LIBER AMICORUM

A handbook for software artists.

Liber Amicorum

Liber Amicorum

by

Published by O'Reilly Media, 1005 Gravenstein Highway, Sebastopol, CA 95472.

Editors: Isabel Draves and Andrew Odewahn

Revision History for the :

See *http://oreilly.com/catalog/errata.csp?isbn=* for release details.

Table of Contents

Foreword

A brilliant, insightful, and moving foreword goes here.

Preface

An informative preface goes here.

Acknowledgments

The following people were critical to this project:

- **Matthew Epler** is a graduate student at NYU's Interactive Telecommunications Program (ITP) specializing in computational art and archives.
- Jorge Just led RapidFTR from student assignment to international volunteer driven software project that was recently deployed in a refugee camp in Uganda. He teaches at ITP.
- Sanders Kleinfeld
- Rune Madsen
- Nellie McKesson
- Sara Winge
- Adam Witwer
- Jen Robbins

CHAPTER 1

Overview

Creative Coding. Software Art. It's a pan-disciplinary field so new that those working in it haven't yet settled on a name. Practitioners are artists, designers, sculptors, computer scientists, vision researchers, hardware hackers—anyone who makes art through the creative application of technology.

Liber Amicorum is a collaborative, community-driven project to map the state of the art and create a useful resource for both new and experienced software artists. Andrew Odewahn of O'Reilly Media, Isabel Draves of LISA (Leaders in Software and Art), and select members of the community will serve as curators. It will operate as open source project on GitHub, and be maintained by O'Reilly. Through Liber Amicorum, we aim to:

- Create a resource—equal parts inspiration, guide, and catalog—that maps this emerging space, advances practitioners'understanding of it, and shows that there's a there there.
- Document inspirational projects—not only the amazing finished product, but the workflow, tools, and hacks the artist used in the process of creating it.
- Help practitioners identify and find each other.
- Surface and address practitioners' pain points and stumbling blocks. For example, what happens when an artist's creative vision is held up by a lack of technical or mathematical knowledge?
- Explore the use of GitHub as a tool for collaborative authoring. How does this new platform, which has fueled community-driven innovation in the software world, serve the creative community?

We'll publish Liber Amicorum periodically as a PDF, on the iBookstore, on the Kindle, and, in the not-too-distant future, on the web.

And that title? *Liber Amicorum* means the "book of friends." We thought that was a pretty good description for what we want to do and how we want to do it. (And, we liked it a lot better than "guide" or "directory").

What We're Looking For

- Useful tools or techniques
- Inspirational projects
- Hands-on tutorials or articles [Coming Soon]
- Sites and resources of interest to practitioners
- Key people or organizations

Areas / Fields

We're interested primarily in these areas:

- 3D graphics
- Algorithmic and Generative Art
- Animation
- App Design
- Circuit Bending
- Data visualization
- Electromechanical sculpture
- Interaction Design
- LED design
- Musical / Sound Arts
- Physical Computing

Please note that we're not seeking to *define* these areas in particular — that's better done on Wikipedia or similar sites. We're interested mainly in identifying trends, tools, people, and projects that are advancing the state of the art in the field.

How the Project is Structured

The project currently has 3 main parts: organizations, tools, and projects.

The **organizations** section lists the groups that are doing interesting and innovative work in the field. This can range from a full academic program to a hackerspace to a conference to a gallery.

The **tools** section attempts to map the key tools. The goal is not so much to provide an exhaustive reference, but more to give a good overview of what the tool is, who the heavyweights in the field are, and list some interesting work. Here are a few examples:

- arduino
- cinder
- kinect
- max
- 3D printers

For each tool, we'd like to see:

- The quickest possible description. We're not looking to recreate wikipedia.
- What it's commonly used for
- Important details, like whether it's free software
- The heavyweights in the field
- A list of interesting projects and examples
- Resources and community support
- Similar tools

Finally, the **projects** section explores a particular projectin more depth. The goal is to articulate the vision and thoughts that go into creating an interesting work or solving an interesting problem, with a particular focus on the project's relevance compared to other computational / serial artists or projects.

We're much more interested in the "why's" of the project — why did you pick certain tools, why did you take the approach, why is this different than what's come before — before the "how's". (But, don't worry — you can put the How's in the tutorial or example sections.) The project descriptions should serve as a guide of your thinking for people who will follow after you.

- What was the project?
- What areas or field does it fall in?
- What tools did you use?
- Why was this an interesting project to you — what was new or innovative?
- What were the key challenges you had to overcome?

- Your name and a brief bio

How to Contribute

The simplest way to get started is to Liber Amicorum prject on GitHub. And if you're wondering, "Why GitHub?," Clay Shirky explains its transformative superpowers in his talk How the Internet will (one day) transform government.

To contribute to Liber Amicorum:

- Create a GitHub account (If you don't have one already)
- Go to the Liber Amicorum repo
- Fork the repository, or just go to the file you want to edit and click "Edit"
- Enter your changes in AsciiDoc format
- Send a pull request

We'll use the feedback and discussion generated from your pull request to decide whether to accept the contribution. Once we do, we'll generate a fresh version of the guide in PDF, EPUB (for iBooks), and Mobi (for Kindle) formats.

License

CHAPTER 2

Organizations

Here are some of the key organizations involved in the Creative Coding movement.

Academic Programs

ITP

ITP is a two-year graduate program located in the Tisch School of the Arts whose mission is to explore the imaginative use of communications technologies — how they might augment, improve, and bring delight and art into people's lives. Perhaps the best way to describe us is as a Center for the Recently Possible.

CMU program

TK

MIT Media Lab

TK

Conferences and Events

LISA

The LISA Conference is…

SIGGRAPH

TK

CHAPTER 3

Tools

This section attempts to map the key tools. The goal is not so much to provide an exhaustive reference, but more to give a good overview of what the tool is, who the heavyweights in the field are, and list some interesting work. Here are a few examples:

- arduino
- cinder
- kinect
- max
- 3D printers

For each tool, we'd like to see:

- A succinct description; we're not looking to recreate Wikipedia
- What the tool is commonly used for
- Important details, such as whether it's free software
- Heavyweights in the field who use the tool
- A list of interesting projects and examples
- Resources and community support
- Similar tools

Arduino

Arduino—one of the most successful open-source hardware projects ever—is a micro-controller and programming environment intended primarily for artists, designers, hobbyists, and students. It is particularly useful for quick prototyping because of the flexibility and ease-of-use of both the software and hardware. The software includes an IDE built on Processing and Wiring.

Both the hardware and software are open source, so you can build-your-own Arduino, or buy one pre-assembled. In addition, many specialized versions have been created for specific uses, such as the Lilypad Arduino for wearable computing (official Arduino project) and the Boarduino, a breadboard-compatible clone. *http://www.ladyada.net/make/boarduino/*

Quickest Possible Description

Description
A micro-controller and programming environment

Site
http://arduino.cc/

Git / Repo
- *https://github.com/arduino/Arduino*
- *http://code.google.com/p/arduino/*

Wikipedia
http://en.wikipedia.org/wiki/Arduino

Twitter
http://twitter.com/arduinoteam

Other wiki/resource
http://www.arduino.cc/playground/

Frequently Used For

Major uses include:

- Physical Computing
- Interactive art and objects
- Translating physical inputs (sound, light, touch, temperature, vibration) into digital applications
- Wearables

Important Details

Details include:

- Both the hardware and software are open source.
- The software is released under the GNU GPL V2 (IDE) and LGPL (micro-controll * libraries).
- The hardware is released under an Creative Commons Attribution Share-Alike license.

