Advanced Scene Manager

Generated by Doxygen 1.10.0

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

Namespace Documentation

3.1 AdvancedSceneManager Namespace Reference

Classes

- class SceneManager

  The central Advanced Scene Manager API. Provides access to the most important things in ASM.

3.2 AdvancedSceneManager.Callbacks Namespace Reference

Classes

- class ActionUtility

  Contains utility functions for Action.
- class CallbackUtility

  An utility class that invokes callbacks (defined in interfaces based on ISceneCallbacks), and tracks performance and provides tools for optimizing and diagnosing bottlenecks in these callbacks.
- interface ICollectionClose
- interface ICollectionCloseAsync
- interface ICollectionExtraDataCallbacks

  Callbacks for a ScriptableObject that has been set as extra data for a collection.
- interface ICollectionExtraDataCallbacksAsync

  Callbacks for a ScriptableObject that has been set as extra data for a collection.
- interface ICollectionOpen
- interface ICollectionOpenAsync
- interface ISceneCallbacks

  Base interface for ISceneOpen, ISceneClose, ICollectionOpen, ICollectionClose. Does nothing on its own, used by CallbackUtility.
- interface ISceneClose

  Callback for when the scene that a MonoBehaviour is contained within is closed.
- interface ISceneCloseAsync
- interface ISceneOpen

  Callback for when the scene that a MonoBehaviour is contained within is opened.
- interface ISceneOpenAsync
- class LoadingScreen

A class that contains callbacks for loading screens.

- class LoadingScreenBase

  A generic base class for loading screens. You probably want to inherit from LoadingScreen though.
- class ParallelASMCallbacks

  Specifies whatever the ASM callbacks should be run in parallel for any callbacks defined in this script.
- class SplashScreen

  A class that contains callbacks for splash screens.

3.3 AdvancedSceneManager.Core Namespace Reference

Classes

- class App

  Manages startup and quit.
- class Callback

  Represents a callback that can be run on Phase change, or right before loading screen hide (or when it would, if it was enabled).
- class Runtime

  Manages runtime functionality for Advanced Scene Manager such as open scenes and collection.
- class SceneLoadArgs

  Specifies arguments for SceneLoader.LoadScene(Models.Scene, SceneLoadArgs).
- class SceneLoader

  Specifies a scene loader.
- class SceneLoaderArgsBase

  Base class for SceneLoadArgs and SceneUnloadArgs.
- class SceneOperation

  A scene operation is a queueable operation that can open or close scenes. See also: SceneAction.
- class SceneUnloadArgs

  Specifies arguments for SceneLoader.UnloadScene(Models.Scene, SceneUnloadArgs).

Enumerations

- enum Phase {
  CloseCallbacks , UnloadScenes , LoadScenes , OpenCallbacks ,
  CustomActions }

  The phase that a SceneOperation is currently in.

3.3.1 Enumeration Type Documentation

3.3.1.1 Phase

```
enum Phase
```

The phase that a SceneOperation is currently in.

Enumerator

CloseCallbacks	The scene operation is currently executing close callbacks on the scenes that are being closed, if any.
UnloadScenes	The scene operation is currently unloading the scenes, if any.
LoadScenes	The scene operation is currently loading the scenes, if any.
OpenCallbacks	The scene operation is currently executing open callbacks on the scenes that are being opened, if any.
CustomActions	The scene operation is currently executing custom actions, added through SceneOperation.WithAction(SceneAction[ ]) or similar methods, if any.

3.4 AdvancedSceneManager.Models Namespace Reference

Classes

- class ASMModel

  A base class for Profile, SceneCollection and Scene.
- class ASMSceneHelper

  Represents scene helper. Contains functions for opening / closing collections and scenes from UnityEngine.Events.↩ UnityEvent.
- class ASMSettings

  Contains the core of ASM assets. Contains projectSettings and assets
- class DynamicCollection

  Represents a collection that can take a path and then gather all scenes within, guaranteeing that they are all added to build, including non-imported and blacklisted scenes.
- interface ILockable

  Specifies a object that can be locked, using LockUtility.
- class InputBinding

  Represents a input binding for InputSystem. Available even when InputSystem is uninstalled.
- struct InputButton

  Specifies a input binding for use with InputSystem.
- interface ISceneCollection

  Represents the core variables of what makes up a scene collection.
- class Profile

  A profile, contains settings, collections.
- class Scene

  Represents a scene.
- class SceneCollection

  Represents a collection of scenes.
- class SceneCollectionExtensions

  Provides utility methods for working with SceneCollection.
- class StandaloneCollection

  Represents a collection of standalone scenes. These scenes are guaranteed to be included in build (if the associated Profile is active).

Enumerations

- enum InputBindingInteractionType { Open , Hold , Toggle }

  Specifies the interaction type to use for scene bindings.

3.4.1 Enumeration Type Documentation

3.4.1.1 InputBindingInteractionType

`enum InputBindingInteractionType`

Specifies the interaction type to use for scene bindings.

Enumerator

Open	Specifies that the scene or collection will be opened automatically, but not closed.
Hold	Specifies that the scene or collection will be opened automatically on button down, then closed on button up.
Toggle	Specifies that the scene or collection will be opened automatically on button down, then closed on next button down.

3.5 AdvancedSceneManager.Models.Enums Namespace Reference

Enumerations

- enum CollectionLoadingThreadPriority {
  Auto = -2 , Low = ThreadPriority.Low , BelowNormal = ThreadPriority.BelowNormal , Normal = Thread↩
  Priority.Normal ,
  High = ThreadPriority.High }

  Wrapper for ThreadPriority, adds CollectionLoadingThreadPriority.Auto.
- enum CollectionStartupOption { Auto , Open , DoNotOpen }

  Specifies what to do with a SceneCollection during startup.
- enum EditorPersistentOption { Never , WhenAnyOfTheFollowingScenesAreOpened , AnySceneOpened }

  Specifies whatever a scene should be automatically opened outside of play-mode.
- enum LoadingScreenUsage { DoNotUse , UseDefault , Override }

  Specifies what loading screen to use, if any.
- enum SceneImportOption { Manual , SceneCreated }

  Specifies how to scenes are imported.
- enum SceneState {
  Unknown , NotOpen , Queued , Opening ,
  Preloading , Preloaded , Open }

  Specifies that state of a scene.

3.5.1 Enumeration Type Documentation

3.5.1.1 CollectionLoadingThreadPriority

`enum CollectionLoadingThreadPriority`

Wrapper for ThreadPriority, adds CollectionLoadingThreadPriority.Auto.

ThreadPriority:

Enumerator

Auto	Automatically decide ThreadPriority based on if loading screen is open.
Low	Lowest thread priority.
BelowNormal	Below normal thread priority.
Normal	Normal thread priority.
High	Highest thread priority.

3.5.1.2 CollectionStartupOption

`enum CollectionStartupOption`

Specifies what to do with a SceneCollection during startup.

Enumerator

Auto	Specifies that ASM should automatically decide if a SceneCollection should be opened during startup. This means that if no collection in the list specifies either Open or OpenAsPersistent, then the first collection in the list that has Auto will be opened.
Open	Specifies that a SceneCollection will open during startup.
DoNotOpen	Specifies that a SceneCollection will not open during startup.

3.5.1.3 EditorPersistentOption

`enum EditorPersistentOption`

Specifies whatever a scene should be automatically opened outside of play-mode.

Enumerator

Never	Never automatically open scene.
WhenAnyOfTheFollowingScenesAreOpened	Automatically open scene when any specified scene is opened.
AnySceneOpened	Automatically open scene when any scene opens.

3.5.1.4 LoadingScreenUsage

`enum LoadingScreenUsage`

Specifies what loading screen to use, if any.

Enumerator

DoNotUse	Specifies no loading screen.
UseDefault	Specifies default loading screen, defined in profile settings.
Override	Specifies overriden loading screen, defined in SceneCollection.

3.5.1.5 SceneImportOption

```
enum SceneImportOption
```

Specifies how to scenes are imported.

Enumerator

Manual	User will manually import scenes.
SceneCreated	Scenes will be automatically imported when created, otherwise manual.

3.5.1.6 SceneState

```
enum SceneState
```

Specifies that state of a scene.

Enumerator

Unknown	The state of the scene is unknown. (An issue probably occured while checking state)
NotOpen	The scene is not open.
Queued	The scene is in queue to be opened.
Opening	The scene is currently being opened. Mutually exclusive to Preloading.
Preloading	The scene is currently being preloaded. Mutually exclusive to Opening.
Preloaded	The scene is currently preloaded.
Open	The scene is open.

3.6 AdvancedSceneManager.Models.Helpers Namespace Reference

Classes

- class AssetsProxy

  Provides access to the scenes, collections and profiles managed by ASM.
- class DefaultScenes

  Provides access to the default ASM scenes.
- class SettingsProxy

  Provides access to ASM settings.

3.7 AdvancedSceneManager.Models.Internal Namespace Reference

Classes

- class Assets

  Manages all ASM assets.

3.8 AdvancedSceneManager.Models.Utility Namespace Reference

Classes

- class BuildOption

  Represents an enabled state depending on build context (editor, dev build, non-dev build).
- class ProfileDependent

  Specifies a T that changes depending on active Profile.
- class ProfileDependentCollection

  Represents a SceneCollection that changes depending on active Profile.
- class ProfileDependentScene

  Represents a Scene that changes depending on active Profile.
- class SceneCollectionTemplate

  Represents a template for a SceneCollection.

3.9 AdvancedSceneManager.Setup Namespace Reference

Classes

- class ASMInfo

  Gets version info about ASM.

3.10 AdvancedSceneManager.Utility Namespace Reference

Classes

- class ASMFilePathAttribute

  A FilePathAttribute that supports build.
- class ASMScriptableSingleton

  A ScriptableSingleton<T> that supports build.
- class AssetSearchUtility

  Provides utility functions for searching ASM assets.
- class Async

  Represents a async operation that returns a value.
- class CanvasGroupExtensions

  Provides extension methods for CanvasGroup.
- class CanvasSortOrderUtility

  An utility class to manage sort order on canvases.
- class DictionaryUtility

  Contains utility functions for working with dictionaries.
- class FallbackSceneUtility

  An utility class that manages the default scene, called 'AdvancedSceneManager'.
- class GuidReference

  Represents a persistent reference to the GameObject that this is attached to, see also GuidReferenceUtility .
- class GuidReferenceUtility

  An utility for referencing objects globally.
- interface IFadeLoadingScreen

Used to pass arguments from LoadingScreenUtility.FadeIn(LoadingScreen, float, Color?)

- class InitializeInEditorAttribute

  Initializes a class in editor on recompile.
- class InitializeInEditorMethodAttribute

  Initializes a class in editor on recompile.
- interface IQueueable

  Represents a queueable item.
- class LerpUtility

  Provides some convinience functions for lerping.
- class LoadingScreenUtility

  Manager for loading screens.
- class QueueUtility

  A utility that provides queuing.
- class SceneDataUtility

  A utility for storing scene related data. Data can only be saved to disk in editor.
- class SceneUtility

  An utility class to perform actions on scenes.
- class ScriptableObjectUtility

  Contains utility methods for ScriptableObject.
- class SerializableDictionary

  A serializable dictionary.
- class SerializableStringBoolDict

  A serializable dictionary of string and bool.
- class SpamCheck

  Provides an easy way to check for spamming.

3.11 AdvancedSceneManager.Utility.CrossSceneReferences Namespace Reference

Classes

- class CrossSceneReference

  A reference to a variable that references another object in some other scene.
- class CrossSceneReferenceUtility

  An utility for saving and restoring cross-scene references.
- class ObjectReference

  A reference to an object in a scene.
- struct ResolvedCrossReference

  Represents a resolved reference.
- struct ResolvedReference

  Represents a resolved ObjectReference.
- class SceneReferenceCollection

  A collection of CrossSceneReference for a scene. More...

Enumerations

- enum ResolveStatus

  Specifies the result of a resolve.
- enum SceneStatus { Default , Restored , Cleared }

  Specifies the state of a scene.

3.11.1 Class Documentation

3.11.1.1 class AdvancedSceneManager::Utility::CrossSceneReferences::SceneReferenceCollection

A collection of CrossSceneReference for a scene.

3.11.2 Enumeration Type Documentation

3.11.2.1 SceneStatus

```
enum SceneStatus
```

Specifies the state of a scene.

Enumerator

Default	Cross-scene reference utility has not done anything to this scene.
Restored	Cross-scene reference utility has restored references in this scene.
Cleared	Cross-scene reference utility has cleared references in this scene.

3.12 Lazy Namespace Reference

3.13 Lazy.Utility Namespace Reference

Classes

- class CoroutineUtility

  An utility class that helps with running coroutines detached from MonoBehaviour.
- class GlobalCoroutine

  Represents a IEnumerator coroutine started using CoroutineUtility.
- class MainThreadUtility

Chapter 4

Class Documentation

4.1 ActionUtility

Contains utility functions for Action.

Static Public Member Functions

- static void **LogInvoke** (this Action action)

  Tries to invoke the action, then logs error to the console if an error occurred.
- static void **TryInvoke** (this Action action)

  Tries to invoke the action, eats the exception.
- static bool TryInvoke (this Action action, [NotNullWhen(false)] out Exception exception)

  Tries to invoke the action.

4.1.1 Detailed Description

Contains utility functions for Action.

4.1.2 Member Function Documentation

4.1.2.1 TryInvoke()

```
static bool TryInvoke (
            this Action action,
            [NotNullWhen(false)] out Exception exception )  [static]
```

Tries to invoke the action.

Parameters

action	The action to invoke.
exception	The exception that occurred when invoking action. `null` if `true` was returned.

Returns

`true` if invoke succeeded with no exception.

4.2 App

Manages startup and quit.

Classes

- class Props

  An object that persists start properties across domain reload, which is needed when configurable enter play mode is set to reload domain on enter play mode.

Public Member Functions

- void **Restart** (Props props=null)
- IEnumerator **RestartAsync** (Props props=null)
- void **Start** (Props props=null)
- IEnumerator **StartAsync** (Props props=null)
- void **RegisterQuitCallback** (IEnumerator coroutine)

  Register a callback to be called before quit.
- void **UnregisterQuitCallback** (IEnumerator coroutine)

  Unregister a callback that was to be called before quit.
- void CancelQuit ()

  Cancels a quit in progress.
- void Quit (bool fade=true, Color? fadeColor=null, float fadeDuration=1)

  Quits the game, and calls quitCallbacks, optionally with a fade animation.

Properties

- bool isBuildMode `[get]`

  Gets whatever we're currently in build mode.
- bool **isStartupFinished** `[get]`

  Gets if startup process is finished.
- bool **isQuitting** `[get]`

  Gets whatever ASM is currently in the process of quitting.

Events

- Action **beforeRestart**

  Occurs before restart process has begun, but has been initiated.
- Action **afterRestart**

  Occurs after restart has been completed.

4.2.1 Detailed Description

Manages startup and quit.

Usage: SceneManager.app.

4.2.2 Member Function Documentation

4.2.2.1 CancelQuit()

```
void CancelQuit ( )
```

Cancels a quit in progress.

Only usable during a RegisterQuitCallback(IEnumerator) or while isQuitting is true.

4.2.2.2 Quit()

```
void Quit (
            bool fade = true,
            Color?  fadeColor = null,
            float fadeDuration = 1 )
```

Quits the game, and calls quitCallbacks, optionally with a fade animation.

Parameters

fade	Specifies whatever screen should fade out.
fadeColor	Defaults to ProjectSettings.buildUnitySplashScreenColor.
fadeDuration	Specifies the duration of the fade out.

4.2.3 Property Documentation

4.2.3.1 isBuildMode

```
bool isBuildMode  [get]
```

Gets whatever we're currently in build mode.

This is `true` when in build or when play button in scene manager window is pressed.

4.3 ASMFilePathAttribute

A FilePathAttribute that supports build.

Inherits Attribute.

Properties

- string **path** `[get]`

  The path to the associated ScriptableSingleton<T>.

4.3.1 Detailed Description

A FilePathAttribute that supports build.

4.4 ASMInfo

Gets version info about ASM.

Static Public Attributes

- static string **version**

  Gets the current version info.

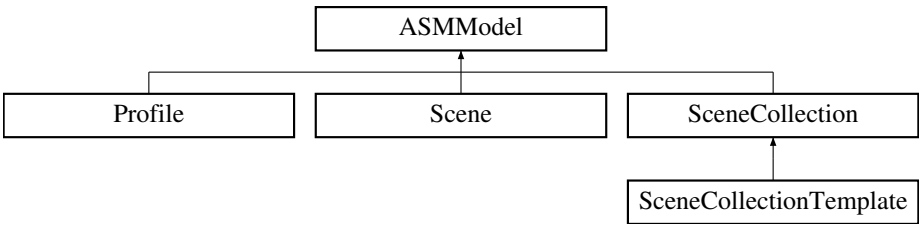
4.4.1 Detailed Description

Gets version info about ASM.

4.5 ASMModel

A base class for Profile, SceneCollection and Scene.

Inheritance diagram for ASMModel:



Public Member Functions

- virtual void Save ()

  Saves the scriptable object after modifying.
- void MarkAsDirty ()

  Mark scriptable object as dirty after modifying.
- virtual bool **IsMatch** (string q)

  Gets if q matches name.
- override string **ToString** ()

  Gets a text summarization of this model.
- virtual string ToString (int indent)

Static Protected Member Functions

- static T **CreateInternal**< **T** > (string name)

  Creates a profile. Throws if name is invalid.

Properties

- string **id** `[get]`

  Gets the id of this ASMModel.
- new string **name** `[get, protected set]`

4.5.1 Detailed Description

A base class for Profile, SceneCollection and Scene.

4.5.2 Member Function Documentation

4.5.2.1 MarkAsDirty()

```
void MarkAsDirty ( )
```

Mark scriptable object as dirty after modifying.

Only available in editor.

4.5.2.2 Save()

```
virtual void Save ( )  [virtual]
```

Saves the scriptable object after modifying.

Only available in editor.

4.5.2.3 ToString()

```
virtual string ToString (
            int indent )  [virtual]
```

Parameters

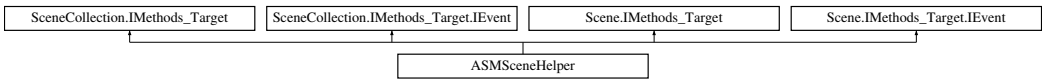
indent	The indentation level, used for nested calls.

Reimplemented in SceneCollection, Profile, and Scene.

4.6 ASMSceneHelper

Represents scene helper. Contains functions for opening / closing collections and scenes from UnityEngine.↩ Events.UnityEvent.

Inheritance diagram for ASMSceneHelper:



Public Member Functions

- SceneOperation **Open** (SceneCollection collection, bool openAll=false)
- SceneOperation **OpenAdditive** (SceneCollection collection, bool openAll=false)
- SceneOperation **ToggleOpen** (SceneCollection collection, bool? openState=null, bool openAll=false)
- SceneOperation **Close** (SceneCollection collection)
- void **_Open** (SceneCollection collection)
- void **_OpenAdditive** (SceneCollection collection)
- void **_ToggleOpen** (SceneCollection collection)
- void **_Close** (SceneCollection collection)
- SceneOperation Open (Scene scene)

  Opens the specified scene.
- SceneOperation **ToggleOpenState** (Scene scene)

  Toggles the open state of this scene.
- SceneOperation **ToggleOpen** (Scene scene, bool? openState=null)

  Toggles the open state of the specified scene, or ensures the state specified.
- SceneOperation Close (Scene scene)

  Closes the specified scene.
- SceneOperation Preload (Scene scene, Action onPreloaded=null)

  Preloads the specified scene, to be displayed at a later time. See also: FinishPreload(Scene), DiscardPreload(↩ Scene).
- SceneOperation FinishPreload (Scene scene)

  Finishes preloading the specified scene, displaying it.
- SceneOperation **DiscardPreload** (Scene scene)

  Discards the specified scene, if preloaded.
- SceneOperation **OpenWithLoadingScreen** (Scene scene, Scene loadingScene)

  Opens the specified scene while a loading screen is open.
- void **SetActive** (Scene scene)

  Sets the specified scene as active in heirarchy.
- void **_Open** (Scene scene)
- void **_ToggleOpen** (Scene scene)
- void **_Close** (Scene scene)
- void **_Preload** (Scene scene)
- void **_FinishPreload** (Scene scene)
- void **_DiscardPreload** (Scene scene)
- void **_SetActive** (Scene scene)
- void **OpenWhereNameStartsWith** (string name)

  Open all scenes that starts with the specified name.
- void **Quit** ()
- void **Restart** ()
- void **RestartCollection** ()

Properties

- new string **name** `[get]`
- static ASMSceneHelper **instance** `[get]`

  Gets a reference to scene helper.

