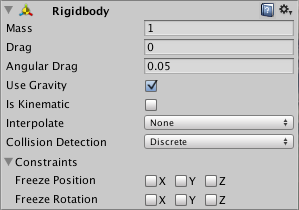
# 刚体

**刚体 (Rigidbody)**使\_\_游戏对象\_\_的行为方式受物理控制。刚体可以接受力和扭矩，使对象以逼真的方式移动。任何游戏对象都必须包含受重力影响的刚体，行为方式基于施加的作用力（通过脚本），或通过 NVIDIA PhysX 物理引擎与其他对象交互。



## 属性

| **属性：** | **功能：** |
| --- | --- |
| **Mass** | 对象的质量（默认为千克）。 |
| **Drag** | 根据力移动对象时影响对象的空气阻力大小。0 表示没有空气阻力，无穷大使对象立即停止移动。 |
| **Angular Drag** | 根据扭矩旋转对象时影响对象的空气阻力大小。0 表示没有空气阻力。请注意，如果直接将对象的 Angular Drag 属性设置为无穷大，无法使对象停止旋转。 |
| **Use Gravity** | 如果启用此属性，则对象将受重力影响。 |
| **Is Kinematic** | 如果启用此选项，则对象将不会被物理引擎驱动，只能通过\_\_变换 (Transform)\_\_ 对其进行操作。对于移动平台，或者如果要动画化附加了 **HingeJoint** 的刚体，此属性将非常有用。 |
| **Interpolate** | 仅当在刚体运动中看到急动时才尝试使用提供的选项之一。 |
| - **None** | 不应用插值。 |
| - **Interpolate** | 根据前一帧的变换来平滑变换。 |
| - **Extrapolate** | 根据下一帧的估计变换来平滑变换。 |
| **Collision Detection** | 用于防止快速移动的对象穿过其他对象而不检测碰撞。 |
| - **Discrete** | Use discrete collision detection against all other Colliders in the Scene. Other colliders will use discrete collision detection when testing for collision against it. Used for normal collisions (This is the default value). |
| - **Continuous** | Use Discrete collision detection against dynamic Colliders (with a Rigidbody) and sweep-based continuous collision detection against static Colliders (without a Rigidbody). Rigidbodies set to **Continuous Dynamic** will use continuous collision detection when testing for collision against this rigidbody. Other rigidbodies will use Discrete Collision detection. Used for objects which the **Continuous Dynamic**detection needs to collide with. (This has a big impact on physics performance, leave it set to **Discrete**, if you don’t have issues with collisions of fast objects) |
| - **Continuous Dynamic** | Use sweep-based continuous collision detection against GameOjects set to **Continuous** and **Continuous Dynamic** collision. It will also use continuous collision detection against static Colliders (without a Rigidbody). For all other colliders, it uses discrete collision detection. Used for fast moving objects. |
| **Constraints** | 对刚体运动的限制： |
| - **Freeze Position** | 有选择地停止刚体沿世界 X、Y 和 Z 轴的移动。 |
| - **Freeze Rotation** | 有选择地停止刚体围绕局部 X、Y 和 Z 轴旋转。 |

## 详细信息

刚体可使游戏对象的行为方式受物理引擎控制。由此可以实现逼真碰撞和不同类型关节等行为。通过向刚体施力来操纵游戏对象产生的效果完全不同于直接调整变换\_\_组件\_\_。通常，不应同时操纵同一个游戏对象的刚体和变换 - 一次只能操纵其中一个。

操纵变换与刚体的最大区别在于力的运用。刚体可以接受力和扭矩，但变换不能。变换可以被平移和旋转，但这与使用物理引擎不同。亲自尝试使用二者会注意到明显的差异。向刚体施加力/扭矩实际上会改变对象的变换组件位置和旋转。这就是为何应该只使用其中一个组件的原因。使用物理引擎时更改变换可能会导致碰撞和其他计算问题。

必须先将刚体显式添加到游戏对象中，然后才能让刚体受到物理引擎的影响。可以从菜单中的 **Components > Physics > Rigidbody** 向所选对象添加刚体。现在对象已经完成物理设置；它会受到重力的影响而掉落，而且可以通过脚本来接受作用力，但可能需要添加一个\_\_碰撞体\_\_或关节来让对象按照期望的方式运行。

### 父子化

当对象处于物理控制下时，对象的移动方式在一定程度上独立于其变换父对象的移动方式。如果移动任何父对象，它们会将刚体子对象拉到自己身边。但是，刚体仍然会因重力而降落，并会对碰撞事件作出反应。

### 脚本

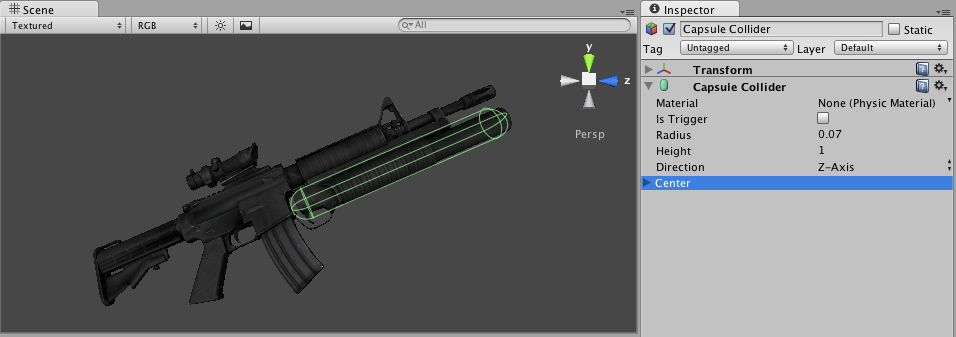
要控制刚体，主要需要使用脚本来施加力或扭矩。要执行此操作，可在对象的刚体上调用 **[AddForce()](file:///E:\\UnityDocumentation\\ScriptReference\\Rigidbody.AddForce.html)** 和 **[AddTorque()](file:///E:\\UnityDocumentation\\ScriptReference\\Rigidbody.AddTorque.html)**。请记住，在使用物理引擎时，不应直接改变对象的变换。

### 动画

在某些情况下，主要需要创建布娃娃效果，因此有必要在动画和物理之间切换对象的控制。为此，可将刚体标记为 **[isKinematic](file:///E:\\UnityDocumentation\\ScriptReference\\Rigidbody-isKinematic.html)**。虽然刚体标记为 \_\_isKinematic\_\_，但不会受到碰撞、力或物理系统任何其他部分的影响。这意味着，必须通过直接操作[变换](file:///E:\UnityDocumentation\Manual\class-Transform.html)组件来控制对象。运动刚体将影响其他对象，但它们自身不会受到物理影响。例如，附加到运动对象的关节将约束附加到关节的所有其他刚体，并且运动刚体将通过碰撞影响其他刚体。

### 碰撞体

碰撞体是另一种必须与刚体一起添加才会实现碰撞的组件。如果两个刚体相互碰撞，只有两个对象都附加了碰撞体时，物理引擎才会计算碰撞。在物理模拟过程中，无碰撞体的刚体将直接相互穿过对方。

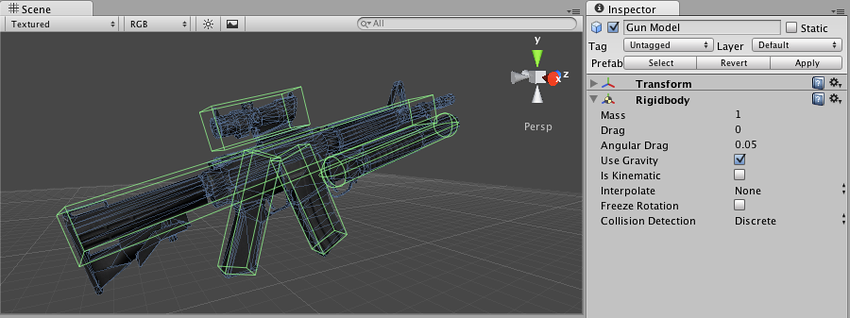
碰撞体定义刚体的物理边界

可通过 **Component > Physics** 菜单添加碰撞体。请查看单个碰撞体的组件参考页面以获取更多具体信息：

* [盒型碰撞体](file:///E:\UnityDocumentation\Manual\class-BoxCollider.html) - 立方体的原始形状
* [球形碰撞体](file:///E:\UnityDocumentation\Manual\class-SphereCollider.html) - 球体的原始形状
* [胶囊碰撞体](file:///E:\UnityDocumentation\Manual\class-CapsuleCollider.html) - 胶囊体的原始形状
* [网格碰撞体](file:///E:\UnityDocumentation\Manual\class-MeshCollider.html) - 根据对象网格创建碰撞体，不能与另一个网格碰撞体碰撞
* [车轮碰撞体](file:///E:\UnityDocumentation\Manual\class-WheelCollider.html) - 专门用于创建汽车或其他移动车辆
* [地形碰撞体](file:///E:\UnityDocumentation\Manual\class-TerrainCollider.html) - 处理与 Unity 地形系统的碰撞

### 复合碰撞体

复合碰撞体是原始碰撞体的组合，共同作为单个碰撞体。如果模型在性能方面过于复杂或成本过高而无法精确模拟，因此希望使用简单的近似值以最佳方式模拟形状的碰撞，复合碰撞体将派上用场。要创建复合碰撞体，请创建碰撞对象的子对象，然后将碰撞体组件添加到每个子对象。这样就能轻松独立定位、旋转和缩放每个碰撞体。可以通过许多原始碰撞体和/或凸面网格碰撞体构建复合碰撞体。

真实复合碰撞体设置

在上图中，游戏对象 Gun Model 附加了一个刚体，使用多个原始碰撞体作为子游戏对象。当刚体父对象由于受力而移动时，子碰撞体会随之移动。原始碰撞体将与环境的网格碰撞体碰撞，而父刚体将根据所受的力以及其子碰撞体与场景中其他碰撞体的交互方式来改变自己的移动方式。

网格碰撞体通常不能相互碰撞。如果网格碰撞体被标记为\_\_凸面体 (Convex)\_\_，则可与另一个网格碰撞体发生碰撞。典型的解决方案是对所有移动的对象使用原始碰撞体，而对静态背景对象使用网格碰撞体。

### 连续碰撞检测

连续碰撞检测是一种阻止快速移动的碰撞体相互穿过的功能。使用正常 (**Discrete**) 碰撞检测时，如果对象在一个帧中位于某个碰撞体的一侧，而在下一帧中已经穿过了碰撞体，便属于彼此穿过的情况。要解决此问题，可在快速移动对象的刚体上启用连续碰撞检测。将碰撞检测模式设置为 **Continuous** 可防止刚体穿过任何静态（即非刚体）网格碰撞体。设置为 **Continuous Dynamic** 也会防止刚体穿过任何其他支持的刚体（即，碰撞检测模式设置为 **Continuous** 或 **Continuous Dynamic** 的刚体）。 盒型碰撞体、球形碰撞体和胶囊碰撞体支持连续碰撞检测。请注意，连续碰撞检测的目的是作为一种安全机制，可在对象会相互穿过的情况下捕获碰撞，但不会提供精确物理碰撞的结果；所以如果遇到快速移动对象的问题，仍然可以考虑在 TimeManager 检视面板中降低固定时间步长值以使模拟更准确。

## 使用合适的大小

游戏对象网格的大小比刚体的质量重要得多。如果发现刚体的行为与预期不符，比如：移动缓慢、漂浮或碰撞方式不正确，请考虑调整网格资源的缩放比例。Unity 的默认单位比例为 1 个单位等于 1 米，因此会维持导入网格的比例并应用于物理计算。例如，摇摇欲坠的摩天大楼在坍塌时与由玩具积木制成的塔楼完全不同，因此不同大小的对象应按照精确比例建模。