Heavyweights

The heavyweights in the field include:

- Massimo Banzi
- David Cuartielles
- Tom Igoe
- Gianluca Martino
- David Mellis

Projects & Examples

Interesting projects and examples include:

- Massimo Banzi: How Arduino is open-sourcing imagination
- Short++
- MudTub
- KickBee

Resources / Community / Support

Because the Arduino project is so well-organized and supported, the very best resource is the Arudino site itself. Some of the most useful sections are included below

Tutorials

- Arduino tutorial site
- Getting Started With Arduino
- Arduino Video Tutorials

Books

- Getting Started With Arduino by Massimo Banzi
- Making Things Talk by Tom Igoe
- Arduino Cookbook by Michael Margolis
- iOS Sensor Apps with Arduino by Alasdair Allan

Blogs & Websites

- TBD

Community / Support

- Arduino Developer Mailing List
- Arduino Forum
- Arduino Playground (community resource)
- Arduino Language Reference
- ITP's Physical Computing site

Similarly Awesome and Useful Tools

Similar tools include:

- Make Controller Kit
- Rasberry Pi—a credit card sized single-board computer:
- BeagleBoard—another popular single-board computer
- Official Arduino Hardware
- AdaFruit
- SparkFun
- Boarduino

Cinder

Cinder is an open source C++ library for creative coding. Think of it as a non-Java alternative to Processing. It was originally developed by the Barbarian Group for their interactive advertising projects, which required something more robust than Processing for programming graphics, audio & video, image pro-

cessing, and computational geometry. It can be considered as a next-step beyond Processing, which is better for learning to code. Cinder shares many attributes with openFrameworks, and an aspiring user should investigate both tools, along with Processing.

Quickest Possible Description

Description
The non-Java alternative to Processing.

Site
http://libcinder.org

Git
https://github.com/cinder/Cinder

Wikipedia
http://en.wikipedia.org/wiki/Cinder_(programming_language)

Frequently Used For

Frequently used for:

- Interactive screen-based work
- Physical installations
- Increasingly used for iphone/ipad apps

Important Details

Important details include:

- Supports Mac OS X (with XCode), Windows, and IOS
- Released under Simplified BSD License

Heavyweights

The heavyweights in the field include:

- Andrew Bell was the originator and lead architect.
- Robert Hodgin wrote the book on it.

Projects & Examples

Interesting projects and examples include:

- Planetary—Visual exploration of your music collection
- Cindermedusae—Generative sea creatures

Resources / Community / Support

Useful resources include:

Tutorials

- Hello Cinder—tutorial by Robert Hodgin

Books

- Exploring Cinder, by Joshua Noble and Robert Hodgin

Community

- Cinder forums

Similarly Awesome and Useful Tools

Similar tools include CCGL and CCGL-Touch, Cinder wrappers for OSX and iOS

Kinect

Quickest Possible Description

Description
Video game console turned into magicmaking box.

Site
http://www.xbox.com/en-US/KINECT

wikipedia
http://en.wikipedia.org/wiki/Kinect

Frequently Used For

Major uses include:

* *

Important Details

Details include:

* * *

Heavyweights

Heavyweights include Greg Borenstein, @atduskgreg

Projects & Examples

Interesting projects and examples include:

* * *

Resources / Community / Support

Because the Arduino project is so well-organized and supported, the very best resource is the Arudino site itself. Some of the most useful sections are included below

Tutorials

* * *

Books

* * *

Blogs & Websites

* * *

Community / Support

* * *

Similarly Awesome and Useful Tools

Similar tools include:

* * *

MechanicalTurk

Sometimes people do things better than code. Don't tell anybody because you can keep that secret hidden with Mechanical Turk.

Quickest possible description

Description
Crowdsourcing as a service

Site
https://www.mturk.com/mturk/welcome

Git / Repo
https://github.com/

wikipedia
http://en.wikipedia.org/wiki/Amazon_Mechanical_Turk

Frequently Used For

Major uses include:

* *

Important Details

Details include:

* * *

Heavyweights

The heavyweights in the field include:

* * *

Projects & Examples

Interesting projects and examples include:

- Bicycle Built For Two Thousand by Aaron Koblin and Daniel Massey
- Emoji Dick, crowdsourced Emoji translation of Moby Dick

Resources / Community / Support

Useful resources include:

Tutorials
* * *

Books
* * *

Blogs & Websites
* * *

Community / Support
* * *

Similarly Awesome and Useful Tools

Similar tools include:

* * *

Max/MSP & Jitter

Max is a visual dataflow programming environment for music and multimedia whose GUI reflects the structure of the program you're writing, where objects in the code are displayed on the screen and are connected to each other via inlets and outlets, in a similar way to how a sound engineer might connect analog inputs to processors and outputs. Following the metaphor, Max operates in response to "bangs," which are initiating events, much like an electric current running through a synthesizer. MSP is a plug-in for Max that allows real-time manipulation of digital audio signals, and is designed to be fairly easy to start using and prototype with. When you think Max, you're probably really thinking Max/MSP.

Jitter is another popular Max package that allows for manipulation of real-time video, 3D graphics, animation, and more.

Quickest Possible Description

Site
http://cycling74.com

Git
TK

Wikipedia
http://en.wikipedia.org/wiki/Max_(software)

Frequently Used For

Frequent uses include:

- Live performance
- Interactive art installations, both large and small

Important Details

Important details include:

- Max is not free software, but there are student and teacher versions.
- Cross-platform and widely supported.

Heavyweights

The heavyweights in the field include:

- Luke Dubois, @RLukeDuBois—Coauthor of Jitter, Director of Experimental Media Center at NYU Poyltechnic
- Josh Goldberg (artist), @wugmump
- Kurt Ralske (artist, creator of Auvi) @kurtralske
- Jeremy Bailey (artist) @jeremybailey

Projects & Examples

Interesting projects and examples include:

- Vertical Music by R. Luke Dubois video::height=*300*, width=*500*, poster=*images/generic_video.png*
- Skube
- Skube
- Monome
- 13 Hours Of The Discovery Channel by Joshua Goldberg (2001)
- Word Sequencer by Justin Sheriff
- Word Sequencer video::height=*300*, width=*500*, poster=*images/generic_video.png*
- Giant Theremin Project by Robin Fox video::height=*300*, width=*500*, poster=*images/generic_video.png*

Resources / Community / Support

Useful resources include:

Tutorials

- The best place to start is the huge amount of resources available at the Max/MSP site.
- Jitter movie playback optimizations.

Books

- Electronic Music & Sound Design by Alessandro Cipriani & Maurizio Giri
- Max/MSP/Jitter for Music by V.J. Manzo

Blogs & Websites

- s373.net

Community / Support

- Again, the best forums and community are the Max/MSP site.

