






# UltimatePooling Namespace

## ▲ Classes

	Class	Description
	<a href="#">GenericPoolGroup</a>	Represents a pool group that can accept either a game object or component prefab as its root prefab.
	<a href="#">PoolBehaviour</a>	Intermediate behaviour script that allows spawn and despawn events to be received. The events are broadcast to the object that is being re-used so the script can be at any level on the objects hierarchy.
	<a href="#">PoolGroup</a>	Represents a spawn pool for a specific type of prefab. Valid types are game objects and components.
	<a href="#">PoolManager</a>	The manager that is responsible for all pool groups and handles the creation and destruction of pools at runtime.
	<a href="#">ResourcesPoolGroup</a>	Represents a pool group that manages a prefab object located within the resources folder.



## UltimatePool

The main class or interacting with the UltimatePooling API. All spawning and despawning methods are found in this class however you can use the individual spawn method on pools if required.

## ▲ Interfaces

	Interface	Description
	<a href="#">IPoolReceiver</a>	Implement this interface when you want to receive spawned and despawned events sent to the pooled object. This interface will typically be implemented by a mono behaviour script that is attached to a pooled object which will then receive the appropriate event when it is spawned or despawned. Alternativley you can inherit from <a href="#">PoolBehaviour</a> which provides default overridable behaviour for these events. (Modifies the objects enabled state to show or hide the object).

## ▲ Enumerations

	Enumeration	Description
	<a href="#">LogLevel</a>	The amount of detail to include in logged messages.
	<a href="#">PoolEventType</a>	The method that is used to inform pooled objects about their current spawn state.



# GenericPoolGroup Class

Represents a pool group that can accept either a game object or component prefab as its root prefab.

## ▲ Inheritance Hierarchy

[SystemObject](#) **Object**  
**Component**  
**Behaviour**  
**MonoBehaviour**  
[UltimatePoolingPoolGroup](#)  
[UltimatePoolingGenericPoolGroup](#)  
[UltimatePoolingResourcesPoolGroup](#)

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax


C#    JavaScript

[Copy](#)

```
public class GenericPoolGroup : PoolGroup
```



The [GenericPoolGroup](#) type exposes the following members.

## ▲ Constructors

Name	Description
 <a href="#">GenericPoolGroup</a>	Initializes a new instance of the <a href="#">GenericPoolGroup</a> class


[Top](#)

## Methods

	Name	Description
	<a href="#">onInstanceDespawned</a>	Handle despawning of a pooled object. By default, this method disables the game object. (Overrides <a href="#">PoolGroup.onInstanceDespawned(Object, PoolEventType)</a> .)
	<a href="#">onInstanceSpawned</a>	Handle the spawning of a pooled object. By default, this method enables the game object. (Overrides <a href="#">PoolGroup.onInstanceSpawned(Object, PoolEventType, Vector3, Quaternion)</a> .)

[Top](#)

## Properties

	Name	Description
	<a href="#">Prefab</a>	Access the component or game object prefab. (Overrides <a href="#">PoolGroup.Prefab</a> .)

[Top](#)

## See Also

Reference

[UltimatePooling Namespace](#)

# GenericPoolGroup Constructor

Initializes a new instance of the [GenericPoolGroup](#) class

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public GenericPoolGroup()
```

## ▲ See Also

Reference



[GenericPoolGroup Class](#)

[UltimatePooling Namespace](#)

# GenericPoolGroup Methods

The [GenericPoolGroup](#) type exposes the following members.

## ▲ Methods

Name	Description
 <a href="#">onInstanceDespawned</a>	Handle despawning of a pooled object. By default, this method disables the game object. (Overrides <a href="#">PoolGroup.onInstanceDespawned(Object, PoolEventType)</a> .)
 <a href="#">onInstanceSpawned</a>	Handle the spawning of a pooled object. By default, this method enables the object. (Overrides <a href="#">PoolGroup.onInstanceSpawned(Object, PoolEventType, Vector3, Quaternion)</a> .)

[Top](#)

## ▲ See Also

Reference

[GenericPoolGroup Class](#)

[UltimatePooling Namespace](#)

# GenericPoolGrouponInstanceDespa Method

Handle despawning of a pooled object. By default, this method disables the game object.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
protected override void onInstanceDespawned(  
    Object instance,  
    PoolEventType type  
)
```

## Parameters

*instance*

Type: **Object**

The instance to handle the despawning of

*type*

Type: [UltimatePoolingPoolEventType](#)

The type of event used to inform the object of its spawned status

## ▲ See Also

### Reference

[GenericPoolGroup Class](#)

[UltimatePooling Namespace](#)



# GenericPoolGrouponInstanceSpawr Method

Handle the spawning of a pooled object. By default, this method enabled the game object.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
protected override void onInstanceSpawned(  
    Object instance,  
    PoolEventType type,  
    Vector3 position,  
    Quaternion rotation  
)
```

## Parameters

*instance*

Type: **Object**

The newly spawned instance to handle

*type*

Type: [UltimatePoolingPoolEventType](#)

The type of event used to inform the object of its spawn status

*position*

Type: **Vector3**

The position to spawn the object at

*rotation*

Type: **Quaternion**

The rotation to spawn the object with

## ▲ See Also

### Reference

[GenericPoolGroup Class](#)


[UltimatePooling Namespace](#)

---

# GenericPoolGroup Properties

The [GenericPoolGroup](#) type exposes the following members.

## ▲ Properties

	Name	Description
	<a href="#">Prefab</a>	Access the component or game object prefab. (Overrides <a href="#">PoolGroupPrefab</a> .)

[Top](#)

## ▲ See Also

Reference

[GenericPoolGroup Class](#)

[UltimatePooling Namespace](#)

# GenericPoolGroupPrefab Property

Access the component or game object prefab.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public override Object Prefab { get; set; }
```

Property Value

Type: **Object**

## ▲ See Also

Reference

[GenericPoolGroup Class](#)

[UltimatePooling Namespace](#)

# IPoolReceiver Interface

Implement this interface when you want to receive spawned and despawned events sent to the pooled object. This interface will typically be implemented by a mono behaviour script that is attached to a pooled object which will then receive the appropriate event when it is spawned or despawned. Alternativley you can inherit from [PoolBehaviour](#) which provides default overridable behaviour for these events. (Modifies the objects enabled state to show or hide the object).

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax



C#    JavaScript

[Copy](#)

```
public interface IPoolReceiver
```

The [IPoolReceiver](#) type exposes the following members.

## ▲ Methods

	Name	Description
	<a href="#">OnDespawned</a>	Called when an object is about to be returned to the pool. Note that this method will not be called when the object is destroyed.
	<a href="#">OnSpawned</a>	Called when an object has been spawned from the pool. This event allows the state of the object to be reset so it can be treaded as a newly created object. Note that this method

will not be called when the object is first created.

---

[Top](#)

## ▲ See Also

Reference



[UltimatePooling Namespace](#)

---

# IPoolReceiver Methods

The [IPoolReceiver](#) type exposes the following members.

## ▲ Methods

	Name	Description
	<a href="#">OnDespawned</a>	Called when an object is about to be returned to the pool. Note that this method will not be called when the object is destroyed.
	<a href="#">OnSpawned</a>	Called when an object has been spawned from the pool. This event allows the state of the object to be reset so it can be treated as a newly created object. Note that this method will not be called when the object is first created.

[Top](#)

## ▲ See Also

Reference

[IPoolReceiver Interface](#)

[UltimatePooling Namespace](#)

# IPoolReceiverOnDespawned Method

Called when an object is about to be returned to the pool. Note that this method will not be called when the object is destroyed.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
void OnDespawned(  
    PoolGroup pool  
)
```

## Parameters

*pool*

Type: [UltimatePoolingPoolGroup](#)

## ▲ See Also

### Reference

[IPoolReceiver Interface](#)

[UltimatePooling Namespace](#)



# IPoolReceiverOnSpawned Method

Called when an object has been spawned from the pool. This event allows the state of the object to be reset so it can be treaded as a newly created object. Note that this method will not be called when the object is first created.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
void OnSpawned(  
    PoolGroup pool  
)
```

## Parameters

*pool*

Type: [UltimatePoolingPoolGroup](#)

## ▲ See Also

### Reference

[IPoolReceiver Interface](#)

[UltimatePooling Namespace](#)

# LogLevel Enumeration

The amount of detail to include in logged messages.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public enum LogLevel
```

## ▲ Members

Member name	Value	Description
<a href="#">None</a>	0	Dont log anything to the console.
<a href="#">Error</a>	1	Only log error messages to the console.
<a href="#">Warning</a>	2	Log errors and warnings to the console.
<a href="#">Message</a>	3	Log all message types to the console, including errors and warnings.

## ▲ See Also

Reference

[UltimatePooling Namespace](#)



# PoolBehaviour Class

Intermediate behaviour script that allows spawn and despawn events to be received. The events are broadcast to the object that is being re-used so the script can be at any level on the objects hierarchy.

## ▾ Inheritance Hierarchy

[SystemObject](#) **Object**  
**Component**  
**Behaviour**  
**MonoBehaviour**  
[UltimatePoolingPoolBehaviour](#)

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▾ Syntax


C#    JavaScript

[Copy](#)

```
public class PoolBehaviour : MonoBehaviour,
    IPoolReceiver
```



The [PoolBehaviour](#) type exposes the following members.

## ▾ Constructors

	Name	Description
	<a href="#">PoolBehaviour</a>	Initializes a new instance of the <a href="#">PoolBehaviour</a> class



[Top](#)

## ▲ Methods

	Name	Description
	<a href="#">OnDespawned</a>	Called by the managing pool to notify that this object is about to be returned to the pool. This method will not be called when the object is about to be destroyed.
	<a href="#">OnSpawned</a>	Called by the managing pool to notify that this object has just been recycled from the pool. This method will not be called when the object is created for the first time.

[Top](#)

## ▲ Fields

	Name	Description
	<a href="#">monoDespawnedEvent</a>	The name of the event that is called when an object is returned to the pool.
	<a href="#">monoSpawnedEvent</a>	The name of the event that is called when an object is spawned from the pool.

[Top](#)

## ▲ See Also

Reference

[UltimatePooling Namespace](#)

# PoolBehaviour Constructor

Initializes a new instance of the [PoolBehaviour](#) class

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public PoolBehaviour()
```

## ▲ See Also

Reference



[PoolBehaviour Class](#)

[UltimatePooling Namespace](#)

# PoolBehaviour Fields

The [PoolBehaviour](#) type exposes the following members.

## ▲ Fields

	Name	Description
 <b>S</b>	<a href="#">monoDespawnedEvent</a>	The name of the event that is called when an object is returned to the pool.
 <b>S</b>	<a href="#">monoSpawnedEvent</a>	The name of the event that is called when an object is spawned from the pool.

[Top](#)

## ▲ See Also

Reference

[PoolBehaviour Class](#)

[UltimatePooling Namespace](#)

# PoolBehaviourmonoDespawneEvent Field

The name of the event that is called when an object is returned to the pool.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▾ Syntax

**C#**    **JavaScript**

[Copy](#)

```
public static string monoDespawneEvent
```

Field Value

Type: [String](#)

## ▾ See Also

Reference

[PoolBehaviour Class](#)

[UltimatePooling Namespace](#)



# PoolBehaviourmonoSpawnedEvent Field

The name of the event that is called when an object is spawned from the pool.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▾ Syntax

**C#**    **JavaScript**

[Copy](#)

```
public static string monoSpawnedEvent
```

Field Value

Type: [String](#)

## ▾ See Also

Reference



[PoolBehaviour Class](#)

[UltimatePooling Namespace](#)

# PoolBehaviour Methods

The [PoolBehaviour](#) type exposes the following members.

