



Autodesk MotionBuilder 2013

Programming in MotionBuilder || Focusing on Python

Naiqi Weng, Developer Consultant
Autodesk Developer Network

Module 2



The Beginning: Python SDK

Module 2

Module's Agenda

- Working with the Python Version
- Navigating MotionBuilder Install folders
- How to Execute Scripts
- The Python Editor
- Programming Preferences
- MotionBuilder Python Modules
- Starting to Build Code for MotionBuilder
- Helpful Coding Standards
- Assignment



Working with the Python Version

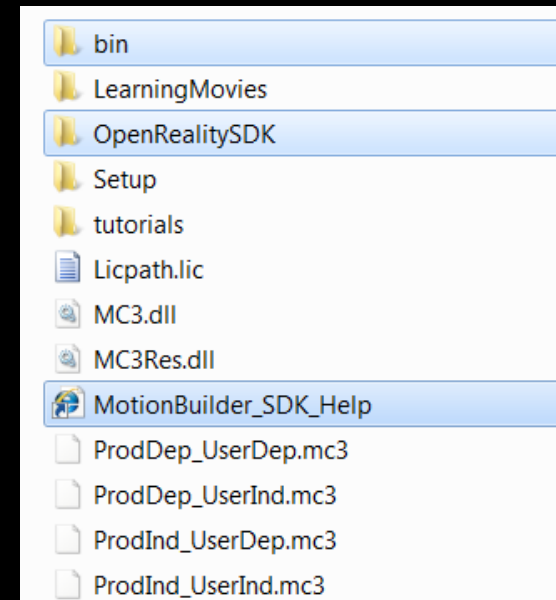
- Python Version in MotionBuilder
 - Version 2.6.4
 - Generally Matches with Maya
 - Cannot change internal version
- Including External Python Modules in MotionBuilder
 - use the environment variable PYTHONPATH and PATH
 - copying the regular .py modules into the MotionBuilder 'Python' folder

Navigating MotionBuilder Install Folders

- *Default installation location*

C:\Program Files\Autodesk\MotionBuilder 2013

- Programming Related:
 - bin
 - OpenRealitySDK
 - MotionBuilder_SDK_Help



Let's take a look at these

- Exploring the installation folders structure and setup...