如果要对人类进行建模，请确保该模型在 Unity 中的高度约为 2 米。要检查对象是否具有适当的大小，请将其与默认立方体进行比较。可以使用 **GameObject > 3D Object > Cube** 来创建立方体。立方体的高度恰好是 1 米，所以人的高度应该是其两倍。

如果无法调整网格本身，可以更改特定网格资源的统一比例，操作方法是在 **Project 视图\_\_中选择该网格资源，然后从菜单中选择**Assets > Import Settings…\_\_。在此处，可以更改比例并重新导入网格。

如果游戏需要在不同比例下实例化游戏对象，则可以调整变换的缩放轴的值。缺点是物理模拟必须在对象实例化时执行更多工作，并且可能导致游戏时性能下降。这倒不是可怕的损失，但此方法不如使用其他两个选项来最终确定比例那么高效。另外请注意，使用父子化 (Parenting) 时，不统一的比例会产生意外行为。出于这些原因，最好在建模应用程序中以正确的比例创建对象。

## 提示

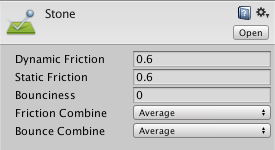
* 两个刚体的相对\_\_质量\_\_决定了它们相互碰撞时的反应情况。
* 使一个刚体具有比另一个刚体更大的\_\_质量\_\_并不会使其在自由落体时降落得更快。要实现这一目的，请使用 **Drag**。
* 较低的 **Drag** 值会让对象看起来较重。较高的值会使其看起来较轻。**Drag** 的典型值为 .001（实心金属块）到 10（羽毛）之间。
* 如果要直接操作对象的变换组件，但仍希望获得物理效果，请附加刚体并将其设为 Kinematic。
* 如果要通过变换组件移动游戏对象，但希望接收 Collision/Trigger 消息，必须将刚体附加到正在移动的对象。
* 如果直接将对象的 Angular Drag 属性设置为无穷大，无法使对象停止旋转。

# 物理材质

**物理材质 (Physic Material)** 用于调整碰撞对象的摩擦力和反弹效果。

要创建物理材质，请从菜单栏中选择 **Assets > Create > Physic Material**。然后将物理材质从 Project 视图拖入到场景中的\_\_碰撞体\_\_。

## 属性



| **属性：** | **功能：** |
| --- | --- |
| **Dynamic Friction** | 已在移动时使用的摩擦力。通常为 0 到 1 之间的值。值为零就像冰一样，值为 1 将使对象迅速静止（除非用很大的力或重力推动对象）。 |
| **Static Friction** | 当对象静止在表面上时使用的摩擦力。通常为 0 到 1 之间的值。值为零就像冰一样，值为 1 将导致很难让对象移动。 |
| **Bounciness** | 表面的弹性如何？值为 0 将不会反弹。值为 1 将在反弹时不产生任何能量损失，预计会有一些近似值，但可能只会给模拟增加少量能量。 |
| **Friction Combine** | 两个碰撞对象的摩擦力的组合方式。 |
| - **Average** | 对两个摩擦值求平均值。 |
| - **Minimum** | 使用两个值中的最小值。 |
| - **Maximum** | 使用两个值中的最大值。 |
| - **Multiply** | 两个摩擦值相乘。 |
| **Bounce Combine** | 两个碰撞对象的弹性的组合方式。其模式与 Friction Combine 模式相同 |

## 详细信息

摩擦力是防止表面相互滑落的量。尝试堆叠对象时，此值很重要。摩擦力有两种形式：动态和静态。对象静止时使用\_\_静态摩擦力**。静态摩擦力会阻止对象开始移动。如果向对象施加足够大的力，对象将开始移动。随后，\_\_动态摩擦力\_\_将发挥作用。**动态摩擦力\_\_现在将尝试在与另一个对象接触时减慢对象的速度。

当两个对象接触时，根据所选择的模式对它们两者施加相同的弹性和摩擦力效果。当两个接触的碰撞体具有不同的组合模式设置时，存在一种特殊情况。在这种特殊情况下，使用具有最高优先级的函数。优先级顺序如下：\_\_Average\_\_ < **Minimum** < **Multiply** < **Maximum**。例如，如果一种材质设置了 **Average\_\_，但另一种材质设置了**Maximum\_\_，那么要使用的组合函数是 \_\_Maximum\_\_，因为它具有更高的优先级。

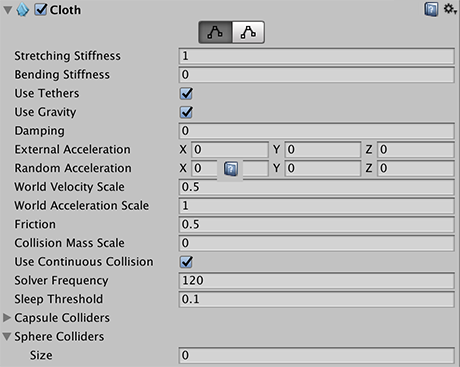
请注意，Nvidia PhysX 引擎使用的摩擦力模型针对模拟的性能和稳定性进行了调整，并不一定代表真实物理的高度近似值。具体而言，大于单个点的接触面（例如两个相互叠放在一起的盒子）将计算为具有两个接触点，其摩擦力将是现实世界物理学中的两倍。在这种情况下，可能希望将摩擦系数乘以 0.5 以获得更真实的结果。

同样的逻辑适用于弹性模型。由于各种模拟细节（如位置校正），Nvidia PhysX 无法保证完美的能量守恒效果。因此，比如说，如果受重力影响的对象的弹性值为 1，并与弹性为 1 的地面碰撞，则对象的弹跳高度应该会高于初始位置。

# 布料

布料 (Cloth) 组件与带蒙皮的网格渲染器 (Skinned Mesh Renderer) 协同工作，从而提供基于物理的面料模拟解决方案。此组件是专为角色服装设计的，仅对蒙皮网格有效。如果向非蒙皮网格中添加 Cloth 组件，则 Unity 会删除非蒙皮网格并添加蒙皮网格。

要将 Cloth 组件附加到蒙皮网格，请在 Editor 中选择游戏对象，在 Inspector 窗口中单击 **Add Component** 按钮，然后选择 **Physics** > **Cloth**。Inspector 中将显示该组件。



## 属性

| **属性：** | **功能：** |
| --- | --- |
| **Stretching Stiffness** | 布料的拉伸刚度。 |
| **Bending Stiffness** | 布料的弯曲刚度。 |
| **Use Tethers** | 施加约束以帮助防止移动的布料粒子离开固定粒子的距离太远。此属性有助于减少过度拉伸。 |
| **Use Gravity** | 是否应该对布料施加重力加速度？ |
| **Damping** | 运动阻尼系数。 |
| **External Acceleration** | 施加在布料上的恒定外部加速度。 |
| **Random Acceleration** | 施加在布料上的随机外部加速度。 |
| **World Velocity Scale** | 角色多大程度的世界空间移动会影响布料顶点。 |
| **World Acceleration Scale** | 角色多大的世界空间加速度会影响布料顶点。 |
| **Friction** | 布料与角色碰撞时的摩擦力。 |
| **Collision Mass Scale** | 碰撞粒子的质量增加量。 |
| **Use Continuous Collision** | 启用连续碰撞来提高碰撞稳定性。 |
| **Use Virtual Particles** | 每个三角形添加一个虚拟粒子，从而提高碰撞稳定性。 |
| **Solver Frequency** | 解算器每秒迭代次数。 |
| **Sleep Threshold** | 布料的睡眠阈值。 |
| **Capsule Colliders** | 应与此 Cloth 实例碰撞的 CapsuleCollider 的数组。 |
| **Sphere Colliders** | 应与此 Cloth 实例碰撞的 ClothSphereColliderPairs 的数组。 |

## 详细信息

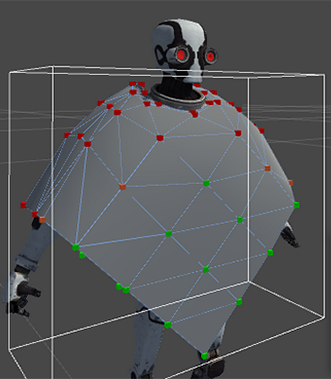
布料不响应场景中的任何碰撞体，也不会将力反射回世界。添加 Cloth 组件后，该组件完全不会响应和影响任何其他实体。因此在手动将碰撞体从世界添加到 Cloth 组件之前，布料和世界无法识别或看到彼此。即使执行了此操作，模拟仍是单向的：布料可以响应实体，但不会反向施力。

此外，只能对布料使用三种类型的碰撞体：球体、胶囊体以及使用两个球形碰撞体构造而成的圆锥胶囊碰撞体。之所以存在这么多限制是为了帮助提高性能。

### 编辑约束工具

选择 **Edit** > **Constraints** 可编辑对布料网格中每个顶点施加的约束。所有顶点都具有基于当前可视化模式的颜色，目的是显示它们各自值之间的差异。通过使用画笔在布料上绘制即可编辑布料约束。

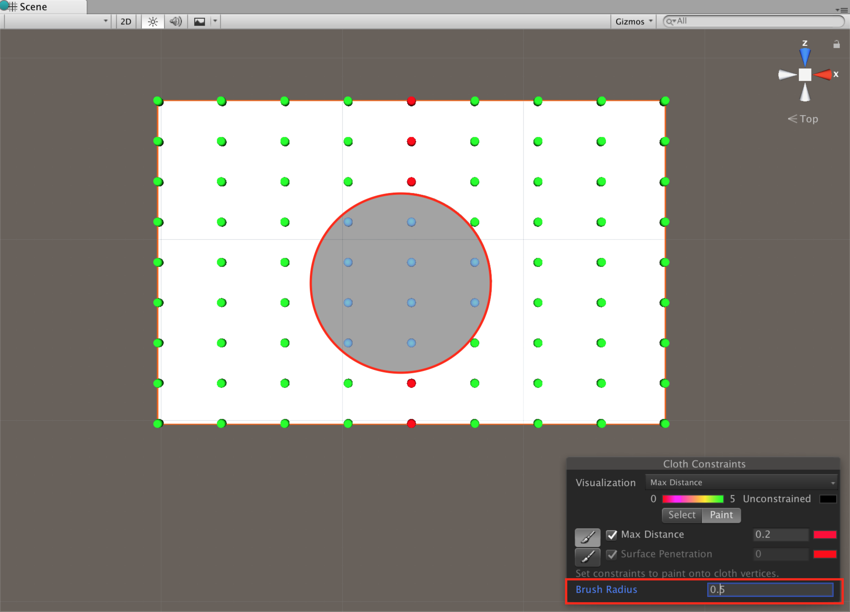
| **属性：** | **功能：** |
| --- | --- |
| **Visualization** | 在 Scene 视图中的 Max Distance 属性值和 Surface Penetration 属性值之间切换工具的视觉外观。此外，还提供了用于操纵背面的开关。 |
| **Max Distance** | 布料粒子可从顶点位置行进的最大距离。 |
| **Surface Penetration** | 布料粒子可穿透网格的深度。 |
| **Brush Radius** | 设置用于在布料上绘制约束的画笔的半径。 |

在带蒙皮的网格渲染器上使用的布料约束工具。

更改每个顶点的值时可使用两种模式：

* 使用\_\_选择\_\_模式可选择一组顶点。要这样做，请使用鼠标光标来绘制一个选框，或一次单击一个顶点。然后，即可启用 **Max Distance** 和/或 \_\_Surface Penetration\_\_，并可设置一个值。
* 使用\_\_绘画\_\_模式可直接调整每个顶点。要这样做，请单击要调整的顶点。然后，即可启用 **Max Distance** 和/或 \_\_Surface Penetration\_\_，并可设置一个值。

