Api documentation

Collider2D

class in UnityEngine

/

Inherits from:Behaviour

/

Implemented in:UnityEngine.Physics2DModule

Description

Parent class for collider types used with 2D gameplay.

See Also: BoxCollider2D, CircleCollider2D, PolygonCollider2D, EdgeCollider2D.

Properties

attachedRigidbody	The Rigidbody2D attached to the Collider2D.
bounciness	Get the bounciness used by the collider.
bounds	The world space bounding area of the collider.
composite	Get the CompositeCollider2D that is available to be attached to the collider.
density	The density of the collider used to calculate its mass (when auto mass is enabled).
friction	Get the friction used by the collider.
isTrigger	Is this collider configured as a trigger?
offset	The local offset of the collider geometry.

shapeCount	The number of separate shaped regions in the collider.
sharedMaterial	The PhysicsMaterial2D that is applied to this collider.
usedByComposite	Sets whether the Collider will be used or not used by a CompositeCollider2D.
usedByEffector	Whether the collider is used by an attached effector or not.

Public Methods

Cast	Casts the Collider shape into the Scene starting at the Collider position ignoring th
ClosestPoint	Returns a point on the perimeter of this Collider that is closest to the specified posi
CreateMesh	Creates a planar Mesh that is identical to the area defined by the Collider2D geome
Distance	Calculates the minimum separation of this collider against another collider.
GetContacts	Retrieves all contact points for this Collider.
GetShapeHash	Generates a simple hash value based upon the geometry of the Collider2D.
IsTouching	Check whether this collider is touching the collider or not.
IsTouchingLayers	Checks whether this collider is touching any colliders on the specified layerMask c
OverlapCollider	Get a list of all colliders that overlap this collider.
OverlapPoint	Check if a collider overlaps a point in space.
Raycast	Casts a ray into the Scene that starts at the Collider position and ignores the Collid

Messages

OnCollisionEnter2D	Sent when an incoming collider makes contact with this object's collider (2D physi

OnCollisionExit2D	Sent when a collider on another object stops touching this object's collider (2D phy
OnCollisionStay2D	Sent each frame where a collider on another object is touching this object's collide
OnTriggerEnter2D	Sent when another object enters a trigger collider attached to this object (2D physic
OnTriggerExit2D	Sent when another object leaves a trigger collider attached to this object (2D physi
OnTriggerStay2D	Sent each frame where another object is within a trigger collider attached to this ob

Inherited Members

Properties

enabled	Enabled Behaviours are Updated, disabled Behaviours are not.
isActiveAndEnabled	Has the Behaviour had active and enabled called?
gameObject	The game object this component is attached to. A component is always attached to
tag	The tag of this game object.
transform	The Transform attached to this GameObject.
hideFlags	Should the object be hidden, saved with the Scene or modifiable by the user?
name	The name of the object.

Public Methods

BroadcastMessage	Calls the method named methodName on every MonoBehaviour in this game obje
CompareTag	Is this game object tagged with tag ?
GetComponent	Returns the component of Type type if the game object has one attached, null if it

GetComponentInChildren	Returns the component of Type type in the GameObject or any of its children usin
GetComponentInParent	Returns the component of Type type in the GameObject or any of its parents.
GetComponents	Returns all components of Type type in the GameObject.
GetComponentsInChildren	Returns all components of Type type in the GameObject or any of its children.
GetComponentsInParent	Returns all components of Type type in the GameObject or any of its parents.
SendMessage	Calls the method named methodName on every MonoBehaviour in this game obje
SendMessageUpwards	Calls the method named methodName on every MonoBehaviour in this game obje
TryGetComponent	Gets the component of the specified type, if it exists.
GetInstanceID	Returns the instance id of the object.
ToString	Returns the name of the object.

Static Methods

Destroy	Removes a GameObject, component or asset.
DestroyImmediate	Destroys the object obj immediately. You are strongly recommended to use Destro
DontDestroyOnLoad	Do not destroy the target Object when loading a new Scene.
FindObjectOfType	Returns the first active loaded object of Type type.
FindObjectsOfType	Returns a list of all active loaded objects of Type type.
Instantiate	Clones the object original and returns the clone.

Operators

bool	Does the object exist?
operator !=	Compares if two objects refer to a different object.
operator ==	Compares two object references to see if they refer to the same object.

Transform

class in UnityEngine

/

Inherits from:Component

/

Implemented in:UnityEngine.CoreModule

Description

Position, rotation and scale of an object.

Every object in a Scene has a Transform. It's used to store and manipulate the position, rotation and scale of the object. Every Transform can have a parent, which allows you to apply position, rotation and scale hierarchically. This is the hierarchy seen in the Hierarchy pane. They also support enumerators so you can loop through children using:

```
using UnityEngine;

public class Example : MonoBehaviour

{

    // Moves all transform children 10 units upwards!

    void Start()

    {

        foreach (Transform child in transform)
```

```
        {
            child.position += Vector3.up * 10.0f;
        }
    }
}
```

See Also: The component reference, Physics class.

Properties

childCount	The number of children the parent Transform has.
eulerAngles	The rotation as Euler angles in degrees.
forward	Returns a normalized vector representing the blue axis of the transform in world sp
hasChanged	Has the transform changed since the last time the flag was set to 'false'?
hierarchyCapacity	The transform capacity of the transform's hierarchy data structure.
hierarchyCount	The number of transforms in the transform's hierarchy data structure.
localEulerAngles	The rotation as Euler angles in degrees relative to the parent transform's rotation.
localPosition	Position of the transform relative to the parent transform.
localRotation	The rotation of the transform relative to the transform rotation of the parent.
localScale	The scale of the transform relative to the GameObjects parent.
localToWorldMatrix	Matrix that transforms a point from local space into world space (Read Only).

lossyScale	The global scale of the object (Read Only).
parent	The parent of the transform.
position	The world space position of the Transform.
right	The red axis of the transform in world space.
root	Returns the topmost transform in the hierarchy.
rotation	A Quaternion that stores the rotation of the Transform in world space.
up	The green axis of the transform in world space.
worldToLocalMatrix	Matrix that transforms a point from world space into local space (Read Only).

Public Methods

DetachChildren	Unparents all children.
Find	Finds a child by n and returns it.
GetChild	Returns a transform child by index.
GetSiblingIndex	Gets the sibling index.
InverseTransformDirection	Transforms a direction from world space to local space. The opposite of Transform
InverseTransformPoint	Transforms position from world space to local space.
InverseTransformVector	Transforms a vector from world space to local space. The opposite of Transform.T
IsChildOf	Is this transform a child of parent?
LookAt	Rotates the transform so the forward vector points at /target/'s current position.

