Main Page

Classes

## **Wwise Unity Integration**

The Wwise Unity Integration is a Unity plug-in that allows the use of the Wwise Sound Engine in Unity games. It is referred to as the "Integration" in the rest of the document. The Integration consists of a Unity plug-in library, the Wwise SDK API bindings in C#, a few helpful components and Editor windows. To understand the fundamental features of the Wwise SDK, refer to its documentation available both online and as a CHM with your Wwise installation.

This Integration documentation provides information about the Unity-Wwise Integration and the Wwise API. If you need information about the Wwise authoring application (for sound designers), refer to the appropriate help file, available both online and with your local installation (usually in c:\Program Files(x86)\Audiokinetic\Wwise [version] \Authoring\Help).

- Release Notes
- Installing or Upgrading the Integration in a Unity Project
- Uninstalling the Wwise Integration Package
- Using the Wwise Unity Integration
- Build your Unity Game for a Target Platform
- Making DLCs with Wwise and Unity
- Build the Native Integration Plug-in from Source
- API Changes and Limitations
- Platform-Specific Information
- Licensing (Free & Commercial)
- Wwise Demo Scene

- Using Wwise Spatial Audio in Unity
- Troubleshooting

## **Contact Us**

If you have questions about this integration, please post your questions to our Q&A community forum. If you have purchased the support package, contact our customer support.

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3



Main Page Classes

# **Wwise Unity Integration**

## **Release Notes**

This page details the major changes to the Wwise Unity Integration by version:

- Wwise Unity Integration 2017.2.0.6500.947
  - New Features 2017.2
  - Important Migration Notes 2017.2
- Wwise Unity Integration 2017.1.4.6407.845
- Wwise Unity Integration 2017.1.3.6377.812
- Wwise Unity Integration 2017.1.2.6361.791
- Wwise Unity Integration 2017.1.1.6340.770
- Wwise Unity Integration 2017.1.0.6302.726
  - New Features 2017.1
  - Important Migration Notes 2017.1
- Wwise Unity Integration 2016.2.4.6098.531
- Wwise Unity Integration 2016.2.3.6077.504
- Wwise Unity Integration 2016.2.2.6022.430
- Wwise Unity Integration 2016.2.1.5995.409
- Wwise Unity Integration 2016.2.1
- Wwise Unity Integration 2016.2.0
- Wwise Unity Integration 2016.1.3
- Wwise Unity Integration 2016.1.2
- Wwise Unity Integration 2016.1.1
- Wwise Unity Integration 2016.1
- Wwise Unity Integration 2015.1.4
- Wwise Unity Integration 2015.1.3
- Wwise Unity Integration 2015.1.2
- Wwise Unity Integration 2015.1.1
- Wwise Unity Integration 2015.1
- Wwise Unity Integration 2014.1.6
- Wwise Unity Integration 2014.1.5
- Wwise Unity Integration 2014.1.4
- Wwise Unity Integration 2014.1.3
- Wwise Unity Integration 2014.1.2
- Wwise Unity Integration 2014.1.1
- Wwise Unity Integration 2014.1
- Wwise Unity Integration 2013.2.9

- Wwise Unity Integration 2013.2.8
- Wwise Unity Integration 2013.2.5
- Wwise Unity Integration 2013.2.4
- Wwise Unity Integration 2013.1.1
- Wwise Unity Integration 2013.1

# **Wwise Unity Integration 2017.2.0.6500.947**

This release delivers the Wwise SDK update for Wwise 2017.2.0, and compatibility with Unity 2017.3.

## **Compatibility:**

• Wwise SDK: 2017.2.0

• Unity: Unity 5.5, 5.6, 2017.1, 2017.2, and 2017.3 (Personal or Pro).

Note: Important note: This integration has **not** been tested on Unity 4.

Platforms:	Wwise Unity Integration tested on:
Mac Windows	Unity 2017.3.0f3
Android iOS Linux PS4 tvOS UWP (Universal Windows Platform) Xbox One	Unity 2017.2.0f3
Switch	Unity for Nintendo Switch 3.1.2 (Unity 5.6.4 with NintendoSDK 3.5.2)

#### **New Features:**

Added several new features, as detailed in the New Features
 2017.2 page.

## **Migration Notes:**

• Breaking changes have been made to avoid name clashes with other software packages. This is detailed in the **Important Migration Notes 2017.2** page.

- WG-25994: Made the audio input source plug-in accessible via C# scripting. See Using the Audio Input Source Plug-in in Unity. for more information.
- **WG-27337**: Added ability to post MIDI events to the sound engine. See **Sending MIDI to Wwise.** for more information.
- **WG-28541**: Added namespaces for all enums not prefixed with Ak. Added Ak prefix to all generated classes.
- WG-33501: Added automatic SoundBank management.
- WG-34446: Reduced memory allocations in AkCallbackManager.

## **Wwise Unity Integration 2017.1.4.6407.845**

## **Compatibility:**

- Wwise SDK: 2017.1.4
- Unity: Unity 5.5, 5.6, 2017.1 and 2017.2 (Personal or Pro).

Note: Important note: This integration has **not** been tested on Unity 4.

Platforms:	Wwise Unity Integration tested on:	
Android		
iOS		
Linux		
Мас		
PS4		
PS Vita	Unity 2017.2.0f3	
tvOS		
UWP (Universal		
Windows Platform)		
Windows		
Xbox One		
Switch	Unity for Nintendo Switch 3.1.2 (Unity 5.6.4	
SWILCH	with NintendoSDK 3.5.2)	

- WG-35168: Fixed crash when loading SoundBanks on Switch.
- **WG-35383**: Fixed update of IDs associated with WwiseTypes when used as properties within custom inspectors.
- WG-35384: AkAudioListener, AkInitializer, and AkTerminator now implement OnEnable() instead of Awake(), so they are disabled when unchecked in the Editor.
- WG-35513: Added full custom positioning for AkGameObj.
- **WG-35958**: Unity 2017.3 support added. Note the integration has not been thoroughly tested in Unity 2017.3, but it should be compatible.

## **Wwise Unity Integration 2017.1.3.6377.812**

This release delivers the Wwise SDK update for Wwise 2017.1.3, and compatibility with Unity 2017.1.

## **Compatibility:**

Wwise SDK: 2017.1.3

• Unity: Unity 5.5, 5.6, 2017.1 and 2017.2 (Personal or Pro).

**Note:** Important note: This integration has **not** been tested on Unity 4.

Platforms:	Wwise Unity Integration tested on:
Android	
iOS	
Linux	
Мас	
PS4	
PS Vita	Unity 2017.2.0f3
tvOS	
UWP (Universal	
Windows Platform)	
Windows	
Xbox One	
Switch	Unity for Switch 1.0.1 (Unity 5.5.0p1 with Nintendo SDK 1.4.0)

- **WG-34855**: Added AkSoundEngine.GetCurrentLanguage().
- **WG-35075**: Fixed: Added support for Unity 2017.2 integration in the Wwise Launcher.

# **Wwise Unity Integration 2017.1.2.6361.791**

This release delivers the Wwise SDK update for Wwise 2017.1.2.

## **Compatibility:**

• Wwise SDK: 2017.1.2

• Unity: Unity 2017.1 (Personal or Pro).

Note: Important note: This integration has **not** been tested on Unity 4.

Platforms:	Wwise Unity Integration tested on:
Android iOS Linux Mac PS4 PS Vita tvOS UWP (Universal Windows Platform) Windows	Unity 2017.1.1f1
Xbox One	
Switch	Unity for Switch 1.0.1 (Unity 5.5.0p1 with Nintendo SDK 1.4.0)

## **Wwise Unity Integration 2017.1.1.6340.770**

This release delivers the Wwise SDK update for Wwise 2017.1.1, and compatibility with Unity 2017.1.

### **Compatibility:**

Wwise SDK: 2017.1.1

• Unity: Unity 2017.1 (Personal or Pro).

**Note:** Important note: This integration has **not** been tested on Unity 4.

Platforms:	Wwise Unity Integration tested on:
Mac	Unity 2017.1.0f3
Windows	,
Android	
iOS	
Linux	
PS4	
PS Vita	Unity 5.6.1p1
tvOS	
UWP (Universal	
Windows Platform)	
Xbox One	
Switch	Unity for Switch 1.0.1 (Unity 5.5.0p1 with Nintendo SDK 1.4.0)

- **WG-33018**: Fixed: No localization folder is created when using the Decode Banks feature.
- WG-33818: Fixed bank decoding on iOS and Android.
- **WG-34090**: Fixed TLS Allocator Error spamming for WSA Unity Plugin.
- **WG-34188**: Fully qualified name of System.EventHandler to avoid name clashes.
- WG-34205: Fully qualified name of UnityEditor.Menu and

UnityEditor.MenuItem to avoid name clashes.

# **Wwise Unity Integration 2017.1.0.6302.726**

This release delivers the Wwise SDK update for Wwise 2017.1.0, and compatibility with Unity 2017.1.

### **Compatibility:**

Wwise SDK: 2017.1.0

• Unity: Unity 2017.1 (Personal or Pro).

**Note:** Important note: This integration has **not** been tested on Unity 4.

Platforms:	Wwise Unity Integration tested on:
Мас	Unity 2017.1.0f3
Windows	Officy 2017.1.013
Android	
iOS	
Linux	
PS4	
PS Vita	Unity 5.6.1p1
tvOS	
UWP (Universal	
Windows Platform)	
Xbox One	
Switch	Unity for Switch 1.0.1 (Unity 5.5.0p1 with Nintendo SDK 1.4.0)

#### **New Features:**

Added several new features, as detailed in the New Features
 2017.1 page.

## **Migration Notes:**

- The Wii U platform is no longer supported.
- The functionality encompassing the auto-registration of GameObject's has been moved to the Unity Integration Extensions. See more

- details about this change in the Wwise Installation and Migration Guide.
- There are a number of breaking changes made to the AkcallbackManager, which are detailed in the Important Migration Notes 2017.1 page.

- WG-27479: Made callback buffer size configurable via AkInitializer.
- WG-30791: Added WwiseTypes. See WwiseTypes for more information.
- WG-31155: Fixed AkmemSettings and AkchannelConfig so that their methods do not check for SoundEngine initialization.
- WG-31735: Added hooks enabling users to easily extend or remove auto-registration of GameObjects. See Unity Integration
   Extensions for more information.
- WG-32348: Fixed XML parsing of SoundBanksInfo file.
- **WG-32657**: Reduced memory allocations when using callback classes, and removed foreach statements.
- **WG-33303**: Added support for AK\_MusicPlaylistSelect.
- **WG-34003**: Installation procedure will not deactivate other native plugins in the project anymore.

# **Wwise Unity Integration 2016.2.4.6098.531**

This release delivers the Wwise SDK update for Wwise 2016.2.4.

## **Compatibility:**

• Wwise SDK: 2016.2.4

• Unity: Unity 5.6 (Personal or Pro).

Note: Important note: This integration has **not** been tested on Unity 4.

Platforms:	<b>Wwise Unity Integration tested on:</b>
Android iOS Linux Mac PS4 PS Vita tvOS Windows Windows Store (Universal Windows Platform) Xbox One	Unity 5.6.1p1
Switch	Unity for Switch 1.0.1 (Unity 5.5.0p1 with NintendoSDK 1.4.0)
Wii U	Unity 5.4.2f2

### **Resolved Issues:**

• WG-33395: Added namespace around potentially conflicting class names

# **Wwise Unity Integration 2016.2.3.6077.504**

This release delivers the Wwise SDK update for Wwise 2016.2.3.

## **Compatibility:**

Wwise SDK: 2016.2.3

• Unity: Unity 5.6 (Personal or Pro).

**Note:** Important note: This integration has **not** been tested on Unity 4.

Platforms:	Wwise Unity Integration tested on:
Android	
iOS	
Linux	
Mac	
PS4	
PS Vita	Unity 5.6.0f3
tvOS	
Windows	
Windows Store (Universal Windows	
Platform)	
Xbox One	
Switch	Unity for Switch 0.6.1
Wii U	Unity 5.4.2f2

- WG-32536: Added Nintendo Switch support.
- WG-32623: Fixed Android music custom cue crash.

# **Wwise Unity Integration 2016.2.2.6022.430**

This release delivers the Wwise SDK update for Wwise 2016.2.2.

## **Compatibility:**

• Wwise SDK: 2016.2.2

• Unity: Unity 5.5 (Personal or Pro).

**Note:** Important note: This integration has **not** been tested on Unity 4.

Platforms:	Wwise Unity Integration tested on:
Android	
iOS	
Linux	
Mac	
PS4	
PS Vita	Unity 5.5.0p4
tvOS	
Windows	
Windows Store (Universal Windows	
Platform)	
Xbox One	
Wii U	Unity 5.4.2f2

### **Resolved Issues:**

• WG-31862: Removed enum values that are not associated with event callbacks from event callback list.

# **Wwise Unity Integration 2016.2.1.5995.409**

This release delivers the Wwise SDK update for Wwise 2016.2.1.

## **Compatibility:**

• Wwise SDK: 2016.2.1

• Unity: Unity 5.5 (Personal or Pro).

**Note:** Important note: This integration has **not** been tested on Unity 4.

Platforms:	Wwise Unity Integration tested on:
Android	
iOS	
Linux	
Mac	
PS4	
PS Vita	Unity 5.5.0f3
tvOS	
Windows	
Windows Store (Universal Windows	
Platform)	
Xbox One	
Wii U	Unity 5.4.2f2

### **Resolved Issues:**

• WG-32006: Fixed "Unknown Game Object ID" error when setting a GameObject's position.

This release delivers the Wwise SDK update for Wwise 2016.2.1.

## **Compatibility:**

Wwise SDK: 2016.2.1

• Unity: Unity 5.5 (Personal or Pro).

**Note:** Important note: This integration has **not** been tested on Unity 4.

Platforms:	Wwise Unity Integration tested on:
Android	
iOS	
Linux	
Mac	
PS4	
PS Vita	Unity 5.5.0f3
tvOS	
Windows	
Windows Store (Universal Windows	
Platform)	
Xbox One	
Wii U	Unity 5.4.2f2

- WG-27085: Reduced the allocations due to AuxSends
- WG-31127: Fixed the returned user cue name for AK MusicSyncUserCue.
- WG-31650: Fixed NullReferenceException in **AkGameObj** when the SoundEngine was not initialized.
- WG-31651: Avoid the automatic creation of **AkGameObj** when calling SetObjectPosition.
- WG-31862: Removed enum values that are not associated with event callbacks from event callback list.

## **Known Issues**

• Unity case #861189: Installation from the Launcher fails if the Unity Windows Store playback engine is installed.

This release delivers the Wwise SDK update for Wwise 2016.2.0.

### **Compatibility:**

Wwise SDK: 2016.2.0

• Unity: Unity 5.4 (Personal or Pro).

**Note:** Important note: This integration has **not** been tested on Unity 4.

Platforms:	Wwise Unity Integration tested on:
Android	
iOS	
Linux	
Mac	
PS4	
PS Vita	Unity 5.4.1p2
tvOS	
Windows	
Windows Store (Universal Windows	
Platform)	
Xbox One	
Wii U	Unity 5.2.4f1

## **Migration Notes**

- Windows Phone 8.1, and Windows Store 8.0 and 8.1 platforms have been deprecated. The only supported Windows Store platform is now Universal Windows Platform (UWP).
- The Xbox 360 platform is no longer supported.
- The PS3 platform is no longer supported.

### **Resolved Issues:**

• WG-30571: Rebuilding the Library folder will no longer trigger an error message saying the Wwise Unity integration is not installed for

a platform.

- WG-30960: Applying a position offset to an AkGameObj is now reflected in Unity's dirty system.
   WG-31507: Add MediaID and bStreaming to duration callback.

This release delivers the Wwise SDK update for Wwise 2016.1.3.

## **Compatibility:**

• Wwise SDK: 2016.1.3

• Unity: Unity 5.4 (Personal or Pro).

**Note:** Important note: We no longer officially support the Integration for Unity 4. While it may work, we will not test this or future versions of the Wwise Integration for Unity on Unity 4.

Platforms:	Wwise Unity Integration tested on:
Android	
iOS	
Linux	
Мас	
PS3	
PS4	Unity 5.4.0p2
PS Vita	
tvOS	
Windows	
Windows Store	
Xbox One	
Wii U	Unity 5.2.4f1
Xbox 360	Unity 5.0.1f1

This release delivers the Wwise SDK update for Wwise 2016.1.2. The Wwise Unity integration documentation is also now available in Japanese.

## **Compatibility:**

• Wwise SDK: 2016.1.2

• Unity: Unity 5.4 (Personal or Pro).

**Note:** Important note: This integration has **not** been tested on Unity 4.

Platforms:	<b>Wwise Unity Integration tested on:</b>
Android	
iOS	
Linux	
Мас	
PS3	
PS4	Unity 5.4.0p2
PS Vita	
tvOS	
Windows	
Windows Store	
Xbox One	
Wii U	Unity 5.2.4f1
Xbox 360	Unity 5.0.1f1

#### **Resolved Issues:**

• WG-30567 Fixed: Crash when using WAV markers having an empty label.

## **Migration Notes:**

• SWIG sources are not distributed with the source ZIP files anymore. The provided ZIP sources contain the necessary files and pre-

- generated SWIG interfaces required to build the integration package without the need for SWIG.
- Projects still using Wwise 2014.1.4 and less are not directly upgradeable to 2016.1. Please first update your project to 2015.1.6, and then to 2016.1.
- If you wish to update your Unity project to Unity 5.4, please perform the Wwise migration to 2016.1.2 first. Some changes in Unity 5.4 broke the older integrations, and updating to the Wwise 2016.1.2 integration first resolves these issues.

This release delivers the Wwise SDK update for Wwise 2016.1.1.

### **Compatibility:**

Wwise SDK: 2016.1.1

• Unity: Unity 5.3 (Personal or Pro).

Note: Important note: This integration has **not** been tested on Unity 4.

Platforms:	Wwise Unity Integration tested on:
Android	
iOS	
Linux	
Мас	
PS3	
PS4	Unity 5.3.5p6
PS Vita	
tvOS	
Windows	
Windows Store	
Xbox One	
Wii U	Unity 5.2.4f1
Xbox 360	Unity 5.0.1f1

- WG-30021: Use the last modified date of a decoded SoundBank to decide if it needs to be re-decoded.
- WG-30228: It is now possible to rebuild the Linux integration from source.
- WG-30231: It is now possible to rebuild the tvOS integration from source.
- WG-30259: Fix AkGameObj registration behavior when calling GC.Collect.

## **Known issues:**

- WG-30128: It is not possible to save decoded SoundBanks to disk on the Vita platform.
- WG-30139: It is not possible to save decoded SoundBanks to disk on the Wii U platform.

This release delivers the Wwise SDK update for Wwise 2016.1.

## **Compatibility:**

Wwise SDK: 2016.1

• Unity: Unity 5.3 (Personal or Pro).

Note: Important note: This integration has **not** been tested on Unity 4.

Platforms:	<b>Wwise Unity Integration tested on:</b>
Android	
iOS	
Linux	
Мас	
PS3	Unity 5.3.4p3
PS4	Offity 5.5.4p5
PS Vita	
Windows	
Windows Store	
Xbox One	
Wii U	Unity 5.2.4f1
Xbox 360	Unity 5.0.1f1

- WG-25675 Fixed: Added a "Generate SoundBanks" button in the Wwise Picker to generate the SoundBanks for all platforms of the Wwise project.
- WG-27583 Fixed: Now automatically managing the SoundBank folders in the Unity assets, pointing to the right SoundBank for the current Unity target platform.
- WG-28175 Fixed: WwiseGlobal is not always automatically created when creating a new scene.
- WG-26011 Fixed: Enabling back Ak Audio Listener after creating a new Scene doesn't properly destroy Default Unity Audio Listener.

## **Migration Notes:**

- SWIG sources are not distributed with the source ZIP files anymore. The provided ZIP sources contain the necessary files and pregenerated SWIG interfaces required to build the integration package without the need for SWIG.
- Projects still using Wwise 2014.1.4 and less are not directly upgradable to 2016.1. Please first update your project to 2015.1.6, and then to 2016.1.

### **Known issues:**

- WG-30021: The DecodeBanks feature is deactivated when previewing a scene in the Editor.
- WG-30021: The DecodeBanks feature does not automatically delete the decoded SoundBanks. If SoundBanks are updated, the DecodedBanks folder should be removed from the target devices.
- WG-30128: It is not possible to save decoded SoundBanks to disk on the Vita platform
- WG-30139: It is not possible to save decoded SoundBanks to disk on the WiiU platform

This release delivers the Wwise SDK update for Wwise 2015.1.4.

## **Compatibility:**

Wwise SDK: 2015.1.4

• Unity: Unity 4.6 Pro or Unity 5.2 (Personal or Pro)

Platforms:	<b>Wwise Unity Integration tested on:</b>
Android	
iOS	
Linux	
Мас	
Metro	
PS3	Unity 5.2.2p1
PS4	
PS Vita	
Windows	
Windows Phone	
Xbox One	
Wii U	Unity 4.3.7f1 with Wii U add-on 2.2.5
Xbox 360	Unity 5.0.1f1

- WG-28412 Fixed: **AkGameObj** position offset can now be saved in a prefab.
- WG-28723 Fixed: It is now possible to build the game as a PS4 package.

This release delivers the Wwise SDK update for Wwise 2015.1.3, and the Universal Windows Platform.

### **Compatibility:**

Wwise SDK: 2015.1.3

• Unity: Unity 4.6 Pro or Unity 5.2 (Personal or Pro)

Platforms:	Wwise Unity Integration tested on:
Android	Unity 5.2.2p1
iOS	
Linux	
Mac	
PS3	
PS4	
PS Vita	
Windows	
Windows Store Apps	
Xbox One	
Wii U	Unity 4.3.7f1 with Wii U add-on 2.2.5
Xbox 360	Unity 5.0.1f1

#### **New Features:**

- Added support for the Universal Windows Platform under the Windows Store Apps package.
- Added support for Windows Store Apps SDK 8.1.
- Removed support for Windows Phone 8.0.
- Removed support for Windows Store Apps SDK 8.0.

- WG-25945 Fixed: Wwise errors were always redirected to the Unity console. Added a menu item in the WwiseGlobal settings to enable/disable the Wwise engine output in the Unity console.
- WG-26011 Fixed: Enabling back Ak Audio Listener after creating a

- new Scene didn't properly destroy Default Unity Audio Listener.
- WG-28108 Fixed: Sound kept playing when pausing editor. Sound is now stopped when pausing in the editor and starts playing again when unpausing. Sound keeps playing (as before) when switching windows.
- WG-28175 Fixed: WwiseGlobal was not always automatically created when creating a new scene.
- WG-28479 Fixed: Setup attempted to add unchecked option. Setup now succeeds when installing the integration in a project without a Main Camera.
- WG-28526 Fixed: Potential Unity Editor crash. Made sure the SoundEngine is initialized before registering and unregistering GameObjects.

This release delivers the Wwise SDK update for Wwise 2015.1.2.

## **Compatibility:**

• Wwise SDK: 2015.1.2

• Unity: Unity 4.6.5 Pro or Unity 5.1.2p2 (Personal or Pro)

Platforms:	<b>Wwise Unity Integration tested on:</b>
Android	
iOS	
Linux	
Мас	
Metro	
PS3	Unity 5.2.0f3
PS4	
PS Vita	
Windows	
Windows Phone	
Xbox One	
Wii U	Unity 4.3.7f1 with Wii U add-on 2.2.5
Xbox 360	Unity 5.0.1f1

- WG-27029 Fixed: Fix rare occurrence where a deactivated Unity GameObject might result in the associated AkGameObj being unregistered.
- WG-28200 Fixed: Default active listener set to Everything affected 3D positioning of GameObjects close to (0, 0, 0). Resolved by setting default listener mask to L0.

This release delivers the Wwise SDK update for Wwise 2015.1.1.

### **Compatibility:**

Wwise SDK: 2015.1.1

• Unity: Unity 4.6.5 Pro or Unity 5.1.2p2 (Personal or Pro)

Platforms:	Wwise Unity Integration tested on:
Android	
iOS	
Linux	
Mac	
Metro	
PS3	Unity 5.1.2p2
PS4	
PS Vita	
Windows	
Windows Phone	
Xbox One	
Wii U	Unity 4.3.7f1 with Wii U add-on 2.2.5
Xbox 360	Unity 5.0.1f1

- WG-27977 Fixed: Unity crash when terminating, if the script order is wrong.
- WG-28030 Fixed: Wii U/Unity: Release RPL is not tagged NDEBUG as required by the TRC.
- WG-28042 Fixed: Unity: Not possible to run game if the Wwise project is not accessible.
- WG-28044 Fixed: Unity: Performance issue, Wwise project is reparsed endlessly.
- WG-28046 Fixed: Remove mandatory dependency on an **AkInitializer** instance.
- WG-28048 Fixed: Properly display the GameObject names in the

Wwise profiler.

This release delivers the Wwise SDK update for Wwise 2015.1.

### **Compatibility:**

• Wwise SDK: 2015.1

#### **Known Issues:**

• WG-25669: Auto Populate of the Wwise Picker doesn't work in the Mac Editor.

#### **Resolved Issues:**

• WG-27079 Fixed: Properly set the encoding of the WwiseSettings.xml files for non-English locales.

This release delivers the Wwise SDK update for Wwise 2014.1.6.

### **Compatibility:**

Wwise SDK: 2014.1.6

• Unity: Unity 4.6.5 Pro, or Unity 5.0.2 (Personal or Pro)

Platforms:	Wwise Unity Integration tested on:
Android iOS Linux Mac Metro PS3	Unity 5.0.2p3
PS4 PSVita Windows Windows Phone	
Xbox 360	Unity 5.0.1f1
Xbox One	Unity 5.0.2p4

#### **Resolved Issues:**

- WG-27585 Fixed: Properly display the Wwise project hierarchy in the Wwise Picker window.
- WG-27624 Fixed: (PS3) SetListenerPostion doesn't work on PS3.

### **Known Unity Issues:**

- Xbox One: SoundEngine initialization fails even if Unity audio is disabled. See **Troubleshooting** for a workaround.
- Windows Store Apps in Unity 4: The built game throws a DllNotFoundException. See **Troubleshooting** for a workaround.

#### **Resolved Issues:**

WG-27585 Fixed: Properly display the Wwise project hierarchy in the Wwise Picker window.

This release delivers the Wwise SDK update for Wwise 2014.1.5.

### **Compatibility:**

Wwise SDK: 2014.1.5

• Unity: Unity 4.6.5 Pro, or Unity 5.0.2 (Personal or Pro)

Platforms:	Wwise Unity Integration tested on:
Android iOS Linux Mac Metro PS3	Unity 5.0.2p3
PS4 PSVita Windows Windows Phone	
Xbox 360	Unity 5.0.1f1
Xbox One	Unity 5.0.2p4

#### **Known Issues:**

 WG-25669: Auto Populate of the Wwise Picker doesn't work in the Mac Editor.

### **Known Unity Issues:**

- Xbox One: SoundEngine initialization fails even if Unity audio is disabled. See **Troubleshooting** for a workaround.
- Windows Store Apps in Unity 4: The built game throws a DllNotFoundException. See **Troubleshooting** for a workaround.

#### **New Features:**

Added support for Android x86.

• Removed the need for Scripting Define Symbols for the Windows Store Apps platform.

#### **Resolved Issues:**

- WG-27108 Fixed: Prevent a Unity error message when posting events during the Destroy phase.
- WG-25733 Fixed: It is now possible to build iOS/Mac games using the Windows editor and vice-versa.
- WG-26875 Fixed: The **AkMemBankLoader** component now allocates aligned memory, preventing failures on certain platforms.

This release delivers the Wwise SDK update for Wwise 2014.1.4, and compatibility with Unity 5.

### **Compatibility:**

Wwise SDK: 2014.1.4

• Unity: Unity 4 Pro, or Unity 5 (Personal or Pro)

#### **Resolved Issues:**

- WG-26780 Fixed: Not having the Wwise project present on the machine will no longer generate errors in the Unity console window.
- WG-26837 Fixed: Fix the SoundBank reference counting in the AkBankManager to prevent premature unloading.

#### **Known Issues:**

- WG-25669: Auto Populate of the Wwise Picker doesn't work in the Mac Editor.
- WG-25733: Cannot build iOS/Mac games using the Windows editor and vice-versa.

#### **New Features:**

- Added support for the Windows Phone 8.1 platform (accessible via the Metro platform in the build settings).
- Now fully compatible with Unity 5. The Wwise Unity Integration now uses the new Plugin Importer API to activate its plug-ins. The <unity\_project\_root>/assets/plugins folder will no longer be used by the integration.

**Migration notes:** The recommended procedure to upgrade your Unity 4 project to Unity 5 is:

- 1. Make sure you have a backup of your project
- 2. Open your Unity 4 project in Unity 5. This will trigger the Unity

- upgrade assistant.
- 3. Once the upgrade to Unity 5 is done, import the 2014.1.4 Unity integration
- 4. A migration window will appear, click "Start"
- 5. Once the migration window closes, the migration is done
- 6. You are now ready to use the Wwise Unity Integration in Unity 5!

This release delivers the Wwise SDK update for Wwise 2014.1.3.

### **Compatibility:**

• Wwise SDK: 2014.1.3

#### **Known Issues:**

- WG-25669: Auto Populate of the Wwise Picker doesn't work in the Mac Editor.
- WG-25733: Cannot build iOS/Mac games using the Windows editor and vice-versa.

This release delivers the Wwise SDK update for Wwise 2014.1.2 and bug fixes.

### **Compatibility:**

- Wwise SDK: 2014.1.2
- This version is unavailable for the Xbox One platform due to XDK incompatibilities between the Unity Editor (October XDK) and the Wwise SDK (November XDK).

#### **Resolved Issues:**

- WG-26305 Fixed: Using null GameObject reports an unregistered GameObject on 64-bit builds.
- WG-26337 Fixed: Added compatibility to the Mac 64-bit platform.
- WG-26385 Fixed: Wwise takes CPU in Unity when in background on Android.
- WG-26395 Fixed: Fixed build failure when building Windows Phone 8 project from the command line using MSBUILD.
- WG-26430 Fixed: Fixed crash when calling SetBasePath() on iOS.

#### **Known Issues:**

- WG-25669: Auto Populate of the Wwise Picker doesn't work in the Mac Editor.
- WG-25733: Cannot build iOS/Mac games using the Windows editor and vice-versa.

This release delivers the Wwise SDK update for Wwise 2014.1.1, final documentation, and bug fixes.

### **Compatibility:**

Wwise SDK: 2014.1.1

• Unity: 4.x (5.0 not supported yet)

#### **New Features:**

- Added the support for the Wii U platform.
- Added Plug-in Registration documentation.
- Exposed the Wwise Sound Engine version in the Unity "Wwise Settings" dialog.
- Updated documentation with more accurate information about the GeneratedSoundBanks and Populate button.

#### **Resolved Issues:**

- WG-26201: Explicitly call the scope of C# System calls Known Issues:
- WG-25669: Auto Populate of the Wwise Picker doesn't work in the Mac Editor.
- WG-25733: Cannot build iOS/Mac games using the Windows editor and vice-versa.
- WG-25669: Auto Populate of the Wwise Picker doesn't work in the Mac Editor.
- WG-25733: Cannot build iOS/Mac games using the Windows editor and vice-versa.

This release delivers the Wwise SDK update for Wwise 2014.1, bug fixes, a new Demonstration scene, and a migration process for projects created with the Wwise Unity Integration 2013.2.

#### **Notable features:**

- Stability fixes
- Added a demonstration scene
- Added a migration process for projects created with the 2013.2
   Wwise Unity Integration

### **Compatibility:**

- Wwise SDK: 2014.1
- Unity: 4.x

### **Supported Platforms:**

- Android
- iOS
- Linux
  - 32-bit
  - 64-bit
- Mac OS X (10.6 and above)
- PS3
- PS4
- PS Vita
- Windows
  - 32-bit
  - 64-bit
- Windows 8 (Metro)
  - Desktop
  - Windows Store App on Intel devices and ARM tablets
- Windows Phone 8.0
- Xbox360
- Xbox One

#### **New Features:**

- Added a demonstration scene, available in a separate package. See
   Wwise Demo Scene for more information.
- Added a migration process for projects created with the 2013.2
   Wwise Unity Integration.
- Wwise Components are now referenced by UUID within the Wwise Unity Integration. This allows renaming them in the Wwise Project without breaking the link with the Unity project.
- Added some customizable triggers for posting events and loading banks. See Adding New Triggers for Wwise Events for more information.
- Added some customizable callbacks for Wwise events. See How to use AkAmbient with the inspector for more information.
- Reduced the AkSoundEngine plug-in size.
- Added an option to create (or destroy) the WwiseGlobal object and the AkListener.

