Computational Thinking Through Computing and Music

Workshop on Interdisciplinary Teaching and Learning

Sponsored by the National Science Foundation
Division of Undergraduate Education Award No. 1118435

Transforming Undergraduate Education in STEM Program
Science, Technology, Engineering and Mathematics

January 15-16, 2015

UMass Lowell Inn and Conference Center
Lowell, Massachusetts

Workshop Leaders

Jesse M. Heines, Ed.D., Prof. of Computer Science, UMass Lowell
Gena R. Greher, Ed.D., Prof. of Music Education, UMass Lowell
S. Alex Ruthmann, Ph.D., Assoc. Prof. of Music Education and Music Technology, New York University

www.performamatics.org

This project is supported by Award No. 1118435 from the National Science Foundation (NSF) Division of Undergraduate Education (DUE). It falls under the TUES program: Transforming Undergraduate Education in STEM (Science, Technology, Engineering, and Mathematics). Any opinions, findings, conclusions, or recommendations expressed in our materials are solely those of the authors and do not necessarily reflect the views of the National Science Foundation.

Notebook Contents

Workshop Information

About This Project

Resources and Publications on Performamatics

Selected Book Excerpts

Selected Reprints

Sound Thinking Course Materials

Interdisciplinary Grading Techniques

List of Workshop Attendees

as of January 3, 2015

(in no particular order)

(1) Applicant Name: Rajeev Agrawal
Institution: North Carolina A & T State University
Department: Computer Systems Technology
Email Address: ragrawal@ncat.edu
Work Phone: (336) 285-3137
Cell Phone: (none entered)
Website URL: (none entered)
Courses Taught: Intro to Programming, Storage Technology, Network Security

(2) Colleague Name: Sambit Bhattacharya
Institution: Fayetteville State University
Department: Mathematics and Computer Science
Email Address: sbhattac@uncfsu.edu
Work Phone: (910) 672-1156
Cell Phone: (none entered)
Website URL: (none entered)
Courses Taught: Intro to Programming, Robotics, Mobile App Development

(3) Applicant Name: Katherine Martinenza
Institution: University of Delaware
Department: Music
Email Address: kamart@udel.edu
Work Phone: (302) 743-3491
Cell Phone: (none entered)
Website URL: (none entered)
Courses Taught: I teach Music in Early Childhood
I am a teaching assistant for:

- Freshman Seminar
- Teaching Elementary General Music
- Teaching Secondary General Music
- The Beat Goes On

(4) Colleague Name: Suzanne Burton
Institution: University of Delaware
Department: Music
Email Address: slburton@udel.edu
Work Phone: (610) 563-1067
Cell Phone: (none entered)
Website URL: (none entered)
Courses Taught: Freshman Seminar
Teaching Elementary General Music
Teaching Secondary General Music
The Beat Goes On
Curriculum
Philosophy of Music Education
Psychology of Music Education
Research Methods

(5) Applicant Name: Scott Byerly
Institution: West Chester Rustin High School
Department: Mathematics
Email Address: sbyerly@wcasd.net
Work Phone: (484) 266-4313
Cell Phone: (none entered)
Website URL: (none entered)
Courses Taught: AP Computer Science
Honors Calculus
Academic Algebra II

(6) Colleague Name: Michael Shoremount
Institution: West Chester Rustin High School
Department: Music
Email Address: mshoremount@wcasd.net
Work Phone: (484) 266-4313
Cell Phone: (none entered)
Website URL: (none entered)
Courses Taught: Concert Band
Honors Wind Ensemble
Jazz Lab Band
Jazz Honors Band

(7) Applicant Name: Kerri Epps
Institution: Richmond Street School
Department: Music
Email Address: kepps@esusd.k12.ca.us
Work Phone: (310) 606-6831
Cell Phone: (310) 968-5614
Website URL: (none entered)
Courses Taught: String Orchestra, General Music

(8) Colleague Name: Danielle Flynn
Institution: Richmond Street School
Department: STEM Teacher
Email Address: dflynn@esusd.k12.ca.us
Work Phone: (310) 606-6831
Cell Phone: (none entered)
Website URL: (none entered)
Courses Taught: STEM Teacher 4th/5th grade

(9) Applicant Name: Jean French
Institution: Coastal Carolina University
Department: Computer Science & Information Systems
Email Address: jennis@coastal.edu
Work Phone: (843) 234-3430
Cell Phone: (843) 455-7373
Website URL: (none entered)
Courses Taught: Multimedia, database, web

(10) Colleague Name: Dan Ennis
Institution: Coastal Carolina University
Department: College of Humanities and Fine Arts
Email Address: dennis@coastal.edu
Work Phone: (843) 349-2746
Cell Phone: (843) 602-0016
Website URL: (none entered)
Courses Taught: Dean of College of Humanities and Fine Arts (including Departments of Music, Visual Arts, Theater, Philosophy, History, English, and Communication). Courses taught: World Drama (English), World Dramatic Lit (Theater)

(11) Applicant Name: Brittney Kerby
Institution: El Segundo Middle School
Department: Music
Email Address: bkerby@esusd.k12.ca.us
Work Phone: (916) 390-6006
Cell Phone: (916) 390-6006
Website URL: (none entered)
Courses Taught: 6th-8th grade Music technology, choir, and musical theater

(12) Colleague Name: Cathie Haynes
Institution: El Segundo Middle School
Department: Science
Email Address: chaynes@esusd.k12.ca.us
Work Phone: (310) 615-2690
Cell Phone: (none entered)
Website URL: (none entered)
Courses Taught: 7th and 8th Grade Science and Design

(13) Applicant Name: Aaron Koehl
Institution: Christopher Newport University
Department: Computer and Information Science
Email Address: aaron.koehl@cnu.edu
Work Phone: (757) 594-7911
Cell Phone: (757) 871-3194
Website URL: http://www.aaronkoehl.com
Courses Taught: Algorithms, Senior Sem./Capstone, Systems Analysis and Design, Databases, Programming Languages, Parallel Processing.
Currently serving as Director of Information Systems and Information Science.

(14) Colleague Name: Kelly Rossum
Institution: Christopher Newport University
Department: Music
Email Address: kelly.rossum@cnu.edu
Work Phone: (757) 594-8812
Cell Phone: (917) 291-8139
Website URL: http://krossum.com
Courses Taught: Applied Trumpet, Jazz Ensemble, Jazz Combo, Jazz History and Literature, Jazz Arranging, Jazz Pedagogy Techniques, Brass Masterclass.
Currently serving as Director of Jazz Studies.

(15) Applicant Name: James Robert Mobley
Institution: Brownstown Middle School
Department: Music
Email Address: mobleyj@wbsdweb.com
Work Phone: (734) 783-3400
Cell Phone: (313) 550-6802
Website URL: http://woodhavenbms.sharpschool.net/select_a_teacher_website/mobley__j
Courses Taught: 5th Grade Band, 6th Grade Band, 7th Grade Band, Beginning Band, Jazz Band

(16) Colleague Name: Shannon McNamara
Institution: Brownstown Middle School
Department: Science
Email Address: mcnamas@wbsdweb.com
Work Phone: (734) 783-3400
Cell Phone: (734) 344-8479
Website URL: http://woodhavenbms.sharpschool.net/select_a_teacher_website/mc_namara__s
Courses Taught: 7th Grade Science

(17) Applicant Name: Andrew Clark
Institution: Brownstown Middle School
Department: Principal/Science Teacher
Email Address: ClarkA@wbsdweb.com
Work Phone: (734) 783-3400
Cell Phone: (734) 915-8076
Website URL: http://Woodhaven.k12.mi.us
Courses Taught: I oversee all math, science and performing arts classes

(18) Applicant Name: Ann Moskol
Institution: Rhode Island College
Department: Mathematics & Computer Science
Email Address: amoskol@ric.edu
Work Phone: (401) 456-9761
Cell Phone: (401) 744-1929
Website URL: (none entered)
Courses Taught: Data Structures, Computer Science 1, Introduction to Computers

(19) Colleague Name: Janice Kowalczyk
Institution: All Saints Academy
Department: Computer Education Program
Email Address: jckowalczyk@gmail.com
Work Phone: (401) 848-4300
Cell Phone: (401) 935-5419
Website URL: http://allsaintsacademy.org/
Courses Taught: Computer/Technical Education courses

(20) Applicant Name: Joseph Plazak
Institution: Illinois Wesleyan University
Department: School of Music
Email Address: jplazak@iwu.edu
Work Phone: (309) 556-3280
Cell Phone: (847) 922-9605
Website URL: (none entered)
Courses Taught: Music Theory I-IV; Ear-Training I-IV; Electroacoustic Music; Music Technology; Counterpoint; Advanced Analysis; Intro to Music Cognition

(21) Colleague Name: Mark Liffiton
Institution: Illinois Wesleyan University
Department: Computer Science
Email Address: mliffito@iwu.edu
Work Phone: (309) 556-3535
Cell Phone: (none entered)
Website URL: (none entered)
Courses Taught: Physical Computing; Computer Science I; Computer Architecture and Organization; Computer Networks; Algorithm Design and Analysis; Operating Systems

(22) Applicant Name: Jeffrey Popyack
Institution: Drexel University
Department: Computer Science
Email Address: jpopyack@cs.drexel.edu
Work Phone: (215) 895-1846
Cell Phone: (484) 883-7452
Website URL: http://www.cs.drexel.edu/~jpopyack
Courses Taught: Introduction to Computer Science, Computer Programming I-II, Artificial Intelligence, Evolutionary Computing

(23) Colleague Name: Luke Abruzzo
Institution: Drexel University
Department: Music
Email Address: laa24@drexel.edu
Work Phone: (215) 571-3528
Cell Phone: (610) 405-1140
Website URL: http://www.drexel.edu/westphal/minors/MUSC/
Courses Taught: Music Theory I-II, Composition, Digital Music Composition

(24) Applicant Name: Daniel Walzer
Institution: UMass Lowell
Department: Music
Email Address: Daniel_Walzer@uml.edu
Work Phone: (978) 934-3881
Cell Phone: (none entered)
Website URL: (none entered)
Courses Taught: Music

(25) Applicant Name: James Caristi
Institution: Valparaiso University
Department: Computing and Information Sciences
Email Address: james.caristi@valpo.edu
Work Phone: (219) 464-5342
Cell Phone: (219) 241-5531
Website URL: http://faculty.valpo.edu/jcaristi
Courses Taught: computer simulation, CS1, CS2, assembly, DBMS, mobile computing, software design

(26) Applicant Name: Kelsey Smith
Institution: Los Feliz Charter School for the Arts
Department: Music
Email Address: ksmith@losfelizarts.org
Work Phone: (310) 924-5584
Cell Phone: 3109245584
Website URL: http://www.losfelizarts.org
Courses Taught: Music TK-6th Grade

(27) Colleague Name: Zelina Munoz
Institution: Los Feliz Charter School for the Arts
Department: General Elementary
Email Address: zmunoz@losfelizarts.org
Work Phone: (310) 924-5584
Cell Phone: (none entered)
Website URL: http://www.losfelizarts.org
Courses Taught: All General Subjects Second Grade

(28) Applicant Name: Brandon Petcaugh
Institution: New Foundations Charter School
Department: K-12 STEM Specialist
Email Address: bpetcaugh@nfcs.k12.pa.us
Work Phone: (215) 344-6410
Cell Phone: (215) 436-5720
Website URL: http://www.nfcsonline.org
Courses Taught: I currently assist in STEM instruction from grades K-12 in two separate buildings. I lead several Robotics programs and teach an "Intro. to Computer Programming" elective to 11-12 graders.

(29) Colleague Name: Rachel Schwartz
Institution: New Foundations Charter School
Department: Math Specialist
Email Address: rschwartz@nfcs.k12.pa.us
Work Phone: (215) 624-8100
Cell Phone: (none entered)
Website URL: http://www.nfcsonline.org
Courses Taught: Responsible for all of our 6-8 grade enrichment programs and assist in developing math skills for underachievers.

(30) Applicant Name: Aaron Thomas Hill, Sr.
Institution: Woodward Academy
Department: Instrumental Music
Email Address: aaron.hill@woodward.edu
Work Phone: (404) 765-1487
Cell Phone: (757) 218-4094
Website URL: (none entered)
Courses Taught: Band

(31) Colleague Name: Alisha Louise Stratton
Institution: Woodward Academy
Department: Foreign Language
Email Address: alisha.stratton@woodward.edu
Work Phone: (404) 765-1487
Cell Phone: (404) 931-4159
Website URL: (none entered)
Courses Taught: Spanish

(32) Applicant Name: Carla Gioia Fernandez
Institution: Los Feliz Charter School for the Arts
Department: General Ed Teacher 4th grade
Email Address: cfernandez@losfelizarts.org
Work Phone: (323) 539-2810
Cell Phone: (818) 397-6686
Website URL: http://www.losfelizarts.org
Courses Taught: Language Arts, Math, Science, Social Studies, Integration of Music, Art, and Dance

(33) Applicant Name: Jeff Tillinghast
Institution: University Prep
Department: Academic Technology
Email Address: jtillinghast@universityprep.org
Work Phone: (206) 832-1160
Cell Phone: (206) 910-4548
Website URL: http://jeff.thetillinghasts.com
Courses Taught: Digital Media, Professional Development (Technology Integration and Curriculum Development for Staff)

(34) Colleague Name: Thane Lewis
Institution: University Prep
Department: Music
Email Address: tlewis@universityprep.org
Work Phone: (206) 525-2714
Cell Phone: (none entered)
Website URL: (none entered)
Courses Taught: Orchestra, Songwriting

(35) Applicant Name: Steve Venz <svenz@ocde.us>
Institution: Orange County Department of Education
Department: Visual & Performing Arts
Email Address: svenz@ocde.us
Work Phone: (818) 445-8993
Cell Phone: (818) 445-8993
Website URL: (none entered)
Courses Taught: I provide professional development to instructors from the 28 school districts in Orange County, California.

I am teaching the following classes for the Longy School of Music of Bard College's Master of Arts in Teaching Program:

- ED 513 Historical & Social Contexts of Teaching and Learning
- ED/MU515 Teaching as Clinical Practice #2
- ED/MU535 Teaching as Clinical Practice #3

WORKSHOP CO-LEADERS

Name: Jesse M. Heines
Institution: University of Massachusetts Lowell, Lowell, MA
Department: Computer Science
Email Address: heines@cs.uml.edu

Name: Gena R. Greher
Institution: University of Massachusetts Lowell, Lowell, MA
Department: Music Education
Email Address: gena_greher@uml.edu

Name: S. Alex Ruthmann
Institution: New York University, Steinhardt School of Culture, Education, and Human Development, New York, NY
Department: Music and Performing Arts Professions
Email Address: alex.ruthmann@nyu.edu

PROJECT EVALUATOR

Name: Scott D. Lipscomb
Institution: Univ. of Minnesota School of Music, Minneapolis, MN
Department: Music Education & Music Therapy
Email Address: lipscomb@umn.edu

ADDITIONAL UMASS LOWELL FACULTY RESEARCHERS

Name: Fred Martin
Institution: University of Massachusetts Lowell, Lowell, MA
Department: Computer Science
Email Address: fredm@cs.uml.edu

Name: Sarah Kuhn
Institution: University of Massachusetts Lowell, Lowell, MA
Department: Psychology
Email Address: sarah_kuhn@uml.edu

Pre-Workshop Questionnaire
31 responses

View all responses Publish analytics

Summary

Please Enter Your Name

Jeffrey L. Popyack

Rajeev Agrawal

Kelly Rossum

Scott Byerly

Brittney

Michael Shoremount

James Mobley

Jean French

Sambit Bhattacharya

Kelsey Smith

Suzanne Burton

Rachel Schwartz

Joseph Plazak

Cathie Haynes

Luke Abruzzo

Kerri Epps

James Caristi

Aaron Koehl

Danielle

Andrew Clark

janice kowalczyk

Katherine Martinenza

Brandon Petcaugh

Carla Gioia Fernandez

Zelina Munoz-Friedman

Dan Ennis

Steve Venz

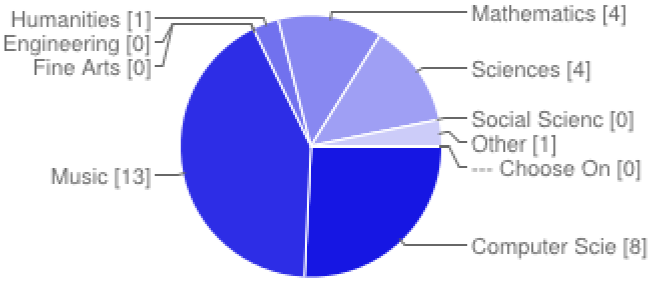
Mark Liffiton

Jeff Tillinghast

Shannon McNamara

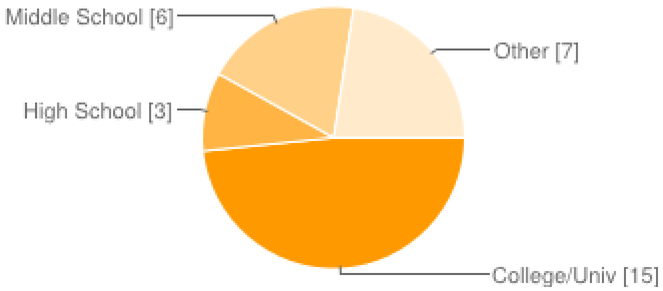
Daniel Walzer

Your Area of Expertise



--- Choose One ---	**0**	0%
Computer Science	**8**	26%
Music	**13**	42%
Engineering	**0**	0%
Fine Arts	**0**	0%
Humanities	**1**	3%
Mathematics	**4**	13%
Sciences	**4**	13%
Social Sciences	**0**	0%
Other	**1**	3%

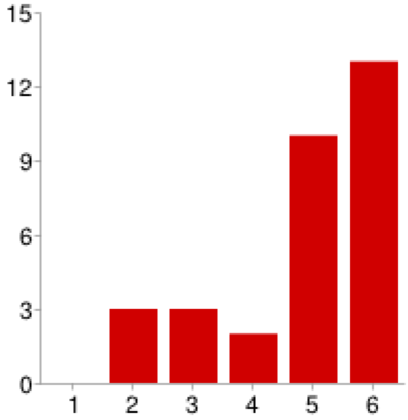
Which of the following best describes your level of teaching.



College/University	**15**	48%
High School	**3**	10%
Middle School	**6**	19%
Other	**7**	23%

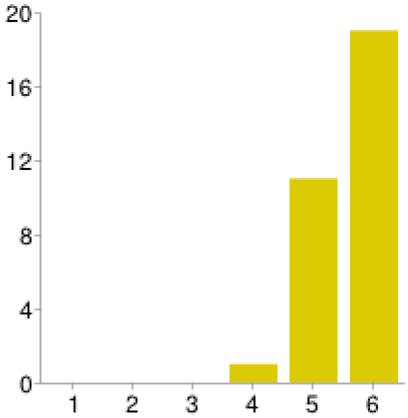
Expectations for This Workshop

(1) I expect to learn about creating music with computers.



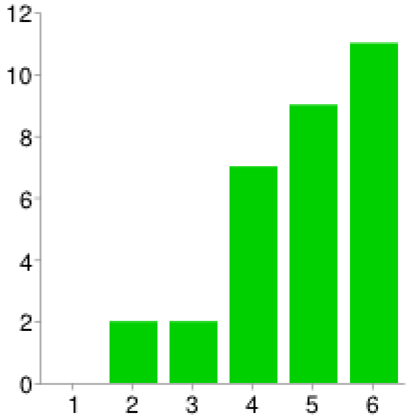
1	**0**	0%	<<< Strongly Disagree
2	**3**	10%	
3	**3**	10%	
4	**2**	6%	
5	**10**	32%	
6	**13**	42%	<<< Strongly Agree

(2) I expect to learn how to plan interdisciplinary projects in music and computing.



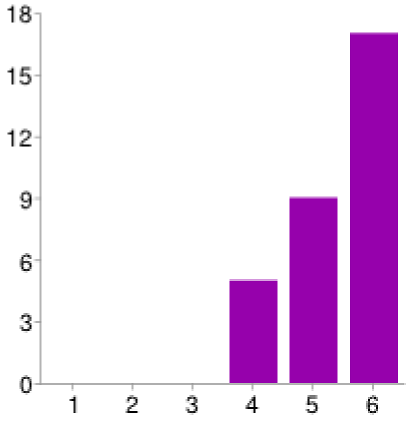
1	**0**	0%
2	**0**	0%
3	**0**	0%
4	**1**	3%
5	**11**	35%
6	**19**	61%

(2a) I have a significant amount of direct, personal experience working with colleagues and peers across traditional disciplinary boundaries.



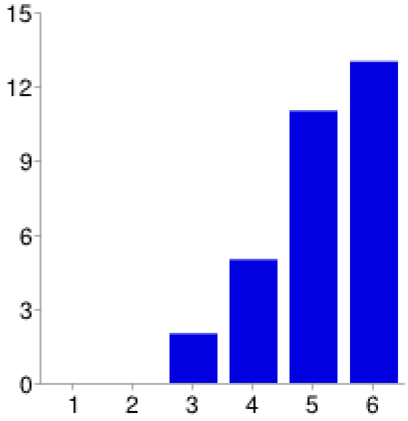
1	**0**	0%
2	**2**	6%
3	**2**	6%
4	**7**	23%
5	**9**	29%
6	**11**	35%

(3) I expect to learn about computational thinking as it relates to computing and music.



1	**0**	0%
2	**0**	0%
3	**0**	0%
4	**5**	16%
5	**9**	29%
6	**17**	55%

(4) I expect to have time to plan and collaborate on an interdisciplinary teaching project.



1	**0**	0%
2	**0**	0%
3	**2**	6%
4	**5**	16%
5	**11**	35%
6	**13**	42%

(5) Please explain any additional reasons why you are participating in this workshop:

On the personal side, higher education is becoming more and more automated via technology. Programming skills seem to be essential for managing the larger workloads that academics are likely to be asked to assume. Within the teaching realm, I know of very few resources that encourage musicians and music educators (beyond composers) to learn about computer coding and automation.

Creating more means of connection-making in inter-disciplinary learning

I am looking forward to learning more about this area of music. We are moving to a new format at our middle school next year, and we have opportunities to create new courses and team teach. This is a wonderful start to developing a middle school level course combining science and music for our district.

I would like to find enrichment opportunities for middle school math students.

I am interested in participating in this workshop to bring back ideas for our general education teachers to develop units using Music.

it just sounds interesting!

I have been a constructivist educator for over 10 years. I believe that music is an integral in teaching children in elementary school. I've collaborated with many teachers over the years to help students learn. I also believe that each student is an individual learner and I try to expose students to different ways of learning. I think this workshop is just another way for me to learn new ways to bring learning to my classroom.

Computational Thinking can provide teachers a way to implement an excellent approach to teaching that is aligned to STEAM. I would like to learn more about the project in order to figure out ways to provide this to teachers in Southern California (Orange County).

I am always looking for new ways to engage students in personal ways in STEM related activities and I feel strongly that Scratch is a good vehicle for this. I think this workshop will enhance my ability to help connect my student's love of music with their developing Scratch expertise. Looking forward to it.

I am a Co-PI on a grant where we are working on including computational approaches which are culturally relevant and interesting. Courses at different undergraduate levels will include this material in future offerings. Music composition using programming is at the top of our list. I hope to get training / materials from this workshop to fullfill some of the grant objectives.

This workshop was strongly recommended by my mentor, Dr. Frank Heuser.

Utilizing this knowledge in STEAM program back at home site.

My colleague, Joe Plazak, and I are considering creating a joint Music/CS course to teach at our institution. It may serve as an introduction to computer science with Music as a lens, or it might fit into our general education curriculum. This workshop can serve as an excellent launching point for the design and planning of the course.

Always looking to connect core subject areas to the arts. To many times arts are being removed from high schools!

I have lifelong interests in music and computers (well, the last 40 years, anyway). I have used some music-related exercises in my Introduction to Computer Science class, and I am interested in finding more significant and interesting experiences. Also, we always have CS students in our program with substantial backgrounds in music that would be interested in applying computing to music.

We are starting a major initiative in digital humanities and I look forward to incorporating musical composition into the program.

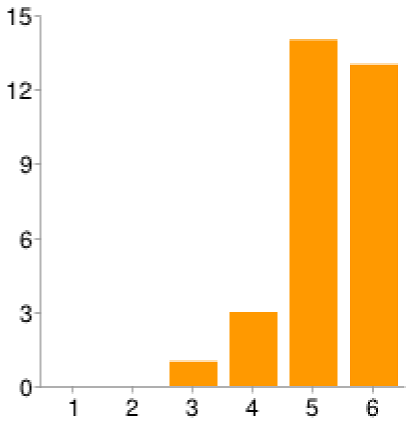
I hope to learn about computer science and the thought process of students/practitioners in this field.

Because I LOVE this topic!

How to incorporate performing art with STEM style classes.

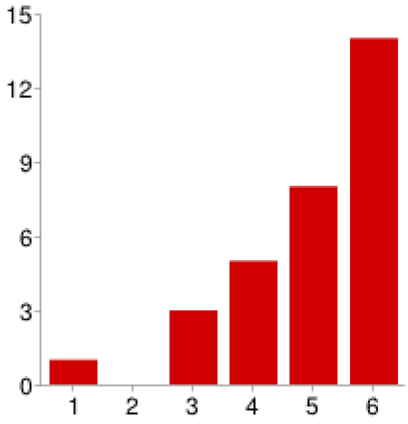
About Yourself

(6) I like to tinker and create new things.



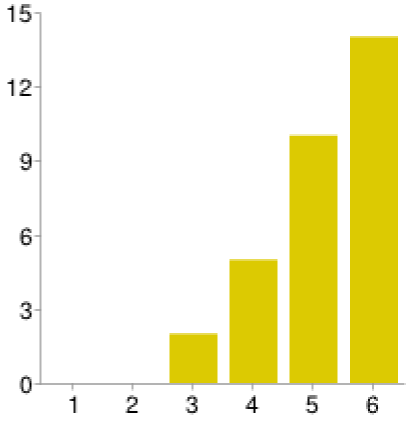
1	**0**	0%
2	**0**	0%
3	**1**	3%
4	**3**	10%
5	**14**	45%
6	**13**	42%

(7) I have personal experience as a member of an interdisciplinary team.



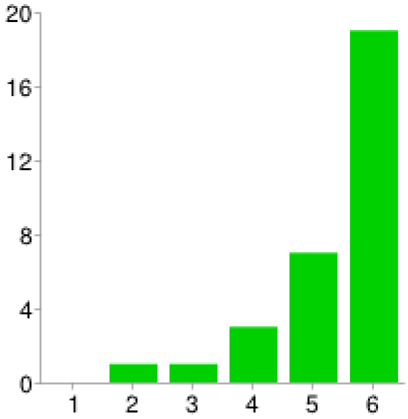
1	**1**	3%
2	**0**	0%
3	**3**	10%
4	**5**	16%
5	**8**	26%
6	**14**	45%

(8) I enjoy interdisciplinary work.



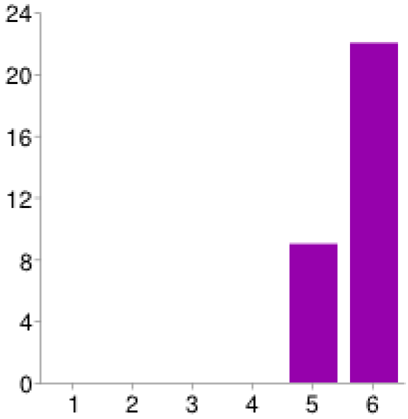
1	**0**	0%
2	**0**	0%
3	**2**	6%
4	**5**	16%
5	**10**	32%
6	**14**	45%

(9) I am not afraid to try new things.



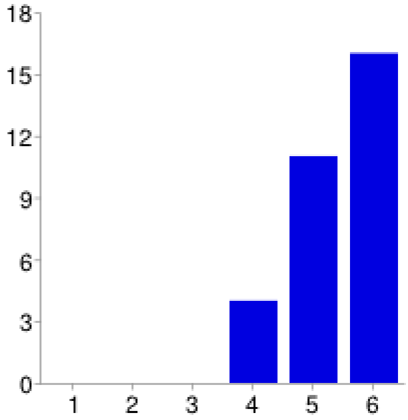
1	**0**	0%
2	**1**	3%
3	**1**	3%
4	**3**	10%
5	**7**	23%
6	**19**	61%

(10) I believe interdisciplinary work is good for my students.



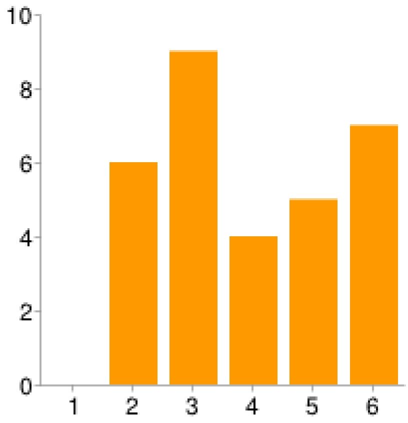
1	**0**	0%
2	**0**	0%
3	**0**	0%
4	**0**	0%
5	**9**	29%
6	**22**	71%

(11) I like to collaborate with others.



1	**0**	0%
2	**0**	0%
3	**0**	0%
4	**4**	13%
5	**11**	35%
6	**16**	52%

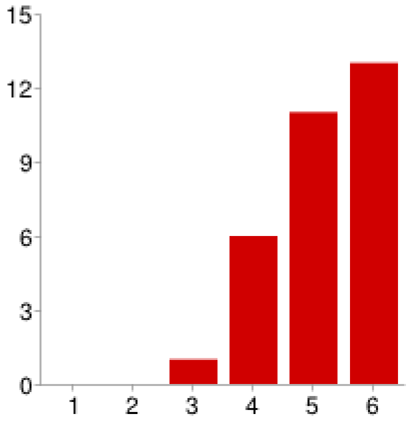
(12) I like to work alone.



1	**0**	0%
2	**6**	19%
3	**9**	29%
4	**4**	13%
5	**5**	16%
6	**7**	23%

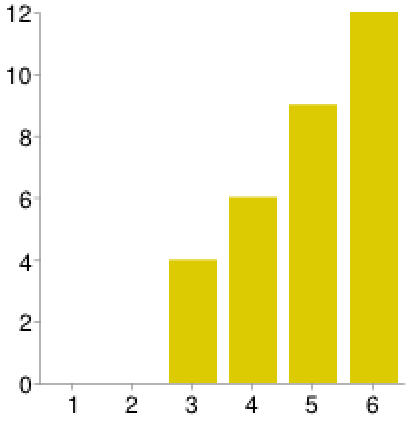
Your Opinions on Interdisciplinary Teaching in Computing+Music

(13) I can successfully implement an interdisciplinary project in one of my courses.



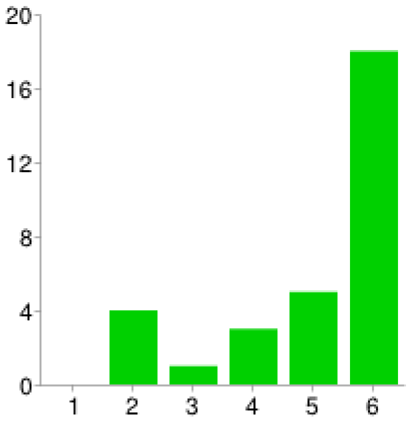
1	**0**	0%
2	**0**	0%
3	**1**	3%
4	**6**	19%
5	**11**	35%
6	**13**	42%

(14) I can successfully implement an interdisciplinary course at my institution.



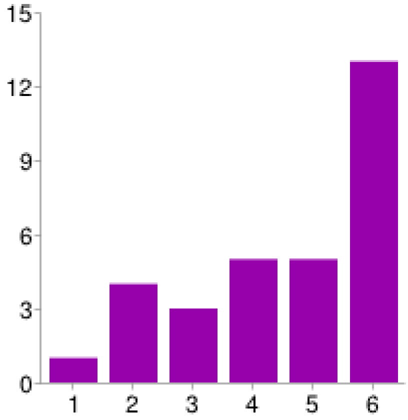
1	**0**	0%
2	**0**	0%
3	**4**	13%
4	**6**	19%
5	**9**	29%
6	**12**	39%

(15) I take advantage of my students' interests in music in my teaching.



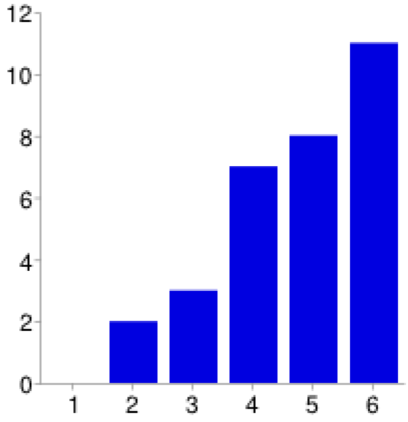
1	**0**	0%
2	**4**	13%
3	**1**	3%
4	**3**	10%
5	**5**	16%
6	**18**	58%

(16) I take advantage of my students' interests in computing in my teaching.



1	**1**	3%
2	**4**	13%
3	**3**	10%
4	**5**	16%
5	**5**	16%
6	**13**	42%

(17) I am comfortable teaching and evaluating interdisciplinary student projects.



1	**0**	0%
2	**2**	6%
3	**3**	10%
4	**7**	23%
5	**8**	26%
6	**11**	35%

(18) Please provide any additional thoughts you might have regarding interdisciplinary teaching in computing+music.

I don't have any real direct experience with it. In all of my teaching of computer science, I haven't yet integrated music, but I am looking forward to finding out how I might.

The main difficulty will be getting music professors and music majors to invest time. Their schedules are very full. Music minors or students just interested in music will be easily attracted.

It remains to be seen whether I will succeed in getting a music professor to collaborate with me for the course(s).

I have anxiety, given that this would be a step out of my normal teaching box of comfort. But at the same time I am excited about the opportunity to pursue this new avenue for reaching students through music and science.

Quality interdisciplinary instruction requires time to plan with partnering teacher(s) in order to guarantee rigor of instruction in all subject matter being taught as well as ensuring students understand connections, allowing application of understanding.

I would like to learn more about implementing Music through my core curriculum.

This is new for me. I teach musically interested students who have backgrounds in computer science or engineering. They are often excellent musical people and if I could understand their respective field(s) perhaps I could relate music to them in a way that speaks to their computer minds. On the other hand, for more than half of a century, cutting edge composers were more than just musical people. They were mathematicians, physicists, and electrical engineers. In this day of computer musical production, I think it behooves all musicians/composers to understand software beyond consumer usability.

My experiences in interdisciplinary curriculum development suggest that it usually requires a lot longer than expected or desired for participants to get on the same page. It is amazing how much one takes for granted, not only in background experience, but in overall approach and attitude. And so I expect the same to be true of working with music faculty. And yet, I am highly optimistic about the end result here.

Computers can do more than compose! I hope that we'll have a bit of time to discuss how music educators, performers, and researchers might also benefit from programming knowledge.

I have always used interdisciplinary teaching in my classrooms. Project-based learning is a good fit for this model.

We recently received laptops as a part of our student resources. I want to be able to make use of these laptops as a part of our standard curriculum in music, especially with my upper graders who are eager to integrate computation and music.

THANK YOU - From Alex, Gena, and Jesse and the Performamatics Team

TUES Performamatics Interdisciplinary Workshop

Thursday and Friday, January 15-16, 2015

UMass Lowell Inn & Conference Center ◆ Lower Locks Room
50 Warren Street ◆ Lowell, MA 01852 ◆ 1-978-934-6920

Don't forget to bring your own laptop! Don't forget to load Scratch and Audacity!

Thursday, January 15

8:00 *continental breakfast (provided)*

8:30 welcome and introductions
8:45 discussion of interdisciplinary teaching goals and approaches
9:15 introduction to collaboration-building exercise: "found instruments"
9:45 building an instrument from provided materials and creating a score using graphical notation (in interdisciplinary pairs)

10:00 *break while continuing to work on found instruments and creative notations*

10:45 sharing of instruments and notations
11:30 debriefing on these projects and discussion of our evolving models of interdisciplinary collaboration and the modes of interaction that made them work

12:00 *lunch (provided) combined with discussion of barriers to interdisciplinary teaching*

1:00 "reporting out" from lunch groups
1:30 introduction to Scratch and our related Sound Thinking assignments
1:45 improvise and perform a drumbeat and melody with the keyboard (in interdisciplinary pairs), changing sounds, tempo, and volume
2:15 debriefing on these projects and discussion of challenges and insights

2:45 *coffee break*

3:00 introduction to sequencing sounds in Scratch
3:15 pairs sequence a drumbeat and melodic riff in Scratch
3:45 pairs share sequences
4:00 introduction to optimizing sequences and songs
4:15 pairs explore methods of optimizing sequences
4:45 debriefing on these projects and discussion of challenges and insights
5:00 interdisciplinary pairs meet to discuss and outline a potential project to work on at their home institution

5:30 *end of first day's formal workshop activities*

Thursday, January 15 (*continued*)

6:30 *reception (cash bar)*

7:00 *dinner (provided)*

“Homework”

If not completed during day, write out the explicit concept and process connections, learning goals and objectives for a collaborative project between the pairs. What experiences would be core for students?

Friday, January 16

8:00 *continental breakfast (provided)*

8:30 reflection on Thursday’s experiences and discussion of what it would take to bring this type of work into participants’ own institutions
9:00 small group breakouts to discuss how to implement an interdisciplinary project at home institutions
9:30 report back and share ideas

10:00 *coffee break*

10:30 time to for pairs to outline collaborative projects and design and develop interdisciplinary assignments and activities for use at their own institutions

12:00 *lunch (provided) combined with discussion of participants’ own projects*

1:00 “reporting out” from group work over lunch
1:15 pairs share, get feedback on, and try out other group’s interdisciplinary assignments and activities

2:00 *informal coffee break while continuing to work on own interdisciplinary assignments and try out other’s assignments and activities*

2:45 pairs create an action plan for implementing their project
3:15 pairs share action plans with whole group
4:00 workshop wrap-up

4:30 *formal workshop ends*

National Science Foundation
WHERE DISCOVERIES BEGIN

SEARCH
NSF Web Site

Award Abstract #1118435

Computational Thinking through Computing and Music

NSF Org:	DUE Division of Undergraduate Education
Initial Amendment Date:	July 27, 2011
Latest Amendment Date:	July 27, 2011
Award Number:	1118435
Award Instrument:	Standard Grant
Program Manager:	Guy-Alain Amoussou DUE Division of Undergraduate Education EHR Directorate for Education & Human Resources
Start Date:	August 1, 2011
Expires:	July 31, 2014 (Estimated)
Awarded Amount to Date:	$449995
Investigator(s):	Jesse Heines heines@cs.uml.edu (Principal Investigator) Gena Greher (Co-Principal Investigator) S. Alex Ruthmann (Co-Principal Investigator)
Sponsor:	University of Massachusetts Lowell 600 Suffolk Street Lowell, MA 01854 978/934-4723
NSF Program(s):	TUES-Type 2 Project, S-STEM:SCHLR SCI TECH ENG&MATH
Field Application(s):	
Program Reference Code(s):	SMET, 9178
Program Element Code(s):	7511, 1536

ABSTRACT

Computational thinking (CT) is an emerging component of computer science education. A common characteristic of successful efforts to introduce CT is the presence of a context to which students can relate. This project builds upon previous efforts that have shown music to be a context that engages students.

A sample of student activities include writing computer programs to play music, developing web pages that incorporate music, and developing data structures and databases to catalog sounds. Upper level courses in computing and music are synchronized by students working on collaborative projects across the disciplines. An alternative format is to offer a hybrid course co-taught by faculty from both disciplines. Expected outcomes include course materials and approaches for measuring CT gains. Course materials include lecture notes, class activities, code examples and homework assignments.

Professional development workshops provide expertise for faculty to adopt new education approaches and to participate in a community of like-minded educators. Attendees are interdisciplinary two-person teams with expertise in computing as well as music. Three summer workshops are expected to attract one hundred faculty from fifty institutions.