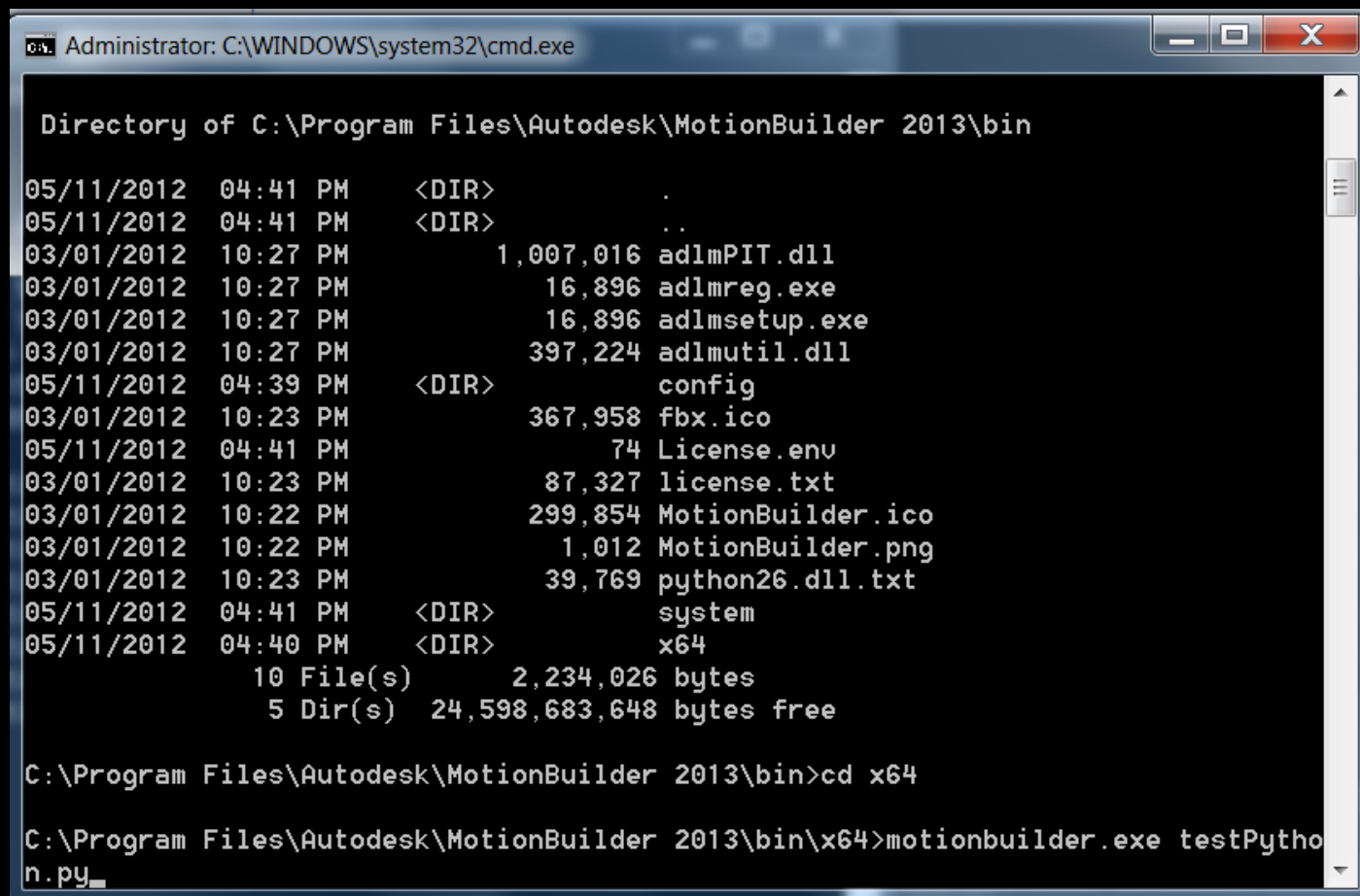
User Configuration Directory

- Comply with Windows 7 permission restrictions
 - C:Users\[user]\Documents\MB\2013-x64\config
- Be cautious: save back-up file before modification
- {ComputerName}.Application.txt
- Environment variable:
MB_CONFIG_DIR

How to Execute Scripts

- Launching Script from Command Line Startup
- Drag and Drop Script from the Asset Browser
- From the Python Editor
- Places Python Tools in 'PythonStartup' Folder
- Running Script on Hot Key Initiation
- Using the Function `FBApplication::ExecuteScript`
- Triggering a script via a script device inside a Relation Constraint

