

# How to Get Started with Maya Plug-ins

Part 1- Setting up the build environment and Compile a C++ code in Eclipse (Linux)

### What you need!

First you need to make sure that you have a '**devkit**'. it comes with maya, so you don't have it make sure you have it. On my computer is located at: /usr/autodesk/maya2012-x64/devkit

This folder includes a lot of important materials which helps you to develop tools for Maya. The most important thing in the 'devkit' is that it includes a directory that includes more than 600 C++ source pluging codes that makes it an important asset for the developer. You can look at the some of the plugins and see how is done and get help from it.

Next important thing included in the plug-ins folder is a code called '**buildconfig**' which is required for making a and building the C++ source code.

You also need to make sure that you have the 'gcc' and 'gcc-c++' installed on your computer. That is a compiler for our code. If you are installing them, Make sure you install the version 4.1.2. It is not a big deal because there is a work around it, but that is a version that 'buildconfig' uses to build the source code. If you already have a different version of 'gcc' installed, don't worry, I'll show you how to use it. To find see what is the version of your 'gcc', in your terminal type: gcc --version

Next thing we need is an editor for writing and compiling the code. I use 'eclipse' but you can use anything else you are comfortable with, you could use terminal to do all this too. Go to <u>eclipse</u> website and download the Eclipse IDE for C/C++ Developers.

## Set it up!

Lets set up our environment:

Open eclipse and go File --> New --> C++ Project. That will open up the dialog box (images below) that you can choose from variety of project types.

++ Project         reate C++ project of selected type         troject name:       basePlugin         Z       Use default location         ocation:       (home/alij/workspace/C++_Plugin/basePlugin)       Brows         Choose file system:       default   ○         roject type:       Toolchains:         > ⇒ Shared Library       - Other Toolchain         > ⇒ Static Library       - Other Toolchain         > ⇒ Makefile project       GNU Autotools Toolchain         ■ Hello World C++ Project       ■         > ⇒ GNU Autotools       Inux GCC	C+-	Project
troject name:       basePlugin         Z       Use default location         cocation:       //home/alij/workspace/C++_Plugin/basePlugin       Brows         Choose file system:       default   \$         troject type:       Toolchains:         >       Executable         >       Shared Library         >       Static Library         >       Empty Project         •       Hello World C++ Project         >       GNU Autotools         Z       Show project types and toolchains only if they are supported on the platform	++ Project reate C++ project of selected type	
☑ Use gefault location       Brows         occation:       //home/alij/workspace/C++_Plugin/basePlugin       Brows         Choose file system:       default   ◇         rroject type:       Toolchains:         ◇ Executable       - Other Toolchain         ◇ Shared Library       - Other Toolchain         ◇ Makefile project       GNU Autotools Toolchain         ● Hello World C++ Project       - Linux GCC         ☑ Show project types and toolchains only if they are supported on the platform	Project name: basePlugin	
cocation:       (home/alij/workspace/C++_Plugin/basePlugin       Brows         Choose file system:       default   ◇         rroject type:       Toolchains:         > Shared Library       - Other Toolchain         > Makefile project       GNU Autotools Toolchain         • Hello World C++ Project       Linux GCC         ✓ Show project types and toolchains only if they are supported on the platform	✓ Use <u>d</u> efault location	
Choose file system:       default            rroject type:       Toolchains:         > Executable       - Other Toolchain Cross GCC         > Static Library       GNU Autotools Toolchain         > Makefile project       Linux GCC         • Empty Project       Hello World C++ Project         > GNU Autotools       Show project types and toolchains only if they are supported on the platform	_ocation: //home/alij/workspace/C++_F	Plugin/basePlugin Browse.
roject type:       Toolchains:         > Static Library       - Other Toolchain Cross GCC         Static Library       GNU Autotools Toolchain         > Makefile project       Inux GCC         • Hello World C++ Project       -         > GNU Autotools       -         Z Show project types and toolchains only if they are supported on the platform	Choose file system: default	0
<ul> <li>Executable</li> <li>Shared Library</li> <li>Static Library</li> <li>Makefile project</li> <li>Empty Project</li> <li>Hello World C++ Project</li> <li>GNU Autotools</li> </ul>	Project type:	Toolchains:
<ul> <li>▷ Shared Library</li> <li>▷ Static Library</li> <li>▷ Makefile project</li> <li>○ Empty Project</li> <li>○ Hello World C++ Project</li> <li>▷ GNU Autotools</li> </ul>	🕨 😑 Executable	Other Toolchain
> Static Library       GNU Autotools Toolchain         > Makefile project       Linux GCC         • Hello World C++ Project       Solution         > GNU Autotools       Linux GCC         2 Show project types and toolchains only if they are supported on the platform	🕨 🗁 Shared Library	Cross GCC
<ul> <li>➢ Makefile project</li> <li>Empty Project</li> <li>Hello World C++ Project</li> <li>➢ GNU Autotools</li> <li>Z Show project types and toolchains only if they are supported on the platform</li> </ul>	🕨 🗁 Static Library	GNU Autotools Toolchain
Empty Project     Hello World C++ Project     GNU Autotools      Show project types and toolchains only if they are supported on the platform	🗢 ≽ Makefile project	Linux GCC
Hello World C++ Project     GNU Autotools     Show project types and toolchains only if they are supported on the platform	🖶 Empty Project	
GNU Autotools Show project types and toolchains only if they are supported on the platform	Hello World C++ Project	
☑ Show project types and toolchains only if they are supported on the platform	👂 📴 GNU Autotools	
	☑ Show project types and toolchains o	nly if they are supported on the platform
Cancel Fini		Next > Cancel Finish