This effort leverages a natural relationship between music and computing to teach CT concepts to undergraduates in all disciplines. Materials are being developed for interdisciplinary general education courses and discipline-specific music and computing courses at more advanced levels.

Please report errors in award information by writing to: awardsearch@nsf.gov.

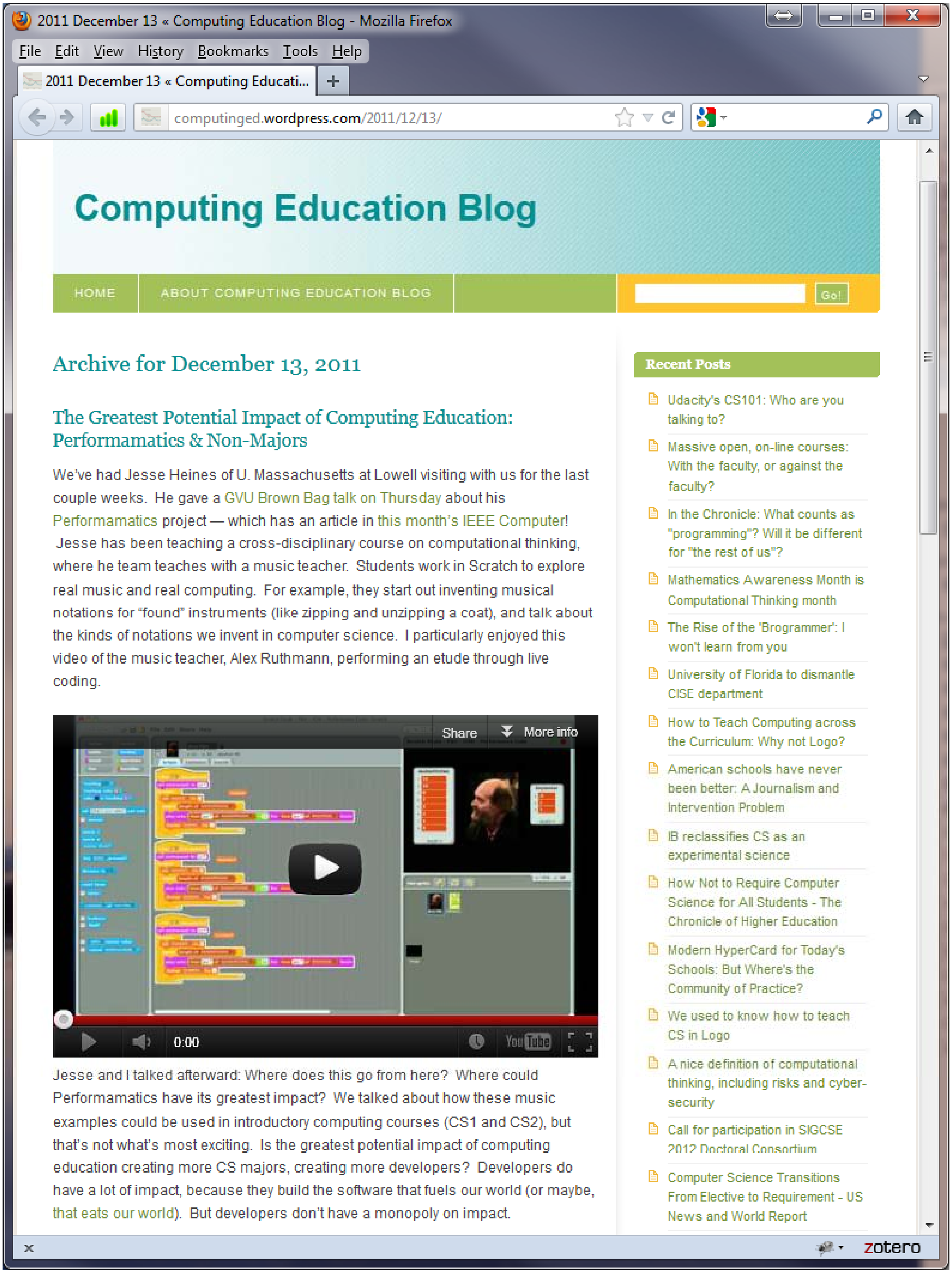
Print this page

Top

Web Policies and Important Links | Privacy | FOIA | Help | Contact NSF | Contact Web Master | SiteMap

The National Science Foundation, 4201 Wilson Boulevard, Arlington, Virginia 22230, USA
Tel: (703) 292-5111, FIRS: (800) 877-8339 | TDD: (800) 281-8749

Last Updated:
April 2, 2007
Text Only


2011 December 13 « Computing Education Blog - Mozilla Firefox
File Edit View History Bookmarks Tools Help
2011 December 13 « Computing Educati...
computinged.wordpress.com/2011/12/13/
HOME
ABOUT COMPUTING EDUCATION BLOG
Go!
Share
More info
0:00
zotero

Computing Education Blog

Archive for December 13, 2011

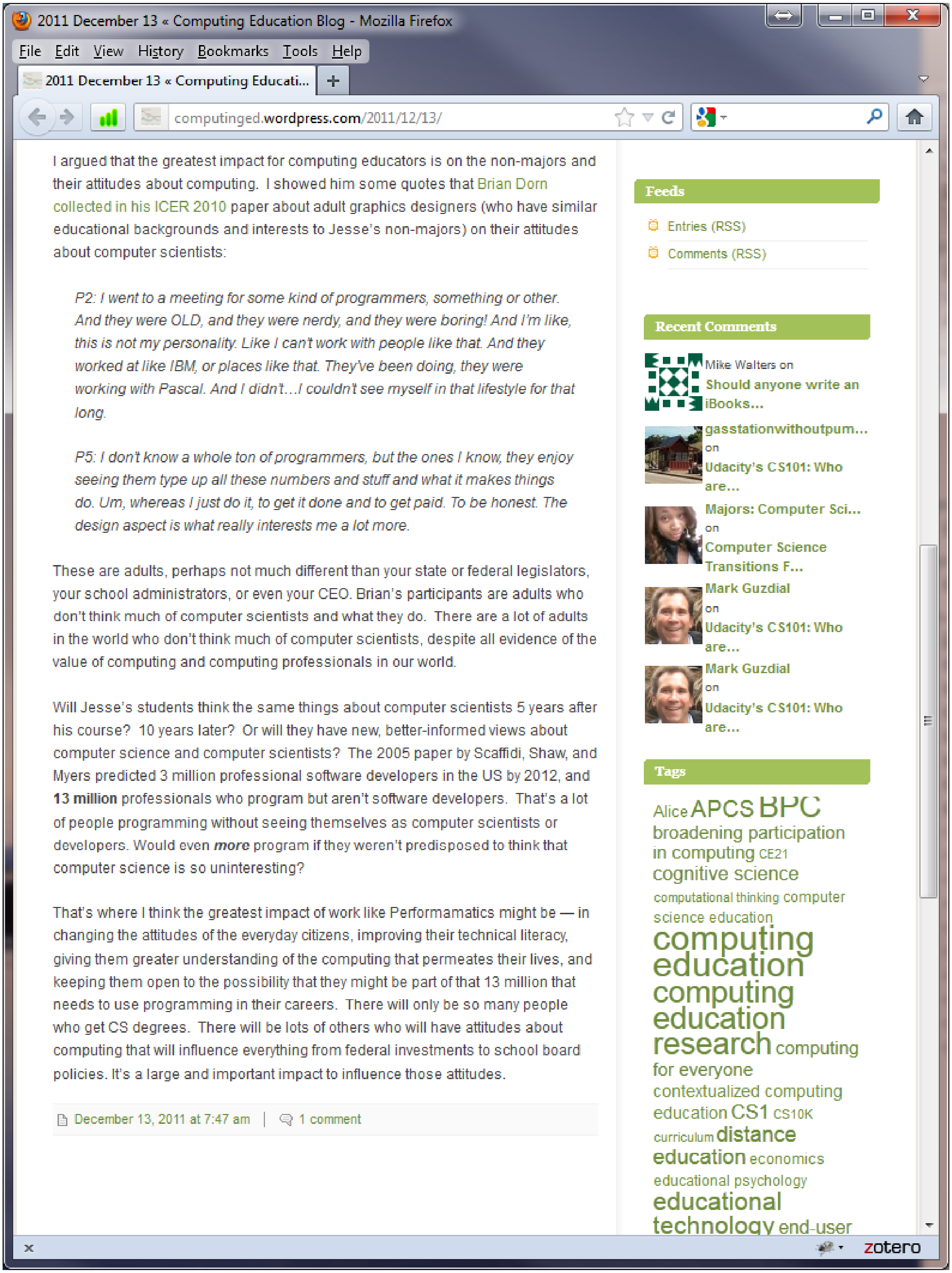
The Greatest Potential Impact of Computing Education: Performamatics & Non-Majors

We've had Jesse Heines of U. Massachusetts at Lowell visiting with us for the last couple weeks. He gave a GVU Brown Bag talk on Thursday about his Performamatics project — which has an article in this month's IEEE Computer! Jesse has been teaching a cross-disciplinary course on computational thinking, where he team teaches with a music teacher. Students work in Scratch to explore real music and real computing. For example, they start out inventing musical notations for "found" instruments (like zipping and unzipping a coat), and talk about the kinds of notations we invent in computer science. I particularly enjoyed this video of the music teacher, Alex Ruthmann, performing an etude through live coding.

Jesse and I talked afterward: Where does this go from here? Where could Performamatics have its greatest impact? We talked about how these music examples could be used in introductory computing courses (CS1 and CS2), but that's not what's most exciting. Is the greatest potential impact of computing education creating more CS majors, creating more developers? Developers do have a lot of impact, because they build the software that fuels our world (or maybe, that eats our world). But developers don't have a monopoly on impact.

Recent Posts

- Udacity's CS101: Who are you talking to?
- Massive open, on-line courses: With the faculty, or against the faculty?
- In the Chronicle: What counts as "programming"? Will it be different for "the rest of us"?
- Mathematics Awareness Month is Computational Thinking month
- The Rise of the 'Brogrammer': I won't learn from you
- University of Florida to dismantle CISE department
- How to Teach Computing across the Curriculum: Why not Logo?
- American schools have never been better: A Journalism and Intervention Problem
- IB reclassifies CS as an experimental science
- How Not to Require Computer Science for All Students - The Chronicle of Higher Education
- Modern HyperCard for Today's Schools: But Where's the Community of Practice?
- We used to know how to teach CS in Logo
- A nice definition of computational thinking, including risks and cyber-security
- Call for participation in SIGCSE 2012 Doctoral Consortium
- Computer Science Transitions From Elective to Requirement - US News and World Report


2011 December 13 « Computing Education Blog - Mozilla Firefox
File Edit View History Bookmarks Tools Help
2011 December 13 « Computing Educati...
computinged.wordpress.com/2011/12/13/
zotero

I argued that the greatest impact for computing educators is on the non-majors and their attitudes about computing. I showed him some quotes that Brian Dorn collected in his ICER 2010 paper about adult graphics designers (who have similar educational backgrounds and interests to Jesse's non-majors) on their attitudes about computer scientists:

> *P2: I went to a meeting for some kind of programmers, something or other. And they were OLD, and they were nerdy, and they were boring! And I'm like, this is not my personality. Like I can't work with people like that. And they worked at like IBM, or places like that. They've been doing, they were working with Pascal. And I didn't…I couldn't see myself in that lifestyle for that long.*
>
> *P5: I don't know a whole ton of programmers, but the ones I know, they enjoy seeing them type up all these numbers and stuff and what it makes things do. Um, whereas I just do it, to get it done and to get paid. To be honest. The design aspect is what really interests me a lot more.*

These are adults, perhaps not much different than your state or federal legislators, your school administrators, or even your CEO. Brian's participants are adults who don't think much of computer scientists and what they do. There are a lot of adults in the world who don't think much of computer scientists, despite all evidence of the value of computing and computing professionals in our world.

Will Jesse's students think the same things about computer scientists 5 years after his course? 10 years later? Or will they have new, better-informed views about computer science and computer scientists? The 2005 paper by Scaffidi, Shaw, and Myers predicted 3 million professional software developers in the US by 2012, and **13 million** professionals who program but aren't software developers. That's a lot of people programming without seeing themselves as computer scientists or developers. Would even ***more*** program if they weren't predisposed to think that computer science is so uninteresting?

That's where I think the greatest impact of work like Performamatics might be — in changing the attitudes of the everyday citizens, improving their technical literacy, giving them greater understanding of the computing that permeates their lives, and keeping them open to the possibility that they might be part of that 13 million that needs to use programming in their careers. There will only be so many people who get CS degrees. There will be lots of others who will have attitudes about computing that will influence everything from federal investments to school board policies. It's a large and important impact to influence those attitudes.

December 13, 2011 at 7:47 am | 1 comment

Feeds

- Entries (RSS)
- Comments (RSS)

Recent Comments

Mike Walters on Should anyone write an iBooks…

gasstationwithoutpum… on Udacity's CS101: Who are…

Majors: Computer Sci… on Computer Science Transitions F…

Mark Guzdial on Udacity's CS101: Who are…

Mark Guzdial on Udacity's CS101: Who are…

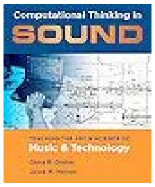
Tags

Alice APCS BPC broadening participation in computing CE21 cognitive science computational thinking computer science education computing education computing education research computing for everyone contextualized computing education CS1 CS10K curriculum distance education economics educational psychology educational technology end-user

University of Massachusetts Lowell
Depts. of Music and Computer Science

Computational Thinking through Computing and Music

an interdisciplinary NSF TUES project



Now Available

Computational Thinking In Sound

by Gena R. Greher and Jesse M. Heines
Oxford University Press, April 2014

To attend our workshop, please complete the ***workshop application form****.*

Participants' comments on our workshops

Our fifth (and potentially our final!) **two-day** interdisciplinary workshop will take place on:

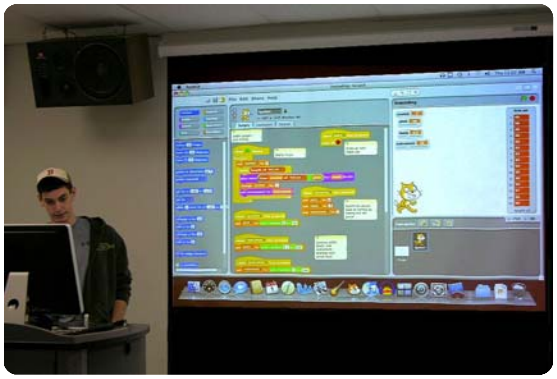
Thursday & Friday, June 19-20, 2014

at New York University
in New York City, New York

Logistics for Workshop Participants

Scenes from the June 21-22, 2012, workshop

Our goal is to develop and disseminate ways to enhance students' grasps of computational thinking by engaging them in fundamental concepts that unite computing and music. Our approach leverages students' near universal interest in music as a context and springboard for engaging in rich computational thinking experiences. Prior work in an NSF CPATH project showed this approach to be effective at creating value in both discipline-specific courses for Computer Science and Music majors, as well as General Education courses for all majors. This project will develop additional activities to deepen students' experiences in computing and music, and explore additional techniques for evaluating learning through those activities. The project will also disseminate our work through workshops for pairs of interdisciplinary faculty at 4- and 2-year colleges.



Our materials teach concepts such as modularization by breaking songs down into their components, looping and subroutines by noting where musical phrases are repeated intact and with small variations (requiring parameters), logic flow by creating musical flowcharts, and algorithms by writing programs that generate music. New materials will explore ways to teach more advanced computing concepts such as threads and synchronization by writing programs that play multiple parts simultaneously and use various Application Programmer Interfaces (APIs), allowing us to combine software platforms into systems that to do more than is possible by one alone.

A major component of this project is the sharing of our techniques and materials through sponsored workshops at conferences and on-site at universities where participants attend as a pair: at least one from Computer Science (or another science or engineering department) and one from Music (or another arts department). This will ensure that collaborations begun in the workshops have a foothold on sustainability when the participants return to their own institutions.

This project is supported by Award No. 1118435 from the National Science Foundation (NSF) Division of Undergraduate Education (DUE). It falls under the TUES program: Transforming Undergraduate Education in STEM (Science, Technology, Engineering, and Mathematics). Any opinions, findings, conclusions, or recommendations expressed in our materials are solely those of the authors and do not necessarily reflect the views of the National Science Foundation.

University of Massachusetts Lowell
Depts. of Music and Computer Science

Computational Thinking through Computing and Music

an interdisciplinary NSF TUES project

Home Workshop About Our Project **About Us** Resources Publications Scratch Laptop Orchestra

Jesse Heines, *Principal Investigator*, is a Professor of Computer Science at UMass Lowell with a strong interest in music and its power to interest students in computing. He teaches courses on graphical user interfaces, web programming, and C++, and co-teaches the Sound Thinking course with Gena and Alex. He was PI on an NSF CPATH award and is currently PI on an NSF TUES award. Gena and Jesse are working on a book on interdisciplinary teaching that is currently under contract with Oxford University Press. To keep his music alive, Jesse sings with the Lowell "Gentlemen Songsters" Chapter of the Barbershop Harmony Society.

Gena Greher, *Co-Principal Investigator*, is a Professor of Music Education at UMass Lowell. Her research focuses on creativity and listening skill development in children and examining the influence of integrating multimedia technology in urban music classrooms, as well as in the music teacher education curriculum and School-University partnerships. Recent projects include: a music technology mentor/partnership with UMass Lowell music education students and two local K-8 schools; *SoundScape*, a technology-infused music intervention program for teenagers with autism spectrum disorders; and *Performamatics*, an NSF-supported project linking computer science to the arts. Before entering the education profession and crossing paths with Jesse and Alex, Gena was a music director in advertising, working for several multinational advertising agencies producing the jingles and underscores for hundreds of commercials.

S. Alex Ruthmann, *Co-Principal Investigator*, is an Associate Professor of Music Education and Music Technology at New York University. Beginning as a middle school music teacher and computational musician, he now teaches undergraduate and graduate courses at the intersection of music education, arts computing, and research. He has published on technology-mediated music learning and teaching, children's musical and compositional processes, and fostering learner agency through music and technology. Alex's research explores social/digital media musicianship and creativity, as well as the development of technologies for music learning, teaching and engagement for use in schools and community-based arts+computing programs.

Brendan Reilly, *Research Assistant*, is an undergraduate Computer Science major at UMass Lowell, expecting to complete his B.S. in Computer Science in 2014. He has played bass since grade school, participating in every musical group available to him. After taking a course on Java, however, he decided to go into CS while keeping music in his

Contact Information

Jesse M. Heines, *Principal Investigator*
Dept. of Computer Science
University of Massachusetts Lowell
Lowell, MA 01854
e-mail: **Jesse_Heines@uml.edu**
phone: +1 (978) 934-3634

Gena R. Greher, *Co-Principal Investigator*
Dept. of Music
University of Massachusetts Lowell
Lowell, MA 01854
e-mail: **Gena_Greher@uml.edu**
phone: +1 (978) 934-3893

S. Alex Ruthmann, *Co-Principal Investigator*
Dept. of Music & Performing Arts Professions
New York University
New York, NY 10003
e-mail: **Alex.Ruthmann@nyu.edu**
phone: +1 (212) 998-5607

Scott Lipscomb, *Project Evaluator*, is an Associate Professor of Music Education at the University of Minnesota. He teaches courses in music education, music cognition, and music technology. Scott's research areas include facilitation of music learning through technology integration, interactive instructional media development, and multimedia cognition. His work has been published in numerous peer-reviewed journals and edited volumes, and he presents frequently at national and international conferences. Scott is Editor of the *Journal of Technology in Music Learning*, immediate past President of the *Association for Technology in Music Instruction*, Research Committee Chair for the *Technology Institute for Music Educators*, and Treasurer for the *Society for Music Perception and Cognition*.

Fred Martin, *Senior Personnel*, is an Associate Professor of Computer Science and serves as Associate Dean of the College of Sciences at UMass Lowell. He teaches courses in robotics, programming languages, software engineering, and artificial intelligence, and co-taught the Performamatics interdisciplinary Tangible Interaction Design course. In 2006, Fred received an NSF Faculty Early Career Development award (REC-0546513, $599,943). His present focus is science inquiry, using electronic sensors for data collection and the web for data-sharing and visualization.

background.

Zachary Robichaud, *Research Assistant*, is a graduate Computer Science major at UMass Lowell expecting to complete his M.S. in Computer Science in 2015. He has been a musician since he was in second grade, playing the guitar, piano, and trombone. Zack received his bachelor’s degree in Sound Recording Technology from UMass Lowell in 2012. He hopes to use his knowledge of music and sound recording with his master degree to create and develop music and audio applications.

Sarah Kuhn, *Senior Personnel*, is a Professor of Psychology at UMass Lowell. She is commited to innovation in learning, particularly blending technical education with the social sciences and arts, which led her to create the UMass Lowell Laboratory for Interdisciplinary Design. Sarah is a member of the Social Science Advisory Board of the NSF-funded National Center for Women & Information Technology and was a member of the National Research Council Committee on Workforce Needs in Information Technology. She was Co-PI of Project TechForce, an NSF-funded study of women and men working in the Massachusetts software and Internet industry.

PERFORMAMATICS

University of Massachusetts Lowell
Depts. of Music and Computer Science

UMASS LOWELL

Computational Thinking through Computing and Music

an interdisciplinary NSF TUES project

Home Workshop About Our Project About Us **Resources** Publications

Resources Linked from This Page

- Presentation Slides
- Materials from Conference Workshops
- Videos (on YouTube and elsewhere)
- Sound Thinking Course Websites

Presentation Slides

Top

Teaching Computing and Music *Together*
Adam Mickiewicz University
Poznań, Poland, May 17, 2012

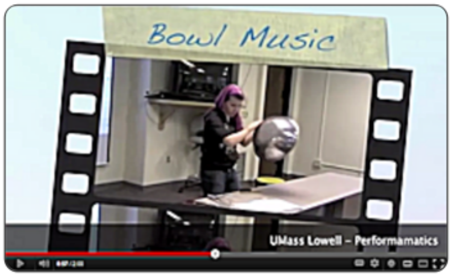
- PowerPoint Slides (PDF file)

Teaching Artsy Types to Think Like Geeks and Vice Versa (*reprise*)
Georgia Tech Graphics, Visualization and Usability (GVU) Group
Atlanta, GA, December 8, 2011

- Abstract and Video (on GVU Website)
- PowerPoint Slides (PDF file)

Teaching Artsy Types to Think Like Geeks and Vice Versa
Georgia Tech Center for Music Technology (CMT) Group
Atlanta, GA, October 2011

- PowerPoint Slides (PDF file)



Materials from Conference Workshops

Top

for materials used in our TUES workshops, please **click here**

Making Music with Scratch Workshop
Scratch@MIT 2012, MIT Media Lab
Cambridge, MA, July 27, 2012

- Workshop Handout (PDF file)

Scratch Day
Brisbane, Australia, May 19, 2012

- Announcement and Abstract (website)
- Scratch Music Projects (website)

Scratch Workshop
Association for Technology in Music Instruction (ATMI)
Richmond, VA, October 2011

- Prezi Presentation (website)
- Scratch Files (ZIP file)

Making Music with Scratch Workshop
Association for Computing Machinery Special Interest Group in Computer Science Education Technical Symposium (ACM SIGCSE)
Dallas, TX, March 2011

- Workshop Handout (PDF file)

Making Music with Scratch Workshop
Scratch@MIT 2010, MIT Media Lab
Cambridge, MA, August 13, 2010

- Workshop Handout (PDF file)

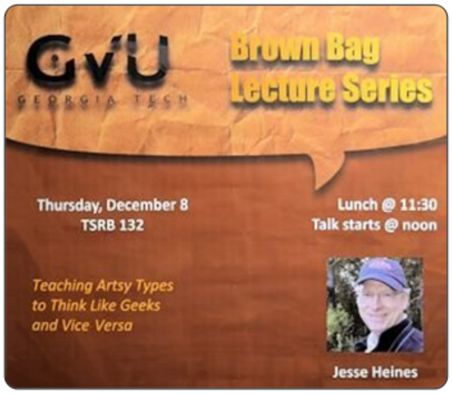
- Attendee Interview (video on YouTube)

Videos (on YouTube and elsewhere)

Top

Computational Thinking in Performamatics and
Found Sound Project Explanation and Examples, July 1, 2012

- Created by Prof. Gena Greher
- Premiered at *ITiCSE 2012*, Haifa, Israel, July 4, 2012, and *ISME 2012*, Thessaloniki, Greece, July 16, 2012



Movement 1 – Pärt, December 9, 2011

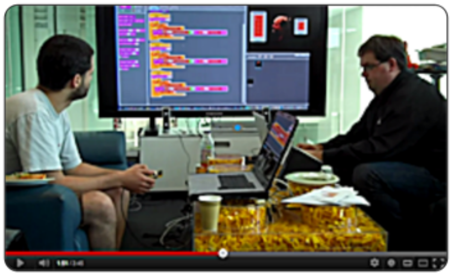
- Live coding composition and performance by Prof. Alex Ruthmann
- Published in *Computer Music Journal* 35(4):127-128, DVD Track 17
- Inspired by the musical organization of Arvo Pärt's *Stabat Mater* and a live coding performance by Andrew Sorensen within his Impromptu software
- Minor pentatonic solo layer was performed live over the drone using an IchiBoard sensor interface

Teaching Artsy Types to Think Like Geeks and Vice Versa, December 8, 2011

- Presentation by Jesse Heines at Georgia Tech
- Introduction by Mark Guzdial
- Commentary by Mark Guzdial in his *Computing Education* blog

Live Coding & IchiBoard-Enhanced Performance, July 22, 2011

- UMass Lowell Band Camp
- Composed and performed by students Amy & Katy



Pärt-Inspired Musical Live Coding Etude in Scratch, April 15, 2011

- Short musical live coding example using MIT's Scratch software and an IchiBoard
- A minor pentatonic improvisation over a drone bass
- Code and live performance by Alex Ruthmann

Musical Live Coding & Improv Performance using Scratch at the MIT Media Lab, July 13, 2010

- A musical live coding and IchiBoard sensor board improvisation duet performance of "Scratch Etude — Pärt"
- Musical live coding by Alex Ruthmann, UMass Lowell
- IchiBoard improvisations by Eric Rosenbaum, MIT Media Lab

Making Music with MIT's Scratch, an Ichiboard, and Sensors, April 29, 2010

- Composed and performed by Jeremy Murray and Nicole Dwyer
  - With special help from Kevin Webb as the "conductive human"
- Spring 2010 Sound Thinking Project Exhibition
- 119 Gallery, Lowell, MA

Sound Thinking Course Websites

Top

Spring 2012 Semester
Spring 2011 Semester
Spring 2010 Semester
Spring 2009 Semester

University of Massachusetts Lowell
Depts. of Music and Computer Science

Computational Thinking through Computing and Music

an interdisciplinary NSF TUES project

Home Workshop About Our Project About Us Resources Publications Scratch Laptop Orchestra

Please click the arrow next to each entry to show or hide its abstract.

Expand All Collapse All

Greher, G.R. and Heines, J.M. (2014). **Computational Thinking in Sound**. New York: Oxford University Press. [Please see www.compthinkinsound.org links and purchase information.]

This book demonstrates to music fundamentals educators how the range of mental tools in computer science — for example, analytical thought, system design, and problem design and solution — can be fruitfully applied to music education. While technology instruction in music education has traditionally focused on teaching how computers and software work to produce music, Greher and Heines offer context: a clear understanding of how music technology can be structured around a set of learning challenges and tasks of the type common in computer science classrooms. Using a learner-centered approach that emphasizes project-based experiences, the book provides music educators with multiple strategies to explore, create, and solve problems with music and technology in equal parts. It also provides examples of hands-on activities that encourage students, alone and in interdisciplinary groups, to explore the basic principles that underlie today's music technology and expose them to current multimedia development tools.

Ruthmann, S.A. (2013). **Exploring New Media Musically and Creatively**. In Burnard, P. & Murphy, R. (eds.), *Teaching Music Creatively*, pp. 85-97. London: Routledge.

This book chapter provides an introduction to projects and tools for exploring the creative dimensions of new media with primary pupils. I begin with an introduction to creative musicianship with new media, followed by an overview of tools for creating and being creative with new media. These tools are discussed in the context of practical projects and creative strategies for teacher and pupil exploration within primary classrooms.

Ruthmann, S.A. (2012). **Drive-By Workshop: Designing Learning Experiences for Kids at the Intersection of Computing and Music**. *Studio for Electro-Instrumental Music*. Amsterdam, The Netherlands, August 7, 2012.

This Drive-By workshop will engage participants in discussions and hands-on work exploring models of experience design for kids at the intersection of computing and music. We will explore the use of the Scratch visual programming environment (MIT Media Lab, scratch.mit.edu) as a tool for teaching children aspects of computer music, live coding, hardware interfacing, and computational thinking.

Heines, J.M. and Greher, G.R. (2012). **Getting into the Digital Music Game With Scratch**. *Scratch@MIT 2012*. Cambridge, MA, July 25-28, 2012.

This workshop introduces participants to Scratch's music-generating capabilities and shows how they can be used to teach computing concepts to students with a wide range of music and computing experience. The workshop demonstrates techniques that we use in "Sound Thinking," a university General Education (GenEd) course open to all students in all majors. Participants will receive an extensive handout with links to our teaching materials, and they will have the opportunity to create music-generating programs themselves using a variety of Scratch constructs. The workshop will conclude with a mini concert in which some participants will play the music that they created.

Ruthmann, S.A., Greher, G.R., & Heines, J.M. (2012). **Real World Projects for Developing Musical and Computational Thinking**. *30th International Society for Music Education (ISME) World Conference on Music Education*. Thessaloniki, Greece, July 15-20, 2012.

Music and the arts are engaging contexts for students to learn, integrate and apply computational concepts and thinking strategies. Our demonstration will begin by sharing our process of interdisciplinary collaboration, following by exemplar integration projects and examples of student work integrating musical and computational thinking developed through our research. These projects were designed to reflect the real-world collaboration skills, disciplinary and interdisciplinary understandings needed in the development of music and arts technologies today and in the future.

Heines, J.M., Greher, G.R., & Ruthmann, S.A. (2012). **Techniques at the Intersection of Computing and Music**. *17th Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE)*. Haifa, Israel, July 3-5, 2012.

Our work on Performamatics aims to enhance students' computational thinking (CT) by engaging them in fundamental concepts that unite computing and music. Our approach leverages students' near universal interest in music as a context for rich CT experiences. The techniques we share are used in a General Education course open to students in any major called *Sound Thinking*, which is now being offered for the fourth time.

Ruthmann, S.A. (2011). **Movement 1 – Pärt** [DVD and program notes]. *Computer Music Journal* 35(4):127-128, DVD Track 17, December 2011.

Movement 1 – Pärt (2010) is an excerpt from a suite of live-coding pieces, *Scratch Etudes*, exploring the live, interactive coding capabilities of the Scratch visual programming environment (scratch.mit.edu). Taking inspiration from the musical organization of Arvo Pärt's *Stabat Mater* and a live coding performance by Andrew Sorensen within his Impromptu software, *Movement 1 – Pärt* utilizes live manipulation of visual code chunks, blocks, lists, and variables through mouse and keyboard control in a crcative exploration of the Aeolian mode. An additional minor pentatonic solo layer was performed live over the drone using an IchiBoard sensor interface (bit.ly/ichiboard) developed by Mark Sherman in the Engaging Computing Group at the University of Massachusetts Lowell.

Video of live coding performance

Heines, J.M., Greher, G.R., Ruthmann, S.A., & Reilly, B. (2011). **Two Approaches to Interdisciplinary Computing+Music Courses**. *IEEE Computer* 44(12):25-32, December 2011.

The intersection of computing and music can enrich pedagogy in numerous ways, from low-level courses that use music to illustrate practical applications of computing concepts to high-level ones that use sophisticated computer algorithms to process audio signals. This paper explores the ground between these extremes by describing our experiences with two types of interdisciplinary courses. In the first, arts and computing students worked together to tackle a joint project even though they were taking independent courses. In the second, all students enrolled in the same course, but every class was taught by two professors: one from music and the other from computer science. This course was designed to teach computing and music *together*, rather than one in service to the other. This paper presents the philosophy and motivation behind these courses, describes some of the assignments students do in them, and shows examples of student work.

Abstract in IEEE Computer Society CS Digital Library

with link to full paper as published (requires member login)

Ruthmann, S.A., Heines, J.M., Greher, G.R., Laidler, P., & Saulters, C. (2010). **Teaching Computational Thinking through Musical Live Coding in Scratch**. *41st ACM SIGCSE Technical Symposium on CS Education*. Milwaukee, WI, March 12, 2010.

This paper discusses our ongoing experiences in developing an interdisciplinary general education course called *Sound Thinking* that is offered jointly by our Dept. of Computer Science and Dept. of Music. It focuses on the student outcomes we are trying to achieve and the projects we are using to help students realize those outcomes. It explains why we are moving from a web-based environment using HTML and JavaScript to Scratch and discusses the potential for Scratch's "musical live coding" capability to reinforce those concepts even more strongly.

Heines, J.M. & Ruthmann, S.A. (2010). **The Scholarship of Teaching and Learning – Sound Thinking: An Interdisciplinary GenEd** (presentation video). *1st Annual Faculty Development Conference*. Lowell, MA, April 9, 2010.

This session presented "Sound Thinking," an interdisciplinary course that explores the intersection of music and computer science. We offered this course for the second time in the Spring 2010 semester, and we made significant changes based on student feedback from the first offering. Any faculty interested in teaching an interdisciplinary course, particularly one designated as a GenEd, will benefit from hearing about our experiences. We cover course planning, maximizing benefits to students in different departments, teaching and learning style differences, preparing a GenEd proposal, addressing different departmental definitions of "scholarly activity," and impacts on professors.

Ruthmann, S.A., & Heines, J.M. (2009). **Designing Music Composing Software with and for Middle School Students: A Collaborative Project among Senior Computer Science and Music Education Majors**. *Association for Technology in Music Instruction (ATMI) 2009 Conference*. Portland, OR, October 23, 2009.

What might be learned through a collaborative project bringing together middle school students, pre-service music educators and senior computer science majors in the development of new web-based music composing software for middle school students? This question framed a collaborative project undertaken by music education majors enrolled in *General Music Methods II* and senior computer science majors enrolled in *GUI Programming II*. Our demonstration will share examples of the software developed through the project as well as student compositions created by the middle school students who designed and tested the software. The challenges and surprises faced during the collaborative development process will be shared detailing our implications for future collaborations among our students. We will also present the rationale for our decision-making as professors. Finally, the lessons learned and insights gained by the middle school students and the music education and computer science majors will be shared and discussed.

Additional Co-Presenters

C. Holly Johnston, Music Teacher, Stony Brook Middle School, Westford, MA

Marie Gleason-Tada, Instructional Technoloty Specialist, Parker Middle School, Chelmsford, MA

PowerPoint Slides (as a PDF file)

Greher, G.R., & Heines, J.M. (2009). **Sound Thinking: Conceptualizing the Art and Science of Digital Audio for an Interdisciplinary General Education Course**. *Association for Technology in Music Instruction (ATMI) 2009 Conference*. Portland, OR, October 22, 2009.

What is sound? How do we capture, manipulate, and harness it in the digital world? Thus began our journey into the creation of a general education course focused on the intersection of art and technology, exploring how sound is integrated into computer applications. Our goal seemed simple enough. We were going to have students explore the art and science of digital audio, from the basic end-user applications that promote creative expression and exploration, to the underlying code that allows these programs to function. Once we began to examine what this means from the perspective of a total novice, and began to think about the software applications we would work with, as well as the projects we would assign, we began to reevaluate and question our own assumptions about the purpose of general education courses and the challenges posed by teaching students outside of our respective disciplines.

PowerPoint Slides (as a PDF file)

Heines, J.M. (2009). **Interdisciplinary Computer Science Courses: Lessons Learned**. *Athens Institute for Education and Research (ATINER) 2009 Conference*. Athens, Greece, July 27, 2009.

In 2007, the National Science Foundation (NSF) funded 19 "Community Building" awards intended to "bring stakeholders together to discuss the challenges and opportunities inherent in transforming undergraduate computing education, and to identify creative strategies to do so." Our "creative strategy" has been to develop interdisciplinary courses that bring Computer Science (CS) majors together with Art, Music, and Theatre majors to work on joint projects in the area of exhibition and performance technologies. We call this strategy "Performamatics," because the common thread in these projects is that "many tasks, performed by multiple people, must come together on a tight schedule by a specific date to achieve a desired result. Performamatics also implies that each team member must 'perform' his or her task in a way that can be integrated into a final product, regardless of whether that team member participates visibly in the culminating event."

This presentation discusses the successes and failures we have experienced in trying to implement Performamatics courses over the last two years. In that time we have experimented with two pedagogical models: (a) "synchronized" courses in which students in different disciplines come together at strategic points to work on joint projects, and (b) "hybrid" courses in which all students enroll in a single course that has two instructors, one from Computer Science and one from Art, Music, or Theatre. Our presentation will describe the content of these courses, provide examples of student work, and suggest ways in which both the student and professor collaborations could be improved, all with the intent to provide others with solid guidance on implementing similar strategies at their own institutions.

Martin, F., Greher, G.R., Heines, J.M., Jeffers, J., Kim, H.J., Kuhn, S., Roehr, K., Selleck, N., Silka, L., and Yanco, H. (2009). **Joining Computing and the Arts at a Mid-Size University**. *2009 Conference of the Consortium for Computing Sciences in Colleges -- Northeastern Region (CCSCNE 2009)*. Plattsburgh, NY, April 24, 2009.

This paper describes two NSF-funded collaborations among faculty members in the Computer Science, Art, Music, and English departments at a public university in the Northeast USA. Our goal has been to create undergraduate learning opportunities across the university, focusing on connecting computer science to creative and expressive domains. In past publications, we have focused on student learning outcomes. This paper reports on the motivations, opportunities, and challenges for the faculty members involved.

Heines, J.M., Greher, G.R., & Kuhn, S. (2009). **Music Performamatics: Interdisciplinary Interaction**. *40th ACM SIGCSE Technical Symposium on CS Education*. Chattanooga, TN, March 7, 2009.

This paper describes how a graphical user interface (GUI) programming course offered by the Dept. of Computer Science (CS) was paired with a general teaching methods course offered by the Dept. of Music in an attempt to revitalize undergraduate CS education and to enrich the experiences of both sets of students. The paper provides details on the joint project done in these classes and the evaluation that assessed its effect on the curriculum, students, and professors.

Heines, J.M., Jeffers, J., & Kuhn, S. (2008). **Performamatics: Experiences with Connecting a Computer Science Course to a Design Arts Course**. *International Journal of Learning* 15(2):9-16.

Our work is based on a partnership between the a Computer Science (CS) and Art, Music, and English departments in the area of exhibition and performance technologies. We define these areas broadly to encompass all CS applications in the creative and performing arts. These areas not only resonate with today's media-rich culture, but reinforce the fact that virtually all computer applications now require the integration of creative elements. CS majors must learn to work with specialists in areas where the perspective is often quite different from their own. We believe that computer scientists have much to learn from those trained in the arts and vice versa. The common thread in performamatics projects is that many tasks, performed by multiple people, must come together on a tight schedule by a specific date to achieve a desired result. Performamatics also implies that each team member must "perform" his or her task(s) in a way that can be integrated into a final product, regardless of whether that team member participates visibly in the culminating event. Our paper reports on initial attempts to couple CS courses and integrate CS elements with courses in Art, Music, and Theater. We describe the techniques we used that were designed to increase the scope and level of creativity in student projects and the impact these techniques and the presence of interdisciplinary teams had on those projects. We discuss changes we will make to improve the experience for both groups of students in the future and suggest new techniques we may try to better achieve our goals.

