

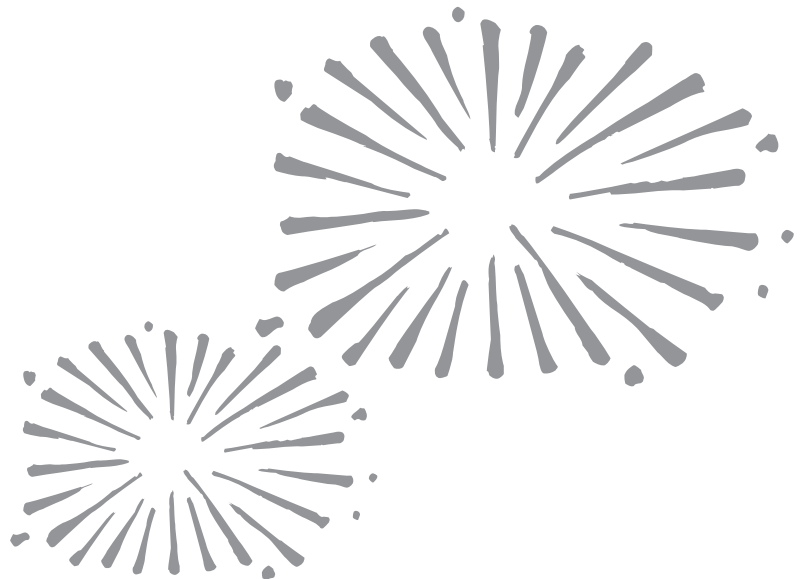


ART AND CODE

programming environments for artists,
young people & the rest of us

7-9 march, 2009

carnegie mellon
pittsburgh, pa



sponsored by

Microsoft®
Research

 Center for
Computational Thinking
Carnegie Mellon

STUDIO
FOR CREATIVE INQUIRY

Saturday, March 7

Half-Day Workshops

The main meeting place for ART AND CODE is the Lounge outside the Giant Eagle Auditorium, located on the first floor of Baker Hall (Baker-GE). Directional signs will be posted at the building's main entrance.

9-10am	LOUNGE	REGISTRATION & BREAKFAST <i>Coffee and tea will be available in the lounge all day.</i>
10am-1pm	BAKER-140F CFA-318 WEAN-5201 WEAN-5202 CFA-323 CFA-303 CFA-310 BAKER-140E BAKER-140C	Patient Intro to Processing, Part 1 (Reas) Intro to Flash Programming with Actionscript (Greenberg) Brisk Intro to Processing (Shiffman) Information Visualization with Processing (Fry) Audio & MIDI with Max/MSP (DuBois) openFrameworks (Lieberman, Watson & Castro) Hijacking Photoshop & Illustrator with Extendscript (Woohoo) Teaching with Scratch (Eastmond) Drawing Cats with Hackety Hack (Why)
1-2pm	LOUNGE	LUNCH & BOOK SALE <i>Authors will sign books at 1:30</i>
2-5pm	BAKER-140F CFA-318 WEAN-5201 WEAN-5202 CFA-323 CFA-303 CFA-317 CFA-310 BAKER-140C BAKER-140E	Patient Intro to Processing, Part 2 (Reas) Teaching with Processing (Greenberg) Processing: Advanced Applications (Shiffman) Information Visualization with Processing (Fry) Video & Graphics with Max/MSP/Jitter (DuBois) openFrameworks (Lieberman, Watson & Castro) Intro to VVVV (Oschatz) Intro to Silverlight (Brown) Scratch for Young People (Maloney & Eastmond) Play Games with Hackety Hack (_Why)
5:30-7:30pm	LOUNGE	OPENING RECEPTION <i>Shuttle buses will depart Frew St at 7:10 and 7:35 to take you to the Oskar Fischinger film festival.</i>
8-9:10pm	Pittsburgh Filmmakers	OSKAR FISCHINGER FILM FESTIVAL <i>Free admission with your conference badge. Shuttle buses will be available at 9:15 and 9:50 to return you to campus and hotels.</i>

CODE and FORM EXHIBIT

Free admission with your conference badge
Pittsburgh Center for the Arts, Saturday, 10am-5pm



DAYLIGHT SAVINGS REMINDER:

Daylight Savings Time starts *tomorrow* (Sunday, March 8.)
Set your clocks ahead 1 hour before you go to bed tonight.

Sunday, March 8

Presentations

8-8:45am	LOUNGE	REGISTRATION & BREAKFAST <i>Coffee and tea will be available in the lounge all day.</i>
8:45-9am	BAKER-GE	Opening Remarks (Levin)
9-9:45am	BAKER-GE	Alice (Slater)
9:45-10:30am	BAKER-GE	Silverlight (Brown)
10:30-11:15am	BAKER-GE	Scratch (Maloney & Eastmond)
11:15-12:00pm	BAKER-GE	Hackety Hack (_Why)
noon-1pm	LOUNGE	LUNCH & BOOK SALE
1-1:45pm	BAKER-GE	Processing (Fry & Reas)
1:45-2:30pm	BAKER-GE	Max/MSP/Jitter (DuBois)
2:30-3:15pm	BAKER-GE	VVVV (Oschatz)
3:15-3:30pm	LOUNGE	COOKIE BREAK
3:30-4:15pm	BAKER-GE	AIR & Extendscript (Woohoo)
4:15-5pm	BAKER-GE	openFrameworks (Lieberman, Watson & Castro)
5-5:45pm	BAKER-GE	Keynote Lecture (McMail)
7:10pm	FREW ST	<i>Shuttle buses will depart Frew St at 7:10 and 7:35 to take you to brillobox and the Oskar Fischinger film festival</i>
7:30pm-1am	brillobox	Open-mic Dorkbot <i>Shuttle buses will be available at 10, 11, 12 and 1 to return you to campus and hotels</i>
8-9:10pm	Pittsburgh Filmmakers	OSKAR FISCHINGER FILM FESTIVAL <i>Free admission with your conference badge. Shuttle buses will be available at 9:15 and 9:50 to take you to brillobox, campus and hotels.</i>

