文件是按行进行组织的，即一行为一条日志。

- 每条日志是一个由逗号分隔的字符串，字符串的不同位置代表不同的含义。服务器根据位置信息来切分日志。一条完整的日志格式如下：

```
0_     257_1479573031.408824_D-VM,2016-11-20 00:30:31:408,1000533192018_IOS-
0000000001,2.0.X.X,2,-,7542B136-5EA8-4C3A-930D-8BF2CA15F3CA,-,event,-,-,-,-,-,-,startApp,-
,u,c,Launcher,-,NativeApp,-,-,-,-,-,2,-,-,-,-,-,-,iPhone 6S,9.3.3,WIFI,-,-
,follow_system_zh-Hans-CN,-,-,-,-,VoiceOver=0$$
```

- 10.1.60 和 10.1.68 版本中，每条日志开头的标记信息不再标识日志上传的状态。
- 10.1.32 版本中，每条日志开头都有标记信息，第一个数字代表日志的上传状态。0 表示日志还没有上传，1 代表日志已上传。
- 日志各字段的具体含义请参考 日志模型。

后续操作

上报日志

5.5. 上报日志

写入客户端本地文件中的日志，会通过以下三种方式同步到日志服务器。

- 自动上报：满足一定条件后自动上报。
- 日志开关上报：在自动上报的基础上，通过服务端下发的开关值，修改自动上报触发的条件。
- 手动上报：调用上报日志接口强制上报。

如需停止日志上报，可通过关闭 iOS 客户端埋点开关来实现。具体参见 停止日志上报。

自动上报

触发日志自动上报的条件如下：

- 程序每次冷启动都会触发检查日志上报的逻辑。
- 程序进入后台会立即触发上报。
- 写日志时，某种类型的日志默认达到 40 条就触发上报。
- 为保证闪退日志能够及时上报，每次发生闪退后，会在应用下次启动时触发上报。

日志开关上报

在上述默认触发日志上报的条件下，您还可以通过控制台 **移动分析** > **日志管理** > **配置上传开关** > **埋点配置** 对日志上报进行动态控制。

在控制台动态控制日志上报，各参数的含义如下：

- **上报开关**：只有打开此开关，此日志和开关配置才生效。
- **网络**：选择 **全网环境** 或 **仅在 WiFi** 环境下上报。
- **业务码**：与客户端埋点时的 bizType 对应。常见的业务码请参见 查看本地日志。
- **最低上报等级**：每条日志写入时都会设置一个等级，小于等于此设置值的日志会被上报。如设置最低上报等级为 2 时，则等级为 1 和 2 的日志会被上报，而等级为 3 的日志不会被上报。
- **日志上报条数**：本地文件中此 bizType 类型的日志达到条数触发上传（即修改上述 40 条的阈值）。
- **日志上报比率**：按用户维度设置日志上报的比率，采用千分制，如 1000 表示全部用户的日志都上报。更多信息，请参见 配置日志上报开关。

手动上报

若业务需要保证某些日志实时上报，您可以调用以下接口强制上报日志：

```
[MPRemoteLoggingInterface upload];
```

停止日志上报

关闭 iOS 客户端埋点开关来停止日志上报。

> **重要**
> 该方法仅适用于 10.1.68.42 及以上基线版本。

停用埋点的方法如下：

```
#import <MPMasAdapter/MPAnalysisHelper.h>

[MPAnalysisHelper enableRemoteLog:NO];
```

6.接入 HarmonyOS NEXT

6.1. 添加 SDK

引入依赖

在项目的 `.ohpmrc` 中添加如下仓库：

```
@mpaas:registry=https://mpaas-ohpm.oss-cn-hangzhou.aliyuncs.com/meta
```

`oh-package.json5` 的 `dependencies` 引入 `@mpaas/masadapter`，版本参考开发指南手册中基于已有工程使用 ohpmrc 接入。

```
{
  "license": "",
  "devDependencies": {},
  "author": "",
  "name": "entry",
  "description": "Please describe the basic information.",
  "main": "",
  "version": "1.0.0",
  "dynamicDependencies": {},
  "dependencies": {
    "@mpaas/masadapter": "填入参考的版本",
  }
}
```

使用 `ohpm install` 安装移动分析依赖。

6.2. 使用 SDK

前置条件

- 检查 `rawfile` 目录下 `mpaas.config` 里的移动分析服务端地址 `logGW` 配置是否正确。
- 使用移动分析 SDK 之前，需要先初始化 mPaaS 框架 SDK，详情请参考开发指南手册中接入 mPaaS 框架。

添加权限

项目中添加网络权限，代码如下：

```
 "requestPermissions":[
      {
        "name" : "ohos.permission.GET_NETWORK_INFO"
      },
      {
        "name" : "ohos.permission.INTERNET"
      }
  ]
```

初始化 MPRemoteLogger

```
import { MPRemoteLogger } from '@mpaas/masadapter';

MPRemoteLogger.init();
```

使用接口

App 报活

```
import { MPRemoteLogger } from '@mpaas/masadapter';

MPRemoteLogger.reportActive();
```

App 登录报活

```
import { MPRemoteLogger } from '@mpaas/masadapter';

MPRemoteLogger.reportUserLogin(userId);
```

> **重要**
> 在用户 ID 发生变化时调用，以保证埋点的 userId 的更新。

App 启动速度

```
import { MPRemoteLogger } from '@mpaas/masadapter';
const startupTime = startup; //单位为毫秒
MPRemoteLogger.reportStartupTime(startupTime);
```

自定义埋点

可使用自定义埋点上报业务埋点，代码如下：

```
import { MPRemoteLogger } from '@mpaas/masadapter';

let param = new Map<string, string>(); // 可选
param.set('a', 'b');
MPRemoteLogger.logBehavior('myBiz', 'myEventId', param);
```

性能监控

- 总初始化接口

  性能监控的总初始化接口，调用之后无需再进行下述的各个（Crash 和 ANR 监控、启动卡死监控）初始化，但是需要在启动完成后调用启动完成通知 `MPRemoteLogger.startupFinish()` 。

  ```
  import { MPRemoteLogger } from '@mpaas/masadapter';

  /**
     * 初始化性能监控埋点（包括 crash 、 anr 、启动卡死）
     * 调用此方法后，无需再调用 initFaultTrack、startPerformanceMonitor 方法
     * 注意：启动完成的时候，需要手动调用 MPRemoteLogger.startupFinish() 方法以通知完成启动
     */
  MPRemoteLogger.initPerformanceTrack();

  // App启动完成的时候，调用下述代码通知完成启动
  MPRemoteLogger.startupFinish();
  ```

  > **重要**

> 当您需要选择性接入 Crash 监控、启动卡死监控时，请不要调用总初始化接口，而是根据需求分别调用下述所需的接口。

- Crash 和 ANR 监控

  需做如下初始化，即可自动化进行 Crash 和 ANR 监控。

```
import { MPRemoteLogger } from '@mpaas/masadapter';

MPRemoteLogger.initFaultTrack();
```

- 启动卡死监控

  需做如下初始化，即可自动化监控启动卡死。

```
import { MPRemoteLogger } from '@mpaas/masadapter';

// App启动时，开启卡死监控
MPRemoteLogger.startPerformanceMonitor();
// App启动完成的时候，调用下述代码通知完成启动
MPRemoteLogger.startupFinish();
```

页面自动化监控

初始化页面自动化监控。

```
import { MPRemoteLogger } from '@mpaas/masadapter';

//在自定义的 UIAbility 里调用下述方法初始化页面自动化监控
//示例
onCreate(want: Want, launchParam: AbilityConstant.LaunchParam): void {
    //全埋点页面自动化初始化
    MPRemoteLogger.initUIAutoTrack(this.context);
}
```

可选配置

有个性化配置需求时，可以使用下述接口。

- 埋点开关

  当需要关闭埋点时，可以参考如下设置。

```
import { MPRemoteLogger } from '@mpaas/masadapter';
/**
* 关闭埋点
*/
MPRemoteLogger.enableLog(false);
```

- 强制触发上报

```
import { MPRemoteLogger } from '@mpaas/masadapter';
/**
* 强制上报（内存中还未上报的）所有的埋点 （非必要时不要使用， 批量上报埋点有网络资源消耗）
*/
MPRemoteLogger.uploadAllLog();
```

- 通用扩展字段

  设置通用的埋点扩展字段。

```
import { MPRemoteLogger } from '@mpaas/masadapter';

MPRemoteLogger.setFoundationExtend('your custom field');
```

- 保留已上报的埋点

  当需要保留已上报成功的埋点文件时，请调用下述方法。

```
import { MPRemoteLogger } from '@mpaas/masadapter';

MPRemoteLogger.reserveUploadedLog(true);
```

> **说明**
> 该方法适用于开发测试阶段调试，生产环境包不建议使用。

- 查看埋点

  当需要查看埋点时，可以在以下手机目录查看： /data/app/el2/100/base/bundleid（替换为应用的 bundleid）/haps/entry/files/mptrack 。

  未上报成功的埋点可在 logs 和 upload 目录中查看；已经上报成功的埋点可在 uploaded 目录查看（前提是开启保留已上报埋点的接口）；埋点开关配置可在 logConfig 文件查看。

7.基础分析

7.1. 数据概览

7.1.1. 数据概览介绍

数据概览从平台、版本、渠道等多个维度展示实时和历史统计数据，包括启动次数、活跃用户（设备 ID）、活跃账号（用户 ID）、新增用户、累计用户、累计账号等 App 核心指标，为您的业务决策提供数据支撑。

前置条件

数据概览中的数据都是根据客户端上报的 **报活埋点日志** 统计得出。您需要确保在接入客户端时，进行了正确的报活日志埋点。更多信息，请参见 接入 Android 或 接入 iOS。

查看数据概览

1. 登录控制台，点击 **产品与服务** > **移动开发平台 mPaaS**，选择应用。
2. 在导航栏左侧，点击 **移动分析** > **数据概览**。
3. 在右侧页面，您可以点击 **实时大盘** 或 **历史趋势** 标签，以查看实时或历史数据。

指标说明

您可以在数据概览中看到以下指标数据：

- **启动次数**：用户打开应用的总次数，包括应用关闭后或切后台超过 30 分钟后再次打开。
- **活跃用户**：登录 App 的去重设备 ID 数量。
- **活跃账号**：登录 App 的去重用户 ID 数量。
- **累计用户**：App 接入移动分析后，累计的用户数量，即累计去重的设备 ID 数量。
- **新增用户**：累计用户数量减去老用户数量。
- **累计账号**：累计的账号数量，即累计去重的用户 ID 数量。

在页面中，将鼠标移动到指标右上角的问号区域（?），您会看到该指标的描述及计算规则。

相关链接

- 实时大盘
- 历史趋势

7.1.2. 实时大盘

实时大盘展示 App 核心指标的实时分析结果，相关指标说明，参见 指标计算规则。

查看数据分析实时大盘的步骤如下：

1. 登录控制台，单击 **产品与服务** > **移动开发平台 mPaaS**，选择应用。
2. 在导航栏左侧，单击 **移动分析** > **数据概览**。
3. 在右侧页面，单击 **实时大盘** 标签，进入实时大盘页面。
4. 选择平台、版本、渠道，以查看对应的细分数据。

实时大盘分为指标概览、指标趋势、指标详细数据三个部分。

指标概览

展示各项指标实时统计数据，包含启动次数、活跃用户、活跃账号三个指标的实时数据以及日环比和周同比数据。

指标趋势

以折线图的形式展现每个小时的聚合数据，包含今日、昨日、7 日前、30 日前的指标对比曲线。

从折线图左上方的下拉列表选择指标，查看各指标的数据变化趋势。

详细数据

以表格的形式展现今日每个小时的详细数据，包含启动次数、活跃用户、活跃账号三个指标。

如需导出分析报告，单击详细数据列表右上方的 **导出** 按钮即可下载相应的 Excel 文件。

7.1.3. 历史趋势

历史趋势页面展示 App 核心指标的历史分析结果，相关指标说明，参见 指标计算规则。

查看指标数据历史趋势的步骤如下：

1. 登录控制台，单击 **产品与服务** > **移动开发平台 mPaaS** ，选择应用。
2. 在导航栏左侧，单击 **移动分析** > **数据概览** 。
3. 在右侧页面，单击 **历史趋势** 标签，进入实时大盘页面。
4. 选择平台、版本、渠道、时间段，以查看对应的细分数据。

实时大盘分为指标趋势和详细数据两个部分。

指标趋势

以折线图的形式展示选定时间段内的指标数据：

- 每个指标单独展现，您可以通过左上角下拉列表选择指标。可选的指标包括启动次数、活跃用户、活跃账号、新增用户、累计用户、累计账号。
- 选择指标数据显示的日期维度，包括按日显示、按周显示和按月显示。
  - **按日显示**：展示选定时间范围内每日的数据曲线。
  - **按周显示**：展示选定时间范围内每周的数据曲线。如果所选时间范围小于 7 日，周维度不可用。
  - **按月显示**：展示选定时间范围内每月的数据曲线。如果所选时间范围小于 30 日，月维度不可用。

详细数据

以表格的形式展现选定时间段内的详细数据，包括启动次数、活跃用户、活跃账号、新增用户、累计用户、累计账号共 6 个指标的每日、每周或每月数据。支持按不同指标排序查看数据概况。

- 单击 **导出** 以 Excel 表格的形式将这些数据导出到本地。
- 单击排序图标（  ）按当前指标进行升序或降序排列。

7.2. 行为分析

行为分析显示了用户行为相关的数据。主要分析 App 用户什么时间在哪里进行了哪些操作，通过什么渠道，用了多长时间，可帮助您了解用户的操作规律、访问路径及行为特点等信息。

查看行为分析数据的步骤如下：

1. 登录控制台，点击 **产品与服务** > **移动开发平台 mPaaS** ，选择应用。
2. 从左侧导航栏进入 **移动分析** > **基础分析** > **行为分析** 页面。
3. 筛选分析数据。在页面上方，选择平台、应用版本、渠道，并选择数据分析的日期，即可查看相应的页面分析数据。

行为分析页面通过图表展示以下行为指标数据：

- 用户活跃分析
- 热门页面 Top5
- 渠道分布
- 启动速度

- 来源去向
- 地域用户数占比

> **说明**
>
> - 将鼠标移动到指标右上角的问号区域（?），您会看到该指标的描述及计算规则。
> - 统计数据基于历史的日级汇总，数据库中记录了所有统计项历史每日的汇总结果，当日数据不计入汇总结果。
> - 更多关于指标计算规则，参考 基础指标计算规则。

用户活跃分析

通过系统字段显示的日志上报时间减去压后台时间段（duration），计算出用户开始使用 App 的具体时间。从 0 点开始，以每两小时为一个时间段（例如 2:00 ~ 4:00 am 为一个时间段），用户的活跃时间依据用户开始使用 App 的具体时间所在的时间段。

- **日活跃时长**：通过 App 的使用时长（duration）总和除以设备 ID 数量（UV），计算出用户平均每日使用 App 的时长。
- **次数 / 日·人（次）**：通过压后台的非去重数据（PV）除以压后台的去重数据（UV），计算出用户平均每日打开 App 次数。

热门页面 Top5

展示当前 App 访问量最多的前 5 个页面。点击页面名称，可跳转至该页面的分析详情页。

渠道分布

基于报活埋点，计算 App 下载次数排名前三的渠道。计算时，去重同一用户在同一渠道的多次下载。

例如，用户 A 通过 C1 渠道下载应用 1 次，通过 C2 渠道下载应用 2 次。在数据归类时，只计算通过 C1 、C2 渠道各下载 1 次。

> **说明**
>
> 要了解 Android 设备的渠道分布，您可以选择生成并上传不同的渠道包至对应的应用市场。渠道包中添加了不同渠道的字段。当用户通过应用市场下载并运行 App 后，客户端会自动上报该埋点字段到服务端。具体的渠道包生成步骤，查看 基于 mPaaS 框架 中关于设置渠道号的相关操作。

启动速度

启动速度统计首次启动和非首次启动的平均启动时间，单位为秒。

首次启动指首次安装后，第一次启动 App；非首次启动指非首次安装 App 后启动 App。

来源去向

显示当前页面的访问来源和去向。根据 pid 和 refer 字段统计当前页面的访问来源及每个访问来源占总来源数比例，和当前页面的所有去向及每个去向占总去向比例。

可以通过图表右上方的下拉框选择切换页面，查看相应的页面数据。下拉列表所展示的页面均为 **页面分析** > **页面配置** 页面中所添加的页面。

地域用户数占比

地域用户数占比报活埋点对地域字段做归类，显示地域归类的去重用户数占比最高的前五个地区。同时，分别显示前五个地区的用户数各占总用户数的比例。

- 如果是中国地图，计算按省份归类的去重用户数占比。
- 如果是世界地图，计算按国家归类的去重用户数占比。

7.3. 留存分析

留存分析展示用户黏性相关的数据，主要分析某一群组用户（通常为新用户）中再次访问 App 的人数和比例。您可以基于留存分析数据，针对流失用户有针对性地调整运营策略，为 App 带来持续流量增长。

查看留存分析数据的步骤如下：

1. 登录控制台，点击 **产品与服务** > **移动平台 mPaaS**，选择应用。
2. 从左侧导航栏进入 **移动分析** > **基础分析** > **留存分析** 页面。
3. 筛选要查看的分析数据。
   - 在页面上方，选择平台、版本、渠道。
   - 选择数据分析的时间段。
   - 选择分析数据展示维度，即按设备或账号展示。
   - 选择分析数据的展示时间粒度，可以是天、周或月。默认为天数据，时间范围默认为昨天之前的 7 天。

留存分析页面分为 留存概览 和 留存详情 两部分。

> **说明**
>
> - 将鼠标移动到指标右上角的问号区域（?），您会看到该指标的描述及计算规则。
> - 统计数据基于历史的日级汇总，数据库中记录了所有统计项历史每日的汇总结果，当日数据不计入汇总结果。欲了解更多关于指标计算规则，参考 指标计算规则。

留存概览

某日的新增用户分别在之后的七日内的每一天再次访问 App 的用户数量除以新增用户总数，结果以折线图显示。

若选择查看日数据，横坐标的 N 天后（例如 7 天后）的留存率为选定日期范围内的 N 天的平均留存率。同样，若选择查看周、月数据，N 周/月后留存率也为日期范围内的 N 周/月的平均留存率。日数据最多展示到 30 天后，周数据最多展现到 6 周后，月数据最多展现到 3 个月后；留存详情同样如此。

留存详情

某日的新增用户分别在之后的七日内的每一天再次访问 App 的用户数量除以新增用户总数，结果以表格显示。

例如，第 T 日的新增用户总数为 N。第 T+1 日用户 N 中再次访问 App 的用户数量为 M1，第 T+7 日用户 N 中再次访问 App 的用户数量为 M7，那么第 T+1 日的留存率为 M1/N，第 T+7 日的留存率为 M7/N。

7.4. 页面分析

页面分析显示了 App 中所有访问页面的情况。

查看页面分析数据的步骤如下：

1. 登录控制台，点击 **产品与服务** > **移动平台 mPaaS**，选择应用。
2. 从左侧导航栏进入 **移动分析** > **基础分析** > **页面分析** 页面。
3. 筛选分析数据。在页面上方，选择平台、版本、渠道，并选择数据分析的日期来筛选数据，也可以通过在页面右上方的搜索栏中输入页面名称关键字来搜索相关页面的分析报告。

通过浏览页面分析报告，您可以看到 App 中所有页面的分析数据。下文将通过图表示例向您展示页面分析结果。

> **说明**
>
> - 将鼠标移动到指标右上角的问号区域（?），您会看到该指标的描述及计算规则。
> - 统计数据基于历史的日级汇总，数据库中记录了所有统计项历史每日的汇总结果，当日数据不计入汇总结果。欲了解更多关于指标计算规则，参考 指标计算规则。

页面分析总览

页面分析列表展示了当前 App 下所有页面的访问情况。

- **用户数**：访问当前页面的设备 ID 去重后的数量。
- **账户数**：访问当前页面的用户 ID 去重后的数量。

> **重要**
> 如果用户未登录，账户数无法统计。

- **访问量**：访问当前页面的总次数（PV）。
- **退出率**：从当前页面退出 App 的次数占该页面总访问量的比值。退出率 =（访问当前页面的非去重总数 - 来源为当前页面的非去重总数）/访问当前页面的总次数。
- **页面平均停留时间**：通过停留时段字段得出的当前页面的所有停留时间除以访问当前页面的非去重总数。

页面分析详情

点击页面名称，可以查看每个页面的详细访问数据。

页面访问概览

展示 PV（页面的总访问量）和 UV（页面去重后的访问量），同时显示日环比和周同比数值。

页面访问趋势

显示页面在过去七天的访问量，以折线图展现页面访问趋势。

来源去向

显示当前页面的访问来源和去向。根据当前页面 ID 和 refer 字段统计当前页面的访问来源及每个访问来源占总来源数比例，和当前页面的所有去向及每个去向占总去向比例。

7.5. 设备分析

设备分析展示了 App 核心指标数据在不同设备型号下的分布情况。

查看设备分析数据的步骤如下：

1. 登录控制台，点击 **产品与服务** > **移动平台 mPaaS**，选择应用。
2. 从左侧导航栏进入 **移动分析** > **基础分析** > **设备分析** 页面。
3. 筛选分析数据。在页面上方，选择平台、版本、渠道，并选择数据分析的日期来筛选数据。

> **说明**
> 统计数据基于历史的日级汇总，数据库中记录了所有统计项历史每日的汇总结果，当日数据不计入汇总结果。如需了解指标计算规则，参考 指标计算规则。

设备分布

以柱状图的形式展现 App 各核心指标在不同设备型号下的分布情况。

每个指标单独展现，可通过左上角下拉列表选择指标。可选的指标包括启动次数、活跃用户、活跃账号、新增用户。默认展现占比前十的设备。

详细数据

以表格的形式展现 App 各核心指标在不同设备型号下的分布情况：

- 指标包括新增用户及占比、活跃用户及占比、活跃账号及占比、累计用户及占比、启动次数及占比。
- 包含所有设备型号下的数据。

7.6. 组件使用分析

7.6.1. 热修复分析

热修复用于在不发版的情况下线上修复 bug。通过热修复报告，您可以了解 RPC（远程过程调用）、修复、回滚信息。

完成以下操作，查看热修复分析报告：

1. 登录控制台，点击 **产品与服务** > **移动开发平台 mPaaS**，并选择应用。
2. 从左侧导航栏进入 **移动分析** > **组件使用分析** > **热修复分析** 页面。
3. 筛选数据。在页面上方，选择平台、应用版本、热修复版本，并选择数据分析的日期，即可查看相应的热修复分析数据。

热修复分析相关指标说明见下表：

指标	说明
RPC 开始数	通过网络发送下载补丁文件的 RPC 请求的总次数。
RPC 成功数	成功发送下载补丁文件的 RPC 请求的次数。
热修复开始数	所有设备加载补丁的次数。
热修复成功数	所有设备成功加载补丁的次数。
回滚开始数	当前应用进行回滚的次数。
回滚成功数	当前应用成功回滚到基线版本或补丁版本的次数。

7.6.2. 离线包分析

离线包是一个包含 HTML、Javascript、CSS 等页面内静态资源的压缩包。用户可预先下载离线包到本地，然后通过客户端打开，直接从本地加载离线包，从而最大程度地摆脱网络环境对 H5 页面的影响。通过离线包分析报告，您可以了解离线包的离线包下发次数、成功下载次数和使用次数。

完成以下步骤，查看离线包分析报告：

1. 登录控制台，点击 **产品与服务** > **移动开发平台 mPaaS**，选择应用。
2. 从左侧导航栏进入 **移动分析** > **组件使用分析** > **离线包分析** 页面。
3. 筛选数据。在页面上方，选择平台、应用版本、离线包或离线包版本，并选择数据分析的日期范围，即可查看相应的离线包分析数据。

指标总览

指标	说明
下发次数	指客户端成功请求到离线包更新提醒的次数。若您下载了离线包但未安装，仍会收到离线包的更新提醒。
到达次数	指客户端成功解压离线包的次数。

打开次数	指您在客户端上打开离线包的次数。

指标走势

展示指定时间范围内的离线包指标走势，支持按 **分钟**、**小时**、**日** 维度展示。

> **说明**
>
> 仅当查询的时间范围为一天时，才支持展示分钟和小时维度的数据。例如，当选择时间范围为 2020-06-01 ~ 2020-08-01 时，小时和分钟维度不可用。

详细数据

在页面下方，您可看到指定时间范围内的离线包下发次数、到达次数、打开次数。数据展示的时间粒度取决于您所选的展示时间维度。

7.6.3. 扫码分析

移动分析支持对 mPaaS 扫码组件的使用数据进行统计分析，包括调用量、调用成功量、调用耗时、调用成功率。

按以下步骤，查看扫码分析报告：

1. 登录控制台，点击 **产品与服务** > **移动开发平台 mPaaS**，选择应用。
2. 从左侧导航栏进入 **移动分析** > **组件使用分析** > **扫码分析** 页面，查看扫码统计报表。
3. 筛选数据。在页面上方，选择平台、版本，并选择数据分析的日期范围，即可查看相应的扫码分析数据。
4. 选择数据展示维度，按分钟、小时或日维度展示数据。

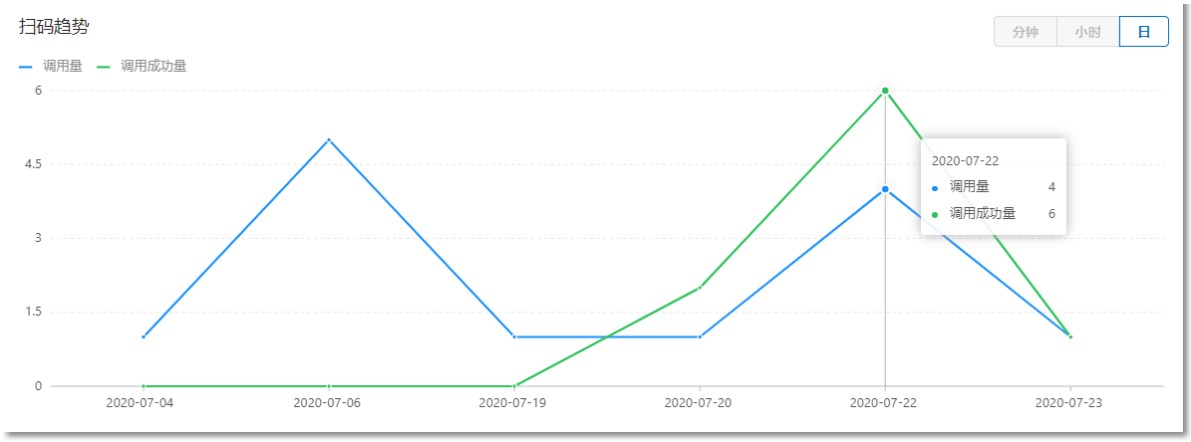
> **重要**
>
> 仅当查询的时间范围为一天时，才支持展示分钟和小时维度的数据。例如，当选择时间范围为 2020-06-01 ~ 2020-08-01 时，小时和分钟维度不可用。

