The requirements according to Autodesk are to be using Xcode 5.0.2 with the 10.8 SDK(comes with it). Xcode 6 does not have this SDK.

Unfortunately, when Apple updates Xcode it breaks everything, every time. Xcode 5.0.2 is not available for download through Apple now that Xcode 6.0.1 is out. However, 10 seconds of internet searching led me to the download for it. I'm not going to give out links for the legal protection of RD, but if you're the least bit internet savvy, you'll be able to get it.

That being said, I compiled a few plugins on Xcode 6.0.1 and did not have a problem, and I used the 10.9 SDK. As it is not officially supported through Autodesk, use at your own risk.

What will follow will be showing you how to set up Xcode to work with Maya and get to a compiled plugin. In addition, I will discuss some differences between Marco's setup in Visual Studio vs. the Xcode template setup and what you can do.

Disclaimer: I do not take any credit for the files in the DMG. They are all property of Autodesk and Cyrille Fauvel who was generous enough to update the Xcode files when I sent him an e-mail asking a few questions about getting set up. If you get this working, *please* send Cyrille an e-mail thanking him. It is only because of him that this exists.

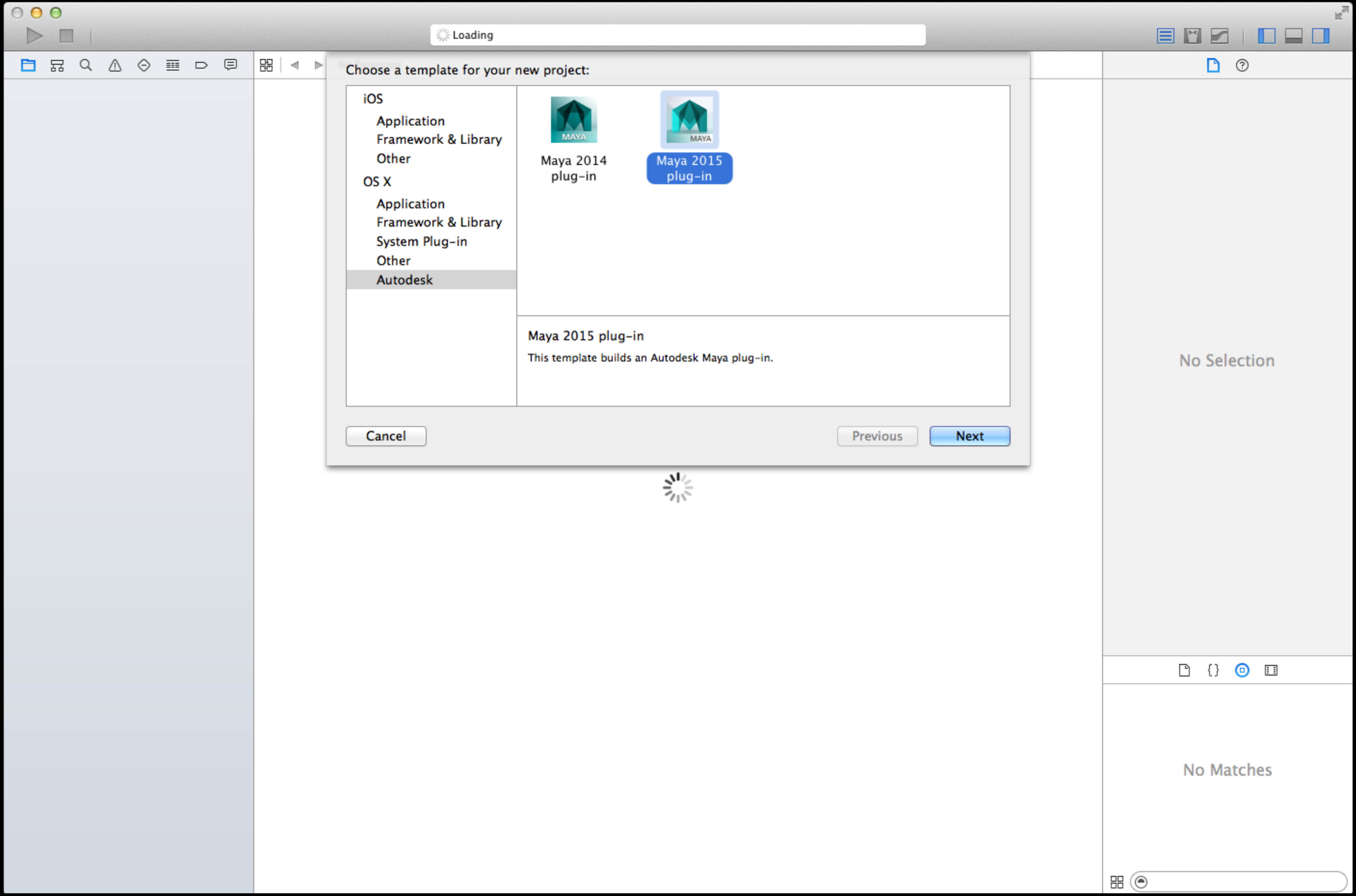
You can visit his website at around-the-corner.typepad.com

1. Make sure Xcode is installed. I’m not going to go over everything for Xcode, but I will go over a few things later. For now, this will focus on setup of everything.
2. In the DMG, there is a folder that says ‘Templates’. Double-click this folder and it should take you to this path: /Applications/Xcode.app/Contents/Developer/Library/Xcode/Templates
3. In that folder, create two more folders:
   File Templates
   Project Templates
   Make sure you have a space and the correct capitalization.
4. In the File Templates folder, create a folder called Maya
   In the Project Templates folder, create a folder called Mac and a folder inside of that called Autodesk
   You should now have these paths:
   /Applications/Xcode.app/Contents/Developer/Library/Xcode/Templates/File Templates/Maya
   /Applications/Xcode.app/Contents/Developer/Library/Xcode/Templates/Project Templates/Mac/Autodesk
5. In the DMG, drag/copy Maya 2014.xctemplate and Maya 2015.xctemplate into Project Templates/Mac/Autodesk
6. In the DMG, drag/copy MayaCommand.xctemplate and MayaNode.xctemplate and MayaNode Any.xctemplate into File Templates/Maya

That’s the installation part. Up next comes the usage.

Boot up Xcode and you will see: nothing.

That's because Xcode doesn't use a splash screen or background or anything. Go to your menu bar(after selecting the Xcode icon on your dash) and go to File > New > Project. Under OS X, you should see an Autodesk menu and see one or two of these icons.

