First steps in creative coding, learning Python while making drawings and animations

Alexandre B A Villares

abav.luguaralgum.com/links

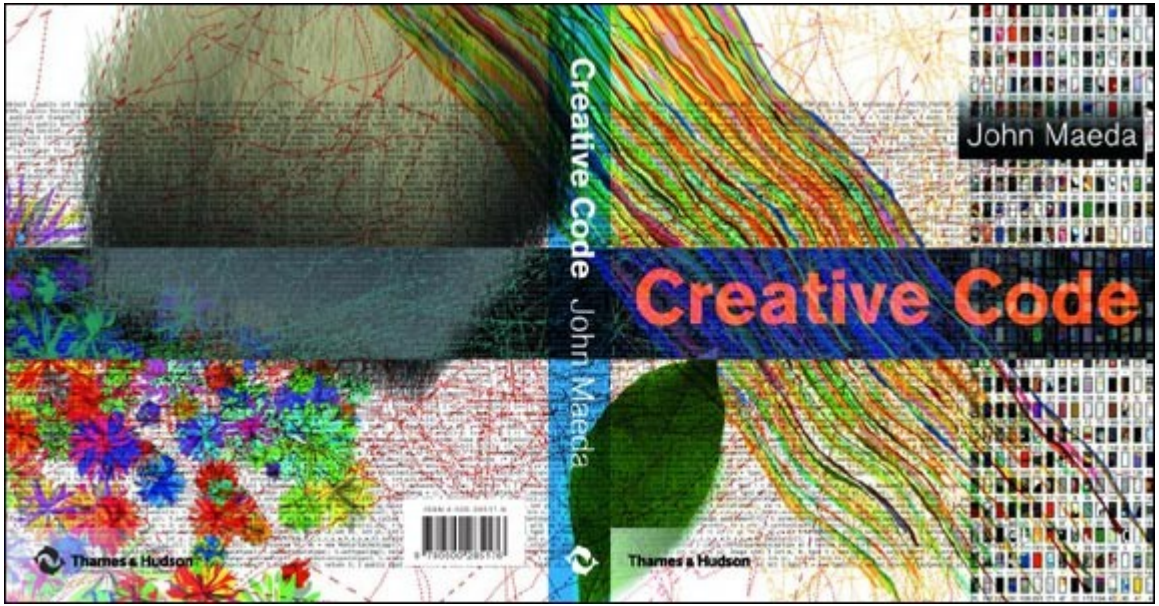
Alexandre B A Villares

Teacher, visual artist and consultant in design and new media projects, doing research on computational art procedures, generative design, creative coding and how to teach programming in visual contexts. Graduated in architecture and urbanism from FAU-USP (2000) and has a master's degree from FECFAU-Unicamp (2019), were he is currently a PhD candidate. Alexandre has taught courses at undergraduate and graduate levels at several institutions, he develops open didactic materials, collaborates with free software projects and works as a technology and arts educator at Sesc-SP. Since 2018, he tries to make one drawing a day with code.