CODE and FORM EXHIBIT

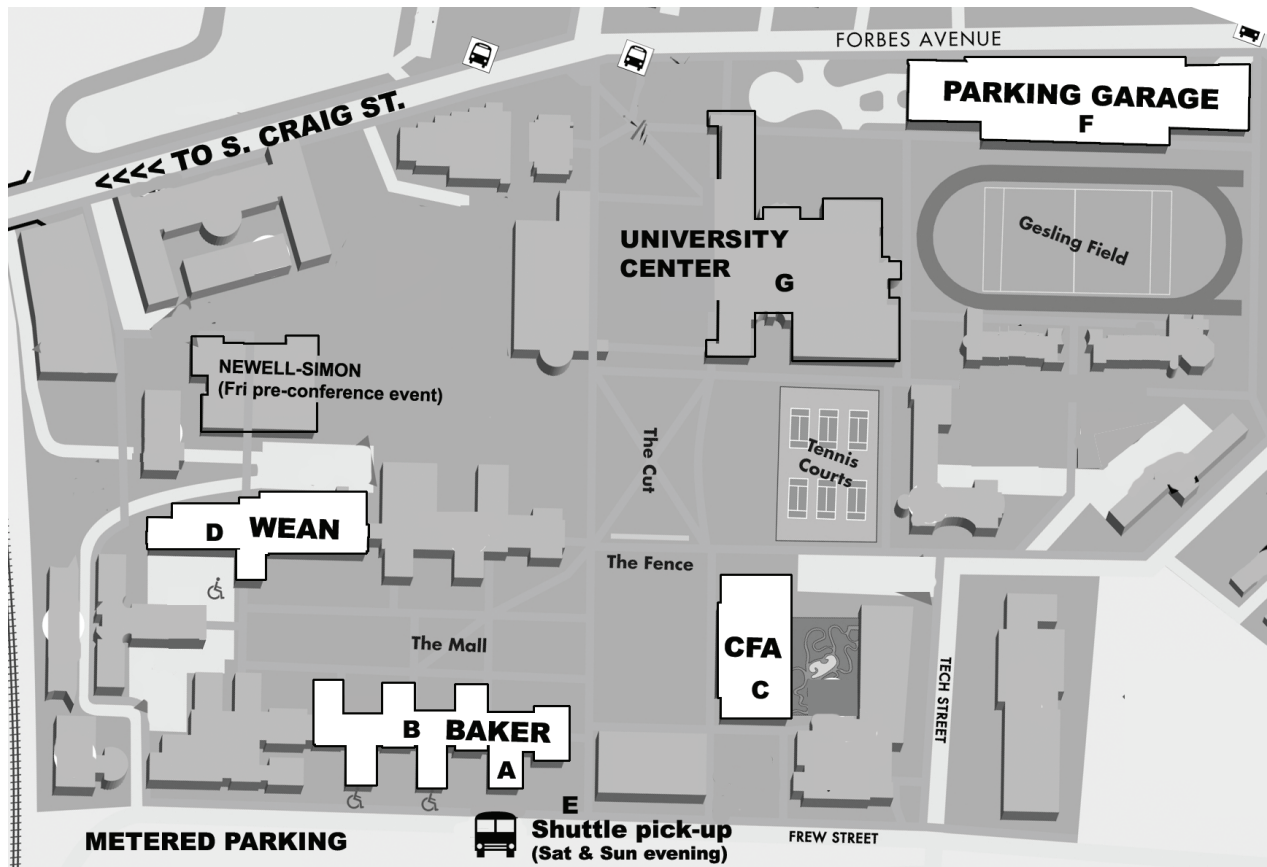
Free admission with your conference badge
Pittsburgh Center for the Arts, Sunday, noon-5pm

Monday, March 9

Short Workshops & Panels

8-9am	LOUNGE	BREAKFAST <i>Coffee and tea will be available in the lounge all day.</i>
9-10:20am	BAKER-140C/E CFA-318 CFA-317 CFA-303	Alice: Intro to Programming (Dann & Slater) Getting Started with VVVV (Oschatz) Getting Started with Processing (Greenberg) Pure Data Demo (Steiner)
10:30-noon	CFA-303 BAKER-140F CFA-318 CFA-310	Getting Started with openFrameworks (Lieberman, Watson & Castro) Getting Started with Scratch (Maloney & Eastmond) Getting Started with Processing (Shiffman) Getting Started with Extendscript (Woohoo)
noon-1pm	LOUNGE	LUNCH
1-1:30pm	BAKER-GE	Developing New Tools (panel)
1:30-2pm	BAKER-GE	Educating in New Ways (panel)

Campus Map



Walking time from the garage to Baker Hall is 5-10 minutes.

- (A)** Giant Eagle Auditorium & Main Lounge Area (in Baker Hall)
- (B)** Baker Hall workshop rooms (140C, 140E and 140F)
- (C)** CFA workshop rooms (303, 310, 317, 318, 323)
- (D)** Wean Hall workshop rooms (5201, 5202)
- (E)** Shuttle pickup area (outside main entrance of Baker Hall, along Frew Street)
- (F)** East Campus Garage
- (G)** University Center:
 - ATMs
 - Skibo Cafe (Coffeehouse): Sat 10am-6pm, Sun 10am-6pm, Mon 9am-3pm.
 - Entropy+ (Convenience Store): Sat 12-6pm; Sun 12-6pm; Mon 7:30am-6pm.
 - Si Senor (Tex/Mex Fast Food): Sat 10am-6pm, Sun 10am-6pm, Mon 11am-2pm.

Please note, Carnegie Mellon is on Spring break this weekend, so many campus facilities are closed. A wide variety of restaurants, cafes and bars are located on S. Craig Street, just two blocks from campus.

Pittsburgh Map

For an interactive Google map with walking and driving directions to hotels, restaurants, bars, galleries and other attractions, please see the conference website:

ARTANDCODE.NING.COM

Motivation

"ART AND CODE is the software wing of the DIY movement."

-- *Tom McMail, Microsoft Research*

Just as true literacy in English means being able to write as well as read, true literacy in software demands not only knowing how to use commercial software tools, but how to create new software for oneself and for others. Today, everyday people are still woefully limited in their ability to create their own software. Many would like to create their own programs and interactive artworks, but fear that programming is "too hard." The problem, it turns out, may not be programming itself so much as the ways in which it is conventionally taught.

Recently, a number of projects dedicated to democratizing the education of computational thinking have coalesced. Emerging primarily from the arts sector, a set of new programming tools (and accompanying pedagogic techniques) have been developed **by artists, and for artists**, to help regular folks and other non-computer-scientists learn to make software. Using visual and musical expression as the "hook", thousands of people have not only learned to code using these new environments, but found new reasons to code in the first place. These toolkits – many of which are free, open-source initiatives – have made enormous inroads towards expanding the computational skills and interests of hundreds of thousands of creative people worldwide. You too can join this movement!