Launching Script from Command Line Startup



```
Administrator: C:\WINDOWS\system32\cmd.exe

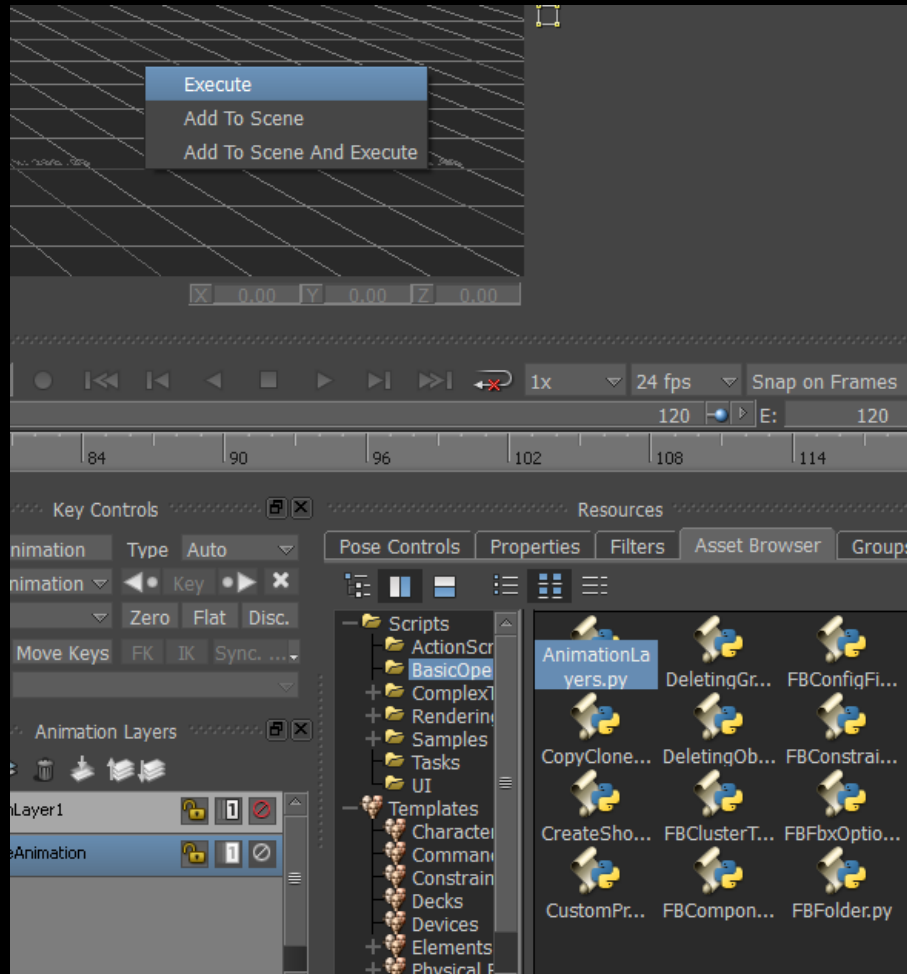
Directory of C:\Program Files\Autodesk\MotionBuilder 2013\bin

05/11/2012  04:41 PM    <DIR>          .
05/11/2012  04:41 PM    <DIR>          ..
03/01/2012  10:27 PM           1,007,016  adlmPIT.dll
03/01/2012  10:27 PM             16,896  adlmreg.exe
03/01/2012  10:27 PM             16,896  adlmsetup.exe
03/01/2012  10:27 PM           397,224  adlmutil.dll
05/11/2012  04:39 PM    <DIR>          config
03/01/2012  10:23 PM           367,958  fbx.ico
05/11/2012  04:41 PM              74  License.env