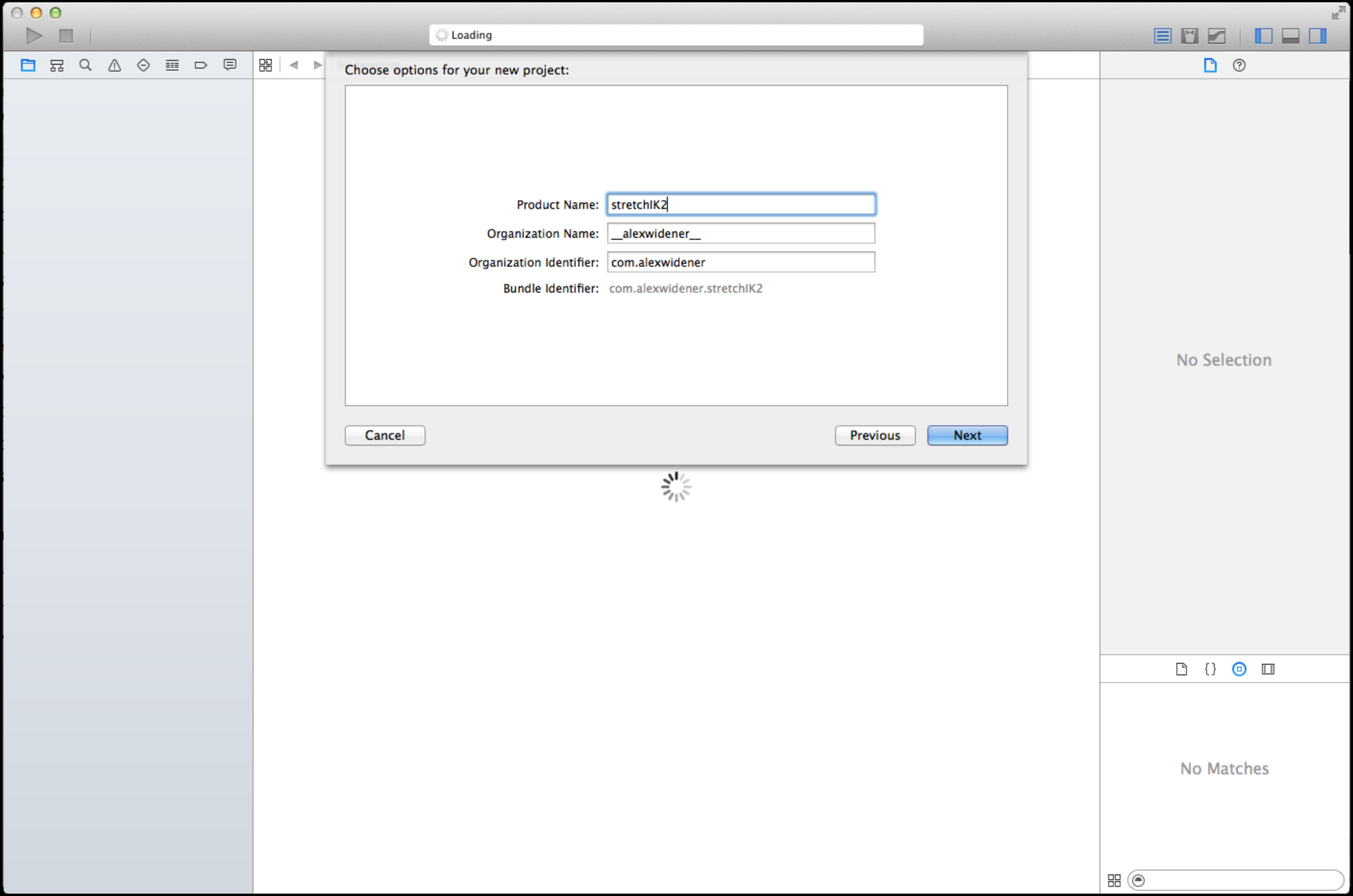


Choose a template for your new project:

Maya 2015 plug-in

This template builds an Autodesk Maya plug-in.

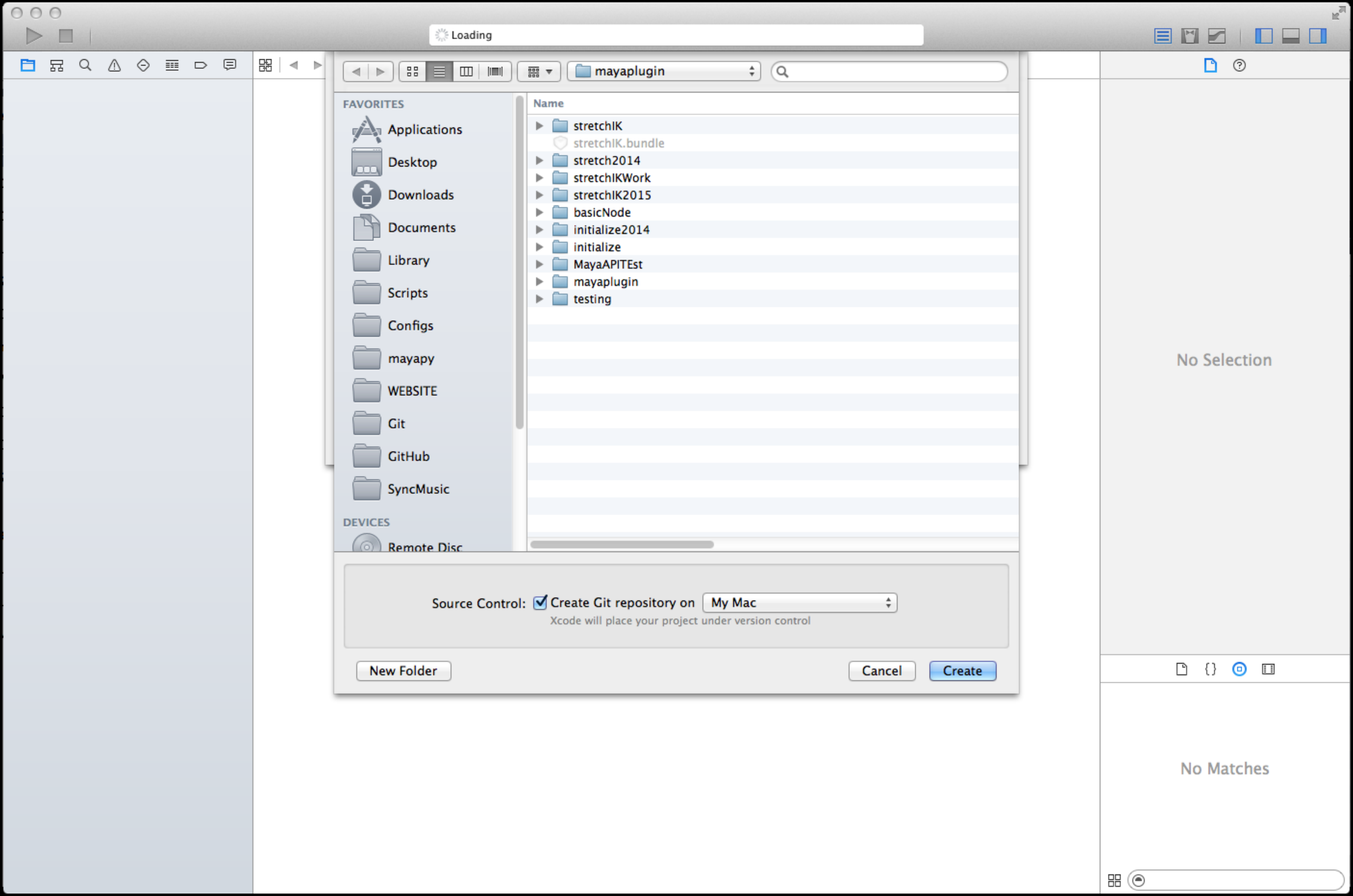
Go to the next page and enter what you would like for the product name(I name it the same as the node). I don’t think Organization Name and Organization Identifier autofill for people without Apple Developer accounts, so you might have to fill that in yourself. Just put your name, it doesn’t matter.



Choose options for your new project:

Product Name:	stretchIK2
Organization Name:	__alexwidener__
Organization Identifier:	com.alexwidener
Bundle Identifier:	com.alexwidener.stretchIK2

On the next screen, choose where you would like your plugin to be saved. I don't recommend doing the default Xcode place, because stuff gets weird with that. I'll explain on another slide. Also, if you're not using Git, uncheck the box at the bottom. If you don't know what it is, look it up. It's amazing. Press Create.



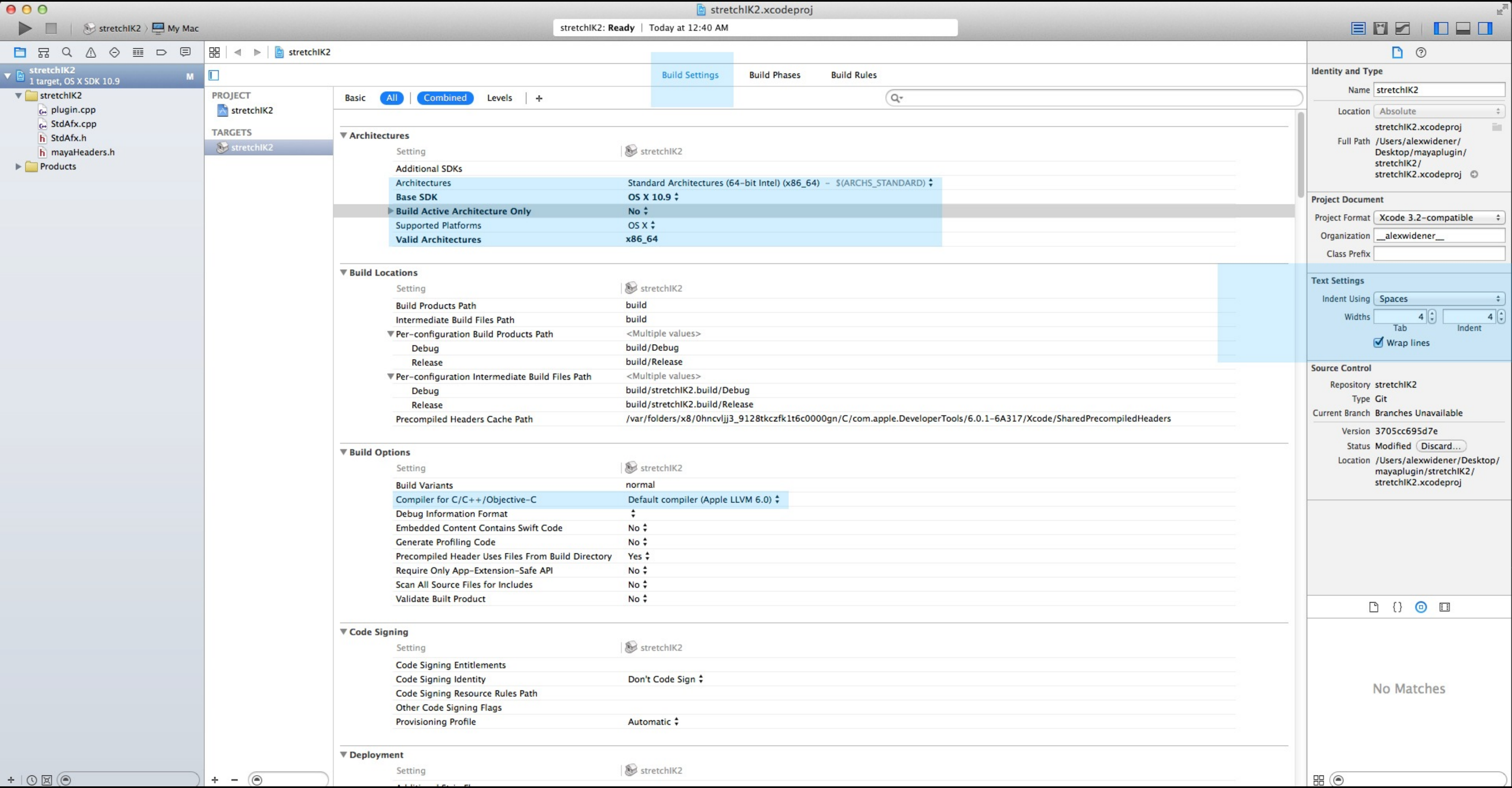
Source Control: ☑ Create Git repository on My Mac