Rotate	Use Transform.Rotate to rotate GameObjects in a variety of ways. The rotation is c
RotateAround	Rotates the transform about axis passing through point in world coordinates by ang
SetAsFirstSibling	Move the transform to the start of the local transform list.
SetAsLastSibling	Move the transform to the end of the local transform list.
SetParent	Set the parent of the transform.
SetPositionAndRotation	Sets the world space position and rotation of the Transform component.
SetSiblingIndex	Sets the sibling index.
TransformDirection	Transforms direction from local space to world space.
TransformPoint	Transforms position from local space to world space.
TransformVector	Transforms vector from local space to world space.
Translate	Moves the transform in the direction and distance of translation.

Inherited Members

Properties

gameObject	The game object this component is attached to. A component is always attached to
tag	The tag of this game object.
transform	The Transform attached to this GameObject.
hideFlags	Should the object be hidden, saved with the Scene or modifiable by the user?
name	The name of the object.

Public Methods

BroadcastMessage	Calls the method named methodName on every MonoBehaviour in this game obje
CompareTag	Is this game object tagged with tag ?
GetComponent	Returns the component of Type type if the game object has one attached, null if it
GetComponentInChildren	Returns the component of Type type in the GameObject or any of its children usin
GetComponentInParent	Returns the component of Type type in the GameObject or any of its parents.
GetComponents	Returns all components of Type type in the GameObject.
GetComponentsInChildren	Returns all components of Type type in the GameObject or any of its children.
GetComponentsInParent	Returns all components of Type type in the GameObject or any of its parents.
SendMessage	Calls the method named methodName on every MonoBehaviour in this game obje
SendMessageUpwards	Calls the method named methodName on every MonoBehaviour in this game obje
TryGetComponent	Gets the component of the specified type, if it exists.
GetInstanceID	Returns the instance id of the object.
ToString	Returns the name of the object.

Static Methods

Destroy	Removes a GameObject, component or asset.
DestroyImmediate	Destroys the object obj immediately. You are strongly recommended to use Destro
DontDestroyOnLoad	Do not destroy the target Object when loading a new Scene.
FindObjectOfType	Returns the first active loaded object of Type type.

FindObjectsOfType	Returns a list of all active loaded objects of Type type.
Instantiate	Clones the object original and returns the clone.

Operators

bool	Does the object exist?
operator !=	Compares if two objects refer to a different object.
operator ==	Compares two object references to see if they refer to the same object.

Vector2

struct in UnityEngine

/

Implemented in:UnityEngine.CoreModule

Description

Representation of 2D vectors and points.

This structure is used in some places to represent 2D positions and vectors (e.g. texture coordinates in a Mesh or texture offsets in Material). In the majority of other cases a Vector3 is used.

Static Properties

down	Shorthand for writing Vector2(0, -1).
left	Shorthand for writing Vector2(-1, 0).
negativeInfinity	Shorthand for writing Vector2(float.NegativeInfinity, float.NegativeInfinity).
one	Shorthand for writing Vector2(1, 1).
positiveInfinity	Shorthand for writing Vector2(float.PositiveInfinity, float.PositiveInfinity).

right	Shorthand for writing Vector2(1, 0).
up	Shorthand for writing Vector2(0, 1).
zero	Shorthand for writing Vector2(0, 0).

Properties

magnitude	Returns the length of this vector (Read Only).
normalized	Returns this vector with a magnitude of 1 (Read Only).
sqrMagnitude	Returns the squared length of this vector (Read Only).
this[int]	Access the x or y component using [0] or [1] respectively.
x	X component of the vector.
y	Y component of the vector.

Constructors

Vector2	Constructs a new vector with given x, y components.

Public Methods

Equals	Returns true if the given vector is exactly equal to this vector.
Normalize	Makes this vector have a magnitude of 1.
Set	Set x and y components of an existing Vector2.
ToString	Returns a nicely formatted string for this vector.

Static Methods

Angle	Returns the unsigned angle in degrees between from and to.
ClampMagnitude	Returns a copy of vector with its magnitude clamped to maxLength.
Distance	Returns the distance between a and b.
Dot	Dot Product of two vectors.
Lerp	Linearly interpolates between vectors a and b by t.
LerpUnclamped	Linearly interpolates between vectors a and b by t.
Max	Returns a vector that is made from the largest components of two vectors.
Min	Returns a vector that is made from the smallest components of two vectors.
MoveTowards	Moves a point current towards target.
Perpendicular	Returns the 2D vector perpendicular to this 2D vector. The result is always rotated
Reflect	Reflects a vector off the vector defined by a normal.
Scale	Multiplies two vectors component-wise.
SignedAngle	Returns the signed angle in degrees between from and to.
SmoothDamp	Gradually changes a vector towards a desired goal over time.

Operators

operator -	Subtracts one vector from another.
operator *	Multiplies a vector by a number.
operator /	Divides a vector by a number.

operator +	Adds two vectors.
operator ==	Returns true if two vectors are approximately equal.
Vector2	Converts a Vector3 to a Vector2.
Vector3	Converts a Vector2 to a Vector3.

Rigidbody2D

class in UnityEngine

/

Inherits from:Component

/

Implemented in:UnityEngine.Physics2DModule

Description

Rigidbody physics component for 2D sprites.

The Rigidbody2D class essentially provides the same functionality in 2D that the Rigidbody class provides in 3D. Adding a Rigidbody2D component to a sprite puts it under the control of the physics engine. By itself, this means that the sprite will be affected by gravity and can be controlled from scripts using forces. By adding the appropriate collider component, the sprite will also respond to collisions with other sprites. This behaviour comes entirely from Unity's physics system; very little code is required to get impressive and authentic physical behaviour and allows for "emergent" gameplay that was not explicitly coded into the game.

See Also: Rigidbody class, SpriteRenderer class, Collider2D class, Joint2D class.