#### **Resolved Issues:**

- WG-25783: Fix a memory leak in the Callback Manager
- WG-25677: Fix path issues on Mac

#### **Known Issues:**

- WG-25669: Auto Populate of the Wwise Picker doesn't work in the Mac Editor.
- WG-25733: Cannot build iOS/Mac games using the Windows editor and vice-versa.

This release delivers the Wwise SDK update for Wwise 2013.2.9, final documentation, and bug fixes.

#### **Notable features:**

- Final version of the documentation
- Stability fixes

### **Compatibility:**

- Wwise SDK: 2013.2.9
- Unity: 4.x

### **Supported Platforms:**

- Android
- iOS
- Linux
  - 32-bit
  - ∘ 64=bit
- Mac OS X (10.6 and above)
- PS3
- PS4
- PS Vita
- Windows
  - 32-bit
  - 64-bit
- Windows 8 (Metro)
  - Desktop
  - Windows Store App on Intel devices and ARM tablets
- Windows Phone 8.0
- Xbox360
- Xbox One

#### **New Features:**

Added the support for the Windows Phone 8.0 platform.

This release delivers the Wwise SDK update for Wwise 2013.2.8, new Wwise-Unity integration tools, additional documentation, and bug fixes.

#### **Notable features:**

- A simpler setup flow for new projects (see Installing or Upgrading the Integration in a Unity Project)
- Easier Wwise project access with the Wwise Picker (see Wwise Picker)
- New and improved Unity components to wrap common audio tasks (see Using the Wwise Unity Integration)
- Wwise-Unity Integration documentation merged and crossreferenced with Wwise SDK documentation (this file!)

### **Compatibility:**

Wwise SDK: 2013.2.8

• Unity: 4.x

### **Supported Platforms:**

- Android
- iOS
- Linux\*
  - 32-bit
  - 64-bit
- Mac OS X (10.6 and above)
- PS3
- PS4
- PS Vita
- Windows
  - 32-bit
  - 64-bit
- Windows 8 (Metro)
  - Desktop
  - Windows Store App on Intel devices and ARM tablets

- Windows Phone 8.0
- Xbox360
- Xbox One

#### **New Features:**

- Added the support for the Xbox One platform.
- Added the support for the PS4 platform.
- Added the support for the Linux platform.
  - \* Unity Integration for Linux is supported in Wwise 2013.2.8 Linux BETA.
- Added the support for the Windows Phone 8.0 platform.

### **Migrations:**

- Multi-architecture Windows and Metro plug-ins are now installed into their designated folders and no longer need to be switched using Unity's Scripting Define Symbol feature.
- Removed the feature for rebuilding the Integration from Unity Editor menu for simplicity.
- Separated the ready-to-use Integration from the source code.
- Upgraded Windows IDE solution from Visual Studio 2008 to 2010.
- Updated the designated platform folders for Windows and Metro.

This release delivers the Wwise SDK update for Wwise 2013.2.5, the support of new platforms, and bug fixes.

### **Compatibility:**

• Wwise SDK: 2013.2, 2013.2.x

• Unity: 4.x

### **Supported Platforms:**

- Android
- iOS
- Mac OS X (10.6 and above)
- PS3
- PS4
- PS Vita
- Windows
  - 32-bit
  - 64-bit
- Windows 8 (Metro)
  - Desktop
  - Windows Store App on Intel devices and ARM tablets
- Xbox 360
- Xbox One

#### **New Features:**

- Added the support for the new platform: PS4.
- Added the support for the new platform: PS Vita.
- Added the support for the new platform: Xbox One.
- Android: Added a new API AddBasePath() to allow specifying a POSIX file system path for I/O.

#### **Resolved Issues:**

• WG-24351: iOS: Remote control interruption is not handled.

This release delivers Wwise SDK update for Wwise 2013.2.4, bug fixes, and improvements.

### **Compatibility:**

• Wwise SDK: 2013.2, 2013.2.x

• Unity: 4.x

### **Supported Platforms:**

- Windows 8 (Metro)
  - Desktop
  - Windows Store App on Intel devices and ARM tablets
- Windows
  - 32-bit
  - 64-bit
- Mac OS X (10.6 and above)
- iOS
- Android
- Xbox360
- PS3

#### **New Features:**

- New menu item Wwise > Help for opening the Integration documentation in the default web browser.
- Added Version.txt metadata file to all distributions to indicate the base Wwse SDK info.

#### **Resolved Issues:**

 WG-24080: Compiler errors after switching to Android with both Windows and Mac APIs present.

### **Migrations:**

• Removed Android armeabi plug-ins.

This release delivers the Wwise SDK update for Wwise 2013.2.1, new features including an overhauled distribution, installation and build experience, and bug fixes.

### Compatibility:

Wwise SDK: 2013.2, 2013.2.1

Unity: 4.x (3.x is no longer supported)

### **Supported Platforms:**

- Windows 8 (Metro)
  - Desktop
  - Windows Store App on Intel devices and ARM tablets
- Windows
  - 32-bit
  - 64-bit
- Mac OS X (10.6 and above)
- iOS
- Android
- Xbox360
- PS3

#### **New Features:**

- Unity Integration is now distributed as a Unity asset bundle that contains the target platform version and the desktop version. It can be imported directly into a Unity project, and function in both the Editor and on target devices.
- A new Wwise menu was added in the Unity Editor UI for the following tasks:
  - Installing the plug-in.
  - Rebuilding the Integration.
  - Converting a SoundBank ID C++ header into its C# version.
- Build and run applications for multi-architecture platforms (for example, Windows) with Unity 4's Scripting Define Symbols feature.

- A build script for rebuilding the Integration for any supported platform through the command line.
- A simplified positioning demo added to the IntegrationDemo.
- Android: SoundBanks can now be saved in and loaded from the Android Expansion Files (OBB).

#### **Resolved Issues:**

- WG-23781: Bank callback does not work.
- WG-23734: Marker callback receives garbled strings on Windows platforms.
- WG-23345: Compiler errors when switching to unsupported platforms in Unity Editor.
- WG-23436: Unity: listener positioning error due to wrong index type cast.
- WG-23423: Performance issue with GameObjects in Unity related to component access.
- WG-22533: Unity: Some API binding functions shouldn't have exception handlers.

### Improvements:

- Simplified iOS plug-in installation and deployment.
- Updated documentation with more accurate prerequisites and other instructions.

### **Migrations:**

- The installed file and folder structure is updated for the new distribution method. See Installing or Upgrading the Integration in a Unity Project for details.
- The plug-ins for multi-architecture platforms are renamed. See BUILD THE APPLICATION FOR A MULTI-ARCHITECTURE PLATFORM for details.
- On iOS, there is no longer a need to configure Xcode project to find the plug-in library. But a new native API call is needed. See Build for iOS for details.
- The previous multi-architecture platform build UI script examples are removed.

- The old build script for the Apple platforms is removed.
  SWIG is upgraded to 2.0.11. Mac users need to refer to pg\_compileswig to update SWIG.

This release includes the Wwise SDK update for Wwise 2013.1.1. No other changes were made in the Integration.

This release includes the Wwise SDK update for Wwise 2013.1, Android-related new features, and general bug fixes.

### **Compatibility:**

Wwise SDK: 2012.2.x, 2013.1

• Unity: 3.4.x, 3.5.x, 4.x

### **Supported Platforms:**

- Windows 8
  - Desktop
  - Windows Store App on Intel devices and ARM tablets
- Windows (32bit)
- Windows (64bit)
- Mac OS X (10.6 and above)
- iOS
- Android
- Xbox360
- PS3

#### **New Features:**

- Android: SoundBanks can now be saved in Android application packages (apk) and are accessible transparently using LoadBank() APIs.
- Support for Windows 64bit.
- Support for Windows 8.

#### **Resolved Issues:**

- WG-22948: Positioning error due to listener index assignment error on PS3.
- WG-22938: Positioning data passed by Unity Integration on Mac OSX is invalid.
- WG-22334: Music userCue names are incorrectly reported to Unity

- as empty strings.
- WG-22329: Cannot PostEvent() to global scope with Unity Integration.
- WG-22255: Unity Android: Sound persists after switching app to background.
- WG-22165: 3D positioning does not work properly under iOS and Android.
- WG-21933: AkCallbackManager.cs handles deserialization of floats incorrectly. UserCue names broken.
- WG-21365: Unity iOS: Cannot hear sounds after iOS audio interruption events come and go.
- WG-22533: Some API binding functions shouldn't have exception handlers.

### Improvements:

- Windows and Mac distributions can coexist in the same Unity project.
- Improved documentation readability and accuracy and added Platform-Specific Information section for easily reading the info about platforms of interest.
- Build logs now support quiet and verbose modes for readability.
- Build pipeline now checks user Python version and warns if it is outdated.
- Deployment script can now be used directly to copy the Integration to Unity projects.

### **Migrations:**

- Visual Studio 2010 is now required to build the Integration for PS3.
- Build scripts no longer support Python 2.6 and older versions. Python 2.7.x or 3.x is required.
- The environment variable UNITY\_PROJECT\_ROOT is no longer required for building the Integration yourself. By default, the Integration is no longer automatically deployed to the folder specified by UNITY\_PROJECT\_ROOT. See Build the Native Integration Plug-in from Source for details.
- Android build scripts no longer deploy PostprocessBuildPlayer scripts to Unity projects.

• IntegrationDemo project is upgraded to Unity4 format.

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3

Main Page Classes

# **Wwise Unity Integration » Release Notes**

## **New Features 2017.2**

There are a number of important new features in Wwise 2017.2 and its Unity Integration. These include the following:

- Edit Mode Support
  - Loading and Unloading Banks in Edit Mode
- Additions to WwiseGlobal Game Object
- Wwise Audio Input Plug-in
- MIDI Events
- Automatic SoundBank Management

## **Edit Mode Support**

The Wwise sound engine is now initialized while Unity is in Edit mode. This means Wwise Events can be triggered from the editor without entering Play mode. In order to demonstrate this, the inspectors for AkEvent (and AkAmbient) have been updated. There are now buttons in the inspector that can be used to play and stop the Events directly from the editor. The Play/Stop button plays and stops individual Events. When multiple objects that have AkEvents are selected, the Play Multiple and Stop Multiple buttons play or stop all currently selected Events. The Stop All button stops all currently playing Events.



### Loading and Unloading Banks in Edit Mode

Note that in order for the edit mode functionality to work correctly, the corresponding **AkBank** components should have their Load On: property set to Awake and their Unload On: property set to Never.

## **Additions to WwiseGlobal Game Object**

The WwiseGlobal game object now has an AkEditorEventPlayer component. This component is editor only, and is used to play WwiseEvents that are associated to **AkEvent** components in Edit mode. The WwiseGlobal object also now has Rigidbody, **AkGameObj**, and **AkAudioListener** components. These are used for interactions between **AkEnvironment**, **AkGameObj**, and **AkRoom** components in Edit mode. See **Changes Made to Your Unity Project**.

# **Wwise Audio Input Plug-in**

The Wwise Audio Input Source Plug-in is now accessible via C# scripting. See **Using the Audio Input Source Plug-in in Unity.** for more information.

## **MIDI Events**

MIDI events can now be posted to Wwise via C# scripting. See **Sending MIDI to Wwise.** for more information.

# **Automatic SoundBank Management**

Automatic SoundBank management has been added. See **Deploying SoundBanks in single-platform projects** and **Deploying SoundBanks in multi-platform projects** for more information.

# **Spatial Audio Integration**

Added spatial audio API to the Unity integration. See **Using Wwise** Spatial Audio in Unity for a tutorial.

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3



Main Page Classes

# **Wwise Unity Integration » Release Notes**

# **Important Migration Notes 2017.2**

The following enumerations have changed names:

Pre-2017.2:	New in 2017.2:
ErrorCode	AK.Monitor.ErrorCode
ErrorLevel	AK.Monitor.ErrorLevel
DynamicSequenceType	AK.SoundEngine.DynamicSequence.DynamicSequenceTy
MultiPositionType	AK.SoundEngine.MultiPositionType
PreparationType	AK.SoundEngine.PreparationType
RTPCValue_type	AK.SoundEngine.Query.RTPCValue_type

The following classes have changed names:

Pre-2017.2:	New in 2017.2:
Iterator	AkIterator
Playlist	AkPlaylist
PlaylistItem	AkPlaylistItem



## **Wwise Unity Integration » Release Notes**

#### **New Features 2017.1**

There are a number of important new features in Wwise 2017.1 and its Unity Integration. These include the following:

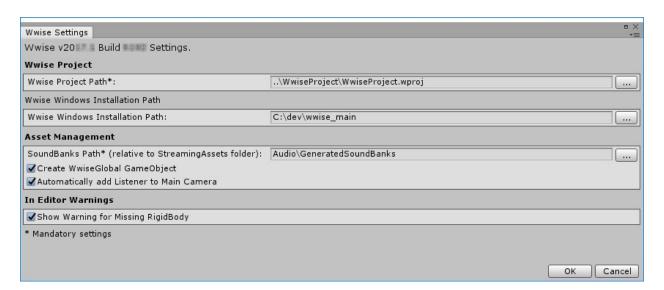
- Required Components
- WwiseTypes
- Unity Integration Extensions
- 3D Busses

### **Required Components**

[RequireComponent(typeof(AkGameObj))] has been added to AkEvent.

AkEnvironment and AkEnvironmentPortal no longer specify Rigidbody as a required component, and the members of Rigidbody are no longer modified. If an AkEnvironment or an AkEnvironmentPortal does not have a Rigidbody attached to its game object or an AkGameObj is set to "Environment aware" and does not have a Rigidbody attached to its game object, the following warning will be displayed in the Editor: AkGameObj-AkEnvironment interactions require a Rigidbody component on the object or the environment.

All instances of the above warning can be removed by unchecking the Show Warning for Missing RigidBody Option within the Wwise Settings window.



### **WwiseTypes**

New "WwiseTypes" have been added that provide users with finer control over features represented by Wwise objects such as Events, game parameters, Switches, and States. These new classes each come with their own property drawers that bring up a Wwise picker when clicked, allowing for rapid prototyping of game audio systems.



Below is a brief example use case:

```
public class Footsteps : MonoBehaviour
{
    [Header("Wwise Types")]
    public AK.Wwise.Event FootStepEvent = null;
    public AK.Wwise.RTPC SpeedRTPC = null;
    public AK.Wwise.Switch UnderFootMaterialSwitch =
null;
    public AK.Wwise.Bank FootStepBank = null;
    void InitializeSound() { FootStepBank.Load(); }
    void FinalizeSound() { FootStepBank.Unload(); }
    void PlayFootStepSound(float speed)
    {
        SpeedRTPC.SetValue(gameObject, speed);
        UnderFootMaterialSwitch.SetValue(gameObject)
        FootStepEvent.Post(gameObject);
    }
}
```

# **Unity Integration Extensions**

AkSoundEngine is now a partial class. Hooks have been added to allow users to extend the auto-registration of game objects or remove this functionality all together. The extensions, though recommended, can optionally not be installed via the Launcher.

#### **3D Busses**

To complement the addition of the 3D busses feature in Wwise, the following modifications have been made. There is no longer a restriction on the number of listeners, and the concept of default listeners has been added to represent the listeners that all game objects are initially set up to be listened by. AkGameObj has been modified to hold a list of AkAudioListeners. By default, an AkGameObj will emit its sound to the default listeners, and an AkAudioListener is a default listener.



## **Wwise Unity Integration » Release Notes**

### **Important Migration Notes 2017.1**

The following breaking changes have been made to the AkCallbackManager.

 AkCallbackManager.AudioInterruptionCallback() has a new signature:

```
public delegate AKRESULT AudioInterruptionCallba
ck(bool in_bEnterInterruption, object in_Cookie)
```

AkCallbackManager.BGMCallback() has a new signature:

```
public delegate AKRESULT BGMCallback(bool in_b0t
herAudioPlaying, object in_Cookie);
```

• AkCallbackManager.EventCallback() has a new signature:

```
public delegate void EventCallback(object in_coo
kie, AkCallbackType in_type, AkCallbackInfo in_i
nfo);
```

- The callback classes are no longer nested within AkcallbackManager and have a hierarchy with many inheriting from AkCallbackInfo. Wherever possible, the associations and members of these classes mimic the syntax of their C++ counterparts.
- The gameObjID member of the callback classes is now of type ulong instead of Intetr.
- The class AkCallbackManager.AkMidiEventCallbackInfo is now AkMIDIEventCallbackInfo. (Please note the capitalization of "MIDI".)
- The class AkCallbackManager. AkMusicSyncCallbackInfoBase has been removed. Use AkMusicSyncCallbackInfo instead.
- All of the members of AkCallbackManager.AkMusicSyncCallbackInfo.segmentInfo have been exposed as members of AkMusicSyncCallbackInfo.segmentInfo\_\*.

### **Wwise Unity Integration**

# Installing or Upgrading the Integration in a Unity Project

The wwiseUnityIntegration UnityPackage is comprehensive, containing Unity components, Editor tools, and documentation to make Wwise work in a Unity project.

Consult the following pages for the various installation and upgrade procedure information you may need.

- Prerequisites
- Installing a Wwise Integration Package for the First Time
- Changes Made to Your Unity Project
- Integrating Sounds in Your Game
- Where Should My SoundBanks Be Located?
- Adding Plug-ins
- Updating/Upgrading the Wwise Integration Package to a Newer Version

Wwise Unity Integration » Installing or Upgrading the Integration in a Unity Project

# **Prerequisites**

Check the Wwise requirements for each of your platforms here: **Platform-Specific Information** 

The integration into Unity requires the following software:

Platform	Required components
All platforms	Unity 5 (Personal or Pro)
iOS	Xcode 7.2 or above with compatible iOS SDK
Linux	libSDL2 (See <b>Build for Linux</b> )
	DirectX End-User runtime The x64 Visual Studio 2013 redistributable
Windows 32-bit Debug	The x86 Visual Studio 2013 redistributable



Wwise Unity Integration » Installing or Upgrading the Integration in a Unity Project

# Installing a Wwise Integration Package for the First Time

The integration procedure is handled by the Wwise Launcher. Please use the Wwise Launcher to integrate Wwise in a Unity project.

#### Warning: Recommendations Before you begin:

- Make sure to back up your Unity project, either by relying on source control or on the backup functionality in the Launcher.
- Before you install the Wwise integration, you need to make sure your Unity project is in a state where it can compile successfully without any errors. (Warnings are OK.)
- If the Unity project is under source control, make sure to check out the whole Unity project folder before starting to integrate.
   Otherwise, the integration will fail because it won't be able to modify the files.



### **Unity Integration Extensions**

Users hooks, in the form of partial methods, have been added to allow users to extend the auto-registration of game objects or remove this functionality all together. The default extensions, though recommended, can optionally be omitted via the Launcher.



Wwise Unity Integration » Installing or Upgrading the Integration in a Unity Project

### **Changes Made to Your Unity Project**

During the installation process, the following changes were made to your Unity project:

- The Disable Audio Option in Edit > Project Settings > Audio Was selected. This disables the built-in audio system.
- If the "Create WwiseGlobal GameObject" check box has been checked, a game object named "Wwise Global" was created in the current scene. This object has the global scripts AkInitializer and AkTerminator to initialize Wwise. There are a few customizable options in the initializer script. The WwiseGlobal object is also given AkEditorEventPlayer, AkAudioListener and AkGameObj scripts which are used for previewing **AkEvent** objects in edit mode. A Rigidbody component is added such that interactions between AkEnvironment, AkRoom and AkGameObj can also be previewed.
- The script Execution Order has been set for sensitive Wwise SCripts (AkInitializer, AkGameObj, AkAudioListener, AkBank, and AkTerminator).
- If the "Add Ak Listener to Main Camera" check box has been checked, the Audio Listener component has been removed from the Main Camera object, and the default AkAudioListener.cs Script has been set on the Main Camera object.
- The Play in Background option has been set to allow you to connect the Wwise Profiler

Wwise Unity Integration » Installing or Upgrading the Integration in a Unity Project

### **Integrating Sounds in Your Game**

At this point, you should be able to try playing sounds from Wwise in your game. If you haven't already set up your Wwise project with a few sounds, Events, and SoundBanks, you will need to start Wwise and create the following:

- 1. Import a WAV file. Drag and drop one from a Windows Explorer window to the Project Explorer's Audio tab. This will create a Sound.
- 2. Create an Event. Right-click on the Sound and select New Event. Name it appropriately.
- 3. Create a SoundBank. Go to the SoundBanks layout (F7). Click New.
- 4. Put your event in the SoundBank. To do so, drag and drop the Event from the Project Explorer to the SoundBank.
- 5. Click Generate SoundBanks. You now have something to play.

Now, to integrate a sound in your Unity project:

- Open the Wwise Picker window (Window > Wwise Picker), it should show you your Events and your SoundBanks.
- 2. Drag and drop your SoundBank on the object in your scene.
- 3. Drag and drop your Event on an object in your scene.
- 4. Play your game! You should hear a sound on start-up.

You didn't hear anything? Read more on the **Troubleshooting** page.

### **Integrate more sounds**

The current integration of Wwise into Unity provides a few Unity components that will cover some of the regular usage cases.

- Use **Akambient** to place scene-bound sound emitters.
- Use AkBank to load a SoundBank.
- Attach AkEnvironment to colliders to set up reverbs in a specific zone.
- Read more about the other components.

However, it is important to have the support of a programmer to write scripts to implement the more complex behaviors you want in your game. You can find more information on the Wwise API in these two sections:

- Integrate Wwise Elements into Your Game
- AK::SoundEngine Namespace Reference

Wwise Unity Integration » Installing or Upgrading the Integration in a Unity Project

### Where Should My SoundBanks Be Located?

When running in the Unity Editor, the SoundBanks are taken directly from the Wwise project's GeneratedSoundBanks folder. This helps in faster prototyping.

When packaging your game, Unity does not have a way to discriminate between assets that are built specifically for one platform or another. It packages all the SoundBanks located under .... Wwise, on the other hand, generates different SoundBanks for each platform in subfolders. To avoid packaging the SoundBanks for all platforms with your game, you should enable the automatic generation of SoundBanks or use a BuildPlayerPipeline script. Read more in this section: Build your Unity **Game for a Target Platform.** 



Wwise Unity Integration » Installing or Upgrading the Integration in a Unity Project

### **Adding Plug-ins**

All effect plug-ins are available as dynamic libraries. The libraries will be automatically included in your Unity project depending on their usage in the Wwise project. The list of effect plugins your project uses is compiled when the SoundBanks are generated. To add a previously unused plug-in to your project, follow these steps:

- Use the effect plug-in in your Wwise project.
- Generate your SoundBanks for the platform you are working on.
- Switch to your Unity project.
- Copy your new SoundBanks to the expected *StreamingAssets* folder. (If Play in Editor, no need to copy.)
- Compile your game (or Play in Editor).

**Note:** The following source and effect plug-ins are included as part of the base Unity integration:

- Wwise Compressor
- Wwise Delay
- Wwise Expander
- Wwise Gain
- Wwise Matrix Reverb
- Wwise Meter
- Wwise Parametric EQ
- Wwise Peak Limiter
- Wwise RoomVerb
- Wwise Silence
- Wwise Sine
- Wwise Tone Generator

If you want to create a brand new effect plug-in for Wwise (and then Unity), please read section Creating New Plug-ins

Licensed plug-ins are not part of the default Wwise license (McDSP, iZotope, Auro, SoundSeed, Crankcase REV, Convolution Reverb, Motion). You can try them in the authoring application, but not in-game. If

you do not have a valid license, please contact us. Evaluation licenses are always free.

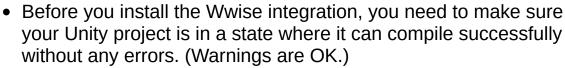


Wwise Unity Integration » Installing or Upgrading the Integration in a Unity Project

## **Updating/Upgrading the Wwise Integration Package to a Newer Version**

#### **Warning: Recommendations**

- The Unity migration operation is not reversible. Make sure to back up your Unity project before you proceed.
- If the Unity project is under source control, make sure to check out the whole Unity project folder before starting the migration. Otherwise, the migration will fail because it won't be able to modify the files successfully.



• If you have modified the C++ Wwise SDK, of the C++ AkSoundEngine plug-in source code, please follow the instructions listed in **Updating/Upgrading the Wwise Integration Package to a Newer Version When You Have Modified the Plug-in's C++ Code** 

This installation procedure assumes that you already have a Unity project containing a previous Wwise Integration package. For new installations, please refer to the Installing a Wwise Integration Package for the First Time page. Unlike other Unity plug-ins, you need to leave the previous Wwise Integration package installed in order to upgrade it to a newer version. The Wwise Launcher will install the new features and bug fixes, and then migrate the previous Wwise Integration to the newer version.



**Warning:** Do not delete the previous Wwise Integration before installing the new one as it will result in losing all the hard work you've done integrating Wwise audio in your game.

The upgrade is done in a 2-step process. First you need to update your Wwise project to the new version you wish to use and then you need to do the same with your Unity project.



## **Updating your Wwise project to the new version**

- 1. Launch the Wwise version that matches the new Wwise Integration package(s) you intend on using in Unity.
- 2. Open your Wwise Project.
- 3. Answer 'Yes' when prompted to update your Wwise project to the current version.
- 4. Once completed, make sure to regenerate your Wwise SoundBanks to be ready to carry on with your work after updating the Unity project to the new Wwise version.

# **Updating your Unity project to the new Wwise** version

Updating the Unity project to the new Wwise version is handled by the Wwise Launcher.



### **Wwise Unity Integration**

# **Uninstalling the Wwise Integration Package**

To uninstall the Wwise Integration Package, you need to manually delete the folder Assets/Wwise from your Unity project. This does not remove the Wwise components from your game objects, nor will it change any other settings.



### **Wwise Unity Integration**

### **Using the Wwise Unity Integration**

This integration provides a few components that can be used without code directly in a scene for the most frequent usage scenarios:

AkAmbient Use this component to attach a Wwise Event to any object in a scene. The sound can be started at various moments, dependent on the selected Unity trigger. This component is more useful for ambient sounds (sounds related to scene-bound objects) but could also be used for other purposes. Since AkAmbient has AkEvent as its base class, it features the play/stop, play multiple, stop multiple and stop all buttons for previewing the associated Wwise event. . See How to use AkAmbient with the inspector.

#### AkAudioListener

Add this script on the game object that represent a listener. This is normally added to the Camera object or the Player object, but can be added to any game object when implementing 3D busses. isDefaultListener determines whether the game object will be considered a default listener - a listener that automatically listens to all game objects that do not have listeners attached to their AkGameObjListenerList's.

#### AkBank

Loads and unloads a SoundBank at a specified moment. Vorbis sounds can be decompressed at a specified moment using the decode compressed data option. In that case, the SoundBank will be prepared.

#### AkEmitterObstructionOcclusion

Obstructs/Occludes the emitter of the current game object from its listeners if at least one object is between them.

#### AkEnvironment

Use this component to define a reverb zone. This needs to be added to a collider object to work properly. How to use AkEvironment and AkEvironmentPortal with the inspector (Reverb Zones).

#### AkEnvironmentPortal

Use this component to define an area that straddles two different AkEnvironments zones and allow mixing between both zones. How to use AkEvironment and AkEvironmentPortal with the inspector (Reverb Zones).

#### AkEvent

Helper class that knows a Wwise Event and when to trigger it in Unity. See **How to use AkAmbient with the inspector**.

#### AkGameObj

This component represents a sound object in your scene tracking its position and other game syncs such as Switches, RTPC and environment values. You can add this to any object that will emit sound, and it will be added to any object that an **AkAudioListener** is attached to. Note that if it is not present, Wwise will add it automatically, with the default values, to any Unity Game Object that is passed to Wwise.

#### AkRoom

An **AkRoom** is an enclosed environment that can only communicate to the outside/other rooms with AkRoomPortals.

#### AkRoomPortal

An AkRoomPortal can connect two AkRoom components together.

#### AkRoomPortalObstruction

Obstructs/Occludes the spatial audio portal of the current game object from the spatial audio listener if at least one object is between them.

#### AkSpatialAudioEmitter

Add this script on the GameObject which represents an emitter that uses the Spatial Audio API.

#### • AkSpatialAudioListener

Add this script on the game object that represent a listener. This is normally added to the Camera object or the Player object, but can be added to any game object when implementing 3D busses. isDefaultListener determines whether the game object will be considered a default listener - a listener that automatically listens to all game objects that do not have listeners attached to their AkGameObjListenerList's.

#### AkState

This will call AksoundEngine.SetState() whenever the selected Unity event is triggered. For example this component could be set on a Unity collider to trigger when an object enters it.

#### AkSurfaceReflector

This component will convert the triangles of the GameObject's geometry into sound reflective surfaces.

#### AkSwitch

This will call AksoundEngine.SetSwitch() whenever the selected Unity event is triggered. For example this component could be set on a Unity collider to trigger when an object enters it.

#### See also:

**Wwise Picker** 

This integration also provides a few classes that can be used, with minimal code, for most remaining usage scenarios:

#### AK.Wwise.AuxBus

This type represents an auxiliary send in the Master-Mixer Hierarchy.

AK.Wwise.Bank

This type can be used to load/unload SoundBanks.

#### AK.Wwise.CallbackFlags

This type represents the values of the flags used when posting an **Event** with a callback.

AK.Wwise.Event

This type can be used to post Events to the sound engine.

AK.Wwise.RTPC

This type can be used to set game parameter values to the sound engine.

AK.Wwise.State

This type can be used to set Wwise States.

AK.Wwise.Switch

This type can be used to set **Switch** values on gameobjects.

• AK.Wwise.Trigger

This type can be used to post triggers to the sound engine.

#### See also:

**Wwise Types** 

# How to add a Wwise sound to a game object

There are four ways to add sounds to your game:

- Using the Wwise Picker. This is the simplest way to add a sound to an object. Drag an Event from the Wwise Picker window to an object in the Unity Viewer or the Inspector. This automatically creates an AkAmbient component on the target Game Object.
- Using the Add Component menu. Add an AkAmbient or an AkEvent component to any Unity Game Object.
- Using Wwise Types. Call AK. Wwise. Event. Post() at any time from a C# script.
- **Using scripts.** Call AkSoundEngine.PostEvent() at any time from a C# script.

# **How to use AkAmbient with the inspector**

#### AkAmbient:

#### Trigger On:

Provides a list of Unity events that can trigger your event. You are not limited to those events. You can trigger an event at any time by calling AkSoundEngine.PostEvent anywhere in your code. You can also code your own triggers, so they appear in the list for your co-workers. See **Adding New Triggers for Wwise Events**.

#### • Event Name:

Specifies the name of the current event. To select an event, click on the current event's name to open the event picker window. Then, you can either click on an event and click on the *Ok* button or double click an event to select it. You can also drag an event from the Wwise picker and drop it on the current event's name to select a new one.

#### Action On Event:

Enables users to override some event parameters defined in Wwise directly from Unity. This allows the reuse of existing events insted of creating new ones.

- Action On Event Type:
   Overrides the event type.
- Curve Interpolation:
   Overrides the interpolation curve.
- Fade Time: Overrides the sound's fade time.

#### Use Callback:

Provides an easy way to make a game object react to an event callback.

- Game Object:
  - The game object that will receive the callback. To select a game object, drag it from the hierarchy and drop it in the game object field.
- Callback Function:
   This is the function that will get called by Game Object when the callback happens. To select a function, type its

name in the Callback Function text field.

For this to work, *Game Object* must define *Callback Function* in one of its components.

The function's definition must be **void FunctionName(AkEventCallbackMsg in\_info)**.

Callback Flags:

Select a flag which specifies when *Callback Function* will be called. More than one flag can be selected at the same time. See the AkCallbackType enumeration in the Wwise SDK documentation for more information about each flag.

#### Play / Stop:

Can be used to preview the Wwise Event when in Edit mode.

Stop All:

Stops all currently playing Wwise events.

Position Type:

Defines the way the event's position will be sent to the audio engine.

Simple\_Mode:

The event's position will be the same as the game object to which it's attached.

■ Large\_Mode:

The event can have multiple positions that are defined by a set of points. You can add a point by clicking the Add button in the *AkAmbient Tool Bar*. You can then move each point by dragging it around. This mode is useful when a sound is coming from multiple positions at the same time. The sound of water in the middle of the ocean is a good example.

MultiPosition\_Mode:

This mode enables us to have only one instance of a sound for all instances of **AkAmbient** using the same event **in order to save memory**.

All **AkAmbient** instances that are using this mode and that have the same event will automatically get detected and the same sound instance will be used for all of them insted of loading the same sound multiple times.

Note that all **AkAmbient** instances in this mode and with the same event will have the same trigger (see *Trigger On* in **AkEvent**). So, changing the trigger of one **AkAmbient** will automatically change the trigger of all the others with the same event.

#### ■ Show Attenuation Sphere:

Shows a sphere that defines the space where the sound played by an event can be heard.