Similarly Awesome and Useful Tools

Similar tools include:

- Pure Data: An open source alternative to Max, originally developed by the same developer, Miller Puckette. Labeled as a *real-time music and multimedia environment* it allows manipulation of video, graphics, and music, and enables live collaboration across networks between musicians. It's multiplatform and portable so it can run on all sorts of devices and can be extended.
- Audio Cubes are smart physical objects for live performance and installations.
- Auvi is a collection of Max/Jitter objects for real-time video processing.
- cvjit is a collection of Max/MSP/Jitter objects for computer vision applications.
- ppooll is a networking system between max patches. Max Objects is a database for Max objects, patches, etc. It looks old but it's still being updated.
- Max For Live is a toolkit for building new devices for Ableton Live.
- vOO2 is an ongoing project to develop new effects and functionality to augment realtime video applications using Quartz Composer based technology. One such project is syphon, an open source OSX technology for sharing frames between applications in real time.

OpenCV

OpenCV is an open source computer vision and machine learning software library. It's widely used and supported by people working on real-time computer vision such as face detection, camera tracking, augmented reality, and anything else cool that you see done with cameras. The library has thousands of algorithms that can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track movement, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together, find similar images from a database, recognize scenery and establish markers for augmented reality applications, and much, much more.

OpenCV is actively managed by itseez and Willow Garage.

Quickest Possible Description

Site
http://opencv.org

Git
git://code.opencv.org/opencv.git

Wikipedia
http://en.wikipedia.org/wiki/OpenCV

Twitter
http://twitter.com/#!/opencvlibrary

OpenCV Wiki
http://code.opencv.org/projects/opencv/wiki

Frequently Used For

Major uses include:

- Face and gesture recognition
- Human and computer interaction
- Motion tracking
- Robotics
- Stereopsis
- Machine learning (detection/recognition)
- Camera calibration
- Image processing

Important Details

Details include:

- Released under a BSD license (free to use)
- Has C++, C, Python & Java interfaces
- Supports Windows, Linux, Android, Mac OSX

Heavyweights

The heavyweights in the field include Gary Bradski and Adrian Kaehler, who wrote the book on OpenCV.

Projects & Examples

Interesting projects and examples include:

- CV Dazzle by Adam Harvey
- Robot operating system
- i3D – Head Tracking for iPad
- Aikon, home of Paul The Robot by Patrick Tresset
- T.I.M.
- Apparel by t normals
- normalfutu.re

Resources / Community / Support

Useful resources include:

Tutorials

- Introduction to embedded vision and the OpenCV library by Eric Gregori

Books

- Introduction to Programming with OpenCV by Gady Agam
- Learning OpenCV, 2nd Edition by Gary Bradski, Adrian Kaehler
- OpenCV 2: Computer Vision Application Programming Cookbook by Robert Laganière

Blogs & Websites

- OpenCV Wiki
- OpenCV Documentation
- OpenCV Questions

- Yahoo OpenCV Group

Similarly Awesome and Useful Tools

Similar tools include:

- SimpleCV website & tumblr
- MatLab

openFrameworks

openFrameworks is an open source C\+\+ toolkit designed to make creative coding easier. Its creators describe it as a "general purpose glue" that wraps together common libraries such as OpenGL, OpenCV, FreeType and many more. It is designed to be cross-platform and extensible, and to provide a promote experimentation with both software interfaces that allow for interesting real-time screen-based and physical interactions. openFrameworks shares many similarities with Processing and especially Cinder, which is also a C\+\+ library framework. It makes sense to investigate all three when deciding on a creative coding tool.

Quickest Possible Description

Site
http://www.openframeworks.cc/

Git
https://github.com/openframeworks/openFrameworks

Wikipedia
http://en.wikipedia.org/wiki/OpenFrameworks

Twitter
https://twitter.com/openframeworks

Other wiki/resource
http://wiki.openframeworks.cc/

Frequently Used For

Frequent uses include:

- Physical installations
- Big screens

- Physical and screen-based interaction projects

Important Details

Details include:

- Open Source, distributed under the MIT License
- Platforms: Windows, OSX, Linux, iOS, Android
- IDEs: XCode, Code::Blocks, Visual Studio, Eclipse

Heavyweights

The heavyweights in the field include:

- Zach Lieberman, @zachlieberman
- Arturo Castro, @arturOcastro
- Kyle McDonald, @kcimc
- Theo Watson, @theowatson
- Chris O'Shea, @chrisoshea
- Evan Roth, @fi5e
- Golan Levin, @golan

Projects & Examples

Made with openFrameworks:

- Eyewriter
- Drawn
- Faces by Arturo Castro and Kyle McDonald
- Body Swap
- Blind Self Portrait by Kyle McDonald
- Night Bright by Design I/O
- Hana by Andreas Muller
- Hand tracking experiment by Ben McChesney
- FaceShift (face tracking)

- Graffiti Analysis by Evan Roth

Resources / Community / Support

Useful resources include:

Tutorials

- openframeworks.cc/tutorials/ has excellent tutorials and is a great place to start
- C and C++ in 5 days by Philip Machanick (PDF)
- OF for Processing Users by Zach Gage, for those transitioning from Processing
- Using openFrameworks with Arduino
- OF Frameworks Tutorials by Ben McChesney

Books

- Programming Interactivity by Joshua Noble

Blogs & Websites

* * *

Community / Support

- Mailing List
- Forum

Similarly Awesome and Useful Tools

- ofxaddons A directory of extensions and libraries for OpenFrameworks
- ofxKinect, an OF add-on for the Xbox Kinect

PureData (PD)

Pure Data: An open source alternative to Max, originally developed by the same developer, Miller Puckette. Labeled as a *real-time music and multimedia environment* it allows manipulation of video, graphics, and music, and enables live collaboration across networks between musicians. It's multi-platform and portable so it can run on all sorts of devices and can be extended, just like Max.

Quickest Possible Description. *** Site:: Project homepage Wikipedia:: Max, Pure Data Other wiki/resource:: Docs from project homepage Code:: *http://www-crca.ucsd.edu/~msp/software.html* Email list:: *http://lists.puredata.info/pipermail/pd-list/* ***

Frequently Used For

Major uses include:

- Realtime media manipulation
- Interactive art with sound and video
- Mobile apps

Important Details

Details include:

- PD is free and licensed under BSD.
- Most associated software is licensed under GNU GPL.

Heavyweights

The heavyweights in the field include Miller Puckette, originator of PD.

Projects & Examples

Interesting projects and examples include Jamie Oliver's Silent Drum.

Resources / Community / Support

Tutorials

- A huge collection of workshops and tutorials is available on the PD site.
- Practical Synthetic Sound Design (tutorial by Andy Farnell).
- Prototyping with Pure Data and Processing.
- The Theory and Technique of Electronic Music by Miller Puckette.
- video lectures by Dr.Hdez.

Books

- Loadbang: Programming Electronic Music in PD by Johannes Kreidler
- Making Musical Apps – Real-Time Audio Synthesis on Android and iOS by Peter Brinkmann (2012)
- Composition: Pure Data as a Meta-Compositional Instrument by Michael Barkl (2012)
- Pd Recipe Book—Pure Data by Sei Matsumura
- Designing Sound by Andy Farnell (2010)

Blogs & Websites
The Pure Data manual

Community / Support
IRC

Similarly Awesome and Useful Tools

Libpd is "PD made embeddable." It strips away the GUI and the concerns with inputs and outputs, to give a lightweight version of PD which is better suited to mobile development. You can find libpd on GitHub.