Properties

angularDrag	Coefficient of angular drag.
angularVelocity	Angular velocity in degrees per second.
attachedColliderCount	Returns the number of Collider2D attached to this Rigidbody2D.

bodyType	The physical behaviour type of the Rigidbody2D.
centerOfMass	The center of mass of the rigidBody in local space.
collisionDetectionMode	The method used by the physics engine to check if two objects have collided.
constraints	Controls which degrees of freedom are allowed for the simulation of this Rigidbod
drag	Coefficient of drag.
freezeRotation	Controls whether physics will change the rotation of the object.
gravityScale	The degree to which this object is affected by gravity.
inertia	The rigidBody rotational inertia.
interpolation	Physics interpolation used between updates.
isKinematic	Should this rigidbody be taken out of physics control?
mass	Mass of the Rigidbody.
position	The position of the rigidbody.
rotation	The rotation of the rigidbody.
sharedMaterial	The PhysicsMaterial2D that is applied to all Collider2D attached to this Rigidbody
simulated	Indicates whether the rigid body should be simulated or not by the physics system.
sleepMode	The sleep state that the rigidbody will initially be in.
useAutoMass	Should the total rigid-body mass be automatically calculated from the [[Collider2D
useFullKinematicContacts	Should kinematic/kinematic and kinematic/static collisions be allowed?

velocity	Linear velocity of the Rigidbody in units per second.
worldCenterOfMass	Gets the center of mass of the rigidBody in global space.

Public Methods

AddForce	Apply a force to the rigidbody.
AddForceAtPosition	Apply a force at a given position in space.
AddRelativeForce	Adds a force to the rigidbody2D relative to its coordinate system.
AddTorque	Apply a torque at the rigidbody's centre of mass.
Cast	All the Collider2D shapes attached to the Rigidbody2D are cast into the Scene star
ClosestPoint	Returns a point on the perimeter of all enabled Colliders attached to this Rigidbody
Distance	Calculates the minimum distance of this collider against all Collider2D attached to
GetAttachedColliders	Returns all Collider2D that are attached to this Rigidbody2D.
GetContacts	Retrieves all contact points for all of the Collider(s) attached to this Rigidbody.
GetPoint	Get a local space point given the point point in rigidBody global space.
GetPointVelocity	The velocity of the rigidbody at the point Point in global space.
GetRelativePoint	Get a global space point given the point relativePoint in rigidBody local space.
GetRelativePointVelocity	The velocity of the rigidbody at the point Point in local space.
GetRelativeVector	Get a global space vector given the vector relativeVector in rigidBody local space.
GetVector	Get a local space vector given the vector vector in rigidBody global space.

IsAwake	Is the rigidbody "awake"?
IsSleeping	Is the rigidbody "sleeping"?
IsTouching	Checks whether the collider is touching any of the collider(s) attached to this rigidl
IsTouchingLayers	Checks whether any of the collider(s) attached to this rigidbody are touching any c
MovePosition	Moves the rigidbody to position.
MoveRotation	Rotates the Rigidbody to angle (given in degrees).
OverlapCollider	Get a list of all Colliders that overlap all Colliders attached to this Rigidbody2D.
OverlapPoint	Check if any of the Rigidbody2D colliders overlap a point in space.
SetRotation	Sets the rotation of the Rigidbody2D to angle (given in degrees).
Sleep	Make the rigidbody "sleep".
WakeUp	Disables the "sleeping" state of a rigidbody.

Inherited Members

Properties

gameObject	The game object this component is attached to. A component is always attached to
tag	The tag of this game object.
transform	The Transform attached to this GameObject.
hideFlags	Should the object be hidden, saved with the Scene or modifiable by the user?
name	The name of the object.

Public Methods

BroadcastMessage	Calls the method named methodName on every MonoBehaviour in this game obje
CompareTag	Is this game object tagged with tag ?
GetComponent	Returns the component of Type type if the game object has one attached, null if it
GetComponentInChildren	Returns the component of Type type in the GameObject or any of its children usin
GetComponentInParent	Returns the component of Type type in the GameObject or any of its parents.
GetComponents	Returns all components of Type type in the GameObject.
GetComponentsInChildren	Returns all components of Type type in the GameObject or any of its children.
GetComponentsInParent	Returns all components of Type type in the GameObject or any of its parents.
SendMessage	Calls the method named methodName on every MonoBehaviour in this game obje
SendMessageUpwards	Calls the method named methodName on every MonoBehaviour in this game obje
TryGetComponent	Gets the component of the specified type, if it exists.
GetInstanceID	Returns the instance id of the object.
ToString	Returns the name of the object.

Static Methods

Destroy	Removes a GameObject, component or asset.
DestroyImmediate	Destroys the object obj immediately. You are strongly recommended to use Destro
DontDestroyOnLoad	Do not destroy the target Object when loading a new Scene.
FindObjectOfType	Returns the first active loaded object of Type type.

FindObjectsOfType	Returns a list of all active loaded objects of Type type.
Instantiate	Clones the object original and returns the clone.

Operators

bool	Does the object exist?
operator !=	Compares if two objects refer to a different object.
operator ==	Compares two object references to see if they refer to the same object.

MonoBehaviour

class in UnityEngine

/

Inherits from:Behaviour

/

Implemented in:UnityEngine.CoreModule

Description

MonoBehaviour is the base class from which every Unity script derives.

When you use C#, you must explicitly derive from MonoBehaviour.

This class doesn't support the null-conditional operator (**?.**) and the null-coalescing operator (**??**).

For code samples, see the individual MonoBehaviour methods.

Note: There is a checkbox for enabling or disabling MonoBehaviour in the Unity Editor. It disables functions when unticked. If none of these functions are present in the script, the Unity Editor does not display the checkbox. The functions are:

Start()
Update()
FixedUpdate()
LateUpdate()
OnGUI()
OnDisable()
OnEnable()

See Also: The Deactivating GameObjects page in the manual.

Properties

runInEditMode	Allow a specific instance of a MonoBehaviour to run in edit mode (only available i
useGUILayout	Disabling this lets you skip the GUI layout phase.

Public Methods

CancelInvoke	Cancels all Invoke calls on this MonoBehaviour.
Invoke	Invokes the method methodName in time seconds.
InvokeRepeating	Invokes the method methodName in time seconds, then repeatedly every repeatRat
IsInvoking	Is any invoke on methodName pending?
StartCoroutine	Starts a Coroutine.
StopAllCoroutines	Stops all coroutines running on this behaviour.
StopCoroutine	Stops the first coroutine named methodName, or the coroutine stored in routine run

Static Methods

print	Logs message to the Unity Console (identical to Debug.Log).

Messages

Awake	Awake is called when the script instance is being loaded.
FixedUpdate	Frame-rate independent MonoBehaviour.FixedUpdate message for physics calcula
LateUpdate	LateUpdate is called every frame, if the Behaviour is enabled.

