MMDI 131: Introduction to Interactive Programming: Creative Coding, 3 credits

Katherine Bennett, Assistant Professor
Email: Kabennett@uarts.edu
Office Hours: Tues 5:30-7, Thurs 7-8:30
Office Location: 1211a Terra (by DJ booth)

This course is an introductory programming class, appropriate for students with no programming experience. Traditionally, introductory programming teaches algorithmic problem-solving, where a sequence of instructions describe the steps necessary to achieve a desired result. In this course, students are trained to go beyond this sequential thinking - to think concurrently and modularly. By its end, students are empowered to write and read code for event-driven creative applications, interactive installations, graphical user interfaces, networked video games and user interfaces.

Required Materials

We will utilize a variety of tools to make this class successful and to create your class projects.

- **Texts and Readings**:
  - Learning Processing by Daniel Shiffman (purchase)
  - Programming Interactivity, 2nd Edition by Joshua Noble (purchase)
  - See the CMT for additionally required ones (handouts)- http://cmt.polishedsolid.com/
- A Github account - https://github.com/
- Processing - http://processing.org/
- openFrameworks - http://www.openframeworks.cc/
- XCode 4
- Laptop
- Proper utilization of resources outlined

Academic Program

This course reinforces the Multimedia Program goals by:

- Integrate multiple mediums of image, text, sound, physical objects, space, and interactivity. (reinforced)
- Adapt to constantly changing technological paradigms by learning how to learn. Skills acquisition.
- Combine research and studio practice through inquiry and iteration. (reinforced)
- Collaborate across disciplines.
- Adopt and use professional practices. (reinforced)
- Integrate Cultural Literacy
- Adapt to constantly changing technological paradigms by learning how to learn. Skills acquisition.

Multimedia Mantra: Investigate. Immerse. Iterate.

We insist that our students:

Look. We search for opportunities for change, and have the flexibility to navigate an ever-changing world.

Question. By questioning and responding today, we shape and envision tomorrow.

Do. Our studio practices bridge disciplines to create new experiences.

Connect. We identify and engage audiences.

Course Outcomes/Goals

- Develop procedural thinking skills
- Develop and refine problem solving and independent learning skills
- An alleviation of the fear of programming (it's not that bad) and acquisition of progamming principles
- Have an ability to write code which controls content and interactivity, actualizing programming principles and creating 3 interactive applications
- Have confidence in building something from the ground up - risk taking and experimentation
- Discuss current work in the field of software & interactive art

Major Learning Objectives/Competencies

- Students will gain an understanding of programming and programming topics: Object Orientated Programming, IDE's, Git & Version Control and Addon's & Libraries

Assessment

At the course level, professors will evaluate student work through critique and written feedback (which will be delivered by email). Students will evaluate how they progressed in a course in two ways: 1) A course evaluation form, which will be developed based on the course's individual objectives. 2) Students will participate in a self reflection assignment due April 4th or as otherwise noted.

- Critically analyze/evaluate how much time was spent learning syntax & structure, programming concepts vs. actually programming, and how does this reflect on the final quality of your end result.
- Comment on your openFrameworks and Processing successes and frustrations.
- Compare and contrast openFrameworks versus Processing

Class Format

This is a studio course. Students will work on individual projects and group collaborations. Work will be completed both inside and outside of the class. Class will be a mix of lectures, class discussion, workshops, studio time and critiques. Guest critics may be present.

The course can be broken down into 3 sections: programming basics, Object Orientated Programming (OOP), and advance topics (automation/behaviors, using addons - JSON, API's, etc).

A typical day will aim for the following schedule:

1:00 - 3:00 Demo/Lecture/Workshop/Discussion
3:00 - 3:30 Show and Tell Presentations
short break
4:00 - 6:30/7 Pair programming workshop for class exercises
6:30 - 7 Showcase of class exercises

Important Aspects of this Course

Pair programming is a popular way to learn and to work on projects. Two people share one computer and write code together. For the second half of class, an exercise will be given which will be worked on in pairs. The exercise will be due at the end of the class period and presented to class. On occasions the deadline will be at another point. Yet, the exercise is meant to be worked on together - on one laptop, in one program. Choose either oF or Processing. Alternate environments for your next exercise.