This conference is for:

- Artists, designers and musicians who want to create interactive art, information visualization, or personal software tools;
- Teens, undergraduates, and graduate students who wish to combine art, design, interaction, and computer science;
- Middle-school and high-school teachers who want a more expressive way of teaching programming and computer arts;
- College educators and professional artists who want to learn the most cutting-edge tools for interaction design;
- Computer-science education researchers interested in visually-oriented learning tools;
- Anyone who has been wanting to learn how to program their own software, but hasn't known where to start!

-- *Golan Levin, ART AND CODE Conference Organizer*

<golan@andrew.cmu.edu>

Tools & Workshops

ActionScript

www.adobe.com/devnet/actionscript

Presented by Ira Greenberg (Author, *Processing for Flash Developers*)

Actionscript is the programming language used in Adobe's popular Flash authoring tool. Originally created as a way for developers to program interactivity in the Flash Player, ActionScript enables efficient programming of Adobe Flash applications for everything from simple animations to complex, data-rich, interactive application interfaces.

Alice

www.alice.org

Presented by Don Slater and Wanda Dann

Alice is an innovative 3D programming environment that makes it easy to create an animation for telling a story, playing an interactive game, or a video to share on the web. Alice is a freely available teaching tool designed to be a student's first exposure to object-oriented programming. It allows students to learn fundamental programming concepts in the context of creating animated movies and simple video games. In Alice, 3-D objects (e.g., people, animals, and vehicles) populate a virtual world and students create a program to animate the objects.

In Alice's interactive interface, students drag and drop graphic tiles to create a program, where the instructions correspond to standard statements in a production oriented programming language, such as Java, C++, and C#. Alice allows students to immediately see how their animation programs run, enabling them to easily understand the relationship between the programming statements and the behavior of objects in their animation. By manipulating the objects in their virtual world, students gain experience with all the programming constructs typically taught in an introductory programming course.

ExtendScript

seminars.adobe.acrobat.com/p51382259/

Presented by Dr. Woohoo

Did you know that you can write scripts to control Photoshop? To automate Illustrator? to abuse AfterEffects? You can with ExtendScript, Adobe's implementation of JavaScript. It is used by all Adobe Creative Suite applications that support a JavaScript interface.

Hackety Hack **hacketyhack.net**

Presented by why the lucky stiff

Hackety Hack is a free Ruby-based environment which aims to make programming easily available to beginners, especially teenagers. Its motivation was an essay entitled The Little Coder's Predicament, written in 2003 by a fellow called (Mr.) why the lucky stiff, which argued that programming isn't as readily available as it was in the days of the Commodore 64, and that something should be done about it to help beginners tinker with their computers.

One of Hackety Hack's sincere pledges is to make the most common code very easy and short. Downloading an MP3 can be done in one line of code. Making a blog takes about 6 lines. Constructing your own Instant Messaging system takes about twice that. Presently, Ruby is the only language taught by Hackety Hack. And it's a great one to start with.

All of this, the whole of it, is totally free to you. My wish is to spread infectious hacking smarts all over the world. And so Hackety Hack is yours forever at no cost: give it away, take it apart, learn-learn-learn without a second thought.

Max/MSP/Jitter **www.cycling74.com/downloads/max5**

Presented by Luke DuBois

Max/MSP/Jitter is an interactive graphical programming environment for music, audio, and media. For over two decades, people have been using Max/MSP/Jitter to make their computers do things that reflect their individual ideas and dreams. Compatible with Mac and Windows, Max/MSP/Jitter provides true cross-platform authoring and a free runtime version.

Max/MSP/Jitter is three things:

1. Max, a graphical programming environment that provides user interface, timing, communications, and MIDI support;
2. MSP, for real-time audio synthesis and digital signal processing;
3. Jitter, for video and matrix data processing.

Max/MSP/Jitter is a common framework in which artists and researchers from varied disciplines can collaborate and cross-pollinate, building upon their respective specialties, all within the same flexible architecture. Recent artistic and research projects that have used Max/MSP/Jitter range from evolutionary systems for generating video filters to a computer controlled paint gun printer, and from music driven by the movement of ballet dancers to a robotic labyrinth that changes form as the participant walks through it.

openFrameworks

www.openframeworks.cc

Presented by Zachary Lieberman, Theodore Watson, and Arturo Castro

Especially created for artists and designers, openFrameworks is the ideal starting point for those who want to take their first steps in C++ programming. At the same time, it is an immensely powerful tool for developing more advanced projects, providing a simple and intuitive structure for experimentation with many other open-source libraries.

The openFrameworks library is open-source, designed to work as a general purpose glue in order to wrap together several commonly-used open source libraries within a tidy interface: OpenGL for graphics, openCV for computer vision, rtAudio for audio analysis and synthesis, freeType for fonts, freeImage for image input and output, Quicktime for video playing and sequence grabbing, and many other free libraries for networking, serial communications, vector graphics output, etcetera.

The code is written to be both cross platform and cross compiler. It works in Mac, Windows and Linux operating systems -- and it compiles in readily-downloadable template projects for the free XCode, CodeBlocks, VisualStudio, and gcc compilers. The API is designed to be minimal and easy to grasp. There are very few classes, and inside of those classes, there are very few functions. The code has been implemented so that within the classes there is minimal cross-referencing, making it quite easy to rip out and reuse, if you need, or to extend.

Processing

processing.org

Presented by Casey Reas, Ben Fry, Ira Greenberg, and Daniel Shiffman

Processing is an open-source programming language, development environment, and online community that since 2001 has promoted software literacy within the visual arts. Initially created to serve as a software sketchbook and to teach fundamentals of computer programming within a visual context, Processing quickly developed into a tool for creating finished professional work as well. It is used by students, artists, designers, researchers, and hobbyists for learning, prototyping, and production. It is created to teach fundamentals of computer programming within a visual context and to serve as a software sketchbook and professional production tool for programming images, animation, and interactions.

Processing is a free alternative to proprietary software tools with expensive licenses, making it accessible to schools and individual students. Its open-source status encourages the community participation and collaboration that is vital to its growth. Contributors share programs, contribute code, answer questions in the discussion forum, and build libraries to extend the possibilities of the software. The Processing community has written over seventy libraries to facilitate computer vision, data visualization, music, networking, and electronics.