## ▲ Methods

	Name	Description
	<a href="#">OnDespawned</a>	Called by the managing pool to notify that this object is about to be returned to the pool. This method will not be called when the object is about to be destroyed.
	<a href="#">OnSpawned</a>	Called by the managing pool to notify that this object has just been recycled from the pool. This method will not be called when the object is created for the first time.

[Top](#)

## ▲ See Also

Reference

[PoolBehaviour Class](#)

[UltimatePooling Namespace](#)

# PoolBehaviourOnDespawned Method

Called by the managing pool to notify that this object is about to be returned to the pool. This method will not be called when the object is about to be destroyed.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public virtual void OnDespawned(  
    PoolGroup pool  
)
```

## Parameters

*pool*

Type: [UltimatePoolingPoolGroup](#)

[Missing <param name="pool"/> documentation for "M:UltimatePooling.PoolBehaviour.OnDespawned(UltimatePooling.PoolGroup)"]

## Implements

[IPoolReceiverOnDespawned\(PoolGroup\)](#)

## ▲ See Also

### Reference

[PoolBehaviour Class](#)

[UltimatePooling Namespace](#)

# PoolBehaviourOnSpawned Method

Called by the managing pool to notify that this object has just been recycled from the pool. This method will not be called when the object is created for the first time.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public virtual void OnSpawned(  
    PoolGroup pool  
)
```

## Parameters

*pool*

Type: [UltimatePoolingPoolGroup](#)

[Missing <param name="pool"/> documentation for "M:UltimatePooling.PoolBehaviour.OnSpawned(UltimatePooling.PoolGroup)"]

## Implements

[IPoolReceiverOnSpawned\(PoolGroup\)](#)

## ▲ See Also

### Reference

[PoolBehaviour Class](#)

[UltimatePooling Namespace](#)

# PoolEventType Enumeration

The method that is used to inform pooled objects about their current spawn state.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public enum PoolEventType
```

## ▲ Members

Member name	Value	Description
<a href="#">BroadcastMessage</a>	0	Broadcast a message to the game object and all scripts with a matching listener method will be informed.
<a href="#">InterfaceCallback</a>	1	

## ▲ See Also

Reference

[UltimatePooling Namespace](#)

# PoolGroup Class

Represents a spawn pool for a specific type of prefab. Valid types are game objects and components.

## ▾ Inheritance Hierarchy

[SystemObject](#) **Object**  
**Component**  
**Behaviour**  
**MonoBehaviour**  
[UltimatePoolingPoolGroup](#)  
[UltimatePoolingGenericPoolGroup](#)

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▾ Syntax


C#    JavaScript

[Copy](#)

```
public abstract class PoolGroup : MonoBehaviour
```







The [PoolGroup](#) type exposes the following members.

## ▾ Constructors

	Name	Description
	<a href="#">PoolGroup</a>	Initializes a new instance of the <a href="#">PoolGroup</a> class

[Top](#)

## ▲ Methods

	Name	Description
	<code>despawn(Object)</code>	Indicates that the specified instance can be returned to the pool and re-used at a later time.
	<code>despawn(Object, Single)</code>	Indicates that the specified instance can be returned to the pool and re-used at a later time.
	<code>despawnAll</code>	Attempts to reclaim all instances spawned by this pool and return them to the pool. Any instances spawned by this pool will be forcefully returned without warning.
	<code>despawnAll(Single)</code>	Attempts to reclaim all instances spawned by this pool after the specified time delay. Any instances spawned by this pool will be forcefully returned without warning.
	<code>destroy</code>	Attempts to destroy a specific instance from the pool. Note that 'OnDespawn' will not be called on the instance. Instead you should handle any cleanup in 'OnDestroy'
	<code>destroyAll</code>	Attempts to destroy all pooled objects effectively emptying the pool and resetting its state.

Note that 'OnDespawn' will not be called on the pooled objects. Instead you should handle any cleanup in 'OnDestroy'



### `destroySelf`

Attempts to destroy all pooled objects effectively emptying the pool as well as destroying the pool instance. This is the preferred way of destroying an object pool as it allows the spawned objects to remain in the scene if required as opposed to being destroyed along with the pool. Note that 'OnDespawn' will not be called on any of the pooled objects. Instead you should handle any cleanup in 'OnDestroy'



### `didSpawn`

Returns true if this spawn group created the instance specified. Useful for spawn validation to make sure multiple pools are not attempting to manage the same instance.



### `onInstanceDespawned`





Should be implemented by the inheriting class. Called when the object is about to be returned to the pool.



### `onInstanceSpawned`






Should be implemented by the inheriting class. Called when the object has been taken from the pool and will be re-used.






	<code>spawn</code>	Spawn an instance from the pool.
	<code>spawn(Vector3, Quaternion)</code>	Spawn an instance from the pool using the specified position and rotation.
	<code>Start</code>	Called by Unity on the first frame.
	<code>ToString</code>	Override the string value to return detailed state information about the pool. (Overrides <b>Object.ToString</b> .)

[Top](#)




## ▲ Fields

	Name	Description
	<code>eventType</code>	The method used to inform a spawned instance when it is added to or removed from the pool.
	<code>maxAmount</code>	The max amount of instances that the pool can contain. If this amount is exceeded then the pool will need to destroy some objects.
	<code>parentInstances</code>	When true, all spawned instances will be added as child objects to the managing pool group.
	<code>pooled</code>	A collection of objects that are ready to be spawned.
	<code>prewarmAmount</code>	The amount of instances to preload.

	<a href="#">prewarmPerFrame</a>	The max amount of instances to preload per frame.
	<a href="#">prewarmPool</a>	Should the pool preload a set number of objects at startup. This can avoid frame spikes cause by calls to 'Instantiate' but may increase loading time.
	<a href="#">tracked</a>	A collection of objects that have been spawned by this pool.

[Top](#)

## ▲ Properties

	Name	Description
	<a href="#">IsFull</a>	Returns true if the pool is unable to store anymore pooled instances.
	<a href="#">IsPrewarming</a>	Returns true if the pool is currently prewarming.
	<a href="#">Prefab</a>	Should be implemented by the inheritng class. Should return the specific prefab type, For example 'GameObject'.

[Top](#)

## ▲ See Also

Reference

[UltimatePooling Namespace](#)

# PoolGroup Constructor

Initializes a new instance of the [PoolGroup](#) class

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
protected PoolGroup()
```

## ▲ See Also

Reference








[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolGroup Fields

The `PoolGroup` type exposes the following members.

## ▲ Fields

	Name	Description
	<code>eventType</code>	The method used to inform a spawned instance when it is added to or removed from the pool.
	<code>maxAmount</code>	The max amount of instances that the pool can contain. If this amount is exceeded then the pool will need to destroy some objects.
	<code>parentInstances</code>	When true, all spawned instances will be added as child objects to the managing pool group.
	<code>pooled</code>	A collection of objects that are ready to be spawned.
	<code>prewarmAmount</code>	The amount of instances to preload.
	<code>prewarmPerFrame</code>	The max amount of instances to preload per frame.
	<code>prewarmPool</code>	Should the pool preload a set number of objects at startup. This can avoid frame spikes cause by calls to 'Instantiate' but may increase loading time.



`tracked`

A collection of objects that have been spawned by this pool.

---

[Top](#)

## ▲ See Also

Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)

---

# PoolGroupeventType Field

The method used to inform a spawned instance when it is added to or removed from the pool.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

**C#**    **JavaScript**

[Copy](#)

```
public PoolEventType eventType
```

Field Value

Type: [PoolEventType](#)

## ▲ See Also

Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolGroupmaxAmount Field

The max amount of instances that the pool can contain. If this amount is exceeded then the pool will need to destroy some objects.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

**C#**    **JavaScript**

[Copy](#)

```
public int maxAmount
```

Field Value

Type: [Int32](#)

## ▲ See Also

Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolGroupparentInstances Field

When true, all spawned instances will be added as child objects to the managing pool group.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

**C#**    **JavaScript**

[Copy](#)

```
public bool parentInstances
```

Field Value

Type: [Boolean](#)

## ▲ See Also

Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)



# PoolGrouppooled Field

A collection of objects that are ready to be spawned.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
protected Stack<Object> pooled
```

Field Value

Type: [StackObject](#)

## ▲ See Also

Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolGroup.prewarmAmount Field

The amount of instances to preload.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public int prewarmAmount
```

Field Value

Type: [Int32](#)

## ▲ See Also

Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolGroupprewarmPerFrame Field

The max amount of instances to preload per frame.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version:  
0.0.0.0

## ▲ Syntax

**C#**    **JavaScript**

[Copy](#)

```
public int prewarmPerFrame
```

Field Value

Type: [Int32](#)

## ▲ See Also

Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolGroup.prewarmPool Field

Should the pool preload a set number of objects at startup. This can avoid frame spikes cause by calls to 'Instantiate' but may increase loading time.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public bool prewarmPool
```

Field Value

Type: [Boolean](#)

## ▲ See Also

Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolGrouptracked Field

A collection of objects that have been spawned by this pool.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
protected HashSet<Object> tracked
```

Field Value

Type: [HashSetObject](#)

## ▲ See Also

Reference




[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolGroup Methods

The [PoolGroup](#) type exposes the following members.

## ▲ Methods

	Name	Description
	<a href="#">despawn(Object)</a>	Indicates that the specified instance can be returned to the pool and re-used at a later time.
	<a href="#">despawn(Object, Single)</a>	Indicates that the specified instance can be returned to the pool and re-used at a later time.
	<a href="#">despawnAll</a>	Attempts to reclaim all instances spawned by this pool and return them to the pool. Any instances spawned by this pool will be forcefully returned without warning.
	<a href="#">despawnAll(Single)</a>	Attempts to reclaim all instances spawned by this pool after the specified time delay. Any instances spawned by this pool will be forcefully returned without warning.
	<a href="#">destroy</a>	Attempts to destroy a specific instance from the pool. Note that 'OnDespawn' will not be called on the instance. Instead

you should handle any cleanup in 'OnDestroy'



### destroyAll

Attempts to destroy all pooled objects effectively emptying the pool and resetting its state. Note that 'OnDespawn' will not be called on the pooled objects. Instead you should handle any cleanup in 'OnDestroy'



### destroySelf

Attempts to destroy all pooled objects effectively emptying the pool as well as destroying the pool instance. This is the preferred way of destroying an object pool as it allows the spawned objects to remain in the scene if required as opposed to being destroyed along with the pool. Note that 'OnDespawn' will not be called on any of the pooled objects. Instead you should handle any cleanup in 'OnDestroy'








### didSpawn

Returns true if this spawn group created the instance specified. Useful for spawn validation to make sure multiple pools are not attempting to manage the same instance.



### onInstanceDespawned

Should be implemented by the inheriting class. Called when the object is about to be returned to the pool.

	<a href="#">onInstanceSpawned</a>	Should be implemented by the inheriting class. Called when the object has been taken from the pool and will be re-used.
	<a href="#">spawn</a>	Spawn an instance from the pool.
	<a href="#">spawn(Vector3, Quaternion)</a>	Spawn an instance from the pool using the specified position and rotation.
	<a href="#">Start</a>	Called by Unity on the first frame.
	<a href="#">ToString</a>	Override the string value to return detailed state information about the pool. (Overrides <b>Object.ToString</b> .)