From the 'MakeFile project' select the 'Empty Project', form the 'toolchains' select the 'Linux Gcc' and give it the project name and click Finish.

In the '**Project Explorer**' tab, right click on your project folder and create a '**source folder**'. Name it whatever you want, I usually call it '**scr**'.

\* If you are not see the 'Project Explorer' tab, go to Window -- > Show View -- > Project Exploerer.

e					
<u>F</u> ile <u>E</u> dit	<u>S</u> ource	Refac	<u>t</u> or	<u>N</u> avi	gate
📬 🔲			<u>c</u> *	62≁	<b>c</b> ~
陷 Project	Explorer	x			• 🗖
		Ē	\$ <u>5</u>	69	▽
🗢 😂 base	Plugin				
Includes					
🔑 scr					

From 'devkit/plug-ins' copy the 'buildconfig' file into your source folder, in my case in 'scr'. If you have a different version of 'gcc', double click on 'buildconfig' file and open it. Find the 'gccVersion' in the file and replace the version to be the same as the version of the gcc on your system. After doing so you can close it.

Now we need a '**Makefile**' to be able to compile the C++ code and make a **.so** file which is a plugin version for Maya on Linux. Right click on source folder, scr, and go to New --> File and make new file, name it exactly '**Makefile**'. In the file type the next code:

include buildconfig %.so : %.cpp \$(C++) -o \$@ \$(INCLUDES) \$(C++FLAGS) \$< \$(LFLAGS) \$(LIB) clean: rm -f \*.o \*.so

Then save it and close it.

There are few more steps until we can actually build and create Maya plug-in:

Right click on the source folder, in my case 'scr', and go to New --> Source File. Name it whatever you want and make sure

you add '.cpp' at end of it, In my case is called helloWorld.cpp.

•	New Source File		×
Source File Create a new s	ource file.		C
Source fol <u>d</u> er:	basePlugin/scr		<u>B</u> rowse
Source fil <u>e</u> :	helloWorld.cpp		
<u>T</u> emplate:	Default C++ source template	\$	Configure
?		Cancel	<u>F</u> inish

Click on finish, and save it.

Now it is time to make targets, which means once you click on it, it'll build or delete the compiled file. Go to 'Make Target' tab, if you don't have it go to Window --> Show View --> Make Target. Open the project folder and right click on source folder, in my case 'scr', then click on new to make a new target.

#### **Clean Target:**

Name it 'clean', or what ever make sense to you. Uncheck the use

Builder Settings, and type the next line:

#### make -f /PATH/TO/YOUR/Makefile

and then click ok

Target name: clean		
Make Target		
Same as the target name		
Make target: clean		
Build Command		
Use builder settings		
Build command: make -f /home/alij/workspace/C++_Plug	gin/basePlugin/s	cr/Makefile
Build Settings		
☑ Stop on first build error		
☑ Run all project builders		
	Cancel	ОК

Create Make Target

#### **Build Target:**

Name it 'Build', or what ever make sense to you.				
Uncheck the use Builder Settings, and type the next line:				
make -f /PATH/TO/YOUR/Makefile				
Uncheck the 'Same as the target name' and put the name				
of the plugin , make sure you add '.so' to the name, that's what				
will be the plugin, eventually.and then click ok				

		Modify Make Targ	et	
Target name: 🕒	lild			
Make Target				
Same as the	target name			
Make target: h	lloWorld.so			
Build Command				
🗆 Use builder s	ettings			
Build command	make -f /home/	/alij/workspace/C++	Plugin/basePlugin/s	cr/Makefile
Build Settings				
Stop on first	build error			
🗹 Run all proje	ct builders			
			Cancel	OK
			Cancel	UK

Right click on your project folder in the Project Explorer tab and go to **Properties**. In the C/C++ Build section click on Builder Setting tab, and then add the name of the source folder, in my case scr, to the build directory section.