R: The R Project for Statistical Computing

R is a language and environment for statistical computing and graphics. It's an open source version / different implementation of S. Originally written by Robert Gentleman and Ross Ihaka of the Univeristy of Auckland, it is now collaborative effort with its own foundation and core contributors. It's widely used for data analytics, data mining, machine learning, data visualization. It's what you need for Big Data projects, basically. R is particularly useful because it's so extensible. Thousands of packages have been written to help R handle certain kinds of data from climate information to speech. This "Task View"page offers an overview of some of the ways R has been applied.

Quickest possible description

Site
http://www.r-project.org/

Git or Code
http://cran.r-project.org/mirrors.html

Wikipedia
http://en.wikipedia.org/wiki/R_%28programming_language%29

Other wiki/resource
R Wiki

Frequently Used For

Major uses include:

- Data visualization
- Analytics
- Machine Learning

- Basic and Advanced Statistics (modeling, distributions, tests, etc.)
- Handling GeoSpatial data, medical research, signal processing, etc.
- Data mining

Important Details

Details include:

- GNU General Public License
- Runs on UNIX including Windows, MacOS, FreeBSD, Linux

Heavyweights

The heavyweights in the field include:

- Dianne Cook
- Bill Venables
- John Chambers (creator of S, core committer of R)
- Jake Porway
- Amanda Cox

Projects & Examples

Interesting projects and examples include:

- 3-D Animation of the changing antarctic ice sheet
- Video Map of the hottest songs in the US
- A bunch of great projects from periscopic

Resources / Community / Support

Useful resources include:

Tutorials

- Introduction to R for Data Mining (Webinar)
- Why and How people Use R by John Cook (presentation)
- *http://jeffreybreen.wordpress.com/2012/10/02/tapping-the-data-deluge-with-r/*
- Tapping the Data Deluge with R – by Jeff Breen (lightning talk)

- Tutorial: R for programmers coming from different languages by John Cook
- Some Hints For The R Beginner by Patrick Burns
- The R Inferno by Patrick Burns (PDF)
- Google's R Style Guide
- A bunch of great tutorials from Flowing Data
- an introduction to R

Books

- datajournalismhandbook.org
- R In A Nutshell
- Visualize This
- R in Action – Data Analysis and Graphics with R by Robert Kabacoff
- Books Related To R (a list, not a book)

Blogs & Websites

- Finding Data on the Internet (This blog post lists data sources that are easily imported into R or already in R format)
- www.statmethods.net A great blog with lots of information from the author of R In Action.
- David Smith's R blog
- Citizen Statistician (a blog about stats that talks a lot about R)
- *http://journal.r-project.org/current.html[R* Journal: Articles & News about R)
- flowingdata.com

Community / Support

- Cran R project FAQ
- inside-R
- R-Help Mailing List
- project documentation

Similarly Awesome and Useful Tools

Similar tools include:

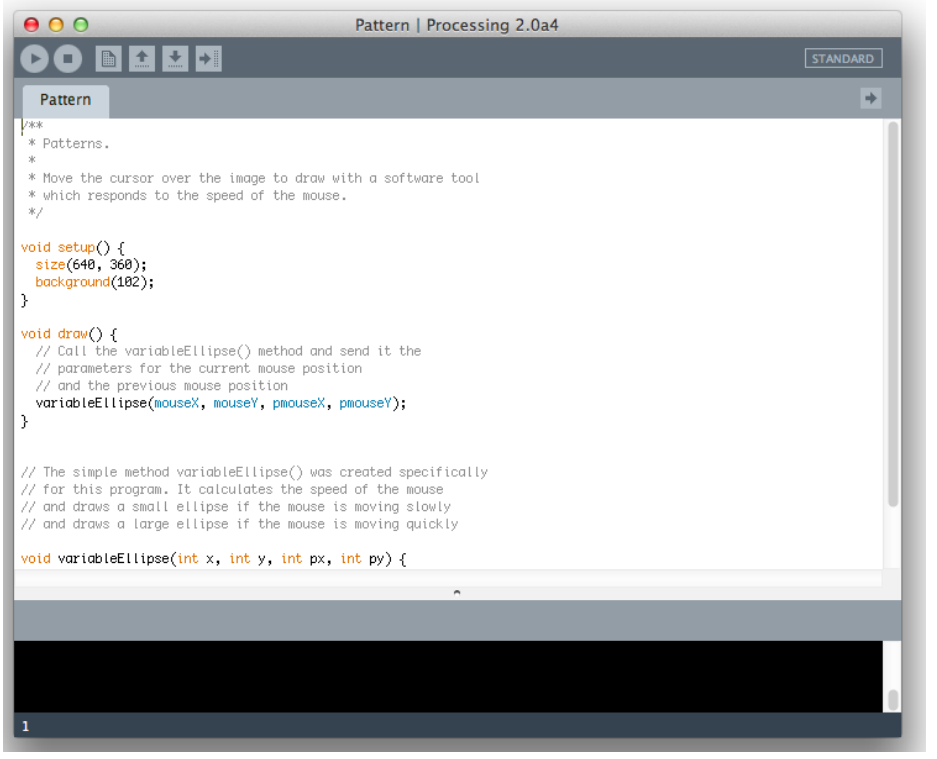
- Rook—R package for building and running R web applications
- RStudio—IDE for R
- elasticR—Package for using EC2 with R

- R Commander—Basic-Statics GUI for R
- RevolutionR—Enterprise software for R (with academic and free version
- Bioconductor open source development project to provide packages and tools for analysis and comprehension of high-throughput genomic data
- CRAN Task Views Page
- Crantastic community site for R Packages
- R 4 IntelliJ:—Integration of R into the intellij DEA

Processing

Processing is an open source programming language and environment for people who want to create images, animations, and interactions. Initially developed to serve as a software sketchbook and to teach fundamentals of computer programming within a visual context, Processing also has evolved into a tool for generating finished professional work. Today, there are tens of thousands of students, artists, designers, researchers, and hobbyists who use Processing for learning, prototyping, and production.

Processing *sketches*, the Processing community's term for interactive programs, are created in a custom integrated devlopment environment (IDE) that makes it very easy to get started writing code. Processing sketches end in the extension ".pde".