Xcode will place your project under version control

And now we get to the ugly stuff. All the beauty of OS X and none of it came here. I recommend going through an Xcode tutorial to learn this, but for the time being, there's only a few things you need to pay attention to(blue boxes). I'm using Xcode 6.0.1 so my Base SDK is set to 10.9 - but if you're using Xcode 5.0.2 set your SDK to 10.8.

Also: make sure you set your Compiler, it tends to not always be set(lowest blue box).

If you'll notice: at the top middle, there's a blue box with "Build Settings" that means there are other parts here(Build Phases and Build Rules), more specific stuff. Don't worry about it now, just be aware it exists. Don't leave this page yet in Xcode. There are more settings to check first.



▼ Architectures

Setting	stretchIK2
Additional SDKs	
Architectures	Standard Architectures (64-bit Intel) (x86_64) – $(ARCHS_STANDARD)
Base SDK	OS X 10.9
▶Build Active Architecture Only	No
Supported Platforms	OS X
Valid Architectures	x86_64

▼ Build Locations

Setting	stretchIK2
Build Products Path	build
Intermediate Build Files Path	build
▼Per-configuration Build Products Path	<Multiple values>
Debug	build/Debug
Release	build/Release
▼Per-configuration Intermediate Build Files Path	<Multiple values>
Debug	build/stretchIK2.build/Debug
Release	build/stretchIK2.build/Release
Precompiled Headers Cache Path	/var/folders/x8/0hncvljj3_9128tkczfk1t6c0000gn/C/com.apple.DeveloperTools/6.0.1-6A317/Xcode/SharedPrecompiledHeaders

▼ Build Options

Setting	stretchIK2
Build Variants	normal
Compiler for C/C++/Objective-C	Default compiler (Apple LLVM 6.0)
Debug Information Format	
Embedded Content Contains Swift Code	No
Generate Profiling Code	No
Precompiled Header Uses Files From Build Directory	Yes
Require Only App-Extension-Safe API	No
Scan All Source Files for Includes	No
Validate Built Product	No

▼ Code Signing

Setting	stretchIK2
Code Signing Entitlements	
Code Signing Identity	Don't Code Sign
Code Signing Resource Rules Path	
Other Code Signing Flags	
Provisioning Profile	Automatic

▼ Deployment

Setting	stretchIK2

Identity and Type

Name: stretchIK2

Location: Absolute

stretchIK2.xcodeproj

Full Path: /Users/alexwidener/Desktop/mayaplugin/stretchIK2/stretchIK2.xcodeproj

Project Document

Project Format: Xcode 3.2-compatible

Organization: __alexwidener__

Class Prefix:

Text Settings

Indent Using: Spaces

Widths: 4 Tab, 4 Indent

☑ Wrap lines

Source Control

Repository: stretchIK2

Type: Git

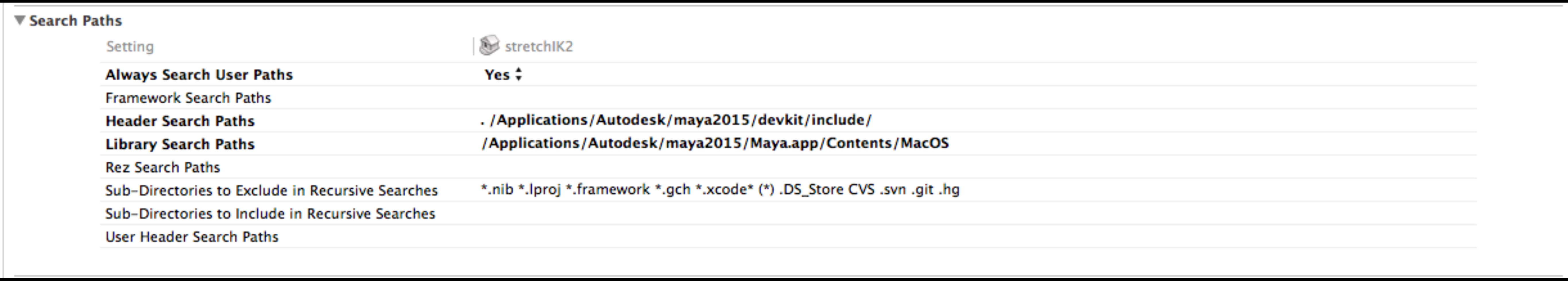
Current Branch: Branches Unavailable

Version: 3705cc695d7e

Status: Modified (Discard...)

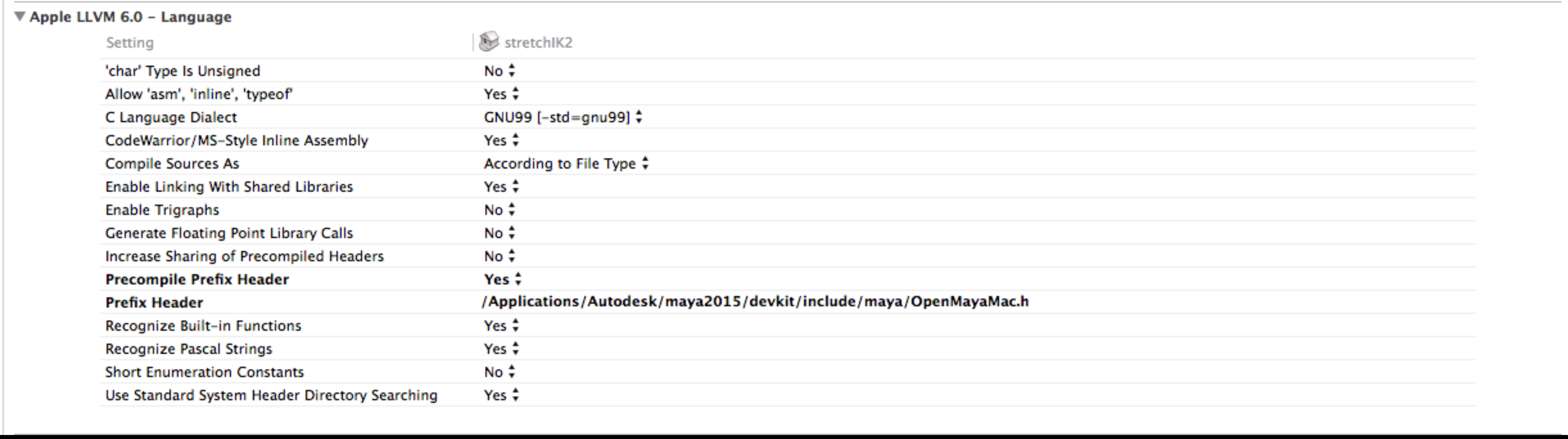
Location: /Users/alexwidener/Desktop/mayaplugin/stretchIK2/stretchIK2.xcodeproj

These are all on the same page as before, underneath “Build Settings”. You shouldn’t have to change them since you’re using the project, but you should be aware of them in case anything goes awry.