OnAnimatorIK	Callback for setting up animation IK (inverse kinematics).
OnAnimatorMove	Callback for processing animation movements for modifying root motion.
OnApplicationFocus	Sent to all GameObjects when the player gets or loses focus.
OnApplicationPause	Sent to all GameObjects when the application pauses.
OnApplicationQuit	Sent to all GameObjects before the application quits.
OnAudioFilterRead	If OnAudioFilterRead is implemented, Unity will insert a custom filter into the au
OnBecameInvisible	OnBecameInvisible is called when the renderer is no longer visible by any camera
OnBecameVisible	OnBecameVisible is called when the renderer became visible by any camera.
OnCollisionEnter	OnCollisionEnter is called when this collider/rigidbody has begun touching anothe
OnCollisionEnter2D	Sent when an incoming collider makes contact with this object's collider (2D phys
OnCollisionExit	OnCollisionExit is called when this collider/rigidbody has stopped touching anoth
OnCollisionExit2D	Sent when a collider on another object stops touching this object's collider (2D ph
OnCollisionStay	:ref::OnCollisionStay is called once per frame for every collider/rigidbody that is t
OnCollisionStay2D	Sent each frame where a collider on another object is touching this object's collide
OnConnectedToServer	Called on the client when you have successfully connected to a server.
OnControllerColliderHit	OnControllerColliderHit is called when the controller hits a collider while perform
OnDestroy	Destroying the attached Behaviour will result in the game or Scene receiving OnD
OnDisable	This function is called when the behaviour becomes disabled.

OnDisconnectedFromServer	Called on the client when the connection was lost or you disconnected from the se
OnDrawGizmos	Implement OnDrawGizmos if you want to draw gizmos that are also pickable and
OnDrawGizmosSelected	Implement OnDrawGizmosSelected to draw a gizmo if the object is selected.
OnEnable	This function is called when the object becomes enabled and active.
OnFailedToConnect	Called on the client when a connection attempt fails for some reason.
OnFailedToConnectToMasterServer	Called on clients or servers when there is a problem connecting to the MasterServe
OnGUI	OnGUI is called for rendering and handling GUI events.
OnJointBreak	Called when a joint attached to the same game object broke.
OnJointBreak2D	Called when a Joint2D attached to the same game object breaks.
OnMasterServerEvent	Called on clients or servers when reporting events from the MasterServer.
OnMouseDown	OnMouseDown is called when the user has pressed the mouse button while over th
OnMouseDrag	OnMouseDrag is called when the user has clicked on a Collider and is still holding
OnMouseEnter	Called when the mouse enters the Collider.
OnMouseExit	Called when the mouse is not any longer over the Collider.
OnMouseOver	Called every frame while the mouse is over the Collider.
OnMouseUp	OnMouseUp is called when the user has released the mouse button.
OnMouseUpAsButton	OnMouseUpAsButton is only called when the mouse is released over the same Co
OnNetworkInstantiate	Called on objects which have been network instantiated with Network.Instantiate.

OnParticleCollision	OnParticleCollision is called when a particle hits a Collider.
OnParticleSystemStopped	OnParticleSystemStopped is called when all particles in the system have died, and non-looping system has been exceeded.
OnParticleTrigger	OnParticleTrigger is called when any particles in a Particle System meet the condi
OnParticleUpdateJobScheduled	OnParticleUpdateJobScheduled is called when a Particle System's built-in update j
OnPlayerConnected	Called on the server whenever a new player has successfully connected.
OnPlayerDisconnected	Called on the server whenever a player disconnected from the server.
OnPostRender	OnPostRender is called after a camera finished rendering the Scene.
OnPreCull	OnPreCull is called before a camera culls the Scene.
OnPreRender	OnPreRender is called before a camera starts rendering the Scene.
OnRenderImage	OnRenderImage is called after all rendering is complete to render image.
OnRenderObject	OnRenderObject is called after camera has rendered the Scene.
OnSerializeNetworkView	Used to customize synchronization of variables in a script watched by a network v
OnServerInitialized	Called on the server whenever a Network.InitializeServer was invoked and has cor
OnTransformChildrenChanged	This function is called when the list of children of the transform of the GameObjec
OnTransformParentChanged	This function is called when the parent property of the transform of the GameObje
OnTriggerEnter	When a GameObject collides with another GameObject, Unity calls OnTriggerEnt
OnTriggerEnter2D	Sent when another object enters a trigger collider attached to this object (2D physi
OnTriggerExit	OnTriggerExit is called when the Collider other has stopped touching the trigger.

OnTriggerExit2D	Sent when another object leaves a trigger collider attached to this object (2D physi
OnTriggerStay	OnTriggerStay is called once per physics update for every Collider other that is tou
OnTriggerStay2D	Sent each frame where another object is within a trigger collider attached to this ob
OnValidate	This function is called when the script is loaded or a value is changed in the Inspec
OnWillRenderObject	OnWillRenderObject is called for each camera if the object is visible and not a UI
Reset	Reset to default values.
Start	Start is called on the frame when a script is enabled just before any of the Update m
Update	Update is called every frame, if the MonoBehaviour is enabled.

Inherited Members

Properties

enabled	Enabled Behaviours are Updated, disabled Behaviours are not.
isActiveAndEnabled	Has the Behaviour had active and enabled called?
gameObject	The game object this component is attached to. A component is always attached to
tag	The tag of this game object.
transform	The Transform attached to this GameObject.
hideFlags	Should the object be hidden, saved with the Scene or modifiable by the user?
name	The name of the object.

Public Methods

BroadcastMessage	Calls the method named methodName on every MonoBehaviour in this game obje
CompareTag	Is this game object tagged with tag ?
GetComponent	Returns the component of Type type if the game object has one attached, null if it
GetComponentInChildren	Returns the component of Type type in the GameObject or any of its children usin
GetComponentInParent	Returns the component of Type type in the GameObject or any of its parents.
GetComponents	Returns all components of Type type in the GameObject.
GetComponentsInChildren	Returns all components of Type type in the GameObject or any of its children.
GetComponentsInParent	Returns all components of Type type in the GameObject or any of its parents.
SendMessage	Calls the method named methodName on every MonoBehaviour in this game obje
SendMessageUpwards	Calls the method named methodName on every MonoBehaviour in this game obje
TryGetComponent	Gets the component of the specified type, if it exists.
GetInstanceID	Returns the instance id of the object.
ToString	Returns the name of the object.