Homework projects will be turned in via *GitHub*. GitHub is a web hosting service that supports Git version control. It is more important than your resume, whether you are an artist or designer. It's used larger for software, but can be used for anything. It's important to become familar with it, develop good habits (it's like Time Machine, but better). See the assigned reading links for more information.

Time. *Time management is everything.* You will need it to have reasonable accomplishments in this class. The material is cumulative. The readings are essential. Writing code is the only way to make progress and to fully synthesize the material. It becomes evident quite quickly. Develop excellent time management. You will need *8-10 hours outside of class* to succeed.

Projects are to be completed in *BOTH* environments - **Processing and oF**. They are not group projects.

Show and Tell

You will be assigned a Show and Tell presentation date. See the CMT for the calendar. Please research the assigned artist/designers/firms/pieces/collectives in advance. Be able to address the following, along with visual examples of their work:

What is original/unique about how this artist uses technology?

(if possible) What kind of code did the artist use? (JavaScript? Flash? Procesing? openFrameworks?)

Why is it necessary that the artist use code? (ie: particularly if it's non-interactive, could it have been achieved with traditional or linear media?)

Required Resources

Github vs Linked in : http://www.ittybittyurl.com/32g6, http://www.ittybittyurl.com/32g7

http://lifedev.net/2008/07/programmer-creativity-boost/

http://www.openframeworks.cc/tutorials/

http://www.openframeworks.cc/

http://www.openframeworks.cc/documentation/

http://wiki.openframeworks.cc/index.php?title=Main_Page

http://ofxaddons.com/

http://natureofcode.com/

Douglas Rushkoff's *Program or Be Programmed*, Ch 1. (CMT)

Fabulous References and Supporting Resources

Programming (general) and other languages

- http://worrydream.com/LearnableProgramming/
- http://www.udacity.com/
- http://www.codeschool.com/
- Code Academy - http://www.codecademy.com/#!/exercises/0
- https://coderwall.com/
- http://www.ittybittyurl.com/32g9 <-- list of online references and learning places for programming
- http://projecteuler.net/problems
- http://teamtreehouse.com/library
- http://learnpythonthehardway.org/
- http://ruby.learncodethehardway.org/
- http://codingbat.com/ (Python and Java)
- http://inventwithpython.com/
- http://creativejs.com/2011/08/31-days-of-canvas-tutorials/

Processing

- http://funprogramming.org/
- http://forum.processing.org/
- http://processing.org/learning/
- http://www.learningprocessing.com/tutorials/
- http://www.plethora-project.com/2011/09/12/processing-tutorials/
- http://flowingdata.com/2010/04/13/data-visualization-tutorial-in-processing/
- http://www.lynda.com/Processing-training-tutorials/1402-0.html

openFrameworks

- https://sites.google.com/site/ofauckland/home
- https://github.com/jefftimesten/CodeForArt
- https://github.com/openframeworks/openFrameworks/wiki/Tutorials,-Examples-and-Documentation
- http://www.quietless.com/kitchen/getting-started-with-openframeworks/
- http://www.memo.tv/simple-openframeworks-application-on-iphone-sample-1/

Git

- http://gitref.org/
- http://ndpsoftware.com/git-cheatsheet.html#loc=local_repo;
- http://git-scm.com/docs
- http://rogerdudler.github.com/git-guide/
- http://www.lynda.com/GitHub-training-tutorials/1384-0.html

Xcode

- http://www.lynda.com/Xcode-training-tutorials/451-0.html

Grading

Exercise 1, 2, 4	10 percent
Exercise 3, 5, 6	20 percent
Presentation	10 percent
Project 1: Adjective	20 percent
Project 2: Choice (critiqued)	30 percent
Participation	10 percent

For each project your grade will be assesed upon the following: **Principles, Creativity/Thoughtfulness, Craft**