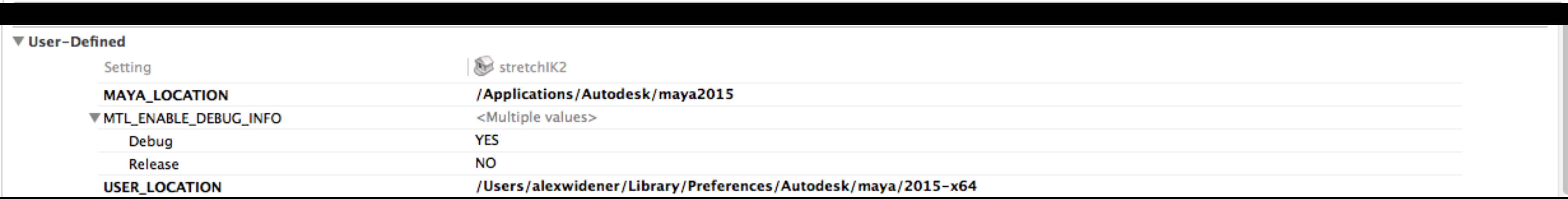
▼ Search Paths

Setting	stretchIK2
Always Search User Paths	**Yes** ⬍
Framework Search Paths	
Header Search Paths	**. /Applications/Autodesk/maya2015/devkit/include/**
Library Search Paths	**/Applications/Autodesk/maya2015/Maya.app/Contents/MacOS**
Rez Search Paths	
Sub-Directories to Exclude in Recursive Searches	*.nib *.lproj *.framework *.gch *.xcode* (*) .DS_Store CVS .svn .git .hg
Sub-Directories to Include in Recursive Searches	
User Header Search Paths	



▼ Apple LLVM 6.0 - Language

Setting	stretchIK2
'char' Type Is Unsigned	No ⬍
Allow 'asm', 'inline', 'typeof'	Yes ⬍
C Language Dialect	GNU99 [-std=gnu99] ⬍
CodeWarrior/MS-Style Inline Assembly	Yes ⬍
Compile Sources As	According to File Type ⬍
Enable Linking With Shared Libraries	Yes ⬍
Enable Trigraphs	No ⬍
Generate Floating Point Library Calls	No ⬍
Increase Sharing of Precompiled Headers	No ⬍
Precompile Prefix Header	**Yes** ⬍
Prefix Header	**/Applications/Autodesk/maya2015/devkit/include/maya/OpenMayaMac.h**
Recognize Built-in Functions	Yes ⬍
Recognize Pascal Strings	Yes ⬍
Short Enumeration Constants	No ⬍
Use Standard System Header Directory Searching	Yes ⬍

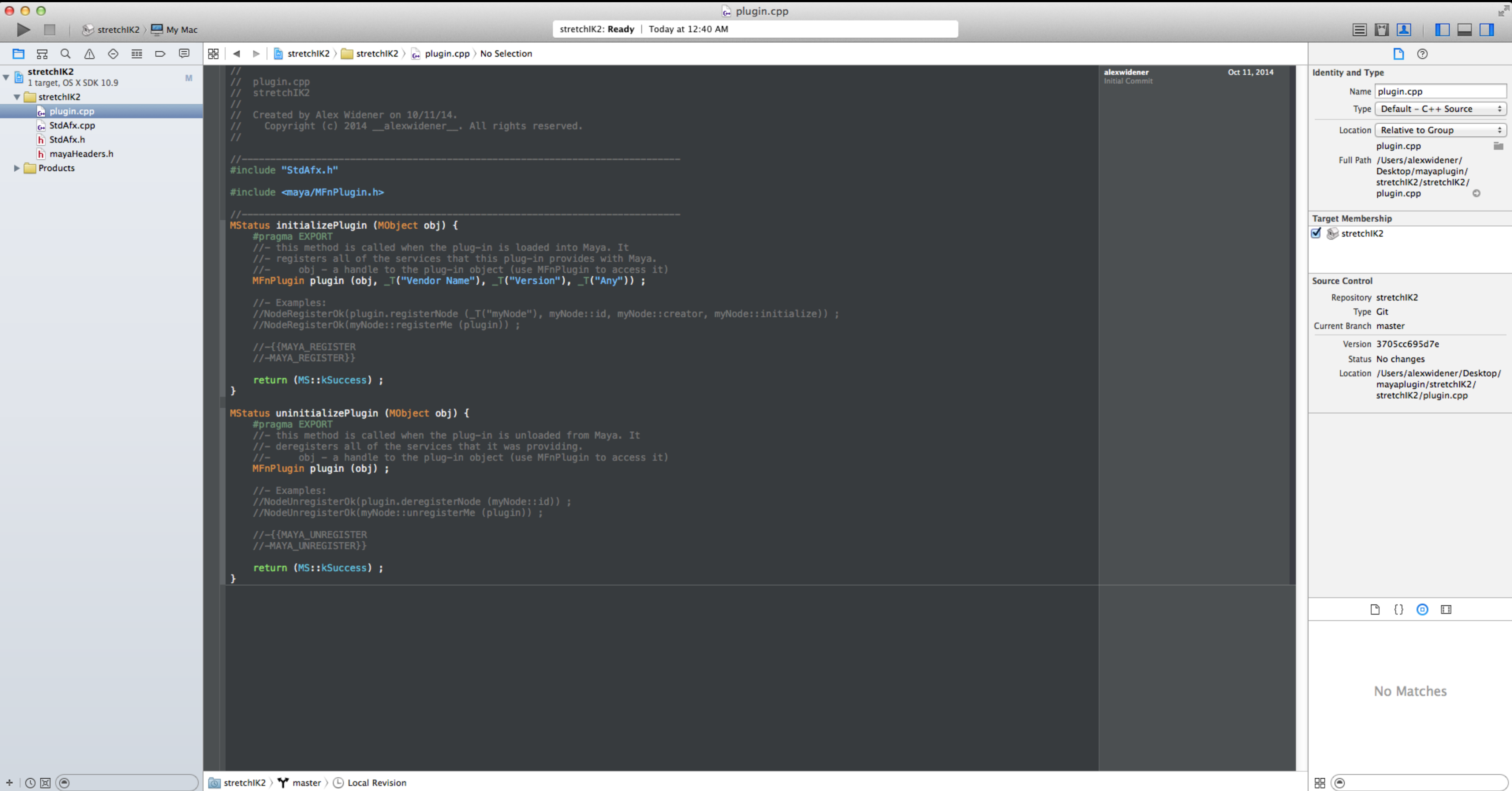


▼ User-Defined

Setting	stretchIK2
MAYA_LOCATION	**/Applications/Autodesk/maya2015**
▼ MTL_ENABLE_DEBUG_INFO	<Multiple values>
Debug	YES
Release	NO
USER_LOCATION	**/Users/alexwidener/Library/Preferences/Autodesk/maya/2015-x64**

So on the left is your Project Browser. The reason all the stuff was showing up before is because you had the entire project selected, so you were looking at the project settings. Now if you'll click on any of the files underneath the project(stretchIK2 in my images), you can see it switches to the Text Editor. You'll also see some files in there that are generated by the template. LEAVE THEM ALL. You don't have to use them, just leave them in there. You need to use plugin.cpp(which Marco calls pluginMain.cpp) but just leave the others.

Notice the bar on the right. You don't need this for now, but might later. It helps me to get rid of it, I like to code in clean spaces. Hit the button closest to the arrow in the top right corner.



```
//
//  plugin.cpp
//  stretchIK2
//
//  Created by Alex Widener on 10/11/14.
//    Copyright (c) 2014 __alexwidener__. All rights reserved.
//

//-----------------------------------------------------------------------------
#include "StdAfx.h"

#include <maya/MFnPlugin.h>

//-----------------------------------------------------------------------------
MStatus initializePlugin (MObject obj) {
    #pragma EXPORT
    //- this method is called when the plug-in is loaded into Maya. It
    //- registers all of the services that this plug-in provides with Maya.
    //-     obj - a handle to the plug-in object (use MFnPlugin to access it)
    MFnPlugin plugin (obj, _T("Vendor Name"), _T("Version"), _T("Any")) ;

    //- Examples:
    //NodeRegisterOk(plugin.registerNode (_T("myNode"), myNode::id, myNode::creator, myNode::initialize)) ;
    //NodeRegisterOk(myNode::registerMe (plugin)) ;

    //-{{MAYA_REGISTER
    //-MAYA_REGISTER}}

    return (MS::kSuccess) ;
}

MStatus uninitializePlugin (MObject obj) {
    #pragma EXPORT
    //- this method is called when the plug-in is unloaded from Maya. It
    //- deregisters all of the services that it was providing.
    //-     obj - a handle to the plug-in object (use MFnPlugin to access it)
    MFnPlugin plugin (obj) ;

    //- Examples:
    //NodeUnregisterOk(plugin.deregisterNode (myNode::id)) ;
    //NodeUnregisterOk(myNode::unregisterMe (plugin)) ;

    //-{{MAYA_UNREGISTER
    //-MAYA_UNREGISTER}}

    return (MS::kSuccess) ;
}
```

Identity and Type

Name: plugin.cpp
Type: Default - C++ Source
Location: Relative to Group
plugin.cpp
Full Path: /Users/alexwidener/Desktop/mayaplugin/stretchIK2/stretchIK2/plugin.cpp