[Top](#)

## ▲ See Also

Reference



[PoolGroup Class](#)

[UltimatePooling Namespace](#)



# PoolGroupdespawn Method

## ▲ Overload List

	Name	Description
	<a href="#">despawn(Object)</a>	Indicates that the specified instance can be returned to the pool and re-used at a later time.
	<a href="#">despawn(Object, Single)</a>	Indicates that the specified instance can be returned to the pool and re-used at a later time.

[Top](#)

## ▲ See Also

Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolGroupdespawn Method (Object)

Indicates that the specified instance can be returned to the pool and re-used at a later time.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public void despawn(  
    Object instance  
)
```

## Parameters

*instance*

Type: **Object**

The instance to despawn

## ▲ See Also

### Reference

[PoolGroup Class](#)

[despawn Overload](#)

[UltimatePooling Namespace](#)

# PoolGroupdespawn Method (Object, Single)

Indicates that the specified instance can be returned to the pool and re-used at a later time.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public void despawn(  
    Object instance,  
    float time  
)
```

## Parameters

*instance*

Type: **Object**

The instance to despawn

*time*

Type: [SystemSingle](#)

The amount of time to wait before the object is despawned

## ▲ See Also

### Reference



[PoolGroup Class](#)

[despawn Overload](#)

[UltimatePooling Namespace](#)

# PoolGroupdespawnAll Method

## ▲ Overload List

	Name	Description
	<a href="#">despawnAll</a>	Attempts to reclaim all instances spawned by this pool and return them to the pool. Any instances spawned by this pool will be forcefully returned without warning.
	<a href="#">despawnAll(Single)</a>	Attempts to reclaim all instances spawned by this pool after the specified time delay. Any instances spawned by this pool will be forcefully returned without warning.

[Top](#)

## ▲ See Also

Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolGroupdespawnAll Method

Attempts to reclaim all instances spawned by this pool and return them to the pool. Any instances spawned by this pool will be forcefully returned without warning.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public void despawnAll()
```

## ▲ See Also

Reference

[PoolGroup Class](#)

[despawnAll Overload](#)

[UltimatePooling Namespace](#)

# PoolGroupdespawnAll Method (Single)

Attempts to reclaim all instances spawned by this pool after the specified time delay. Any instances spawned by this pool will be forcefully returned without warning.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public void despawnAll(  
    float time  
)
```

## Parameters

*time*

Type: [SystemSingle](#)

The amount of time to wait before despawning all instances

## ▲ See Also

### Reference

[PoolGroup Class](#)

[despawnAll Overload](#)

[UltimatePooling Namespace](#)

# PoolGroupdestroy Method

Attempts to destroy a specific instance from the pool. Note that 'OnDespawn' will not be called on the instance. Instead you should handle any cleanup in 'OnDestroy'

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public void destroy(  
    Object instance,  
    bool keepSpawnedInstances = true  
)
```

## Parameters

*instance*

Type: **Object**

The instance to remove from the pool

*keepSpawnedInstances* (**Optional**)

Type: [SystemBoolean](#)

If true, the pool will also try to locate this instance in its spawned list

## ▲ See Also

### Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolGroupdestroyAll Method

Attempts to destroy all pooled objects effectively emptying the pool and resetting its state. Note that 'OnDespawn' will not be called on the pooled objects. Instead you should handle any cleanup in 'OnDestroy'

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public void destroyAll(  
    bool keepSpawnedInstances = true  
)
```

## Parameters

*keepSpawnedInstances* (**Optional**)

Type: [SystemBoolean](#)

If true, all spawned objects created by this pool will also be destroyed

## ▲ See Also

### Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)



# PoolGroupdestroySelf Method

Attempts to destroy all pooled objects effectively emptying the pool as well as destroying the pool instance. This is the preferred way of destroying an object pool as it allows the spawned objects to remain in the scene if required as opposed to being destroyed along with the pool. Note that 'OnDespawn' will not be called on any of the pooled objects. Instead you should handle any cleanup in 'OnDestroy'

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public void destroySelf(  
    bool keepSpawnedInstances = true  
)
```

## Parameters

*keepSpawnedInstances* (**Optional**)

Type: [SystemBoolean](#)

When true, the pool will avoid destroying its parent object if the objects spawned by this pool are parented to it. This allows them to remain in the scene even though the pool will be destroyed

## ▲ See Also

### Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolGroupdidSpawn Method

Returns true if this spawn group created the instance specified. Useful for spawn validation to make sure multiple pools are not attempting to manage the same instance.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public bool didSpawn(  
    Object instance  
)
```

## Parameters

*instance*

Type: **Object**

The instance to check

## Return Value

Type: [Boolean](#)

True if this pool spawned the specified instance otherwise false

## ▲ See Also

### Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolGroup.onInstanceDespawned Method

Should be implemented by the inheriting class. Called when the object is about to be returned to the pool.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
protected abstract void onInstanceDespawned(  
    Object instance,  
    PoolEventType type  
)
```

## Parameters

*instance*

Type: **Object**

The object that is about to be pooled

*type*

Type: [UltimatePoolingPoolEventType](#)

The event type that should be used to inform the object that it is about to be despawned

## ▲ See Also

### Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolGrouponInstanceSpawned Method

Should be implemented by the inheriting class. Called when the object has been taken from the pool and will be re-used.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
protected abstract void onInstanceSpawned(  
    Object instance,  
    PoolEventType type,  
    Vector3 position,  
    Quaternion rotation  
)
```

## Parameters

*instance*

Type: **Object**

The object that has been re-used

*type*

Type: [UltimatePoolingPoolEventType](#)

The event type that should be used to inform the object that it has been spawned

*position*

Type: **Vector3**

The position to spawn the object at

*rotation*

Type: **Quaternion**

The rotation to spawn the object with

## ▲ See Also

Reference



[PoolGroup Class](#)

[UltimatePooling Namespace](#)

---

# PoolGroupspawn Method

## ▲ Overload List

	Name	Description
	<a href="#">spawn</a>	Spawn an instance from the pool.
	<a href="#">spawn(Vector3, Quaternion)</a>	Spawn an instance from the pool using the specified position and rotation.

[Top](#)

## ▲ See Also

Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolGroupspawn Method

Spawn an instance from the pool.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public Object spawn()
```

## Return Value

Type: **Object**

An instance of a pooled object

## ▲ See Also

### Reference

[PoolGroup Class](#)

[spawn Overload](#)

[UltimatePooling Namespace](#)

# PoolGroupspawn Method (Vector3, Quaternion)

Spawn an instance from the pool using the specified position and rotation.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public Object spawn(  
    Vector3 position,  
    Quaternion rotation  
)
```

## Parameters

*position*

Type: **Vector3**

The position in 3D space to place the spawned object

*rotation*

Type: **Quaternion**

The initial rotation to give the spawned object

## Return Value

Type: **Object**

An instance of a pooled object

## ▲ See Also

## Reference



PoolGroup Class  
spawn Overload  
UltimatePooling Namespace

---

# PoolGroupStart Method

Called by Unity on the first frame.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
protected virtual void Start()
```

## ▲ See Also

Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolGroupToString Method

Override the string value to return detailed state information about the pool.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

**C#**    **JavaScript**

[Copy](#)

```
public override string ToString()
```

## Return Value

Type: [String](#)

A string representation of the current pool state

## ▲ See Also

### Reference




[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolGroup Properties

The [PoolGroup](#) type exposes the following members.

## ▲ Properties

	Name	Description
	<a href="#">IsFull</a>	Returns true if the pool is unable to store anymore pooled instances.
	<a href="#">IsPrewarming</a>	Returns true if the pool is currently prewarming.
	<a href="#">Prefab</a>	Should be implemented by the inheritng class. Should return the specific prefab type, For example 'GameObject'.

[Top](#)

## ▲ See Also

Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolGroupsFull Property

Returns true if the pool is unable to store anymore pooled instances.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public bool IsFull { get; }
```

Property Value

Type: [Boolean](#)

## ▲ See Also

Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolGroupsPrewarming Property

Returns true if the pool is currently prewarming.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public bool IsPrewarming { get; }
```

Property Value

Type: [Boolean](#)

## ▲ See Also

Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolGroupPrefab Property

Should be implemented by the inheriting class. Should return the specific prefab type, For example 'GameObject'.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

**C#**    **JavaScript**

[Copy](#)

```
public abstract Object Prefab { get; set; }
```

Property Value

Type: **Object**

## ▲ See Also

Reference

[PoolGroup Class](#)

[UltimatePooling Namespace](#)

# PoolManager Class

The manager that is responsible for all pool groups and handles the creation and destruction of pools at runtime.

## ▾ Inheritance Hierarchy

[SystemObject](#) [UltimatePoolingPoolManager](#)

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▾ Syntax



C#    JavaScript

[Copy](#)

```
public sealed class PoolManager
```

The `PoolManager` type exposes the following members.

## ▾ Methods

	Name	Description
	<a href="#">createPool(String)</a>	Attempts to create a new object pool for a prefab located in the resources folder. If a pool already exists for the specified prefab name then this method will simply return the existing pool.
	<a href="#">createPool(Component, String)</a>	Attempts to create a new object pool for the component prefab type. If a



pool already exists for the specified prefab then this method will simply return the existing pool.



`createPool(GameObject, String)`

Attempt to create a new object pool for prefab type. If a pool already exists for the specified prefab then this method will simply return the existing pool.



`destroyPool`

Destroys a pool group and call of its pooled instances.



`findPool(String)`

Find the pool for the prefab with the specified name.



`findPool(Object)`

Find the pool for the specified prefab.



`findPoolWithInstance`

Find the pool that initially spawned the specified instance. This method will fail if the pool that spawned this instance has been destroyed. In this case it will be up to the user to destroy the object manually, or call `despawn(Object)` which will result in the same thing.



`hasPool`

Returns true if there is an existing pool for the specified prefab type.

[Top](#)

▲ **See Also**

Reference






[UltimatePooling Namespace](#)

---

# PoolManager Methods

The [PoolManager](#) type exposes the following members.

## ▲ Methods

	Name	Description
	<a href="#">createPool(String)</a>	Attempts to create a new object pool for a prefab located in the resources folder. If a pool already exists for the specified prefab name then this method will simply return the existing pool.
	<a href="#">createPool(Component, String)</a>	Attempts to create a new object pool for the component prefab type. If a pool already exists for the specified prefab then this method will simply return the existing pool.
	<a href="#">createPool(GameObject, String)</a>	Attempt to create a new object pool for prefab type. If a pool already exists for the specified prefab then this method will simply return the existing pool.
	<a href="#">destroyPool</a>	Destroys a pool group and call of its pooled instances.
	<a href="#">findPool(String)</a>	Find the pool for the prefab with the specified name.



[findPool\(Object\)](#)

Find the pool for the specified prefab.



[findPoolWithInstance](#)

Find the pool that initially spawned the specified instance. This method will fail if the pool that spawned this instance has been destroyed. In this case it will be up to the user to destroy the object manually, or call [despawn\(Object\)](#) which will result in the same thing.



[hasPool](#)

Returns true if there is an existing pool for the specified prefab type.

[Top](#)

## ▲ See Also




Reference

[PoolManager Class](#)

[UltimatePooling Namespace](#)

# PoolManagercreatePool Method

## ▲ Overload List

	Name	Description
	<a href="#">createPool(String)</a>	Attempts to create a new object pool for a prefab located in the resources folder. If a pool already exists for the specified prefab name then this method will simply return the existing pool.
	<a href="#">createPool(Component, String)</a>	Attempts to create a new object pool for the component prefab type. If a pool already exists for the specified prefab then this method will simply return the existing pool.
	<a href="#">createPool(GameObject, String)</a>	Attempt to create a new object pool for prefab type. If a pool already exists for the specified prefab then this method will simply return the existing pool.

[Top](#)

## ▲ See Also

Reference

[PoolManager Class](#)

## UltimatePooling Namespace

---

# PoolManagercreatePool Method (String)

Attempts to create a new object pool for a prefab located in the resources folder. If a pool already exists for the specified prefab name then this method will simply return the existing pool.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public PoolGroup createPool(  
    string prefabName  
)
```