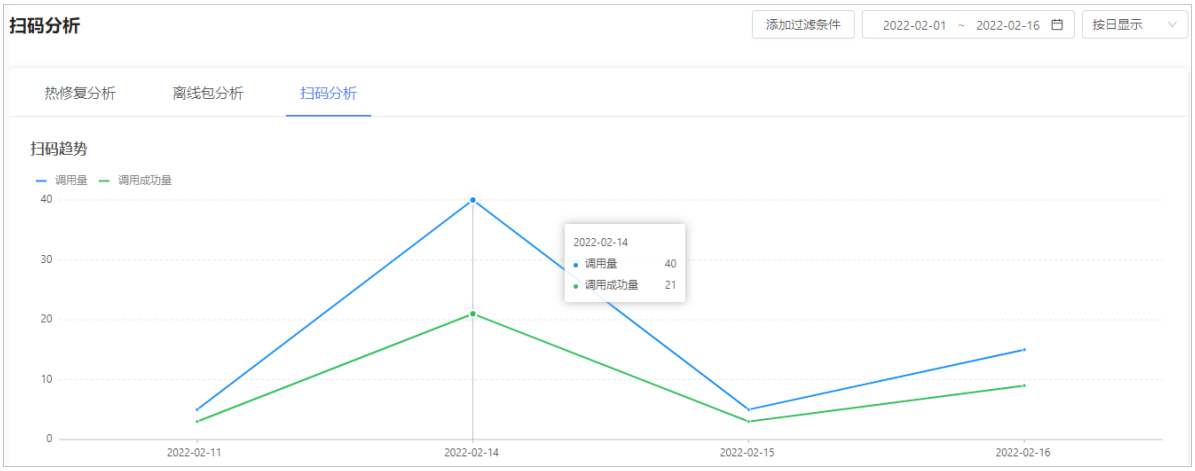
扫码趋势

展示指定时间范围内的扫码调用趋势。

- **调用量**：扫码次数，即调用扫码组件的次数。

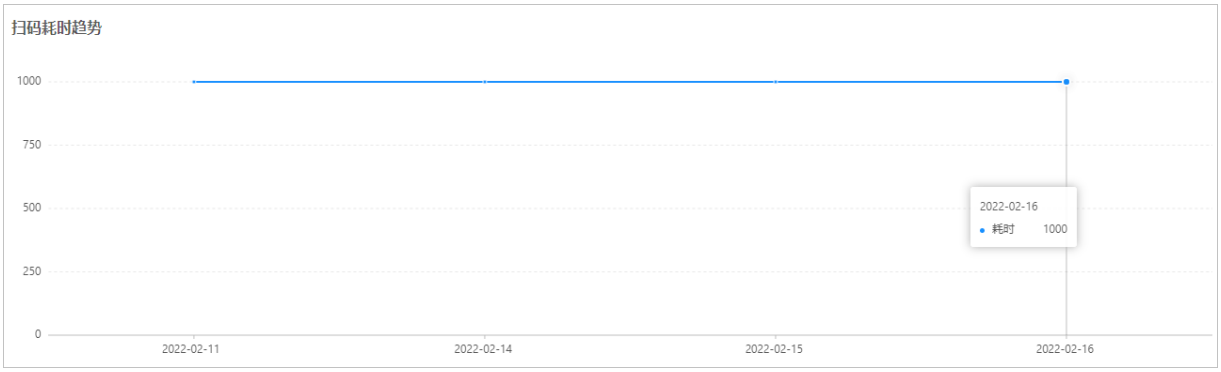
- **调用成功量**：扫码成功次数，即成功调用扫码组件完成扫码的次数。

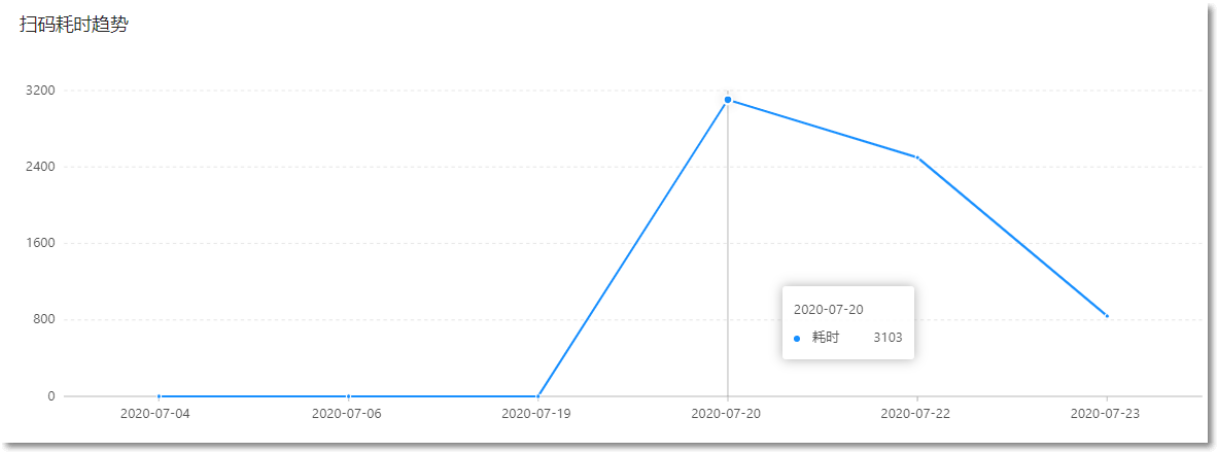




扫码耗时趋势

展示指定时间范围内的扫码耗时趋势。扫码耗时指调用扫码组件完成扫码所花的时间。在 MAS 扫码分析中，扫码耗时的时间单位为毫秒（ms）。





扫码趋势详情

展示指定时间范围内每天的扫码详细数据。数据展示的时间粒度取决于您所选的展示时间维度。

扫码趋势详情

时间	调用量	调用成功量	调用成功率	调用耗时(毫秒)
2022-02-16	15	9	60%	1000
2022-02-15	5	3	60%	1000
2022-02-14	40	21	53%	1000
2022-02-11	5	3	60%	1000

扫码趋势详情

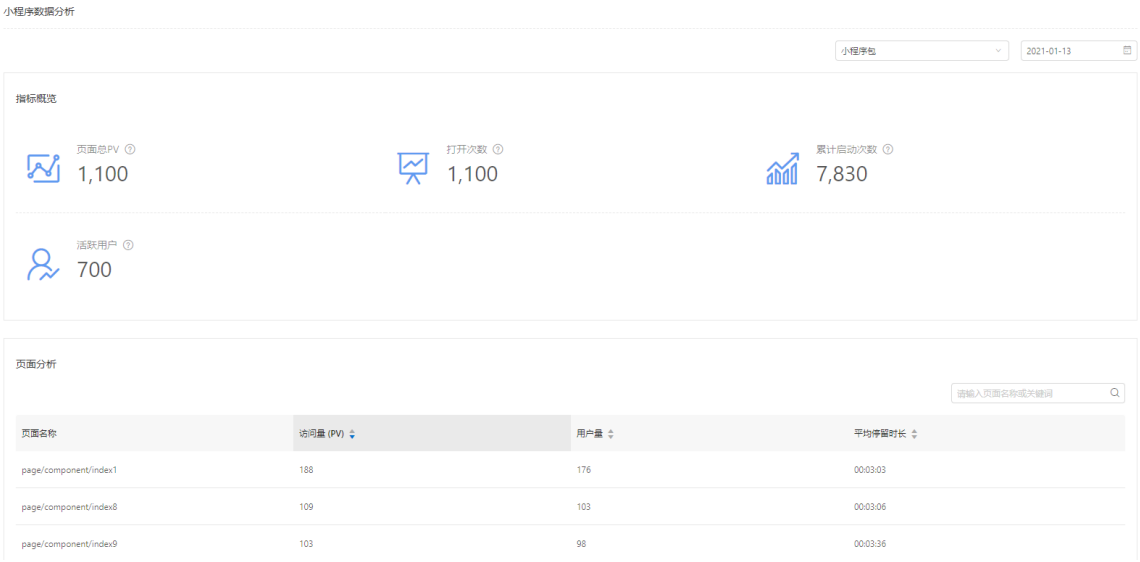
时间	调用量	调用成功量	调用成功率	调用耗时(毫秒)
2020-07-23	1	1	100%	841
2020-07-22	4	6	150%	2499
2020-07-20	1	2	200%	3103
2020-07-19	1	0	0%	0
2020-07-06	5	0	0%	0
2020-07-04	1	0	0%	0

7.6.4. 小程序分析

移动分析服务支持对当前 App 的小程序使用数据进行统计分析。您可以在小程序分析页面查看小程序基础核心指标（如页面 PV、打开次数、启动次数、活跃用户）和页面级的分析数据。

完成以下步骤，查看小程序分析报告：

1. 登录控制台，点击 **产品与服务** > **移动开发平台 mPaaS** ，选择应用。
2. 从左侧导航栏进入 **移动分析** > **组件使用分析** > **小程序分析** 页面。
3. 筛选数据。在页面右上方，选择小程序，并选择数据分析的日期，即可查看相应的小程序分析数据。



指标概览

您可以在指标概览中看到以下指标数据：

指标	说明
页面 PV	选定日期内当前小程序所有页面的访问次数之和。
打开次数	选定日期内小程序启动的次数。
累计启动次数	自小程序上线至选定日期，启动的总次数。
活跃用户	设备 ID 维度去重，选定日期内使用过小程序的总用户数

页面分析

在页面分析中，您可查看小程序不同页面的分析数据，包括访问量（PV）、用户量、平均停留时长。

默认展示所有页面的数据。同时，您也可以在右上方的搜索框中输入页面名称或关键字来搜索相关页面的分析数据。

7.7. 页面配置

页面分析展示当前 App 在指定时间的所有访问页面以及各页面的访问数据报表。为方便数据结果查询，您可以通过页面配置设置各页面在 iOS 和 Android 系统中的 ID 以及该页面 ID 在报表中对应的名称。

添加页面配置

完成以下步骤，添加页面配置：

1. 登录 mPaaS 控制台，选择应用后，在左侧的导航栏，单击 **移动分析 > 基础分析** 菜单。
2. 在右侧页面上，单击 **页面配置** 标签。
3. 单击 **添加**，添加页面信息：
   - **名称**：页面在报表中的名称。
   - **iOS**：页面在 iOS 系统中的 ID，为 VC 类名，例如 `viewControlleName` 。
   - **Android**：页面在 Android 系统中的 ID，为页面的 Activity 全路径类名，例如 `com.mpaas.demo.launcher.MainActivity` 。

     > **说明**
     > - 同一个页面在 iOS 和 Android 系统中的名称需保持一致。
     > - 每个 iOS ID 与 Android ID 均为唯一，不可重复。

修改页面配置

选择目标页面，在操作列中点击 **修改** 即可编辑页面在不同操作系统中的 ID 和页面名称。

删除页面配置

选择目标页面，在操作列中点击 **删除** 并确认即可删除该页面配置。该操作不影响页面在 **页面分析** 中的展示。

7.8. 指标计算规则

以下表格列出了移动分析报表中涉及的指标详情，包括指标名称、对应的埋点日志以及指标计算规则等。根据您的实际环境，您可能会看到以下部分或全部指标。

> **重要**
> 凡是涉及到时间的指标，均以服务端收到日志的时间点为准。

数据概览

实时大盘

指标	埋点	计算规则	是否去重	是否实时
启动次数	报活埋点	用户打开应用的总次数，包括客户端冷启动或压后台后 App 界面回到前台。	非去重	实时
活跃用户	报活埋点	登录 App 的设备 ID 数量。	去重	实时
活跃账号	报活埋点	登录 App 的用户 ID 数量。	去重	实时

历史趋势

指标	埋点	计算规则	是否去重	是否实时

指标	埋点	计算规则	是否去重	是否实时
启动次数	报活埋点	用户打开应用的总次数，包括客户端冷启动或压后台后 App 界面回到前台。	非去重	非实时
活跃用户	报活埋点	登录 App 的设备 ID 数量。	去重	非实时
累计用户	报活埋点	累计登录 App 的设备 ID 数量。	去重	非实时
新增用户	报活埋点	新增的登录 App 的设备 ID 数量。	去重	非实时
活跃账号	报活埋点	登录 App 的用户 ID 数量。	去重	非实时
累计账号	报活埋点	累计登录 App 的用户 ID 数量。	去重	非实时

基础分析

行为分析

指标	埋点	计算规则	是否去重	是否实时
用户活跃时间分布	压后台埋点	使用日志上报时间减去压后台时间段（duration），计算出用户开始使用 App 的具体时间。从 0 点开始，以每两小时为一个时间段（例如 2:00~4:00 am 为一个时间段），用户的活跃时间依据用户开始使用 App 的具体时间所在的时间段。 例如，使用该 App 的用户从 7:55 am 开始使用该 App，8:05 am 把 App 压后台，那么该用户的活跃时间分布时间只算是 6:00~8:00 am。	非去重	非实时
用户日参与度：时长/日	压后台埋点	通过 App 的使用时长总和除以用户 ID 数量，计算出用户平均每日使用 App 的时长。压后台的埋点会记录每次 App 页面展现到前台的时间，根据该时间累加统计得出 App 的使用时长。	去重	非实时
渠道分布	报活埋点	基于报活埋点，计算 App 下载次数排名前三的渠道。计算时，去重了同一用户在同一渠道的多次下载。 例如，用户 A 通过 C1 渠道下载应用 1 次，通过 C2 渠道下载应用 2 次。在数据归类时，只计算通过 C1 、C2渠道各下载 1 次。	去重	实时
启动速度	性能埋点	计算应用平均启动速度，单位是秒，区分首次启动和非首次启动。首次启动指首次安装后，第一次启动 App；非首次启动指非首次安装 App 后启动 App。	不适用	实时
来源去向	页面自动埋点	日志中会记录当前页面 ID 和来源页面 ID 等信息。据此可以统计当前页面的访问来源及每个访问来源占总来源数比例，和当前页面的所有去向及每个去向占总去向比例。	非去重	非实时
地域用户数占比	报活埋点	报活埋点对地域字段做归类，显示地域归类的去重用户数占比最高的前五个地区。同时，分别显示前五个地区的用户数各占总用户数的比例。对于中国地图，计算按省份归类的去重用户数占比；对于世界地图，计算按国家归类的去重用户数占比。 例如，中国地图中，用户某日上午在 A 省登录 2 次 App，下午在 B 省登录 1 次 App，在数据归类时，只计算用户 A、B 省各 1 次有效登录。	去重	实时

留存分析

指标	埋点	计算规则	是否去重	是否实时
留存率	报活埋点	某日的新增用户分别在之后的七日内的每一天再次访问 App 的用户数量除以新增用户总数。例如，第 T 日的新增用户数为 N，这 N 个用户在第 T+1 日再次访问 App 的数量为 M1，第 T+7 日再次访问 App 的数量为 M7，那么第 T+1 日的留存率为 M1/N，第 T+7 日的留存率为 M7/N。	去重	非实时

页面分析

指标	埋点	计算规则	是否去重	是否实时
用户数	页面自动埋点	访问当前页面的设备 ID 数量。	去重	非实时
账户数	页面自动埋点	访问当前页面的用户 ID 数量。	去重	非实时
访问量	页面自动埋点	访问当前页面的总次数。	非去重	非实时
退出率	页面自动埋点	从当前页面退出 App 的次数占该页面总访问量的比值。计算公式： 退出率 = （访问当前页面的非去重总数 - 来源为当前页面的非去重总数）/访问当前页面的非去重总次数。	非去重	非实时
页面停留时间	页面自动埋点	通过停留时长字段得出的当前页面的总停留时长除以访问当前页面的非去重总数。	非去重	非实时

设备分析

指标	埋点	计算规则	是否去重	是否实时
启动次数	报活埋点	从设备型号维度，统计用户打开应用包括客户端冷启动和压后台后 App 界面回到前台的总次数。	非去重	非实时
活跃用户	报活埋点	从设备型号维度，统计登录 App 的设备 ID 数量。	去重	非实时
新增用户	报活埋点	从设备型号维度，统计新增的登录 App 的设备 ID 数量。	去重	非实时
活跃账号	报活埋点	从设备型号维度，统计登录 App 的用户 ID 数量。	去重	非实时

8.自定义分析

8.1. 自定义大盘

8.1.1. 创建自定义大盘

大盘是将各种数据分析结果集中在一个页面上以报表形式呈现。移动分析的自定义大盘支持根据业务需要，将自定义分析的数据以不同类型的报表展示。通过查看报表上的各项指标数据和变化趋势，用户能够快速判断 App 的业务情况，并做出相应的决策。

创建自定义大盘需要添加报表，并完成相关配置。您可以通过新建自定义分析报表或者选择现有分析报表将其添加到大盘中。

类型	对象
选择已有分析报表	事件
	漏斗
新建自定义分析	事件
	漏斗

前置条件

- 添加自定义分析报表时，如果选择添加已有分析报表，则需要先通过事件分析或漏斗分析添加相应的报表；如果选择新建自定义分析，则需要事先配置好事件。
- 针对事件分析或漏斗分析中涉及的事件，已完成客户端自定义事件埋点。详细信息，参见 Android 自定义事件埋点 或 iOS 自定义事件埋点。

操作步骤

登录 mPaaS 控制台，完成以下操作。本操作以新建自定义分析为例：

1. 点击左侧导航栏的 **移动分析** > **自定义分析**。
2. 在右侧页面，点击 **自定义大盘** > **添加大盘**。
3. 在 **添加大盘** 窗口，输入大盘名称，选择模板后，点击 **确定**。

> **重要**
> 当前仅支持空白模板。

4. 添加自定义分析报表。在新建的大盘中，单击 **添加自定义分析**，选择报表添加方式：**新建自定义分析** 或 **添加已有分析模块**。
   - **新建自定义分析**：
     a. 选择分析报表类型，并根据所选报表类型，完成相应的自定义分析配置。
        - 如果选择添加的对象是 **事件分析**，则完成事件分析相关的配置。具体步骤，参考 添加事件分析。
        - 如果选择添加的对象是 **漏斗分析**，完成漏斗分析相关的配置。具体步骤，参见 创建漏斗。
     b. 配置完事件分析或漏斗分析后，单击 **提交** 按钮，将该分析报表添加至大盘。

- **添加已有分析模块**：
  a. 选择事件或漏斗分析模块。
  b. 从列表中选择要添加到大盘的事件分析或漏斗分析报表，单击 **提交** 按钮，将该分析报表添加至大盘。

在 **自定义大盘** 标签页，您可以看到添加的大盘。点击该大盘，您将看到已添加的自定义分析。

默认情况下，自定义分析报表显示的是最近 4 天的数据。

后续操作

根据业务需求 管理自定义大盘，例如调整大盘布局、修改大盘中的报表等。

8.1.2. 管理自定义大盘

管理已创建的自定义分析大盘。

大盘管理操作包括：

- 调整大盘页面布局
- 筛选报表数据
- 修改大盘中的报表
- 删除大盘中的报表
- 修改大盘
- 删除大盘

调整大盘页面布局

登录 mPaaS 控制台，选择目标应用后，完成以下步骤：

1. 从左侧导航栏进入 **移动分析** > **自定义分析** 页面，然后单击右侧的 **自定义大盘** 标签。
2. 在自定义大盘页面上，单击要调整页面布局的大盘名称，进入大盘。
3. 在大盘中，单击右上角的 **编辑布局**，使用鼠标拖拽该页面的分析报表位置。
4. 调整完成后，单击右上角的 **保存布局**。

筛选报表数据

1. 在 **自定义大盘** 页面上，单击要进行分析数据筛选的大盘名称，进入大盘。
2. 在 **添加自定义分析** 按钮下方，选择分析时间段、聚合维度进行简单的数据筛选。如需进一步筛选数据，可单击 **高级过滤**，设置过滤条件。

   筛选后的自定义分析报表结果在当前页面显示。时间聚合维度可以是分钟、小时、天、周或月，具体因所选时间段而异。

修改大盘中的报表

每创建一张事件分析报表都会相应生成一张卡片。如要对已创建的事件分析进行配置修改，您可以在卡片管理标签页中进行统一操作。

除了在 **卡片管理** 标签页中修改自定义分析，您还可以通过 **自定义大盘** 标签页，单击想要修改的大盘，通过自定义报表右上方的修改图标修改配置。

> **重要**
> 您无法修改漏斗分析报表。

前置条件

- 已经创建过事件分析或漏斗分析。
- 确认拥有修改自定义分析的权限。

操作步骤

1. 单击左侧导航栏的 **移动分析** > **自定义分析**。
2. 在右侧页面上，单击 **自定义配置** > **卡片管理** 标签。
3. 选择要修改的自定义分析的指标名称，单击操作栏的 **修改**。
4. 在 **修改自定义分析** 页面修改配置。
5. 修改完毕后，单击 **提交**。

删除大盘中的报表

您可通过以下两种方式来删除自定义分析报表。

删除事件分析报表

- 通过 **卡片管理** 标签页：
  i. 单击左侧导航栏的 **移动分析** > **自定义分析**。
  ii. 在右侧页面上，单击 **自定义配置** > **卡片管理** 标签。
  iii. 选择要删除的自定义分析的指标名称，单击操作栏的 **删除** 后确认即可。
- 通过 **自定义大盘** 标签页：
  i. 单击左侧导航栏的 **移动分析** > **自定义分析**。
  ii. 在右侧页面，单击 **自定义大盘** 标签。
  iii. 选择要删除的事件报表所在的大盘名称。
  iv. 选择要删除的事件报表，单击报表右上方的删除图标后确认即可。

删除漏斗分析报表

您可通过以下两种方式来删除漏斗分析报表。

- 通过 **漏斗分析** 标签页：
  i. 单击左侧导航栏的 **移动分析** > **自定义分析**。
  ii. 在右侧页面，单击 **漏斗分析** 标签。
  iii. 选择要删除的漏斗，单击操作栏的 **删除** 后确认即可。
- 通过 **自定义大盘** 标签页：
  i. 单击左侧导航栏的 **移动分析** > **自定义分析**。
  ii. 在右侧页面，单击 **自定义大盘** 标签。
  iii. 选择要删除的漏斗报表所在的大盘名称。
  iv. 选择要删除的漏斗报表，单击报表右上方的删除图标后确认即可。

修改大盘

在大盘页面上，单击页面左上方大盘名称旁的编辑图标（  ），修改大盘名称。

删除大盘

删除不需要的大盘。

在自定义大盘页面，选择要删除的大盘，单击卡片上的 **删除** 菜单并确认删除即可。

8.2. 分群管理

8.2.1. 用户分群介绍

自定义用户分群以用户的视角划分用户群体，您可以以具备某几种特征的用户（who）在某段时间（when）做了某件事（what）为条件，将用户按需划分为不同的群组。

移动分析服务（MAS）提供基于事件创建人群的能力，生产的人群可供智能投放等组件直接调用或导出供其它组件使用。

应用场景

- 通过用户分群功能，筛选出您所需要的特定用户群体，并用于精准投放、精准营销、个性化推送等场景，以满足不同用户的不同需求。
- 基于用户分群，直接分析某个群体在基础、高阶分析功能里的数据表现，让运营、营销相关的分析更加精细化。

8.2.2. 创建用户群组

用户分群指将用户按照行为特征划分为不同的群组，帮助您更好地分析不同群组的人群属性、行为特点及用户在关键路径转化中的差异，帮助运营人员更好地发掘产品问题的背后原因，从而进行有效的改进及优化。

下面介绍如何创建用户群组。

前置条件

您已在 **自定义分析** > **自定义配置** 中创建了相关事件，详情参见 配置事件。

操作步骤

用户群组的创建步骤如下：

1. 登录 mPaaS 控制台，选择应用后，从左侧导航栏进入 **移动分析** > **自定义分析** 页面。
2. 在右侧页面，点击 **用户分群** 标签。
3. 点击页面左上方的 **创建群组** 进入创建用户群组页面。
4. 在页面中配置人群信息：
   - **群组名称**：用户群组名称，支持任意字符，不超过 20 个字符。
   - **群组描述**：输入该用户群组的简单介绍，支持任意字符，不超过 100 个字符。
   - **计算方式**：可选择 **单次** 或 **每日例行**。
     - **单次**：每次创建或编辑群组时手动执行一次分群计算。
     - **每日例行**：每日凌晨自动执行一次分群计算。
   - **用户维度**：分群统计用户的维度，直接影响到群组导出的内容，可选择 **用户 ID** 或 **设备 ID**。
   - **群组用户事件**：最多可添加 10 个事件。可对每个事件单独选择发生时间段、发生次数及其他属性筛选条件。
     - 在 **时间** 选择框中选择您想筛选的时间段。
       - 当 **计算方式** 为 **单次** 时，可选择任意时间段。
       - 当 **计算方式** 为 **每日例行** 时，只可选择 **昨日**、**过去 7 日**、**过去 14 日**、**过去 30 日** 这几个固定时间段。
     - 在 **请选择事件** 中选择您在 **自定义配置** 中创建过的事件。
   - **人群标签**：对群组中的用户打上标签。仅支持对按用户 ID 维度统计的分群添加标签。

     > **说明**
     > - 单次群组：每日最多允许创建 10 个单次群组。
     > - 每日例行群组：单个 App 最多共可创建 10 个每日例行群组。
5. 点击 **提交** 完成创建。提交后，当前分群任务开始以离线任务形式计算，最长约 2 小时内产出计算结果。

8.2.3. 管理用户群组

已创建的用户群组将以列表形式展示在分群管理页面中。您可以对已创建的群组进行管理，包括查找、删除、编辑用户分群，导出群组用户信息，以及查看群组内的用户行为轨迹和用户详情。

查看群组

登录 mPaaS 控制台后，选择目标应用后，从左侧导航栏进入 **移动分析** > **自定义分析** > **用户分群** 页面查看用户分群信息。

用户群组列表默认展示所有计算方式的群组，您可以通过页面右上方的计算方式下拉列表，选择展示 **全部**、**单次** 或 **每日例行** 群组，也可以在输入框中输入关键字进行查找。

群组信息包括：

- **群组名称**：点击群组名称可进入群组详情页，查看群组详情，也可以编辑群组信息。编辑筛选条件后，会重新进行计算。
- **用户数**：对应群组下的用户数量。
  - 若计算方式为 **单次**：
    - 群组计算完成后显示具体用户数。
    - 未计算完成时显示 **计算中...**。
  - 若计算方式为 **每日例行**：
    - 如果已经计算过至少 1 次，则展现最近一次的计算结果；下次计算完成后，直接更新用户数及时间。
    - 如果第一次计算还未完成，则显示 **计算中...**。
- **用户维度**：分群统计用户的维度，用户 ID 或设备 ID。
- **更新时间**：展示最近一次计算的时间。如果最近一次计算还未完成，则显示 —。
- **计算方式**：分群统计的计算方式。
  - **单次**：群组只会计算一次，群组创建后会马上进入计算队列，最长约 2 小时内产出计算结果。
  - **每日例行**：群组每天会例行计算一次，群组创建后首次计算会马上进入计算队列，最长约 2 小时产出计算结果，后续每日凌晨产出最新结果。

编辑群组

在群组列表中，点击目标群组名称，进入群组详情页后，点击 **编辑** 修改群组信息，修改完毕后，点击 **提交** 即可。

支持修改群组名称、描述以及群组用户事件，用户维度与计算方式不可修改。重新编辑后，会马上进入计算队列，最长约 2 小时内产出计算结果。

删除群组

在群组列表中，选择目标群组，点击 **操作** 列下的 **删除**，并确认删除即可。删除群组后，正在计算的任务也将停止。

导出群组用户信息

支持以 Excel 文件的形式导出群组对应的用户 ID 或设备 ID 信息。分群用户量导出上限为 100 万条。

在群组列表中，选择目标群组，点击 **操作** 列下的 **导出** 即可。

> **重要**
>
> 正在计算的分群无法导出用户信息。

查看用户轨迹/用户详情

在群组列表中，点击目标群组对应的用户数，跳转至相应的群组用户详情页，该页面展示群组内的详细用户/设备 ID 列表，支持按用户/设备 ID 搜索目标用户。

- **查看用户轨迹**

  在群组详情页中，点击目标用户/设备 ID 右侧的 **查看用户轨迹** 链接后跳转至 **轨迹分析** 页面，然后按当前 ID 查询用户的行为轨迹分析结果。更多信息，参见 轨迹分析。

- **查看用户详情**

  点击 **查看用户详情** 链接后跳转至 **日志回放** 页面，然后按当前 ID 搜索用户相关历史日志。更多信息，参见 查询历史日志。

8.3. 关于自定义分析

自定义分析在 mPaaS 自定义埋点的基础上，通过联机分析处理（OLAP）对客户端自定义事件进行多维即时自定义分析，以满足云上不同业务的特殊需求，同时能在前端根据定义的不同查询条件、查询场景生成保存不同的报表。

多维即时自定义分析指基于客户端自定义埋点模型的即时分析，主要包括以下部分：

- 事件元数据配置管理
- 自定义分析模型
- 自定义分析模式
- 前端自定义分析报表

其中：

- **事件元数据配置管理** ：维护当前 App 需要即时分析的各种指标数据，是分析的基础，也是 OLAP 数据库建表、查询的依据。
- **自定义分析模型** 和 **自定义分析模式** ：共同代表一种特定类型的自定义分析功能，例如事件分析、漏斗分析等，每种模型都有自己特定的查询模式。
- **前端自定义分析报表** ：需要根据各个不同的模型及查询条件产生不同的前端展示报表，方便用户分析处理。

您可以创建自定义大盘将自定义分析的结果以不同类型的报表展示。目前的版本支持以下类型的分析：

- **自定义事件分析** ：可以根据自己的业务情况上报自定义分析事件，mPaaS 提供强大的 OLAP 引擎可以多维度分析自定义事件。
- **行为分析** ：mPaaS 行为分析封装了众多的通用行为分析指标，比如留存率、用户访问趋势、设备访问趋势、设备占比、版本占比、热点页面、用户访问时间分布等。