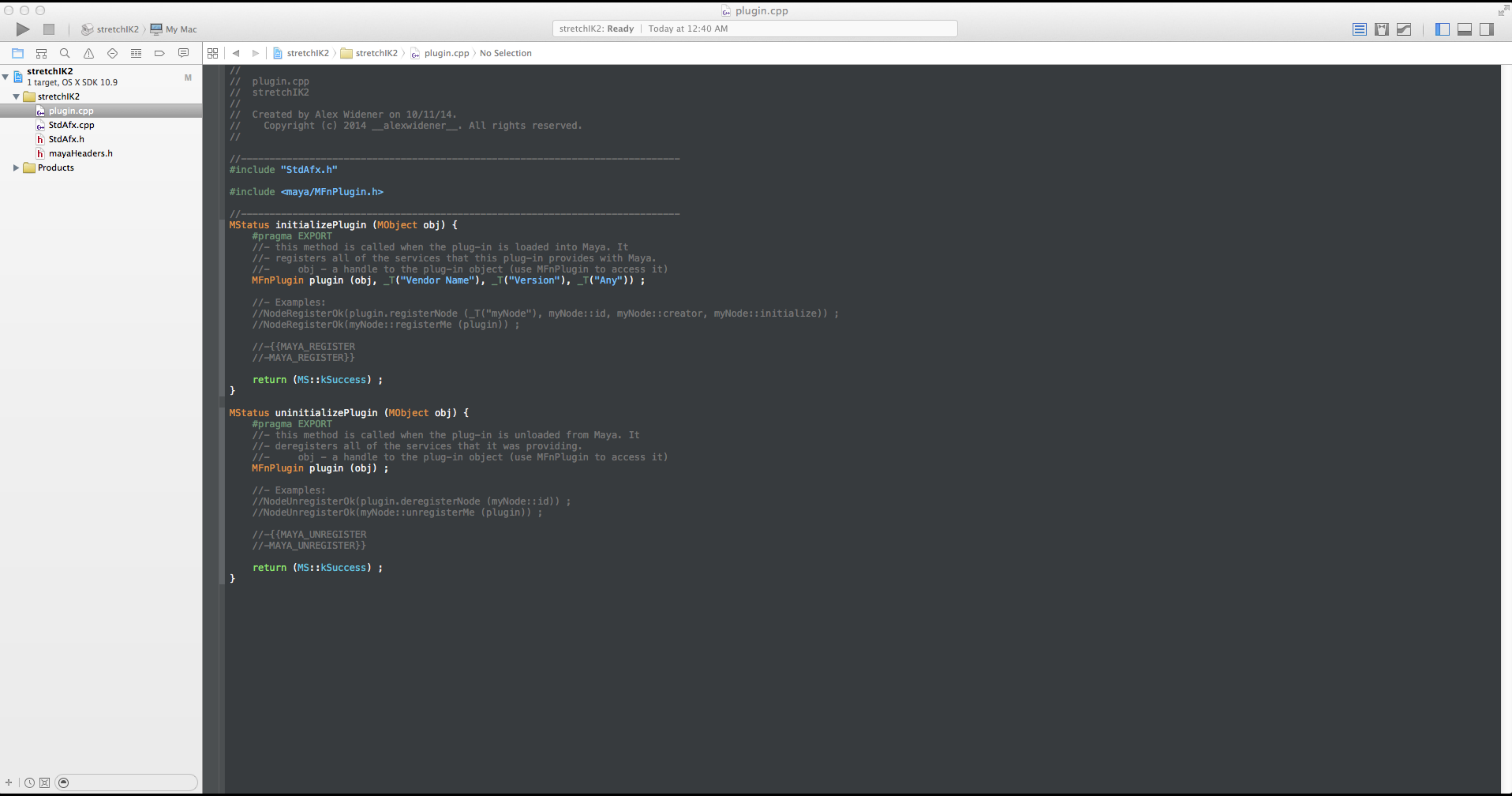
Target Membership

☑ stretchIK2

Source Control

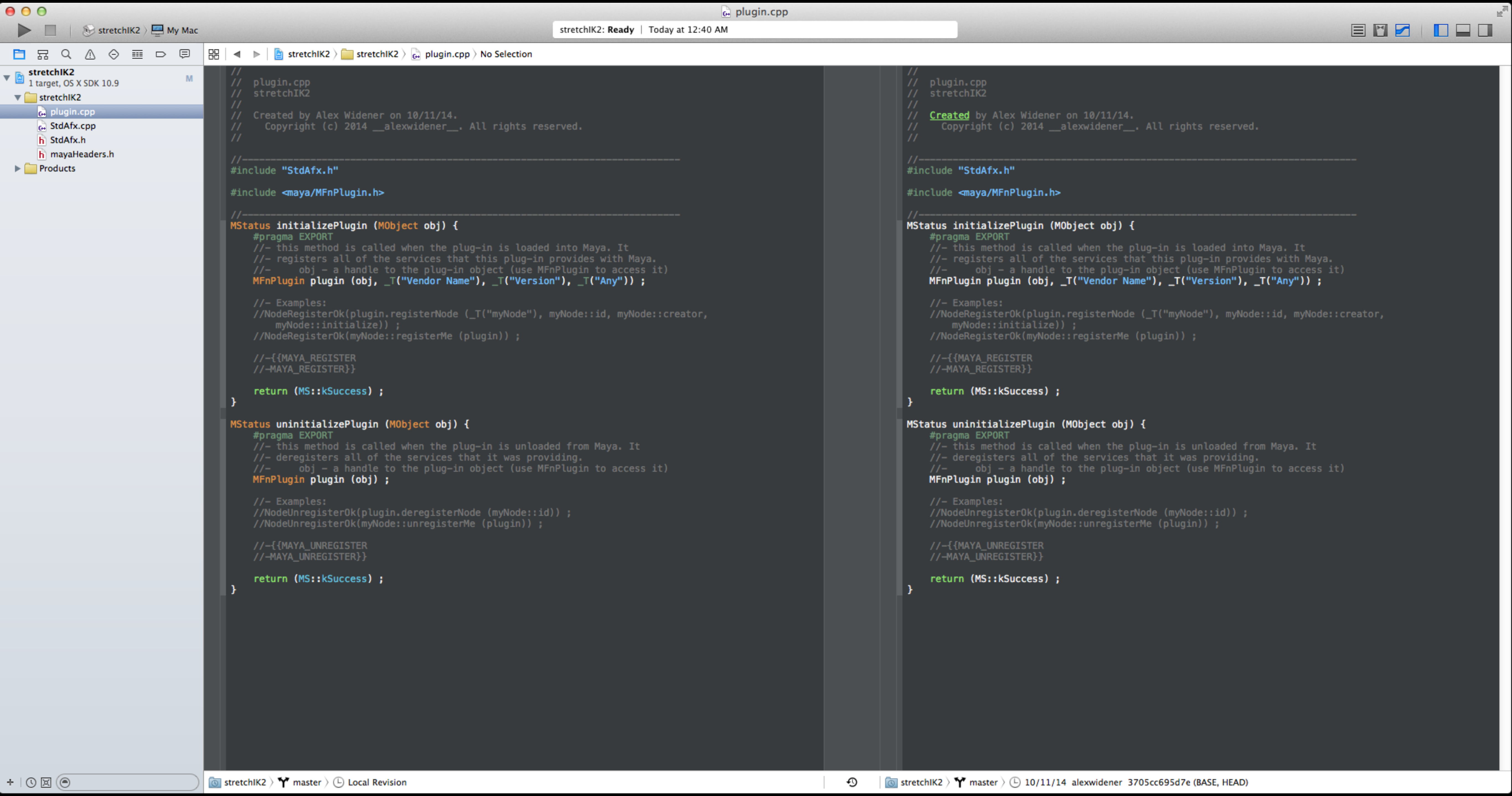
Repository: stretchIK2
Type: Git
Current Branch: master
Version: 3705cc695d7e
Status: No changes
Location: /Users/alexwidener/Desktop/mayaplugin/stretchIK2/stretchIK2/plugin.cpp

That leaves you with this, much cleaner.



```
//
//  plugin.cpp
//  stretchIK2
//
//  Created by Alex Widener on 10/11/14.
//    Copyright (c) 2014 __alexwidener__. All rights reserved.
//

//-----------------------------------------------------------------------------
#include "StdAfx.h"

#include <maya/MFnPlugin.h>

//-----------------------------------------------------------------------------
MStatus initializePlugin (MObject obj) {
    #pragma EXPORT
    //- this method is called when the plug-in is loaded into Maya. It
    //- registers all of the services that this plug-in provides with Maya.
    //-     obj - a handle to the plug-in object (use MFnPlugin to access it)
    MFnPlugin plugin (obj, _T("Vendor Name"), _T("Version"), _T("Any")) ;

    //- Examples:
    //NodeRegisterOk(plugin.registerNode (_T("myNode"), myNode::id, myNode::creator, myNode::initialize)) ;
    //NodeRegisterOk(myNode::registerMe (plugin)) ;

    //-{{MAYA_REGISTER
    //-MAYA_REGISTER}}

    return (MS::kSuccess) ;
}

MStatus uninitializePlugin (MObject obj) {
    #pragma EXPORT
    //- this method is called when the plug-in is unloaded from Maya. It
    //- deregisters all of the services that it was providing.
    //-     obj - a handle to the plug-in object (use MFnPlugin to access it)
    MFnPlugin plugin (obj) ;

    //- Examples:
    //NodeUnregisterOk(plugin.deregisterNode (myNode::id)) ;
    //NodeUnregisterOk(myNode::unregisterMe (plugin)) ;

    //-{{MAYA_UNREGISTER
    //-MAYA_UNREGISTER}}

    return (MS::kSuccess) ;
}
```

There's another useful feature up in the top right. Notice the icon that looks like a hill. If you press that, it's a version comparison system, which(if you're running Git), you can use to compare files that you created in the past to what you're currently working on. Very helpful if something has changed and messed stuff up and you don't know what happened.

