

DevKitchen 2018

Python in Cinema 4D R20

Python in R20

Changes

New Console

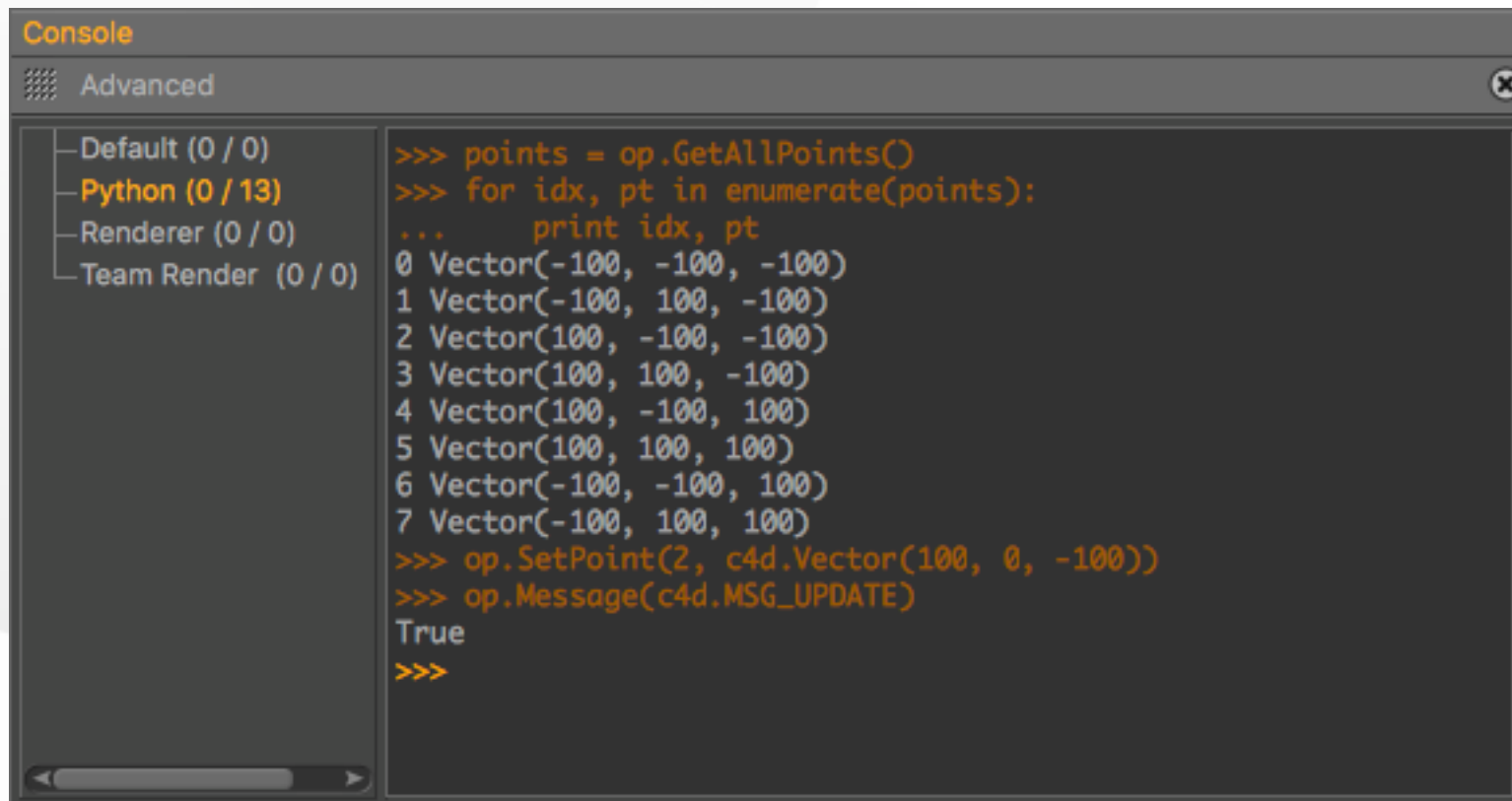
c4dpy

MAXON API

Python in R20 – Changes

- Python API not affected by C++ API backward compatibility break
- See Python SDK documentation for the API changes in R20
Pages [What is New](#) and [API Changelist R20](#)
- Python version 2.7.14
- Preferences
Removed loading at startup of *python_init.py* from *prefs\python*
Moved 3rd party modules/packages location to *python27\libs*