4.6.1 Detailed Description

Represents scene helper. Contains functions for opening / closing collections and scenes from UnityEngine.↩ Events.UnityEvent.

4.6.2 Member Function Documentation

4.6.2.1 Close()

```
SceneOperation Close (
            Scene scene )
```

Closes the specified scene.

Already closed scenes not affected.

Implements Scene.IMethods_Target.

4.6.2.2 FinishPreload()

```
SceneOperation FinishPreload (
            Scene scene )
```

Finishes preloading the specified scene, displaying it.

Scene must be preloaded beforehand.

Implements Scene.IMethods_Target.

4.6.2.3 Open()

```
SceneOperation Open (
            Scene scene )
```

Opens the specified scene.

Already open scenes not affected.

Implements Scene.IMethods_Target.

4.6.2.4 Preload()

```
SceneOperation Preload (
            Scene scene,
            Action onPreloaded = null )
```

Preloads the specified scene, to be displayed at a later time. See also: FinishPreload(Scene), DiscardPreload(↩ Scene).

Scene must be closed beforehand.

Implements Scene.IMethods_Target.

4.7 ASMScriptableSingleton< T >

A ScriptableSingleton<T> that supports build.

Inherits ScriptableObject.

Public Member Functions

- void Save ()

  Saves the singleton to disk.

Properties

- virtual bool **editorOnly** `[get]`

  Specifies that build support will not be applied to this ScriptableSingleton<T>.

4.7.1 Detailed Description

A ScriptableSingleton<T> that supports build.

Type Constraints

***T* : *ASMScriptableSingleton*<*T*>**

4.7.2 Member Function Documentation

4.7.2.1 Save()

```
void Save ( )
```

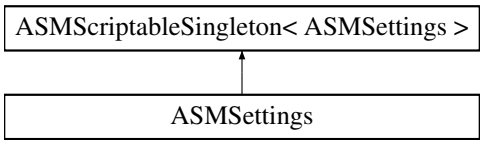
Saves the singleton to disk.

Can be called outside of editor, but has no effect.

4.8 ASMSettings

Contains the core of ASM assets. Contains projectSettings and assets

Inheritance diagram for ASMSettings:



Static Public Member Functions

- static void **OnInitialized** (Action action)

  Runs the callback when ASMSettings has initialized.

Properties

- Profile **defaultProfile** `[get, set]`

  The profile to use when none is set.
- Profile **forceProfile** `[get, set]`

  The profile to force everyone in this project to use.
- Profile **buildProfile** `[get]`

  The profile to use during build.
- bool **checkForDuplicateSceneOperations** `[get, set]`

  By default, ASM checks for duplicate scene operations, since this is usually caused by mistake, but this will disable that.
- bool **preventSpammingEventMethods** `[get, set]`

  By default, ASM will prevent spam calling event methods (i.e. calling Scene.Open() from a button press), but this will disable that.
- float **spamCheckCooldown** `[get, set]`

  Sets the default cooldown for SpamCheck.
- bool enableCrossSceneReferences `[get, set]`

  Gets or sets whatever cross-scene references should be enabled.
- SceneImportOption **sceneImportOption** `[get, set]`

  Gets or sets when to automatically import scenes.
- bool allowExcludingCollectionsFromBuild `[get, set]`

  Specifies whatever collections can be excluded from build.
- bool **reverseUnloadOrderOnCollectionClose** `[get, set]`

  Specifies whatever collections should unload scenes in the reverse order.
- string **assetPath** `[get, set]`

  Specifies the path where profiles and imported scenes should be generated to.
- CustomData **customData** `[get]`

  Specifies custom data.
- Color **buildUnitySplashScreenColor** `[get]`

  This is the color of the unity splash screen, used to make the transition from unity splash screen to ASM smooth, this is set before building. Color.black is used when the unity splash screen is disabled.
- bool **allowSceneLocking** `[get, set]`

  Specifies whatever asm will allow locking scenes.
- bool **allowCollectionLocking** `[get, set]`

  Specifies whatever asm will allow locking collections.

Properties inherited from ASMScriptableSingleton< ASMSettings >

- virtual bool **editorOnly** `[get]`

  Specifies that build support will not be applied to this ScriptableSingleton<T>.

Additional Inherited Members

Public Member Functions inherited from ASMScriptableSingleton< ASMSettings >

- void Save ()

  Saves the singleton to disk.

4.8.1 Detailed Description

Contains the core of ASM assets. Contains projectSettings and assets

Only available in editor.

4.8.2 Property Documentation

4.8.2.1 allowExcludingCollectionsFromBuild

```
bool allowExcludingCollectionsFromBuild [get], [set]
```

Specifies whatever collections can be excluded from build.

When `true`, a toggle will be shown in scene manager window.

4.8.2.2 enableCrossSceneReferences

```
bool enableCrossSceneReferences [get], [set]
```

Gets or sets whatever cross-scene references should be enabled.

Experimental.

4.9 Assets

Manages all ASM assets.

Properties

- static IEnumerable< Profile > **profiles** `[get]`

  Enumerates all imported profiles.
- static IEnumerable< SceneCollection > **collections** `[get]`

  Enumerates all imported collections.
- static IEnumerable< SceneCollectionTemplate > **collectionTemplates** `[get]`

  Enumerates all imported collection templates.
- static IEnumerable< Scene > **scenes** `[get]`

  Enumerates all imported scenes.
- static ASMSceneHelper **sceneHelper** `[get]`

  Gets scene helper singleton.
- static IEnumerable< Object > **allAssets** `[get]`

  Enumerates all imported assets.
- static string assetPath `[get]`

  Gets the import path.
- static string **fallbackScenePath** `[get]`

  Gets the path to the fallback scene.

4.9.1 Detailed Description

Manages all ASM assets.

4.9.2 Property Documentation

4.9.2.1 assetPath

```
string assetPath  [static], [get]
```

Gets the import path.

Can be changed using ProjectSettings.assetPath

4.10 AssetSearchUtility

Provides utility functions for searching ASM assets.

Static Public Member Functions

- static T **Find**< **T** > (string q)

  Finds the T with the specified name.
- static bool **TryFind**< **T** > (string q, out T result)

  Finds the T with the specified name.
- static T **Find**< **T** > (this T[ ] list, string q)
- static bool **TryFind**< **T** > (this T[ ] list, string q, out T result)
- static T **Find**< **T** > (this IEnumerable< T > list, string q)
- static bool **TryFind**< **T** > (this IEnumerable< T > list, string q, out T result)

4.10.1 Detailed Description

Provides utility functions for searching ASM assets.

4.11 AssetsProxy

Provides access to the scenes, collections and profiles managed by ASM.

Public Member Functions

- IEnumerable< T > **Enumerate< T >** ()
  Enumerates T .

Properties

- IEnumerable< Profile > **profiles** `[get]`
  Enumerates all profiles in the project.
- IEnumerable< Scene > **scenes** `[get]`
  Enumerates all scenes.
- IEnumerable< SceneCollection > **collections** `[get]`
  Enumerates all collections.
- IEnumerable< SceneCollectionTemplate > **templates** `[get]`
  Enumerates all templates.
- DefaultScenes **defaults** = new() `[get]`
  Provides access to the default ASM scenes.

4.11.1 Detailed Description

Provides access to the scenes, collections and profiles managed by ASM.

4.12 Async< T >

Represents a async operation that returns a value.

Inherits CustomYieldInstruction.

Public Member Functions

- void **OnComplete** (Action< T > callback)
  Calls the callback when the async operation is complete.

Properties

- static Async< T > **complete** = new(null) `[get]`

  Gets a Async<T> that is already completed.
- T **value** `[get, set]`

  Gets the value that was produced by the async operation.

4.12.1 Detailed Description

Represents a async operation that returns a value.

4.13 BuildOption

Represents an enabled state depending on build context (editor, dev build, non-dev build).

Inherits INotifyPropertyChanged.

Public Member Functions

- bool **GetIsEnabledInCurrentContext** ()

  Get whatever we should be enabled in the current context.

Properties

- bool **enableInEditor** `[get, set]`

  Gets whatever we should be enabled in editor.
- bool **enableInDevBuild** `[get, set]`

  Gets whatever we should be enabled in dev build.
- bool **enableInNonDevBuild** `[get, set]`

  Gets whatever we should be enabled in non-dev build.

4.13.1 Detailed Description

Represents an enabled state depending on build context (editor, dev build, non-dev build).

4.14 Callback

Represents a callback that can be run on Phase change, or right before loading screen hide (or when it would, if it was enabled).

Public Types

- enum When { Before , After }

  Specifies when to run the callback on a given Phase.

Public Member Functions

- Callback **Do** (Action action)

  Performs the specified callback.
- Callback **Do** (Func< IEnumerator > enumerator)

  Performs the specified callback.
- Callback **Do** (GlobalCoroutine coroutine)

  Performs the specified callback.
- Callback **Do** (float delay)

  Performs the specified callback.

Static Public Member Functions

- static implicit **operator Callback** (Action action)

  Converts to Callback.
- static implicit **operator Callback** (Func< IEnumerator > enumerator)

  Converts to Callback.
- static implicit **operator Callback** (GlobalCoroutine coroutine)

  Converts to Callback.
- static Callback **AfterLoadingScreenOpen** ()

  Runs a callback after loading screen would have opened, if one was specified.
- static Callback **BeforeLoadingScreenClose** ()

  Runs a callback before loading screen would close, if one was opened.
- static Callback Before (Phase phase, Scene on)

  Runs a callback before the specified phase, when processing the specified scene.
- static Callback **After** (Phase phase, Scene on)

  Runs a callback after the specified phase.
- static Callback Before (Phase phase)

  Runs a callback before the specified phase, when processing the specified scene.
- static Callback **After** (Phase phase)

  Runs a callback after the specified phase.

Properties

- Phase? **phase** `[get]`

  Specifies on what phase this callback should run at.
- When **when** `[get]`

  Specifies when to run the callback on a given Phase.
- Action **action** `[get]`

  The Action to run.
- Func< IEnumerator > **enumerator** `[get]`

  The IEnumerator coroutine to run.
- GlobalCoroutine **coroutine** `[get]`

  The GlobalCoroutine to run.
- Scene scene `[get]`

  Specifies the scene that this callback should run on.

4.14.1 Detailed Description

Represents a callback that can be run on Phase change, or right before loading screen hide (or when it would, if it was enabled).

4.14.2 Member Enumeration Documentation

4.14.2.1 When

```
enum When
```

Specifies when to run the callback on a given Phase.

Enumerator

Before	Run callback before any scene actions have started during a given Phase.
After	Run callback after all scene actions have run during a given Phase.

4.14.3 Member Function Documentation

4.14.3.1 Before() [1/2]

```
static Callback Before (
            Phase phase )  [static]
```

Runs a callback before the specified phase, when processing the specified scene.

Phase will still have changed to the next though, but scene actions won't run until after callback.

4.14.3.2 Before() [2/2]

```
static Callback Before (
            Phase phase,
            Scene on )  [static]
```

Runs a callback before the specified phase, when processing the specified scene.

Phase will still have changed to the next though, but scene actions won't run until after callback.

4.14.4 Property Documentation

4.14.4.1 scene

```
Scene scene  [get]
```

Specifies the scene that this callback should run on.

Specify `null` to run on all scenes.

4.15 CallbackUtility

An utility class that invokes callbacks (defined in interfaces based on ISceneCallbacks), and tracks performance and provides tools for optimizing and diagnosing bottlenecks in these callbacks.

Classes

- class FluentInvokeAPI

  An helper class to facilitate a fluent api.

4.15.1 Detailed Description

An utility class that invokes callbacks (defined in interfaces based on ISceneCallbacks), and tracks performance and provides tools for optimizing and diagnosing bottlenecks in these callbacks.

4.16 CanvasGroupExtensions

Provides extension methods for CanvasGroup.

Static Public Member Functions

- static IEnumerator **Fade** (this CanvasGroup group, float to, float duration, bool setBlocksRaycasts=true)

  Animates the alpha of a CanvasGroup.

4.16.1 Detailed Description

Provides extension methods for CanvasGroup.

4.17 CanvasSortOrderUtility

An utility class to manage sort order on canvases.

Static Public Member Functions

- static void **Remove** (Canvas canvas)

  Removes this canvas from the managed list.
- static void **PutOnTop** (this Canvas canvas)

  Sets the sort order on this canvas to be on top of all other canvases managed by CanvasSortOrderUtility.
- static void **PutAtBottom** (this Canvas canvas)

  Sets the sort order on this canvas to be on bottom of all other canvases managed by CanvasSortOrderUtility.
- static void MakeSure (this Canvas canvas, Canvas above=null, Canvas below=null)

  Adds a constraint on the sort order of this Canvas based on one or two other canvases.

4.17.1 Detailed Description

An utility class to manage sort order on canvases.

4.17.2 Member Function Documentation

4.17.2.1 MakeSure()

```
static void MakeSure (
            this Canvas canvas,
            Canvas above = null,
            Canvas below = null )  [static]
```

Adds a constraint on the sort order of this Canvas based on one or two other canvases.

Parameters

canvas	The canvas to constrain.
above	Makes sure that this canvas is always above this one.
below	Makes sure that this canvas is always below this one.

See parameter comments for more info.

4.18 CoroutineUtility

An utility class that helps with running coroutines detached from MonoBehaviour.

Classes

- class Events

  Provides events for coroutine events.

Static Public Member Functions

- static void **Run** (Action action, TimeSpan after, [CallerFilePath] string callerFile="", [CallerLineNumber] int callerLine=0, [CallerMemberName] string callerName="")

  Runs the action after the specified time.
- static void **Run** (Action action, float? after=null, bool nextFrame=false, Func< bool > when=null, [CallerFile↩Path] string callerFile="", [CallerLineNumber] int callerLine=0, [CallerMemberName] string callerName="")

  Runs the action after the specified time.
- static GlobalCoroutine StartCoroutineGlobal (this MonoBehaviour _, IEnumerator coroutine, Action on↩Complete=null, string description="", [CallerFilePath] string callerFile="", [CallerLineNumber] int callerLine=0)

  Runs the coroutine using CoroutineUtility, which means it won't be tied to a MonoBehaviour and will persist through scene close.
- static GlobalCoroutine **StartCoroutine** (this IEnumerator coroutine, Action onComplete=null, string description="", [CallerFilePath] string callerFile="", [CallerLineNumber] int callerLine=0)
- static GlobalCoroutine **Chain** (params Func< IEnumerator >[] coroutines)

Runs the coroutines in sequence, wrapped in a single GlobalCoroutine.

- static void **StopCoroutine** (GlobalCoroutine coroutine)

  Stops the coroutine.
- static void **StopAllCoroutines** ()

  Stops all global coroutines.
- static IEnumerator **WaitAll** (params IEnumerator[ ] coroutines)

  Wait for all coroutines to complete.
- static IEnumerator **WaitAll** (this IEnumerable< IEnumerator > coroutines, Func< bool > isCancelled=null, string debugText=null)

  Wait for all coroutines to complete.
- static IEnumerator **WaitAll** (params GlobalCoroutine[ ] coroutines)

  Wait for all coroutines to complete.
- static IEnumerator **WaitAll** (this GlobalCoroutine[ ] coroutines, Func< bool > isCancelled=null)

  Wait for all coroutines to complete.

4.18.1 Detailed Description

An utility class that helps with running coroutines detached from MonoBehaviour.

4.18.2 Member Function Documentation

4.18.2.1 StartCoroutineGlobal()

```
static GlobalCoroutine StartCoroutineGlobal (
            this MonoBehaviour _,
            IEnumerator coroutine,
            Action onComplete = null,
            string description = "",
            [CallerFilePath] string callerFile = "",
            [CallerLineNumber] int callerLine = 0 )  [static]
```

Runs the coroutine using CoroutineUtility, which means it won't be tied to a MonoBehaviour and will persist through scene close.

You may yield return this method.

4.19 CrossSceneReference

A reference to a variable that references another object in some other scene.

4.19.1 Detailed Description

A reference to a variable that references another object in some other scene.

4.20 CrossSceneReferenceUtility

An utility for saving and restoring cross-scene references.

Static Public Member Functions

- static void **Initialize** (bool? enabled=null)

  Initializes cross-scene references, if it is enabled in settings.
- static IEnumerable< ResolvedCrossReference > **GetResolvedReferences** ()

  Gets all references for all scenes.
- static IEnumerable< ResolvedCrossReference > **GetResolvedReferences** (scene scene)

  Gets all references for this scene.
- static IEnumerable< ResolvedCrossReference > **GetResolvedReferences** (GameObject obj)

  Gets all references for this game object.
- static IEnumerable< ResolvedCrossReference > **GetResolvedReferencesValue** (GameObject obj)

  Gets all references for this game object.
- static bool CanSceneBeSaved (scene scene)

  Gets if the cross-scene references can be saved.
- static bool **GetResolved** (CrossSceneReference reference, out ResolvedCrossReference? resolved)

  Get the resolve result for a cross scene reference, if it has been resolved.
- static ResolvedCrossReference **GetResolved** (CrossSceneReference reference)

  Get the resolve result for a cross scene reference, if it has been resolved.
- static SceneReferenceCollection **Load** (Scene scene)

  Loads cross-scene references for a scene.
- static SceneReferenceCollection[ ] **Enumerate** ()

  Loads cross-scene references for all scenes.
- static void ResolveAllScenes ()

  Resolves all scenes.
- static IEnumerable< ResolvedCrossReference > **ResolveScene** (scene scene)

  Resolves cross-scene references in the scene.
- static void **ResetAllScenes** ()

  Resets all cross-scene references in all scenes.
- static void **ResetScene** (scene scene)

  Resets all cross-scene references in scene.
- static IEnumerable< CrossSceneReference > **FindCrossSceneReferences** (params scene[ ] scenes)

  Finds all cross-scene references in the scenes.

4.20.1 Detailed Description

An utility for saving and restoring cross-scene references.

4.20.2 Member Function Documentation

4.20.2.1 CanSceneBeSaved()

```
static bool CanSceneBeSaved (
            scene scene )  [static]
```

Gets if the cross-scene references can be saved.

This would be if status: SceneStatus.Restored and no resolve errors.

4.20.2.2 ResolveAllScenes()

```
static void ResolveAllScenes ( )  [static]
```

Resolves all scenes.

This runs within a single frame.

4.21 DefaultScenes

Provides access to the default ASM scenes.

Public Member Functions

- Scene GetScene (string name)
  Gets a default scene.
- IEnumerable< Scene > Enumerate (bool listNulls=false)
  Enumerates all default scenes.

Properties

- Scene splashScreen `[get]`
  Gets the default splash screen.
- Scene fadeScreen `[get]`
  Gets the default fade loading screen.
- Scene progressBarScreen `[get]`
  Gets the default progress bar loading screen.
- Scene iconBounceScreen `[get]`
  Gets the default icon bounce loading screen.
- Scene pressAnyButtonScreen `[get]`
  Gets the default press any button loading screen.
- Scene quoteScreen `[get]`
  Gets the default quote loading screen.
- Scene videoScreen `[get]`
  Gets the default video loading screen.
- Scene pauseScreen `[get]`
  Gets the default pause screen.
- Scene inGameToolbar `[get]`
  Gets the default in-game-toolbar scene.

4.21.1 Detailed Description

Provides access to the default ASM scenes.

4.21.2 Member Function Documentation

4.21.2.1 Enumerate()

```
IEnumerable< Scene > Enumerate (
            bool listNulls = false )
```

Enumerates all default scenes.

Parameters

listNulls	Specifies whatever `null` will be returned for scenes that could not be found.

4.21.2.2 GetScene()

```
Scene GetScene (
            string name )
```

Gets a default scene.

May be `null` if scene has been removed, or is not imported.

4.21.3 Property Documentation

4.21.3.1 fadeScreen

```
Scene fadeScreen  [get]
```

Gets the default fade loading screen.

