

208: Scene Kit

Part 3: Lab Instructions

Copyright © 2015 Razeware LLC.

All rights reserved. No part of this book or corresponding materials (such as text, images, or source code) may be reproduced or distributed by any means without prior written per- mission of the copyright owner.

This book and all corresponding materials (such as source code) are provided on an "as is" basis, without warranty of any kind, express or implied, including but not limited to the warranties of merchantability, fitness for a particular purpose, and noninfringement. In no event shall the authors or copyright holders be liable for any claim, damages or other liability, whether in action of contract, tort or otherwise, arising from, out of or in connec- tion with the software or the use or other dealings in the software.

All trademarks and registered trademarks appearing in this book are the property of their respective owners.

# Thinking outside the box

Your frogger-like game is shaping up quite nicely at this point. You can move the “frog” around using swipes and taps, and you can complete the level.

However, unless you have a very good imagination, a box bears little resemblance to a frog. So, time to kiss the box and turn it into a handsome voxelized frog.

## Piña.collada

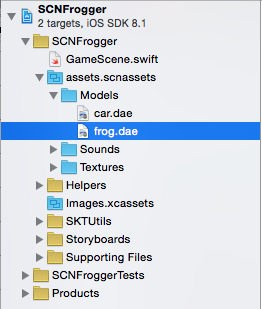
Scene Kit is able to load 3D assets from COLLADA (.dae) files. COLLADA is an open format for exchanging 3D contents and is supported by all the most popular 3D modeling tools on the market.

COLLADA files are XML files that define the node graph and geometries, animations, textures, lighting, cameras etc for a scene. A scene could be a single 3D model but could also be a full level. Therefore, having a tool to inspect the COLLADA file will be very handy.

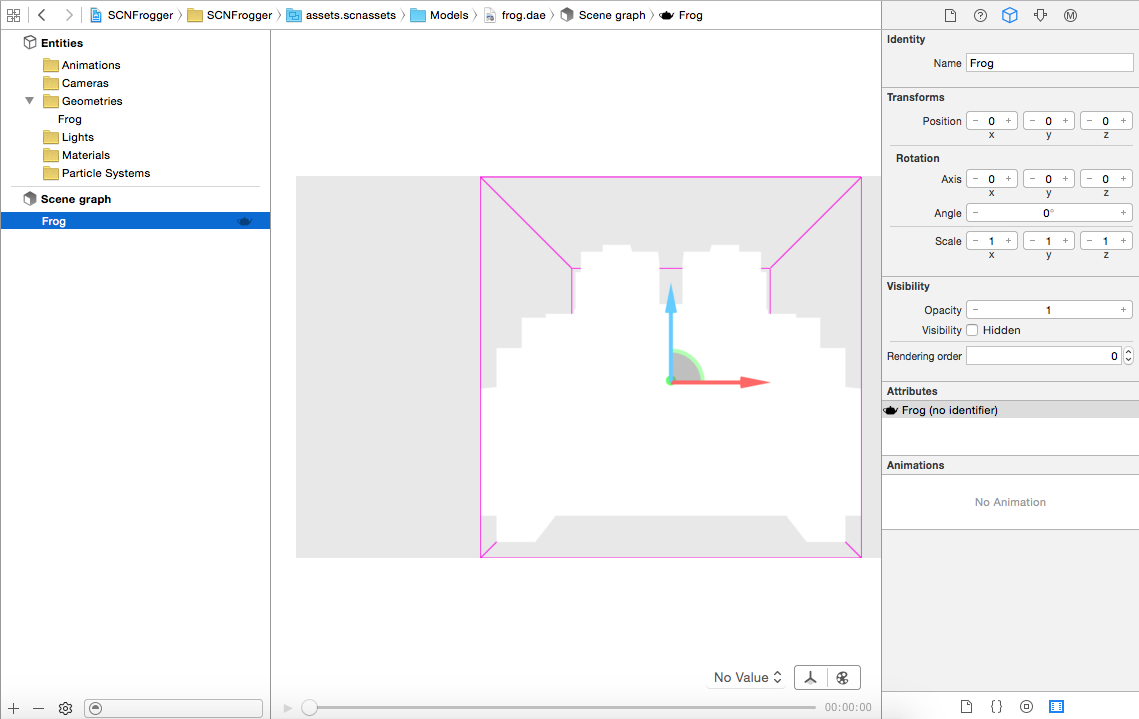
Thankfully, Apple has built such a tool right into Xcode: The SceneKit Editor. You will spend the first few minutes in this lab getting familiar with it.

## SceneKit Editor

You activate the Scene Kit Editor by selecting a COLLADA file in the Project navigator.

In the Project Navigator, navigate to **assets.scnassets/Models** and select **frog.dae**.

You will now see the following window:

  
On the left, you have a list of **Entities** in the file (animations, cameras, geometries etc). This can be used for easily getting an overview of the unique entities in the scene. The **frog.dae** file only has one entity: The geometry for **Frog**.

Below the list of entities is the **Scene Graph**. This part is important for two reasons:

1. When you load a scene programmatically, the scene graph will define all the nodes in the scene. The name of these nodes is what you will use when referring to the nodes from code. You will learn more about this shortly.
2. You can change the Scene Graph from within the SceneKit Editor without having to open the file in a 3D modeling tool. This is handy if you need to make small adjustments to the scene graph – for instance, changing the rotation of a node so the node faces in the correct direction.

Select **Frog** in the Scene Graph if it is not already selected. On the right, you should now see the **Utilities** window. The Utilities window has five icons on the top:

From left to right these are:

* File inspector
* Quick help inspector
* Node inspector
* Attributes inspector
* Material inspector

The File and Quick help inspectors you are probably already familiar with if you have used Xcode for some time. What is interesting here are the three other inspectors.

