Create a new Swift file named **Texturable.swift**, and include it in both the macOS and iOS targets. Replace the code with:

```
import MetalKit

protocol Texturable {}

extension Texturable {
}
```

Inside the protocol extension, add a default method:

```
static func loadTexture(imageName: String) throws -> MTLTexture?
{
  // 1
  let textureLoader = MTKTextureLoader(device: Renderer.device)

  // 2
  let textureLoaderOptions: [MTKTextureLoader.Option: Any] =
            [.origin: MTKTextureLoader.Origin.bottomLeft]
  // 3
  let fileExtension =
    URL(fileURLWithPath: imageName).pathExtension.isEmpty ?
      "png" : nil
  // 4
  guard let url = Bundle.main.url(forResource: imageName,
                                  withExtension: fileExtension)
    else {
      print("Failed to load \(imageName)")
      return nil
  }

  let texture =
    try textureLoader.newTexture(URL: url,
                                 options: textureLoaderOptions)
  print("loaded texture: \(url.lastPathComponent)")
  return texture
}
```

Going through the code:

1. Loading textures can get complicated. When Metal was first released, you had to specify everything about the image — including pixel format, dimensions and usage — using `MTLTextureDescriptor`. MetalKit introduced `MTKTextureLoader` which provides defaults you can optionally change using loading options.

2. Here, you change a loading option to ensure that the texture loads with the origin at the bottom-left. If you don't specify this option, the texture will be flipped. Try it later: `lowpoly-house-color.png` is almost vertically symmetrical, but with