May be `null` if scene has been removed, or is not imported.

4.21.3.2 iconBounceScreen

```
Scene iconBounceScreen  [get]
```

Gets the default icon bounce loading screen.

May be `null` if scene has been removed, or is not imported.

4.21.3.3 inGameToolbar

```
Scene inGameToolbar  [get]
```

Gets the default in-game-toolbar scene.

May be `null` if scene has been removed, or is not imported.

4.21.3.4 pauseScreen

```
Scene pauseScreen  [get]
```

Gets the default pause screen.

May be `null` if scene has been removed, or is not imported.

4.21.3.5 pressAnyButtonScreen

`Scene pressAnyButtonScreen [get]`

Gets the default press any button loading screen.

May be `null` if scene has been removed, or is not imported.

4.21.3.6 progressBarScreen

`Scene progressBarScreen [get]`

Gets the default progress bar loading screen.

May be `null` if scene has been removed, or is not imported.

4.21.3.7 quoteScreen

`Scene quoteScreen [get]`

Gets the default quote loading screen.

May be `null` if scene has been removed, or is not imported.

4.21.3.8 splashScreen

`Scene splashScreen [get]`

Gets the default splash screen.

May be `null` if scene has been removed, or is not imported.

4.21.3.9 videoScreen

`Scene videoScreen [get]`

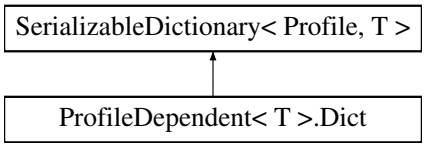
Gets the default video loading screen.

May be `null` if scene has been removed, or is not imported.

4.22 ProfileDependent< T >.Dict

A dictionary of type Profile, *T*.

Inheritance diagram for ProfileDependent< T >.Dict:



4.22.1 Detailed Description

A dictionary of type Profile, *T* .

4.23 DictionaryUtility

Contains utility functions for working with dictionaries.

Static Public Member Functions

- static void **Add**< **TKey, TValue** > (this Dictionary< TKey, TValue > d, TKey key, TValue value)

  Adds or sets the value of a key.
- static void **Add**< **TKey, TList, TItem** > (this Dictionary< TKey, TList > d, TKey key, TItem item)

  Adds the value to the list with the specified key. Creates list automatically if null and adds key if necessary.
- static void **AddRange**< **TKey, TList, TItem** > (this Dictionary< TKey, TList > d, TKey key, IEnumerable< TItem > items)

  Adds the values to the list with the specified key. Creates list automatically if null and adds key if necessary.
- static void **AddRange**< **TKey, TList, TItem** > (this Dictionary< TKey, TList > d, TKey key, params TItem[ ] items)

  Adds the values to the list with the specified key. Creates list automatically if null and adds key if necessary.
- static void **Remove**< **TKey, TList, TItem** > (this Dictionary< TKey, TList > d, TKey key, TItem value)

  Removes the value to the list with the specified key.
- static TValue **GetValue**< **TKey, TValue** > (this Dictionary< TKey, TValue > d, TKey key, TValue default↩ Value=default)

  Gets the value of the specified key, returns default if it does not exist.

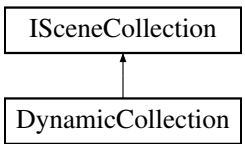
4.23.1 Detailed Description

Contains utility functions for working with dictionaries.

4.24 DynamicCollection

Represents a collection that can take a path and then gather all scenes within, guaranteeing that they are all added to build, including non-imported and blacklisted scenes.

Inheritance diagram for DynamicCollection:



Public Member Functions

- bool **Contains** (string path)

  Gets if the specified SceneAsset path is tracked by this dynamic collection.

Properties

- string **id** `[get]`

  Gets the id of this collection.
- Profile **profile** `[get]`

  Finds the profile associated with this dynamic collection.
- string **path** `[get, set]`

  Specifies the path that this dynamic collection will gather scenes from.
- string **title** `[get, set]`

  Gets the title of this collection.
- string **description** `[get, set]`

  Gets the description of this collection.
- IEnumerable< string > **scenePaths** `[get]`

  Gets the scenes of this collection.
- Scene **this[int index]** `[get]`

  Gets the scene at the specified index.
- int **count** `[get]`

  Gets the scene count of this collection.

Properties inherited from ISceneCollection

4.24.1 Detailed Description

Represents a collection that can take a path and then gather all scenes within, guaranteeing that they are all added to build, including non-imported and blacklisted scenes.

Represents a dynamic scene collection.

4.25 CoroutineUtility.Events

Provides events for coroutine events.

Public Member Functions

- delegate void **CoroutineEvent** (GlobalCoroutine coroutine)

Parameters

coroutine	*The coroutine that this event was called for.*

- delegate object CoroutineFrameStartEvent (GlobalCoroutine coroutine, object data, int level, object parent↩ UserData, bool isPause)
- delegate void CoroutineFrameEndEvent (GlobalCoroutine coroutine, object userData)

Static Public Attributes

- static CoroutineEvent **onCreated**

Occurs when created. Note that GlobalCoroutine is pooled, the same object instance will be used multiple times, and this event is called when the pooled instance is 'constructed', meaning this event will be called multiple times for the same object instance.

- static CoroutineEvent **onDestroyed**

  Occurs when a GlobalCoroutine is 'destroyed'. Note that GlobalCoroutine is pooled, the same object instance will be used multiple times, and this event is called when the pooled instance is 'destroyed', meaning this event will be called multiple times for the same object instance.

- static CoroutineEvent **onCoroutineStarted**

  Occurs when a GlobalCoroutine is started.

- static CoroutineEvent **onCoroutineEnded**

  Occurs when a GlobalCoroutine is ended.

- static CoroutineFrameStartEvent onSubroutineStart

  Occurs before a subroutine in an executing GlobalCoroutine is started.

- static CoroutineFrameEndEvent **onSubroutineEnd**

  Occurs when a subroutine in an executing GlobalCoroutine has ended.

Properties

- static bool **enableEvents** `[get]`

  Enables or disables events. Setter not available, and getter always returns false, in build. Default is `false`.

4.25.1 Detailed Description

Provides events for coroutine events.

4.25.2 Member Function Documentation

4.25.2.1 CoroutineFrameEndEvent()

```
delegate void CoroutineFrameEndEvent (
            GlobalCoroutine coroutine,
            object userData )
```

Parameters

coroutine	The coroutine that this event was called for.
userData	The userdata that was passed to onSubroutineStart.

4.25.2.2 CoroutineFrameStartEvent()

```
delegate object CoroutineFrameStartEvent (
            GlobalCoroutine coroutine,
            object data,
            int level,
            object parentUserData,
            bool isPause )
```

Parameters

coroutine	The coroutine that this event was called for.
data	The object returned from IEnumerator.Current.
level	The level, or depth, of the current subroutine.
parentUserData	The userdata of the subroutine above this one, depth-wise.
isPause	GlobalCoroutine.Pause is reported as a subroutine, this is true when that is the case.

4.25.3 Member Data Documentation

4.25.3.1 onSubroutineStart

```
CoroutineFrameStartEvent onSubroutineStart  [static]
```

Occurs before a subroutine in an executing GlobalCoroutine is started.

A user object can be returned, which is then passed to onSubroutineEnd.

4.26 FallbackSceneUtility

An utility class that manages the default scene, called 'AdvancedSceneManager'.

Static Public Member Functions

- static bool **IsFallbackScene** (scene scene)

  Gets whatever the specified scene is the default scene.

4.26.1 Detailed Description

An utility class that manages the default scene, called 'AdvancedSceneManager'.

The default scene allows us to more easily close all scenes when we need to, since unity requires at least one scene to be open at any time.

4.27 CallbackUtility.FluentInvokeAPI< T >

An helper class to facilitate a fluent api.

Public Member Functions

- FluentInvokeAPI< T > WithCallback (Callback callback)

  Specify a callback, this should point to the interface method which provides a IEnumerator.
- FluentInvokeAPI< T > **WithParam** (object param)

  Specify a parameter to use when invoking the callback.
- IEnumerator **On** (SceneCollection collection, params Scene[ ] additionalScenes)

  Specify the collection scenes to run this callback on and start execution.
- IEnumerator **OnAllOpenScenes** ()

  Specify the collection scenes to run this callback on and start execution..
- IEnumerator **On** (params Scene[ ] scenes)

  Specify the scenes to run this callback on and start execution.
- IEnumerator **On** (params ScriptableObject[ ] scriptableObjects)

  Specify the scenes to run this callback on and start execution.

Properties

- bool **hasDefaultCallback** `[get]`

  Gets whatever T has a default callback. All callbacks inheriting from ISceneCallbacks should have one.

4.27.1 Detailed Description

An helper class to facilitate a fluent api.

Usage: Invoke<T>

4.27.2 Member Function Documentation

4.27.2.1 WithCallback()

```
FluentInvokeAPI< T > WithCallback (
            Callback callback )
```

Specify a callback, this should point to the interface method which provides a IEnumerator.

This is not needed for callback interfaces inheriting from ISceneCallbacks.

4.28 GlobalCoroutine

Represents a IEnumerator coroutine started using CoroutineUtility.

Inherits CustomYieldInstruction.

Public Member Functions

- void **Pause** ()

  Pauses the coroutine. Make sure to not use this from within a coroutine, unless you also make sure to unpause it from outside. No effect if already paused.
- void **Resume** ()

  Resumes a paused coroutine. No effect if not paused.
- void **Stop** ()

  Stops the coroutine.
- override string **ToString** ()

  />

Public Attributes

- MethodBase **method**

  Gets the caller info of this coroutine.

Properties

- Action **onComplete** `[get]`

  The callback that is executed when coroutine is finished.
- bool **isPaused** `[get]`

  Gets whatever this coroutine is paused.
- bool **isComplete** `[get]`

  Gets whatever this coroutine is completed.
- bool **isRunning** `[get]`

  Gets whatever this coroutine is currently running. This will still return `true` *when paused.*
- bool **wasCancelled** `[get]`

  Gets whatever this coroutine was cancelled.
- string **description** `[get, set]`

  Gets the user defined message that was associated with this coroutine.
- override bool **keepWaiting** `[get]`

  CustomYieldInstruction.keepWaiting, this is how unity knows if this coroutine is done or not.

4.28.1 Detailed Description

Represents a IEnumerator coroutine started using CoroutineUtility.

4.29 GuidReference

Represents a persistent reference to the GameObject that this is attached to, see also GuidReferenceUtility .

Inherits MonoBehaviour.

4.29.1 Detailed Description

Represents a persistent reference to the GameObject that this is attached to, see also GuidReferenceUtility .

4.30 GuidReferenceUtility

An utility for referencing objects globally.

Static Public Member Functions

- static string **AddRuntime** (Object obj)

  Adds a reference to the object, returns the id that will be used to find it again.
- static void **RemoveRuntime** (Object obj)

  Removes the reference to this object.
- static void **RemoveRuntime** (string id)

  Removes the reference to the object with this id.
- static bool **HasReference** (string id)

  Gets if reference exists.
- static bool **TryFind**< **T** > (string id, out T obj)

  Gets if reference exists.
- static Object **TryFind** (string id, out Object obj)

  Tries to find the reference.
- static Object **Find** (string id)

  Finds a reference if it exists.
- static T **Find**< **T** > (string id)

  Finds a reference if it exists.
- static IEnumerator **Find** (string id, Action< Object > callback)

  Finds a reference if it exists.
- static string GetOrAddPersistent (GameObject obj)

  Adds a persistent reference to this GameObject.
- static string GenerateID ()

  Generates an id.
- static GameObject FindPersistent (string guid)

  Finds the persistent reference in the currently open scenes.
- static bool TryFindPersistent (string guid, out GameObject obj)

  Finds the persistent reference in the currently open scenes.

4.30.1 Detailed Description

An utility for referencing objects globally.

4.30.2 Member Function Documentation

4.30.2.1 FindPersistent()

```
static GameObject FindPersistent (
            string guid )  [static]
```

Finds the persistent reference in the currently open scenes.

Parameters

guid	The guid of the GameObject to find.

4.30.2.2 GenerateID()

```
static string GenerateID ( )  [static]
```

Generates an id.

Uses https://blog.codinghorror.com/equipping-our-ascii-armor.

4.30.2.3 GetOrAddPersistent()

```
static string GetOrAddPersistent (
            GameObject obj )  [static]
```

Adds a persistent reference to this GameObject.

Can only add in editor, returns `null` otherwise.

4.30.2.4 TryFindPersistent()

```
static bool TryFindPersistent (
            string guid,
            out GameObject obj )  [static]
```

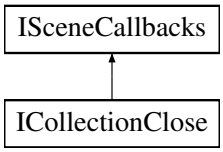
Finds the persistent reference in the currently open scenes.

Parameters

guid	The guid of the GameObject to find.
obj	The found GameObject.

4.31 ICollectionClose

Inheritance diagram for ICollectionClose:



Public Member Functions

- void **OnCollectionClose** (SceneCollection collection)

4.31.1 Detailed Description

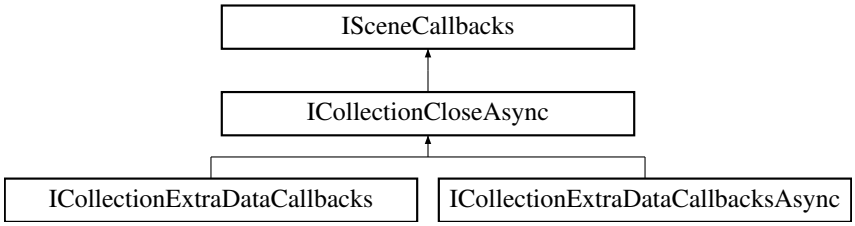
Callback for when a scene in a collection that a MonoBehaviour is contained within is closed.

Called after loading screen has opened, if one is defined, or else just before collection is closed.

See also: ICollectionCloseAsync.

4.32 ICollectionCloseAsync

Inheritance diagram for ICollectionCloseAsync:



Public Member Functions

- IEnumerator **OnCollectionClose** (SceneCollection collection)

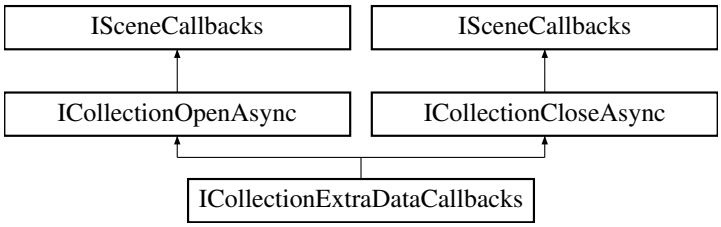
4.32.1 Detailed Description

Scene operation will wait for coroutine callback before continuing.

4.33 ICollectionExtraDataCallbacks

Callbacks for a ScriptableObject that has been set as extra data for a collection.

Inheritance diagram for ICollectionExtraDataCallbacks:



Additional Inherited Members

Public Member Functions inherited from ICollectionOpenAsync

- IEnumerator **OnCollectionOpen** (SceneCollection collection)

Public Member Functions inherited from ICollectionCloseAsync

- IEnumerator **OnCollectionClose** (SceneCollection collection)

4.33.1 Detailed Description

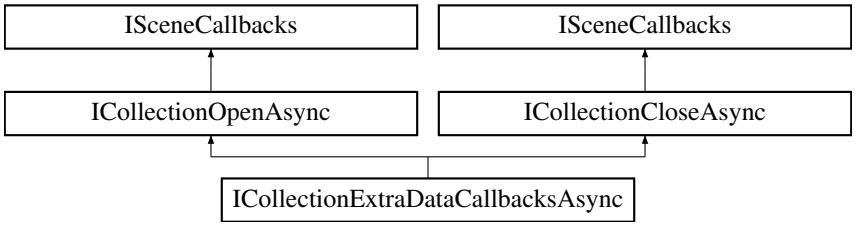
Callbacks for a ScriptableObject that has been set as extra data for a collection.

See also: ICollectionExtraDataCallbacksAsync.

4.34 ICollectionExtraDataCallbacksAsync

Callbacks for a ScriptableObject that has been set as extra data for a collection.

Inheritance diagram for ICollectionExtraDataCallbacksAsync:



Additional Inherited Members

Public Member Functions inherited from ICollectionOpenAsync

- IEnumerator **OnCollectionOpen** (SceneCollection collection)

Public Member Functions inherited from ICollectionCloseAsync

- IEnumerator **OnCollectionClose** (SceneCollection collection)

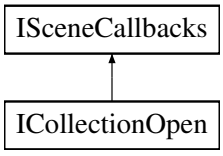
4.34.1 Detailed Description

Callbacks for a ScriptableObject that has been set as extra data for a collection.

Scene operation will wait for coroutine callback before continuing.

4.35 ICollectionOpen

Inheritance diagram for ICollectionOpen:



Public Member Functions

- void **OnCollectionOpen** (SceneCollection collection)

4.35.1 Detailed Description

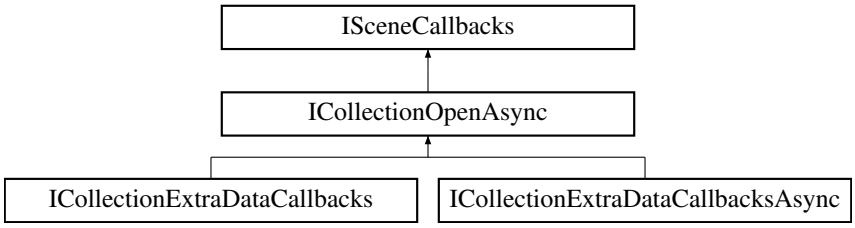
Callback for when a scene in a collection that a MonoBehaviour is contained within is opened.

Called before loading screen is hidden, if one is defined, or else just when collection has opened.

See also: ICollectionOpenAsync.

4.36 ICollectionOpenAsync

Inheritance diagram for ICollectionOpenAsync:



Public Member Functions

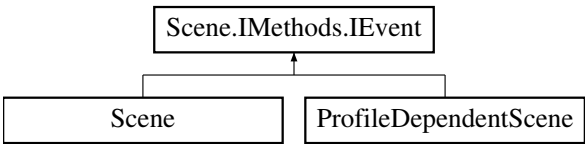
- IEnumerator **OnCollectionOpen** (SceneCollection collection)

4.36.1 Detailed Description

Scene operation will wait for coroutine callback before continuing.

4.37 Scene.IMethods.IEvent

Inheritance diagram for Scene.IMethods.IEvent:



Public Member Functions

- void **_Open** ()

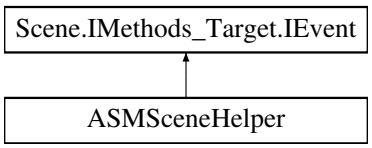
  Event method. Its meant for UnityEngine.Events.UnityEvent.
- void **_ToggleOpenState** ()
- void **_ToggleOpen** (bool? openState=null)
- void **_Close** ()
- void **_Preload** ()
- void **_FinishPreload** ()
- void **_DiscardPreload** ()
- void **_OpenWithLoadingScreen** (Scene loadingScene)
- void **_SetActive** ()

4.37.1 Detailed Description

Specifies methods to be used in UnityEvent, using the scene itself.

4.38 Scene.IMethods_Target.IEvent

Inheritance diagram for Scene.IMethods_Target.IEvent:



Public Member Functions

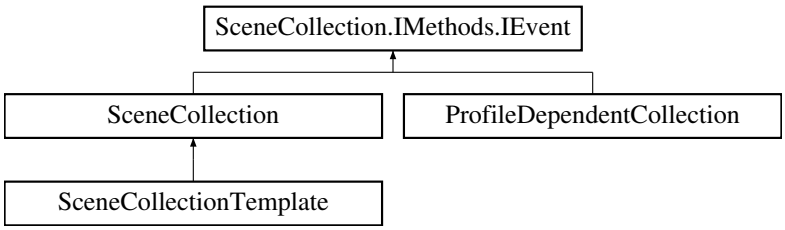
- void **_Open** (Scene scene)
- void **_ToggleOpen** (Scene scene)
- void **_Close** (Scene scene)
- void **_Preload** (Scene scene)
- void **_FinishPreload** (Scene scene)
- void **_DiscardPreload** (Scene scene)
- void **_SetActive** (Scene scene)

4.38.1 Detailed Description

Specifies methods to be used in UnityEvent, when not using scene itself.