## Parameters

*prefabName*

Type: [SystemString](#)

The name of the prefab in the resources folder

## Return Value

Type: [PoolGroup](#)

An instance of a pool group

## ▲ See Also

### Reference

[PoolManager Class](#)

[createPool Overload](#)

[UltimatePooling Namespace](#)





# PoolManagercreatePool Method (Component, String)

Attempts to create a new object pool for the component prefab type. If a pool already exists for the specified prefab then this method will simply return the existing pool.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public PoolGroup createPool(  
    Component prefab,  
    string name = ""  
)
```

## Parameters

*prefab*

Type: **Component**

The prefab to create the pool for

*name* (**Optional**)

Type: [SystemString](#)

The name of the pool

## Return Value

Type: [PoolGroup](#)

An instance of a pool group

## ▲ See Also

## Reference

[PoolManager Class](#)

[createPool Overload](#)

[UltimatePooling Namespace](#)

---

# PoolManagercreatePool Method (GameObject, String)

Attempt to create a new object pool for prefab type. If a pool already exists for the specified prefab then this method will simply return the existing pool.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public PoolGroup createPool(  
    GameObject prefab,  
    string name = ""  
)
```

## Parameters

*prefab*

Type: **GameObject**

The prefab to create the pool for

*name* (**Optional**)

Type: [SystemString](#)

The name of the pool

## Return Value

Type: [PoolGroup](#)

An instance of a pool group

## ▲ See Also

## Reference

[PoolManager Class](#)

[createPool Overload](#)

[UltimatePooling Namespace](#)

---

# PoolManagerdestroyPool Method

Destroys a pool group and call of its pooled instances.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public void destroyPool(  
    PoolGroup pool,  
    bool keepSpawnedInstances = true  
)
```

## Parameters

*pool*

Type: [UltimatePoolingPoolGroup](#)

The pool to destroy

*keepSpawnedInstances* **(Optional)**

Type: [SystemBoolean](#)

When true, all spawned instances will be kept alive

## ▲ See Also



### Reference

[PoolManager Class](#)

[UltimatePooling Namespace](#)

# PoolManagerfindPool Method

## ▲ Overload List

	Name	Description
	<a href="#">findPool(String)</a>	Find the pool for the prefab with the specified name.
	<a href="#">findPool(Object)</a>	Find the pool for the specified prefab.

[Top](#)

## ▲ See Also

Reference

[PoolManager Class](#)

[UltimatePooling Namespace](#)

# PoolManagerfindPool Method (String)

Find the pool for the prefab with the specified name.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public PoolGroup findPool(  
    string name  
)
```

## Parameters

*name*

Type: [SystemString](#)

The name of the prefab to find the pool for

## Return Value

Type: [PoolGroup](#)

An instance of the pool group responsible for the prefab with the specified name

## ▲ See Also

### Reference

[PoolManager Class](#)

[findPool Overload](#)

[UltimatePooling Namespace](#)

# PoolManagerfindPool Method (Object)

Find the pool for the specified prefab.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

**C#**    **JavaScript**

[Copy](#)

```
public PoolGroup findPool(  
    Object prefab  
)
```

## Parameters

*prefab*

Type: **Object**

The prefab to find the pool for

## Return Value

Type: [PoolGroup](#)

An insatnce of the pool group responsible for the specified prefab

## ▲ See Also

### Reference

[PoolManager Class](#)

[findPool Overload](#)

[UltimatePooling Namespace](#)



# PoolManagerfindPoolWithInstance Method

Find the pool that initially spawned the specified instance. This method will fail if the pool that spawned this instance has been destroyed. In this case it will be up to the user to destroy the object manually, or call [despawn\(Object\)](#) which will result in the same thing.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public PoolGroup findPoolWithInstance(  
    Object instance  
)
```

## Parameters

*instance*

Type: **Object**

The instance to find the managing pool for

## Return Value

Type: [PoolGroup](#)

The managin pool group if found

## ▲ See Also

### Reference

[PoolManager Class](#)

[UltimatePooling Namespace](#)



# PoolManagerhasPool Method

Returns true if there is an existing pool for the specified prefab type.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public bool hasPool(  
    Object prefab  
)
```

## Parameters

*prefab*

Type: **Object**

The prefab to check for

## Return Value

Type: [Boolean](#)

True if the specified prefab is already associated with a pool group

## ▲ See Also

### Reference

[PoolManager Class](#)

[UltimatePooling Namespace](#)

# ResourcesPoolGroup Class

Represents a pool group that manages a prefab object located within the resources folder.

## ▲ Inheritance Hierarchy

[SystemObject](#) **Object**

**Component**

**Behaviour**

**MonoBehaviour**

[UltimatePoolingPoolGroup](#)

[UltimatePoolingGenericPoolGroup](#)

[UltimatePoolingResourcesPoolGroup](#)

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public class ResourcesPoolGroup : GenericPoolGroup
```


The [ResourcesPoolGroup](#) type exposes the following members.

## ▲ Constructors

	Name	Description
	<a href="#">ResourcesPoolGroup</a>	Initializes a new instance of the <a href="#">ResourcesPoolGroup</a> class

[Top](#)


## ▲ Methods

	Name	Description
	<a href="#">Start</a>	Called by Unity when the pool is created. (Overrides <a href="#">PoolGroupStart</a> .)

---

[Top](#)


## ▲ Fields

	Name	Description
	<a href="#">prefabName</a>	The name of the prefab to load from the resources folder.

---

[Top](#)

## ▲ Properties

	Name	Description
	<a href="#">Prefab</a>	We need to modify the way that the prefab is retrieved. (Overrides <a href="#">GenericPoolGroupPrefab</a> .)

---

[Top](#)

## ▲ See Also

Reference

[UltimatePooling Namespace](#)

---

# ResourcesPoolGroup Constructor

Initializes a new instance of the [ResourcesPoolGroup](#) class

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public ResourcesPoolGroup()
```

## ▲ See Also

Reference


[ResourcesPoolGroup Class](#)

[UltimatePooling Namespace](#)

# ResourcesPoolGroup Fields

The [ResourcesPoolGroup](#) type exposes the following members.

## ▲ Fields

	Name	Description
	<a href="#">prefabName</a>	The name of the prefab to load from the resources folder.

---

[Top](#)

## ▲ See Also

Reference

[ResourcesPoolGroup Class](#)

[UltimatePooling Namespace](#)

# ResourcesPoolGroup prefabName Field

The name of the prefab to load from the resources folder.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public string prefabName
```

## Field Value

Type: [String](#)

## ▲ See Also

### Reference

[ResourcesPoolGroup Class](#)


[UltimatePooling Namespace](#)



# ResourcesPoolGroup Methods

The [ResourcesPoolGroup](#) type exposes the following members.

## ▲ Methods

	Name	Description
	<a href="#">Start</a>	Called by Unity when the pool is created. (Overrides <a href="#">PoolGroupStart</a> .)

[Top](#)

## ▲ See Also

Reference

[ResourcesPoolGroup Class](#)

[UltimatePooling Namespace](#)

# ResourcesPoolGroupStart Method

Called by Unity when the pool is created.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
protected override void Start()
```

## ▲ See Also

Reference


[ResourcesPoolGroup Class](#)

[UltimatePooling Namespace](#)

# ResourcesPoolGroup Properties

The [ResourcesPoolGroup](#) type exposes the following members.

## ▲ Properties

	Name	Description
	<a href="#">Prefab</a>	We need to modify the way that the prefab is retrieved. (Overrides <a href="#">GenericPoolGroupPrefab.</a> )

[Top](#)

## ▲ See Also

Reference

[ResourcesPoolGroup Class](#)

[UltimatePooling Namespace](#)

# ResourcesPoolGroupPrefab Property

We need to modify the way that the prefab is retrieved.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version:  
0.0.0.0

## ▸ Syntax

**C#**    **JavaScript**

[Copy](#)

```
public override Object Prefab { get; set; }
```

Property Value

Type: **Object**

## ▸ See Also

Reference

[ResourcesPoolGroup Class](#)

[UltimatePooling Namespace](#)

# UltimatePool Class

The main class for interacting with the UltimatePooling API. All spawning and despawning methods are found in this class however you can use the individual spawn method on pools if required.

## ▾ Inheritance Hierarchy

[SystemObject](#) [UltimatePoolingUltimatePool](#)

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▾ Syntax


C#    JavaScript

[Copy](#)

```
public static class UltimatePool
```

The [UltimatePool](#) type exposes the following members.

## ▾ Methods

Name	Description
 <a href="#">batchDespawn(IEnumerableComponent)</a>	Attempts to Despawn all objects and return them to their pool group. Important: All objects in the enumerable collection must have been spawned from the same pool group. If you attempt

to return instance:  
from multiple pool  
using this method  
then you will  
invalidate all  
associated pool  
groups.



### `batchDespawn(IEnumerableGameObject)`

Attempts to  
Despawn all object  
and return them to  
their pool group.  
Important: All object  
in the enumerable  
collection must have  
been spawned from  
the same pool  
group. If you attempt  
to return instance:  
from multiple pool  
using this method  
then you will  
invalidate all  
associated pool  
groups.



### `batchSpawn(String, Int32)`

Spawn a number  
instance of a prefab  
with the specified  
name from the  
appropriate pool.  
This method will  
succeed if the pool  
has been created  
before hand. Batch  
spawning is quicker  
than multiple calls  
`spawn(String)`  
because the pool

cached on the first spawn.



`batchSpawn(Component, Int32)`

Spawn a number of instances of the specified component prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type. Batch spawning is quicker than multiple calls to `spawn(Component)` because the pool is cached on the first spawn.



`batchSpawn(GameObject, Int32)`

Spawn a number of instances of the specified prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type. Batch spawning is quicker than multiple calls to `spawn(GameObject)` because the pool is cached on the first spawn.



`batchSpawn(String, Object, Int32)`

Spawn a number of instances of a prefab with the specified name from the

appropriate pool. This overload allows the user to manage the array where the objects will be spawned to avoid garbage generation. This method will succeed if the pool has been created beforehand. Batch spawning is quicker than multiple calls to `spawn(String)` because the pool is cached on the first spawn. If amount is specified then the value should be less than or equal to the length of the array, otherwise an out-of-bounds exception may occur.



### `batchSpawn(Component, Object, Int32)`

Spawn a number of instances of the specified prefab and place the results in the specified array. This overload allows the user to manage the array where the objects will be spawned to avoid garbage generation. If no pool exists, then one is automatically



created for the prefab type. Batch spawning is quick than multiple calls `spawn(Component)` because the pool is cached on the first spawn. If amount specified then the value should be less than or equal to the length of the array otherwise an out of bounds exception may occur.