Additional Web Resources

Performamatics Project Website
.... http://www.performamatics.org

Computational Thinking in Sound Website
.... http://www.compthinkinsound.org

Sound Thinking Course Websites
.... http://soundthinking.uml.edu/

Scratch Performamatics Resources
.... http://scratch.mit.edu/users/performamatics/

NYU Performamatics Scratch Studio
.... http://scratch.mit.edu/studios/222541

Ruthmann Blog Post re Being More Musical In and With Scratch
.... http://www.alexruthmann.com/blog1/?p=641

MaKey MaKey Website
.... http://makeymakey.com

Ruthmann Experiencing Audio page on
MaKey MaKey Music
.... http://www.experiencingaudio.org/2012/11/
makey-makey-music-workshop-materials.html

MaKey MaKey Musical Construction Kit and links to
Ruthmann students' MaKey MaKey music projects
.... http://makeymakeymusicalconstructionkit.blogspot.com

Eric Rosenbaum Home Page
.... http://web.media.mit.edu/~ericr

Getting Started on an Interdisciplinary Project

You may want to each take an inventory of the types of projects you currently do in your classes and think about ways each project could be extended to include aspects of the other person's discipline.

Think about what concepts each project may share in common with particular emphasis on how this project could support the development of computational thinking.

What concepts do you wish to cover and what types of projects will you create to address those concepts? What software will you use?

What type of project or situation could you devise to support setting your students' imaginations loose? In other words, try to think in terms of projects that support divergent outcomes.

Creating a project that is intriguing to both sets of students that provides learning outcomes benefitting both your disciplines will promote "buy in" for all involved.

Consistency is another area you and your colleague must work out through communication. There needs to be a consistent set of guidelines for both groups of students regarding due dates and grading policies.

What if you and your colleague discover you have completely different philosophical views towards teaching and grading? Do you rank your students? Are you more of a lecturer or a facilitator? How might you mesh your stylistic differences?

All projects need to be designed so that anyone, regardless of their major or educational level can succeed.

Computational Thinking Inventory for Found Sounds Project

Music

Aural analysis through making decisions about timbre, rhythm, dynamics, form and texture, not to mention the performance aspect.

For the creative notation part of the project, your students will be involved in visual analysis and some serious decision making regarding symbolic representation.

CS

The CT involved in these musical decision-making processes involves your students in temporal structuring, pattern recognition, the beginnings of procedural thinking, and categorizing.

The creative notation part of the project was where we felt the music making and computing could intersect the interests of both sets of students in a natural way. Both disciplines rely on a unique symbol system that is used to create and perform.

Computational Thinking Inventory for Musical Flowchart Project

Music

- Structural/macro analysis – through aural (listening) and lyrics
- Chunking, structural dictation, form, texture
- Representational encoding/decoding, repeats

Computer Science

- Procedural thinking: repetition, branching, `if` and `while` statements, signal flow, serial and parallel processing
- Representational encoding/decoding, pseudo-code, control structures, algorithms, loops

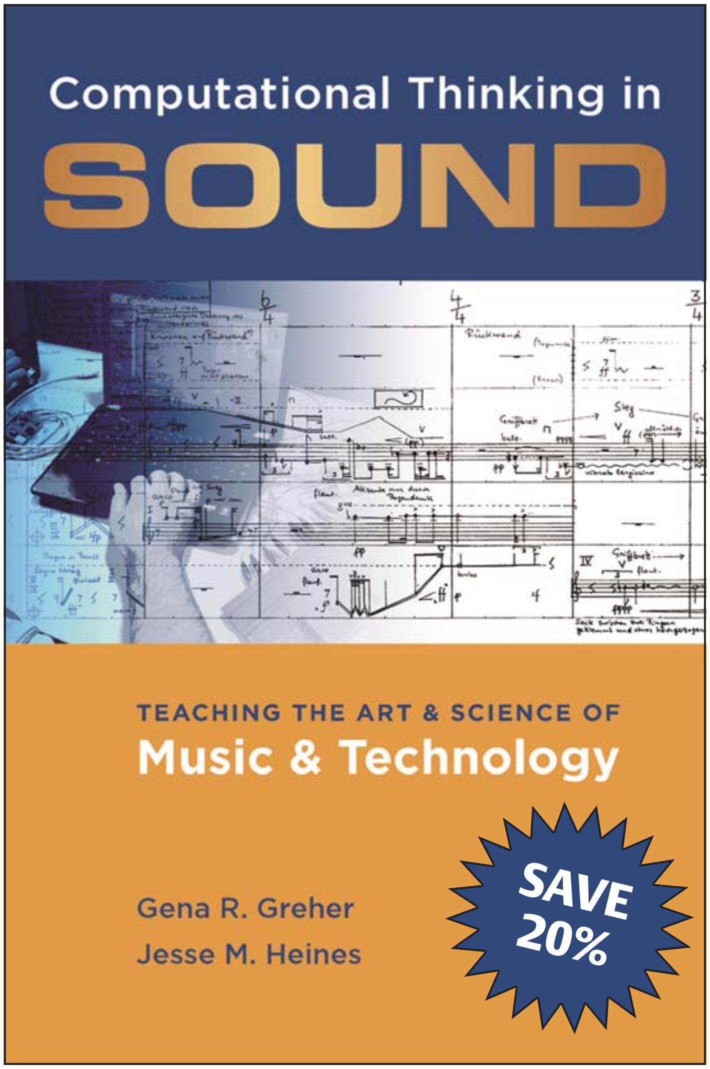
NEW FROM OXFORD

Computational Thinking in Sound

Teaching the Art and Science of Music and Technology

By Gena R. Greher and Jesse M. Heines

With *Computational Thinking in Sound*, veteran educators Gena R. Greher and Jesse M. Heines provide the first book ever written for music fundamentals educators that is devoted specifically to music, sound, and technology. Using a student-centered approach that emphasizes project-based experiences, the book provides music educators with multiple strategies to explore, create, and solve problems with music and technology in equal parts. It also provides examples of hands-on activities that encourage students, alone and in groups, to explore the basic principles that underlie today's music technology and freely available multimedia creation tools. *Computational Thinking in Sound* is an effective tool for educators to introduce students to the complex process of computational thinking in the context of the creative arts through the more accessible medium of music.



March 2014

Hardcover | 272 pp. | ~~$35.00~~ **$28.00**
978-0-199-826190-3

Gena R. Greher is a Professor of Music Education at UMass Lowell. Her research focuses on creativity and listening skill development in children and examining the influence of integrating multimedia technology in urban music classrooms and music teacher education through School-University partnerships.

OXFORD
UNIVERSITY PRESS

Jesse M. Heines is a Professor of Computer Science at UMass Lowell. His primary teaching responsibilities include object-oriented programming and graphical user interfaces. His research focuses on computer science education, particularly interdisciplinary approaches that blend computer science with music and other fields to enhance instructional effectiveness in both.

Order online at **www.oup.com/us**. Enter promotion code **28862** to save 20%

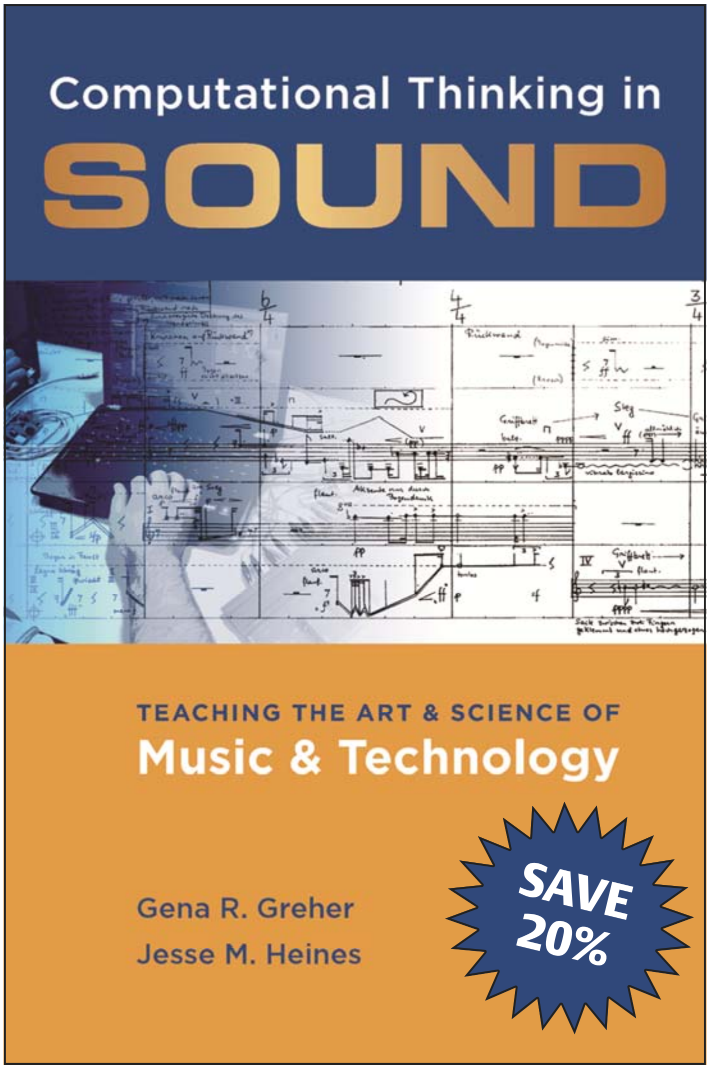
NEW FROM OXFORD

Computational Thinking in Sound

Teaching the Art and Science of Music and Technology

By Gena R. Greher and Jesse M. Heines

With *Computational Thinking in Sound*, veteran educators Gena R. Greher and Jesse M. Heines provide the first book ever written for music fundamentals educators that is devoted specifically to music, sound, and technology. Using a student-centered approach that emphasizes project-based experiences, the book provides music educators with multiple strategies to explore, create, and solve problems with music and technology in equal parts. It also provides examples of hands-on activities that encourage students, alone and in groups, to explore the basic principles that underlie today's music technology and freely available multimedia creation tools. *Computational Thinking in Sound* is an effective tool for educators to introduce students to the complex process of computational thinking in the context of the creative arts through the more accessible medium of music.



March 2014

Hardcover | 272 pp. | ~~$99.00~~ **$79.00**
978-0-19-982617-9

Gena R. Greher is a Professor of Music Education at UMass Lowell. Her research focuses on creativity and listening skill development in children and examining the influence of integrating multimedia technology in urban music classrooms and music teacher education through School-University partnerships.

OXFORD
UNIVERSITY PRESS

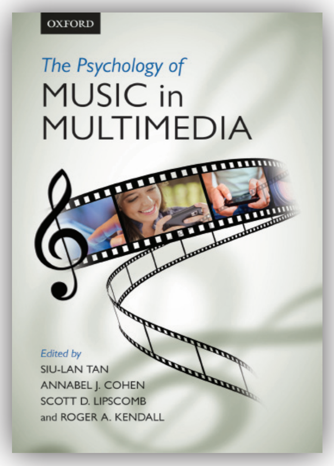
Jesse M. Heines is a Professor of Computer Science at UMass Lowell. His primary teaching responsibilities include object-oriented programming and graphical user interfaces. His research focuses on computer science education, particularly interdisciplinary approaches that blend computer science with music and other fields to enhance instructional effectiveness in both.

Order online at **www.oup.com/us**. Enter promotion code **28862** to save **20%**

Brain Sciences from Oxford University Press

THE PSYCHOLOGY OF MUSIC IN MULTIMEDIA

Edited by SIU-LAN TAN, Department of Psychology, Kalamazoo College, USA; ANNABEL J. COHEN, Department of Psychology, University of Prince Edward Island, CANADA; SCOTT D. LIPSCOMB, School of Music, University of Minnesota, USA; and ROGER A. KENDALL, Herb Alpert School of Music, University of California, Los Angeles, USA



For most of the history of film-making, music has played an integral role serving many functions - such as conveying emotion, heightening tension, and influencing interpretation and inferences about events and characters. More recently, with the enormous growth of the gaming industry and the Internet, a new role for music has emerged. However, all of these applications of music depend on complex mental processes that are being identified through research on human participants in multimedia contexts. *The Psychology of Music in Multimedia* is the first book dedicated to this fascinating topic.

The Psychology of Music in Multimedia presents a wide range of scientific research on the psychological processes involved in the integration of sound and image when engaging with film, television, video, interactive games, and computer interfaces. Collectively, the rich chapters in this edited volume represent a comprehensive treatment of the existing research on the multimedia experience, with the aim of disseminating the current knowledge base and inspiring future scholarship. The focus on empirical research and the strong psychological framework make this book an exceptional and distinctive contribution to the field. The international collection of contributors represents eight countries and a broad range of disciplines including psychology, musicology, neuroscience, media studies, film, and communications. Each chapter includes a comprehensive review of the topic and, where appropriate, identifies models that can be empirically tested.

Part One presents contrasting theoretical approaches from cognitive psychology, philosophy, semiotics, communication, musicology, and neuroscience. Part Two reviews research on the structural aspects of music and multimedia, while Part Three focuses on research examining the influence of music on perceived meaning in the multimedia experience. Part Four explores empirical findings in a variety of real-world applications of music in multimedia including entertainment and educational media for children, video and computer games, television and online advertising, and auditory displays of information. Finally, the closing chapter in Part Five identifies emerging themes and points to the value of broadening the scope of research to encompass multisensory, multidisciplinary, and cross-cultural perspectives to advance our understanding of the role of music in multimedia.

This is a valuable book for those in the fields of music psychology and musicology, as well as film and media studies.

Features:

- Makes a unique and distinctive contribution to the field through its focus on empirical research
- Explains what research with human participants has shown about the role of music in the multimedia experience
- Includes companion website featuring: audio-visual materials, audio files, and animations, including original laboratory stimuli used in some research studies discussed in the book
- Represents diverse and rich perspectives from an international and interdisciplinary roster of authors each conducting research in their area of expertise; useful to a wide range of disciplines

"*The Psychology of Music in Multimedia* marks a critical turning point in re-envisioning the established methods for analyzing music and moving image within multimedia. Grounded in the burgeoning empirical research of leading scholars in psychology and music perception, scientific approaches are merged with analytical modalities of musicology, music technology, and film studies... The authors provide an exemplary blueprint for critical inquiry, gearing the concerted efforts of diverse scholars toward interdisciplinary and broad-based analytical strategies."
-- Ronald Sadoff, Associate Professor and Director, Scoring for Film and Multimedia, NYU Steinhardt, USA and composer for acclaimed films, including 2006 Academy Award-winning *The Moon and The Son: An Imagined Conversation*

4 EASY WAYS TO ORDER

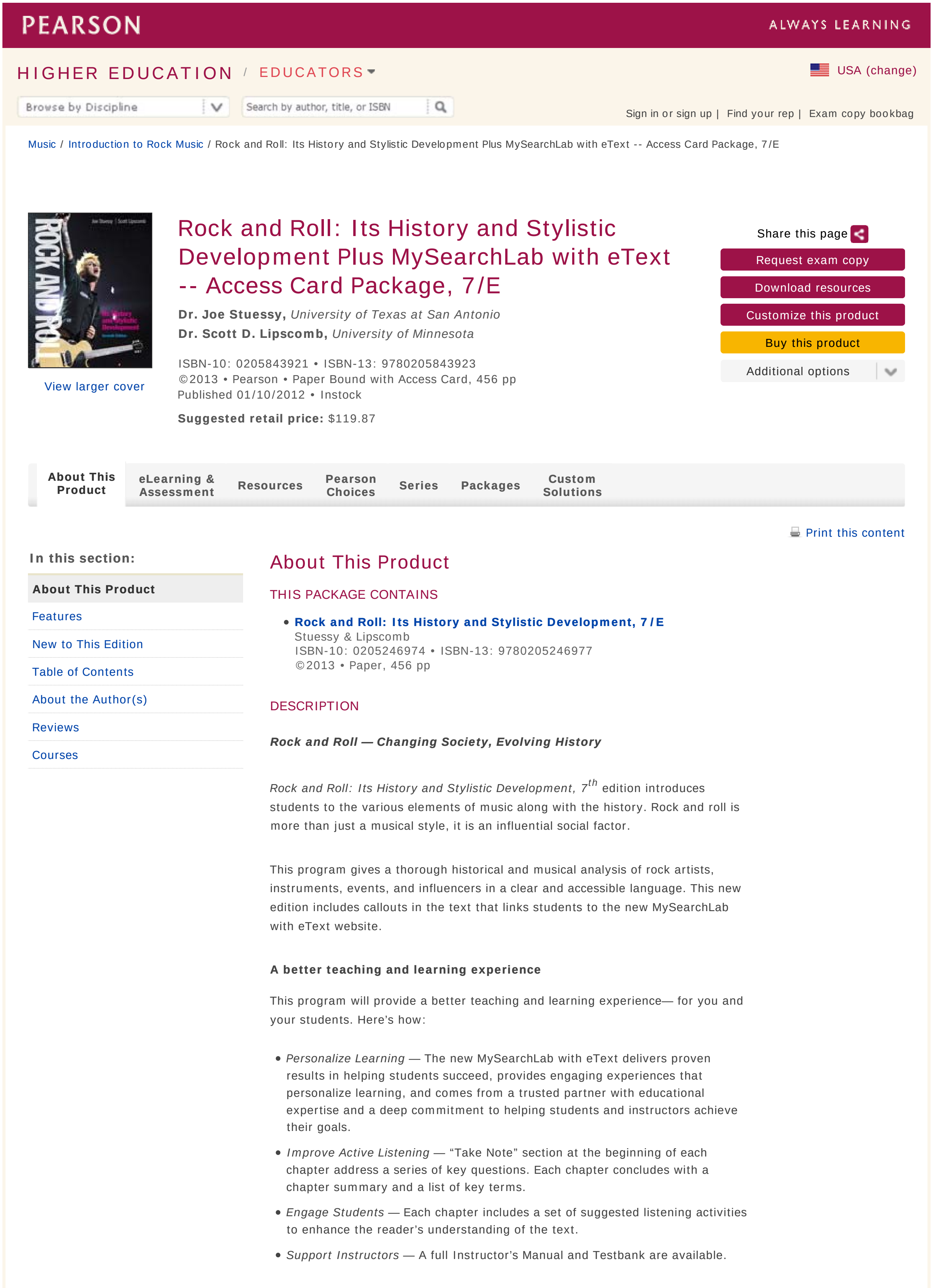
PROMO CODE:31980

•Phone: 800.451.7556 •Fax: 919.677.1303 •Web: www.oup.com/us

•Mail: Oxford University Press, Order Dept., 2001 Evans Road, Cary NC 27513

Save 20% with promo code 31980!

Sept 2013 | 456 pages | 9780199608157 | Paperback | ~~$59.95~~/$**47.95**



Music / Introduction to Rock Music / Rock and Roll: Its History and Stylistic Development Plus MySearchLab with eText -- Access Card Package, 7/E

Rock and Roll: Its History and Stylistic Development Plus MySearchLab with eText -- Access Card Package, 7/E

Dr. Joe Stuessy, *University of Texas at San Antonio*
Dr. Scott D. Lipscomb, *University of Minnesota*

ISBN-10: 0205843921 • ISBN-13: 9780205843923
©2013 • Pearson • Paper Bound with Access Card, 456 pp
Published 01/10/2012 • Instock

Suggested retail price: $119.87

About This Product

THIS PACKAGE CONTAINS

- **Rock and Roll: Its History and Stylistic Development, 7/E**
  Stuessy & Lipscomb
  ISBN-10: 0205246974 • ISBN-13: 9780205246977
  ©2013 • Paper, 456 pp

DESCRIPTION

Rock and Roll — Changing Society, Evolving History

Rock and Roll: Its History and Stylistic Development, 7th edition introduces students to the various elements of music along with the history. Rock and roll is more than just a musical style, it is an influential social factor.

This program gives a thorough historical and musical analysis of rock artists, instruments, events, and influencers in a clear and accessible language. This new edition includes callouts in the text that links students to the new MySearchLab with eText website.

A better teaching and learning experience

This program will provide a better teaching and learning experience— for you and your students. Here's how:

- *Personalize Learning* — The new MySearchLab with eText delivers proven results in helping students succeed, provides engaging experiences that personalize learning, and comes from a trusted partner with educational expertise and a deep commitment to helping students and instructors achieve their goals.
- *Improve Active Listening* — "Take Note" section at the beginning of each chapter address a series of key questions. Each chapter concludes with a chapter summary and a list of key terms.
- *Engage Students* — Each chapter includes a set of suggested listening activities to enhance the reader's understanding of the text.
- *Support Instructors* — A full Instructor's Manual and Testbank are available.

3

Interdisciplinary Teaching and Learning: Two Heads Might Actually Be Better Than One

Background

Learning to work with others is a life-long endeavor. These skill sets don't develop in a vacuum. They need to be nurtured through modeling and experience. As suggested by John-Steiner [3], students need to be socialized into the culture of collaborative work and the kinds of creative and critical thinking the new workplace requires.

As you will discover, collaborative work yields processes and results that are far richer than any that a single person's expertise can produce. John-Steiner calls this "creative collaboration": "In collaborative work we learn from each other by teaching what we know; we engage in mutual appropriation. Solo practices are insufficient to meet the challenges and the new complexities of classrooms, parenting, and the changing workplace" [*op. cit.*, p. 3]. However, we found out both by observing the students in their project teams and by reflecting on our own interactions as teachers, that collaborative work creates a complex dynamic where diverse ideas and opinions are continually being challenged and negotiation is a constant presence. It quickly became clear that creating interdisciplinary assignments and forming multidisciplinary student teams to share the work is not enough. Getting team members to actually learn new skills from their peers in other disciplines simply would not happen without a great deal of mediation and guidance from us, the teachers.

Defining Interdisciplinary Teaching

So what exactly do we mean when we say that a course is "interdisciplinary"? There are many definitions and variations, some involving one teacher presenting multiple perspectives and others involving team teaching.

Interdisciplinarity is often described as “the integration of disciplinary perspectives” [4], which in our case would be music and computer science. It is often confused with “multidisciplinary,” which Spelt et al. [8] describe as “additive,” presenting multiple perspectives without the integration of the disciplines or opportunities for students to synthesize knowledge from them. Davis [1] points to the need for “blurring genres” and “synthesizing disparate sources into new knowledge.” He claims that the disciplinary focus of much of higher education cannot adequately address the issues confronting 21st century thinking. Davis is an advocate for faculty to work in teams, believing that teams can achieve what individual disciplinary specialists cannot.

We embrace Davis’s last point wholeheartedly. To us, it takes two to tango. We feel that the only way to give students a truly interdisciplinary experience is to have a professor from each discipline present at all class meetings. We understand the logistical problems that this presents, and we address those in a later chapter. We also understand the differences of opinion and perspective that will arise in class, but we see no need to hide those from students….

“Synchronized” vs. “Hybrid” Courses

One of the first decisions you will need to make is whether to take the plunge into interdisciplinary teaching, either by revising and rethinking an entire course or by beginning with a small well-defined project. Our first foray was project-based and probably closer to the “additive” multidisciplinary approach. We put students together from regular classes that we were already teaching, *General Music Methods* and *GUI Programming.* (“GUI” stands for “Graphical User Interface”.) Our colleague Fred Martin dubbed these “synchronized” courses [5]. Students did a project that we will describe later in detail, but suffice it to say at this point that the students in each course did their parts of the project independently. They then came together a few times during the semester to share, discuss, and evaluate their work.

The synchronized courses showed some positive results, but we wanted more. We wanted students from different disciplines to work together throughout an entire semester. We therefore developed what Martin [*op cit.*] termed a "hybrid" course, co-taught with both of us in the classroom simultaneously. This is *Sound Thinking*, focusing on the study of sound from the perspective of digital musicianship.

Learning from Experience

As you begin to develop your interdisciplinary experience, be mindful of setting up regular meeting times before and during the semester for planning, reflecting, and modifying some of the course details. Strong communication and the ability to evaluate one's work "in process" are essential to a successful collaboration. You and your colleague should attempt to make a conscious effort to find a common class meeting time and try to set aside specific days when the classes will meet as one. As we both learned from previous attempts at this, it's a good idea to introduce the project together and build it in phases. You may also wish to think about setting up multiple planning meetings before the beginning of the semester as well as scheduling regular meetings throughout the semester to evaluate, reflect upon, and make modifications to the project as needed.

You may also wish to consider building in time for the students to evaluate each other's work. Peer-to-peer feedback is invaluable to the learning process, particularly so with a diverse group of students with different strengths. As previously suggested with regard to the definition of computational thinking, we sometimes get hampered by our habits of mind and modes of thinking. Getting students to think about solutions from another person's perspective will help develop their analytical thinking skills, one of the benefits of a computational thinking mindset. In our case, when the teams were required to present their work, we did it as one class. This made it possible for CS students to comment on the work of and give feedback to the music students, and vice-versa. In addition, the students developed an appreciation for the differences in thinking and learning styles and the creative thinking that formed a common link between them.

What resulted was a successful collaboration all around, based on what, for the most part, fits the definition of an interdisciplinary project. The general sentiment of the Music Ed class with regard to this new collaboration was summed up by one student:

> Boy, do I really like to have the CS students in our class! I feel we are really becoming one class, not just two classes in the same room. It's great to have other voices in the class, and to provide perspective from students outside the music department.

"Not just two classes in the same room." What better confirmation could we have asked for?

Building an entire interdisciplinary course around a long-range project or thematic idea can be a daunting task without a clear idea of your purpose for doing this in the first place. In the case of our own class, our goal was to create an entire interdisciplinary course for all students interested in music technology and how to manipulate it. Or, more to the point, we were going to have students "get under the hood" to discover how these programs work. We therefore based our strategies and projects on the interdisciplinary synchronized course module we created for computer science and music majors.

Whether you are attempting a single interdisciplinary project or developing an entire course, you may want to use the following questions to help frame your thinking and planning:

1. What is it that you hope to gain with regard to your students' learning?
2. How will this impact your own personal and professional growth?
3. What will be the overall benefits? What barriers will you have to overcome?
4. What compromises will you have to make when you enter into a long-term collaborative endeavor?
5. And, most importantly, how exactly are you defining interdisciplinarity with regard to your content and teaching?

In our case, our rationales for developing the new course were straightforward. For Music Education majors, there are few opportunities to gain immersion into areas of study outside their discipline, let alone the technologies that may support and enhance their work. Even students in our Sound Recording Technology program sometimes lack sufficient opportunities to understand the programming and visual aspects of multimedia technology that support much of their work. At the other end of the spectrum, those who design and build software applications have few opportunities to pursue in-depth study of multimedia applications from the perspectives of the audio and visual artists who are their end users.

Whatever your rationale, one of your goals should be to break down boundaries created by compartmentalized instruction and have your students see their own work through an interdisciplinary lens. You will find that such experience is critical in preparing students for the multidisciplinary workplace, regardless of their major field.

Benefits to Students

We set out to explore broad concepts through the lenses of our respective disciplines and integrate those concepts within a project-based learning environment. We put students into the position of decision makers, a role they do not often occupy [2]. Boix Mansilla and Gardner suggest that such projects give students multiple opportunities to develop "performances of understanding," in which students are invited "to think with knowledge in multiple novel situations" [*op cit.*, p. 105]. They discuss at length the importance of training students to think like the practitioners in the various fields of study they encounter during their schooling.

So what does this mean for your students and why should you care? In our case, one of our goals was to give students "real-world" experiences. Perhaps the fact that we both began our careers in the world of business might have something to do with shaping our perspectives on the skills and thinking that our students will need once they leave our classrooms.

Among the many advantages of interdisciplinary courses is that these courses encourage and support creative risk taking and the ability to accept ambiguity [7]. This would seem to benefit professors, as well as students.

Benefits to the Professors

Here are some of the benefits we experienced. We are fairly certain that others can experience these, as well:

- You will each learn a lot more about each other's discipline.
- You can attend and present your work at conferences in each other's field, further expanding your respective knowledge of each other's disciplines.
- You will most likely receive significant recognition within your own university, raising your profiles. In our case we were each invited to serve on university-level committees that may influence the future of interdisciplinary teaching at our institution.
- You will be introduced to other colleagues in your respective departments, thereby expanding the scope of your work, leading to new collaborations and possibly to grant applications.

On top of all that, if our experiences are any indication, you will quite simply have a lot of fun. This last point should not be taken lightly. The National Science Foundation program that funded our own work was conceived to "revitalize undergraduate education in computing" [6]. As we strove to achieve that goal, we found that the process revitalized us as educators. Such faculty revitalization is clearly key to educational transformation, because while faculty are not always the major source of curriculum *innovation*, they are — and for the foreseeable future will remain — the major component of curriculum *implementation*.

Bibliography for Chapter 3

[1] Davis, J.R. (1995). *Reengineering Teaching for 21st Century Learning.* The Educational Record **76**(4):16-23.

[2] Greher, G.R. (2006). *Transforming Music Teacher Preparation through the Lens of Video Technology*. Jrnl. of Music Teacher Ed. **15**(2):49-60.

[3] John-Steiner, V. (2000). *Creative Collaboration*. New York: Oxford Univ. Press.

[4] Lattuca, L.R., Voigt, L.J., & Fath, K.Q. (2004). *Does Interdisciplinarity Promote Learning? Theoretical Support and Researchable Questions.* The Review of Higher Education **28**(1):23-48.

[5] Martin, F., Greher, G.R., Heines, J.M., Jeffers, J., Kim, H.-J., Kuhn, S., Roehr, K., Selleck, N., Silka, L., & Yanco, H. (2009). *Joining Computing and the Arts at a Mid-Size University.* Jrnl. of Computing Sciences in Colleges **24**(6):87-94.

[6] National Science Foundation (2010). *CISE Pathways to Revitalized Undergraduate Computing Education (CPATH).* www.nsf.gov/funding/pgm_summ.jsp?pims_id=500025 *accessed* 4/19/2010.

[7] Nikitina, S. (2005). *Pathways of Interdisciplinary Cognition.* Cognition and Instruction **23**(3):389-425

[8] Spelt, E.J.H., Biemans, H.J.A., Tobi, H., Luning, P.A., & Mulder, M. (2009). *Teaching and Learning in Interdisciplinary Higher Education: A Systematic Review.* Educational Psychology Review **21**:365-378.

Engaging Adolescents with Music and Technology

Engaging Musical Practices: A Sourcebook for Middle School General Music

Music and technology are inseparable within the lives of today's adolescents. It seems like our students spend all of their time with their cell phones - tucked under their pillow texting at night and stealing time in between (and during!) class to text their friends and update their Facebook statuses. With the advent of portable digital music players, their music is always nearby providing the soundtrack to their life. This soundtrack accompanies the profound changes – physical, mental and social – occurring at home, with friends and at school. Our students' lives revolve around their relationships with each other. They are simultaneously exploring and hiding from the world, testing the waters of friendships and their own self-expression, while negotiating their changing identity. Music, mediated by all kinds of technologies, performs a key role in this process.

This chapter presents an emerging picture of promising practices with digital media and technology and advocates a relational pedagogy (Ruthmann & Dillon, in press) where teachers actively design musical experiences informed by their students' musical and technological lives. Practical strategies for easing into initial creative experiences with music and technology are shared, followed by projects that explore the affordances of new mobile and online music technologies in the classroom and community. This chapter concludes with two digital remix projects – one for student-created video games and another exploring music videos as source material for original creative works. All of the projects are united by the common theme of active, social music making with technology.

First Steps: Connecting with Your Students

When integrating technology into the classroom, it is all too easy to be drawn in by the flash and quick spark of technology. In my own practice, I've found that when the planning for the music learning experience begins with the technology as the starting point, it's not as successful and often does not sustain the students' interest beyond a class or two. Even when introducing technologies like the iPad to students with all of the possibilities afforded by that tool, it helps to have musical and educational goals in mind first. Rather than integrate technology for the sake of technology, we need to understand the how we might develop relationships among our goals and experiences as teachers, the affordances and constraints of the technology, our broader curricular and community contexts, and our students' interests and needs (Ruthmann & Dillon, in press).

Getting to Know Your Students' Musical and Technological Lives

As teachers, it is a never-ending challenge to stay connected to and knowledgeable about our students' ever-changing interests and tastes in music, let alone adolescent culture in general. We know what we see in our classrooms and in the hallways between classes, but this is only a small window into our students' musical world. When working with adolescents, I begin by asking my students a set of questions as a way to get to know them and the place of music and technology in their everyday lives. My first step is to take a stab at answering the questions myself, making predictions of how they might answer. This helps me as a teacher document and push my own assumptions about who my students are and where music and technology might be in their lives out of my head and onto paper or the computer screen. After I have done this preparation work, I distribute the following questions to my students:[1]

1. Where do you experience music in your life?
2. How do you use technology in your everyday life? What technologies do you use?
3. What does music mean to you?
4. What music do you listen to?
5. Do you ever make music?
6. Are you a musician? Why/Why not?
7. If you could learn anything about music, what would it be?
8. What do you like about music in school?
9. What do you not like about music in school?

After reading through my students' answers, I compare them to my own predictions. This process no doubt surprises me each time I do this. I always gain new insights into what music is currently popular, which technologies are being used by my students, and how music is meaningful in their lives. Starting with their answers, I make time to explore the Internet and sites like YouTube listening to and getting to know *their* music. And, many of the pieces discovered then find their way into future lessons and discussions.

Building Community in and Around Our Classrooms

[1] These questions are available online as a freely shareable template on Google Docs at http://bit.ly/musictechquestions.

To best support our students' music learning, our classrooms need to be safe spaces for exploring, creating, learning and expressing ourselves through music. Technology can be leveraged to support these goals both in and outside of school. While many schools prohibit personal technologies, such as cell phones and iPods, to be used by students in schools, these technologies present multiple possibilities for music learning and making inside and outside of school. In addition, many schools also prohibit access to social media sites such as YouTube and other online media providers. It is important to know the policies that exist in your school so that you can be an advocate for the music learning and expressive possibilities for your students.

Part of establishing a community is building relationships with building administrators, your colleague teachers, and the technology staff. A good place to start is to collaborate on a project with other teachers in your building. Your students can begin to use technology to create their own music for projects in other classes and compose school songs. Once you have small projects like these up and running and relationships with other teachers and staff, it can be easier to pursue larger technology integration projects within your classroom and throughout your school.

A Focus on DOING Music, Not Just Learning ABOUT Music

Many middle school general music courses can be classified as either advanced versions of elementary general music classes or diluted versions of introductory college music history or music appreciation classes. Courses that take these forms neglect to consider the unique nature and needs of adolescent students. Middle School students are neither advanced primary students nor are they premature college students. They are experiencing a unique developmental phase - a period of rapid maturing and growth –

physically, mentally, and socially. As such, the nature and scope of musical experiences for adolescents in schools should support and reflect these realities.

Nothing bores a middle school student more than another worksheet or quiz about a dead composer or doing a report on musical instruments. Even Internet-based research and Webquests lose their appeal quickly among students. More importantly, these technologically-mediated tasks are really information technology tasks applied in the context of learning *about* music, rather than engaging students directly in making, creating and responding to sound and music. These approaches to using web technologies do keep students occupied with online busy work, but how can we as teachers leverage technology in support of active, social music *making* – DOING music, rather than just learning ABOUT music. The curricular ideas shared throughout this chapter are specifically designed to share musical experiences where students are active music makers with and through technology.

Promoting and Sharing Student Work

When a technology-integrated music curriculum is organized around active music making, the sharing and celebration of students' music becomes essential. Technology can play an important role in the creation and making of music and in the documentation, reflection on and sharing of their musical works. One easy way to set this in motion is to establish an online site where student work is posted and published. This website can serve many purposes beyond an online public exhibition space. It also can be a highly visible vehicle to advocate for your music program. Your website becomes a forum for sharing the creativity and music learning happening in your classes with parents, administrators, and your community. This public advocacy work serves to raise awareness and build support for your programs.

A step beyond a music program website is to set up a curated blog where students have input in what gets shared and posted online. All content to be posted to the site can and should remain moderated by the teacher. However, by sharing responsibility for content, your students can engage in learning about professional presentation, copyright and permission issues, awareness of a broader audience for their works, and you can reinforce broader school-wide goals such as “writing across the curriculum” through creating new opportunities for student authored blog posts and commentaries on their music.

A further step is to create a school-wide online radio station and music label (Ruthmann, 2007) where students are responsible for the production, selection and promotion of their music online. In this model, student compositions can be published online through services like iTunes where there music resides next to all of the music they listen to. This can be an extremely motivating proposition for students. They begin to see their music as valued and themselves as musicians. The music shared online not only helps create community within the class that created the music, but it extends to the whole school and community, including future students who can listen to that music in future years.

Easing into Creative Music Projects

Many traditional music and non-music software programs (e.g., notation, sequencing, looping, audio editing, word processing) are based on the metaphor of a blank canvas or void. When the program is launched the user is presented with a blank slate upon which to place notes, audio waveforms, images or words. For many students, it can be intimidating starting from scratch. In my own teaching I have seen many students who have been reluctant to add their first notes to the page, sometimes wondering if they have anything of value to say. Of course they do have something valuable to say, but starting from scratch is not always the best place for them to begin. What follows are examples of how to use a variety of technologies to ease students into creating music for the first time.

Improvising with Generative Media Software

A new class of generative music software is emerging for both desktop computers and portable and hand-held computers. *Jam2jam* (http://www.savetodisc.net/jam2jam/) is a new family of collaborative media performance software where users interact together and networked online to improvise and perform with preexisting sets of music, images and video (Brown & Dillon, 2007; Dillon & Hirche, 2010). Designed for children and adolescents, users manipulate broad musical and visual parameters such as balance, filter, frame-rate, tempo, texture, and density in a live, interactive setting. If a user does not interact with the software, the preexisting sound and images stay in a repetitive state, which quickly gets boring. This "feature" motivates users to interact with the software to change its state and move the composition forward. Through exploration, users begin to discern finer and finer details about the music and gain more control of the performance environment. This software can also be used with acoustic instruments as part of a multimedia e-Jay performance.

Legendary music producer Brian Eno's *Bloom* and *Trope* apps for the iPhone and iPad are similar generative applications in that the user interacts with the software through touch to perform an improvised composition. Based on pre-programmed musical algorithms or sets of computational instructions, the software becomes an improvisational duet partner with the user, providing both visual and aural feedback. For students who are more hesitant to begin creating their own music, these apps can be a great way to begin to teach critical listening and improvisation skills, leading toward future work with more advanced applications.

Subtractive Processes: Learning from Sculpture

Another way of easing students into creating is to begin with experiences where students begin by critically listening and making adjustments to pre-existing musical material. Working with loop software (e.g., *Mixcraft* or *GarageBand*), I assign a musical sculpture project where students begin by crafting a solid "block" of musical sound (Figure 1.). Students explore and select 10-15 pre-existing loops and paint each of them into a separate track from time zero to 2 minutes. After saving the file, students switch seats with another class member and adopt a new sound block with which to work.

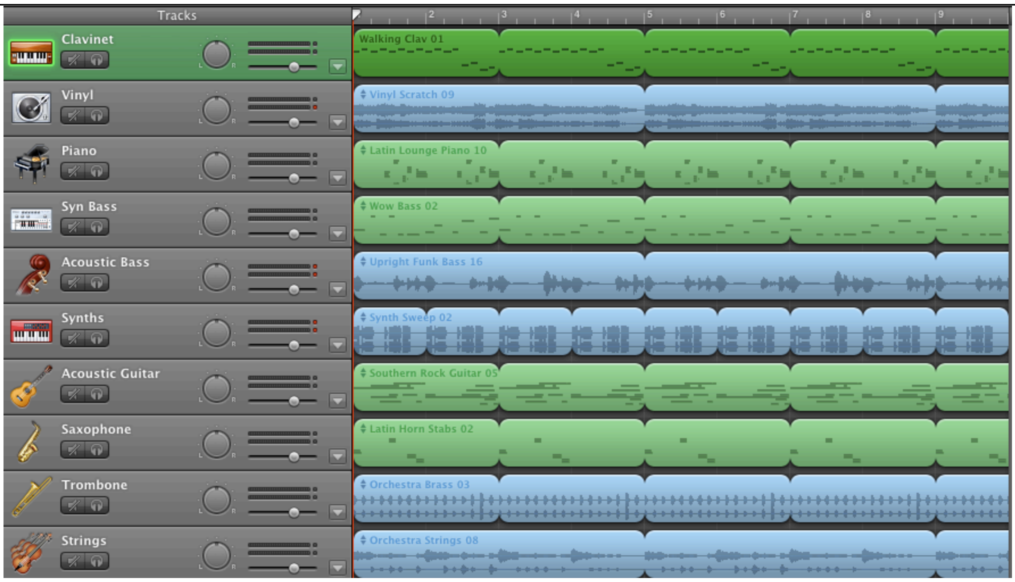


Figure 1. An unedited sound block in GarageBand.

