Getting Started with Physics

Simulate gravity, acceleration, collision detection, or joints.

Overview

Although you can control the exact position of every node in a scene, often you want these nodes to interact with each other, colliding with each other and imparting velocity changes in the process. You might also want to do things that are not handled by the action system, such as simulating gravity and other forces. To do this, you create physics bodies (SKPhysicsBody) and attach them to nodes in your scene.

SpriteKit uses the International System of Units, also known as SI, or the meter-kilogram-second system. Where necessary, you may need to consult other reference materials online to learn more about the physics equations used by SpriteKit.

Customize Physics by Configuring Bodies

A physics body uses its node’s position and orientation to place itself in the simulation. Every physics body has other characteristics that define how the simulation operates on it. These include innate properties of the physical object, such as its mass and density, and properties imposed on it, such as its velocity. These characteristics define how a body moves, how it is affected by forces in the simulation, and how it responds to collisions with other physics bodies.

Interact with Position and Velocity

Each time a scene computes a new frame of animation, it simulates the effects of forces and collisions on physics bodies connected to the node tree. It computes a final position, orientation, and velocity for each physics body. Then, the scene updates the position and rotation of each corresponding node. Other forces can be applied automatically to multiple physics bodies by adding SKFieldNode objects to the scene. You can also directly affect a specific field body by modifying its velocity directly or by applying forces or impulses directly to it. The acceleration and velocity of each body is computed and the bodies collide with each other. Then, after the simulation is complete, the positions and rotations of the corresponding nodes are updated.

Control the Behavior of Individual Bodies

You have precise control over which physics effects interact with each other. For example, you can specify that a particular physics field node only affects a subset of the physics bodies in the scene. You also decide which physics bodies can collide with each other and separately decide which interactions cause your app to be called. You use these callbacks to add game logic. For example, your game might destroy a node when its physics body is struck by another physics body.

Decide on Physics Behaviors

To use physics in your game, you need to:

>> Attach physics bodies to nodes in the node tree and configure their physical properties. See SKPhysicsBody.

>> Define global characteristics of the scene's physics simulation, such as gravity. See SKPhysicsWorld.

>> Where necessary to support your gameplay, set the velocity of physics bodies in the scene or apply forces or impulses to them. See SKFieldNode and SKPhysicsBody.

>> Decide whether physics bodies in the scene should be connected with each other. See SKPhysicsJointand the pinned property on SKPhysicsBody.

>> Define how the physics bodies in the scene interact when they come in contact with each other. See SKPhysicsContactDelegate.

>> Optimize your physics simulation to limit the number of calculations it must perform.