

Python Interface to the Plugin SDK (V1.02) (Differences from SDK)

NOTE

**The plugin SDK documentation should always be referred to.
This document only lists the functional differences for a python script.**

This document describes the differences between how a plugin is programmed in Python and how it would be programmed in C/C++.

These are the only differences and any functions not in this document can be used the same way in C/C++ or Python.

There are sections at the end of this document that explain the extra utility modules that I have added.

Notes

All scripts files must have the PI_ prefix in order for them to be loaded by the plugin. This distinguishes them from any other support scripts that you may be using.

Some of the functions require self to be passed in. Failure to do this will cause problems.

When registering callbacks you need to pass in a copy of the callback.

e.g.

```
self.MyOrbitPlaneFuncCB = self.MyOrbitPlaneFunc  
XPLMControlCamera(self, xplm_ControlCameraUntilViewChanges,  
self.MyOrbitPlaneFuncCB, 0)
```

Where “self.MyOrbitPlaneFunc” is the actual callback.

```
def MyOrbitPlaneFunc(self, outCameraPosition, inIsLosingControl, inRefcon):  
..  
..  
..
```

It is a good idea to use the examples to get a feel for how plugins are programmed in Python.

Python Interface to the Plugin SDK (V1.02) (Differences from SDK)

XPLMCamera Module

XPLMControlCamera(self, inHowLong, CameraControlCallback, inRefcon)

e.g.

```
self.MyOrbitPlaneFuncCB = self.MyOrbitPlaneFunc  
XPLMControlCamera(self, xplm_ControlCameraUntilViewChanges,  
self.MyOrbitPlaneFuncCB, 0)
```

XPLMDontControlCamera(self)

XPLMReadCameraPosition(outCameraPosition)

Pass in “outCameraPosition” as an empty list.

e.g.

```
CameraPosition = []  
XPLMReadCameraPosition(CameraPosition)
```

Python Interface to the Plugin SDK (V1.02) (Differences from SDK)

XPLMDataAccess Module

Count = XPLMGetDataavi(DataRef, DataRefValues, Offset, Max)

Pass in “DataRefValues” as an empty list.

Count = XPLMGetDatavf(DataRef, DataRefValues, Offset, Max)

Pass in “DataRefValues” as an empty list.

Count = XPLMGetDatab(DataRef, DataRefValues, Offset, Max)

Pass in “DataRefValues” as an empty list.

**XPLMDataRef = XPLMRegisterDataAccessor(
self, inDataName, inDataType, inIsWritable,
inReadInt, inWriteInt,
inReadFloat, inWriteFloat,
inReadDouble, inWriteDouble,
inReadIntArray, inWriteIntArray,
inReadFloatArray, inWriteFloatArray,
inReadData, inWriteData,
inReadRefcon, inWriteRefcon)**

NOTE

More than one dataref using the same callback is not supported.

XPLMUnregisterDataAccessor(self, inDataRef)

**RetVal = XPLMShareData(self, inDataName, inDataType, SharedDataCallback,
inNotificationRefcon)**

**RetVal = XPLMUnshareData(self, inDataName, inDataType, SharedDataCallback,
inNotificationRefcon)**

**Python Interface to the Plugin SDK (V1.02)
(Differences from SDK)**

XPLMDisplay Module

**RetVal = XPLMRegisterDrawCallback(self, inCallback, inPhase,
inWantsBefore, inRefcon)**

**RetVal = XPLMUnregisterDrawCallback(self, inCallback, inPhase,
inWantsBefore, inRefcon)**

**RetVal = XPLMRegisterKeySniffer(self, KeySnifferCallback, inBeforeWindows,
inRefcon)**

**RetVal = XPLMUnregisterKeySniffer(KeySnifferCallback, inBeforeWindows,
inRefcon)**

**HotKeyID = XPLMRegisterHotKey(inVirtualKey, inFlags, inDescription,
HotKeyCallback, inRefcon)**

XPLMUnregisterHotKey(self, HotKeyID)

XPLMGetScreenSize(outWidth, outHeight)

For outWidth, outHeight pass in None if you don't want any data back.

DO NOT pass in 0 (zero).

XPLMGetMouseLocation(outX, outY)

For outX, outY pass in None if you don't want any data back.

DO NOT pass in 0 (zero).

XPLMGetWindowGeometry(WindowID, outnLeft, outTop, outRight, outBottom)

For outLeft, outTop, outRight, outBottom pass in None if you don't want any data back.

DO NOT pass in 0 (zero).