The Processing software runs on the Mac, Windows, and GNU/Linux platforms. With the click of a button, it exports applets for the Web or standalone applications for Mac, Windows, and GNU/Linux. Graphics from Processing programs may also be exported as PDF, DXF, or TIFF files and many other file formats.

Processing was founded by Ben Fry and Casey Reas in 2001 while both were John Maeda's students at the MIT Media Lab. Since that time, Processing has received many significant awards and recognitions. The Cooper-Hewitt National Design Museum (a Smithsonian Institution) included Processing in its National Design Triennial. Works created with Processing were featured prominently in the Design and the Elastic Mind show at the Museum of Modern Art. Numerous design magazines, including Print, Eye, and Creativity, have highlighted the software. For their work on Processing, Fry and Reas received the 2008 Muriel Cooper Prize from the Design Management Institute. The Processing community was awarded the 2005 Prix Ars Electronica Golden Nica award and the 2005 Interactive Design Prize from the Tokyo Type Director's Club.

Pure Data

`puredata.info`

Presented by Hans-Christoph Steiner

Pd (aka Pure Data) is a real-time graphical programming environment for audio, video, and graphical processing. It is the third major branch of the family of patcher programming languages known as Max (Max/FTS, ISPW Max, Max/MSP, jMax, etc.) originally developed by Miller Puckette and company at IRCAM. The core of Pd is written and maintained by Miller Puckette and includes the work of many developers, making the whole package very much a community effort.

Pd was created to explore ideas of how to further refine the Max paradigm with the core ideas of allowing data to be treated in a more open-ended way and opening it up to applications outside of audio and MIDI, such as graphics and video.

It is easy to extend Pd by writing object classes ("externals") or patches ("abstractions"). The work of many developers is already available as part of the standard Pd packages and the Pd developer community is growing rapidly. Recent developments include a system of abstractions for building performance environments; a library of objects for physical modeling; and a library of objects for generating and processing video in realtime.

Pd is free software and can be downloaded either as an OS-specific package, source package, or directly from CVS. Pd was written to be multi-platform and therefore is quite portable; versions exist for Win32, IRIX, GNU/Linux, BSD, and MacOS X running on anything from a PocketPC to an old Mac to a brand new PC. It is possible to write externals and patches that work with Max/MSP and Pd using flex and cyclone.

Scratch

scratch.mit.edu

Presented by John Maloney and Evelyn Eastmond

Scratch is a new graphical-programming environment that enables young people (ages 8 and up) to create their own interactive stories, games, and animations - and share their creations on the web. Scratch is designed to make programming more tinkerable, more meaningful, and more social. Since Scratch was launched in May 2007, more than 300,000 projects have been shared on the Scratch website, which has been called “the YouTube of interactive media.” As young people create and share Scratch projects, they learn to think creatively, reason systematically, and work collaboratively. Scratch is a project of the Lifelong Kindergarten group at the MIT Media Lab, directed by the distinguished educational researcher, Mitchel Resnick.

Scratch aims to broaden the audience for computer programming by making programming:

- * More tinkerable. To create programs in Scratch, you simply snap graphical blocks together into stacks. The blocks are designed to fit together only in ways that make syntactic sense, so there are no syntax errors. Different data types have different shapes, eliminating type mismatches. You can make changes to stacks even as programs are running, so it is easy to experiment with new ideas incrementally and iteratively.

- * More meaningful. We designed Scratch to support a wide range of different types of projects, so that people with diverse backgrounds and interests can all work on personally-meaningful projects. In particular, Scratch enables people to programmatically control graphics, animations, music, and sound, extending the media-manipulation activities that are popular in today’s youth culture.

- * More social. Scratch makes it easy to share projects on the web – and remix projects created by others. On the Scratch website, members are constantly borrowing and building upon one another’s ideas, images, and programs. More than 15% of the projects on the website are modified versions of other projects on the site.

Silverlight

www.microsoft.com/SILVERLIGHT/

Presented by DeVaris Brown

Microsoft Silverlight is a programmable web browser plugin that enables features such as animation, vector graphics and audio-video playback that characterise rich Internet applications. Version 2.0, released October 2008, brings additional interactivity features and support for .NET languages and development tools. It is compatible with multiple web browser products used on Microsoft Windows and Mac OS X operating systems. Mobile devices, starting with Windows Mobile 6 and Symbian (Series 60) phones, will also be supported.

VVVV

vvvv.org

Presented by Sebastian Oschatz

VVVV is a toolkit for real time video synthesis -- and a graphical programming language which allows you to draw a program while it is running. It is designed to facilitate the handling of large media environments with physical interfaces, real-time motion graphics, audio and video that can interact with many users simultaneously. VVVV uses a “visual programming” interface, wherein user programs are created by connecting various function blocks by “wires”. In this way, VVVV provides a graphical programming language for easy prototyping and development.

VVVV is real time. Whereas many other languages have distinct modes for building and running programs, VVVV only has one mode -- runtime. Write code and see the results simultaneously!

Other key features of VVVV include:

- Effortless handling of multiple objects
- Seamless multi-projection setups
- DirectX-based 3D rendering
- Realtime programming of HLSL shaders, directly on the graphics card
- Interfacing with a wide variety of external devices and protocols, including UDP, TCP, RS232, OSC, MIDI, DMX, HTTP and plugin standards like VST and FreeFrame.

Presenters



DeVaris Brown is the Academic Relations Manager for the Heartland Region of Microsoft Corporation. DeVaris has been with Microsoft since August of 2006, first serving as a Systems Engineer within the Windows Live Operations group. In this role, he was responsible for creating automation for the largest installation of Windows/SQL Server and coding libraries for the Hotmail division of Windows Live. In his current role, DeVaris travels throughout the Heartland Region of the United States, building relationships with students and faculty to increase awareness about Microsoft products, such as Silverlight, and to get people excited about technology.

DeVaris received his BS in 2006 from the University of Illinois at Urbana Champaign, with a double major in Math and Computer Science and a minor in Management Information Systems. He is a recognized expert in the fields of intrusion detection, high performance computing, and virtualization. During his college years he interned with Intel, IBM, and Cisco. Currently he is a member of ACM, IEEE, and the National Society of Black Engineers.