在这两种模式下，为 **Max Distance** 和 **Surface Penetration** 赋值时，Scene 视图中的可视化表示都会自动更新。

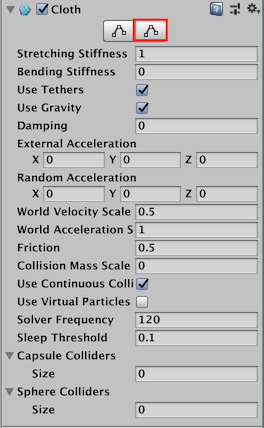
绘画模式下的布料约束工具

### 自碰撞和互碰撞

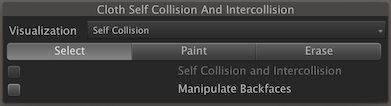
布料碰撞使游戏中的角色服装和其他面料更加逼真。在 Unity 中，布料有几种处理碰撞的布料粒子。可将布料粒子设置为：

* 自碰撞，防止布料穿透自身。
* 互碰撞，允许布料粒子相互碰撞。

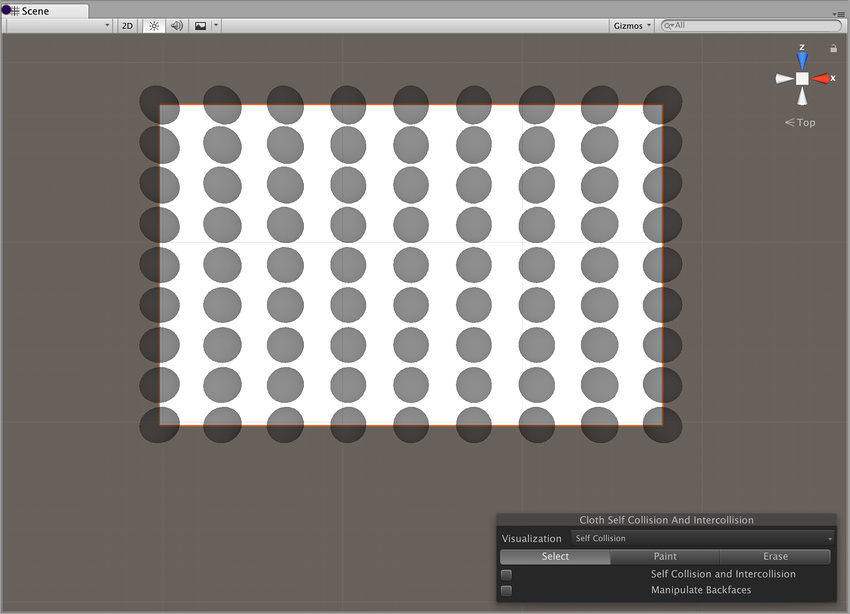
要为布料设置碰撞粒子，请在 Cloth Inspector 中选择 **Self Collision and Intercollision** 按钮：

Cloth Inspector 中的 Self Collision and Intercollision 按钮

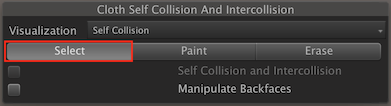
此时将在 Scene 视图中显示 **Cloth Self Collision And Intercollision** 窗口：

Cloth Self Collision And Intercollision 窗口

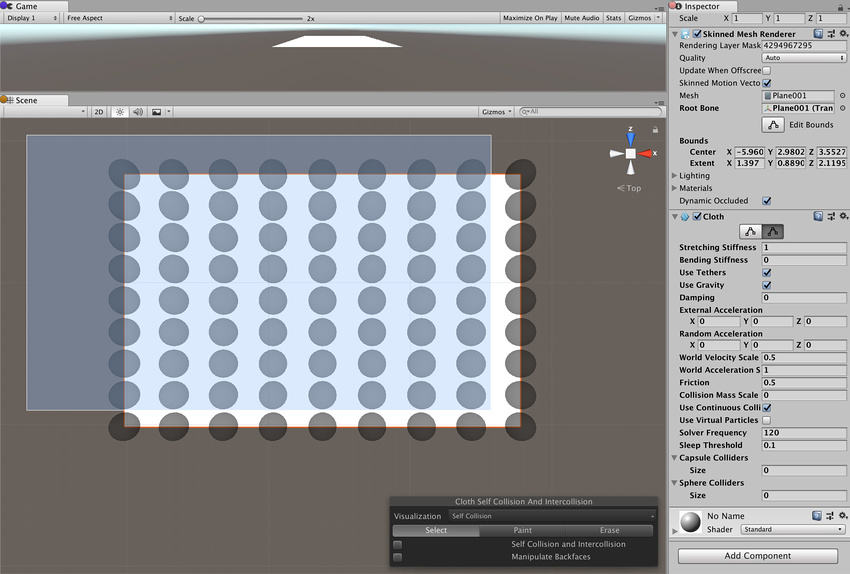
包含 Cloth 组件的蒙皮网格将自动显示布料粒子。最初没有任何布料粒子设置为使用碰撞。这些未使用的粒子显示为黑色：

未使用的布料粒子

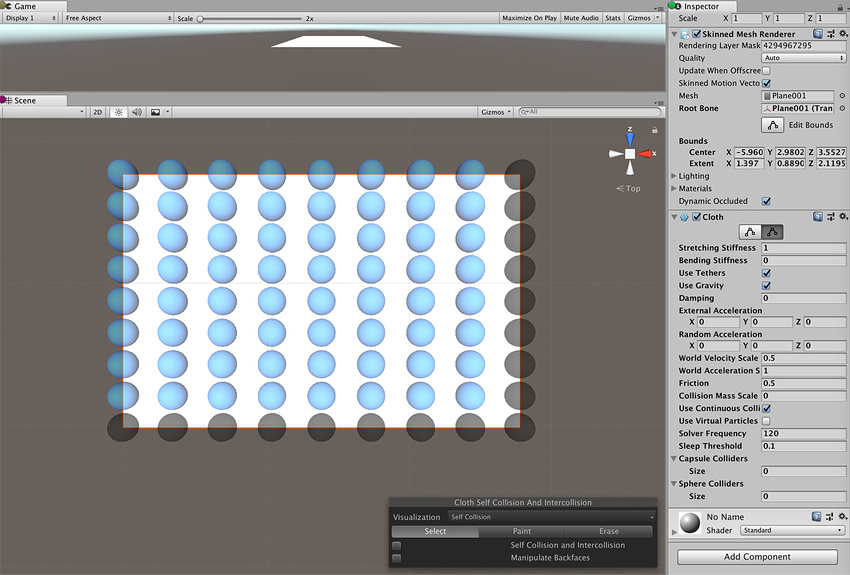
要应用自碰撞或互碰撞，必须选择一组粒子来应用碰撞。要选择一组粒子进行碰撞，请单击 **Select** 按钮：

Select 布料粒子按钮

现在左键单击并拖动以便选择要应用碰撞的粒子：

使用单击并拖动的方式选择布料粒子

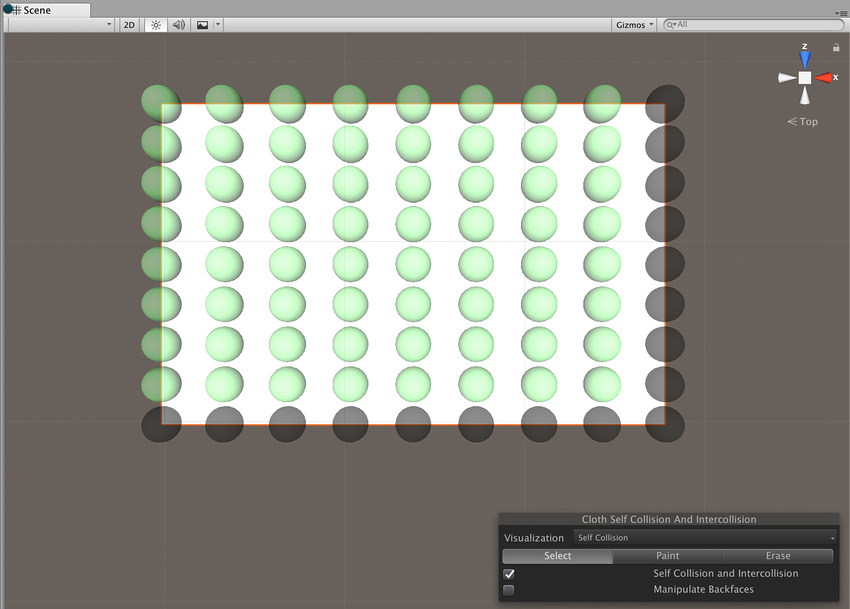
选定粒子显示为蓝色：

选定的布料粒子为蓝色

勾选 **Self Collision and Intercollision** 复选框可将碰撞应用于选定粒子：

Self Collision and Intercollision 复选框

指定用于碰撞的粒子将显示为绿色：

选定粒子为绿色

要为布料启用自碰撞行为，请选择 Cloth Inspector 窗口的 **Self Collision** 部分，并将 **Distance** 和 **Stiffness** 设置为非零值：

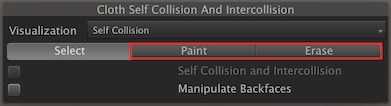
Self Collision 属性的参数

| **属性：** | **功能：** |
| --- | --- |
| **Distance** | 每个粒子的包裹球体的直径。Unity 确保这些球体在模拟过程中不会重叠。**Distance** 属性的值应小于配置中的两个粒子之间的最小距离。如果距离较大，则自碰撞可能违反某些距离约束并导致抖动。 |
| **Stiffness** | 粒子之间的分离冲力的强度。此值由布料解算器进行计算，应足以保持粒子分离。 |

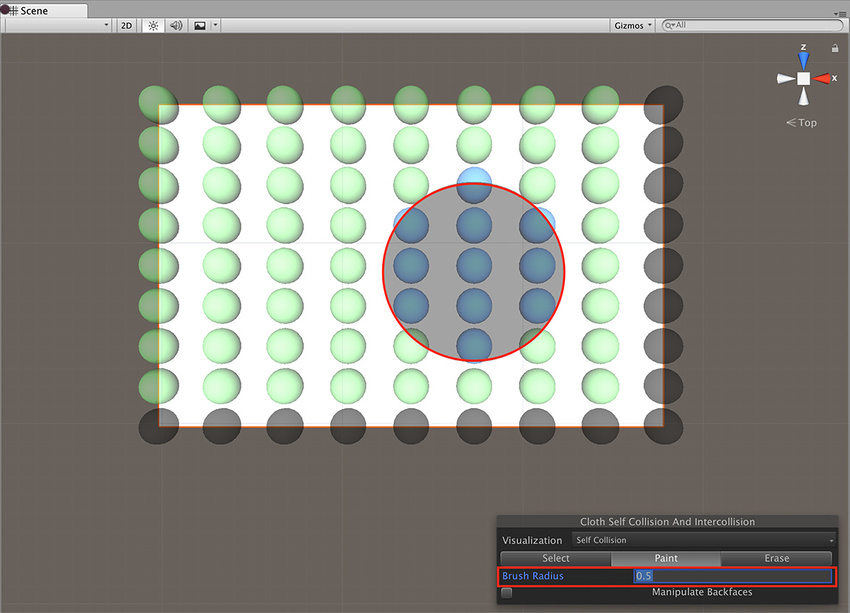
自碰撞和互碰撞可能需要大量的总模拟时间。请考虑缩小碰撞距离并使用自碰撞索引来减少彼此碰撞的粒子数。