4.39 SceneCollection.IMethods.IEvent

Inheritance diagram for SceneCollection.IMethods.IEvent:



Public Member Functions

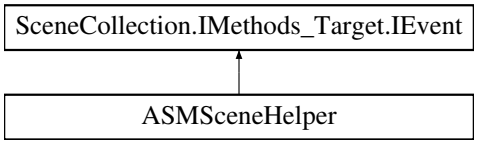
- void **_Open** (bool openAll=false)
- void **_OpenAdditive** (bool openAll=false)
- void **_ToggleOpen** (bool? openState=null)
- void **_Close** ()
- void **_ToggleOpenState** ()

4.39.1 Detailed Description

Specifies methods to be used in UnityEvent, using the collection itself.

4.40 SceneCollection.IMethods_Target.IEvent

Inheritance diagram for SceneCollection.IMethods_Target.IEvent:



Public Member Functions

- void **_Open** (SceneCollection collection)
- void **_OpenAdditive** (SceneCollection collection)
- void **_ToggleOpen** (SceneCollection collection)
- void **_Close** (SceneCollection collection)

4.40.1 Detailed Description

Specifies methods to be used in UnityEvent, when not using collection itself.

4.41 IFadeLoadingScreen

Used to pass arguments from LoadingScreenUtility.FadeIn(LoadingScreen, float, Color?)

Properties

- float **fadeDuration** `[get, set]`

  Specifies the fade duration.
- Color **color** `[get, set]`

  Specifies the color of the fade.

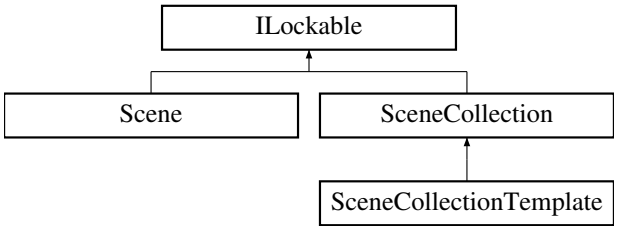
4.41.1 Detailed Description

Used to pass arguments from LoadingScreenUtility.FadeIn(LoadingScreen, float, Color?)

4.42 ILockable

Specifies a object that can be locked, using LockUtility.

Inheritance diagram for ILockable:



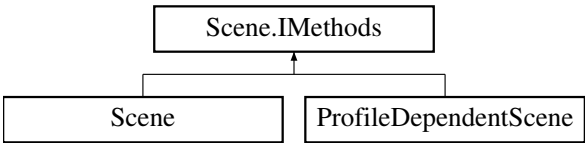
4.42.1 Detailed Description

Specifies a object that can be locked, using LockUtility.

Available, but no effect in build.

4.43 Scene.IMethods

Inheritance diagram for Scene.IMethods:



Classes

- interface IEvent

Public Member Functions

- SceneOperation Open ()

  Opens the scene.
- SceneOperation ToggleOpen (bool? openState=null)

  Toggles this scene open or closed.
- SceneOperation Close ()

  Closes the scene.
- SceneOperation Preload (Action onPreloaded=null)

  Preloads the scene, to be displayed at a later time. See also: FinishPreload, DiscardPreload.
- SceneOperation FinishPreload ()

  Finishes preloading the scene, displaying it.
- SceneOperation **DiscardPreload** ()

  Discards the scene, if preloaded.
- SceneOperation **OpenWithLoadingScreen** (Scene loadingScene)

  Opens the scene while a loading screen is open.
- void **SetActive** ()

  Sets the scene as active in heirarchy.

4.43.1 Detailed Description

Specified methods to be used programmatically, on the scene itself.

4.43.2 Member Function Documentation

4.43.2.1 Close()

```
SceneOperation Close ( )
```

Closes the scene.

No effect if scene is already closed.

Implemented in ProfileDependentScene, and Scene.

4.43.2.2 FinishPreload()

```
SceneOperation FinishPreload ( )
```

Finishes preloading the scene, displaying it.

Scene must be preloaded beforehand.

Implemented in ProfileDependentScene, and Scene.

4.43.2.3 Open()

```
SceneOperation Open ( )
```

Opens the scene.

No effect if scene is already open.

Implemented in ProfileDependentScene, and Scene.

4.43.2.4 Preload()

```
SceneOperation Preload (
            Action onPreloaded = null )
```

Preloads the scene, to be displayed at a later time. See also: FinishPreload, DiscardPreload.

Scene must be closed beforehand.

Implemented in ProfileDependentScene, and Scene.

4.43.2.5 ToggleOpen()

```
SceneOperation ToggleOpen (
            bool?  openState = null )
```

Toggles this scene open or closed.

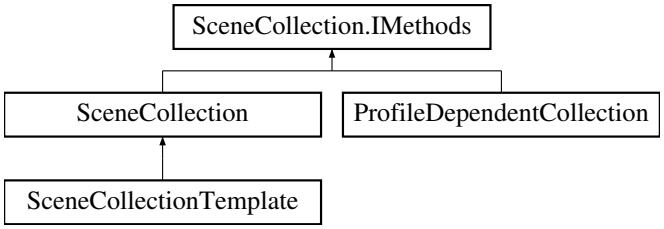
Parameters

openState	Specifies whatever you have a preferred state to toggle to, this means scene will not be closed if `true` is passed. This can be used to scene collection is open, without having an explicit check beforehand. The inverse is also the case for `false`.

Implemented in ProfileDependentScene, and Scene.

4.44 SceneCollection.IMethods

Inheritance diagram for SceneCollection.IMethods:



Classes

- interface IEvent

Public Member Functions

- SceneOperation Open (bool openAll=false)

  Opens this collection.
- SceneOperation OpenAdditive (bool openAll=false)

  Opens this collection as additive.
- SceneOperation ToggleOpen (bool? openState=null, bool openAll=false)

  Toggles this collection open or closed.
- SceneOperation Close ()

  Closes this collection.

4.44.1 Detailed Description

Specified methods to be used programmatically, on the collection itself.

4.44.2 Member Function Documentation

4.44.2.1 Close()

```
SceneOperation Close ( )
```

Closes this collection.

No effect if collection is already closed. Note that "additive collections" are not actually opened, only its scenes are.

Implemented in SceneCollection, and ProfileDependentCollection.

4.44.2.2 Open()

```
SceneOperation Open (
            bool openAll = false )
```

Opens this collection.

Parameters

openAll	Specifies whatever scenes flagged to not open with collection, should.

Reopens all non-persistent scenes.

Implemented in SceneCollection, and ProfileDependentCollection.

4.44.2.3 OpenAdditive()

```
SceneOperation OpenAdditive (
            bool openAll = false )
```

Opens this collection as additive.

Parameters

openAll	Specifies whatever scenes flagged to not open with collection, should.

Additive collections are not "opened", all scenes within are merely opened like normal scenes. Mostly intended for convenience.

Implemented in SceneCollection, and ProfileDependentCollection.

4.44.2.4 ToggleOpen()

```
SceneOperation ToggleOpen (
            bool?  openState = null,
            bool openAll = false )
```

Toggles this collection open or closed.

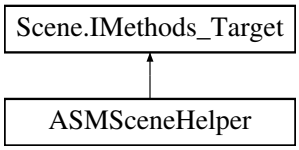
Parameters

openState	Specifies whatever you have a preferred state to toggle to, this means collection will not be closed if `true` is passed. This can be used to ensure collection is open, without having an explicit check beforehand. The inverse is also the case for `false`.
openAll	Specifies whatever scenes flagged to not open with collection, should.

Implemented in SceneCollection, and ProfileDependentCollection.

4.45 Scene.IMethods_Target

Inheritance diagram for Scene.IMethods_Target:



Classes

- interface IEvent

Public Member Functions

- SceneOperation Open (Scene scene)

  Opens the specified scene.
- SceneOperation **ToggleOpenState** (Scene scene)

  Toggles the open state of this scene.
- SceneOperation **ToggleOpen** (Scene scene, bool? openState=null)

  Toggles the open state of the specified scene, or ensures the state specified.
- SceneOperation Close (Scene scene)

  Closes the specified scene.
- SceneOperation Preload (Scene scene, Action onPreloaded=null)

  Preloads the specified scene, to be displayed at a later time. See also: FinishPreload(Scene), DiscardPreload(↩ Scene).
- SceneOperation FinishPreload (Scene scene)

  Finishes preloading the specified scene, displaying it.
- SceneOperation **DiscardPreload** (Scene scene)

  Discards the specified scene, if preloaded.
- SceneOperation **OpenWithLoadingScreen** (Scene scene, Scene loadingScene)

  Opens the specified scene while a loading screen is open.
- void **SetActive** (Scene scene)

  Sets the specified scene as active in heirarchy.

4.45.1 Detailed Description

Specifies methods to be used programmatically, using scene as first parameter.

4.45.2 Member Function Documentation

4.45.2.1 Close()

```
SceneOperation Close (
            Scene scene )
```

Closes the specified scene.

Already closed scenes not affected.

Implemented in ASMSceneHelper.

4.45.2.2 FinishPreload()

```
SceneOperation FinishPreload (
            Scene scene )
```

Finishes preloading the specified scene, displaying it.

Scene must be preloaded beforehand.

Implemented in ASMSceneHelper.

4.45.2.3 Open()

```
SceneOperation Open (
            Scene scene )
```

Opens the specified scene.

Already open scenes not affected.

Implemented in ASMSceneHelper.

4.45.2.4 Preload()

```
SceneOperation Preload (
            Scene scene,
            Action onPreloaded = null )
```

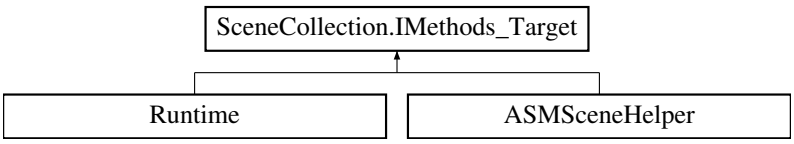
Preloads the specified scene, to be displayed at a later time. See also: FinishPreload(Scene), DiscardPreload(↩ Scene).

Scene must be closed beforehand.

Implemented in ASMSceneHelper.

4.46 SceneCollection.IMethods_Target

Inheritance diagram for SceneCollection.IMethods_Target:



Classes

- interface IEvent

Public Member Functions

- SceneOperation **Open** (SceneCollection collection, bool openAll=false)
- SceneOperation **OpenAdditive** (SceneCollection collection, bool openAll=false)
- SceneOperation **ToggleOpen** (SceneCollection collection, bool? openState=null, bool openAll=false)
- SceneOperation **Close** (SceneCollection collection)

4.46.1 Detailed Description

Specifies methods to be used programmatically, using collection as first parameter.

4.47 InitializeInEditorAttribute

Initializes a class in editor on recompile.

Inherits Attribute.

4.47.1 Detailed Description

Initializes a class in editor on recompile.

Available in build, but no effect.

4.48 InitializeInEditorMethodAttribute

Initializes a class in editor on recompile.

Inherits Attribute.

4.48.1 Detailed Description

Initializes a class in editor on recompile.

Available in build, but no effect.

4.49 InputBinding

Represents a input binding for InputSystem. Available even when InputSystem is uninstalled.

Properties

- List< InputButton > **buttons** `[get]`
  Specifies the buttons.
- bool **openCollectionAsAdditive** `[get, set]`
  Specifies whatever collection should be opened as a collection.
- InputBindingInteractionType **interactionType** `[get, set]`
  Specifies the interaction type.

4.49.1 Detailed Description

Represents a input binding for InputSystem. Available even when InputSystem is uninstalled.

4.50 InputButton

Specifies a input binding for use with InputSystem.

Public Attributes

- string **name**

  Specifies the name of this binding.
- string path

  Specifies the path of this binding.

Properties

- readonly bool **isValid** `[get]`

  Gets if this binding is valid.

4.50.1 Detailed Description

Specifies a input binding for use with InputSystem.

4.50.2 Member Data Documentation

4.50.2.1 path

```
string path
```

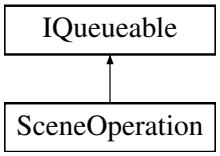
Specifies the path of this binding.

This would be UnityEngine.InputSystem.InputBinding.path.

4.51 IQueueable

Represents a queueable item.

Inheritance diagram for IQueueable:



Public Member Functions

- void OnTurn (Action onComplete)

  Called when it is this queueables turn.
- void **OnCancel** ()

  Called when queueable is cancelled.
- bool **CanQueue** ()

  Called to make sure the item can actually be queued.

4.51.1 Detailed Description

Represents a queueable item.

See also QueueUtility<T>.

4.51.2 Member Function Documentation

4.51.2.1 OnTurn()

```
void OnTurn (
            Action onComplete )
```

Called when it is this queueables turn.

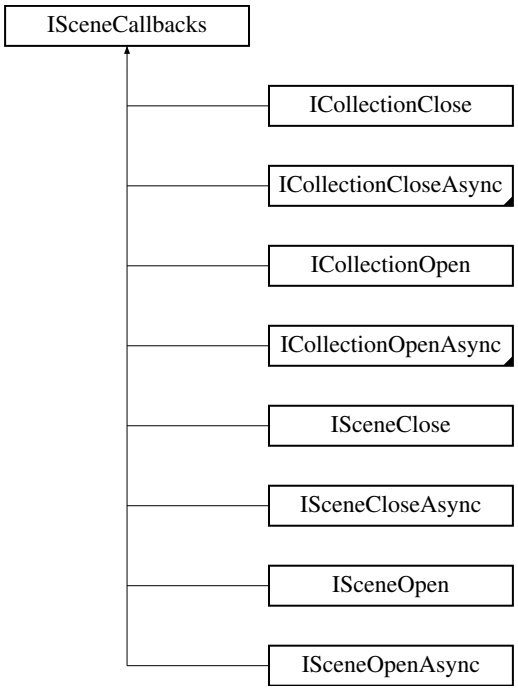
Parameters

onComplete	Must be called when operation is done, otherwise queue will be stuck.

4.52 ISceneCallbacks

Base interface for ISceneOpen, ISceneClose, ICollectionOpen, ICollectionClose. Does nothing on its own, used by CallbackUtility.

Inheritance diagram for ISceneCallbacks:



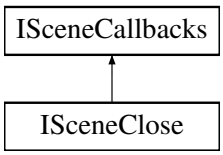
4.52.1 Detailed Description

Base interface for ISceneOpen, ISceneClose, ICollectionOpen, ICollectionClose. Does nothing on its own, used by CallbackUtility.

4.53 ISceneClose

Callback for when the scene that a MonoBehaviour is contained within is closed.

Inheritance diagram for ISceneClose:



Public Member Functions

- void **OnSceneClose** ()

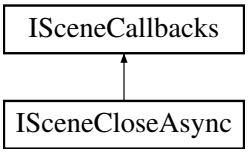
4.53.1 Detailed Description

Callback for when the scene that a MonoBehaviour is contained within is closed.

See also: ISceneCloseAsync.

4.54 ISceneCloseAsync

Inheritance diagram for ISceneCloseAsync:



Public Member Functions

- IEnumerator **OnSceneClose** ()

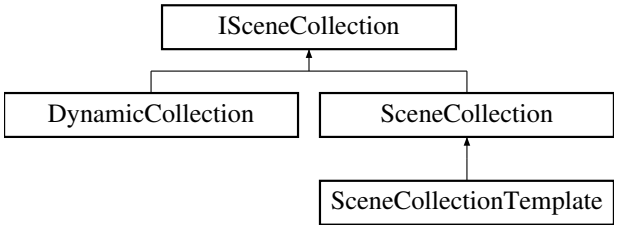
4.54.1 Detailed Description

Scene operation will wait for coroutine callback before continuing.

4.55 ISceneCollection

Represents the core variables of what makes up a scene collection.

Inheritance diagram for ISceneCollection:



Properties

- IEnumerable< Scene > **scenes** `[get]`

  Gets the scenes of this collection.
- IEnumerable< string > **scenePaths** `[get]`

  Gets the scenes of this collection.
- string **title** `[get]`

  Gets the title of this collection.
- string **description** `[get]`

  Gets the description of this collection.
- int **count** `[get]`

  Gets the scene count of this collection.
- string **id** `[get]`

  Gets the id of this collection.
- Scene **this[int index]** `[get]`

  Gets the scene at the specified index.

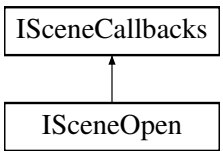
4.55.1 Detailed Description

Represents the core variables of what makes up a scene collection.

4.56 ISceneOpen

Callback for when the scene that a MonoBehaviour is contained within is opened.

Inheritance diagram for ISceneOpen:



Public Member Functions

- void **OnSceneOpen** ()

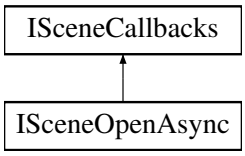
4.56.1 Detailed Description

Callback for when the scene that a MonoBehaviour is contained within is opened.

See also: ISceneOpenAsync.

4.57 ISceneOpenAsync

Inheritance diagram for ISceneOpenAsync:



Public Member Functions

- IEnumerator **OnSceneOpen** ()

4.57.1 Detailed Description

Scene operation will wait for coroutine callback before continuing.

4.58 LerpUtility

Provides some convinience functions for lerping.

Static Public Member Functions

- static IEnumerator Lerp (float start, float end, float duration, Action< float > callback, Action onComplete=null)

  Lerp from start to end over duration seconds.
- static IEnumerator **Lerp** (Vector3 start, Vector3 end, float duration, Action< Vector3 > callback, Action onComplete=null)
- static IEnumerator **Lerp** (Vector2 start, Vector2 end, float duration, Action< Vector2 > callback, Action onComplete=null)

4.58.1 Detailed Description

Provides some convinience functions for lerping.

4.58.2 Member Function Documentation

4.58.2.1 Lerp()

```
static IEnumerator Lerp (
            float start,
            float end,
            float duration,
            Action< float > callback,
            Action onComplete = null )  [static]
```

Lerp from *start* to *end* over *duration* seconds.

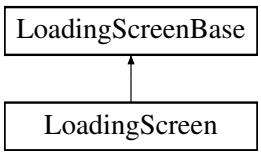
Parameters

start	The start value.
end	The end value.
duration	The duration in seconds to lerp for.
callback	The callback each lerp interval.
onComplete	Callback when complete.

4.59 LoadingScreen

A class that contains callbacks for loading screens.

Inheritance diagram for LoadingScreen:



Public Member Functions

- virtual void **OnProgressChanged** (float progress)

  Called when progress has changed.
- override IEnumerator OnOpen ()
- override IEnumerator OnClose ()

Public Member Functions inherited from LoadingScreenBase

- IEnumerator **OnOpen** ()

  Called when the loading screen is opened.
- IEnumerator **OnClose** ()

  Called when the loading screen is about to close.

Properties

- SceneOperation **operation** `[get, set]`

  The current scene operation that this loading screen is associated with. May be null for the first few frames, before loading has actually begun.

Additional Inherited Members

Public Attributes inherited from LoadingScreenBase

- Action< LoadingScreenBase > **onDestroy**

  Occurs when loading screen is destroyed.
- Canvas canvas

4.59.1 Detailed Description

A class that contains callbacks for loading screens.

SplashScreen and LoadingScreen cannot co-exist within the same scene.

4.59.2 Member Function Documentation

4.59.2.1 OnClose()

```
override IEnumerator OnClose ( )  [abstract]
```

Use this callback to hide your loading screen.

4.59.2.2 OnOpen()

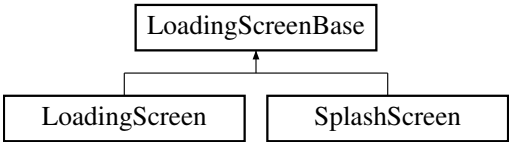
```
override IEnumerator OnOpen ( )  [abstract]
```

Use this callback to show your loading screen, the scene manager will wait until its done.

4.60 LoadingScreenBase

A generic base class for loading screens. You probably want to inherit from LoadingScreen though.

Inheritance diagram for LoadingScreenBase:



Public Member Functions

- IEnumerator **OnOpen** ()
  Called when the loading screen is opened.
- IEnumerator **OnClose** ()
  Called when the loading screen is about to close.