8.4. 配置自定义分析

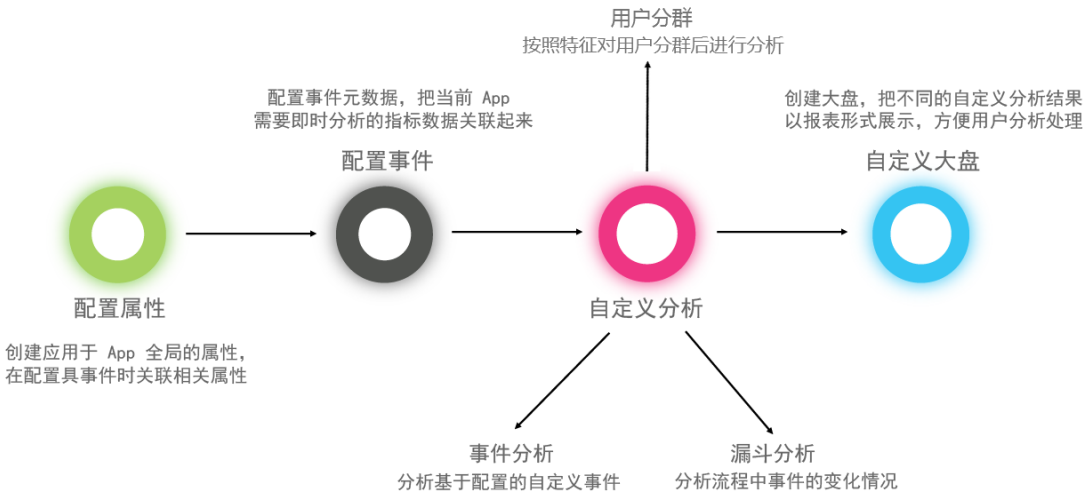
要通过 mPaaS 移动分析进行自定义分析，您需要先完成客户端自定义事件埋点，然后登录 mPaaS 控制台，并且完成相关操作。

前置条件

您已完成客户端自定义事件埋点。详细信息，参见 Android 自定义事件埋点 或 iOS 自定义事件埋点。

关于此任务

以下流程图显示了在 mPaaS 控制台进行自定义分析的步骤：



1. 配置属性：创建应用于 App 全局的属性，在配置具体事件时关联相关属性。
2. 配置事件：配置事件元数据，把当前 App 需要即时分析的指标数据关联起来。
3. 自定义分析：
   - 事件分析：分析基于配置的自定义事件。
   - 漏斗分析：分析流程中事件的变化情况。
   - 用户分群：分析不同特征的用户群体。
4. 自定义大盘：创建大盘，把不同的自定义分析结果以报表形式展示，方便用户分析处理。

8.5. 配置属性

一个事件包含一些信息，如触发事件的用户 ID、App 版本、设备型号等，这些信息即为属性。移动分析平台预置了一些常用属性，同时支持根据业务需要自定义一些应用于 App 全局的属性，以便在配置具体分析事件时进行关联。

创建属性

操作步骤如下：

1. 登录 mPaaS 控制台，从左侧导航栏进入 **移动分析** > **自定义分析** 页面，然后单击 **自定义配置** 标签。
2. 在 **自定义配置** 下方，切换到 **属性** 标签，单击 **新建**。
3. 在 **新建属性** 窗口，完成以下属性的配置：

字段	是否必填	描述
属性名称	是	属性 ID 对应的中文名称，可自定义。
属性 ID	是	属性 ID，与客户端埋点的对应关系，请参见 Android 自定义事件埋点 extParam 参数 或 iOS 自定义事件埋点 extParams 参数 的说明。

数据类型	是	从下拉菜单中选择当前属性对应的数据类型。可选的数据类型包括字符型、整型、浮点型。 ◦ 在事件分析中，可以统计字符型属性的去重次数，计算浮点型和整型属性的总和、平均值、最大值或最小值。 ◦ 对于浮点型或整型属性，客户端上报的数据会进行相应的类型转换。
单位	否	填写当前属性的单位。
是否可枚举	是	设置属性值是否可枚举。
显示状态	是	设置是否显示当前属性，默认开启。

4. 单击 **确定** 完成创建。属性列表将展示新增的属性信息。

修改属性

在属性列表中，选择目标属性，单击操作列下的 **修改** 菜单，修改属性配置。

删除属性

在属性列表中，选择目标属性，单击操作列下的 **删除** 菜单，删除属性。

> **重要**
> - 预置属性不支持删除。
> - 如果属性已关联了事件，则相关事件中将删除该属性；如果属性已关联了大盘，则该属性无法删除。

后续操作

配置事件

8.6. 配置事件

事件是对用户使用 App 过程中的互动行为的描述，例如点击注册、加入购物车、完成付款等，描述用户在什么时间、什么位置做了什么样的事。

由于自定义事件分析是基于客户端自定义埋点多维度即时组合进行的用户事件分析，在自定义事件分析前，您需要配置事件元数据，将当前 App 需要即时分析的指标数据关联起来。

创建事件

操作步骤如下：

1. 登录 mPaaS 控制台，从左侧导航栏进入 **移动分析 > 自定义分析** 页面，然后单击 **自定义配置** 标签。
2. 在 **自定义配置** 下方，切换到 **事件** 标签，单击 **新建**。
3. 在 **新建事件** 窗口，完成以下事件的配置：
   - **事件 ID**：必填。与客户端埋点的对应关系，请参见 Android 自定义事件埋点 seedID 参数 或 iOS 自定义事件埋点 seed 参数 的说明。
   - **显示状态**：是否在事件分析列表中显示该事件。如果选择 **不显示**，则该事件将不会在事件分析列表中展示。
   - **事件名称**：输入新建事件的名称。
4. 在 **属性管理** 栏，单击 **添加**，在弹出的 **添加属性** 窗口中，勾选您要关联的一个或多个指标数据，即属性，然后单击 **确定**。

> **说明**
> 所有的平台预置属性都会自动添加，您无需额外操作。

5. 返回 **新建事件** 页面的 **属性管理** 栏，查看并确认属性信息，单击 **提交**。

创建后的事件及其信息将展示在事件列表中。配置完成后，一旦有对应事件上报，该事件会出现在 **事件分析** 页面。

管理事件

在事件列表页面，您可对事件进行如下操作：

- 更改显示状态：可通过滑动开关，手动开启或关闭该事件的显示状态。
- 查看事件详情：单击事件名称，进入事件详情页，查看事件的基本信息与关联属性。
- 修改事件信息：单击事件名称，进入事件详情页，可修改事件名称，添加或删除关联属性。
- 删除事件：在事件右侧的 **操作** 列中，单击 **删除**，并确认删除操作即可。

后续操作

- 在 **事件分析** 标签页，添加事件分析
- 在 **漏斗分析** 标签页，添加漏斗。

8.7. 事件分析

8.7.1. 添加事件分析

事件分析页面展示了在事件配置页面中配置的且上报了的事件。要添加自定义的事件分析报表，在 **事件分析** 标签页选择想要分析的事件，并完成相关配置。

操作步骤

登录 mPaaS 控制台，完成以下操作添加自定义的事件分析报表：

1. 单击左侧导航栏的 **移动分析** > **自定义分析** 菜单。
2. 在右侧的 **事件分析** 标签页中，单击要添加自定义分析的事件名称，进入事件分析详情页面。
3. 单击 **添加自定义分析** 按钮，在当前事件的 **添加自定义分析** 页面，完成以下配置：
   - **分析名称**：填写该自定义分析的名称。
   - **显示指标**：选择当前事件要在报表上展示的属性。可以为该显示指标添加过滤条件，具体操作如下：
     a. 点击 **展开过滤规则**，并点击 **+ AND** 开始添加过滤规则。
     b. 配置事件的参数以及相应的过滤条件后，点击 **保存**。
   - **对比指标**：选择与当前事件对比的其他事件以及对比指标，对比的数据会在报表中显示。可以为该对比指标添加过滤条件，具体操作如下：
     a. 点击 **展开过滤规则**，并点击 **+ AND** 开始添加过滤规则。
     b. 配置事件的参数以及相应的过滤条件后，点击 **保存**。
   - **聚合指标**：选择上述指标聚合的维度。您可以添加多个聚合规则。
     聚合规则相当于 SQL 中的 `Group by` 语句，作用是将一个数据集划分成若干个小的区域，然后针对若干个小区域进行数据处理。例如日活指标，如果按照 **机型** 的聚合规则，那么同一类机型对应的是一个日活曲线。
   - **展示方式**：选择当前事件分析的展示报表类型。

4. 单击 **确定** 完成配置。
5. 如要对当前事件添加多种分析报表，重复步骤 3 ～ 4。

在 **事件分析** 标签页，点击该事件名称，您将看到该事件的分析报表。每创建完一张事件分析报表，就会生成一张卡片，这些卡片在 **自定义分析** > **自定义配置** > **卡片管理** 页面中统一展示。

新建事件后等待一分钟，客户端上报相关日志后，您就能看到分析报表，但报表中不包含事件创建前的数据。

相关操作

根据您的业务需求，管理已添加的自定义分析。具体操作，参见 管理自定义分析。

8.7.2. 管理事件分析

您可以管理已创建的事件分析。

管理操作包括以下内容：

- 查看事件
- 调整页面布局
- 筛选事件数据
- 修改事件分析报表
- 删除事件分析报表

查看事件

事件分析列表展示了在 **自定义配置** > **事件** 页面中配置的且上报了的事件。

查看上报事件的步骤如下：

1. 登录 mPaaS 控制台，选择目标应用后，从左侧导航栏进入 **移动分析** > **自定义分析** > **事件分析** 页面。
2. 在事件分析列表的上方，选择查询时间范围，单击 **高级过滤** 添加过滤条件并保存。下方的事件列表将展示查询的上报事件的访问数据。

   或者，在页面右上方的搜索框中输入事件名称搜索事件。

调整页面布局

操作步骤如下：

1. 从左侧导航栏进入 **移动分析** > **自定义分析** > **事件分析** 页面。
2. 在事件分析列表中，选择目标事件，单击事件名称，进入事件分析详情页面。
3. 单击右上角的 **编辑布局**，使用鼠标拖拽该页面的分析报表位置。
4. 调整完成后，单击右上角的 **保存布局**。

筛选事件数据

在事件分析详情页面，选择分析时间段、时间聚合维度（月、周、天、小时，可选的聚合维度因时间段长短而异）进行简单的数据筛选。

如需进一步筛选数据，可单击 **高级过滤**，设置过滤条件。 筛选后的自定义分析报表结果在当前页面显示。

修改事件分析报表

每创建完一张事件分析报表后，就会生成一张卡片。如要对已创建的事件分析进行配置修改，您可以在卡片管理标签页进行统一操作。

除了在 **卡片管理** 标签页中修改事件分析报表，您还可以在 **事件分析** 标签页，单击事件分析报表右上方的编辑图标修改配置。

- 通过 **卡片管理** 标签页：
  i. 从左侧导航栏进入 **移动分析** > **自定义分析** > **自定义配置** 页面。
  ii. 单击 **卡片管理** 标签。
  iii. 选择要修改的自定义分析的指标名称，单击操作栏的 **修改**。
  iv. 在 **修改自定义分析** 页面修改配置。
  v. 修改完毕后，单击 **提交** 即可。
- 通过 **事件分析** 标签页：
  i. 从左侧导航栏进入 **移动分析** > **自定义分析** > **事件分析** 页面。
  ii. 选择要修改的报表所在的事件，进入事件详情页。
  iii. 选择要修改的事件报表，单击报表右上方的编辑图标，修改配置后，单击 **提交** 即可。

删除事件分析报表

支持删除不需要的事件分析报表。

- 通过 **卡片管理** 标签页：
  i. 从左侧导航栏进入 **移动分析** > **自定义分析** > **自定义配置** 页面。
  ii. 单击 **卡片管理** 标签。
  iii. 选择要删除的自定义分析的指标名称，单击操作栏的 **删除** 后确认即可。
- 通过 **事件分析** 标签页：
  i. 从左侧导航栏进入 **移动分析** > **自定义分析** > **事件分析** 页面。
  ii. 选择要删除的报表所在的事件，进入事件详情页。
  iii. 选择要删除的事件报表，单击报表右上方的删除图标后确认删除即可。

8.8. 漏斗分析

8.8.1. 漏斗分析简介

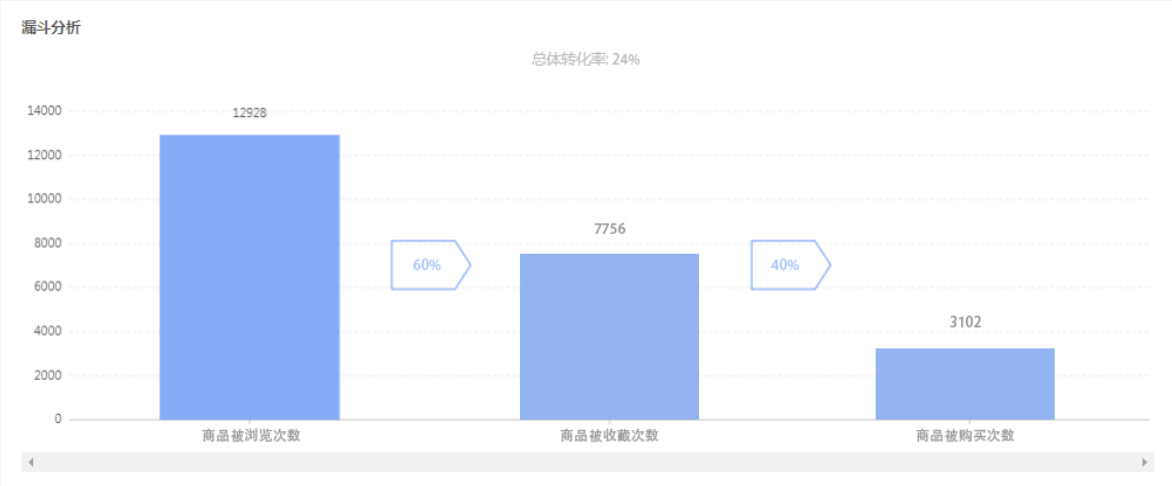
漏斗分析是指通过漏斗图等方式展示某个特定流程中事件的变化情况，主要用于统计和计算转化率等关键数据，供数据和业务人员分析以确认和改善企业的运营和营销策略。

例如，用户通过某个 App 进行购物的整体行为可以大致分为搜索产品、浏览产品、加入购物车、付款等步骤。为了分析用户群体在搜索产品开始后的一定时间内最终到达付款步骤的分步转化率，您可以通过创建漏斗，添加步骤实现数据分析，并以漏斗图和转化率趋势折线图展示。

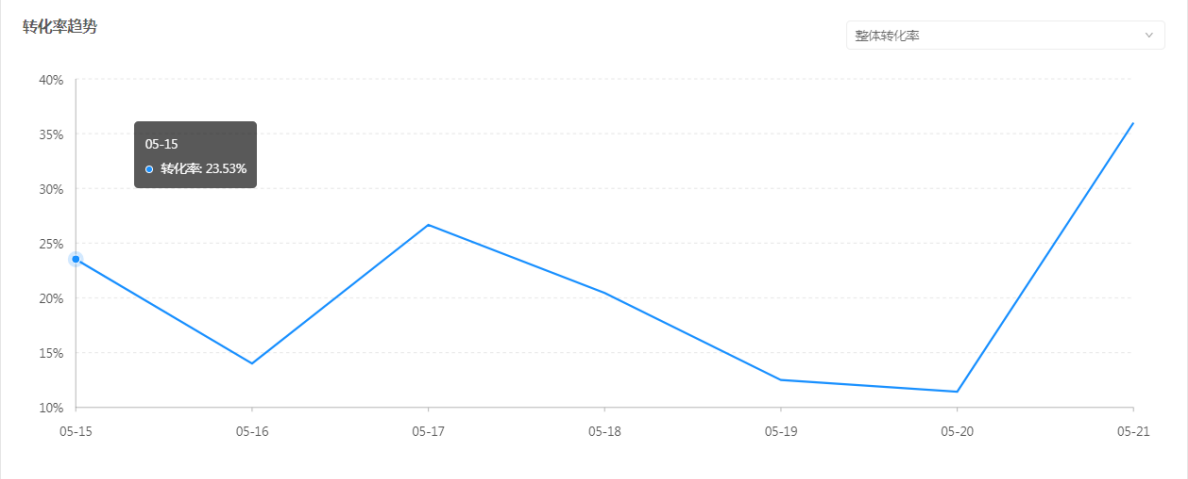
- 漏斗图展示了多个步骤之间的页面访问去重数据以及步骤之间的转化率。

  总体转化率 = 漏斗图中最后一步的数量 ÷ 第一步的数量

  下面的示例漏斗图显示了浏览、收藏到购买商品这三个步骤之间的页面访问去重数据，以及各个步骤之间的转化率。



- 转化率趋势折线图展示了在特定时间段内流程中事件的转化率。



- 详细数据表展示了当前漏斗事件在各个设备类型上的发生次数以及漏斗的具体转化数据。

8.8.2. 创建漏斗

漏斗模型主要用于分析一个多步骤过程中每一步的转化与流失情况，帮助您定位用户流失环节，发现流失原因，进而通过产品优化或者运营活动提升用户转化率。

本文介绍如何创建漏斗，在漏斗中添加步骤和数据计算的时间窗口及维度。

前置条件

漏斗分析通过漏斗图的方式展示某个特定流程中事件的变化情况。因此，在创建漏斗前，您必须事先配置好漏斗下关联的事件。关于配置事件的步骤，请参见 配置事件。

关于此任务

- 由于漏斗分析计算的是步骤之间的转化率，因此创建漏斗时，至少配置两个步骤。每一个步骤都由一个事件加一个或多个触发条件组成。
- 创建漏斗后，漏斗数据在 T+1 天后才显示。

操作步骤

登录 mPaaS 控制台，完成以下步骤：

1. 从左侧导航进入 **移动分析** > **自定义分析** > **漏斗分析** 页面。
2. 单击页面右上方的 **创建漏斗** 按钮，在漏斗创建页面，完成以下配置：
   - **漏斗名称**：必填，填写漏斗名称。
   - **漏斗描述**：选填，填写漏斗描述。
   - **时间窗口**：必填，选择漏斗下所有步骤的完成时间周期。

     > **说明**
     > 当用户触发第一个步骤之后，在指定的时间窗口内完成所有步骤，转化就算达成。

   - **计算维度**：必填，可选择 **用户 ID** 或者 **设备 ID**。

     > **说明**
     > 选择漏斗的数据统计维度，数据去重。

   - **第 1 步**：必填，从已配置的事件列表中选择第 1 步事件，并添加触发条件。
   - **第 2 步**：必填，从已配置的事件列表中选择第 2 步事件，并添加触发条件。

     > **说明**
     > 如果要在漏斗下关联更多的步骤，您可点击 **添加步骤**，并从已配置的事件列表中选择第 3 步或更多步事件。

3. 点击 **提交** 完成漏斗创建。

漏斗数据将会在 T+1 天后显示。在 **漏斗分析** 页面，点击操作栏下的 **详情** 查看漏斗图和转化率折线图。

8.8.3. 管理漏斗

您可以对已创建的漏斗进行管理。管理操作包括查看漏斗设置、查看并筛选漏斗数据和删除漏斗。

关于此任务

已创建的漏斗无法修改。

查看漏斗设置

登录 mPaaS 控制台，完成以下操作：

1. 从左侧导航进入 **移动分析** > **自定义分析** > **漏斗分析** 页面。
2. 在漏斗列表中，选择要查看的漏斗，点击操作栏下的 **详情**。在打开的页面，您可以看到当前漏斗的设置，包括时间窗口、计算维度以及各步骤的事件等。

查看并筛选漏斗数据

1. 在漏斗分析页面，选择要查看的漏斗，在打开的页面，可以看到漏斗转化分析图、转化率趋势图和详细数据。

2. 在漏斗列表中，选择要查看的漏斗名称，在打开的页面，您可以看到漏斗分析图和转化率趋势图。
3. 对漏斗分析报表展示的数据，进行过滤展示。
   - 在页面上方的日期栏，选择数据统计时间段。例如，您选择 2022-01-10 ~ 2022-01-13，那么该时间段内，只要发生过第 1 步事件的数据都会被统计。
   - 点击日期栏右侧的 **高级过滤**，选择 **参数**、**规则** 并输入 **数值** 后，**保存** 即可。还可以点击 **+ OR** 或 **+ AND** 来增加更多筛选条件。

删除漏斗

在漏斗分析页面的漏斗列表中，选择要删除的漏斗，点击操作栏下的 **删除** 并确认即可。

> **重要**
>
> 删除漏斗将会删除该漏斗下关联的所有报表。

8.9. 轨迹分析

移动分析服务结合自动化埋点，通过实时计算，将用户在 App 内发生的行为串联，按时序（客户端时间）排列，形成用户行为轨迹。通过查看用户行为轨迹，可以了解用户在 App 内的操作行为路径，例如何时启动了 App，浏览了哪些页面，以及在页面内进行了哪些操作。

轨迹分析展示具体某个用户在一段时间内的行为轨迹，各个轨迹节点按照日志埋点类型展示行为相关信息，包括行为发生的时间、平台、系统版本、渠道、设备（型号）以及发生的行为等。支持按用户 ID 或设备 ID 查询用户的行为轨迹。

查看轨迹分析报告

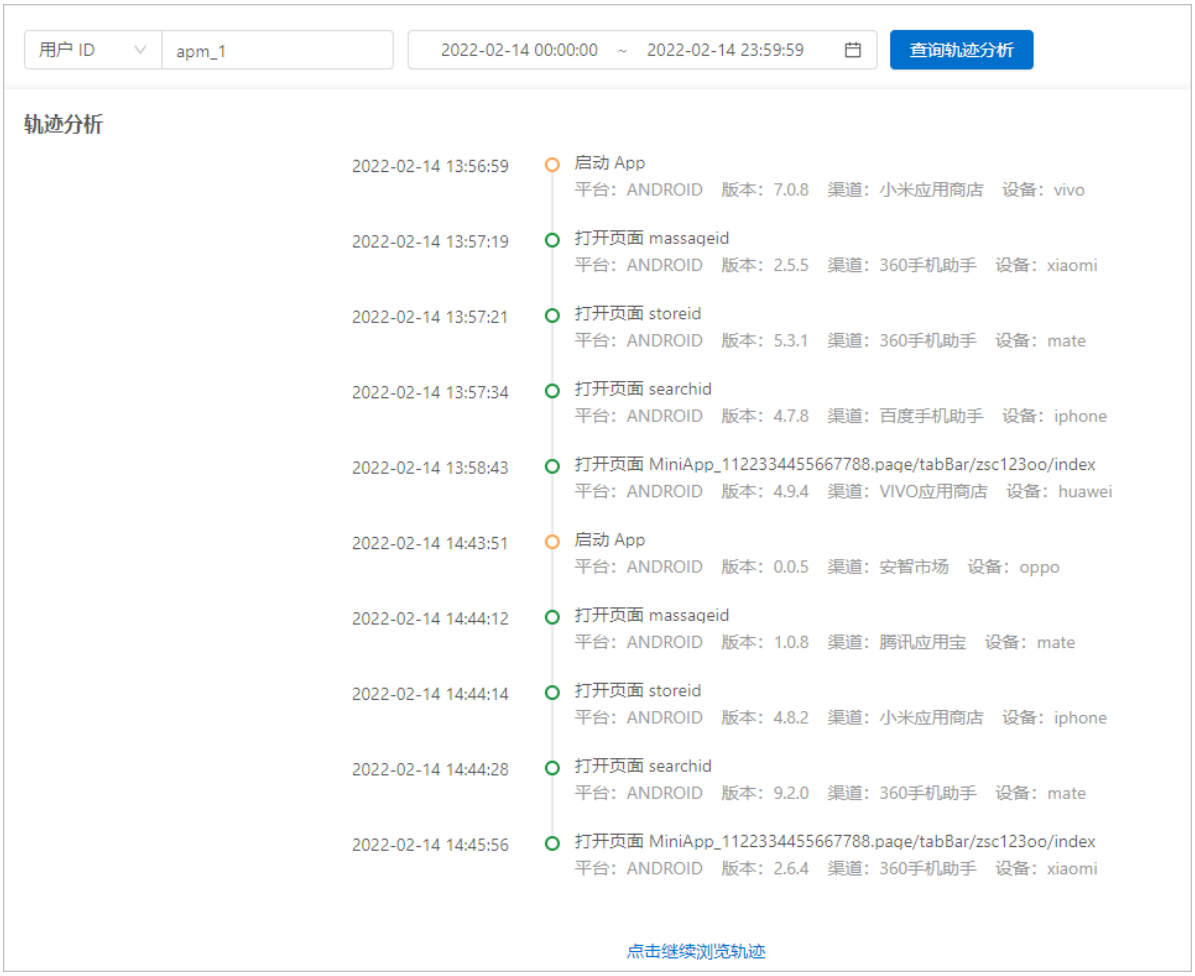
按照以下步骤查看轨迹分析报告：

1. 登录 mPaaS 控制台，选择目标应用，从左侧导航栏进入 **移动分析** > **自定义分析** > **轨迹分析** 页面。
2. 选择查询维度 **用户 ID** 或 **设备 ID**，输入相应的用户或设备 ID ，并选择时间段后，点击 **查询轨迹分析** 即可查看相应的用户轨迹结果。
3. 轨迹分析结果每次加载展现 10 条，点击下方的 **点击继续浏览轨迹** 可加载后续 10 条轨迹数据。轨迹分析结果全部显示在 1 页内，如果内容较多，可滚动鼠标查看轨迹图。

各轨迹节点按照日志埋点类型展现不同行为描述。其中：

- 报活埋点：记录启动 App 的行为信息。
- 页面自动化埋点：记录打开页面的行为信息。

- 点击自动化埋点：记录点击页面上的控件的行为信息。



相关日志埋点

- 报活埋点

  关键字段：日志标识 `reportActive` （第 17 个字段）

  埋点样例：

```
  2020-07-05 12:30:33.002,ip=183.95.xxx.xxx^country=中国^province=湖北省^city=武汉市
^district=,2020-07-05 12:30:32:532,363452B111425_IOS-
default,2.1.1,3,XElQBnytt2YDAJfdBoO/uAAq,C3638BAE-C601-4D68-ADCF-
8AB52E3D539C,368800339590,event,-,-,-,-,-,-,reportActive,1,mPaaSAliveiOS,c,-,-,-,mp_baseli
ne=10.1.60.27^mp_module=mPaaSAliveiOS,-,-,XElQBnytt2YDAJfdBoO/uAAq,-,-,-,-,-,-,1000,iPhone
11,13.3,LTE|中国联通,-,-,follow_system_zh-Hans-CN,-,-,-,-
,VoiceOver=0^TimeZone=Asia/Shanghai,-,750*1624,-,3,-
```

- 页面自动化埋点

  关键字段：日志标识 `auto_openPage` （第 10 个字段），当前页面 ID（第 17 个字段）

  埋点样例：

```
  2020-07-05 12:30:33.007,ip=117.136.x.xxx^country=中国^province=上海市^city=上海市^district=
,2020-07-05 12:30:32:902,84EFA9A281942_IOS-
default,2.4.01,2,Wo4BWJSmXjoDAOGmjiPm1HdH,AD2B567B-B0F9-4A78-A1C5-
FFFCEDB2975B,13681855793,auto_openPage,-
,SHM_H5WebViewController|SHM_H5WebViewController__Wo4BWJSmXjoDAOGmjiPm1HdH__NCTFKzA_,-,NCT
FMp8,//SHM_H5WebViewController,-,SHM_H5WebViewController,2,autotrack,c,-
,1278,NCTFMp8,referSpm=^respond=83565^openpagetime=2020-07-05 12:30:31:625^staytime=1278,-
,1278,Wo4BWJSmXjoDAOGmjiPm1HdH,-
,SHM_H5WebViewController__Wo4BWJSmXjoDAOGmjiPm1HdH__NCTFMp8_,SHM_H5WebViewController,SHM_H5W
ViewController,https://__bridge_loaded__/|flash,A93ac1e3a7d2c9d60bbe895acabea8bf9,1000,iPhon
6S Plus,13.3,LTE|中国移动,-,-,follow_system_zh-Hans-CN,-,-,-,-
,VoiceOver=0^TimeZone=Asia/Shanghai,-,1242*2208,-,298,-
```