03/01/2012  10:23 PM           87,327  license.txt
03/01/2012  10:22 PM          299,854  MotionBuilder.ico
03/01/2012  10:22 PM           1,012  MotionBuilder.png
03/01/2012  10:23 PM           39,769  python26.dll.txt
05/11/2012  04:41 PM    <DIR>          system
05/11/2012  04:40 PM    <DIR>          x64
               10 File(s)      2,234,026 bytes
               5 Dir(s)  24,598,683,648 bytes free

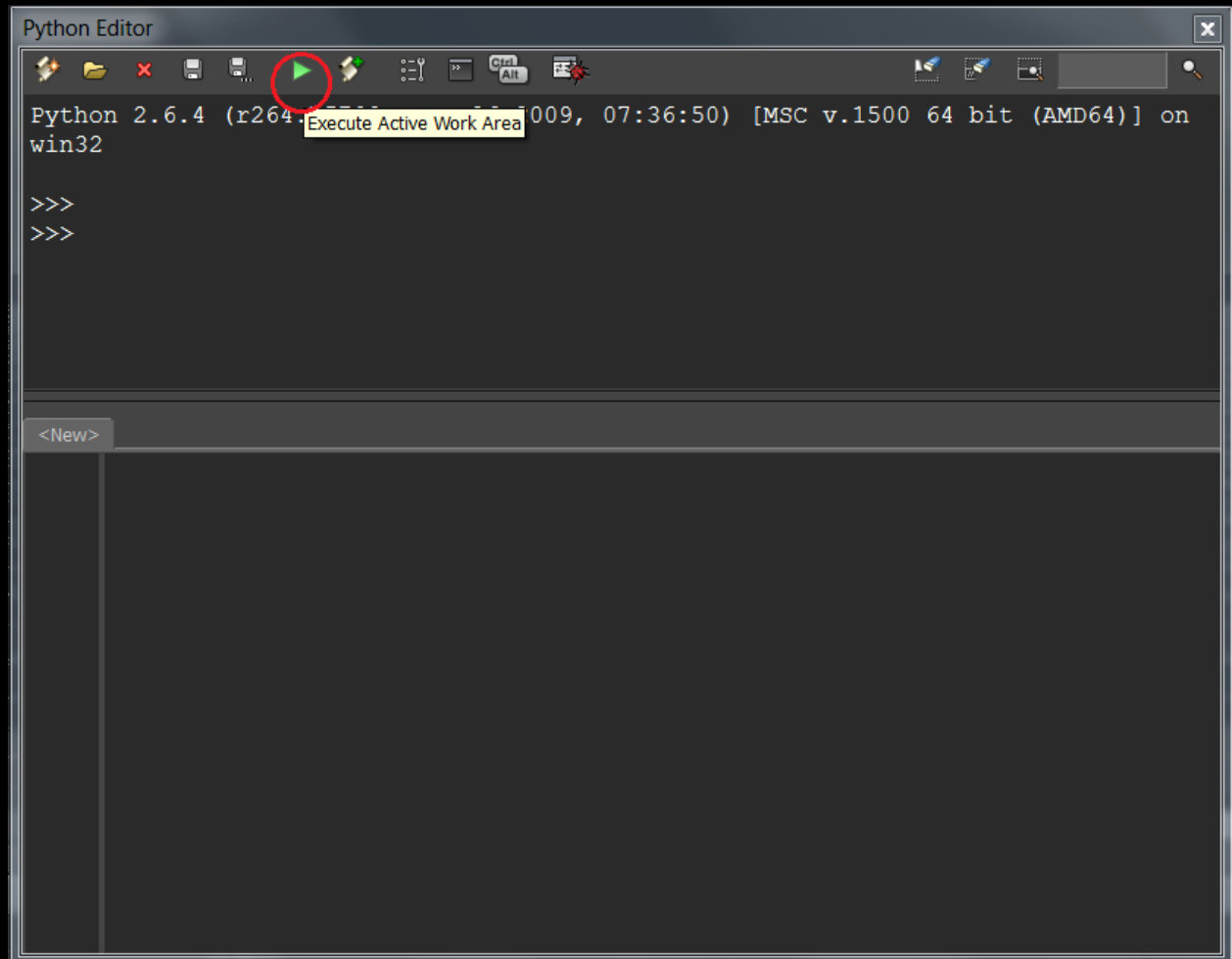
C:\Program Files\Autodesk\MotionBuilder 2013\bin>cd x64

C:\Program Files\Autodesk\MotionBuilder 2013\bin\x64>motionbuilder.exe testPython.py
```