Arturo Castro is currently based in Barcelona. He studied computer science and has collaborated in several creative projects since 2004. Arturo works with Zachary Lieberman and Theodore Watson on the development of openFrameworks. He is the maintainer of the Linux version and leads the openFrameworks working group at Hangar (Barcelona).

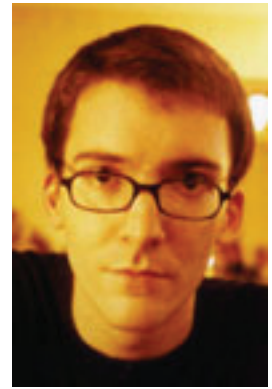


Dr. Wanda Dann, an active member of the Alice team for the last decade, has recently assumed leadership of the team. She is currently transitioning into a faculty position at Carnegie Mellon University from Associate Professor of Computer Science at Ithaca College. Wanda's research interests include visualization in programming and programming languages and innovative approaches to introductory programming.

With Dr. Steve Cooper and Dr. Randy Pausch, she has published papers on the use of program visualization in teaching and learning introductory programming. Papers have appeared in ACM's Special Interest Group on Computer Science Education (SIGCSE) inroads, the Computer Science Education Journal, and other related publications. She is co-author of Learning to Program with Alice (2006, Prentice-Hall).

Dr. Dann's leadership as a computer science educator has been recognized in her various roles as SIGCSE Technical Symposium publications editor, special projects chair, program chair, and symposium chair. She is now a member of the SIGCSE Board.

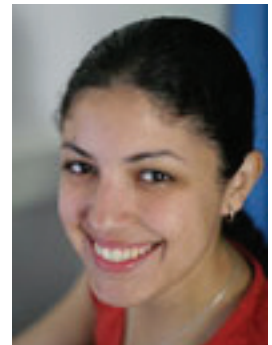
R. Luke DuBois is a composer, artist, and performer who explores the temporal, verbal, and visual structures of cultural and personal ephemera. He holds a doctorate in music composition from Columbia University, and has lectured and taught worldwide on interactive sound and video performance. He has collaborated on interactive performance, installation, and music production work with many artists and organizations including Toni Dove, Matthew Ritchie, Todd Reynolds, Michael Joaquin Grey, Elliott Sharp, Michael Gordon, Bang on a Can, Engine27, Harvestworks, and LEMUR, and was the director of the Princeton Laptop Orchestra for its 2007 season.



An active visual and musical collaborator, DuBois is the co-author of Jitter, a software suite for the real-time manipulation of matrix data. He appears on nearly twenty-five albums both individually and as part of the avant-garde electronic group The Freight Elevator Quartet. He currently performs as part of Bioluminescence, a duo with vocalist Lesley Flanigan that explores the modality of the human voice, and in Fair Use, a trio with Zach Layton and Matthew Ostrowski, that looks at our accelerating culture through electronic performance and remixing of cinema.

DuBois has lived for the last fifteen years in New York City. He teaches at the Brooklyn Experimental Media Center at NYU's Polytechnic Institute. His records are available on Caipirinha/Sire, Liquid Sky, C74, and Cantaloupe Music. His artwork is represented by bitforms gallery in New York City. DuBois holds both a bachelor's and a doctorate in music composition from Columbia University, and is a staff researcher at Columbia's Computer Music Center.

Evelyn Eastmond has been a member of the Scratch team in the Lifelong Kindergarten group at the MIT Media Lab for six years: first as an undergraduate researcher, then graduating with an MEng in Computer Science and now working fulltime as a developer for Scratch, splitting her time between her home in Madison, WI and the MIT Media Lab in Cambridge. She continues to learn and grow as Scratch itself evolves and is excited to connect with Scratch and *art and code* enthusiasts from all corners of the world. Evelyn is currently teaching herself Flex, Processing and other cool things.



Ben Fry 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 complex data. 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 Institute of MIT & Harvard. During the 2006-2007 school year, Ben was the Nierenberg Chair of Design for the the Carnegie Mellon School of Design. At the end of 2007, he finished the book Visualizing Data for O'Reilly. He currently works as a designer in Cambridge, MA.



With Casey Reas of UCLA, he currently develops Processing, an open source programming environment for teaching computational design and sketching interactive media software that won a Golden Nica from the Prix Ars Electronica in 2005.

In 2006, Fry received a New Media Fellowship from the Rockefeller Foundation to support the project. Processing was also featured in the 2006 Cooper-Hewitt Design Triennial. In 2007, Reas and Fry published *Processing: A Programming Handbook for Visual Designers and Artists* with MIT Press.



Ira Greenberg is an associate professor at Miami University of Ohio, with a joint appointment in the Department of Art and the Armstrong Institute for Interactive Media Studies. He is also an affiliate faculty member of the Department of Computer Science and Systems Analysis. His research interests include aesthetics and computation, expressive programming, emergent forms, net-based art, and computer art pedagogy. During the last few years, he has been torturing defenseless art students with trigonometry, algorithms, and object-oriented programming and is excited to spread this passion to the rest of the world. He is the author of *Processing: Creative Coding and Computational Art*, Friends of Ed, 2007 and the upcoming *Processing for Flash Developers*, Friends of Ed, (summer, 2009).



Golan Levin is Director of the STUDIO for Creative Inquiry and Associate Professor of Electronic Art at Carnegie Mellon University, where he also holds courtesy appointments in the School of Design and the School of Computer Science. Golan is the organizer of the Art and Code Conference.

Golan's pedagogy is concerned with reclaiming computation as a personal medium of expression. To that end, his courses are designed to give students the confidence to program their own software creations from first principles. His studio classes focus on significant themes in contemporary electronic media arts, such as interaction design, computational form generation, information visualization, and audiovisual performance. These function as "studio art courses in computer science," in which the objective is to produce personally and socially relevant expressions, but the medium is software created by the students themselves. Golan's own work investigates formal languages for visualization and interactivity in cybernetic systems. He is known for the conception and creation of *Telesymphony*, a concert whose sounds are wholly performed through the carefully choreographed ringing of the audience's own mobile phones, and for interactive information visualizations like *Secret Lives of Numbers* and *Dumpster*, which offer novel perspectives onto millions of online communications. Golan has exhibited and performed widely in Europe, America and Asia.



Zachary Lieberman's work uses technology in a playful and enigmatic way to explore the nature of communication and the delicate boundary between the visible and the invisible. He creates performances, installations, and on-line works that investigate gestural input, augmentation of the body, and kinetic response.

