Overview

Welcome to PlayHaven! We are excited to have you join our network and look forward to helping you increase downloads for your games. This document outlines the PlayHaven integration, testing, and launch process as well as links to resources and guides to help you improve your performance over time.

Getting Started

At this point, you should have already received your PlayHaven SDK. If not, then make sure that you visit your **Developer Dashboard** to download the SDK. If you are unable to find your SDK on the dashboard, please email **support@playhaven.com**. Once you have your SDK, we are ready to move onto the integration steps, which are largely standard aside from a few special cases outlined below.

- **Unity version 1.7 PlayHaven 1.7: Unity Integration Steps**
- **Unity Version 3.0: Unity 3.0 Integration Steps**
- **Unity and OpenFeint: Unity Integration with OpenFeint Steps**
- **Cocos2D: Cocos2D Integration**

If your game was built with any other technology other than the cases listed above then please proceed directly to the next section.

Implementation Instructions

1. Add the PlayHaven library code and headers to your project. There are three files in all:

```
libPlayHaven.a
libPlayHavenDebug.a
PlayHaven.h
```

a. The PlayHaven SDK should allow you to target iPhone OS 3.0 and above.
b. You should use a 4.0 Base SDK, targeting the armv6/7 device or simulator.

2. Add the SystemConfiguration, QuartzCore, and libicucore.dylib frameworks to your project. They are required to compile. **They are dynamic libraries so they will not increase the size of your app.**

3. In your project's build properties, ensure that -ObjC has been added to "Other linker flags".

4. Get a publisher token from **http://developers.playhaven.com** (*It's in your Developer Dashboard under "My Games"*).

a. Integrate PlayHaven into your app. Here is some sample code:

Prepare PlayHaven for future requests in your Application Delegate.

** Preload PlayHaven when your application has finished launching.*

```
@interface ApplicationDelegate : NSObject<UIApplicationDelegate> ...
-(void)applicationDidFinishLaunching:(UIApplication *)application {
[PlayHaven preloadWithPublisherToken:@"YOUR PUBLISHER TOKEN HERE" testing:YES];
...
}
```

** Before publishing your game to the App Store, you should set the testing parameter to NO. While testing is YES, you will not affect your cross-promotion rankings by using the system.*

Implementation Instructions cont.

b. Load the Charts service.

** In whichever class is convenient, after a user has completed an action indicating they would like to see "More Games/Apps", implement the vPHRequestDelegate protocol.*

```
@interface MyViewController : UIViewController <PHRequestDelegate>
...
```

** After the user has taken the prerequisite action, load the "charts" service*

```
[PlayHaven loadChartsWithDelegate:self context:nil];
```

** Listen for the necessary messages when requesting a service.*

```
-(void)playhaven:(UIView *)view didLoadWithContext:(id)contextValue
{
[myFullScreenContainerView addSubview:view];
 }

-(void)playhaven:(UIView *)view didFailWithError:(NSString *)message
context:(id)contextValue {//NSLog(@"playhaven didFailWithError:
%@", message);
[view removeFromSuperview];
}

-(void)playhaven:(UIView *)view
wasDismissedWithContext:(id)contextValue {
[view removeFromSuperview];
}
```

** Note that the context object you pass to the loadChartsWithDelegate:context: message will be passed along to each of the other callbacks. If you don't need it, just pass nil. The context object will be retained until the service has been dismissed or has failed.*

5. That's it!

Contact **support@playhaven.com** with any bugs, questions, or feedback, or check the developer community at **http://developers.playhaven.com/forum**

Integration Tips:

Now that you have integrated PlayHaven from a technical perspective, it's time to find the right place(s) for your "More Games" button. The placement of your button can dramatically affect how well the service works for you so this aspect is just as important as the technical integration. With more than 400+ games live we have learned a lot about what works well and what to avoid. Please see the **Optimization Guide** (pdf) for more details and find the right implementation strategy for you. If you still have questions regarding optimizing your "More Games" button please do not hestitate to conact **support@playhaven.com**. We would be happy to work with you to assure optimal implementation.

Submission to App Store and Launch

1. When you are ready to launch your game with PlayHaven, please visit your **Developer Dashboard** and let us know that the game has been submitted. This allows us to prepare for a smooth launch on our end.

2. **IMPORTANT**: When you are ready to launch there is a simple, but very important step to take before submitting to the App Store. If this step is not completed then PlayHaven will not work properly and you will not be able to earn any impressions:

   Turn testing mode off! Change the TESTING parameter from YES to NO:

```
-(void) applicationDidFinishLaunching:(UIApplication *)application {
        [PlayHaven preloadWithPublisherToken:@"YOUR PUBLISHER TOKEN HERE" testing:NO];
```

3. Please visit your **Developer Dashboard** when your game has gone live on the app store. This allows us to test the implementation for you and make sure everything is working properly.