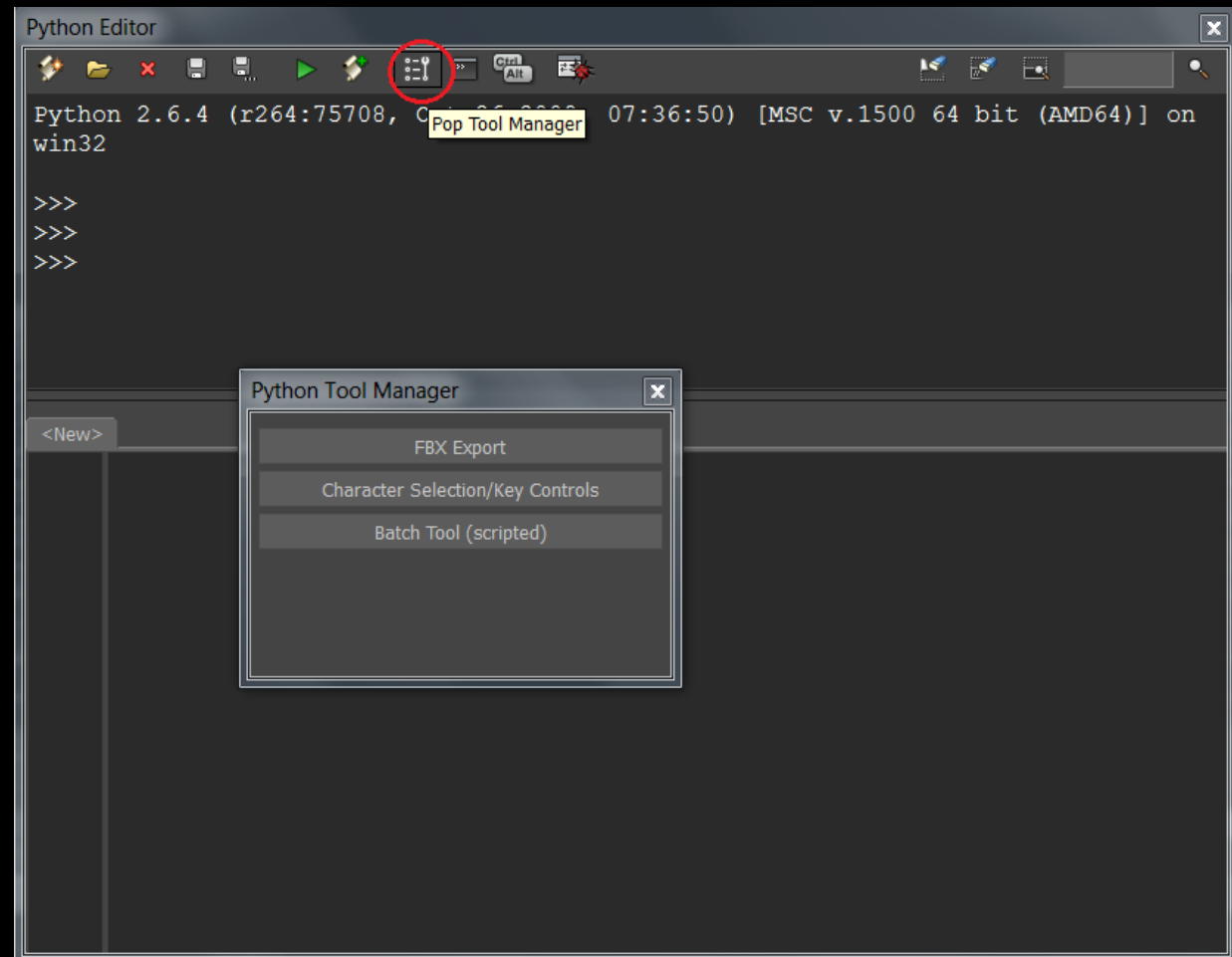
Drag and Drop Script from the Asset Browser



