Python on the Landscape of Programming Tools for Design and Architectural Education

Python on the Landscape of Programming Tools for Design and Architectural Education

Alexandre Barrozo do Amaral Villares
FEC-Unicamp, Brazil
abav@lugaralgum.com

Daniel de Carvalho Moreira
FEC-Unicamp, Brazil
damore@fec.unicamp.br

Abstract

Currently most professional modeling and computer graphics software packages embed a scripting language. This is an early report on collecting data about software applications and coding tools geared towards the educational environment, preparing a listing for further evaluation and analysis of platforms. An increase in the adoption of Python as the embedded scripting syntax in many established tools can already be recognized, therefore the creation of educational materials on Python for design and architectural education merits further attention. Other insights on the educational potential of the available tools might be gained by advancing the data collection and evaluation work.

Keywords: Education; Design; Architecture; Programming; Python.

Introduction

Nowadays most CAD and 3D modeling software packages used by architects and designers embed a scripting language, and they have also been used as introductory tools to teach coding from early on. Mark Burry in Scripting Cultures: Architectural design and programming (2011), identifies different uses of computer programming, like productivity aids and exploratory code, as well as describes his teaching experience using scripting tools embedded in CAD software:

"[...] Once I had a handle on this coding caper, I could see then that I could attempt to transcend whatever limitations software might impose on me as a designer, guiding this electronic instrument with the same authority I applied to my pen and compass. [...] The two problems that had encouraged me to step outside my professional comfort zone of compliant passenger to become front-seat driver were stimulated by a need to rid myself of repetitive work [...]. Very quickly, however, I could see that a prime motivation for coding on top of software was to augment my design practice by allowing me to work in ways hitherto impractical, and so scripting became a medium of experimentation ahead of productivity gain. [...] Within a year, scripting had infiltrated my teaching, and in 1993 I instituted an elective course in which the participants had to come up with two pieces of code: a productivity tool (this was to appease my senior CAD teaching colleagues and satisfy the school's curriculum priorities) and a design experimentation script. [...]"(p.29)

Burry also compiled a list of programming tools used by several correspondents. It included textual scripting languages, like VBA, and tools with node diagrams, like Generative Components, embedded on mainstream computer graphics applications, like AutoCAD, as well as stand alone development environments, like Processing (Reas & Fry, 2011).

Collecting and organising information about programming languages embedded in applications used by designers and architects, and also coding tools with educational aim, could be useful for teachers who need to choose suitable tools and supporting resources. Others researchers in the field of design and architectural education might engage in further investigations like comparing platforms (Celani & Vaz, 2012) and analyzing trends.

Resilience Design is a contemporary aspect of design that might greatly benefit from model simulations, tackling design complexity and other exploratory strategies that are most effective through engagement in coding by design practitioners.

Methods

The records started to be collected in 2015 and the following selection criteria was adopted:

- Drawing or 3D modeling software that embeds a scripting language on the user interface or allows automation with a very limited number of steps between programming and code execution.

- Tools aimed at teaching programming in a visual or graphic context.

Software Development Kits (SDKs), usually provided by mainstream computer graphics software houses, mostly offering C or C++ resources, were excluded. These tools are aimed at professional programmers who wish to create

plugins, and are not immediately accessible to designers and architects, the main application end users.

The following fields are being recorded:

- Evaluation level: 'Yes' means the authors saw first hand the tool being used or used them themselves, 'partial' means some documentation of use was consulted at the time of entry, 'no' means only a description of the tool was available, possibly provided by a correspondent;
- OS: Describes if the tool will run on MacOS, Windows and/or GNU/Linux;
- License: 'FOSS' (Free/Libre and Open Source Software), or 'Proprietary', and in this last case, reference to, if available, any educational license, free or at reduced cost;
- Host/IDE: The main CAD, modelling, game application or Integrated Development Environment to be used. (i.e. SketchUp, Rhinoceros, Minecraft, or Processing IDE);
- Language/Library: Other software components or libraries that enable the programming interface. (i.e. Ruby, Grasshopper, ComputerCraftEdu, or P5*JS);
- Syntax: An attempt to group tools by language syntax features (i.e. Ruby, nodes, blocks, or JavaScript). 'Nodes' groups visual scripting tools like Generative Components, Grasshopper and Marionette). 'Blocks' groups programming interfaces similar to Scratch;
- Main Uses: A brief description of uses encountered;
- Official Site: Attempt to locate an online resource regarded as the official source of the tool;
- Recommended reference: Attempt to locate useful reference resources.