Written by Sandy Barbour – 5th May, 2005

Updated by Sandy Barbour -Monday, 29 December 2008

Python Interface to the Plugin SDK (V1.02) (Differences from SDK)

**XPLMGetHotKeyInfo(inHotKey, outVirtualKey, outFlags, outDescription,
outPlugin)**

For outVirtualKey, outFlags, outDescription, outPlugin pass in None if you don't want any data back.

DO NOT pass in 0 (zero).

**WindowId = XPLMCreateWindow(self, inLeft, inTop, inRight, inBottom,
inIsVisible,
inDrawWindowCallback,
inKeyCallback,
inMouseClickedCallback,
inRefcon)**

XPLMDestroyWindow(self, WindowId)

**Python Interface to the Plugin SDK (V1.02)
(Differences from SDK)**

XPLMGraphics Module

outX, outY, outZ = XPLMWorldToLocal(inLatitude, inLongitude, inAltitude)

outLatitude, outLongitude, outAltitude = XPLMLocalToWorld(inX, inY, inZ)

**XPLMGetFontDimensions(inFontID, outCharWidth, outCharHeight,
outDigitsOnly)**

For outCharWidth, outCharHeight, outDigitsOnly pass in None if you don't want any data back.

DO NOT pass in 0 (zero).

Python Interface to the Plugin SDK (V1.02)
(Differences from SDK)

XPLMMenu Module

**MenuID = XPLMCreateMenu(self, inName, inParentMenu, inParentItem,
MenuCallback, inMenuRef)**

XPLMDestroyMenu(self, inMenuID)

Python Interface to the Plugin SDK (V1.02) (Differences from SDK)

XPLMNavigation Module

XPLMFindNavAid(inNameFragment, inIDFragment, inLat, inLon, inFrequency, inType)

For inIDFragment, inLat, inLon, inFrequency pass in None if you don't want any data back.

DO NOT pass in 0 (zero).

XPLMGetNavAidInfo(inRef, outType, outLatitude, outLongitude, outHeight, outFrequency, outHeading, outID, outName, outReg)

For outType, outLatitude, outLongitude, outHeight, outFrequency, outHeading, outID, outName, outReg pass in None if you don't want any data back.

DO NOT pass in 0 (zero).

Pass in a list with one value if you want the function to use that value.
Pass in an empty list if you only want data back.

XPLMGetFMSEntryInfo(Index, outType, outID, outRef, outAltitude, outLat, outLon)

For outType, outID, outRef, outAltitude, outLat, outLon pass in None if you don't want any data back.

DO NOT pass in 0 (zero).

**Python Interface to the Plugin SDK (V1.02)
(Differences from SDK)**

XPLMPlanes Module

outTotalAircraft, outActiveAircraft, outController = XPLMCountAircraft()

outFileName, outPath = XPLMGetNthAircraftModel(inIndex)

XPLMAcquirePlanes(self, pAircraft, PlanesAvailableCallback, inRefcon)

Pass in “pAircraft” as a list of lists.

XPLMReleasePlanes(self)

**XPLMDrawAircraft(inPlaneIndex, inX, inY, inZ, inPitch, inRoll, inYaw,
inFullDraw, inDrawStateInfo)**

Pass in the “inDrawStateInfo” structure as a list.

Python Interface to the Plugin SDK (V1.02)
(Differences from SDK)

XPLMPlugin Module

**XPLMGetPluginInfo(inPlugin, outName, outFilePath, outSignature,
outDescription)**

For outName, outFilePath, outSignature, outDescription pass in None if you don't want any data back.

DO NOT pass in 0 (zero).

Python Interface to the Plugin SDK (V1.02)
(Differences from SDK)

XPLMProcessing Module

**XPLMRegisterFlightLoopCallback(self, inFlightLoopCallback, inInterval,
inRefcon)**

XPLMUnregisterFlightLoopCallback(self, inFlightLoopCallback, inRefcon)

**Python Interface to the Plugin SDK (V1.02)
(Differences from SDK)**

XPLMUtilities Module

VirtualKeyDescription = XPLMGetVirtualKeyDescription(VirtualKey)

SystemPath = XPLMGetSystemPath(SystemPath)

PrefsPath = XPLMGetPrefsPath(PrefsPath)

DirectorySeparator = XPLMGetDirectorySeparator()

PathOnly, FullPath = XPLMExtractFileAndPath(FullPath)

**RetVal, outFileNames, outIndices, outTotalFiles, outReturnedFiles =
XPLMGetDirectoryContents(inDirectoryPath, inFirstReturn, inFileNameBufSize,
inIndexCount)**

XPlaneVersion, XPLMVersion, HostID = XPLMGetVersions()

**Python Interface to the Plugin SDK (V1.02)
(Differences from SDK)**

XPUIGraphics Module

XPGetWindowDefaultDimensions(inStyle, outWidth, outHeight)

For outWidth, outHeight pass in None if you don't want any data back.

