Appendix A:

A.1 Converting the VST3 TemplateSynth (9 Steps)

Example: Converting the TemplateSynth to NewSynth (where NewSynth is the name of your new project) - you will need a simple text editor to edit the .sln, .vcproj and .vcxproj files

1. copy the TemplateSynth directory and change the name of the new directory to NewSynth
2. open that directory and change all file names from TemplateSynth.xxx to NewSynth.xxx
3. open the .sln file with a text editor and Replace All TemplateSynth with NewSynth and save
4. open the .vcproj file with a text editor and Replace All TemplateSynth with NewSynth and save
5. open the .vcxproj file with a text editor and Replace All TemplateSynth with NewSynth and save
6. open the .sln file in Visual Studio (everything else is done in VS) and do a Replace in Current Project to replace all instances of TemplateSynth with NewSynth
7. IMPORTANT: use guidgen.exe to make new GUIDs for the following:

```
FUID Controller::cid (0xB561D747, 0xBA004597, 0xA3BF911A, 0x5DA2AFA4);
FUID Processor::cid (0x91F037DC, 0xA35343AB, 0x852C37B1, 0x3774DC90);
```

Guidgen.exe and FUIDs are explained in Chapter 2. Replace the existing FUIDs at the top of the *VSTSynthProcessor.cpp* and *VSTSynthController.cpp* files with your newly generated and properly formatted FUIDs.

8. Set your Output Directories: open the Project Properties for your **NewSynth** and on the General panel, browse for your Output Directory and Intermediate Directory; this is where your final .vst3 will be delivered. Then, do the same thing for the **base_vc** project and make sure to use the same directories as in the NewSynth project.
9. Set your debugger's VST3 host: open the Project Properties and open the Debugging panel; in the Command field, browse to find your VST3 host executable file, for example *Cubase LE AI Elements 7.exe* and then set the Attach field to YES. If you forget the Attach part, the debugger will never launch

A.2 Converting the AU TemplateSynth (10 Steps)

Example: Converting the TemplateSynth to NewSynth (where NewSynth is the name of your new project)

1. copy the TemplateSynth directory and change the name of the new directory to NewSynth

2. open the .xcodeproj file (you don't need to rename it): at upper left of Xcode, double click on the TemplateSynth project (it says "Two Targets" below it) and change the name to NewSynth; when prompted to Rename Project Items, answer YES
3. Use Find/Replace to do a brute force replacement of TemplateSynth -> NewSynth
4. Close and re-open Xcode - click on the project in the upper left (now renamed NewSynth) and you will see two targets, Info, Build Settings, and Build Phases: In the **Info** settings:
    - Open the AudioComponents array, then open the Item 0 dictionary (See Chapter 2 for more information about this) and change:
        - manufacturer: <your 4 digit company code> (mine is WILL)
        - subtype: <your 4 digit plug-in code> (for MiniSynth mine is MS00)
        - name: concatenation of your Company Name : plug-in title (mine is Will Pirkle: MiniSynth)
5. Open the AUSynth.r file and change the following to match (for backwards compatibility with Logic 9)
    - COMP_MANUF: <your 4 digit company code> (mine is WILL)
    - COMP_SUBTYPE: <your 4 digit plug-in code> (for MiniSynth mine is MS00)
    - NAME: concatenation of your Company Name : plug-in title (mine is Will Pirkle: MiniSynth)
7. Change the names of:
    TemplateSynthView.h
    TemplateSynthView.cpp
    TemplateSynthViewFactory.h
    TemplateSynthViewFactory.cpp
    to
    NewSynthView.h
    NewSynthView.cpp
    NewSynthViewFactory.h
    NewSynthViewFactory.cpp
8. In XCode, do a brute force Find and Replace to replace:
    - WPEditBoxTS --> WPEditBoxNS (NS = NewSynth)
    - WPRotaryKnobTS --> WPRotaryKnobNS
    - WPPopUpButtonCellTS --> WPPopUpButtonCallNS
    - WPOptionMenuGroupTS --> WPOptionMenuGroupNS
9. Click on the CocoaSynthView.nib file to open it in Interface Builder and:
    - set the File's Owner to *NewSynthViewFactory*
    - set the View (the NSView laying in the center of the window you see) to *NewSynthView*
10. Set up the debugger by clicking on the Scheme and choose Edit Scheme
    - Click on Run (left) and then choose Development from the drop-down list
    - Click on the Executable box and browse to find your AU host (e.g. Logic 9, Ableton, etc…) and you are ready to debug

Validate your plug-in by opening Terminal and using the AU validation:

auval -v aumu <PLUG> <COMP)

Where <PLUG> is your 4-character plug-in code and <COMP> is your 4 character company name, for my MiniSynth that would be:

auval -v aumu MS00 WILL

If your validation does not succeed, go back and check your conversion. Once it does succeed, you are ready to code the rest of the synth.

For debugging, edit the Scheme for your project, go to the Run panel and choose the Executable drop-down. Browse to find your AU client's executable file.