Public Attributes

- Action< LoadingScreenBase > **onDestroy**
  Occurs when loading screen is destroyed.
- Canvas canvas

4.60.1 Detailed Description

A generic base class for loading screens. You probably want to inherit from LoadingScreen though.

When multiple loading screens exist within the same scene, only the first found one will be used.

4.60.2 Member Data Documentation

4.60.2.1 canvas

```
Canvas canvas
```

The canvas that this loading screen uses.

This will automatically register canvas with CanvasSortOrderUtility, to automatically manage canvas sort order.

You probably want to set this through the inspector.

4.61 LoadingScreenUtility

Manager for loading screens.

Static Public Member Functions

- static bool **IsLoadingScreenOpen** (Scene scene)

  Gets if this scene is a loading screen.
- static Async< T > **OpenLoadingScreen**< **T** > (Scene loadingScene, SceneOperation operation=null, Action< T > callbackBeforeBegin=null, Action< float > progress=null)

  Shows a loading screen.
- static IEnumerator **CloseLoadingScreen** (Scene scene)
- static IEnumerator CloseLoadingScreen (LoadingScreenBase loadingScreen, Action< float > progress=null, bool closeScene=true)

  Hide the loading screen.
- static IEnumerator **CloseLoadingScreenScene** (Scene scene, Action< float > progress=null)

  Close the scene that contained a loading screen.
- static IEnumerator **CloseAll** ()

  Hide all loading screens.
- static Async< LoadingScreen > **FadeOut** (float duration=1, Color? color=null, Action< float > progress=null)

  Fades out the screen.
- static IEnumerator **FadeIn** (LoadingScreenBase loadingScreen, float duration=1, Color? color=null, Action< float > progress=null)

  Fades in the screen.
- static IEnumerator **DoAction** (Scene scene, Action action, Action< LoadingScreenBase > loadingScreen↩ Callback=null)
- static IEnumerator DoAction (Scene scene, Func< IEnumerator > coroutine, Action< LoadingScreenBase > loadingScreenCallback=null)

  Opens loading screen, performs action and hides loading screen again.
- static IEnumerator **WithProgress** (this AsyncOperation asyncOperation, Action< float > onProgress)

  Returns a coroutine that returns when AsyncOperation.isDone becomes `true`. onProgress will be called every frame with AsyncOperation.progress.
- static AsyncOperation **Preload** (this AsyncOperation asyncOperation, out Func< IEnumerator > activate↩ Callback)

  Sets AsyncOperation.allowSceneActivation to `false`.

Properties

- static bool **isAnyLoadingScreenOpen** `[get]`

  Gets if any loading screens are open.
- static Scene **fade** `[get]`

  Finds the default fade loading screen. Will be null if not included in build.
- static Scene **defaultLoadingScreen** `[get]`

  Gets the current default loading screen.
- static IEnumerable< LoadingScreenBase > **loadingScreens** `[get]`

  The currently open loading screens.

4.61.1 Detailed Description

Manager for loading screens.

4.61.2 Member Function Documentation

4.61.2.1 CloseLoadingScreen()

```
static IEnumerator CloseLoadingScreen (
            LoadingScreenBase loadingScreen,
            Action< float > progress = null,
            bool closeScene = true )  [static]
```

Hide the loading screen.

Parameters

loadingScreen	The loading screen to hide.
progress	The callback to receive progress.
closeScene	Specifies whatever the scene should be closed afterwards. Use CloseLoadingScreenScene(Scene, Action<float>) if `false`.

4.61.2.2 DoAction()

```
static IEnumerator DoAction (
            Scene scene,
            Func< IEnumerator > coroutine,
            Action< LoadingScreenBase > loadingScreenCallback = null )  [static]
```

Opens loading screen, performs action and hides loading screen again.

Parameters

scene	The loading screen scene.
coroutine	To coroutine to execute.
loadingScreenCallback	The callback to perform when loading script is loaded, but before ASM has called LoadingScreenBase.OnOpen().

4.62 MainThreadUtility

Static Public Member Functions

- static T Invoke< T > (Func< T > func)

  Queues the function to be run on the main thread, during the next frame.
- static void Invoke (Action action)

  Queues the action to be run on the main thread, during the next frame.
- static T Invoke< T > (this Func< T > func, bool mainThread=false)

  Invokes the action .
- static void Invoke (this Action action, bool mainThread=false)

  Invokes the action .
- static void **Start** ()

  Starts main thread utility coroutine.
- static void **Stop** ()

  Stops main thread utility coroutine.

Properties

- static bool **isEnabled** `[get]`

  Gets whatever MainThreadUtility is enabled, set to `false` in source code to disable.
- static bool **isOnMainThread** `[get]`

  Gets if the thread we're currently on is the main thread.
- static bool **IsRunning** `[get]`

  Gets if main thread utility is running.

4.62.1 Detailed Description

An utility for running actions on the main thread.

Only usable in play mode.

4.62.2 Member Function Documentation

4.62.2.1 Invoke() [1/2]

```
static void Invoke (
            Action action )  [static]
```

Queues the action to be run on the main thread, during the next frame.

Parameters

action	The action to invoke.

4.62.2.2 Invoke() [2/2]

```
static void Invoke (
            this Action action,
            bool mainThread = false )  [static]
```

Invokes the *action* .

Parameters

action	The action to invoke.
mainThread	Queues the action to be run on the main thread, during the next frame.

4.62.2.3 Invoke< T >() [1/2]

```
static T Invoke< T > (
            Func< T > func )  [static]
```

Queues the function to be run on the main thread, during the next frame.

Parameters

func	The action to invoke.

4.62.2.4 Invoke< T >() [2/2]

```
static T Invoke< T > (
            this Func< T > func,
            bool mainThread = false )  [static]
```

Invokes the *action* .

Parameters

func	The action to invoke.
mainThread	Queues the action to be run on the main thread, during the next frame.

4.63 ObjectReference

A reference to an object in a scene.

Inherits IEqualityComparer< ObjectReference >.

Public Member Functions

- ObjectReference **With** (Component component)

  Adds data about a component.
- ObjectReference **With** (int? unityEventIndex=null, int? arrayIndex=null)

  Adds data about an unity event.
- bool **IsValid** (bool returnTrueWhenSceneIsUnloaded=false)

  Returns true if the reference is still valid.

4.63.1 Detailed Description

A reference to an object in a scene.

4.64 ParallelASMCallbacks

Specifies whatever the ASM callbacks should be run in parallel for any callbacks defined in this script.

Inherits Attribute.

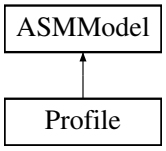
4.64.1 Detailed Description

Specifies whatever the ASM callbacks should be run in parallel for any callbacks defined in this script.

4.65 Profile

A profile, contains settings, collections.

Inheritance diagram for Profile:



Public Member Functions

- bool **IsStartupCollection** (SceneCollection collection)

  Gets whatever the specified collection is a startup collection.
- int **IndexOf** (SceneCollection collection)

  Gets the index of the specified collection.
- int **IndexOf** (DynamicCollection collection)

  Gets the index of the specified collection.
- bool **Contains** (ISceneCollection collection, bool checkRemoved=false)

  Gets whatever this profile contains the specified collection.
- IEnumerable< ISceneCollection > **FindCollections** (Scene scene)

  Finds all collection that the scene is included in. Includes dynamic collections.
- override string ToString (int indent)

Public Member Functions inherited from ASMModel

- virtual void Save ()

  Saves the scriptable object after modifying.
- void MarkAsDirty ()

  Mark scriptable object as dirty after modifying.
- virtual bool **IsMatch** (string q)

  Gets if q matches name.
- override string **ToString** ()

  Gets a text summarization of this model.

Static Public Member Functions

- static Profile **Find** (string q)

  Finds the profile with the specified name or id.
- static bool **TryFind** (string q, out Profile profile)

  Finds the profile with the specified name or id.

Static Public Attributes

- static readonly string **AssetSearchString** = "t:" + typeof(Profile).FullName

  Gets 't:AdvancedSceneManager.Models.Profile', the string to use in AssetDatabase.FindAssets(string).

Properties

- static Profile **current** `[get]`

  Gets the currently active profile.
- IEnumerable< Scene > scenes `[get]`

  Gets the scenes managed by this profile.
- IEnumerable< SceneCollection > **collections** `[get]`

  Gets the collections contained within this profile.
- IEnumerable< DynamicCollection > **dynamicCollections** `[get]`

  Gets the dynamic collections contained within this profile.
- StandaloneCollection **standaloneScenes** `[get]`

  Gets the standalone scenes contained within this profile.
- IEnumerable< Scene > **startupScenes** `[get]`

  Gets the scenes flagged to open on startup.
- IEnumerable< ISceneCollection > removedCollections `[get]`

  Gets all removed collections in this profile.
- IEnumerable< ISceneCollection > **allCollections** `[get]`

  Gets collections, standaloneScenes, dynamicCollections.
- IEnumerable< Scene > specialScenes `[get]`

  Gets default loading screen, splash screen and startup loading screen.
- IEnumerable< SceneCollection > startupCollections `[get]`

  Gets the collections that will be opened on startup.
- Scene **startupScene** `[get, set]`

  The startup scene.
- Scene **startupLoadingScreen** `[get, set]`

  The loading screen to use during startup.
- Scene **loadingScreen** `[get, set]`

  The default loading screen.
- Scene **splashScreen** `[get, set]`

  The splash screen.
- ThreadPriority **backgroundLoadingPriority** `[get, set]`

  Application.backgroundLoadingPriority setting is not saved, and must be manually set every time build or editor starts, this property persists the value and automatically sets it during startup.
- bool **enableChangingBackgroundLoadingPriority** `[get, set]`

  Enable or disable ASM automatically changing Application.backgroundLoadingPriority.
- bool **createCameraDuringStartup** `[get, set]`

  Enable or disable ASM automatically creating a camera during startup.
- bool **unloadUnusedAssetsForStandalone** `[get, set]`

  Enable or disable ASM calling Resources.UnloadUnusedAssets after standalone scenes has been opened or closed.

Properties inherited from ASMModel

- string **id** `[get]`

  Gets the id of this ASMModel.
- new string **name** `[get, protected set]`

Additional Inherited Members

Static Protected Member Functions inherited from ASMModel

- static T **CreateInternal**< **T** > (string name)

  Creates a profile. Throws if name is invalid.

4.65.1 Detailed Description

A profile, contains settings, collections.

4.65.2 Member Function Documentation

4.65.2.1 ToString()

```
override string ToString (
            int indent )  [virtual]
```

Parameters

indent	The indentation level, used for nested calls.

Parameters

indent	The indentation level, used for nested calls.

Reimplemented from ASMModel.

4.65.3 Property Documentation

4.65.3.1 removedCollections

```
IEnumerable<ISceneCollection> removedCollections  [get]
```

Gets all removed collections in this profile.

Removed collections still exist until deleted, and may be manually opened, but they will not be listed in collections or dynamicCollections.

4.65.3.2 scenes

```
IEnumerable<Scene> scenes  [get]
```

Gets the scenes managed by this profile.

Includes both collection and standalone scenes.

4.65.3.3 specialScenes

```
IEnumerable<Scene> specialScenes [get]
```

Gets default loading screen, splash screen and startup loading screen.

`null` is filtered out.

4.65.3.4 startupCollections

```
IEnumerable<SceneCollection> startupCollections [get]
```

Gets the collections that will be opened on startup.

If no collection is explicitly defined to be opened during startup, then the first available collection in list will be returned.

4.66 ProfileDependent< T >

Specifies a *T* that changes depending on active Profile.

Inherits ScriptableObject.

Classes

- class Dict

  A dictionary of type Profile, T .

Public Member Functions

- bool GetModel (out T scene)

  Gets the selected scene.
- T2 DoAction< T2 > (Func< T, T2 > action)

  Performs an action on the scene.
- void DoAction (Action< T > action)

  Performs an action on the scene.

Public Attributes

- Dict **list** = new Dict()

  The list of proxies for this T .

Properties

- bool **isValid** `[get]`

  Gets if the current state of this T is valid.

4.66.1 Detailed Description

Specifies a *T* that changes depending on active Profile.

Type Constraints

T **:** ***ASMModel***

4.66.2 Member Function Documentation

4.66.2.1 DoAction()

```
void DoAction (
            Action< T > action )
```

Performs an action on the scene.

Does nothing if isValid is `false`.

4.66.2.2 DoAction< T2 >()

```
T2 DoAction< T2 > (
            Func< T, T2 > action )
```

Performs an action on the scene.

Does nothing if isValid is `false`.

4.66.2.3 GetModel()

```
bool GetModel (
            out T scene )
```

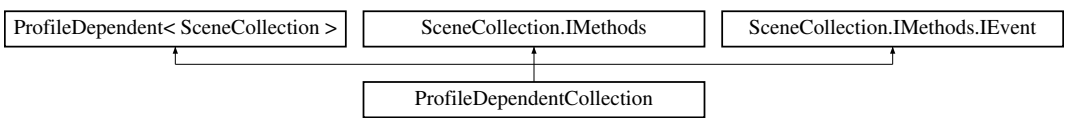
Gets the selected scene.

Returns null if scene something went wrong.

4.67 ProfileDependentCollection

Represents a SceneCollection that changes depending on active Profile.

Inheritance diagram for ProfileDependentCollection:



Public Member Functions

- SceneOperation Open (bool openAll=false)
  Opens this collection.
- SceneOperation OpenAdditive (bool openAll=false)
  Opens this collection as additive.
- SceneOperation ToggleOpen (bool? openState=null, bool openAll=false)
  Toggles this collection open or closed.
- SceneOperation Close ()
  Closes this collection.
- void **_Open** (bool openAll=false)
- void **_OpenAdditive** (bool openAll=false)
- void **_ToggleOpen** (bool? openState=null)
- void **_ToggleOpenState** ()
- void **_Close** ()

Public Member Functions inherited from ProfileDependent< SceneCollection >

- bool GetModel (out T scene)
  Gets the selected scene.
- T2 DoAction< T2 > (Func< T, T2 > action)
  Performs an action on the scene.
- void DoAction (Action< T > action)
  Performs an action on the scene.

Additional Inherited Members

Public Attributes inherited from ProfileDependent< SceneCollection >

- Dict **list**
  The list of proxies for this T .

Properties inherited from ProfileDependent< SceneCollection >

- bool **isValid** `[get]`
  Gets if the current state of this T is valid.

4.67.1 Detailed Description

Represents a SceneCollection that changes depending on active Profile.

4.67.2 Member Function Documentation

4.67.2.1 Close()

```
SceneOperation Close ( )
```

Closes this collection.

No effect if collection is already closed. Note that "additive collections" are not actually opened, only its scenes are.

Implements SceneCollection.IMethods.

4.67.2.2 Open()

```
SceneOperation Open (
            bool openAll = false )
```

Opens this collection.

Parameters

openAll	Specifies whatever scenes flagged to not open with collection, should.

Reopens all non-persistent scenes.

Implements SceneCollection.IMethods.

4.67.2.3 OpenAdditive()

```
SceneOperation OpenAdditive (
            bool openAll = false )
```

Opens this collection as additive.

Parameters

openAll	Specifies whatever scenes flagged to not open with collection, should.

Additive collections are not "opened", all scenes within are merely opened like normal scenes. Mostly intended for convenience.

Implements SceneCollection.IMethods.

4.67.2.4 ToggleOpen()

```
SceneOperation ToggleOpen (
            bool?  openState = null,
            bool openAll = false )
```

Toggles this collection open or closed.

Parameters

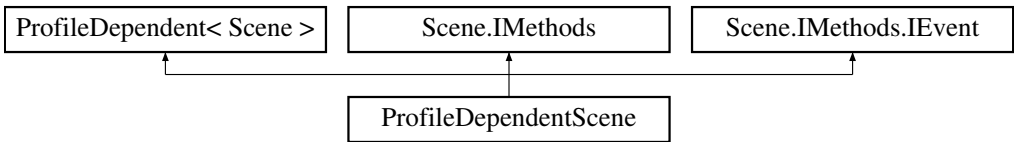
openState	Specifies whatever you have a preferred state to toggle to, this means collection will not be closed if `true` is passed. This can be used to ensure collection is open, without having an explicit check beforehand. The inverse is also the case for `false`.
openAll	Specifies whatever scenes flagged to not open with collection, should.

Implements SceneCollection.IMethods.

4.68 ProfileDependentScene

Represents a Scene that changes depending on active Profile.

Inheritance diagram for ProfileDependentScene:



Public Member Functions

- SceneOperation Open ()

  Opens the scene.
- SceneOperation ToggleOpen (bool? openState=null)

  Toggles this scene open or closed.
- SceneOperation Close ()

  Closes the scene.
- SceneOperation Preload (Action onPreloaded=null)

  Preloads the scene, to be displayed at a later time. See also: FinishPreload, DiscardPreload.
- SceneOperation FinishPreload ()

  Finishes preloading the scene, displaying it.
- SceneOperation **DiscardPreload** ()

  Discards the scene, if preloaded.
- SceneOperation **OpenWithLoadingScreen** (Scene loadingScreen)

  Opens the scene while a loading screen is open.
- void **SetActive** ()

  Sets the scene as active in heirarchy.
- void **_Open** ()

  Event method. Its meant for UnityEngine.Events.UnityEvent.
- void **_ToggleOpenState** ()
- void **_ToggleOpen** (bool? openState=null)
- void **_Close** ()
- void **_Preload** ()
- void **_FinishPreload** ()
- void **_DiscardPreload** ()
- void **_OpenWithLoadingScreen** (Scene loadingScene)
- void **_SetActive** ()

Public Member Functions inherited from ProfileDependent< Scene >

- bool GetModel (out T scene)

  Gets the selected scene.
- T2 DoAction< T2 > (Func< T, T2 > action)

  Performs an action on the scene.
- void DoAction (Action< T > action)

  Performs an action on the scene.

Additional Inherited Members

Public Attributes inherited from ProfileDependent< Scene >

- Dict **list**

  The list of proxies for this T .

Properties inherited from ProfileDependent< Scene >

- bool **isValid** `[get]`

  Gets if the current state of this T is valid.

4.68.1 Detailed Description

Represents a Scene that changes depending on active Profile.

4.68.2 Member Function Documentation

4.68.2.1 Close()

```
SceneOperation Close ( )
```

Closes the scene.

No effect if scene is already closed.

Implements Scene.IMethods.

4.68.2.2 FinishPreload()

```
SceneOperation FinishPreload ( )
```

Finishes preloading the scene, displaying it.

Scene must be preloaded beforehand.

Implements Scene.IMethods.

4.68.2.3 Open()

```
SceneOperation Open ( )
```

Opens the scene.

No effect if scene is already open.

Implements Scene.IMethods.

4.68.2.4 Preload()

```
SceneOperation Preload (
            Action onPreloaded = null )
```

Preloads the scene, to be displayed at a later time. See also: FinishPreload, DiscardPreload.

Scene must be closed beforehand.

Implements Scene.IMethods.

4.68.2.5 ToggleOpen()

```
SceneOperation ToggleOpen (
            bool?  openState = null )
```

Toggles this scene open or closed.

Parameters

openState	Specifies whatever you have a preferred state to toggle to, this means scene will not be closed if `true` is passed. This can be used to scene collection is open, without having an explicit check beforehand. The inverse is also the case for `false`.

Implements Scene.IMethods.

4.69 App.Props

An object that persists start properties across domain reload, which is needed when configurable enter play mode is set to reload domain on enter play mode.

Public Member Functions

- **Props** ()
  Creates a new props.
- **Props** (Props props)
  Creates a new props, from the specified props, copying its values.

Public Attributes

- bool? **displaySplashScreen** = null
  Specifies whatever the splash screen should be played.
- bool **forceOpenAllScenesOnCollection** = false
  Specifies whatever all scenes on openCollection should be opened.
- Color? fadeColor
  The color for the fade out.
- float **fadeInDuration** = 1f
  Specifies the duration for the fade out animation.
- float fadeOutDuration = 1f
  Specifies the duration for the fade in animation.
- SceneCollection **openCollection**
  Specifies a collection to be opened after startup process is done.