自碰撞使用顶点，而不是三角形，因此对于三角形远大于布料厚度的网格，不要指望碰撞能够完美运行。

**Paint** 和 **Erase** 模式允许通过按住鼠标左键并拖动单个布料粒子来添加或删除用于碰撞的粒子：

Self Collision 参数

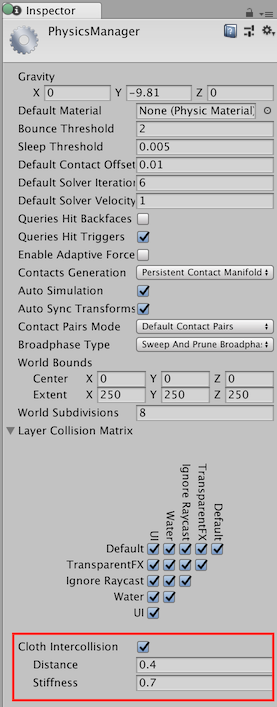
在 **Paint** 或 **Erase** 模式下，指定用于碰撞的粒子为绿色，未指定的粒子为黑色，画笔下方的粒子为蓝色：

正在绘制的粒子为蓝色

#### 布料互碰撞

可使用与指定自碰撞粒子相同的方式（如上所述）为互碰撞指定粒子。与自碰撞一样，可以指定一组用于互碰撞的粒子。

要启用互碰撞行为，请打开 PhysicsManager Inspector (**Edit** > **Project Settings** > **Physics**)，并在 **Cloth InterCollision** 部分将 **Distance** 和 **Stiffness** 设置为非零值：

正在绘制的粒子为蓝色

布料互碰撞的 **Distance** 和 **Stiffness** 属性与以上描述的自碰撞的 Distance 和 Stiffness 属性功能相同。

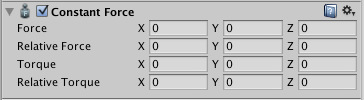
### 碰撞体碰撞

布料无法直接与任意世界几何体碰撞，现在只会与[胶囊碰撞体](file:///E:\UnityDocumentation\ScriptReference\Cloth-capsuleColliders.html)或[球形碰撞体](file:///E:\UnityDocumentation\ScriptReference\Cloth-sphereColliders.html)数组中指定的碰撞体相互作用。

球形碰撞体数组可以包含单个有效的 [SphereCollider](file:///E:\\UnityDocumentation\\ScriptReference\\SphereCollider.html) 实例（第二个为 null）或一对实例。在前一种情况下，[ClothSphereColliderPair](file:///E:\UnityDocumentation\ScriptReference\ClothSphereColliderPair.html) 仅表示待碰撞布料的单个球形碰撞体。在后一种情况下，它表示由两个球体以及连接这两个球体的锥体所定义的圆锥形胶囊形状。圆锥形胶囊形状在对角色的四肢进行建模时非常有用。

# 恒定力

**恒定力 (Constant Force)** 可用于快速向\_\_刚体\_\_添加恒定力。如果不希望某些一次性对象以较大的速度开始而是逐渐加速（比如火箭），则很适合使用恒定力。



## 属性

| **属性：** | **功能：** |
| --- | --- |
| **Force** | 要在世界空间中应用的力的矢量。 |
| **Relative Force** | 要在对象的局部空间中应用的力的矢量。 |
| **Torque** | 在世界空间中应用的扭矩的矢量。对象将开始\_围绕\_此矢量旋转。矢量越长，旋转越快。 |
| **Relative Torque** | 在局部空间中应用的扭矩的矢量。对象将开始\_围绕\_此矢量旋转。矢量越长，旋转越快。 |

### 详细信息

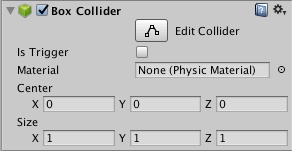
要制作一个向前加速的火箭，请将 **Relative Force** 设定为沿正 z 轴。然后，使用刚体的 **Drag** 属性使其不超过某个最大速度（阻力越高，最大速度越低）。在刚体中，还要确保关闭重力，以便火箭始终保持在其路径上。

## 提示

* 要使对象向上运动，请添加具有正 Y 值 **Force** 属性的恒定力。
* 要使对象向前飞行，请添加具有正 Z 值 **Relative Force** 属性的恒定力。

**盒型碰撞体**

**盒型碰撞体 (Box Collider)** 是一种基本的立方体形状原始碰撞体。



**属性**

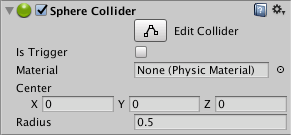
| ***属性：*** | ***功能：*** |
| --- | --- |
| **Is Trigger** | 如果启用此属性，则该碰撞体将用于触发事件，并被物理引擎忽略。 |
| **Material** | 引用[物理材质](file:///E:\\UnityDocumentation\\Manual\\class-PhysicMaterial.html)，可确定该碰撞体与其他对象的交互方式。 |
| **Center** | 碰撞体在对象局部空间中的位置。 |
| **Size** | 碰撞体在 X、Y、Z 方向上的大小。 |

**详细信息**

盒型碰撞体显然可用于形状大致类似于盒体的任何东西，例如板条箱或木箱。但是，可以使用薄形盒体作为地板、墙壁或坡道。盒体形状也是复合碰撞体中的有用元素。

# 球形碰撞体

**球形碰撞体 (Sphere Collider)** 是一种基本的球体形状原始碰撞体。



## 属性

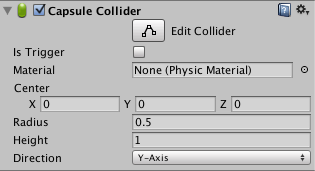
| **属性：** | **功能：** |
| --- | --- |
| **Is Trigger** | 如果启用此属性，则该碰撞体将用于触发事件，并被物理引擎忽略。 |
| **Material** | 引用[物理材质](file:///E:\UnityDocumentation\Manual\class-PhysicMaterial.html)，可确定该碰撞体与其他对象的交互方式。 |
| **Center** | 碰撞体在对象局部空间中的位置。 |
| **Radius** | 碰撞体的大小。 |

## 详细信息

可以通过 **Radius** 属性调整碰撞体的大小，但不能单独沿三个轴缩放（即，不能将球体展平为椭圆）。除了网球等球形对象的明显用途外，球体也适用于坠落的巨石和其他需要翻滚的对象。

**胶囊碰撞体**

**胶囊碰撞体 (Capsule Collider)** 由两个半球与一个圆柱体连接在一起组成。胶囊碰撞体与胶囊原始碰撞体的形状相同。



**属性**

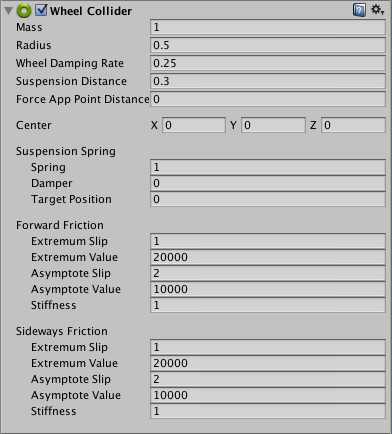
| ***属性：*** | ***功能：*** |
| --- | --- |
| **Is Trigger** | 如果启用此属性，则该碰撞体将用于触发事件，并被物理引擎忽略。 |
| **Material** | 引用[物理材质](file:///E:\\UnityDocumentation\\Manual\\class-PhysicMaterial.html)，可确定该碰撞体与其他对象的交互方式。 |
| **Center** | 碰撞体在对象局部空间中的位置。 |
| **Radius** | 碰撞体的局部宽度的半径。 |
| **Height** | 碰撞体的总高度。 |
| **Direction** | 胶囊体在对象局部空间中纵向方向的轴。 |

**详细信息**

可以独立调整胶囊碰撞体的 **Radius** 和 **Height**。胶囊碰撞体在[角色控制器](file:///E:\\UnityDocumentation\\Manual\\class-CharacterController.html)中用于模拟杆体，也可与其他碰撞体组合用于表现不寻常的形状。

# 车轮碰撞体

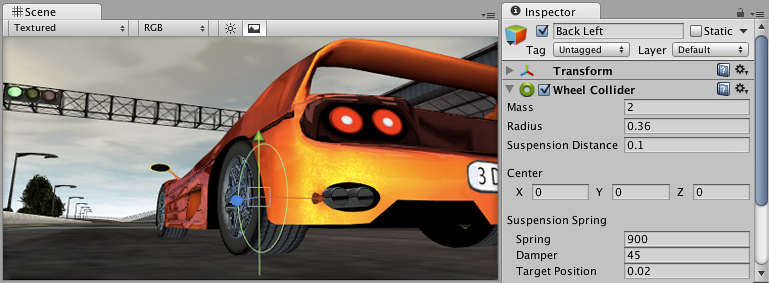
**车轮碰撞体 (Wheel Collider)** 是一种用于地面交通工具的特殊碰撞体。此碰撞体内置了碰撞检测、车轮物理组件和基于打滑的轮胎摩擦模型。此碰撞体可以用于除车轮以外的其他对象，但专门设计用于有轮的交通工具。



有关使用车轮碰撞体的指南，请参阅 [Unity 车轮碰撞体教程](file:///E:\UnityDocumentation\Manual\WheelColliderTutorial.html)。

## 属性

| **属性：** | **功能：** |
| --- | --- |
| **Mass** | 车轮的质量。 |
| **Radius** | 车轮的半径。 |
| **Wheel Damping Rate** | 这是应用于车轮的阻尼值。 |
| **Suspension Distance** | 车轮悬架的最大延伸距离（在局部空间中测量）。悬架始终向下延伸穿过局部 Y 轴。 |
| **Force App Point Distance** | 此参数定义车轮上的受力点。此距离应该是距车轮底部静止位置的距离（沿悬架行程方向），以米为单位。当 forceAppPointDistance = 0 时，受力点位于静止的车轮底部。较好的车辆会使受力点略低于车辆质心。 |
| **Center** | 车轮在对象局部空间中的中心位置。 |
| **Suspension Spring** | 悬架尝试通过增加弹簧力和阻尼力来到达\_\_目标位置 (Target Position)**。 | |&#160;&#160;&#160;&#160;&#160;&#160;&#160;&#160;**Spring\_\_ | 弹簧力尝试到达\_\_目标位置**。值越大，悬架达到\_\_目标位置\_\_就越快。 | |&#160;&#160;&#160;&#160;&#160;&#160;&#160;&#160;**Damper\_\_ | 抑制悬架速度。值越大，\_\_悬架弹簧\_\_移动就越慢。 |
| &#160;&#160;&#160;&#160;&#160;&#160;&#160;&#160;**Target Position** | 悬架沿悬架距离 (Suspension Distance) 的静止距离。1 对应于完全展开的悬架，0 对应于完全压缩的悬架。默认值为 0.5，与常规汽车的悬架行为匹配。 |  |  |
| **Forward/Sideways Friction** | 车轮向前和侧向滚动时轮胎摩擦的特性。请参阅下面的\_车轮摩擦曲线\_部分。 |  |  |

车轮碰撞体 (Wheel Collider) 组件。汽车模型由 ATI Technologies Inc. 提供

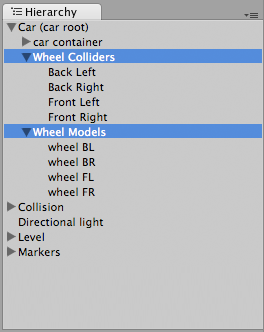
## 详细信息

通过从\_\_中心 (Center)\_\_ 向下穿过局部 Y 轴进行射线投射来执行车轮的碰撞检测。车轮具有\_\_半径 (Radius)**，并且根据\_\_悬架距离 (Suspension Distance)** 向下延伸。通过脚本使用不同属性来控制车辆，这些属性包括：\_\_motorTorque**、**brakeTorque\_\_ 和 **steerAngle**。有关更多信息，请参阅[车轮碰撞体脚本参考](file:///E:\UnityDocumentation\ScriptReference\WheelCollider.html)。