Working with collaborator Golan Levin, he has created a series of installations - "Remark" and "Hidden Worlds" - which presented different interpretations of what the voice might look like if we could see our own speech. These were followed with "Messa Di Voce," a concert performance in which the speech, shouts and songs of

two abstract vocalists were radically augmented in real-time by interactive visualization software. The collaborators have toured and exhibited their works widely, much to the delight of their audiences. Lieberman's installation / performance "Drawn," in which live painted forms appear to come to life, rising off the page and reacting to the world around them, recently won awards in the Ars Electronica and CYNETart competitions. Most recently, he presented "Opensourcery," collaboration with Spanish magician Mago Julian, in which open source software is combined with traditional close-magic to create a completely new realm of tricks.

Lieberman has held artist residencies at Ars Electronica Futurelab, Eyebeam, Dance Theater Workshop, and most recently at the Hangar Center for the Arts. He is also the founder, with Theo Watson and Arturo Castro, of openFrameworks, an open source toolkit for creative coding in C++.

John Maloney is the lead programmer for Scratch, a new programmable toolkit that lets kids create and share their own games, animated stories, and interactive art. Scratch is designed to help people learn programming and problem solving as they create personally meaningful artifacts.

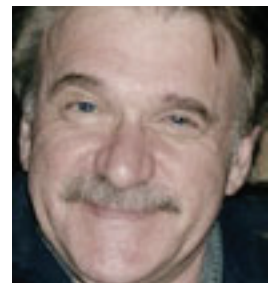
Prior to joining the Lifelong Kindergarten John worked for computer pioneer Alan Kay. Under Alan, John developed key parts of Squeak, an experimental programming system for elementary school children. He also worked at Walt Disney Imagineering, where he built an experimental handheld electronic guide for theme park guests and software for a new attraction that earned the theme park equivalent of an Academy Award. Prior to working at Disney, John worked at Apple, Sun Microsystems Labs, and Xerox.

John has a Ph.D. in Computer Science from the University of Washington and M.S./B.S. degrees in Electrical Engineering and Computer Science from MIT. John enjoys singing and playing the recorder. He also likes bicycling, hiking, and cross-country skiing.

Tom McMail is passionate about innovation, education and technology as means for improving the human condition. After his first university training in Psychology and Psycholinguistics, he taught K-12 in many subjects and grade levels, then took a role as social worker for Head Start. He has also worked as a professional musician and composer, plays more than 20 instruments and has taught improvisation, as well as music theory and composition.

After returning to school for a degree in Computer Science, he worked in the electronic gaming industry as a computer music and technical audio expert, and later became a producer and developer of educational software.

Overall, Tom has spent nearly 14 years at Microsoft in a variety of roles. At MSN, he used new technologies and techniques in support of special events and also created innovative plans for online community adopted across Microsoft.com properties.



With MSR University Relations, Tom migrated regional efforts to a more strategic approach, directly addressing academic concerns about declining enrollments, gender disparity, and software security, introducing Tablet PCs to universities and driving experimentation to find best uses of technology to transform education. He was instrumental in introducing programs employing Gaming and Robotics as change agents for reinvigorating CS curriculum.

At Microsoft Research, in the External Research group, for the past two years he focused on seeking the most innovative new research and researchers emerging today as part of the Breakthrough Research initiative. Currently he is responsible for strategic collaboration programs with North American academic institutions and researchers.

Tom lives outside of Monroe, Washington with his family in the beautiful rural foothills of the Cascade Mountains.



Sebastian Oschatz is one of the founders of the Frankfurt-based media company MESO Digital Interiors, established in 1997 to work with experimental media interfaces and interactive installations. MESO creates computational and interactive exhibition designs for clients like Mercedes Benz, BMW, Nikon and Sony Ericsson, among others.

MESO has also been the home of the developers of the visual programming tool VVVV, created originally to run MESO's own projects. VVVV has since snowballed into a freely available multipurpose toolkit with an independent group of developers and a growing fan base. VVVV is well-suited for realtime video synthesis, and gives easy access to DirectX-shaders and a range of control protocols like MIDI, OSC and DMX-512. Little-known in the United States but heavily used throughout Europe, VVVV is an excellent tool for creating sound-responsive visual performances, although it has also been developed with interactive installations in mind.



Casey Reas is an associate professor and chair of the Department of Design Media Arts at UCLA. His classes provide a foundation for thinking about software as a dynamic visual medium and set a structure for inquiry into synthesis of culture, technology, and aesthetics. With Ben Fry, Reas initiated Processing.org in 2001. Processing is an open source programming language and environment for creating images, animation, and interaction. In September 2007, they published *Processing: A Programming Handbook for Visual Designers and Artists*, a comprehensive introduction to programming within the context of visual media (MIT Press). Reas' essays have appeared in the books *Network Practices* (Princeton Architectural Press), *Aesthetic Computing* (MIT Press), *Code: The Language of Our Time* (Hatje Cantz), and the *Programming Cultures* issue of *Architectural Design* (Wiley).

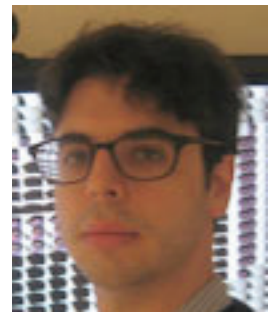
Don Slater was already a secondary school teacher in Western Pennsylvania when the first personal computers were introduced into his building. Don became a computer science teacher, when he realized he wanted to help others discover the cool things that they might be able to build with computers, as he had. Don taught and served as the technology director at Sewickley Academy, a private school in the Pittsburgh area. In 1998, he began teaching introductory programming part time at Carnegie Mellon. In 2000, he was invited to become a full time member of the Computer Science Department as a Lecturer.



Don joined the Alice Team in the fall of 2005 as he introduced Alice into his Introductory Programming course. Wanda Dann happened to be on sabbatical at Carnegie Mellon that fall, and she became a valuable advisor as they explored the using Alice in his course. With Wanda and Steve Cooper, Don has since presented at teacher training workshops around the country, as well as at various conferences, including NECC, SIGCSE, IMICT, and ISECON.

Don is a consultant for the College Board in Advanced Placement Computer Science. He has served as a reader and leader at the yearly Advanced Placement Computer Science Exam grading since 1991. He has written articles for the College Board and led teacher workshops throughout the east, and now directs the Advanced Placement Teacher Workshop held each summer at Carnegie Mellon.