Properties

- static Props defaultProps = new() `[get]`

  Gets the default Props.
- bool **runStartupProcessWhenPlayingCollection** `[get, set]`

  Specifies whatever startup process should run before openCollection is opened.
- bool **runStartupProcess** `[get]`

  Gets if startup process should run.
- Color **effectiveFadeColor** `[get]`

  Gets the effective fade animation color, uses fadeColor if specified, but falls back to ProjectSettings.buildUnity↩ SplashScreenColor.

4.69.1 Detailed Description

An object that persists start properties across domain reload, which is needed when configurable enter play mode is set to reload domain on enter play mode.

4.69.2 Member Data Documentation

4.69.2.1 fadeColor

```
Color?  fadeColor
```

The color for the fade out.

Unity splash screen color will be used if `null`.

4.69.2.2 fadeOutDuration

```
float fadeOutDuration = 1f
```

Specifies the duration for the fade in animation.

This would normally be 0 during first startup, then on restart it would be > 0.

4.69.3 Property Documentation

4.69.3.1 defaultProps

```
Props defaultProps = new()  [static], [get]
```

Gets the default Props.

Cannot be called during Object constructor.

4.70 QueueUtility< T >

A utility that provides queuing.

Static Public Member Functions

- static bool **IsQueued** (T queueable)

  Get if the item is queued.
- static bool **IsRunning** (T queueable)

  Gets if the item is running.
- static void **Stop** (T queueable)

  Cancels the queuable.
- static void **StopAll** ()

  Cancels all queued and running items.

Properties

- static bool **isBusy** `[get]`

  Gets whatever any items in the queue are running.
- static IEnumerable< T > **queue** `[get]`

  Gets the items currently in queue.
- static IEnumerable< T > **running** `[get]`

  Gets the items that are currently running.

Events

- static Action **queueEmpty**

  Occurs when an queued item finishes and queue is empty.
- static Action **queueFilled**

  Occurs when an queued is added.

4.70.1 Detailed Description

A utility that provides queuing.

Type Constraints

T **:** ***IQueueable***

4.71 ResolvedCrossReference

Represents a resolved reference.

Public Attributes

- CrossSceneReference reference

  The unresolved reference.
- ObjectReference reference

  The unresolved and resolved reference to the variable.
- ResolveStatus **result**

  The result when setting value.

4.71.1 Detailed Description

Represents a resolved reference.

4.71.2 Member Data Documentation

4.71.2.1 reference [1/2]

```
ObjectReference reference
```

The unresolved reference.

The unresolved and resolved reference to the value.

4.71.2.2 reference [2/2]

```
ObjectReference reference
```

The unresolved and resolved reference to the variable.

The unresolved and resolved reference to the value.

4.72 ResolvedReference

Represents a resolved ObjectReference.

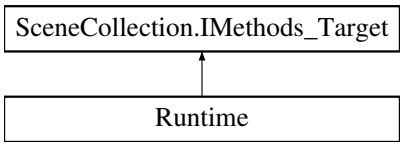
4.72.1 Detailed Description

Represents a resolved ObjectReference.

4.73 Runtime

Manages runtime functionality for Advanced Scene Manager such as open scenes and collection.

Inheritance diagram for Runtime:



Public Member Functions

- IEnumerable< SceneLoader > **GetToggleableSceneLoaders** ()

  Gets a list of all added scene loaders that can be toggled scene by scene.
- SceneLoader **GetLoaderForScene** (Scene scene)

  Gets the loader for scene .
- void **AddSceneLoader**< **T** > ()

  Adds a scene loader.
- void **RemoveSceneLoader**< **T** > ()

  Removes a scene loader.
- SceneOperation **Open** (params Scene[ ] scenes)
- SceneOperation Open (IEnumerable< Scene > scenes)

  Opens the scenes.
- SceneOperation **OpenWithLoadingScreen** (IEnumerable< Scene > scene, Scene loadingScreen)

  Opens a scene with a loading screen.
- SceneOperation **Close** (params Scene[ ] scenes)
- SceneOperation Close (IEnumerable< Scene > scenes)

  Closes the scenes.
- SceneOperation **FinishPreload** ()

  Finishes the preload of the currently preloaded scene.
- SceneOperation **DiscardPreload** (Scene scene)

  Discards preload of the scene, if preloaded.
- SceneOperation **DiscardPreload** ()

  Discards the preload of the currently preloaded scene.
- SceneOperation **ToggleOpenState** (Scene scene)
- SceneOperation **ToggleOpen** (Scene scene, bool? openState=null)

  Toggles the open state of this scene.
- void SetActive (Scene scene)

  Sets the scene as active.
- SceneOperation **Open** (SceneCollection collection, bool openAll=false)
- SceneOperation OpenAdditive (SceneCollection collection, bool openAll=false)

  Opens the collection without closing existing scenes.
- SceneOperation **Close** (SceneCollection collection)
- SceneOperation **ToggleOpenState** (SceneCollection collection, bool openAll=false)
- SceneOperation **ToggleOpen** (SceneCollection collection, bool? openState=null, bool openAll=false)
- SceneState **GetState** (Scene scene)

  Gets the current state of the scene.
- void Track (Scene scene, UnityEngine.SceneManagement.Scene unityScene)

Tracks the specified scene as open.

- void **Track** (Scene scene)
- bool Untrack (Scene scene)

  Untracks the specified scene as open.
- void UntrackScenes ()

  Untracks all open scenes.
- void Track (SceneCollection collection, bool isAdditive=false)

  Tracks the collection as open.
- void Untrack (SceneCollection collection, bool isAdditive=false)

  Untracks the collection.
- void UntrackCollections ()

  Untracks all collections.
- bool **IsTracked** (Scene scene)

  Gets whatever this scene is tracked as open.
- bool **IsTracked** (SceneCollection collection)

  Gets whatever this collection is tracked as open.
- SceneOperation **CloseAll** (bool exceptLoadingScreens=true, bool exceptUnimported=true, params Scene[] except)

  Closes all scenes and collections.

Public Attributes

- Action onAllScenesClosed

  Occurs when the last user scene closes.

Properties

- IEnumerable< Scene > **openScenes** `[get]`

  Gets the scenes that are open.
- IEnumerable< SceneCollection > **openAdditiveCollections** `[get]`

  Gets the collections that are opened as additive.
- SceneCollection **openCollection** `[get]`

  Gets the collection that is currently open.
- Scene **preloadedScene** `[get]`

  Gets the scene that is currently preloaded.
- Scene activeScene `[get]`

  Gets the active scene.
- Scene dontDestroyOnLoad `[get]`

  Gets the dontDestroyOnLoad scene.
- bool **isBusy** `[get]`

  Gets whatever ASM is busy with any scene operations.
- IEnumerable< SceneOperation > **runningOperations** `[get]`

  The currently running scene operations.
- IEnumerable< SceneOperation > **queuedOperations** `[get]`

  Gets the current scene operation queue.
- SceneOperation **currentOperation** `[get]`

  Gets the current active operation in the queue.

Events

- Action< Scene > **sceneOpened**

  Occurs when a scene is opened.
- Action< Scene > **sceneClosed**

  Occurs when a scene is closed.
- Action< SceneCollection > **collectionOpened**

  Occurs when a collection is opened.
- Action< SceneCollection > **collectionClosed**

  Occurs when a collection is closed.
- Action< Scene > **scenePreloaded**

  Occurs when a scene is preloaded.
- Action **startedWorking**

  Occurs when ASM has started working and is running scene operations.
- Action **stoppedWorking**

  Occurs when ASM has finished working and no scene operations are running.

4.73.1 Detailed Description

Manages runtime functionality for Advanced Scene Manager such as open scenes and collection.

4.73.2 Member Function Documentation

4.73.2.1 Close()

```
SceneOperation Close (
            IEnumerable< Scene > scenes )
```

Closes the scenes.

Closes persistent scenes.

4.73.2.2 Open()

```
SceneOperation Open (
            IEnumerable< Scene > scenes )
```

Opens the scenes.

Open scenes will not be re-opened, please close it first.

4.73.2.3 OpenAdditive()

```
SceneOperation OpenAdditive (
            SceneCollection collection,
            bool openAll = false )
```

Opens the collection without closing existing scenes.

Parameters

collection	The collection to open.
openAll	Specifies whatever all scenes should open, regardless of open flag.

Implements SceneCollection.IMethods_Target.

4.73.2.4 SetActive()

```
void SetActive (
            Scene scene )
```

Sets the scene as active.

No effect if not open.

4.73.2.5 Track() [1/2]

```
void Track (
            Scene scene,
            UnityEngine::SceneManagement::Scene unityScene )
```

Tracks the specified scene as open.

Does not open scene.

4.73.2.6 Track() [2/2]

```
void Track (
            SceneCollection collection,
            bool isAdditive = false )
```

Tracks the collection as open.

Does not open collection.

4.73.2.7 Untrack() [1/2]

```
bool Untrack (
            Scene scene )
```

Untracks the specified scene as open.

Does not close scene.

4.73.2.8 Untrack() [2/2]

```
void Untrack (
            SceneCollection collection,
            bool isAdditive = false )
```

Untracks the collection.

Does not close the collection.

4.73.2.9 UntrackCollections()

```
void UntrackCollections ( )
```

Untracks all collections.

Does not close collections.

4.73.2.10 UntrackScenes()

```
void UntrackScenes ( )
```

Untracks all open scenes.

Does not close scenes.

4.73.3 Member Data Documentation

4.73.3.1 onAllScenesClosed

```
Action onAllScenesClosed
```

Occurs when the last user scene closes.

This usually happens by mistake, and likely means that no user code would run, this is your chance to restore to a known state (return to main menu, for example), or crash to desktop.

Returning to main menu can be done like this:
`SceneManager.app.Restart()`

4.73.4 Property Documentation

4.73.4.1 activeScene

```
Scene activeScene  [get]
```

Gets the active scene.

Returns `null` if the active scene is not imported.

4.73.4.2 dontDestroyOnLoad

```
Scene dontDestroyOnLoad  [get]
```

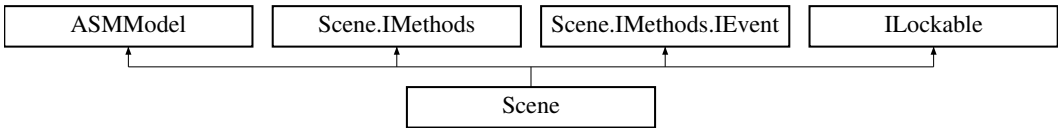
Gets the dontDestroyOnLoad scene.

Returns `null` outside of play mode.

4.74 Scene

Represents a scene.

Inheritance diagram for Scene:



Classes

- interface IMethods
- interface IMethods_Target

Public Member Functions

- IEnumerable< GameObject > GetRootGameObjects ()

  Gets the root game objects in this Scene.
- T FindObject< T > ()

  Finds the object in the hierarchy of this Scene.
- bool **FindObject< T >** (out T component)
- IEnumerable< T > FindObjects< T > ()

  Finds the objects in the hierarchy of this Scene.
- SceneOperation Open ()

  Opens the scene.
- SceneOperation ToggleOpen (bool? openState=null)

  Toggles this scene open or closed.
- SceneOperation Close ()

  Closes the scene.
- SceneOperation Preload (Action onPreloaded=null)

  Preloads the scene, to be displayed at a later time. See also: FinishPreload, DiscardPreload.
- SceneOperation FinishPreload ()

  Finishes preloading the scene, displaying it.
- SceneOperation **DiscardPreload** ()

  Discards the scene, if preloaded.
- SceneOperation **OpenWithLoadingScreen** (Scene loadingScreen)

  Opens the scene while a loading screen is open.
- void **SetActive** ()

Sets the scene as active in heirarchy.

- void **_Open** ()

  Event method. Its meant for UnityEngine.Events.UnityEvent.
- void **_ToggleOpenState** ()
- void **_ToggleOpen** (bool? openState=null)
- void **_Close** ()
- void **_Preload** ()
- void **_FinishPreload** ()
- void **_DiscardPreload** ()
- void **_OpenWithLoadingScreen** (Scene loadingScene)
- void **_SetActive** ()
- override bool **IsMatch** (string q)

  Gets if q matches ASMModel.name, id, path.
- T **UserData**< **T** > (T defaultValue=default)

  Gets user specified data.
- T **UserData**< **T** > (string key, T defaultValue=default)

  Gets user specified data.
- void **SetUserData**< **T** > (T value)

  Sets user specified data.
- void **SetUserData**< **T** > (string key, T value)

  Sets user specified data.
- void **UnsetUserData**< **T** > ()

  Unsets user specified data.
- void **UnsetUserData** (string key)

  Unsets user specified data.
- bool EvalOpenAsPersistent (SceneCollection parentCollection, SceneCollection collectionToOpen=null)

  Gets whatever this scene will be opened as persistent.
- void SetSceneLoader< T > ()

  Specifies the scene loader to use for this scene.
- Type **GetSceneLoader** ()

  Gets the scene loader specified for this scene. `null` *if none set.*
- SceneLoader **GetEffectiveSceneLoader** ()

  Gets the effective, contextual, scene loader for this scene. `null` *if none found (this means normal ASM loader will be used).*
- void **ClearSceneLoader** ()

  Clears custom scene loader for this scene. This means normal ASM functionality will be used.
- override string ToString (int indent)

Public Member Functions inherited from ASMModel

- virtual void Save ()

  Saves the scriptable object after modifying.
- void MarkAsDirty ()

  Mark scriptable object as dirty after modifying.
- override string **ToString** ()

  Gets a text summarization of this model.

Static Public Member Functions

- static IEnumerable< Scene > **Find** (Func< Scene, bool > predicate)

  Find scenes by predicate, in the specified collection or profile, if defined.
- static Scene **Find** (string q)

  Find scene by name, path, or id, in the specified collection or profile, if defined.
- static bool **TryFind** (string q, out Scene scene)

Static Public Attributes

- static readonly string **AssetSearchString** = "t:" + typeof(Scene).FullName

  Gets 't:AdvancedSceneManager.Models.Scene', the string to use in AssetDatabase.FindAssets(string).

Properties

- string **path** `[get, set]`

  Gets the path of the associated SceneAsset.
- bool isLoadingScreen `[get, set]`

  Gets if this scene is a loading screen.
- bool isSplashScreen `[get, set]`

  Gets if this scene is a splash screen.
- bool isSpecial `[get]`

  Gets if this is a 'special' scene.
- bool **isIncluded** `[get]`

  Gets whatever this scene is included in build.
- bool keepOpenWhenCollectionsClose `[get, set]`

  Specifies whatever this scene will remain open when collections close.
- bool keepOpenWhenNewCollectionWouldReopen `[get, set]`

  Specifies whatever this will remain open when a newly opened collection would have reopened it.
- bool **isNonPersistant** `[get]`

  Gets whatever this scene will close normally after a collection closes.
- bool openOnStartup `[get, set]`

  Specifies whatever this scene should be opened on startup.
- bool openOnPlayMode `[get, set]`

  Specifies whatever this scene should be opened when entering playmode.
- EditorPersistentOption **autoOpenInEditor** `[get, set]`

  Specifies whatever this scene should be opened automatically outside of play-mode.
- List< Scene > **autoOpenInEditorScenes** `[get]`

  Specifies the scenes that should trigger this scene to open when autoOpenInEditor is set to EditorPersistentOption.↩ WhenAnyOfTheFollowingScenesAreOpened.
- bool **isLocked** `[get, set]`

  Gets if this scene is locked.
- string **lockMessage** `[get, set]`

  Gets the lock message for this scene.
- bool **isActive** `[get]`

  Gets if this scene is currently active.
- bool **isOpenInHierarchy** `[get]`

  Gets whatever the scene is open in the hierarchy, this is `true` if scene is currently loading, if scene is preloaded, if scene is fully open..
- SceneState **state** `[get]`

- bool **isOpen** `[get]`
  Gets whatever the scene is open.
- bool **isPreloaded** `[get]`
  Gets whatever the scene is preloaded.
- unityScene? internalScene `[get, set]`
  Gets the unityScene that this scene is associated with.
- bool **isPersistent** `[get]`
  Gets if this scene is opened as persistent.
- bool **isDefaultScene** `[get]`
  Gets if this is a default ASM scene. These are located in '/AdvancedSceneManager/Defaults/'.
- bool **isDontDestroyOnLoad** `[get]`
  Gets if this scene is the dontDestroyOnLoad scene.
- bool **isDynamic** `[get]`
  Gets if this scene is dynamic, it is not persisted to disk.
- string **sceneLoader** `[get, set]`
  Specifies what SceneManagement.SceneLoader to use.

Properties inherited from ASMModel

- string **id** `[get]`
  Gets the id of this ASMModel.
- new string **name** `[get, protected set]`

Additional Inherited Members

Static Protected Member Functions inherited from ASMModel

- static T **CreateInternal**< **T** > (string name)
  Creates a profile. Throws if name is invalid.

4.74.1 Detailed Description

Represents a scene.

4.74.2 Member Function Documentation

4.74.2.1 Close()

```
SceneOperation Close ( )
```

Closes the scene.

No effect if scene is already closed.

Implements Scene.IMethods.

4.74.2.2 EvalOpenAsPersistent()

```
bool EvalOpenAsPersistent (
            SceneCollection parentCollection,
            SceneCollection collectionToOpen = null )
```

Gets whatever this scene will be opened as persistent.

Parameters

parentCollection	Specifies the parent collection that was opened before *finalCollection* .
collectionToOpen	Specifies the collection that will be opened, if you are not evaluating state after it would have opened, pass `null`. If multiple collections are opened in sequence, then pass the final one.

4.74.2.3 FindObject< T >()

```
T FindObject< T > ( )
```

Finds the object in the hierarchy of this Scene.

Only works if scene is loaded.

Type Constraints

T* : *Component

4.74.2.4 FindObjects< T >()

```
IEnumerable< T > FindObjects< T > ( )
```

Finds the objects in the hierarchy of this Scene.

Only works if scene is loaded.

Type Constraints

T* : *Component

4.74.2.5 FinishPreload()

```
SceneOperation FinishPreload ( )
```

Finishes preloading the scene, displaying it.

Scene must be preloaded beforehand.

Implements Scene.IMethods.

4.74.2.6 GetRootGameObjects()

```
IEnumerable< GameObject > GetRootGameObjects ( )
```

Gets the root game objects in this Scene.

Only usable if scene is open.

4.74.2.7 Open()

```
SceneOperation Open ( )
```

Opens the scene.

No effect if scene is already open.

Implements Scene.IMethods.

4.74.2.8 Preload()

```
SceneOperation Preload (
            Action onPreloaded = null )
```

Preloads the scene, to be displayed at a later time. See also: FinishPreload, DiscardPreload.

Scene must be closed beforehand.

Implements Scene.IMethods.

4.74.2.9 SetSceneLoader< T >()

```
void SetSceneLoader< T > ( )
```

Specifies the scene loader to use for this scene.

If the specified scene loader is not registered when scene is opened, then ASM will fallback to other scene loaders, if any (normal ASM functionality is used if not).

Type Constraints

T* : *SceneLoader

4.74.2.10 ToggleOpen()

```
SceneOperation ToggleOpen (
            bool?  openState = null )
```

Toggles this scene open or closed.

Parameters

openState	Specifies whatever you have a preferred state to toggle to, this means scene will not be closed if `true` is passed. This can be used to scene collection is open, without having an explicit check beforehand. The inverse is also the case for `false`.

Implements Scene.IMethods.

4.74.2.11 ToString()

```
override string ToString (
            int indent )  [virtual]
```

Parameters

indent	The indentation level, used for nested calls.

Parameters

indent	The indentation level, used for nested calls.

Reimplemented from ASMModel.

4.74.3 Property Documentation

4.74.3.1 internalScene

```
unityScene?  internalScene  [get], [set]
```

Gets the unityScene that this scene is associated with.

`null` if scene is not open.

4.74.3.2 isLoadingScreen

```
bool isLoadingScreen  [get], [set]
```

Gets if this scene is a loading screen.

Automatically updated.

If this is `false` for an actual loading screen, please make sure scene contains a Callbacks.LoadingScreen script.

Scene might sometimes have to be re-saved for this flag to appear.

Might not be 100% reliable.

4.74.3.3 isSpecial

```
bool isSpecial  [get]
```

Gets if this is a 'special' scene.

A scene is special if any of the following is `true`: isSplashScreen, isLoadingScreen or isDontDestroyOnLoad.