Before moving on to the next step, I search Google and share examples of *bas relief* sculptures and discuss the artist's processes for revealing the final work through a subtractive process of slowly removing bits and pieces material from a solid block over time, revealing a final artwork. In *bas relief*, the artist can only transform or remove material. She cannot add it back. Following this model, students transfer that process to their newly found sound block in the loop software, exploring transformative functions like panning, balance, silence, and sound effects and subtractive elements such as slicing and removing portions of tracks creating variations in texture and a final form. This process attunes students to listen for subtle differences in the music when a small piece is removed or transformed. This focusing of critical listening, in turn, helps students better understand the building blocks of music they listen to.

Decoding and Remixing Musicians' Multitrack Audio Files

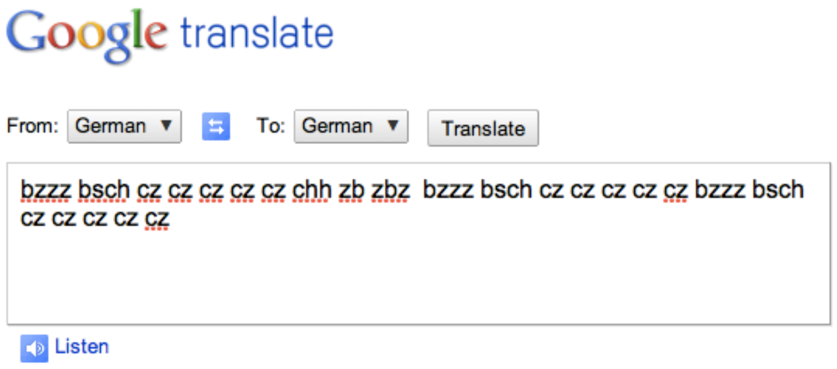
Another great place to begin with adolescents is to provide them opportunities to work directly with the music they might listen to and hear on the radio. The adolescents I work with want nothing more than to be able to remix and work directly with their favorite music. Until recently, educators have only had access to the basic stereo audio files released on CD or online. A new culture has emerged where artists are beginning to release multitrack recordings of their works for fans to remix and make their own. Though a variety of unauthorized multitracks float around the Internet, artists such as Lily Allen, David Bowie, Nine Inch Nails, and Radiohead have all released their music this way and are actively being used in music classrooms across the country.

Once these files are loaded into software such as *GarageBand*, *Logic* or *Mixcraft*, a good starting place is to listen and compare the original CD release to the multitrack recording. Encourage students to solo and mute various tracks as they listen to the multitrack. Using the pre-existing material, students can then remove, rearrange and add to the song. Students can also record their own singing or rapping and mix it with or replace the original.

Beat-Boxing with Google Translate and Audacity

It's amazing what can be done with a cheap microphone, simple audio editing software, and Google's online translation tools (http://translate.google.com/). Beat-boxing is a favorite activity among a number of adolescents I work with. Sometimes, however, students are hesitant to try beat-boxing in front of their peers and friends. Google's translate tools provide a safe, private way of exploring the techniques of beat-boxing without having to practice and share it aloud.

Google recently added computer-generated speech capabilities sonifying the text input of the translation box (Figure 2.). In the United States, the default setting is to translate from English. Changing the “From” and “To” languages to German results in consonant groupings being “spoken” similarly to beat-box sounds.



From: German

To: German

bzzz bsch cz cz cz cz cz chh zb zbz bzzz bsch cz cz cz cz cz bzzz bsch cz cz cz cz cz

Figure 2. Screenshot: Using Google Translate to Beat Box - http://translate.google.com/

Once some sample text is input into the box, the user can click on the “Listen” button to hear the faux computer-generated beat-boxing. The audio files generated by Google Translate can then be recorded into music software like *GarageBand* or an audio editor like *Audacity*, which is free and available for download to all common computing platforms. The computer-spoken files can then be cut up and arranged and integrated into larger compositions as a creative effect. Students who choose to begin beat-boxing projects this way often rehearse with and open up to sharing and recording their own voices in later projects. Beat-boxing and Hip-Hop culture in general is an engaging context for sampling, musical expression and working with the music of others (see Thibeault, 2010).

These example introductory projects help create safe community for musical exploration and sharing. Providing adolescent students the option to explore their voice and music privately behind headphones before sharing with a group, or having the option to begin with critical listening and arranging projects, rather than being presented with a blank canvas, can help diffuse students' anxiety to create their own music. As experienced musicians, it is often hard to remember what it was like to make music or explore a piece of music software for the first time, particularly during the turbulent adolescent years. Creating safe introductory experiences to creative musical work can set the stage for more social music making and exploration in later projects.

Mobile & Online Musicianship

New mobile technologies such as iPads, iPhones and iPod Touches are powerful portable music production tools. Unchained from the tether of power cables and computer mice, these devices can go anywhere and by design promote collaboration through their multi-touch interfaces. Since their launch, a wide variety of music making and production applications have been developed for this platform. Everything from digital emulations of harmonicas, xylophones, synthesizers and pianos, to multitrack audio recording software, to generative music composition tools, and beyond have been invented. Whereas the traditional computer is designed for one person to use at a time, many of the application designers have created programs that rely on social interactions to make and experience music.

Innovations in online social media have also influenced online music making and sharing technologies, building on and extending social relationships in real life to virtual and online spaces. While there is legitimate reason to be cautious in bringing these technologies into school settings, the reality is that social media is here and an integral part of our students lives. Students use it to communicate and collaborate outside of school. These same affordances can be leveraged in safe and beneficial ways inside music classrooms.

iPad & iPhone Ensembles

One of the most promising areas of exploration for adolescent music classes is in using iPads, iPhones, and iPods Touches as instruments in small ensembles. Recent pilot projects with middle school students have involved using iPads and iPods in like and mixed-instrument ensembles for both covering songs and creating original compositions. Modeled after the first iBand power trio from Austria (http://iband.at), students begin by finding tunes they know on acoustic instrument emulation applications (e.g., *Virtuoso Piano* or *Band*). Once they've found a tune they like, they teach it to the other members of the group. From there, students work together to create, rehearse and perform an original arrangement of the tune using only iPod or iPad applications. Follow up projects include integrating instruments the students play and other classroom instruments as desired into either another cover tune or an original composition. After a basic experience covering a song, students are encouraged to explore other musical applications and submit ideas for future projects.

One of the challenges of this project is how to amplify the sound of each iPod or iPad so that the other students can hear. One solution is to have each iPod and iPad connected to headphones and a series of headphone splitters. Another solution is to plug the inputs of each device into an audio mixer which mixes and amplifies the audio to an external set of speakers. The mixer/speaker solution is particularly useful when students want to perform using their own or the classroom instruments.

iPads, iPhones and other smartphones often come equipped with audio and image recording capabilities. These can be leveraged in the music classroom as tools for conducting portable field recordings, capturing images and sounds to be used in multimedia projects, and for conducting composer interviews and commentaries on their music (Ruthmann, 2007).

Social Music Notation: Composing Music Together using Noteflight

Innovations in online Internet technologies have led to the launch of the social music creation and collaboration platforms Indaba Music and Noteflight.com. Noteflight is a free, fully-featured online platform for social music notation. Students and teachers can create free accounts and notate and share music they create alone or together. What differentiates Noteflight from other music notation programs is that it is a software-as-service delivered to the user through an Internet browser, similar to Google Documents. Users can share their scores and enable others to view and even contribute to them. Noteflight also periodically saves your progress and keeps track of different versions of your compositions, which can be helpful for tracking individual contributions to collaborative notation projects.

A recent research project documented adolescent students' initial explorations of Noteflight in a classroom setting (Ruthmann & Dillon, in press). The project began by sharing a contemporary piece of art (Figure 3.) and asking students to work in groups to create an original composition in Noteflight that expressed that artwork. This project was conducted with students who had not been formally introduced to music notation in class.

The goal of this project was to see what students knew and were able to express in music through notation, supported by a social, online environment.

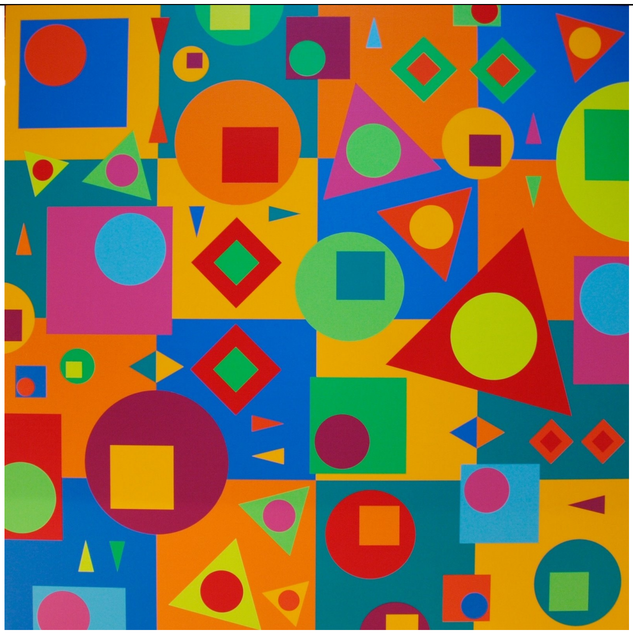


Figure 3. Geoshape 6, by Frank Jonas

The student compositions that emerged showed a much greater depth of musical understanding among the students than the teacher had expected. Through starting with an open-ended composing experience translating a piece of art to a notated composition, the students had the opportunity to bring their musical ideas to the project, rather than work within a fixed set of parameters. As a result, each of the group compositions showed a complex understanding of form, melodic structure, and tension and release. The complexity of the students' compositions caused the teacher to develop more advanced follow-up projects building on the strengths and weaknesses shown in part by the composing project. Example compositions from the project can be viewed online at http://web.me.com/prhsfapa/Site/Art-Infused_Composing.html.

The Music of Lowell: A Digital Ethnomusicology Project

A current, ongoing project integrates iPads, Noteflight and Audacity as part of a community-based digital ethnomusicology project. Inspired by a found music composition posted to Noteflight and the ethnomusicological processes of Australian composer Percy Grainger (see Ruthmann & Hebert, in press), middle school general music students and a class of college music education majors are applying digital ethnomusicology to documenting the traditional music of immigrant families in Lowell, Massachusetts. Using the audio recording and portability features of iPads, middle school students working side by side with college students are digitally recording the songs and stories of students' parents and grandparents in the community. These recordings are downloaded to computers, edited in Audacity, and translated into English by the middle school students and community members. Transcriptions of the songs are then notated in Noteflight and are posted to a community website, along with the edited audio recordings and translated stories.

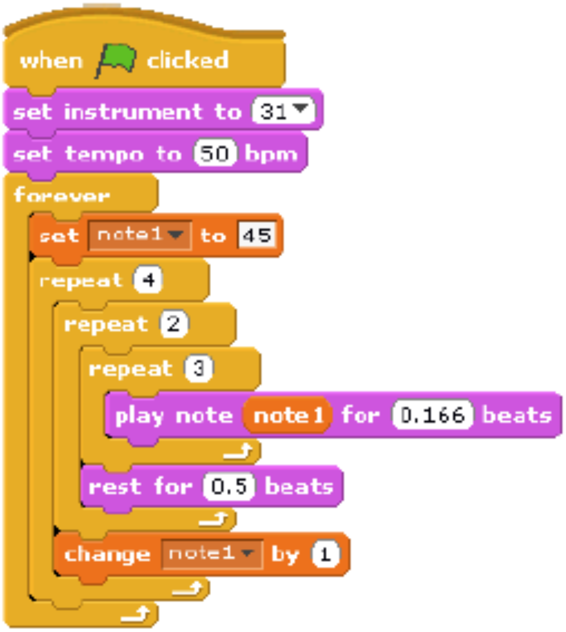
After gathering all of this material, the middle school students begin a digital remixing project using the recorded and transcribed songs and stories as musical source material for original compositions and remixes. This process enables students to integrate aspects of their musical culture with that of their parents and grandparents. When the project is complete, the student remixes will be posted back to the community web site, to be shared alongside the originally recorded source material.

Increasingly powerful and portable mobile computers, and an emerging sophistication of online social music software are affording new and exciting possibilities for technology-infused music experiences in schools. Though mobile technologies such as iPhones and iPads are currently quite expensive, the possibilities afforded by only one or two iPads or iPod Touches in a music classroom are worth exploration. Over time, these costs will come down and the tools students will have access to these tools at school and at home.

Remixing Games, Music and Video

Creating and Remixing Music and Games

A tool that many middle school technology and media specialists are beginning to use in their work with adolescent students is the Scratch visual programming environment (http://scratch.mit.edu). Originally designed to adapt the metaphor of "scratching" and remixing of samples of sound in Hip-Hop music to teaching computer programming and math, Scratch enables students to remix, share and create original interactive games, stories and other programs. A quick visit to the Scratch website will reveal a number of existing games and musical applications programmed by middle school students from around the world. A key feature of this site and software is that all programs can be downloaded and portions remixed and adapted into new games. A companion site for teachers (http://scratched.media.mit.edu) presents free lesson plans and support for beginning projects in Scratch, including integrating music.



```
when ⚑ clicked
set instrument to 31▾
set tempo to 50 bpm
forever
  set note1▾ to 45
  repeat 4
    repeat 2
      repeat 3
        play note note1 for 0.166 beats
      rest for 0.5 beats
    change note1▾ by 1
```

Figure 4. A musical algorithm represented in Scratch.

Scratch has nearly unlimited potential for music projects, especially in the realm of composing music for games. Students can embed digital audio files they record or edit in *Audacity*, or code original MIDI music using algorithms similar to those underpinning *jam2jam*, *Bloom* and *Trope*. Some of our current work with middle school students and Scratch centers around creating music that responds to video game play. Learning experiences include creating algorithms for mapping musical tempo and dynamics to character movement through a game level. If you ask around, you might find that students and teachers are already using this software at your school.

Found and Recorded Video Remixes

With the advent of social media sites like YouTube, entire new worlds of musical practice and musicianship emerge every day. The video/musical artist Kutiman's Thru-

You project (http://thru-you.org/) is a sophisticated example of the new social media musicianship that's emerging online. Music videos uploaded to YouTube serve as Kutiman's sonic and video palette. As a musician, he scours YouTube for musical snippets, which he then orchestrates into entirely new compositions using video editing software. Rather than working exclusively with audio, clips from videos serve as the musical material.

A Norwegian "amateur" musician, Lasse Gjertsen, takes a related approach to creating his own music. Self-professing to not being able to play a traditional musical instrument, he sets up a video camera and records himself playing sounds on a drum set and individual keys on the piano. He then edits these videos and isolates each pitch and drum sound. Again, using video editing software, he arranges his edited clips into complex musical compositions and arrangements.

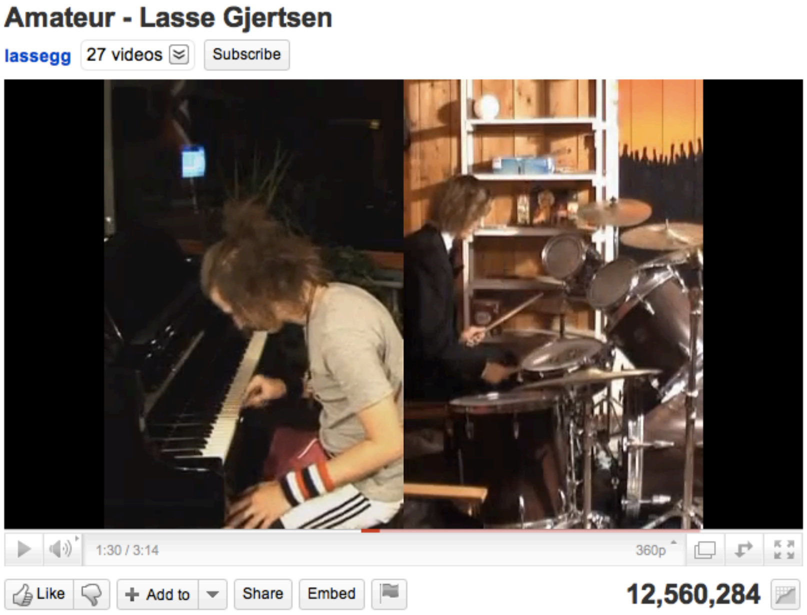


Figure 5. *View online at http://www.youtube.com/watch?v=JzqumbhfxRo*

Projects like these can be accomplished by middle school students using a computer with a web cam or a portable video camera. One way to proceed is to ask students to record pitched and non-pitched sounds found around the music classroom or school via video camera. The recorded video file can then be split up by the students in video editing software (e.g., *iMovie* or *Windows Movie Maker*) isolating each recorded sound. From there, students can create or cover a melody, rhythmic pattern, or entire composition. By switching the composing environment from audio software to video software, students have to rely on and develop their aural and visual skills to accurately realize rhythms and melodies.

Coda

Music is no longer just an aural experience, if it ever was. This is especially so for today's students. Music, image, video and technology are inseparably linked together in the lived experiences of our students. Technology defines their music as much as they perceive their music to define them. As such, projects that connect to and draw upon students' youth culture have the potential to be engaging and relevant in schools. The starting place for engaging music technology projects begins with the building of relationships between ourselves as teachers, our students, their culture, our schools and community.

Because policy often lags behind possibility, teachers may initially run into barriers using technologies as suggested in this chapter, particularly those that call upon the open use of the Internet to gather musical resources or the use of iPods in the classroom. However, I encourage you to first adopt some of these technologies personally, and figure out how to best integrate them with your students in your school and community contexts. Each school and set of students is different and music technologies should be selected based on the needs of your curriculum and students.

Technology is more than just a tool; it's a platform for creating, sharing, interacting, performing, assessing and reflecting on the music we and our students make together. Through making time to get to know and understand our students' musical and technological lives, we can better design music learning experiences that are culturally relevant to our students. When technologies are chosen and learning experiences designed within this framework, technology can be used in support of meaningful musical engagement in middle school music classrooms.

Selected Music and Media Applications for iPad/iPhone/iPod Touch

Band – A virtual drum set, guitar, bass, and keyboard.
Bebot – Theremin-like touch based synthesizer.
Bloom – An interactive, generative music environment composed by Brian Eno.
GraphMusic – An interactive music drawing program.
Reactable – Easy to use, but feature-rich touch-based audio exploration environment.
Singing Fingers – Records sounds to drawings, which can be performed via touch.
Sound Cloud – Application for audio playback and recording directly from/to an online file
SoundDrop – A music and physics composition environment driven by falling orbs.
Sound Thingie – Drawing based loop software.
Space Oddity – Interactive multitrack remix recording of David Bowie's song *Space Oddity.*
Talking Carl – Interactive robot which parrots back audio input and plays sound effects.
Trope – An interactive, generative music environment composed by Brian Eno.
Virtuoso Piano – A social piano application for multiple performers.

Recommended Online Resources

MusicPLN: Professional Learning Network for Music Educators – http://www.musicpln.org/. The MusicPLN is a network of music teachers from around the world, primarily from the US sharing best practices for learning and teaching music with technology.

Phil Kirkman's Blog – http://www.kirki.co.uk/. Phil's blog presents a perspective on music education and technology from the United Kingdom.

Jam2Jam: Collaborative Media Performance Software– http://www.savetodisc.net/jam2jam/. Jam2Jam is free software for improvising and performing collaboratively with images, video, and sound.

Scratch – http://scratch.mit.edu/ & ScratchEd – http://scratched.media.mit.edu/. Scratch is a free software for programming original video games, interactive software and for composing music. ScratchEd is an online space for collaboratively sharing educational resources around Scratch.

Music Resources for Scratch – http://www.scratchmusic.org/. This website is a collection of music-specific resources for working in and with Scratch.

SoundCloud – http://www.soundcloud.com/. This website is a free audio storage, sharing and social commenting website.

Audacity – http://audacity.sourceforge.net/. Audacity is free audio editing, analysis and production software.

Recommended Further Reading

Brown, A. (2007). *Computers in Music Education: Amplifying Musicality*. New York: Routledge.

Finney, J., & Burnard, P. (Eds.). (2007). *Music Education with Digital Technology*. London: Continuum Press.

Ruthmann, A. (2007). The composers' workshop: An emergent approach to composing in the classroom. *Music Educators Journal, 93*(4), 38-44.

Thibeault, M. (2010). Hip-hop, digital media, and the changing face of music education. *General Music Today*, *24*, 46-49.

Watson, S. (in press). *Using Technology to Unlock Musical Creativity.* New York: Oxford University Press.

References

Brown, A., & Dillon, S. (2007). Networked improvisational musical environments: Learning through online collaborative music making. In J. Finney & P. Burnard (Eds.), *Music Education with Digital Technology* (pp. 96-106). London: Continuum.

Dillon, S., & Hirche. K. (2010). Navigating technological contexts and experience design in music education. In J. Ballantyne (Ed.), *Navigating Music and Sound Education* (pp.173-190). Newcastle upon Tyne: Cambridge Scholars Publishing.

Ruthmann, A. & Dillon, S. (in press). Technology in the lives and schools of adolescents. In G. McPherson & G. Welch (Eds.), *Oxford Handbook of Music Education*. New York: Oxford University Press.

Ruthmann, A. & Hebert, D. (in press). Music learning and new media in virtual and online environments. In G. McPherson & G. Welch (Eds.), *Oxford Handbook of Music Education*. New York: Oxford University Press.

Thibeault, M. (2010). Hip-hop, digital media, and the changing face of music education. *General Music Today*, *24*, 46-49.

CHAPTER 8 EXPLORING NEW MEDIA MUSICALLY AND CREATIVELY

INTRODUCTION

Innovations in new media and ICT are inspiring children in their creative play at home and in schools (Somekh, 2007). Today's computers, both desktop and mobile, can now be considered musical instruments in and of themselves (Ruthmann & Dillon, 2012; Thibeault, 2012), and are inspiring new practices that integrate sound, image, touch, and video as the medium and method of musical expression[1]. These practices parallel the emerging trend that today's youth are now more often going to You Tube to find and listen to music, rather than listening to mp3s and CDs[2]. With the growing prevalence of mobile devices like iPhones, iPads, and smart phones, our children now have access to powerful music making applications and new models of creative musicianship *at their fingertips*, providing new avenues for creative engagement within and outside of our music classrooms.

This chapter provides an introduction to projects and tools for exploring the creative dimensions of new media with primary pupils. I begin with an introduction to creative musicianship with new media, followed by an overview of tools for creating and being creative with new media. These tools are discussed in the context of practical projects and creative strategies for teacher and pupil exploration within primary classrooms. The first project I share focuses on a mobile app for Apple's iDevices[3] - the application *Singing Fingers*[4], which enables children to "finger paint in sound" connecting physical gesture, drawing, and sound. The second project focuses on creative performing, improvising and composing experiences with the Scratch multimedia programming environment[5] created by the Lifelong Kindergarten Group[6] at the MIT Media Lab.

[1] See Kutiman's 'Mother of all funk chords' - http://www.youtube.com/watch?v=tprMEs-zfQA, Lasse Gjertsen's 'Amateur' - http://www.youtube.com/watch?v=JzqumbhfxRo, and Pomplamoose's 'Hail Mary' – http://www.youtube.com/watch?v=fYy2p_0DVMU.

[2] Smith, E. (August 14, 2012). Forget CDs: Teens are tuning into You Tube. Wall Street Journal. Retrieved from http://online.wsj.com/article/SB10000872396390444042704577587570410556212.html.

[3] iPad, iPhone, and iPod Touch.

[4] http://singingfingers.com/

[5] http://scratch.mit.edu/

WHAT IS NEW MEDIA MUSICIANSHIP?

New media musicianship is a broad collection of creative musical practices where video, images and sounds are interactively used as the medium for musical expression. Today, artists such as Kutiman[7] browse YouTube for videos and remix them using video-editors into completely new musical compositions. The band Pomplamoose[8] has pioneered the genre of the *video song* where all layers of the musical texture recorded in their songs appears via split screen technology in the video, providing a window into contemporary music recording production techniques. Choral composer Eric Whitacre invites and *crowdsources* singers from around the world to submit videos of their singing, which are stitched together into his multimedia Virtual Choir[9] performances. One of the common aspects across these three examples is that music, image and sound are present together in both the final product and throughout the creating process.

The innovative practices and technologies described and linked to throughout this chapter provide a glimpse into ways children use these technologies to make music with media they find culturally valuable and relevant (Ruthmann & Dillon, 2012). Children can more easily take on the musical roles of curator, video editor, producer, re-mixer and programmer of musical media with these technologies. New web-based and mobile tools open access to children's music making experiences using interactive webpages running on a computer or mobile device they have at home or at school. The simplicity of not having to download expensive and complicated music production software affords children and their teachers access to musically complex creative media making experiences producing digital stories, creating animations, designing musical instruments, video games, and interactive soundtracks. For example, the Scratch visual programming environment[10] is home to over 2.6 million child-created interactive multimedia projects; over 500,000 of them include creative use of music and sound.

[6] http://llk.media.mit.edu/

[7] http://thru-you.com/

[8] http://pomplamoose.com/

[9] http://ericwhitacre.com/the-virtual-choir

[10] http://scratch.mit.edu/

The tools for exploring new media musicianship can be sub-categorised into tools for *social media musicianship*, *video-edited musicianship*, *tangible media musicianship* and *computational media musicianship*. Social media musicianship tools include blogs, wikis and social networking sites where music is discussed, shared and collaboratively created, such as online music notation platform Noteflight.com and audio remixing platform Indaba.com.

Video-edited musicianship tools are online and mobile apps where children can remix and edit videos as the musical medium. YouTube's interactive video editor, Smule's *MadPad*[11] application for mobile music video sampling, and interactive websites like InBflat.net provide children access to basic creative video editing in musical contexts within internet browsers and on their mobile devices.

Tangible media musicianship tools harness children's touch and gesture as the central method of exploring and being curious with musical media. Examples of these technologies include the iPod and iPad app *Singing Fingers* (discussed later in this chapter), the interactive Sifteo Cube[12] game environment, and the MaKey MaKey[13] sensor interface, where children can easily turn the physical environment around them into a musical instrument.

Computational media musicianship tools engage children in making music through computational, mathematical and pattern-based processes. The Scratch multimedia programming environment, as well as the interactive musical media websites Bohemian Rhapsichord[14] and Bangarang Boomerang[15] engage children in building, designing, remixing and making their own musical instruments, games, and music.

The common thread across all of these categories of new media musicianship is that children are new producers, designers and creators of musical environments, in addition to the traditional roles of performer and listener. These tools provide opportunities for students to use their voices, sounds and the environment around them as musical instruments inside our classrooms and also at home. It is incumbent upon us, as teachers, to encourage and study these practices and develop new pedagogical strategies

[11] http://www.smule.com/MadPad
[12] http://www.sifteo.com/
[13] http://www.makeymakey.com/
[14] http://bohemianrhapsichord.com/
[15] http://static.echonest.com/BeatDriver/

that foster our children's musical agency and learning. I encourage you to take plenty of time to explore the linked websites, tools, and videos yourself to experience first hand these new tools. If our children have access to these tools and are making music with them in their free time, it is our responsibility to get to know them and make space for them within our classrooms.

CREATIVE STRATEGIES FOR SINGING FINGERS

Singing Fingers is an interactive media application that allows young children to "finger paint" and play with sound. When the application is loaded, the user is presented with a blank drawing surface. The user simply sings or makes sounds while touching and drawing along the surface of the mobile device (see Figure 1). When you have finished singing and drawing, the application automatically turns into playback mode, where the user can retrace their drawing, performing back the drawing they recorded. The user can touch or trace the drawing forward, backwards, or anywhere colours are drawn on the screen. If you do not like what you've recorded, you can wipe the screen and start again. And, if you really like your creation, you can save it and share it for playback and interaction at a later time.

Drawings made with Singing Fingers are also designed to be performed via multi-touch. This feature enables you to play chords with your drawings, or even a multi-part drum kit if you record non-pitched sounds. Melodic sounds are mapped to the colours of the rainbow, which correspond to each note of the chromatic scale. For example, a note in any octave is always the same colour. Non-pitched sounds show up in various shades of grey. This visual and musical design distinction opens up many possibilities for creative visual, musical and gestural expressions with Singing Fingers.

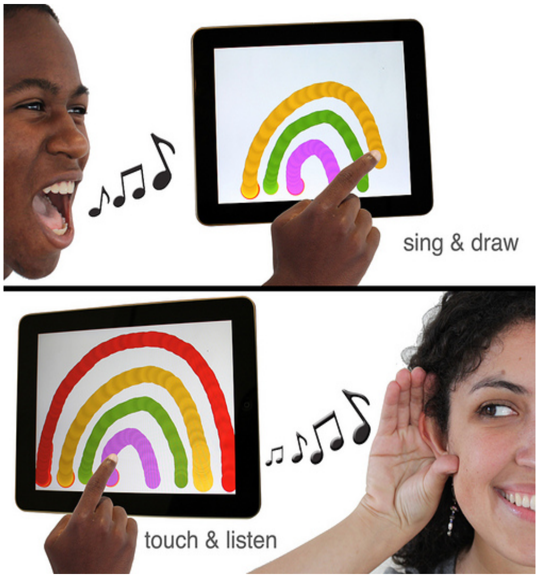


Figure 1. *Singing Fingers* – http://www.singingfingers.com/.

Fingerpainting in Sound

Activity 8.1: Fingerpainting Your First Music with Singing Fingers

1. Invite children to sing a song they know.
2. Using one iPad, invite the class to sing a phrase of the song and select a pupil to draw while the class sings the song.
3. Invite children to come up and try different ways of performing the drawing. What happens when you trace it fast? What happens when you trace it slow? Forwards? Backwards? Stopping and starting?
4. Ask the children if they can figure out why certain notes are different colours. Do they notice a relationship? Can they explain the reason behind it?
5. Clear the screen and draw a staircase while singing an ascending scale. Each step of the staircase should display as a different colour. Lead the children in the song again and ask them to think of the first note. Invite a confident pupil to find that note on the staircase. Work with that pupil and the whole class in finding and performing the melody of the song they just sang on the drawn staircase.

6. Invite children to come up and perform parts of the melody while the whole class sings.

Activity 8.1 is designed as an orienting, whole class introduction to *Singing Fingers* assuming that you only have one mobile device available. As the teacher, you have the opportunity to model not only how *Singing Fingers* works, but also to model and introduce creative thinking and performance of the drawing. Because *Singing Fingers* is also a visual drawing tool, it is ideal for experimenting with and reinforcing concepts such as pitch relationships, rhythm, timbre, tempo and texture, many of which are visual and gestural metaphors for sound (higher, lower, shorter, longer, slower, faster, etc.).

In addition to using *Singing Fingers* to record and perform melodies, it is great for recording and performing percussive sounds and rhythmic ostinati. A possible extension to Activity 8.1 is to record vocal percussive sounds emulating a drum kit. Invite children to think of sounds and record and draw them on *Singing Fingers*. You with the pupils could then perform a rhythmic accompaniment to the song you sang using Singing Fingers.

Another strategy that is useful with primary pupils is to pre-record and draw various musical examples. Sometimes it is easier for younger pupils to work with drawings that have already be created, rather than go through the process of creating the drawing and then performing it. An idea you might consider is to record melodies that have different pitch contours on the *Singing Fingers* screen:

- Record and draw a melody that ascends
- Record and draw a melody the descends
- Record and draw a melody that ascends and descends
- Record a melody using only one pitch

At this point, you can invite children to look at the four straight lines you drew and recorded. Ask them if they see any visual differences between them. Before you perform them, ask them to hypothesise what they think the lines will sound like when traced. Invite the children to trace the lines and test their hypotheses about the melodic contour. What happens to the melodic contour when you trace the line backwards? *Singing*

Fingers can be a wonderful platform for musical problem solving[16], hypothesising, and exploring one's curiosities[17] in music and sound.

Singing Fingers provides an environment where children can be creative and explore the musical possibilities of their own voice. It is a space where children can make their musical ideas visual, audible and tangible, and easily share them with others. As a teacher, having access to children's musical ideas across multiple forms of representation provides additional insight into their ideas, challenges and learning process. The opportunity to learn and experiment with music and gesture through exploratory play can lead to deeper creative engagement with music.

BEYOND JUST PERFORMING

With Singing Fingers, children can explore multiple musical roles. The most obvious role is that of the *performer* and *curator* of the sounds that accompany THE drawing. Children either generate the sound themselves through singing, speaking, or making other types of sounds, or they can have others create the sounds while they draw. Once the drawing is on the screen, pupils take on the role of *improviser*, exploring and experimenting with the drawing, looking for interesting sounds and patterns. Once the drawing is set, it becomes a platform for *composing* and *arranging*, either alone by the pupil, or as part of a shared duet or trio small ensemble with other friends. The drawings can be saved for use by other children in the class or in other classes.

Once you and your pupils have had some beginning experiences with Singing Fingers, invite them to create their own free drawings and compositions[18]. For older, more experienced pupils, it may be easier to share mobile devices and integrate them into small group work. Singing Fingers works well in duets or trios of children, as they can all interact with the drawings. Singing Fingers also lends itself to integration with classroom singing, collaborative performance with other classroom instruments, and improvisatory

[16] For more practical ideas exploring musical problem solving in primary music education, see Wiggins, J. (2010). *Teaching for Musical Understanding* (2nd edition). Rochester, MI: Center for Applied Research in Musical Understanding.
[17] For practical strategies facilitating children's musical curiosities, see Greher & Ruthmann's (2012) summarization of the work of Jeanne Bamberger.
[18] If you have multiple mobile devices, it is helpful to have several headphone splitters and headphones to keep the sound levels low enough for optimal recording.

works. Pupils can all start from the same starting picture, or can be asked to create their own, based on their level of comfort and experience with music and Singing Fingers.

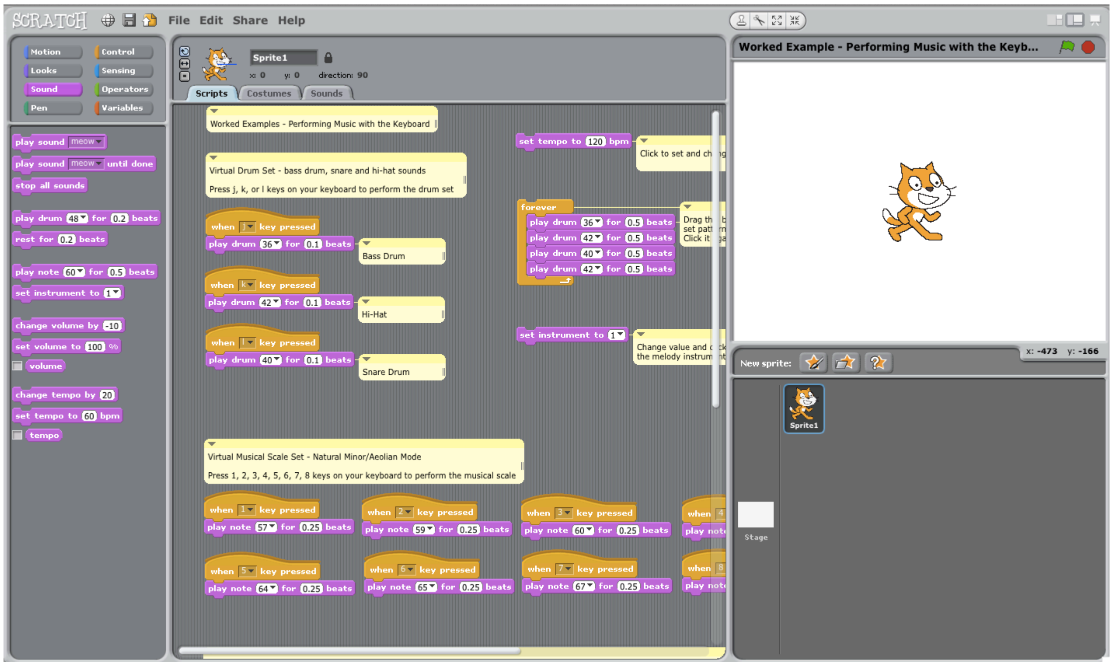
Activity 8.2: Design Your Own Instrument

1. Invite children to imagine an original musical instrument. To help get their ideas flowing, ask them: If you could design you own personal musical instrument, what sounds would it make? How would you play it? What would it look like? Provide blank sheets of paper and coloured markers for pupils to explore and draft their ideas. This is useful if you have a limited number of mobile devices with Singing Fingers available.
2. Invite children to create, draw and record their instrument into Singing Fingers. If you have multiple mobile devices, this can be an individual or partner project. If you only have one, or a few, you can make this a whole class or small group project.
3. Once children have created their instruments, give them time to explore it and to make changes. As with any musical instrument, it takes time and practice to master it. Be sure to provide enough time so that pupils can explore the creative possibilities of their instruments.
4. Invite children to create an original composition with their Singing Fingers instrument and rehearse it.
5. Invite pupils to present their instrument to the class and give them time to perform the composition they created with their Singing Fingers instrument. Be sure to have them save their drawn instrument for archiving.

CREATIVE STRATEGIES FOR USING SCRATCH

Scratch is a free, visual programming environment created by the Lifelong Kindergarten Group at the MIT Media Lab[19]. This environment was created as a means for children to "remix" and "play" with computer code in the same way that DJs and turntablists remix, play with, and recombine musical samples and drumbeats. Also inspired by LEGO building blocks, children are asked to *imagine, program, and share* blocks of computer code, creating simple to very complex interactive computer programs and animations as a result. Rather than just being *consumers* of technology and games, Scratch enables children to be the *designers*, *producers*, and *creators* of the games. Additionally, the Scratch website serves as an online community of practice for both Scratch users and teachers to upload, discuss, and share the projects they create.

Kafai, Peppler, and Chapman (2009) profiled children's use of Scratch in afterschool *computer clubhouses* in greater Boston, Massachusetts. Their work documents how Scratch and the clubhouse teachers and environment inspired children to think of themselves as competent, creative and critical thinkers and learners. In the clubhouses, this was accomplished through a approach to the design of Scratch and the children's experiences where they actively build and design the tools with which they work.



```
play sound meow
play sound meow until done
stop all sounds

play drum 48 for 0.2 beats
rest for 0.2 beats

play note 60 for 0.5 beats
set instrument to 1

change volume by -10
set volume to 100 %
volume

change tempo by 20
set tempo to 60 bpm
tempo
```

```
Worked Examples - Performing Music with the Keyboard

Virtual Drum Set - bass drum, snare and hi-hat sounds
Press j, k, or l keys on your keyboard to perform the drum set

when j key pressed
play drum 36 for 0.1 beats      Bass Drum

when k key pressed
play drum 42 for 0.1 beats      Hi-Hat

when l key pressed
play drum 40 for 0.1 beats      Snare Drum

set tempo to 120 bpm      Click to set and ch

forever
  play drum 36 for 0.5 beats      Drag th b
  play drum 42 for 0.5 beats      set patt rn
  play drum 40 for 0.5 beats      Click it ga
  play drum 42 for 0.5 beats

set instrument to 1      Change value and c ck
                         the melody instrum nt

Virtual Musical Scale Set - Natural Minor/Aeolian Mode
Press 1, 2, 3, 4, 5, 6, 7, 8 keys on your keyboard to perform the musical scale

when 1 key pressed            when 2 key pressed            when 3 key pressed            when 4
play note 57 for 0.25 beats   play note 59 for 0.25 beats   play note 60 for 0.25 beats   play not

when 5 key pressed            when 6 key pressed            when 7 key pressed            when 8
play note 64 for 0.25 beats   play note 65 for 0.25 beats   play note 67 for 0.25 beats   play not
```

[19] Scratch can be accessed at http://scratch.mit.edu/.