Daniel Shiffman is an Assistant Arts Professor at the Interactive Telecommunications Program (ITP) 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 is the author of Learning Processing: A Beginner's Guide to Programming Images, Animation, and Interaction (Morgan Kaufmann Publishers, Elsevier Inc.)



Hans-Christoph Steiner spends his time designing interactive software with a focus on human perceptual capabilities, building networks with free software, and composing music with computers. With an emphasis on collaboration, he has worked in many forms, including responsive sound environments, free wireless networks that help build community, musical robots that listen, software environments that allow people to play with math, and a jet-powered fish that you can ride. To further his research, he teaches and works at various media art centers and organizes open, collaborative hacklabs and barcamp conferences. He is currently teaching courses in dataflow programming NYU's Interactive Telecommunications Program.



His research focuses on making software tools enable read/write literacy, mostly focusing on Pure Data, a graphical dataflow programming language. He also is an active sound designer and artist, and draws the inspiration for shaping the software he works with from the creative projects he works on.



Theodore Watson is an artist, designer and experimenter whose work is born out of the curiosity and excitement of designing experiences that come alive and invite people to play. Theodore's work ranges from creating new tools for artistic expression, experimental musical systems, to immersive, interactive environments with full-body interaction. His recent work includes the Graffiti Research Lab's Laser Tag, laser graffiti system and Funky Forest, an immersive interactive ecosystem for young children. Theodore works together with Zachary Lieberman and Arturo Castro on openFrameworks, which is an open source library for writing creative code in C++.

Watson's work has been shown at MoMA, Tate Modern, Ars Electronica, The Sundance Film Festival, Res Fest, REMF, Cinekid, Montevideo, OFFF, SHIFT, ICHIM, The Creators Series, Deitch Projects, Eyebeam, Pixel Gallery, Museum N8 Amsterdam.



why the lucky stiff (or `_why`) is a computer programmer. His best known work may be Why's (poignant) Guide to Ruby, a book which teaches the Ruby programming language with stories; its eclectic style has been compared to a "collaboration between Stanislaw Lem and Edward Lear". Chapter 3 of this Guide was republished in *The Best Software Writing I*, edited by Joel Spolsky.

Most recently, Why has focused his efforts on the problem of how to better teach programming, and how to make programming more appealing to young people. His latest project, Hackety Hack, is a Ruby-based environment used to teach programming to children. His most vocal critics describe him as "a fledgling freelance professor, should one adhere to the most fraudulent of definitions." His biographical information, matter of point, is riddled with accolades which are clearly either lifted or falsified in order to cast him in a good light -- let's say 40 watts of violet."



Dr. Woohoo (translation: serious fun) is a New Mexico-based artist and programmer. Woohoo sculpts the form of his generative artwork and then harnesses natural phenomena (wind, gravity, etc.) and music, through algorithms, to drive his brushstrokes and composition. With a DIY attitude, he creates his own color, paint and visual effects applications using Adobe AIR and ExtendScript, and rather than recreate existing features and functionality, Woohoo integrates his tools with applications like Photoshop, Illustrator and Maya. Woohoo's latest work explores the dichotomy between Population vs. Consumption from the abstract perspective of Mother Nature. For more information on Dr. Woohoo, you can find him at DrWoohoo.com or in the featured article of the March issue of Photoshop User magazine.

Afterword

The Little Coder's Predicament by Why the Lucky Stiff (2003)

Used with permission

Okay, then, children of the modern age (where we live in a world so tied together with wires that Pangaea ain't goin' nowhere!), you tell me if this is a predicament or not.

In the 1980s, you could look up from your Commodore 64, hours after purchasing it, with a glossy feeling of empowerment, achieved by the pattern of notes spewing from the speaker grille in an endless loop. You were part of the movement to help machines sing! You were a programmer! The Atari 800 people had BASIC. They know what I'm talking about. And the TI-994A guys don't need to say a word, because the TI could say it for them!

The old machines don't compare to the desktops of today, or to the consoles of today. But, sadly, current versions of Windows have no immediately accessible programming languages. And what's a kid going to do with Visual Basic? Build a modal dialog? Forget coding for Xbox. Requires registration in the Xbox Developer Program. Otherwise, you gotta crack the sucker open. GameCube? GameBoy? Playstation 2?

Coding Just Isn't Accessible

Yes, there are burgeoning free SDKs for many of these platforms. But they are obscure and most children have no means of actually deploying or executing the code on their own hardware! This is obvious to us all and likely doesn't seem such a big deal. But ask yourself what might have happened had you not had access to a programming language on an Atari 800 or a Commodore. You tell me if this is a predicament.

It turns out, most of the kids in my neighborhood are exposed to coding through the TI calculator. A handful of languages are available on the TI and its processor is interesting enough to evoke some curiosity. But this hasn't spread to its PDA big brothers, where young people could have more exposure to programming. And undoubtedly the utility of a language on the Palm, Pocket PC and others would be useful to many.

So what's the problem here? We have no shortage of new languages, but they become increasingly distanced from the populace. Are the companies behind these platforms weary of placing the power of a programming language in the hands of users? Is there not a demand any longer? It's got to be some kind of greed, power, money thing, right?

Perhaps this is just another reason to push Linux and BSD on consumer systems. Still, are scripting languages easily accessible to beginners on those systems? OSX has made several scripting languages available (including Ruby and Python), but most users are unaware of their presence.

I should mention that Windows is equipped with its own scripting host for developing in JScript and VBScript. But the use of the scripting host is (I believe) under-documented and limited for beginners. Try doing something useful in a script without using `Server.CreateObject`. Let's not let kids touch the COM objects, please!

The Christmas List

I'm thinking a toy language for consoles and desktops alike could be monumental. I'm not saying it needs to be cross-platform. A language for GameCube that took advantage of platform-specific features could be more appealing to GameCube users than a language that used a reduced featureset, but could execute on a handheld. Really, we live in a world where both choices should be available.

As for essential features:

1. Transportable code.

On my TI-994A, I could make a little, animated Optimus Prime from pixels. Insert cassette. Record. Pass around to friends. Receive high fives from friends. Put on wraparound shades. Thank you, TI! Thank you, Optimus Prime!

A little language for the consoles could be wildly popular if combined with the good nature of sharing code. This could be done by trading memory cards, but would be more effective if code could be easily obtained and posted on the Web. Learning would accelerate and collaborative development could take place.