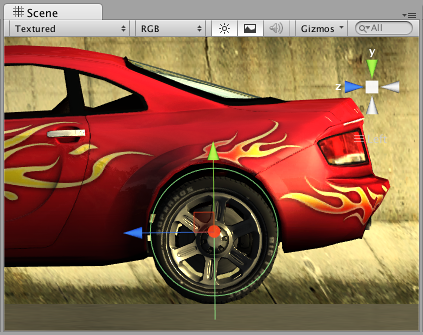
车轮碰撞体使用基于打滑的摩擦模型计算摩擦力（独立于物理引擎的其余部分）。这样可以实现更逼真的行为，但也会导致车轮碰撞体忽略标准[物理材质](file:///E:\UnityDocumentation\Manual\class-PhysicMaterial.html)设置。

### 车轮碰撞体设置

不必通过转动或滚动 WheelCollider 对象来控制汽车；附加了 WheelCollider 的对象应始终相对于汽车本身固定。但是，可能希望转动并滚动图形化的车轮表示。最好的方法是为车轮碰撞体和可见车轮设置单独的对象：

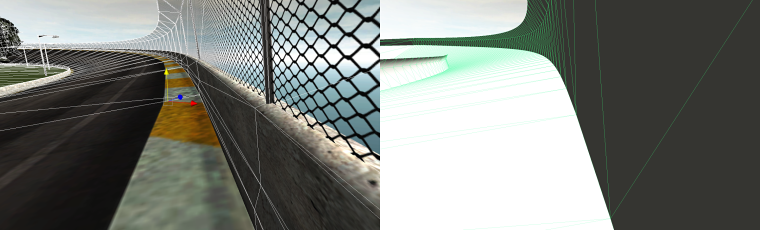
车轮碰撞体 (Wheel Colliders) 与可见车轮模型 (Wheel Models) 彼此独立

请注意，在播放模式下，WheelCollider 位置的辅助图标图形不会更新：

使用悬架距离 0.15 的 WheelCollider 辅助图标在运行时的位置

### 碰撞几何体

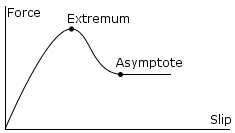
因为赛车可以达到很高的速度，所以正确设置赛道碰撞几何体非常重要。具体而言，[碰撞网格](file:///E:\UnityDocumentation\Manual\class-MeshCollider.html)不应具有构成可见模型（例如栅栏杆）的微小凹凸痕迹。通常，用于赛道的碰撞网格与可见网格分开制作，使碰撞网格尽可能平滑。此外不应有薄型对象；如果有薄型轨道边界，请在碰撞网格中使其加宽（如果汽车绝不会到达该处，应完全移除另一侧）。

可见几何体（左侧）比碰撞几何体（右侧）复杂得多

### 车轮摩擦曲线

下文中显示的\_车轮摩擦曲线\_可以描述轮胎摩擦。车轮的前进（滚动）方向和侧向方向有单独的曲线。在这两个方向上，首先确定轮胎打滑的程度（基于轮胎橡胶和道路之间的速度差异）。然后，将该打滑值用于计算施加在接触点上的轮胎力。

曲线以轮胎打滑的度量值作为输入，并以力作为输出。曲线由包含两部分的样条图近似模拟。第一部分从 (0 , 0) 到 (***ExtremumSlip*** , ***ExtremumValue***)，目标点处曲线的正切值为零。第二部分从 (***ExtremumSlip*** , ***ExtremumValue***) 到 (***AsymptoteSlip*** , ***AsymptoteValue***)，目标点处曲线的正切值再次为零：

车轮摩擦曲线的典型形状

根据真实轮胎的特性，在低打滑条件下，轮胎可能会施加很大的力，因为橡胶会通过拉伸来补偿打滑。随后，当打滑变得非常高时，随着轮胎开始滑动或旋转，力会减小。因此，轮胎摩擦曲线的形状与上图相似。

| **属性：** | **功能：** |
| --- | --- |
| **Extremum Slip/Value** | 曲线的极值点。 |
| **Asymptote Slip/Value** | 曲线的渐近点。 |
| **Stiffness** | **Extremum Value** 和 **Asymptote Value** 的乘数（默认值为 1）。改变摩擦力的刚度。将此值设置为零将完全禁用车轮的所有摩擦力。通常在运行时修改刚度以使用脚本来模拟各种地面材质。 |

## 提示

* 可能希望在 [Time Manager](file:///E:\UnityDocumentation\Manual\class-TimeManager.html) 中降低物理时间步长长度以获得更稳定的汽车物理特性，特别是在高速赛车的情况下。
* 为防止汽车太容易翻转，可以通过脚本稍微降低其[刚体](file:///E:\UnityDocumentation\Manual\class-Rigidbody.html)质心，并施加取决于汽车速度的“下压”力。

# 网格碰撞器

该**网格碰撞器**需要一个网格资源和建立了基于该网格的碰撞器。对于碰撞检测而言，它比使用复杂网格的基元要准确得多。标记为**凸面的**网格碰撞器可以与其他网格碰撞器碰撞。



## 属性

| **属性** | | **功能** |
| --- | --- | --- |
| **Convex** | | 勾选复选框以启用**Convex**。如果启用，此网格碰撞器将与其他网格碰撞器发生碰撞。**凸面**网格碰撞器限制为255个三角形。 |
| **Is Trigger** | | 如果启用，Unity会使用此Collider来触发事件，而物理引擎会忽略它。 |
| **Cooking Options** | | 启用或禁用影响物理引擎处理网格的方式的[网格烘焙](file:///E:\UnityDocumentation\Manual\class-MeshCollider.html#cooking)选项。 |
|  | **None** | 禁用下面列出的所有**烘焙选项**。 |
|  | **Everything** | 启用下面列出的所有**烘焙选项**。 |
|  | **Inflate Convex Mesh** | 允许物理引擎增加输入网格的体积，以生成有效的凸网格。 |
|  | **Cook for Faster Simulation** | 让物理引擎烘焙网格以加快模拟速度。启用后，会运行一些额外的步骤，以确保生成的Mesh最适合运行时性能。这会影响物理查询和联系人生成的性能。禁用此设置时，物理引擎会使用更快的烘焙时间，并尽可能快地生成结果。因此，煮熟的网格碰撞器可能不是最佳的。 |
|  | **Enable Mesh Cleaning** | 使物理引擎清洁网格。启用后，烘焙过程会尝试消除网格的[退化三角形](https://en.wikipedia.org/wiki/Degeneracy_(mathematics)#Triangle)以及其他几何图形。这导致网格更适合用于碰撞检测并且倾向于产生更准确的生命值。 |
|  | **Weld Colocated Vertices** | 使物理引擎在网格中删除相等的顶点。启用后，物理引擎会组合具有相同位置的顶点。这对于在运行时发生的碰撞反馈很重要。 |
| **Material** | | 引用[物理材质](file:///E:\UnityDocumentation\Manual\class-PhysicMaterial.html)，可确定该碰撞体与其他对象的交互方式。 |
| **Mesh** | | 引用用于碰撞的网格。 |

## 详细信息

网格碰撞器构建从它的碰撞表示[网](file:///E:\UnityDocumentation\Manual\class-Mesh.html)附接至游戏物体，并且读取所连接的属性[变换](file:///E:\UnityDocumentation\Manual\class-Transform.html)来设置其位置和正确缩放。这样做的好处是，您可以使Collider的形状与GameObject的可见Mesh的形状完全相同，从而产生更精确和真实的碰撞。但是，与包含原始碰撞器（例如Sphere，Box和Capsule）的碰撞相比，这种精度会带来更高的处理开销，因此最好谨慎使用网格碰撞器。

碰撞网格中的面是片面的。这意味着物体可以从一个方向穿过它们，但是从另一个方向与它们碰撞。

## 网格烘焙

网格烘焙将正常的网格物体更改为可以在物理引擎中使用的网格物体。Cooking为物理查询构建空间搜索结构，例如[Physics.Raycast](file:///E:\UnityDocumentation\ScriptReference\Physics.Raycast.html)，以及用于生成联系人的支持结构。Unity在碰撞检测中使用它们之前烘焙所有网格物体。这可能发生在导入时（**导入设置>模型>生成碰撞器**）或运行时。

在运行时生成网格物体时（例如，对于程序曲面），将**烘焙选项**设置为更快地生成结果并禁用其他数据清洁步骤是很有用的。缺点是你不需要生成退化三角形和没有共同定位的顶点，但烘焙工作更快。

如果禁用“ **启用网格清理”**或“ **焊接共同顶点”**，则需要确保不使用这些算法可能会过滤的数据。如果禁用“ **焊接**共同顶点”，请确保没有任何共同定位的顶点，并且**启用“启用网格清理”**时，请确保没有小三角形，其面积接近零，没有细三角形，也没有大三角形面积接近无限。

**注意**：将**烘焙选项**设置为除默认设置之外的任何其他值意味着网格碰撞器必须使用[isReadable](file:///E:\UnityDocumentation\ScriptReference\Mesh-isReadable.html)值为true 的网格。

## 限制

使用网格碰撞器时有一些限制：

只有没有Rigidbody组件的GameObjects支持没有启用**Convex的**网格碰撞器。要将“网格碰撞器”应用于“刚体”组件，请勾选“ **凸”**复选框。

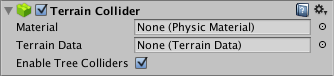
要使网格碰撞器正常工作，必须在以下任何情况下启用网格读取/写入：

* 网格碰撞器的变换具有负缩放（例如，（-1,1,1））。
* 网格碰撞器的变换是倾斜或剪切的（例如，当旋转的变换具有缩放的父变换时）。
* Mesh Collider的烘焙选项标志设置为默认值以外的任何值。

**优化提示：**如果网格仅由网格碰撞器使用，则可以在**导入设置中**禁用**法线**，因为物理系统不需要它们。

# 地形碰撞体

**地形碰撞体 (Terrain Collider)** 实现了一个碰撞表面，其形状与其所附加到的 [Terrain](file:///E:\UnityDocumentation\Manual\script-Terrain.html) 对象相同。



## 属性

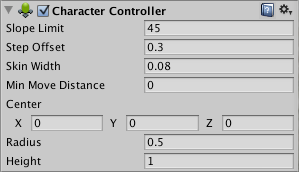
| **属性：** | **功能：** |
| --- | --- |
| **Material** | 引用[物理材质](file:///E:\UnityDocumentation\Manual\class-PhysicMaterial.html)，可确定该碰撞体与其他对象的交互方式。 |
| **Terrain Data** | 地形数据。 |
| **Enable Tree Colliders** | 选中此属性时，将启用树碰撞体。 |

## 详细信息

应注意，Unity 5.0 之前的版本中，地形碰撞体具有 **Smooth Sphere Collisions** 属性，用于改善地形和球体之间的相互作用。此属性现已废弃，因为平滑交互是物理引擎的标准行为，将其关闭没有特别的优势。