Static Methods

Destroy	Removes a GameObject, component or asset.
DestroyImmediate	Destroys the object obj immediately. You are strongly recommended to use Destro
DontDestroyOnLoad	Do not destroy the target Object when loading a new Scene.
FindObjectOfType	Returns the first active loaded object of Type type.

FindObjectsOfType	Returns a list of all active loaded objects of Type type.
Instantiate	Clones the object original and returns the clone.

Operators

bool	Does the object exist?
operator !=	Compares if two objects refer to a different object.
operator ==	Compares two object references to see if they refer to the same object.

FixedUpdate

struct in UnityEngine.PlayerLoop

/

Implemented in:UnityEngine.CoreModule

Description

Update phase in the native player loop. **UnityEngine.AIModule**

Description

The AI module implements the path finding features in Unity.

Classes

NavMesh	Singleton class to access the baked NavMesh.
NavMeshAgent	Navigation mesh agent.
NavMeshBuilder	Navigation mesh builder interface.
NavMeshData	Contains and represents NavMesh data.

NavMeshObstacle	An obstacle for NavMeshAgents to avoid.
NavMeshPath	A path as calculated by the navigation system.
OffMeshLink	Link allowing movement outside the planar navigation mesh.

Structs

NavMeshBuildDebugSettings	Specify which of the temporary data generated while building the NavMesh should
NavMeshBuildMarkup	The NavMesh build markup allows you to control how certain objects are treated d
NavMeshBuildSettings	The NavMeshBuildSettings struct allows you to specify a collection of settings wh
NavMeshBuildSource	The input to the NavMesh builder is a list of NavMesh build sources.
NavMeshDataInstance	The instance is returned when adding NavMesh data.
NavMeshHit	Result information for NavMesh queries.
NavMeshLinkData	Used for runtime manipulation of links connecting polygons of the NavMesh.
NavMeshLinkInstance	An instance representing a link available for pathfinding.
NavMeshLocation	A world position that is guaranteed to be on the surface of the NavMesh.
NavMeshQuery	Object used for doing navigation operations in a NavMeshWorld.
NavMeshQueryFilter	Specifies which agent type and areas to consider when searching the NavMesh.
NavMeshTriangulation	Contains data describing a triangulation of a navmesh.
NavMeshWorld	Assembles together a collection of NavMesh surfaces and links that are used as a w
OffMeshLinkData	State of OffMeshLink.

PolygonId	Represents a compact identifier for the data of a NavMesh node.

Enumerations

NavMeshBuildDebugFlags	Bitmask used for operating with debug data from the NavMesh build process.
NavMeshBuildSourceShape	Used with NavMeshBuildSource to define the shape for building NavMesh.
NavMeshCollectGeometry	Used for specifying the type of geometry to collect. Used with NavMeshBuilder.C
NavMeshObstacleShape	Shape of the obstacle.
NavMeshPathStatus	Status of path.
NavMeshPolyTypes	The types of nodes in the navigation data.
ObstacleAvoidanceType	Level of obstacle avoidance.
OffMeshLinkType	Link type specifier.
PathQueryStatus	Bit flags representing the resulting state of NavMeshQuery operations.

This is the C# representation of an update phase in the native player loop. It can only be used to identify the update phase in native.

Collider2D.OnTriggerEnter2D(Collider2D)

Parameters

other	The other Collider2D involved in this collision.

Description

Sent when another object enters a trigger collider attached to this object (2D physics only).

Further information about the other collider is reported in the Collider2D parameter passed during the call.

Trigger events will be sent to disabled MonoBehaviours, to allow enabling Behaviours in response to collisions.

See Also: The Collider2D class and the OnTriggerExit2D and OnTriggerStay2D messages.

An OnTriggerEnter2D example is shown. This example has two empty GameObjects, called `GameObject1` and `GameObject2`. These both have script files which makes the example work. The first script, `Example1`, creates a Sprite and adds a BoxCollider2D and a Rigidbody2D. This object falls under gravity and collides with `Example2`. `Example2` has no visibility. (The rectangle it creates is visible in the Scene window.) Both GameObjects report the collision.

The script below is Example1.

```
using System.Collections;

using System.Collections.Generic;

using UnityEngine;

// Create GameObject1 that falls under gravity.  It will pass through

// Example2 and cause a collision.  GameObject1 is moved back to

// the start position and it will again start to fall under gravity.

public class Example1 : MonoBehaviour

{

    void Awake()

    {

        SpriteRenderer sr;

        sr = gameObject.AddComponent<SpriteRenderer>() as SpriteRenderer;

        sr.color = new Color(0.9f, 0.9f, 0.1f, 1.0f);

        BoxCollider2D bc;
```

```
        bc = gameObject.AddComponent<BoxCollider2D>() as BoxCollider2D;

        bc.size = new Vector2(1.0f, 1.0f);

        Rigidbody2D rb;

        rb = gameObject.AddComponent<Rigidbody2D>() as Rigidbody2D;

        rb.gravityScale = 1.0f;

        // A square in the Resources folder is used.

        gameObject.GetComponent<SpriteRenderer>().sprite =
Resources.Load<Sprite>("square");

        // GameObject1 starts 3 units in the Up direction.

        gameObject.transform.position = new Vector3(0.0f, 3.0f, 0.0f);

        gameObject.transform.localScale = new Vector3(0.5f, 0.5f, 1.0f);

    }

    private float timer = 0.0f;

    private bool restart = false;

    void FixedUpdate()

    {

        if (restart == true)

        {

            timer = timer + Time.deltaTime;

            if (timer > 0.25f)
```

```
            {

                gameObject.transform.position = new Vector3(0.0f, 3.0f,
0.0f);

                gameObject.GetComponent<Rigidbody2D>().velocity = new
Vector2(0.0f, 0.0f);

                restart = false;

            }

        }

    }

    // called when this GameObject collides with GameObject2.

    void OnTriggerEnter2D(Collider2D col)

    {

        Debug.Log("GameObject1 collided with " + col.name);

        restart = true;

        timer = 0.0f;

    }

}
```

This is `Example2` which is the collide script for the second GameObject:

```
using System.Collections;

using System.Collections.Generic;
```

```
using UnityEngine;

// Create a rectangle that the other GameObject will collide with.

// Note that this GameObject has no visibility.

// (View in the Scene view to see this GameObject.)
public class Example2 : MonoBehaviour

{

    void Awake()

    {

        BoxCollider2D bc;

        bc = gameObject.AddComponent<BoxCollider2D>() as BoxCollider2D;

        bc.size = new Vector2(3.0f, 1.0f);

        bc.isTrigger = true;
        gameObject.transform.localScale = new Vector3(3.0f, 1.0f, 1.0f);

        gameObject.transform.position = new Vector3(0.0f, -2.0f, 0.0f);

    }
    void OnTriggerEnter2D(Collider2D col)

    {

        Debug.Log("GameObject2 collided with " + col.name);

    }
```

```
}
```

Collider.OnCollisionEnter(Collision)

Parameters

other	The Collision data associated with this collision event.

