Using Analytics API in client applications

Overview

To populate your custom charts you can use the Anlaytics API to send customized data. Custom data is any key/value pair that you would like to collect that is not an out of the box feature, like button presses.

When collecting custom analytics data, the app writes to the devices file system and is not sent until the `send` API is called. After the `send` API is called the app deletes the data from the file system and begins collecting again.

Before you being in the native app import the Analytics SDK.

Android

```
import com.worklight.common.WLAnalytics;
```

iOS

```
import "WLAnalytics.h"
```

Jump to:

Initializing Analytics

Before you can start collecting the out of the box data that Operational Analytics provides you first need to initialize Analytics.

In cordova this is done through your native application. So the code snippets below are going to show you how to initialize analytics in iOS and Android.

Android:

```
WLAnalytics.init(this.getApplication());
```

iOS:

```

```

After initializing analytics the app will start collecting user information, like app sessions.

Enabling/Disabling Client Event Types

The developer now has the option to choose to record certian event types if the wishes through an api call.

Android:

```
//DeviceEvent.LIFECYCLE records app sessions
WLAnalytics.addDeviceEventListener(DeviceEvent.LIFECYCLE);
WLAnalytics.removeDeviceEventListener(DeviceEvent.LIFECYCLE);
//DeviceEvent.Network records client information about adapters like 'Average Procedure Response Size'
WLAnalytics.addDeviceEventListener(DeviceEvent.NETWORK);
WLAnalytics.removeDeviceEventListener(DeviceEvent.NETWORK);
```

iOS

```
```

Custom Analytics

JavaScript API

JavaScript API is used in Cordova applications.

Creating custom events in Cordova is simply just calling:

```
WL.Analytics.log({"key" : 'value'});
WL.Analytics.send();
```

Android

After setting the first two configurations you can start to log data like in the example below.

```
JSONObject json = new JSONObject();
try {
    json.put("key", "value");
} catch (JSONException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}

WLAnalytics.log("Message", json);
```

Be sure to send your data with:

```
WLAnalytics.send();
```

Objective-C API

Objective-C API is used in iOS applications.

To collect custom analytics in iOS you need to import WLAnalytics.

```
#import "WLAnalytics.h";
```

After importing WLAnalytics you can now use the API to collect custom data like below:

```
NSDictionary *inventory = @{
    @"property" : @"value",
};

[[WLAnalytics sharedInstance] log:@"Custom event" withMetadata:inventory];
[[WLAnalytics sharedInstance] send];
```