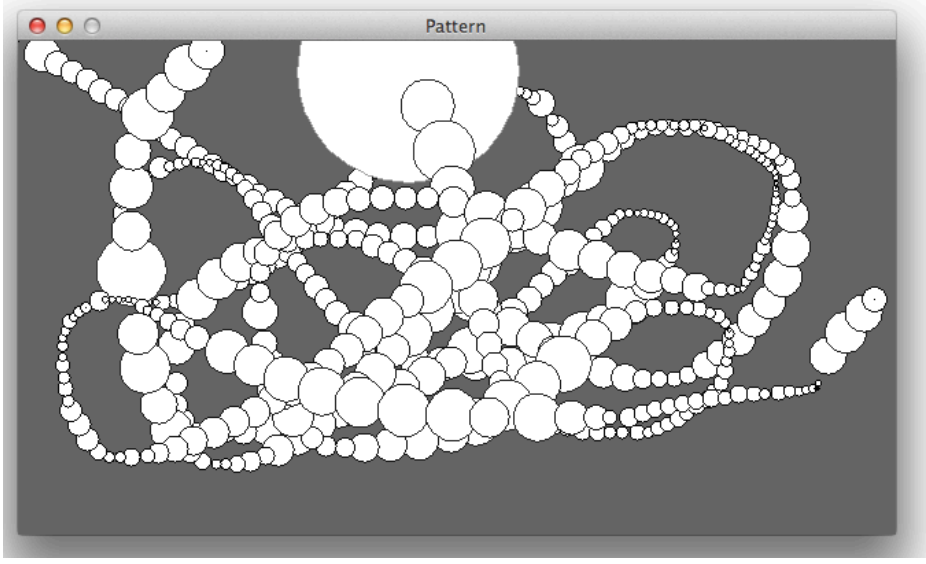


The language has a Java-like syntax (in fact, it basically is Java with a few new commands) that will be immediately familiar to most coders. Here, for example, is a Processing sketch called "Patternd.pde" that is included in the "Examples"that are shipped with the environment. It makes interesting patterns as you move the mouse across the canvas area:

```
/**
 * Patterns.
 *
 * Move the cursor over the image to draw with a software tool
 * which responds to the speed of the mouse.
 */

void setup() {
  size(500, 360);
  background(102);
}

void draw() {
  // Call the variableEllipse() method and send it the
  // parameters for the current mouse position
  // and the previous mouse position
  variableEllipse(mouseX, mouseY, pmouseX, pmouseY);
}
```

```
// The simple method variableEllipse() was created specifically
// for this program. It calculates the speed of the mouse
// and draws a small ellipse if the mouse is moving slowly
// and draws a large ellipse if the mouse is moving quickly

void variableEllipse(int x, int y, int px, int py) {
  float speed = abs(x-px) + abs(y-py);
  stroke(speed);
  ellipse(x, y, speed, speed);
}
```

Here's a sample of the output:



Most of the action revolves around painting images on the *canvas*, which is the main output display area. Each sketch contains a method called *draw()* that executes as a continuous loop on the main output of the program. The sketches typically implement basic interactions, such as detecting mouse movements or button clicks. There are additional libraries for more complex interactions.

Quickest Possible Description

URL

http://processing.org/

Key areas

,,

Frequently Used For

Major uses include:

* *

Important Details

Details include:

* * *

Heavyweights

Processing has a large and vital community. Some of the heavyweights include:

Ben Fry

Ben Fry is a cofounder of the Processsing project, and is the principal of Fathom, a design and software consultancy located in Boston. He received his doctoral degree from the Aesthetics + Computation Group at the MIT Media Laboratory, where his research focused on combining fields such as computer science, statistics, graphic design, and data visualization as a means for understanding information. After completing his thesis, he spent time developing tools for visualization of genetic data as a postdoc with Eric Lander at the Eli & Edythe L. Broad Insitute of MIT & Harvard. During the 2006–2007 school year, Ben was the Nierenberg Chair of Design for the Carnegie Mellon School of Design. At the end of 2007, he finished writing Visualizing Data for O'Reilly. In 2011, he won the National Design Award for Interaction Design from the Cooper-Hewitt.

Casey Reas

Casey Reas, a cofounder of the Processsing project, explores the relationship between naturally evolved systems and those that are synthetic. The imagery evokes transformation, and visualizes systems in motion and at rest. Equally embracing the qualitative human perception and the quantitative rules that define digital culture, organic form emerges from precise mechanical structures. Reas' software, prints, and installations have been featured in numerous solo and group exhibitions at museums and galleries in the United States, Europe, and Asia.

Dan Shiffman

Daniel Shiffman works as an Assistant Arts Professor at the Interactive Telecommunications Program at NYU's Tisch School of the Arts. Originally from Baltimore, Daniel received a BA in Mathematics and Philosophy from Yale University and a Master's Degree from the Interactive Telecommunications Program. He works on developing tutorials, examples, and libraries for Processing, the open source programming language and environment created

by Casey Reas and Ben Fry. He is the author of Learning Processing: A Beginner's Guide to Programming Images, Animation, and Interaction and The Nature of Code (self-published via Kickstarter), an upcoming text and series of code examples about simulating natural phenomenon in Processing.

Projects & Examples

Interesting projects and examples include:

* * *

Resources / Community / Support

Useful resources include:

Tutorials

*

Books

- Processing: A Programming Handbook for Visual Designers and Artists. A giant, beautifully designed and richly detailed exploration of Processing by languages cofounders.
- Learning Processing. A refreshingly brief introduction to Processing by the project's founders, Ben Fry and Casey Reas. This is a great book for total beginners to Processing and programing in general.

Blogs & Websites

- Learning Processing. This site is home for ITP Professor Daniel Shiffman's book Learning Processing. Although this book doesn't cover Arduino, it covers almost everything you would want to do in Processing, and is a great book to tackle once you've learned the language. The section on video has many creative mirrors, like "Video Pixelation."
- The Nature of Code. Another great site from Daniel Shiffman, this site covers "topics ranging from basic mathematics and physics concepts to more advanced simulations of complex systems. Subjects covered will include forces, trigonometry, fractals, cellular automata, self-organization, and genetic algorithms." In addition to all the great Processing code, you'll also learn the physics behind the code. For example, the material on vectors and path following will be helpful to anyone doing robotics.

Community / Support

- Processing Forums. An old school community forum for asking questions and sharing techniques. This is a great place for beginners and seasoned hands alike to discuss how to use the language.

- OpenProcessing. OpenProcessing is a community-driven site where people can share their portfolios. You'll find a vast array of creative sketches that range from art to science. For example, GUIGUITRO-CHOID by Guigui simulates the gears in a Spirograph. In addition to enjoying the visual beauty, you can also view the source code from any sketch to see how it works. 

Similarly Awesome and Useful Tools

Similar tools include:

* * *

xbee

xBees make things talk, wirelessly. Free your projects.

Quickest Possible Description

Description
: xBees make things talk, wirelessly. Free your projects.

Site
: *http://www.digi.com/xbee*

wikipedia
: *http://en.wikipedia.org/wiki/XBee*

Frequently Used For

Major uses include:

* *

Important Details

Details include:

* * *

Heavyweights

The heavyweights in the field include:

- Rob Faludi — @faludi * *

Projects & Examples

Interesting projects and examples include:

* * *

Resources / Community / Support

Useful resources include:

Tutorials

* * *

Books

* * *

Blogs & Websites

* * *

Community / Support

* * *

Similarly Awesome and Useful Tools

Similar tools include:

* * *

CHAPTER 4

Projects

The goal of the projects section is to articulate the vision and thoughts that go into creating an interesting work or solving an interesting problem, with a particular focus on the project's relevance compared to other computational / serial artists or projects.