Description

OnCollisionEnter is called when this collider/rigidbody has begun touching another rigidbody/collider.

In contrast to OnTriggerEnter, OnCollisionEnter is passed the Collision class and not a Collider. The Collision class contains information, for example, about contact points and impact velocity. Notes: Collision events are only sent if one of the colliders also has a non-kinematic rigidbody attached. Collision events will be sent to disabled MonoBehaviours, to allow enabling Behaviours in response to collisions.

```
using UnityEngine;

using System.Collections;

public class ExampleClass : MonoBehaviour

{

    AudioSource audioSource;

    void Start()

    {

        audioSource = GetComponent<AudioSource>();

    }

    void OnCollisionEnter(Collision collision)

    {
```

```
        foreach (ContactPoint contact in collision.contacts)
        {
            Debug.DrawRay(contact.point, contact.normal, Color.white);
        }
        if (collision.relativeVelocity.magnitude > 2)
            audioSource.Play();
    }
}
```

Another example:

```
// A grenade
// - instantiates an explosion Prefab when hitting a surface
// - then destroys itself
using UnityEngine;
using System.Collections;
public class ExampleClass : MonoBehaviour
{
    public Transform explosionPrefab;
    void OnCollisionEnter(Collision collision)
    {
```

```
        ContactPoint contact = collision.contacts[0];

        Quaternion rotation = Quaternion.FromToRotation(Vector3.up,
```

Vector2.ClampMagnitude

public static Vector2 **ClampMagnitude**(Vector2 **vector**, float **maxLength**);

Description

Returns a copy of `vector` with its magnitude clamped to `maxLength`.

```
contact.normal);

        Vector3 position = contact.point;

        Instantiate(explosionPrefab, position, rotation);

        Destroy(gameObject);
```

}GameObject

class in UnityEngine

/

Inherits from:Object

/

Implemented in:UnityEngine.CoreModule

Description

Base class for all entities in Unity Scenes.

Note: Many variables in the GameObject class have been removed. To access, for example GameObject.renderer in csharp use GetComponent<Renderer>() instead.

See Also: Component.

Properties

activeInHierarchy	Defines whether the GameObject is active in the Scene.
activeSelf	The local active state of this GameObject. (Read Only)
isStatic	Gets and sets the GameObject's StaticEditorFlags.
layer	The layer the game object is in.
scene	Scene that the GameObject is part of.
sceneCullingMask	Scene culling mask Unity uses to determine which scene to render the GameObjec
tag	The tag of this game object.
transform	The Transform attached to this GameObject.

Constructors

GameObject	Creates a new game object, named name.

Public Methods

AddComponent	Adds a component class named className to the game object.
BroadcastMessage	Calls the method named methodName on every MonoBehaviour in this game obje
CompareTag	Is this game object tagged with tag ?
GetComponent	Returns the component of Type type if the game object has one attached, null if it
GetComponentInChildren	Returns the component of Type type in the GameObject or any of its children usin
GetComponentInParent	Retrieves the component of Type type in the GameObject or any of its parents.
GetComponents	Returns all components of Type type in the GameObject.

GetComponentsInChildren	Returns all components of Type type in the GameObject or any of its children.
GetComponentsInParent	Returns all components of Type type in the GameObject or any of its parents.
SendMessage	Calls the method named methodName on every MonoBehaviour in this game obje
SendMessageUpwards	Calls the method named methodName on every MonoBehaviour in this game obje
SetActive	Activates/Deactivates the GameObject, depending on the given true or false value.
TryGetComponent	Gets the component of the specified type, if it exists.

Static Methods

CreatePrimitive	Creates a game object with a primitive mesh renderer and appropriate collider.
Find	Finds a GameObject by name and returns it.
FindGameObjectsWithTag	Returns an array of active GameObjects tagged tag. Returns empty array if no Gam
FindWithTag	Returns one active GameObject tagged tag. Returns null if no GameObject was fou

Inherited Members

Properties

hideFlags	Should the object be hidden, saved with the Scene or modifiable by the user?
name	The name of the object.

Public Methods

GetInstanceID	Returns the instance id of the object.
ToString	Returns the name of the object.

Static Methods

Destroy	Removes a GameObject, component or asset.
DestroyImmediate	Destroys the object obj immediately. You are strongly recommended to use Destro
DontDestroyOnLoad	Do not destroy the target Object when loading a new Scene.
FindObjectOfType	Returns the first active loaded object of Type type.
FindObjectsOfType	Returns a list of all active loaded objects of Type type.
Instantiate	Clones the object original and returns the clone.

Operators

bool	Does the object exist?
operator !=	Compares if two objects refer to a different object.
operator ==	Compares two object references to see if they refer to the same object.

}Component.GetComponent

1public Component **GetComponent**(Type **type**);

Parameters

type	The type of Component to retrieve.

Description

Returns the component of Type `type` if the game object has one attached, null if it doesn't.

```
using UnityEngine;

public class ScriptExample : MonoBehaviour
```

```
{

    void Start()

    {

        // Disable the spring on the HingeJoint component.

        HingeJoint hinge = GetComponent<HingeJoint>();

        hinge.useSpring = false;

    }

}
```

public T **GetComponent**();

Description

Generic version of this method.

public Component **GetComponent**(string **type**);

Description

Returns the component with name `type` if the game object has one attached, null if it doesn't.

It is better to use GetComponent with a Type instead of a string for performance reasons. Sometimes you might not be able to get to the type however, for example when trying to access a C# script from Javascript. In that case you can simply access the component by name instead of type. Example:

```
using UnityEngine;

public class ScriptExample : MonoBehaviour


{

    void Start()
```

```
    {

        // Disable the spring on the HingeJoint component.

        HingeJoint hinge = GetComponent("HingeJoint") as HingeJoint;

        hinge.useSpring = false;

    }

}
```

InputDeviceCharacteristics.Controller

Description

The InputDevice is a game controller.