	Superior - A	Satisfactory - C	Unacceptable - D/F
Principles	Your work shows evidence and understanding of programming concepts dicussed in readings, lectures, and exercises, as you bend them to your will.	Your work shows evidence of concepts and is still developing an understanding of course material. You understand some aspects, but aren't utilizing them fully. Usage is cursory. Keep pushing your work and review the material to revisit how you can integrate it to your work.	Your work shows some evidence of concepts discussed, but lacks key understanding, confidence, roubustness and authority. Aspects are lacking. Reviewing course material is required. Ask questions in class. Manage your time better.
Creativity, Thoughtfulness	Your work demonstrates your personality and a great depth of engagement with the material. It's extremely evident that you are thinking, exploring, playing and taking risks. You are creating wonderful experiences.	You are executing your ideas, but more time is needed to consider more deeply about what is conceptually and physically happening. They are barely getting off the ground. Or have large bits that are broken. Do more research and exploring. Play.	Your work is so straightforward that it's flat. It's barely coded (or badly broken). It's copied from elsewhere and not expanded upon. Question and Iterate your work to push your it further. Read. Play. Get off the lame track.
Craft	Your work shows delicate care and consideration to presentation and professionalism. You code is neat, clean, commented and structured. Your friends consider you "Type A." Your style is evident.	Your work shows the birth of your ideas, but further time and iteration can really push your work to accel. Your code is there, but messy. It can be simplifed and made cleaner. What you put into it, is what you get out of it. Practice makes perfect!	Your work is rushed and looks like it was done on your train-ride in or the night before. Make your work something you are proud of. You are here to build your portfolio after all, aren't you?

Participation will be based upon:

Professionalism
Engagement
Improvement

	Superior - A	Satisfactory - C	Unacceptable - D/F
Professionalism	Always arrives on time; assignments turned in properly and on time; respectfil of others in class and gives feedback appropriately.	Ususally arrives on time; most assisgnments turned in on time; listens to others.	Often tardy; turns in assignments late; fails to prepare for class; in attentitive to instructor or other students
Engagement	Always contributes appropriately to class discussion; frequently offers to demonstrate technique; shows leadership in group projects.	Usually contributes to class discussion; has demonstrated technique; participates actively in group projects.	Does not participate in class discussion; no evidence of technique; fails to contribute adequately to group projects.
Improvement	Shows an exceptional and growing understanding of technique; builds on previous lessons; accepts critique and makes proper adjustments.	Technique is developing; has shown some ability to build on previous lessons; generally able to accept critique.	Little or no development of technique; is unable or unwilling to accept critique; unable to make adjustments.

The numerical breakdown for letter grades is as follows:

A	100-90	Excellent
B	89-80	Very Good/Good
C	79-70	Satisfactory
D	69-63	Poor; Below Average
F	63 & below	Unacceptable

The University of the Arts Grading Policy can be found in the current Course Catalog.

Attendance

Unexcused absences (without a note from a doctor or agency) will affect your grade. Attendance is taken at the beginning of class.

- One unexcused absence is allowed; after that, your final overall grade for the course will drop by 2.5 points out of 100 for each additional absence.
- 3 unexcused absences is grounds for dismissal from the course and a failing grade.
- All presentation and critique days are mandatory. Being absent on critique days, affects your grade for that project as well as your participation grade. Work is still due on due dates, regardless of whether you are in class or not. Critiques can not be made up.
- Contact me in advance if you will not be in class. (Email is preferred) Demonstrate time management, communication and respect.
- Be on time! Lateness will affect your grade. For every 15 minutes you are late, your participation grade drops.
- Material missed is the responsibility of the student to make up. Speak to other classmates in order to find out what was lectured and discussed. It is your responsibility to go over that material independently. It is your responsibility to contact me if you continue to have difficulty with the information.

The University Policy on absences can be found in the Course Catalogue.

Contacting Me

Email is the preferred initial contact (kabennett@uarts.edu). Email is suitable for short questions (to answers that cannot be found in the syllabus), to set up appointments, or to notify me about being late or absent. Emails should not be longer than 5 sentences.

For matters longer than 5 sentences, please email me to make an appointment during office hours. This is the proper way to address longer questions, issues, to ask me about an assignment, review a grade you received, or to discuss other matters. If you send me a long email, I am simply going to respond by asking you to meet with me to resolve the matter. I will not resolve matters over email.