For this to work you need to enable *Max Attenuation* in the SoundBank settings in your Wwise project (Project>Project Settings->Soundbanks->Max attenuation)

- Dont\_Show: No attenuation sphere is shown.
- Current\_Event\_Only:

  Shows the attenuation spheres for all the sounds that would be played after a call to

  AkSoundEngine.PostEvent while in the current mode.

  If in Simple\_Mode, then only the attenuation sphere of the sound coming from the game object is shown. if in Large\_Mode, then an attenuation sphere is shown for each point.
  - If in *MultiPosition\_Mode*, then an attenuation sphere is shown for every other **AkAmbient** in *MultiPosition\_Mode* with the same event.
- All\_Events:

Shows the attenuation sphere of all **AkAmbient** instances in the scene.

# **Using Wwise with Unity Timeline**

For Unity's Timeline feature, there are custom Wwise tracks for triggering Wwise events and setting Wwise RTPC values.

### See also:

**Wwise Timeline Integration** 

# How to use AkEvironment and AkEvironmentPortal with the inspector (Reverb Zones)

In Wwise, Reverb Zones are called Environment Or Auxiliary Sends. Reverb Zones are not limited to being reverb effects and are defined in the Wwise project.

An **AkEnvironment** component embodies a very simple environment zone. You can attach an **AkEnvironment** to any type of collider. To add an **AkEnvironment** to your scene:

- Using the Wwise Picker. This is the simplest way to add an AkEnvironment. Drag an AuxBus from the Wwise Picker window to an object in the Unity Viewer or the Inspector. This will automatically create an AkEnvironment component on the target Game Object.
- Using the "Add Component" menu. Add an AkEnvironment component to any Unity Game Object. Select the desired environment from the selector in the inspector.
- Using scripts. You can call
   AkSoundEngine.SetGameObjectAuxSendValues() at any time from a C#
   script.

We also have portals which can be used to combine the effects of two environments. The contribution of each of the two environments is relative to their distance from the game object.

This is useful if a game object is standing between two rooms or in a tunnel connecting two environments.

 To add an environment portal to your project, go to GameObject->Wwise->Environment Portal in Unity's menu bar.

To use environments and environment portals, you need a game object with an **AkGameObj** component that is environment-aware. **AkEnvironmentPortal** objects will automatically detect **AkEnvironment** objects that overlap it. The overlapping environments will appear in the two select-lists in the portal's inspector. If too many environments overlap

the portal, you can select which ones the portal will mix together.

In Wwise, only 4 environments can be active at the same time. Those 4 environments are selected as follows:

- The environments that are connected to a portal and that have the highest priority are selected until we reach 4 environments or until there are no more environments connected to a portal.
- If we still don't have 4 selected environments, we select the environments that are not connected to a portal as follows:
  - Environments with the highest priority will be selected until we reach 4 environments (if the *Default* and *Exclude Others* flags are not set).
  - A Default environment will be selected only if no other environment is selected.
  - If your game object is inside an environment with the *Exclude Others* flag, then it will be selected and all other environments will get discarded.

#### AkEnvironment component:

Only 4 environments can be active at the same time.

#### • Priority:

Defines the priority of an environment.

A smaller number has a higher priority.

If a game object is inside more than 4 environments, only the 4 environments with the highest priority will be active (if the *Default* and *Exclude Others* flags are not set).

#### Default:

A default environment will be active only if it's the only environment containing your game object.

If your game object is inside more than one default environment, then only the one with the highest priority will be active.

#### Exclude Others:

An environment with this flag can't be overlapped by other environments.

If your game object is inside an environment with the *Exclude Others* flag, then all other environments will get discarded. If your game object is inside more than one environment with the *Exclude Others* flag, only the one with the highest priority

will be active.

#### AuxBus Name:

Specifies the name of the current AuxBus. To select an AuxBus, click on the current AuxBus's name to open the AuxBus picker window. Then, you can either click on an AuxBus and click on the *Ok* button or double-click an AuxBus to select it. You can also drag an AuxBus from the Wwise picker and drop it on the current AuxBus's name to select a new one.

#### AkEnvironmentPortal component:

You can create an environment portal in Unity by going to GameObject->Wwise->Environment Portal.

You can place an environment portal between two environments to combine their effects while your game object is inside the portal. The portal must intersect with both environments for this to work. The contribution of each of the two environments is relative to their distance from the game object. The closer the game object is from an environment, the more that environment will contribute towards the final effect.

#### • Environment #1:

The portal will automatically detect all environments that intersect the portal. Of those environments, the ones that are placed on the negative side of the portal (opposite to the direction of the chosen axis) will be available in the drop down menu. The reason the environments are sorted this way is to reduce the computation needed to determine the contribution of each environment at runtime.

#### Environment #2:

The portal will automatically detect all environments that intersect the portal. Of those environments, the ones that are placed on the positive side of the portal (same direction as the chosen axis) will be available in the menu. The reason the environments are sorted this way is to reduce the computation needed to determine the contribution of each environment at runtime.

#### Axis:

The axis is used to find the contribution of each environment. For example, if the z axis is chosen, then moving along the x axis won't have any effect on the contribution of each

environment. Only movement on the z axis will have an effect on their contribution.

Note that the axis is in object space. So, rotating the portal will also rotate the axis.

#### See also:

- Integrating Environments and Game-defined Auxiliary Sends
- AK::SoundEngine::SetGameObjectAuxSendValues

# **Using C# code to control the sound engine**

Most Wwise SDK functions are available in Unity through the AksoundEngine class. Think of it as the replacement of C++ namespaces AK::SoundEngine, AK::MusicEngine, and so on. See API Changes and Limitations for changes made in the API binding compared to the original SDK. For more complex situations, you'll need to call Wwise functions from code. In the API, the GameObjectID in all functions is replaced by the Unity flavor of the GameObject. At runtime, an AkGameObj component is automatically added to this GameObject, unless you have already manually added it before.

## Using numeric IDs instead of strings for Events and Banks.

The native Wwise API allows you to use strings or IDs to trigger events and other named objects in the Wwise project. You can still do this in the C# world by converting the file wwise\_IDs.h to wwise\_IDs.cs. Click **Assets** > **Wwise** > **Convert Wwise SoundBank IDs**. You need to have Python installed to make this work.

## Sending MIDI to Wwise.

MIDI can be sent to Wwise by filling the Akmidipost members of Akmidipostarray class and calling any of the following methods:

- AkMIDIPostArray.PostOnEvent()
- AkSoundEngine.PostMIDIOnEvent()
- AK.Wwise.Event.PostMIDI()

The following is a basic script that sends MIDI messages to the sound engine:

```
public class MyMIDIBehaviour : UnityEngine.MonoBehav
iour
{
   public AK.Wwise.Event SynthEvent;
```

```
private void Start()
        AkMIDIPostArray MIDIPostArrayBuffer = new Ak
MIDIPostArray(6);
        AkMIDIPost midiEvent = new AkMIDIPost();
        midiEvent.byType = AkMIDIEventTypes.NOTE_ON;
        midiEvent.byChan = 0;
        midiEvent.byOnOffNote = 56;
        midiEvent.byVelocity = 127;
        midiEvent.uOffset = 0;
        MIDIPostArrayBuffer[0] = midiEvent;
        midiEvent.byOnOffNote = 60;
        MIDIPostArrayBuffer[1] = midiEvent;
        midiEvent.byOnOffNote = 64;
        MIDIPostArrayBuffer[2] = midiEvent;
        midiEvent.byType = AkMIDIEventTypes.NOTE_OFF
        midiEvent.byOnOffNote = 56;
        midiEvent.byVelocity = 0;
        midiEvent.uOffset = 48000 * 8;
        MIDIPostArrayBuffer[3] = midiEvent;
        midiEvent.byOnOffNote = 60;
        MIDIPostArrayBuffer[4] = midiEvent;
        midiEvent.byOnOffNote = 64;
        MIDIPostArrayBuffer[5] = midiEvent;
        SynthEvent.PostMIDI(gameObject, MIDIPostArra
yBuffer);
    }
}
```

## Using the Audio Input Source Plug-in in Unity.

The audio input source plug-in can be used via C# scripting. See Audio Input Source Plug-in from the Wwise SDK documentation.

The following is a basic script that sends a test tone to the audio input source plug-in:

```
public class MyAudioInputBehaviour : UnityEngine.Mon
oBehaviour
{
    public AK.Wwise.Event AudioInputEvent;
    public uint SampleRate = 48000;
    public uint NumberOfChannels = 1;
    public uint SampleIndex = 0;
    public uint Frequency = 880;
    private bool IsPlaying = true;
    bool AudioSamplesDelegate(uint playingID, uint c
hannelIndex, float[] samples)
    {
        for (uint i = 0; i < samples.Length; ++i)
            samples[i] = UnityEngine.Mathf.Sin(Frequ
ency * 2 * UnityEngine.Mathf.PI * (i + SampleIndex)
/ SampleRate);
        if (channelIndex == NumberOfChannels - 1)
            SampleIndex = (uint)(SampleIndex + sampl
es.Length) % SampleRate;
        // Return false to indicate that there is no
more data to provide. This will also stop the assoc
iated event.
        return IsPlaying;
    }
    void AudioFormatDelegate(uint playingID, AkAudio
```

```
Format audioFormat)
    {
        audioFormat.channelConfig.uNumChannels = Num
berOfChannels;
        audioFormat.uSampleRate = SampleRate;
    }
    private void Start()
    {
        AkAudioInputManager.PostAudioInputEvent(Audi
oInputEvent, gameObject, AudioSamplesDelegate, Audio
FormatDelegate);
    }
    // This method can be called by other scripts to
 stop the callback
    public void StopSound()
    {
        IsPlaying = false;
    }
    private void OnDestroy()
    {
        AudioInputEvent.Stop(gameObject);
    }
}
```

## **Apply Custom Positioning in Unity**

By default, the AkGameObj component is attached to a specific Unity gameObject and uses its transform (with an optional offset) for full positioning. This is usually adequate for many games, such as first-person shooters. However, games with custom camera angles, such as many third-person games, may find it difficult to accommodate the two aspects of positioning (distance attenuation and spatialization) by simply attaching the audio listener to one game object, such as the main camera in Unity. Other games may want players to experience other custom

positioning.

To this end, the AkGameObj component class provides overridable positioning to Unity users. Through the three virtual methods GetPosition(), GetForward(), and GetUpward(), users can derive a subclass from AkGameObj and use that subclass component to customize any number of Unity gameObjects' positioning.

Here is a simple example of how to use a custom component to override the default <code>AkAudioListener</code> behavior. With a third-person project integrated with Wwise, remove the existing <code>AkAudioListener</code> and its associated <code>AkGameObj</code>. Then attach the following script to the MainCamera object, attach <code>AkAudioListener</code>, and finally specify the target Unity <code>gameObject</code> (such as the player avatar) that the audio listener's position will follow. After this, the distance attenuation of all the emitters will rely on the selected target Unity <code>gameObject's</code> position as the listener position (an on-screen distance listener), while the orientation of all the emitters is still based on the main camera orientation as the listener orientation (an off-screen orientation listener).

```
public class AkGameObj3rdPersonCam : AkGameObj
    public Transform target;
                                        // The posit
ion that this camera will be following. User can spe
cify this to the player character's Unity gameObject
in the Inspector.
    // Sets the camera position to the player's posi
tion to handle distance attenuation.
    public override Vector3 GetPosition ()
        return target.GetComponent<AkGameObj> ().Get
Position ();
    }
#endif // #if ! (UNITY_DASHBOARD_WIDGET || UNITY_WEB
PLAYER || UNITY WII || UNITY WIIU || UNITY NACL || U
NITY FLASH || UNITY BLACKBERRY) // Disable under uns
upported platforms.
```

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3

Main Page Classes

**Wwise Unity Integration » Using the Wwise Unity Integration** 

## **Wwise Picker**

The Wwise Picker window shows the available objects (such as Events and Switches) from your Wwise project, using the same structure. The project displayed in the Wwise Picker view is the one that is entered in the Wwise Settings window (menu **Edit > Wwise Settings...**), that was normally specified when the Wwise plug-in was installed in Unity.

The available objects are updated in the Wwise Picker view when the Wwise project is saved. So, if you don't see objects that you created, it is probably because your project wasn't saved. You can force a refresh of the wwise Picker view by clicking Refresh Project. Note that this also refreshes all Wwise-related data, including data found in the XML metadata files that are created when generating the SoundBanks.

Use the filter box near the top of the Wwise Picker to search for a specific item. Simply write the name of the item you're looking for and the Picker will automatically filter the items that don't match your search. You might need to expand some items if the one you're looking for has a closed parent.

You can expand and close items by clicking on the + and - buttons or by double clicking the item's name.

# **Generating SoundBanks**

Click Generate SoundBanks to generate all SoundBanks for all platforms and languages. The generated SoundBanks will appear under the Wwise Picker's SoundBanks folder, the same as if they were generated from within Wwise. Additionally, the Unity status bar will display one of a possible two messages: WwiseUnity: SoundBanks generation successful, if there were no errors or warnings; WwiseUnity: SoundBanks generation has warning(s), if there were warnings; and wwiseUnity: SoundBanks generation error, if there were errors. Click on the message to move focus to the console tab, which displays information from the SoundBank generation process.



**Note:** It is necessary to specify the **Wwise Windows Installation Path** (or **Wwise Application** on the Mac) within the Wwise Settings dialog. Unity needs to use the Wwise path when clicking Generate SoundBanks in Unity.

# Adding audio to your scenes with the Picker

To add any object from the Wwise Picker to your scene, drag the object onto an existing Game Object or into the Inspector window. Refer to each component's documentation for further details about the components:

- Dragging and dropping an Event creates an AkAmbient.
- Dragging and dropping a SoundBank creates an AkBank.
- Dragging and dropping a Switch Value creates an AkSwitch.
- Dragging and dropping a State Value creates an AkState.
- Dragging and dropping an Aux Bus creates an **AkEnvironment**.

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3

Main Page Classes

**Wwise Unity Integration » Using the Wwise Unity Integration** 

# **Wwise Types**

Wwise Types are a collection of classes that represent core objects within Wwise such as SoundBanks, Events, game parameters, States, and Switches. Each of these classes has an associated property drawer that allows users to easily select their required object. Wwise Types also contain a support class that represents the callback flags that can be sent to the sound engine when posting Events.

Here is an example script that uses a few Wwise Types:

```
public class WwiseTypesExample : UnityEngine.MonoBeh
aviour
{
    public AK.Wwise.Bank MyBank = null;
    public AK.Wwise.Event MyEvent = null;
    public AK.Wwise.RTPC MyRTPC = null;
    public void Awake()
    {
        MyBank.Load();
    public void Start()
    {
        MyEvent.Post(gameObject);
    private float CalculateMyValue()
    {
        return (float)System.Math.Sin(System.Math.PI
 * UnityEngine.Time.timeSinceLevelLoad);
    public void Update()
    {
```

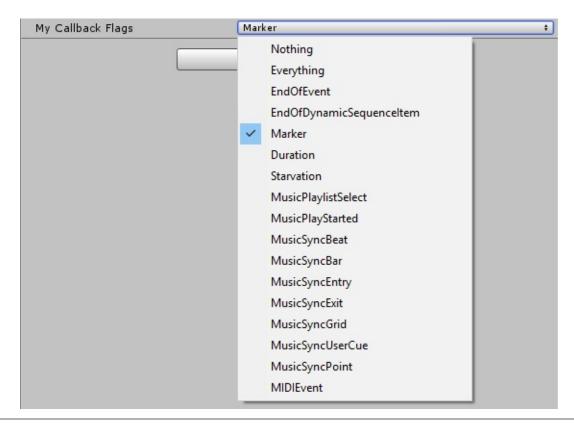
```
MyRTPC.SetValue(gameObject, CalculateMyValue
());
    }
}
```

The above script automatically generates the following inspector:



The following modifications can be made to the wwiseTypesExample class to use marker callbacks:

```
public AK.Wwise.CallbackFlags MyCallbackFlags = null
public void Start()
{
    MyEvent.Post(gameObject, MyCallbackFlags, EventC
allback);
}
private void EventCallback(object cookie, AkCallback
Type type, AkCallbackInfo info)
{
    if (type == AkCallbackType.AK_Marker)
    {
        var markerInfo = info as AkMarkerCallbackInf
0;
        if (markerInfo != null)
        {
            // ...
        }
    }
}
```



Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3



Main Page Classes

**Wwise Unity Integration » Using the Wwise Unity Integration** 

# **Adding New Triggers for Wwise Events**

In most of Wwise components for Unity, there is a "Trigger On" property from which you can select which Unity notification/event will trigger the Wwise component (Event, Switch, State, etc). This list contain only a subset of what is possible in Unity. However, the triggering system is extensible, without modifying the Integration code itself.

To add a new trigger, you simply have to write a C# class that derives from AkTriggerBase. All classes derived from AkTriggerBase will be found through reflection and added to the "Trigger On" drop list. This class contains a delegate, triggerDelegate(GameObject in\_target), that you must call when the conditions for your trigger occur. The "target" object is optional, and currently used to give collided objects when used with Colliders. This allow Wwise components to Post events on the "collidee" instead of the Collider.

While entirely optional, you could use this system to trigger audio components from your game code too. You can simply call GetComponent<YourTriggerClass>().triggerDelegate(GameObject in target) whenever your trigger condition occur from your code (or have a custom function).



Note: Currently the maximum number of derivative classes of AkTriggerBase is 32.

Here is an example, with a custom function:

```
public class TriggerOnGunHit : AkTriggerBase
{
    void Hit()
    {
        if(triggerDelegate != null)
        {
            triggerDelegate(null);
        }
    }
```

```
}
```

From your game code, you could have this code:

```
if (playerIsShot)
{
    GetComponent<TriggerOnGunHit>().Hit();
}
```

Any Wwise component linked to the TriggerOnGunHit trigger would then do its job. Note that in many simple situations, this is completely superfluous; you could also simply call the base Wwise SDK through AkSoundEngine.PostEvent("GunHit", gameObject) and let the sound designer handle the effect of this event in Wwise.

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by



Main Page Classes

**Wwise Unity Integration » Using the Wwise Unity Integration** 

# **Wwise Timeline Integration**

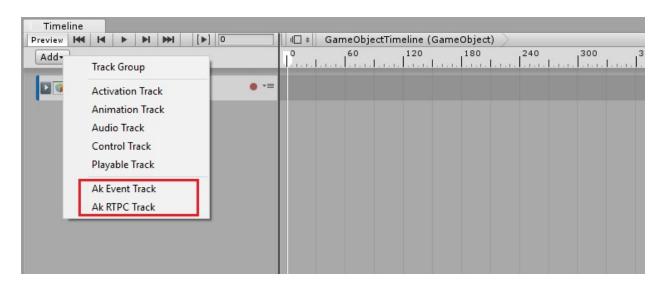
Timeline is Unity's cinematic editting tool. It can be used to create gameplay sequences, audio sequences, and other cinematic content.

For more information on Timeline refer to Timeline Docs.

For instructions on how to create a Timeline Asset and Timeline instance, refer to Timeline Usage.

## **Wwise Timeline Tracks**

The Wwise Unity Integration adds two tracks: the **AkRTPC** track and the **AkEvent** track. The **AkRTPC** track is used to set Game Parameter values, and the **AkEvent** track is used to post Wwise Events. **AkRTPC** tracks and **AkEvent** tracks can be added to a Timeline in Unity using the "Add" dropdown button.



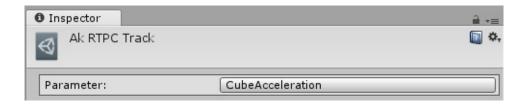
Adding AK tracks to Timeline

## Ak Track Object Bindings

**AkEvent** tracks and **AkRTPC** tracks have object bindings which can be set to any Game Object. If this is left as None, the Event messages or RTPC messages in the track will be triggered on the Game Object that owns the Timeline (this is the Game Object that has a PlayableDirector component, which is used to trigger the Timeline). If the track binding is set to a different Game Object, the messages will be triggered on that Game Object.

## The AkRTPC Parameter Property

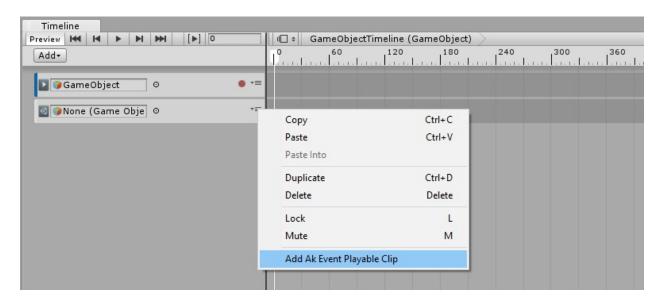
**AkRTPC** tracks have one property in their property inspector: **Parameter**. This is the Wwise RTPC parameter that will be set from the various clips in the track.



**RTPC Track Parameter Property** 

## Wwise Timeline Playable Clips

The **AkEvent** track and the **AkRTPC** tracks each hold corresponding clips: the **AkEventPlayable** clip and the **AkRTPCPlayable** clip. These can be added to a track using the track's menu.



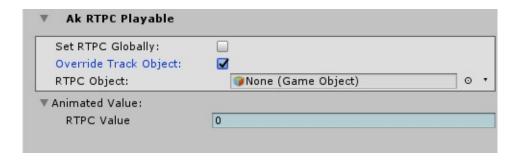
Adding AK clips to AK tracks

## **Timeline Clip Properties**

All clips in Timeline have timing properties that are displayed in the top portion of the inspector. For more information on these properties, refer to Clip Properties.

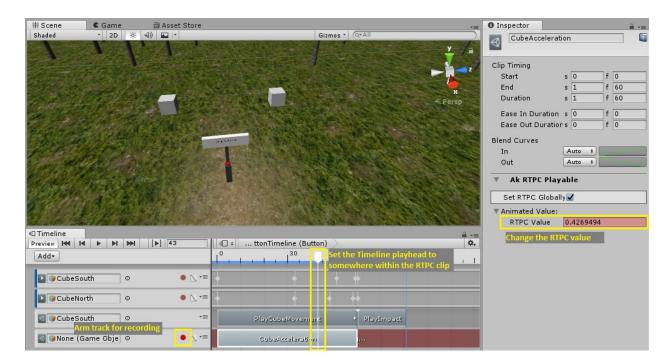
#### **AkRTPCPlayable Clip Properties**

The following properties are accessible from the inspector of an **AkRTPCPlayable** clip:



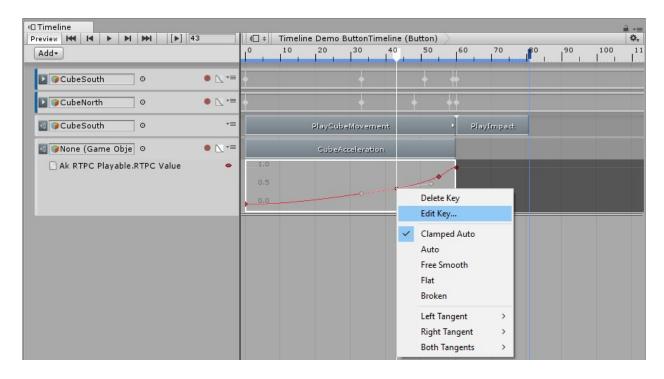
#### **AkRTPCPlayable Clip Properties**

- **Set RTPC Globally**: If this is checked, the RTPC will be set globally for all objects.
- Override Track Object: If this is checked, the RTPC will be set for the selected Game Object in the RTPC Object property (only available when Set RTPC Globally is unchecked).
- RTPC Object: The Game Object for which the RTPC will be updated. If this is set to None, the RTPC will be set on the track object (or the Timeline's owner object if the track object is set to None). This is only available when Override Track Object is checked.
- Animated Value: RTPC Value: This is the value that will be animated by the RTPC track. In order to begin animating this value you need to arm the track for recording, set the playhead to somewhere within the RTPCPlayable clip, and then edit the value from the property inspector.

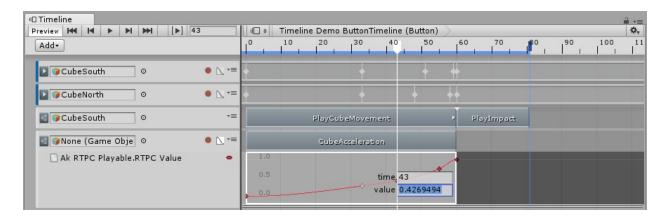


## **Animating RTPC Values from Timeline**

Following these steps will insert a keyframe into the RTPC track, below the RTPC clip. You can then use this Unity curve editor to add more keyframes. The keyframes have a context menu that can be used for more precise values.



#### **RTPC Keyframe Context Menu**



**Editting RTPC Keyframe Values** 

### AkEventPlayable Clip Properties

The following properties are accessible from the inspector of an **AkEventPlayable** clip:



## **AkEventPlayable Clip Properties**

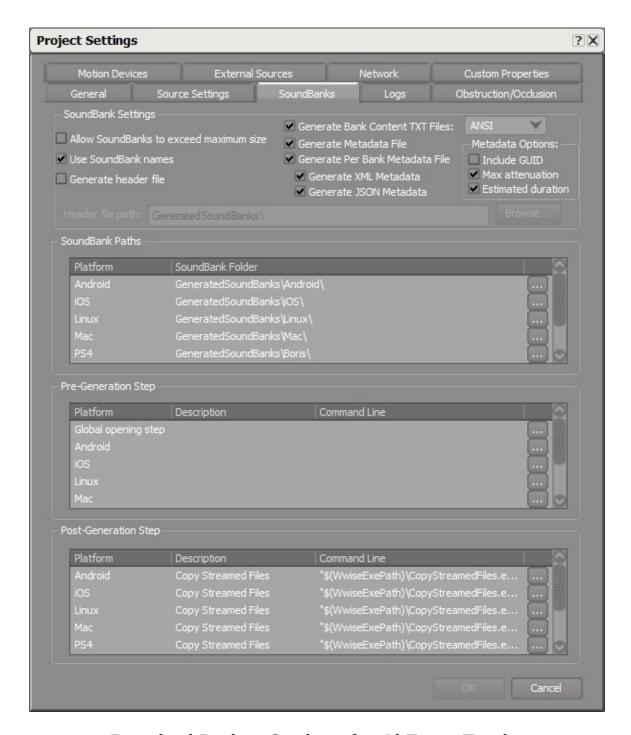
- Override Track Object: This can be checked in order to have the Event emitted from a specific Game Object, different to the track binding object. This will also override the Timeline owner object if the track binding is set to None (see Ak Track Object Bindings).
- Emitter Object Ref: The Game Object from which the Event should be emitted (only available when Override Track Object is checked).
- Event: The Wwise Event that will be triggered from this clip.

## **Known Issues & Limitations**

Currently scrubbing is only supported in Play mode in the Unity editor. This is due to an issue with Unity Playable callbacks in edit mode. Unity are aware of the issue: 953109. This means that "Play from anywhere" behavior is not supported in edit mode either. If you place the playhead somewhere between the beginning and end of a clip and play the timeline, while in edit mode, the event will not be triggered.

# **Wwise Project Setup**

To ensure that Event lengths are properly represented in the **AkEvent** Tracks, the Wwise project needs to be set up to estimate the duration of its audio Events and JSON metadata must be generated. To enable these project-wide settings, go to the Project Settings' SoundBanks tab. Enable the **Estimated duration** check box within the **Metadata Options** group box and enable the **Generate JSON Metadata** check box within the **SoundBank Settings** group box, as seen below.

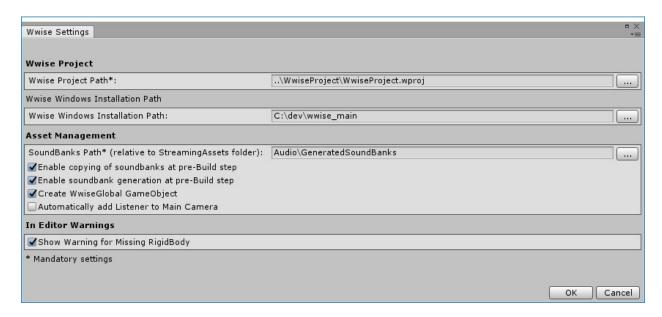


**Required Project Settings for AkEvent Tracks** 

#### **Wwise Unity Integration**

#### **Build your Unity Game for a Target Platform**

Building your Unity game for a target platform is now as simple as building your application using Unity's standard pipeline or a build script. Pre- and post-build steps that generate, copy and delete SoundBanks can be enabled/disabled within the **Edit** > **Wwise Settings...** 



#### **Wwise plug-in deployment**

The Unity build system automatically packages the Wwise plug-in with the game. By default, the **Profile** configuration of the plug-in is used. This is the most practical configuration for game development because it allows you to connect Wwise to your game for profiling. However, when you are ready to release your game, go to the **Assets > Wwise >** Activate Plugins menu to change your Wwise plug-in configuration to the **Release** option.



**Note: Debug** is only used for debugging specific issues with the sound engine, which is usually done with the assistance of Audiokinetic support.

#### See also:

- Platform-Specific Information
- Build for Android
- Build for iOS
- Build for Linux
- Using Wwise Custom Platforms in Unity

#### **StreamingAssets folder**

Unity normally creates a streamingAssets folder (manually create one if it does not exist in your Unity project) in the Unity project's Assets folder, as Shown here: <uNITY\_PROJECT\_ROOT>\Assets\StreamingAssets. Unity automatically deploys this folder's content to the target device. See Unity's StreamingAssets feature for details.

The default SoundBanks path, Audio\GeneratedSoundBanks, is found under the streamingAssets folder. At the end of this path, Unity adds a folder for the current platform, as it was named in your Wwise project's Platform Manager. This gives a full final path of <UNITY PROJECT\_ROOT>\Assets\StreamingAssets\Audio\GeneratedSoundBank <YourPlatform>.



Note: If necessary, you can modify AkInitializer::basePath to change the default SoundBanks path.

When running in the Unity editor, the SoundBanks path is always set to point to the Wwise project's GeneratedSoundBank path, which is defined in Wwise's Project Settings dialog. This way, when integrating sounds in your game in Windows or on a Mac, you can avoid bank deployment issues.

# **Deploying SoundBanks in single-platform** projects

If you are working on a single platform title, instead of physically moving SoundBank files into the StreamingAssets folder, the SoundBank output path in a Wwise project can be set to the desired folder under the **StreamingAssets folder** so that the SoundBanks can be generated directly into the desired location. Using this method, be sure to disable the pre-build steps related to generating and copying SoundBanks within the **Wwise Settings** window.

## Deploying SoundBanks in multi-platform projects

For productions that ship on multiple platforms, the pre-build steps can generate the SoundBanks and copy them over to the <code>streamingAssets</code> folder before the binary is built, and then delete the SoundBanks in a post-build step to ensure that only SoundBanks for the target platforms are deployed.

Final implementations should be decided by game developers to accommodate the potential need for more elaborate management.

## Handling multiple custom platforms for a reference platform

If, in your Wwise project, you have defined multiple versions of the same reference platform, such as an iPad and an iPhone platform targeting iOS, some more scripting will be needed. See **Using Wwise Custom Platforms in Unity** for more information.

#### **Exclude SoundBank metadata**

Wwise generates SoundBank metadata files (such as TXTs and XMLs). They contain information about the SoundBanks to be used by your other pipeline tools, but are useless to the game. It is not necessary to copy them into your final streamingAssets folder.

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3



**Wwise Unity Integration » Build your Unity Game for a Target Platform** 

#### **Build for Android**

SoundBanks for Android included in the APK can be directly accessed via the LoadBank() APIs. SoundBanks placed under the StreamingAssets folder are included in the Assets folder of the APK when building the Unity project. The Low-Level IO handles accessing SoundBanks from the APK transparently.

The Low-Level IO can also access SoundBank files in the Android file system, such as from an SD card. This is not the default behavior. If you want to access the native file system you need to call AkSoundEngine.AddBasePath(YourPath). This new path will take precedence over the default APK files. If a file has the same name in the external path and in the APK, the external path is used. This is useful to implement DLCs or patches to audio content after the APK is built.

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3



**Wwise Unity Integration » Build your Unity Game for a Target Platform** 

#### **Build for iOS**

To test your Unity application on an iOS device, you need to build and deploy your app to the device. When you execute Build or Build and Run in the Unity Editor, Unity generates an Xcode project based on the Unity scripts.

UNITY\_PROJECT\_ROOT/Assets/Plugins/iOS

Unity does not yet support thumb instructions; therefore, employing thumb instructions is disabled in the Xcode project.

### **Building for release**

The Wwise Integration library, libakSoundEngine.a, normally contains all debug symbols. This makes the library very big. Make sure you strip the debug symbols by specifying the **Stripping Level** option in the **Other Settings** section of the (iOS) Player Settings. Normally, the integration should add between one and two MB to your game.

