Getting Started Guide for Marmalade v.1.0 June - 2013

Introduction

The following guide will instruct you on how to integrate the PlayHaven's Plug-In for Marmalade.

About PlayHaven

PlayHaven helps you acquire, engage, and monetize your players. Built specifically for game developers, we offer an easy to integrate SDK for both iOS and Android with plugins available for Unity, Adobe Air and now Marmalade. And we offer powerful user segmentation and targeting tools that you can use to take action in real time based on relevant information on our web dashboard.

PlayHaven's network spans thousands of games, 523 million unique users, 130 active monthly users, and 2.5 billion monthly game sessions. Some of the most successful mobile studios rely on PlayHaven, including Cartoon Network, Digital Chocolate, Glu Mobile, Game Circus, Namco Bandai, Natural Motion, Nickelodeon, Playtika, and Sega.

About Marmalade

Marmalade is a cross-platform tool that helps developers spread apps to any device by empowering them to create apps in Native (C++), Lua or hybrid (HTML5) to access platforms features such as location, notifications, social media links, camera functionality and more. Utilizing the provided tools, developers can deploy to iOS, Android, Windows Phone 8 and BlackBerry 10, Windows and Mac, as well as other platforms including selected Smart TVs - with just one click. Chart-smashing mobile games rely on Marmalade, from casuals such as Cut the Rope, Plants vs Zombies and Draw Something to high-performance titles like Call of Duty, Need for Speed and Pro Evolution Soccer.

Notes & Pre-Requisites

- This extension has been built and tested with Marmalade 6.2.
- The iOS version will not build on earlier versions of Marmalade due to the dependency on the iOS 6 AdSupport framework included for the first time in Marmalade 6.2.
- iOS is built against Playhaven SDK v1.13.1
- Android is built against Playhaven SDK version 1.12.5
- There is no keychain on Windows, so you need to export your private and public signing keys to your PC and then put them in a specific folder inside Marmalade – for more information, see this guide: (http://docs.madewithmarmalade.com/native/platformguides/iosguide/iossigning/iossigningassetssetup.html). If you don't do this, your IPA will not be signed correctly and will not deploy. You should export your provisioning profile to PC and include it in the Deploy stage.

iOS Integration

Integrating the s3ePlayhaven extension into your own project

- Create an account on http://dashboard.playhaven.com/ if you haven't already.
- Add a new iOS game with the name of your project.
- In your mkb add the extension to the subprojects section.
- You can either copy the entire extension folder to your Marmalade extension folder, in which case add the below:

```
subprojects
{
	s3ePlayhaven
}
```

Or you can include the extension mkf file using a relative path

```
subprojects
{
	../s3ePlayhaven
}
```

- Add your test device ID to your Playhaven account to enable test content to be sent.
- Copy the generated App token and secret from the dashboard into your project.
- Initialize Playhaven using `s3ePlayhavenInitWithKeys`.
- Send `s3ePHSendAppOpen` whenever your app starts or comes into the foreground.
- Trigger placements in your code using s3ePHSendContentRequest.
- Preload placement content data in the background for a specific placement using `s3ePHPreloadContentRequest`.
- Set optional callbacks to various content events using `s3ePHRegisterCallback`.
- See /h/s3ePlayhaven.h for detailed documentation on each function.

Notes:

- As with any third party Marmalade extensions, s3ePlayhaven can only be tested on a device. It is only linked at the deploy stage. It will be unavailable in the emulator.
- The extension also requires the UIKit, CoreGraphics, SystemConfiguration, CFNetwork, AdSupport and StoreKit iOS frameworks, these are added to the load linker for you by the extension .mkf file.
- We define `PH_NAMESPACE_LIBS`, which causes the third party libraries such as OpenUDID into their own namespace to prevent collisions if other copies are included by other extensions. They are pretty lightweight and the convenience was seen to outweigh the downside of potential duplicated functionality.

Example Application with the s3ePlayhaven extension

As part of the package, we have provided you with a simple Marmalade Application that integrates the s3ePlayhaven extension.

MAC (OSX) to IOS

- As with any third party Marmalade extensions, s3ePlayhaven can only be tested on a device. It is only linked at the deploy stage. It will be unavailable in the emulator.
- Select/Create test game on Dashboard
- Open the Marmalade folder, Navigate to Example App folder
- Open s3ePlayhavenExample.mbk file
- Once finished xCode such launch
- Set deploy to: RELEASE/DEBUG ARM GCC > Mac 32bit (In xCode)
- Change placements names and placements being called to match those with selected game and make changes to .cpp file (In xCode)
- Edit Secret/Token to match select game in the .cpp file
- Open Deploy Tool from inside the Marmalade directory
- Navigate to the xCode project that was just created and select it
- Select GCC ARM check box
- Select “k” leave all other settings default
- Set package to install and run
- Click on Deploy all.
- iTunes should automatically launch
- Select your device and click on Apps Tab (in iTunes)
- Install Marmalade