* **Node inspector:** Will allow you to change the properties of a node. This is handy if you want to ensure a node is rotated to face in a specific direction when the scene I loaded.
* **Attributes inspector:** Will show useful information about the node. For instance, if the node has geometry you can see how many vertices and polygons are in the geometry.
* **Material inspector:** Will allow you to view or change the material assigned to the node geometry.

Now that you have basic knowledge about how to inspect a COLLADA file your next task is to load the frog.dae scene and assign the Frog node to the player instead of the box.

## Loading scenes

You’ll add a new property to the top of **GameScene** for the **frog.dae** scene:

let playerScene = SCNScene(named: "assets.scnassets/Models/frog.dae")

This will load the **frog.dae** scene into the property **playerScene** every time a new **GameScene** object is created.

Next, go to the createPlayerAtPosition(…) method and change the line playerModelNode = SCNNode() to:

let playerModelNode = playerScene!.rootNode.childNodeWithName(

"Frog", recursively: false)!

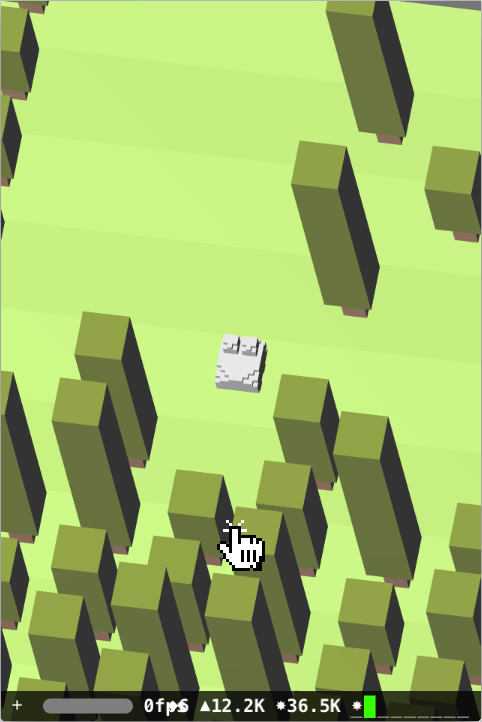
This will get the “Frog” child node of the playerScene scene and assign the node to playerModelNode. You set recursively to false as the Frog node is the only child of the rootNode. Try inspecting the scene graph of the **frog.dae** file with the SceneKit editor in case you are in doubt.

**Note:** When you reference the “Frog” node in the playerScene this way, you are referencing the node in the scene directly. This is OK in this case as we only have one frog in our game. If you need to use the same node several times, you might need to create a clone like so:

let playerNode = playerScene!.rootNode.childNodeWithName(“Frog”, recursively: false)!**.clone()** as SCNNode

This will recursively copy the node and it’s child nodes. Remember this for the challenge :]

Build and run, and your box is now a nice – but boring – voxel frog.



Why is the frog gray when we all know that frogs are green? There are actually two reasons for this.

First, the createPlayerAtPosition(…) method creates a material with a gray diffuse color and assigns it to the playerModelNode geometry.

Second, if you remember, the **frog.dae** file contained no materials. Hence, if a material had not been assigned in createPlayerAtPosition(…) the frog would just have been white like you see in the SceneKit editor when inspecting the file.

Your next step is to create a material from a texture and assign this to the frog instead of the dull grey material it has now.

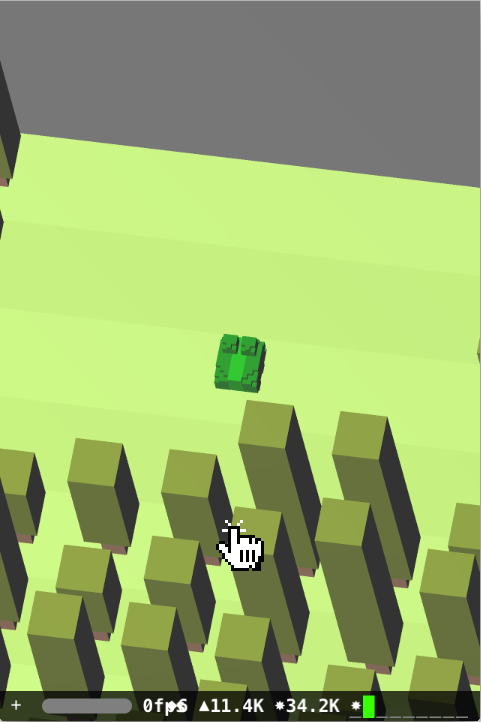
## Materials and textures

Materials in Scene Kit can use textures loaded from image files as well as colors. In fact, you could even use a video as a texture. That is how versatile Scene Kit materials are :]

You will load a texture from a file in this lab. Go to createPlayerAtPosition(…) and replace the line playerMaterial.diffuse.contents = UIColor(red: 225.0/255.0, green: 225.0/255.0, blue: 225.0/255.0, alpha: 1.0) with the following:

let playerMaterial.diffuse.contents = UIImage(named: "assets.scnassets/Textures/model\_texture.tga")

This will load an image into the contents of the diffuse property of the material. When rendering the frog, the image will be used instead of the color that was previously set.

Do another build and run. The frog is now green as a frog should be:

Materials are very powerful and allow you to change the appearance of your scene drastically thanks to the power of Scene Kit. Behind the scenes, shaders take care of everything for you. Shaders are beyond the scope of this tutorial but expect to find a more advanced tutorial on Scene Kit on our website in the future.

## End of the lab

Congratulations, you now know how to load 3D scenes into your Scene Kit game and how to load textures into materials. In the challenge, you will use that knowledge to add cars onto the roads for the frog to avoid.