Introduction

Thank you for purchasing the iOS Haptic asset!

If you have questions or require assistance in deploying the iOS Haptic plugin in your project, please email the publisher at alexander@hodge.io

The iOS Haptic plugin allows you to easily add haptic feedback to your Unity iOS project using the Taptic Engine on supported devices (iPhone 7 and iPhone 7 Plus and newer iPhones).

The categories of haptic feedback exposed by the Taptic Engine API are:

1) Physical impacts (3 types: light, medium, and heavy)
2) UI Notification feedback (3 types: Success, Warning, and Error)
3) UI Selection Change feedback (single type)

New in iOS 13:

4) Transient and continuous haptic playback control
5) Two new haptic parameters: intensity and sharpness

It is strongly advised that you read and understand Apple's Human Interface Guidelines for using the Taptic Engine before submitting your app to the App Store:

https://developer.apple.com/ios/human-interface-guidelines/interaction/feedback/

Important note: Use of the Taptic Engine will only work when the Unity project containing the plugin is deployed to a supported physical device. It will not work when the project is run in the Unity Editor with the Unity Remote 5 app, or in the Xcode simulator, or on an unsupported device.

Setup

1. Import the iOS Haptic asset to your Unity project
2. Drag an instance of the iOSHapticController prefab into your scene.
3. Trigger haptic feedback from your game scripts at any time with simple calls to the iOSHapticController singleton. For example:

```
void Start()
{
        // triggers a haptic feedback simulating an impact of medium size
        iOSHapticController.instance.TriggerImpactMedium();
}
```

The full script reference is given below.

Script Reference

A prefab controller, iOSHapticController, is provided with an example scene for reference which demonstrates how to trigger haptic feedback using UI Buttons. The controller's singleton class exposes the following functions, which are responsible for triggering feedback or managing the Taptic Engine's generators:

Haptic Feedback:
TriggerImpactLight() - simulates a light physical impact
TriggerImpactMedium() - simulates a medium physical impact
TriggerImpactHeavy() - simulates a heavy physical impact
TriggerNotificationSuccess() - a type of feedback that suggests an action has been performed successfully
TriggerNotificationWarning() - a type of feedback that suggests a warning to the user
TriggerNotificationError() - a type of feedback that suggests an error has occurred
TriggerSelectionChange() - a type of feedback that communicates that a selection made by the user has changed (example scene hooks into a horizontal slider's OnValueChanged method)

Managing the Haptic Generators:
PrepareTapticEngine() - call this optional function in advance of triggering feedback in order to reduce latency (has no effect on latency if called immediately before triggering haptic feedback)
ReleaseHapticGenerators() - call this optional function if you no longer require the Taptic Engine
SetupHapticGenerators() - call this function if you've previously released the haptic generators and need to use the Taptic Engine again

For iOS 13+:
PlayTransientHaptic(intensity, sharpness) - plays a one-shot haptic event with intensity and sharpness parameters
PlayContinuousHaptic(intensity, sharpness, time) - plays a continuous haptic event with intensity and sharpness parameters for a duration of time (maximum 30 seconds)
UpdateContinuousHaptic(intensity, sharpness) - updates a playing haptic event with new intensity and sharpness values
StopHaptic() - stops the currently playing haptic event