DO NOT pass in 0 (zero).

XPGetElementDefaultDimensions(inStyle, outWidth, outHeight, outCanBeLit)

For outWidth, outHeight, outCanBeLit pass in None if you don't want any data back.

DO NOT pass in 0 (zero).

outWidth, outCanBeLit = XPGetTrackDefaultDimensions(inStyle)

**outIsVertical, outDownBtnSize, outDownPageSize, outThumbSize, outUpPageSize,
outUpBtnSize = XPGetTrackMetrics(inX1, inY1, inX2, inY2, inMin, inMax,
inValue, inTrackStyle)**

Python Interface to the Plugin SDK (V1.02) (Differences from SDK)

XPWidgets Module

XPGetWidgetGeometry(inWidget, outLeft, outTop, outRight, outBottom)

For outLeft, outTop, outRight, outBottom pass in None if you don't want any data back.

DO NOT pass in 0 (zero).

XPGetWidgetExposedGeometry(inWidgetID, outLeft, outTop, outRight, outBottom)

For outLeft, outTop, outRight, outBottom pass in None if you don't want any data back.

DO NOT pass in 0 (zero).

WidgetDescriptorLength = XPGetWidgetDescriptor(inWidget, outDescriptor, inMaxDescLength)

Pass in "outDescriptor" as an empty list.

XPAddWidgetCallback(self, inWidget, inNewCallback)

XPDestroyWidget(self, inWidget, inDestroyChildren)

outWidgetID = XPCreateCustomWidget(self, inLeft, inTop, inRight, inBottom, inVisible, inDescriptor, inIsRoot, inContainer, WidgetCallback)

Python Interface to the Plugin SDK (V1.02)
(Differences from SDK)

XPWidgetUtils Module

XPUCreateWidgets(inWidgetDefs, inCount, inParamParent, ioWidgets)

Pass in “ioWidgets” as an empty list.

Python Interface to the Plugin SDK (V1.02) (Differences from SDK)

XPWidgetDefs Module

x, y, button = PI_GetMouseState(inParam1)

Example

```
def MainWidgetHandler(self, inMessage, inWidget, inParam1, inParam2):
```

```
    if (inMessage == xpMsg_MouseDown):
        x, y, button = PI_GetMouseState(inParam1)

    return 0
```

dx, dy, dwidth, dheight = PI_GetWidgetGeometryChange(inParam2)

Example

```
def MainWidgetHandler (self, inMessage, inWidget,          inParam1, inParam2):
```

```
    if (inMessage == xpMsg_Reshape):
        dx, dy, dwidth, dheight = PI_GetWidgetGeometryChange(inParam2)

    return 0
```

key, flags, vkey = PI_GetKeyState(inParam1)

Example

```
def EditWidgetHandler(self, inMessage, inWidget, inParam1, inParam2):
```

```
    if (inMessage == xpMsg_KeyPress):
        key, flags, vkey = PI_GetKeyState(inParam1)
```

```
    return 0
```

**Python Interface to the Plugin SDK (V1.02)
(Differences from SDK)**

**PythonScriptMessaging Module
(Unique to the PythonInterface)**

NumberOfScripts = PI_CountScripts()

ScriptId = PI_GetNthScript(inIndex)

ScriptId = PI_FindScriptBySignature(inSignature)

PI_GetScriptInfo(ScriptId, outName, outSignature, outDescription)

EnableStatus = PI_IsScriptEnabled(inScriptId)

EnableStatus = PI_EnableScript(inScriptId)

PI_DisableScript(inScriptId)

PI_SendMessageToScript(self, inScriptId, inMessage, inParam)

Pass in None instead of inScriptId to send to all scripts.

Python Interface to the Plugin SDK (V1.02) (Differences from SDK)

SandyBarbourUtilities Module (Unique to the PythonInterface)

This module will send text to various areas.
It also redirects stdout and stderr.

In windows, if you use my standalone Python Interface Plugin Console app, the text will be displayed there as well.
This is good for debugging startup problems.

In linux it will be written to the terminal.

On the mac there is only the plugin control panel at the moment.