- **点击自动化埋点**

  关键字段：日志标识 `auto_click` **（第 10 个字段），当前点击的控件 ID （第 17 个字段）**

  埋点样例：

```
  2020-07-05 12:30:33.023,ip=223.104.xxx.xxx^country=中国^province=上海市^city=上海市^distric
t=,2020-06-25 12:27:44:106,84EFA9A281942_ANDROID-
default,2.0.55,2,460001605536199|865611032563288,d5e39965-2def-45d3-8f61-
44efdbb0fea6,13501677084,auto_click,-
,MyWalletAct|MyWalletAct__W/OMuBtMJZ4DAMBf8XDfluoI__NBfkkVv_,ActivityApplicationStub,-,//c
om.app.shanghai.metro.ui.mine.wallet.MyWalletAct/LinearLayout[0]:-/FrameLayout[0]:-/TextVi
ew[0]:tvLeftTitle|-1,-,tvLeftTitle,2,autotrack,c,-,-,-,header=D-AM,-,-
,W/OMuBtMJZ4DAMBf8XDfluoI,-,-
,com.app.shanghai.metro.ui.activities.H5ActivitiesActivity,com.app.shanghai.metro.ui.mine.wa
et.MyWalletAct,com.app.shanghai.metro.ui.mine.wallet.MyWalletAct|tvLeftTitle,A5473178d535a5e
6c8ebf3425d4d2f8,1000,1711-A01,7.1.1,LTE|cmnet,-,-,-
,20190424130858,8,1401,3602,romVersion=v076^brand=360,-,1920x1080,main,171
```

9.性能分析

9.1. 闪退报告

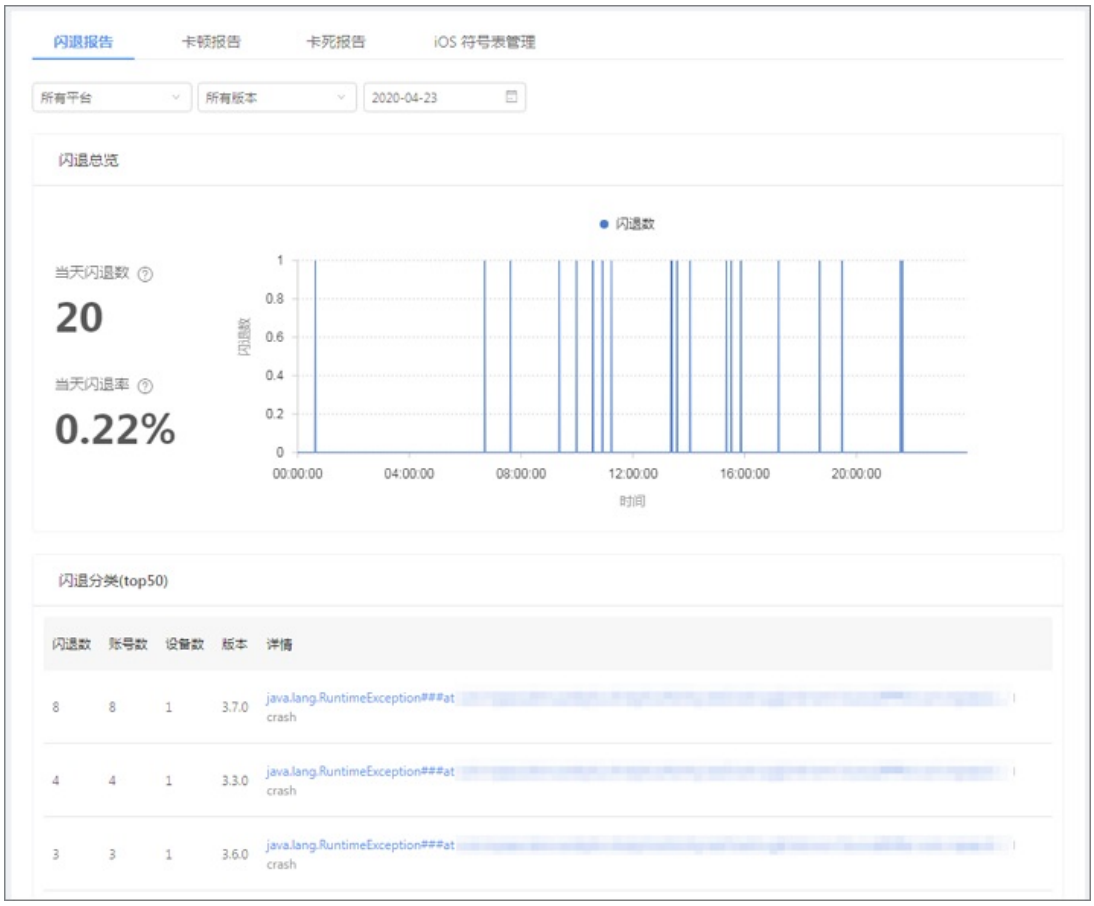
闪退一般是指 App 非正常退出。当发生闪退时，客户端会实时上传闪退信息。这些信息会在控制台上展示出来，总体时间延迟一般在几秒钟到几分钟。

移动分析为应用提供了闪退统计功能，支持统计闪退个数、闪退率、闪退趋势，支持按照问题原因聚合闪退报告，并统计某一类闪退的个数、设备数、主要机型等信息。

> **重要**
>
> 在查看闪退报告之前，确保您已完成客户端 SDK 接入和埋点，具体参见 接入 Android 客户端 和 接入 iOS 客户端。

完成以下操作查看闪退报告：

1. 登录控制台，单击 **产品与服务** > **移动开发平台 mPaaS**，并选择应用。
2. 在左侧的导航栏，单击 **移动分析** > **性能分析** > **闪退报告**。
3. 通过选择平台、App 版本、时间等条件来筛选闪退统计分析数据。



闪退总览

展示每分钟的闪退数据，并以折线图的形式展现数据变化趋势。

- **当天闪退数**：当天的闪退日志总数（未去重）。
- **当天闪退率**：当天的闪退日志总数（未去重） / 客户端上报的报活日志总数，即闪退率 = 闪退数 / 报活数。

闪退详细数据

此区域的数据报表展现所选日期的闪退数据，并根据闪退的原因对日志分类统计。mPaaS 支持符号化闪退日志，iOS 闪退日志需要符号化，安卓闪退日志不需要。

- **闪退数**：同一类闪退的总个数（闪退日志个数）。
- **账号数**：同一类闪退中，不同的 userID 的个数（根据日志中的 **userID** 字段来统计），如果日志中没有填写 userID 字段，默认值为 1。
- **设备数**：同一类闪退中，不同的设备 ID 的个数（根据日志中的 **设备 ID** 字段来统计），如果日志中没有填写设备 ID 字段，默认值为 1。
- **版本**：闪退日志中记录的版本号。
- **详情**：闪退日志中记录的闪退调用堆栈。

> **说明**
> Android 和 iOS 机型均支持在闪退报告的详情数据中查看相应的堆栈信息。

闪退分类详情

在闪退分类列表中，单击 **详情** 列中的内容，可查看该类闪退的错误详情，包括错误组信息以及错误样本信息。

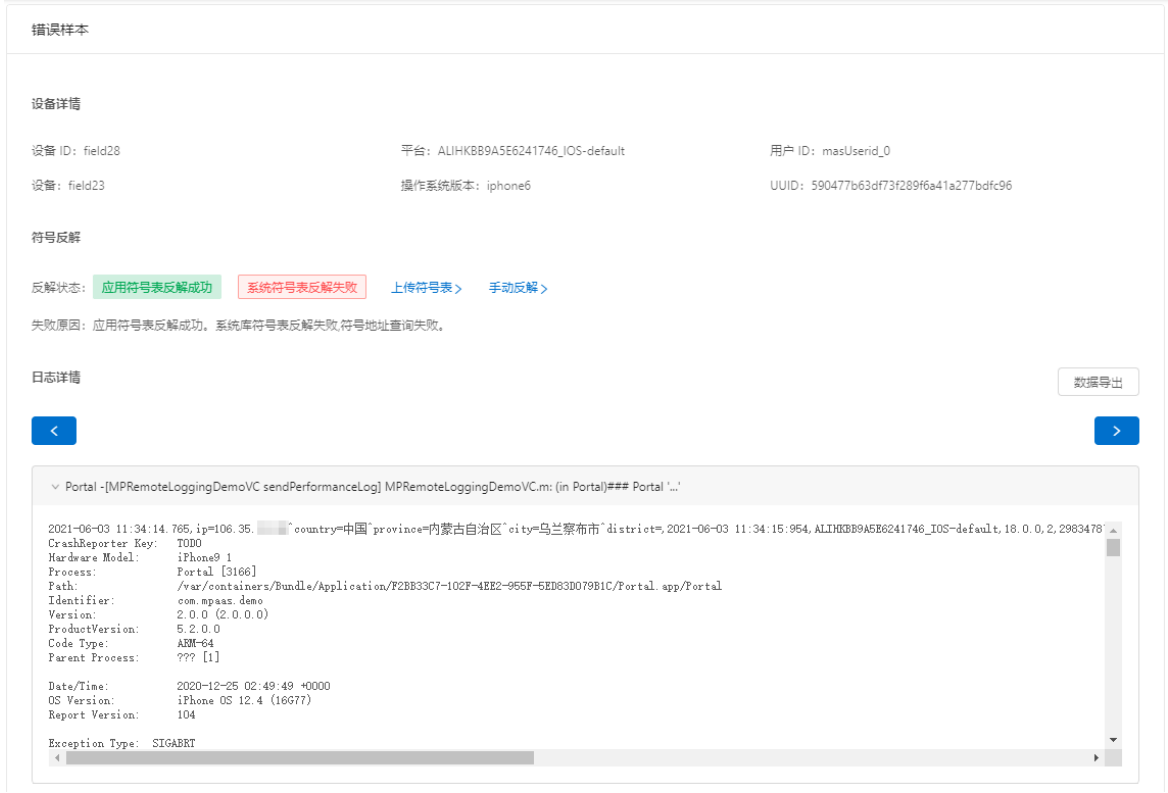
- **错误组**：
  - **闪退数**：同一类闪退的总个数（闪退日志个数）。
  - **影响设备数**：发生该类闪退的设备量，按设备 ID 去重。

    > **说明**
    > 如果设备 ID 为空或者为“-”，设备数不累加。

  - **设备占比**：此类闪退影响的设备数 / 发生闪退的总设备数。
  - **机型**：按闪退次数由高到低展示不同机型的闪退次数占比。
- **错误样本**：展示当前样本的设备详情、日志详情等信息，可通过两侧的 **<** 与 **>** 按钮来切换样本。
  - **设备详情**：展示当前样本的设备 ID、平台、用户 ID、设备型号、操作系统版本 、UUID 等信息。
  - **符号反解**：仅针对 iOS 设备。展示当前闪退日志的反解状态。对于反解失败的日志，同时会展示失败原因，如未找到符号表文件、符号表文件无效、UUID 不匹配等。支持 上传符号表文件，进行手动反解。

    > **说明**
    > 支持通过符号表对 iOS App 的闪退日志进行符号反解。如果 App 发布后尚未上传符号表文件，则默认展示原始日志内容；如果已上传 dSYM 符号表文件，闪退日志符号化会在闪退日志上传后实时进行（延迟在分钟级）。

- **日志详情**：展示当前样本的闪退日志，并支持导出日志。关于日志详情，可查看 闪退埋点。



9.2. 卡顿报告

卡顿是指主线程超过一定时间（Android 2.25 秒，iOS 2 秒）未执行完一个方法。当发生卡顿时，客户端会实时上传卡顿信息。这些信息会在控制台上展示出来，总体时间延迟一般在几秒钟到几分钟。

> **重要**
>
> 在查看卡顿报告之前，确保您已完成客户端 SDK 接入和埋点，具体参见 接入 Android 客户端 和 接入 iOS 客户端。

通过卡顿报告，您可以了解设备卡顿次数、设备卡顿率、影响设备数，并查看每个卡顿类别的详细情况。

完成以下操作，查看卡顿报告：

1. 登录控制台，点击 **产品与服务** > **移动开发平台 mPaaS**，并选择应用。
2. 在左侧的导航栏，点击 **移动分析** > **性能分析** > **卡顿报告**。
3. 选择平台、版本、时间等条件来筛选卡顿统计分析数据。

卡顿总览

展示每分钟的卡顿数据，并以折线图的形式展现数据变化趋势。

- **卡顿次数**：开启卡顿监控的设备发生的卡顿总次数，卡顿监控设备采样率为 10%。
- **卡顿数**：开启卡顿监控的设备发生的卡顿总次数，卡顿监控设备采样率为 10%。
- **卡顿率**：卡顿次数/开启卡顿监控设备的总 PV 数。
- **影响设备数**：卡顿次数，按用户去重。

卡顿详细数据

此区域的数据报表可展现所选日期的卡顿数据，并根据卡顿的原因对日志分类统计。

- **卡顿数**：同一类卡顿的总个数（卡顿日志个数）。
- **账号数**：同一类卡顿中，不同的 userID 的个数（根据日志中的 userID 字段来统计），如果日志中没有填写 userID 字段，默认值为 1。
- **设备数**：同一类卡顿中，不同的设备 ID 的个数（根据日志中的设备 ID 字段来统计），如果日志中没有填写设备 ID 字段，默认值为 1。
- **版本**：卡顿日志中记录的版本号。
- **详情**：卡顿日志中记录的卡顿调用堆栈。

> **说明**
>
> Android 机型支持在卡顿报告的详情数据中查看相应的堆栈信息；针对 iOS 机型，卡顿报告的详情数据不提供堆栈信息。

卡顿分类详情

在卡顿分类列表中，单击 **详情** 列中的内容，可查看该类卡顿的错误详情，包括错误组信息以及错误样本。

- **错误组**：
  - **卡顿数**：同一类卡顿的总个数（卡顿日志个数）。
  - **影响设备数**：发生该类卡顿的设备量，按设备 ID 去重。

    > **说明**
    >
    > 如果设备 ID 为空或者为“-”，则设备数不累加。

  - **设备占比**：此类卡顿影响的设备数 / 发生卡顿的总设备数。
  - **机型**：按卡顿次数由高到低展示不同机型的卡顿次数占比。
- **错误样本**：展示当前样本的设备详情、日志详情等信息，您可通过两侧的 **<** 与 **>** 按钮来切换样本。
  - **设备详情**：展示当前样本的 **设备 ID**、**平台**、**用户 ID**、**设备型号**、**操作系统版本** 信息。
  - **日志详情**：展示当前样本的卡顿日志，并支持 **数据导出**。关于日志详情，可查看 卡顿埋点。

9.3. 卡死报告

卡死包括启动卡死和 ANR 卡死两种情况。当发生卡死时，客户端会实时上传卡死信息。这些信息会在控制台上展示出来，总体时间延迟一般在几秒钟到几分钟。

卡死类型	Android	iOS
启动卡死	App 启动后 30 秒内未能离开欢迎页和进入首页。	App 启动时主线程在 15 秒（iPhone6 及以下机型 30 秒）内未执行完一个方法。
ANR 卡死	即系统 ANR 卡死，定义详见 Android 官网 ANR。	App 运行时主线程在 10 秒（iPhone6 及以下机型 20 秒）内未执行完一个方法。

> **重要**
>
> 在查看卡死报告之前，确保您已完成客户端 SDK 接入和埋点，具体参见 接入 Android 客户端 和 接入 iOS 客户端。

通过卡死报告，您可以了解卡死对应的总次数、卡死率、影响设备数，并分类查看启动卡死和 ANR 卡死的详细情况。

完成以下操作，查看卡死报告：

1. 登录控制台，单击 **产品与服务** > **移动开发平台 mPaaS** ，并选择应用。
2. 在左侧的导航栏，单击 **移动分析** > **性能分析** > **卡死报告**。
3. 选择平台、版本、时间等条件来筛选卡死统计分析数据。

卡死总览

展示每分钟的启动卡死和 ANR 卡死数据，并以折线图的形式展现数据变化。

指标	启动卡死	ANR 卡死
卡死数	当日应用启动卡死次数。	当日应用 ANR 卡死次数。
卡死率	应用启动卡死的次数/应用启动次数。	应用 ANR 卡死的次数/应用启动次数。
影响设备数	指定时间内发生的启动卡死总数，按设备 ID 去重。	指定时间内发生的 ANR 卡死总数，按设备 ID 去重。

启动/ANR 卡死详细数据

此区域的数据报表可展现所选日期的 **启动卡死** 和 **ANR 卡死** 数据，并根据卡死的原因对日志分类统计。

- **卡死数**：同一类卡死的总个数（卡死日志个数）。
- **账号数**：同一类卡死中，不同的 userID 的个数（根据日志中的 userID 字段来统计），如果日志中没有填写 userID 字段，默认值为 1。
- **设备数**：同一类卡死中，不同的设备 ID 的个数（根据日志中的设备 ID 字段来统计），如果日志中没有填写设备 ID 字段，默认值为 1。
- **版本**：卡死日志中记录的版本号。
- **详情**：卡死日志中记录的卡死调用堆栈。

> **说明**
>
> - 针对 Android 机型，支持在卡死报告的详情数据中查看相关的堆栈信息。其中，启动卡死的堆栈信息提供了当前全部线程的堆栈内容，供开发人员排查。
> - 针对 iOS 机型，仅支持在卡死报告的详情数据中查看非启动卡死对应的堆栈信息。卡死报告不提供启动卡死的堆栈信息。

启动/ANR 卡死分类详情

在卡死分类列表中，单击 **详情** 列中的内容，可查看该类卡死的错误详情，包括错误组信息以及错误样本。

- **错误组**：
  - **卡死数**：同一类卡死的总个数（卡死日志个数）。

- **影响设备数**：发生该类卡死的设备量，按设备 ID 去重。

  > **说明**
  > 如果设备 ID 为空或者为“-”，设备数不累加。

- **设备占比**：此类卡死影响的设备数 / 发生卡死的总设备数。
- **机型**：按卡死次数由高到低展示不同机型的卡死次数占比。

- **错误样本**：展示当前样本的设备详情、日志详情等信息，您可通过两侧的 < 与 > 按钮来切换样本。
  - **设备详情**：展示当前样本的 **设备 ID**、**平台**、**用户 ID**、**设备型号**、**操作系统版本** 信息。
  - **日志详情**：展示当前样本的卡死日志，并支持数据导出。关于日志详情，可查看 卡死埋点。

9.4. iOS 符号表管理

移动分析支持通过符号表对 iOS App 的闪退日志进行反向解析，以便定位 App 中的问题代码，帮助提高排查、解决线上异常问题的效率，同时提供 iOS 符号表管理功能以便导入、查询符号表并进行符号表反解测试。

什么是符号表

符号表是内存地址与函数名、文件名、行号的映射表。符号表元素如下所示：

```
<起始地址> <结束地址> <函数> [<文件名:行号>]
```

iOS App 出现闪退时，闪退日志中的 Crash 堆栈为混淆后的二进制信息，需要通过符号表对这些二进制堆栈信息进行反向解析，将其转换为可读的函数名和行数，以便定位 App 中出现问题的代码。

导入 iOS 符号表

对闪退日志进行符号反解前，要先上传符号表。在 iOS 平台中，dSYM 文件是保存符号表的目标文件，文件名通常为 `xxx.app.dSYM` ，建议每次构建或发布 App 的时候，备份好 dSYM 文件。

上传 iOS 符号表的步骤如下：

1. 在 dSYM 文件当前目录下，通过 Linux 命令 `tar -czvf symbol.tgz ./xxx.app.dSYM` 将 dSYM 文件压缩成 tgz 包。
2. 登录 mPaaS 控制台，选择目标应用后，从左侧导航栏进入 **移动分析 > 性能分析 > iOS 符号表管理** 页面。
3. 点击 **导入**，在导入符号表弹窗中，输入符号表信息，并上传相应的符号表。
   - **版本**：App 版本号。
   - **Module Name**：iOS App 二进制文件名称，存储在符号表文件里面用来标识出对应的 App 二进制文件，以方便用户匹配 App 二进制文件和 App 的符号表文件。

     此处填写 App 主 module 的 moduleName，例如打出的包是 `Produce.app`，那么 moduleName 为 `Produce` 。
   - **UUID**：通用唯一识别码（Universally Unique Identifier，简称 UUID），是机器生成的唯一标识符。iOS 应用每次编译时都会生成一个 UUID，为保证反解成功，堆栈的 UUID 和应用的符号表文件中的 UUID 必须保持一致（也就是来自同一次编译）。还原 Crash 堆栈信息时，仅当导入的符号表文件的 UUID 与闪退日志中的 UUID 一致时，才能准确地对堆栈进行解析还原。

     此处填写 App 主 module 的 UUID，如果有多个，选择其中一个填写即可。例如有 armv7、arm64 两种架构，则会有两个 UUID，任选一个填写即可。格式上，UUID 字符串需去掉“-”，并且全部小写，例如：b7583434dc5e377bb4d8e7b69bf4c1fb。
   - **URL 上传**：输入符号表压缩后的 tgz 文件的 URL。如果指定 URL 下没有符号表文件，会返回报错信息。
4. 点击 **导入**，导入符号表。
5. 在 iOS 符号表管理页面中查看符号表的导入状态。如果符号表状态为 **Done**，表示导入成功；如果为 **Failed**，则表示导入失败，请根据错误提示重新导入符号表文件。

导入的符号表都将展示在符号表管理页面，可根据 App 版本进行查询。

反解测试

反解测试用于验证导入的符号表文件是否有效。

反解测试的步骤如下：

1. 在 iOS 符号表列表中，选择导入成功的目标符号表，点击操作列下的 **反解测试**。
2. 在 **原始日志文本** 栏中输入需要反解的闪退日志内容后，点击 **反解日志**。**日志反解结果** 栏将显示反解后的日志文本，如果反解失败，会提供失败原因，例如“UUID 不匹配”等。

10.日志管理

10.1. 拉取实时日志

mPaaS 的客户端 SDK 为用户提供写入诊断日志的接口。该诊断日志是指您根据开发需求或排查问题的需要，调用埋点接口写入的日志。此类日志默认只会记录在磁盘中，并不上传。

需要排查问题时，您可以通过日志提取功能下发诊断任务到客户端，客户端收到任务后上传日志到服务器，您便可以在应用分析控制台上下载上传的日志。

前置条件

拉取实时日志前，确保已完成客户端诊断日志埋点。有关日志埋点操作，请参见 iOS 客户端诊断 和 Android 客户端诊断。

下载服务端日志

下载服务端日志的操作如下：

1. 登录 mPaaS 控制台，从左侧导航栏进入 **移动分析** > **日志管理** > **拉取实时日志** 页面。
2. 单击 **添加** 按钮进入新增诊断任务页面，然后根据页面提示填写诊断任务内容。

   在配置 Android 平台的诊断任务时，如果日志类型选择 **自定义拉取路径** ，则在填写自定义路径时需注意以下两点：

   - 需确保自定义路径为单个文件，而非目录。若为手机 SD 卡下的文件，则路径前须加上 `/storage/emulated/0/` ，例如 `/storage/emulated/0/Android/data/com.mpaas.aar.demo.analytics/files/mdap/upload/log.txt` 。
   - 文件路径必须包含应用包名，例如 `com.mpaas.aar.demo.analytics` ，考虑到应用行为规范，该包名必须与您在 代码配置 中填写的 Package Name 保持一致。无论是应用内部存储的文件还是 SD 卡上的文件，都应该只拉取应用自身目录下的文件。
3. 填写完诊断任务信息后，单击 **确定** 按钮生成一条诊断任务。
4. 在诊断任务列表中，选择刚创建的任务，并选择 **触发通道** ，然后单击 **触发** 按钮触发诊断任务的下发。任务下发成功后，页面上的任务状态会更新。

   触发通道需要与客户端接入诊断的方式保持一致，即如果客户端使用的诊断方式为 **数据同步** ，则触发通道必须选择 **通过数据同步触发（Sync）** ；如果诊断方式为 **消息推送** ，则触发通道必须选择 **通过消息推送（Push）** 。
5. 客户端收到诊断任务后，将日志上传到服务器，同时任务状态更新为 **处理完成** 。此时，您可以单击 **查看** 按钮进入子任务查看页面，单击 **下载** 按钮下载日志。

10.2. 查询历史日志

您可以根据业务需求，查询客户端上报的历史日志。可以查询的日志类型包括行为日志、自动化日志、异常日志、客户端性能日志。查询历史日志时，可以使用关键字类型和高级过滤规则快速定位到日志位置，并且下载搜索结果。

前置条件

确保您具备日志查询的权限。具体的权限问题，请咨询您的系统管理员。

关于此任务

对于任意一种日志类型，您可以通过任意关键字进行查询。

- 任意关键字使用全文索引，并根据英文分词，因此支持以下搜索内容：
  - 字母和数字。例如，全文中 `hello Peter how are you` 会被分成 `hello` 、 `Peter` 、 `how` 、 `are` 、 `you` 等单词，输入以上任意一个单词都可以搜索出来。
  - 不包含 `path` 的 URL。例如，支持 `www.google.com` 搜索， 但是 `www.google.com.hk/webhp` 中 `/webhp` 不会被索引，无法进行搜索。

- 任意关键字支持多关键词查询，不同的关键字之间用空白字符（包括空格键、Tab 按键）分开。多关键字遵循叠加规则。

> **重要**
>
> - 支持查询最近 32 天的原始日志（不包含诊断日志）以及最近 7 天的诊断日志，但是每次查询的时间跨度不能超过 2 天。
> - 针对分析结果，实时数据在服务端的保留时长为 90 天，离线数据为 360 天。
> - 对于积压未上传的客户端日志，达到限度时会择旧删除 1/4 以释放存储空间。限度说明如下：
>   - 在 Android 客户端，埋点日志总量超过 15MB，或 applog 总量超过 50MB 时。
>   - 在 iOS 客户端，埋点日志或 applog 超过 30MB 时。

操作步骤

登录 mPaaS 控制台，完成以下步骤：

1. 在左侧导航栏，点击 **移动分析** > **日志管理** > **日志回放**。
2. 选择要查询的日志类型、日志查询的时间顺序，并输入关键字，然后点击 **搜索** 图标。

   也可通过点击搜索框下方显示的历史查询记录，进行快速查询。默认查询最近 12 小时的日志，支持正序和倒序查询。在 **日志列表** 下方，您可以看到查询的日志结果。

   如要精确日志查询时间，点击 **时间条件** 展开日志查询的条件，选择时间类型，设置日志查询的开始和结束时间，选择时间区间、每页显示的日志条数。

   > **说明**
   >
   > 如果您选择的时间区间和开始/结束时间冲突，开始/结束时间会根据您选择的时间区间自动调整。

3. 要添加高级过滤规则，点击关键字输入列右边的加号按钮（⊕）。
4. 在 **高级过滤规则** 栏中，选择规则类型，并添加对应的过滤内容。
   - 支持的规则类型包括 **埋点（Seed）**、**平台（Platform）**、**版本（Version）**。
   - 您可以添加多条高级过滤规则，过滤规则会叠加。如果要移除当前的过滤规则，点击删除按钮（⊖）。
5. 在 **日志列表** 下方，查看筛选后的日志查询结果，关键字高亮显示。查询结果默认折叠显示。
   - 点击单条查询结果，可以展开单条结果的完整日志内容。
   - 勾选 **展开全部**，可以展开所有日志内容；取消勾选，则会收起日志内容。
6. 如要下载搜索结果，点击关键字输入列右边的下载按钮（⬇）。搜索结果汇总在 .xlsx 文件中。

相关链接

- Android 客户端埋点
- iOS 客户端埋点
- 日志埋点说明

10.3. 配置日志上报开关

开关配置功能指通过服务端下发的开关值，修改日志自动上报的触发条件，对日志上报进行动态控制。

移动分析依赖客户端 SDK 来进行埋点，收集用户行为以及 App 性能等相关数据生成日志并上报到服务端，然后通过实时或离线计算形成各种指标和大盘供用户查看。

mPaaS 会对埋点日志的上报收取流量费用，为避免产生一些不必要的费用，您可以通过开关配置对日志上报进行管理。更多关于日志上报流量费用的信息，请参见 后付费模式。

下文介绍如何添加埋点配置并进行管理。

新增埋点配置

1. 登录 mPaaS 控制台，从左侧导航栏进入 **移动分析** > **日志管理** > **配置上传开关** 页面，单击左上方的 **埋点配置**，进入日志开关列表页。
2. 单击 **新增业务** 按钮，配置埋点信息。
   - **业务 code**：业务码。其中，自定义行为埋点的业务码配置因平台而异。
     - Android：客户端通过代码 `MPLogger.event(String logId, String bizType, Map<String, String> params)` 设置的 `bizType`，bizType不能带下划线_。
     - iOS：默认为 `behavior`，客户端可以通过 `writeLogWithActionId` 接口中的 `bizType` 参数自定义。更多信息，请参见 Android 业务码 或 iOS 业务码。
   - **业务名称**：埋点业务的说明。您可以自定义业务名称。
   - **日志头**：日志模型中的 **字段 01**，用于区分不同的日志类型。其中，自定义行为埋点的日志头为 `D-VM`。更多信息，请参见 日志模型。
3. 单击 **新增** 按钮，完成埋点配置新增。更进一步的配置，参见下文 修改埋点配置。