If I have asked you to schedule an appointment with me, I am not trying to avoid you. I am offering you my full attention and preparation. Due to my schedule and other campus responsibilities, I ask you to respond to

Twitter Resources and Inspiration

Joshua Noble - @fctry2
Joshua Nimoy - @jtnimoy
Red Paper Heart - @redpaperhearts
Creative Coding - @creativecoding
Casey Reas - @Reas
Daniel Shiffman - @shiffman
Marius Watz - @mariuswatz
Amit Pitaru - @pitaru
Processing Tips - @ProcessingTips
Creative Applications - @creativeapps
Memo Akten - @memotv
Seb Lee-Delisle - @seb_ly
Eyeo organizers - @eyeofestival
Kyle McDonald - @kcimc
Jer Thorp - @blprnt
Golan Levin - @golan
Hacker School - @hackerschool

Technology in the Classroom

Laptop use is fine if you are using your laptop to present in class, or if we're in the middle of an exercise that makes use of it. Otherwise, however, please keep your laptop closed. The quality of the class depends in large part on the quality of your attention and active participation. In particular, give your fellow students the respect you deserve in return, and close your laptop and give them your full attention when they are presenting work. Cellphone usage, texting and social media usage are frowned upon. You don't want to be embarassed by being caught.

University Policies on Technology can be found in the Course Catalogue.

Academic Integrity

"Violations of academic integrity are considered to be acts of academic dishonesty and include (but are not limited to) cheating, plagiarizing, fabrication, denying other access to information or material, and facilitating academic dishonesty, and are subject to the policies and procedures noted in the Student Handbook (p. 27-28) and within the Course Catalog, including the Student Code of Conduct and the Student Judicial System. Please note that lack of knowledge of citations procedures, for example, is an unacceptable explanation for plagiarism, as is having studied together to produce remarkable similar papers or creative works submitted separately by two students, or recycling work from a previous class."

Please see the Student Handbook for further information regarding this, including procedures taken for addressing such violations. Procedures may include, but are not limited to: failing the assignment, failing the course, going in front of an academic judicial council and possible suspension from school. Violations will not be tolerated.

The University of the Arts policy on Academic Integrity may be found in the Course Catalogue.

Disability Access Statement

The University of the Arts is committed to assuring equal educational opportunity and full participation for all students. The mission of the Office of Educational Accessibility is to provide individuals with disabilities the same access to programs and activities as other students. We assist students to maximize their potential while helping them develop and maintain independence.

If you have a disability-related need for modifications or reasonable accommodations in this course, please contact OEA, Hamilton Hall, room 221C, (215) 717.6616. Or visit their website at: http://www.uarts.edu/studentlife/disability.html. Instructors should be notified as early as possible regarding the need for modifications or reasonable accommodations. I need to be notified within the first 2 weeks of class. If you anticipate needing any type of accommodation in this course or have questions about physical access, please contact UArts' Disability Services to verify your eligibility accommodations. Appropriate documentation from the Disability Services verifying eligibility must be provided to the Instructor before accommodations can be extended.

Class Schedule is subject to change. Please refer to the CMT for updates.

Date:	Class:	Lab:	For Homework:
Week 1: Jan 24	Introductions Course Outline Expectancies	Lynda Rescue Time - https://www. rescuetime.com/ http://selfcontrolapp. com/	Download and install Xcode 4 Download oF - run oF Test files Join the CMT Read: Rushkoff's *Program or Be Programmed* - Ch. 1 (CMT) Programming, Creative Boost (RR) PI - Ch. 2
Week 2: Jan 31 Feb 4- last day to add	Programming Basics	PP: Exercise 1	Project 1 Read: LP - Ch. 1-3 Git Hub vs Linked In OF Tutorials - Intro (RR)
Week 3: Feb 7	Loops, Control, Interaction Git Hub	PP: Exercise 2	Project 1 Read: oF Tutorials Ch 1 (RR) LP - Ch. 4-7
Week 4: Feb 14	Random Motion	Studio Time	Project 1 Read: LP -Ch. 13-17 Nature of Code - Intro (RR)
Week 5: Feb 21	**Project 1 DUE** Images	PP: Exercise 3	Read: LP - Ch. 8-11; 14 oF - Tutorials/First Steps Particle System (RR)
Week 6: Feb 28	Pixels - breaking apart images	PP: Exercise 3	Read: IP - Ch. 8-9
Week 7: Mar 7	Video OOP	PP: Exercise 4a	Read: IP - Ch. 5 LP - Ch. 22
March 11-15	No Class Spring Break	Spring Break	Proposal for Project 2 Read: oF Wiki - OOPS! - OOP +