Many entries have not yet been evaluated, and the evaluation and categorization methods themselves require further work. It can be noted there is some overlap between categories due to the way entries have been recorded. The research would benefit from clearer criteria for listing platforms with several scripting and syntax options (at this point some are listed as separate entries, others grouped together).

Results

The preliminary results are presented in Table 1, also available as a CSV table published on GitHub inviting collaboration by means of corrections and additions: <https://github.com/villares/Resources-for-teaching-programming>

At the time of this writing 43 tools have been listed, of which 32 at least superficially investigated, of which at least 20% (7 of 32) have substantial educational aims.

A preliminary look at this landscape seems to indicate strong presence of Python as an embedded language or option in programming tools, about 40% (18 of 43 listed, 14 of 32 partially reviewed entries), followed by visual/node based tools (like GC, Dynamo and Grasshopper) about 15% (5 of 32), Lua and BASIC-related (like GLD and VB.NET) about 15% (5 of 32).

Discussion

There seems to be a trend of adoption of Python as an embedded scripting language. Python was added to Rhinoceros, arguably as replacement of RhinoScript, added to Vectorworks replacing VectorScript (a Pascal based language) and similarly Python was added to Maya, 3D Max and Cinema 4D.

On the Free/Libre and Open-Source category, Blender and FreeCAD have strong Python integration from the start. Rosetta, a CAD controlling extension used within DrRacket IDE, can now be accessed via Processing or Python as well (Caetano & Leitão, 2016; Ramos & Lleitão, 2014). Recently the Processing Foundation incorporated the Processing Python Mode as an official project (Parrish, 2016).

It can be noted that the Python programming language has grown in use at introductory courses in Computer Science and Engineering (Guo, 2014) and many other educational environments. According to Tollervey (2015), Python popularity in education might be explained from its origins on ABC, designed for teaching and aimed at nonprofessional programmers as well as on the open-source and extensible platform ecosystem “capable of simply and effectively addressing many different types of computational problems”.

This leads to the preliminary conclusion that documenting Python use, and related software tools, as well as the creation of educational materials on Python for design and architectural education merit further work. It could be investigated, as further research, if knowledge of Python can be usefully transferred by students between platforms and tools.

Other insights on the evolution of the available scripting tools might be expected to follow from the accumulation of detailed data, it is mostly the intention of this work to provide a useful database for other researchers and specially teachers willing to explore further those tools.

Table 1: Data collected up to May 2017 (part 1 of 2)