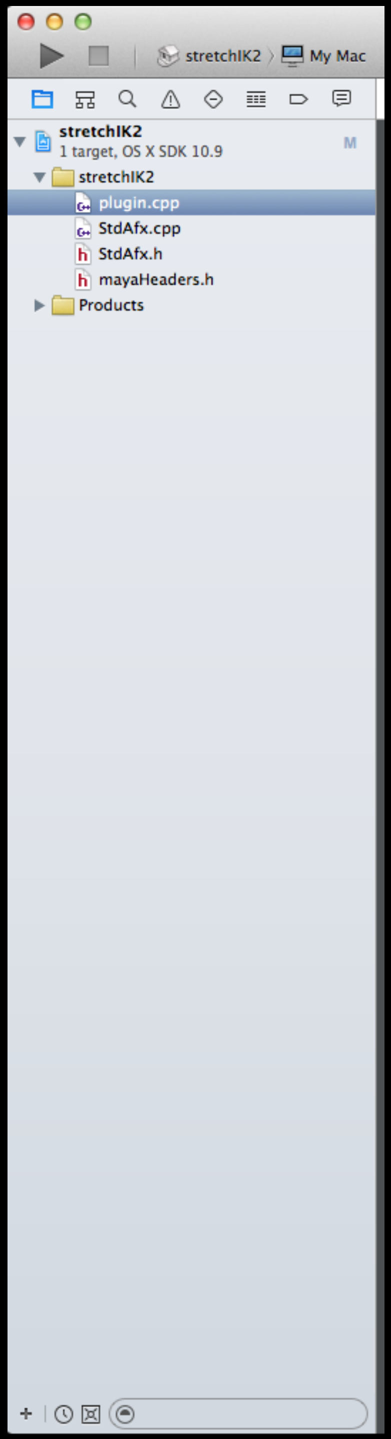


```
//
//  plugin.cpp
//  stretchIK2
//
//  Created by Alex Widener on 10/11/14.
//    Copyright (c) 2014 __alexwidener__. All rights reserved.
//

//-----------------------------------------------------------------------------
#include "StdAfx.h"

#include <maya/MFnPlugin.h>

//-----------------------------------------------------------------------------
MStatus initializePlugin (MObject obj) {
    #pragma EXPORT
    //- this method is called when the plug-in is loaded into Maya. It
    //- registers all of the services that this plug-in provides with Maya.
    //-     obj - a handle to the plug-in object (use MFnPlugin to access it)
    MFnPlugin plugin (obj, _T("Vendor Name"), _T("Version"), _T("Any")) ;

    //- Examples:
    //NodeRegisterOk(plugin.registerNode (_T("myNode"), myNode::id, myNode::creator,
        myNode::initialize)) ;
    //NodeRegisterOk(myNode::registerMe (plugin)) ;

    //-{{MAYA_REGISTER
    //-MAYA_REGISTER}}

    return (MS::kSuccess) ;
}

MStatus uninitializePlugin (MObject obj) {
    #pragma EXPORT
    //- this method is called when the plug-in is unloaded from Maya. It
    //- deregisters all of the services that it was providing.
    //-     obj - a handle to the plug-in object (use MFnPlugin to access it)
    MFnPlugin plugin (obj) ;

    //- Examples:
    //NodeUnregisterOk(plugin.deregisterNode (myNode::id)) ;
    //NodeUnregisterOk(myNode::unregisterMe (plugin)) ;

    //-{{MAYA_UNREGISTER
    //-MAYA_UNREGISTER}}

    return (MS::kSuccess) ;
}
```

```
//
//  plugin.cpp
//  stretchIK2
//
//  Created by Alex Widener on 10/11/14.
//    Copyright (c) 2014 __alexwidener__. All rights reserved.
//

//-----------------------------------------------------------------------------
#include "StdAfx.h"

#include <maya/MFnPlugin.h>

//-----------------------------------------------------------------------------
MStatus initializePlugin (MObject obj) {
    #pragma EXPORT
    //- this method is called when the plug-in is loaded into Maya. It
    //- registers all of the services that this plug-in provides with Maya.
    //-     obj - a handle to the plug-in object (use MFnPlugin to access it)
    MFnPlugin plugin (obj, _T("Vendor Name"), _T("Version"), _T("Any")) ;

    //- Examples:
    //NodeRegisterOk(plugin.registerNode (_T("myNode"), myNode::id, myNode::creator,
        myNode::initialize)) ;
    //NodeRegisterOk(myNode::registerMe (plugin)) ;

    //-{{MAYA_REGISTER
    //-MAYA_REGISTER}}

    return (MS::kSuccess) ;
}

MStatus uninitializePlugin (MObject obj) {
    #pragma EXPORT
    //- this method is called when the plug-in is unloaded from Maya. It
    //- deregisters all of the services that it was providing.
    //-     obj - a handle to the plug-in object (use MFnPlugin to access it)
    MFnPlugin plugin (obj) ;

    //- Examples:
    //NodeUnregisterOk(plugin.deregisterNode (myNode::id)) ;
    //NodeUnregisterOk(myNode::unregisterMe (plugin)) ;

    //-{{MAYA_UNREGISTER
    //-MAYA_UNREGISTER}}

    return (MS::kSuccess) ;
}
```

stretchIK2 › master › Local Revision
stretchIK2 › master › 10/11/14 alexwidener 3705cc695d7e (BASE, HEAD)

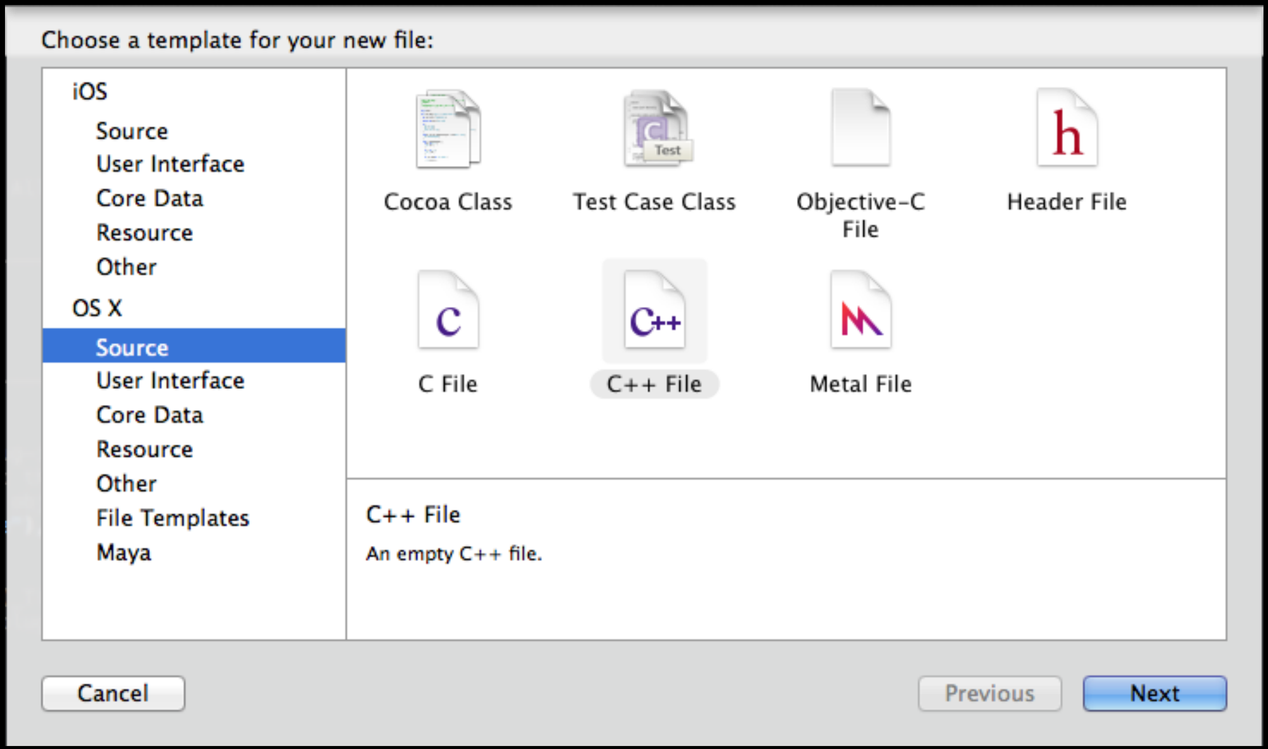
The next thing to pay attention to is the bar on the left. Keep this one open(although you can close it as needed with the buttons in the top right). It's your project structure and it's pretty similar to what you'll see in Marco's Visual Studio setup. The Folders that you see are not called folders, they're called Groups, and if you right-click on anything in there, you can see "New Group" - these Groups do not exist on disk, they only exist within the project. You can structure your stuff how you see fit. I prefer to structure with headers in one group and Source Files in another group, each named respectively.