Figure 2. *Screenshot from Scratch 1.4 – http://scratch.mit.edu/.*

Scratch is designed so that primary school age children can easily enter into computer programming experiences, while providing enough complexity to enable the creation of sophisticated projects by motivated pupils. This is, in part, accomplished through careful design and by putting the processes of play and remixing at the centre of the learning experience.

While Scratch is a general purpose programming environment designed for kids, it has a strong set of musical and sonic capabilities[20]. The music and sound blocks (see left side of Figure 3 on previous page) enable children to work directly with musical concepts such as pitch, melody, duration, rhythm, tempo, tone colour, form, texture, and volume, among others. Children can record in and edit their own sounds and sound effects, or they can create drumbeats and musical riffs, sequences and songs that accompany and interact with their animations and computer games. As a result, Scratch is a sophisticated, yet accessible, platform for musical and computational play that can foster creative expression among primary school age children.

GETTING INTO SCRATCH

With the metaphor of "remix" at the centre of the design of Scratch, children are encouraged to first visit the Scratch website - http://scratch.mit.edu - to browse and play one of the over 2.6 million projects created by kids across the world. Once children find a project or game that they like, they can simply click a button to literally "see inside" the project, which exposes all of the programming code, animations, sounds, and images that were used in the creation of the project.

A simple introduction to working with Scratch is for pupils to replace the images and sounds of the main objects (called "Sprites" in Scratch) in one of the Scratch games with their own, using Scratch's built-in image and sound editors. Pupils who are more

[20] For more examples of projects and information on teaching music with Scratch, visit http://scratchmusicprojects.com and http://performamatics.org/.

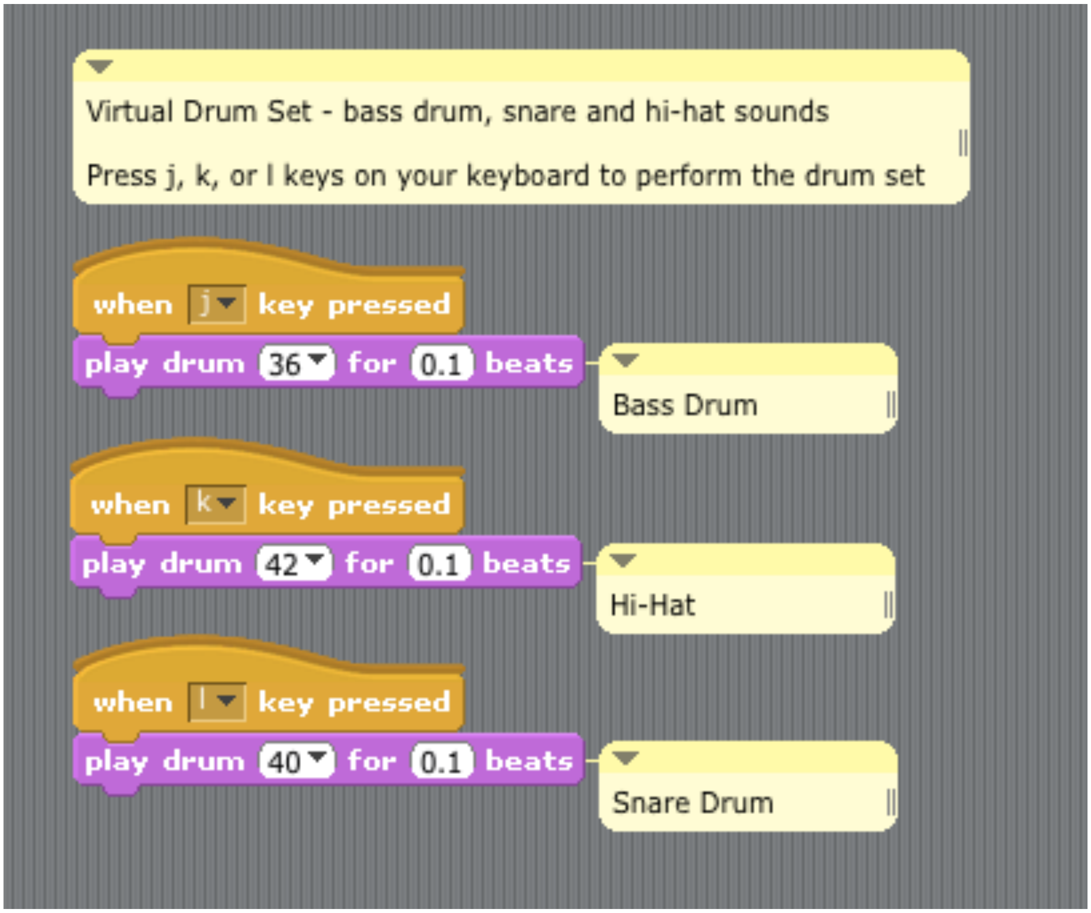
experienced with Scratch can further play with, explore, and remix the inner code to any project as a way to learn how it was created or to make it their own. Though one can begin with a blank project and start from the ground up, this can be intimidating to pupils (and teachers!) who are new to computer programming. Starting in the "middle" by remixing existing programs is often the most accessible entry point for primary age pupils.

There are many tutorials and related resources for getting started with Scratch available online. Many of these can be found on social media sites, such as YouTube and on the educators' page for Scratch at http://scratched.media.mit.edu/. I encourage you to search out introductory tutorial videos to watch and also to connect with other primary educators working with Scratch on the ScratchEd site as you explore the project in this section of the chapter.

MAKING MUSIC WITH SCRATCH: PERFORMING WITH A KEYBOARD

The project I am sharing with you starts with the principles of creative musical play and sound exploration. To begin with, download and open the example Scratch program - ScratchMusicKeyboard.sb[21] - in your version of Scratch. In this project, we're starting with a small, working program that turns your computer keyboard into an interactive drumset and melody player. You can choose to start from the example program, or open a new Scratch file and duplicate the code, as shown in Figure 4.

[21] You can download the ScratchMusicKeyboard.sb file that accompanies this chapter at http://bit.ly/scratchmusickeyboard.



```
Virtual Drum Set - bass drum, snare and hi-hat sounds

Press j, k, or l keys on your keyboard to perform the drum set

when [j] key pressed
play drum (36) for (0.1) beats        Bass Drum

when [k] key pressed
play drum (42) for (0.1) beats        Hi-Hat

when [l] key pressed
play drum (40) for (0.1) beats        Snare Drum
```

Figure 3. *Code for a keyboard drum kit in Scratch*.

When you've downloaded opened the ScratchMusic.sb file in Scratch, take a moment to view the first set of blocks (see Figure 4 on the previous page) and press the 'j', 'k', and 'l' keys on your computer keyboard. If your volume is turned up, you should hear the three basic drum set sounds of a bass drum when you press the 'j' key, a hi-hat when you press the 'k' key and a snare drum when you press the 'l' key. This simple set of Scratch blocks maps the computer keyboard keys to perform three drum sounds when the keys are pressed. In Scratch, the blocks can be *read* similarly to regular language. For example, you can convert the top set of two blocks into the following statement:

When the 'k' key is pressed,
Play drum sound '36' for '0.1' beats.

Take a couple of minutes to play around with those keys and drum sounds. Can you find and perform a drumbeat?

The yellow blocks in Scratch are referred to as “event” blocks. In this case, they are “listening” for when a specific key on the computer keyboard is pressed. If it is pressed, Scratch detects that *event* and does what the next block in line tells it to. In this case, each of the three “when ____ key pressed” blocks is connected to a “play drum” block. Whenever one of the specified keys is pressed, the “play drum” block plays the sound of the specified drum.

In the ScratchMusicKeyboard.sb project file, the ‘j,’ ‘k,’ and ‘l’ keys are assigned to the drum sounds and the drum instrument values of ’36’ (bass drum), ’42’ (hi-hat), and ’40’ (snare drum). Other drum sounds can be selected by clicking on the black triangle to the left of each number and changing the values in the “play drum” block. This action brings up a list of many different drum sounds that can be used and assigned to various keys on the keyboard. Additionally, any key on the keyboard can be mapped to any sound in Scratch.

REMIXING THE SCRATCH DRUMSET

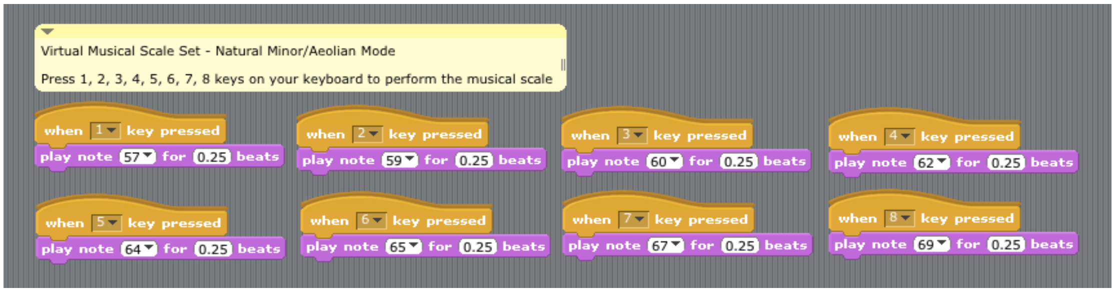
While it is possible to create interesting drum rhythms using just a bass drum, hi-hat and snare, encourage your pupils to explore additional sounds by remixing and duplicating the existing ScratchMusicKeyboard.sb code. Pupils are also not limited to only three drum set sounds at a time. They can create additional sounds by dragging a new yellow “when ____ key pressed” block to screen and connecting a new purple “play drum” block. By repeating this process, pupils can build an entire percussion orchestra in Scratch for their own creative play and performance.

This example harnesses the interactive aspects of computer programming. Computers take input from various devices (such as mice, touch events, and keys) and put those actions into motion to do various things, in this case to play musical sounds.

CREATING AND EXPLORING RIFFS, PATTERNS AND MELODIES

Just like with drum sounds, melodic pitches can be also mapped to keys on the computer program. Take a look at the example code a bit further down on the screen in the ScratchMusicKeyboard.sb example. In this example, the number keys on the computer keyboard are mapped to the pitches of a minor scale. Take a moment to press the numbers 1 through 8 on your computer keyboard and listen to the result.

Just as the drum set sounds and computer keys could be changed, so can the musical pitches. Again, by clicking on the black triangle to the right of the note value in the "play note" block, you can change the pitches in the scale (see Figure 5.). Convert the scale from a minor scale to a major scale by adding 1 to the "play note" values under the "when 3 key pressed" and the "when 7 key pressed" blocks. Changing these values from 60 to 61 and 67 to 68, respectively, will create a major scale across the number keys on the computer keyboard. Take some time to see if you can find a melody or short musical riff using the number keys.



```
Virtual Musical Scale Set - Natural Minor/Aeolian Mode
Press 1, 2, 3, 4, 5, 6, 7, 8 keys on your keyboard to perform the musical scale

when 1 key pressed
play note 57 for 0.25 beats

when 2 key pressed
play note 59 for 0.25 beats

when 3 key pressed
play note 60 for 0.25 beats

when 4 key pressed
play note 62 for 0.25 beats

when 5 key pressed
play note 64 for 0.25 beats

when 6 key pressed
play note 65 for 0.25 beats

when 7 key pressed
play note 67 for 0.25 beats

when 8 key pressed
play note 69 for 0.25 beats
```

Figure 4. *Code for performing minor melodies with the keyboard in Scratch.*

Tips for Play and Further Exploration

- Duplicate the blocks to add more notes to the scale.
- Adjust the existing values to create new scales.
- Play around and try to find a melody or musical riff or pattern that you know.
- Try performing a melody *and* a drum beat at the same time, or with a partner.
- Assign the same keyboard key to multiple notes to create chords.

LISTENING FOR PATTERNS AND BUILDING LOOPS

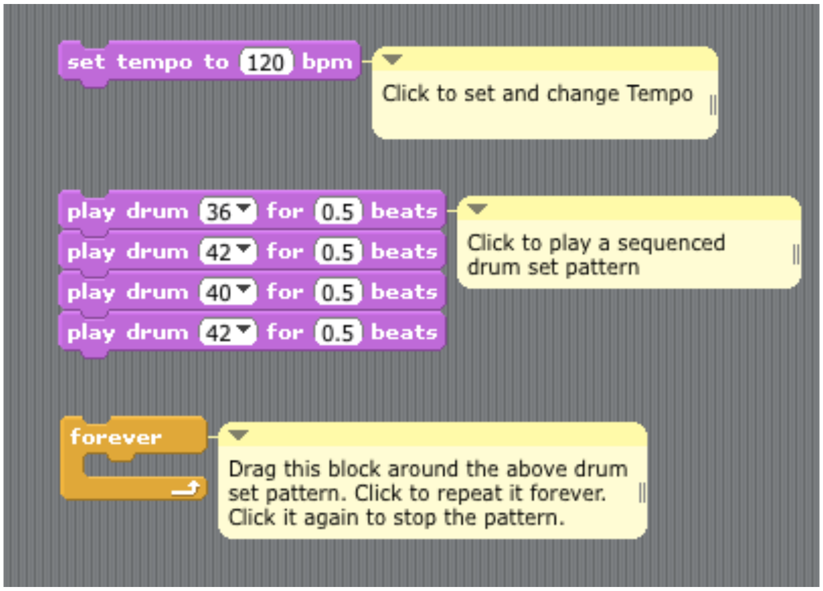
Computers are great at doing repetitive tasks. We can write a very small piece of code that tells the computer to do something over and over, and it will do that for us. The computer never gets bored repeating a task. We can harness this power of computers to our advantage when working creatively with music. Professional musicians spend hours practicing in order to learn to play and perform well the music they play. In fact, I'm sure you spent some time practicing the drumbeat and melody patterns you found above before you felt successful.

Patterns and sequences are key concepts in music. They are everywhere, from the riffs and drum beats we find in popular songs, to the ostinati of Stravinsky, and motivic development of Beethoven. Below is a basic four sound drum set pattern with a bass drum on beat 1, hi-hat on beat 2, snare drum on beat 3, and hi-hat on beat four:

36 – 42 – 40 – 42	36 – 42 – 40 – 42	36 – 42 – 40 – 42
J - K - L - K	J - K - L - K	J - K - L - K

Take a moment to practice and perform the pattern above in tempo.

The pattern you just practiced above has been re-created as a connected set of blocks in Figure 6. If you connect multiple "play drum" blocks together, you create a *sequence* of blocks, which in turn performs a musical sequence of drum sounds. Take a moment to duplicate and click on the above set of blocks in Figure 6. When you click on the blocks, Scratch will play drum 36 for 0.5 beats, drum 42 for 0.5 beats, drum 40 for 0.5 beats, and finally drum 42 again for 0.5 beats. If you want the beats to be performed



```
set tempo to (120) bpm
```

Click to set and change Tempo

```
play drum (36) for (0.5) beats
play drum (42) for (0.5) beats
play drum (40) for (0.5) beats
play drum (42) for (0.5) beats
```

Click to play a sequenced drum set pattern

```
forever
```

Drag this block around the above drum set pattern. Click to repeat it forever. Click it again to stop the pattern.

at a faster or slower speed, change the value of the “set tempo” block and be sure to click the block after changing the value.

Figure 5. *Code for sequencing drum sounds in Scratch.*

The real power of Scratch comes when you need it to repeat things and perform code blocks over and over. You can play with this feature by dragging the yellow “forever” control block around the four purple “play drum” blocks. Once that is set, click on the yellow block and your drumset pattern will repeat *forever*, until your computer loses power or you click on the yellow block again. You can use this feature to set up a regular drum pattern accompaniment.

TIPS FOR PLAY AND FURTHER EXPLORATION

- Start a drum set loop going within a “forever” block and use the number keys on your keyboard to perform a melody over it.
- Create a more complicated drum pattern by adding “play drum” blocks within the “forever” block.
- Change the “beats” values to create different rhythms.

- Drag "play drum" blocks in and out of the "forever" block while the loop is going to create variety and to explore different patterns.
- Apply the same "forever" block repetition technique to "play note" blocks to create *melodic* riffs and patterns.

CONCLUSION

In this chapter, you have read about tools and projects in tangible media and computational media musicianship. The same creative threads and themes discussed in other chapters throughout this book also apply to new media musicianship. New media tools provide access to new communities of musical practice and help to scaffold pupils' musical expression. All of these tools engage pupils as makers, designers, and curators of musical instruments and experiences. In these roles, children have new opportunities for musical agency through participation in the design of instruments and environments for musical exploration, performance, improvisation, and composing in addition to the simple use of these engaging tools. And, these tools provide opportunities for creative musical experiences anytime pupils have access to a mobile devices or a computer, at home or at school.

In *Singing Fingers*, timbre and melodic contour are made audible, visible and tangible for children. This shifting of representations can help strengthen children's understanding of musical concepts and music making skills. *Singing Fingers is* designed in ways that allow the students to participate in music making and creating at multiple levels, and with any sound imaginable. This aspect is further developed within the Scratch environment as a platform for musical interaction, creativity, and interdisciplinary connections with math and computing. The sheer volume of student created projects on the Scratch website and the worldwide network of Scratch educators is a great resource for beginning your work teaching music creatively with new media.

Because children are working with these tools on their own outside of school, the challenge, then, is for the teacher to cultivate a culture and community of new media musicians inside and around their classroom. This requires that the teacher take on the role of musician and learner *alongside* their pupils, creating spaces for pupils to share their music, and actively engaging with and learning these new media musicianship practices. These new media musicianship tools enable you to extend your pupils' engagement with your music curriculum outside the physical and temporal bounds of your classroom, and provide a space for students to bring their own musical ideas, *familiar and unexpected*, to the classroom.

One strategy to create this community of practice is to adopt a process advocated by ethnomusicologist Bruno Nettl - *musical cartography*[22]. When confronted with new technologies, it is natural to look for pre-existing maps to guide the way. In fact, one way to view this book is as a general map to follow and to orient you to the possibilities of learning and teaching music creatively. However, in order to facilitate creativity, one needs first hand experience traversing the landscape and making one's own map. This is strengthened for pupils when the teacher sets off with their pupils on that same journey of creative exploration and collaborative map-making.

Now that you've read through this chapter, take time to review the tools and projects described, and view the website links. Explore these yourself, make your own music with new media, and think about what connections you might make to your curriculum, your musicianship, and the interests and proclivities of your pupils. Document your own creative exploration with these new media musicianship projects, and read other accounts of teachers sharing the findings of their own action research projects with new media technologies (e.g., Finney & Burnard, 2007; Somekh, 2007). And, join online communities of practice focused around primary classroom issues.[23] Many of these communities have dedicated discussion groups around technology and new media. They are a great place to discuss your challenges and interests with other like-minded educators.

The tools for making and creating music have always been changing. Our children are making music with new media on their own, finding their own way and discovering new pathways for musical expression. Join them on their journey, and create

[22] See Nettl, B. (1960). Musical cartography and the distribution of music. *Southwestern Journal of Anthropology*, 16(3), 338-347.

[23] http://www.facebook.com/group/musicpln & http://www.musicteachersproject.net/.

maps together helping them achieve deeper levels of musical understanding, expression and creativity with new media.

WEBSITES AND APPLICATIONS

Singing Fingers	http://singingfingers.com/
MadPad	http://www.smule.com/madpad/
Bohemian Rhapsichord	http://bohemianrhapsichord.com/
Bangarang Boomerang	http://static.echonest.com/BeatDriver/
InBflat Project	http://inbflat.net/
Scratch	http://scratch.mit.edu/
MaKey MaKey	http://makeymakey.com/
Scratch Music Projects	http://scratchmusicprojects.com/

REFERENCES

Burnard, P. (2012). *Musical creativities in practice*. New York: Oxford University Press.

Finney, J. & Burnard, P. (Eds.). (2009). *Music Education with Digital Technology* (2nd edition). London: Continuum.

Greher, G. & Ruthmann, S.A. (2012). On chunking, simples, and paradoxes: Why Jeanne Bamberger's research matters. *Visions of Research in Music Education*, 20. Retrieved from http://www.rider.edu/~vrme/v20n1/.

Kafai, Y., Peppler, K., & Chapman, R. (2009). *The Computer Clubhouse: Constructionism and Creativity in Youth Communities.* New York: Teachers College Press.

Nettl, B. (1960). Musical cartography and the distribution of music. *Southwestern Journal of Anthropology*, *16*(3), 338-347.

Ruthmann, S.A. & Dillon, S.C. (2012). Technology in the lives and schools of adolescents. In G. McPherson and G. Welch (Eds), *Oxford Handbook of Music Education*, Volume 1, pp. 529-547. New York: Oxford University Press.

Somekh, B. (2007). *Pedagogy and Learning with ICT: Researching the Art of Innovation.* New York: Routledge.

Thibeault, M.D. (2012). Music education in a post-performance world. In G. McPherson and G. Welch (Eds.), *Oxford Handbook of Music Education*, Volume 2, pp. 517-530. New York: Oxford University Press.

Wiggins, J. (2010). *Teaching for Musical Understanding* (2nd edition). Rochester, MI: Center for Applied Research in Musical Understanding.

Scratch@MIT 2014 Session Proposal

Accepted for Scratch@MIT 2014, August 6-9, 2014, Cambridge, Massachusetts

Session Title: **Scratch Projects to Enhance for Middle School Music**

Proposer Name: Jesse M. Heines

Affiliation: Dept. of Computer Science, University of Massachusetts Lowell

Email: heines@cs.uml.edu

Names and affiliations of co-presenters:
Gena R Greher, Dept. of Music, University of Massachusetts Lowell
S. Alex Ruthmann, Dept. of Music and Performing Arts Professions, New York University

Session Type:

○ Poster/Demonstration
⦿ Hands-On Workshop
○ Interactive Panel
○ Ignite Talk

SESSION DESCRIPTION (400-800 words)

We propose a hands-on workshop geared specifically to middle school teachers on ways to infuse computing into the music curriculum. We are targeting middle schools because while the arts seem to be victims of budget cuts at the high school level, most states still require all middle school students to take courses in the arts, which of course includes music. As a result, middle school music teachers come into contact with a large percentage of the students in their schools. An interdisciplinary approach in which the arts are integrated with STEM — into what has been called STEAM — could therefore include virtually every student in a middle school, having a huge impact on the program's reach and effectiveness within an entire school.

This work has grown out of our NSF-funded Performamatics project, through which we have developed numerous resources, models, and tools that integrate computing and music. We will involve workshop participants by having them actually do one or two of the activities that we have used successfully with middle school students. We will then lead a discussion of the barriers to implementation of interdisciplinary teaching in middle schools and possible ways to address those. Unlike our previous presentations at Scratch@MIT conferences, we will do less presenting and demonstrating and more hands-on activities and interactive follow-up discussions. Like our previous presentations, however, we will prepare a handout of resources for teachers to take away.

Scratch@MIT 2012 Session Proposal

Accepted for Scratch@MIT 2012, July 25-28, 2012

Session Title: **Getting into the Digital Music Game With Scratch**

Proposer Name: Jesse M. Heines

Affiliation: Dept. of Computer Science, University of Massachusetts Lowell

Email: heines@cs.uml.edu

Names and affiliations of co-presenters:
Gena R Greher, Dept. of Music, University of Massachusetts Lowell

Session Type:

○ Poster/Demo
○ Presentation
○ Panel discussion
◉ Workshop

SESSION DESCRIPTION (400-800 words)

> "There is nothing like making music and messing with sound to inspire people to learn how to program."
> -- Prof. Dan Trueman, *co-founder of the Princeton Laptop Orchestra*

We couldn't agree more. However, the cost of "getting into the digital music game" can be prohibitive on many fronts. Our work strives to make the excitement of music technology accessible to students of all levels.

This workshop presents a subset of the techniques we've developed to allow students to explore the intersection of computing and music using the sound capabilities built into Scratch. In addition, it gives participants the opportunity to create music-generating programs themselves using a variety of Scratch constructs. The workshop will conclude with a mini concert in which some participants will play the music they created using these techniques.

One of the distinctive characteristics of the workshop is that it is presented in true interdisciplinary fashion: by a CS and a Music professor working together. This is a hallmark of our work. Participants will be exposed to a fresh approach to teaching that they may be able to implement to a greater or lesser degree in their own schools and institutions.

Background

Music applications are incredibly powerful and engaging tools for getting students interested in learning about technology. They can range from music cataloging applications such as the popular musicbrainz.org site, to music playing applications such as the ubiquitous Apple and Adobe products, to music transcribing and composing applications such as Finale, Sibelius, and Noteflight.

We are interested in harnessing the engaging power of music to stimulate student interest in computing in general and computational thinking in particular. Toward that end we have designed an interdisciplinary, general education (GenEd) course that teaches computing *and* music to undergraduates in novel ways and that is open to all students in all majors. Our project is called "Performamatics," and it has been funded by two grants from the National Science Foundation. (Please see www.performamatics.org for further information on this project.)

The centerpiece of Performamatics is "Sound Thinking," an interdisciplinary course taught with a music and a computer science professor in the room for all class meetings. This approach models the interdisciplinary environments that students will encounter when they graduate and provides valuable lessons for life in the world of work.

The computational thinking component of our approach not only helps arts majors learn to think analytically, but also helps technical majors understand computing concepts at a deeper level through applications that employ the engaging power of music. We take advantage of the "low floor and high ceiling" of Scratch to appeal to students at both ends of the curricular spectrum. We believe that this is one of the real powers of this media-rich visual programming system. We have seen that harnessing its music capabilities is a tremendous "hook" for making programming appealing to a wide range of students, from those in middle school to those in undergraduate programs.

Workshop Take-Aways

In addition to seeing demonstrations of the techniques we use in our "Sound Thinking" course and examples of student work, participants will be provided with an extensive handout containing some of our teaching materials and links to our course websites, where even more materials are publicly available.

SHORT SESSION DESCRIPTION FOR THE PROGRAM (100-200 words)

This workshop introduces participants to Scratch's music-generating capabilities and shows how they can be used to teach computing concepts to students with a wide range of music and computing experience. The workshop demonstrates techniques that we use in "Sound Thinking," a university General Education (GenEd) course open to all students in all majors. Participants will receive an extensive handout with links to our teaching materials, and they will have the opportunity to create music-generating programs themselves using a variety of Scratch constructs. The workshop will conclude with a mini concert in which some participants will play the music that they created.

Although our experience is based primarily on a university-level course, we have used some of the techniques we've developed with middle and secondary school students in after-school programs. This workshop is therefore appropriate for teachers at all levels, including those with little or no music or computing experience and those who are just beginning to work with Scratch.

OTHER NOTES

The concepts and techniques in this workshop were developed as part of an NSF-funded project called Performamatics. This project is ongoing, and we are now offering two-day workshops on the use of our techniques to foster interdisciplinary education. These workshops are intended to be attended by pairs of professors from the same institution, one from a technical field and the other from an arts field. We hope that this short workshop at Scratch@MIT 2012 will interest participants in attending one of our longer workshops and learning more about Scratch, interdisciplinary teaching, and the power of music to engage students in computing fields. There is no charge to attend our larger workshops, and we also offer limited travel support to those who need it.

Our workshop at Scratch@MIT 2010 entitled "Exploring Musical and Computational Thinking Through Musical Live Coding with Kids in Scratch" received rave reviews. See, for example, the short video at `http://www.youtube.com/watch?v=oA-PxrvlJxM` which is a snippet of an interview with a participant in our 2010 workshop just after it concluded.

We look forward to sharing our work with a new population of Scratch users, and of course introducing things we have learned in the two years since we last presented at Scratch@MIT.

30th ISME World Conference on Music Education Workshop Proposal

Accepted for ISME World Conference, July 15-20, 2012, Thessaloniki, Greece

Workshop Title:

Real World Projects for Developing Musical and Computational Thinking

Proposers:

Gena R. Greher, S. Alex Ruthmann, and Jesse M. Heines

Workshop Description:

Music and the arts are engaging contexts for students to learn, integrate and apply computational concepts and thinking strategies. With the recent boom in the development of web-enabled, personal, and mobile technologies in the consumer marketplace, new jobs in fields such as video game design, interactive media and social media technologies are emerging requiring experience, competency and fluency in both musical and computational thinking. At the tertiary level in the US, few computer science programs offer the specialized training in music, and few music programs offer the computational foundation needed to prepare graduates for these emerging careers. To help address this challenge, we were awarded two National Science Foundation grants to develop interdisciplinary curricular modules and models of collaboration for use in tertiary general education and upper level music and computer science courses. Recently, we have also extended this work to K-12 and community music education contexts through collaborations with middle and high school students and teachers as a means of creating new career pathways for into computer science and music.

Our demonstration will begin by sharing our process of interdisciplinary collaboration, following by exemplar integration projects and examples of student work integrating musical and computational thinking developed through our research. For the past two years, our primary environment for exploring musical and computational thinking has been through the Scratch visual programming language developed at the MIT Media Lab - http://scratch.mit.edu/. Scratch was originally conceived as an interactive environment for children and young adults to learn programming and computing concepts through developing interactive animations and games. Members of our research team have worked closely with the developers of Scratch to further refine and extend its interactive musical capabilities. As a result, middle, high school and undergraduate students in our courses and community music workshops are developing their own electronic musical instruments, composing environments, games and other interactive projects including new sensor-integrated performance controllers. These projects were designed to reflect the real-world collaboration skills, disciplinary and interdisciplinary understandings needed in the development of music and arts technologies today and in the future.

This research is supported by the National Science Foundation under Grant No. 0722161, “CPATH CB: Performamatics: Connecting Computer Science to the Performing, Fine, and Design Arts” and Grant No: 1118435, “DUE: Computational Thinking through Computing and Music.”

Techniques at the Intersection of Computing and Music

Jesse M. Heines
Dept. of Computer Science
Univ. of Massachusetts Lowell
Lowell, MA 01854
1-978-934-3634
heines@cs.uml.edu

Gena R. Greher
Dept. of Music
Univ. of Massachusetts Lowell
Lowell, MA 01854
1-978-934-3893
Gena_Greher@uml.edu

S. Alex Ruthmann
Dept. of Music
Univ. of Massachusetts Lowell
Lowell, MA 01854
1-978-934-3879
Alex_Ruthmann@uml.edu

Abstract

Our work on *Performamatics* aims to enhance students' computational thinking (CT) by engaging them in fundamental concepts that unite computing and music. Our approach leverages students' near universal interest in music as a context for rich CT experiences. The techniques we share are used in a General Education course open to students in any major called *Sound Thinking*, which is now being offered for the fourth time.

Categories and Subject Descriptors

K.3.2 [**Computers and Education**]: Computer and Information Science Education – *computer science education, curriculum.*

Keywords

Performamatics, computer science education, interdisciplinary programs, music+computing, Audacity, Scratch.

Course Structure

Our course teaches both music and computing. We begin by having students analyze music through listening exercises and sequencing and manipulating sounds using Audacity (audacity.sourceforge.net). We then have them begin to work with pre-recorded sounds in Scratch (scratch.mit.edu) before getting into composing in Scratch with MIDI.

Audacity-Based Techniques

Students begin the course by making instruments from "found objects," that is, common things that they find in their living area. We specify that they should look for objects that can produce multiple pitches with multiple timbres. This gets them thinking about what, exactly, constitutes "music."

The next step is to record the sounds their instruments make and load them into Audacity. They learn to use the editing capabilities of that program to break their sounds into component elements and the various effects to manipulate them. They then arrange the component sounds and apply effects to create a composition that further explores their own interpretation of "music."

In the third assignment, students choose a song of their own liking and listen to it critically to identify the components from which it is built. With these in hand, they create a flowchart of the song to illustrate how those components go together.

Scratch-Based Techniques

We then get into programing with Scratch, which is very easy for students of all majors to pick up and has some terrific capabilities for working with music.

Students begin by sequencing sounds in Scratch in a manner similar to what they did in Audacity. There is much more they can do, however, using loops and control structures. They also learn about the limitations of computer software as they try to do things that the system just can't handle.

Next we introduce intervals, focusing on the major 2nd and the perfect 5th. We show students how to program these and we have them create a composition using only these intervals. More advanced students program the intervals algorithmically and add randomization to have the music change each time it is played.

To introduce lists, we have students work on a piece that they must transpose from one key to another. This forces them to use offsets controlled by variables.

Finally, we introduce physical interaction between Scratch and the IchiBoard, a small device packed with sensors, a slider, a button, and accelerometers from which Scratch can read values that control the notes it plays (see www.cs.uml.edu/ecg/index.php/IchiBoard/IchiBoard). This technique gets students doing the most sophisticated programming in the course, as they must scale the sensor values to ranges that are appropriate for playing music.

Culminating Performances

Our course builds to a "recital" in which students showcase final projects based on the course assignments. Performances are in a venue other than our normal classroom, and students are welcome to invite their friends and family. Middle and secondary school students also attend, not only for their own benefit, but for the benefit of our students, as well. By explaining their work to younger students, our students expand and deepen their own understanding of the intersection between computing and music.

Our presentation will feature videos, recordings, and live examples of student work to demonstrate the types of results that students produce for the assignments described above.

Project URLs and Locations of Materials

Further information on our work and the materials that we use in our classes may be found at:

- *Performamatics:* http://www.performamatics.org
- *Sound Thinking:* http://soundthinking.uml.edu

Acknowledgments

This work is currently supported by National Science Foundation Award No. 1118435, "Computational Thinking through Computing and Music." *Principal Investigator:* Jesse M. Heines. *Co-Principal Investigators:* Gena Greher and S. Alex Ruthmann. *Senior Personnel:* Fred G. Martin and Sarah Kuhn. Any opinions, findings, and conclusions or recommendations expressed in these materials are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

ITiCSE'12, July 3–5, 2012, Haifa, Israel.
ACM 978-1-4503-1246-2/12/07.

Two Approaches to Interdisciplinary Computing+Music Courses

Jesse M. Heines, Gena R. Greher, S. Alex Ruthmann, and Brendan L. Reilly

University of Massachusetts Lowell

The developers of a university curriculum designed to bridge the gaps between the two disciplines have found that there are numerous ways to introduce arts majors to computing, and science and engineering majors to the arts.

The intersection of computing and music can enrich pedagogy in numerous ways, from low-level courses that use music to illustrate practical applications of computing concepts to high-level ones that use sophisticated computer algorithms to process audio signals.

We explore the ground between these extremes by describing our experiences with two types of interdisciplinary courses. In the first, arts and computing students worked together to tackle a joint project, even though they were taking independent courses. In the second, all students enrolled in the same course, but every class was taught by two professors: one from music and the other from computer science. This course was designed to teach computing and music *together*, rather than as one in service to the other.

WHO'S THE MORE CREATIVE?

It is the first day of a new semester. Two students walk into your class. You have never seen them before, and you know nothing about them. When you identify them and check their primary fields of study, you see that one is a computer science major, the other a music major. Which do you assume is the more creative?

Surely, most of us would say it is the music major. The general perception is that people in the arts are more creative than those in the sciences, particularly those in computing. But is this truly the case?

Consider the types of learning experiences that characterize each field. In music, students mainly focus on the *re-creation* of art. They learn to master their instruments by studying someone else's original creations. The *composition* of original works is advanced study, typically pursued by only a handful of music majors and usually at the graduate level.

In CS, students might initially re-create programs that implement known algorithms, but they quickly progress to writing original programs to solve problems. Those problems might be carefully bounded, but good students tend to devise solutions that exhibit a wide range of approaches.

It is interesting to note that music students also must learn concepts and syntax. Think of staves, notes, key signatures, accidentals, fingerings, and so on. The difference is what they do with these. In general, they apply what they have learned to try to play a piece exactly as their teachers say it should be played. CS students try to apply what they have learned to *solve a problem* outlined by their teachers.

So, on reconsideration, which do you now judge to be the more creative?

0018-9162/11/$26.00 © 2011 IEEE

Table 1. Computing+Music course offerings.

Listing department	Number	Percent
Music	40	77
Computer science	9	17
Co-listed	3	6
Type of instruction	**Number**	**Percent**
Single instructor	28	54
Team teaching	8	15
Not identified	16	31
Student level targeted	**Number**	**Percent**
1st- and 2nd-year undergraduates	21	40
3rd- and 4th-year undergraduates	19	37
Graduate students	5	10
Multiple levels	7	13

INTERDISCIPLINARY LEARNING

It is not our purpose, of course, to instigate an argument over who is more creative than whom. But it certainly is our purpose to break stereotypes and to stress that when we look at science and engineering majors versus their peers in the arts, business, and other supposedly nontechnical majors, it is clear that they have much to learn *from each other*. It is not much of a stretch to assert that the technologies most of our CS graduates will be working on 5 to 10 years after they graduate probably have not been invented yet. This can make it a bit hard to decide what or how we should teach them. We have therefore based our work on the following postulates.

Once our CS students graduate, it is very likely that they will never again write a program of any significant size by themselves. Instead, they will work in teams, and those teams will undoubtedly be interdisciplinary. Even if certain members of the team do not write a single line of code, they will have a say not only in what a program does, but also in how it is implemented.

Basic skills will remain basic. An array will always be an array, and a linked list will always be a linked list. With all the buzz about students seeking CS programs with concentrations in game development, programmers who succeed in that subfield will be those who understand that interesting games are built on the fundamentals of algorithms and data structures, just as musicians understand that interesting music is built on the fundamentals of melody, rhythm, and harmony. As Michael Zyda stated, "The game industry … wants graduates with a strong background in computer science. It does not want graduates with watered-down computer science degrees, but rather an enhanced set of skills."[1]

The need for everyone to have basic computer skills will only increase. Jeanette Wing stated that the basic skill in problem solving is "computational thinking," which "involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science."[2] According to Wing, this "is a fundamental skill for everyone, not just for computer scientists." We strongly agree, and we feel that exposing arts students to computational thinking *within their own field* has huge potential for enhancing their education.

Everyone has something to learn from everyone else. Virtually all jobs today involve interdisciplinary teams, and working in such teams usually requires abandoning assumptions about our coworkers' fields. Reflecting on one of the assignments in our interdisciplinary course, a CS major wrote, "It was great to work with someone as musically (and graphically) inclined as Maria [a music major]. I lack a lot of knowledge about both of those, and her ideas made very notable improvements in the programming as well as the music and graphics." Note that the CS major specifically mentions improvements to the *programming* based on ideas from the music major.

COMPUTING+MUSIC COURSES

To address these issues, we developed two interdisciplinary course models that our colleague Fred Martin dubbed *synchronized* and *hybrid*.[3] The synchronized model pairs two independent, upper-level courses in different disciplines and requires interdisciplinary teams of students to complete a project collaboratively. The hybrid model is a single course taught by two professors from different disciplines, with both in the classroom throughout the semester.

These are, of course, but two of myriad models employed in interdisciplinary computing+music courses. To put our work in perspective, we took an *informal* look at 52 courses at 40 colleges and universities that cover computing through music or music through computing. Some of these were identified by attendees at a March 2011 workshop on this topic under the auspices of the ACM SIGCSE Music Committee[4] and sponsored by the NSF-funded LIKES project[5] (www.likes.org.vt.edu). Additional courses were found by the student researcher on our team, who searched the Web for syllabi that combined computing and music in interdisciplinary courses.

Our search criteria specifically *excluded* audio recording and production courses that have the shaping of sound through electronics and signal processing as their primary objectives. Although these courses fall at the intersection of computing and music, they focus on *using* technology to achieve desired sounds rather than *teaching* computational and musical concepts together. Table 1 presents general information about the courses we discovered and gives an overall picture of the landscape.

Table 2 presents the content of the 52 courses, as gleaned from their posted syllabi. This is an inexact measure, to be sure, but it still gives a somewhat reasonable view of the field. (The numbers in each section do not add up to 52 and the percentages do not total 100 percent because some entries fall into more than one category.)