ID	Evaluation	OS	License	Host/IDE	Language/Library	Syntax	Main uses
10	yes	Windows, Mac OS	Proprietary, educational use at no cost	ArchiCAD	GDL	*BASIC-like*	Parametric objects, complex 3D geometry
20	yes	Windows, Mac OS, Linux	FOSS	Arduino	C/Wiring	C-like	Physical computing, home automation, interactive art.
30	yes	Windows	Proprietary, no cost licence	Bentley	Generative Components	*nodes*	generative 3D geometry, complex 3D geometry
40	yes	Windows, Mac OS, Linux	FOSS	Blender	Python	Python	Plug-in tools, complex 3D geometry
50	yes	Windows, Mac OS	FOSS	Cinder	C++	C-like	2D graphics, generative art, dataviz
60	yes	Mac OS	FOSS	Drawbot	Python	Python	2D graphics, generative art, dataviz
70	yes	Windows, Mac OS, Linux	FOSS	Processing	Processing Java Mode	Java	2D and 3D graphics, intetactive art, generative art, dataviz
71	yes	Windows, Mac OS, Linux	FOSS	Processing	Processing Python Mode	Python	2D and 3D graphics, intetactive art, generative art, dataviz
72	yes	Windows, Mac OS, Linux	FOSS	P5*JS (Processing Foundation)	P5*JS	JavaScript	2D and 3D graphics, intetactive art, generative art, dataviz
80	yes	Windows	Proprietary, educational use at no cost or reduced cost	Revit, Vasari & Dynamo Studio	Dynamo	*nodes*, DesignScript	Generative geometry in 3D
90	yes	Windows	Proprietary, educational at reduced cost	Rhinoceros	Grasshopper	*nodes*	Generative geometry in 3D
91	yes	Windows, Mac OS	Proprietary, educational at reduced cost	Rhinoceros	Python	Python	Plug-in tools, complex 3D geometry
92	yes	Windows, Mac OS	Proprietary, educational at reduced cost	Rhinoceros	Rhinoscript	VBScript	Plug-in tools, complex 3D geometry
100	yes	Windows, Mac OS, Linux	FOSS	Scratch	Scratch	*blocks*	Education, games
110	yes	Windows, Mac OS	Proprietary, educational use at no cost or reduced cost	SketchUp	Ruby	Ruby	Plug-in tools, complex 3D geometry
120	yes	Windows, Mac OS	Proprietary, educational use at no cost	Vectorworks	Marionette	*nodes*	Generative geometry in 3D
121	yes	Windows, Mac OS	Proprietary, educational use at no cost	Vectorworks	Vectorscript	Pascal	Plug-in tools, parametric objects, complex 3D geometry
122	yes	Windows, Mac OS	Proprietary, educational use at no cost	Vectorworks	Python	Python	Plug-in tools, parametric objects, complex 3D geometry
130	yes	online	-	Trinket.io (Browser based)	Python	*blocks*, Python, JavaScript	Education
140	partial	Windows	Proprietary	ArcGIS	Python	Python	Plug-in tools, GIS, dataviz
150	partial	Windows	Proprietary, educational use at no cost	Autocad	AutoILISP & VisualLISP	Lisp-like	Plug-in tools, complex 2D and 3D geometry
151	partial	Windows	Proprietary, educational use at no cost	Autocad	VB.NET	VB.NET	Plug-in tools, complex 2D and 3D geometry
160	partial	Windows, Mac OS, Linux	FOSS	FreeCAD	Python	Python	Plug-in tools, complex 2D and 3D geometry
170	partial	Windows, Mac OS, Linux	Proprietary	Minecraft	[various]	blocks, Python, JavaScript, Lua	Education, games
180	partial	Windows, Mac OS, Linux	FOSS	NodeBox 3	NodeBox	*nodes*, Python	2D graphics, generative art, dataviz
181	partial	Windows, Mac OS, Linux	FOSS	NodeBox for OpenGL	Python	Python	2D graphics, generative art, dataviz
190	partial	Windows, Mac OS, Linux	FOSS	OpenFrameworks	C++	C-like	2D and 3D graphics, interactive art, dataviz
200	partial	Mac OS (aplicativo ou módulo Python), Windows e Linux (módulo Python)	FOSS	PlotDevice	Python	Python	2D graphics, generative art, dataviz
210	partial	Windows, Mac OS, Linux	FOSS	QGIS	Python	Python	Plug-in tools, GIS, dataviz
81	partial	Windows	Proprietary, educational use at no cost	Revit	Revit Macros (VB.NET or C#)	VB.NET, C-like	Plug-in tools, generative 3D geometry, complex 3D geometry
220	partial	online	-	repl.it (Browser based)	Python [and many others]	Python [and others]	Education
230	partial	Windows, Mac OS, Linux	FOSS	DrRackett	Racket/Rosetta	Lisp-like, Python, Java	Education, generative 3D geometry, complex 3D geometry
82	no	Windows	Proprietary, educational use at no cost	Revit, Vasari	RevitPythonShell	Python	Plug-in tools, complex 3D geometry
240	no	Windows, Mac OS, Linux		BlueJ	Java	Java	Education
250	no	Windows, Mac OS, Linux		Greenfoot	Java	Java	Education
260	no	Windows	Proprietary	3D Max	Python	Python	3D
261	no	Windows	Proprietary	3D Max	MAXScript	MAXScript	3D
270	no	Windows, Mac OS	Proprietary	Cinema 4D	Python	Python	3D
271	no	Windows, Mac OS	Proprietary	Cinema 4D	Coffee	Coffee	3D
280	no		Proprietary	Maya	Python	Python	3D
290	no			Unity	C#	C-like	games, dataviz
300	no			Unreal Engine	Blueprints	*blocks*	games, dataviz
310	no	Windows, Mac OS, Linux	Proprietary	Stencyl	Stencyl	*blocks*, Java	Education, games

Table 1: Data collected up to May 2017 (part 2 of 2)

