# 高级功能

## 面包屑

```
1、相关接口
/*
 面包屑功能：是指用户在程序中通过该接口添加的一些信息,当程序发生崩溃时，将会把这些添加的信息，按顺序收集起来，和崩溃信息都发送给服务器
 @breadcrumb:自定义信息
*/
+(void)leaveBreadcrumb:(NSString *)breadcrumb;

2、代码示例
-(BOOL)application:(UIApplication *)application didFinishLaunchingWithOption:(NSDictionary *)launchOptions
{
    [NBSAppAgent leaveBreadcrumb:@"didFinishLaunchingWithOptions"]; //设置面包屑
    return YES;
}
```

## 自定义Event

```
1、相关接口
/*
 自定义事件：最多包含32个字符，支持中文、英文、数字、下划线，但不能包含空格或其他的转义字符
 */
+ (void)trackEvent:(NSString *)eventID;

2、代码示例
-(void)purchaseFailedCallback
{
    [NBSAppAgent trackEvent:@"购买失败"];
}
```

## 自定义Trace

```
1、相关接口
//Name 为当前方法所在方法名或自定义名称，支持中文、英文、数字、下划线，但不能包含空格或其他的转义字符
beginTracer(@“String Name")
endTracer(@“String   Name")
/用户可以在SDK初始化后的任意方法前后添加自定义Trace

2、代码示例
- (void)doSomething
{
  beginTracer(@"String Name")
 //write you code here

 endTracer(@"String  Name")
}
```

## 自定义ID

```
1、相关接口
/*
 设置用户标识符,不能超过64个字符
 @userId:能唯一标识一个用户的信息
 warning:为保证数据的准确性，该接口最好只设置一次
 */
+(void)setUserIdentifier:(NSString *)userId;

2、代码示例
-(void)viewDidLoad
{
    [super viewDidLoad];
    [NBSAppAgent setUserIdentifier:@"UserIdentifier"];//    设置自定义Id,可为邮箱等标识用户身份的信息，如xxx@tingyun.com
}
```

## 自定义操作ID

```
1、相关接口
/*
 设置自定义的操作Id，在需要改变对应操作名称的实现中调用
 @target:方法的receiver，一般为self
 @actionId:自定义名称，最多包含32个字符，支持英文、数字、下划线，不能使用“#”
 @warning:只能在主线程中调用
 */
+ (void)setTarget:(id)target actionId:(NSString *)actionId;

2、代码示例
- (void)tap
 {
    [NBSAppAgent setTarget:self actionId:@"xxx"];//会把操作名称tap改为xxx，应该首先调用
    ...
 }
```

## 设置采集App版本号

```
1、相关接口
/*
 同时指定启动概率、是否使用位置服务、渠道ID
 useBuildVersion：YES优先使用CFBundleVersion版本号，NO使用CFBundleShortVersionString，默认为NO。
 */
+(void)startWithAppID:(NSString*)appId location:(BOOL)locationAllowed rateOfLaunch:(double) rate channelId:(NSString *)channelId useBuildVersion:(BOOL)useBuildVersion;

2、代码示例
int main(int argc, char * argv[]) {
 @autoreleasepool {
        [NBSAppAgent startWithAppID:@"appkey" location:YES rateOfLaunch:1.0f channelId:@"App Store" useBuildVersion:YES];
    ...
    }
}
```

## 设置首次启动功能模块

**从SDK 2.12.0开始，首次启动应用时，SDK默认只开启崩溃数据采集(之后的启动按连接服务器下发的配置开启相应功能)。用户可以通过接口自定义首次启动SDK的功能开关，仅首次启动有效**

1、相关接口

```
typedef NS_ENUM(NSInteger, NBS_OPTION)
{
    NBS_OPTION_NET           = 1<<0,
    NBS_OPTION_UI            = 1<<1,
    NBS_OPTION_CRASH         = 1<<2,
    NBS_OPTION_HYBRID        = 1<<3,
    NBS_OPTION_SOCKET        = 1<<4,
    NBS_OPTION_STRIDE        = 1<<5,
    NBS_OPTION_ANR           = 1<<6,
    NBS_OPTION_BEHAVIOUR     = 1<<7,
    NBS_OPTION_CND           = 1<<8,

    NBSOption_Net       = NBS_OPTION_NET,
    NBSOption_UI        = NBS_OPTION_UI,
    NBSOption_Crash     = NBS_OPTION_CRASH,
    NBSOption_hybrid    = NBS_OPTION_HYBRID,
    NBSOption_Socket    = NBS_OPTION_SOCKET,
    NBSOption_StrideApp = NBS_OPTION_STRIDE,
    NBSOption_ANR       = NBS_OPTION_ANR,
    NBSOption_Behaviour = NBS_OPTION_BEHAVIOUR,
    NBSOption_CDNHeader = NBS_OPTION_CND
};
/*
 设置启动选项，SDK有几个功能，借此可以关闭某个。此函数应该在其他函数之前调用。option的值应该是NBSOption_Net、NBSOption_UI、NBSOption_Crash、NBSOption_hybrid、NBSOption_Socket的组合
 */
+(void)setStartOption:(int)option;
```

2、代码示例

```
int main(int argc, char * argv[]) {
 @autoreleasepool {
    [NBSAppAgent setStartOption:NBSOption_Net|NBSOption_Crash]; //首次启动只开启网络和crash采集功能。
    [NBSAppAgent startWithAppID:@"appkey" location:YES];
    ...
    }
}
```

## 关闭更新提示

```
1、相关接口
/*
 关闭更新提示log。
 SDKVersion为最新的SDK版本
 */
+ (void)closeLogForUpdateHint:(NSString *)SDKVersion;

2、代码示例
int main(int argc, char * argv[]) {
 @autoreleasepool {
    [NBSAppAgent closeLogForUpdateHint:@"sdkversion"]; //首次启动只开启网络和crash采集功能。
    [NBSAppAgent startWithAppID:@"appkey" location:YES];
    ...
    }
}
```

## 设置自定义用户信息

```
1、相关接口
/*
 setCustomerData:forKey:的目的是这样的：通过这个方法设置的信息，会在crash时作为环境信息提交，帮助用户分析问题。此函数可以在任何地方多次调用
 */
+(void)setCustomerData:(NSString*)data forKey:(NSString*)key;

2、代码示例
- (void)doSomething
{
    [NBSAppAgent setCustomerData:@"value" forKey:@"key"];
    ...
}
```