Controlling Actions Precisely by Using Names

Set an action's name property so you can access it later without needing an instance variable.

Overview

Normally, you can't see which actions a node is executing, and if you want to remove actions, you must remove all of them. If you need to see whether a particular action is executing, or if you want to remove a specific action, use named actions. A named action is characterized by a unique string that identifies the action. Using named actions, you can start, remove, find and replace the actions of a node.

Create and Run a Named Action

The following code creates and runs a new action identified with the ignition key.

```
let moveNodeUp = SKAction.moveBy(x: 0.0,
                                 y: 100.0,
                                 duration: 1.0)
rocketNode.run(moveNodeUp,
               withKey: "ignition")
```

The following key-based methods are available:

- >> run(_:withKey:) method to run the action. If an action with the same key is already executing, it is removed before the new action is added.
- >> action(forKey:) method to determine if an action with that key is already running.
- >> removeAction(forKey:) method to remove the action.

Find and Replace an In-Flight Action

The code below shows how you use a name to replace an action that's currently running. The code determines where a mouse click occurred and then runs an action that moves a sprite to the click location. The action duration is calculated ahead of time to move the sprite at a fixed speed. Because this code uses `run(_:withKey:)`, any previous move that's still executing is stopped and replaced by the new action with the same name.

```
override func mouseDown(with event: NSEvent) {
     guard let playerNodeParent = self.playerNode.parent else { return }

     let clickPoint = event.location(in: playerNodeParent)

     let charPos = playerNode.position

     let distance = hypot(clickPoint.x-charPos.x,
          clickPoint.y-charPos.y)

     let moveToClick = SKAction.move(to: clickPoint,
          duration: TimeInterval(distance / characterSpeed))

     self.playerNode.run(moveToClick, withKey: "moveToClick")
}
```