Game Controllers have axes and buttons that can be accessed through UnityEngine.Input.

Undo.IncrementCurrentGroup

public static void **IncrementCurrentGroup**();

Description

Unity automatically groups undo operations by the current group index.

The current group is automatically incremented based on events eg. mouse down events, executing a menu item increments the current group. But sometimes it is necessary to manually group undo operations.

NavMesh

class in UnityEngine.AI

/

Implemented in:UnityEngine.AIModule

Description

Singleton class to access the baked NavMesh.

The NavMesh is a class that can be used to do spatial queries, like pathfinding and walkability tests, set the pathfinding cost for specific area types, and to tweak global behavior of pathfinding and avoidance.

In order to use the spatial queries, you need to first bake NavMesh for your Scene.

See also:
• Building a NavMesh – for more information on how to setup and bake NavMesh
• Areas and Costs – to learn how to use different Area types.
• NavMeshAgent – to learn how to control and move NavMesh Agents.
• NavMeshObstacle – to learn how to control NavMesh Obstacles using scripting.
• OffMeshLink – to learn how to control Off-Mesh Links using scripting.

Static Properties

AllAreas	Area mask constant that includes all NavMesh areas.
avoidancePredictionTime	Describes how far in the future the agents predict collisions for avoidance.
onPreUpdate	Set a function to be called before the NavMesh is updated during the frame update
pathfindingIterationsPerFrame	The maximum amount of nodes processed each frame in the asynchronous pathfin

Static Methods

AddLink	Adds a link to the NavMesh. The link is described by the NavMeshLinkData struct
AddNavMeshData	Adds the specified NavMeshData to the game.
CalculatePath	Calculate a path between two points and store the resulting path.
CalculateTriangulation	Calculates triangulation of the current navmesh.
CreateSettings	Creates and returns a new entry of NavMesh build settings available for runtime N
FindClosestEdge	Locate the closest NavMesh edge from a point on the NavMesh.

GetAreaCost	Gets the cost for path finding over geometry of the area type.
GetAreaFromName	Returns the area index for a named NavMesh area type.
GetSettingsByID	Returns an existing entry of NavMesh build settings.
GetSettingsByIndex	Returns an existing entry of NavMesh build settings by its ordered index.
GetSettingsCount	Returns the number of registered NavMesh build settings.
GetSettingsNameFromID	Returns the name associated with the NavMesh build settings matching the provide
Raycast	Trace a line between two points on the NavMesh.
RemoveAllNavMeshData	Removes all NavMesh surfaces and links from the game.
RemoveLink	Removes a link from the NavMesh.
RemoveNavMeshData	Removes the specified NavMeshDataInstance from the game, making it unavailabl
RemoveSettings	Removes the build settings matching the agent type ID.
SamplePosition	Finds the closest point on NavMesh within specified range.
SetAreaCost	Sets the cost for finding path over geometry of the area type on all agents.

DelegatesCanvas

class in UnityEngine

/

Inherits from:Behaviour

/

Implemented in:UnityEngine.UIModule

Description

Element that can be used for screen rendering.

Elements on a canvas are rendered AFTER Scene rendering, either from an attached camera or using overlay mode.

```
using System.Collections;

using System.Collections.Generic;

using UnityEditor;

using UnityEngine;

using UnityEngine.UI;

// Create a Canvas that holds a Text GameObject.

public class ExampleClass : MonoBehaviour

{

    void Start()

    {

        GameObject myGO;

        GameObject myText;

        Canvas myCanvas;

        Text text;

        RectTransform rectTransform;

        // Canvas

        myGO = new GameObject();
```

```
        myGO.name = "TestCanvas";

        myGO.AddComponent<Canvas>();
        myCanvas = myGO.GetComponent<Canvas>();

        myCanvas.renderMode = RenderMode.ScreenSpaceOverlay;

        myGO.AddComponent<CanvasScaler>();

        myGO.AddComponent<GraphicRaycaster>();
        // Text

        myText = new GameObject();

        myText.transform.parent = myGO.transform;

        myText.name = "wibble";
        text = myText.AddComponent<Text>();

        text.font = (Font)Resources.Load("MyFont");

        text.text = "wobble";

        text.fontSize = 100;
        // Text position

        rectTransform = text.GetComponent<RectTransform>();

        rectTransform.localPosition = new Vector3(0, 0, 0);

        rectTransform.sizeDelta = new Vector2(400, 200);

    }
```

```
}
```

Properties

additionalShaderChannels	Get or set the mask of additional shader channels to be used when creating the Can
cachedSortingLayerValue	Cached calculated value based upon SortingLayerID.
isRootCanvas	Is this the root Canvas?
normalizedSortingGridSize	The normalized grid size that the canvas will split the renderable area into.
overridePixelPerfect	Allows for nested canvases to override pixelPerfect settings inherited from parent c
overrideSorting	Override the sorting of canvas.
pixelPerfect	Force elements in the canvas to be aligned with pixels. Only applies with renderM
pixelRect	Get the render rect for the Canvas.
planeDistance	How far away from the camera is the Canvas generated.
referencePixelsPerUnit	The number of pixels per unit that is considered the default.
renderMode	Is the Canvas in World or Overlay mode?
renderOrder	The render order in which the canvas is being emitted to the Scene. (Read Only)
rootCanvas	Returns the Canvas closest to root, by checking through each parent and returning
scaleFactor	Used to scale the entire canvas, while still making it fit the screen. Only applies wi
sortingLayerID	Unique ID of the Canvas' sorting layer.
sortingLayerName	Name of the Canvas' sorting layer.

sortingOrder	Canvas' order within a sorting layer.
targetDisplay	For Overlay mode, display index on which the UI canvas will appear.
worldCamera	Camera used for sizing the Canvas when in Screen Space - Camera. Also used as t

Static Methods

ForceUpdateCanvases	Force all canvases to update their content.
GetDefaultCanvasMaterial	Returns the default material that can be used for rendering normal elements on the
GetETC1SupportedCanvasMaterial	Gets or generates the ETC1 Material.

EventsGameObject

class in UnityEngine

/

Inherits from:Object

/

Implemented in:UnityEngine.CoreModule

Description

Base class for all entities in Unity Scenes.

Note: Many variables in the GameObject class have been removed. To access, for example GameObject.renderer in csharp use GetComponent<Renderer>() instead.