Notes:

- You will need to add the appropriate options such as a provisioning profile during Deploy.
- If you receive error code 7, xCode is not in right directory use this command in terminal

```
$ sudo xcode-select -switch /Applications/Xcode.app/Contents/Developer
```

Windows to IOS

- Select/Create test game on Dashboard
- Install Visual Studio Express 2010 and set Visual Studio as the default build environment
- Install Marmalade SDK Plus Edition 30 day trial
- Double click on the s3ePlayhavenExample.mkb file in the root Marmalade folder. Visual Studio should be launched with the Marmalade project loaded.
- Open the s3ePlayhavenExample.cpp file
- Edit the token / secret and placements to match selected game, then save the changes (Build in Visual Studio)
- Open Marmalade "Deploy Tool", navigate to Marmalade Directory and select Config.py file (Check Example Application Folder in Marmalade Directory)
- Select: Arm GCC, IOS, and leave rest as Default.
- Set to package to install

- Click on Deploy all
- iTunes should automatically launch
- Select your device and click on Apps Tab (In iTunes)
- Install Marmalade

Notes:

- You will need to add the appropriate options such as a provisioning profile during Deploy.

Android Integration

Integrating the s3ePlayhaven extension into your own project

- Create an account on http://dashboard.playhaven.com/ if you haven't already.
- Add a new Android game with the name of your project.
- In your mkb, add the extension to the subprojects section.
- You can either copy the entire extension folder to your Marmalade extension folder, in which case add the below:

```
subprojects
{
      s3ePlayhaven
}
```

Or you can include the extension mkf file using a relative path

```
subprojects
{
      ../s3ePlayhaven
}
```

- Add your test Mac address to your Playhaven account to enable test content to be sent.
- Copy the generated App token and secret from the dashboard into your project.
- Initialize Playhaven using `s3ePlayhavenInitWithKeys`
- Send `s3ePHSendAppOpen` whenever your app starts or comes into the foreground.
- Trigger placements in your code using `s3ePHSendContentRequest`.
- Preload placement content data in the background for a specific placement using `s3ePHPreloadContentRequest`.
- Set optional callbacks to various content events using `s3ePHRegisterCallback`.
- See /h/s3ePlayhaven.h for detailed documentation on each function.

Notes:

- As with any third party Marmalade extensions, s3ePlayhaven can only be tested on a device. It is only linked at the deploy stage. It will be unavailable in the emulator.
- The extension also requires the Playhaven SDK jar (PlayhavenSDK_Android/playhaven-x.xx.x.jar), which is added to the load linker for you by the extension mkf file.

Example Application with the s3ePlayhaven extension

The Example Application implements a simple Marmalade application that integrates the s3ePlayhaven extension.

MAC (OSX) to Android

- Select/Create test game on Dashboard
- Open the Marmalade folder, Navigate to Example App folder
- OPEN s3ePlayhavenExample.MBK FILE
- Once finished, xCode should launch

- Set deploy to: RELEASE/DEBUG ARM GCC > Mac 32bit (In xCode)
- Change placements names and placements being called to match those with selected game and make change to .cpp file (In xCode)
- Edit (token / secret) to match selected game in the .cpp file
- Open Deploy Tool from inside the marmalade directory
- Navigate to the xCode project that was just created and select it
- Select GCC ARM check box
- Select Android leave all other settings default
- Set to package, install and run
- Connect Device to computer
- Click on Deploy all
- Marmalade should install

Notes:

- If during Deploy you have Android_Root SDK path issues, you can click on “Explore” as it will take you directly to where the .APK is created and you can manually install the .apk.
- You will need to add the appropriate options during Deploy.

Windows to Android

- Select/Create test game on Dashboard
- Install Visual Studio Express 2010 and set Visual Studio as the default build environment
- Install Marmalade SDK Plus Edition 30 day trial
- Double click on the s3ePlayhavenExample.mkb file in the root Marmalade folder. Visual Studio should be launched with the Marmalade project loaded.
- Open the s3ePlayhavenExample.cpp file
- Edit the (token / secret) and placements to match the selected game and save the changes (Build in Visual Studio)
- Open Marmalade "Deploy Tool", navigate to the Marmalade Directory and select Config.py file (Check Example Application Folder in Marmalade Directory)
- Select GCC ARM check box
- Select Android leave all other settings default
- Set to package, install and run
- Connect Device to computer
- Click on Deploy all

Notes:

- If during Deploy you have Android_Root SDK path issues, you can click on “Explore” as it will take you directly to where the .APK is created and you can manually install the .apk.
- Please be sure to add the appropriate options during Deploy.