SandyBarbourDisplay()

This will send text to the top widget listbox in the python interface control panel.

SandyBarbourClearDisplay()

This will clear the text in the top widget listbox in the python interface control panel.

SandyBarbourPrint()

This will send text to the bottom widget listbox in the python interface control panel.

SandyBarbourClearPrint()

This will clear the text in the bottom widget listbox in the python interface control panel.

Python Interface to the Plugin SDK (V1.02) (Differences from SDK)

SandyBarbourVCUtilities Module (Unique to the PythonInterface and the VirtualCamera Plugin)

This module allows access to my VirtualCamera plugin.
The main purpose is so that a python script can be used to orchestrate camera sequences.
I wanted to add scripting to my VirtualCamera plugin, what better way to do it than use
my existing PythonInterface.

CameraPluginReady = VC_CameraPluginReady()

Because of the way our plugin SDK starts up, you can't be sure if another plugin is in a
ready state.
So use this to determine when the Virtual Camera plugin is ready.

VC_AcquireCamera()

This will allow the script to use the Virtual Camera.
This does the same as pressing the Start Button in the VC plugin.

VC_ReleaseCamera()

Call this when the script has finished with the Virtual Camera.
This does the same as pressing the Stop Button in the VC plugin.

VC_SetCamera()

Camera types like TOWER etc require this to be called to load the tower position into the
camera data.
This does the same as pressing the Select Button in the VC plugin.

Python Interface to the Plugin SDK (V1.02) **(Differences from SDK)**

CameraIndex = VC_GetCameraIndex()

Get index of current camera.

CameraTypeIndex = VC_GetCameraTypeIndex()

Get current camera type index.

XOffset = VC_GetXOffset()

Get current camera X Offset.

YOffset = VC_GetYOffset()

Get current camera Y Offset.

ZOffset = VC_GetZOffset()

Get current camera Z Offset.

HeadingOffset = VC_GetHeadingOffset()

Get current camera Heading Offset.

PitchOffset = VC_GetPitchOffset()

Get current camera Pitch Offset.

RollOffset = VC_GetRollOffset()

Get current camera Roll Offset.

ZoomRatio = VC_GetZoomRatio()

Get current camera Zoom Ratio.

LookAt = VC_GetLookAtAircraft()

Get current aircraft that is being looked at.

Python Interface to the Plugin SDK (V1.02) **(Differences from SDK)**

VC_SetCameraIndex(CameraIndex)

Sets the current camera to CameraIndex.

VC_SetCameraTypeIndex(CameraTypeIndex)

Sets the current camera type to CameraTypeIndex.

VC_SetXOffset(XOffset)

Sets the current camera XOffset.

VC_SetYOffset(YOffset)

Sets the current camera YOffset.

VC_SetZOffset(ZOffset)

Sets the current camera ZOffset.

VC_SetHeadingOffset(HeadingOffset)

Sets the current camera HeadingOffset.

VC_SetPitchOffset(PitchOffset)

Sets the current camera PitchOffset.

VC_SetRollOffset(RollOffset)

Sets the current camera RollOffset.

VC_SetZoomRatio(ZoomRatio)

Sets the current camera ZoomRatio.

VC_SetLookAtAircraft(LookAt)

Sets the current aircraft that the camera will follow..

Python Interface to the Plugin SDK (V1.02) **(Differences from SDK)**

Refer to these enums when using VC_GetCameraTypeIndex() and VC_SetCameraTypeIndex(CameraTypeIndex)

- 0 NOT_SET
- 1 TOWER
- 2 RUNWAY
- 3 CHASE
- 4 FREE1
- 5 FREE2
- 6 SPOT
- 7 FULLSCREEN1
- 8 FULLSCREEN2
- 9 AIRCRAFT0
- 10 AIRCRAFT1
- 11 AIRCRAFT2
- 12 AIRCRAFT3
- 13 AIRCRAFT4
- 14 AIRCRAFT5
- 15 AIRCRAFT6
- 16 AIRCRAFT7
- 17 AIRCRAFT8
- 18 AIRCRAFT9

Refer to these enums when using VC_GetLookAtAircraft() and VC_SetLookAtAircraft(LookAt)

- 0 NOT_SET
- 1 AIRCRAFT0
- 2 AIRCRAFT1
- 3 AIRCRAFT2
- 4 AIRCRAFT3
- 5 AIRCRAFT4
- 6 AIRCRAFT5
- 7 AIRCRAFT6
- 8 AIRCRAFT7
- 9 AIRCRAFT8
- 10 AIRCRAFT9