There is indeed a large range of courses offered, subjects covered, perspectives taken, teaching styles employed, and software systems used. Based on a review of this data and reflections on our familiarity with some of the teachers of these courses, the following overall picture emerges:

- At the upper end of the curriculum, virtually all courses that cover computing+music are advanced offerings by music departments. We know of no upper-level CS courses dedicated to addressing issues faced by musicians (although of course there may be some unknown to us).
- Courses and research at the upper end require deep understanding of *both* computation and music. Examples include the algorithmic composition work by Michael Edwards[6] and by Andrew Brown and Andrew Sorenson.[7]
- At the lower end of the curriculum, music is typically used to demonstrate or to introduce concepts. This is music *in service to* computing, not music *integrated with* computing. An example is the media computation work by Mark Guzdial and Barbara Ericson.[8]

Our work attempts to fill some of the gaps between these types of courses by integrating computing and music at a high conceptual level. The synchronized course targets mid- to upper-level music and CS majors with the intent of furthering students' knowledge of both. The hybrid course is a general education offering open to all students in the university. It attempts to provide an understanding of where computing and music interact, at a level that is accessible to students without deep knowledge of one or the other. Thus, our work is at both ends of the instructional spectrum.

COMBINED GUI PROGRAMMING AND MUSIC METHODS

One way to get started in interdisciplinary teaching and learning is to connect the students in two existing courses through a joint project. Administratively, this is a "low-hanging fruit" approach because it does not involve getting a new course approved or making any changes to the course catalog. All that is needed are professors who agree to collaborate with each other to build an interdisciplinary project into their courses.

In our case, the CS professor teaches a project-based course in graphical user interface programming, which fit nicely with a project-based course on teaching methods taught by the music professors. After reviewing the projects that we assign in our respective courses, we decided to make our initial foray into interdisciplinary teaching using a "found instruments" project that has been used in music for years.

Table 2. Computing+Music course content.

Disciplines covered	Number	Percent
Sound/audio	37	71
Computer science	36	69
Music (composition)	22	42
Music (theoretical)	12	25
Media	5	5
Primary focus	**Number**	**Percent**
Composition	31	60
Sound symbols	27	52
CS (introductory)	18	35
Sound processing	17	33
CS (specialized)	12	23
Music theory	7	13
Interactive media	1	2
Software used	**Number**	**Percent**
Max/MSP	11	21
Audacity	4	8
Processing	4	8
SuperCollider	3	6
ChucK, Disklavier, Pro Tools, Reason	2 each	4 each
Audition, Garage Band, Matlab, Peak, PureData, Reaktor, Scratch, Sibelius	1 each	2 each

The music assignment

For the musicians, the purpose of our assignment is similar to Andrew Hugill's description of a project intended "to strip away previous ideas of 'musicianship,' [by] reevaluating the sounding properties of objects, how they may be made into instruments, how playing techniques might be developed, and how music may be created as a result."[9] Music students are asked to do the following:

- Using only household object(s), create a musical "instrument" that can produce several different pitches or timbres. Your instrument must be able to produce several different types of sounds, or sounds with several different characteristics.
- Create a composition for your instrument that employs a specific musical form of your choice. It need not be long. A 2-3 minute piece is sufficient, but it must include distinct sections that give it form. That is, your composition must include distinctive opening, middle, and closing sections.
- Devise a system of creative notation that others will be able to understand well enough to perform your composition. Your notational system should not resemble traditional musical notation in any way.

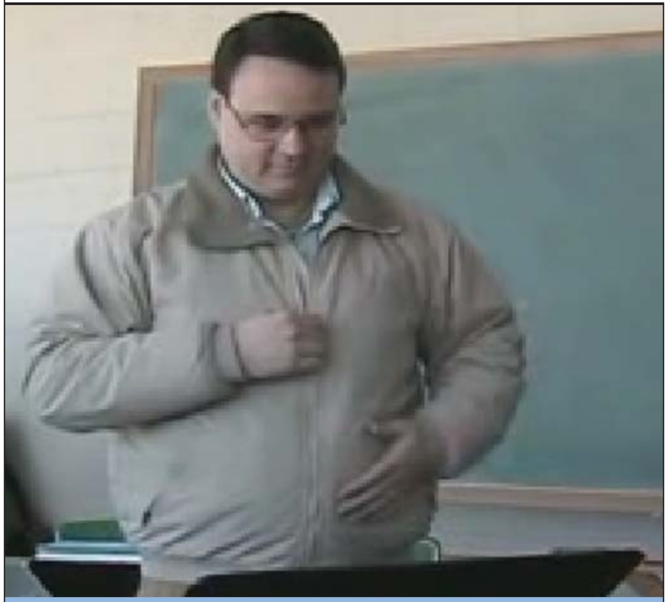


Figure 1. Mike playing his jacket as a found instrument.

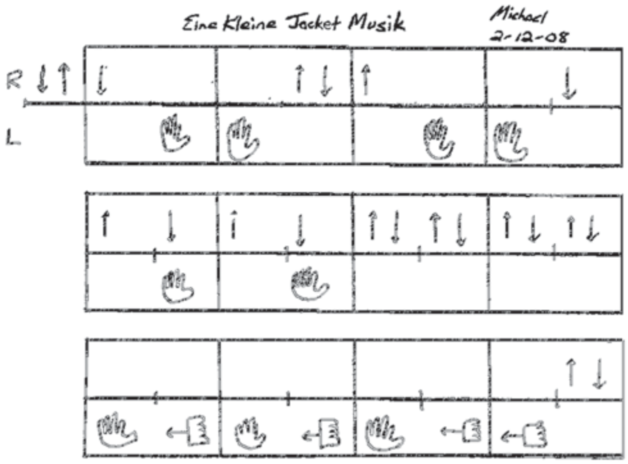


Figure 2. Mike's notation for his composition.

- Bring your instrument and notated composition to class. Come prepared to explain your work and to perform your piece.

To achieve camaraderie and pique interest, the CS majors are also given this assignment. Our experience is that the CS students "find" instruments that exhibit just as much novelty as those of their music counterparts. When we get the students from the two courses together, we do several things to build community, including having them jam on their instruments in mini-ensembles. Again, the CS students "get into" this project just as much as the music students, and the resultant "music" is, well, "interesting."

In another class activity, students try to play each other's found instruments from the notations created for those instruments. We have them do this without first hearing the original composer play the piece and without any verbal explanation of the notational system. This is a good test of the communicability of the notation by itself, and it opens up several avenues for discussion of human factors. As an example of this activity, see www.youtube.com/watch?v=IJuGoYnCxSs.

The computing assignment

The found instruments project connected to computing through the creative notation. In this project, we

- introduced CS students to standard music notation software using Finale NotePad (www.finalemusic.com/NotePad) and Noteflight (www.noteflight.com);
- assigned CS and music teams and charged the CS students with creating a music notation program for the notation devised by their music partners; and
- scheduled several joint classes in which the music students could work with the CS students on the programs' designs, review the CS students' works in progress and offer comments and suggestions for improving the programs, and finally act as usability test subjects on the finished products.

Some of the programs produced as a result of these collaborations and the lessons learned from them were truly astounding.

As Figure 1 shows, in one of the best of these projects, Mike, a music student, used his jacket as a found instrument, creating sounds by slapping it, rubbing it, working the zipper, and so on. He then created a piece satirically named *Eine Kleine Jacket Musik*. Figure 2 shows an excerpt from Mike's creative notation. Performances of Mike's piece first by Chase, a CS student, and then by Mike himself are posted at www.youtube.com/watch?v=iD4dEZOTiIg.

Figure 3 shows part of Mike's partner Chris's composition to demonstrate the CS concepts and skills involved in developing such a program.

In Figure 3a, a few icons from the tool palette on the left in the composition program have been placed onto the right- (R) and left-hand (L) staves in the composing area by either dragging and dropping them or double-clicking on them in the tool palette. In Figure 3b, the insertion cursor is positioned between the sixth and seventh icons on the left-hand staff, as indicated by the thick vertical bar. At this point, double-clicking on an icon in the tool palette would insert the icon to the right of the insertion cursor, which is to the left of the last icon on staff L.

In Figure 3c, the backspace key has just been pressed, and the blank (or "rest") icon that the arrow cursor in Figure 3b pointed to has disappeared. The issue is that the thick vertical bar insertion cursor has also disappeared,

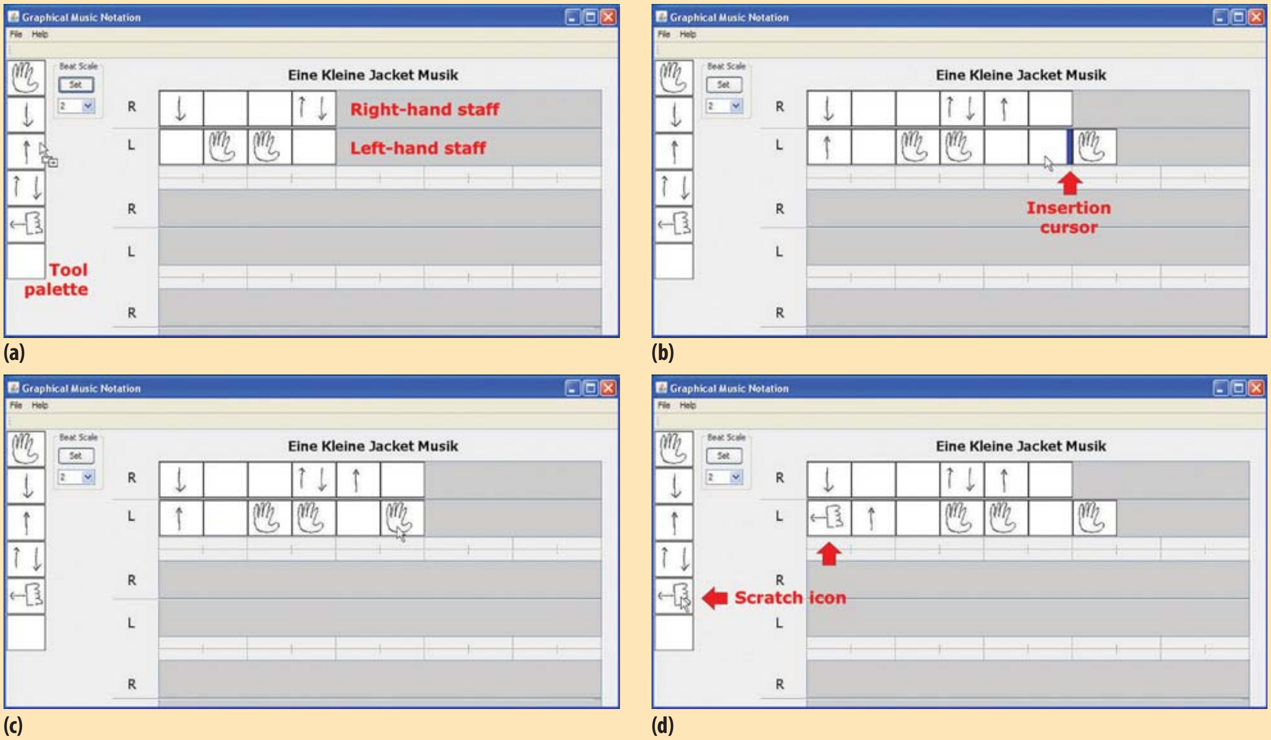


Figure 3. Chris's music composition program for Mike's jacket notation. (a) State 1: a few icons from the tool palette on the left in the composition program have been placed onto the right- (R) and left-hand (L) staves in the composing area. (b) State 2: the insertion cursor is positioned between the sixth and seventh icons on the left-hand staff. (c) State 3: the icon pointed to by the arrow cursor in Figure 3b has been deleted. (d) State 4: the scratch icon in the tool palette (indicated by the arrow cursor) has been double-clicked to insert it into the composition.

leaving users to wonder where the insertion point is. In most editors, the insertion point would not change—that is, if the user double-clicked an icon in the tool palette at this point, that icon would still be inserted to the left of the right-most hand icon on staff L.

Unfortunately, this is not what happens. Instead, when the "scratch" icon is double-clicked, it is inserted at the beginning of the staff, as Figure 3d shows. This may be fully logical to a programmer who has implemented the composition area as a pair of linked lists, but it is not at all logical to someone used to working with any sort of text editor.

When the anomaly was pointed out to Chris, he immediately recognized the problem and said, "I can't believe I didn't notice that." But that's exactly why usability tests are needed. Programmers are often "too close" to their work to see even the most obvious user interface issues. Teaching this point in a lecture setting requires students to mentally connect theory and practice. When it is learned from a peer while testing the student's own software, the connection is far more concrete, and the lesson is learned at a deeper level that is more personal and, therefore, more effective.

Thus, the fresh views of students in other disciplines can teach valuable lessons to our computing students. Likewise, for music majors, helping nonmusicians translate their musical concepts into computer programs can shed light on the clarity of their thinking—or lack thereof. Such reciprocal learning,[10] in which students learn from each other instead of just from their professors, exemplifies one of the best characteristics of interdisciplinary courses.

SOUND THINKING

Our synchronized courses worked well at the upper end of our curricula, but we also wanted to work at the lower end so that we could introduce more students to the benefits of interdisciplinary courses. Following the pioneering work of Holly Yanco and colleagues in combining art and robotics at our university,[11] we developed Sound Thinking, a new hybrid course that could be offered to all students in the university (http://soundthinking.uml.edu).

Two characteristics about the way in which Sound Thinking was put into the course catalog contributed significantly to its success. First, it was co-listed in both the music and CS departments. Second, we applied for and were granted general education status for the course. Arts students who take it register using the CS department number and receive science and technology general education credit. Science students register using the music department number and

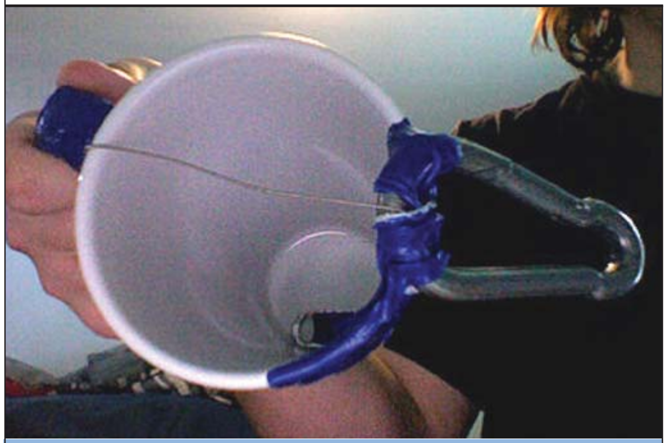


Figure 4. Top view of the lever drumitar.

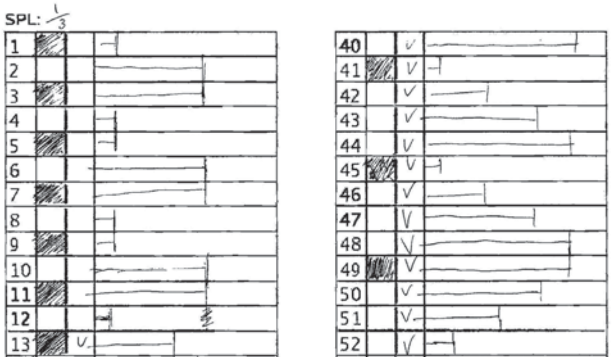


Figure 5. Notation for playing the lever drumitar.

receive arts and humanities credit. These characteristics were essential to achieving the critical number of registrations needed for the course to run, especially with two professors present at all class meetings.

Revisiting found instruments

We also used the found instruments project at the beginning of Sound Thinking, but we took it in another direction. After students created their instruments and notations, we had them record the sounds their instruments could make and then used those as an introduction to sound editing.

Eric, a CS student, created what he called a "lever drumitar," shown in Figure 4. He strung a guitar string across the opening of a cup, secured it with strong tape, and rigged up a carabiner to use as a lever for changing the cable's tension. This allowed him to produce different sounds when he strummed the cable with a soda can tab.

Figure 5 shows the original notation that Eric created for his instrument. Each row represents an action. If the square in the second column is filled in, the string is to be strummed. A V in the third column indicates that the time duration is to be shortened. The length of the line in the fourth column indicates the carabiner's position.

For the next assignment, students recorded the various sounds their found instruments could generate and loaded them into Audacity. They then created original compositions by looping and combining those sounds. To hear Eric's original lever drumitar sounds and his remixed composition, go to www.youtube.com/watch?v=_zA_hn_4T8k.

Extending found instruments

For the next assignment, students loaded their sounds using the Scratch programming language[12] and sequenced those sounds by chaining "play sound until done" blocks together. Initially, they just created linear chains like that shown in Figure 6a. When they wanted to repeat a sound or just use it again, they simply dragged in another block and selected the sound they wanted it to play.

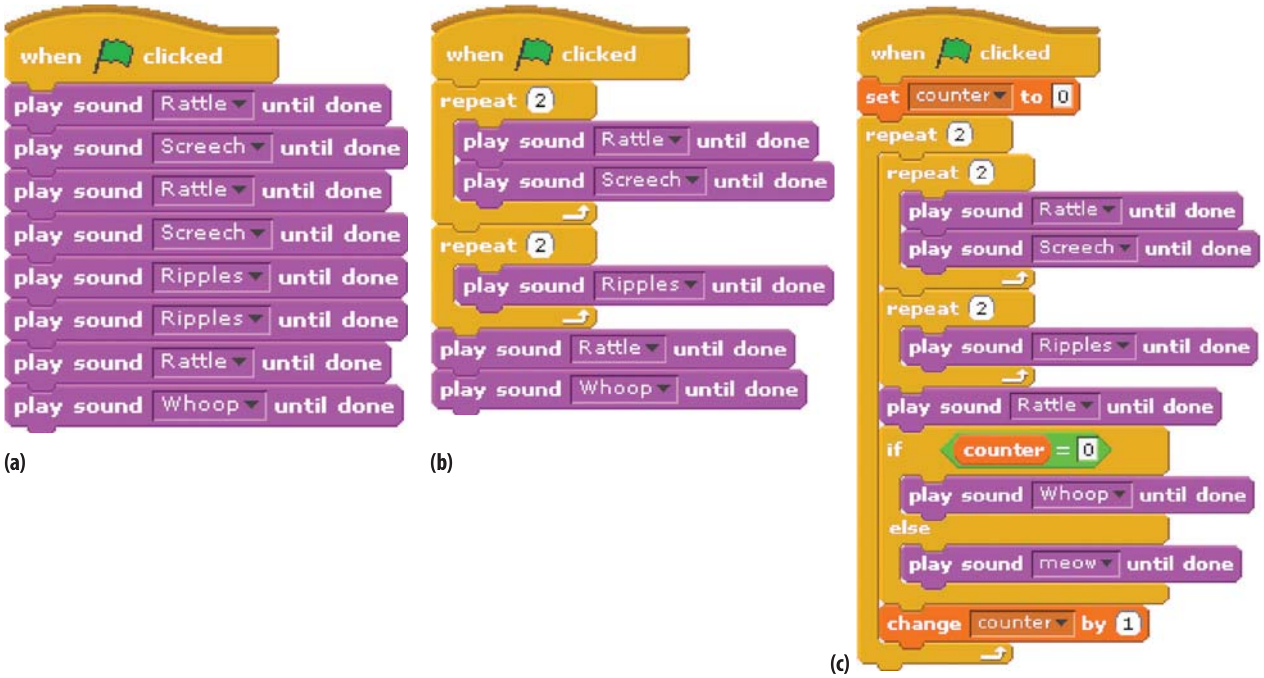
With a bit of experimentation, all the students succeeded in creating Scratch programs that used looping as shown in Figure 6b. With a bit more instruction and encouragement, most students were able to incorporate variables, nested loops, and conditional structures as shown in Figure 6c, as well.

Finally, with help from each other rather than from the professors—which indicates true student involvement in the course and is the best way for them to learn: by teaching others—some students figured out how to do more advanced things, such as playing two or more sounds simultaneously using the "play sound" and "broadcast" and its complementary "when I receive" block, leading to interesting and sometimes relatively complex discussions about synchronization.

Many CS concepts are at play here, and we use the word "play" intentionally. The Scratch development group at the MIT Media Lab is called the Lifelong Kindergarten Group for good reason. The ability to learn through *thoughtful* play that involves the use of creativity is at the heart of what we are trying to achieve. The music and arts students learn about computing, to be sure, but so do the CS and engineering students.

Using a visual programming environment like Scratch forces CS majors—who have been "brought up" on languages like C/C++ and Java and on text-based coding environments—out of their comfort zone. It is amazing how many of them stumble when they discover that a Scratch loop does not provide access to its index (counter) variable. It is pretty easy for them to implement a counter, but solving this problem requires a bit of creative thinking. In addition, explaining to their nontechnical peers what they are doing not only increases their partners' understanding, but solidifies their own as well. As the saying goes, "If you really want to learn something, teach it to someone else."

Sound Thinking builds on the found instruments project and its related assignments by introducing MIDI concepts and generating music using Scratch's various



(a)

```
when ⚑ clicked
play sound Rattle until done
play sound Screech until done
play sound Rattle until done
play sound Screech until done
play sound Ripples until done
play sound Ripples until done
play sound Rattle until done
play sound Whoop until done
```

(b)

```
when ⚑ clicked
repeat 2
    play sound Rattle until done
    play sound Screech until done
repeat 2
    play sound Ripples until done
play sound Rattle until done
play sound Whoop until done
```

(c)

```
when ⚑ clicked
set counter to 0
repeat 2
    repeat 2
        play sound Rattle until done
        play sound Screech until done
    repeat 2
        play sound Ripples until done
    play sound Rattle until done
    if counter = 0
        play sound Whoop until done
    else
        play sound meow until done
    change counter by 1
```

Figure 6. Scratch programs that chain sound blocks together to play (a) a straight sequence of sounds, (b) a sequence of sounds using loops, and (c) a looped sequence with conditionals.

"sound" blocks, shown in Figure 7. We have created several different types of assignments using these blocks, including having students write a composition based on only major 2nds and perfect 5ths (to take music majors out of their Western music comfort zone), writing algorithms to transpose lists as either MIDI values or interval deltas into different keys, and coding multiple parts that must be carefully synchronized.

These and other assignments are described in detail at soundthinking.uml.edu. Through these assignments, music majors learn about computing, and CS students learn about music.

INTERDISCIPLINARY TEACHING

One measure of the success of our work is the lasting effect it has on students. This is difficult to assess, but the number of students who return semesters later to tell us how they applied the concepts they learned in a different context gives us confidence that at least some of the activities we developed have generated good results. We are currently working to devise more rigorous evaluations to substantiate this belief.

In addition, the effects of our interdisciplinary experiences were not limited to the students. The professors also learned from each other, not only about discipline-specific content, but also about teaching and pedagogy. As the NSF evaluator of our Performamatics project wrote in her final report:

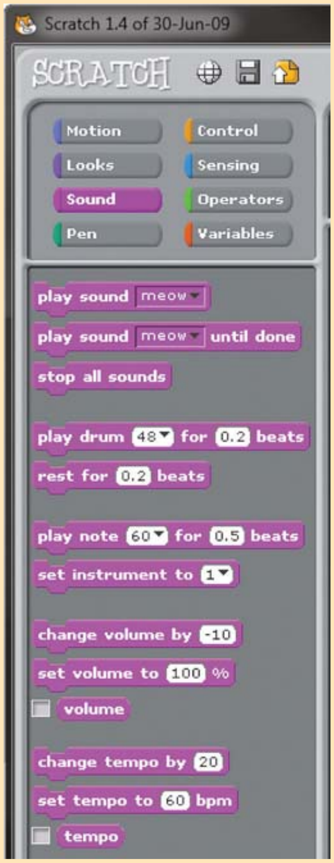


Figure 7. Blocks available from the Scratch sound panel.

> One CS faculty member … changed his approach to teaching significantly in some situations, assigning more open-ended projects, a change well received by students. … Change in faculty is an essential but often overlooked element of institutional and curricular change.

The professors' experiences in teaching with each other were so positive that they continued to do so even after the original NSF funding expired. Then in 2011, we were awarded a grant from the NSF TUES program to disseminate our work in a series of workshops for interdisciplinary pairs of professors. The first of these free workshops will be offered 21-22 June 2012. Faculty interested in attending are invited to visit www.performamatics.org for further information and to apply.

Our explorations of ways to bridge the gaps in computing+music education are really just beginning. We believe that there are many more ways to introduce arts majors to computing and science and engineering majors to the arts, and that our approaches offer effective ways to work toward that goal in an undergraduate institution. We are constantly working to improve our current methods and to extend our work into more advanced offerings that move into live coding[13-15] and text-based music coding environments such as SuperCollider, Impromptu, Processing, and Max/MSP.

Acknowledgments

This work is supported by National Science Foundation Awards 0722161 and 1118435. In addition to the authors, Fred Martin and Sarah Kuhn of the University of Massachusetts Lowell and Scott Lipscomb of the University of Minnesota are members of these project teams. Lipscomb thoroughly reviewed an early draft of this article and provided significant suggestions for improvement. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF. Please see www.performamatics.org for further information, including how to apply to attend our NSF-sponsored workshop.

References

1. M. Zyda, "Computer Science in the Conceptual Age," *Comm. ACM*, vol. 52, no. 12, 2009, pp. 66-72.
2. J.M. Wing, "Computational Thinking," *Comm. ACM*, vol. 49, no. 3, 2006, pp. 33-35.
3. F. Martin et al., "Joining Computing and the Arts at a Mid-size University," *J. Computing Sciences in Colleges*, vol. 24, no. 6, 2009, pp. 87-94.
4. R. Beck and J. Burg, "Report on the LIKES Workshop on Computing and Music," ACM SIGCSE Music Committee, to be published in 2012.
5. W. Chung et al., "LIKES: Educating the Next Generation of Knowledge Society Builders," *Proc. 15th Americas Conf. Information Systems* (AMCIS 09), Assoc. for Information Systems, 2009; www.likes.org.vt.edu/files/LIKES_AMCIS09_pub.pdf.
6. M. Edwards, "Algorithmic Composition: Computational Thinking in Music," *Comm. ACM*, vol. 54, no. 7, 2011, pp. 58-67.
7. A.R. Brown and A. Sorensen, "Interacting with Generative Music through Live Coding," *Contemporary Music Rev.*, vol. 28, no. 1, 2009, pp. 17-29.
8. M. Guzdial and B. Ericson, *Introduction to Computing and Programming in Java: A Multimedia Approach*, Prentice Hall, 2005.
9. A. Hugill, *The Digital Musician*, Routledge, 2008.
10. H.F. Silver, R.W. Strong, and M.J. Perini, *Strategic Teacher: Selecting the Right Research-Based Strategy for Every Lesson*, chapt. 13, Assoc. for Supervision & Curriculum Development, 2008.
11. H.A. Yanco et al., "Artbotics: Combining Art and Robotics to Broaden Participation in Computing," *Proc. AAAI Spring Symp. Robots and Robot Venues: Resources for AI Education*, 2007; www.cs.hmc.edu/roboteducation/papers2007/c39_yancoArtbotics.pdf.
12. M. Resnick et al., "Scratch Programming for All," *Comm. ACM*, vol. 52, no. 11, 2009, pp. 60-67.
13. A. Brown, "Generative Structures Performance," 2011; http://vimeo.com/26193440.
14. A. Ruthmann, "Live Coding & Ichiboard-Enhanced Performance," 2011; www.youtube.com/watch?v=qehSEroHn4E.
15. A. Sorenson and A. Brown, "aa-cell Live Coding at the Loft 2," 2007; www.youtube.com/watch?v=tj74-q_Mxrg.

***Jesse M. Heines** is a professor of computer science at the University of Massachusetts Lowell, with a strong interest in music and its power to interest students in computing. Heines received an EdD in educational media and technology from Boston University. Heines and Gena Greher are writing a book on interdisciplinary teaching that is currently under contract with Oxford University Press. Contact him at jesse_heines@uml.edu.*

***Gena R. Greher** is an associate professor of music education at the University of Massachusetts Lowell. Her research focuses on creativity and listening skill development in children and on examining the influence of integrating multimedia technology in urban music classrooms. Greher received an EdD in music and music education from the Teachers College of Columbia University. Contact her at gena_greher@uml.edu.*

***S. Alex Ruthmann** is an assistant professor of music education at the University of Massachusetts Lowell, where he teaches courses at the intersection of music education and arts computing. His research explores social/digital media musicianship and creativity, as well as the development of technologies for music learning. Ruthmann received a PhD in music education from Oakland University, Michigan. Contact him at alex_ruthmann@uml.edu.*

***Brendan L. Reilly** is an undergraduate computer science major at the University of Massachusetts Lowell. He has played bass since grade school, participating in every musical group available to him. Contact him at brendan_reilly@student.uml.edu.*

Teaching Computational Thinking through Musical Live Coding in Scratch

Alex Ruthmann
Dept. of Music
Univ. of Massachusetts Lowell
Lowell, MA 01854
1-978-934-3879
Alex_Ruthmann@uml.edu

Jesse M. Heines
Dept. of Computer Science
Univ. of Massachusetts Lowell
Lowell, MA 01854
1-978-934-3634
heines@cs.uml.edu

Gena R. Greher
Dept. of Music
Univ. of Massachusetts Lowell
Lowell, MA 01854
1-978-934-3893
Gena_Greher@uml.edu

Paul Laidler
Student, Dept. of Computer Science
Univ. of Massachusetts Lowell
Lowell, MA 01854
1-978-934-3634
plaidler@gmail.com

Charles Saulters II
Student, Dept. of Music
Univ. of Massachusetts Lowell
Lowell, MA 01854
1-978-934-3634
kingd615@hotmail.com

ABSTRACT

This paper discusses our ongoing experiences in developing an interdisciplinary general education course called *Sound Thinking* that is offered jointly by our Dept. of Computer Science and Dept. of Music. It focuses on the student outcomes we are trying to achieve and the projects we are using to help students realize those outcomes. It explains why we are moving from a web-based environment using HTML and JavaScript to Scratch and discusses the potential for Scratch's "musical live coding" capability to reinforce those concepts even more strongly.

Categories and Subject Descriptors

K.3.2 [**Computers and Education**]: Computer and Information Science Education — *computer science education, curriculum*.

General Terms

Design, Languages

Keywords

Performamatics, Scratch, computer science education, interdisciplinary courses, musical live coding, generative music, curriculum design.

1. PERFORMAMATICS BACKGROUND

Performamatics is a series of courses intended to attract students to computer science (CS) by tapping their inherent interest in performance and the arts. Toward that end, two CS professors have teamed with five Music, Theater, and Art professors to offer both introductory and advanced courses where assignments designed to reinforce CS concepts center around applications in the Arts. These courses are in the spirit of pioneering work done by Cooper, Dann, & Pausch [3], Guzdial [6], and Yanco et al. [15], and have been described in many other papers and presentations [5, 7, 8, 9, 12]. Readers are also referred to www.performamatics.com for links to online materials.

The most successful Performamatics courses to date (based on enrollment and student feedback) have clearly been the introductory ones. These are general education (GenEd) courses co-listed in two departments, allowing CS majors to earn Arts & Humanities GenEd credit while Arts majors earn Science & Technology GenEd credit. *Tangible Interaction Design* is a collaboration between CS and Art, while *Sound Thinking*, the course on which this paper focuses, is a collaboration between CS and Music.

2. SOUND THINKING

One of the hurdles in getting our first offering of *Sound Thinking* approved for dual GenEd credit was to convince the GenEd committee that Music majors would learn something about technology and CS majors would learn something about music. The committee feared that if project teams had both CS and Music majors, each group would naturally navigate to its own discipline and there might be relatively little true cross-over. We therefore established the following behavioral objectives for all students.

Upon completion of this course, students should be able to:

1. Identify properties of sound and describe the organization of sound into music.
2. Design a simple notation system and describe the differences between formal and informal notation.
3. Distinguish between analog and digital audio.
4. Discuss the basic differences between various audio file formats and sound compression techniques.
5. Create a web-based computer program that plays a music file.
6. Create a web-based computer program that plays a user-definable sequence of music files.

In the first half of the semester, students created compositions for "found" instruments, invented notations for those instruments, recorded the instruments' sounds, manipulated those sounds with

SIGCSE'10, March 10-3, 2010, Milwaukee, Wisconsin, USA.

audio editors, and remixed and recomposed the sounds into original compositions. In the second half, they created webpages that incorporated music, used a JavaScript Application Programmer Interface (API), and developed interactive web applications in which music played an integral part. Looking back, we see that objectives 1, 2, 5, and 6 held the most interest for students, and that's where we spent most of our class time.

Before the course was taught, there was much discussion among the Performamatics faculty about the development platform to be used for programming assignments. A CS professor said, "The Music majors will cringe if you make them code. You've got to use a visual programming environment." But a Music professor strongly disagreed, saying "One of our goals is to have them overcome any fear they have of code. We *want* them to see real code." Given that we thought students would enjoy creating webpages that they could share with their friends, we therefore chose Dreamweaver as our development platform, because it allowed viewing the page layout and its underlying code simultaneously. This helped students easily see the cause-and-effect relationship between the code and its result. We taught the basics of underlying HTML and JavaScript (with a custom API to play sounds and sound files), and only one of the 13 students had any real trouble doing the webpage development assignments.

On the contrary, most of the Music majors were very technically savvy and the post course student evaluations revealed that they wanted to know *more* about what was "under the hood." They enjoyed referring to the CS professor on the faculty team as a "magician" (because he could almost always make their pages do what they wanted when their CS partners were stumped), and they suggested that when the next the course is taught, the programming should be spread throughout the semester rather than confined to the second half.

We are now revising *Sound Thinking* for its next offering. With the help of students funded through a Research Experience for Undergraduates supplement to our NSF CPATH grant, we investigated a number of other platforms for use in the course. We have settled on Scratch [10, 11]. The remainder of this paper discusses why we feel that Scratch is an appropriate platform for teaching computational thinking through music.

3. MAKING MUSIC WITH SCRATCH

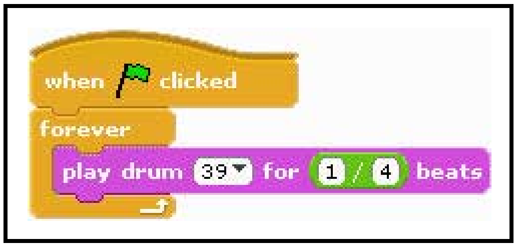
Scratch has the ability to generate and play sounds using various components in its Sound category. But if one wants to begin making *music* with more than one sound line in Scratch, one needs to address the issue of synchronization.

This issue is best addressed once students are familiar with looping, an essential concept in the implementation of many musical models. As students begin working with models where multiple voices are layered, it becomes necessary to maintain the tempo using the Scratch timer. Figure 1 shows a loop that will play a hand clap (MIDI drum instrument #39) every quarter of a beat. However, this example will not have a steady tempo, and if it was used to layer multiple voices, each would eventually fall out of synchronization.

The Scratch timer offers a way to address this problem. We first determine how many seconds our hand claps should be apart and then use the Scratch timer to control when each clap should occur. Figure 2 shows a more complex loop that will remain synchronized with the tempo regardless of the number of iterations performed. Each iteration waits until the Scratch timer reaches the value stored in variable now. This variable holds the time when a hand clap should sound. A message is then broadcast to play the hand clap. This ensures that the loop will complete and return to the "wait until" statement before the next hand clap needs to be played by delegating the actual playing to a "when I receive" event handler. The value of variable now is then changed to contain the time at which the next hand clap should be played. The Scratch timer ensures that the hand claps remain in tempo.

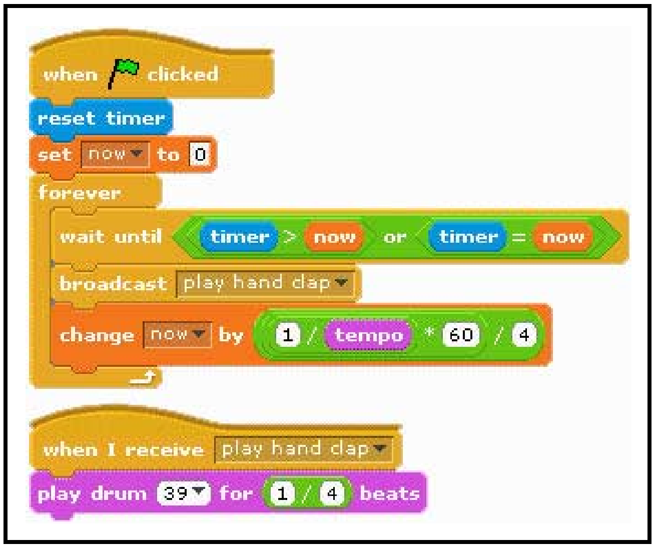
Note that in addition to synchronization, many other computational thinking concepts are touched upon by this example.

- looping
- initialization
- use of variables
- changing variables algorithmically
- modularization
- event processing



```
when ⚑ clicked
forever
  play drum 39▾ for 1 / 4 beats
```

Figure 1. A hand clap loop.



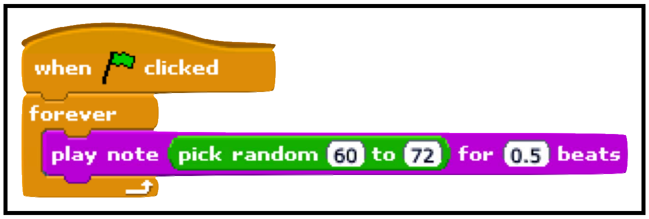
```
when ⚑ clicked
reset timer
set now▾ to 0
forever
  wait until < timer > now or timer = now >
  broadcast play hand clap▾
  change now▾ by 1 / tempo * 60 / 4

when I receive play hand clap▾
play drum 39▾ for 1 / 4 beats
```

Figure 2. Hand clap loop synchronized via the Scratch timer.

4. FROM CODE TO MUSIC

Once students understand basic note and sound generation in Scratch and can implement synchronization, more musical, generative algorithms for creating and manipulating sequences of notes can be explored.

One possible starting point is to use the "forever" loop to generate random melodies constrained by lower and upper boundaries as shown in Figure 3. In this example, random musical pitches from middle C (MIDI note #60) and the C above that (MIDI note #72) are chosen and played for half a beat. Because the "play note" function is surrounded by a "forever" loop, Scratch continues to generate notes until the Scratch stop button (●) is pressed.



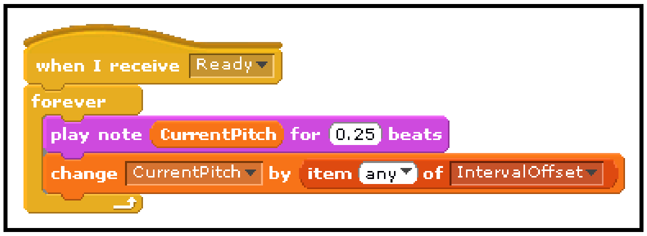
```
when ⚑ clicked
forever
  play note pick random 60 to 72 for 0.5 beats
```

Figure 3. Code for a random melody by boundary constraints.

The fact that Scratch also functions as a live interpreter/compiler makes things more interesting. This feature allows the boundaries of the random pitches as well as the duration of the sound played to be manipulated in real time through "musical live coding" [2, 13, 14] without disrupting the sounds being generated. That is, the resultant melody can be changed in real time by adjusting the upper and lower bounds of the random function and changing the duration value for the beat without stopping program execution.

The code in Figure 3 can be expanded to generate a random melody from notes provided in a pitch set as shown in Figure 4. This code implements a Scratch "list" that contains a Pentatonic (five note) pitch set. It then selects a random note from that pitch set and adds a pitch offset (MIDI note 38). In a real-time performance, the pitch could be changed by manipulating the offset to move the randomly chosen notes higher and lower through the pitch register. Additionally, through the use of the "pick random" function, the bounds of the notes chosen from the Pentatonic pitch set could be further constrained. For example, if we wanted to choose only the 2nd, 3rd, or 4th note of the set, we could change the function to "pick random 2 to 4." This technique enables live coders to create more variety in the resultant musical output.