`batchSpawn(GameObject, GameObject, Int32)`

Spawn a number instances of the specified prefab and place the results in the specified array. This overload allows the user to manage the array where the objects will be spawned to avoid garbage generation. If no pool exists, then one is automatically created for the prefab type. Batch spawning is quick than multiple calls `spawn(GameObject)` because the pool is cached on the first spawn. If amount specified then the

value should be less than or equal to the length of the array otherwise an out of bounds exception may occur.



### `batchSpawnT(Component, Int32)`

Spawn a number of instances of the specified component prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type. Batch spawning is quicker than multiple calls to `spawnT(Component)` because the pool is cached on the first spawn.



### `batchSpawnT(Component, T, Int32)`

Spawn a number of instances of the specified prefab and place the results in the specified array. This overload allows the user to manage the array where the objects will be spawned to avoid garbage generation. If no pool exists, then one is automatically created for the prefab type. Batch

spawning is quick that multiple calls `spawnT(Component)` because the pool cached on the first spawn. If amount specified then the value should be less than or equal to the length of the array otherwise an out of bounds exception may occur.



`despawn(Object)`

Direct replacement for 'Object.Destroy' for pooling. Allows the specified instance to be returned to the pool and re-used at a later time.



`despawn(Object, Single)`

Direct overload replacement for 'Object.Destroy' for pooling. Allows the specified instance be returned to the pool and re-used at a later time.



`despawnAll(GameObject)`

Calls all spawned instances of the specified prefab back to their pool. This method allow you to pass a prefab such as a 'Bullet'

which will subsequently cause all 'Bullet' instances to be despawned.



`despawnAll(GameObject, Single)`

Calls all spawned instances of the specified prefab back to their pool. This method allows you to pass a prefab such as a 'Bullet' which will subsequently cause all 'Bullet' instances to be despawned.







`spawn(String)`

Spawn an instance of a prefab with the specified name. The prefab name is the same name used when creating the pool, or if no name is used then the name of the prefab supplied is substituted. This method will only succeed if the pool has been created before hand.



`spawn(Component)`

Spawn an instance of the specified component prefab from the appropriate pool. If no pool exists, then one is

		automatically created for the prefab type.
	<code>spawn(GameObject)</code>	Spawn an instance of the specified prefab from the appropriate pool. If no pool exists, one is automatically created for the prefab type.
	<code>spawn(String, Vector3, Quaternion)</code>	Spawn an instance of a prefab with the specified name. This method will only succeed if the pool has been created beforehand.
	<code>spawn(Component, Vector3, Quaternion)</code>	Spawn an instance of the specified component prefab from the appropriate pool. If no pool exists, then one is automatically created for the component prefab type.
	<code>spawn(GameObject, Vector3, Quaternion)</code>	Spawn an instance of the specified prefab from the appropriate pool. If no pool exists, one is automatically created for the

prefab type.



`spawnT(Component)`

Spawn an instance of the specified component prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type.



`spawnT(Component, Vector3, Quaternion)`

Spawn an instance of the specified component prefab from the appropriate pool. If no pools exist, then one is automatically created for the component prefab type.

[Top](#)

## ▲ Fields

	Name	Description
	<code>LogLevel</code>	The amount level of logging that is allowed. Default is 'Message' - Full logging.

[Top](#)

## ▲ Properties

	Name	Description
	<code>Pools</code>	Access the pool manager which maintains all

existing pools. Allows pools to be created and destroyed.

---

[Top](#)

## ▲ See Also

Reference


[UltimatePooling Namespace](#)

---

# UltimatePool Fields

The [UltimatePool](#) type exposes the following members.

## ▲ Fields

	Name	Description
	<a href="#">logLevel</a>	The amount level of logging that is allowed. Default is 'Message' - Full logging.

[Top](#)

## ▲ See Also

Reference

[UltimatePool Class](#)

[UltimatePooling Namespace](#)



# UltimatePoolLogLevel Field

The amount level of logging that is allowed. Default is 'Message' - Full logging.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

**C#**    **JavaScript**

[Copy](#)

```
public static LogLevel logLevel
```

Field Value

Type: [LogLevel](#)

## ▲ See Also





Reference

[UltimatePool Class](#)

[UltimatePooling Namespace](#)

# UltimatePool Methods

## ▲ Methods

	Name	Description
 	<code>batchDespawn(IEnumerableComponent)</code>	Attempts to Despawn all object and return them to their pool group. Important: All object in the enumerable collection must have been spawned from the same pool group. If you attempt to return instances from multiple pool using this method then you will invalidate all associated pool groups.
 	<code>batchDespawn(IEnumerableGameObject)</code>	Attempts to Despawn all object and return them to their pool group. Important: All object in the enumerable collection must have been spawned from the same pool group. If you attempt to return instances

from multiple pool using this method then you will invalidate all associated pool groups.



### `batchSpawn(String, Int32)`

Spawn a number instance of a prefab with the specified name from the appropriate pool. This method will succeed if the pool has been created before hand. Batch spawning is quick than multiple calls `spawn(String)` because the pool is cached on the first spawn.



### `batchSpawn(Component, Int32)`

Spawn a number instances of the specified component prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type. Batch spawning is quick than multiple calls `spawn(Component)` because the pool is cached on the first spawn.

---



## batchSpawn(GameObject, Int32)

Spawn a number instances of the specified prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type. Batch spawning is quicker than multiple calls to [spawn\(GameObject\)](#) because the pool is cached on the first spawn.



## batchSpawn(String, Object, Int32)

Spawn a number instances of a prefab with the specified name from the appropriate pool. This overload allows the user to manage the array where the objects will be spawned to avoid garbage generation. This method will succeed if the pool has been created beforehand. Batch spawning is quicker than multiple calls to [spawn\(String\)](#) because the pool is cached on the first spawn. If amount specified then the value should be less

than or equal to the length of the array otherwise an out of bounds exception may occur.



`batchSpawn(Component, Object, Int32)`

Spawn a number instances of the specified prefab at the place the results in the specified array. This overload allows the user to manage the array where the objects will be spawned to avoid garbage generation. If no pool exists, then one is automatically created for the prefab type. Batch spawning is quicker than multiple calls to `spawn(Component)` because the pool is cached on the first spawn. If amount specified then the value should be less than or equal to the length of the array otherwise an out of bounds exception may occur.



`batchSpawn(GameObject, GameObject, Int32)`

Spawn a number instances of the specified prefab at

place the results in the specified array. This overload allows the user to manage the array where the objects will be spawned to avoid garbage generation. If no pool exists, then one is automatically created for the prefab type. Batch spawning is quicker than multiple calls to `spawn(GameObject)` because the pool is cached on the first spawn. If amount is specified then the value should be less than or equal to the length of the array, otherwise an out of bounds exception may occur.



### `batchSpawnT(Component, Int32)`

Spawn a number of instances of the specified component prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type. Batch spawning is quicker than multiple calls to `spawnT(Component)`.

because the pool  
cached on the first  
spawn.



### `batchSpawnT(Component, T, Int32)`

Spawn a number  
instance of the  
specified prefab at  
place the results in  
the specified array.  
This overload allows  
the user to manage  
the array where the  
objects will be  
spawned to avoid  
garbage generation.  
If no pool exists,  
then one is  
automatically  
created for the  
prefab type. Batch  
spawning is quicker  
than multiple calls  
to `spawnT(Component)`  
because the pool  
is cached on the first  
spawn. If amount  
specified then the  
value should be less  
than or equal to the  
length of the array,  
otherwise an out of  
bounds exception  
may occur.



### `despawn(Object)`

Direct replacement  
for `Object.Destroy`  
for pooling. Allows  
the specified  
instance to be

returned to the pool and re-used at a later time.



`despawn(Object, Single)`

Direct overload replacement for 'Object.Destroy' for pooling. Allows the specified instance be returned to the pool and re-used at a later time.



`despawnAll(GameObject)`

Calls all spawned instances of the specified prefab back to their pool. This method allow you to pass a prefab such as a 'Bullet' which will subsequently cause all 'Bullet' instances to be despawned.



`despawnAll(GameObject, Single)`

Calls all spawned instances of the specified prefab back to their pool. This method allow you to pass a prefab such as a 'Bullet' which will subsequently cause all 'Bullet' instances to be despawned.



`spawn(String)`

Spawn an instance of a prefab with the



specified name. The prefab name is the same name used when creating the pool, or if no name is used then the name of the prefab supplied is substituted. This method will only succeed if the pool has been created before hand.



`spawn(Component)`

Spawn an instance of the specified component prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type.



`spawn(GameObject)`

Spawn an instance of the specified prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type.



`spawn(String, Vector3, Quaternion)`

Spawn an instance of a prefab with the specified name. This method will only succeed if the pool has been created

before hand.



`spawn(Component, Vector3, Quaternion)`

Spawn an instance of the specified component prefab from the appropriate pool. If no pool exists, then one is automatically created for the component prefab type.



`spawn(GameObject, Vector3, Quaternion)`

Spawn an instance of the specified prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type.



`spawnT(Component)`

Spawn an instance of the specified component prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type.



`spawnT(Component, Vector3, Quaternion)`

Spawn an instance of the specified component prefab from the appropriate pool. If no pools exists, then one is automatically

created for the component prefat type.

---

[Top](#)

## ▲ See Also

Reference


[UltimatePool Class](#)

[UltimatePooling Namespace](#)

---

# UltimatePoolbatchDespawn Method

## ▲ Overload List

	Name	Description
	<code>batchDespawn(IEnumerableComponent)</code>	Attempts to Despawn all objects and return them to their pool group. Important: All objects in the enumerable collection must have been spawned from the same pool group. If you attempt to return instances from multiple pools using this method then you will invalidate all associated

pool groups.



[batchDespawn\(IEnumerableGameObject\)](#)

Attempts to Despawn all objects and return them to their pool group.

Important:  
All objects in the enumerable collection must have been spawned from the same pool group. If you attempt to return instances from multiple pools using this method then you will invalidate all associated pool groups.

---

[Top](#)

## ▲ See Also

Reference

[UltimatePool Class](#)

[UltimatePooling Namespace](#)

---



# UltimatePoolbatchDespawn Method (IEnumerableComponent)

Attempts to Despawn all objects and return them to their pool group. Important: All objects in the enumerable collection must have been spawned from the same pool group. If you attempt to return instances from multiple pools using this method then you will invalidate all associated pool groups.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static void batchDespawn(  
    IEnumerable<Component> objects  
)
```

## Parameters

*objects*

Type: [System.Collections.GenericIEnumerableComponent](#)

An enumerable collection of components that should be despawned

## ▲ See Also

### Reference

[UltimatePool Class](#)

[batchDespawn Overload](#)

[UltimatePooling Namespace](#)

# UltimatePoolbatchDespawn Method (IEnumerableGameObject)

Attempts to Despawn all objects and return them to their pool group. Important: All objects in the enumerable collection must have been spawned from the same pool group. If you attempt to return instances from multiple pools using this method then you will invalidate all associated pool groups.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static void batchDespawn(  
    IEnumerable<GameObject> objects  
)
```

## Parameters

*objects*

Type: [System.Collections.GenericIEnumerableGameObject](#)

An enumerable collection of objects that should be despawned

## ▲ See Also

### Reference

[UltimatePool Class](#)

[batchDespawn Overload](#)







[UltimatePooling Namespace](#)





# UltimatePoolbatchSpawn Method

## ▲ Overload List

	Name	Description
 	<a href="#">batchSpawn(String, Int32)</a>	Spawn a number of instance of a prefab with the specified name from the appropriate pool. This method will only succeed if the pool has been created before hand. Batch spawning is quicker than multiple calls to <a href="#">spawn(String)</a> because the pool is cached on the first spawn.
 	<a href="#">batchSpawn(Component, Int32)</a>	Spawn a number of instances of the specified component prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type. Batch spawning is quicker than multiple calls to <a href="#">spawn(Component)</a> because the pool is cached on the first spawn.
 	<a href="#">batchSpawnT(Component, Int32)</a>	Spawn a number of instances of the specified component prefab from the

appropriate pool. If no pool exists, then one is automatically created for the prefab type. Batch spawning is quicker than multiple calls to [spawnT\(Component\)](#) because the pool is cached on the first spawn.