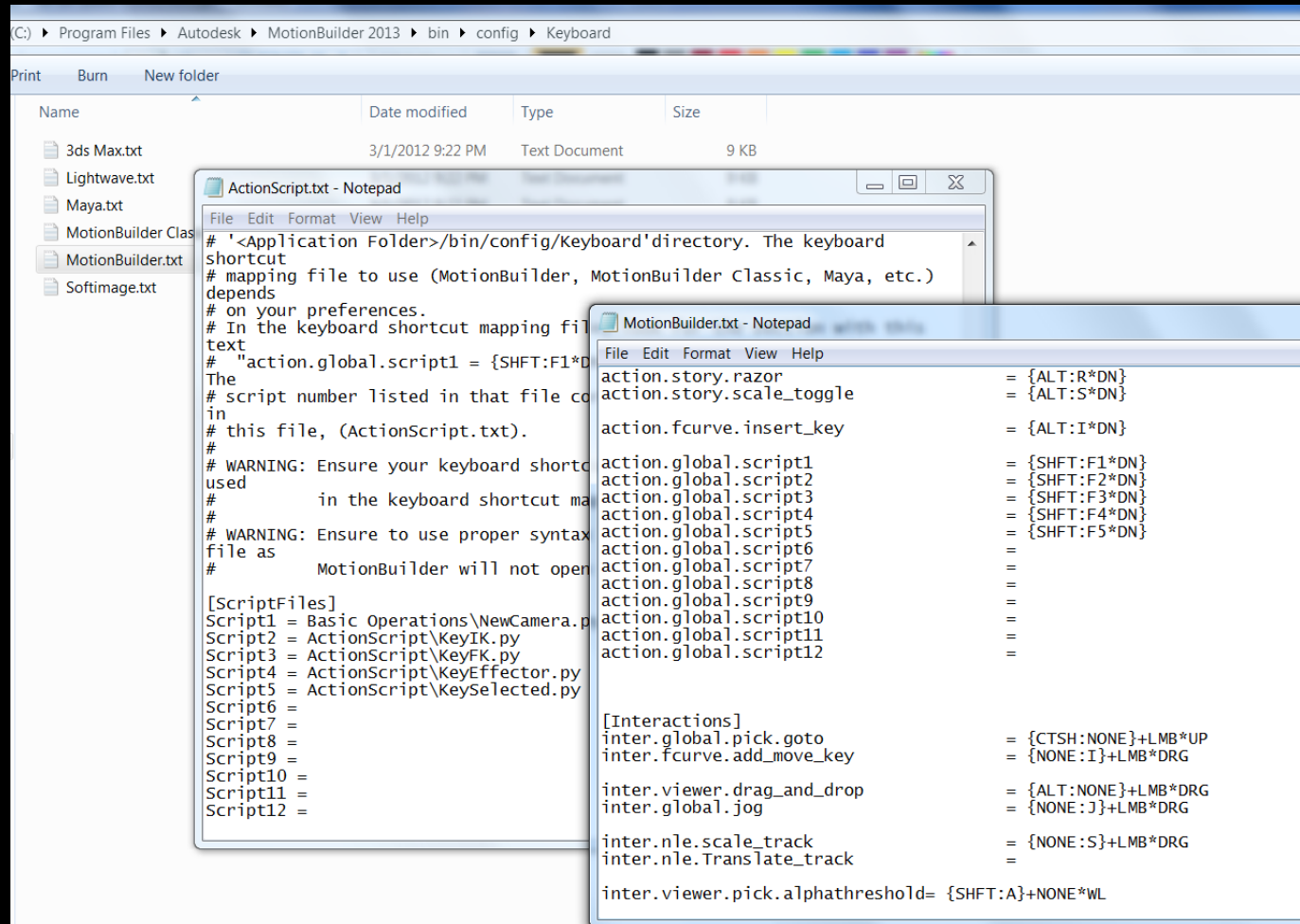
From the Python Editor