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by



**Wwise Unity Integration » Build your Unity Game for a Target Platform** 

#### **Build for Linux**

To test your Unity application on a Linux machine, build and deploy your app to the machine. When executing Build or Build and Run in Unity Editor, Unity generates a folder containing the compiled game. Copy this folder to the Linux machine and double-click on the .x86 (for 32-bit builds) or .x86\_64 (for 64-bit builds) file in the build folder to launch the game.

**Note:** The Wwise Unity Integration for Linux needs libsdl2 installed on the machine to work. Without this, a "DLLNotFoundException" message is displayed when launching the game.

To install libsdl2 for **Ubuntu 12.04**, do the following:

- sudo apt-get install build-essential libasound2-dev
- wget http://www.libsdl.org/release/SDL2-2.0.3.tar.gz
- tar -zxvf SDL2-2.0.3.tar.gz
- cd SDL2-2.0.3
- ./configure
- make
- sudo make install
- sudo ldconfig



**Note:** The requirements may vary depending on the machine's configuration. Refer to libsdl.org for installation instructions on other setups.

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3

**Wwise Unity Integration » Build your Unity Game for a Target Platform** 

### **Using Wwise Custom Platforms in Unity**

By default, the Wwise Unity Integration assumes the platforms defined in the associated Wwise project all have the default name. If you wish to give your platforms a custom name or define multiple custom platforms for a base platform, you will need to create a C# script to handle this.



Note: This page assumes that the subfolder in which your SoundBanks are generated has the same name as your platform. If this is not the case, note that the string returned by the GetPlatformName method is actually used as the in-game SoundBank subfolder name.

### **Custom platforms**

The default platform resolving script can be found in the file UNITY\_PROJECT\_ROOT/Assets/Wwise/Deployment/Components/AkBasePathGett As seen in the GetPlatformName() method, default names for all platforms are returned by this function. If you wish to use your own platform names, you will have to extend this partial class.

#### **Example**

This example will only cover a use case for the iOS platform. In the example, you wish to have three different custom platforms for the iOS base platform: one for iPods, one for iPhones, and one for iPads.

- First, in the Wwise project, you will add three different platforms in the platform manager: "iPod", "iPhone", and "iPad". (For more information on the Platform Manager in Wwise, see the Wwise Help > Setting Up Your Projects > Managing Platforms section of the Wwise documentation).
- 2. Then, in Unity, create a C# script that extends AkBasePathGetter. The contents of this file will be like this:

```
public partial class AkBasePathGetter
#if UNTIY IOS
    static partial void GetCustomPlatformName(re
f string platformName)
    {
        switch(UnityEngine.iOS.Device.generation
)
        {
            case UnityEngine.iOS.DeviceGeneratio
n.iPodTouch1Gen:
            case UnityEngine.iOS.DeviceGeneratio
n.iPodTouch2Gen:
            case UnityEngine.iOS.DeviceGeneratio
n.iPodTouch3Gen:
            case UnityEngine.iOS.DeviceGeneratio
n.iPodTouch4Gen:
            case UnityEngine.iOS.DeviceGeneratio
n.iPodTouch5Gen:
            case UnityEngine.iOS.DeviceGeneratio
n.iPodTouchUnknown:
                platformName = "iPod";
                break;
```

```
case UnityEngine.iOS.DeviceGeneratio
n.iPad1Gen:
            case UnityEngine.iOS.DeviceGeneratio
n.iPad2Gen:
            case UnityEngine.iOS.DeviceGeneratio
n.iPad3Gen:
            case UnityEngine.iOS.DeviceGeneratio
n.iPadMini1Gen:
            case UnityEngine.iOS.DeviceGeneratio
n.iPad4Gen:
            case UnityEngine.iOS.DeviceGeneratio
n.iPadAir1:
            case UnityEngine.iOS.DeviceGeneratio
n.iPadMini2Gen:
            case UnityEngine.iOS.DeviceGeneratio
n.iPadMini3Gen:
            case UnityEngine.iOS.DeviceGeneratio
n.iPadAir2:
            case UnityEngine.iOS.DeviceGeneratio
n.iPadUnknown:
                platformName = "iPad";
                break;
            case UnityEngine.iOS.DeviceGeneratio
n.iPhone:
            case UnityEngine.iOS.DeviceGeneratio
n.iPhone3G:
            case UnityEngine.iOS.DeviceGeneratio
n.iPhone3GS:
            case UnityEngine.iOS.DeviceGeneratio
n.iPhone4:
            case UnityEngine.iOS.DeviceGeneratio
n.iPhone4S:
            case UnityEngine.iOS.DeviceGeneratio
n.iPhone5:
            case UnityEngine.iOS.DeviceGeneratio
```

- Note: If platformName is left unmodified, the default platform name for the current active Unity platform will be used.
- 3. Either, (a) create a C# that extends AkBuildPreprocessor or (b) create a C# script that uses the functionality within AkBuildPreprocessor. The contents of the file could be as follows:

```
public class WwiseIOSBuildPreprocessor : IPrepro
cessBuild, IPostprocessBuild
{
    public int callbackOrder { get { return 0; } }
    string iPodDestinationSoundBankFolder = stri
ng.Empty;
    string iPadDestinationSoundBankFolder = stri
ng.Empty;
    string iPhoneDestinationSoundBankFolder = st
ring.Empty;
    public void OnPreprocessBuild(BuildTarget ta
rget, string path)
    {
```

```
if (target == BuildTarget.iOS)
            AkBuildPreprocessor.CopySoundbanks(t
     "iPod",
               iPodDestinationSoundBankFolder);
rue,
            AkBuildPreprocessor.CopySoundbanks(t
               iPadDestinationSoundBankFolder);
rue,
            AkBuildPreprocessor.CopySoundbanks(t
     "iPhone", iPhoneDestinationSoundBankFolder)
rue,
        }
    }
    public void OnPostprocessBuild(BuildTarget t
arget, string path)
    {
        DeleteSoundbanks(iPodDestinationSoundBan
kFolder);
        DeleteSoundbanks(iPadDestinationSoundBan
kFolder);
        DeleteSoundbanks(iPhoneDestinationSoundB
ankFolder);
    }
}
```

- 4. Or in Wwise, generate the SoundBanks for all three platforms "iPhone", "iPod", and "iPad", and copy the three resulting folders to UNITY\_PROJECT\_ROOT/Assets/StreamingAssets/Audio/GeneratedSoundBa
- 5. In Unity, build your game for the iOS platform.
- 6. From now on, a different set of SoundBanks will be used depending on the type of device used.

#### **Wwise Unity Integration**

#### **Making DLCs with Wwise and Unity**

#### **Using Wwise file packages**

The default way to support DLCs with Wwise is to use file packages. This is explained in part in the Authoring documentation: Wwise > Help > Finishing Your Project > Managing File Packages > Downloadable Content Overview. The short explanation is that these packages can contain BNK and WEM replacement or additional files, and they can be loaded and unloaded at will to enable or disable the DLC. This package format is optional, but it is currently built-in the Wwise/Unity plug-in. If you want to write your own package format, you'll need to write a new Low-Level IO to replace the default one.

To use packages in Unity, you simply have to place them in the Base Path, the path where your banks are, as defined in AkInitializer's properties. Then call AkSoundEngine.LoadFilePackage(). Wwise will always try to load files through the file packages first, in the reverse order (last added is the first searched).

This method will work on all platforms that allow you to write on disk, into your Base Path, after the game is shipped. This is unfortunately not the case with iOS and Android.

On Android and iOS, since all the data is bundled into a read-only file system, you will need to put your packages in a different directory. This is done by calling AkSoundEngine.AddBasePath(), before AkSoundEngine.LoadFilePackage(). You can add multiple paths, the last added will be searched first. Please check the appropriate Operating System documentation to find what are the restrictions. The code below is an example of code to select a path where you can store your files:

```
#if UNITY_IPHONE
    string fileNameBase = Application.dataPath.Subst
ring(0, Application.dataPath.LastIndexOf('/'));
    fileName = fileNameBase.Substring(0, fileNameBas
e.LastIndexOf('/')) + "/Documents/" + FILE_NAME;
```

```
#elif UNITY_ANDROID
    fileName = Application.persistentDataPath + "/"
+ FILE_NAME ;
#else
    fileName = Application.dataPath + "/" + FILE_NAM
E;
#endif
```

Note that there are several other writable paths on Android, such as the SD card or similar storage devices.

#### See also:

• Using OBB Files (Android)

#### **Streaming banks from Unity WWW service**

It is also possible to use the WWW class from Unity to get your DLC audio files and load them directly. This is done through in-memory loading of banks, as demonstrated in AkMemBankLoader.cs. If you are replacing existing banks with new ones, you must take care of properly unloading the previous banks first; the Wwise IO system would be unaware that code so it can't know if you're reloading the same bank by mistake or want to replace it.

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3



#### **Wwise Unity Integration**

### **Build the Native Integration Plug-in from Source**

The Wwise-Unity integration is a thin layer of C# calling the native Wwise API. It comes pre-built for all platforms. In most cases, it is unnecessary to rebuild the Integration plug-in. However, if you have the source code, you may want to customize it to integrate custom changes to the native sound engine or to modify the C++ code that interfaces with the sound engine. This should be done by a programmer who has some C++ knowledge.

#### The procedure

- 1. Install the Wwise SDK for the platforms you wish to build using the Wwise Launcher. Make sure to logoff and login again.
- 2. Locate the source zip for the operating system you are using (Windows or Mac), it is installed in your Unity project by the Launcher when you add the Wwise integration to it (the zip is post-fixed with Src.zip).
- 3. Locate the source zip for the platform you wish to build (if different from Windows or Mac).
- 4. Verify the prerequisites for building the Integration. See **Prerequisites** for details.
- 5. Unzip the source zips, all in the same folder. See **Installing or Upgrading the Integration in a Unity Project** for details.
- 6. Launch the build process using one of these two methods: **Build** using console utility or **Build** using Integrated Development Environment (IDE).
- 7. Wait until the build finishes. If the build succeeds, the plug-ins and API binding under the Deployment folder will be updated or created if none existed before.

### Notes

 During the build, you can monitor warnings, errors, or critical messages, if any, by reading the detailed log messages found in the Build Log.

## **Requirements to build the Integration yourself:**

Platform	Required Components
All platforms	<ul> <li>Unity 5 (Personal or Pro)</li> <li>Compatible Wwise SDK for your target platform(s). See Release Notes.</li> <li>Compatible platform SDK for your target platform(s). Refer to Wwise SDK help (Section Reference Materials &gt; Platform Requirements).</li> <li>Python 2.7.x or 3.x, with Python executable folder in your system executable path.</li> </ul>
Android	<ul> <li>Cygwin (Windows only).</li> <li>Android SDK: Minimum API 9 for 32 bits, Minimum API 21 for 64 bits</li> <li>Android NDK r10e.</li> <li>Apache Ant 1.8.4.</li> <li>The path to the Wwise SDK should not contain any white spaces.</li> <li>The path to Unity project root should not contain any white spaces.</li> <li>Environment variables: <ul> <li>CYGWIN_HOME should point to the Cygwin root folder (Windows Only).</li> <li>ANDROID_HOME should point to the Android SDK root folder.</li> <li>ANDROID_NDK_ROOT should point to the Android NDK root folder.</li> <li>ANT_HOME should point to the Apache Ant root folder.</li> </ul> </li> </ul>
iOS	<ul> <li>Xcode 7.2 or above.</li> <li>The Wwise SDK for iOS, and a wwisesdk environment variable pointing to it</li> </ul>

Linux	<ul> <li>Build-essential (sudo apt-get install build-essential)</li> <li>SDL2 (see Build for Linux on how to install the SDL2 library)</li> <li>The Wwise SDK for Linux, and a wwisesdk environment variable pointing to it</li> </ul>
Мас	<ul> <li>Xcode 7.2 or above.</li> <li>The Wwise SDK for Mac, and a wwisesdk environment variable pointing to it</li> </ul>
PS4	Visual Studio 2012 Professional Edition.
Windows	Visual Studio 2013 Professional Edition.
Windows Store	Visual Studio 2015.
Xbox One	Visual Studio 2012 Professional Edition.

#### How to install the source package

Unzip the source package. You can then rebuild and generate the content of the Integration using just the source package and a Wwise SDK installation for your target platforms. Optionally, you can merge the unzipped content under the Wwise folder with your imported Integration unity package in the Assets folder of your Unity project. However, this may generate unwanted intermediate files inside your Assets folder, which need to be cleaned up manually.

# The folder structure of the installed Integration package

- **StreamingAssets**: The **folder** used for automatic SoundBank deployment.
- Wwise: The top-level folder.
  - Deployment: The main Integration files used in your application.
    - API: C++ to C# binding files. This is the translation of the native Wwise SDK.
    - **Dependencies**: 3rd-party Unity scripts required by certain platforms.
    - Components: The Unity components scripts that can be used directly and serve as examples for custom components.
    - Plug-ins: The Unity plug-in libraries of various configurations and debug symbol databases.
      - Platform
        - Architecture: Files shared by architectures of a multi-architecture platform.
          - Debug: The Debug version of the plug-in binaries. See Wwise plug-in deployment for details.
          - **Profile**: The Profile version of the plug-in binaries. See **Wwise plug-in deployment** for details.
          - Release: The Release version of the plug-in binaries. See Wwise plug-in deployment for details.
          - **DSP**: The Wwise plug-in Effects.
  - **Documentation**: The full documents about the Integration.
  - Editor: All editor windows and inspectors used by the WwiseUnityIntegration.
  - Tools: Application tools that provide extra features.

## The folder structure of the installed source package

- **Wwise**: The top-level folder.
  - AkSoundEngine: The source code and IDE projects needed for building the Integration, organized by platforms.
    - Common: The source code and build scripts used by all supported platforms.
    - **Platform**: The IDE solution or project, the source code, and platform-specific build scripts for the target platform.
  - Integration/Assets/Wwise/Deployment: The main Integration files used in your application.
    - API: The root-level folder of the API binding.
      - Generated: API binding files generated during the SWIG binding step.
      - **Handwritten**: API binding files written by hand.
    - **Components**: The factory Unity component scripts that can be used directly and serve as examples for custom components.
    - Plugins: All the Wwise plug-ins (written after building AkSoundEngine).
      - <plan</p>**platform>**: The applicable platform folder.
        - <architecture>: The applicable platform architecture.

### **Build using console utility**

You can build the Integration from the command line using the build script located at <a href="https://www.negration.py">www.negration.py</a>. This is useful for continuous integration. To see the usage and examples, assuming your current working directory is the parent folder of the script, type in a command console on your computer:

python BuildWwiseUnityIntegration.py -h

## **Build using Integrated Development Environment (IDE)**

Located in the Wwise Unity Integration source package under:

WwiseUnityIntegration\_version\_platform\_Src.zip\Wwise
\AkSoundEngine\YourPlatform

the solution (or Xcode project) allows you to build the Integration for the target platform in a supported IDE.

#### Building for the Mac or iOS platform with Xcode

When building the integration from the command line, the wwisesder path is provided to Xcode automatically based on the one set in the environment variable \$\text{\$\text{\$\text{\$WWISESDK}\$}}\$ or the one provided to the build script using the -\text{\$\tex

#### **Building for the Linux platform**

A premake script is included in the integration source package that generated the necessary makefiles. To build the Linux plug-in from source, simply enter the following commands into a terminal:

- cd <Integration source location>/AkSoundEngine/Common
- ./premake4 --akplatform=Linux gmake
- make -f AkSoundEngineLinux.make config=<config>

Where <config> is either debug32, profile32, release32, debug64,

profile64, Or release64. The resulting plug-in will be located in <Integration source location>/Deployment/Plugins/Linux.

## **Deploy the new lib**

The resulting binaries will be found in Wwise\Deployment\Plugins\ [Platform]. If you haven't unzipped the sources in your Assets\Wwise directory, copy the new binaries from that folder to the corresponding folder in your project's Assets directory.

#### Under the hood

Prior to packaging the sources for building the integration code, we use SWIG to generate the API binding in the Unity for Wwise SDK. Most users will never need SWIG to work with Unity for Wwise. Even advanced users with access to the Wwise for Unity sources for building the integration code, which contains the converted APIs, do not need to install or use SWIG. The only exceptions to this are for integrating changes to the sound engine or modifying the C++ code that interfaces with it. In such rare cases, we recommend contacting support before proceeding.

The build process performs the following task:

1. The Wwise SDK libraries are wrapped into a single library as the Unity plug-in, a dynamic library (.dll) on Windows, a loadable bundle (.bundle) on Mac OS X, a static library (.a) on iOS, or a shared library (.so) on Android. They are directly built into the Deployment folder. See Installing or Upgrading the Integration in a Unity **Project** for the folder structure.

#### See also:

- Platform-Specific Information
- Build Loa
- Updating/Upgrading the Wwise Integration Package to a Newer Version When You Have Modified the Plug-in's C++ Code

**Wwise Unity Integration » Build the Native Integration Plug-in from Source** 

#### **Build Log**

Warnings and error messages are displayed in the Unity Editor console and in the Build Log. The Build Log file contains more detailed messages created while building the Integration. The Build Log is stored in the following location:

```
UNITY_PROJECT_ROOT\Assets\Wwise\Logs\BuildWwiseUnity
Integration.log
```

The log messages are displayed in the following format:

```
Time: Message type: Source Code File name (Module na
me): Line number: Message body
```

#### Example:

```
2013-09-26 09:29:56,490: INFO: BuildWwiseUnityIntegration.py (WindowsBuilder): 91: Building: Windows (Win32, Debug) ...
```

The log file accumulates and rotates to a new file after an hour. You can modify the rotation behavior by configuring the BuildUtil.CreateLogger() function in the following file from the Wwise Unity Integration source package:

```
WwiseUnityIntegration_version_platform_Src.zip\Wwise
\AkSoundEngine\Common\AkSoundEngine\Common\BuildUtil
.py
```

Refer to the Python logging.handlers module for more information.

On certain platforms, additional log files may be generated by an IDE such as Visual Studio. The additional files are available under the same folder with the keyword IDE in their filenames.

**Wwise Unity Integration » Build the Native Integration Plug-in from Source** 

# Updating/Upgrading the Wwise Integration Package to a Newer Version When You Have Modified the Plug-in's C++ Code

#### **Warning: Recommendations**

- The Unity migration operation is not reversible. Make sure to back up your Unity project before you proceed.
- If the Unity project is under source control, make sure to check out the whole Unity project folder before starting the migration. Otherwise, the migration will fail because it won't be able to modify the files successfully.
- Before you install the Wwise integration, you need to make sure your Unity project is in a state where it can compile successfully without any errors. (Warnings are OK.)

This installation procedure assumes that you already have a Unity project containing a previous Wwise Integration package. For new installations, please refer to the Installing a Wwise Integration Package for the First Time page. Unlike other Unity plug-ins, you need to leave the previous Wwise Integration package installed in order to upgrade it to a newer version. The Launcher will install the new features and bug fixes, and then migrate the previous Wwise Integration to the newer version.



**Warning:** Do not delete the previous Wwise Integration before installing the new one as it will result in losing all the hard work you've done integrating Wwise audio in your game.

The upgrade is done in a 3-step process. First you need to update your Wwise project to the new version you wish to use and then you need to do the same with your Unity project.



## **Updating your Unity project to the new Wwise version**

Update your Unity project to the new Wwise version using the Wwise Launcher.

#### **Updating your Wwise project to the new version**

- 1. Launch the Wwise version that matches the new Wwise Integration version you intend on using in Unity.
- 2. Open your Wwise Project.
- 3. Answer 'Yes' when prompted to update your Wwise project to the current version.
- 4. Once completed, make sure to regenerate your Wwise SoundBanks to be ready to carry on with your work after updating the Unity project to the new Wwise version.
- 5. Copy the generated SoundBanks to your Unity project's StreamingAssets folder.

#### **Updating your C++ code modifications**

- 1. Install the updated Wwise SDK on your machine, for all your desired platforms.
- 2. Locate the Unity Integration Source Code zips that were copied in your Unity project for all your desired platforms.
- 3. Extract all the downloaded Unity Integration Source Code packages to the same directory
- 4. Merge the modifications you have made to the Unity Integration Source Code into the updated location
- 5. Compile the updated and merged integration source code

#### **Wwise Unity Integration**

#### **API Changes and Limitations**

The following features and functions of the native Wwise SDK are not supported, or their behaviors have been modified in the Integration (most recent change first):

- UnloadBank() no longer takes a single argument overloaded version.
   A memory-location pointer has to be provided as the second argument.
- An iOS-specific API is added to handle audio interruptions: AK::SoundEngine::iOS::ListenToAudioSessionInterruption().
- The Windows version of GetGameObjectFromPlayingID() returns GameObject ID in 32bit for both Windows 32bit and 64bit (truncation).
- PostEvent() and GameObject-related APIs now accept null as the GameObject input argument to affect the global scope.
- PostEvent() callbacks are serialized and occur on the start of the next game frame instead of being called from the audio thread.
   Therefore no synchronization mechanism is necessary.
- AK::Monitoring::SetLocalOutput() has been replaced by AkCallbackManager.SetMonitoringCallback().
- AK::SoundEngine::SetPosition() has been replaced by AkSoundEngine.SetObjectPosition().
- External Sources arrays in PostEvent() are limited to one source.
- Streaming sounds should not be done at the same time as level data is loaded from disk.
- Several AK::SoundEngine::DynamicSequence API functions are renamed in their Unity API binding to maintain clarity. This is a remedy to compensate for the namespace-flattening side effects caused by SWIG. For example,
  - AK::SoundEngine::DynamicSequence::Open() would have become AkSoundEngine.Open() in the generated API. So we rename it into AkSoundEngine.DynamicSequenceOpen() in the API binding.
- C++ overloaded operators are renamed into functions in the Unity scripting language; for example, AkArray::operator[] becomes AkPlaylistArray.ItemAtIndex(uint uiIndex). This is done according to how SWIG is designed to handle C++ overloaded operators.
- Unsupported API and components:

- iOS audio input callback API
- AkSpeakerVolumeMatrixCallbackInfo
- AkSpeakerVolumeMatrixBusCallbackInfo
- AkBusCallbackFunc
- AK::SoundEngine::RegisterBusVolumeCallback
- AK::SoundEngine::RegisterCodec
- AK::SoundEngine::RegisterGlobalCallback
- AK::SoundEngine::RegisterPlugin
- AK::SoundEngine::Query::AkGameObjectsList
- AK::SoundEngine::Query::GetActiveGameObjects()
- AK::SoundEngine::Query::GameObjDst
- AK::SoundEngine::Query::AkRadiusList
- AK::SoundEngine::Query::GetMaxRadius(AkRadiusList& io\_RadiusList)
- all variants of

AK::SoundEngine::DynamicDialogue::ResolveDialogueEvent that accept callbacks or event names

#### See also:

• Platform-Specific Information.

#### **Wwise Unity Integration**

## **Platform-Specific Information**

#### **Android**

- Build for Android
- Background Mode (Android)
- Using OBB Files (Android)
- Making DLCs with Wwise and Unity

### iOS

- Build for iOS
- Audio Session Interruptions (iOS)Making DLCs with Wwise and Unity

## Linux

• Build for Linux

## **Windows Store Apps**

• Windows Store Apps Specific Information



**Wwise Unity Integration » Platform-Specific Information** 

## **Background Mode (Android)**

When the app is put in background, you should call AkSoundEngine.Suspend(). When coming back to the foreground, restore the sound with AkSoundEngine.WakeupFromSuspend(). This is currently implemented in AkInitializer, but you could move it to other parts of your game if it makes sense.



**Wwise Unity Integration » Platform-Specific Information** 

## **Using OBB Files (Android)**

#### Overview of the OBB and Wwise IO

For the Android platform, when checking the Unity option Player Settings > Publishing Settings > Split Application Binary, APK Expansion Files are used for bundling the project assets. This effectively creates a zip file with the .obb extension that contains the StreamingAssets folder, including the SoundBanks. The default Android SoundBank loading support assumes that SoundBanks are located in either the APK file or OBB file.

If you are constructing your own OBB file externally, note that the inner path given to AkSoundEngine.SetBasePath must be the same in the OBB. OBB files must be uncompressed and unencrypted for the default I/O system to work. If this is not the case, then you will need to uncompress the OBB in the persistent data path. The CPU cost to uncompress is too high to uncompress on the fly.

#### **Loading banks in memory**

Alternatively, you can use in-memory loading of banks. An example is provided in AkMemBankLoader.cs. This script tries to load the specified SoundBank at the start time of the scene, and unload it when the scene is destroyed. The following changes need to be made to use this example:

- 1. Add the script as a component to any game object that needs to load a SoundBank.
- 2. In the Inspector, enter the file name of the SoundBank into the **Bank Name** field of the script component.
- 3. In the Inspector, select the Is Localized Bank Option if the SoundBank is localized. It will use the global language defined in AkInitializer.cs.
- 4. Disable any other SoundBank-loading methods and save the scene.

Two methods AkMemBankLoader.LoadNonLocalizedBank() and AkMemBankLoader.LoadLocalizedBank() can also be called directly in code for the non-localized and localized SoundBanks.

#### **Limitations**

- The in-memory method cannot load the streamed SoundBanks. As a better but more difficult alternative. low-level IO hooks can be implemented to decompress the zip file on the fly to support both streamed and non-streamed SoundBanks. This example, however, will be left for future work. Users are free to implement such hooks and rebuild the Integration. Refer to the section streamingmanager lowlevel to do so.
- For simplicity, the example script loads only one SoundBank per script component, but one can easily extend it to a collection-based version that can batch-load a list of SoundBanks.
- The example only wraps the simplest in-memory SoundBank-loading API; there is another callback-based version which you can easily add when you need. The API binding is already available in the Integration.



**Wwise Unity Integration » Platform-Specific Information** 

### **Audio Session Interruptions (iOS)**

The Unity Integration supports two audio session categories.

- AkAudioSessionCategorySoloAmbient (Default): When you use this category, audio from other apps is silenced. Your app's audio is also silenced by screen locking and by the Silent switch (called the Ring/Silent switch on the iPhone).
- AkAudioSessionCategoryAmbient: AmbientSound allows your app's audio to be mixed with other apps. Any audio interruptions, such as an incoming phone call, or playing music with the Music app using a remote control, will be mixed with your app's audio by default. However, you can specify route sounds and music to a special bus so that they get muted when user music plays and unmuted when user music stops. The muting and unmuting are done by the sound engine internally. You can still define the BGM (BackGround Music) callback to respond to user music behaviors for other useful tasks, such as updating a UI element to notify users of the user music playback status. Use AkcallbackManager.SetBGMCallback() to register that callback.

Playing user music using the remote-control will trigger an interruption (under the SoloAmbient category) or the BGM event (under the Ambient category), however, stopping user music using the remote-control will trigger a BGM event under all categories. Again, the sound engine takes care of all the audio pipeline tasks regarding such events. The app can register a BGM callback to respond to these events if extra work is needed.



**Note:** The background and foreground switching is handled internally on iOS, no need to call Suspend and WakeupFromSuspend manually on this platform.

**Wwise Unity Integration » Platform-Specific Information** 

## **Windows Store Apps Specific Information**

• Universal Windows Platform use regular Windows SoundBanks. There is no separate platform SoundBanks for those two SDKs.



#### **Wwise Unity Integration**

## **Licensing (Free & Commercial)**

Wwise licenses are required for any games that are released publicly. Licenses are issued per project, so only one license is needed for the whole development team. For more information about licensing, go to the Audiokinetic website.

- Free Limited Commercial License
- Educational & Non Commercial License
- Royalty-based License
- Commercial License
- Licensing FAQ

#### **Wwise Unity Integration**

#### **Wwise Demo Scene**

A demonstration scene containing the Wwise Unity Integration, is available to download from the Wwise Launcher Unity page, under the contextual menu in the "Recent Unity Projects" title. This simple scene, created using Unity's standard assets exclusively, demonstrates how to use some of the Integration's basic features.

This First-Person 3D map contains "stations" along a path. For each station, a small description was made available on a sign next to the station. For easy reference, each station's assets have been nested in the scene hierarchy.

#### Installation

The Wwise Demo Scene is a standalone project. It is available to download in the Wwise Launcher. **It should not be used as a foundation for your game.** Use the Wwise Launcher to integrate Wwise in a new Unity project for this purpose.

#### NOTE:

- The SoundBank generation are included in the package.

#### **Deployment**

The Wwise Demo Scene is intended as a way to preview and show how the Wwise Unity Integration should be done in the Unity Editor.

To deploy the Wwise Demo Scene to a game console or mobile device, follow these steps:

- 1. Within the Launcher's Unity tab, select the **Modify Wwise in Project...** option from your Wwise Demo Scene Unity Project.
  - 1. The Launcher's Unity integration page opens.
- 2. Add the desired Deployment Platforms and click **Modify**.
- 3. Open the updated version of the project in Unity.
- 4. Generate the SoundBanks for the platform.
- 5. Copy the Generated SoundBanks folder to the StreamingAssets/Audio folder.
- 6. Within Unity, build the scene for your desired platform then deploy to the device.

#### Footsteps demo

The footpath, along with the first station, demonstrates how to use scripting and trigger volumes to create a footstep system.

In the Wwise Project, footsteps have been implemented in the recommended way: Random containers for each surface type, nested under a Switch Container. A footstep can be posted using the Footstep event, and the surface material is controlled with the Footstep\_material switch group.

The scene's terrain has been painted with four different textures: grass, gravel, wood, and dirt. Walking around the different surface types changes the footstep sound accordingly.

To accomplish this, Box Colliders have been placed over each zone. Setting the Footstep\_Material switch is then done simply by dragging a Switch Value from the Wwise Picker Window to a Box Collider. In order to set the switch on the First Person Controller when it enters the collider, we need to trigger the Ak Switch script on the "AkTriggerEnter" trigger, and make sure the "Use Other Object" check box is enabled (as seen in the Inspector Window).

A very rudimentary footstep script has been implemented, which simply posts the Footstep event every 0.3 seconds when the player is moving. This script has been attached to the First Person Controller. This illustrates a way to post an event to the Wwise SoundEngine via scripting.

## **Subtitle demo**

This station illustrates how to create custom event triggers in Unity, as well as using event callbacks.

A simple button script contains a delegate to be run when the player is close enough to the button, and presses a key on the keyboard or controller. Another script, AktriggerButtonPress, registers itself on that delegate, and then calls triggerDelegate from its parent class AktriggerBase. Inheriting from AktriggerBase allows the custom trigger to show up in the "trigger" list in Wwise Component Inspector windows (for example, the Ak Ambient attached to the button). For more information on custom triggers, see Adding New Triggers for Wwise Events.

The event starts playing a sound file containing WAV markers. Registering the marker callbacks associated with the event allows the updating of the subtitle on the panel. To achieve this, in the Ak Ambient's inspector, the "Use Callback" option was checked. Then the GameObject (SubtitleSign), containing a script (SubtitleDemo.cs), was dragged & dropped onto the "Game Object" box. The name of the Callback Function (MarkerCallback), to execute when the callback is triggered, was typed and "Marker" was chosen as the Callback Flags. Looking at SubtitleDemo's code, it can be seen that the Callback function is simply reading the subtitles from a pre-defined string array, using the MarkerCallback's uIdentifier field as an index. See How to use AkAmbient with the inspector for more information on callbacks.

#### **Environment demo**

This station demonstrates how to apply effects on sound within a zone in the scene. Two caves contain an EnvironmentZone (once again, a Box Collider is acting as a trigger) on which an AuxBus has been added (Dragged & Dropped from the Wwise Picker window).

In Wwise, two Auxiliary Busses have been created, each containing a different effect. Furthermore, the footsteps and the Little Sequence have "Use game-defined Auxiliary sends" checked.

When a Wwise event is posted from within the EnvironmentZone, its sound is routed to the AuxBus associated to the zone.

This station also demonstrates the Environment Portal component. This allows a spatial transition between two environments. The three buttons in this demonstration are used to illustrate how an Environment Portal affects the mixing of the Auxiliary Busses.

To create an environment portal, first create a Box Collider that touches two Ak Environments on one of its axes (in this demonstration, the portal touches the Red zone and the Blue zone on the 'z' axis). Then add the Ak Environment Portal component, and select the correct axis. The two environments, which will be mixed, will be automatically filled in.

For more information on Environments and Environment Portals, see How to use AkEvironment and AkEvironmentPortal with the inspector (Reverb Zones).

## **Timeline Demo**

This station demonstrates the use of an AkEventTrack and an AkRTPCTrack within a Timeline in Unity. There are two cubes, which, when the button is pressed, are animated towards each other and stopped when they reach impact.

In order to inspect the Timeline, select Window->Timeline. Then in the Hierarchy, expand the TimelineDemo object, and then expand the Timeline Demo Button object. Then select the Button object. The Timeline editor will now show the Timeline that controls the animation and Wwise Events for the cubes. The Timeline is controlled by the Playable Director component of the Button object.