https://abav.lugaralgum.com
https://ciberlandia.pt/@villares

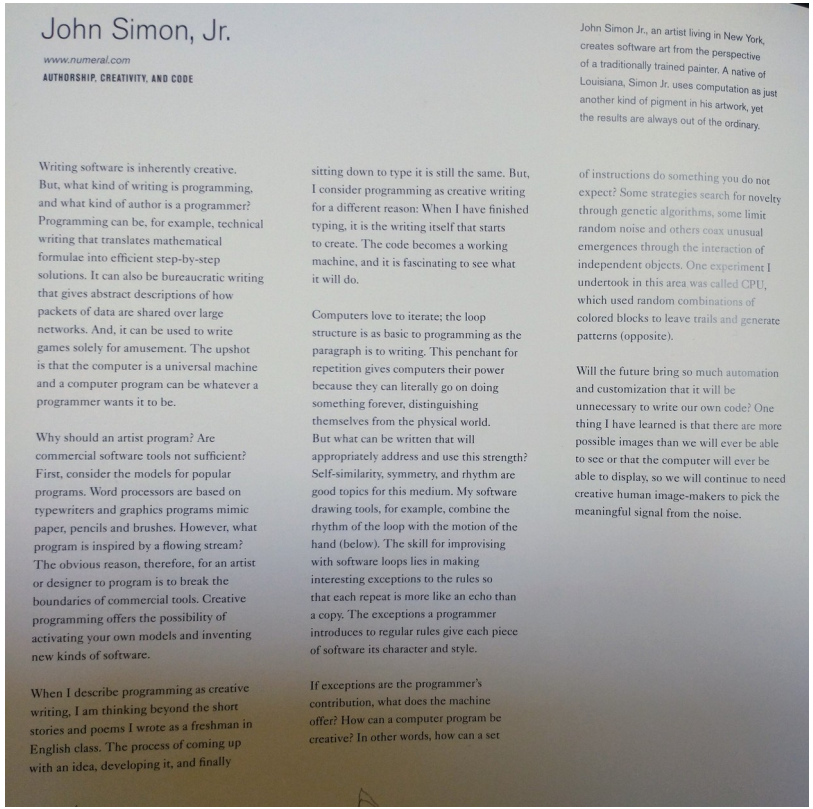
“Creative Coding” term is John Maeda’s fault

Artists have been using code to produce images and express themselves since the '60s. Lot’s of names have been used, with weird implications, like Generative Art, New Media Art, Algorithmic Art, etc.



Maeda, 2004 Creative Code: Aesthetics + Computation
http://maedastudio.com/2004/creativecode/

Why should an artist program?



John Simon, Jr.

www.numeral.com

AUTHORSHIP, CREATIVITY, AND CODE

John Simon Jr., an artist living in New York, creates software art from the perspective of a traditionally trained painter. A native of Louisiana, Simon Jr. uses computation as just another kind of pigment in his artwork, yet the results are always out of the ordinary.

Writing software is inherently creative. But, what kind of writing is programming, and what kind of author is a programmer? Programming can be, for example, technical writing that translates mathematical formulae into efficient step-by-step solutions. It can also be bureaucratic writing that gives abstract descriptions of how packets of data are shared over large networks. And, it can be used to write games solely for amusement. The upshot is that the computer is a universal machine and a computer program can be whatever a programmer wants it to be.

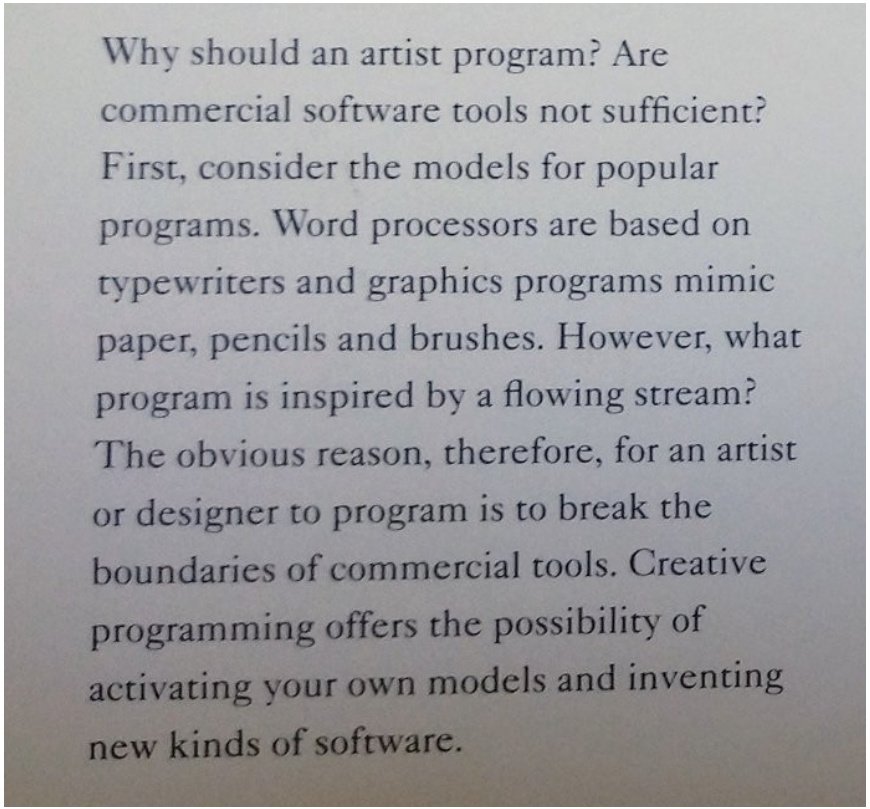
Why should an artist program? Are commercial software tools not sufficient? First, consider the models for popular programs. Word processors are based on typewriters and graphics programs mimic paper, pencils and brushes. However, what program is inspired by a flowing stream? The obvious reason, therefore, for an artist or designer to program is to break the boundaries of commercial tools. Creative programming offers the possibility of activating your own models and inventing new kinds of software.