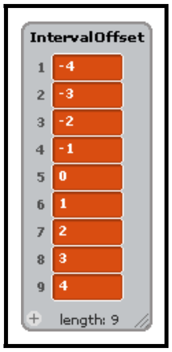
In most music, melodies do not move by random intervals. If one has a large pitch set, random intervals could result in very large leaps from one pitch to the next. A more natural sounding melody can be generated by implementing a "random walk" algorithm to change subsequent pitches as shown in Figures 5a and 5b. This results in a more musical melody by constraining the interval movement between -4 and +4 of the prior pitch. This approach also enables the melody to move freely across the MIDI pitch spectrum rather than to be constrained by the length of the pitch set as was the case in the prior examples.



```
when I receive Ready
forever
  play note CurrentPitch for 0.25 beats
  change CurrentPitch by item any of IntervalOffset
```

Figure 5a. Code for a melody via a "random walk" algorithm.

Eventually, these techniques can be expanded to model the musical styles of various composers. Our initial explorations have centered on generating music in the style of Arvo Pärt and Philip Glass. Figure 6 shows a small part of a larger algorithm inspired by Arvo Pärt's *Stabat Mater* [1]. This example iterates through the AeolianPitchSet list (organized as a descending minor scale) to select pitches and then chooses random rhythm values from the RhythmSet list. The values of the RhythmSet list were derived from an aural analysis of the *Stabat Mater* and weight the probability of selecting a whole note twice that of selecting a half note. In the full algorithm, this code is duplicated twice to create three multithreaded musical parts that are triggered via keystrokes. The addition of human control to starting and stopping threads enables the performer to create dynamic variations in musical form and texture by starting and stopping sections of the overall code.



IntervalOffset

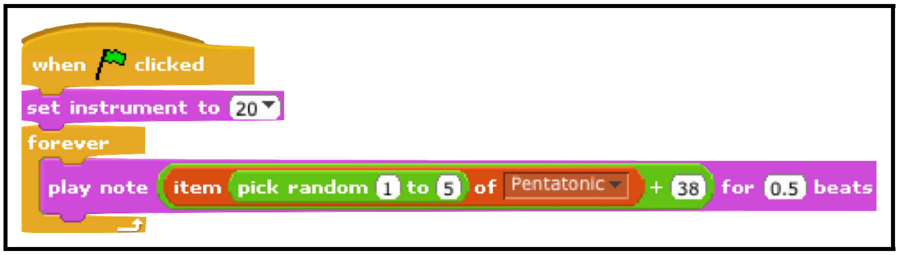
1	-4
2	-3
3	-2
4	-1
5	0
6	1
7	2
8	3
9	4

+ length: 9

Figure 5b. Interval list for use with the "random walk" algorithm.

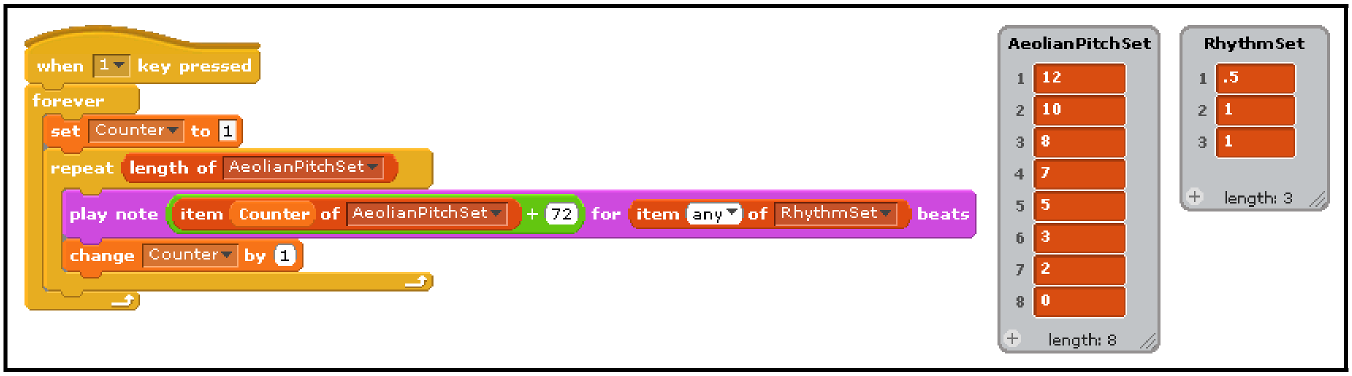
In addition to the live manipulation of lists, variables, and offsets shown in prior examples, Figure 6 also enables the selection of multiple pitch set lists, modification of the direction in which the list is iterated, and the ability to choose randomly or to isolate and repeat pitch values. Additional functions from the Scratch Sound and Numbers menus can be added, removed, and manipulated in real time to generate more musical control and expression. For example, a "change volume by x" function could be added to create changes in musical dynamics or to realize dynamic fading in or fading out of sections of the code. Additionally, a "change tempo by x" function could be inserted at various points to slow down or speed up the tempo. This could be set to a discrete value or by a mathematical function as shown in Figure 7.

Performing effective real-time manipulation of code (musical live coding) to create and shape generated music requires both musical and computational understanding. From a musical perspective, one needs to understand how the ongoing, generative music should sound. From a computational perspective, one needs to understand how the code can be adjusted and manipulated in real time to achieve the aural and musical changes and outcomes one



```
when ⚑ clicked
set instrument to 20
forever
  play note item pick random 1 to 5 of Pentatonic + 38 for 0.5 beats
```

Figure 4. Code for a melody from pitch and rhythm set lists.

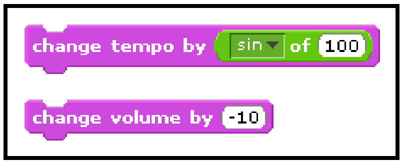


```
when [1] key pressed
forever
  set [Counter] to (1)
  repeat (length of [AeolianPitchSet])
    play note ((item (Counter) of [AeolianPitchSet]) + (72)) for (item (any) of [RhythmSet]) beats
    change [Counter] by (1)
```

AeolianPitchSet	
1	12
2	10
3	8
4	7
5	5
6	3
7	2
8	0
	length: 8

RhythmSet	
1	.5
2	1
3	1
	length: 3

Figure 6. Melodic code inspired by Arvo Pärt's *Stabat Mater*.



```
change tempo by ([sin] of (100))
change volume by (-10)
```

Figure 7. Examples of built-in Scratch functions well suited to musical live coding.

desires. These are advanced skills, but they can be learned through experimentation and exploration that is both educational and fun. Scratch provides a unique, easy-to-learn platform that enables musical live coding by allowing nearly all aspects of the code to be adjusted in real time. Students can then share and showcase their work in live or pre-coded performances, which is the essence of Performamatics.

5. ADDING A TANGIBLE INTERFACE

As the course develops, we plan to integrate tangible computing using IchiBoards [4] (Figure 8). Using these devices' live sensing capabilities, we can implement gestural musical input and design new instruments to perform musical algorithms implemented in Scratch.

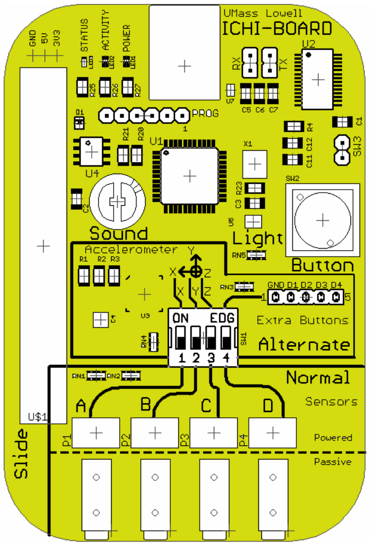
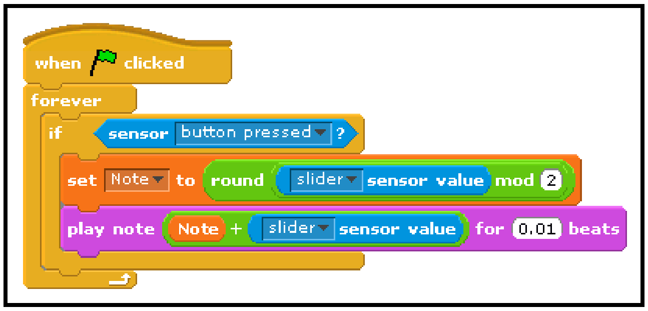


Figure 8. IchiBoard [4].

Figure 9 shows a simple program that converts an IchiBoard into a musical instrument. A "forever" loop is used to enable continuous live sensing of the board's button and slider sensors. When the button is pressed, the slider value is read and a note is played whose pitch corresponds to that value. Computational thinking comes into play because the IchiBoard's slider returns values between 0 and 100. To convert those values to a 7-note whole tone musical scale in which each interval is two equal half-steps apart, the value returned by the slider is combined with a copy of itself on which a modulus 2 operation has been applied. This ensures that when the slider is moved, the pitch of the note being played always jumps by a whole step rather than a half step. With the beat value set to 0.01, a continuous stream of pitches sounds when the button is pressed. The result is a surprisingly expressive instrument with which the user can establish a rhythm through interaction with the button and play gestural pitch sweeps through manipulation of the slider.



```
when green flag clicked
forever
  if <sensor [button pressed]?>
    set [Note] to (round ([slider] sensor value) mod (2))
    play note ((Note) + ([slider] sensor value)) for (0.01) beats
```

Figure 9. Code for a simple IchiBoard musical instrument.

The previous example only takes advantage of two of the eight possible sensor inputs on the IchiBoard. More complex interface configurations and Scratch code are currently in development that will enable more interesting musical performances and live coding demonstrations of computational thinking.

The integration of IchiBoards as an interface for tangible computing enables discussion of CS hardware concepts such as:

- What is a device?
- What is a sensor?
- What is a signal, and how is it detected in software?
- What is an event, and how is it detected in software?
- What different types of events are triggered by various devices (real and virtual)?

Integrating physical computing with Scratch's graphical coding environment provides a unique platform for expressive computing in real time. Programs can not only be written to create music, but they can be written to model musical environments that are performed through musical live coding or the design and interfacing of tangible computing devices such as the IchiBoard.

6. INTERDISCIPLINARY BENEFITS

After taking *Sound Thinking* in the Spring 2009 semester, Music major Charles Saulters developed a strong interest in using computational thinking as a means of developing more expressive gestural music controllers. He pursued a Research Experience for Undergraduates with us over the summer, exploring ways to apply these Performamatics concepts in even more exciting ways. He describes his work as follows.

> I am interested in enabling others to achieve more than they ever thought possible through the use of computational thinking in real world situations that are relevant and interesting to students. One "hook" that I found particularly interesting is the manipulation of virtual instruments, composing for and performing using nontraditional devices such as the iPod Touch. Now more than ever, we musicians find ourselves in an age where technologically almost anything is possible. It is therefore crucial that we understand what makes computers function and acquire a strong working knowledge of programs and the coding behind them.
>
> Interdisciplinary collaboration helps cultivate new and exciting innovations that can bring about the revitalization of CS education for which Performamatics was conceived. Using music as a hook, we can create innovative live performances and interesting visuals in conjunction with "musical live coding" to tap the imagination of people who might never have considered CS as a possible major. People (like myself) tend to be intimidated by the mystifying technical jargon. However, with more exposure to interesting multi-disciplinary projects, students can start thinking computationally and actively using that new way of thinking in a hands-on way without even realizing they are doing so. At that point, the fear is gone.
>
> Devices such as the iPod Touch and iPhone are ideal tools for exploring computational thinking. They are easy to use, have simple, intuitive user interfaces, and have a wide range of functionality: file transfer, web browsing, MIDI control through accelerometers, light sensors, microphones, and touch sensors.

While no Scratch interface to iPhones or iPods yet exists, these sensor-rich input devices have tremendous potential as expressive interfaces to musical live coding and performance. We see our work in integrating IchiBoards into *Sound Thinking* Version 2 as an initial step in providing these benefits.

7. ACKNOWLEDGMENTS

The work described in this paper is based upon work supported by the National Science Foundation under Grant No. 0722161, "CPATH CB: Performamatics: Connecting Computer Science to the Performing, Fine, and Design Arts" and a complementary Research Experience for Undergraduates (REU) supplement. *Principal Investigator:* Jesse M. Heines. *Co-Principal Investigators:* Fred G. Martin, Gena Greher, Jim Jeffers, and Karen Roehr. *Senior Personnel:* Sarah Kuhn and Nancy Selleck. *Student Researchers:* Paul Laidler and Charles Saulters II. Additional information on the Performamatics project can be found at www.performamatics.org.

Alex Ruthmann's adaptation of musical live coding in Scratch is based on his collaboration with Andrew R. Brown and Andrew C. Sorensen at the Queensland Univ. of Tech., Brisbane, Australia.

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

8. REFERENCES

[1] Brown, A.R. (2005). *Making Music with Java*. Brisbane, Australia: Lulu.com.

[2] Brown, A.R., & Sorensen, A.C. (2009). Interacting with generative music through live coding. *Contemporary Music Review* **28**(1):17-29.

[3] Cooper, S., Dann, W., & Pausch, R. (2000). Alice: a 3-D tool for introductory programming concepts. *Jrnl. of Computing Sciences in Colleges* **15**(5):107-116.

[4] Engaging Computing Group (2009). *IchiBoard*. www.cs.uml.edu/ecg/index.php/IchiBoard/IchiBoard *accessed* Nov. 16, 2009.

[5] Greher, G.R., & Heines, J.M. (2008). Connecting Computer Science and Music Students to the Benefit of Both. *Assoc. for Technology in Music Instruction (ATMI) 2008 Conf.* Atlanta, GA.

[6] Guzdial, M. (2003). A media computation course for non-majors. *SIGCSE Bulletin* **35**(3):104-108.

[7] Heines, J.M., Goldman, K.J., Jeffers, J., Fox, E.A., & Beck, R. (2008). Interdisciplinary approaches to revitalizing undergraduate computing education. *Jrnl. of Computing in Small Colleges* **23**(5):68-72.

[8] Heines, J.M., Jeffers, J., & Kuhn, S. (2008). Performamatics: Experiences With Connecting a Computer Science Course to a Design Arts Course. *The Int'l. Jrnl. of Learning* **15**(2):9-16.

[9] Heines, J.M., Greher, G.R., & Kuhn, S. (2009). Music Performamatics: Interdisciplinary Interaction. *Proc. of the 40th ACM SIGCSE Technical Symposium on Computer Science Education*, pp. 478-482. Chattanooga, TN: ACM.

[10] Malan, D.J., & Leitner, H.H. (2007). Scratch for budding computer scientists. *Proc. of the 38th ACM SIGCSE Technical Symposium on Computer Science Education*. Covington, Kentucky, USA: ACM.

[11] Maloney, J*., et al.* (2004). Scratch: A Sneak Preview. *Second Int'l. Conf. on Creating, Connecting and Collaborating through Computing (C5'04)*, pp. 104-109. Kyoto, Japan.

[12] Martin, F*., et al.* (2009). Joining Computing and the Arts at a Mid-Size University. *Jrnl. of Computing Sciences in Colleges* **24**(6):87-94.

[13] Sorensen, A.C., & Brown, A.R. (2007). aa-cell in practice: An approach to musical live coding. *Proc. of the Int'l. Computer Music Conf.* Copenhagen, Denmark.

[14] Wang, G., & Cook, P.R. (2004). On-the-fly programming: using code as an expressive musical instrument. *Proc. of the 2004 Conf. on New Interfaces for Musical Expression*, pp. 138-143. Hamamatsu, Shizuoka, Japan: National Univ. of Singapore.

[15] Yanco, H.A., Kim, H.J., Martin, F., & Silka, L. (2007). Artbotics: Combining Art and Robotics to Broaden Participation in Computing. *Proc. of the AAAI Spring Symposium on Robots & Robot Venues*. Stanford Univ., CA.

UMass Lowell Depts. of Music and Computer Science

73.212 / 91.212 Sound Thinking

Spring 2012 Semester, Section 201

Prof. Jesse M. Heines and Prof. Gena R. Greher

Site Home
Teaching
Research
Advising
Calendar
Projects

Course Home Information Class Notes Assignments Grades Resources (Instructor)

Class Notes

Class videos are posted at: http://echo360.uml.edu/heines/sound.htm

Next and Past Classes (in reverse order by date)

No.	Class Date	Reading	Class Topic
28	Thu May 3		Course Wrap-Up and Assessment *Note:* You must bring your final project to this class.
27	Tue May 1		Final Project Performances at The Revolving Museum *Note:* This class will take place at The Revolving Museum, which is located at 290 Jackson Street in Lowell
	Fri Apr 27		*Academic Calendar Note:* University Day — No Classes
26	Thu Apr 26		Class Time Devoted to Group Work on Final Projects
25	Tue Apr 24		Class Time Devoted to Group Work on Final Projects
	Mon Apr 23		*Academic Calendar Note:* **Faculty advising period for the Fall 2012 semester ends**
24	Thu Apr 19		Exploring Interactivity and Class Time Devoted to Group Work on Final Projects
23	Tue Apr 17		Working with the Scratch Timer and Class Time Devoted to Group Work on Final Projects
	Mon Apr 16		*Academic Calendar Note:* Patriots' Day — University Closed
22	Thu Apr 12		Using Additional IchiBoard Sensors *Academic Calendar Note:* Last day for students to complete work for incomplete Fall and Intersession 2011 semester courses
21	Tue Apr 10		Creating Music with the IchiBoard (*continued*) *Academic Calendar Note:* Last day for students to withdraw from courses with a grade of "W"
	Mon Apr 9		*Academic Calendar Note:* **Faculty advising period for the Fall 2012 semester begins**
20	Thu Apr 5		Introduction to the IchiBoard (*continued*)
19	Tue Apr 3		Synchronization Using the Scratch Timer and Introduction to the IchiBoard
18	Thu Mar 29		Playing and Transposing Using Lists (*continued*) *Note:* Prof. Heines will lead this class, as Prof. Greher will be away at a conference.
17	Tue Mar 27		Playing Music Using Lists
16	Thu Mar 22		Midterm Course Assessment
15	Tue Mar 20		Revisiting Conditional Structures and Introduction to Lists
	Thu Mar 15		*Academic Calendar Note:* Spring Break -- No Classes
	Tue Mar 13		*Academic Calendar Note:* Spring Break -- No Classes
14	Thu Mar 8		Generating 2nds and 5ths Using Variables
13	Tue Mar 6		Introduction to Intervals
12	Thu Mar 1		*Note:* Weather-Related University Closing -- Class Cancelled

11	Tue Feb 28		Introduction to MIDI Sounds in Scratch
10	Thu Feb 23		Working with Prerecorded Sounds in Scratch (*continued*)
9	Tue Feb 21		Working with Prerecorded Sounds in Scratch
	Mon Feb 20		*Academic Calendar Note:* Presidents' Day -- University Closed
8	Thu Feb 16		Introduction to Scratch and Working on Song Flowcharting
7	Tue Feb 14		Introduction to Song Flowcharting
6	Thu Feb 9		Working with Sounds in Audacity (*continued*)
5	Tue Feb 7	Handouts	Working with Sounds in Audacity
	Fri Feb 3		*Academic Calendar Note:* Last day for registered students to add a course *with* a permission number or drop a course without record
4	Thu Feb 2		Presentations of Instruments from Found Objects (*continued*)
3	Tue Jan 31		Presentations of Instruments from Found Objects
	Fri Jan 27		*Academic Calendar Note:* Last day for undergraduate students to add courses *without* a permission number
2	Thu Jan 26	Handouts	Introduction to Working with Sounds in Audacity
1	Tue Jan 24	Syllabus	Course Orientation and Introduction to the Found Instruments Project
	Mon Jan 23		*Academic Calendar Note:* First day of Spring 2012 semester classes

Tentative Future Classes

(in order by upcoming date; topics and dates are subject to change) [top]

No.	Class Date	Reading	Class Topic
	Mon May 7		*Academic Calendar Note:* Last day of Spring 2012 semester classes
	Tue May 8		*Academic Calendar Note:* Reading day
	Wed May 9		*Academic Calendar Note:* Spring semester examinations begin
	Thu May 17		*Academic Calendar Note:* Spring semester examinations end
	Sat May 26		*Academic Calendar Note:* University Commencement
	Mon May 28		*Academic Calendar Note:* Memorial Day — University Closed

This is document http://localhost/~heines/91.212/91.212-2011-12s/classnotes.jsp. *It was last modified on Thursday, March 22, 2012 at 3:55 PM.*

UMass Lowell Depts. of Music and Computer Science

73.212 / 91.212 Sound Thinking

Spring 2012 Semester, Section 201

Prof. Jesse M. Heines and Prof. Gena R. Greher

Site Home
Teaching
Research
Advising
Calendar
Projects

Course Home | Information | Class Notes | Assignments | Grades | Resources | (Instructor)

Assignments

TBA = due date to be announced

No.	Due Date	Assignment Title
1	Tue Jan 31	Creating a Composition for a Found Objects Instrument
2	Tue Feb 14	Creating a Composition from Digitized Found Sounds
3	Tue Feb 21	Creating a Song Flowchart
4	Thu Mar 1	Sequencing Sounds with Scratch
5	Thu Mar 8	Creating a Composition Based on Major Seconds and Perfect Fifths
6	Tue Apr 3	Transposing with Scratch
7	Tue Apr 17	Using IchiBoards and Sensors
8	Tue May 1	Final Sound Thinking Project and Performance

This is document http://localhost/~heines/91.212/91.212-2011-12s/assignments.jsp. It was last modified on Thursday, March 22, 2012 at 3:56 PM.

UMass Lowell Depts. of Music and Computer Science

73.212 / 91.212 Sound Thinking

Spring 2012 Semester, Section 201

Prof. Jesse M. Heines and Prof. Gena R. Greher

Site Home
Teaching
Research
Advising
Calendar
Projects

Course Home Information Class Notes Assignments Grades Resources (Instructor)

Assignment No. 1

Creating a Composition for a Found Objects Instrument

Date Due: **Tuesday, January 31, 2012**

Contents

What This Assignment Is About

This assignment involves creating an instrument from a commonly found object and devising a notation for the music that that instrument can create. While we have our own twists on this assignment, it is closely related a project that Andrew Hugill describes in his book *The Digital Musician* on page 255:

> This project is designed to strip away previous ideas of “musicianship,” [by] reevaluating the sounding properties of objects, how they may be made into instruments, how playing techniques might be developed, and how music may be created as a result. It is not necessarily a “digital” project as such, but its value to anybody working in a digital context should quickly become apparent, especially through its ability to awaken the ears.

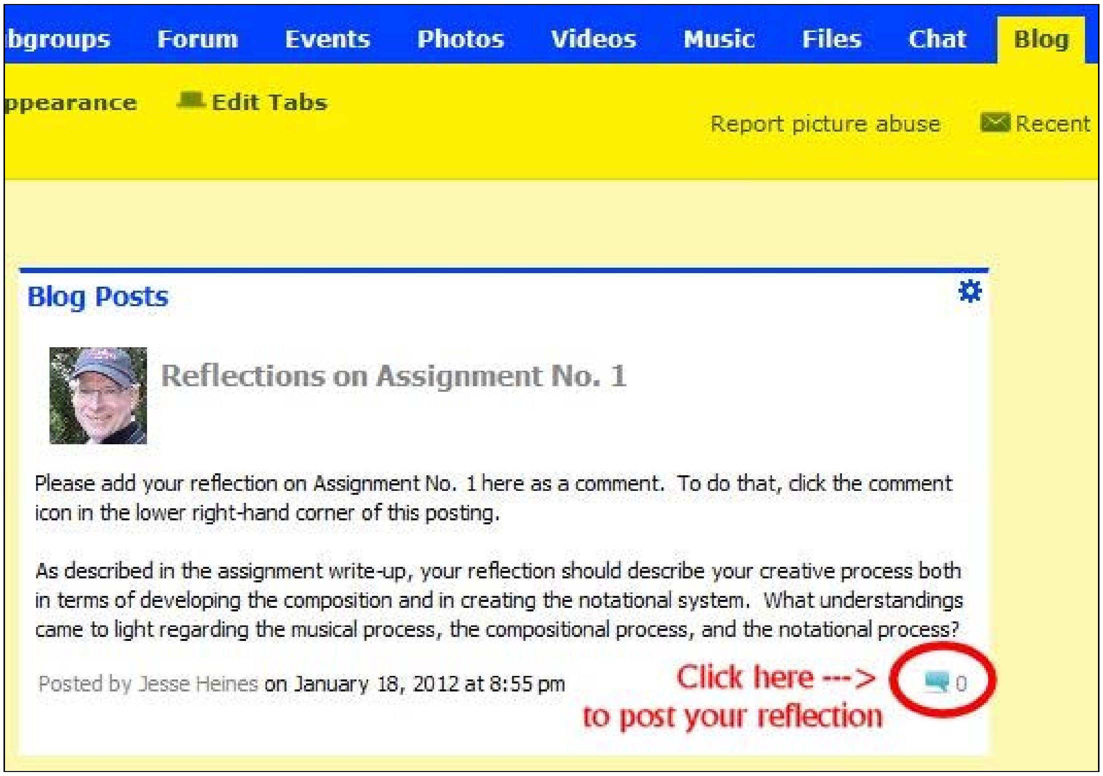
What You Are To Do

1. Using only household object(s), create a musical “instrument” that can produce several different pitches and/or timbres. It is critical that your “instrument” must be able to produce *several different types of sounds, or sounds with several different characteristics*. For example, a table that one can only tap to produce one sound would ***not*** be sufficient for this assignment. See the section below entitled How You Will Be Graded for more information on this.

2. Create a composition for your instrument. The composition should be indicative of a specific musical form of your choice. It need not be long. A 2-3 minute piece is fully sufficient, but it should include distinct sections that give it form. That is, at a minimum *your composition should include distinctive opening, middle, and closing sections*.

3. Once you have created your composition, devise a system of creative notation that others would be able to understand enough to perform your piece. The notational system should be reflective of your unique type of instrument. It should not resemble standard musical notation in any way, shape, or form.

4. Write a reflection on your experiences with this assignment that desribes your creative process both in terms of developing the composition and in creating the notational system. What understandings came to light regarding the musical process, the compositional process, and the notational process? Post your reflection in the Blog area of the Grouply site as a comment to the post entitled “Reflections on Assignment No. 1.” To do this, click the Comment icon in the lower right-hand corner of the original posting, as shown below.



Reflections on Assignment No. 1

Please add your reflection on Assignment No. 1 here as a comment. To do that, click the comment icon in the lower right-hand corner of this posting.

As described in the assignment write-up, your reflection should describe your creative process both in terms of developing the composition and in creating the notational system. What understandings came to light regarding the musical process, the compositional process, and the notational process?

Posted by Jesse Heines on January 18, 2012 at 8:55 pm

5. Bring your "found instrument" to class on the assignment's due date along with your notation. Come prepared to explain your work and to perform your piece. Please note that these classes will be videoed unless you specifically request that we ***not*** video your "performance."

Submitting Your Assignment for Grading

We professors will take notes on the "performances" during class and you will turn in your creative notation for us to review. We will review your reflection directly from the Grouply site, so it is critical that you get it posted there by the assignment's due date.

How You Will Be Graded

This assignment has four major parts:

- the instrument you choose
- the composition you compose for it
- the notation you devise for your composition
- your reflective write-up

Your grade for these parts will be determined by evaluating your work on the following criteria.

- For Your Instrument
  - Does your instrument produce two or more different timbres?
  - Did you just grab a spoon to fulfill the assignment or did you take an ordinary household object or objects and put them together and use them in an unusual and novel way?
  - Did you use only the normal sounds made by the object or did you use one or more non-traditional (unexpected) sound that it can make?
  - Was it evident that you put some thought into this assignment?

- For Your Composition
  - Does your composition exhibit a clearly discernable form with an opening, middle and end?
  - Does it use a variety of musical elements such as textural changes, dynamics, rhythmic variety, and melodic elements?

- For Your Notation
  - Is your notation just simple dots and dashes showing little effort or problem solving, or did you attempt to create an unusual way of representing your music?
  - Does it represent a clear sequence of events?
  - Does it provide guidance on the timing of events?
  - Does it indicate expressive parameters for performance such as rhythm and dynamics?
  - As with your choice of instrument, was it evident that you put some thought into this assignment?

- For Your Reflective Write-Up
  - Did you leave this until the last minute or, as above, is it clear that you thought about what you wanted to write and put some effort into doing the writing?
  - Were you mindful of formatting, grammar, spelling, etc., or did you just throw a few sentences up on the Grouply site in a sloppy manner?

⏮ ◀ ▶ ⏭

This is document http://localhost/~heines/91.212/91.212-2011-12s/212-assn/FoundObjects-03.jsp. It was last modified on Monday, March 19, 2012 at 1:10 PM.

UMass Lowell Depts. of Music and Computer Science

73.212 / 91.212 Sound Thinking

Spring 2013 Semester, Section 201

Prof. Jesse M. Heines and Prof. S. Alex Ruthmann

Site Home
Teaching
Research
Advising
Calendar
Projects

Course Home Information Class Notes Assignments Piazza Grades Resources

Assignment No. 1

Designing and Performing an Original Musical Instrument Using a MaKey MaKey Board

Date Due: **Thursday, January 31, 2013**

Contents

What This Assignment Is About

This assignment involves designing and creating a musical “instrument” from common conductive materials and interfacing them with a MaKey MaKey and Scratch to perform a variety of sounds. Once your “instrument” is built, you will create a short composition for your instrument and devise a notation for the music that that instrument can create.

In this assignment, you will explore the musical principles of form, rhythm, duration, melody, texture, timbre, composition and performance. You will also experience the computational & physical principles of sensor interfacing, electrical conductivity, triggering, and event listening.

What You Are To Do

Top

1. Using the simple conductive materials provided in class (and others you might have access to), design a musical “instrument” that can be connected to the MaKey MaKey to perform drum set, melodic, or recorded sounds. It is critical that your “instrument” must be able to produce several different types of sounds, or sounds with several different characteristics. An “instrument” that only plays one or two sounds will not meet the requirements for this assignment. See the section below entitled How You Will Be Graded for more information on this.

2. Practice playing your instrument and create a composition. The composition should be indicative of a specific musical form of your choice. It need not be long. A 2-3 minute piece is fully sufficient, but it should include distinct sections that give it form. That is, at a minimum *your composition should include distinctive opening, middle, and closing sections*.

3. Once you have created your composition, devise a system of creative notation that others will be able to understand enough to perform your piece. The notational system should be reflective of your unique type of instrument. It should not resemble standard musical notation in any way, shape, or form.

4. Write a reflection on your experiences with this assignment that describes your creative process both in terms of developing your instrument, the composition, and in creating the notational system. What understandings came to light regarding the musical process, the compositional process, and the notational process? Post your reflection in the Google Form at https://docs.google.com/spreadsheet/viewform?formkey=dFNJZWZET3B3TkVCNjhhRW1OXzNhZkE6MA#gid=0.

5. Bring your “instrument” to class on the assignment’s due date along with your notation. Come prepared to explain your work and to perform your piece. Remember, you can practice your composition without connection to a MaKey MaKey using the keys on the computer keyboard. Please note that these classes will be videoed unless you specifically request that we ***not*** video your “performance.”

Submitting Your Assignment for Grading Top

We professors will take notes on the “performances” during class and you will turn in your creative notation for us to review. We will review your reflection directly from the Google Form, so it is critical that you get it posted there by the assignment’s due date (shown at the top of this page).

Submit your reflection by answering the questions posted at:

> https://docs.google.com/spreadsheet/viewform?formkey=dFNJZWZET3B3TkVCNjhhRW1OXzNhZkE6MA#gid=0

How You Will Be Graded Top

This assignment has four major parts:

- the instrument you design and build
- the composition you compose for it
- the notation you devise for your composition
- your reflective write-up

Your grade for these parts will be determined by evaluating your work on the following criteria for a total of 20 points.

- For Your Instrument (5 points)
  - Is it connected properly to the MaKey MaKey with each trigger making a conductive connection playing a sound?
  - Does your instrument produce three or more different timbres?
  - Did you use the conductive materials in an unusual and novel way?
  - Did you use a diversity of sounds in your composition?
  - Was it evident that you put some thought into this assignment?

- For Your Composition (5 points)
  - Does your composition exhibit a clearly discernable form with an opening, middle and end?
  - Does it use a variety of musical elements such as textural changes, rhythmic variety, and melodic elements?

- For Your Notation (5 points)
  - Is your notation just simple dots and dashes showing little effort or problem solving, or did you attempt to create an unusual way of representing your music?
  - Does it represent a clear sequence of events?
  - Does it provide guidance on the timing of events?
  - Does it indicate expressive parameters for performance such as rhythm and dynamics?
  - As with your choice of instrument, was it evident that you put some thought into this assignment?

- For Your Reflective Write-Up (5 points)
  - Did you leave this until the last minute or, as above, is it clear that you thought about what you wanted to write and put some effort into doing the writing?
  - Were you mindful of formatting, grammar, spelling, etc., or did you just throw a few sentences up on the Google Form in a sloppy manner?

This is document http://teaching.cs.uml.edu/~heines/91.212/91.212-2012-13s/212-assn/FoundObjects-04.jsp. It was last modified on Monday, January 28, 2013 at 8:12 PM.

UMass Lowell Depts. of Music and Computer Science

73.212 / 91.212 Sound Thinking

Spring 2012 Semester, Section 201

Prof. Jesse M. Heines and Prof. Gena R. Greher

Site Home
Teaching
Research
Advising
Calendar
Projects

Course Home Information Class Notes Assignments Grades Resources (Instructor)

Assignment No. 2

Creating a Composition from Digitized Found Sounds

Date Due: **Tuesday, February 14, 2012**

Contents

What This Assignment Is About

This assignment is a natural extension of the Found Objects Composition that you created for the last assignment. You are to create a new composition from the sounds that can be created by your found instrument, including sounds that can only be created digitally using Audacity. The purpose of this assignment is to get you familiar with digital editing and to encourage you to continue to explore sounds and to develop your creativity.

This assignment also requires you to work with a partner. We will set up partner pairings so that students in the arts are (for the most part) paired with students in the sciences, or at least students in different majors. The pairings will be announced in class and posted in the class notes at a later date. You can then consult the class roster page to find your partner's (or partners') e-mail address(es).

What You Are To Do

1. Remember that all aspects of this project except your reflection are to be done with your partner.

2. Record the compositions that both of you created for your found object instruments and/or record all the different sounds that both of your found object instruments can make.

3. Transfer your sounds to your computers and import them into Audacity.

4. Work with your partner to break your sounds into pieces so that you can reassemble into a completely new composition. Experiment with different effects to create even more sounds from your recorded sounds. Mix the sounds from both of your found object instruments. See the notes for Class No. 2 for references in the Hugill text that might give you some ideas about interesting things to do.

5. As you work, write notes on what you did so that someone else (or even yourself a few months from now) can reproduce what you did. Take notes on things that you did and then undid, too, so that you do not forget the things you tried that you don't feel worked out well.

6. Save your work in MP3 format and post that MP3 file to the web as instructed in class. You and your partner are to submit ***ONE*** MP3 file that you created together.

7. Make a screen capture of your composition as it appears Audacity. See the next section for directions on how to do this.

8. Write a blog post reflecting on this assignment, what you learned about your own creative process, and how things went with your partner. This is the individual part of the assignment. Be sure to answer the

following two questions within your blog post, along with any other comments you wish to make.

a. What did you learn about music and what did you learn about computing?
b. How did the two of you work together?
    - That is, did you work in parallel where each person did his/her own thing and then piece it together?
    - Or did you truly collaborate by brainstorming and exploring together with shared decision making?

Post your reflection along with your notes (see #5 above) on the Grouply site as a comment to the blog post that we will create, just as you did for Assignment No. 1.

Submitting Your Assignment for Grading

For this assignment you and your partner are to email FOUR files to umlsoundthinking2012@gmail.com. Do ***not*** email your Audacity `.aup` file or the contents of your Audacity `data` directory. Just email the three MP3 files and the one screen shot listed below. The screen shot may be a JPG or BMP or PNG or GIF file.

- **TWO** MP3 files containing the original recordings from each of your found instruments (#2 above)
- **ONE** MP3 file containing the composition that the two of you together created in Audacity (#6 above)
- **ONE** screen shot of your composition as it appears in Audacity (#7 above)

These files should obviously be emailed as attachments. In the body of your message, be sure to identify all the partner(s) who worked on this assignment.

EACH of you is then to upload the following items to the Grouply site as comments to the blog post we will create specifically for posting this information.

- **TWO** sets of notes (#5 above) — each of you must post *separate* notes
- **TWO** descriptions of your creative process and reflections on this assignment (#8) — each of you must post *separate* descriptions and reflections

We will review these online, so it is important that you get everything posted by the assignment's due date.

Capturing your Audacity project screen

There are several ways to capture a screen shot and save it to a file. We will talk about some of these in class. As usual, you can also find other techniques by googling "screen shot" and specifying your operating system (such as "Windows 7" or "Mac OS/X").

- regardless of the type of system you're working on, begin by maximizing your Audacity window as large as it can be
- on Windows machines, press the **Print Screen** key to capture the screen to the clipboard
    - you will then have to paste that into a graphic editor such as Windows Paint and save it as a graphic file such as a JPG
    - alternatively, you can paste the captured screen into a Microsoft Word document and save that
- for more control over what's captured, use the built-in Windows **Snipping Tool**
    - see "How To Capture a Screen Shot with the Snipping Tool in Windows Vista / Windows 7" at http://graphicssoft.about.com/od/microsoft/ht/snippingtool.htm
- on Macs, use **Shift-Command-3** to capture the screen to a *file* or **Shift-Control-Command-3** to capture the screen to the *clipboard*
    - you can also use **Shift-Command-4** to capture a *selection* to a file or **Shift-Control-Command-4** to capture a selection to the Clipboard
    - see http://support.apple.com/kb/ht1343

How You Will Be Graded

This assignment has three major parts:

- the composition you and your partner created
- the notes you wrote on what you did
- your reflective blog post

Your grade for these parts will be determined by evaluating your work on the following criteria.

- For the Sounds you recorded and the Composition you created

    - Clarity — Did you record a few different sounds clearly or just one or two with considerable distrotion?
    - Chunks — Did you use large, unweildly clips or did you break them down into manageable sizes for maximum flexibility?
    - Creativity — Did you just rearrange the sounds or did you try to put together a truly new composition with some interesting characteristics?
    - Problem Solving — Was it evident that you put some thought into this assignment?
    - Structure — Did you use sounds from both (or all three) students' found instruments and create a piece with a clear form and a variety of musical elements such as textural changes, dynamics, rhythmic variety, melodic elements, etc.?
    - Submission — Did you save your work in MP3 format and submit the correct file?

- For the Notes on what you did
    - Screen shot — Did you submit a screen shot of your Audacity project window?
    - Clarity — Could someone else reproduce your work from your notes?
    - Comprehensiveness — Could you yourself reproduce your own work 6 months from now?

- For the Reflective Write-Up
    - Content — Did you discuss the computational concepts involved?
    - Effort — Did you leave this until the last minute or is it clear that you thought about what you wanted to write and put some effort into doing the writing?
    - Professionalism — Were you mindful of formatting, grammar, spelling, etc., or did you just throw a few sentences up on the Ning site in a sloppy manner?

This is document http://localhost/~heines/91.212/91.212-2011-12s/212-assn/DigitizedFoundSounds-03.jsp. It was last modified on Monday, March 19, 2012 at 1:10 PM.

UMass Lowell Depts. of Music and Computer Science

73.212 / 91.212 Sound Thinking

Spring 2012 Semester, Section 201

Prof. Jesse M. Heines and Prof. Gena R. Greher

Site Home
Teaching
Research
Advising
Calendar
Projects

Course Home | Information | Class Notes | **Assignments** | Grades | Resources | (Instructor)

Assignment No. 3

Creating a Song Flowchart

Date Due: **Tuesday, February 21, 2012**

Contents

What This Assignment Is About

For this assignment you are to create a flowchart for a song of your choice, similar to the ones introduced in class on February 15, 2011.