The Timeline contains two animation tracks, one for each cube. These tracks are used to animate the z position of each cube over time. The Timeline also contains an AkEventTrack and an AkRTPCTrack, marked by a white and red tab, respectively. The AkEventTrack contains two AkEventPlayable clips: the first triggers the PlayCubeMovement Wwise Event, and the next triggers the PlayImpact Wwise Event. The names of these Wwise Events are displayed in the clips. You may need to increase the size of the Timeline editor view and zoom in in order for the name to be displayed correctly. The PlayCubeMovement Wwise Event plays a sine wave source, which has an RTPC affecting its pitch. The name of this RTPC is CubeAcceleration. The AkRTPCTrack increases this RTPC over time as the cubes move towards each other. This causes the pitch of the sine wave to increase, producing a simple sound effect to indicate the acceleration of each cube towards the other.

For more information on the Timeline integration, see **Wwise Timeline** Integration

Main Page Classes

# **Wwise Unity Integration**

# **Using Wwise Spatial Audio in Unity**

This chapter gives an overview of how to use the Spatial Audio features of the Unity integration. It is broken down into:

- A preparation page you need to follow before going to the tutorials:
  - Preparation for the Spatial Audio Tutorials
    - 1. Create a Unity Project
    - 2. Wwise Project Preparation
    - 3. Unity Project Preparation
- Two independent tutorials:
  - Surface Reflectors Tutorial
  - Rooms and Portals Tutorial
- And a supplementary page addressing the use of obstruction and occlusion:
  - Obstruction and Occlusion

**Note:** Completion of sections using the Wwise Reflect plug-in require the appropriate license.

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3

Main Page Classes

**Wwise Unity Integration » Using Wwise Spatial Audio in Unity** 

# **Preparation for the Spatial Audio Tutorials**

The following steps must be followed beforehand to successfully complete the **Surface Reflectors Tutorial** and the **Rooms and Portals Tutorial**:

- 1. Create a Unity Project
- 2. Wwise Project Preparation
- 3. Unity Project Preparation
  - 3.A. Environment
  - 3.B. Character Listener
  - 3.C. Third-Person Emitters
  - o 3.D. Scene

# 1. Create a Unity Project

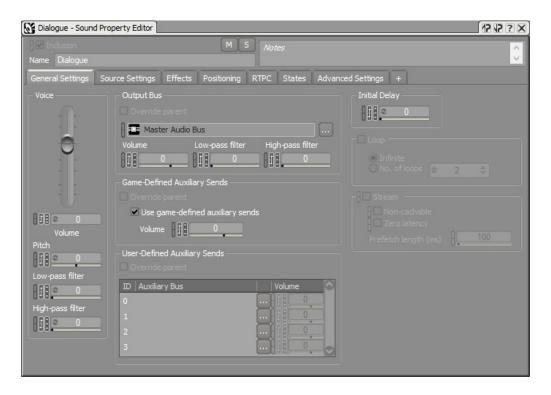
Using the latest version of Wwise and Unity, follow the steps below in order to build your working environment.

- 1. Launch Unity and create a new project.
- 2. Close Unity.
- 3. Start the Wwise Launcher and integrate Wwise into your Unity project.

## 2. Wwise Project Preparation

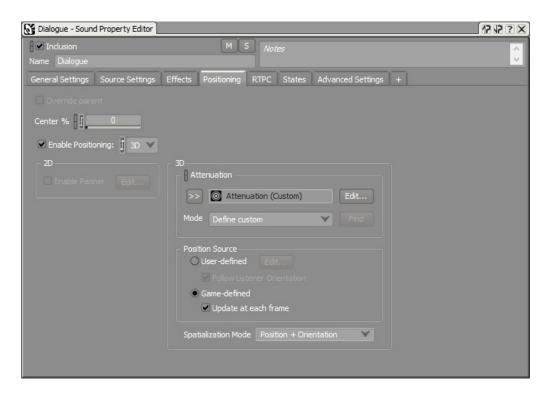
For the tutorials, you will need a Sound SFX, an Event to play it, and a SoundBank.

- 1. From the Launcher, open your project in Wwise.
- 2. Add a Sound SFX under the Actor Mixer Hierarchy's **Default Work Unit**. In the integration Wwise Project, we added a dialogue.
  - 1. Import a sound into it.
  - 2. In the Sound Property Editor's
    - 1. General Settings tab, enable **Use game-defined auxiliary** sends.



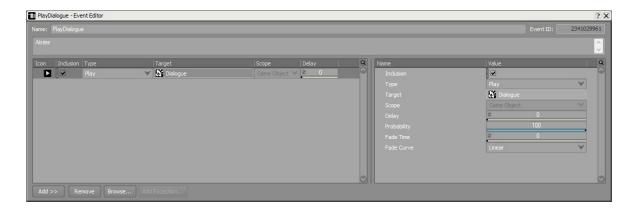
**Sound Property Editor General Settings tab** 

2. Positioning tab, enable positioning and choose **3D**. Optionally, add an Attenuation.



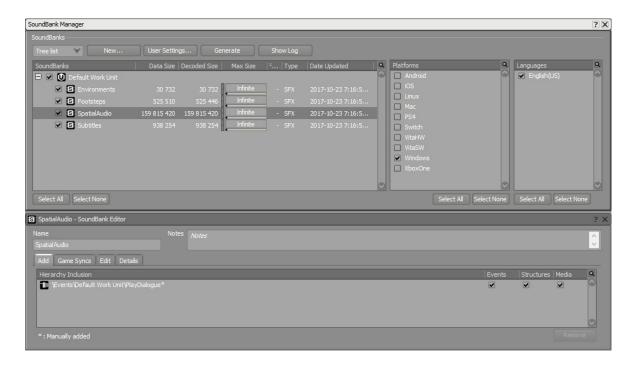
**Sound Property Editor Positioning tab** 

3. Right-click on the Sound SFX within the Actor-Mixer Hierarchy, then select **New Event > Play**.



**Event Editor** 

4. Go to the SoundBank layout (shortcut F7) and add a new SoundBank. Drag the Event into the new SoundBank. Here's how the SoundBank Manager should look in the integration Wwise Project:

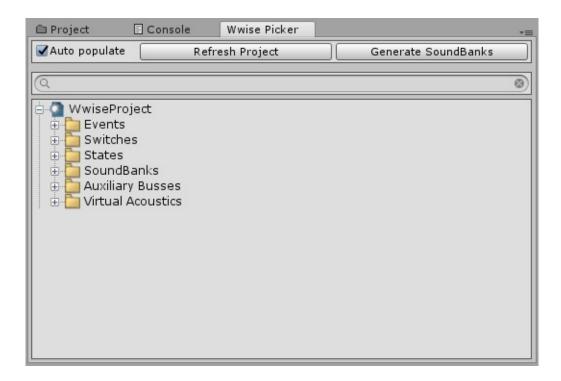


**SoundBank Manager and Editor** 

5. Save your project.

# 3. Unity Project Preparation

In this section we will create our game area and the objects in it. We will need a listener, emitters, and two rooms. From the Launcher, open your project in Unity. Open the Wwise Picker (Windows > Wwise Picker), Refresh Project, and Generate SoundBanks.



**Wwise Picker** 

#### 3.A. Environment

 Create a floor, walls, and ceilings using cubes: GameObject > 3D Object > Cube.

<b>-</b>			
Object Name	Position	Rotation	Scale
Floor	(0, 0, 0)	(0, 0, 0)	(50, 0.5, 50)
Wall Front Left	(-4.5, 3, -10)	(0, 0, 0)	(7, 6, 0.5)
Wall Front Middle	(0, 5, -10)	(0, 0, 0)	(2, 2, 0.5)
Wall Front Right	(2.5, 3, -10)	(0, 0, 0)	(3, 6, 0.5)

Wall Left Small	(-7.75, 3, -6)	(0, 0, 0)	(0.5, 6, 8)
Wall Back Small	(-2, 3, -2)	(0, 0, 0)	(12, 6, 0.5)
Ceiling Small	(-2, 5.75, -6)	(0, 0, 0)	(12, 0.5, 8)
Wall Right Small	(3.75, 3, -8)	(0, 0, 0)	(0.5, 6, 3.5)
Wall Front Large	(11.5, 3, -7.5)	(0, 0, 0)	(16, 6, 0.5)
Wall Right Large	(19.25, 3, 3.25)	(0, 0, 0)	(0.5, 6, 21.5)
Wall Middle	(3.75, 5, -4.5)	(0, 0, 0)	(0.5, 2, 3.5)
Wall Left Large	(3.75, 3, 5.625)	(0, 0, 0)	(0.5, 6, 16.75)
Wall Back Large	(11.5, 3, 14)	(0, 0, 0)	(16, 6, 0.5)
Ceiling Large	(11.5, 5.75, 3.25)	(0, 0, 0)	(16, 0.5, 22)
Barrier	(-4.5, 3, -17)	(0, 0, 0)	(7, 6, 0.5)

2. Add spotlights inside rooms ( **GameObject** > **Light** > **Spotlight** ).

Object Name	Position	Rotation	Scale
Spotlight Small Room	(-3, 5.75, -6)	(0, 0, 0)	(1, 1, 1)
Spotlight Large Room	(11.5, 5.75, 5)	(0, 0, 0)	(1, 1, 1)

- 1. In the **Light** component, change:
  - 1. **Range** to 30.
  - 2. **Spot Angle** to 179.

#### 3.B. Character Listener

1. Create a character with the method of your choice.

<b>Object Name</b>			
MainCharacter	(0, 1.3, -20)	(0, 0, 0)	(1, 1, 1)

- 1. Remove any Unity Audio Source.
- 2. Add an **Ak Audio Listener** component to the camera.



## **Ak Audio Listener component**

3. Add an **Ak Spatial Audio Listener** component to the camera.



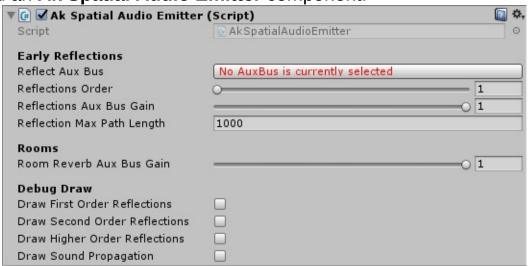
#### **Ak Spatial Audio Listener component**

#### 3.C. Third-Person Emitters

Create buttons using cylinders ( GameObject > 3D Object > Cylinder ).

Object Name	Position	Rotation	Scale
Button Outside	(-3, 0.75, -15)	(0, 0, 0)	(0.15, 0.5, 0.15)
Button Small Room	(-3, 0.75, -5)	(0, 0, 0)	(0.15, 0.5, 0.15)
Button Large Room	(7.5, 0.75, 5)	(0, 0, 0)	(0.15, 0.5, 0.15)

2. Add an Ak Spatial Audio Emitter component.



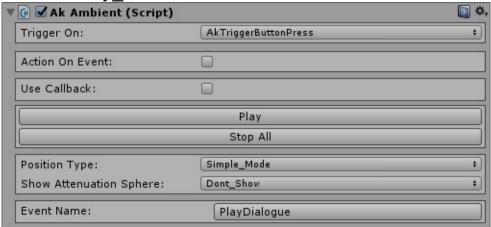
**Ak Spatial Audio Emitter component** 

- 3. In the Ak Game Obj component:
  - 1. Enable the Environment Aware option.
  - 2. Enable the Use Default Listener option.



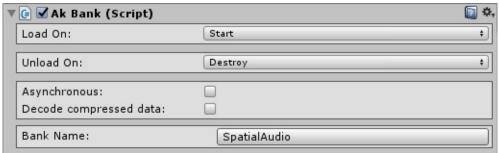
**Ak Game Obj component** 

- 4. Add an **Ak Ambient** component:
  - 1. Choose your preferred way to start the sound in **Trigger On**.
  - Choose the Play\_sound Event in Event Name.



**Ak Ambient component** 

- 5. Add an Ak Bank component:
  - Add the SoundBank created in 2. Wwise Project Preparation to Bank Name.



Ak Bank component

- 6. Start the game, connect it to Wwise, and open the Profiler layout (shortcut F6).
- 7. Play the sound from a button. You should hear the sound you imported in the Wwise project.

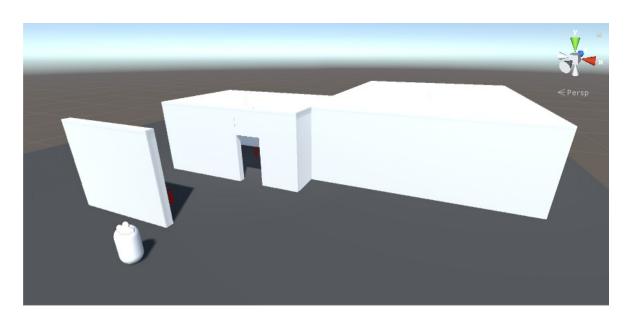
8. In the Voices Graph tab, you should see the following graph.



**Button Outside Voice Graph with no Effect** 

## 3.D. Scene

- 1. Save your Scene to a convenient place.
- 2. The final look of the scene is shown in the following image. This scene is available with the Unity integration package. Colors were added to elements of the scene for convenience. A mobile first-person controller was also added.



**Tutorial Unity Scene** 

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3



Main Page Classes

**Wwise Unity Integration » Using Wwise Spatial Audio in Unity** 

## **Surface Reflectors Tutorial**

In the following steps, this tutorial will show you how to use the new Reflect plug-in with Spatial Audio. You will need a Reflect plug-in license to make the project work.

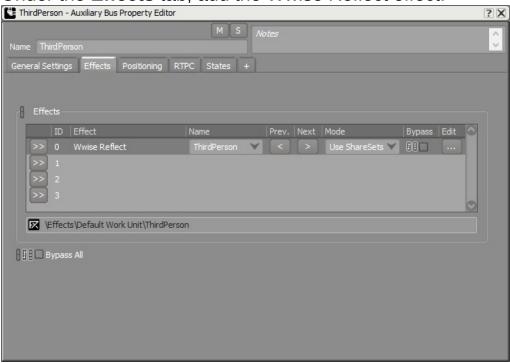
- A. Wwise Project
- B. Spatial Audio Emitter
- C. Surface Reflector Component
  - C.1. Use Existing Meshes
  - C.2. Alternative: Create a New Volume

**Note**: This tutorial presumes that you have completed the **Preparation for the Spatial Audio Tutorials**. However, it does not consider if the **Rooms and Portals Tutorial** was done; both tutorials are independent.

# A. Wwise Project

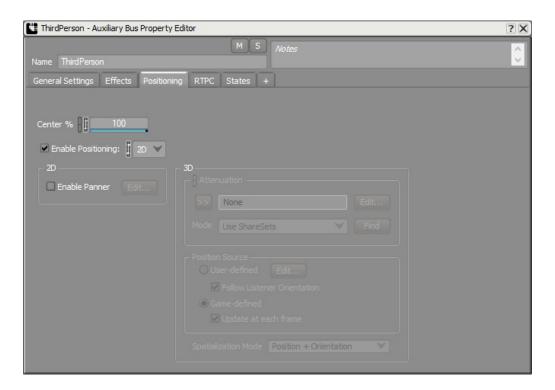
We need to update our wwise project with an Auxiliary Bus with the Reflect plug-in and Acoustic Textures.

- 1. Add an Auxiliary Bus under the Master Audio Bus.
  - 1. In the Auxiliary Bus Property Editor,
    - 1. Under the **Effects** tab, add the Wwise Reflect effect.



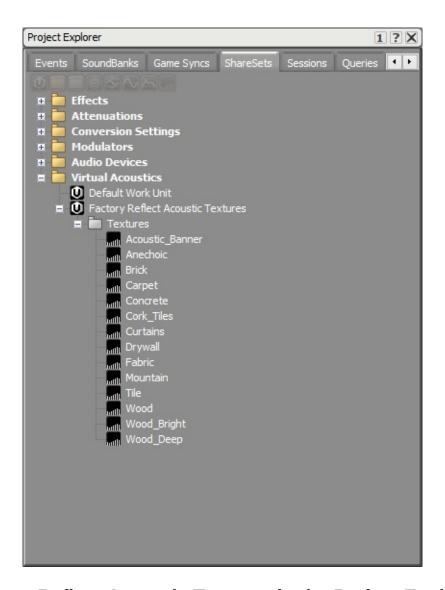
**Auxiliary Bus Property Editor Effects tab for Reflect** 

2. Under the **Positioning** tab, enable positioning and choose 2D.



Auxiliary Bus Property Editor Positioning tab for Reflect

- 2. Import Reflect Acoustic Textures from: Project > Import Factory Assets...
  - 1. They will be added to the **Virtual Acoustics** folder in the **ShareSets** tab of the **Project Explorer** view.



# Factory Reflect Acoustic Textures in the Project Explorer ShareSets tab

- 2. (optional) You can add custom Acoustic Textures in the **Virtual Acoustics** folder under **Default Work Unit**.
- 3. Save your project.

# **B. Spatial Audio Emitter**

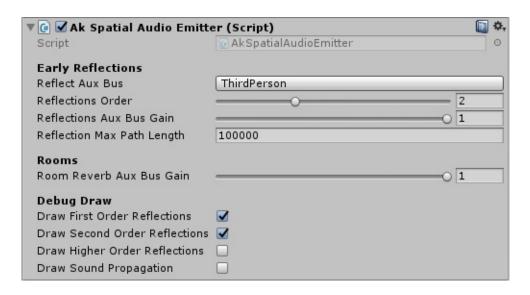
We need to modify the spatial audio emitter to use our newly created reflect aux bus.

- Refresh your Wwise project in the Wwise Picker: Windows > Wwise Picker
  - 1. Generate SoundBanks



#### **Wwise Picker**

- 2. Modify the **Ak Spatial Audio Emitter** component of each button.
  - 1. Pick the Reflect Auxiliary Bus for Reflect Aux Bus.
  - 2. Choose 2 for the Reflections Order.
  - 3. Choose 1 for the **Reflections Aux Bus Gain**.
  - 4. Choose 100000 for the Reflection Max Path Length.
  - 5. Check the **Draw First Order Reflections** and **Draw Second Order Reflections** under **Debug Draw**.



**Ak Spatial Audio Emitter for Surface Reflector** 

## **C. Surface Reflector Component**

You can simulate early reflections with the **Ak Surface Reflector** component. One way to have surface reflectors is to use existing meshes of the game. If the shape is too complex, you may want to add a new object as a surface reflector.

## C.1. Use Existing Meshes

The advantages of using the existing meshes are that you don't need to create new objects and each mesh can have a different acoustic texture. The disadvantages are that if you have a complex figure, you may send too much triangles to the Spatial Audio API and this could slow down your game. Also, another disadvantage is that each mesh only gets assigned one texture. In the case of the this tutorial, this means that it is not possible to have a different acoustic texture on the outside than on the inside of the rooms. In the scene provided by the unity integration, this is the method used for surface reflectors.

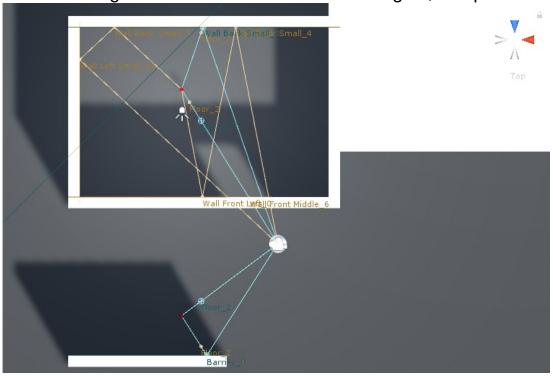
- 1. Select all the walls, ceiling and floor from section **3.A. Environment** and add an **Ak Surface Reflector** component.
  - 1. In the **Acoustic Texture** parameter, choose an acoustic texture or leave it at **None** to have an unfiltered reflection.



**Ak Surface Reflector component** 

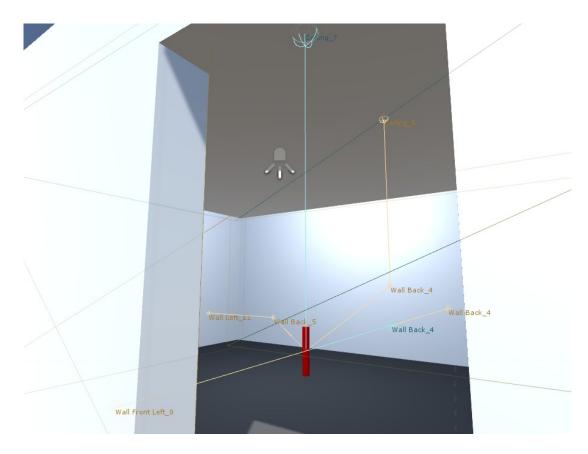
- 2. Start the game.
  - 1. Since we have enabled the drawing of reflections under **Debug Draw** in section , you should see rays going from the emitter to the listener in the **Scene** window. (The ceiling was removed in the following image to see the rays clearly) The blue rays are first order reflections, i.e. they reflect from one surface before reaching the listener, and the yellow rays are second order reflections. The name of the triangles are displayed at the reflection point. A name is made up of the game object's name

and the triangle's number. A cube has 12 triangles, two per face.



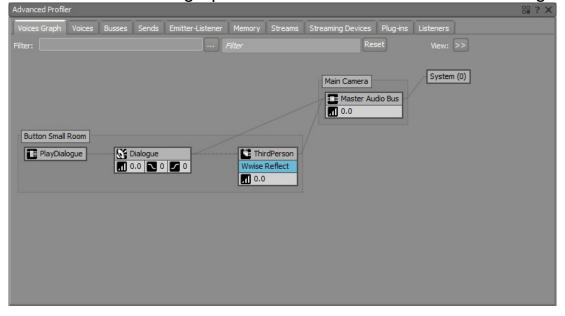
Top view of early reflection drawings for existing mesh surface reflectors

2. You can see the rays in the **Game** window by enabling **Gizmos**; as we can see in the next image. From this point of view, we can also see that the **Debug Draw** option draws the triangles of the surfaces that are being reflected on. Leaving the ceiling in this view, we can see that the sound will also reflect from the ceiling on triangles 7 and 6.



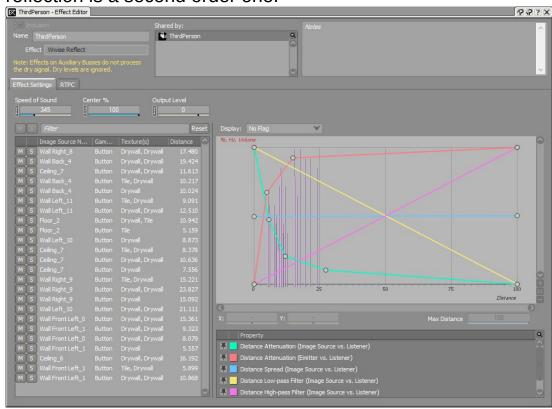
First person view of early reflection drawings for existing mesh surface reflectors

3. Connect the game in Wwise and go to the Profiler Layout. You should see a similar graph when Button Small Room is emitting.



#### **Button Small Room Voice Graph with Reflect Effect**

4. Double click on the Auxiliary Bus with the Reflect Effect in the graph. Navigate to the Effects tab and double click on the Reflect Effect. When playing the sound, you should see the current reflections in the graph and the list of the Reflect Effect Editor. In thw following image, there is a Drywall acoustic texture for all the walls and the ceiling and a Tile acoustic texture for the floor. When there are two textures in the list, it means that the reflection is a second order one.



Reflect Effect Editor View

#### C.2. Alternative: Create a New Volume

Creating a new game object with no mesh renderer can simplify a complex shape. In this optional section, we will show you how to have a surface reflector volume for the interior of the small room. You can also mix solutions by having some meshes be Surface Reflectors and having an object with no mesh renderer as well. For example. Having each mesh be a surface reflector to have different texture per wall inside of the

house and adding and exterior volume to have a completly different texture on the outside of the house. It is also possible to use **Quads** instead of cubes to have a different texture wherever desired. But be careful with intersections when using quads; you don't want your sound to get out of your shape.

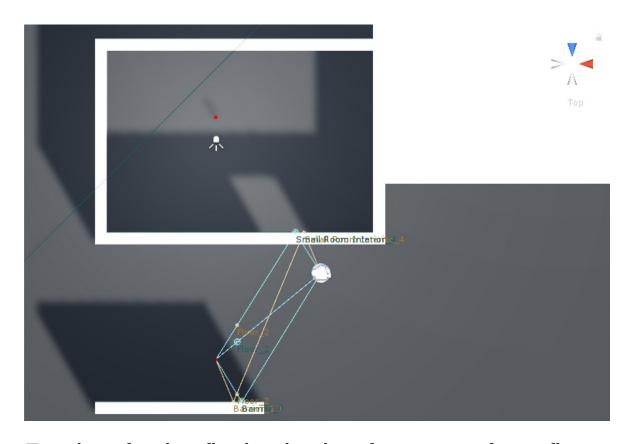
 Create a simpler unrendered mesh using a cube: GameObject > 3D Object > Cube

<b>Object Name</b>	Position	Rotation	Scale
Small Room Interior	(-2, 2.875, -6)	(0, 0, 0)	(11, 5.25, 7.5)

- 1. Remove the **Mesh Renderer** component
- 2. Check the Is Trigger in the Box Collider component
- 3. add an Ak Surface Reflector.

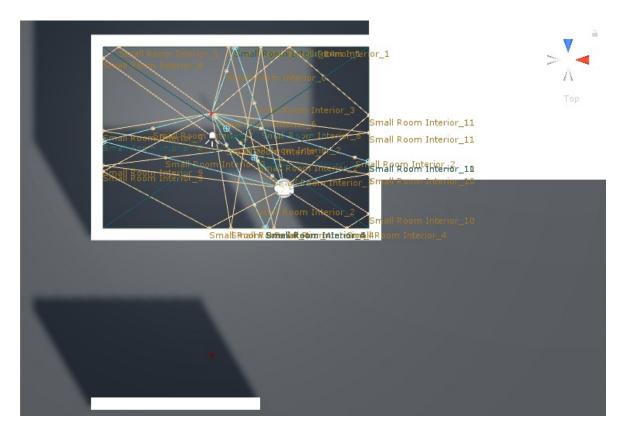
#### Start the game.

 Since we simplified the shape of our room, the door is now part of the reflective surface. We don't have any reflections from Button Small Room if we are standing outside. We even get a reflection from Button Outside on the door opening. You may have noticed the name of the reflection point is now referencing to Small Room Interior.



Top view of early reflection drawings for a new surface reflector volume when the emitter is outside

2. Entering the small room, Button Small Room's debug drawings appear and the ones for Button Outside disappear.



Top view of early reflection drawings for a new surface reflector volume when the emitter is inside

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3



Main Page Classes

**Wwise Unity Integration » Using Wwise Spatial Audio in Unity** 

#### **Rooms and Portals Tutorial**

This tutorial will show you how to use Rooms and Portals with Spatial Audio. The following sections detail each step in the tutorial:

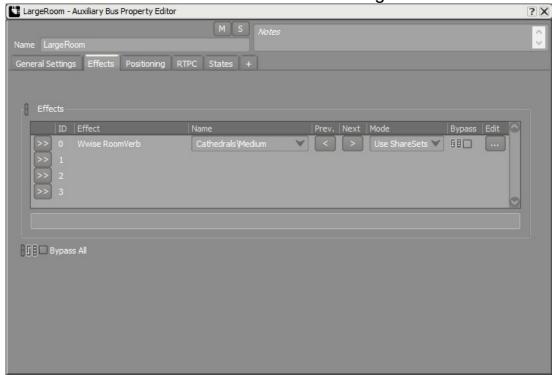
- A. Wwise Project
- B. Spatial Audio Emitter
- C. Rooms
- D. Portals
- E. Portals and Reverb
- F. Surface Reflectors and Reverb

Note: This tutorial presumes that you have completed the Preparation for the Spatial Audio Tutorials. However, it does not consider if the Surface Reflectors Tutorial was done; both tutorials are independent. If you did the Surface Reflectors Tutorial and wish to continue with this tutorial, you can do so without changing anything to your scene except for the optional step of C.2. Alternative: Create a New Volume, which should be reverted. If you wish to start anew, and you've done the Surface Reflectors Tutorial, you can still keep the same scene and just remove any Ak Surface Reflector components.

# A. Wwise Project

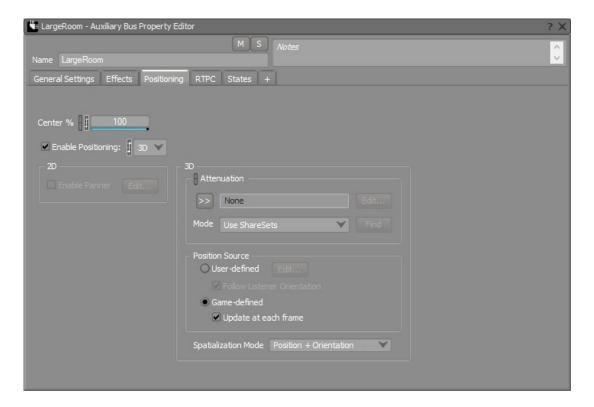
For the rooms and portals tutorial, we will need to add Reverb Effects and modify Obstruction curves to hear diffraction through a portal when loosing sight of an emitter.

- 1. Add two Auxiliary Busses under the **Master Audio Bus** called SmallRoom and LargeRoom. In the **Auxiliary Bus Property Editor**,
  - Under the Effects tab, add the Wwise RoomVerb effect. I chose two different factory presets: Rooms/Room\_Medium for SmallRoom and Cathedrals/Medium for LargeRoom.



**Auxiliary Bus Property Editor Effects tab for rooms** 

2. Under the **Positioning** tab, enable positioning and choose 3D.

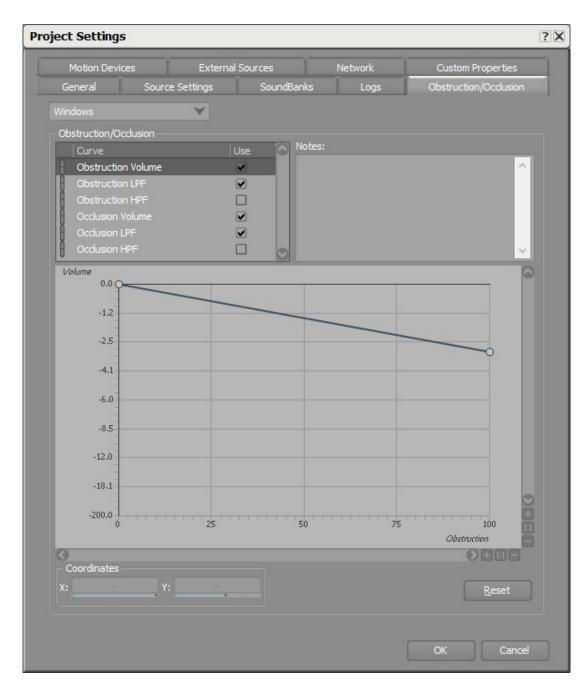


## **Auxiliary Bus Property Editor Positioning tab for rooms**

- 2. Project > Project Settings > Obstruction/Occlusion
  - 1. Change the curves:

Curve		Point 1	
	Х	Υ	
Obstruction Volume	0	0	
Obstruction LPF	0	0	

2. Here's how the Obstruction Volume curve looks like:



**Obstruction/Occlusion curves in the Wwise Project Settings** 

3. Save your project.

## **B. Spatial Audio Emitter**

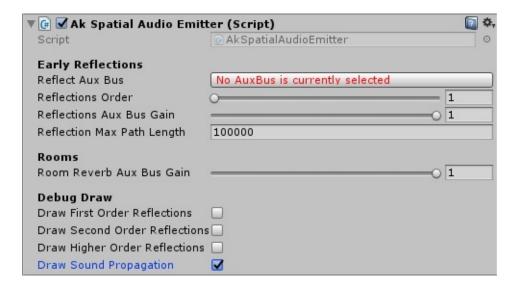
We need to modify the spatial audio emitter to use our newly created reflect aux bus.

- Refresh your Wwise project in the Wwise Picker: Windows > Wwise Picker
  - 1. Generate SoundBanks



**Wwise Picker** 

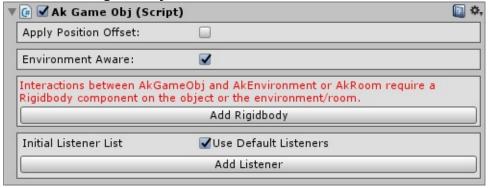
- 2. Modify the **Ak Spatial Audio Emitter** component of each button.
  - 1. Apply a gain of 1 for the **Room Reverb Aux Bus Gain** parameter.
  - 2. Check **Draw Sound Propagation** under **Debug Draw** and uncheck all the others.