Python in R20 – New Console



The screenshot shows a console window titled "Console" with a sub-tab "Advanced". On the left, there is a tree view with the following items: "Default (0 / 0)", "Python (0 / 13)", "Renderer (0 / 0)", and "Team Render (0 / 0)". The main area of the console displays the following Python code and its output:

```
>>> points = op.GetAllPoints()
>>> for idx, pt in enumerate(points):
...     print idx, pt
0 Vector(-100, -100, -100)
1 Vector(-100, 100, -100)
2 Vector(100, -100, -100)
3 Vector(100, 100, -100)
4 Vector(100, -100, 100)
5 Vector(100, 100, 100)
6 Vector(-100, -100, 100)
7 Vector(-100, 100, 100)
>>> op.SetPoint(2, c4d.Vector(100, 0, -100))
>>> op.Message(c4d.MSG_UPDATE)
True
>>>
```

- Python logger
- Improved text selection
- Drag&drop parameters
- Multi-line statements: for, while, if
- Save console output to text file
 - Advanced ▶ Logger Settings... ▶ Write To File
- Command line field still available
 - Advanced ▶ Command Line...

Python in R20 – c4dpy

- c4dpy = Cinema 4D + Python interpreter
Cinema 4D running in command line mode (no GUI operations)
Python command line and interactive
- app/exe distributed on developers.maxon.net Downloads
The version of Cinema 4D and c4dpy must match
Get pip and install modules/packages
- Greatly improves Python development for Cinema 4D
Write, run and debug Python code in external editors
- Python SDK documentation [c4dpy](#) page

Python in R20 – c4dpy Command Line

```
E:\CINEMA\R20>c4dpy --version
Python 2.7.14

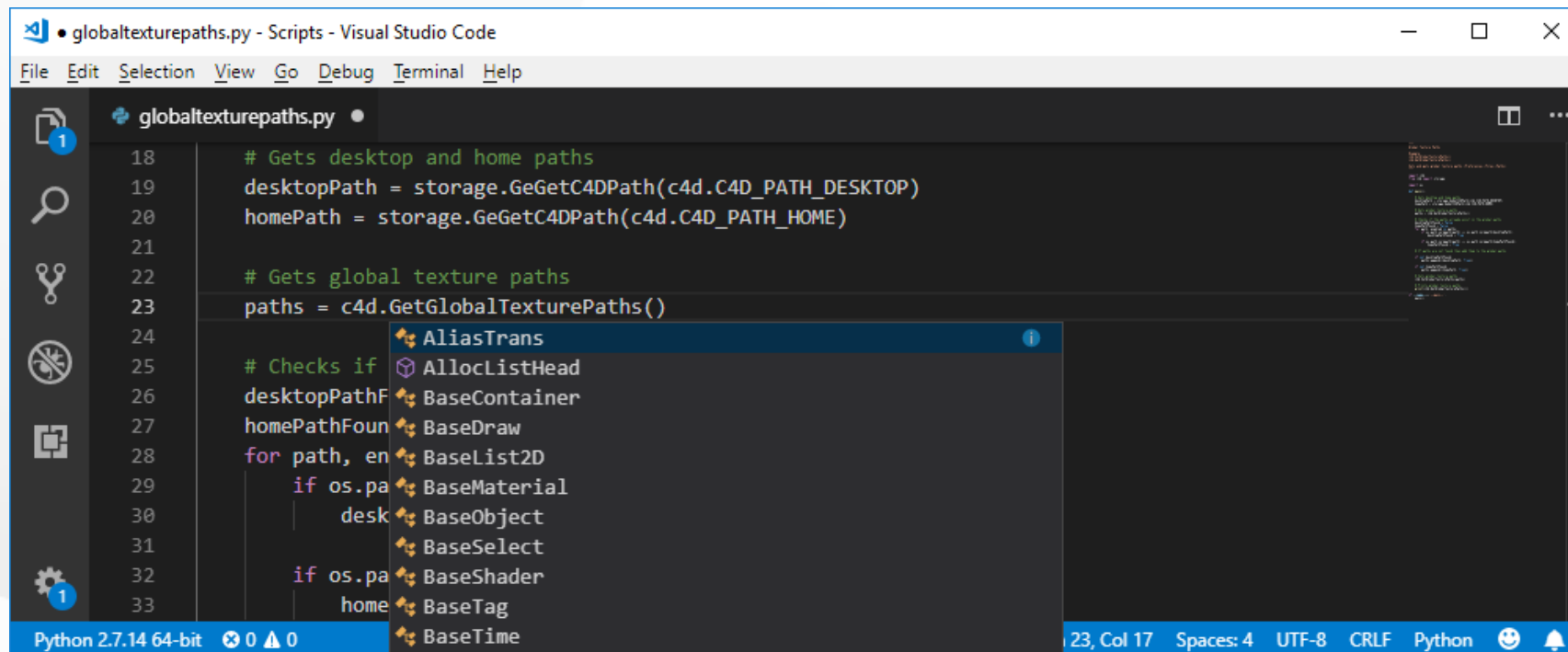
E:\CINEMA\R20>c4dpy -c "print('Hello!')"
Hello!

E:\CINEMA\R20>c4dpy E:\Scripts\cinemaversion.py
20028

E:\CINEMA\R20>_
```