修改埋点配置

新增埋点配置展示在日志开关列表中，您可以进行日志上报开关设置、网络设置，埋点修改以及删除操作。

设置日志上报开关

开启或关闭日志上报。开启上报开关后，该业务的埋点日志会自动上报。

设置上报网络

您可以选择在 **全网环境**（2G、3G、4G 和 Wi-Fi 等）或只有在 **Wi-Fi** 环境中上报埋点日志。

修改埋点信息

单击操作列下的 **修改** 按钮，进入埋点配置修改页面，您可以对埋点配置作进一步的修改：

- **业务码**：对应新增埋点配置时填写的业务码。
- **说明**：对应新增埋点配置时填写的业务名称。
- **日志头**：对应新增埋点配置时填写的日志头。
- **最低上报等级**：每条日志的重要程度分为 1、2、3 三个等级；1 级表示最重要，2 级次之，3 级最次。如设置最低上报等级为 2，则等级为 1 和 2 的日志会上报，而等级为 3 的日志不上报。其中，自定义行为埋点默认等级为 2；iOS 应用可以在客户端埋点时修改日志等级，请参见 iOS 自定义行为埋点。
- **日志上报条数**：客户端埋点日志会先写入本地，本地文件中此业务码类型的日志达到一定条数后会自动上报到移动分析服务端。更多信息，请参见 Android 日志自动上报 或 iOS 日志自动上报。
- **日志上报比率**：按用户维度设置日志上报的比率，采用千分制，如 1000 表示全部用户都上报。
- **策略**：勾选 **压后台上报** 策略后，不同平台的上报时机会有所不同。
  - Android：应用进入后台、超过半小时后，会触发日志自动上报。
  - iOS：应用进入后台会立即触发日志自动上报。更多信息，请参见 Android 上报日志 或 iOS 上报日志。

初始化业务

客户端写入本地的日志，在满足一定条件后，会自动上报到移动分析服务端。触发自动上报的条件由默认的埋点配置决定。新建应用后，您可以通过 **初始化业务** 功能，将默认的埋点配置同步到新应用埋点配置列表中。

如需初始化业务，单击页面上的 **初始化业务** 按钮，并确认即可。初始化成功后，您将看到默认的埋点配置列表。

> **重要**
>
> 即使您未通过 **初始化业务** 功能将默认埋点配置同步到新应用的埋点配置列表中，默认的埋点配置仍然起作用，只不过不会在埋点配置列表中显示。

Mock

如上文所述，您可以通过页面操作新增并修改埋点配置，以此控制日志的上报策略。此外，控制台还允许您直接使用 `JSON` 进行埋点配置。但 **不建议** 您使用此功能，理由如下：

- Mock 功能仅仅是埋点配置的一种简便方式，不提供更多的功能。
- 假如您同时通过页面操作和 Mock 功能对同一业务埋点进行了配置，则 Mock 配置会优先生效。
- 在不熟悉埋点配置格式时，您很难正确使用 Mock 功能。很可能会影响默认的日志自动上报。

新增 Mock 配置

新增 Mock 配置的步骤如下：

1. 在 **配置上传开关** 页面，单击 **新增 Mock 配置** 按钮。
2. 填写配置信息。
   - **版本**：选择您应用的版本。
   - **平台**：选择应用类型。
   - **值**：JSON 类型的配置信息。具体参见下文的 Mock 配置格式。
3. 信息确认无误后，单击 **确定**，完成 Mock 配置新增。

Mock 配置格式

Mock 配置格式如下：

```
{
                        "日志头1": {
                        "业务码1": {
                        "write": "yes",
                        "send": [
                        "wifi"
                        ],
                        "maxLogCount": 50,
                        "level": 3,
                        "uploadRate": 1000,
                        "event": [
                        "gotoBackground"
                        ]
                        },
                        "业务码2": {
                        ...
                        },
                        ...
                        },
                        "日志头2": {
                        ...
                        },
                        ...
                        }
```

- 日志头：对应埋点配置中的日志头。
- 业务码：对应埋点配置中的业务 code。

- write：对应埋点配置中的上报开关。开关开启时值为 `"yes"`，否则为 `"no"`。
- send：对应埋点配置中的网络。网络为 **全网** 时值为 `["2g", "3g", "4g", "wifi"]`，**仅 Wi-Fi** 时为 `["wifi"]`。
- maxLogCount：对应埋点配置中的日志上报条数。
- level：对应埋点配置中的最低上报等级。
- uploadRate：对应埋点配置中的日志上报率。
- event：对应埋点配置中的策略。选择压后台策略时值为 `["gotoBackground"]`，否则为 `[]`。

10.4. 客户端诊断

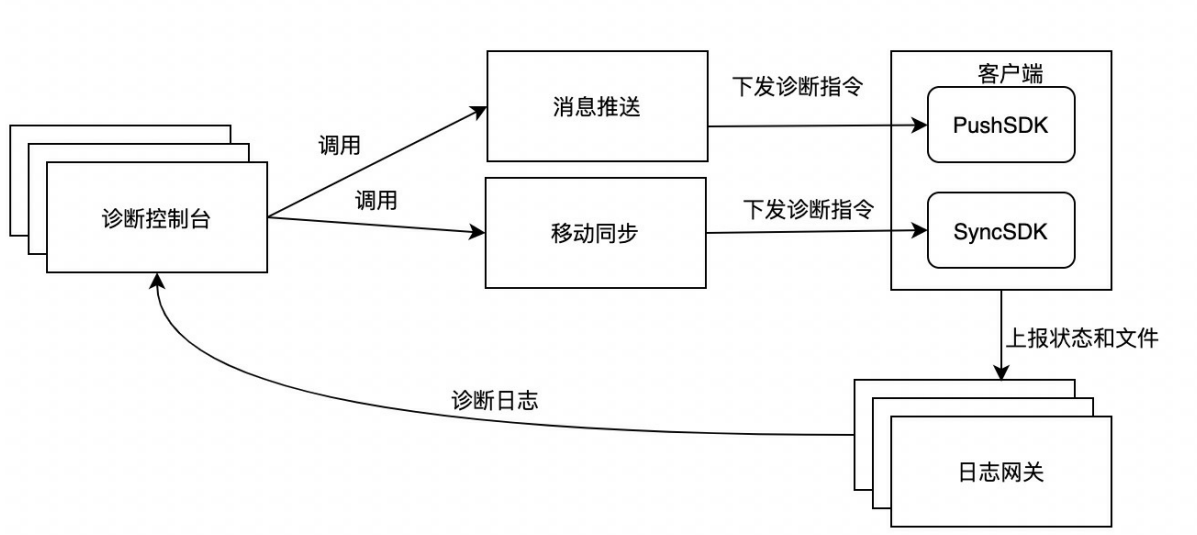
10.4.1. 客户端诊断介绍

客户端诊断是指通过服务端获取客户端写入的诊断日志，并根据诊断日志进行客户端问题的诊断。

您可以在客户端关键链路写入诊断日志，当出现线上问题时，在控制台针对用户下发诊断指令拉取诊断日志，从而排查问题。客户端诊断功能支持通过 数据同步简介 或者 消息推送 通道拉取诊断日志。

原理框架

控制台根据不同的诊断通道（消息推送或移动同步）下发日志诊断指令，客户端接收到诊断指令后，将诊断日志上报到日志网关，当诊断文件上报成功后，在控制台可以下载诊断文件。基本框架如下所示：



接入和使用

Android 端的接入及使用方式参见 Android 客户端诊断；iOS 端的接入及使用方式分别参见 添加 iOS SDK 和 使用 iOS SDK。

10.4.2. Android 客户端诊断

本文介绍如何在 Android 客户端中使用日志诊断服务。日志诊断在控制台下发任务时支持消息推送和数据同步双通道。

进行 Android 客户端诊断的步骤如下：

1. 初始化诊断服务
2. 写入诊断日志
3. 在控制台拉取日志

前置条件

- 您已接入移动分析组件，且日志上报功能正常。参见 接入移动分析到 Android 客户端 了解接入步骤。
- 您已接入消息推送或数据同步组件。
  - 如果使用消息推送通道，参见 快速开始接入消息推送到 Android 客户端 了解接入步骤。
  - 如果使用数据同步通道，参见 接入 Android接入数据同步到 Android 客户端 了解接入步骤。

初始化诊断服务

诊断服务支持使用数据同步或消息推送通道拉取诊断日志，两种通道的初始方式不同。

使用消息推送通道

使用消息推送通道时，您需要在 App 启动后完成以下操作：

- 初始化。初始化推送服务，建立客户端和移动推送网关之间的长连接，这个长连接由推送 SDK 维护，即自建通道。初始化推送 SDK
- 绑定用户 ID。此用户 ID 由开发者自定义，既可以是真实用户系统的用户标识，也可以是能和每个用户形成映射关系的其他参数，例如账号、手机号等。上报用户 ID

使用数据同步通道

使用数据同步通道时，您需要在 App 启动后调用以下方法，完成通道初始化。

```
// 设置 userId
MPLogger.setUserId(String userId);
// 初始化 Sync 通道，必须先设置 userId，否则会初始化失败
MPDiagnose.initSyncChannel(Context context);
```

写入诊断日志

客户端收到下发的诊断任务时，只有使用 mPaaS 的日志工具 MPLogger 写入的日志才会被上传，因此推荐您使用 MPLogger 代替 `android.util.Log` 写日志。MPLogger 提供和原生 Log 类似的日志级别方法：

```
void verbose(String tag, String msg);
void debug(String tag, String msg);
void info(String tag, String msg);
void warn(String tag, String msg);
void warn(String tag, Throwable t);
void warn(String tag, String msg, Throwable tr);
void error(String tag, String msg);
void error(String tag, Throwable t);
void error(String tag, String msg, Throwable t);
void print(String tag, String msg);
void print(String tag, Throwable t);
```

在通过 MPLogger 写入的日志中，debug 包会在 logcat 中显示；release 包不会在 logcat 中显示。诊断日志在设备上的存储目录为：

- debug 包：`/sdcard/[PackageName]/applog`，当该目录无法写入时，日志将写入 release 包的目录。

> **重要**
>
> 当 **targetSdkVersion 大于等于 30 且 手机系统大于等于 11** 时，存放目录为：`/storage/emulated/0/Android/data/com.mpaas.demo/cache/[PackageName]/applog/`。

- release 包：`/data/data/[PackageName]/files/applog`。

在控制台拉取日志

您可以在控制台拉取使用 mPaaS 的日志工具打印的日志，从而快速便捷地分析 App 在指定机型或用户上出现的崩溃或异常问题。

步骤 1：创建日志拉取任务

1. 进入 mPaaS 控制台，选择目标应用。
2. 在左侧导航栏中，点击 **移动分析** > **日志管理**。
3. 在 **拉取实时日志** 标签页中，点击 **添加** 按钮。
4. 填写任务信息。其中，**用户 ID** 为您的应用登录系统中用户的标识。通过 `MPLogger.setUserId(String userId)` 或消息推送上报用户 ID 的方法设置。
5. 点击 **确认** 按钮，完成日志拉取任务创建。

步骤 2：触发日志拉取任务

1. 在日志拉取任务列表中，找到刚刚创建的任务，选择 **触发通道**，然后点击 **操作** 列的 **触发**。
2. 稍等片刻刷新页面，若任务状态为：
   - **任务处理完成**：点击 **查看** 按钮即可下载诊断日志。
   - **调用 Push/Sync 服务成功**：表示已下发上传诊断日志的消息，但客户端还未收到或收到但未上传日志。 针对这种情况，请确认您的 App 进程在系统中仍然存在；如果不存在请重启 App。如果重启后任务状态仍未变化，请参考下文进行初步排查。

问题排查

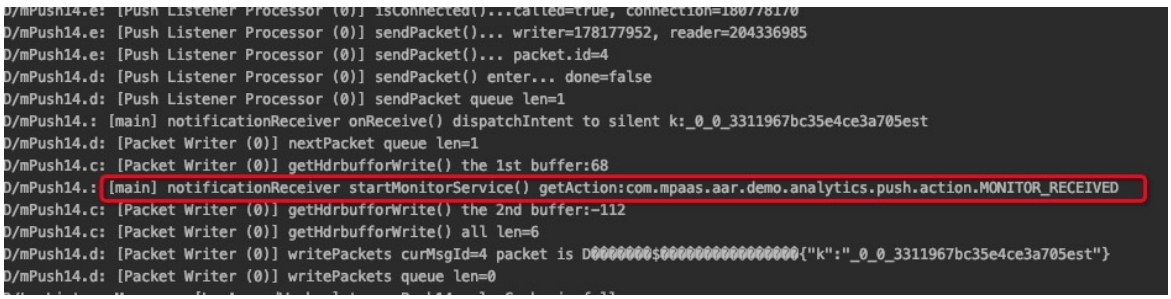
如果出现无法拉取到日志的问题，请根据所使用的诊断日志下发通道，按照对应的步骤进行排查。

使用消息推送通道

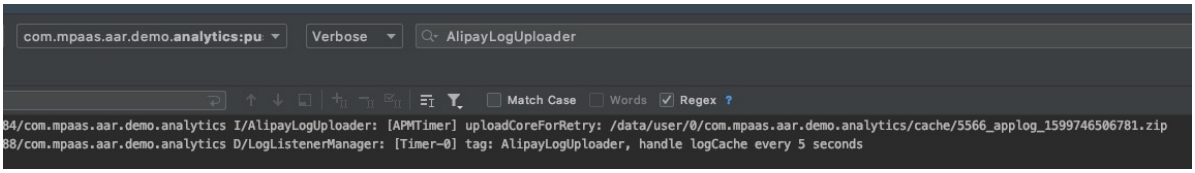
排查步骤如下：

1. 进入 **mPaaS 控制台** > **消息推送** 页面，根据 userId 推送一条普通的消息到您的 App，确认消息推送通道畅通。
2. 在消息推送通道畅通的情况下，先清空 logcat 日志，切换到 push 进程，然后在控制台诊断任务中再次点击 **触发** 按钮，观察 logcat 日志。
3. 过滤 `mPush14`，观察是否收到诊断消息。若未收到，请确认推送通道是否已断开连接。

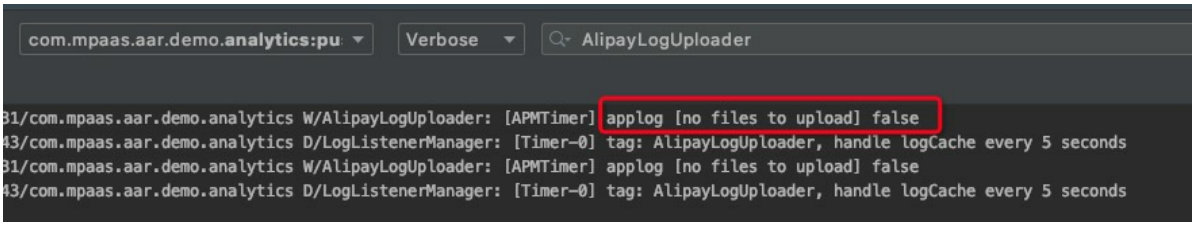
4. 确认收到诊断消息后，可查找是否有以下日志，确认诊断消息是否被转发到 `MonitorService`。 若报错找不到 `ClientMonitorService`，请使用 mPaaS 插件更新 SDK：
   - 10.1.32 基线请升级到 10.1.68，并更新组件到最新版本。
   - 10.1.60 或 10.1.68 基线请更新组件到最新版本。



5. 确认 `MonitorService` 启动后，过滤 `AlipayLogUploader` ，查看本地是否存在诊断日志。如果出现以下日志，说明本地存在诊断日志可以上传。



如果出现以下日志，说明不存在诊断日志。

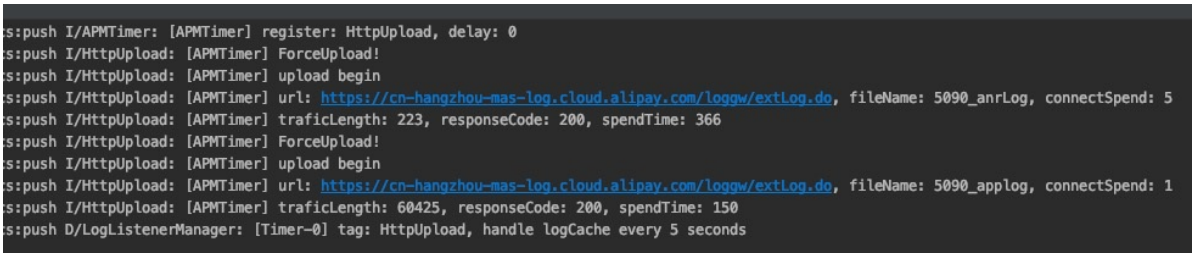


若不存在诊断日志，请检查以下项目：

- 控制台创建的拉取实时日志任务，选取的时间段不能少于 1 小时；在选取的时间段内 App 须运行且输出诊断日志。
- App 声明并动态申请了以下权限：

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
```

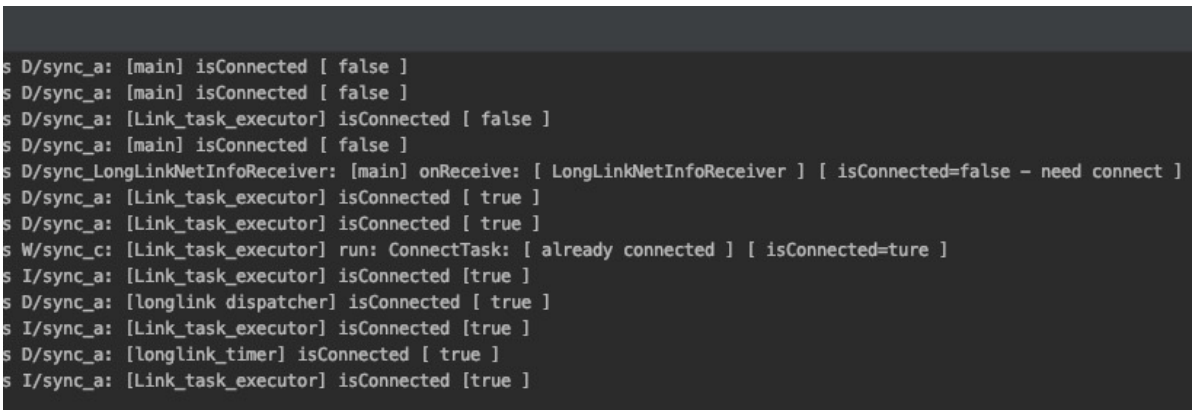
6. 确认本地存在诊断日志后，过滤 `HttpUpload` ，查看日志上传是否成功，若 `responseCode` 为 200 则表示上传成功。



使用数据同步通道

排查步骤如下：

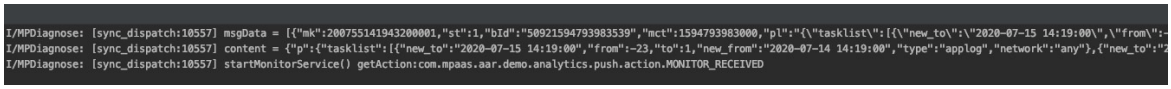
1. 清空 logcat 日志，切换到主进程，初始化同步通道后，过滤 `isConnected` ，观察同步通道建连是否成功。



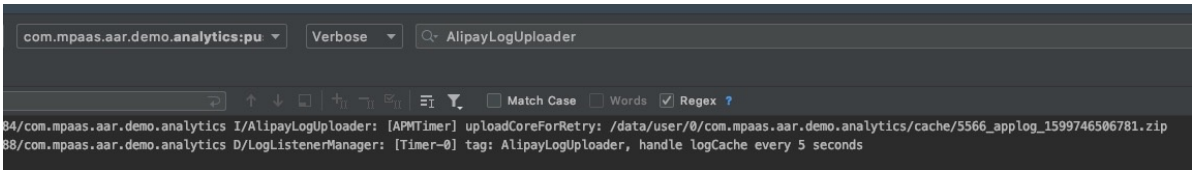
2. 在 mPaaS 控制台诊断任务中再次点击 **触发** 按钮，过滤 `MPDiagnose` ，观察是否收到诊断消息，以及是否启动了 `MonitorService` 。

   若未收到消息，请确认设置的 userId 和诊断任务中填写的 userId 是否一致；若报错找不到 `ClientMonitorService` ，请使用 mPaaS 插件更新 SDK：

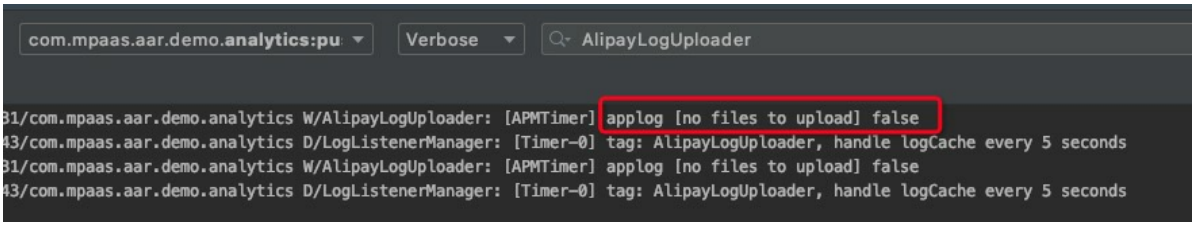
- 10.1.32 基线请升级到 10.1.68，并更新组件到最新版本。
- 10.1.60 或 10.1.68 基线请更新组件到最新版本。



3. 确认 `MonitorService` 启动后，切换到 push 进程，过滤 `AlipayLogUploader`，查看本地是否存在诊断日志。如果出现以下日志，说明本地存在诊断日志可以上传。



如果出现以下日志，说明不存在诊断日志。

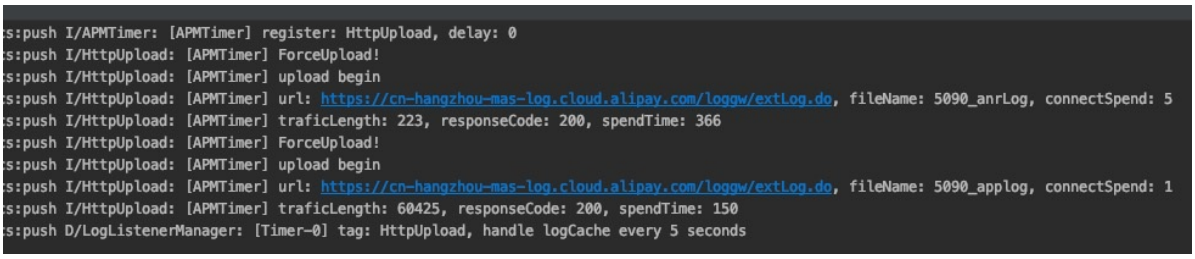


请检查以下项目：

- 控制台创建的拉取实时日志任务，选取的时间段不能少于 1 小时；在选取的时间段内 App 须运行且输出诊断日志。
- App 声明并动态申请了以下权限：

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
```

4. 确认本地存在诊断日志后，过滤 `HttpUpload`，查看日志上传是否成功，若 `responseCode` 为 200 则表示上传成功。



10.4.3. iOS 客户端诊断

10.4.3.1. 添加 SDK

本文将引导您基于原生框架使用 CocoaPods 添加诊断或消息推送 SDK。

诊断功能支持通过 **移动同步** 和 **消息推送** 两种方式进行日志拉取。您可以根据需要选择其中一种方式，并在接入时相应地选择 **诊断** 或 **消息推送** 模块。移动同步方式成功率更高，依赖移动同步服务；消息推送方式要求在下发诊断指令时用户在线，依赖消息推送服务。

前置条件

已参考 基于已有工程且使用 CocoaPods 接入 文档，完成通用接入步骤。

添加 SDK

下面针对两种不同的诊断方式，介绍如何接入诊断功能。

移动同步方式

在 `Podfile` 里配置 `mPaaS_pod "mPaaS_Diagnosis"`，具体操作参考 CocoaPods 接入说明中的 Podfile 文件配置操作说明。

消息推送方式

在 `Podfile` 里配置 `mPaaS_pod "mPaaS_Push"`，具体操作参考 CocoaPods 接入说明中的 Podfile 文件配置操作说明。

10.4.3.2. 使用 SDK

完成以下步骤进行 iOS 客户端诊断：

1. 初始化诊断服务
2. 设置用户 ID
3. 写诊断日志
4. 查看本地诊断日志
5. 获取线上用户诊断日志

初始化诊断服务

诊断功能在使用之前需要进行初始化：

```
#import <MPDiagnosis/MPDiagnoseAdapter.h>
[MPDiagnoseAdapter initDiagnose];
```

诊断服务支持使用 **移动同步** 或 **消息推送** 方式拉取诊断日志，两种方式的初始化代码不同。

移动同步方式初始化

若采用移动同步的方式拉取诊断日志，则在使用诊断功能之前需要进行初始化。

```
#import <MPDiagnosis/MPDiagnoseAdapter.h>
[MPDiagnoseAdapter initDiagnose];
```

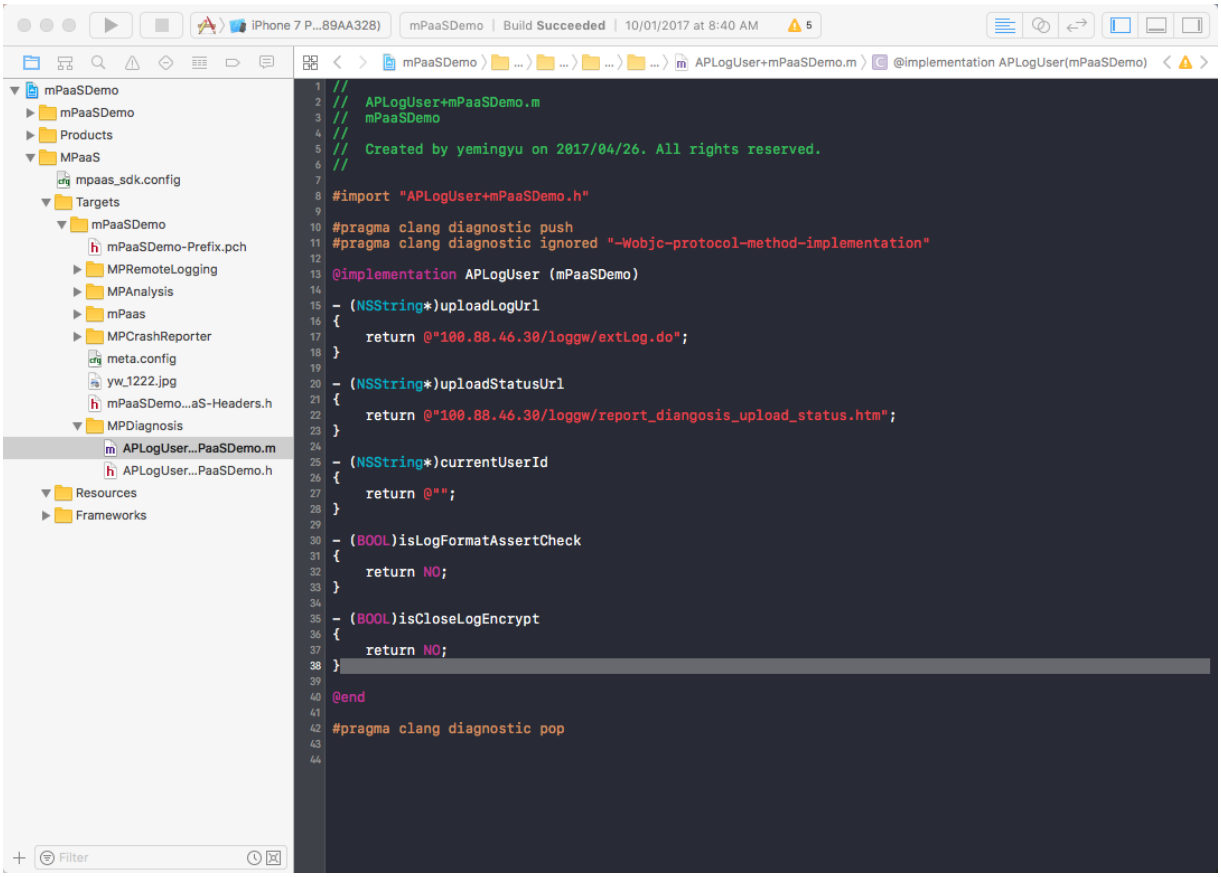
消息推送方式初始化

您需要先接入消息推送组件，完成 消息推送初始化。然后，在接收到消息推送之后，进行如下方法调用：

```
#import <APLog/APLogMgr.h>
[[APLogMgr sharedInstance] handlePushDiagnosisCmd:[notification.userInfo
objectForKey:@"content"]];
```