**Ak Spatial Audio Emitter for Rooms and Portals** 

3. We will use rooms in the next section, so we need to add a Rigidbody component to our emitter; as explained in the warning message on the Ak Game Obj component.

1. Click on Add Rigidbody



Ak Game Obj component

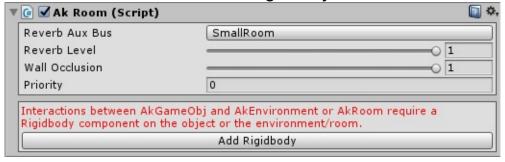
#### C. Rooms

Each of the rooms will have their own room reverb. To do so, we will use the Ak Room component.

Create unrendered meshes using cubes: GameObject > 3D Object > Cube

Object Name	Position	Rotation
Small Room Interior	(-2, 2.875, -6)	(0, 0, 0)
Large Room Interior	(11.5, 2.875, 3.25)	(0, 0, 0)

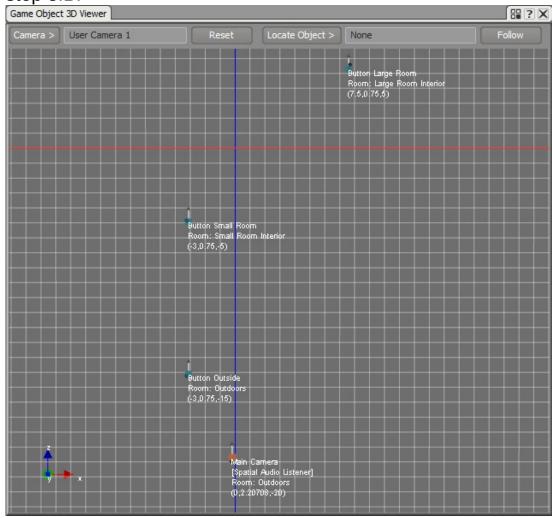
- 1. Remove the **Mesh Renderer** component.
- 2. Check the **Is Trigger** in the **Box Collider** component.
- 3. Add an **Ak Room** component to the Small and Large Room Interior Volumes.
  - 1. Pick the corresponding SmallRoom or LargeRoom Auxiliary Busses for **Reverb Aux Bus**.
  - 2. If you added a **Rigidbody** to your emitters, you don't need one here. If not, click on Add Rigidbody.



#### Ak Room component

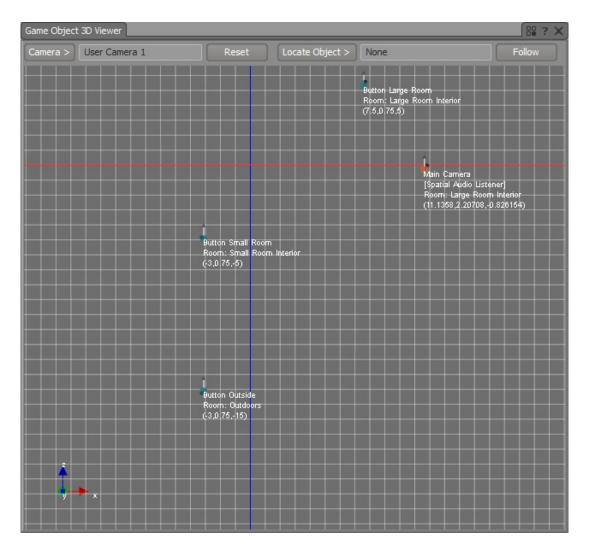
- 2. Start the game, connect to Wwise and go to the Game Object Profiler layout
  - 1. Watch all the emitters and the listener.
  - 2. Open a Game Object 3D Viewer view. You should see each emitter in their respective rooms. The next image is a top view of the scene. The listener and Button Outside are not in a room, so they are put by default in the "Outdoors" room. If everything is in the Outdoors room instead of their respective rooms, it means spatial audio is disabled. You must have forgotten to add

an Ak Spatial Audio Listener component to the listener. Follow step 3.B.



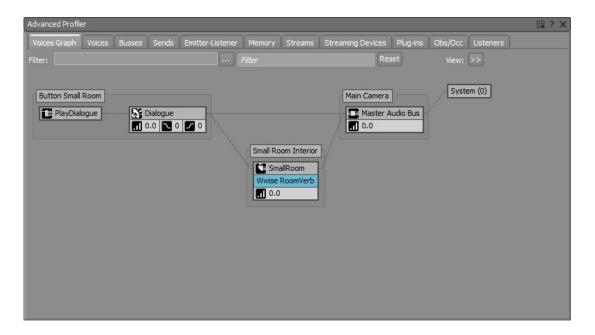
**Emitters in their respective rooms and the Listener Outdoors** 

3. Move the listener to a room and see the room name change under your listener game object in the Game Object 3d Viewer. In the following image, the listener is in the large room.



Emitters in their respective rooms and the Listener in the Large Room Interior

- 3. Press the button in one of the rooms to hear the reverb.
  - 1. In the small room, the voice graph should look like this:



#### **Button Small Room Voice Graph with Reverb**

2. When a listener is in a different room than the emitter, the sound is completely occluded. You will need to add portals to hear the sound coming from rooms through desired openings.

**Note:** To add a reverb outside, add a Room around the entire area. Make sure that the Room has a lower priority than the Small Room Interior and Large Room Interior.

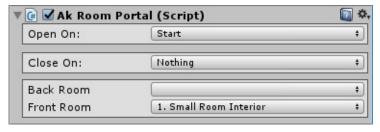
#### **D. Portals**

Each Room needs a Portal to be connected to another area. We need two Portals: one on the door between the outside area and the small Room and one between both Rooms.

 Create two Ak Room Portal components: GameObject > Wwise > Room Portal

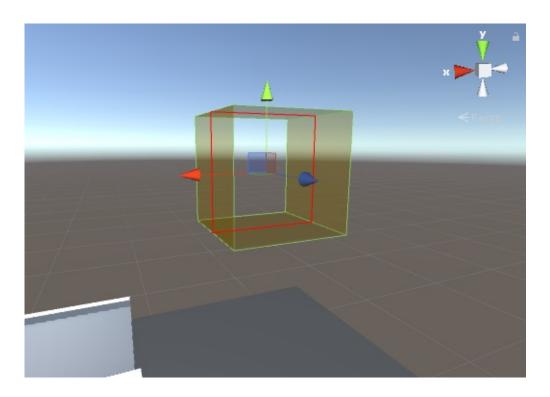
Object Name	Position	Rotation
Outside Portal	(0, 2, -10)	(0, 0, 0)
Inside Portal	(3.75, 2, -4.5)	(0, 90, 0)

- 2. In the Ak Room Portal component:
  - 1. **Open On** Start.
  - 2. Close On Nothing.
  - 3. Make sure that the Back Room and Front Room display the right **Ak Room** components. If not, refresh the inspector by deselecting and selecting the Portal again.
- 3. This is how the Inside Portal Ak Room Portal component will look like:



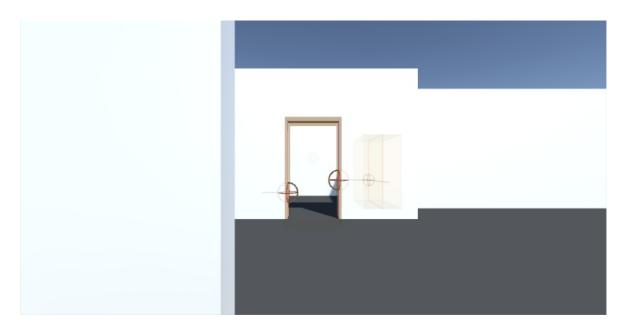
## **Ak Room Portal component**

4. In the **Scene** window, a Portal is represented with a yellow ribbon around it indicating its size and the orientation in which it should be placed. The red line shows the separation between front and back areas of the Portal. The front is in the same direction as the local z axis.



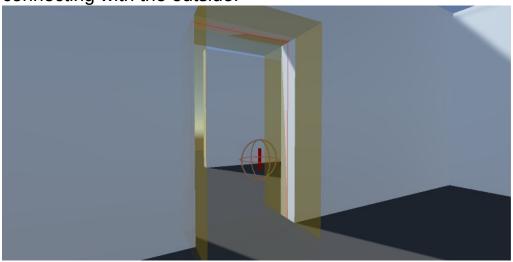
Ak Room Portal in the Scene window

5. Start the game. You should already see the Draw Sound Propagation drawings indicating where the sound will be coming from through the portal. The sphere on the left comes from the small room emitter and the one on the right from the large room emitter. You can see the latter is actually coming from the Portal connecting the small and large Rooms. The sound propagation is represented by red and blue spheres that change in size depending on the wet and dry diffraction angles respectively.



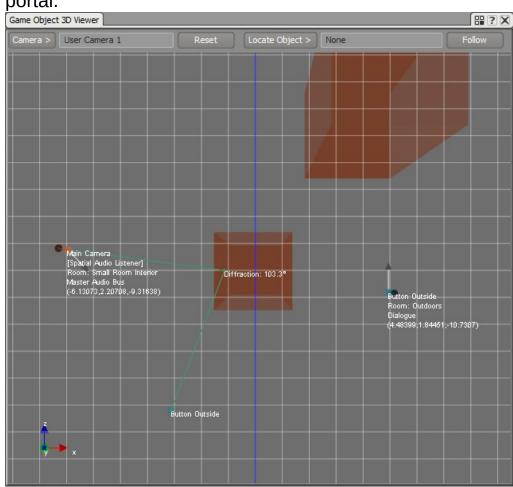
**Game window at Start** 

- 1. Connect to Wwise.
- 2. Play the button outside and go inside the small room.
  - 1. You should see the same sound propagation drawings coming from the outside emitter through the portal connecting with the outside.



Sound Propagation drawings from Button Outside when the listener is inside Small Room Interior

2. Move in and out of sight to hear the diffraction applied on the sound. You will hear more or less diffraction depending on the angle of the listener vs. the portal. In the next image, we are watching the listener and Button Outside. You can see a top view of the scene in the Game Object 3D Viewer. In dark red, we can see the portals. The one in the middle is the Outside Portal, the area on top of the Portal is the Small Room Interior and the area on the bottom is the outside. The Listener is in the Small Room Interior and the playing emitter is Button Outside. On the image, you can see the sound propagation in green between the listener and the emitter. The Diffraction angle is 103.3. Also, you can see the virtual position of the Button Outside emitter on the right. That's where the sound you hear will go through the portal.

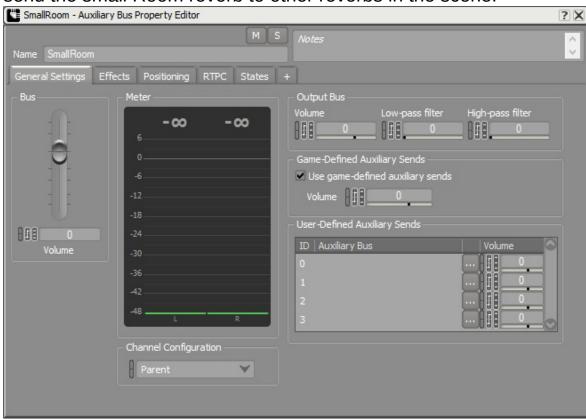


Portal diffraction angle on Game Object 3D Viewer

### E. Portals and Reverb

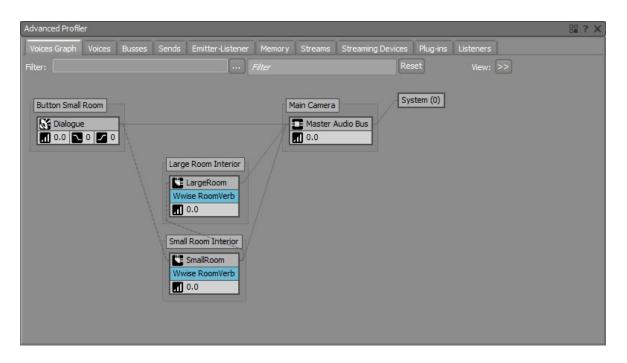
Sound emitted through Portals can also use the reverb from the current Room the listener is in.

1. Open the SmallRoom Auxiliary Bus Property Editor and check **Use game-defined auxiliary sends** in the General Settings tab. This will send the small Room reverb to other reverbs in the scene.



Auxiliary Bus Property Editor General Settings tab to send room reverb to other reverbs

2. When playing the small room emitter, and going into the large room, you will see the small room's reverb feeding into the large room's reverb.

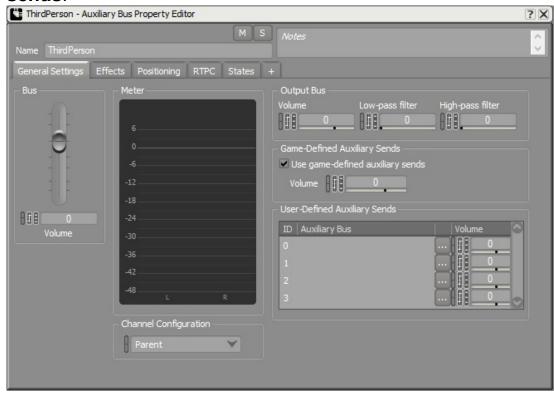


Button Small Room reverb sending to Large Room Interior room's reverb on the Voices Graph

### F. Surface Reflectors and Reverb

It is also possible to feed early reflections into the reverb of a Room. Let's add surface reflectors in our scene.

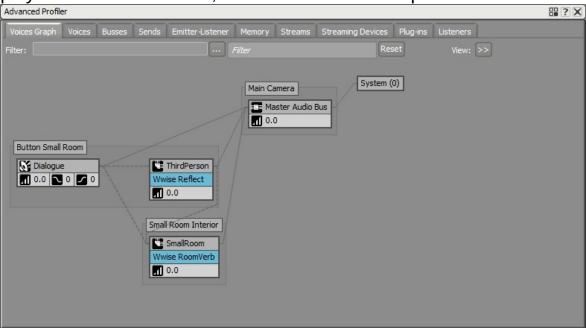
- 1. Following section Surface Reflectors Tutorial,
  - 1. Add an Auxiliary Bus with the Reflect Effect.
  - 2. Link each **Ak Spatial Audio Emitter** to the new Reflect Auxiliary Bus.
  - Add Ak Surface Reflector components on all the walls and the floor.
- 2. In the Wwise project, open the Auxiliary Bus with the Reflect Effect applied to it.
  - 1. In the General Settings tab, check **Use game-defined auxiliary** sends.



Auxiliary Bus Property Editor General Settings tab to send early reflection to the room reverb

3. Play the scene and press the button in the small Room. With the

player in the small Room, here is the Voices Graph we see:



Early reflections in the small room are sent to the small room's reverb in the Voices Graph

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3



Main Page Classes

**Wwise Unity Integration » Using Wwise Spatial Audio in Unity** 

## **Obstruction and Occlusion**

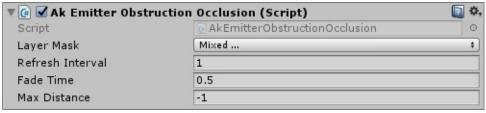
It is possible to obstruct or occlude sounds in Unity. See the following sections for details:

- A. Emitter Obstruction/Occlusion
- B. Portal Obstruction

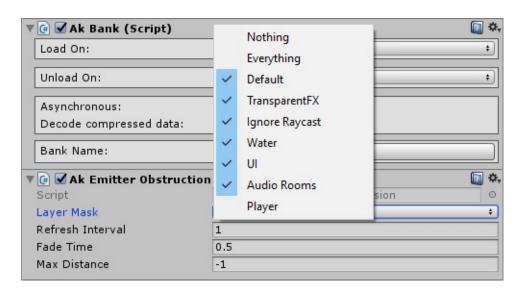
#### A. Emitter Obstruction/Occlusion

An emitter can be obstructed or occluded in Unity when the **Ak Emitter Obstruction Occlusion** component is added to its GameObject. When your map doesn't use spatial audio, that is it doesn't have any **Ak Room** components, the **Ak Emitter Obstruction Occlusion** component will only use occlusion. In a spatial audio scene, this same component will use obstruction with the spatial audio listener only; the GameObject with the **Ak Spatial Audio Listener** component. Emitter obstruction is applied to emitters that are in the same room as the spatial audio listener.

- Add an Ak Emitter Obstruction Occlusion component to Button Outside.
  - In Layer Mask, select layers that would block audio. In the SpatialAudioTutorial scene provided with the integration, we have put the first person character on a user created layer called "Player". This layer is not selected in the Layer Mask option.



## **Ak Emitter Obstruction Occlusion component**



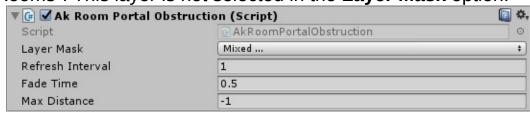
## Layer Mask options for the Ak Emitter Obstruction Occlusion component

- 2. Play the scene.
  - 1. Play Button Outside.
  - 2. Go behind the Barrier GameObject to hear it occluded or obstructed depending on the presence of any GameObjects with the **Ak Room** component in your scene.

#### **B.** Portal Obstruction

Sound emitted through portals can also be obstructed. Use the **Ak Room Portal Obstruction** component on a game object with an **Ak Room Portal** component to do so. See section **Rooms and Portals Tutorial** to add rooms and portals to your scene, if not done already.

- 1. Add an Ak Room Portal Obstruction component to Outside Portal.
  - 1. In **Layer Mask**, select layers that would block audio. In the SpatialAudioTutorial scene provided with the integration, we have put the rooms on a user created layer called "Audio Rooms". This layer is not selected in the **Layer Mask** option.



#### **Ak Room Portal Obstruction component**



Layer Mask options for the Ak Room Portal Obstruction component

- 2. Play the scene.
  - 1. Play Button Small Room.
  - 2. Go behind the Barrier GameObject to obstruct the portal.

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3

Main Page Classes

## **Wwise Unity Integration**

## **Troubleshooting**

- Compiling errors after installation
- DIINotFoundException in the Unity console
- There is no sound
- Can't connect the Wwise Profiler
- Error during installation Access to the path is denied in the Unity console
- Error during SoundBank generation on Wwise for Mac Cannot write to Wwise IDs.h
- Error: Plugin Not Registered
- "Multiple plugins with the same name" in the Unity Console on Mac
- "Failed to initialize the sound engine" using the Xbox One plugin

## **Compiling errors after installation**

If you have just upgraded Wwise from a previous version, you may have duplicate scripts. In version 2013.2.8, the UNITY\_PROJECT\_ROOT\Assets\Wwise\Deployment\Examples folder was renamed to components. If you haven't made any changes to the scripts in the Examples folder, you can simply delete it.

Also, if you develop on multiple platforms, make sure that you have installed the UnityPackage for each of your development platforms. There are some scripts that are platform-specific.

## **DIINotFoundException in the Unity console**

If you see a *DllNotFoundException* message about AkSoundEngine in the Unity console on Windows, this can be caused by two missing dependencies:

- Make sure you have DirectX installed on your machine.
- If you are using the Debug config of the Wwise-Unity Integration, you also need the *Microsoft Visual Studio 2010 Debug Redistributables* installed. Normally, the Profile configuration is installed by default, so this should not be an issue unless you manually installed the Debug config.

### There is no sound

If you hear no sound in your game, check the following:

- If there is no sound when you run the game in standalone, try to run your game in the Editor on Windows or Mac. It is easier to fix the issues in the editor first.
- Ensure your SoundBanks have been properly generated, and that Unity can find them. If you are running the game in the editor, the integration will look for the SoundBanks in the Generated SoundBanks path from your Wwise project. If none are found there, it will then look in the StreamingAssets folder. Note that if you chose to override the project SoundBank path in Wwise, the integration will not be able to load them automatically. For more information on the issue, see **Build your Unity Game for a Target Platform**
- Ensure that the Sound Engine is initialized properly before any other audio calls are made, and terminated after all engine calls are completed. You need to ensure the Script Execution Order (menu Edit > Project Settings > Script Execution Order) is properly set for AkInitializer and AkTerminator.
- Are there any errors reported in the Unity console? Fix them...
- If you connect the Wwise Profiler (F7, in Wwise) while running your game, are there errors reported in the Capture Log?
- In the Wwise Profiler, in the SoundBank tab can you see your bank(s) loaded? If the tab is not visible, open the Profiling Settings (Alt-G) and enable the SoundBanks. If they are not loaded, you should check when and how you load them. Look for AkBank components and AkSoundEngine.LoadBank calls.
- In the Wwise Profiler, in the Capture Log, can you see your expected Event posted? If not, you should check that your SoundBanks contain these Events and that they are properly loaded (see previous point). If the banks are loaded, you should check the AkEvent, AkAmbient components and AkSoundEngine.PostEvent calls in your game scripts.

## **Can't connect the Wwise Profiler**

If your game is running on Windows, check the following:

- Verify that the Run in background setting is enabled in Unity (menu File > Build Settings > Player Settings).
- Verify that the firewall on your machine allows port 24024 and that Wwise is set as an exception. If the firewall is disabled it is not an issue.
- Which configuration of the Wwise-Unity Plug-in are you running? Only Debug and Profile have the communication code enabled.
- Open the Task Manager, kill any process named "adb.exe". This is the Android driver and it locks TCP ports. This is true even if you don't develop on Android or have any Android device connected. (This is an issue on Google's side.)

If your game is running on another computer or console, check the following:

- Are your Wwise computer and game console on the same subnet? If not, the game will not appear in the Remote Connection dialog.
   Enter the IP manually with the connect To IP button.
- Is your network allowing traffic on port 24024 or dynamic port allocations?

## **Error during installation - Access to the path is denied in the Unity console**

If you see a *Wwise: Error during installation: Access to the path is denied* message in the Unity console, check the following:

- Did you try to install a new Plug-in configuration (Debug, Profile, Release) from the Assets > Wwise > Install Plugins menu right after:
  - installing the new Unity Integration from the wwise Setup window?
  - o previewing a Scene in the Unity Editor?
- Reopen your Unity project and install the new Plug-in configuration (Debug, Profile, Release) right away.

## **Error during SoundBank generation on Wwise for Mac - Cannot write to Wwise\_IDs.h**

If, using Wwise for Mac, you see an error during SoundBank generation about not being able to write to Wwise\_IDs.h, check the following:

- Open the Wwise project.
- In the project settings, go to the SoundBanks tab.
- Set the Header file path to where the SoundBanks are generated.

## **Error: Plugin Not Registered**

Your Wwise project uses a licensed plug-in which hasn't been included yet in your AkSoundEngine dynamic library. See **Adding Plug-ins**.

# "Multiple plugins with the same name" in the Unity Console on Mac

Using Unity for Mac, upon importing the Integration, the Unity Console might display many errors that say:

Multiple plug-ins with the same name 'aksoundengine' (found at 'Assets/Wwise/Deployment/Plugins/Mac/Release/AkSoundEngine.bu and

'Assets/Wwise/Deployment/Plugins/Mac/Debug/AkSoundEngine.bun That means one or more plug-ins are set to be compatible with Editor. Only one plug-in at a time can be used by Editor.

These errors can be safely ignored, as the installation process will properly activate the correct plug-ins.

## "Failed to initialize the sound engine" using the Xbox One plug-in

On the Xbox One, the Unity built-in audio system initializes the necessary hardware for audio, therefore Wwise's initialization fails. To fix the issue, disable the built-in audio by going to Assets > Project Settings > Audio, and selecting the "Disable audio" check box.

Note that because of a known Unity 5.0.2 issue, the audio hardware gets initialized even though the "Disable audio" check box is enabled. As a workaround for this, in AkInitializer.cs, change the following code:

```
AkPlatformInitSettings platformSettings = new Ak
PlatformInitSettings();
   AkSoundEngine.GetDefaultPlatformInitSettings(pla
tformSettings);
   platformSettings.uLEngineDefaultPoolSize = (uint
)lowerPoolSize * 1024;
   platformSettings.fLEngineDefaultPoolRatioThresho
ld = memoryCutoffThreshold;
```

#### To this:

```
AkPlatformInitSettings platformSettings = new Ak
PlatformInitSettings();
   AkSoundEngine.GetDefaultPlatformInitSettings(pla
tformSettings);
   platformSettings.uLEngineDefaultPoolSize = (uint
)lowerPoolSize * 1024;
   platformSettings.fLEngineDefaultPoolRatioThresho
ld = memoryCutoffThreshold;
#if UNITY_XBOXONE && !UNITY_EDITOR
        platformSettings.uMaxXMAVoices = 0;
#endif
```

Note that this will disable the XMA plug-in.

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3

Main Page Classes

Class List Class Index Class Hierarchy Class Members

## **Class List**

Here are the classes, structs, unions and interfaces with brief descriptions:

AK::Wwise::AcousticTexture
AK::Wwise::AuxBus
AK::Wwise::Bank
AK::Wwise::BaseGroupType
AK::Wwise::BaseType
AK::Wwise::CallbackFlags
AK::Wwise::Event
AK::Wwise::RTPC
AK::Wwise::State
AK::Wwise::Switch

K::Wwise::Trigger	
kAmbient	
Is A redical interner	
kAudioListener	

AkBank
AkCallbackManager
AkEmitterObstructionOcclusion
AkEnvironment

AkEnvironment::AkEnvironment_CompareBySelectionAlgorithm
AkEnvironmentPortal
AkEvent
AkEventCallbackMsg

AkGameObj
AkInitializer
AkMemBankLoader
AkRoom
AkRoomPortal
AkRoomPortalObstruction
AkSpatialAudioEmitter

AkSpatialAudioListener	
AkState	
AkSurfaceReflector	
AkSwitch	
AkTerminator	
AkTriggerBase	

Main Page Classes

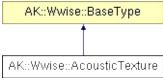
Class List Class Index Class Hierarchy Class Members

AK::Wwise::AcousticTexture

#### **AK::Wwise::AcousticTexture Class Reference**

This type represents an Acoustic Texture. More...

Inheritance diagram for AK::Wwise::AcousticTexture:



This type represents an Acoustic Texture.



Main Page Classes

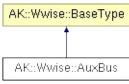
Class List Class Index Class Hierarchy Class Members

AK::Wwise::AuxBus

#### **AK::Wwise::AuxBus Class Reference**

This type represents an auxiliary send in the Master-Mixer Hierarchy. More...

Inheritance diagram for AK::Wwise::AuxBus:



This type represents an auxiliary send in the Master-Mixer Hierarchy.



Main Page Classes

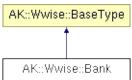
Class List Class Index Class Hierarchy Class Members

AK::Wwise::Bank

#### **AK::Wwise::Bank Class Reference**

This type can be used to load/unload SoundBanks. More...

Inheritance diagram for AK::Wwise::Bank:



This type can be used to load/unload SoundBanks.



Main Page Classes

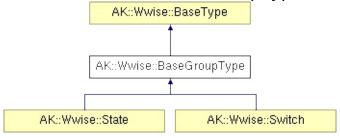
Class List Class Index Class Hierarchy Class Members

AK::Wwise::BaseGroupType

### AK::Wwise::BaseGroupType Class Reference

This type represents the base for all Wwise Types that also require a group GUID, such as **State** and **Switch**. More...

Inheritance diagram for AK::Wwise::BaseGroupType:



This type represents the base for all Wwise Types that also require a group GUID, such as State and Switch.

Main Page Classes

Class List Class Index Class Hierarchy Class Members

AK::Wwise::BaseType

### **AK::Wwise::BaseType Class Reference**

This type represents the base for all Wwise Types that require a GUID. More...

Inheritance diagram for AK::Wwise::BaseType:



This type represents the base for all Wwise Types that require a GUID.



Main Page Classes

Class List Class Index Class Hierarchy Class Members

AK::Wwise::CallbackFlags

# **AK::Wwise::CallbackFlags Class Reference**

This type represents the values of the flags used when posting an **Event** with a callback. More...

This type represents the values of the flags used when posting an **Event** with a callback.



Main Page Classes

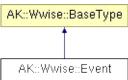
Class List Class Index Class Hierarchy Class Members

AK::Wwise::Event

#### **AK::Wwise::Event Class Reference**

This type can be used to post Events to the sound engine. More...

Inheritance diagram for AK::Wwise::Event:



# **Public Member Functions**

uint	Post (GameObject gameObject)
	Posts this <b>Event</b> on a GameObject.
uint	Post (GameObject gameObject, CallbackFlags flags, AkCallbackManager.EventCallback callback, object cookie=null)
	Posts this <b>Event</b> on a GameObject.
uint	<b>Post</b> (GameObject gameObject, uint flags, AkCallbackManager.EventCallback callback, object cookie=null)
	Posts this <b>Event</b> on a GameObject.
void	<b>ExecuteAction</b> (GameObject gameObject, AkActionOnEventType actionOnEventType, int transitionDuration, AkCurveInterpolation curveInterpolation)
	Executes various actions on this event associated with a GameObject.
void	PostMIDI (GameObject gameObject, AkMIDIPostArray array)
	Posts MIDI Events on this <b>Event</b> associated with a GameObject.
void	<b>PostMIDI</b> (GameObject gameObject, AkMIDIPostArray array, int count)
	Posts MIDI Events on this <b>Event</b> associated with a GameObject.
void	StopMIDI (GameObject gameObject)
	Stops MIDI Events on this <b>Event</b> associated with a GameObject.
void	StopMIDI ()
	Stops all MIDI Events on this <b>Event</b> .

This type can be used to post Events to the sound engine.



Main Page Classes

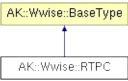
Class List Class Index Class Hierarchy Class Members

AK::Wwise::RTPC

#### **AK::Wwise::RTPC Class Reference**

This type can be used to set game parameter values to the sound engine. More...

Inheritance diagram for AK::Wwise::RTPC:



This type can be used to set game parameter values to the sound engine.



Main Page Classes

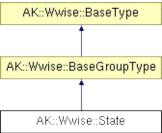
Class List Class Index Class Hierarchy Class Members

AK::Wwise::State

#### **AK::Wwise::State Class Reference**

This type can be used to set Wwise States. More...

Inheritance diagram for AK::Wwise::State:



This type can be used to set Wwise States.



Main Page Classes

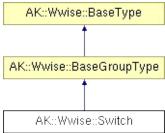
Class List Class Index Class Hierarchy Class Members

AK::Wwise::Switch

#### **AK::Wwise::Switch Class Reference**

This type can be used to set **Switch** values on gameobjects. More...

Inheritance diagram for AK::Wwise::Switch:



This type can be used to set **Switch** values on gameobjects.



Main Page Classes

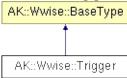
Class List Class Index Class Hierarchy Class Members

AK::Wwise::Trigger

# **AK::Wwise::Trigger Class Reference**

This type can be used to post triggers to the sound engine. More...

Inheritance diagram for AK::Wwise::Trigger:



This type can be used to post triggers to the sound engine.



Main Page Classes

Class List Class Index Class Hierarchy Class Members

#### **AkAmbient Class Reference**

Use this component to attach a Wwise Event to any object in a scene. The sound can be started at various moments, dependent on the selected Unity trigger. This component is more useful for ambient sounds (sounds related to scene-bound objects) but could also be used for other purposes. Since **AkAmbient** has **AkEvent** as its base class, it features the play/stop, play multiple, stop multiple and stop all buttons for previewing the associated Wwise event. More...

Inheritance diagram for AkAmbient:



## **Public Attributes**

int	eventID = 0
	ID of the Event as found in the WwiseID.cs file.
GameObject	soundEmitterObject = null
	Game object onto which the Event will be posted By default, when empty, it is posted on the same object on which the component was added.
bool	enableActionOnEvent = false
	Enables additional options to reuse existing events. Use it to transform a Play event into a Stevent without having to define one in the Wwise Project.
AkActionOnEventType	<pre>actionOnEventType = AkActionOnEventType.AkActionOnEventType_S</pre>
	Replacement action. See AK::SoundEngine::ExecuteEventOnAction().
AkCurveInterpolation	curveInterpolation = AkCurveInterpolation_Line
	Fade curve to use with the new Action. See AK::SoundEngine::ExecuteEventOnAction().
float	transitionDuration = 0.0f
	Duration of the fade. See AK::SoundEngine::ExecuteEventOnAction().
const int	MAX_NB_TRIGGERS = 32
	Since our mask is a 32 bits integer, we can't hav more than 32 triggers.
List< int >	<pre>triggerList = new List<int>() { START_TRIGGER_ID }</int></pre>
	List containing the enabled triggers.
bool	useOtherObject = false
	This property is usefull only when used with colliders. When enabled, the target of the action

will be the other colliding object. When disabled, will be the current object.

# **Static Public Attributes**

static Dictionary< uint, string >	<pre>triggerTypes = AkTriggerBase.GetAllDerivedTypes ()</pre>
	Will contain the types of all the triggers derived from <b>AkTriggerBase</b> at runtime.