4.74.3.4 isSplashScreen

```
bool isSplashScreen  [get], [set]
```

Gets if this scene is a splash screen.

Automatically updated.

If this is `false` for an actual splash screen screen, please make sure scene contains a Callbacks.SplashScreen script.

Scene might sometimes have to be re-saved for this flag to appear.

Might not be 100% reliable.

4.74.3.5 keepOpenWhenCollectionsClose

```
bool keepOpenWhenCollectionsClose  [get], [set]
```

Specifies whatever this scene will remain open when collections close.

You'll have to close it manually, if needed.

4.74.3.6 keepOpenWhenNewCollectionWouldReopen

```
bool keepOpenWhenNewCollectionWouldReopen  [get], [set]
```

Specifies whatever this will remain open when a newly opened collection would have reopened it.

You'll have to close it manually, if needed.

4.74.3.7 openOnPlayMode

```
bool openOnPlayMode  [get], [set]
```

Specifies whatever this scene should be opened when entering playmode.

Only effective when scene added to Profile.standaloneScenes.

4.74.3.8 openOnStartup

```
bool openOnStartup  [get], [set]
```

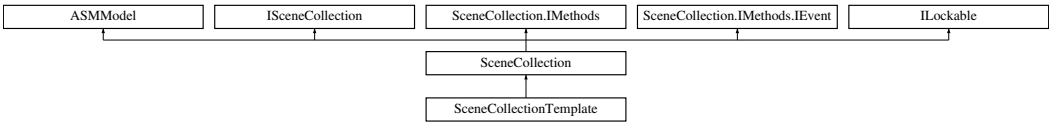
Specifies whatever this scene should be opened on startup.

Only effective when scene added to Profile.standaloneScenes.

4.75 SceneCollection

Represents a collection of scenes.

Inheritance diagram for SceneCollection:



Classes

- interface IMethods
- interface IMethods_Target

Public Member Functions

- override bool **IsMatch** (string q)

  Gets if q matches ASMModel.name.
- SceneOperation Open (bool openAll=false)

  Opens this collection.
- SceneOperation OpenAdditive (bool openAll=false)

  Opens this collection as additive.
- SceneOperation ToggleOpen (bool? openState=null, bool openAll=false)

  Toggles this collection open or closed.
- SceneOperation Close ()

  Closes this collection.
- void **_Open** (bool openAll=false)
- void **_OpenAdditive** (bool openAll=false)
- void **_ToggleOpenState** ()
- void **_ToggleOpen** (bool? openState=null)
- void **_Close** ()
- bool **FindProfile** (out Profile profile)

  Find the Profile that this collection is associated with.
- Profile **FindProfile** ()

  Find the Profile that this collection is associated with.
- T **UserData**< **T** > ()

  Casts and returns userData as the specified type. Returns null if invalid type.
- bool **Contains** (Scene scene)

  Gets if this collection contains scene .
- bool **AutomaticallyOpenScene** (Scene scene, bool? value=null)

  Gets or sets whatever the scene should automatically open, when this collection is open. Default is `true`.
- override string ToString (int indent)

Public Member Functions inherited from ASMModel

- virtual void Save ()

  Saves the scriptable object after modifying.
- void MarkAsDirty ()

  Mark scriptable object as dirty after modifying.
- override string **ToString** ()

  Gets a text summarization of this model.

Static Public Member Functions

- static SceneCollection **Find** (string q, bool activeProfile=true)

  Finds a collection based on its title or id.
- static bool **TryFind** (string q, out SceneCollection collection, bool activeProfile=true)

  Finds a collection based on its title or id.

Static Public Attributes

- static readonly string **AssetSearchString** = "t:" + typeof(SceneCollection).FullName

  Gets 't:AdvancedSceneManager.Models.SceneCollection', the string to use in AssetDatabase.FindAssets(string).

Properties

- int **count** `[get]`

  Gets the scene count of this collection.
- Scene **this[int index]** `[get]`

  Gets the scene at the specified index.
- string **title** `[get]`

  Gets the title of this collection.
- string **description** `[get, set]`

  Gets the description of this collection.
- IEnumerable< string > **scenePaths** `[get]`

  Gets the scenes of this collection.
- IEnumerable< Scene > **scenes** `[get]`

  Gets the scenes of this collection.
- IEnumerable< Scene > allScenes `[get]`

  Gets both scenes and loadingScreen.
- bool **hasScenes** `[get]`

  Gets if this collection has any scenes.
- bool isStartupCollection `[get]`

  Gets if this is a startup collection.
- ScriptableObject userData `[get, set]`

  The extra data that is associated with this collection.
- bool **isIncluded** `[get, set]`

  Gets whatever this collection should be included in build.
- Scene **loadingScreen** `[get, set]`

  The loading screen that is associated with this collection.
- Scene **effectiveLoadingScreen** `[get]`

  Gets effective loading screen depending on loadingScreenUsage.

- LoadingScreenUsage **loadingScreenUsage** `[get, set]`

  Specifies what loading screen to use.
- Scene **activeScene** `[get, set]`

  Specifies the scene that should be activated after collection is opened.
- CollectionStartupOption **startupOption** `[get, set]`

  Specifies startup option.
- CollectionLoadingThreadPriority **loadingPriority** `[get, set]`

  Specifies the thread priority to use when opening this collection.
- bool **openAsPersistent** `[get, set]`

  Specifies whatever this collection should be opened as persistent.
- bool **openAsDisabled** `[get, set]`

  Specifies whatever this collection should be opened as disabled.
- bool **unloadUnusedAssets** `[get, set]`

  Calls Resources.UnloadUnusedAssets after collection is opened or closed.
- List< Scene > **scenesThatShouldNotAutomaticallyOpen** `[get]`

  Specifies scenes that should not open automatically.
- InputBinding **binding** `[get]`

  Specifies bindings for this scene.
- bool **isLocked** `[get, set]`

  Gets if this collection is locked.
- string **lockMessage** `[get, set]`

  Gets the lock message for this collection.
- bool **isOpen** `[get]`

  Gets if this collection is open.
- bool **isOpenAdditive** `[get]`

  Gets if this collection is opened additively.
- bool **isOpenNonAdditive** `[get]`

  Gets if this collection is opened additively.

Properties inherited from ASMModel

- string **id** `[get]`

  Gets the id of this ASMModel.
- new string **name** `[get, protected set]`

Properties inherited from ISceneCollection

- string **id** `[get]`

  Gets the id of this collection.

Additional Inherited Members

Static Protected Member Functions inherited from ASMModel

- static T **CreateInternal**< **T** > (string name)

  Creates a profile. Throws if name is invalid.

4.75.1 Detailed Description

Represents a collection of scenes.

Only one collection can be open at a time.

4.75.2 Member Function Documentation

4.75.2.1 Close()

```
SceneOperation Close ( )
```

Closes this collection.

No effect if collection is already closed. Note that "additive collections" are not actually opened, only its scenes are.

Implements SceneCollection.IMethods.

4.75.2.2 Open()

```
SceneOperation Open (
            bool openAll = false )
```

Opens this collection.

Parameters

openAll	Specifies whatever scenes flagged to not open with collection, should.

Reopens all non-persistent scenes.

Implements SceneCollection.IMethods.

4.75.2.3 OpenAdditive()

```
SceneOperation OpenAdditive (
            bool openAll = false )
```

Opens this collection as additive.

Parameters

openAll	Specifies whatever scenes flagged to not open with collection, should.

Additive collections are not "opened", all scenes within are merely opened like normal scenes. Mostly intended for convenience.

Implements SceneCollection.IMethods.

4.75.2.4 ToggleOpen()

```
SceneOperation ToggleOpen (
            bool?  openState = null,
            bool openAll = false )
```

Toggles this collection open or closed.

Parameters

openState	Specifies whatever you have a preferred state to toggle to, this means collection will not be closed if `true` is passed. This can be used to ensure collection is open, without having an explicit check beforehand. The inverse is also the case for `false`.
openAll	Specifies whatever scenes flagged to not open with collection, should.

Implements SceneCollection.IMethods.

4.75.2.5 ToString()

```
override string ToString (
            int indent )  [virtual]
```

Parameters

indent	The indentation level, used for nested calls.

Parameters

indent	The indentation level, used for nested calls.

Reimplemented from ASMModel.

4.75.3 Property Documentation

4.75.3.1 allScenes

```
IEnumerable<Scene> allScenes  [get]
```

Gets both scenes and loadingScreen.

`null` is filtered out.

4.75.3.2 isStartupCollection

```
bool isStartupCollection  [get]
```

Gets if this is a startup collection.

Only available in editor.

4.75.3.3 userData

```
ScriptableObject userData  [get], [set]
```

The extra data that is associated with this collection.

Use UserData<T> to cast it to the desired type.

4.76 SceneCollectionExtensions

Provides utility methods for working with SceneCollection.

Static Public Member Functions

- static int IndexOf< T > (this T collection, Scene scene)
  Finds the index of scene .

4.76.1 Detailed Description

Provides utility methods for working with SceneCollection.

4.76.2 Member Function Documentation

4.76.2.1 IndexOf< T >()

```
static int IndexOf< T > (
            this T collection,
            Scene scene )  [static]
```

Finds the index of *scene* .

Returns -1 if it does not exist.

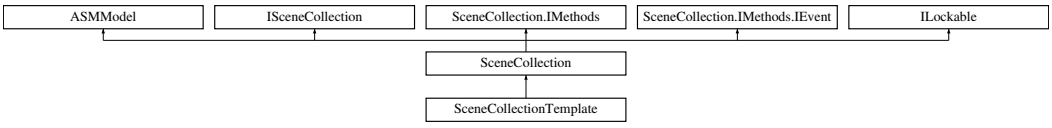
Type Constraints

T* : *ISceneCollection

4.77 SceneCollectionTemplate

Represents a template for a SceneCollection.

Inheritance diagram for SceneCollectionTemplate:



Properties

- new string **name** [get]

Properties inherited from SceneCollection

- int **count** [get]

  Gets the scene count of this collection.
- Scene **this[int index]** [get]

  Gets the scene at the specified index.
- string **title** [get]

  Gets the title of this collection.
- string **description** [get, set]

  Gets the description of this collection.
- IEnumerable< string > **scenePaths** [get]

  Gets the scenes of this collection.
- IEnumerable< Scene > **scenes** [get]

  Gets the scenes of this collection.
- IEnumerable< Scene > allScenes [get]

  Gets both scenes and loadingScreen.
- bool **hasScenes** [get]

  Gets if this collection has any scenes.
- bool isStartupCollection [get]

  Gets if this is a startup collection.
- ScriptableObject userData [get, set]

  The extra data that is associated with this collection.
- bool **isIncluded** [get, set]

  Gets whatever this collection should be included in build.
- Scene **loadingScreen** [get, set]

  The loading screen that is associated with this collection.
- Scene **effectiveLoadingScreen** [get]

  Gets effective loading screen depending on loadingScreenUsage.
- LoadingScreenUsage **loadingScreenUsage** [get, set]

  Specifies what loading screen to use.
- Scene **activeScene** [get, set]

  Specifies the scene that should be activated after collection is opened.
- CollectionStartupOption **startupOption** [get, set]

  Specifies startup option.
- CollectionLoadingThreadPriority **loadingPriority** [get, set]

  Specifies the thread priority to use when opening this collection.
- bool **openAsPersistent** [get, set]

  Specifies whatever this collection should be opened as persistent.
- bool **openAsDisabled** [get, set]

  Specifies whatever this collection should be opened as disabled.
- bool **unloadUnusedAssets** [get, set]

  Calls Resources.UnloadUnusedAssets after collection is opened or closed.
- List< Scene > **scenesThatShouldNotAutomaticallyOpen** [get]

  Specifies scenes that should not open automatically.
- InputBinding **binding** [get]

  Specifies bindings for this scene.

- bool **isLocked** `[get, set]`

  Gets if this collection is locked.
- string **lockMessage** `[get, set]`

  Gets the lock message for this collection.
- bool **isOpen** `[get]`

  Gets if this collection is open.
- bool **isOpenAdditive** `[get]`

  Gets if this collection is opened additively.
- bool **isOpenNonAdditive** `[get]`

  Gets if this collection is opened additively.

Properties inherited from ASMModel

- string **id** `[get]`

  Gets the id of this ASMModel.
- new string **name** `[get, protected set]`

Properties inherited from ISceneCollection

- string **id** `[get]`

  Gets the id of this collection.

Additional Inherited Members

Public Member Functions inherited from SceneCollection

- override bool **IsMatch** (string q)

  Gets if q matches ASMModel.name.
- SceneOperation Open (bool openAll=false)

  Opens this collection.
- SceneOperation OpenAdditive (bool openAll=false)

  Opens this collection as additive.
- SceneOperation ToggleOpen (bool? openState=null, bool openAll=false)

  Toggles this collection open or closed.
- SceneOperation Close ()

  Closes this collection.
- void **_Open** (bool openAll=false)
- void **_OpenAdditive** (bool openAll=false)
- void **_ToggleOpenState** ()
- void **_ToggleOpen** (bool? openState=null)
- void **_Close** ()
- bool **FindProfile** (out Profile profile)

  Find the Profile that this collection is associated with.
- Profile **FindProfile** ()

  Find the Profile that this collection is associated with.
- T **UserData**< **T** > ()

  Casts and returns userData as the specified type. Returns null if invalid type.
- bool **Contains** (Scene scene)

  Gets if this collection contains scene .
- bool **AutomaticallyOpenScene** (Scene scene, bool? value=null)

  Gets or sets whatever the scene should automatically open, when this collection is open. Default is `true`.
- override string ToString (int indent)

Public Member Functions inherited from ASMModel

- virtual void Save ()

  Saves the scriptable object after modifying.
- void MarkAsDirty ()

  Mark scriptable object as dirty after modifying.
- override string **ToString** ()

  Gets a text summarization of this model.

Static Public Member Functions inherited from SceneCollection

- static SceneCollection **Find** (string q, bool activeProfile=true)

  Finds a collection based on its title or id.
- static bool **TryFind** (string q, out SceneCollection collection, bool activeProfile=true)

  Finds a collection based on its title or id.

Static Public Attributes inherited from SceneCollection

- static readonly string **AssetSearchString** = "t:" + typeof(SceneCollection).FullName

  Gets 't:AdvancedSceneManager.Models.SceneCollection', the string to use in AssetDatabase.FindAssets(string).

Static Protected Member Functions inherited from ASMModel

- static T **CreateInternal**< **T** > (string name)

  Creates a profile. Throws if name is invalid.

4.77.1 Detailed Description

Represents a template for a SceneCollection.

4.78 SceneDataUtility

A utility for storing scene related data. Data can only be saved to disk in editor.

Static Public Member Functions

- static IEnumerable<(Scene scene, T data)> **Enumerate**< **T** > (string key)

  Enumerates data using the specified key on all scenes that uses it.
- static T **Get**< **T** > (Scene scene, string key, T defaultValue=default)

  Gets the value with the specified key, for the specified scene.
- static T **Get**< **T** > (string scene, string key, T defaultValue=default)

  Gets the value with the specified key, for the specified scene.
- static string **GetRaw** (Scene scene, string key)

  Gets the raw value with the specified key, for the specified scene.
- static void Set< T > (Scene scene, string key, T value)

  Sets the value with the specified key, for the specified scene.
- static void Set< T > (string scene, string key, T value)

  Sets the value with the specified key, for the specified scene.
- static void SetRaw (string scene, string key, string value)

  Sets the value with the specified key, for the specified scene. This is the direct version, all values are stores as string, which means Get<T>(string, string, T) must convert value beforehand, this method doesn't.
- static void Unset (Scene scene, string key)

  Unsets the value with the specified key, for the specified scene.

4.78.1 Detailed Description

A utility for storing scene related data. Data can only be saved to disk in editor.

4.78.2 Member Function Documentation

4.78.2.1 Set< T >() [1/2]

```
static void Set< T > (
            Scene scene,
            string key,
            T value )  [static]
```

Sets the value with the specified key, for the specified scene.

Changes will only be persisted in editor.

4.78.2.2 Set< T >() [2/2]

```
static void Set< T > (
            string scene,
            string key,
            T value )  [static]
```

Sets the value with the specified key, for the specified scene.

Changes will only be persisted in editor.

4.78.2.3 SetRaw()

```
static void SetRaw (
            string scene,
            string key,
            string value )  [static]
```

Sets the value with the specified key, for the specified scene. This is the direct version, all values are stores as string, which means Get<T>(string, string, T) must convert value beforehand, this method doesn't.

Changes will only be persisted in editor.

4.78.2.4 Unset()

```
static void Unset (
            Scene scene,
            string key )  [static]
```

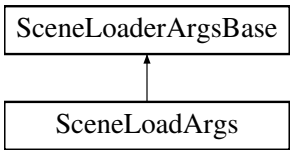
Unsets the value with the specified key, for the specified scene.

Changes will only be persisted in editor.

4.79 SceneLoadArgs

Specifies arguments for SceneLoader.LoadScene(Models.Scene, SceneLoadArgs).

Inheritance diagram for SceneLoadArgs:



Public Member Functions

- void SetCompleted (UnityEngine.SceneManagement.Scene scene)

  Notifies ASM that the load is done.
- void SetCompleted (UnityEngine.SceneManagement.Scene scene, Func< IEnumerator > preloadCallback)
- void **SetCompletedWithoutScene** ()

  Sets this loader as complete even though no scene was loaded.
- bool **CheckIsIncluded** (bool logError=true)

  Checks if the scene is actually included in build.
- UnityEngine.SceneManagement.Scene GetOpenedScene ()

  Gets the UnityEngine.SceneManagement.Scene that was opened by this override.

Properties

- bool **isPreload** `[get, set]`

  Specifies if the scene should be preloaded.

Properties inherited from SceneLoaderArgsBase

- bool **isLoadingScreen** `[get]`

  Gets if this scene is a loading screen.
- bool **isSplashScreen** `[get]`

  Gets if this scene is a splash screen.

4.79.1 Detailed Description

Specifies arguments for SceneLoader.LoadScene(Models.Scene, SceneLoadArgs).

4.79.2 Member Function Documentation

4.79.2.1 GetOpenedScene()

```
UnityEngine.SceneManagement.Scene GetOpenedScene ( )
```

Gets the UnityEngine.SceneManagement.Scene that was opened by this override.

Will return `default` if not found.

4.79.2.2 SetCompleted() [1/2]

```
void SetCompleted (
            UnityEngine::SceneManagement::Scene scene )
```

Notifies ASM that the load is done.

Parameters

scene	The opened scene.

4.79.2.3 SetCompleted() [2/2]

```
void SetCompleted (
            UnityEngine::SceneManagement::Scene scene,
            Func< IEnumerator > preloadCallback )
```

Parameters

scene	The opened scene.
preloadCallback	Specifies a callback that will be called when it is time to activate preloaded scene.

4.80 SceneLoader

Specifies a scene loader.

Inherited by RuntimeSceneLoader.

Public Member Functions

- virtual bool **CanOpen** (Scene scene)

  Gets whatever this scene loader can open the scene.
- IEnumerator **LoadScene** (Scene scene, SceneLoadArgs e)

  Loads the scene specified in e.scene.
- IEnumerator **UnloadScene** (Scene scene, SceneUnloadArgs e)

  Unloads the scene specified in e.scene.

Static Public Member Functions

- static string **GetKey**< **T** > ()

  Gets the key for the specified scene loader.
- static string **GetKey**< **T** > (T obj)

  Gets the key for the specified scene loader.

Properties

- string Key [get]

  Gets the key for this scene loader.
- virtual string **sceneToggleText** [get]

  Specifies the text to display on the toggle in scene popup. Only has an effect if isGlobal is `false`.
- virtual Indicator **indicator** [get]

  Specifies the indicator on scene fields for this scene loader.
- virtual bool isGlobal = true [get]

  Specifies if this scene loader will can be applied to all scenes. Otherwise scenes will have to be explicitly flagged to open with this loader.
- virtual bool **activeOutsideOfPlayMode** [get]

  Specifies whatever this loader will run outside of play mode or not.
- virtual bool **activeInPlayMode** = true [get]

  Specifies whatever this loader will run in play mode or not.
- bool **canBeActivated** [get]

  Gets whatever this loader may be activated in the current context.

4.80.1 Detailed Description

Specifies a scene loader.