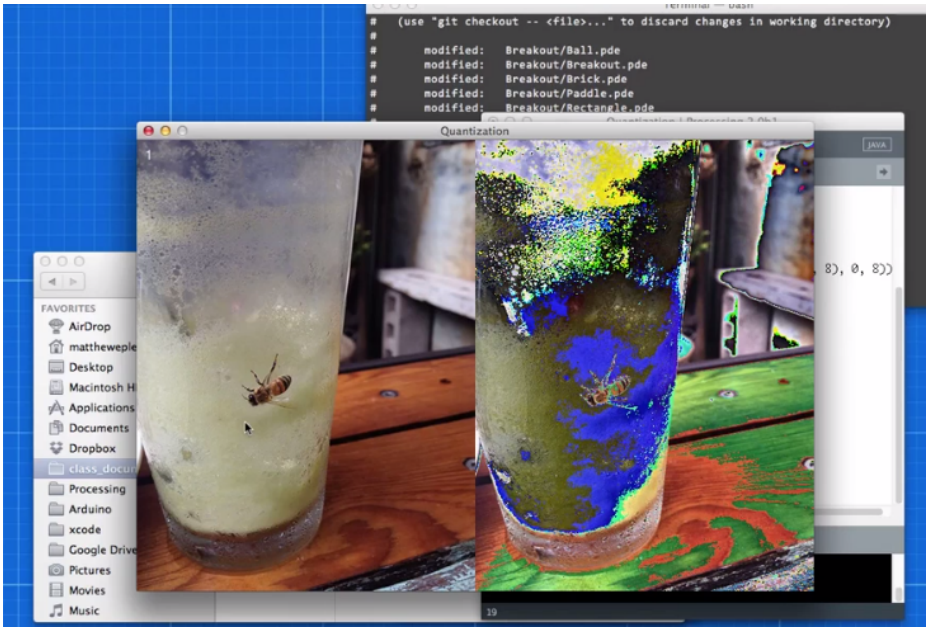
We're much more interested in the "whys" of the project — why did you pick certain tools, why did you take the approach, why is this different than what's come before — before the "hows". (But, don't worry — you can put the hows in the tutorial or example sections.) The project descriptions should serve as a guide of your thinking for people who will follow after you.

- What was the project?
- What areas or field does it fall in?
- What tools did you use?
- Why was this an interesting project to you — what was new or innovative?
- What were the key challenges you had to overcome?
- Your name and a brief bio

Bit-shifting with Kyle McDonald

What areas or field does it fall in

code, math What tools did you use? Processing



Description

Ever wondered what bit-shifting is, exactly? How can you use it in your projects? Well you're not alone. I recently sat down with creative wizard samurai Kyle McDonald to get some answers. Check out this screen recording of our conversation and download the code for yourself. NOTE: sorry for the low volume. I didn't think this would be entering the web-o-spheres at the time. But it's just too good to keep to myself!

- Bitwise (Bit-shift visualizer)
- Quantization (using bit-shifting to manipulate images)

Bios

Kyle McDonald is a media artist who works with code, with a background in philosophy and computer science.

Matthew Epler is a graduate student at NYU's Interactive Telecommunications Program (ITP) specializing in computational art and archives.

URL

- *http://kylemcdonald.net/*
- *http://mepler.com*

Contact Information

- *kyle@kylemcdonald.net*
- *matthewepler@gmail.com*

Tags

processing

Inferential 3D Scanner with Goldfish

What areas or field does it fall in

processing, 3d scanning What tools did you use? Processing PlayStation Eye Camera

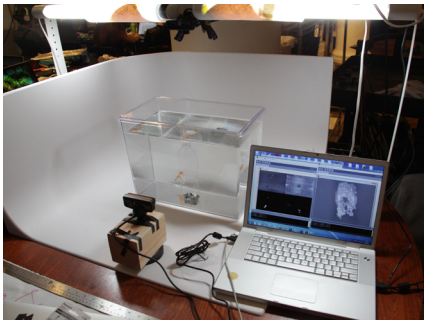
Description

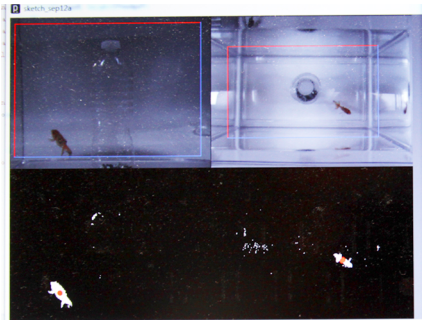
This project was done in Kyle McDonald's 3D Sensing and Visualization class at NYU's Interactive Master's Program (ITP) in the Fall of 2012 by Deqing Sun and Tak Cheung.

For the first class assignment, we were asked to build a 3D sensor from scratch. The concept came from brainstorming what devices are available to detect space. Off-the-shelf sensors like a laser range finders can gather distance and implementing this within a matrix we can create a set of 3D data. We thought sensory inputs are also inherent in every living organism too. There's a lot we can do with that concept like putting a GPS collar on dogs. Using their travel paths to scan out pedestrian sidewalks over time. We chose goldfish as they're conveniently swimming in a 3 dimensional medium and for their bright color it'll be easy for our program/camera to pick them out.

We devised 2 cameras, front view and top view to track the goldfish. The video is passed through Processing that calculates red values in the scene which separates the fish from its white environment.

A central point (red dot) is calculated from both views tracking the X,Y,Z coordinates of the fish. An object is placed inside the tank to be 'scanned.' Clear bottle was our choice so to not obstruct the camera's view as the fish swims behind it.





Bios

Deqing Sun and Tak Cheung are students at NYU's Interactive Telecommunications Program.