Use this component to attach a Wwise Event to any object in a scene. The sound can be started at various moments, dependent on the selected Unity trigger. This component is more useful for ambient sounds (sounds related to scene-bound objects) but could also be used for other purposes. Since **AkAmbient** has **AkEvent** as its base class, it features the play/stop, play multiple, stop multiple and stop all buttons for previewing the associated Wwise event.

#### See also:

- How to use AkAmbient with the inspector
- AkGameObj
- AkEvent
- Integration Details Events (Note: This is described in the Wwise SDK documentation.)

Main Page Classes

Class List Class Index Class Hierarchy Class Members

#### **AkAudioListener Class Reference**

Add this script on the game object that represent a listener. This is normally added to the Camera object or the Player object, but can be added to any game object when implementing 3D busses. isDefaultListener determines whether the game object will be considered a default listener - a listener that automatically listens to all game objects that do not have listeners attached to their AkGameObjListenerList's. More...

1 10+	$\circ$ + $\circ$ 1	1 100 0 100	horo.
1 101 1	M AI	1 1111111	11010
ப்பட	oı aı	I mem	DCI 3.

Add this script on the game object that represent a listener. This is normally added to the Camera object or the Player object, but can be added to any game object when implementing 3D busses. isDefaultListener determines whether the game object will be considered a default listener - a listener that automatically listens to all game objects that do not have listeners attached to their AkGameObjListenerList's.

#### See also:

• Integrating Listeners (Note: This is described in the Wwise SDK documentation.)



Main Page Classes

Class List Class Index Class Hierarchy Class Members

#### **AkBank Class Reference**

Loads and unloads a SoundBank at a specified moment. Vorbis sounds can be decompressed at a specified moment using the decode compressed data option. In that case, the SoundBank will be prepared. More...

Inherits AkUnityEventHandler.

# **Public Member Functions**

override void	HandleEvent (GameObject in_gameObject)
	Loads the SoundBank.
void	<pre>UnloadBank (GameObject in_gameObject)</pre>

## **Public Attributes**

string	bankName = ""
	Name of the SoundBank, as specified in the Wwise project.
bool	loadAsynchronous = false
	Check this to load the SoundBank in the background. Be careful, if Events are triggered and the SoundBank hasn't finished loading, you'll have "Event not found" errors.
bool	decodeBank = false
	Decode this SoundBank upon load.
bool	saveDecodedBank = false
	Save the decoded SoundBank to disk for faster loads in the future.
List< int >	<pre>unloadTriggerList = new List<int>() {AkUnityEventHandler.DESTROY_TRIGGER_ID }</int></pre>
	Reserved.
const int	MAX_NB_TRIGGERS = 32
	Since our mask is a 32 bits integer, we can't have more than 32 triggers.
List< int >	<pre>triggerList = new List<int>() { START_TRIGGER_ID }</int></pre>
	List containing the enabled triggers.
bool	useOtherObject = false
	This property is usefull only when used with colliders. When enabled, the target of the action will be the other colliding object. When disabled, it will be the current object.

# **Static Public Attributes**

static Dictionary< uint, string >	<pre>triggerTypes = AkTriggerBase.GetAllDerivedTypes ()</pre>
	Will contain the types of all the triggers derived from <b>AkTriggerBase</b> at runtime.

Loads and unloads a SoundBank at a specified moment. Vorbis sounds can be decompressed at a specified moment using the decode compressed data option. In that case, the SoundBank will be prepared.



Main Page Classes

Class List Class Index Class Hierarchy Class Members

## **AkCallbackManager Class Reference**

This class manages the callback queue. All callbacks from the native Wwise SDK go through this queue. The queue needs to be driven by regular calls to **PostCallbacks()**. This is currently done in AkInitializer.cs, in LateUpdate(). More...

# **Static Public Member Functions**

static void	SetMonitoringCallback (AK.Monitor.ErrorLevel in_Level, MonitoringCallback in_CB)
	Call this to set a function to call whenever Wwise prints a message (warnings or errors).
static void	SetBGMCallback (BGMCallback in_CB, object in_cookie)
static int	PostCallbacks ()

This class manages the callback queue. All callbacks from the native Wwise SDK go through this queue. The queue needs to be driven by regular calls to PostCallbacks(). This is currently done in AkInitializer.cs, in LateUpdate().



Main Page Classes

Class List Class Index Class Hierarchy Class Members

# AkEmitterObstructionOcclusion Class Reference

Obstructs/Occludes the emitter of the current game object from its listeners if at least one object is between them. More...

Inherits AkObstructionOcclusion.

## **Public Attributes**

LayerMask	LayerMask = -1
	Indicates which layers act as obstructers/occluders.
float	refreshInterval = 1
	The number of seconds between obstruction/occlusion checks.
float	fadeTime = 0.5f
	The number of seconds for fade ins and fade outs.
float	maxDistance = -1.0f
	The maximum distance at which to perform obstruction/occlusion. A negative value will be interpreted as inifinite distance.

Obstructs/Occludes the emitter of the current game object from its listeners if at least one object is between them.

The current implementation does not support occlusion.

Main Page Classes

Class List Class Index Class Hierarchy Class Members

#### **AkEnvironment Class Reference**

Use this component to define a reverb zone. This needs to be added to a collider object to work properly. How to use AkEvironment and AkEvironmentPortal with the inspector (Reverb Zones). More...

# Classes

class AkEnvironment\_CompareBySelectionAlgorithm

Use this component to define a reverb zone. This needs to be added to a collider object to work properly. How to use AkEvironment and AkEvironmentPortal with the inspector (Reverb Zones).

This component can be attached to any collider. You can specify a roll-off to fade-in/out of the reverb. The reverb parameters will be defined in the Wwise project, by the sound designer. All **AkGameObj** that are "environment"-aware will receive a send value when entering the attached collider.

#### See also:

- How to use AkEvironment and AkEvironmentPortal with the inspector (Reverb Zones)
- Integrating Environments and Game-defined Auxiliary Sends (Note: This is described in the Wwise SDK documentation.)
- AK::SoundEngine::SetGameObjectAuxSendValues (Note: This is described in the Wwise SDK documentation.)

Main Page Classes

Class List Class Index Class Hierarchy Class Members

AkEnvironment::AkEnvironment\_CompareBySelectionAlgorithm

# AkEnvironment::AkEnvironment\_CompareBySel Class Reference

Inherits AkEnvironment::AkEnvironment\_CompareByPriority.

The selection algorithm is as follow:

- 1. Environments have priorities.
- 2. Environments have a "Default" flag. This flag effectively says that this environment will be bumped out if any other is present.
- 3. Environments have an "Exclude Other" flag. This flag will tell that this env is not overlappable with others. So, only one (the highest priority) should be selected.



Main Page Classes

Class List Class Index Class Hierarchy Class Members

#### **AkEnvironmentPortal Class Reference**

Use this component to define an area that straddles two different AkEnvironments zones and allow mixing between both zones. How to use AkEvironment and AkEvironmentPortal with the inspector (Reverb Zones). More...

#### **Public Member Functions**

float **GetAuxSendValueForPosition** (Vector3 in\_position, int index) The axis used to find the contribution of each environment.

# **Public Attributes**

Vector3 axis = new Vector3(1,0,0)

Use this component to define an area that straddles two different AkEnvironments zones and allow mixing between both zones. How to use AkEvironment and AkEvironmentPortal with the inspector (Reverb Zones).

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3



#### **AkEvent Class Reference**

Helper class that knows a Wwise Event and when to trigger it in Unity. As of 2017.2.0, the **AkEvent** inspector has buttons for play/stop, play multiple, stop multiple, and stop all. Play/Stop will play or stop the event such that it can be previewed both in edit mode and play mode. When multiple objects are selected, Play Multiple and Stop Multiple will play or stop the associated **AkEvent** for each object. More...

Inheritance diagram for AkEvent:



int	eventID = 0
	ID of the Event as found in the WwiseID.cs file.
GameObject	soundEmitterObject = null
	Game object onto which the Event will be posted By default, when empty, it is posted on the same object on which the component was added.
bool	enableActionOnEvent = false
	Enables additional options to reuse existing events. Use it to transform a Play event into a Stevent without having to define one in the Wwise Project.
AkActionOnEventType	<pre>actionOnEventType = AkActionOnEventType.AkActionOnEventType_5</pre>
	Replacement action. See AK::SoundEngine::ExecuteEventOnAction().
AkCurveInterpolation	curveInterpolation = AkCurveInterpolation_Line
	Fade curve to use with the new Action. See AK::SoundEngine::ExecuteEventOnAction().
float	transitionDuration = 0.0f
	Duration of the fade. See
	AK::SoundEngine::ExecuteEventOnAction().
const int	
	Since our mask is a 32 bits integer, we can't hav more than 32 triggers.
List< int >	<pre>triggerList = new List<int>() { START_TRIGGER_ID }</int></pre>
	List containing the enabled triggers.
bool	useOtherObject = false
	This property is usefull only when used with colliders. When enabled, the target of the action

will be the other colliding object. When disabled, will be the current object.

## **Static Public Attributes**

static Dictionary< uint, string >	<pre>triggerTypes = AkTriggerBase.GetAllDerivedTypes ()</pre>
	Will contain the types of all the triggers derived from <b>AkTriggerBase</b> at runtime.

Helper class that knows a Wwise Event and when to trigger it in Unity. As of 2017.2.0, the **AkEvent** inspector has buttons for play/stop, play multiple, stop multiple, and stop all. Play/Stop will play or stop the event such that it can be previewed both in edit mode and play mode. When multiple objects are selected, Play Multiple and Stop Multiple will play or stop the associated **AkEvent** for each object.

#### See also:

- Edit Mode Support
- How to use AkAmbient with the inspector
- Integration Details Events (Note: This is described in the Wwise SDK documentation.)

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3

# **AkEventCallbackMsg Class Reference**

Event callback information. Event callback functions can receive this structure as a parameter. More...

GameObject	sender
	AkSoundEngine.PostEvent callback flags. See the AkCallbackType enumeration for a list of all callbacks.
AkCallbackInfo	info
	GameObject from whom the callback function was called.

Event callback information. Event callback functions can receive this structure as a parameter.

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3



## **AkGameObj Class Reference**

This component represents a sound object in your scene tracking its position and other game syncs such as Switches, RTPC and environment values. You can add this to any object that will emit sound, and it will be added to any object that an **AkAudioListener** is attached to. Note that if it is not present, Wwise will add it automatically, with the default values, to any Unity Game Object that is passed to Wwise. More...

# **Public Member Functions**

bool	AddListener (AkAudioListener listener)
	Adds an <b>AkAudioListener</b> to the container of listeners listening to this gameobject.
bool	RemoveListener (AkAudioListener listener)
	Removes an <b>AkAudioListener</b> from the container of listeners listening to this gameobject.
virtual Vector3	GetPosition ()
virtual Vector3	GetForward ()
virtual Vector3	GetUpward ()

AkGameObjPositionOffsetData	m_positionOffsetData = null When not set to null, the position will be offset relative to the Game Object position by the Position Offset.
bool	isEnvironmentAware = true
	Is this object affected by Environment changes? Set to false if not affected in order to save some useless calls. Default is true.

This component represents a sound object in your scene tracking its position and other game syncs such as Switches, RTPC and environment values. You can add this to any object that will emit sound, and it will be added to any object that an **AkAudioListener** is attached to. Note that if it is not present, Wwise will add it automatically, with the default values, to any Unity Game Object that is passed to Wwise.

#### See also:

- Integration Details Game Objects (Note: This is described in the Wwise SDK documentation.)
- Integration Details Events (Note: This is described in the Wwise SDK documentation.)
- Integrating Listeners (Note: This is described in the Wwise SDK documentation.)
- Integration Details Switches (Note: This is described in the Wwise SDK documentation.)
- Integration Details States (Note: This is described in the Wwise SDK documentation.)
- Integration Details Environments and Game-defined Auxiliary Sends (Note: This is described in the Wwise SDK documentation.)

## **AkInitializer Class Reference**

string	<b>basePath</b> = AkSoundEngineController.s_DefaultBas Path for the soundbanks. This must contain one sub per platform, with the same as in the Wwise project.
string	language = AkSoundEngineController.s_Language Language sub-folder.
int	defaultPoolSize = AkSoundEngineController.s_DefaultPoolSize
	Default Pool size. This contains the meta data for yo project. Default size is 4 MB, but you should adjust for needs.
int	<pre>lowerPoolSize = AkSoundEngineController.s_Lowe</pre>
	Lower Pool size. This contains the audio processing and DSP data. Default size is 2 MB, but you should a for your needs.
int	<pre>streamingPoolSize = AkSoundEngineController.s_StreamingPoolSize</pre>
	Streaming Pool size. This contains the streaming but Default size is 1 MB, but you should adjust for your r
int	<pre>preparePoolSize = AkSoundEngineController.s_PreparePoolSize</pre>
	Prepare Pool size. This contains the banks loaded use PrepareBank (Banks decoded on load use this). Defis 0 MB, but you should adjust for your needs.
float	memoryCutoffThreshold = AkSoundEngineController.s_MemoryCutoffThreshold
	This setting will trigger the killing of sounds when the memory is reaching 95% of capacity. Lowest priority are killed.
int	monitorPoolSize = AkSoundEngineController.s_MonitorPoolSize
	Monitor Pool size. Size of the monitoring pool, in byte parameter is not used in Release build.

int	monitorQueuePoolSize = AkSoundEngineController.s_MonitorQueuePoolSize Monitor Queue Pool size. Size of the monitoring que in bytes. This parameter is not used in Release build
int	callbackManagerBufferSize = AkSoundEngineController.s_CallbackManagerBuffer CallbackManager buffer size. The size of the buffer to per-frame to transfer callback data. Default size is 4 you should increase this, if required.
int	<pre>spatialAudioPoolSize = AkSoundEngineController.s_SpatialAudioPoolSize</pre>
	Spatial Audio Lower Pool size. Default size is 4 MB, should adjust for your needs.
uint	maxSoundPropagationDepth = AkSoundEngine.AK_MAX_SOUND_PROPAGATION Spatial Audio Max Sound Propagation Depth. Maxim number of rooms that sound can propagate through; less than or equal to AK_MAX_SOUND_PROPAGATION_DEPTH.
AkDiffractionFlags	diffractionFlags = AkDiffractionFlags.DefaultDiffractionFlags.DefaultDiffractionFlags.DefaultDiffractionFlags.See
	Enable or disable specific diffraction features. See AkDiffractionFlags.
bool	<pre>engineLogging = AkSoundEngineController.s_EngineLogging Enable Wwise engine logging. Option to turn on/off t logging of the Wwise engine.</pre>

This script deals with initialization, and frame updates of the Wwise audio engine. It is marked as DontDestroyOnLoad so it stays active for the life of the game, not only one scene. You can, and probably should, modify this script to change the initialization parameters for the sound engine. A few are already exposed in the property inspector. It must be present on one Game Object at the beginning of the game to initialize the audio properly. It must be executed BEFORE any other MonoBehaviors that use AkSoundEngine.

#### See also:

- Initialize the Different Modules of the Sound Engine (Note: This is described in the Wwise SDK documentation.)
- AK::SoundEngine::Init() (Note: This is described in the Wwise SDK documentation.)
- AK::SoundEngine::Term() (Note: This is described in the Wwise SDK documentation.)
- AkCallbackManager

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3

## **AkMemBankLoader Class Reference**

# **Public Member Functions**

void	LoadNonLocalizedBank (string in_bankFilename)	
	Load a sound bank from WWW object.	
void	LoadLocalizedBank (string in_bankFilename)	

string	bankName = ""	
	Name of the bank to load.	
bool	isLocalizedBank = false	
	Is the bank localized (situated in the language specific folders).	

This class is an example of how to load banks in Wwise, if the bank data was preloaded in memory. This would be useful for situations where you use the WWW class

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3



### **AkRoom Class Reference**

An **AkRoom** is an enclosed environment that can only communicate to the outside/other rooms with AkRoomPortals. More...

## **Public Member Functions**

ulong GetID ()

Access the room's ID.

AK.Wwise.AuxBus	reverbAuxBus
	The reverb auxiliary bus.
float	reverbLevel = 1
	The reverb control value for the send to the reverb aux bus.
float	wallOcclusion = 1
	Occlusion level modeling transmission through walls.
int	priority = 0

An AkRoom is an enclosed environment that can only communicate to the outside/other rooms with AkRoomPortals.

Generated on Mon Jan 8 10:46:14 2018 for Wwise Unity Integration by 1.6.3



#### **AkRoomPortal Class Reference**

An **AkRoomPortal** can connect two **AkRoom** components together. More...

Inherits AkUnityEventHandler.

# **Public Member Functions**

ulong	GetID ()	
	Access the portal's ID.	
override void	HandleEvent (GameObject in_gameObject)	
	Opens the portal on trigger event.	
void	ClosePortal (GameObject in_gameObject)	
	Closes the portal on trigger event.	

const int	MAX_ROOMS_PER_PORTAL = 2	
	AkRoomPortals can only connect a maximum of 2 rooms.	
AkRoom[]	rooms = new AkRoom[MAX_ROOMS_PER_PORTAL]	
const int	MAX_NB_TRIGGERS = 32	
	Since our mask is a 32 bits integer, we can't have more than 32 triggers.	
List< int >	<pre>triggerList = new List<int>() { START_TRIGGER_ID }</int></pre>	
	List containing the enabled triggers.	
bool	useOtherObject = false	
	This property is usefull only when used with colliders. When enabled, the target of the action will be the other colliding object. When disabled, it will be the current object.	

## **Static Public Attributes**

static Dictionary< uint, string >	<pre>triggerTypes = AkTriggerBase.GetAllDerivedTypes ()</pre>
	Will contain the types of all the triggers derived from <b>AkTriggerBase</b> at runtime.

An AkRoomPortal can connect two AkRoom components together.



#### **AkRoomPortalObstruction Class Reference**

Obstructs/Occludes the spatial audio portal of the current game object from the spatial audio listener if at least one object is between them. More...

Inherits AkObstructionOcclusion.

List of all members.

## **Public Attributes**

LayerMask	LayerMask = -1
	Indicates which layers act as obstructers/occluders.
float	refreshInterval = 1
	The number of seconds between obstruction/occlusion checks.
float	fadeTime = 0.5f
	The number of seconds for fade ins and fade outs.
float	maxDistance = -1.0f
	The maximum distance at which to perform obstruction/occlusion. A negative value will be interpreted as inifinite distance.

Obstructs/Occludes the spatial audio portal of the current game object from the spatial audio listener if at least one object is between them.

If no spatial audio listener has been registered, there will be no obstruction.



# **AkSpatialAudioEmitter Class Reference**

Add this script on the GameObject which represents an emitter that uses the Spatial Audio API. More...

Inherits AkSpatialAudioBase.

List of all members.

## **Public Attributes**

AK.Wwise.AuxBus	reflectAuxBus
	The Auxiliary Bus with a Reflect plug-in Effect applied.
uint	reflectionsOrder = 1
float	reflectionsAuxBusGain = 1
	The gain [0, 1] applied to the reflect auxiliary bus.
float reflectionMaxPathLength = 1000	
	The maximum path length a sound path can have from the emitter to the listener after reflecting on surfaces.
float	roomReverbAuxBusGain = 1
	Send gain (0.f-1.f) that is applied when sending to the aux bus associated with the room that the emitter is in.

Add this script on the GameObject which represents an emitter that uses the Spatial Audio API.



#### **AkSpatialAudioListener Class Reference**

Add this script on the game object that represent a listener. This is normally added to the Camera object or the Player object, but can be added to any game object when implementing 3D busses. isDefaultListener determines whether the game object will be considered a default listener - a listener that automatically listens to all game objects that do not have listeners attached to their AkGameObjListenerList's. More...

Inherits AkSpatialAudioBase.

List of all members.

Add this script on the game object that represent a listener. This is normally added to the Camera object or the Player object, but can be added to any game object when implementing 3D busses. isDefaultListener determines whether the game object will be considered a default listener - a listener that automatically listens to all game objects that do not have listeners attached to their AkGameObjListenerList's.

#### See also:

• Integrating Listeners (Note: This is described in the Wwise SDK documentation.)



#### **AkState Class Reference**

This will call AkSoundEngine.SetState() whenever the selected Unity event is triggered. For example this component could be set on a Unity collider to trigger when an object enters it. More...

Inherits AkUnityEventHandler.

List of all members.

# **Public Attributes**

int	groupID		
	State Group ID, as defined in WwiseID.cs.		
int	valueID		
	State Value ID, as defined in WwiseID.cs.		
const int	st int MAX_NB_TRIGGERS = 32		
	Since our mask is a 32 bits integer, we can't have more than 32 triggers.		
List< int >	<pre>triggerList = new List<int>() { START_TRIGGER_ID }</int></pre>		
	List containing the enabled triggers.		
bool	useOtherObject = false		
	This property is usefull only when used with colliders. When enabled, the target of the action will be the other colliding object. When disabled, it will be the current object.		

## **Static Public Attributes**

static Dictionary< uint, string >	<pre>triggerTypes = AkTriggerBase.GetAllDerivedTypes ()</pre>
	Will contain the types of all the triggers derived from <b>AkTriggerBase</b> at runtime.

This will call AkSoundEngine.SetState() whenever the selected Unity event is triggered. For example this component could be set on a Unity collider to trigger when an object enters it.

#### See also:

• Integration Details - States (Note: This is described in the Wwise SDK documentation.)

#### **AkSurfaceReflector Class Reference**

This component will convert the triangles of the GameObject's geometry into sound reflective surfaces. More...

List of all members.

# **Static Public Member Functions**

static void	AddGeometrySet (AK.Wwise.AcousticTexture acousticTexture, MeshFilter meshFilter)		
	Sends the mesh filter's triangles and their acoustic texture to Spatial Audio.		
static void	RemoveGeometrySet (MeshFilter meshFilter)		
	Remove the corresponding mesh filter's geometry from Spatial Audio.		

## **Public Attributes**

AK.Wwise.AcousticTexture	AcousticTexture
	All triangles of the component's mesh will be applied with this texture. The texture will change the filter parameters of the sound reflected from this
	component.

This component will convert the triangles of the GameObject's geometry into sound reflective surfaces.

This component requires a Mesh Filter component. The triangles of the mesh will be sent to the Spatial Audio wrapper by calling SpatialAudio::AddGeometrySet(). The triangles will reflect the sound emitted from AkSpatialAudioEmitter components.



#### **AkSwitch Class Reference**

This will call AkSoundEngine.SetSwitch() whenever the selected Unity event is triggered. For example this component could be set on a Unity collider to trigger when an object enters it. More...

Inherits AkUnityEventHandler.

List of all members.

# **Public Attributes**

int	groupID Switch Group ID, as defined in WwiseID.cs.		
int	valueID		
	Switch Value ID, as defined in WwiseID.cs.		
const int	MAX_NB_TRIGGERS = 32		
	Since our mask is a 32 bits integer, we can't have more than 32 triggers.		
List< int >	<pre>triggerList = new List<int>() { START_TRIGGER_ID }</int></pre>		
	List containing the enabled triggers.		
bool	useOtherObject = false		
	This property is usefull only when used with colliders. When enabled, the target of the action will be the other colliding object. When disabled, it will be the current object.		

## **Static Public Attributes**

static Dictionary< uint, string >	<pre>triggerTypes = AkTriggerBase.GetAllDerivedTypes ()</pre>
	Will contain the types of all the triggers derived from <b>AkTriggerBase</b> at runtime.

This will call AkSoundEngine.SetSwitch() whenever the selected Unity event is triggered. For example this component could be set on a Unity collider to trigger when an object enters it.

#### See also:

• Integration Details - Switches (Note: This is described in the Wwise SDK documentation.)



## **AkTerminator Class Reference**

i.	:-4	-4 -	II .aa		م بر م ما	
П	IST	or a	II M	iem	hers.	

This script deals with termination of the Wwise audio engine. It must be present on one Game Object that gets destroyed last in the game. It must be executed AFTER any other monoBehaviors that use AkSoundEngine.

#### See also:

- Terminate the Different Modules of the Sound Engine (Note: This is described in the Wwise SDK documentation.)
- AK::SoundEngine::Term() (Note: This is described in the Wwise SDK documentation.)

## **AkTriggerBase Class Reference**

Inherited by AkTriggerCollisionEnter, AkTriggerCollisionExit, AkTriggerDisable, AkTriggerEnable, AkTriggerEnter, AkTriggerExit, AkTriggerMouseDown, AkTriggerMouseEnter, AkTriggerMouseExit, and AkTriggerMouseUp.

List of all members.

### **Public Member Functions**

delegate void Trigger (GameObject in\_gameObject)
Delegate declaration for all Wwise Triggers.

## **Public Attributes**

Trigger	triggerDelegate = null
	All components reacting to the trigger will be registered in this delegate.

Base class for the generic triggering mechanism for Wwise Integration. All Wwise components will use this mechanism to drive their behavior. Derive from this class to add your own triggering condition, as decribed in **Adding New Triggers for Wwise Events** 



# **Class Hierarchy**

This inheritance list is sorted roughly, but not completely, alphabetically:

- AK::Wwise::BaseType
  - AK::Wwise::AcousticTexture
  - AK::Wwise::AuxBus
  - AK::Wwise::Bank
  - AK::Wwise::BaseGroupType
    - AK::Wwise::State
    - AK::Wwise::Switch
  - AK::Wwise::Event
  - AK::Wwise::RTPC
  - AK::Wwise::Trigger
- AK::Wwise::CallbackFlags
- AkAudioListener
- AkBank
- AkCallbackManager
- AkEmitterObstructionOcclusion
- AkEnvironment
- AkEnvironment::AkEnvironment\_CompareBySelectionAlgorithm
- AkEnvironmentPortal
- AkEvent
  - AkAmbient
- AkEventCallbackMsg
- AkGameObj
- AkInitializer
- AkMemBankLoader
- AkRoom
- AkRoomPortal
- AkRoomPortalObstruction
- AkSpatialAudioEmitter
- AkSpatialAudioListener
- AkState
- AkSurfaceReflector
- AkSwitch
- AkTerminator
- AkTriggerBase



Here is a list of all documented class members with links to the class documentation for each member:

#### - a -

AcousticTexture : AkSurfaceReflector

actionOnEventType : AkEvent

AddGeometrySet(): AkSurfaceReflector

AddListener(): AkGameObjaxis: AkEnvironmentPortal

### - b -

• bankName : AkBank , AkMemBankLoader

• basePath : AkInitializer

#### - C -

• callbackManagerBufferSize : AkInitializer

ClosePortal(): AkRoomPortalcurveInterpolation: AkEvent

### - d -

decodeBank : AkBank

defaultPoolSize : AkInitializerdiffractionFlags : AkInitializer

#### - e -

• enableActionOnEvent : AkEvent

• engineLogging : AkInitializer

- eventID : AkEvent
- ExecuteAction(): AK::Wwise::Event

## *- g -*

- GetAuxSendValueForPosition(): AkEnvironmentPortal
- GetForward(): AkGameObj
- GetID(): AkRoomPortal, AkRoom
- GetPosition(): AkGameObj
- GetUpward(): AkGameObj
- groupID : AkSwitch , AkState

#### - h -

• HandleEvent(): AkBank, AkRoomPortal

## - j -

- info: AkEventCallbackMsg
- isEnvironmentAware : AkGameObj
- isLocalizedBank : AkMemBankLoader

## - 1 -

- language : AkInitializer
- loadAsynchronous : AkBank
- LoadLocalizedBank(): AkMemBankLoader
- LoadNonLocalizedBank(): AkMemBankLoader
- lowerPoolSize : AkInitializer

#### - m -

- m\_positionOffsetData : AkGameObj
- MAX\_ROOMS\_PER\_PORTAL : AkRoomPortal
- maxSoundPropagationDepth : AkInitializer
- memoryCutoffThreshold : AkInitializer
- monitorPoolSize : AkInitializer
- monitorQueuePoolSize : AkInitializer

### - p -

Post(): AK::Wwise::Event

PostCallbacks(): AkCallbackManager

PostMIDI(): AK::Wwise::Event
 preparePoolSize: AkInitializer

• priority : AkRoom

#### - r -

• reflectAuxBus : AkSpatialAudioEmitter

• reflectionMaxPathLength : AkSpatialAudioEmitter

• reflectionsAuxBusGain : AkSpatialAudioEmitter

• reflectionsOrder : AkSpatialAudioEmitter

RemoveGeometrySet(): AkSurfaceReflector

RemoveListener(): AkGameObj

reverbAuxBus : AkRoomreverbLevel : AkRoom

roomReverbAuxBusGain : AkSpatialAudioEmitter

• rooms : AkRoomPortal

#### - S -

• saveDecodedBank : AkBank

• sender : AkEventCallbackMsq

• SetBGMCallback(): AkCallbackManager

• SetMonitoringCallback(): AkCallbackManager

soundEmitterObject : AkEvent

• spatialAudioPoolSize : AkInitializer

• StopMIDI(): AK::Wwise::Event

• streamingPoolSize : AkInitializer

#### - t -

• transitionDuration : AkEvent

Trigger(): AkTriggerBase

• triggerDelegate : AkTriggerBase

#### - u -

UnloadBank() : AkBankunloadTriggerList : AkBank

#### - V -

• valueID : AkState , AkSwitch

#### - W -

• wallOcclusion : AkRoom

**Main Page** Classes

AK::Wwise::Event

**ExecuteAction** 

Post

**Post** 

**Post** 

**PostMIDI** 

**PostMIDI** 

**StopMIDI** 

**StopMIDI** 

uint AK::Wwise::Event::Post (GameObject game

Posts this **Event** on a GameObject.

**Parameters:** 

gameObject The GameObject

**Returns:** 

Returns the playing ID.



**Main Page** Classes

AK::Wwise::Event

**ExecuteAction** 

**Post** 

**Post** 

**Post** 

PostMIDI

**PostMIDI** 

**StopMIDI** 

**StopMIDI** 

void AK::Wwise::Event::PostMIDI (GameObject

**AkMIDIPostAr** 

Posts MIDI Events on this **Event** associated with a G

**Parameters:** 

gameObject The GameObject

The array of AkMIDIPost that are p array



#### **AkInitializer**

## basePath

callbackManagerBufferSize
defaultPoolSize
diffractionFlags
engineLogging
language
lowerPoolSize
maxSoundPropagationDepth
memoryCutoffThreshold
monitorPoolSize
monitorQueuePoolSize
preparePoolSize
spatialAudioPoolSize
streamingPoolSize

# string AkInitializer::basePath = Ak

Path for the soundbanks. This must cosame as in the Wwise project.

**Main Page** Classes

## AkCallbackManager

**PostCallbacks SetBGMCallback** SetMonitoringCallback static void AkCallbackManager::SetMoni

Call this to set a function to call whenever W



**Main Page** Classes

### **AkMemBankLoader**

bankName **isLocalizedBank** LoadLocalizedBank LoadNonLocalizedBank

void AkMemBankLoader::LoadNonLoca

Load a sound bank from WWW object.



**Main Page** 

Classes

## AkCallbackManager

**PostCallbacks** SetBGMCallback SetMonitoringCallback static void AkCallbackManager::SetBGM

Call this to set a iOS callback interruption fur



# **Class Index**

# A|B|C|E|R|S|T

AcousticTexture (AK::Wwise)

**AkAmbient** 

**AkAudioListener** 

AkBank

AkCallbackManager

**AkEmitterObstructionOcclusion** 

**AkEnvironment** 

AkEnvironment::AkEnvironment\_CompareBySelectionAlgorithm

A|B|C|E|R|S|T

Main Page | Classes

# AkCallbackManager

PostCallbacks
SetBGMCallback
SetMonitoringCallback

static int AkCallbackManager::PostCallba

This function dispatches all the accumulated sound engine. It must be called regularly. By AkInitializer.cs.

# **AK::Wwise::AcousticTexture Member List**

This is the complete list of members for AK::Wwise::AcousticTexture, including all inherited members.

# **AK::Wwise::AuxBus Member List**

This is the complete list of members for AK::Wwise::AuxBus, including all inherited members.

# **AK::Wwise::Bank Member List**

This is the complete list of members for AK::Wwise::Bank, including all inherited members.

# **AK::Wwise::BaseGroupType Member List**

This is the complete list of members for AK::Wwise::BaseGroupType, including all inherited members.

# **AK::Wwise::BaseType Member List**

This is the complete list of members for AK::Wwise::BaseType, including all inherited members.

# **AK::Wwise::CallbackFlags Member List**

This is the complete list of members for AK::Wwise::CallbackFlags, including all inherited members.

# **AK::Wwise::Event Member List**

This is the complete list of members for **AK::Wwise::Event**, including all inherited members.