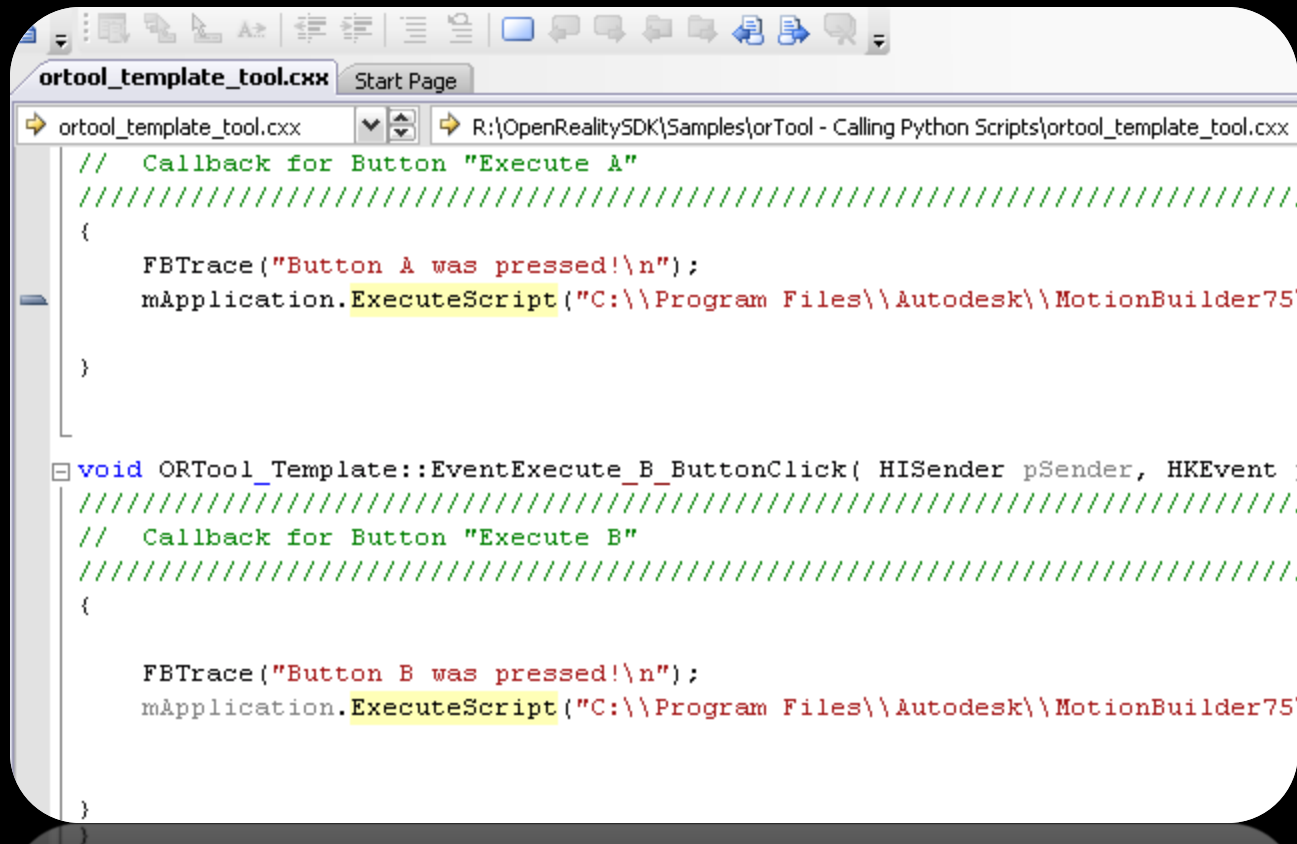
Places Python Tools in 'PythonStartup' Folder