URL

- *http://gettak.com/*
- *http://www.thinkcreate.org/*

Contact Information

- *Tak@gettak.com*
- *iamseer@gmail.com*

Tags

processing 3d scanning diy hack

Cover for the Liber Amicorum

What areas or field does it fall in

generative art, design What tools did you use? Processing

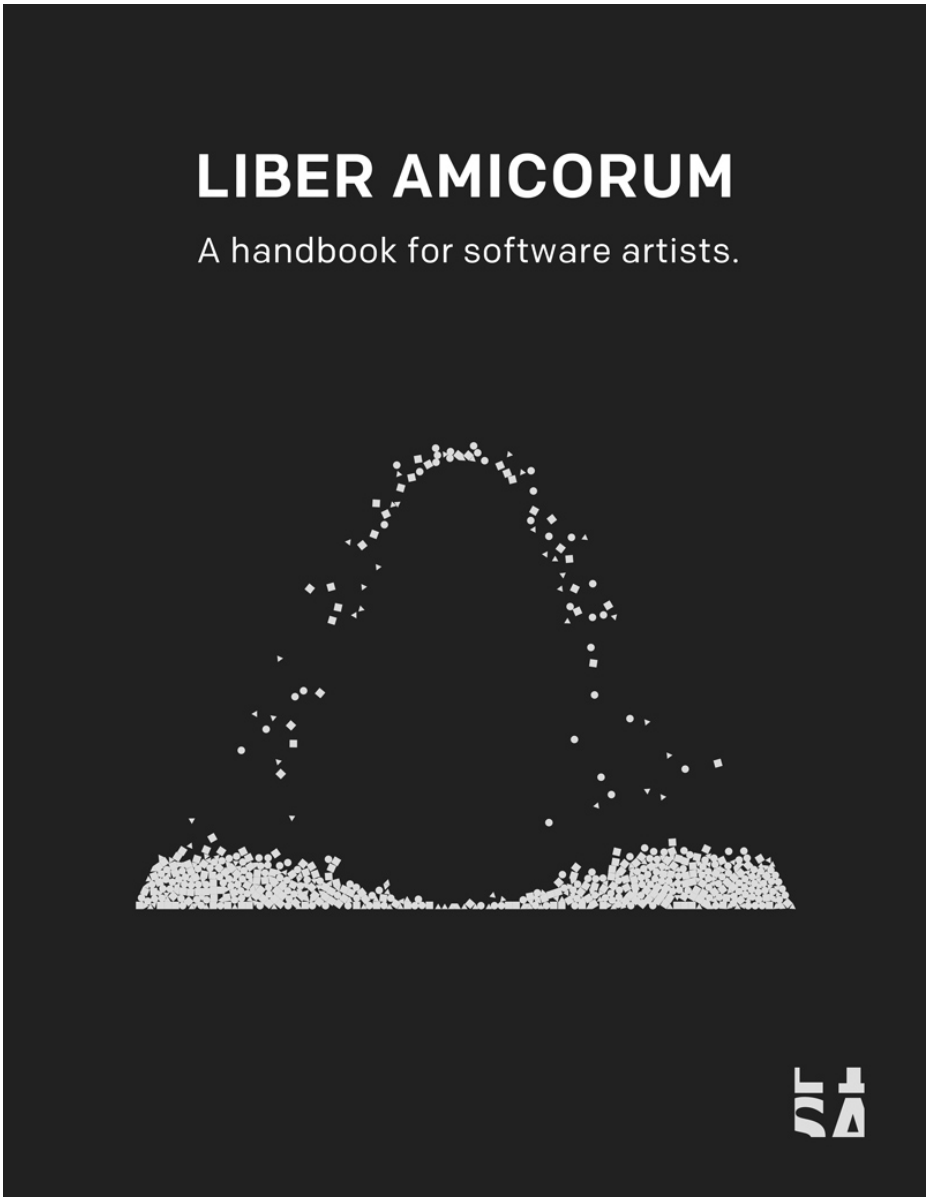
Description

I've was playing around with the thought of making a generative book cover for the LISA book. I tend to have a very minimalistic style in all of my print design. Most books about creative coding tend to have extremely algorithmic book covers, and I thought it would be fun to do something very different.

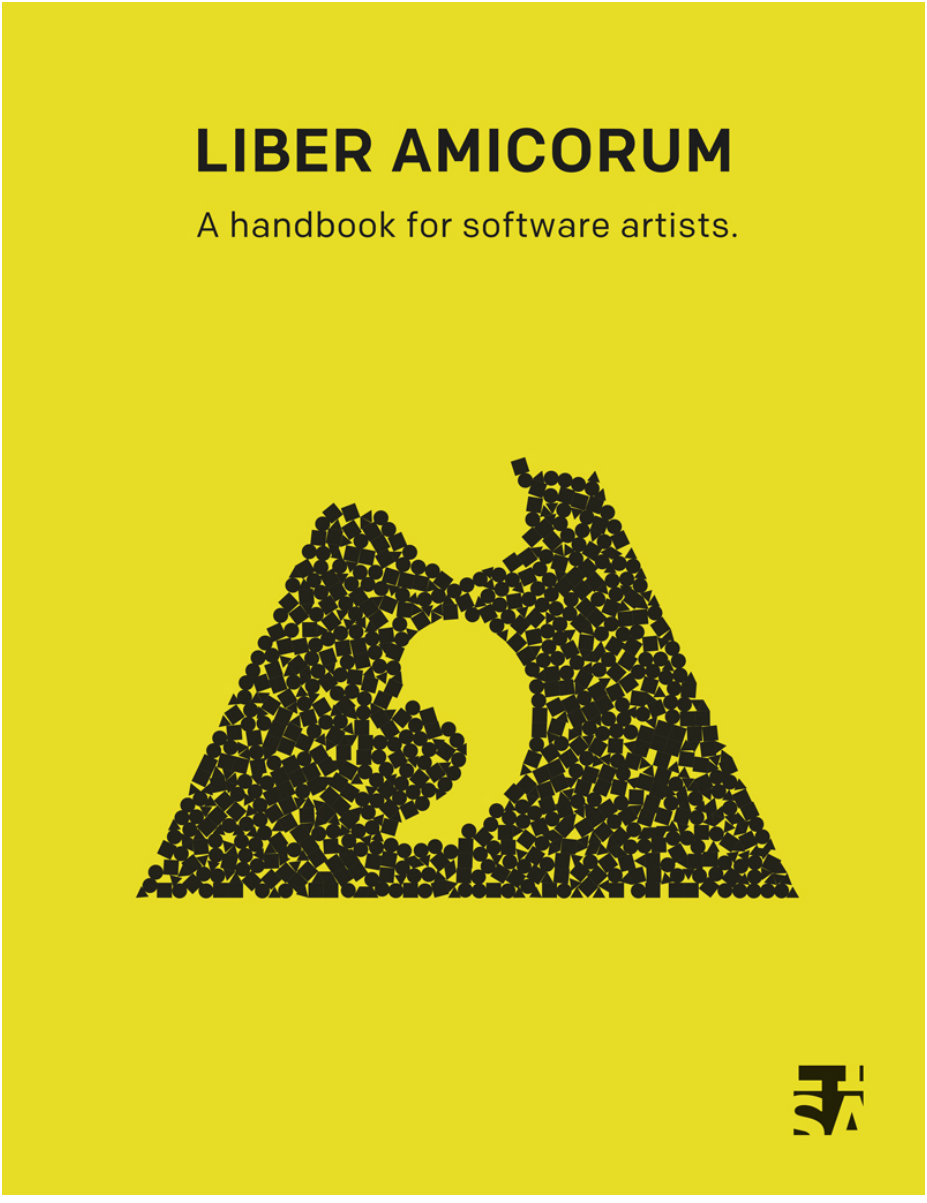
So, the cover is created by a Processing sketch that called generative-book-cover. When it runs, it chooses a random background color, and starts a physics simulation that drops random sized shapes into a big triangle container. Inside of this triangle container is another container in the shape of a semi-colon, and all the small, dropping shapes will bounce of these 2 containers. At any given point in time, the Processing sketch will save whatever is in the window and quit.

Here are a couple of samples:

Cover for the
Liber Amicorum



Here's the second iteration:



I ran the sketch 4 times, and the initial cover was the 4th image that came out of it. We'll use it to generate a completely new cover for each major edition of teh book. It's a good story with some good symbolism (the semi-colon made up by a lot of smaller bits), and it fits well with the kind of stuff that I like (*http://bit.ly/PstKmV*).

Bio

Rune Madsen is a designer and programmer from Copenhagen, Denmark. He moved to New York City 3 years ago to study at the Interactive Telecommunications Program (ITP) at NYU, and spent a year afterwards creating data visualizations for the New York Times.

Rune also makes art with computers, which means everything from print design (*http://runemadsen.com/work/tiny-artists*) and gallery installations (*http://runemadsen.com/work/write-me*) to video sculptures (*http://runemadsen.com/work/tile-puzzle*) and interactive visuals for performers (*http://runemadsen.com/work/ohland-balloon-visuals*).