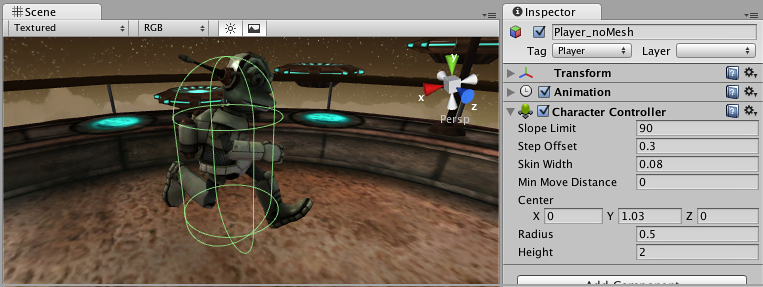
# 角色控制器

**角色控制器 (Character Controller)** 主要用于第三人称玩家控制或者是不使用\_\_刚体\_\_物理组件的第一人称玩家控制。



## 属性

| **属性：** | **功能：** |
| --- | --- |
| **Slope Limit** | 将碰撞体限制为爬坡的斜率不超过指示值（以度为单位）。 |
| **Step Offset** | 仅当角色比指示值更接近地面时，角色才会升高一个台阶。该值不应该大于角色控制器的高度，否则会产生错误。 |
| **Skin width** | 两个碰撞体可以穿透彼此且穿透深度最多为皮肤宽度 (Skin Width)。较大的皮肤宽度可减少抖动。较小的皮肤宽度可能导致角色卡住。合理设置是将此值设为半径的 10%。 |
| **Min Move Distance** | 如果角色试图移动到指示值以下，根本移动不了。此设置可以用来减少抖动。在大多数情况下，此值应保留为 0。 |
| **Center** | 此设置将使胶囊碰撞体在世界空间中偏移，并且不会影响角色的枢转方式。 |
| **Radius** | 胶囊碰撞体的半径长度。此值本质上是碰撞体的宽度。 |
| **Height** | 角色的\_\_胶囊碰撞体\_\_高度。更改此设置将在正方向和负方向沿 Y 轴缩放碰撞体。 |

Character Controller

## 详细信息

传统末日风格的第一人称控制在现实中并不真实。该角色每小时能跑 90 英里，可以立即停止并急转弯。因为该角色非常不真实，所以使用刚体和物理组件来创造这种行为有点不切实际，并会让玩家产生错觉。解决方案是使用专门的角色控制器。角色控制器只是一个胶囊形状的\_\_碰撞体\_\_，可以通过脚本来命令这个碰撞体向某个方向移动。然后，控制器将执行运动，但会受到碰撞的约束。控制器将沿着墙壁滑动，走上楼梯（如果低于 **Step Offset** 值），并走上 **Slope Limit** 设置范围内的斜坡。

控制器本身不会对力作出反应，也不会自动推开刚体。

如果要通过角色控制器来推动刚体或对象，可以编写脚本通过 **OnControllerColliderHit()** 函数对与控制器碰撞的任何对象施力。

另一方面，如果希望玩家角色受到物理组件的影响，那么可能更适合使用[刚体](file:///E:\UnityDocumentation\Manual\class-Rigidbody.html)，而不是角色控制器。

### 微调角色

可以修改 **Height** 和 **Radius** 属性来适应角色的网格。对于人形角色，建议始终使用 2 米左右的值。如果轴心点并非刚好在角色的中心，还可以修改胶囊体的 **Center** 属性。

**Step Offset** 属性也可能有影响，对于身高 2 米的人，请确保此值在 0.1 到 0.4 之间。

**Slope Limit** 不应太小。通常，使用 90 度的值效果最佳。由于胶囊体形状的原因，角色控制器将无法爬墙。

### 不要被卡住

要正确调整角色控制器，\_\_Skin Width\_\_ 属性是最重要的属性之一。 如果角色被卡住，那么很可能是因为 **Skin Width** 设置过小。**Skin Width** 允许对象轻微穿透控制器，但可消除抖动并防止被卡住。

最好是让 **Skin Width** 的值至少大于 0.01 并且比 **Radius** 的值大 10%。

建议将 **Min Move Distance** 保持为 0。

请参阅[此处](file:///E:\UnityDocumentation\ScriptReference\CharacterController.html)的角色控制器脚本参考

可以从我们网站上的[资源 (Resources)](https://unity3d.com/learn/resources) 区域下载一个示例项目（其中显示了预设置的动画角色控制器和移动角色控制器）。

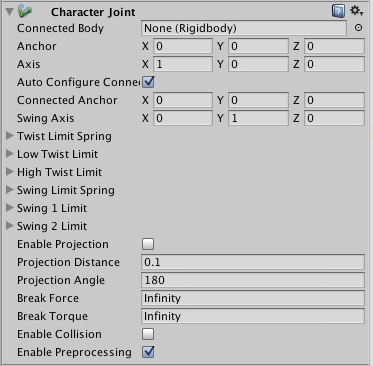
## 提示

* 如果发现角色经常被卡住，请尝试调整 **Skin Width**。
* 如果是自己编写脚本，则角色控制器可能会影响使用物理组件的对象。
* 对象无法通过物理组件来影响角色控制器。
* 请注意，在 Inspector 中更改角色控制器属性将在场景中重新创建控制器，因此任何现有的触发器触点都将丢失，并且在再次移动控制器之前，不会收到任何 OnTriggerEntered 消息。
* 在查询中使用的角色控制器胶囊体（比如射线投射）可能会略有缩小。因此，在某些极端情况下，即使查询似乎命中了角色控制器的辅助图标，但实际可能未命中。

# 角色关节

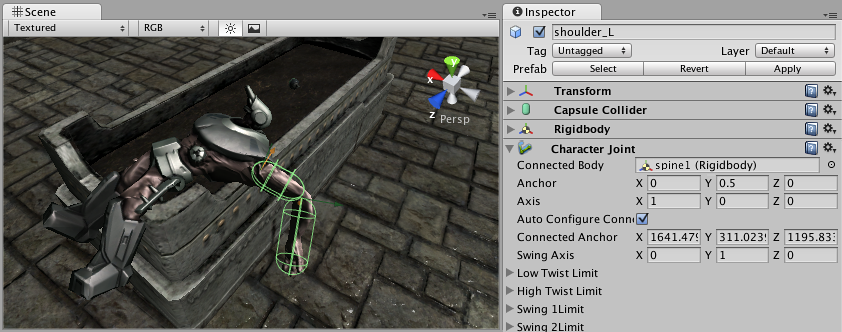
**角色关节 (Character Joint)** 主要用于布娃娃效果。此类关节是延长的球窝关节，可在每个轴上限制该关节。

如果仅希望设置布娃娃，请阅读[布娃娃向导](file:///E:\UnityDocumentation\Manual\wizard-RagdollWizard.html)。



## 属性

| **属性：** | **功能：** |
| --- | --- |
| **Connected Body** | 对关节所依赖的\_\_刚体\_\_的引用（可选）。如果未设置，则关节连接到世界。 |
| **Anchor** | 关节在\_\_游戏对象\_\_的局部空间中旋转时围绕的点。 |
| **Axis** | 扭转轴。用橙色的辅助图标椎体可视化。 |
| **Auto Configure Connected Anchor** | 如果启用此属性，则会自动计算连接锚点 (Connected Anchor) 位置以便与锚点属性的全局位置匹配。这是默认行为。如果禁用此属性，则可以手动配置连接锚点的位置。 |
| **Connected Anchor** | 手动配置连接锚点位置。 |
| **Swing Axis** | 摆动轴。用绿色的辅助图标椎体可视化。 |
| **Low Twist Limit** | 关节的下限。请参阅下文。 |
| **High Twist Limit** | 关节的上限。请参阅下文。 |
| **Swing 1 Limit** | 限制围绕定义的\_\_摆动轴 (Swing Axis)\_\_ 的一个元素的旋转（用辅助图标上的绿色轴可视化）。请参阅下文。 |
| **Swing 2 Limit** | 限制围绕定义的\_\_摆动轴\_\_的一个元素的移动。请参阅下文。 |
| **Break Force** | 为破坏此关节而需要施加的力。 |
| **Break Torque** | 为破坏此关节而需要施加的扭矩。 |
| **Enable Collision** | 选中此复选框后，允许关节连接的连接体之间发生碰撞。 |
| **Enable Preprocessing** | 禁用预处理有助于稳定无法满足的配置。 |

布娃娃上的角色关节

## 详细信息

角色关节提供了很多约束运动的可能性，就像使用万向节一样。

扭转轴（用辅助图标上的橙色椎体可视化）可在很大程度上控制上限和下限，允许按照度数指定上限和下限（限制角度是相对于开始位置进行测量的）。**Low Twist Limit** > **Limit** 中的值 –30 和 **High Twist Limit** > **Limit** 中的值 60 可将围绕扭转轴（橙色辅助图标）的旋转范围限制在 –30 度到 60 度之间。

**Swing 1 Limit** 可限制摆动轴的旋转范围（用辅助图标上的绿色轴可视化）。限制角度是对称的。因此，值 30 会将旋转限制在 –30 到 30 之间。

**Swing 2 Limit** 轴未显示在辅助图标上，但该轴垂直于其他两个轴（即辅助图标上用橙色可视化的扭转轴和辅助图标上用绿色可视化的 **Swing 1 Limit** 轴）。 角度是对称的，因此值 40 可将围绕该轴的旋转范围限制在 –40 度到 40 度之间。

对于每个限制，可以设置以下值：

| **属性：** | **功能：** |
| --- | --- |
| **Bounciness** | 值为 0 将不会反弹。值为 1 将在反弹时不产生任何能量损失。 |
| **Spring** | 用于将两个对象保持在一起的弹簧力。 |
| **Damper** | 用于抑制弹簧力的阻尼力。 |
| **Contact Distance** | 在距离极限位置的接触距离内，接触将持续存在以免发生抖动。 |

### 破坏关节

可使用 **Break Force** 和 **Break Torque** 属性来设置关节强度的限制。如果这些值小于无穷大，并对该对象施加大于这些限制的力/扭矩，则其固定关节将被破坏并将摆脱其约束的束缚。

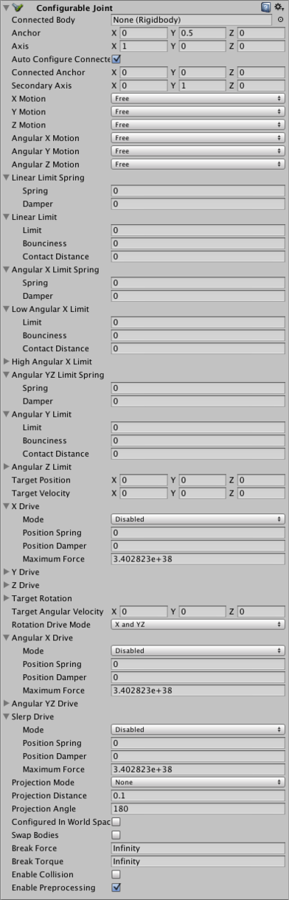
## 提示

* 不需要为关节分配 **Connected Body** 便可让关节运作。
* 角色关节要求对象附加一个刚体。
* 对于通过布娃娃向导制作的角色关节，请注意设置：关节的扭转轴与肢体的最大摆动轴对应，关节的 Swing 1 轴与肢体的较小摆动轴对应，而关节的 Swing 2 表示扭转肢体。采用这种命名方案是为了沿用旧版。