See Also: Component.

Properties

activeInHierarchy	Defines whether the GameObject is active in the Scene.
activeSelf	The local active state of this GameObject. (Read Only)

isStatic	Gets and sets the GameObject's StaticEditorFlags.
layer	The layer the game object is in.
scene	Scene that the GameObject is part of.
sceneCullingMask	Scene culling mask Unity uses to determine which scene to render the GameObjec
tag	The tag of this game object.
transform	The Transform attached to this GameObject.

Constructors

GameObject	Creates a new game object, named name.

Public Methods

AddComponent	Adds a component class named className to the game object.
BroadcastMessage	Calls the method named methodName on every MonoBehaviour in this game obje
CompareTag	Is this game object tagged with tag ?
GetComponent	Returns the component of Type type if the game object has one attached, null if it
GetComponentInChildren	Returns the component of Type type in the GameObject or any of its children usin
GetComponentInParent	Retrieves the component of Type type in the GameObject or any of its parents.
GetComponents	Returns all components of Type type in the GameObject.
GetComponentsInChildren	Returns all components of Type type in the GameObject or any of its children.
GetComponentsInParent	Returns all components of Type type in the GameObject or any of its parents.

SendMessage	Calls the method named methodName on every MonoBehaviour in this game obje
SendMessageUpwards	Calls the method named methodName on every MonoBehaviour in this game obje
SetActive	Activates/Deactivates the GameObject, depending on the given true or false value.
TryGetComponent	Gets the component of the specified type, if it exists.

Static Methods

CreatePrimitive	Creates a game object with a primitive mesh renderer and appropriate collider.
Find	Finds a GameObject by name and returns it.
FindGameObjectsWithTag	Returns an array of active GameObjects tagged tag. Returns empty array if no Gam
FindWithTag	Returns one active GameObject tagged tag. Returns null if no GameObject was fou

Inherited Members

Properties

hideFlags	Should the object be hidden, saved with the Scene or modifiable by the user?
name	The name of the object.

Public Methods

GetInstanceID	Returns the instance id of the object.
ToString	Returns the name of the object.

Static Methods

Destroy	Removes a GameObject, component or asset.
DestroyImmediate	Destroys the object obj immediately. You are strongly recommended to use Destro

DontDestroyOnLoad	Do not destroy the target Object when loading a new Scene.
FindObjectOfType	Returns the first active loaded object of Type type.
FindObjectsOfType	Returns a list of all active loaded objects of Type type.
Instantiate	Clones the object original and returns the clone.

Operators

bool	Does the object exist?
operator !=	Compares if two objects refer to a different object.
operator ==	Compares two object references to see if they refer to the same object.

willRenderCanvases	Event that is called just before Canvas rendering happens.

Inherited Members

Properties

enabled	Enabled Behaviours are Updated, disabled Behaviours are not.
isActiveAndEnabled	Has the Behaviour had active and enabled called?
gameObject	The game object this component is attached to. A component is always attached to
tag	The tag of this game object.
transform	The Transform attached to this GameObject.
hideFlags	Should the object be hidden, saved with the Scene or modifiable by the user?

name	The name of the object.

Public Methods

BroadcastMessage	Calls the method named methodName on every MonoBehaviour in this game obje
CompareTag	Is this game object tagged with tag ?
GetComponent	Returns the component of Type type if the game object has one attached, null if it
GetComponentInChildren	Returns the component of Type type in the GameObject or any of its children usin
GetComponentInParent	Returns the component of Type type in the GameObject or any of its parents.
GetComponents	Returns all components of Type type in the GameObject.
GetComponentsInChildren	Returns all components of Type type in the GameObject or any of its children.
GetComponentsInParent	Returns all components of Type type in the GameObject or any of its parents.
SendMessage	Calls the method named methodName on every MonoBehaviour in this game obje
SendMessageUpwards	Calls the method named methodName on every MonoBehaviour in this game obje
TryGetComponent	Gets the component of the specified type, if it exists.
GetInstanceID	Returns the instance id of the object.
ToString	Returns the name of the object.

Static Methods

Destroy	Removes a GameObject, component or asset.
DestroyImmediate	Destroys the object obj immediately. You are strongly recommended to use Destro

DontDestroyOnLoad	Do not destroy the target Object when loading a new Scene.
FindObjectOfType	Returns the first active loaded object of Type type.
FindObjectsOfType	Returns a list of all active loaded objects of Type type.
Instantiate	Clones the object original and returns the clone.

Operators

bool	Does the object exist?
operator !=	Compares if two objects refer to a different object.
operator ==	Compares two object references to see if they refer to the same object.

OnNavMeshPreUpdate	A delegate which can be used to register callback methods to be invoked before th

SerializedPropertyType.Enum

Description

Enumeration property.

See Also: SerializedProperty.

GameObject.GetComponent

public Component **GetComponent**(Type **type**);

Parameters

type	The type of Component to retrieve.

Description

Returns the component of Type `type` if the game object has one attached, null if it doesn't.

Using gameObject.GetComponent will return the first component that is found and the order is undefined. If you expect there to be more than one component of the same type, use gameObject.GetComponents instead, and cycle through the returned components testing for some unique property.

```
using UnityEngine;

public class GetComponentExample : MonoBehaviour

{

    void Start()

    {

        HingeJoint hinge = gameObject.GetComponent(typeof(HingeJoint)) as
HingeJoint;

        if (hinge != null)

            hinge.useSpring = false;

    }

}
```

public T **GetComponent**();

Description

Generic version of this method.

```
using UnityEngine;

public class GetComponentGenericExample : MonoBehaviour

{

    void Start()
```

```
    {

        HingeJoint hinge = gameObject.GetComponent<HingeJoint>();

        if (hinge != null)

            hinge.useSpring = false;

    }

}
```

public Component **GetComponent**(string **type**);

Parameters

type	The type of Component to retrieve.

Description

Returns the component with name `type` if the game object has one attached, null if it doesn't.

It is better to use GetComponent with a Type instead of a string for performance reasons. Sometimes you might not be able to get to the type however, for example when trying to access a C# script from Javascript. In that case you can simply access the component by name instead of type.

```
using UnityEngine;

public class GetComponentNonPerformantExample : MonoBehaviour

{

    void Start()

    {

        HingeJoint hinge = gameObject.GetComponent("HingeJoint") as
HingeJoint;
```

```
        if (hinge != null)

            hinge.useSpring = false;

    }

}
```