When I describe programming as creative writing, I am thinking beyond the short stories and poems I wrote as a freshman in English class. The process of coming up with an idea, developing it, and finally sitting down to type it is still the same. But, I consider programming as creative writing for a different reason: When I have finished typing, it is the writing itself that starts to create. The code becomes a working machine, and it is fascinating to see what it will do.

Computers love to iterate; the loop structure is as basic to programming as the paragraph is to writing. This penchant for repetition gives computers their power because they can literally go on doing something forever, distinguishing themselves from the physical world. But what can be written that will appropriately address and use this strength? Self-similarity, symmetry, and rhythm are good topics for this medium. My software drawing tools, for example, combine the rhythm of the loop with the motion of the hand (below). The skill for improvising with software loops lies in making interesting exceptions to the rules so that each repeat is more like an echo than a copy. The exceptions a programmer introduces to regular rules give each piece of software its character and style.

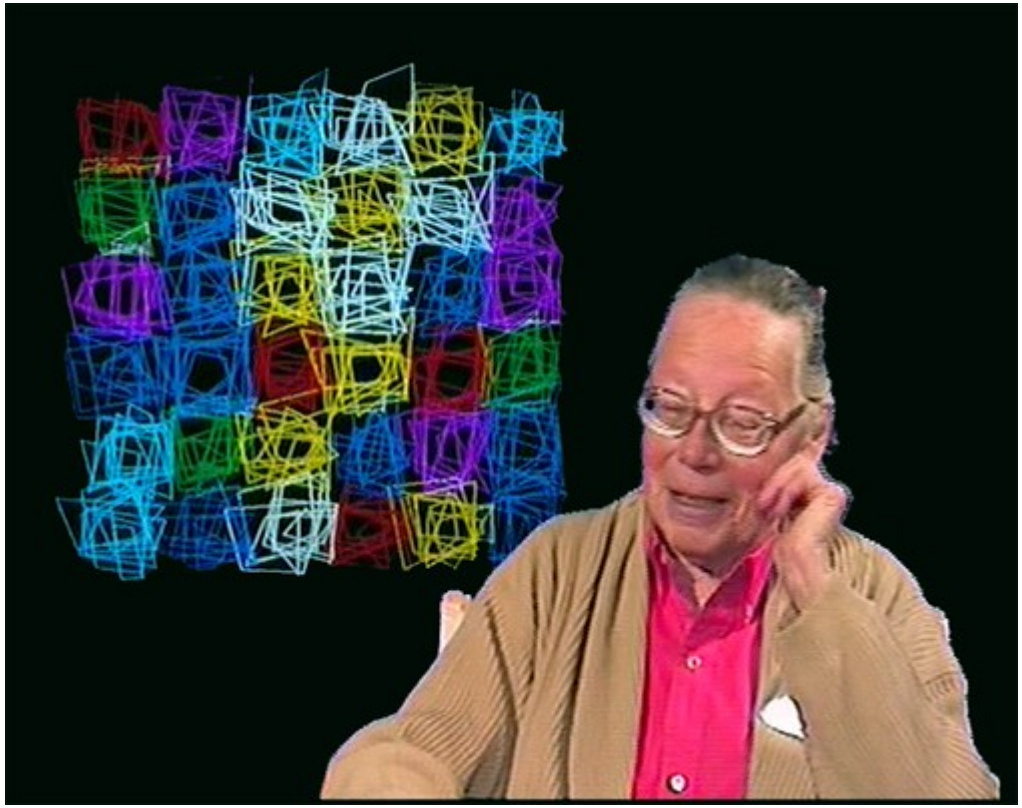
If exceptions are the programmer's contribution, what does the machine offer? How can a computer program be creative? In other words, how can a set of instructions do something you do not expect? Some strategies search for novelty through genetic algorithms, some limit random noise and others coax unusual emergences through the interaction of independent objects. One experiment I undertook in this area was called CPU, which used random combinations of colored blocks to leave trails and generate patterns (opposite).