A suitable language should give coders access to I/O devices, to allow experimentation with network devices and the ability to enhance one's connectivity with others. For the consoles, games could provide hooks for user mods. This has long proven a successful staple of the desktop gaming world.

2. Simplicity.

You've got to be able to write a single line of code and see a result. We need some instant results to give absolute beginners confidence. Simple methods for sending an e-mail, reading a web page, playing music. Demonstrable in a one-liner.

Admittedly, as our systems have grown complex, it is difficult to balance simplicity and capability. Most users will be unimpressed by code that emits beeps and bleeps from a PlayStation 2. If Ruby were available on the PS2, then I would hope that I could hear rich symphonic sounds from a wee bit of code.

```
Orchestra.play( "A:2", "C:4", "E:1", "G:1" )
```

Access to the graphic engine might require more complex code. But simple drawing methods could be provided for beginners. Or images could be stored alongside code and accessed programmatically.

```
ImageLibrary.load( "GolfingOldMan" ).drawAt( 12, 10 )
```

The trick would be to uncover what small applications might entice novices and still provide the ability to write large applications that would drive developers to master the language and not limit their growth.

3. *Sensible environment.*

Considering that many won't want to purchase a keyboard for their gaming unit, let's make sure that a reasonable environment is provided for entry of text. Controllers could be worked like the Twiddler. Or code could be transferred via IR, TCP/IP. (Dare I say cassette? :D)

4. *Give it away!*

It used to be that programming was practically an inalienable right for users. Include a language with the system, situated in a friendly spot. Each of the game consoles I've mentioned has launchers. (With the exception of Game Boy and its successors.) Provide a development prompt from the launcher. From desktop software, provide shortcuts for both the command prompt and a development prompt.

Remember, we're looking for a language that requires no system hacks. No obscure links. No warranty violation. We've become so used to these techniques that it seems to be an essential part of getting our way.

And in many ways it is essential. Tinkering with hardware is learning. Lobotomizing and renovating is meaningful, magical. On behalf of those who prefer to code, I make these wishes. Not to take away jobs from the Phillips screwdriver.

The Ultimatum

My challenge is to Sony, Nintendo, Microsoft, Apple, and to those who manufacture and develop our interactive technology. Let us interact with these machines more deeply. Provide us a channel for having a dialogue with the entertainment boxes we nurture and care for. I swear to you, the relationship between the public and your product will assuredly blossom. That box will become more of a chest for our personal works.

In addition, if your developers start putting out crap, then you have a whole world of people to pick up the slack.

My challenge is for you to bundle a useful programming language with your product. Ruby, Squeak, REBOL, Python. Take your pick. It will be inexpensive to add any of these languages to your systems. And people will seriously pray to you. You know how geeks get when they pledge allegiance to something. But, yes, Ruby is preferable.

Acknowledgments

ART AND CODE is made possible by a generous grant from Microsoft Research, with oversight by the Center for Computational Thinking at Carnegie Mellon University. ART AND CODE is a project of the CMU STUDIO for Creative Inquiry, directed by Golan Levin. We express our gratitude to Microsoft Research and the CMU Center for Computational Thinking for their sponsorship, and to the STUDIO for Creative Inquiry and the CMU School of Art for their administrative support.

Conference Production Credits

Conference Organizer: Golan Levin (STUDIO for Creative Inquiry and School of Art, CMU)

Executive Producer: Peter Lee (Center for Computational Thinking and Computer Science Department, CMU)

Business Manager and Logistics: Jennifer Brodt (STUDIO for Creative Inquiry, CMU)

Additional Communications and Logistics: Marge Myers (STUDIO for Creative Inquiry, CMU)

Registration, Catering, Additional Logistics: Carolyn Manley (Conference Services, CMU)

Web Design and Communications: Drue Miller (drue.com)

Production Assistant and Coordinator: Courtney Dow (CMU School of Art)

Registration Database Rehabilitation: Andrea Boykowycz (Twirt.com)

Volunteers: Ally Reeves, Alyssa Reuter, Angela Wang, Caryn Audenried, David Kjos Jr., David Yen, Eamae Mirkin, Jet Townsend, Julie Mallis, Lea Allbaugh, Remy Tseng, Riley Harmon, Robert Kotcher, Robyn Gray, Xiaoyuan Jiang, Yang Shan

Additional Credits and Acknowledgments

Exhibition Support: Adam Welch and Stefanie Moser (Pittsburgh Center for the Arts)

Exhibition Fabrication & Production: Jeremy Ficca and P. Zach Ali (Digital Fabrication Laboratory, CMU)

Film Series Programming: Gary Kaboly (Pittsburgh Filmmakers)

Dorkbot Co-Director & Liaison: Jet Townsend (Tangible Interaction Design program, CMU)

Great Good Place Maker: Eric Stern (brillobox)

We are grateful to CMU Cluster Services for their efforts in providing the best computers available for our workshops. We also express our gratitude to CMU Conference Services, for assistance in coordinating our lecture spaces and food service; and to the CMU School of Art for a variety of support, and in particular, Lauren Goshinski, Robert Kollar, Cindy Lammert, and Janice Hart of the CMU School of Art for their flexibility and assistance.

ART AND CODE is a symposium on programming environments for artists, young people, and the rest of us. The event takes place the weekend of March 7-9, 2009 on the campus of Carnegie Mellon University in Pittsburgh, PA. It features hands-on workshops and a conference showcase for 11 different creative toolkits -- programming languages made by artists, for artists.

Saturday March 7th features intensive three-hour workshops, taught by leading developers and authors, in a variety of programming environments. Some workshops are intended for more advanced users, but many are introductory in nature and are suited for all ages. Some workshops are specifically for teens and 'tweens.

Sunday March 8th offers a day-long sequence of 45-minute presentations about each programming environment. Learn about who is using each programming environment, and why. Many of these lectures are given by the main people creating these toolkits.

Monday March 9th begins with two morning sessions of "Hello World" workshops that are ideal for beginners or people wanting a quick "leg up" into learning a new toolkit. These are followed, after lunch, by panel discussions featuring developers and educators.

Throughout the weekend are a variety of related events, including an exhibition of computational art at the Pittsburgh Center for the Arts, a festival of abstract film at Pittsburgh Filmmakers, an open-mic Dorkbot (show-and-tell / pecha-kucha) at the brillobox, and plenty of social meals.



printed on recycled paper