Alongside his O'Reilly job as a software developer, Rune is teaching a graduate class at ITP on algorithmic graphic design.

URL

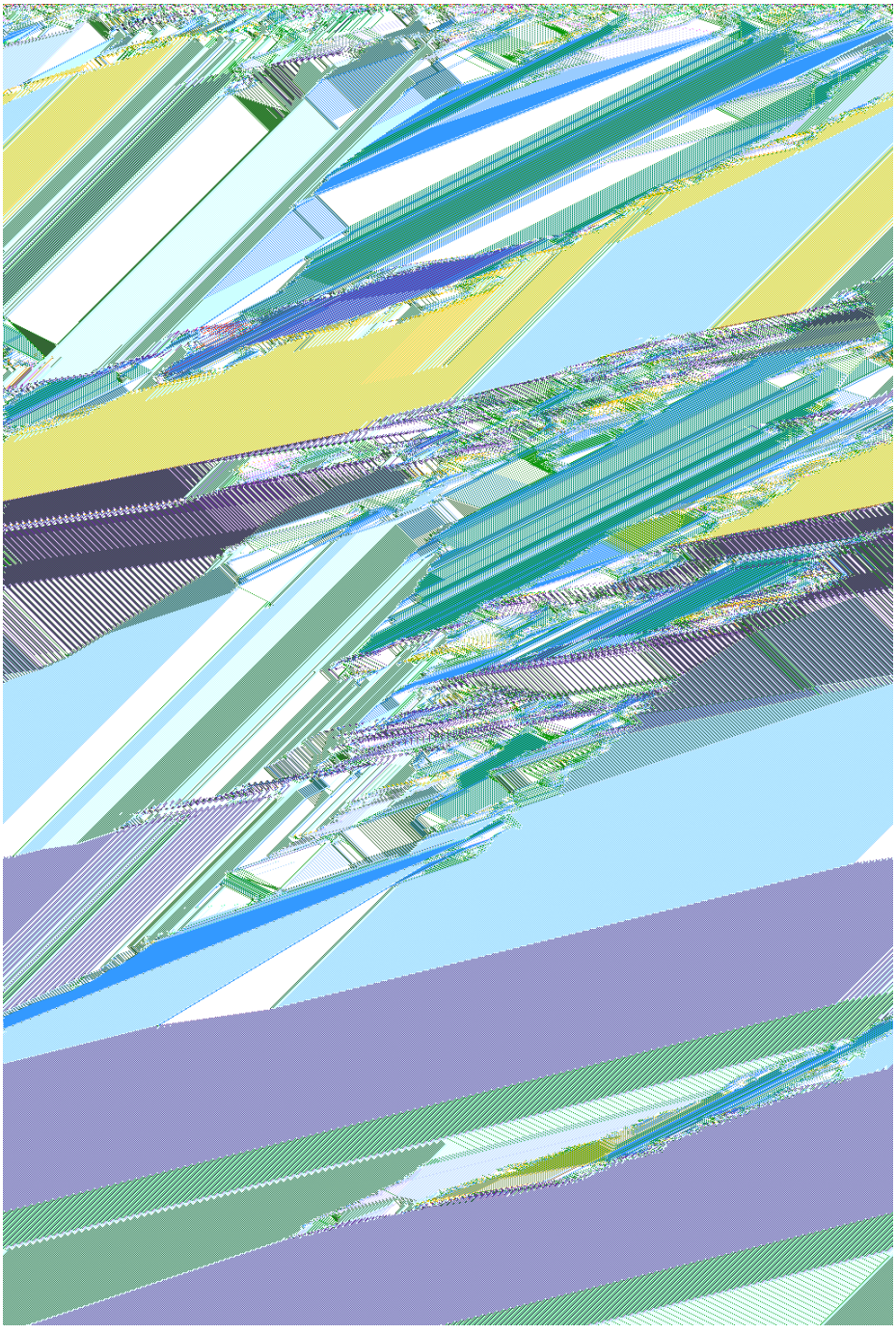
http://runemadsen.com/

Creative Archeology: finding inspiration in the archives

What areas or field does it fall in

History, Genetics Algorithms What tools did you use? Processing

Description



In the summer of 2011, Alex Galloway published a fantastic article on the work of Nils Aall Barricelli in Cabinet Magazine. It has remained somewhat underground among creative coders and I believe it to be not only a fascinating read, but an example of a creative-historical process that could change the way we approach our work as digital artists.

Nils Aal Barricelli was a mathematician who refused to play by the rules and who, in 1953, created numeric organisms of his own design. The results are visually stunning and his story fascinating. But I'll let you read the article for more on that.

More important, I believe, is recognizing the work that has come before us and resurrecting it in an accessible langugage. There is a peculiar habit among creative code practicioners of creating in a historical vaccuum. We are often only aware of the latest developments in computational art as covered by a small handful of blogs. There are, however, countless examples of scientists, researchers, artists and meddlers using computers to do extraordinary tasks going back as far as there have been computational machines. Their work is occaisonally dug up published in an exhibit catalogue or history book. But this doesn't serve our community, who could use their code to build upon.

Galloway offers a rich alternative practice of digital archeology/translation that illuminates the work of a maverick Whose work resonates still today. Go to the library. You never know what you might find.

Code for this project is available on Galloway's site.

Other LInks

- Computer Graphics & Art 1976–1978 – digitized PDFs of a short-lived magazine
- A Little-Known Story about a Movement, a Magazine, and the Computer's Arrival in Art: New Tendencies and Bit International, 1961–1973 – book published by MIT Press

Bios

Alexander R. Galloway is a writer and computer programer working on issues in philosophy, technology, and theories of mediation at NYU. He is the author of a new book, "The Interface Effect"

Matthew Epler is a graduate student at NYU's Interactive Telecommunications Program (ITP) specializing in computational art and archives.

URL

- *http://cultureandcommunication.org/galloway/*

- *http://mepler.com/*

Contact Information

- *galloway@nyu.edu*
- *matthewepler@gmail.com*

Tags

processing

Generative Typography Platform with Processing - Project in Progress

What areas or field does it fall in

Typography, Generative, Design What tools did you use? Processing, Geomerative Library

Description

This project is a basic prototype for a generative typography tool made with Processing using the Geomerative Library by Ricard Marxer. It came out of Rune Madsen's class at ITP in generative design. In his lectures, Madsen stresses the important of limitations alongside flexibility. Building software systems that can provide flexibility while allowing a designer to work within their self-definied limits is difficult but ultimately extremely useful. Additionally, in the world of

generative design there area a lot of generative fonts popping up, but no way to customize them on with your own tools. This interface offers an alternative system that is open, fully customizeable, and flexible. A lot of work still needs to be done to make this a full-fledged tool, so have at it!

I highly recommend checking out the documentation for the Geomerative library to get a better idea of what it's capable of doing inside this platform.

Bio

Matthew Epler is a graduate student at NYU's Interactive Telecommunications Program (ITP) specializing in computational art and archives.

URL
http://mepler.com/

Contact Information
matthewepler@gmail.com

Tags
processing typography generative student

Index

We'd like to hear your suggestions for improving our indexes. Send email to index@oreilly.com.

W