[batchSpawn\(GameObject, Int32\)](#)

Spawn a number of instances of the specified prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type. Batch spawning is quicker than multiple calls to [spawn\(GameObject\)](#) because the pool is cached on the first spawn.



[batchSpawn\(String, Object, Int32\)](#)

Spawn a number of instances of a prefab with the specified name from the appropriate pool. This overload allows the user to manage the array where the objects will be spawned to avoid garbage generation. This method will only succeed if the pool has been created before hand. Batch spawning is quicker than multiple calls to [spawn\(String\)](#) because the pool is cached on the first spawn. If amount is specified then the value

should be less than or equal to the length of the array otherwise an out of bounds exception may occur.



[batchSpawnT\(Component, T, Int32\)](#)

Spawn a number of instance of the specified prefab and place the results in the specified array. This overload allows the user to manage the array where the objects will be spawned to avoid garbage generation. If no pool exists, then one is automatically created for the prefab type. Batch spawning is quicker than multiple calls to [spawnT\(Component\)](#) because the pool is cached on the first spawn. If amount is specified then the value should be less than or equal to the length of the array otherwise an out of bounds exception may occur.



[batchSpawn\(Component, Object, Int32\)](#)

Spawn a number of instances of the specified prefab and place the results in the specified array. This overload allows the user to manager the array where the objects will be spawned to avoid garbage generation. If no

pool exists, then one is automatically created for the prefab type. Batch spawning is quicker than multiple calls to `spawn(Component)` because the pool is cached on the first spawn. If amount is specified then the value should be less than or equal to the length of the array otherwise an out of bounds exception may occur.



`batchSpawn(GameObject, GameObject, Int32)`

Spawn a number of instances of the specified prefab and place the results in the specified array. This overload allows the user to manage the array where the objects will be spawned to avoid garbage generation. If no pool exists, then one is automatically created for the prefab type. Batch spawning is quicker than multiple calls to `spawn(GameObject)` because the pool is cached on the first spawn. If amount is specified then the value should be less than or equal to the length of the array otherwise an out of bounds exception may occur.

---

[Top](#)

## ▲ See Also

Reference

[UltimatePool Class](#)

[UltimatePooling Namespace](#)

---

# UltimatePoolbatchSpawn Method (String, Int32)

Spawn a number of instance of a prefab with the specified name from the appropriate pool. This method will only succeed if the pool has been created before hand. Batch spawning is quicker than multiple calls to [spawn\(String\)](#) because the pool is cached on the first spawn.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static IEnumerable<Object> batchSpawn(  
    string prefabName,  
    int amount  
)
```

## Parameters

*prefabName*

Type: [SystemString](#)

The name of the prefab to spawn from

*amount*

Type: [SystemInt32](#)

The amount of instances to create from this prefab

## Return Value

Type: [IEnumerableObject](#)

An enumeration of the spawned instances

## ▲ See Also

## Reference

[UltimatePool Class](#)

[batchSpawn Overload](#)

[UltimatePooling Namespace](#)

---



# UltimatePoolbatchSpawn Method (Component, Int32)

Spawn a number of instances of the specified component prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type. Batch spawning is quicker than multiple calls to [spawn\(Component\)](#) because the pool is cached on the first spawn.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static IEnumerable<Object> batchSpawn(  
    Component prefab,  
    int amount  
)
```

## Parameters

*prefab*

Type: **Component**

The component prefab to spawn from

*amount*

Type: [SystemInt32](#)

The amount of instances to create from this prefab

## Return Value

Type: [IEnumerableObject](#)

An enumeration of the spawned instances

## ▲ See Also

### Reference

[UltimatePool Class](#)

[batchSpawn Overload](#)

[UltimatePooling Namespace](#)

---

# UltimatePoolbatchSpawnT Method (Component, Int32)

Spawn a number of instances of the specified component prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type. Batch spawning is quicker than multiple calls to [spawnT\(Component\)](#) because the pool is cached on the first spawn.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static IEnumerable<T> batchSpawn<T>(
    Component prefab,
    int amount
)
where T : Object
```

## Parameters

*prefab*

Type: **Component**

The component prefab to spawn from

*amount*

Type: [SystemInt32](#)

The amount of instances to create from this prefab

## Type Parameters

*T*

The type of object to return the instances as

## Return Value

Type: [IEnumerable<T>](#)

An enumeration of the spawned instances

## ▲ See Also

### Reference

[UltimatePool Class](#)

[batchSpawn Overload](#)

[UltimatePooling Namespace](#)

---

# UltimatePoolbatchSpawn Method (GameObject, Int32)

Spawn a number of instances of the specified prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type. Batch spawning is quicker than multiple calls to [spawn\(GameObject\)](#) because the pool is cached on the first spawn.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static IEnumerable<GameObject> batchSpawn(  
    GameObject prefab,  
    int amount  
)
```

## Parameters

*prefab*

Type: **GameObject**

The prefab to spawn from

*amount*

Type: [SystemInt32](#)

The amount of instances to create from this prefab

## Return Value

Type: [IEnumerableGameObject](#)

An enumeration of the spawned instances

## ▲ See Also

## Reference

[UltimatePool Class](#)

[batchSpawn Overload](#)

[UltimatePooling Namespace](#)

---

# UltimatePoolbatchSpawn Method (String, Object, Int32)

Spawn a number of instances of a prefab with the specified name from the appropriate pool. This overload allows the user to manage the array where the objects will be spawned to avoid garbage generation. This method will only succeed if the pool has been created before hand. Batch spawning is quicker than multiple calls to [spawn\(String\)](#) because the pool is cached on the first spawn. If amount is specified then the value should be less than or equal to the length of the array otherwise an out of bounds exception may occur.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static void batchSpawn(  
    string prefabName,  
    Object[] objects,  
    int amount = -1  
)
```

## Parameters

*prefabName*

Type: [SystemString](#)

The name of the prefab to spawn from

*objects*

Type: **Object**

The array to store the spawned objects in

*amount* (**Optional**)

Type: [SystemInt32](#)

The amount of objects to spawn. If the value is set to -1 then the array is filled

## ▲ See Also

### Reference

[UltimatePool Class](#)

[batchSpawn Overload](#)

[UltimatePooling Namespace](#)

---



# UltimatePoolbatchSpawnT Method (Component, T, Int32)

Spawn a number of instance of the specified prefab and place the results in the specified array. This overload allows the user to manage the array where the objects will be spawned to avoid garbage generation. If no pool exists, then one is automatically created for the prefab type. Batch spawning is quicker than multiple calls to [spawnT\(Component\)](#) because the pool is cached on the first spawn. If amount is specified then the value should be less than or equal to the length of the array otherwise an out of bounds exception may occur.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static void batchSpawn<T>(
    Component prefab,
    T[] objects,
    int amount = -1
)
where T : Object
```

## Parameters

*prefab*

Type: **Component**

The prefab to spawn from

*objects*

Type: *T*

The array to store the spawned objects in

### *amount* (Optional)

Type: [SystemInt32](#)

The amount of objects to spawn. If the value is set to -1 then the array is filled

### Type Parameters

*T*

The type of prefab that will be spawned

## ▲ See Also

### Reference

[UltimatePool Class](#)

[batchSpawn Overload](#)

[UltimatePooling Namespace](#)

---

# UltimatePoolbatchSpawn Method (Component, Object, Int32)

Spawn a number of instances of the specified prefab and place the results in the specified array. This overload allows the user to manager the array where the objects will be spawned to avoid garbage generation. If no pool exists, then one is automatically created for the prefab type. Batch spawning is quicker than multiple calls to [spawn\(Component\)](#) because the pool is cached on the first spawn. If amount is specified then the value should be less than or equal to the length of the array otherwise an out of bounds exception may occur.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static void batchSpawn(  
    Component prefab,  
    Object[] objects,  
    int amount = -1  
)
```

## Parameters

*prefab*

Type: **Component**

The prefab to spawn from

*objects*

Type: **Object**

The array to store the spawned objects in

*amount* (**Optional**)

Type: [SystemInt32](#)

The amount of objects to spawn. If the value is set to -1 then the array is filled

## ▲ See Also

### Reference

[UltimatePool Class](#)

[batchSpawn Overload](#)

[UltimatePooling Namespace](#)

---

# UltimatePoolbatchSpawn Method (GameObject, GameObject, Int32)

Spawn a number of instances of the specified prefab and place the results in the specified array. This overload allows the user to manage the array where the objects will be spawned to avoid garbage generation. If no pool exists, then one is automatically created for the prefab type. Batch spawning is quicker than multiple calls to [spawn\(GameObject\)](#) because the pool is cached on the first spawn. If amount is specified then the value should be less than or equal to the length of the array otherwise an out of bounds exception may occur.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static void batchSpawn(  
    GameObject prefab,  
    GameObject[] objects,  
    int amount = -1  
)
```

## Parameters

*prefab*

Type: **GameObject**

The prefab to spawn from

*objects*

Type: **GameObject**

The array to store the spawned objects in

*amount* (**Optional**)

Type: [SystemInt32](#)

The amount of objects to spawn. If the value is set to -1 then the array is filled

## ▲ See Also

### Reference

[UltimatePool Class](#)


[batchSpawn Overload](#)

[UltimatePooling Namespace](#)

---

# UltimatePooldespawn Method

## ▲ Overload List

	Name	Description
	<a href="#">despawn(Object)</a>	Direct replacement for 'Object.Destroy' for pooling. Allows the specified instance to be returned to the pool and re-used at a later time.
	<a href="#">despawn(Object, Single)</a>	Direct overload replacement for 'Object.Destroy' for pooling. Allows the specified instance to be returned to the pool and re-used at a later time.

[Top](#)

## ▲ See Also

Reference

[UltimatePool Class](#)

[UltimatePooling Namespace](#)

# UltimatePoolDespawn Method (Object)

Direct replacement for 'Object.Destroy' for pooling. Allows the specified instance to be returned to the pool and re-used at a later time.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static void despawn(  
    Object instance  
)
```

## Parameters

*instance*

Type: **Object**

A reference to a spawned instance

## ▲ See Also

### Reference

[UltimatePool Class](#)

[despawn Overload](#)

[UltimatePooling Namespace](#)



# UltimatePool.Despawn Method (Object, Single)

Direct overload replacement for 'Object.Destroy' for pooling. Allows the specified instance to be returned to the pool and re-used at a later time.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static void despawn(  
    Object instance,  
    float time  
)
```

## Parameters

*instance*

Type: **Object**

A reference to a spawned instance

*time*

Type: [System.Single](#)

The amount of time to wait before despawning the instance

## ▲ See Also

### Reference

[UltimatePool Class](#)



[despawn Overload](#)

[UltimatePooling Namespace](#)



# UltimatePoolDespawnAll Method

## ▲ Overload List

	Name	Description
	<a href="#">despawnAll(GameObject)</a>	Calls all spawned instances of the specified prefab back to their pool. This method allows you to pass a prefab such as a 'Bullet' which will subsequently cause all 'Bullet' instances to be despawned.
	<a href="#">despawnAll(GameObject, Single)</a>	Calls all spawned instances of the specified prefab back to their pool. This method allows you to pass a prefab such as a 'Bullet' which will subsequently cause all 'Bullet' instances to be despawned.