# 可配置关节

**可配置关节 (Configurable Joint)** 的可定制性极高，因为此类关节包含其他关节类型的所有功能。使用可配置关节可以创建任何关节，从现有关节的改编版到自行设计的高度专业化关节，不一而足。

## 属性



| **属性：** | **功能：** |
| --- | --- |
| **Connected Body** | 关节连接到的另一个刚体对象。可将此属性设置为 None 来表示关节连接到空间中的固定位置，而不是另一个刚体。 |
| **Anchor** | 用于定义关节中心的点。所有基于物理的模拟都将使用此点作为计算中的中心 |
| **Axis** | 用于基于物理模拟来定义对象自然旋转的局部轴 |
| **Auto Configure Connected Anchor** | 如果启用此属性，则会自动计算连接锚点 (Connected Anchor) 位置以便与锚点属性的全局位置匹配。这是默认行为。如果禁用此属性，则可以手动配置连接锚点的位置。 |
| **Connected Anchor** | 手动配置连接锚点位置。 |
| **Secondary Axis** | **Axis** 和 **Secondary Axis** 共同定义了关节的局部坐标系。第三个轴设置为与另外两个轴正交。 |
| **X, Y, Z Motion** | 根据以下描述的限制属性，允许沿 X、Y 或 Z 轴的移动是 Free、完全 Locked 还是 Limited。 |
| **Angular X, Y, Z Motion** | 根据以下描述的限制属性，允许沿 X、Y 或 Z 轴的旋转是 Free、完全 Locked 还是 Limited。 |
| **Linear Limit Spring** | 当对象超过了限制位置时要拉回对象而施加的弹簧力。 |
| **Spring** | 弹簧力。如果此值设置为零，则无法逾越限制；零以外的值将使限制变得有弹性。 |
| **Damper** | 根据关节运动的速度按比例减小弹簧力。设置为大于零的值可让关节“抑制”振荡（否则将无限期进行振荡）。 |
| **Linear Limit** | 关节线性移动的限制（即，移动距离而不是旋转），指定为距关节原点的距离。 |
| **Limit** | 从原点到限制位置的距离（采用世界单位）。 |
| **Bounciness** | 当对象达到限制距离时，要将对象拉回而施加的弹力。 |
| **Contact Distance** | 需要强制执行限制时的最小距离公差（关节位置和限制位置之间）。公差越大，对象快速移动时违反限制的可能性就越低。但是，这种情况下也需要通过更频繁进行物理模拟来考虑限制，而这往往会略微降低性能。 |
| **Angular X Limit Spring** | 当对象超过了关节的限制角度时要反向旋转对象而施加的弹簧扭矩。 |
| **Spring** | 弹簧扭矩。如果此值设置为零，则无法逾越限制；零以外的值将使限制变得有弹性。 |
| **Damper** | 根据关节旋转的速度按比例减小弹簧扭矩。设置为大于零的值可让关节“抑制”振荡（否则将无限期进行振荡）。 |
| **Low Angular X Limit** | 关节绕 X 轴旋转的下限，指定为距关节原始旋转的角度。 |
| **Limit** | 限制角度。 |
| **Bounciness** | 当对象的旋转达到限制角度时在对象上施加的反弹扭矩。 |
| **Contact Distance** | 需要强制执行限制时的最小角度公差（关节角度和限制位置之间）。公差越大，对象快速移动时违反限制的可能性就越低。但是，这种情况下也需要通过更频繁进行物理模拟来考虑限制，而这往往会略微降低性能。 |
| **High Angular XLimit** | 类似于上述 Low Angular X Limit 属性，但确定的是关节旋转的角度上限，而不是下限。 |
| **Angular YZ Limit Spring** | 类似于上述 Angular X Limit Spring\_，但适用于围绕 Y 轴和 Z 轴的旋转。 | |***Angular Y Limit*** |类似于上述 Angular X Limit\_ 属性，但适用于 Y 轴，并将角度的上限和下限视为相同。 |
| **Angular Z Limit** | 类似于上述 Angular X Limit 属性，但适用于 Z 轴，并将角度的上限和下限视为相同。 |
| **Target Position** | 关节的驱动力应该关节移动到的目标位置。 |
| **Target Velocity** | 关节在驱动力下移动到\_目标位置 (Target Position)\_ 时应该采用的所需速度。 |
| **XDrive** | 将关节沿局部 X 轴线性移动的驱动力。 |
| **Mode** | 该模式确定关节在移动时应该达到指定\_\_位置 (Position)\_\_ 还是达到指定\_\_速度 (Velocity)**，还是同时达到指定位置和指定速度。 | |**Position Spring\_\_ | 将关节向目标位置移动的弹簧力。仅当驱动模式设置为 Position 或 Position and Velocity时才会使用此属性。 |
| **Position Damper** | 根据关节运动的速度按比例减小弹簧力。设置为大于零的值可让关节“抑制”振荡（否则将无限期进行振荡）。仅当驱动模式设置为 Position 或 Position and Velocity 时才会使用此属性。 |  |
| **Maximum Force** | 将关节向目标速度加速时使用的力。仅当驱动模式设置为 Velocity 或 Position and Velocity 时才会使用此属性。 |  |
| **YDrive** | 类似于上述 X Drive\_，但适用于关节的 Y 轴。 | |***ZDrive*** |类似于上述 X Drive\_，但适用于关节的 Z 轴。 |  |
| **Target Rotation** | 关节旋转驱动应朝向的方向，指定为[四元数](file:///E:\UnityDocumentation\ScriptReference\Quaternion.html)。 |  |
| **Target Angular Velocity** | 关节的旋转驱动应达到的角速度。该值指定为矢量（其长度指定旋转速度，而其方向定义旋转轴）。 |  |
| **Rotation Drive Mode** | 将驱动力应用于对象以将其旋转到目标方向的方式。如果该模式设置为 X and YZ\_，则将围绕这些轴施加扭矩（由如下所述的 Angular X/YZ Drive\_ 属性指定）。如果使用 **Slerp** 模式，则 Slerp Drive 属性将确定驱动扭矩。 |  |
| **Angular X Drive** | 此属性指定了驱动扭矩将使关节如何围绕局部 X 轴旋转。仅当上述 Rotation Drive Mode 属性设置为 X & YZ 时，才会使用此属性。 |  |
| **Mode** | 该模式确定关节在移动时应该达到指定角\_\_位置\_\_还是达到指定角\_\_速度\_\_，还是同时达到指定角位置和指定角速度。 |  |
| **Position Spring** | 将关节向目标位置旋转的弹簧扭矩。仅当驱动模式设置为 Position 或 Position and Velocity 时才会使用此属性。 |  |
| **Position Damper** | 根据关节移动的速度按比例减小弹簧扭矩。设置为大于零的值可让关节“抑制”振荡（否则将无限期进行振荡）。仅当驱动模式设置为 Position 或 Position and Velocity 时才会使用此属性。 |  |
| **Maximum Force** | 将关节向目标速度加速时使用的扭矩。仅当驱动模式设置为 Velocity 或 Position and Velocity 时才会使用此属性。 |  |
| **Angular YZDrive** | 类似于上述 Angular X Drive\_，但适用于关节的 Y 轴和 Z 轴。 | |***Slerp Drive*** |此属性指定了驱动扭矩将使关节如何围绕所有局部轴旋转。仅当上述 Rotation Drive Mode\_ 属性设置为 Slerp 时，才会使用此属性。 |  |
| **Mode** | 该模式确定关节在移动时应该达到指定角\_\_位置\_\_还是达到指定角\_\_速度\_\_，还是同时达到指定角位置和指定角速度。 |  |
| **Position Spring** | 将关节向目标位置旋转的弹簧扭矩。仅当驱动模式设置为 Position 或 Position and Velocity 时才会使用此属性。 |  |
| **Position Damper** | 根据关节移动的速度按比例减小弹簧扭矩。设置为大于零的值可让关节“抑制”振荡（否则将无限期进行振荡）。仅当驱动模式设置为 Position 或 Position and Velocity 时才会使用此属性。 |  |
| **Maximum Force** | 将关节向目标速度加速时使用的扭矩。仅当驱动模式设置为 Velocity 或 Position and Velocity 时才会使用此属性。 |  |
| **Projection Mode** | 此属性定义了当关节意外地超过自身的约束（由于物理引擎无法协调模拟中当前的作用力组合）时如何快速恢复约束。选项为 None 和 Position and Rotation。 |  |
| **Projection Distance** | 关节超过约束的距离，必须超过此距离才能让物理引擎尝试将关节拉回可接受位置。 |  |
| **Projection Angle** | 关节超过约束的旋转角度，必须超过此角度才能让物理引擎尝试将关节拉回可接受位置。 |  |
| **Configured in World Space** | 是否应该在世界空间而不是对象的局部空间中计算由各种目标和驱动属性设置的值？ |  |
| **Swap Bodies** | 如果启用此属性，则会使关节表现得好像组件已附加到连接的刚体（即关节的另一端）。 |  |
| **Break Force** | 如果通过大于该值的力推动关节超过约束，则关节将被永久“破坏”并被删除。 |  |
| **Break Torque** | 如果通过大于该值的扭矩旋转关节超过约束，则关节将被永久“破坏”并被删除。 |  |
| **Enable Collision** | 包含此关节的对象是否应能够与连接的对象碰撞（而不是仅仅相互穿过）？ |  |
| **Enable Preprocessing** | 如果禁用预处理，则关节某些“不可能”的配置将保持更稳定，而不会在失控状态下狂乱移动。 |  |

## 详细信息

与其他关节一样，使用可配置关节可以限制对象的移动，而且可以使用作用力将对象驱动到目标速度或位置。但是，有许多可用的配置选项在组合使用时可能非常微妙；可能需要尝试不同的选项，才能使关节完全按照所需的方式运行。

### 约束运动

可使用 X, Y, Z Motion 和 X, Y, Z Rotation 属性独立约束每个轴上的平移运动和旋转。如果启用 Configured In World Space\_，则运动将被约束到世界轴而不是对象的局部轴。所有这些属性都可以设置为 Locked、Limited\_ 或 \_Free\_：

* **Locked** 轴不允许任何移动。例如，世界 Y 轴中锁定的对象无法上下移动。
* **Limited** 轴允许在预定义的限制范围内自由移动，如下所述。例如，通过将炮塔的 Y 旋转限制到特定角度范围，可以给炮塔设置受限制的火弧。
* **Free** 轴允许任意移动。

可使用 Linear Limit 属性限制平移运动，该属性定义了关节可从其原点移动的最大距离。（分别沿每个轴进行测量）。例如，为了能够约束空气曲棍球台的曲棍球，可让关节在 Y 轴为锁定状态（在世界空间中），在 Z 轴为自由状态，并在 X 轴为受限状态以便适应球台的宽度限制；这样，曲棍球将被限制在游戏区域内。

还可以使用 Angular Limit 属性来限制旋转。与线性限制不同，这些属性允许为每个轴指定不同的限制值。此外，还可以为 X 轴的旋转角度定义单独的上限和下限（其他两个轴在原始旋转的两侧使用相同的角度）。例如，可使用平面构造一个“摇摆平台”，将关节约束为允许在 X 和 Z 方向轻微摇摆，同时锁定 Y 旋转。

#### 弹性和弹簧

默认情况下，关节只有在达到限制时才会停止移动。然而，像这样的非弹性碰撞在现实世界中是罕见的，因此向受约束的关节添加一些弹跳感会很有用。可使用线性和角度限制的 Bounciness 属性使受约束的对象在达到限制后反弹。大部分的碰撞在有了少量弹性之后会显得更自然，但也可以将该属性设置为更高的值，从而模拟弹性异常大的边界，比如台球桌垫。

有弹性的关节不会超越限制

使用弹簧属性（用于平移的 Linear Limit Spring 和用于旋转的 Angular X/YZ Limit Spring\_）可以进一步增强关节限制。如果将 Spring\_ 属性设置为大于零的值，关节达到限制时将不会突然停止移动，而是会通过弹簧力（力的强度是由 Spring 值确定）被拉回到限制位置。默认情况下，弹簧具有完美的弹性，倾向于按照与碰撞相反的方向弹回关节。但是，可使用 Damper 属性来减少弹性并以更温和的方式将关节恢复到限制位置。例如，可使用弹簧关节创建一个可向左或向右拉动的杠杆，然后弹回到直立位置。如果弹簧具有完美弹性，则杠杆在释放后将倾向于围绕中心点来回摆动。但是，如果添加足够的阻尼，弹簧将迅速稳定到中立位置。