This is called your Navigator, and all of those icons on the top bar there are different Navigators: (in order)
Project Navigator
Symbol Navigator
Find Navigator
Issue Navigator(bugs)
Test Navigator
Debug Navigator
Breakpoint Navigator
Report Navigator

You'll spend most time in the Project one, going back and forth between files.

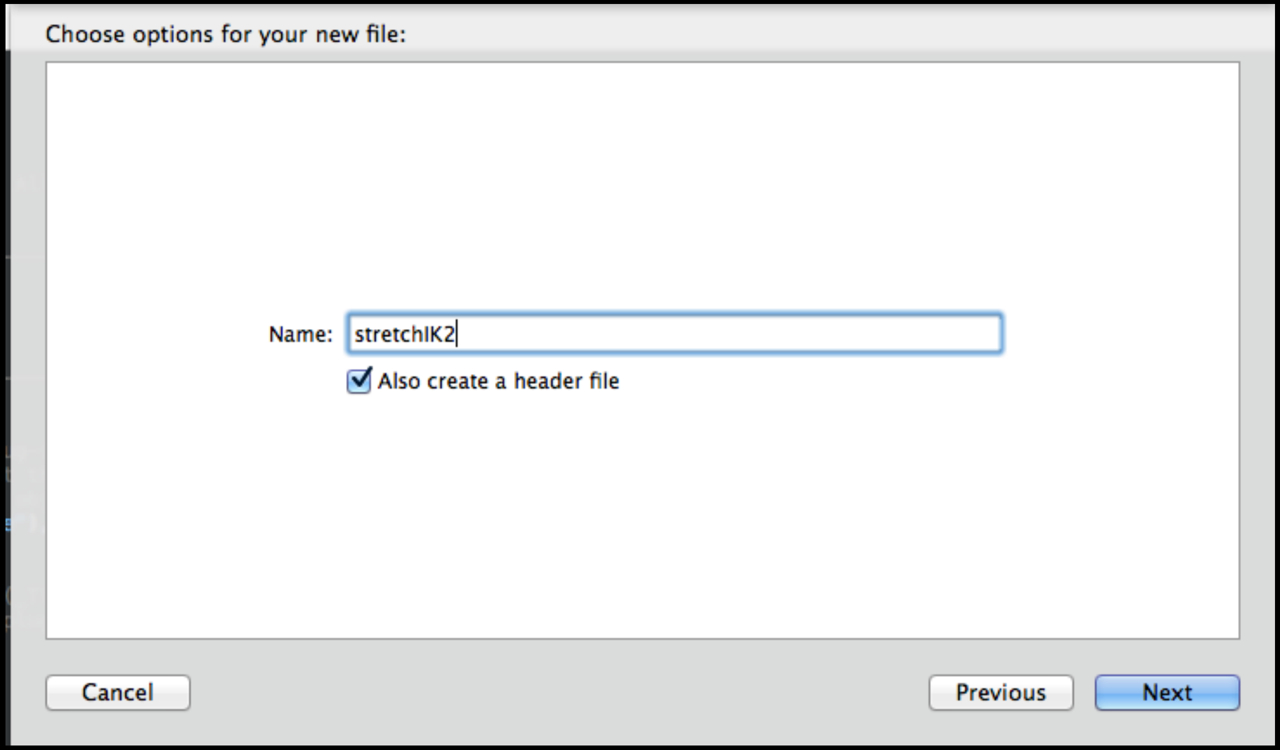
Next, I'm going to add in my files. These are already written, so I'm just copying and pasting them. If you right-click on a group and click New File, it will put that file in that group. It will bring up a window in which(under the OS X menu) you choose Source > C++ File or Header File. The stuff under Maya is for more advanced stuff.



Choose a template for your new file:

C++ File

An empty C++ file.



Choose options for your new file:

Name: stretchIK2

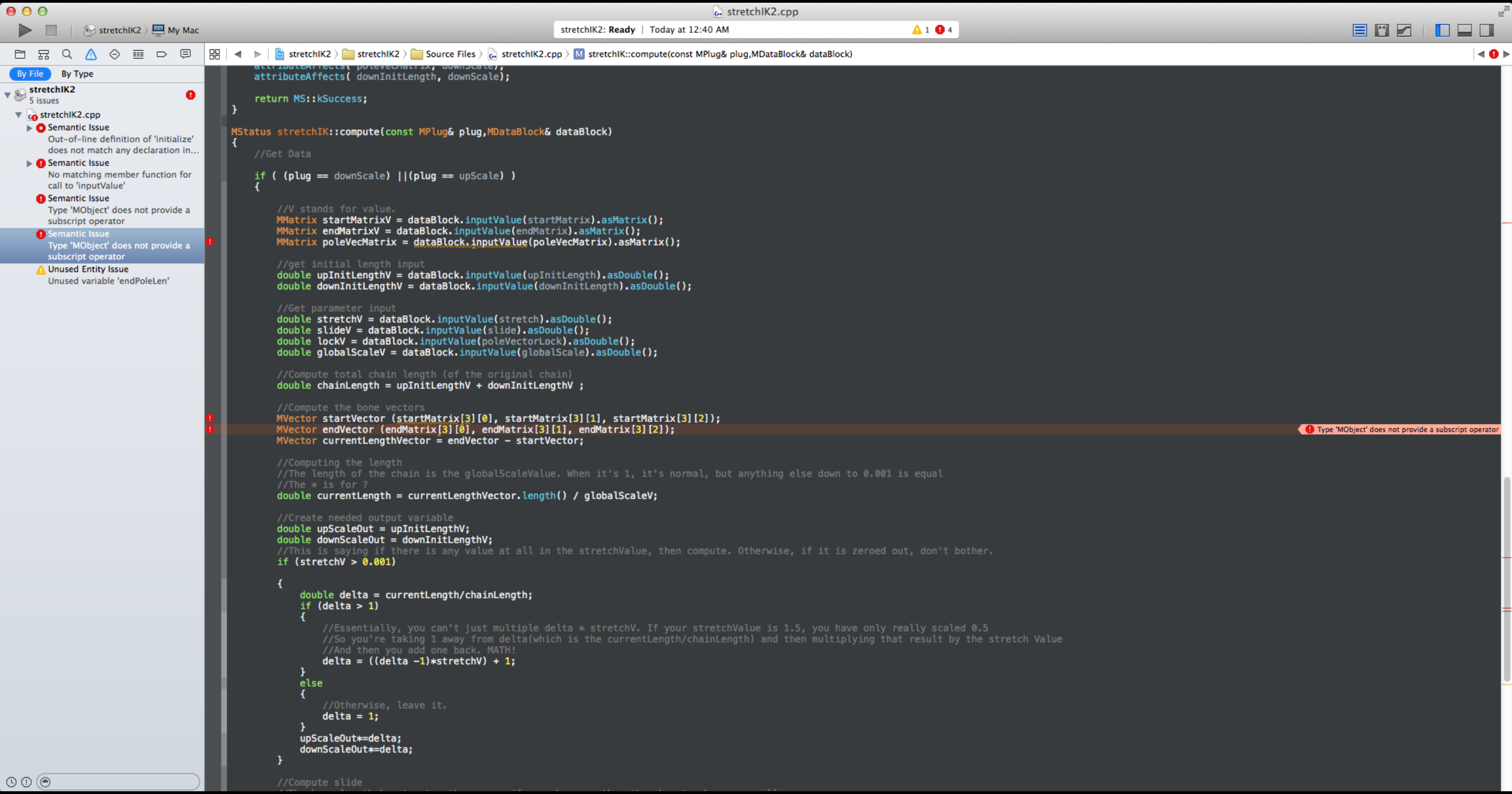
☑ Also create a header file

Go through the next two windows and name the files, and you can turn on the checkbox for Also create a header file. The following window just asks if you want to keep it in the same project, and also allows you to choose which group you want the file to be under(should be the one you clicked on) and what Targets you want to build toward.