<b>ExecuteAction</b> (GameObject gameObject, AkActionOnEventType actionOnEventType, int transitionDuration, AkCurveInterpolation curveInterpolation)	AK::Wwise::Event [inline]
Post(GameObject gameObject)	AK::Wwise::Event [inline]
Post(GameObject gameObject, CallbackFlags flags, AkCallbackManager.EventCallback callback, object cookie=null)	AK::Wwise::Event [inline]
Post(GameObject gameObject, uint flags, AkCallbackManager.EventCallback callback, object cookie=null)	AK::Wwise::Event [inline]
PostMIDI(GameObject gameObject, AkMIDIPostArray array)	AK::Wwise::Event [inline]
PostMIDI(GameObject gameObject, AkMIDIPostArray array, int count)	AK::Wwise::Event [inline]
StopMIDI(GameObject gameObject)	AK::Wwise::Event [inline]
StopMIDI()	AK::Wwise::Event [inline]

**ExecuteAction** 

**Post** 

Post

**Post** 

**PostMIDI** 

**PostMIDI** 

**StopMIDI** 

**StopMIDI** 

uint AK::Wwise::Event::Post ( GameObject

**CallbackFlags** 

AkCallbackManage object

Posts this **Event** on a GameObject.

**Parameters:** 

gameObject The GameObject

flags

callback

cookie Optional cookie received by the ca

**Returns:** 

Returns the playing ID.

**ExecuteAction** 

**Post** 

**Post** 

Post

**PostMIDI** 

**PostMIDI** 

**StopMIDI** 

**StopMIDI** 

uint AK::Wwise::Event::Post ( GameObject

uint

AkCallbackManage

object

Posts this **Event** on a GameObject.

**Parameters:** 

gameObject The GameObject

flags

callback

cookie Optional cookie received by the ca

**Returns:** 

Returns the playing ID.

## ExecuteAction

**Post** 

Post

**Post** 

**PostMIDI** 

**PostMIDI** 

**StopMIDI** 

**StopMIDI** 

void AK::Wwise::Event::ExecuteAction ( GameOl AkActio

int

**AkCurve** 

Executes various actions on this event associated wi

#### **Parameters:**

gameObject The GameObject actionOnEventType transitionDuration curveInterpolation

**ExecuteAction** 

**Post** 

**Post** 

**Post** 

**PostMIDI** 

PostMIDI

**StopMIDI** 

**StopMIDI** 

void AK::Wwise::Event::PostMIDI ( GameObject

**AkMIDIPostAr** 

int

Posts MIDI Events on this **Event** associated with a G

**Parameters:** 

gameObject The GameObject

The array of AkMIDIPost that are r array

The number of elements from the count

**Main Page** Classes

AK::Wwise::Event

**ExecuteAction** 

**Post** 

**Post** 

**Post** 

**PostMIDI** 

**PostMIDI** 

StopMIDI

**StopMIDI** 

void AK::Wwise::Event::StopMIDI ( GameObject

Stops MIDI Events on this **Event** associated with a G

**Parameters:** 

gameObject The GameObject



Main Page | Classes

AK::Wwise::Event

**ExecuteAction** 

**Post** 

**Post** 

**Post** 

**PostMIDI** 

**PostMIDI** 

**StopMIDI** 

StopMIDI

void AK::Wwise::Event::StopMIDI( ) [inline]

Stops all MIDI Events on this **Event**.

## **AK::Wwise::RTPC Member List**

This is the complete list of members for **AK::Wwise::RTPC**, including all inherited members.

## **AK::Wwise::State Member List**

This is the complete list of members for **AK::Wwise::State**, including all inherited members.

## **AK::Wwise::Switch Member List**

This is the complete list of members for **AK::Wwise::Switch**, including all inherited members.

# **AK::Wwise::Trigger Member List**

This is the complete list of members for AK::Wwise::Trigger, including all inherited members.

## **AkAmbient Member List**

This is the complete list of members for AkAmbient, including all inherited members.

actionOnEventType	AkEvent
curvelnterpolation	AkEvent
enableActionOnEvent	AkEvent
eventID	AkEvent
soundEmitterObject	AkEvent
transitionDuration	AkEvent



actionOnEventType curveInterpolation **enableActionOnEvent** eventID

soundEmitterObject transitionDuration

int AkEvent::eventID = 0

ID of the Event as found in the WwiseID.cs file.



actionOnEventType curveInterpolation **enableActionOnEvent** eventID

soundEmitterObject transitionDuration

**GameObject AkEvent::soundEmitterObject** 

Game object onto which the Event will be pos default, when empty, it is posted on the same on which the component was added.



actionOnEventType curvelnterpolation <u>enableActionOnEvent</u> eventID soundEmitterObject transitionDuration

bool AkEvent::enableActionOnEvent = fal

Enables additional options to reuse existing events. Use it to transform a Play event into a Stop event without having to define one in the Wwise Project.



actionOnEventTypecurveInterpolation **enableActionOnEvent** eventID soundEmitterObject transitionDuration

AkActionOnEventType AkEvent::actionOr

Replacement action. See AK::SoundEngine::



actionOnEventType curveInterpolation **enableActionOnEvent** eventID soundEmitterObject transitionDuration

**AkCurveInterpolation AkEvent::curveInter** 

Fade curve to use with the new Action. See A



actionOnEventType curveInterpolation **enableActionOnEvent** eventID soundEmitterObject transitionDuration

**float AkEvent::transitionDuration = 0.0f** 

Duration of the fade. See

AK::SoundEngine::ExecuteEventOnAction().



**Main Page** Classes

### **AkSwitch**

groupID valueID

## const int AkUnityEventHandler::MAX\_NB\_TRIGGERS =

Since our mask is a 32 bits integer, we can't have more than



Main Page | Classes

### **AkSwitch**

groupID valueID

# List<int> AkUnityEventHandler::triggerList = new List<i

List containing the enabled triggers.

### **AkSwitch**

# bool AkUnityEventHandler::useOtherObject = false [int

groupID valueID

This property is usefull only when used with colliders. When enabled, the target of the action will be the other colliding ob When disabled, it will be the current object.

**Main Page** 

Classes

### **AkSwitch**

groupID valueID

Dictionary<uint, string> AkUnityEventHandler::triggerTy

Will contain the types of all the triggers derived from AkTrige



## **AkAudioListener Member List**

This is the complete list of members for AkAudioListener, including all inherited members.



## **AkBank Member List**

This is the complete list of members for **AkBank**, including all inherited members.

bankName	<b>AkBank</b>	
decodeBank	<b>AkBank</b>	
HandleEvent(GameObject in_gameObject)	<b>AkBank</b>	[inline]
loadAsynchronous	<b>AkBank</b>	
saveDecodedBank	<b>AkBank</b>	
<pre>UnloadBank(GameObject in_gameObject)</pre>	<b>AkBank</b>	[inline]
unloadTriggerList	AkBank	

**Main Page** Classes

#### **AkBank**

bankName decodeBank HandleEvent **loadAsynchronous** saveDecodedBank UnloadBank unloadTriggerList

override void AkBank::HandleEvent (GameC

Loads the SoundBank.



**Main Page** Classes

#### **AkBank**

bankName decodeBank **HandleEvent loadAsynchronous** saveDecodedBank UnloadBank unload Trigger List

void AkBank::UnloadBank (GameObject in\_

Unloads a SoundBank.



## bankName

decodeBank **HandleEvent loadAsynchronous** saveDecodedBank UnloadBank unloadTriggerList

## string AkBank::bankName = ""

Name of the SoundBank, as specified in the Wwise project.



bankName decodeBank **HandleEvent** loadAsynchronous saveDecodedBank **UnloadBank** unloadTriggerList

**bool** AkBank::loadAsynchronous = false

Check this to load the SoundBank in the background. Be careful, if Events are triggered and the SoundBank hasn't finished loading, you'll have "Event not found" errors.



bankName decodeBank **HandleEvent loadAsynchronous** saveDecodedBank UnloadBank

unload Trigger List

bool AkBank::decodeBank = false

Decode this SoundBank upon load.



bankName decodeBank **HandleEvent loadAsynchronous** saveDecodedBank **UnloadBank** unloadTriggerList

bool AkBank::saveDecodedBank = false

Save the decoded SoundBank to disk for faster loads in the future.



**Main Page** 

Classes

### **AkBank**

bankName decodeBank **HandleEvent loadAsynchronous** saveDecodedBank UnloadBank  ${\sf unloadTriggerList}$ 

List<int> AkBank::unloadTriggerList = new L

Reserved.



# **AkCallbackManager Member List**

This is the complete list of members for **AkCallbackManager**, including all inherited members.

	AkCallbackManager	0 00
<b>SetBGMCallback</b> (BGMCallback in_CB, object in_cookie)	AkCallbackManager	[in sta
<b>SetMonitoringCallback</b> (AK.Monitor.ErrorLevel in_Level, MonitoringCallback in_CB)	AkCallbackManager	[in sta

# **AkEmitterObstructionOcclusion Member List**

This is the complete list of members for AkEmitterObstructionOcclusion, including all inherited members.

Main Page Classes

### **AkRoomPortalObstruction**

LayerMask AkObstructionOcclusion::LayerMask = -1 [inherited]

Indicates which layers act as obstructers/occluders.

**Main Page** Classes

### **AkRoomPortalObstruction**

float AkObstructionOcclusion::refreshInterval = 1 [inherited]

The number of seconds between obstruction/occlusion checks.

**Main Page** Classes

### **AkRoomPortalObstruction**

float AkObstructionOcclusion::fadeTime = 0.5f [inherited]

The number of seconds for fade ins and fade outs.



**Main Page** Classes

### **AkRoomPortalObstruction**

# float AkObstructionOcclusion::maxDistance = -1.0f [inherited]

The maximum distance at which to perform obstruction/occlusion. A negative value will be interpreted as inifinite distance.



Main Page Classes

Class List Class Index Class Hierarchy Class Members

# **AkEnvironment Member List**

This is the complete list of members for AkEnvironment, including all inherited members.

Main Page Classes

Class List Class Index Class Hierarchy Class Members

# AkEnvironment::AkEnvironment\_CompareBySel **Member List**

This is the complete list of members for AkEnvironment::AkEnvironment\_CompareBySelectionAlgorithm, including all inherited members.



Main Page Classes

Class List Class Index Class Hierarchy Class Members

# **AkEnvironmentPortal Member List**

This is the complete list of members for AkEnvironmentPortal, including all inherited members.

	AkEnvironmentPortal	
<b>GetAuxSendValueForPosition</b> (Vector3 in_position, int index)	AkEnvironmentPortal	[inline]



**Main Page** 

Classes

### **AkEnvironmentPortal**

axis

**GetAuxSendValueForPosition** 

float AkEnvironmentPortal::GetAt

The axis used to find the contribution



### **AkEnvironmentPortal**



**GetAuxSendValueForPosition** 

### **Vector3 AkEnvironmentPortal::axi**

The array is already sortet by position is on the negative side of the portal(c of the chosen axis) The second envir positive side of the portal



Main Page Classes

Class List Class Index Class Hierarchy Class Members

# **AkEvent Member List**

This is the complete list of members for AkEvent, including all inherited members.

actionOnEventType	AkEvent
curveInterpolation	AkEvent
enableActionOnEvent	AkEvent
eventID	AkEvent
soundEmitterObject	AkEvent
transitionDuration	AkEvent



Main Page Classes

Class List Class Index Class Hierarchy Class Members

# **AkEventCallbackMsg Member List**

This is the complete list of members for AkEventCallbackMsg, including all inherited members.

**AkEventCallbackMsg** sender AkEventCallbackMsg



**Main Page** Classes

# AkEventCallbackMsg

# **GameObject AkEventCallbackMsg::sender**



AkSoundEngine.PostEvent callback flags. See the AkCallbackType enumeration for a list of all callbacks.



**Main Page** Classes

# AkEventCallbackMsg



# AkCallbackInfo AkEventCallbackMsg::info

GameObject from whom the callback function was called.



Main Page Classes

Class List Class Index Class Hierarchy Class Members

# **AkGameObj Member List**

This is the complete list of members for AkGameObj, including all inherited members.

AddListener(AkAudioListener listener)	AkGameObj [inline]
GetForward()	AkGameObj [inline, virtual]
GetPosition()	AkGameObj [inline, virtual]
GetUpward()	AkGameObj [inline, virtual]
isEnvironmentAware	AkGameObj
m_positionOffsetData	AkGameObj
RemoveListener(AkAudioListener listener)	AkGameObj [inline]



AddListener

**GetForward** 

**GetPosition** 

**GetUpward** 

**isEnvironmentAware** m\_positionOffsetData

RemoveListener

bool AkGameObj::AddListener ( AkAudiol

Adds an **AkAudioListener** to the container of gameobject.

**Parameters:** 

listener

**Returns:** 

Returns true if the listener was not previo otherwise.



AddListener
GetForward
GetPosition
GetUpward
isEnvironmentAware
m\_positionOffsetData
RemoveListener

bool AkGameObj::RemoveListener ( AkAu

Removes an **AkAudioListener** from the cont gameobject.

#### **Parameters:**

listener

#### **Returns:**

Returns true if the listener was previously

AddListener

**GetForward** 

GetPosition

**GetUpward** 

**isEnvironmentAware** m\_positionOffsetData

RemoveListener

virtual Vector3 AkGameObj::GetPosition (

Gets the position including the position offset, applyPositionOffset is enabled. User can also to calculate an arbitrary position.

#### **Returns:**

The position.



#### AddListener

GetForward

**GetPosition** 

**GetUpward** 

**isEnvironmentAware** m\_positionOffsetData

RemoveListener

## virtual Vector3 AkGameObj::GetForward (

Gets the orientation forward vector. User can method to calculate an arbitrary vector.

#### **Returns:**

The forward vector of orientation.



AddListener

**GetForward** 

**GetPosition** 

GetUpward

**isEnvironmentAware** m\_positionOffsetData

RemoveListener

virtual Vector3 AkGameObj::GetUpward (

Gets the orientation upward vector. User can method to calculate an arbitrary vector.

#### **Returns:**

The upward vector of orientation.



**AddListener GetForward GetPosition GetUpward isEnvironmentAware** 

m\_positionOffsetData

RemoveListener

AkGameObjPositionOffsetData AkGameO

When not set to null, the position will be offse position by the Position Offset.



**AddListener GetForward GetPosition GetUpward** 

isEnvironmentAware m\_positionOffsetData RemoveListener

**bool** AkGameObj::isEnvironmentAware =

Is this object affected by Environment change Set to false if not affected in order to save sor useless calls. Default is true.



Main Page Classes

Class List Class Index Class Hierarchy Class Members

# **AkInitializer Member List**

This is the complete list of members for **AkInitializer**, including all inherited members.

basePath	AkInitializer
callbackManagerBufferSize	<b>AkInitializer</b>
defaultPoolSize	AkInitializer
diffractionFlags	AkInitializer
engineLogging	AkInitializer
language	AkInitializer
lowerPoolSize	AkInitializer
maxSoundPropagationDepth	<b>AkInitializer</b>
memoryCutoffThreshold	<b>AkInitializer</b>
monitorPoolSize	<b>AkInitializer</b>
monitorQueuePoolSize	AkInitializer
preparePoolSize	AkInitializer
spatialAudioPoolSize	AkInitializer
streamingPoolSize	AkInitializer

basePath
callbackManagerBufferSize
defaultPoolSize
diffractionFlags
engineLogging
language
lowerPoolSize
maxSoundPropagationDepth
memoryCutoffThreshold
monitorPoolSize
monitorQueuePoolSize
preparePoolSize
spatialAudioPoolSize

streamingPoolSize

string AkInitializer::language = Ak\$

Language sub-folder.

basePath

call back Manager Buffer Size

defaultPoolSize

diffractionFlags

engineLogging

language

**IowerPoolSize** 

maxSoundPropagationDepth

memoryCutoffThreshold

monitorPoolSize

monitorQueuePoolSize

preparePoolSize

spatialAudioPoolSize

streamingPoolSize

int AkInitializer::defaultPoolSize =

Default Pool size. This contains the m MB, but you should adjust for your ne

basePath

call back Manager Buffer Size

defaultPoolSize

diffractionFlags

engineLogging

language

**lowerPoolSize** 

maxSoundPropagationDepth

 ${\bf memory Cutoff Threshold}$ 

monitorPoolSize

monitorQueuePoolSize

preparePoolSize

**spatialAudioPoolSize** 

streamingPoolSize

int Aklnitializer::lowerPoolSize = A

Lower Pool size. This contains the au size is 2 MB, but you should adjust for

basePath
callbackManagerBufferSize
defaultPoolSize
diffractionFlags
engineLogging
language
lowerPoolSize
maxSoundPropagationDepth
memoryCutoffThreshold
monitorPoolSize
monitorQueuePoolSize
preparePoolSize
spatialAudioPoolSize
streamingPoolSize

int Aklnitializer::streamingPoolSize

Streaming Pool size. This contains the should adjust for your needs.

basePath callbackManagerBufferSize defaultPoolSize diffractionFlags engineLogging language **IowerPoolSize** maxSoundPropagationDepth memoryCutoffThreshold monitorPoolSize monitorQueuePoolSize preparePoolSize **spatialAudioPoolSize** streamingPoolSize

int AkInitializer::preparePoolSize =

Prepare Pool size. This contains the t on load use this). Default size is 0 MB



basePath
callbackManagerBufferSize
defaultPoolSize
diffractionFlags
engineLogging
language
lowerPoolSize
maxSoundPropagationDepth
memoryCutoffThreshold
monitorPoolSize
monitorQueuePoolSize
preparePoolSize
spatialAudioPoolSize
streamingPoolSize

## float AkInitializer::memoryCutoffTl

This setting will trigger the killing of so sounds are killed.

basePath
callbackManagerBufferSize
defaultPoolSize
diffractionFlags
engineLogging
language
lowerPoolSize
maxSoundPropagationDepth
memoryCutoffThreshold
monitorPoolSize
monitorQueuePoolSize
preparePoolSize
spatialAudioPoolSize
streamingPoolSize

int AkInitializer::monitorPoolSize =

Monitor Pool size. Size of the monitor Release build.

basePath callbackManagerBufferSize defaultPoolSize diffractionFlags engineLogging language **IowerPoolSize** maxSoundPropagationDepth memoryCutoffThreshold monitorPoolSize monitorQueuePoolSize preparePoolSize spatialAudioPoolSize streamingPoolSize

# int AkInitializer::monitorQueuePoo

Monitor Queue Pool size. Size of the Release build.



#### basePath

callbackManagerBufferSize

defaultPoolSize diffractionFlags engineLogging

language

**IowerPoolSize** 

maxSoundPropagationDepth

 ${\bf memory Cutoff Threshold}$ 

monitorPoolSize

monitorQueuePoolSize

preparePoolSize

**spatialAudioPoolSize** 

streamingPoolSize

int Aklnitializer::callbackManagerB

CallbackManager buffer size. The size KB, but you should increase this, if re

basePath callbackManagerBufferSize defaultPoolSize diffractionFlags engineLogging language **IowerPoolSize** maxSoundPropagationDepth memoryCutoffThreshold monitorPoolSize monitorQueuePoolSize preparePoolSize **spatialAudioPoolSize** streamingPoolSize

int AkInitializer::spatialAudioPoolS

Spatial Audio Lower Pool size. Defaul



basePath
callbackManagerBufferSize
defaultPoolSize
diffractionFlags
engineLogging
language
lowerPoolSize
maxSoundPropagationDepth
memoryCutoffThreshold
monitorPoolSize
monitorQueuePoolSize
preparePoolSize
spatialAudioPoolSize
streamingPoolSize

uint AkInitializer::maxSoundPropa

Spatial Audio Max Sound Propagation be less than or equal to AK\_MAX\_SC

basePath callbackManagerBufferSize defaultPoolSize

diffractionFlags

engineLogging

language

**IowerPoolSize** 

maxSoundPropagationDepth

memoryCutoffThreshold

monitorPoolSize

monitorQueuePoolSize

preparePoolSize

**spatialAudioPoolSize** 

streamingPoolSize

AkDiffractionFlags AkInitializer::di

Enable or disable specific diffraction for

basePath

call back Manager Buffer Size

defaultPoolSize

diffractionFlags

engineLogging

language

**IowerPoolSize** 

maxSoundPropagationDepth

memoryCutoffThreshold

monitorPoolSize

monitorQueuePoolSize

preparePoolSize

spatialAudioPoolSize

streamingPoolSize

bool AkInitializer::engineLogging =

Enable Wwise engine logging. Option

Main Page Classes

Class List Class Index Class Hierarchy Class Members

# **AkMemBankLoader Member List**

This is the complete list of members for AkMemBankLoader, including all inherited members.

bankName	AkMemBankLoader	
isLocalizedBank	AkMemBankLoader	
<b>LoadLocalizedBank</b> (string in_bankFilename)	AkMemBankLoader [inline]	ne]
<b>LoadNonLocalizedBank</b> (string in_bankFilename)	AkMemBankLoader [inline]	ne]



### **AkMemBankLoader**

bankName **isLocalizedBank** LoadLocalizedBank LoadNonLocalizedBank

void AkMemBankLoader::LoadLocalized

Load a language-specific bank from WWW



### **AkMemBankLoader**

bankName

**isLocalizedBank** LoadLocalizedBank LoadNonLocalizedBank string AkMemBankLoader::bankName =

Name of the bank to load.



### **AkMemBankLoader**

bankName

isLocalizedBank

LoadLocalizedBank LoadNonLocalizedBank bool AkMemBankLoader::isLocalizedBa

Is the bank localized (situated in the langua specific folders).



Main Page Classes

Class List Class Index Class Hierarchy Class Members

# **AkRoom Member List**

This is the complete list of members for AkRoom, including all inherited members.

GetID()	AkRoom	[inline]
priority	AkRoom	
reverbAuxBus	AkRoom	
reverbLevel	AkRoom	
wallOcclusion	AkRoom	



### **AkRoom**

GetID priority reverbAuxBus reverbLevel wallOcclusion

ulong AkRoom::GetID() [inline]

Access the room's ID.



#### **AkRoom**

**GetID** priority reverbAuxBus reverbLevel wallOcclusion

AK.Wwise.AuxBus AkRoom::reverbAuxBus

The reverb auxiliary bus.



### **AkRoom**

**GetID** priority reverbAuxBus reverbLevel wallOcclusion

float AkRoom::reverbLevel = 1

The reverb control value for the send to the reverb aux bus.



### **AkRoom**

**GetID** priority reverbAuxBus reverbLevel wallOcclusion

float AkRoom::wallOcclusion = 1

Occlusion level modeling transmission through walls.



#### **AkRoom**

**GetID** priority reverbAuxBus reverbLevel wallOcclusion

### int AkRoom::priority = 0

In cases where a game object is in an area with two rooms, the higher priority room will be chosen for AK::SpatialAudio::SetGameObjectInRoom() The higher the priority number, the higher the priority of a room.



Main Page Classes

Class List Class Index Class Hierarchy Class Members

# **AkRoomPortal Member List**

This is the complete list of members for AkRoomPortal, including all inherited members.

ClosePortal(GameObject in_gameObject)	<b>AkRoomPortal</b>	[inline]
GetID()	<b>AkRoomPortal</b>	[inline]
HandleEvent(GameObject in_gameObject)	<b>AkRoomPortal</b>	[inline]
MAX_ROOMS_PER_PORTAL	<b>AkRoomPortal</b>	
rooms	<b>AkRoomPortal</b>	



### **AkRoomPortal**

ClosePortal
GetID
HandleEvent
MAX\_ROOMS\_PER\_PORTAL
rooms

ulong AkRoomPortal::GetID() [i

Access the portal's ID.

### **AkRoomPortal**

ClosePortal

**GetID** 

HandleEvent

MAX\_ROOMS\_PER\_PORTAL

rooms

override void AkRoomPortal::Hanc

Opens the portal on trigger event.



### **AkRoomPortal**

ClosePortal
GetID
HandleEvent
MAX\_ROOMS\_PER\_PORTAL
rooms

void AkRoomPortal::ClosePortal (

Closes the portal on trigger event.

#### **AkRoomPortal**

ClosePortal
GetID
HandleEvent
MAX\_ROOMS\_PER\_PORTAL
rooms

const int AkRoomPortal::MAX\_RO

AkRoomPortals can only connect a m

#### **AkRoomPortal**

ClosePortal
GetID
HandleEvent
MAX\_ROOMS\_PER\_PORTAL
rooms

# AkRoom [] AkRoomPortal::rooms

The front and back rooms connected side of the portal(opposite to the direc the positive side of the portal.

Main Page Classes

Class List Class Index Class Hierarchy Class Members

# **AkRoomPortalObstruction Member List**

This is the complete list of members for AkRoomPortalObstruction, including all inherited members.

Main Page Classes

Class List Class Index Class Hierarchy Class Members

# **AkSpatialAudioEmitter Member List**

This is the complete list of members for AkSpatialAudioEmitter, including all inherited members.

reflectAuxBus **AkSpatialAudioEmitter** reflectionMaxPathLength AkSpatialAudioEmitter reflectionsAuxBusGain **AkSpatialAudioEmitter** reflectionsOrder **AkSpatialAudioEmitter** roomReverbAuxBusGain AkSpatialAudioEmitter



### AkSpatialAudioEmitter

# reflectAuxBus

reflection MaxPathLengthreflectionsAuxBusGain reflectionsOrder roomReverbAuxBusGain

### AK.Wwise.AuxBus AkSpatialAudioEmi

The Auxiliary Bus with a Reflect plug-in Ef



### **AkSpatialAudioEmitter**

reflectAuxBus reflectionMaxPathLength reflectionsAuxBusGain reflectionsOrder roomReverbAuxBusGain

## uint AkSpatialAudioEmitter::reflection:

The maximum number of reflections that v processed for a sound path before it reach listener. Reflection processing grows expc with the order of reflections, so this number be kept low. Valid range: 1-4.



### AkSpatialAudioEmitter

reflectAuxBus reflection MaxPathLengthreflectionsAuxBusGain reflectionsOrder roomReverbAuxBusGain

float AkSpatialAudioEmitter::reflection

The gain [0, 1] applied to the reflect auxilia



### AkSpatialAudioEmitter

reflectAuxBus reflectionMaxPathLengthreflectionsAuxBusGain reflectionsOrder roomReverbAuxBusGain

float AkSpatialAudioEmitter::reflectior

The maximum path length a sound path c to the listener after reflecting on surfaces.



### AkSpatialAudioEmitter

reflectAuxBus reflection MaxPathLengthreflectionsAuxBusGain reflectionsOrder roomReverbAuxBusGain

float AkSpatialAudioEmitter::roomRev

Send gain (0.f-1.f) that is applied when se associated with the room that the emitter i



Main Page Classes

Class List Class Index Class Hierarchy Class Members

# **AkSpatialAudioListener Member List**

This is the complete list of members for AkSpatialAudioListener, including all inherited members.

Main Page Classes

Class List Class Index Class Hierarchy Class Members

# **AkState Member List**

This is the complete list of members for AkState, including all inherited members.

groupID AkState valueID AkState



**Main Page** Classes

## **AkState**



# int AkState::groupID

State Group ID, as defined in WwiseID.cs.



**Main Page** Classes

## **AkState**



# int AkState::valueID

State Value ID, as defined in WwiseID.cs.



Main Page Classes

Class List Class Index Class Hierarchy Class Members

# **AkSurfaceReflector Member List**

This is the complete list of members for **AkSurfaceReflector**, including all inherited members.

	AkSurfaceReflector	
<b>AddGeometrySet</b> (AK.Wwise.AcousticTexture acousticTexture, MeshFilter meshFilter)	AkSurfaceReflector	[inli stati
RemoveGeometrySet(MeshFilter meshFilter)	AkSurfaceReflector	[inli stati

## **AkSurfaceReflector**

static void AkSurfaceReflector::AddGeom

**AcousticTexture** AddGeometrySet RemoveGeometrySet

Sends the mesh filter's triangles and their aco

### **Parameters:**

acousticTexture meshFilter



Main Page | Classes

## **AkSurfaceReflector**

AcousticTexture AddGeometrySet RemoveGeometrySet static void AkSurfaceReflector::RemoveG

Remove the corresponding mesh filter's geom

**Parameters:** 

meshFilter

## AkSurfaceReflector

AcousticTexture
AddGeometrySet
RemoveGeometrySet

## AK.Wwise.AcousticTexture AkSurfaceRefl

All triangles of the component's mesh will be  $\varepsilon$  The texture will change the filter parameters o from this component.

Main Page Classes

Class List Class Index Class Hierarchy Class Members

# **AkSwitch Member List**

This is the complete list of members for AkSwitch, including all inherited members.

groupID AkSwitch valueID AkSwitch



**Main Page** Classes

# **AkSwitch**



# int AkSwitch::groupID

Switch Group ID, as defined in WwiseID.cs.



**Main Page** Classes

## **AkSwitch**



# int AkSwitch::valueID

Switch Value ID, as defined in WwiseID.cs.



Main Page Classes

Class List Class Index Class Hierarchy Class Members

# **AkTerminator Member List**

This is the complete list of members for **AkTerminator**, including all inherited members.

Main Page Classes

Class List Class Index Class Hierarchy Class Members

# **AkTriggerBase Member List**

This is the complete list of members for AkTriggerBase, including all inherited members.

Trigger(GameObject in\_gameObject) AkTriggerBase triggerDelegate **AkTriggerBase** 



# **AkTriggerBase**

# delegate void AkTriggerBase::Trigger ( GameObj

Delegate declaration for all Wwise Triggers.

Trigger trigger Delegate

#### **Parameters:**

in\_gameObject is used to pass in\_gameObject when Colliders are used. Some option "Use other object", this i

**Main Page** 

Classes

# AkTriggerBase

**Trigger** triggerDelegate

# **Trigger AkTriggerBase::triggerDelegate = null**

All components reacting to the trigger will be registered in this delegate.



**Main Page Classes** Class List | Class Index Class Hierarchy **Class Members** All Functions **Variables** 

- AddGeometrySet(): AkSurfaceReflector
- AddListener(): AkGameObj
- ClosePortal(): AkRoomPortal
- ExecuteAction(): AK::Wwise::Event
- GetAuxSendValueForPosition(): AkEnvironmentPortal
- GetForward(): AkGameObj
- GetID(): AkRoomPortal , AkRoom
- GetPosition(): AkGameObj
- GetUpward(): AkGameObj
- HandleEvent(): AkBank, AkRoomPortal
- LoadLocalizedBank(): AkMemBankLoader
- LoadNonLocalizedBank(): AkMemBankLoader
- Post(): AK::Wwise::Event
- PostCallbacks(): AkCallbackManager
- PostMIDI(): AK::Wwise::Event
- RemoveGeometrySet(): AkSurfaceReflector
- RemoveListener(): AkGameObj
- SetBGMCallback(): AkCallbackManager
- SetMonitoringCallback(): AkCallbackManager
- StopMIDI(): AK::Wwise::Event
- Trigger(): AkTriggerBase
- UnloadBank(): AkBank



#### - a -

• AcousticTexture : AkSurfaceReflector

actionOnEventType : AkEventaxis : AkEnvironmentPortal

### - b -

• bankName : AkBank , AkMemBankLoader

• basePath : AkInitializer

#### - C -

• callbackManagerBufferSize : AkInitializer

• curveInterpolation : AkEvent

# - d -

• decodeBank : AkBank

defaultPoolSize : AkInitializerdiffractionFlags : AkInitializer

#### - e -

• enableActionOnEvent : AkEvent

• engineLogging : AkInitializer

• eventID : AkEvent

# - g -

groupID : AkState , AkSwitch

## - i -

• info : AkEventCallbackMsg

• isEnvironmentAware : AkGameObj

• isLocalizedBank : AkMemBankLoader

#### - | -

• language : AkInitializer

loadAsynchronous : AkBanklowerPoolSize : AkInitializer

#### - m -

• m positionOffsetData : AkGameObj

• MAX\_ROOMS\_PER\_PORTAL : AkRoomPortal

• maxSoundPropagationDepth : AkInitializer

• memoryCutoffThreshold : AkInitializer

• monitorPoolSize : AkInitializer

• monitorQueuePoolSize : AkInitializer

# - p -

• preparePoolSize : AkInitializer

• priority : AkRoom

#### - r -

• reflectAuxBus : AkSpatialAudioEmitter

• reflectionMaxPathLength : AkSpatialAudioEmitter

• reflectionsAuxBusGain : AkSpatialAudioEmitter

• reflectionsOrder : AkSpatialAudioEmitter

reverbAuxBus : AkRoom

reverbLevel : AkRoom

• roomReverbAuxBusGain : AkSpatialAudioEmitter

rooms : AkRoomPortal

#### - S -

- saveDecodedBank : AkBank • sender : AkEventCallbackMsg
- soundEmitterObject : AkEvent
- spatialAudioPoolSize : AkInitializer
- streamingPoolSize : AkInitializer

## - t -

- transitionDuration : AkEvent
- triggerDelegate : AkTriggerBase
- u -
  - unloadTriggerList : AkBank
- V -
  - valueID : AkState , AkSwitch
- W -
  - wallOcclusion : AkRoom