弹簧关节越过限制位置但被拉回到限制位置

### 驱动力

关节不仅可以对附加的对象做出反应，而且还可以主动施加\_驱动力\_使对象运动。一些关节只需要保持对象以恒定速度移动，例如转动风扇叶片的旋转电机。可使用 Target Velocity 和 Target Angluar Velocity 属性为此类关节设置所需的速度。可能还需要通过关节将对象移动到空间（或特定方向）的特定位置；此情况下，可使用 Target Position 和 Target Rotation 属性来设置关节。例如，可将叉车的叉子安装在可配置关节上，然后使用脚本设置目标高度以升高叉子，即可实现叉车。

设置了目标后，使用 X, Y, Z Drive 和 Angular X/YZ Drive\_（或者是 Slerp Drive\_）属性指定用于将关节推向目标的作用力。驱动的 Mode 属性选择关节是否应该寻找目标位置和/或速度。在寻找目标位置时，\_Position Spring\_ 和 Position Damper 的工作方式与关节限制相同。在速度模式下，弹簧力取决于当前速度和目标速度之间的差距；阻尼器有助于速度稳定在所选值，而不是在该值周围无限振荡。Maximum Force 属性用于最终微调，无论关节距其目标有多远，均可防止弹簧施加的力超过限制值。这样可以防止远离目标的关节快速以不受控制的方式将对象拉回的情况。

请注意，对于所有三个驱动力（下面描述的 Slerp Drive 除外），力都在每个轴上单独施加。因此，举例来说，可实现一个航天器，使之具有较高的向前飞行速度，但在侧向转向运动中具有相对较低的速度。

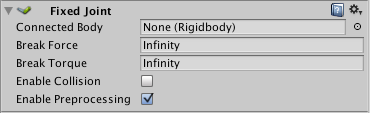
#### Slerp Drive

其他驱动模式在不同的轴上施加力，但 Slerp Drive 使用四元数的球面插值或“Slerp”功能来重新定向关节。Slerp 过程不会隔离单个轴，而是整体上寻找将对象从当前方向带到目标的最小总旋转，并根据需要将该旋转应用于所有轴。Slerp Drive 的配置比较简单并适合大多数用途，但不允许为 X 和 Y/Z 轴指定不同的驱动力。

要启用 Slerp Drive，应将 Rotation Drive Mode 属性从 X and YZ 更改为 Slerp。注意，模式是互斥的；关节将使用 Angular X/YZ Drive 值或 Slerp Drive 值，但不能同时使用两者。

# 固定关节

**固定关节 (Fixed Joint)** 将对象的移动限制为依赖于另一个对象。这有点类似于\_\_管控 (Parenting)**，但是实现的方式是通过物理系统而不是\_\_变换 (Transform)** 层级视图。使用固定关节的最佳场合是在希望对象可以轻松相互分离时，或者在没有管控情况下连接两个对象的移动。



## 属性

| **属性：** | **功能：** |
| --- | --- |
| **Connected Body** | 对关节所依赖的刚体的引用（可选）。如果未设置，则关节连接到世界。 |
| **Break Force** | 为破坏此关节而需要施加的力。 |
| **Break Torque** | 为破坏此关节而需要施加的扭矩。 |
| **Enable Collision** | 选中此复选框后，允许关节连接的连接体之间发生碰撞。 |
| **Enable Preprocessing** | 禁用预处理有助于稳定无法满足的配置。 |

## 详细信息

在游戏中有时可能希望对象永久或暂时粘在一起。固定关节可能是比较适合用于这些情况的\_\_组件\_\_，因为不必通过脚本更改对象的层级视图来实现所需的效果。代价是所有使用固定关节的对象都必须使用\_\_刚体\_\_。

例如，如果要使用“粘性手榴弹”，可写一个脚本来检测与另一刚体（如敌人）的碰撞，然后创建一个固定关节并附加到该刚体。然后，当敌人四处移动时，关节将使手榴弹紧贴在他们身上。

### 破坏关节

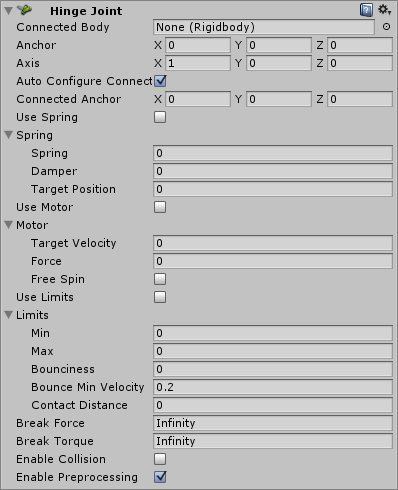
可使用 **Break Force** 和 **Break Torque** 属性来设置关节强度的限制。如果这些值小于无穷大，并对该对象施加大于这些限制的力/扭矩，则其固定关节将被破坏并将摆脱其约束的束缚。

## 提示

* 不需要为关节分配 **Connected Body** 便可让关节运作。
* 固定关节需要一个刚体。

# 铰链关节

**铰链关节 (Hinge Joint)** 将两个[刚体](file:///E:\UnityDocumentation\Manual\class-Rigidbody.html)组合在一起，对刚体进行约束，让它们就像通过铰链连接一样移动。铰链关节非常适合用于门，但也可用于模拟链条、钟摆等对象。



## 属性

| **属性：** | **功能：** |
| --- | --- |
| **Connected Body** | 对关节所依赖的刚体的引用（可选）。如果未设置，则关节连接到世界。 |
| **Anchor** | 连接体围绕摆动的轴位置。该位置在局部空间中定义。 |
| **轴** | 连接体围绕摆动的轴方向。该方向在局部空间中定义。 |
| **Auto Configure Connected Anchor** | 如果启用此属性，则会自动计算连接锚点 (Connected Anchor) 位置以便与锚点属性的全局位置匹配。这是默认行为。如果禁用此属性，则可以手动配置连接锚点的位置。 |
| **Connected Anchor** | 手动配置连接锚点位置。 |
| **Use Spring** | 弹簧使刚体相对于其连接体呈现特定角度。 |
| **Spring** | 在启用 **Use Spring** 的情况下使用的弹簧的属性。 |
| **Spring** | 对象声称移动到位时施加的力。 |
| **Damper** | 此值越高，对象减速越快。 |
| **Target Position** | 弹簧的目标角度。弹簧朝着该角度拉伸（以度为单位）。 |
| **Use Motor** | 电机使对象旋转。 |
| **Motor** | 在启用 **Use Motor** 的情况下使用的电机的属性。 |
| **Target Velocity** | 对象试图获得的速度。 |
| **Force** | 为获得该速度而施加的力。 |
| **Free Spin** | 如果启用此属性，则绝不会使用电机来制动旋转，只会进行加速。 |
| **Use Limits** | 如果启用此属性，则铰链的角度将被限制在 **Min** 到 **Max** 值范围内。 |
| **Limits \_\_ |在启用**Use Limits\_\_ 的情况下使用的限制的属性。 |  |
| **Min** | 旋转可以达到的最小角度。 |
| **Max** | 旋转可以达到的最大角度。 |
| **Bounciness** | 当对象达到了最小或最大停止限制时对象的反弹力大小。 |
| **Contact Distance** | 在距离极限位置的接触距离内，接触将持续存在以免发生抖动。 |
| **Break Force** | 为破坏此关节而需要施加的力。 |
| **Break Torque** | 为破坏此关节而需要施加的扭矩。 |
| **Enable Collision** | 选中此复选框后，允许关节连接的连接体之间发生碰撞。 |
| **Enable Preprocessing** | 禁用预处理有助于稳定无法满足的配置。 |

## 详细信息

应将单个铰链关节应用于\_\_游戏对象**。铰链将在**Anchor\_\_ 属性指定的位置旋转，并围绕指定的 **Axis** 属性移动。**不**需要为关节的 **Connected Body** 属性分配游戏对象。仅当希望关节的\_\_变换\_\_依赖于附加对象的变换时，才应为 **Connected Body** 属性分配游戏对象。

请思考门铰链的工作原理。在这种情况下，\_\_轴 (Axis)\_\_ 沿 Y 轴正方向朝上。**锚点 (Anchor)** 位于门和墙之间交叉处的某个位置。不需要将墙分配给\_\_连接体 (Connected Body)\_\_，因为默认情况下，关节将连接到世界。

现在请思考狗窝门的铰链。狗窝门的\_\_轴\_\_将是侧向的，并沿着正向的相对 X 轴。应将主门分配为\_\_连接体\_\_，因此狗窝门的铰链取决于主门的刚体。

### 链条

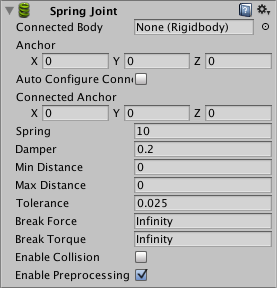
还可以将多个铰链关节串在一起以形成链条。为链条中的每个链接添加一个关节，并将下一个链接作为\_\_连接体\_\_附加。

## 提示

* 不需要为关节分配 **Connected Body** 便可让关节运作。
* 使用 **Break Force** 属性来创建动态损坏系统。使用此属性可让玩家破坏环境（例如，通过用火箭发射器爆炸或用汽车冲撞的方式破坏门的铰链）。
* **Spring**、**Motor** 和 **Limits** 属性可用于微调关节的行为。
* **Spring** 和 **Motor** 是互斥的。同时使用这两者会导致不可预测的结果。

# 弹簧关节

**弹簧关节 (Spring Joint)** 将两个[刚体](file:///E:\UnityDocumentation\Manual\class-Rigidbody.html)连接在一起，但允许两者之间的距离改变，就好像它们通过弹簧连接一样。



## 属性

| **属性：** | **功能：** |
| --- | --- |
| **Connected Body** | 包含弹簧关节的对象连接到的刚体对象。如果未指定对象，则弹簧将连接到空间中的固定点。 |
| **Anchor** | 关节在对象的局部空间中所附加到的点。 |
| **Auto Configure Connected Anchor** | Unity 是否应该自动计算连接锚点的位置？ |
| **Connected Anchor** | 关节在连接对象的局部空间中所附加到的点。 |
| **Spring** | 弹簧的强度。 |
| **Damper** | 弹簧为活性状态时的压缩程度。 |
| **Min Distance** | 弹簧不施加任何力的距离范围的下限。 |
| **Max Distance** | 弹簧不施加任何力的距离范围的上限。 |
| **Tolerance** | 更改容错。允许弹簧具有不同的静止长度。 |
| **Break Force** | 为破坏此关节而需要施加的力。 |
| **Break Torque** | 为破坏此关节而需要施加的扭矩。 |
| **Enable Collision** | 是否应启用两个连接对象之间的相互碰撞？ |
| **Enable Preprocessing** | 禁用预处理有助于稳定无法满足的配置。 |

## 详细信息

弹簧就像一块弹性物，试图将两个锚点一起拉到完全相同的位置。拉力的强度与两个点之间的当前距离成比例，其中每单位距离的力由 Spring 属性设定。为了防止弹簧无休止振荡，可以设置 Damper 值，从而根据与两个对象之间的相对速度按比例减小弹簧力。值越高，振荡消失的速度越快。

可以手动设置锚点，但如果启用 \_Auto Configure Connected Anchor\_，Unity 将自动设置连接锚点，以保持它们之间的初始距离（即，在定位对象时在 Scene 视图中设置的距离）。

Min Distance 和 Max Distance 值用于设置弹簧不施加任何力的距离范围。例如，可以使用该距离范围允许对象进行少量的独立移动，但当对象之间的距离太大时将它们拉到一起。