ID	Official site	Recommended Ref.
10	gdl.graphisoft.com/	GDL Cookbook
20	arduino.cc	
30	bentley.com/en/products/product-line/modeling-and-visualization-software/generativecomponents	
40	blender.org/manual/advanced/scripting/introduction.html	
50	libcinder.org/	
60	www.drawbot.com/	
70	processing.org	[listed separately on the GitHub repository]
71	py.processing.org	[listed separately on the GitHub repository]
72	p5js.org	[listed separately on the GitHub repository]
80	dynamobim.org/	
90	grasshopper3d.com/	
91	wiki.mcneel.com/developer/python	
92	wiki.mcneel.com/developer/rhinoscript	
100	scratch.mit.edu/	scratch.mit.edu
110	www.sketchup.com/intl/en/developer/	Livro Automatic SketchUp
120	developer.vectorworks.net/index.php/Marionette	www.techlimits.com/index.php/vectorworks-2016-marionete
121	developer.vectorworks.net/index.php/VectorScript	https://stackoverflow.com/questions/263774/where-can-i-find-resources-on-vectorscript-programming-language
122	developer.vectorworks.net/index.php/Python	
130	trinket.io	hourofpython.com
140	resources.arcgis.com/en/communities/python/	
150	help.autodesk.com/view/ACD/2016/ENU/?guid=GUID-265AADB3-FB89-4D34-AA9D-6ADF70FF7D4B	
151	http://help.autodesk.com/view/ACD/2016/ENU/?guid=GUID-2CD40631-D67B-4DF0-A2C4-606E9B613252	
160	www.freecadweb.org	http://www.freecadweb.org/wiki/index.php?title=FreeCAD_Scripting_Basics
170		http://computercraftedu.com/ http://pi.minecraft.net/?page_id=14
180	www.nodebox.net/node/	
181	www.cityinabottle.org/nodebox/	
190	openframeworks.cc/	
200	plotdevice.io	http://plotdevice.io/extras/plotdevice-docs.zip
210	docs.qgis.org/testing/en/docs/pyqgis_developer_cookbook/intro.html	qgistutorials.com/en/docs/getting_started_with_pyqgis.html
81		https://knowledge.autodesk.com/support/revit-products/learn-explore/caas/CloudHelp/cloudhelp/2016/ENU/Revit-Customize/files/GUID-4DFDA8CD-B0FD-492E-8EDE-A28C29B1E316-htm.html
220	repl.it	
230	http://web.ist.utl.pt/antonio.menezes.leitao/Rosetta/tutorials/introduction.html	http://web.ist.utl.pt/antonio.menezes.leitao/Rosetta/tutorials/introduction.html
82	github.com/architecture-building-systems/revitpythonshell/blob/master/README.md	
240	www.bluej.org/	http://www.bluej.org/objects-first/
250	www.greenfoot.org/door	http://www.greenfoot.org/book/
260	docs.autodesk.com/3DSMAX/16/ENU/3ds-Max-Python-API-Documentation/index.html	
261		
270		
271		
280		
290		https://unity3d.com/learn/tutorials/topics/scripting
300		https://docs.unrealengine.com/latest/INT/Engine/Blueprints/GettingStarted/index.html
310	http://www.stencyl.com/	

Acknowledgments

The authors would like to thank the many correspondents who have contributed to this research over the last two years.

References

Burry, M. (2011). Scripting cultures: architectural design and programming. Chichester: Wiley.

Reas, C., & Fry, B. (2011). Processing Architecture. Perspecta, 44, 153–202. https://doi.org/10.2307/41662956

Celani, G., & Vaz, C. E. V. (2012). CAD Scripting and Visual Programming Languages for Implementing Computational Design Concepts: A Comparison from a Pedagogical Point of View. International Journal of Architectural Computing, 10(1), 121–137. https://doi.org/10.1260/1478-0771.10.1.121

Guo, P. (2014). Python is now the most popular introductory teaching language at top us universities. BLOG @ Communications of the ACM, July, 47. http://cacm.acm.org/blogs/blog-cacm/176450-python-is-now-the-most-popular-introductory-teaching-language-at-top-us-universities/fulltext

Ramos, P. P., & Leitão, A. M. (2014). Implementing Python for DrRacket. In M. J. V. Pereira, J. P. Leal, & A. Simões (Eds.), 3rd Symposium on Languages, Applications and Technologies (Vol. 38, pp. 127–141). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. https://doi.org/10.4230/OASIcs.SLATE.2014.127

Tollervey, N. (2015). Python in Education. Sebastopol, CA: O'Reilly Media. Retrieved from http://www.oreilly.com/programming/free/python-in-education.csp

Caetano, I., & Leitão, A. M. (2016). Using Processing with Architectural 3D Modelling. Herneoja, Aulikki; Toni Österlund and Piia Markkanen (Eds.), Complexity & Simplicity - Proceedings of the 34th ECAADe Conference - Volume 1, University of Oulu, Oulu, Finland, 22-26 August 2016, Pp. 405-412. Retrieved from http://papers.cumincad.org/cgi-bin/works/Show?ecaade2016_199

Parrish, A., Fry, B., & Reas, C. (2016). Getting Started with Processing.py: Making Interactive Graphics with Processing's Python Mode (1 edition). San Francisco, CA: Maker Media, Inc.