旧版本升级注意事项

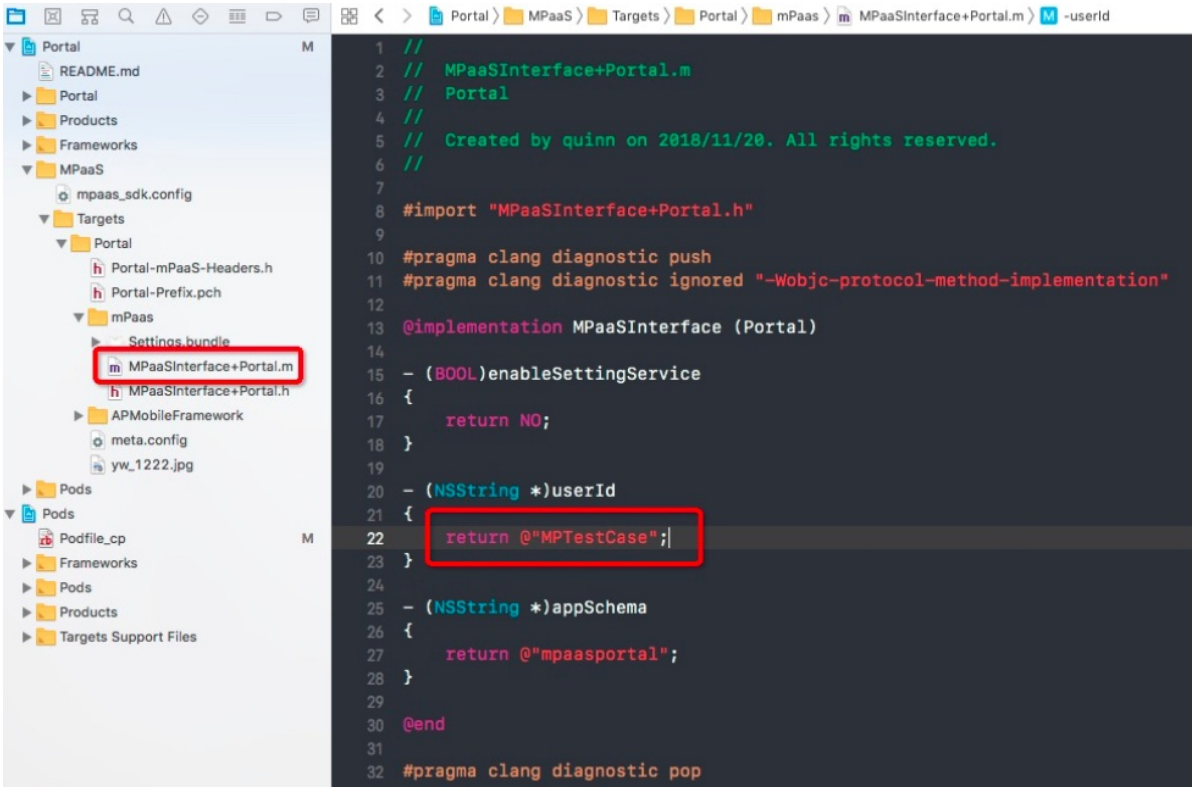
10.1.32 版本之后不再需要添加 `APLogUser` 类的 `Category` 文件，中间层会实现包装，升级版本后请检查工程中是否存在旧版本配置，如果有请移除。下面为新版本应移除的配置。



设置用户 ID

诊断服务根据 **用户 ID** 拉取日志。

1. 在 `MPaaSInterface` 的实现中通过 `userId` 函数配置用户 ID。



2. 当用户切换时，即 `MPaaSInterface` 的 `userId` 函数中配置的值发生变化时，调用以下函数：

```
[MPDiagnoseAdapter userChange];
```

详情可参考 `MPDiagnosis` 下的 `MPDiagnoseAdapter.h` 文件。

写诊断日志

调用以下方法在 App 关键链路写诊断日志：

```
/**
 * Log a message with kAPLogLevelInfo level.
 *
 * @param message   An NSString object that contains a printf-style string containing a log
message and placeholders for the arguments.
 * @param ...       The arguments displayed in the format string.
 */
#define APLogInfo(tag,fmt, ...) \
APLogToFile(tag, kAPLogLevelInfo, fmt, ##__VA_ARGS__)
```

详情可参考 `APLog` 下的 `APLog.h` 文件。

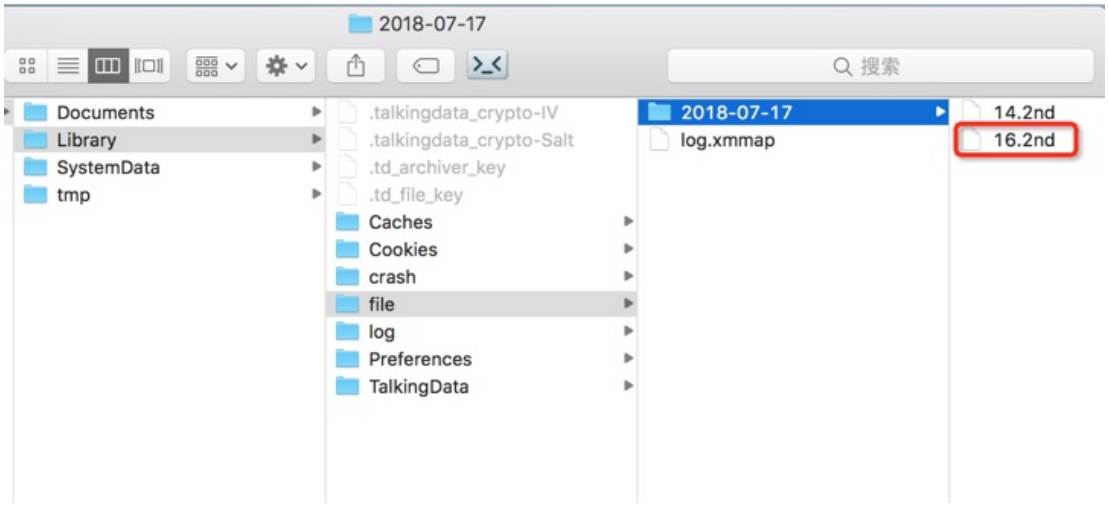
例如，可以在启动完成后，调用下面的语句进行诊断日志的写入：

```
APLogInfo(@"mPaaS", @"Start Cost %d", time);
```

> **说明**
>
> - APLogInfo 写入的日志默认不会在 Xcode 控制台打印，若希望在开发阶段控制台打印相关日志，可以在工程中添加 ConsoleLog 文件。
> - 出于安全考虑，若您希望应用上线后不打印任何日志（包括 NSLog 和 APLogInfo 打印的日志），可在生成应用 release 包时在工程中添加 RemoveNSLog 文件。

查看本地诊断日志

您可以在沙盒目录下找到诊断日志，如下图所示。该日志默认不会上报，只有在需要时才通过控制台下发指令拉取。



保存在客户端本地的诊断日志有 **保存期限** 和 **文件大小限制**。客户端诊断 SDK 会在应用压后台或杀进程时按下述保存策略检查处理：

- 保存期限： 默认保留 6 天。如果检测到前 3 天的日志文件总大小超过 30 MB，前 3 天的日志都会被删除。

- 文件大小限制：默认不超过 100 MB。超过 100 MB 后，会按时间顺序从前到后删除前一半大小的日志。例如有 120 MB 的日志，会删除 60 MB。

获取在线用户诊断日志

应用发布上线后，要获取客户端本地诊断日志排查问题，可通过 mPaaS 控制台下发指令获取相关诊断日志。具体操作，查看 移动分析 > 日志拉取。

10.5. 埋点日志模型

10.5.1. 日志埋点说明

本文对移动分析中涉及到的各种类型的日志埋点进行介绍。

信息收集说明

移动分析功能依赖于客户端上报的埋点日志。为了提供更为精准、丰富的分析能力，埋点日志会收集用户如下设备相关的信息：公网 IP、IMEI、IMSI、设备型号、系统版本、网络类型（如 Wi-Fi、3G 或 4G）、操作系统语言、CPU 核数、CPU 转速、内存大小、屏幕分辨率、客户端渠道号、客户端版本号等。

埋点日志模型

不同类型的日志，其格式各不相同。日志是一个由逗号分隔的字符串，字符串的不同位置代表不同的含义，服务器根据位置信息来切分日志。

常见的埋点类型如下：

- Android 和 iOS 埋点
  - 自定义事件埋点：记录按钮、链接点击等操作，可在 App 内任意动作触发时机埋入，用于自定义事件分析和漏斗分析等功能。
  - 行为埋点
    - **报活埋点**：记录 App 的启动操作，包括客户端冷启动或压后台后 App 界面回到前台。用于统计启动次数、新增用户、活跃用户、活跃账号等核心指标。
      - Android 端默认 App 压后台超过 30 分钟后回到前台记一次报活。
      - iOS 端默认 App 每次从后台回到前台记一次报活。如需修改为 30 分钟上报一次，请设置 `[[DTFrameworkInterface sharedInstance] logReportActiveMinInterval];` 的返回值为 1800。
    - **页面自动埋点**：自动记录页面的打开、来源、停留时长等信息。用于分析页面 PV、UV、来源去向等指标。
    - **压后台埋点**：记录 App 前后台切换相关信息。用于分析用户使用应用的时长、活跃时间等指标。
  - 性能埋点
    - **启动速度埋点**：记录 App 的启动速度，区分首次启动（首次安装后，第一次启动 App）和非首次启动（非首次安装 App 后，启动App）。
    - **卡死埋点**：记录 App 卡死及相关错误日志。包括以下情况：
      - Android 启动卡死：App 启动后 30 秒内未能离开欢迎页和进入首页。
      - Android ANR 卡死：即系统 ANR 卡死，定义详见 Android 官网 ANR。
      - iOS 启动卡死：App 启动时主线程 **5 秒** 未执行完一个方法。
      - iOS ANR 卡死：App 运行时主线程 **5 秒** 未执行完一个方法。
    - **卡顿埋点**：卡顿是指主线程超过一定时间（Android **2.25 秒**，iOS **2 秒**）未执行完一个方法。卡顿埋点记录 App 卡顿及相关错误日志。
    - **闪退埋点**：记录 App 闪退及错误堆栈。
- H5 和 PC 埋点
  - 页面埋点：自动记录网页的打开、来源等信息，用于统计页面 PV、UV、来源去向等指标。

- 点击埋点：记录网页页面内某个按钮、链接的点击操作。
- 曝光埋点：记录网页页面内某段内容的曝光情况。

10.5.2. Android/iOS 自定义事件埋点

本文分别对自定义事件埋点的客户端和服务端日志模型进行说明。

> **说明**
> 若字段含义为“-”，则表明该字段未被使用，您无需关心。

客户端日志模型

> **说明**
> 客户端和后端共用一份日志模型。

序号	示例	字段含义
00	D-VM	日志头，固定为 D-VM。
01	2018-12-19 10:35:47.196	客户端日志时间。
02	A111111****11_IOS-default	格式：后台创建的应用 ID_应用平台-workspaceId
03	1.0.1	客户端版本，即应用版本。
04	2	日志埋点 SDK 版本，固定为2。
05	411111111****11|811111****11111	格式：IMSI|IMEI。
06	d5557b75-ff80-4aab-86a6-9b1a522b****	会话ID。
07	user****	客户端C端用户注册后产生的ID，即用户 ID。
08	event	固定为event。
09	-	-
10	first	格式：上一级页面 ID|上一级页面位置 ID；若当前页为首页则该字段为“-”或“first”。
11	-	-

12	-	-
13	-	-
14	-	-
15	PayResults	自定义事件 ID
16	-	-
17	Pay	自定义业务码，用来控制日志上传策略。
18	-	-
19	-	-
20	-	-
21	-	-
22	time=2018-07-27^amount=10.05	用于存储自定义属性和属性值，格式：key=value^key=value...。
23	-	-
24	-	-
25	Wn11111111111111111****	UTDID，设备ID。
26	-	-
27	-	-
28	-	-
29	-	-
30	-	-
31	-	-

32	-	-
33	vivo Xplay3S	设备型号。
34	4.4.2	操作系统版本。
35	WIFI	网络类型，如：Wi-Fi、2G、3G、4G。
36	-	-
37	alipay	渠道号。
38	zh-CN	操作系统语言。
39	0	热修复包版本号。
40	4	CPU 核数。
41	2265	CPU最大频率，单位：MHz。
42	2853	内存大小，单位：MB。
43	-	-
44	-	-
45	2560x1440	屏幕分辨率。
46	-	-
47	-	-

服务端日志模型

序号	示例	字段含义
01	2018-12-19 10:35:47.996	服务端日志时间。
02	ip=182.11.xx.xx^country=中国^province=北京市^city=北京市^district=朝阳区	包含服务端请求 IP、国家、省份、城市等信息。

03	2018-12-19 10:35:47.196	客户端日志时间。
04	A11111111****_IOS-default	格式：后台创建的应用 ID_应用平台-workspaceId。
05	1.0.1	客户端版本，即应用版本。
06	2	日志埋点 SDK 版本，固定为 2。
07	41111111111****|81111111111****	格式：IMSI|IMEI
08	d5557b75-ff80-4aab-86a6-9b1a522b****	会话 ID。
09	user****	客户端 C 端用户注册后产生的 ID，即用户 ID。
10	event	固定为 event。
11	-	-
12	first	格式：上一级页面 ID|上一级页面位置 ID；若当前页为首页则该字段为“-”或“first”。
13	-	-
14	-	-
15	-	-
16	-	-
17	PayResults	自定义事件 ID。
18	-	-
19	Pay	自定义业务码，用来控制日志上传策略。
20	-	-
21	-	-

22	-	-
23	-	-
24	time=2018-07-27^amount=10.05	用于存储自定义属性和属性值，格式：key=value^key=value...。
25	-	-
26	-	-
27	Wn111111111111111****	UTDID，设备 ID。
28	-	-
29	-	-
30	-	-
31	-	-
32	-	-
33	-	-
34	-	-
35	vivo Xplay3S	设备型号。
36	4.4.2	操作系统版本。
37	WIFI	网络类型，如：Wi-Fi、2G、3G、4G。
38	-	-
39	alipay	渠道号。
40	zh-CN	操作系统语言。
41	0	热修复包版本号。

42	4	CPU 核数。
43	2265	CPU 最大频率，单位：MHz。
44	2853	内存大小，单位：MB。
45	-	-
46	-	-
47	2560x1440	屏幕分辨率。
48	-	-
49	-	-

10.5.3. Android/iOS 行为埋点

本文分别对行为埋点的客户端和服务端日志模型进行说明。

行为埋点包括：

- 报活埋点
- 页面自动埋点
- 压后台埋点

> **说明**
> 若字段含义为“-”，则表明该字段未被使用，您无需关心。

客户端日志模型

> **说明**
> 客户端和后端共用一份日志模型。

报活埋点

序号	示例	字段含义
00	D-VM	日志头，固定为 D-VM。
01	2018-12-19 10:35:47.196	客户端日志时间
02	A11111111****_IOS-default	格式：后台创建的应用 ID_应用平台-workspaceId

03	1.0.1	客户端版本，即应用版本。
04	2	日志埋点 SDK 版本，固定为 2。
05	41111111111****|81111111111****	格式：IMSI|IMEI，仅 Android 设备有值。 **说明** • Android 10 及以上系统，不获取 IMSI 和 IMEI。 • Android 10 以下系统，在已获得 READ_PHONE_STATE 权限和用户同意隐私协议的前提下会获取 IMSI 和 IMEI。 • 不获取 IMSI 和 IMEI 时使用安装后首次启动生成的时间戳代替。
06	d5557b75-ff80-4aab-86a6-9b1a522b****	会话 ID。
07	user****	客户端 C 端用户注册后产生的 ID，即用户 ID。
08	event	用于标识行为类型，固定为 event。
09	-	-
10	first	格式：上一级页面 ID|上一级页面位置 ID；若当前页为首页则该字段为“-”或“first”。
11	-	-
12	-	-
13	-	-
14	-	-
15	reportActive	固定为 reportActive。
16	-	-
17	-	-
18	-	-

19	-	-
20	-	-
21	-	-
22	-	-
23	-	-
24	-	-
25	Wn11111111111111111*** *	UTDID，设备 ID。
26	-	-
27	-	-
28	-	-
29	-	-
30	-	-
31	-	-
32	-	-
33	vivo Xplay3S	设备型号。
34	4.4.2	操作系统版本。
35	WIFI	网络类型，如：Wi-Fi、2G、3G、4G。
36	-	-
37	alipay	渠道号。
38	zh-CN	操作系统语言。

39	0	热修复包版本号。
40	4	CPU 核数。
41	2265	CPU 最大频率，单位：MHz。
42	2853	内存大小，单位：MB。
43	-	-
44	-	-
45	2560x1440	屏幕分辨率。
46	-	-
47	-	-

页面自动埋点

序号	示例	字段含义
00	D-VM	日志头，固定为 D-VM。
01	2018-12-19 10:35:47.196	客户端日志时间。
02	A11111111****_IOS-default	格式：后台创建的应用 ID_应用平台-workspaceId。
03	1.0.1	客户端版本，即应用版本。
04	2	日志埋点 SDK 版本，固定为2。
05	41111111111****|811111111111****	格式：IMSI|IMEI。
06	d5557b75-ff80-4aab-86a6-9b1a522b****	会话 ID
07	user****	客户端C端用户注册后产生的 ID，即用户ID。
08	auto_openPage	用于标识行为类型，固定为 auto_openPage。

09	-	-
10	Referer_Page_ID|Referer_Page_Position	格式：上一级页面 ID|上一级页面位置 ID；若当前页为首页则该字段为“-”或“first”。
11	-	-
12	-	-
13	-	-
14	-	-
15	Current_Page_ID	当前页面ID。
16	-	-
17	-	-
18	-	-
19	-	-
20	-	-
21	-	-
22	-	-
23	-	-
24	9180	页面停留时长，单位：毫秒。
25	Wn111111111111111****	UTDID，设备ID。
26	-	-
27	Current_Page_Position	当前页面位置ID。
28	-	-

29	-	-
30	-	-
31	-	-
32	-	-
33	vivo Xplay3S	设备型号。
34	4.4.2	操作系统版本。
35	WIFI	网络类型，如：Wi-Fi、2G、3G、4G。
36	-	-
37	alipay	渠道号。
38	zh-CN	操作系统语言。
39	0	热修复包版本号。
40	4	CPU核数。
41	2265	CPU最大频率，单位：MHz。
42	2853	内存大小，单位：MB。
43	-	-
44	-	-
45	2560x1440	屏幕分辨率。
46	-	-
47	-	-

压后台埋点

序号	示例	字段含义
00	D-VM	日志头，固定为 D-VM。
01	2018-12-19 10:35:47.196	客户端日志时间。
02	A11111111****_IOS-default	格式：后台创建的应用 ID_应用平台-workspaceId。
03	1.0.1	客户端版本，即应用版本。
04	2	日志埋点 SDK 版本，固定为 2。
05	41111111111****|81111111111****	格式：IMSI|IMEI。
06	d5557b75-ff80-4aab-86a6-9b1a522****	会话ID。
07	user****	客户端 C 端用户注册后产生的 ID，即用户 ID。
08	auto_event	用于标识行为类型，固定为 auto_event。
09	-	-
10	first	格式：上一级页面 ID|上一级页面位置 ID；若当前页为首页则该字段为“-”或“first”。
11	-	-
12	-	-
13	-	-
14	-	-
15	-	-
16	-	-
17	-	-

18	-	-
19	-	-
20	-	-
21	leavehint	固定为leavehint。
22	uid=1****^pid=1****^stayTime=266083	格式：key=value^key=value... 其中 stayTime 为页面停留时间，单位：毫秒。
23	-	-
24	-	-
25	Wn1111111111111****	UTDID，设备ID。
26	-	-
27	-	-
28	-	-
29	-	-
30	-	-
31	-	-
32	-	-
33	vivo Xplay3S	设备型号。
34	4.4.2	操作系统版本。
35	WIFI	网络类型，如：Wi-Fi、2G、3G、4G。
36	-	-
37	alipay	渠道号。

38	zh-CN	操作系统语言。
39	0	热修复包版本号。
40	4	CPU 核数。
41	2265	CPU 最大频率，单位：MHz。
42	2853	内存大小，单位：MB。
43	-	-
44	-	-
45	2560x1440	屏幕分辨率。
46	-	-
47	-	-

服务端日志模型

报活埋点

序号	示例	字段含义
01	2018-12-19 10:35:47.996	服务端日志时间。
02	ip=182.xx.xxx.7^country=中国^province=北京市^city=北京市^district=朝阳区	包含服务端请求 IP、国家、省份、城市等信息。
03	2018-12-19 10:35:47.196	客户端日志时间
04	A11111111****_IOS-default	格式：后台创建的应用 ID_应用平台-workspaceId
05	1.0.1	客户端版本，即应用版本。
06	2	日志埋点 SDK 版本，固定为 2。

07	41111111111****|81111111111****	格式：IMSI|IMEI，仅 Android 设备有值。 **说明** • Android 10 及以上系统，不获取 IMSI 和 IMEI。 • Android 10 以下系统，在已获得 READ_PHONE_STATE 权限和用户同意隐私协议的前提下会获取 IMSI 和 IMEI。 • 不获取 IMSI 和 IMEI 时使用安装后首次启动生成的时间戳代替。
08	d5557b75-ff80-4aab-86a6-9b1a522b****	会话ID。
09	user****	客户端 C 端用户注册后产生的 ID，即用户ID。
10	event	用于标识行为类型，固定为event。
11	-	-
12	first	格式：上一级页面 ID|上一级页面位置 ID；若当前页为首页则该字段为“-”或“first”。
13	-	-
14	-	-
15	-	-
16	-	-
17	reportActive	固定为 reportActive。
18	-	-
19	-	-
20	-	-
21	-	-
22	-	-

23	-	-
24	-	-
25	-	-
26	-	-
27	Wn111111111111****	UTDID，设备ID。
28	-	-
29	-	-
30	-	-
31	-	-
32	-	-
33	-	-
34	-	-
35	vivo Xplay3S	设备型号。
36	4.4.2	操作系统版本。
37	WIFI	网络类型，如：Wi-Fi、2G、3G、4G。
38	-	-
39	alipay	渠道号。
40	zh-CN	操作系统语言。
41	0	热修复包版本号。
42	4	CPU 核数。

43	2265	CPU 最大频率，单位：MHz。
44	2853	内存大小，单位：MB。
45	-	-
46	-	-
47	2560x1440	屏幕分辨率。
48	-	-
49	-	-

页面自动埋点

序号	示例	字段含义
01	2018-12-19 10:35:47.996	服务端日志时间。
02	ip=182.xx.xxx.7^country=中国^province=北京市^city=北京市^district=朝阳区	包含服务端请求 IP、国家、省份、城市等信息。
03	2018-12-19 10:35:47.196	客户端日志时间。
04	A11111111****_IOS-default	格式：后台创建的应用 ID_应用平台-workspaceId。
05	1.0.1	客户端版本，即应用版本。
06	2	日志埋点 SDK 版本，固定为 2。
07	41111111111****|811111111111****	格式：IMSI|IMEI。
08	d5557b75-ff80-4aab-86a6-9b1a522b****	会话ID。
09	user****	客户端 C 端用户注册后产生的 ID，即用户 ID。
10	auto_openPage	用于标识行为类型，固定为 auto_openPage。

11	-	-
12	Referer_Page_ID|Referer_Page_Position	格式：上一级页面 ID|上一级页面位置 ID；若当前页为首页则该字段为“-”或“first”。
13	-	-
14	-	-
15	-	-
16	-	-
17	Current_Page_ID	当前页面ID。
18	-	-
19	-	-
20	-	-
21	-	-
22	-	-
23	-	-
24	-	-
25	-	-
26	9180	页面停留时长，单位：毫秒。
27	Wn11111111111111****	UTDID，设备ID。
28	-	-
29	Current_Page_Position	当前页面位置ID。
30	-	-

31	-	-
32	-	-
33	-	-
34	-	-
35	vivo Xplay3S	设备型号。
36	4.4.2	操作系统版本。
37	WIFI	网络类型，如：Wi-Fi、2G、3G、4G。
38	-	-
39	alipay	渠道号。
40	zh-CN	操作系统语言。
41	0	热修复包版本号。
42	4	CPU 核数。
43	2265	CPU 最大频率，单位：MHz。
44	2853	内存大小，单位：MB。
45	-	-
46	-	-
47	2560x1440	屏幕分辨率。
48	-	-
49	-	-

压后台埋点

序号	示例	字段含义
01	2018-12-19 10:35:47.996	服务端日志时间。
02	ip=182.xx.xxx.7^country=中国^province=北京市^city=北京市^district=朝阳区	包含服务端请求 IP、国家、省份、城市等信息。
03	2018-12-19 10:35:47.196	客户端日志时间。
04	A111111111111_IOS-default	格式：后台创建的应用 ID_应用平台-workspaceId。
05	1.0.1	客户端版本，即应用版本。
06	2	日志埋点 SDK 版本，固定为 2。
07	411111111111111|811111111111111	格式：IMSI|IMEI。
08	d5557b75-ff80-4aab-86a6-9b1a522bbbce	会话 ID。
09	user11	客户端 C 端用户注册后产生的 ID，即用户 ID。
10	auto_event	用于标识行为类型，固定为 auto_event。
11	-	-
12	first	格式：上一级页面 ID|上一级页面位置 ID；若当前页为首页则该字段为“-”或“first”。
13	-	-
14	-	-
15	-	-
16	-	-
17	-	-

18	-	-
19	-	-
20	-	-
21	-	-
22	-	-
23	leavehint	固定为 leavehint。
24	uid=10206^pid=16992^stayTime=266083	格式：key=value^key=value... 其中 stayTime 为页面停留时间，单位：毫秒。
25	-	-
26	-	-
27	Wn11111111111111****	UTDID，设备 ID。
28	-	-
29	-	-
30	-	-
31	-	-
32	-	-
33	-	-
34	-	-
35	vivo Xplay3S	设备型号。
36	4.4.2	操作系统版本。
37	WIFI	网络类型，如：Wi-Fi、2G、3G、4G。

38	-	-
39	alipay	渠道号。
40	zh-CN	操作系统语言。
41	0	热修复包版本号。
42	4	CPU 核数。
43	2265	CPU最大频率，单位：MHz。
44	2853	内存大小，单位：MB。
45	-	-
46	-	-
47	2560x1440	屏幕分辨率。
48	-	-
49	-	-

10.5.4. Android/iOS 性能埋点

本文分别对性能埋点的客户端和服务端日志模型进行说明。

性能埋点包括：

- 启动埋点
- 闪退埋点
- 卡顿埋点
- 卡死埋点

> **说明**
> - 若字段含义为“-”，则表明该字段未被使用，您无需关心。
> - 对于设备数，如果设备 ID 为空或者为“-”，设备数不累加。

客户端日志模型

> **说明**
> 客户端和后端共用一份日志模型。

启动埋点

序号	示例	字段含义
00	D-MM	日志头，固定为 D-MM。
01	2018-12-19 10:35:47.196	客户端日志时间
02	A11111111****_IOS-default	格式：后台创建的应用 ID_应用平台-workspaceId
03	1.0.1	客户端版本，即应用版本。
04	2	日志埋点 SDK 版本，固定为 2。
05	41111111111****|81111111111****	格式：IMSI|IMEI
06	d5557b75-ff80-4aab-86a6-9b1a522b****	会话 ID
07	user****	客户端 C 端用户注册后产生的 ID，即用户 ID。
08	-	-
09	-	-
10	-	-
11	performance	固定为 performance
12	time_startup	固定为 time_startup
13	3785	启动速度，单位：毫秒
14	1	是否首次启动：0 首次，1 非首次。
15	-	-

16	-	-
17	android	操作系统类别
18	4.4.2	操作系统版本
19	WIFI	网络类型，如： Wi-Fi、2G、3G、4G。
20	vivo Xplay3S	设备型号
21	-	-
22	alipay	渠道号
23	Wn1111111111111****	UTDID，设备 ID
24	zh-CN	操作系统语言
25	4	CPU 核数
26	2265	CPU 最大频率，单位：MHz
27	2853	内存大小，单位：MB
28	0	热修复包版本号
29	-	-
30	-	-
31	-	-
32	2560x1440	屏幕分辨率
33	-	-
34	-	-
35	-	-