Date:	Class Discussion:	Class Workshop:	For Homework:
Week 8: Mar 21	OOP	Exercise 4a/b	Proposal for Project 2
Week 9: Mar 28	**Project 2: round table proposals** Strings & Reading files	Exercise 5	Self Assessment Read: LP - Ch. 12, 18, 19 IP - Ch. 12, 14
Week 10: Apr 4 April 8th - last day to withdrawal	**Self Assessment Due** Round table for Project 3 Addon's Libraries	Exercise 6	Read: http://vimeo.com/34092591
Week 11: Apr 11	JSon	Stuio Time	Prototype for Project 3
Week 12: Apr 18th	OpenCV	Studio Time	Project 3
Week 13: Apr 25	Studio Time	Studio Time	Project 3
Week 14: May 2 Last Day of class	**Project 3: Due - Group Critique**		
Finals Week: May 9th			Have a great summer!

Paired Programming Exercises:

Exercise 1: Crop a section of Kandinsky's Composition 8 (CMT). First select an interesting/ambitious crop, then load it into a program such as Photoshop or Illustrator to read the color and coordinate data. Use integer values for coordinates and only use the following functions for drawing: line(), triangle(), quad(), rect(), ellipse(), arc(), and beginShape(), endShape(), vertex().

Exercise 2: Write a program that creates a monster. Animate the monster using variables and mouse interaction. Use different control constructs to animate the character for a set duration, or state.

Exercise 3: Use what you've learned about motion and images to load images into a kinetic collage, that changes upon input. Use only your own images or images that you have permission to use. Flickr is a good source if you don't want to use your own photos: http://www.flickr.com/commons/

Exercise 4a: Take monsters made from others in the class (See exercise 2). Create a classes for each. Then create a monster army, with each monster being able to be drawn uniquely, based on the call to the class. You must have at least 3 methods. Create a program that draws your monster army to the screen at once. You may even make classes out of individual parts and swap them among monsters (ex, hairy arm, fleshy arm, buffer arm, robotic arm, etc).

Exercise 4b: Take your monsters and give them individual behaviors. Are certain ones angry or happy? Weak or strong? Playful or humourous? What happens when certain monsters come in contact with other monsters? How do they interact? What happens over time? Do they age? Get fat? Intergrate time, motion, Perlin noise, control structures, etc.

Exercise 5: Strings

Exercise 6: XML, Yahoo, Twitter- Libraries & API

Project 1: Adjective - 3wks

Pick an adjective and make a program that illustrates that adjective interactively. You may not use images. Only drawing functions, meaning mostly stuff found in the 'graphics' section.
The focus of this project is procedural intensity. How expressive can you be using only lines, shapes, colors?

Ingredients: Graphics, Control Structures, Interaction, Custom functions, Random

Project 2: Choice - 6wks

Option 1- Persuasion: Choose a current event, news item, or issue that you feel strongly about and make a piece that addressees it. Think of it as an interactive, algorithmic documentary. The emphasis of this project is using code to be persuasive. How can you convey a point of view in using procedural generation and interactivity?

Ingredients: Control structures, interaction, addon's, libraries, advance topics

Option 2 - High Score: Design and program a game. This project focuses on motion and interaction. To place the focus on those two components exclusively, the visual elements will be minimal. The visual components of the game are restricted to ~6 basic shapes on the screen at a time, but you may use any motion or interaction technique that you can imagine. You many use only lines, circles, and rectangles, but you may also use simple typography to keep score or show other basic data. These restrictions have been defined to minimize the complexity of the project, to encourage you to be creative within contraints, and to place the emphasis on the qualities of the interaction. After you have completed the game as geometry, you may choose to "skin" the game.