Running Script on Hot Key Initiation



Function FBApplication::ExecuteScript

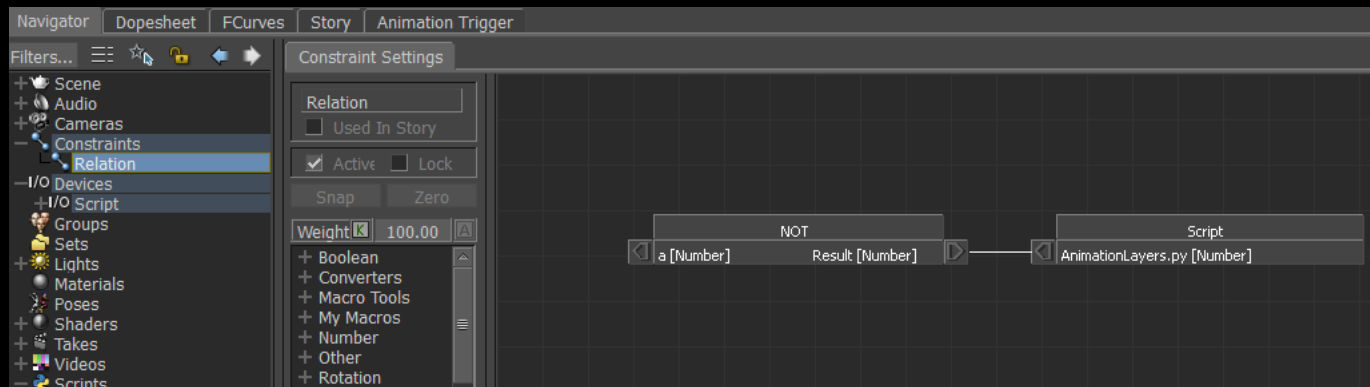


The screenshot shows a code editor window titled 'ortool_template_tool.cxx' with a 'Start Page' tab. The editor displays two callback functions. The first function, 'Execute A', calls 'FBTrace' with the message 'Button A was pressed!\n' and 'mApplication.ExecuteScript' with the path 'C:\\Program Files\\Autodesk\\MotionBuilder75'. The second function, 'EventExecute_B_ButtonClick', also calls 'FBTrace' with 'Button B was pressed!\n' and 'mApplication.ExecuteScript' with the same path. The 'ExecuteScript' method calls are highlighted in yellow.

```
// Callback for Button "Execute A"
////////////////////////////////////
{
    FBTrace("Button A was pressed!\n");
    mApplication.ExecuteScript("C:\\Program Files\\Autodesk\\MotionBuilder75");
}

void ORTool_Template::EventExecute_B_ButtonClick( HISender pSender, HKEvent
////////////////////////////////////
// Callback for Button "Execute B"
////////////////////////////////////
{
    FBTrace("Button B was pressed!\n");
    mApplication.ExecuteScript("C:\\Program Files\\Autodesk\\MotionBuilder75");
}
```

Triggering a script via a script device inside Constraint



The Python Editor

- The Python Editor UI
 - The Python Console
 - The Python Active Work Area
 - Exploring the Python Editor Toolbar and Functionality
 - Exploring the Python Editor Search Functionality



Finding our way around the Python Editor

- Looking at the real deal in MotionBuilder

Programming Preferences

- To access the programming preferences go to Settings > Preferences...
 - Python Preferences
 - SDK Preferences
- All changes to preferences must have MoBu re-start before they take affect

Programming Preferences

- Environment Variables:
 - MOTIONBUILDER_PLUGIN_PATH
The list of directories determines the location from which MotionBuilder will load compiled plugins.
`\bin\{platform}\plugins`
 - MOTIONBUILDER_PYTHON_STARTUP_PATH
The list of directories determines the location from which MotionBuilder will load Python startup scripts.
`\\%USERPROFILE%\Documents\MB\2013-x64\config\PythonStartup`

MoBu Python Modules

- Two MotionBuilder Python Modules:
 - pyfbSDK
For all classes listed in the Python Reference Documentation
 - pyfbSDK_additions
This is a helper module to build UI for Python tools



Starting to Build Code for MotionBuilder

- Open a new scene
 - ‘FBApplication’ class
- Modify configuration file
 - FBConfigFile class



Into MotionBuilder and the Documentation we go

- Open up your MotionBuilder Python Reference Documentation
- Let's get started...

Helpful Coding Standards

1. Comment your code, so that you and others can clearly read it
2. Put introduction comments at the top of your code, so the code is informative.
3. Name your variables appropriately, so it is again clear to read.
4. Use spaces, it makes easier to read

Bonus Material

- Checkout the 'Appendix One' in today's class notes:

MotionBuilder Command Line Startup Flags

Assignment

- A. Write a script that creates a new FBX scene file, this script will use the class FBApplication. If the operation is successful, print out a message.
- B. Write a script that changes the background color, font and font size of Python Editor.

Next Agenda

- General Concepts
- MotionBuilder Architecture
- MotionBuilder Classes Introduction
- Reviewing the Python Documentation Closer
- More Control over Loading and Saving Scenes
- Assignment