- Arguments:
 - Print Python version: `c4dpy -V` or `--version`
 - Execute code: `c4dpy -c 'some code'`
 - Run script: `c4dpy script.py`
 - Invoke module: `c4dpy -m module`
- No argument: interactive mode

Python in R20 – c4dpy and Code Editors



```
globaltexturepaths.py - Scripts - Visual Studio Code
File Edit Selection View Go Debug Terminal Help
globaltexturepaths.py
18 # Gets desktop and home paths
19 desktopPath = storage.GeGetC4DPath(c4d.C4D_PATH_DESKTOP)
20 homePath = storage.GeGetC4DPath(c4d.C4D_PATH_HOME)
21
22 # Gets global texture paths
23 paths = c4d.GetGlobalTexturePaths()
24
25 # Checks if
26 desktopPathF
27 homePathFoun
28 for path, en
29     if os.pa
30         desk
31
32     if os.pa
33         home
AliasTrans
AllocListHead
BaseContainer
BaseDraw
BaseList2D
BaseMaterial
BaseObject
BaseSelect
BaseShader
BaseTag
BaseTime
```

Python 2.7.14 64-bit 0 0 0 23, Col 17 Spaces: 4 UTF-8 CRLF Python

- Supported Editors
 - Visual Studio Code
 - Install Python extension
 - PyCharm
- Workspaces/projects
- Auto-completion
- Debug scripts

Python MAXON API

Introduction

Goals

Package / Frameworks

Datatypes Conversion

From C++ To Python

Python MAXON API – Introduction

- First exposure in R20
- Still considered in beta stage (almost stable, but lacking features)
- Architecture based on *python.framework*
- No documentation

Python MAXON API – Goals

- Use any MAXON_COMPONENT in Python as you do in C++
- Exposure from C++ to Python mostly automatic
- Integrate concepts of the new core
- Easier maintenance and testing

Python MAXON API – Package

- Located in *{INSTALL}/resource/modules/python/libs/python27/maxon*
- Python *maxon* module = C++ *maxon.core* framework
- To know what's available take a look at *__init__.py*
- Some interfaces are not fully exposed e.g. `BaseArray.Append()`

Python MAXON API – Frameworks

- Located in *frameworks* package subfolder
- One Python file per framework
- In R20 exposed C++ API frameworks: *mesh_misc.py* and *volume.py*
- Third party frameworks can also be exposed!

Python MAXON API – Datatypes Conversion

- Datatypes convertible with `maxon.MaxonConvert()`
 - int to `maxon.Int`
 - float to `maxon.Float`
 - bool to `maxon.Bool`
 - str / Unicode to `maxon.String`

Python MAXON API – Datatypes Conversion

- Datatypes manual conversion
 - list to maxon.BaseArray
 - dict to maxon.DataDictionary
 - tuple to maxon.Tuple / maxon.Pair
 - maxon.BaseArray can be converted to list/set/tuple
 - maxon.DataDictionary can be converted to list/set/tuple of tuple

Python MAXON API – Datatypes Conversion

list to maxon.BaseArray

```
import maxon

pyList = [1, 2, 3, 5, 7]
mArray = maxon.BaseArray(maxon.Int32)
mArray.Resize(len(pyList))

for idx, value in enumerate(pyList):
    mArray[idx] = value
```

Python MAXON API – Datatypes Conversion

tuple to maxon.Tuple

```
import maxon

pyTuple = ('test', 2, 4)
mTuple = maxon.Tuple((maxon.String, maxon.Int32, maxon.Int32))

for idx, value in enumerate(pyTuple):
    mTuple.Set(idx, value)
```


Python MAXON API – Datatypes Conversion

dict to maxon.DataDictionary

```
import maxon

pyDict = {'foo':1, 'something':20.05, 'test':'A string'}
mDict = maxon.DataDictionary()

for key, value in pyDict.iteritems():
    mDict.Set(key, value)
```

Python MAXON API – From C++ to Python

- Supported MAXON_COMPONENT
 - MAXON_INTERFACE
 - MAXON_REFERENCE
 - MAXON_ATTRIBUTE
- MAXON_ENUMS_FLAGS
 - exported automatically to *maxon* module
- MAXON_CONFIGURATION
 - exported automatically to *maxon.config* module

Questions?

plugincafe.maxon.net