	<ul> <li>C/C++ Build</li> <li>Build Variables</li> <li>Discovery Options</li> <li>Environment</li> <li>Logging</li> <li>Settings</li> <li>Tool Chain Editor</li> <li>C/C++ General</li> <li>Project References</li> <li>Run/Debug Settings</li> <li>Task Repository</li> <li>WikiText</li> </ul>	<ul> <li>Builder Settings          <ul> <li>Behaviour</li> <li>Refresh Policy</li> </ul> </li> <li>Builder         <ul> <li>Builder type:</li> <li>External builder</li> <li>Use default build command</li> <li>Build command:</li> <li>make</li> </ul> </li> <li>Makefile generation         <ul> <li>Generate Makefiles automatically</li> <li>Expand</li> <li>Build location</li> <li>Build directory: \${workspace_loc:/basePlugin/scr}}</li> </ul> </li> </ul>
Then got to <b>Behaviour</b> tab. In the Build (Incremental build section <b>delete 'all'</b> .	<ul> <li>C/C++ Build</li> <li>C/C++ General Project References Run/Debug Settings</li> <li>Task Repository WikiText</li> </ul>	Builder Settings Behaviour Refresh Policy Build settings Stop on first build error
		Workbench Build Behavior Workbench build type: Make build Build on resource save (Auto build) all Note: See Workbench automatic build preference
Don't close the Properties window,		
Thoro is more stuff to do		✓ Clean clean

There is more stuff to do.

This is a last step, hang in there.

While you are in the **Properties** tab, go to the **Path and Symbols** section. In the **Includes** tab, click on the **GNU C++** and then click on the **Add** button on left. That will open a window where you can add the path of Maya's API library. I would suggest that you check both **Add to all configuration** and **Add to all languages**. Then add the Maya's include path, in my case is */usr/autodesk/maya/include*, by typing out or click on **File system** and direct it to the Include directory. Click ok, and the apply all the settings in the Properties window and close it.

![](_page_7_Figure_2.jpeg)

You are pretty much all set, Just double click on the **build**, in the Make Target tab, and you should get something like the image below in your Console tab.

🖹 Problems 🧟 Tasks 📮 Console 🕱 🛛 🔲 Properties	-0 0 🔄 🖬 🚮 🜬 🖻 🖬 🖓 🖓
CDT Build Console [basePlugin]	
<pre>**** Build of configuration Default for project basePlugin **** make -f /home/alij/workspace/C++_Plugin/basePlugin/scr/Makefile helloWor cc -o helloWorld.so -III/usr/autodesk/maya/include -I/usr/X11R6/i DBits64m64 -DUNIX -D_BOOL -DLINUX -DFUNCPROTO -D_GNU_SOURCE -DLINUX_6- DREQUIRE_IOSTREAM -Wno-deprecated -03 -Wall -Wno-multichar -Wno-comment -fno-gnu-keywords -ftemplate-depth-25 -pthread -Wno-deprecated -fno-gnu shared **** Build Finished ****</pre>	ld.so nclude -I/opt/nvidia/nvparse/include/glh - 4 -fPIC -fno-strict-aliasing - -Wno-sign-compare -funsigned-char -Wno-reorder -keywords helloWorld.cpp -Wl,-Bsymbolic -
Now if you take a look at your folder tree in your Project Explorer, You should see the ' <b>.so</b> ' file is added to your folder.	Project Explorer 🕱 🗖 🗖
Done :) $\longrightarrow$	<ul> <li>✓ SesePlugin</li> <li>▷ Sinaries</li> <li>▷ Includes</li> <li>✓ Scr</li> <li>▷ helloWorld.cpp</li> <li>▷ helloWorld.so - [x86_64/le]</li> <li>□ buildconfig</li> <li>□ Makefile</li> </ul>