If you look in the header file that was created, it refers to it differently. In stretchIK2.h, it lists it as
#ifndef __stretchIK2__stretchIK2
#define __stretchIK2__stretchIK2

I just remove the first __stretchIK2__ from each set and go on.

So I put in some code that I knew had errors so I could demonstrate what would happen and where to go. First, pay attention to the top bar, which is white. You'll see a warning and a Stop Sign with a 4 beside of it. It's pretty responsive to the code, so as you're going it will let you know what errors are popping up. You can also see in the Navigator that I am inside the Issue Navigator and it's giving me detailed information. My error in this case is that I put the wrong information in my matrices, and forgot the V on them(the lines that say 'MObject does not provide a subscript operator')


stretchIK2.cpp
stretchIK2 〉 My Mac
stretchIK2: Ready | Today at 12:40 AM
⚠ 1 ● 4
stretchIK2 〉 stretchIK2 〉 Source Files 〉 stretchIK2.cpp 〉 stretchIK::compute(const MPlug& plug,MDataBlock& dataBlock)
By File
By Type

stretchIK2
5 issues

stretchIK2.cpp

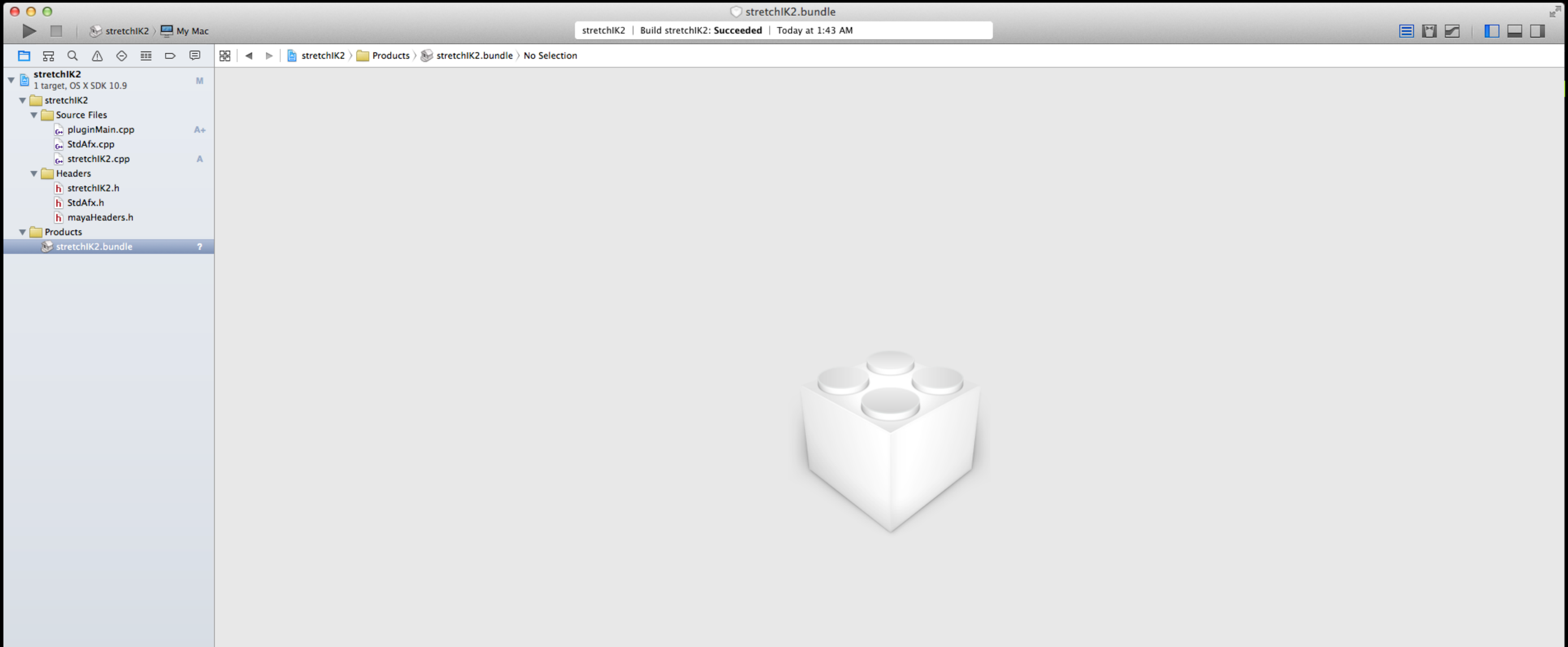
- Semantic Issue
  Out-of-line definition of 'initialize' does not match any declaration in...
- Semantic Issue
  No matching member function for call to 'inputValue'
- Semantic Issue
  Type 'MObject' does not provide a subscript operator
- Semantic Issue
  Type 'MObject' does not provide a subscript operator
- Unused Entity Issue
  Unused variable 'endPoleLen'

```
    attributeAffects( downInitLength, downScale);

    return MS::kSuccess;
}

MStatus stretchIK::compute(const MPlug& plug,MDataBlock& dataBlock)
{
    //Get Data

    if ( (plug == downScale) ||(plug == upScale) )
    {

        //V stands for value.
        MMatrix startMatrixV = dataBlock.inputValue(startMatrix).asMatrix();
        MMatrix endMatrixV = dataBlock.inputValue(endMatrix).asMatrix();
        MMatrix poleVecMatrix = dataBlock.inputValue(poleVecMatrix).asMatrix();

        //get initial length input
        double upInitLengthV = dataBlock.inputValue(upInitLength).asDouble();
        double downInitLengthV = dataBlock.inputValue(downInitLength).asDouble();

        //Get parameter input
        double stretchV = dataBlock.inputValue(stretch).asDouble();
        double slideV = dataBlock.inputValue(slide).asDouble();
        double lockV = dataBlock.inputValue(poleVectorLock).asDouble();
        double globalScaleV = dataBlock.inputValue(globalScale).asDouble();

        //Compute total chain length (of the original chain)
        double chainLength = upInitLengthV + downInitLengthV ;

        //Compute the bone vectors
        MVector startVector (startMatrix[3][0], startMatrix[3][1], startMatrix[3][2]);
        MVector endVector (endMatrix[3][0], endMatrix[3][1], endMatrix[3][2]);
        MVector currentLengthVector = endVector - startVector;

        //Computing the length
        //The length of the chain is the globalScaleValue. When it's 1, it's normal, but anything else down to 0.001 is equal
        //The * is for ?
        double currentLength = currentLengthVector.length() / globalScaleV;

        //Create needed output variable
        double upScaleOut = upInitLengthV;
        double downScaleOut = downInitLengthV;
        //This is saying if there is any value at all in the stretchValue, then compute. Otherwise, if it is zeroed out, don't bother.
        if (stretchV > 0.001)

        {
            double delta = currentLength/chainLength;
            if (delta > 1)
            {
                //Essentially, you can't just multiple delta * stretchV. If your stretchValue is 1.5, you have only really scaled 0.5
                //So you're taking 1 away from delta(which is the currentLength/chainLength) and then multiplying that result by the stretch Value
                //And then you add one back. MATH!
                delta = ((delta -1)*stretchV) + 1;
            }
            else
            {
                //Otherwise, leave it.
                delta = 1;
            }
            upScaleOut*=delta;
            downScaleOut*=delta;
        }

        //Compute slide
```

Type 'MObject' does not provide a subscript operator

Whenever you're ready to build, select Build from the Product menu or hit CMD+B. It should succeed. When it succeeds, there will be a Products folder in the Project Navigator with xxxxxnode.bundle - however, you might not see this right away.

This is because after XCode 4.x.x, the location where .bundle files were built was randomized, I guess? But this is fixable.
Go to XCode > Preferences > Locations Tab
Look at Derived Data but don't touch the menu beside of it. Look for the Advanced... button in the same row.
In the window that pops up, change it from Unique to Legacy. This will create the .bundle file underneath your project folder now. It will create it in a folder called "Debug". To find the file, you can just right-click on it and say "Show In Finder."



stretchIK2 | Build stretchIK2: **Succeeded** | Today at 1:43 AM

Tips:
Learn as much about the IDE as you can. Some people will tell you to not worry about the IDE, it's the programming that counts - this is true, but also incorrect. Your IDE is one of your main tools and is there to help you. A construction worker doesn't take a hammer made out of hamster-cage grade wood chips to work, he takes something he knows how to use and knows will give him the best work. Use your IDE well, but don't rely on it, don't use it as a crutch.

Read about the language outside of classwork. Every day, stay on it.

Keep up-to-date on CLion, a C++ IDE by Jetbrains. There is no Maya support as of the time of this writing(because it was announced a month ago), but who knows. Maybe I'll get bored one day and write a template file for it like the ones we just went over.

Rewatch and redo the videos from Marco, over and over and over. You're not going to understand it by just watching it once.

If you need more help, my e-mail address is alexwidener@gmail.com or you can message me on the Rigging Dojo website.