4.80.2 Property Documentation

4.80.2.1 isGlobal

```
virtual bool isGlobal = true  [get]
```

Specifies if this scene loader will can be applied to all scenes. Otherwise scenes will have to be explicitly flagged to open with this loader.

To flag a scene to be opened with this loader, the following two methods can be used:

If sceneToggleText is non-empty, a toggle will be displayed in scene popup.

Programmatically Scene.SetSceneLoader<T> can be used.

4.80.2.2 Key

```
string Key  [get]
```

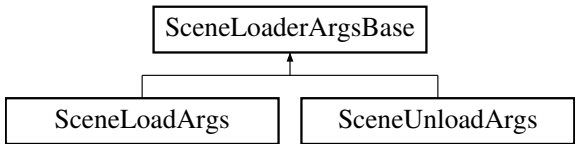
Gets the key for this scene loader.

This is equal to System.Type.FullName.

4.81 SceneLoaderArgsBase

Base class for SceneLoadArgs and SceneUnloadArgs.

Inheritance diagram for SceneLoaderArgsBase:



Properties

- bool **isLoadingScreen** `[get]`

  Gets if this scene is a loading screen.
- bool **isSplashScreen** `[get]`

  Gets if this scene is a splash screen.

4.81.1 Detailed Description

Base class for SceneLoadArgs and SceneUnloadArgs.

4.82 SceneManager

The central Advanced Scene Manager API. Provides access to the most important things in ASM.

Static Public Member Functions

- static void OnInitialized (Action action)

  Call action when ASM has initialized.

Properties

- static AssetsProxy **assets** = new() `[get]`
- static IEnumerable< Scene > **openScenes** `[get]`
- static SceneCollection **openCollection** `[get]`
- static Scene **preloadedScene** `[get]`
- static Runtime **runtime** = new() `[get]`
- static App **app** = new App() `[get]`
- static SettingsProxy **settings** = new() `[get]`
- static Profile **profile** `[get]`
- static bool isInitialized `[get]`

  Gets whatever ASM is initialized. Calling ASM methods may fail if `false`, this is due to ASMSettings singleton not being loaded yet.

4.82.1 Detailed Description

The central Advanced Scene Manager API. Provides access to the most important things in ASM.

4.82.2 Member Function Documentation

4.82.2.1 OnInitialized()

```
static void OnInitialized (
            Action action )  [static]
```

Call *action* when ASM has initialized.

Will call immediately if already initialized.

4.82.3 Property Documentation

4.82.3.1 isInitialized

```
bool isInitialized  [static], [get]
```

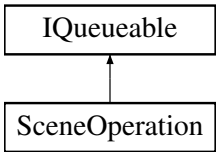
Gets whatever ASM is initialized. Calling ASM methods may fail if `false`, this is due to ASMSettings singleton not being loaded yet.

See also OnInitialized(Action).

4.83 SceneOperation

A scene operation is a queueable operation that can open or close scenes. See also: SceneAction.

Inheritance diagram for SceneOperation:



Public Member Functions

- SceneOperation **WithFriendlyText** (string text)

  Specifies description for coroutine.
- void Cancel ()

  Cancel this operation.
- SceneOperation **With** (SceneCollection collection, bool setActiveScene=false)

  Specifies an associated collection.
- SceneOperation **Open** (SceneCollection collection, bool openAll=false)
- SceneOperation **Close** (SceneCollection collection)
- SceneOperation Focus (Scene scene)

  Sets focus to the specified scene. Overrides selected scene in collections.
- SceneOperation Open (params Scene[ ] scenes)

  Specifies the scenes to open.
- SceneOperation Close (params Scene[ ] scenes)

  Specifies the scenes to close.
- SceneOperation **Callback** (params Callback[ ] callbacks)

  Specifies user callbacks.
- SceneOperation Preload (Scene scene)

  Specifies a scene to preload.
- SceneOperation **Open** (IEnumerable< Scene > scenes)
- SceneOperation **Close** (IEnumerable< Scene > scenes)
- SceneOperation **Callback** (IEnumerable< Callback > callbacks)
- SceneOperation With (Scene loadingScene)

  Specifies loading screen to use.
- SceneOperation **With** (Action< LoadingScreen > loadingScreenCallback)

  Specifies a callback when loading screen is opened, before LoadingScreen.OnOpen is called.
- SceneOperation **DisableLoadingScreen** (bool useLoadingScene=false)

  Specifies whatever loading screen should be used.
- SceneOperation **EnableLoadingScreen** (bool useLoadingScene=true)

  Specifies whatever loading screen should be used.
- SceneOperation **With** (ThreadPriority loadingPriority)

  Specifies the ThreadPriority to use.
- SceneOperation **UnloadUsedAssets** ()

  Specifies whatever Resources.UnloadUnusedAssets should be called at the end (before loading screen).
- SceneOperation **With** (Progress< float > customProgress)

  Specifies custom progress that will be counted as part of progress.
- void **CloseAll** (params Scene[ ] except)

  Closes all scenes prior to opening any scenes.
- void **CloseAllNonPersistent** (params Scene[ ] except)

  Closes all non-persistent scenes prior to opening any scenes.
- SceneOperation ReportProgress (Action< float > progress)

  Specifies a callback for when progress changes.

Static Public Member Functions

- static SceneOperation **Queue** ()

  Queues a new scene operation.
- static SceneOperation **Start** ()

  Starts a new scene operation, ignoring queue.
- static void **AddCallback** (Callback callback)

  Adds the callback to every scene operation.
- static void **RemoveCallback** (Callback callback)

  Removes a callback that was added to every scene operation.

Properties

- static SceneOperation **done** = new SceneOperation() `[get]`

  Gets a SceneOperation that has already completed.
- string **description** `[get, protected set]`

  Specifies description for coroutine.
- SceneCollection **collection** `[get]`

  Specifies the collection that is being opened or closed.
- bool **setActiveCollectionScene** `[get]`

  Specifies whatever active scene should be set when possible.
- Scene focus `[get]`

  Sets focus to the specified scene. Overrides selected scene in collections.
- IEnumerable< Scene > open `[get]`

  Gets the scenes specified to open.
- IEnumerable< Scene > close `[get]`

  Gets the scenes specified to close.
- IEnumerable< Callback > **callbacks** `[get]`

  Gets the user defined callbacks.
- Scene **preload** `[get]`

  Gets the scene specified to preload.
- Scene **loadingScene** `[get]`

  Gets the specified loading screen.
- Action< LoadingScreen > **loadingScreenCallback** `[get]`

  Gets the specified loading screen callback.
- bool **useLoadingScene** `[get]`

  Gets whatever a loading screen should be used.
- ThreadPriority? **loadingPriority** `[get]`

  Gets the specified ThreadPriority to be used.
- bool? **unloadUnusedAssets** `[get]`

  Gets whatever Resources.UnloadUnusedAssets should be called at the end (before loading screen).
- IEnumerable< Scene > **closedScenes** `[get]`

  Gets the scenes that was closed during this operation.
- IEnumerable< Scene > **openedScenes** `[get]`

  Gets the scenes that was opened during this operation.
- bool **isLoadingScreen** `[get, set]`

  Specifies whatever this scene operation was started by ASM to open a loading screen.
- override bool **keepWaiting** `[get]`

  Inherited from CustomYieldInstruction. Tells unity whatever the operation is done or not.
- Phase **phase** `[get]`

  The phase the this scene operation is currently in.
- bool **wasCancelled** `[get]`

  Gets if this scene operation is cancelled.
- Progress< float > **customProgress** `[get]`

  Gets custom progress, if there is any. Will be counted as part of progress.
- LoadingScreen **openedLoadingScreen** `[get]`

  Gets the loading screen that was opened by this operation.
- float **progress** `[get]`

  Gets the current progress.

Events

- static OnBeforeStart **beforeStart**

  Occurs before operation has started working.

4.83.1 Detailed Description

A scene operation is a queueable operation that can open or close scenes. See also: SceneAction.

4.83.2 Member Function Documentation

4.83.2.1 Cancel()

```
void Cancel ( )
```

Cancel this operation.

Note that the operation might not be cancelled immediately, if user defined callbacks are currently running.

4.83.2.2 Close()

```
SceneOperation Close (
            params Scene[] scenes )
```

Specifies the scenes to close.

Can be called multiple times to add more scenes.

4.83.2.3 Focus()

```
SceneOperation Focus (
            Scene scene )
```

Sets focus to the specified scene. Overrides selected scene in collections.

No effect if preloading.

4.83.2.4 Open()

```
SceneOperation Open (
            params Scene[] scenes )
```

Specifies the scenes to open.

Can be called multiple times to add more scenes.

4.83.2.5 Preload()

```
SceneOperation Preload (
            Scene scene )
```

Specifies a scene to preload.

A scene specified to preload cannot also be added to open or close lists.

4.83.2.6 ReportProgress()

```
SceneOperation ReportProgress (
            Action< float > progress )
```

Specifies a callback for when progress changes.

Only one callback can be registered, previous one will be replaced by *progress* .

4.83.2.7 With()

```
SceneOperation With (
            Scene loadingScene )
```

Specifies loading screen to use.

Has no effect if useLoadingScene is `false`.

4.83.3 Property Documentation

4.83.3.1 close

```
IEnumerable<Scene> close  [get]
```

Gets the scenes specified to close.

List will change depending on when its called (i.e. only open scenes can be closed).

4.83.3.2 focus

```
Scene focus  [get]
```

Sets focus to the specified scene. Overrides selected scene in collections.

No effect if preloading.

4.83.3.3 open

```
IEnumerable<Scene> open  [get]
```

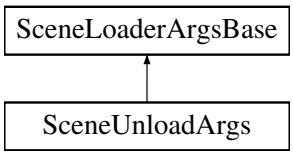
Gets the scenes specified to open.

List will change depending on when its called (i.e. only closed scenes can be opened).

4.84 SceneUnloadArgs

Specifies arguments for SceneLoader.UnloadScene(Models.Scene, SceneUnloadArgs).

Inheritance diagram for SceneUnloadArgs:



Public Member Functions

- void **SetCompleted** ()

  Notifies ASM that the unload is done.

Additional Inherited Members

Properties inherited from SceneLoaderArgsBase

- bool **isLoadingScreen** `[get]`

  Gets if this scene is a loading screen.
- bool **isSplashScreen** `[get]`

  Gets if this scene is a splash screen.

4.84.1 Detailed Description

Specifies arguments for SceneLoader.UnloadScene(Models.Scene, SceneUnloadArgs).

4.85 SceneUtility

An utility class to perform actions on scenes.

Static Public Member Functions

- static IEnumerable< scene > **GetAllOpenUnityScenes** ()

  Get all open unity scenes.
- static bool **IsIncluded** (Scene scene)

  Gets if the scene is included in build.
- static void **Move** (this GameObject obj, Scene scene)
- static void **Move** (this GameObject obj, scene scene)
- static Scene CreateDynamic (string name, UnityEngine.SceneManagement.LocalPhysicsMode local↩ PhysicsMode=UnityEngine.SceneManagement.LocalPhysicsMode.None)

  Creates a scene at runtime, that is not saved to disk.
- static bool **FindCollection** (this Scene scene, out SceneCollection collection)
- static SceneCollection FindCollection (this Scene scene)

  Attempts to find best match for collection.
- static IEnumerable< SceneCollection > **FindCollections** (this Scene scene, bool allProfiles=false)

  Finds which collections that this scene is a part of.
- static IEnumerable< SceneCollection > **FindCollections** (this Scene scene, Profile profile)

  Finds which collections that this scene is a part of.
- static IEnumerable< Scene > **FindOpen** (string q)

  Find open scenes by name or path.
- static Scene **Find** (string q)

  Find scenes by name or path, in the specified collection or profile, if defined.
- static IEnumerable< Scene > **FindOpen** (Func< Scene, bool > predicate)

  Find open scenes by predicate.
- static IEnumerable< Scene > **Find** (Func< Scene, bool > predicate)

  Find scenes by predicate, in the specified collection or profile, if defined.
- static bool **ASMScene** (this Component component, out Scene scene)
- static Scene **ASMScene** (this GameObject gameObject, out Scene scene)
- static Scene **ASMScene** (this Component component)
- static Scene **ASMScene** (this GameObject gameObject)
- static bool **ASMScene** (this scene thisScene, out Scene scene)
- static Scene **ASMScene** (this scene scene)

  Gets the associated ASM Scene.
- static IEnumerable< Scene > EvaluateFinalSceneList (Profile profile, App.Props props)

  Evaluate the final scene list after startup.
- static IEnumerable< Scene > EvaluateFinalSceneList (IEnumerable< SceneCollection > collections)

  Evaluate the final scene list after opening a sequence of collections.
- static void SetEnabled (this Scene scene, bool isEnabled)

  Sets all root objects as enabled / disabled.
- static void Enable (this Scene scene)

  Sets all root objects as enabled.
- static void Disable (this Scene scene)

  Sets all root objects as disabled.

Properties

- static bool **isStartupScene** `[get]`

  Gets if current, and only, scene is the startup scene.
- static scene **dontDestroyOnLoadScene** `[get]`

  Gets the dontDestroyOnLoad scene. Returns null if not open.
- static bool **hasAnyScenes** `[get]`

  Gets if there are any scenes open that are not dynamically created, and not yet saved to disk.
- static int **unitySceneCount** `[get]`

4.85.1 Detailed Description

An utility class to perform actions on scenes.

4.85.2 Member Function Documentation

4.85.2.1 CreateDynamic()

```
static Scene CreateDynamic (
            string name,
            UnityEngine::SceneManagement::LocalPhysicsMode localPhysicsMode = UnityEngine:↩
:SceneManagement::LocalPhysicsMode::None )  [static]
```

Creates a scene at runtime, that is not saved to disk.

Returns `null` if scene could not be created.

4.85.2.2 Disable()

```
static void Disable (
            this Scene scene )  [static]
```

Sets all root objects as disabled.

Only has an effect if scene is open.

4.85.2.3 Enable()

```
static void Enable (
            this Scene scene )  [static]
```

Sets all root objects as enabled.

Only has an effect if scene is open.

4.85.2.4 EvaluateFinalSceneList() [1/2]

```
static IEnumerable< Scene > EvaluateFinalSceneList (
            IEnumerable< SceneCollection > collections )  [static]
```

Evaluate the final scene list after opening a sequence of collections.

Parameters

collections	The sequence of collections that would be opened.

4.85.2.5 EvaluateFinalSceneList() [2/2]

```
static IEnumerable< Scene > EvaluateFinalSceneList (
            Profile profile,
            App::Props props )  [static]
```

Evaluate the final scene list after startup.

Parameters

profile	The profile that would be used to run startup process with.
props	The startup props that would be used to run process with.

4.85.2.6 FindCollection()

```
static SceneCollection FindCollection (
            this Scene scene )  [static]
```

Attempts to find best match for collection.

Only checks current profile.

4.85.2.7 SetEnabled()

```
static void SetEnabled (
            this Scene scene,
            bool isEnabled )  [static]
```

Sets all root objects as enabled / disabled.

Only has an effect if scene is open.

4.86 ScriptableObjectUtility

Contains utility methods for ScriptableObject.

Static Public Member Functions

- static void Save (this ScriptableObject obj)

  Saves the ScriptableObject.

4.86.1 Detailed Description

Contains utility methods for ScriptableObject.

4.86.2 Member Function Documentation

4.86.2.1 Save()

```
static void Save (
            this ScriptableObject obj )  [static]
```

Saves the ScriptableObject.

Safe to call from outside editor, but has no effect.

4.87 SerializableDictionary< TKey, TValue >

A serializable dictionary.

Inherits Dictionary< TKey, TValue >, and ISerializationCallbackReceiver.

4.87.1 Detailed Description

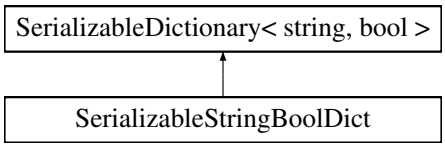
A serializable dictionary.

Older unity versions might need a wrapper class, since they won't support serializing generic types. Don't forget SerializableAttribute on wrapper!

4.88 SerializableStringBoolDict

A serializable dictionary of string and bool.

Inheritance diagram for SerializableStringBoolDict:



4.88.1 Detailed Description

A serializable dictionary of string and bool.

4.89 SettingsProxy

Provides access to ASM settings.

Properties

- ASMSettings **project** `[get]`

  The project-wide ASM settings.
- Profile profile `[get]`

  The current ASM profile.

4.89.1 Detailed Description

Provides access to ASM settings.

4.89.2 Property Documentation

4.89.2.1 profile

```
Profile profile [get]
```

The current ASM profile.

Could be `null`.

4.90 SpamCheck

Provides an easy way to check for spamming.

Public Member Functions

- bool **IsSpam** ()

  Gets if an action was executed not long enough ago.
- void **MarkAsExecuted** ()

  Marks this spam check as executed, disallowing any actions until cooldown has run out.
- void **Execute** (Action action)

  Runs action if allowed.

Properties

- static SpamCheck Global = new SpamCheck() `[get]`

  Gets the global spam check.
- bool isEnabled `[get, set]`

  Gets or sets if this SpamCheck is enabled.
- float **executeCooldown** `[get, set]`

  Gets or sets the cooldown.
- float **lastExecute** `[get]`

  Gets the time an action was executed last.
- float **timeSinceLastExecute** `[get]`

  Gets the time an action was executed last.

4.90.1 Detailed Description

Provides an easy way to check for spamming.

4.90.2 Property Documentation

4.90.2.1 Global

```
SpamCheck Global = new SpamCheck()  [static], [get]
```

Gets the global spam check.

Don't worry about conflicts with ASM stuff, we use a separate one.

4.90.2.2 isEnabled

```
bool isEnabled  [get], [set]
```

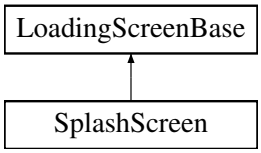
Gets or sets if this SpamCheck is enabled.

When disabled actions will run without checking whatever it is a spam call.

4.91 SplashScreen

A class that contains callbacks for splash screens.

Inheritance diagram for SplashScreen:



Public Member Functions

- IEnumerator DisplaySplashScreen ()
  Called when scene manager is ready to display the splash screen.
- override IEnumerator **OnOpen** ()
  Called when the splash screen is opened.
- override IEnumerator OnClose ()
  Called when the loading screen is about to close.

Public Member Functions inherited from LoadingScreenBase

- IEnumerator **OnOpen** ()
  Called when the loading screen is opened.
- IEnumerator **OnClose** ()
  Called when the loading screen is about to close.

Additional Inherited Members

Public Attributes inherited from LoadingScreenBase

- Action< LoadingScreenBase > **onDestroy**

  Occurs when loading screen is destroyed.
- Canvas canvas

4.91.1 Detailed Description

A class that contains callbacks for splash screens.

SplashScreen and LoadingScreen cannot coexist within the same scene.

4.91.2 Member Function Documentation

4.91.2.1 DisplaySplashScreen()

```
IEnumerator DisplaySplashScreen ( )  [abstract]
```

Called when scene manager is ready to display the splash screen.

Example: yielding new WaitForSeconds(5) will show the splash screen for 5 seconds.

4.91.2.2 OnClose()

```
override IEnumerator OnClose ( )
```

Called when the loading screen is about to close.

Calls DisplaySplashScreen, so make sure to call it manually or call base if overridden.

4.92 StandaloneCollection

Represents a collection of standalone scenes. These scenes are guaranteed to be included in build (if the associated Profile is active).

Inherits ISceneCollection.IEditable.

Properties

- string **id** [get]

  Gets the id of this collection.
- IEnumerable< Scene > **scenes** [get]

  Gets the scenes of this collection.
- IEnumerable< string > **scenePaths** [get]

  Gets the scenes of this collection.
- string **title** [get]

  Gets the title of this collection.
- string **description** [get]

  Gets the description of this collection.
- IEnumerable< Scene > **startupScenes** [get]

  Gets all scenes that will be opened on startup.
- int **count** [get]

  Gets the scene count of this collection.
- Scene **this[int index]** [get]

  Gets the scene at the specified index.

4.92.1 Detailed Description

Represents a collection of standalone scenes. These scenes are guaranteed to be included in build (if the associated Profile is active).

Usage: Profile.standaloneScenes.

Index