Next Steps

Once your game is live, we have several tools and resources available to help you get the most out of the system and improve your performance over time. Here are some important references:

1. Developer Dashboard (**http://developers.playhaven.com**). Your dashboard provides all of the daily metrics for your games and is your tool for managing your games and campaigns.

2. Optimize your performance. For all of the latest tips on how to get more out of the PlayHaven system please refer to the **Optimization Guide** document.

Thank you again for joining PlayHaven and we look forward to seeing our service in your game soon! If you have any questions please contact your Developer Relations team member or **support@playhaven.com** at any time and we will be happy to assists you.

Appendix A: Troubleshooting

- ***"Unauthorized bootstrap components detected" or Bootstrap HTML failed to load:***

  The request timed out.
  The PlayHaven servers are likely down for maintenance. Please wait a few minutes and try again. This should very rarely happen.

- ***"Failure running bootstrap commands: Application core failed to validate."***

  If you're running on the simulator, click on "Reset content and settings" and try again. If you're on the device, delete the app, rebuild it and reinstall.

  "Tried to obtain the web lock from a thread other than the main thread or the web thread. This may be a result of calling to UIKit from a secondary thread."
  This occurs when you are building with iPhone 4 SDK and using a version before 0.3.1. Please ensure you have at least v0.3.1 - if not, please email us.

- ***"Unexpected publisher token:..."***

  You are using the incorrect lib for your App. This error most likely occurs when you are inserting the PlayHaven SDK into multiple app projects. Each app must have its own unique SDK (the lib name indicates which game it is for). There are a few ways to try to get around this:
  In your Targets -> App Name -> Link Binary with Libraries, make sure it is linking to the correct lib.
  Reset Content and Settings and attempt to build again (Simulator only).
  Rename the new PlayHaven lib file to something different and change your code accordingly. This will force Xcode to attempt to use the correct SDK.

- ***"ld: duplicate symbol OBJC_METACLASS$_PHResources..."***

  You have libPlayHaven.a and libPlayHavenDebug.a both attached to the project at the same time. Currently, you can only have one attached. Use the debug lib to find specific errors with the PlayHaven SDK and the regular PlayHaven lib for your release.

Appendix B: FAQ

Q: ***What is a/my publisher key?***

A: This key to your application. If you have multiple applications you will have multiple keys. You can find your key at any time by visiting your **Developer Dashboard** and looking under the "My Games" tab.

Q: ***Why is the library so big?***

A: libPlayHaven.a and libPlayHavenDebug.a are "fat binaries," containing code to link against both the device and the simulator. During your build process, the linker will pick out the parts of the static library that work with the architecture you are targeting. So, to estimate the size of your product after including PlayHaven, you could use an equation something like this: total_size = size_of_my_app + (size_of_libPlayHaven / number_of_architectures_in_libPlayHaven). Currently, number_of_architectures_in_libPlayHaven is 2.

Q: ***How do I implement PlayHaven alongside SDK X:***

A: Please refer to other articles in the Knowledge Base for your specific SDK.

Cocos2D Integration:

```
-(void)playhaven:(UIView *)view didLoadWithContext:(id)contextValue {
// for Cocos version >= 0.7   [[[Director sharedDirector] openGLView]
addSubview:view];   // for Cocos < 0.7   // [[[Director sharedDirector]
window] addSubview:view]; }
```

Q: ***What is the minimum OS version supported by libPlayHaven.a?***

A: iPhone OS 3.0

Q: ***Can I use libPlayHaven.a on the iPad?***

A: Not yet, though we expect to support this sooner than later.

Q: ***Are horizontal/locked orientations supported?***

A: Yes.

Appendix B: FAQ cont.

Q: ***When I try to open PlayHaven, the log is displaying a message like: "Couldn't begin service "charts": not bootstrapped"***

A: This may or may not be expected behavior. See the "Debugging" section in the Integration Instructions.

malloc: * error for object 0x308f000: pointer being freed was not allocated**

This appears to be a problem internal to the 3.0 iPhone SDK for the Simulator only. If you see these messages while running on the device, please let us know!

Q: ***Can I run other code (AdMob, MobClix, your app's code, etc) while loading a PlayHaven service?***

A: No, or do so at your own peril. We're doing our best to make sure that performance of our services are snappy, but iDevices have limited resources. Throwing other codepaths into the mix is definitely unsupported.