[Top](#)

## ▲ See Also

Reference

[UltimatePool Class](#)

[UltimatePooling Namespace](#)

# UltimatePool.despawnAll Method (GameObject)

Calls all spawned instances of the specified prefab back to their pool. This method allows you to pass a prefab such as a 'Bullet' which will subsequently cause all 'Bullet' instances to be despawned.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static void despawnAll(  
    GameObject prefab  
)
```

## Parameters

*prefab*

Type: **GameObject**

The prefab to despawn all instances of

## ▲ See Also

### Reference

[UltimatePool Class](#)

[despawnAll Overload](#)

[UltimatePooling Namespace](#)

# UltimatePoolDespawnAll Method (GameObject, Single)

Calls all spawned instances of the specified prefab back to their pool. This method allows you to pass a prefab such as a 'Bullet' which will subsequently cause all 'Bullet' instances to be despawned.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static void despawnAll(  
    GameObject prefab,  
    float time  
)
```

## Parameters

*prefab*

Type: **GameObject**

The prefab to despawn all instances of

*time*

Type: [SystemSingle](#)

The amount of time to wait before despawning

## ▲ See Also

### Reference

[UltimatePool Class](#)











[despawnAll Overload](#)

[UltimatePooling Namespace](#)



# UltimatePoolspawn Method

## ▲ Overload List

	Name	Description
 	<code>spawn(String)</code>	Spawn an instance of a prefab with the specified name. The prefab name is the same name used when creating the pool, or if no name is used then the name of the prefab supplied is substituted. This method will only succeed if the pool has been created before hand.
 	<code>spawn(Component)</code>	Spawn an instance of the specified component prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type.
 	<code>spawnT(Component)</code>	Spawn an instance of the specified component prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type.
 	<code>spawn(GameObject)</code>	Spawn an instance of the specified prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type.
 	<code>spawn(String, Vector3, Quaternion)</code>	Spawn an instance of a prefab with the specified name. This

method will only succeed if the pool has been created before hand.



`spawn(Component, Vector3, Quaternion)`

Spawn an instance of the specified component prefab from the appropriate pool. If no pool exists, then one is automatically created for the component prefab type.



`spawnT(Component, Vector3, Quaternion)`

Spawn an instance of the specified component prefab from the appropriate pool. If no pools exists, then one is automatically created for the component prefab type.



`spawn(GameObject, Vector3, Quaternion)`

Spawn an instance of the specified prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type.

[Top](#)

## ▲ See Also

Reference

[UltimatePool Class](#)

[UltimatePooling Namespace](#)



# UltimatePoolspawn Method (String)

Spawn an instance of a prefab with the specified name. The prefab name is the same name used when creating the pool, or if no name is used then the name of the prefab supplied is substituted. This method will only succeed if the pool has been created before hand.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#

JavaScript

[Copy](#)

```
public static GameObject spawn(  
    string prefabName  
)
```

## Parameters

*prefabName*

Type: [SystemString](#)

The name of the prefab to spawn from

## Return Value

Type: **GameObject**

An instance of the prefab with the specified name

## ▲ See Also

### Reference

[UltimatePool Class](#)

[spawn Overload](#)

[UltimatePooling Namespace](#)



# UltimatePoolspawn Method (Component)

Spawn an instance of the specified component prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static Object spawn(  
    Component prefab  
)
```

## Parameters

*prefab*

Type: **Component**

The component prefab to spawn from

## Return Value

Type: **Object**

An instance of the prefab supplied

## ▲ See Also

### Reference

[UltimatePool Class](#)

[spawn Overload](#)

[UltimatePooling Namespace](#)



# UltimatePoolspawnT Method (Component)

Spawn an instance of the specified component prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static T spawn<T>(
    Component prefab
)
where T : Object
```

## Parameters

*prefab*

Type: **Component**

The component prefab to spawn from

## Type Parameters

*T*

The type of object to return the instance as

## Return Value

Type: *T*

An instance of the prefab supplied

## ▲ See Also

## Reference

[UltimatePool Class](#)

[spawn Overload](#)

[UltimatePooling Namespace](#)

---

# UltimatePoolspawn Method (GameObject)

Spawn an instance of the specified prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static GameObject spawn(  
    GameObject prefab  
)
```

## Parameters

*prefab*

Type: **GameObject**

The prefab to spawn from

## Return Value

Type: **GameObject**

An instance of the prefab supplied

## ▲ See Also

### Reference

[UltimatePool Class](#)

[spawn Overload](#)

[UltimatePooling Namespace](#)

# UltimatePoolspawn Method (String, Vector3, Quaternion)

Spawn an instance of a prefab with the specified name. This method will only succeed if the pool has been created before hand.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static GameObject spawn(  
    string prefabName,  
    Vector3 position,  
    Quaternion identity  
)
```

## Parameters

*prefabName*

Type: [SystemString](#)

The name of the prefab to spawn from

*position*

Type: **Vector3**

The position to spawn the prefab at

*identity*

Type: **Quaternion**

The initial rotation to spawn the prefab with

## Return Value

Type: **GameObject**

An instance of the prefab with the specified name



## ▲ See Also

### Reference

[UltimatePool Class](#)

[spawn Overload](#)

[UltimatePooling Namespace](#)

---

# UltimatePoolspawn Method (Component, Vector3, Quaternion)

Spawn an instance of the specified component prefab from the appropriate pool. If no pool exists, then one is automatically created for the component prefab type.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static Object spawn(  
    Component prefab,  
    Vector3 position,  
    Quaternion rotation  
)
```

## Parameters

*prefab*

Type: **Component**

The component prefab to spawn from

*position*

Type: **Vector3**

The position to spawn the prefab at

*rotation*

Type: **Quaternion**

The initial rotation to spawn the prefab with

## Return Value

Type: **Object**

An instance of the component prefab supplied

## ▲ See Also

### Reference

[UltimatePool Class](#)

[spawn Overload](#)

[UltimatePooling Namespace](#)

---

# UltimatePoolspawnT Method (Component, Vector3, Quaternion)

Spawn an instance of the specified component prefab from the appropriate pool. If no pools exists, then one is automatically created for the component prefab type.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static T spawn<T>(
    Component prefab,
    Vector3 position,
    Quaternion rotation
)
where T : Object
```

## Parameters

*prefab*

Type: **Component**

The component prefab to spawn from

*position*

Type: **Vector3**

The position to spawn the prefab at

*rotation*

Type: **Quaternion**

The initial rotation to spawn the prefab with

## Type Parameters

*T*

The type of object to return the instance as

## Return Value

Type: *T*

An instance of the component prefab supplied

## ▲ See Also

### Reference

[UltimatePool Class](#)

[spawn Overload](#)

[UltimatePooling Namespace](#)

---

# UltimatePoolspawn Method (GameObject, Vector3, Quaternion)

Spawn an instance of the specified prefab from the appropriate pool. If no pool exists, then one is automatically created for the prefab type.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public static GameObject spawn(  
    GameObject prefab,  
    Vector3 position,  
    Quaternion rotation  
)
```

## Parameters

*prefab*

Type: **GameObject**

The prefab to spawn from

*position*

Type: **Vector3**

The position to spawn the prefab at

*rotation*

Type: **Quaternion**

The initial rotation to spawn the prefab with

## Return Value

Type: **GameObject**

An instance of the prefab supplied

## ▲ See Also

### Reference

[UltimatePool Class](#)

[spawn Overload](#)


[UltimatePooling Namespace](#)

---

# UltimatePool Properties

The [UltimatePool](#) type exposes the following members.

## ▲ Properties

	Name	Description
 <b>S</b>	<a href="#">Pools</a>	Access the pool manager which maintains all existing pools. Allows pools to be created and destroyed.

---

[Top](#)

## ▲ See Also

Reference

[UltimatePool Class](#)

[UltimatePooling Namespace](#)

---



# UltimatePoolPools Property

Access the pool manager which maintains all existing pools. Allows pools to be created and destroyed.

**Namespace:** [UltimatePooling](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

**C#**    **JavaScript**

[Copy](#)

```
public static PoolManager Pools { get; }
```

Property Value

Type: [PoolManager](#)

## ▲ See Also






Reference

[UltimatePool Class](#)

[UltimatePooling Namespace](#)

# UltimatePooling.Demo Namespace

## ▲ Classes

	Class	Description
	<a href="#">Benchmark</a>	
	<a href="#">Ex0_SpawnExample</a>	This class shows how to spawn and despawn prefab objects.
	<a href="#">Ex1_SpawnAtExample</a>	This class shows how to spawn prefab objects using the overloaded methods.
	<a href="#">Ex3_CreatePoolExample</a>	This class shows how to create a new pool for a prefab type and initialize its spawning values.
	<a href="#">Ex4_CreateResourcesPoolExample</a>	This example shows how to create a new resources pool for a prefab in the

resources folder.



## Ex5\_DestroyPoolExample

This example shows how to destroy a pool at runtime. Note that all pooled items are destroyed but all items spawned by this pool will remain in the scene.

# Benchmark Class

[Missing <summary> documentation for "T:UltimatePooling.Demo.Benchmark"]

## ▾ Inheritance Hierarchy

[SystemObject](#) **Object**  
**Component**  
**Behaviour**  
**MonoBehaviour**  
[UltimatePooling.DemoBenchmark](#)

**Namespace:** [UltimatePooling.Demo](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▾ Syntax


C#    JavaScript

[Copy](#)

```
public class Benchmark : MonoBehaviour
```


The [Benchmark](#) type exposes the following members.

## ▾ Constructors

	Name	Description
	<a href="#">Benchmark</a>	Initializes a new instance of the <a href="#">Benchmark</a> class

[Top](#)

## ▾ Fields

Name	Description
 <a href="#">prefab</a>	

---

[Top](#)

## ▲ See Also

Reference

[UltimatePooling.Demo Namespace](#)

---

# Benchmark Constructor

Initializes a new instance of the [Benchmark](#) class

**Namespace:** [UltimatePooling.Demo](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

**C#**    **JavaScript**

[Copy](#)

```
public Benchmark()
```

## ▲ See Also

Reference


[Benchmark Class](#)

[UltimatePooling.Demo Namespace](#)

# Benchmark Fields

The [Benchmark](#) type exposes the following members.

## ▲ Fields

	Name	Description
	<a href="#">prefab</a>	

[Top](#)

## ▲ See Also

Reference

[Benchmark Class](#)

[UltimatePooling.Demo Namespace](#)

# Benchmarkprefab Field

[Missing <summary> documentation for "F:UltimatePooling.Demo.Benchmark.prefab"]

**Namespace:** [UltimatePooling.Demo](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public GameObject prefab
```

Field Value

Type: **GameObject**

## ▲ See Also

Reference

[Benchmark Class](#)

[UltimatePooling.Demo Namespace](#)



# Ex0\_SpawnExample Class

This class shows how to spawn and despawn prefab objects.