36	-	-

闪退埋点

序号	示例	字段含义
00	e	日志头，固定为 e。
01	2018-12-19 10:35:47.196	客户端日志时间
02	A11111111****_IOS-default	格式：后台创建的应用 ID_应用平台-workspaceId
03	1.0.1	客户端版本，即应用版本。
04	-	-
05	41111111111****|81111111111****	格式：IMSI|IMEI
06	d5557b75-ff80-4aab-86a6-9b1a522b****	会话 ID
07	user****	客户端 C 端用户注册后产生的 ID，即用户 ID。
08	exception	固定为 exception。
09	-	-
10	-	-
11	-	-
12	-	-
13	MonitorPoint_Crash	固定为 MonitorPoint_Crash。
14	java.lang.RuntimeException:xx	异常堆栈
15	-	-
16	alipay	渠道号

17	-	-
18	-	-
19	-	-
20	-	-
21	-	-
22	-	-
23	vivo Xplay3S	设备型号
24	4.4.2	操作系统版本
25	WIFI	网络类型，如：Wi-Fi、2G、3G、4G。
26	-	-
27	-	-
28	-	-
29	-	-
30	-	-
31	-	-
32	-	-
33	-	-
34	-	-
35	-	-
36	-	-

37	-	-

卡顿埋点

序号	示例	字段含义
00	D-EM	日志头，固定为 D-EM。
01	2018-12-19 10:35:47.196	客户端日志时间
02	A11111111****_IOS-default	格式：后台创建的应用 ID_应用平台-workspaceId
03	1.0.1	客户端版本，即应用版本。
04	2	日志埋点 SDK 版本，固定为 2。
05	41111111111****|81111111111****	格式：IMSI|IMEI
06	user****11	客户端 C 端用户注册后产生的 ID，即用户 ID。
07	WIFI	网络类型，如：Wi-Fi、2G、3G、4G。
08	vivo Xplay3S	设备型号
09	7.1.2	操作系统版本
10	-	-
11	alipay	渠道号
12	-	-
13	-	-
14	performance	固定为 performance。
15	lag	固定值。如果是 Android 设备，该值为 lag；如果是 iOS 设备，该值为 lag_stack。

16	stackFrame=xx	具体错误堆栈
17	-	-
18	-	-
19	-	-
20	-	-
21	-	-
22	-	-
23	Wn11111111111****QxL	UTDID，设备 ID
24	-	-
25	-	-
26	-	-
27	-	-

卡死埋点

序号	示例	字段含义
00	D-MM	日志头，固定为 D-MM。
01	2018-12-19 10:35:47.196	客户端日志时间
02	A11111111****_IOS-default	格式：后台创建的应用 ID_应用平台-workspaceId
03	1.0.1	客户端版本，即应用版本。
04	2	日志埋点 SDK 版本，固定为 2。
05	41111111111****|81111111111****	格式：IMSI|IMEI

06	d5557b75-ff80-4aab-86a6-9b1a522bbbce	会话 ID
07	user****	客户端 C 端用户注册后产生的 ID，即用户 ID。
08	-	-
09	-	-
10	-	-
11	keybiztrace	固定为 keybiztrace。
12	BizCanNotUse	固定为 BizCanNotUse。
13	BIZ_APM	BIZ_FRAME 为启动卡死；BIZ_APM 为 ANR 卡死。
14	APM_ANR	FRAME_CLIENT_STARTUP_DEAD 为启动卡死；APM_ANR 为 ANR 卡死。
15	1000	1111 为启动卡死；1114 为 iOS ANR 卡死；1000 为 Android ANR 卡死。
16	historyStacks=xxx	具体错误堆栈
17	android	操作系统类别
18	4.4.2	ANR 卡死时为操作系统版本。
19	WIFI	网络类型，如：Wi-Fi、2G、3G、4G。
20	vivo Xplay3S	ANR 卡死时为设备型号。
21	-	-
22	alipay	渠道号
23	Wn1111111111111111****xL	UTDID，设备 ID

24	zh-CN	操作系统语言
25	4	CPU 核数
26	2265	CPU 最大频率，单位：MHz
27	2853	内存大小，单位：MB
28	0	热修复包版本号
29	-	-
30	-	-
31	-	-
32	2560x1440	屏幕分辨率
33	-	-
34	-	-
35	-	-
36	-	-

服务端日志模型

启动埋点

序号	示例	字段含义
01	2018-12-19 10:35:47.996	服务端日志时间
02	ip=182.11.xx.xx^country=中国^province=北京市^city=北京市^district=朝阳区	包含服务端请求 IP、国家、省份、城市等信息。
03	2018-12-19 10:35:47.196	客户端日志时间
04	A11111111****_IOS-default	格式：后台创建的应用 ID_应用平台-workspaceId

05	1.0.1	客户端版本，即应用版本。
06	2	日志埋点 SDK 版本，固定为 2。
07	41111111111****|81111111111****	格式：IMSI|IMEI
08	d5557b75-ff80-4aab-86a6-9b1a522b****	会话 ID
09	user****	客户端 C 端用户注册后产生的 ID，即用户 ID。
10	-	-
11	-	-
12	-	-
13	performance	固定为 performance
14	time_startup	固定为 time_startup
15	3785	启动速度，单位：毫秒
16	1	是否首次启动：0 首次，1 非首次。
17	-	-
18	-	-
19	android	操作系统类别
20	4.4.2	操作系统版本
21	WIFI	网络类型，如： Wi-Fi、2G、3G、4G。
22	vivo Xplay3S	设备型号
23	-	-
24	alipay	渠道号

25	Wn111111111111111****xL	UTDID，设备 ID
26	zh-CN	操作系统语言
27	4	CPU 核数
28	2265	CPU 最大频率，单位：MHz
29	2853	内存大小，单位：MB
30	0	热修复包版本号
31	-	-
32	-	-
33	-	-
34	2560x1440	屏幕分辨率
35	-	-
36	-	-
37	-	-
38	-	-

闪退埋点

序号	示例	字段含义
01	2018-12-19 10:35:47.996	服务端日志时间
02	ip=182.11.xx.xx^country=中国^province=北京市^city=北京市^district=朝阳区	包含服务端请求 IP、国家、省份、城市等信息。
03	2018-12-19 10:35:47.196	客户端日志时间

04	A11111111****_IOS-default	格式：后台创建的应用 ID_应用平台-workspaceId
05	1.0.1	客户端版本，即应用版本。
06	-	-
07	41111111111****|8111111****1111	格式：IMSI|IMEI
08	d5557b75-ff80-4aab-86a6-9b1a522b****	会话 ID
09	user****	客户端 C 端用户注册后产生的 ID，即用户 ID。
10	exception	固定为 exception。
11	-	-
12	-	-
13	-	-
14	-	-
15	MonitorPoint_Crash	固定为 MonitorPoint_Crash。
16	java.lang.RuntimeException:xx	异常堆栈
17	-	-
18	alipay	渠道号
19	-	-
20	-	-
21	-	-
22	-	-
23	-	-

24	-	-
25	vivo Xplay3S	设备型号
26	4.4.2	操作系统版本
27	WIFI	网络类型，如：Wi-Fi、2G、3G、4G。
28	-	-
29	-	-
30	-	-
31	-	-
32	-	-
33	-	-
34	-	-
35	-	-
36	-	-
37	-	-
38	-	-
39	-	-

卡顿埋点

序号	示例	字段含义
01	2018-12-19 10:35:47.996	服务端日志时间。
02	ip=182.11.xx.xx^country=中国^province=北京市^city=北京市^district=朝阳区	包含服务端请求 IP、国家、省份、城市等信息。

03	2018-12-19 10:35:47.196	客户端日志时间
04	A11111111****_IOS-default	格式：后台创建的应用 ID_应用平台-workspaceId
05	1.0.1	客户端版本，即应用版本。
06	2	日志埋点 SDK 版本，固定为 2。
07	41111111111****|81111111111****	格式：IMSI|IMEI
08	user****	客户端 C 端用户注册后产生的 ID，即用户 ID。
09	WIFI	网络类型，如：Wi-Fi、2G、3G、4G。
10	vivo Xplay3S	设备型号
11	7.1.2	操作系统版本
12	-	-
13	alipay	渠道号
14	-	-
15	-	-
16	performance	固定为 performance。
17	lag	固定值。如果是 Android 设备，该值为 lag；如果是 iOS 设备，该值为 lag_stack。
18	stackFrame=xx	具体错误堆栈
19	-	-
20	-	-
21	-	-

22	-	-
23	-	-
24	-	-
25	-	-
26	-	-
27	Wn1111111111111111****xL	UTDID，设备 ID。
28	-	-
29	-	-

卡死埋点

序号	示例	字段含义
01	2018-12-19 10:35:47.996	服务端日志时间。
02	ip=182.11.xx.xx^country=中国^province=北京市^city=北京市^district=朝阳区	包含服务端请求 IP、国家、省份、城市等信息。
03	2018-12-19 10:35:47.196	客户端日志时间。
04	A11111111****_IOS-default	格式：后台创建的应用 ID_应用平台-workspaceId。
05	1.0.1	客户端版本，即应用版本。
06	2	日志埋点 SDK 版本，固定为 2。
07	41111111111****|81111111111****	格式：IMSI|IMEI
08	d5557b75-ff80-4aab-86a6-9b1a522b****	会话 ID
09	user****	客户端 C 端用户注册后产生的 ID，即用户 ID。

10	-	-
11	-	-
12	-	-
13	keybiztrace	固定为 keybiztrace。
14	BizCanNotUse	固定为 BizCanNotUse。
15	BIZ_APM	BIZ_FRAME 为启动卡死；BIZ_APM 为 ANR 卡死。
16	APM_ANR	FRAME_CLIENT_STARTUP_DEAD 为启动卡死；APM_ANR 为 ANR 卡死。
17	1000	1111 为启动卡死；1114 为 iOS ANR 卡死；1000 为 Android ANR 卡死。
18	historyStacks=xxx	具体错误堆栈
19	android	操作系统类别
20	4.4.2	ANR 卡死时为操作系统版本。
21	WIFI	网络类型，如：Wi-Fi、2G、3G、4G。
22	vivo Xplay3S	ANR 卡死时为设备型号。
23	-	-
24	alipay	渠道号
25	Wn1111111111111****xL	UTDID，设备 ID。
26	zh-CN	操作系统语言。
27	4	CPU 核数。
28	2265	CPU 最大频率，单位：MHz。

29	2853	内存大小，单位：MB。
30	0	热修复包版本号。
31	-	-
32	-	-
33	-	-
34	2560x1440	屏幕分辨率。
35	-	-
36	-	-
37	-	-
38	-	-

10.5.5. H5/PC 页面埋点

H5/PC 页面埋点模型如下：

> **说明**
> 若字段含义为“-”，则表明该字段未被使用，您无需关心。客户端上报的日志头为 D-VM，服务端处理后会删除掉。

序号	示例	字段含义
01	2018-12-19 10:35:47.996	服务端日志时间。该字段由服务端填入，客户端日志中不包含此字段。
02	ip=182.11.XX.XX^country=中国^province=北京市^city=北京市^district=朝阳区	包含服务端请求 IP、国家、省份、城市等信息。由服务端填入，客户端日志中不包含此字段。
03	2018-12-19 10:35:47.196	客户端日志时间。
04	A11111111****_IOS-default	格式：后台创建的应用 ID_应用平台-workspaceId。

序号	示例	字段含义
05	1.0.1	客户端版本，即应用版本。
06	2	日志埋点 SDK 版本，固定为2。
07	-	-
08	9ac4990a-7d4d-45fc-9f1d-b0963d9f****	会话 ID，基于 session 生成（非 mPaaS H5 容器，基于 UUID 生成，存放于 sessionStorage 中）。
09	userId	手动传入的 userId。请参考 配置通用埋点。
10	auto_openPage	固定为 auto_openPage。
11	-	-
12	file:///Users/work/mtracker/mtrackerRefer.htm|-	上一级页面 ID。若无页面信息，则显示“-”。
13	-	-
14	-	-
15	-	-
16	-	-
17	file:///Users/work/mtracker/mtrackerDemo.htm	当前页面 URL。
18	-	-
19	UserBehaviorH5	业务码，默认值为 UserBehaviorH5。
20	-	-
21	-	-
22	-	-

序号	示例	字段含义
23	-	-
24	userAgent=Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_0) AppleWebKit/537.36 (KHTML%2C like Gecko) Chrome/81.0.XX.XXSafari/537.36^fullURL=file:///Users/work/mtracker/mtrackerDemo.htm^mBizScenario=mPaaS	格式：key=value^key=value... 其中： • userAgent：设备型号、浏览器。 • mBizScenario：埋点时传入的 bizScenario（渠道来源），请参考 配置通用埋点。 • fullURL：页面 URL。
25	-	-
26	-	-
27	267aa54d-a1f3-472e-a36f-e188f4732f42-154356150****	设备 ID，基于 localStorage 生成。
28	-	-
29	-	-
30	-	-
31	-	-
32	-	-
33	-	-
34	-	-
35	Mac OS X	设备型号。
36	10_14_2	操作系统版本。
37	-	-
38	-	-

序号	示例	字段含义
39	-	-
40	zh-CN	操作系统语言。
41	-	-
42	-	-
43	-	-
44	-	-
45	-	-
46	-	-
47	320x568	屏幕分辨率。
48	-	-
49	-	-

10.5.6. H5/PC 点击埋点

H5/PC 点击埋点模型如下：

> **说明**
> 若字段含义为“-”，则表明该字段未被使用，您无需关心。客户端上报的日志头为 D-VM，服务端处理后会删除掉。

序号	示例	字段含义
01	2018-12-19 10:35:47.996	服务端日志时间。该字段由服务端填入，客户端日志中不包含此字段。
02	ip=182.11.xx.xx^country=中国^province=北京市^city=北京市^district=朝阳区	包含服务端请求 IP、国家、省份、城市等信息。由服务端填入，客户端日志中不包含此字段。

03	2018-12-19 10:35:47.196	客户端日志时间。
04	A11111111****_IOS-default	格式：后台创建的应用 ID_应用平台-workspaceId。
05	1.0.1	客户端版本，即应用版本。
06	2	日志埋点 SDK 版本，固定为 2。
07	-	-
08	9ac4990a-7d4d-45fc-9f1d-b0963d9f****	会话 ID，基于 session 生成。
09	userId	手动传入的 userId。请参考 H5 通用埋点类型。
10	clicked	固定为 clicked。
11	-	-
12	-	-
13	-	-
14	-	-
15	-	-
16	-	-
17	h5testSeed	点击按钮 seedName。请参考 H5 通用埋点类型。
18	-	-
19	UserBehaviorH5	业务码，默认值为 UserBehaviorH5。
20	-	-
21	-	-

22	-	-
23	-	-
24	userAgent=Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_0) AppleWebKit/537.36 (KHTML%2C like Gecko) Chrome/81.0.xx.xxSafari/537.36^fullURL =file:///Users/work/mtracker/mtrackerDemo.htm^mBizScenario=mPaaS	格式：`key=value^key=value…` 其中： • userAgent：设备型号、浏览器。 • mBizScenario：埋点时传入的 bizScenario（渠道来源），详细信息，请参见 配置通用埋点。 • fullURL：页面 URL。
25	-	-
26	-	-
27	267aa54d-a1f3-472e-a36f-e188f4732f42-154356150****	设备 ID，基于 localStorage 生成。
28	-	-
29	-	-
30	-	-
31	-	-
32	-	-
33	-	-
34	-	-
35	Mac OS X	设备型号。
36	10_14_2	操作系统版本。
37	-	-
38	-	-

39	-	-
40	zh-CN	操作系统语言。
41	-	-
42	-	-
43	-	-
44	-	-
45	-	-
46	-	-
47	320x568	屏幕分辨率。
48	-	-
49	-	-

10.5.7. H5/PC 曝光埋点

H5/PC 曝光埋点模型见下表。

> **说明**
> 若字段含义为“-”，则表明该字段未被使用，您无需关心。客户端上报的日志头为 D-VM，服务端处理后会删除掉。

序号	示例	字段含义
01	2018-12-19 10:35:47.996	服务端日志时间。该字段由服务端填入，客户端日志中不包含此字段。
02	ip=182.xx.xxx.7^country=中国^province=北京市^city=北京市^district=朝阳区	包含服务端请求 IP、国家、省份、城市等信息。由服务端填入，客户端日志中不包含此字段。
03	2018-12-19 10:35:47.196	客户端日志时间

序号	示例	字段含义
04	A111111111111_IOS-default	格式：后台创建的应用 ID_应用平台-workspaceId
05	1.0.1	客户端版本，即应用版本
06	2	日志埋点 SDK 版本，固定为 2。
07	-	-
08	9ac4990a-7d4d-45fc-9f1d-b0963d9fe62e,	会话 ID，基于 session 生成。
09	userId	手动传入的 userId。请参考 H5 通用埋点类型。
10	exposure	固定为 exposure。
11	-	-
12	-	-
13	-	-
14	-	-
15	-	-
16	-	-
17	h5testSeed	点击按钮 seedName。请参考 H5 通用埋点类型。
18	-	-
19	UserBehaviorH5	业务码，默认值为 UserBehaviorH5。
20	-	-
21	-	-
22	-	-

序号	示例	字段含义
23	-	-
24	userAgent=Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_0) AppleWebKit/537.36 (KHTML%2C like Gecko) Chrome/81.0.4044.138 Safari/537.36^fullURL=file:///Users/work/mtracker/mtrackerDemo.htm^mBizScenario=mPaaS	格式为“key=value^key=value...”，其中： • userAgent：设备型号、浏览器 • mBizScenario：埋点时传入的 bizScenario（渠道来源），请参考 配置通用埋点。 • fullURL：页面 URL
25	-	-
26	-	-
27	267aa54d-a1f3-472e-a36f-e188f4732f42-1543561503926	设备 ID，基于 localStorage 生成。
28	-	-
29	-	-
30	-	-
31	-	-
32	-	-
33	-	-
34	-	-
35	Mac OS X	设备型号
36	10_14_2	操作系统版本
37	-	-
38	-	-

序号	示例	字段含义
39	-	-
40	zh-CN	操作系统语言
41	-	-
42	-	-
43	-	-
44	-	-
45	-	-
46	-	-
47	320x568	屏幕分辨率
48	-	-
49	-	-

11.H5 通用埋点

11.1. 通用埋点类型

Web 通用自动埋点解决方案（mtracker）是 PC 端和移动端 H5 页面通用的埋点方案。该方案通过设置标签属性来实现自动上报埋点，实现在移动端 H5 页面中使用该方案上报页面显示、点击、曝光埋点。

在配置通用埋点之前，您需要了解不同的业务场景使用的不同埋点类型：

- 访问量埋点
- 点击埋点
- 曝光埋点

访问量埋点

引入了 mtracker 之后，每次打开页面都会在日志中记录一次访问，不需要额外的操作。

点击埋点

记录网页页面内某个按钮、链接的点击操作。

标签埋点

- mtracker 会监听点击事件，自动为有 `data-seed` 属性的标签添加点击的日志。

```
<div data-seed="seedname"></div>
```

- 如果业务需要 `bizType` ，增加以 `data-biztype` 为前缀的属性，也会跟随 `seed` 埋入日志。

```
<div data-seed="seedname" data-biztype="xxx"></div>
```

- 如果业务需要额外的点击埋点信息，增加以 `data-mtr-` 为前缀的属性，也会跟随 `seed` 埋入日志，实际埋入的扩展属性会去掉 `data-mtr-` 的前缀。

```
// 上报的 seed 为 seedname，扩展属性为 extra1=111^extra2=222
<div data-seed="seedname" data-mtr-extra1="111" data-mtr-extra2="222" >
</div>
```

主动触发

有些业务场景需要手动触发埋点点击事件，比如点击了同一个标签后，要判断埋入不同的点，这个时候就需要用到 JS 点击埋点方法。

```
Tracker.click(eventId, options)
```

- 参数说明

参数	类型	示例	描述
eventId	String	clickseedname	事件 ID
options	Object	-	选项配置，包含 `bizType` 、 `ext` 。
bizType	String	Pay	业务码

参数	类型	示例	描述
ext	Object	`{ productId: 'xxx' }`	扩展参数

- 代码示例

```
Tracker.click('clickseedname', { bizType: 'Pay', ext: { productId: 'xxx' } });
```

曝光埋点

记录网页页面内某段内容的曝光情况。

如果需要手动触发埋点曝光事件，比如轮播图显示，需要用到 JS 曝光埋点方法 `Tracker. expo()` 。

```
Tracker.expo(eventId, options)
```

- 参数说明

参数	类型	示例	描述
eventId	String	clickseedname	事件 ID
options	Object	-	选项配置，包含 `bizType` 、 `ext` 。
bizType	String	Pay	业务码
ext	Object	`{ productId: 'xxx' }`	扩展参数

- 代码示例

```
Tracker.expo('exposeedname', { bizType: 'Pay', ext: { productId: 'xxx' } });
```

相关链接

- 配置通用埋点
- 分析通用埋点

11.2. 配置通用埋点

PC 端和移动端 H5 页面可以使用统一的 H5 埋点方案。通过配置埋点，您可以实现统一的 H5 埋点。

关于此任务

基于不同的业务场景，使用不同的 H5 通用埋点类型。要了解具体的业务场景及对应的埋点类型，查看 通用埋点类型。

操作步骤

1. 引入 CDN 版本的 mtracker。引入 mtracker 后，会在全局 `window` 中注入 `Tracker` 对象。

针对下载的 mtracker 埋点 JS 文件中文出现乱码的情况，正常使用该 JS 文件即可，不会影响埋点配置。该中文乱码问题应该是因文件编码格式与浏览器解释的编码格式不同所导致。

2. 初始化配置。根据不同场景，在 H5 埋点 JS 文件中注入相应的信息。
   - mPaaS 容器内，即 App 集成了 mPaaS H5 容器。

     代码示例：

     ```
     <script>
     window._to = {
       bizScenario: 'alipay',    // 选填，渠道来源，默认为空
       mtrDebug: true,          // 选填，默认为 false
     };
     </script>
     ```

参数	描述
bizScenario	渠道来源，默认为空，选填。
mtrDebug	是否开启 mtracker 的 debug 模式，在 debug 模式下会打印上报的日志，默认为 false 不开启。

   - mPaaS 容器外，即 App 未集成 mPaaS H5 容器或浏览器端。

     代码示例：

     ```
     <script>
     window._to = {
       server: 'https://cn-hangzhou-mas-log.cloud.alipay.com/loggw/webLog.do', // 必填，接受埋点的服务地址
       appId: 'xxxxxxxxxx', // 必填，App 唯一标识
       workspaceId: 'default', // 必填，环境标识
       h5version: '1.0.0',     // 必填，客户端 App/H5 页面版本
       userId: '1234567890',       // 选填，默认为空
       bizScenario: 'alipay',    // 选填，渠道来源，默认为空
       mtrDebug: true,          // 选填，默认为 false
       extendParams: { test: 111 } // 选填，全局扩展参数，默认为空，mtracker 1.2.0 版本以上支持
     };
     </script>
     ```

参数	描述
server	接受埋点的服务地址。
appId	App 唯一标识。
workspaceId	环境标识。
h5version	客户端 App 版本或 H5 页面版本。

userId	用户 ID，选填
bizScenario	渠道来源，默认为空，选填。
mtrDebug	是否开启 mtracker 的 debug 模式，在 debug 模式下会打印上报的日志，默认为 false 不开启。
extendParams	全局扩展参数，默认为空，mtracker 1.2.0 及以上版本支持。

3. 初始化 mtracker 对象。

   默认情况下，mtracker 在引入 JS 文件后会自动初始化并注入到 `window` 对象中。如某些场景中需要手动初始化，需按照下列步骤进行：

   i. 在引入 JS 文件的位置前加入如下代码，禁止自动初始化。

   ```
   window.notInitTrackerOnStart = true;
   ```

   ii. 添加初始化代码。

   ```
   window.initTracker();
   ```

> **说明**
> mtracker 1.2.0 版本起支持修改全局扩展参数，如需使用该功能，请先进行版本升级。

可根据需要修改全局扩展参数。通过在 `window._to` 中设置 `extendParams` 参数，可实现在之后的埋点上报中一直添加设置的扩展参数。若在 `click` 或 `expo` 方法中设置的 `ext` 值中，有重复属性名，传递的值以 `click` 或 `expo` 方法中的为准。

可调用以下代码改变已设置的 `extendParams`，新设置的对象将覆盖以往设置的 `extendParams` 所有值。

```
window.changeTrackerExtendParams({ newValue: 11111 });
```

后续操作

登录移动分析控制台，通过自定义分析页面，分析 mtracker 上报的 H5 通用埋点。具体操作步骤，请参见 分析通用埋点。

11.3. 分析通用埋点

使用移动分析组件提供的自定义分析功能分析 mtracker 上报的埋点。

前置条件

确保您已了解自定义分析功能。具体内容，参见 自定义分析。

关于此任务

添加 mtracker 相关的埋点事件时，埋点事件 ID 为自定义事件ID ，参见 通用埋点类型 中的 `eventId` 参数。

mtracker 埋点事件的预置扩展属性说明如下：

属性 ID	说明
userAgent	浏览器 `userAgent`
fullURL	当前页面的完整 URL
mBizScenario	渠道来源

操作步骤

1. 添加 mtracker 相关的埋点事件。具体的添加步骤，参见 配置事件。
2. 添加 mtracker 中的扩展属性。以 `data-mtr-` 为前缀的属性为自定义属性，可自定义添加，实际上报会去掉 `data-mtr-` 前缀。具体的添加步骤，参见 配置属性。
3. 通过分析事件来分析 mtracker 上报的数据。具体的事件分析步骤，参见 添加事件分析。
4. 通过自定义大盘展示数据。具体的大盘数据展示方法，参见 创建自定义大盘。

12.使用教程

12.1. 自定义事件分析

12.1.1. 教程场景说明

在您按照本教程进行操作前，需先了解以下概念：

概念	说明
事件	事件用于记录用户在 App 内的一个动作。您可以在任意动作（如按钮点击）触发时，埋入一个自定义事件。
事件 ID	用于唯一标识一个事件。事件是 App 全局的，因此，同一个 mPaaS 应用中事件 ID 必须 **唯一**。
属性	一个事件包含一些信息，如触发事件的用户 ID、App 版本、设备型号等，这些信息即为属性。移动分析平台预置了一些常用属性；此外，您可以根据实际情况自定义属性。
属性 ID	用于唯一标识一个属性。属性是 App 全局的，因此，同一个 mPaaS 应用中属性 ID 必须 **唯一**。
事件分析	事件及其属性信息会以日志的形式先存储在本地客户端，然后上报至移动分析服务器。在控制台完成相关配置和操作后，您可以查看事件分析报表。

教程场景说明

当用户完成一笔支付时，您可以记录一个 **支付完成事件**。事件包含 **支付时间**、**用户 ID**、**支付方式** 属性。

事件 ID 和 **属性 ID** 在客户端开发和控制台操作中都会用到，因此，Android 和 iOS 的开发者，以及控制台操作人员需要事先商定这些 ID。教程中假定：

- **事件 ID**：`PayResults`。
- **属性 ID**：`pay_time`、`user_id`、`payment_method`。

  自定义属性的 ID 不能和预置属性的 ID 重复。预置属性可以在 **mPaaS 控制台** > **移动分析** > **自定义分析** > **事件与属性配置** > **属性** 页面查看。

本教程将帮助您完成 **支付完成** 事件分析，包括：

1. 完成客户端开发：Android，iOS
2. 创建属性
3. 创建事件
4. 查看事件 PV 和 UV
5. 添加自定义分析
6. 添加大盘

如遇问题，可参见 常见问题 文档进行处理。

12.1.2. Android 客户端开发

本文将引导您完成 Android 客户端开发，包括：

1. 接入移动分析组件
2. 记录事件日志
3. 上报日志

1. 接入移动分析组件

参见 Android 接入文档 接入移动分析组件。

2. 记录事件日志

下面通过代码示例指导如何记录事件日志，并对涉及的相关参数进行说明。

示例代码

在下面的代码示例中，客户端将记录事件对应的业务 ID、事件 ID、支付时间、用户 ID 以及支付方式。

```
import com.mpaas.mas.adapter.api.MPLogger;

import java.util.HashMap;
import java.util.Map;

// 业务 ID
String bizType = "Pay";
// 事件 ID
String logId = "PayResults";
// 添加属性
Map<String, String> params = new HashMap<>(4);
// 属性：支付时间。Key 对应属性 ID；Value 对应属性值
params.put("pay_time", String.valueOf(System.currentTimeMillis()));
// 属性：用户 ID
params.put("user_id", "the-userId");
// 属性：支付方式
params.put("payment_method", "alipay");
// 打印日志
MPLogger.event(logId, bizType, params);
```