Will the future bring so much automation and customization that it will be unnecessary to write our own code? One thing I have learned is that there are more possible images than we will ever be able to see or that the computer will ever be able to display, so we will continue to need creative human image-makers to pick the meaningful signal from the noise.



Why should an artist program? Are commercial software tools not sufficient? First, consider the models for popular programs. Word processors are based on typewriters and graphics programs mimic paper, pencils and brushes. However, what program is inspired by a flowing stream? The obvious reason, therefore, for an artist or designer to program is to break the boundaries of commercial tools. Creative programming offers the possibility of activating your own models and inventing new kinds of software.

Some inspiration: Vera Molnár



<https://veramolnar.com>

Processing

"Our mission is to promote software literacy within the visual arts, and visual literacy within technology-related fields — and to make these fields accessible to diverse communities. Our goal is to empower people of all interests and backgrounds to learn how to program and make creative work with code, especially those who might not otherwise have access to these tools and resources"

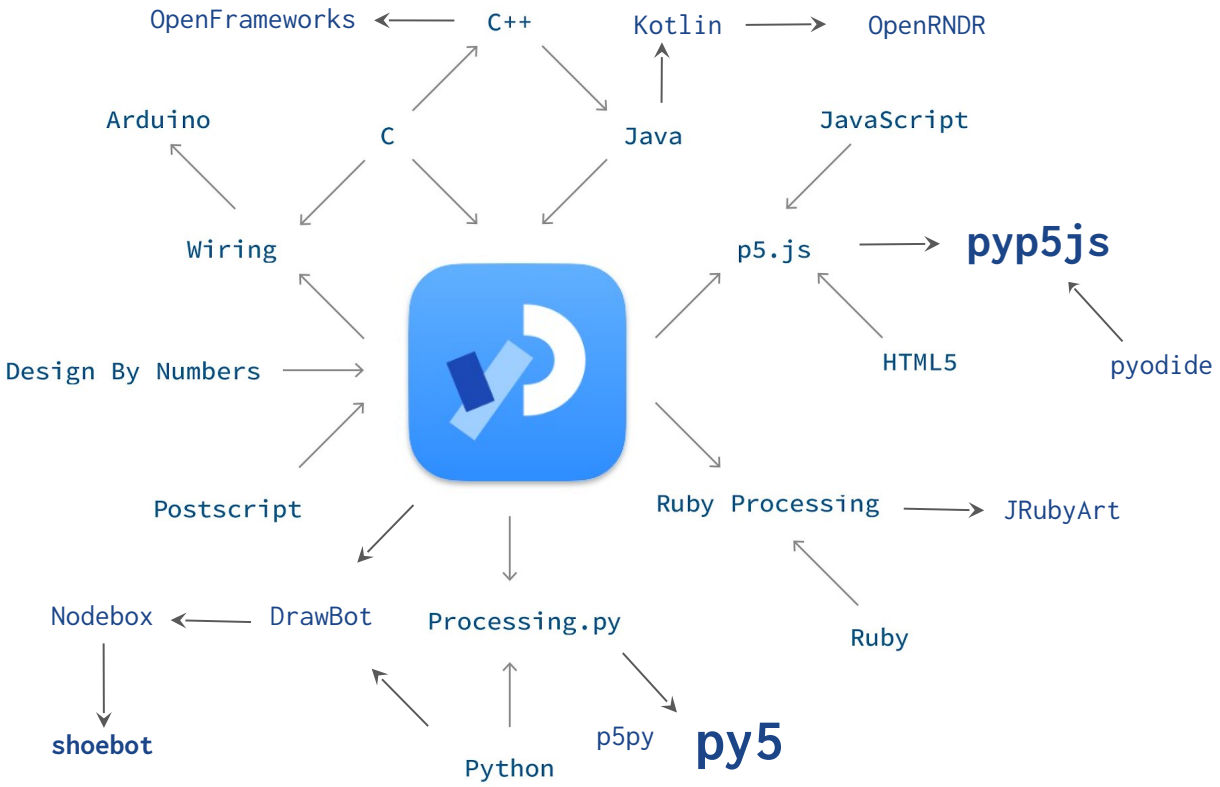
`<processing.org>`

Python

Python is a general-purpose programming language, widely adopted in the software industry, in scientific research, in data analysis and visualization, by journalists, activists, designers and people from the most diverse areas, and also in introductory computing education... <python.org>

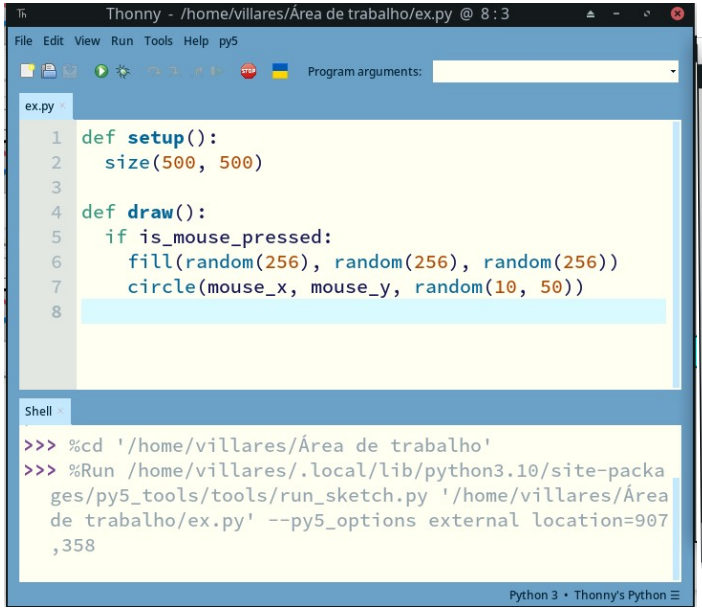
Some drawing, design and 3D tools that embed Python:
Blender, Drawbot, shoebot, NodeBox, Revit/Dynamo/Vasari, Rhinoceros, Rhino+Grasshopper, Vectorworks, ArcGIS, FreeCAD, PlotDevice, QGIS, 3D Max, Cinema 4D, Maya, RoboFont, Glyphs, FontLab, FontForge. <https://github.com/villares/Resources-for-teaching-programming>

Don’t worry about this slide!
It just shows we are in good company

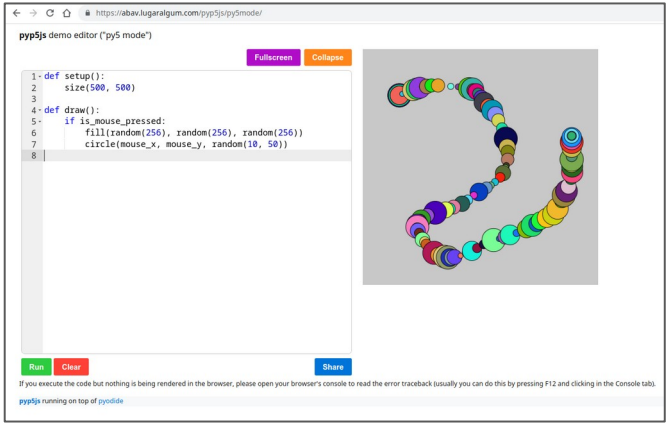


About tools we are going to use

Thonny IDE comes with Python 3, and we add a plug-in named **thonny-py5mode**
To make it easier to use **py5** https://abav.lugaralgum.com/como-instalar-py5/



Or the **online editor**
https://abav.lugaralgum.com/pyp5js/py5mode/



So let’s go!

hackmd.io/@villares/pycon-us-2024

Thank you!