## ▲ Inheritance Hierarchy

[SystemObject](#) **Object**  
**Component**  
**Behaviour**  
**MonoBehaviour**  
[UltimatePooling.DemoEx0\\_SpawnExample](#)

**Namespace:** [UltimatePooling.Demo](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax


C#    JavaScript

[Copy](#)

```
public class Ex0_SpawnExample : MonoBehaviour
```


The [Ex0\\_SpawnExample](#) type exposes the following members.

## ▲ Constructors

Name	Description
 <a href="#">Ex0_SpawnExample</a>	Initializes a new instance of the <a href="#">Ex0_SpawnExample</a> class

[Top](#)

## ▲ Fields

	Name	Description
	<a href="#">prefab</a>	The prefab we want to spawn - Assigned in the editor inspector.

---

[Top](#)

## ▲ See Also

Reference

[UltimatePooling.Demo Namespace](#)

---

# Ex0\_SpawnExample Constructor

Initializes a new instance of the [Ex0\\_SpawnExample](#) class

**Namespace:** [UltimatePooling.Demo](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public Ex0_SpawnExample()
```

## ▲ See Also

### Reference


[Ex0\\_SpawnExample Class](#)

[UltimatePooling.Demo Namespace](#)

# Ex0\_SpawnExample Fields

The [Ex0\\_SpawnExample](#) type exposes the following members.

## ▲ Fields

	Name	Description
	<a href="#">prefab</a>	The prefab we want to spawn - Assigned in the editor inspector.

[Top](#)

## ▲ See Also

Reference

[Ex0\\_SpawnExample Class](#)

[UltimatePooling.Demo Namespace](#)

# Ex0\_SpawnExampleprefab Field

The prefab we want to spawn - Assigned in the editor inspector.

**Namespace:** [UltimatePooling.Demo](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public GameObject prefab
```

Field Value

Type: **GameObject**

## ▲ See Also

Reference

[Ex0\\_SpawnExample Class](#)

[UltimatePooling.Demo Namespace](#)

# Ex1\_SpawnAtExample Class

This class shows how to spawn prefab objects using the overloaded methods.

## ▲ Inheritance Hierarchy

[SystemObject](#) **Object**

**Component**

**Behaviour**

**MonoBehaviour**

UltimatePooling.DemoEx1\_SpawnAtExample

**Namespace:** [UltimatePooling.Demo](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax


C#    JavaScript

[Copy](#)

```
public class Ex1_SpawnAtExample : MonoBehaviour
```


The [Ex1\\_SpawnAtExample](#) type exposes the following members.

## ▲ Constructors

	Name	Description
	<a href="#">Ex1_SpawnAtExample</a>	Initializes a new instance of the <a href="#">Ex1_SpawnAtExample</a> class

[Top](#)

## ▲ Fields

	Name	Description
	<a href="#">prefab</a>	The prefab we want to spawn - Assigned in the editor inspector.

---

[Top](#)

## ▲ See Also

Reference

[UltimatePooling.Demo Namespace](#)

# Ex1\_SpawnAtExample Constructor

Initializes a new instance of the [Ex1\\_SpawnAtExample](#) class

**Namespace:** [UltimatePooling.Demo](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version:  
0.0.0.0

## ▲ Syntax

**C#**    **JavaScript**

[Copy](#)

```
public Ex1_SpawnAtExample()
```

## ▲ See Also

### Reference

[Ex1\\_SpawnAtExample Class](#)


[UltimatePooling.Demo Namespace](#)



# Ex1\_SpawnAtExample Fields

The [Ex1\\_SpawnAtExample](#) type exposes the following members.

## ▲ Fields

	Name	Description
	<a href="#">prefab</a>	The prefab we want to spawn - Assigned in the editor inspector.

[Top](#)

## ▲ See Also

Reference

[Ex1\\_SpawnAtExample Class](#)

[UltimatePooling.Demo Namespace](#)

# Ex1\_SpawnAtExampleprefab Field

The prefab we want to spawn - Assigned in the editor inspector.

**Namespace:** [UltimatePooling.Demo](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public GameObject prefab
```

Field Value

Type: **GameObject**

## ▲ See Also

Reference

[Ex1\\_SpawnAtExample Class](#)

[UltimatePooling.Demo Namespace](#)

# Ex3\_CreatePoolExample Class

This class shows how to create a new pool for a prefab type and initialize its spawning values.

## ▲ Inheritance Hierarchy

[SystemObject](#) **Object**

**Component**

**Behaviour**

**MonoBehaviour**

[UltimatePooling.DemoEx3\\_CreatePoolExample](#)

**Namespace:** [UltimatePooling.Demo](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax


**C#**    **JavaScript**

[Copy](#)

```
public class Ex3_CreatePoolExample : MonoBehaviour
```


The [Ex3\\_CreatePoolExample](#) type exposes the following members.

## ▲ Constructors

Name	Description
 <a href="#">Ex3_CreatePoolExample</a>	Initializes a new instance of the <a href="#">Ex3_CreatePoolExample</a> class

[Top](#)

## ▲ Fields

	Name	Description
	<a href="#">prefab</a>	The prefab we want to spawn - Assigned in the editor inspector.

---

[Top](#)

## ▲ See Also

Reference

[UltimatePooling.Demo Namespace](#)

---

# Ex3\_CreatePoolExample Constructor

Initializes a new instance of the [Ex3\\_CreatePoolExample](#) class

**Namespace:** [UltimatePooling.Demo](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version:  
0.0.0.0

## ▲ Syntax

**C#**    **JavaScript**

[Copy](#)

```
public Ex3_CreatePoolExample()
```

## ▲ See Also

### Reference


[Ex3\\_CreatePoolExample Class](#)

[UltimatePooling.Demo Namespace](#)

# Ex3\_CreatePoolExample Fields

The [Ex3\\_CreatePoolExample](#) type exposes the following members.

## ▲ Fields

	Name	Description
	<a href="#">prefab</a>	The prefab we want to spawn - Assigned in the editor inspector.

[Top](#)

## ▲ See Also

Reference

[Ex3\\_CreatePoolExample Class](#)

[UltimatePooling.Demo Namespace](#)

# Ex3\_CreatePoolExampleprefab Field

The prefab we want to spawn - Assigned in the editor inspector.

**Namespace:** [UltimatePooling.Demo](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public GameObject prefab
```

Field Value

Type: **GameObject**

## ▲ See Also

Reference

[Ex3\\_CreatePoolExample Class](#)

[UltimatePooling.Demo Namespace](#)

# Ex4\_CreateResourcesPoolExample Class

This example shows how to create a new resources pool for a prefab in the resources folder.

## ▾ Inheritance Hierarchy

[SystemObject](#) **Object**  
**Component**  
**Behaviour**  
**MonoBehaviour**  
[UltimatePooling.DemoEx4\\_CreateResourcesPoolExample](#)

**Namespace:** [UltimatePooling.Demo](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▾ Syntax


C#    JavaScript

[Copy](#)

```
public class Ex4_CreateResourcesPoolExample : MonoBehaviour
```

The [Ex4\\_CreateResourcesPoolExample](#) type exposes the following members.

## ▾ Constructors

Name	Description
 <a href="#">Ex4_CreateResourcesPoolExample</a>	Initializes a new instance of the <a href="#">Ex4_CreateResourcesPoolExample</a> class



[Top](#)

## ▲ Fields

	Name	Description
◆	<a href="#">prefabName</a>	The name of the prefab we want to spawn - Located in the resources folder.

---

[Top](#)

## ▲ See Also

Reference

[UltimatePooling.Demo Namespace](#)

---

# Ex4\_CreateResourcesPoolExample Constructor

Initializes a new instance of the [Ex4\\_CreateResourcesPoolExample](#) class

**Namespace:** [UltimatePooling.Demo](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax

C#    JavaScript

[Copy](#)

```
public Ex4_CreateResourcesPoolExample()
```

## ▲ See Also

Reference


[Ex4\\_CreateResourcesPoolExample Class](#)

[UltimatePooling.Demo Namespace](#)

# Ex4\_CreateResourcesPoolExample Fields

The [Ex4\\_CreateResourcesPoolExample](#) type exposes the following members.

## ▲ Fields

	Name	Description
	<a href="#">prefabName</a>	The name of the prefab we want to spawn - Located in the resources folder.

[Top](#)

## ▲ See Also

Reference

[Ex4\\_CreateResourcesPoolExample Class](#)

[UltimatePooling.Demo Namespace](#)

# Ex4\_CreateResourcesPoolExample Field

The name of the prefab we want to spawn - Located in the resources folder.

**Namespace:** [UltimatePooling.Demo](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▾ Syntax

C#    JavaScript

[Copy](#)

```
public string prefabName
```

Field Value

Type: [String](#)

## ▾ See Also

Reference

[Ex4\\_CreateResourcesPoolExample Class](#)

[UltimatePooling.Demo Namespace](#)

# Ex5\_DestroyPoolExample Class

This example shows how to destroy a pool at runtime. Note that all pooled iters are destroyed but all items spawned by this pool will remain in the scene.

## ▲ Inheritance Hierarchy

[SystemObject](#) **Object**

**Component**

**Behaviour**

**MonoBehaviour**

[UltimatePooling.DemoEx5\\_DestroyPoolExample](#)

**Namespace:** [UltimatePooling.Demo](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▲ Syntax


C#    JavaScript

[Copy](#)

```
public class Ex5_DestroyPoolExample : MonoBehaviour
```


The [Ex5\\_DestroyPoolExample](#) type exposes the following members.

## ▲ Constructors

Name	Description
 <a href="#">Ex5_DestroyPoolExample</a>	Initializes a new instance of the <a href="#">Ex5_DestroyPoolExample</a> class

[Top](#)

## ▲ Fields

	Name	Description
	<a href="#">prefab</a>	The prefab we want to spawn - Assigned in the editor inspector.

---

[Top](#)

## ▲ See Also

Reference

[UltimatePooling.Demo Namespace](#)

# Ex5\_DestroyPoolExample Constructor

Initializes a new instance of the [Ex5\\_DestroyPoolExample](#) class

**Namespace:** [UltimatePooling.Demo](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version:  
0.0.0.0

## ▲ Syntax

**C#**    **JavaScript**

[Copy](#)

```
public Ex5_DestroyPoolExample()
```

## ▲ See Also

### Reference


[Ex5\\_DestroyPoolExample Class](#)

[UltimatePooling.Demo Namespace](#)

# Ex5\_DestroyPoolExample Fields

The [Ex5\\_DestroyPoolExample](#) type exposes the following members.

## ▲ Fields

	Name	Description
	<a href="#">prefab</a>	The prefab we want to spawn - Assigned in the editor inspector.

[Top](#)

## ▲ See Also

### Reference

[Ex5\\_DestroyPoolExample Class](#)

[UltimatePooling.Demo Namespace](#)



# Ex5\_DestroyPoolExampleprefab Field

The prefab we want to spawn - Assigned in the editor inspector.

**Namespace:** [UltimatePooling.Demo](#)

**Assembly:** Assembly-CSharp (in Assembly-CSharp.dll) Version: 0.0.0.0

## ▸ Syntax

C#    JavaScript

[Copy](#)

```
public GameObject prefab
```

Field Value

Type: **GameObject**

## ▸ See Also

Reference

[Ex5\\_DestroyPoolExample Class](#)

[UltimatePooling.Demo Namespace](#)