参数说明

参数	说明
bizType	业务 ID（也称为 **业务码**、**业务类型**），是业务的唯一标识。示例代码中设置为 `Pay`，意为支付业务。 `bizType` 影响客户端日志的文件名；日志文件名规则为 `时间戳_包名-进程_bizType`。
logId	事件 ID，是事件的唯一标识。更多信息，参见 教程场景说明。

参数	说明
params	用于存储事件关联的属性，输出格式：`params.put("param-key", "param-value")`。 • `param-key`：对应属性 ID。更多信息，参见 教程场景说明。 • `param-value`：对应属性的值。在客户端以字符串形式存储；在实际分析中，服务端支持自动转化为字符型、整型、浮点型。

3. 上报日志

默认情况下，当客户端本地日志缓存到一定条数或程序压后台超过一定时间后，本地日志会自动上报给移动分析服务器。在开发测试时，您可以调用以下接口，强制将本地日志立刻上报给服务器：

```
import com.mpaas.mas.adapter.api.MPLogger;
MPLogger.uploadAll();
```

相关链接

- 接入方式介绍
- 发布说明
- 问题排查

12.1.3. iOS 客户端开发

> **重要**
>
> 自 2020 年 6 月 28 日起，mPaaS 停止维护 10.1.32 基线。请使用 10.1.68 或 10.1.60 系列基线。可以参考 mPaaS 10.1.68 升级指南 或 mPaaS 10.1.60 升级指南 进行基线版本升级。

本文将引导您完成 iOS 客户端开发，包括：

1. 接入移动分析组件
2. 记录事件日志

1. 接入移动分析组件

参考 iOS 接入文档 接入移动分析组件。

2. 记录事件日志

下文将以 10.1.68 版本 SDK 为例，引导您记录事件日志。

示例代码

```
#import <MPMasAdapter/MPMasAdapter.h>

// 目前 actionId 只支持 KActionID_Event，您无需关心
NSString * actionId = KActionID_Event;
// 事件 ID
NSString * eventId = @"PayResults";
// 添加属性
NSMutableDictionary * extParam = [NSMutableDictionary dictionary];
// 属性：支付时间。Key 对应属性 ID；Value 对应属性值
[extParam setObject:@"2017-05-01 12:03:16" forKey:@"pay_time"];
// 属性：用户 ID
[extParam setObject:@"the-userId" forKey:@"user_id"];
// 属性：支付方式
[extParam setObject:@"alipay" forKey:@"payment_method"];

// 打印日志
[MPRemoteLoggingInterface writeLogWithActionId:actionId eventId:eventId extParam:extParam];
```

参数说明

参数	说明
eventId	事件 ID，是事件的唯一标识。更多信息，参见 教程场景说明。
extParam	事件属性，NSDictionary 类型的 `extParam` 用于存储事件关联的属性： • `Key`：对应属性 ID。更多信息，参见 教程场景说明。 • `Value`：对应属性的值。在客户端以字符串形式存储；在实际分析中，服务端支持自动转化为字符型、整型、浮点型。

相关链接

- 接入方式介绍
- 发布说明
- 问题排查

12.1.4. 创建属性

进行事件分析前，需要先配置事件要关联的属性。

操作步骤

进入控制台 **移动分析** > **自定义分析** > **自定义配置** > **属性** 标签页，单击 **新建** 添加属性。

属性字段的说明如下：

- **属性 ID**：属性唯一标识。属性是应用的全局属性，因此，同一个 mPaaS 应用中属性 ID 必须唯一。
- **属性名称**：属性 ID 对应的中文名称。您可以自定义属性名称。
- **数据类型**：属性值对应的数据类型。
- **单位**：当前属性的单位。
- **显示状态**：选择是否显示当前属性。

本教程中，需要创建以下三个属性：

属性 ID	属性名称	数据类型	单位	是否可枚举	显示状态
pay_time	支付时间	字符型	-	否	开
user_id	用户 ID	字符型	-	否	开
payment_method	支付方式	字符型	-	是	开

后续操作

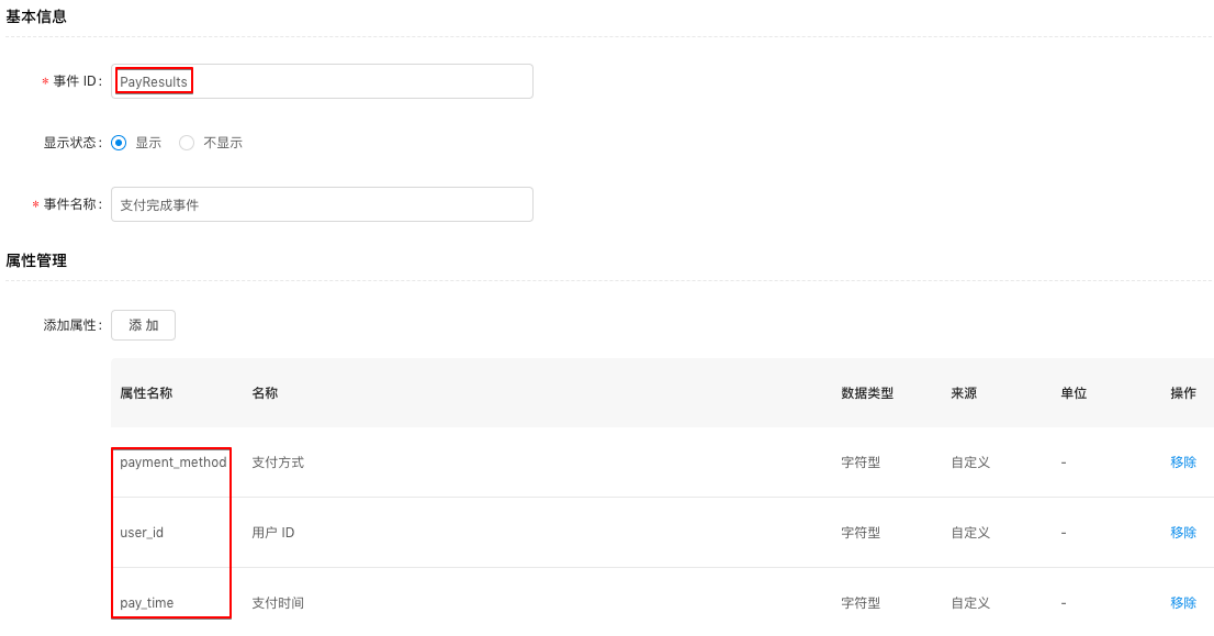
创建事件

12.1.5. 创建事件

创建要分析的用户行为事件。

操作步骤

在控制台创建事件属性后，从左侧导航栏进入 **移动分析** > **自定义分析** > **自定义配置** > **事件** 标签页，创建“支付完成”事件。



- **事件 ID**：事件的唯一标识，例如 `PayResults`。更多信息，请参见 教程场景说明。
- **显示状态**：是否在事件分析列表中显示该事件，这里可以设置为显示。
- **事件名称**：根据实际情况填写易于识别的事件名称，如”支付完成事件“。
- **添加属性**：进入属性添加页面后，您可以勾选事先创建的事件属性。所有的平台预置属性都会自动添加，您无需额外操作。

后续操作

添加自定义分析

12.1.6. 查看事件 PV 和 UV

您可以在 **移动分析** > **自定义分析** > **事件分析** 页面查看 **事件发生的次数**（PV） 和 **去重后的用户数**（UV）。在此例中，PV 代表支付完成的次数，UV 代表支付完成的用户数。

如果您未看到数据或数据与预想不符，请参考 常见问题 进行问题排查。

12.1.7. 添加自定义分析

除事件 PV 和 UV 数据外，您还可以使用属性信息对事件作更进一步的分析。

例如，教程中记录了 **支付方式** 属性，该属性是枚举型的。因此，可以具体分析每种支付方式对应的事件 PV，从而分析用户的支付方式偏好。

操作步骤

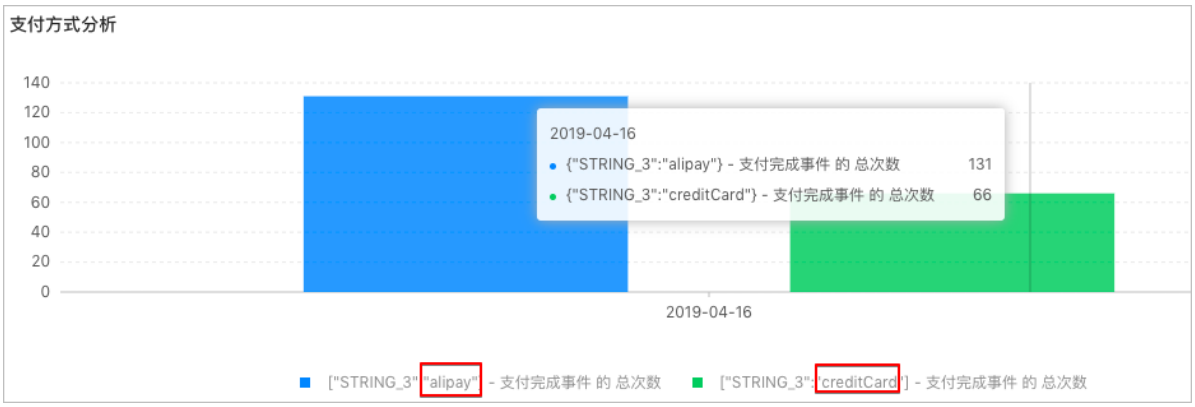
1. 在 事件分析 页面中，单击事件名称，进入对应事件的自定义分析页面。

   您将看到平台默认创建 DashBoard 报表，报表以折线图的形式展示事件每日 PV 和 UV。

2. 在自定义大盘页面，单击 **添加自定义分析** 按钮，进入自定义分析配置页面，完成分析配置，包括分析名称、显示指标、聚合规则、展示方式等。

3. 单击 **提交** 按钮，等待片刻，将看到如下报表：

   由报表可知，在选定的时间段内，用户使用 `alipay`（支付宝）支付的次数为 131、使用 `creditCard`（信用卡）支付的次数为 66，用户更喜欢支付宝。



相关链接

添加事件分析

12.1.8. 添加大盘

当有多个事件时，您可以在控制台创建大盘，从而快速查看关键业务指标。

操作步骤如下：

1. 登录 **mPaaS 控制台**，选择应用，然后从左侧导航栏进入 **移动分析** > **自定义分析** > **自定义大盘** 页面。
2. 单击 **添加大盘** 按钮，在窗口中填写大盘名称，选择大盘模板，并确认添加。
3. 单击刚刚添加的大盘。
   - 创建大盘时默认添加一个名称为 **未命名** 的分类。将鼠标移动到分类名称上，您可以看到修改分类名称的图标。
   - 您可以单击右侧 **添加分类** 按钮，新增一个分类。
4. 单击 **添加自定义分析** 按钮，在大盘中添加关键指标报表。
   - 添加已有分析模块：若您已创建了自定义分析模块，此处可以直接使用。已有的分析模块可以在 **移动分析** > **自定义分析** > **自定义配置** > **卡片管理** 中查看。
   - 新建自定义分析：您也可以新建分析模块。具体步骤，参见 添加自定义分析 和 添加漏斗分析。
5. 添加好自定义分析后，单击 **提交**。至此，您已创建好了大盘。

12.1.9. 相关步骤

通过学习本教程，您已完成了对单个事件的分析。

在实际业务中，一个完整的流程可能包含多个事件，如购物流程大致包含搜索产品、浏览产品、加入购物车、下单、付款等事件。此时，您可以对整个业务过程中的事件进行漏斗分析，分析用户从第一个事件到后续事件的转化率。相关操作，参见 漏斗分析。

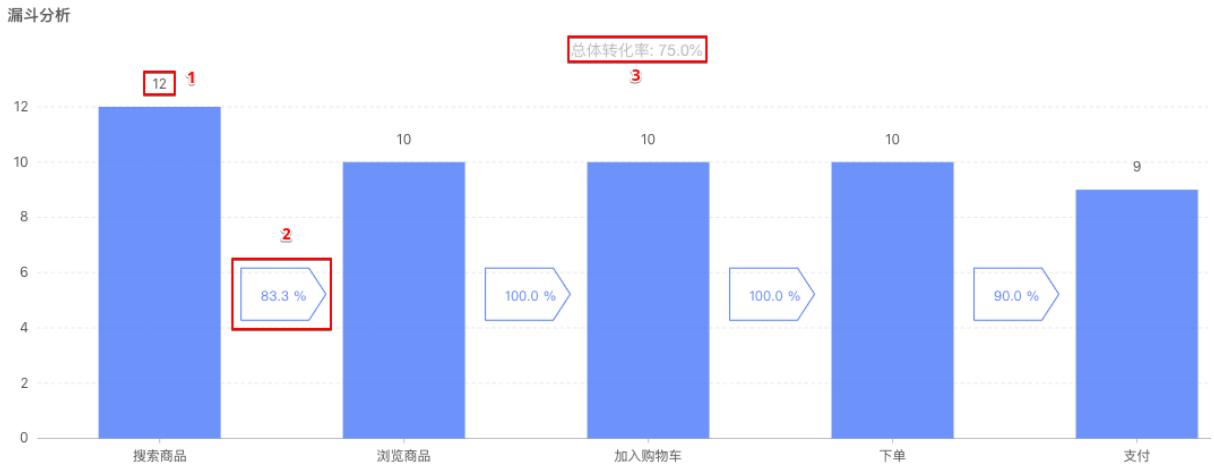
12.2. 漏斗分析

在实际业务中，一个完整的流程通常包含多个步骤，每个步骤可以记录一个事件。漏斗分析可以帮助您分析从第一个步骤到后续步骤的转化率。针对转化率低的步骤，您可以优化该步骤的业务逻辑，也可以使用 消息推送、智能投放 等产品做精细化运营。

为更好地理解漏斗分析，您需要了解以下概念：

- 转化率：在特定条件下，事件 A 到事件 B 的转化率 = 同时触发事件 A 和 B 的用户数/触发事件 A 的用户数。
  - 用户数统计支持用户 ID、设备 ID 两个维度，在创建漏斗时可以任选一种。
  - 特定条件包括时间段、自定义的过滤条件。
- 整体转化率：完整流程中第一个事件到最后一个事件的转化率。

下图展示的是购物流程的漏斗分析：



购物流程包含 5 个步骤（事件），图中数字标号对应的说明如下：

1. 12：对应事件（即搜索商品）的用户数。
2. 83.3%：相邻事件（即搜索商品到浏览商品）的转化率，即 10/12 = 83.3%。
3. 75.0%：整体转化率，即 9/12 = 75.0%。

前置条件

漏斗用于分析事件之间的转化率，因此，进行漏斗分析之前需要配置好自定义事件。更多信息，参见 自定义事件分析教程。

操作步骤

步骤 1：创建漏斗

创建漏斗的步骤如下：

1. 登录 mPaaS 控制台，从左侧导航栏进入 **移动分析** > **自定义分析** > **漏斗分析** 页面，单击 **创建漏斗** 按钮。
2. 在创建漏斗页面，完成相关配置。其中：
   - **时间窗口**：指完成漏斗中相邻两个步骤（事件）允许的最长时间间隔。
   - **计算维度**：分析维度，选择 **用户 ID** 或 **设备 ID**。

3. 根据实际业务流程，添加流程中的步骤（事件）。
4. 单击 **提交** 完成漏斗创建。

步骤 2：查看漏斗

在漏斗分析页面，单击漏斗名称可查看漏斗。

漏斗分析是 T+1 的，每日凌晨计算，您可以查询昨天及以前的数据。

12.3. 应用启动速度分析

mPaaS 的优势之一是能帮助您打造超级稳定、高性能的 App。本教程将引导您进行应用启动速度分析。

Android 开发

1. 接入移动分析组件

移动分析支持原生 AAR 接入、mPaaS Inside 接入和组件化接入（Portal & Bundle）三种接入方式。但是，目前只有基于 mPaaS 框架的应用才能使用 SDK 中封装的接口来统计应用启动速度。

参见 Android 接入文档 完成组件接入，包括完成通用步骤、添加 SDK、配置工程等。其中，在添加 SDK 时推荐您选择最新版本的 SDK。

2. 记录启动速度日志

```
import com.mpaas.mas.adapter.api.MPLogger;

MPLogger.reportLaunchTime(Context context);
```

更多信息，请参见 性能日志 > 启动速度。

iOS 开发

1. 接入移动分析组件

参见 iOS 接入文档 接入移动分析组件。

2. 记录启动速度日志

具体的接口信息，请参见 性能日志。

查看分析报表

在控制台 **移动分析** > **基础分析** > **行为分析** 页面，选择日期后，您可以看到应用平均启动速度报表。

- 首次启动：首次安装后，第一次启动 App。
- 非首次启动：非首次安装 App 后，启动 App。

启动速度统计的都是冷启动，且启动速度分析是 **准实时** 的。即客户端正确接入、日志上报到日志服务器之后，您就可以看到如上报表。

12.4. 闪退分析

闪退一般是指 App 非正常退出。闪退分析提供闪退统计功能，支持统计闪退数、闪退率和受影响设备数，支持按照问题原因聚合闪退报告，并统计某一类闪退的个数、影响的用户数、主要机型等信息。

完成客户端开发

- **Android**

  具体步骤如下：

  i. **接入移动分析组件。更多信息，参见 接入 Android。**

  ii. **开启闪退监控。更多信息，参见 Android 闪退日志。**

- **iOS**

  具体步骤如下：

  i. 接入移动分析组件。更多信息，参见 接入 iOS。

  ii. 开启闪退监控。更多信息，参见 iOS 闪退日志。

查看闪退报告

登录 mPaaS 控制台，进入 **移动分析** > **性能分析** > **闪退报告** 页面，查看闪退报告。

在 **闪退分类** 列表中，点击 **详情** 列的内容，即可查看某一类闪退的详情。

- 设备详情：
  - **设备 ID**：格式为 IMSI|IMEI。
  - **平台**：由 3 个字段组合而成，即 `appId-平台-workspaceId`。
- 日志详情：日志以逗号分隔，其中包含完整的错误堆栈。每个字段的具体含义，请参见 闪退埋点。

闪退分析是 **准实时** 的。即在应用发生闪退且日志上报之后，您就可以看到如上报告。

12.5. 卡死分析

卡死分析提供卡死统计功能，支持统计卡死次数、卡死率、受影响设备数等。

卡死包括以下情况：

卡死类型	Android	iOS
启动卡死	App 启动后 30 秒内未能离开欢迎页和进入首页。	App 启动后 15s（低端机 30s）未接收到 UIApplicationDidFinishLaunchingNotification 通知。
ANR 卡死	即系统 ANR 卡死，定义详见 Android 官网 ANR。	App 运行时主线程在 5 秒内未执行完一个方法。

完成客户端开发

- **Android**

  具体步骤如下：

  i. **接入移动分析组件。更多信息，参见 接入 Android。**

  ii. **开启卡死监控。更多信息，参见 Android 性能日志。**

- **iOS**

  具体步骤如下：

  i. 接入移动分析组件。更多信息，参见 接入 iOS。

  ii. 开启卡死监控。更多信息，参见 iOS 性能日志。

查看卡死报告

登录 mPaaS 控制台，进入 **移动分析** > **性能分析** > **卡死报告** 页面，查看设备卡死率、卡死次数、影响设备数、卡死日志等信息。

在 **卡死分类** 列表中，点击 **详情** 列的内容，即可查看某一类卡死的详情。

- 设备详情：
  - 设备 ID：格式为 IMSI|IMEI。

- 平台：由 3 个字段组合而成，即 `appId-平台-workspaceId` 。

- 日志详情：日志以逗号分隔，其中包含完整的错误堆栈。每个字段的具体含义，请参见 日志模型 > 性能埋点 > 卡死埋点。

卡死分析是 **准实时** 的。即在应用发生卡死且日志上报之后，您就可以看到如上报告。

12.6. 卡顿分析

卡顿是指主线程超过一定时间（Android 为 2.25 秒；iOS 为 2 秒）未执行完一个方法。

完成客户端开发

- **Android**

  具体步骤如下：

  i. **接入移动分析组件。更多信息，参见 接入 Android。**

  ii. **开启卡顿监控。更多信息，参见 Android 性能日志。**

- **iOS**

  具体步骤如下：

  i. 接入移动分析组件。更多信息，参见 接入 iOS。

  ii. 开启卡顿监控。更多信息，参见 iOS 性能日志。

查看卡顿报告

登录 mPaaS 控制台，进入 **移动分析 > 性能分析 > 卡顿报告** 页面，查看设备卡顿次数、卡顿率、影响设备数、卡顿日志等信息。

在 **卡顿分类** 列表中，点击 **详情** 列的内容，即可查看某一类卡顿的详情。

- 设备详情：
  - 设备 ID：格式为 IMSI|IMEI。
  - 平台：由三个字段组合而成，即 `appId_平台-workspaceId` 。

- 日志详情：日志以逗号分隔，其中包含完整的错误堆栈。每个字段的具体含义，请参见 日志模型 > 性能埋点 > 卡顿埋点。

卡顿分析是 **准实时** 的。即在符合抽样规则的设备上，应用发生卡顿且日志上报之后，您就可以看到如上报告。有关抽样规则的说明以及抽样比率的设置方法，请在客户端接入时查看相关的文档。

12.7. 报活分析

移动分析支持在应用启动时记录报活日志，以报告当前用户处于活跃状态。

报活分为：

- 设备报活：以设备 ID 唯一标识当前用户。
- 用户报活：以用户 ID 唯一标识当前用户。

完成客户端开发

- **Android**

  具体步骤如下：

  i. 接入移动分析组件。更多信息，参见 接入 Android。

  ii. 记录报活日志。根据 SDK 版本，参见 添加报活日志。

- **iOS**

具体步骤如下：

i. **接入移动分析组件。更多信息，参见 接入 iOS。**

ii. 记录报活日志。根据 SDK 版本，参见 添加报活日志。

查看报活数据

登录 mPaaS 控制台，进入 **移动分析** > **数据概览** 页面，可以查看实时或历史的 **应用装机量**（对应活跃用户数）、**注册用户数**（对应活跃账号数）等信息。

进入 **移动分析** > **基础分析** > **留存分析** 页面，可以查看用户留存情况。

进入 **移动分析** > **基础分析** > **行为分析** 页面，可以查看地域用户数占比。

相关链接

- 指标计算规则
- 用户 ID

13.常见问题

下面是使用移动分析时会遇到的一些常见问题及排查方法。

如何检查客户端是否已正确接入

您可以查看本地日志（Android/iOS）或 在控制台查询日志，从而检查客户端是否已正确接入。

正确接入客户端之后，控制台上仍然看不到数据

只有在满足一定条件（如本地日志满一定条数或应用压后台超过一定时间）后，客户端日志才会自动上报到日志服务器。在测试时，为了尽快看到数据，您可以在客户端通过手动上报的方式，强制日志立刻上报。

更多信息，请参见 Android 手动上报日志 或 iOS 手动上报日志。

在控制台创建事件后，看不到事件 PV 和 UV 数据

要看到事件 PV 和 UV 数据，您需要确保：

- 已正确接入客户端。更多信息，请参考 如何检查客户端是否已正确接入。
- 日志已上报到日志服务器。更多信息，请参考问题 正确接入客户端之后，控制台上仍然看不到数据。
- 在控制台创建事件之后，用户触发了该事件。

在控制台创建事件后，事件 UV 始终显示为 0

您需要确保在客户端设置了用户 ID。设置方法，请参见 用户 ID。

自定义大盘中没有数据展示

参照以下步骤进行排查：

1. 检查是否开启日志上报开关。进入控制台后，从左侧导航栏点击 **移动分析 > 日志管理 > 配置上传开关 > 埋点配置**，进入日志开关列表页面，查看相应的埋点是否开启上报开关，确保已开启日志上报开关。更多信息，参见 配置日志上传开关。
2. 检查大盘对应的日志是否上报到服务端。通过控制台的 **日志管理 > 日志回放** 功能，查询历史日志。
   - 如果能查询到对应的日志，则表示日志已上报。更多信息，参见 查询历史日志。
   - 如果未查询到对应的日志，则需要排查 App 是否触发日志。操作步骤如下：
     a. 触发日志前，需先断开手机网络，然后触发日志。
     b. 在 App 压后台后，在本地日志目录中查看是否存在要查询的日志。
        - iOS 客户端：日志保存在沙盒目录 `Library > atrack > logs` 下。
        - Android 客户端：日志保存在 `/data/data/[PackageName]/files/mdap` 或 `/sdcard/Android/data/[PackageName]/files/mdap` 下。日志保存路径因 `assets/channel.config` 中的 `release_type` 字段值而异，具体参见 查看本地日志。
     c. 客户端触发生成日志之后，再次进行日志回放，检查日志是否上报到服务端。

        客户端生成日志后，本地缓存的日志要满一定条数才会自动触发日志上报。不同日志类型，默认缓存的日志条数可能不同。您可以先通过 mPaaS 控制台的 **移动分析 > 日志管理 > 配置上传开关 > 埋点配置** 页面，将 **日志上报条数**（本地文件中当前类型的日志到达条数触发日志上报） 修改为 1，以便触发日志上报，调试完毕后，再改回原本的条数。更多关于日志上报触发的内容，参见日志上报说明文档（Android 日志上报/iOS 日志上报）。

3. 如果通过日志回放发现日志仍然没有上报到服务端，则根据日志类型，参考 Android 埋点接入 或 iOS 埋点接入 说明文档，检查是否埋点接入过程中操作有误。根据文档正确接入埋点后，再执行第 2 步，确保日志正常上报。
4. 如果日志已上报至服务端，但大盘没有数据，则需要检查日志的数据格式是否正确。

   对比 **日志回放** 中的原始日志和 埋点日志模型，确认日志格式是否正确。如日志格式有误，则参考对应埋点类型的日志模型，修改日志格式。
5. 如果按照上述步骤排查后，确认日志已上报至服务端，且日志数据格式正确，但大盘中仍无数据展示，请搜索群号 41708565 加入钉钉群进行咨询。

移动分析控制台上 iOS 客户端的闪退日志没有反解

iOS 客户端的闪退日志需要配合打包生成的 dSYM 符号表才能进行反解。

iOS 的闪退统计功能支持闪退日志符号化。对于需要此功能的 App，需要在 mPaaS 控制台的 **移动分析** > **性能分析** > **iOS 符号表管理** 页面中上传 dSYM 符号表文件。参见 iOS 符号表管理 了解更多内容。

14.参考

14.1. 用户 ID

在很多场景中，用户 ID 都是很重要的信息。

- 移动分析平台预置了属性名为 `userId` 的字符型属性。您可以到控制台 **移动分析 > 自定义分析 > 自定义配置 > 属性** 页面中查看属性详情。
- 不同类型的埋点日志都包含用户 ID 字段。详情请参见 日志模型 > 自定义事件。
- 事件分析中，和用户量（UV）相关的分析，都依赖用户 ID。
- 使用 mPaaS 实时发布平台，对特定用户白名单进行灰度发布时，要求客户端设置了用户 ID。

用户 ID 与设备 ID

用户使用您的 App 时可能处于未登录状态，这意味着用户 ID 可能为空，但设备 ID 通常可以获取到。此时，可以使用设备 ID 替代用户 ID，进而分析用户行为。

例如，在使用 漏斗分析功能 时，您可以指定计算维度为用户 ID 或设备 ID。计算维度为用户 ID 时，用户数是去重后的用户 ID 数；反之则为去重后的设备 ID 数。

设置用户 ID

为了使用和 **用户 ID** 相关的分析功能，您需要调用 SDK 接口设置用户 ID。

> **说明**
>
> - 避免在设置用户 ID 时使用特殊符号，推荐使用数字、字母。
> - 勿将 UTDID 读取的设备 ID 作为 `userId`。

Android

- 情形一：调用用户报活接口 `MPLogger.reportUserLogin("userId");`。假如需要统计 App 注册用户量，那么您需要调用用户报活接口 `MPLogger.reportUserLogin("userId");`；该接口会自动将入参设置为全局的用户 ID，您无需额外操作。
- 其他情形：调用 `MPLogger.setUserId("userId");` 设置用户 ID。

更多信息，请参考 报活日志。

iOS

参考 配置工程，在 `MPaaSInterface` 的 `Category` 中配置用户 ID：

```
@implementation MPaaSInterface (Demo)
- (NSString *)userId
{
    return @"the-user-id";
}
@end
```