The partner pairings for this assignment are listed below. As always, consult the class roster page to find your partner’s (or partners’) e-mail address(es).

> *Names removed per IRB regulations.*

What You Are To Do

1. Remember that all aspects of this project except your reflection are to be done with your partner.

2. Search the web for sample song flowcharts and review the ones we show you, as well as the flowchart symbols introduced in in the Notes for Class No. 7.

3. Pick a song to work with that has a simple structure. E-mail that song — or a public link to it — to umlsoundthinking2012@gmail.com so that we have it to listen to as we review your flowchart.

4. Create a flowchart for your song. You may do this on paper, but if you’re adventurous, try to do it in PowerPoint. There is a basic discussion on how to do this at http://office.microsoft.com/en-us/word/ha010552661033.aspx. You can also use Word, but the flowchart drawing tools work better in PowerPoint, at least on the Macs in Durgin 406.

   > You can also find software designed specifically for drawing flowcharts by Googling “download free flowchart software” (without the quotes). One such program that gives you a 7-day free trial is SmartDraw, which you can download from http://www.smartdraw.com/specials/flowchart.asp. Others, including a web-based one, can be found at http://download.cnet.com. (You have to search that site once you get there.)
   >
   > Another interesting free online website that provides facilities to create flowcharts is http://www.flowchart.com. You have to sign up to use this program, but it is completely free. Signing up is done through a form and then confirming e-mails. This program isn't the easiest to use, but it has some good features and capabilities. It is actually written in Flash, the same program that is often used to play movies and sounds on web pages.

5. If you created your flowchart with a program or web-based tool, capture it as a graphic file to submit. See (1) under “Submitting Your Assignment for Grading” below for specific instructions on how to do this. If you created it on paper, turn it in so that we can scan it and create a graphic file from it.

6. Write a blog post reflecting on this assignment, what you learned about your own creative process, and how

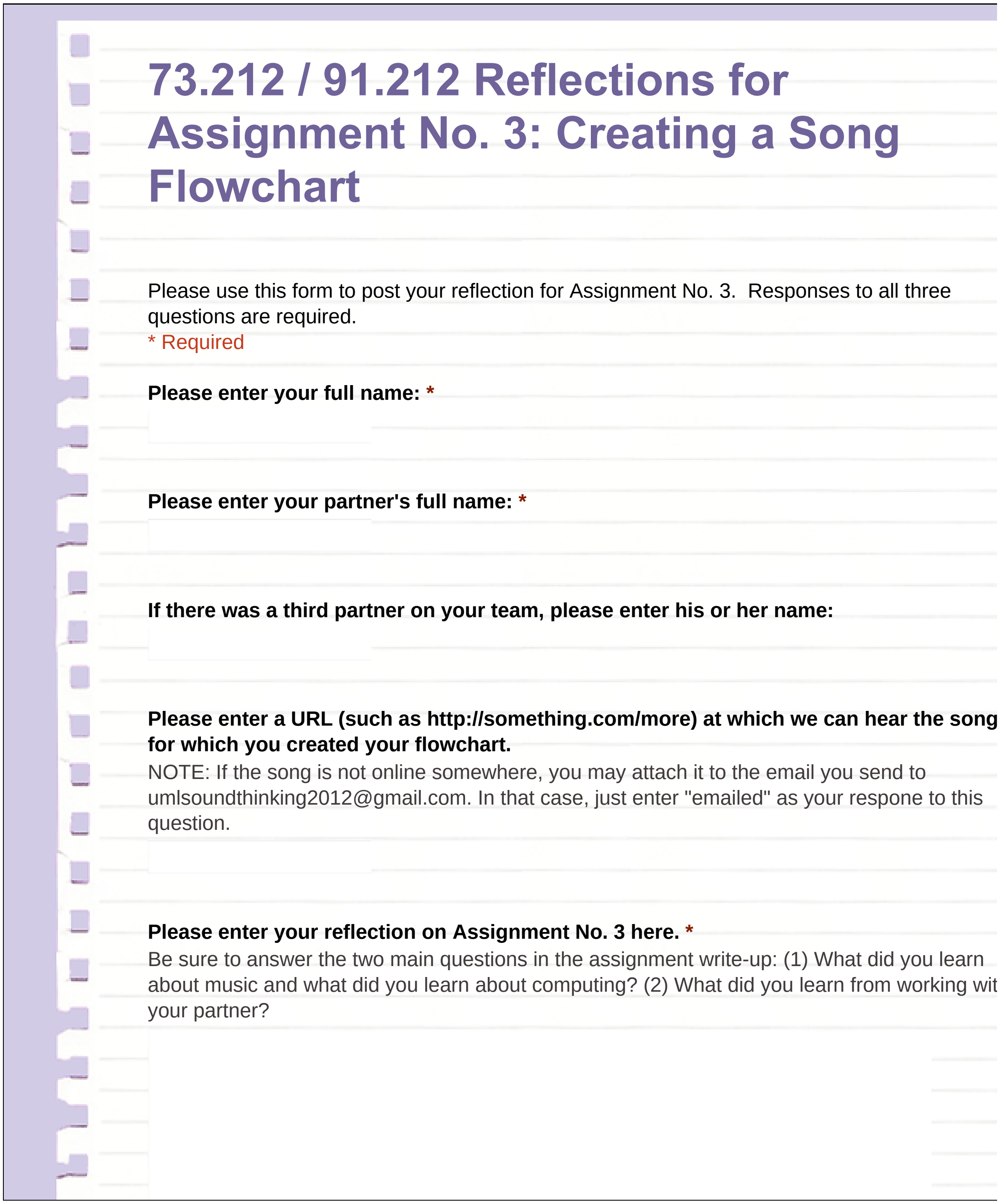
things went with your partner. This is the individual part of the assignment. Be sure to answer the following two questions within your blog post, along with any other comments you wish to make.

a. What did you learn about music and what did you learn about computing?
b. How did the two of you work together?

~~Post your reflection on the Grouply site as a comment to the blog post that we will create, just as you did for our previous assignments.~~

7. ***Update Saturday, February 25, 2012:***

Due to the Grouply website shutting down, please now post your reflection by filling out the form below or doing so at: http://bit.ly/stassn3



73.212 / 91.212 Reflections for Assignment No. 3: Creating a Song Flowchart

Please use this form to post your reflection for Assignment No. 3. Responses to all three questions are required.
* Required

Please enter your full name: *

Please enter your partner's full name: *

If there was a third partner on your team, please enter his or her name:

Please enter a URL (such as http://something.com/more) at which we can hear the song for which you created your flowchart.
NOTE: If the song is not online somewhere, you may attach it to the email you send to umlsoundthinking2012@gmail.com. In that case, just enter "emailed" as your respone to this question.

Please enter your reflection on Assignment No. 3 here. *
Be sure to answer the two main questions in the assignment write-up: (1) What did you learn about music and what did you learn about computing? (2) What did you learn from working wit your partner?

Submitting Your Assignment for Grading

(1) For this assignment you are either to e-mail a screen capture of your flowchart to umlsoundthinking2012@gmail.com or to hand in a paper version of your flowchart. If you submit an e-mail screen capture, be sure to maximize the image on your screen before you capture it so that it is of the highest quality poassible. You then save the screen capture to a file named ***exactly*** as shown below, paying particular attention to the uppercase and lowercase letters in the parts that are not your name.

> ***YourLastName_YourFirstName*_Assn3_SongFlowchart.*ext***

The "Assn3_SongFlowchart" part is standard and should appear exactly like that in everyone's submission. The "ext" part is the proper file extension (GIF, JPG, PNG, etc.) for the type of graphic file that you are submitting. Finally, attach your screen capture file to your e-mail message sent to umlsoundthinking2012@gmail.com.

(2) You must also e-mail the song you worked with (or a public link to it) to umlsoundthinking2012@gmail.com so that we have it to listen to as we review your work.

(3) The final part of the assignment is to submit your reflection as described in step 7 above. Either do so by filling out the form above or by going to http://bit.ly/stassn3. Please review our comments on your previous assignment submissions regarding your reflections on previous assignments and address any issues we mentioned to maximize your grade on this part of the assignment.

How You Will Be Graded

The two parts of this assignment will be evaluated on the following criteria.

- For the Flowchart
  - Creativity — How well does your flowchart represent the song you chose?
  - Problem Solving — Was it evident that you put some thought into this assignment?

- For the Reflection
  - Thoroughness — Did you think about the various aspects of this assignment and their relation to musical composition?
  - Professionalism — Was your writing and its formatting, grammar, spelling, etc. done professionally?

This is document http://localhost/~heines/91.212/91.212-2011-12s/212-assn/SongFlowchart-03.jsp. *It was last modified on Saturday, April 21, 2012 at 12:07 PM.*

UMass Lowell Depts. of Music and Computer Science

73.212 / 91.212 Sound Thinking

Spring 2012 Semester, Section 201

Prof. Jesse M. Heines and Prof. Gena R. Greher

Site Home
Teaching
Research
Advising
Calendar
Projects

Course Home Information Class Notes Assignments Grades Resources (Instructor)

Assignment No. 4

Sequencing Sounds with Scratch

Date Due: **Thursday, March 1, 2012**

Contents

What This Assignment Is About

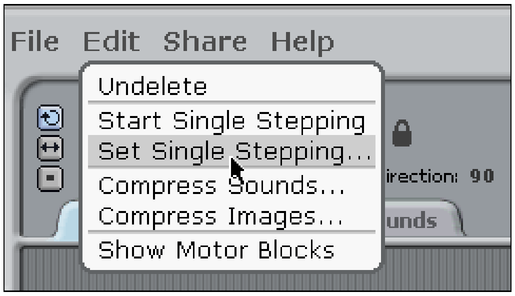
For this assignment you are to create a Scratch program that plays a sequence of sounds to form a complete composition. What we're looking for is a computer program that sequences clips using Scratch blocks rather than linearly as you did using Audacity. Note that Scratch blocks *can* be arranged linearly, but they don't have to be. Think about using loops to write as little code as possible. This means that each clip should really be an indivisible musical tone or phrase and you should use loops and/or broadcasts in the main thread of your program to repeat those clips when desired.

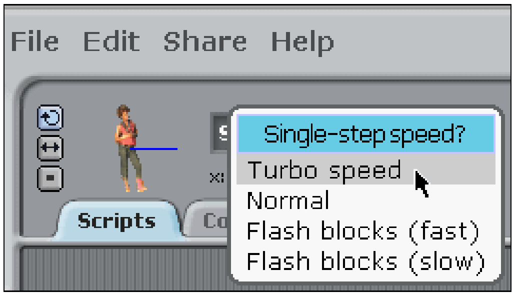
This assignment requires you to work with a partner. We have set up new partner pairings so that, for the most part, students familiar with programming are paired with students who are not. The pairings are listed below, and you should consult the class roster page to find your partner's (or partners') e-mail address(es).

Names removed per IRB regulations.

A Note on Timing

We do not expect you to get the timing perfect in this assignment. That's almost impossible when you play MP3 and other external sound files with Scratch. However, you can improve the timing considerably if you remember to set Turbo speed as discussed in class and shown again below. Click the **Edit** top menu item to display the submenu shown in the first screen capture below, and then click **Set Single Stepping...** and finally click **Turbo speed**.





What You Are To Do

1. Remember that all aspects of this project except your reflection are to be done with your partner.

2. Begin by listening to the sound file named FullMelody.mp3 (click the link to download that file). If you are reading this assignment off-line, the full URL to that file is:

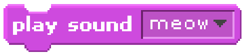
http://teaching.cs.uml.edu/~heines/91.212/Resources/Scratch/SequencingInScratch/FullMelody.mp3

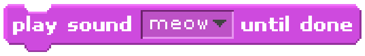
3. Next, opening the Scratch file called SequencingChunks.sb. Again, if you are reading this assignment off-line, the full URL to that file is:

   http://teaching.cs.uml.edu/~heines/91.212/Resources/Scratch/SequencingInScratch/SequencingChunks.sb

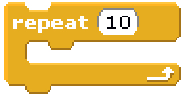
   You will notice there a number of sound files have been loaded into this file. Click the **Sounds** tab to see them. These files have been randomly named and ordered.

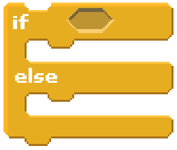
4. Listen to all the different file chunks and create a Scratch program that sequences the full melody. Use the **play sound** and/or **play sound until done** controls:





   and **loops** and **if** structures to make the code as compact as possible:





   Remember that the purpose is to write as ***little*** code as possible. Each musical phrase or idea or pattern should appear only once in your code. Also remember that you can nest controls inside each other.

   Note: You will need to create some additional chunks to make the music work right. In addition, it will be worthwhile to create a flowchart for the song to help you with the sequencing.

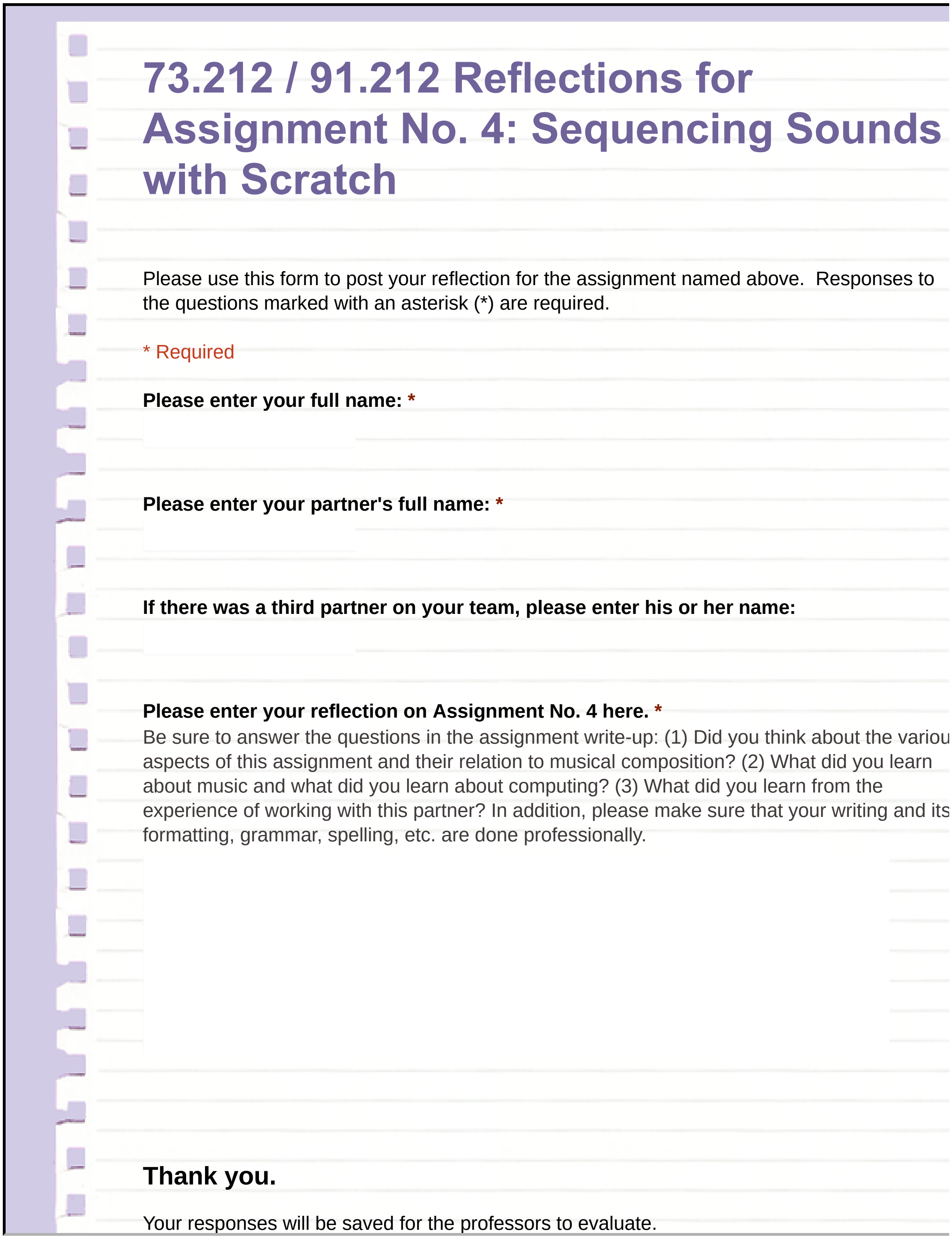
5. Share your composition with other students to help them learn and to get their suggestions for improving your work.

6. ~~Write a blog post reflecting on this assignment, what you learned about your own creative process, and how things went with your partner. This is the individual part of the assignment. Be sure to answer the following two questions within your blog post, along with any other comments you wish to make.~~
   a. ~~What did you learn about music and what did you learn about computing?~~
   b. ~~How did the two of you work together?~~

   ~~Post your reflection along with your notes (see #5 above) on the Grouply site as a comment to the blog post that we will create, just as you did for our previous assignments.~~

7. ***Update Monday, February 27, 2012:***

   Due to the Grouply website shutting down, please now post your reflection by filling out the form below or doing so at: http://bit.ly/stassn4



73.212 / 91.212 Reflections for Assignment No. 4: Sequencing Sounds with Scratch

Please use this form to post your reflection for the assignment named above. Responses to the questions marked with an asterisk (*) are required.

* Required

Please enter your full name: *

Please enter your partner's full name: *

If there was a third partner on your team, please enter his or her name:

Please enter your reflection on Assignment No. 4 here. *
Be sure to answer the questions in the assignment write-up: (1) Did you think about the variou aspects of this assignment and their relation to musical composition? (2) What did you learn about music and what did you learn about computing? (3) What did you learn from the experience of working with this partner? In addition, please make sure that your writing and its formatting, grammar, spelling, etc. are done professionally.

Thank you.

Your responses will be saved for the professors to evaluate.

Submitting Your Assignment for Grading

(1) For this assignment you are to e-mail your Scratch program (only the **.sb** file) to umlsoundthinking2012@gmail.com. Please be sure to name the file ***exactly*** as shown below, paying particular attention to the uppercase and lowercase letters in the parts that are not your name.

`YourLastName_YourFirstName_PartnerLastName_PartnerFirstName_Assn4_ScratchSequence.sb`

The "`_Assn4_ScratchSequence.sb`" part is standard and should appear ***exactly*** like that in everyone's submission.

(2) The second part of the assignment is to create a blog entry as described in step 6 above. Please review our comments on your previous assignment submissions regarding your reflections and address any issues we mentioned to maximize your grade on this part of the assignment.

How You Will Be Graded

The two parts of this assignment will be evaluated on the following criteria.

- For the Program
  - Do the musical chunks match the melodic structure?
  - Are the additional chunks your created appropriate for the composition?
  - Is your program as short as possible?
  - Did you compensate for some of the major timing issues?
  - Did you add comments to the program to identify the major sections?
- For the Reflection
  - Did you think about the various aspects of this assignment and their relation to musical composition?
  - What did you learn about music and what did you learn about computing?
  - What did you learn from the experience of working with this partner?
  - Was your writing and its formatting, grammar, spelling, etc. done professionally?

⏮ ◀ ▶ ⏭

This is document http://localhost/~heines/91.212/91.212-2011-12s/212-assn/ScratchMusicSequence-03.jsp. It was last modified on Saturday, April 21, 2012 at 12:10 PM.

UMass Lowell Depts. of Music and Computer Science

73.212 / 91.212 Sound Thinking

Spring 2012 Semester, Section 201

Prof. Jesse M. Heines and Prof. Gena R. Greher

Site Home
Teaching
Research
Advising
Calendar
Projects

Course Home | Information | Class Notes | Assignments | Grades | Resources | (Instructor)

Assignment No. 5

Creating a Composition Based on Major Seconds and Perfect Fifths

Date Due: **Thursday, March 8, 2012**

Contents

What This Assignment Is About

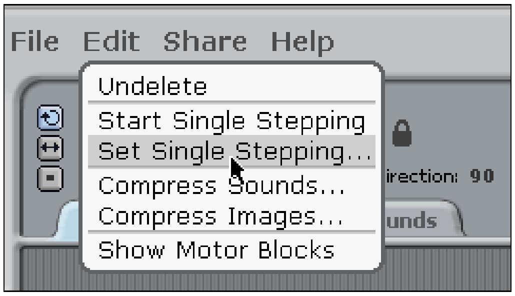
This assignment extends your work with Scratch into the use of MIDI. You are to create a MIDI composition that conforms to the parameters enumerated below. Please be sure to read these carefully!

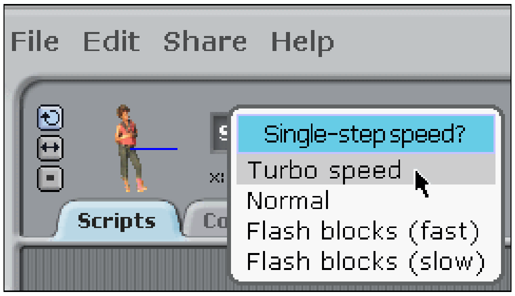
As always in this course, this assignment requires you to work with a partner. We have set up the new partner pairings below, and you should consult the class roster page to find your partner's (or partners') e-mail address(es).

> *Names removed per IRB regulations.*

A Note on Timing

As in the last assignment, we do not expect you to get the timing perfect in this assignment. However, you can improve the timing considerably if you remember to set Turbo speed as discussed repeatedly in class and shown again below. Click the **Edit** top menu item to display the submenu shown in the first screen capture below, and then click **Set Single Stepping...** and finally click **Turbo speed**.





What You Are To Do

1. Remember that all aspects of this project except your reflection are to be done with your partner.
2. Using the pitch class "C" (MIDI note value 60 in Scratch) as your starting point, create a melodic composition based on the intervals of major seconds, perfect fifths and their inversions. (For a review of their inversions, please refer to Class Notes for March 1).

   READ THE NEXT SENTENCE VERY CAREFULLY!

   >>> These are the * ONLY *** intervals that you may use in your composition.**

READ THE PREVIOUS SENTENCE AGAIN!

3. At this point we are only concerned with a viable single line melodic-composition. You need not include an accompaniment, but you may create a simple rhythmic accompaniment if you wish.

4. Your Scratch composition should demonstrate your understanding of how to create loops, use variables, and add comments to identify major sections and explain what your code is doing.

5. Make sure that your composition has a title and displays your name.

6. As you work, write notes on what you did so that someone else (or even yourself a few months from now) can reproduce what you did. Take notes on things that you did and then undid, too, so that you do not forget the things you tried that you don't feel worked out well.

7. As always, you are to write a reflective blog that traces your thinking and sequential steps in arriving at your composition and why you believe you were given this type of assignment. Post your reflection by filling out the form below or doing so at: http://bit.ly/stassn5

73.212 / 91.212 Reflections for Assignment No. 5: Creating a Composition Based on Major Seconds and Perfect Fifths

Please use this form to post your reflection for the assignment named above. Responses to the questions marked with an asterisk (*) are required.

* Required

Please enter your full name: *

Please enter your partner's full name: *

If there was a third partner on your team, please enter his or her name:

Please enter your reflection by answering the following questions.

Please make sure that your writing and its formatting, grammar, spelling, etc. are done professionally.

(1) What did you think about the various aspects of this assignment and their relation t musical composition? *

Submitting Your Assignment for Grading

(1) For this assignment you are to e-mail your Scratch program (only the **.sb** file) to umlsoundthinking2012@gmail.com. Please be sure to name the file ***exactly*** as shown below, paying particular attention to the uppercase and lowercase letters in the parts that are not your name.

`YourLastName_YourFirstName_PartnerLastName_PartnerFirstName_Assn5_2nds5thsComposition.sb`

The "`_Assn5_2nds5thsComposition.sb`" part is standard and should appear ***exactly*** like that in everyone's submission.

(2) The second part of the assignment is to create a blog entry as described in step 7 above. Please review our comments on your previous assignment submissions regarding your reflections and address any issues we mentioned to maximize your grade on this part of the assignment.

How You Will Be Graded

The two parts of this assignment will be evaluated on the following criteria.

- For the Program
  - Does the composition play as intended? Is it ****ONLY**** made up of 2nds and 5ths?
  - Is your program as short as possible?
  - Did you compensate for some of the major timing issues?
  - Did you add comments to the program to identify the major sections?

- For the Reflection
  - Did you think about the various aspects of this assignment and their relation to musical composition?
  - What did you learn about music and what did you learn about computing?
  - What did you learn from the experience of working with this partner?
  - Was your writing and its formatting, grammar, spelling, etc. done professionally?

⏮ ◀ ▶ ⏭

This is document http://localhost/~heines/91.212/91.212-2011-12s/212-assn/CompositionFrom2nds5ths-02.jsp. *It was last modified on Saturday, April 21, 2012 at 12:12 PM.*

UMass Lowell Depts. of Music and Computer Science

73.212 / 91.212 Sound Thinking

Spring 2012 Semester, Section 201

Prof. Jesse M. Heines and Prof. Gena R. Greher

Site Home
Teaching
Research
Advising
Calendar
Projects

Course Home Information Class Notes Assignments Grades Resources (Instructor)

Assignment No. 6

Transposing with Scratch

Date Due: **Tuesday, April 3, 2012**

Contents

What This Assignment Is About

This assignment extends your work with Scratch into the use of MIDI. You are to create a MIDI composition that conforms to the parameters enumerated below. Please be sure to read these carefully!

As always in this course, this assignment requires you to work with a partner. We have set up the new partner pairings below, and you should consult the class roster page to find your partner's (or partners') e-mail address(es).

> *Names removed per IRB regulations.*

What You Are To Do

1. Remember that all aspects of this project except your reflection are to be done with your partner.
2. Encode a song into a list (or set of lists) as we will have done in class. This may be an original composition or a song you like, but it should be encoded using deltas rather than absolute notes.
3. Write the Scratch code to read your list(s) and play the song they encode. Do this in such a way that allows you to transpose the song into any key simply by changing the starting note.
4. Devise a way for the user to change the starting note ***without*** having to explicitly change it in the code.
5. As you work, write notes on what you did so that someone else (or even yourself a few months from now) can reproduce what you did. Take notes on things that you did and then undid, too, so that you do not forget the things you tried that you don't feel worked out well.
6. As always, when you finish, enter your notes and reflection into the form at http://bit.ly/stassn6. That form is also appended to the online version of this write-up.

Students who wish to extend this assignment may do so by adding multiple parts as we will have done in class and/or adding graphics that provide buttons that users can click to change the key up or down.

Submitting Your Assignment for Grading

(1) As you did for Assignment No. 5, e-mail your Scratch program (only the **.sb** file) to umlsoundthinking2012@gmail.com. Please be sure to name the file ***exactly*** as shown below, paying particular attention to the uppercase and lowercase letters in the parts that are not your name.

> YourLastName_YourFirstName_PartnerLastName_PartnerFirstName_Assn6_TransposingUsingLists.sb

The “_Assn6_TransposingUsingLists.sb” part is standard and should appear ***exactly*** like that in everyone’s submission.

(2) The second part of this assignment is to submit your notes and reflection as described in step 6 above. Please review our comments on your previous assignment submissions regarding your reflections and address any issues we mentioned to maximize your grade on this part of the assignment.

How You Will Be Graded

This assignment will be evaluated on the following criteria.

- For the Program
    - Is the song implemented using lists?
    - Can the song be played in any key?
    - Can the key be changed *without* changing the value of a variable in the code?
    - Is the program as short as possible?
    - Does the program compensate for any timing issues?
    - Does the program include comments to identify the major sections?

- For the Notes
    - Are your notes thorough and clear?
    - Could someone reproduce your work from the notes you provided?

- For the Reflection
    - Did you think about the various aspects of this assignment and discuss their relation to musical composition?
    - What did you learn about music and what did you learn about computing?
    - What did you learn from the experience of working with this partner?
    - Was your writing and its formatting, grammar, spelling, etc. done professionally?

This is document http://localhost/~heines/91.212/91.212-2011-12s/212-assn/TransposingUsingLists-02.jsp. It was last modified on Saturday, April 21, 2012 at 12:13 PM.

73.212 / 91.212 Reflections for Assignment No. 6: Transposing with Scratch

Please use this form to post your reflection for the assignment named above. Responses to the questions marked with an asterisk (*) are required.

* Required

Please enter your full name: *

Please enter your partner's full name: *

If there was a third partner on your team, please enter his or her name:

Please enter your notes and reflection by answering the following questions.

Please make sure that your writing and its formatting, grammar, spelling, etc. are done professionally.

(1) Please document what you did to complete this assignment as if you were writing directions for another student to duplicate your work. *

(2) What do you think about the various aspects of this assignment and their relation to musical composition? *

UMass Lowell Depts. of Music and Computer Science

73.212 / 91.212 Sound Thinking

Spring 2012 Semester, Section 201

Prof. Jesse M. Heines and Prof. Gena R. Greher

Site Home
Teaching
Research
Advising
Calendar
Projects

Course Home Information Class Notes Assignments Grades Resources (Instructor)

Assignment No. 7

Using IchiBoards and Sensors

Date Due: **Tuesday, April 17, 2012**

Contents

What This Assignment Is About

The next step in generating music is to have Scratch read from external sensors and create music in response to the signals it receives from those sensors. We will be using IchiBoards, which are built to interface with Scratch, to accomplish these goals.

As always in this course, this assignment requires you to work with a partner. We have set up the new partner pairings below, and you should consult the class roster page to find your partner's (or partners') e-mail address(es).

Names removed per IRB regulations.

What You Are To Do

This assignment can be both fun and challenging. When coupled with Scratch, the IchiBoard can be a musical instrument itself, but generating precise effects based on the sensor values is probably not practical. For example, if you try to program the slider so that the IchiBoard works like a trombone, you will find it difficult to hit notes cleanly. It is better to have the sensor values set variables that are then part of an algorithm to produce the sounds you want. We will, of course, talk more about this concept and hopefully make it clear to everyone in class.

We suggest that you keep your aspirations for this assignment modest, but that you try to use more than just the slider and the button on the IchiBoard. One approach is to start with the composition that you created for either Assignment No. 5 (*Creating a Composition Based on Major Seconds and Perfect Fifths*) or Assignment No. 6 (*Transposing with Scratch*) and modify it in some way to respond to signals received from the IchiBoard. We will demonstrate a number of techniques for using the IchiBoard in class, but of course what you produce will be limited only by your creativity.

Since this assignment involves manipulating the IchiBoard sensors to create your piece, the actual code you hand in will not be a complete representation of your work. Therefore, we strongly suggest that you include detailed notes for us using the File->Project Notes... facility so that we fully understand your work. In addition, as always, you should include Scratch comments on specific parts of your code to explain what's going on. You may also include an explanation of how we should test your program and "play" your piece as part of your blog post.

Do not make this assignment harder than it is! You only have a week to complete this assignment — we want to leave the final two weeks of the class for you to work on your final performance — so you should build on existing code as much as you can rather than completely starting over with new code.

Submitting Your Assignment for Grading

(1) As you did for Assignment No. 5, e-mail your Scratch program (only the **.sb** file) to

umlsoundthinking2012@gmail.com. Please be sure to name the file ***exactly*** as shown below, paying particular attention to the uppercase and lowercase letters in the parts that are not your name.

`YourLastName_YourFirstName_PartnerLastName_PartnerFirstName_Assn7_IchiBoardSensors.sb`

The "`_Assn7_IchiBoardSensors.sb`" part is standard and should appear ***exactly*** like that in everyone's submission.

(2) The second part of this assignment is to submit your notes and reflection by answering the questions in the form at http://bit.ly/stassn7. Please review our comments on your previous assignment submissions regarding your reflections and address any issues we mentioned to maximize your grade on this part of the assignment.

How You Will Be Graded

This assignment will be evaluated on the following criteria.

- For the Program
  - Does the piece depend on what the user does with the IchiBoard?
  - Are the more advanced features of the IchiBoard used?
  - Can the aspects of the music be changed *without* changing the value of a variable in the code?
  - Is the program as short as possible?
  - Does the program compensate for any timing issues?
  - Does the program include comments to identify the major sections?

- For the Notes
  - Are your notes thorough and clear?
  - Could someone reproduce your work from the notes you provided?

- For the Reflection
  - Did you think about the various aspects of this assignment and discuss their relation to musical composition?
  - What did you learn about music and what did you learn about computing?
  - What did you learn from the experience of working with this partner?
  - Was your writing and its formatting, grammar, spelling, etc. done professionally?

This is document http://localhost/~heines/91.212/91.212-2011-12s/212-assn/IchiBoards-03.jsp. It was last modified on Saturday, April 21, 2012 at 12:15 PM.

73.212 / 91.212 Reflections for Assignment No. 7: Using IchiBoards and Sensors

Please use this form to post your reflection for the assignment named above. Responses to the questions marked with an asterisk (*) are required.

* Required

Please enter your full name: *

Please enter your partner's full name: *

If there was a third partner on your team, please enter his or her name:

Please enter your notes and reflection by answering the following questions.

Please make sure that your writing and its formatting, grammar, spelling, etc. are done professionally.

(1) Please document what you did to complete this assignment as if you were writing directions for another student to duplicate your work. *

(2) What do you think about the various aspects of this assignment and their relation to musical composition or performance? *

UMass Lowell Depts. of Music and Computer Science

73.212 / 91.212 Sound Thinking

Spring 2012 Semester, Section 201

Prof. Jesse M. Heines and Prof. Gena R. Greher

Site Home
Teaching
Research
Advising
Calendar
Projects

Course Home Information Class Notes Assignments Grades Resources (Instructor)

Assignment No. 8

Final Sound Thinking Project and Performance

Date Due: **Tuesday, May 1, 2012**

Contents

What This Assignment Is About

This assignment is the capstone project in this course. It will culminate in a performance of an original project on May 1, 2012, at The Revolving Museum in Lowell. (Details are in the What You Are To Do section.)

You may pick your own partner(s) for this assignment, but each team must consist of either two or three partners. Thus, you may not work alone, and no team can have more than three omembers.

Your final project can take any one of the following three forms.

1. You can create a music composition in Scratch with multiple voicings that is a variation of an existing, well-known composition. Your program can include both MIDI-generated sounds and pre-recorded sounds that you may have manipulated in some way using Audacity. If you choose this option, you must provide the original score you worked from. The performance of your piece may simply consist of pressing the green flag icon to play it for the audience.

2. You can create a piece that uses the sensors on the IchiBoard to control what is played. This option is an extension of Assignment Nos. 6 and 7 and perhaps 5.

3. You can create an interactive Scratch program that incorporates music in some way. To see examples of such programs, go to http://scratch.mit.edu and search the 2,453,617 projects there (as of Wednesday, April 10, 2012, at 3:20 PM) for music-related programs or, for a more direct approach, check out http://scratch.mit.edu/galleries/view/69197, "Newest Projects in Cool Music Projects."

What You Are To Do

1. Pick a partner or partners.

2. Fill out the form at http://bit.ly/stassn8registration to tell us who is on your team, what you're planning to do, your project's title (for the evening's program), and whether there is a specific time slot in the evening that you prefer to perform.

3. Complete your project with your partners. Feel free to consult Gena and Jesse about the various aspects of your project to get additional ideas for how to implement what you want and/or for cool things to add as enhancements.

4. Prepare to perform your project on May 1, 2012, at The Revolving Museum in Lowell. Here are the details:

   - *date:* Tuesday, May 1
   - *time:* setup may begin at 4:00 PM, presentations begin at 5:00 PM
   - *location:* The Revolving Museum, 290 Jackson Street, Lowell, MA 01852
   - *attendance:* you have to be there!
   - *problems:* call if you're going to be late
      - The Revolving Museum: 978-937-2787
      - Jesse's cell phone: 978-710-9627 (Google Voice, for phone or text messages)

5. Email your final Scratch program to umlsoundthinking2012@gmail.com by Wednesday, May 2, as described in the next section.

6. Fill out the form at http://bit.ly/stassn8 with your notes and reflection by Wednesday, May 2, as described in the next section.

Submitting Your Assignment for Grading

Performances of these final projects will take place on **Tuesday, May 1, 2012, at The Revolving Museum in Lowell**. You are then to submit the final version of your Scratch code and enter your **notes and reflection** on the Google form created for that purpose by **Wednesday, May 2**.

> ***Important Note:*** As stated numerous times in class, no late submissions of this assignment will be accepted. You must perform your project on May 1 and you must submit your final Scratch code and your notes and reflection by Wednesday, May 2. No late submissions will be accepted.

As you did for the last few assignments, email your Scratch program (only the **.sb** file) to umlsoundthinking2012@gmail.com. Please be sure to name the file ***exactly*** as shown below, paying particular attention to the uppercase and lowercase letters in the parts that are not your name.

> `YourLastName_YourFirstName_PartnerLastName_PartnerFirstName_Assn8_FinalProject.sb`

The "`_Assn8_FinalProject.sb`" part is standard and should appear ***exactly*** like that in everyone's submission.

Submit your notes and reflection by answering the questions in the form at http://bit.ly/stassn8.

> ***Important Note:*** The notes and reflection form was not yet available at the time this assignment was written. We will announce in class when the form is available.

Please review our comments on your previous assignment submissions regarding your reflections and address any issues we mentioned to maximize your grade on this part of the assignment.

How You Will Be Graded

This assignment will be evaluated on the following criteria.

- For the Program and Performance
   - Does the piece use a variety of the techniques we've studied this semester?
   - Does it use the more advanced techniques we've studied or just the basic ones?
   - Did your performance go smoothly, indicating that you proacticed it before you had to perform it?
   - Is your Scratch program as short and efficient as possible?
   - Does the program attempt to compensate for any timing issues?
   - Does the program include comments at least to identify the major sections, what they do and how they work?

- For the Notes
   - Are your notes thorough and clear?
   - Could someone reproduce your work from the notes you provided?

- For the Reflection
   - Did you think about the various aspects of this assignment and discuss their relation to musical composition?
   - What did you learn about music and what did you learn about computing?
   - What did you learn from the experience of working with this partner?
   - Was your writing and its formatting, grammar, spelling, etc. done professionally?

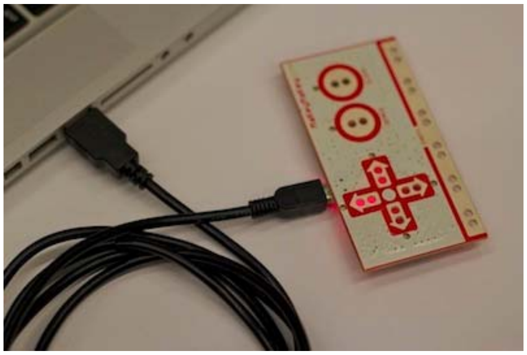
BUY FAQ HOW-TO PROJECTS GUIDES FORUMS ABOUT PRESS

MaKey MaKey Quick Start Guide

Setup | Materials | Software | Troubleshooting

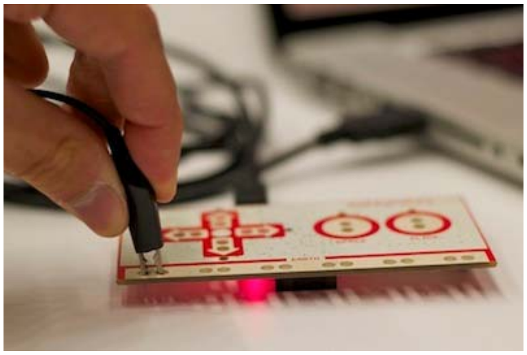
Setup

1) Plug in USB
Small side of USB cable plugs into MaKey MaKey, big side plugs into computer.



2) Close Popup Window
Your computer may ask you to install drivers or do other setup. You can click cancel or close the window.

3) Connect to Earth
Connect one end of an alligator clip to "Earth" on the bottom of the front side of MaKey MaKey.



4) Connect to Yourself
Hold the metal part of the other end of the alligator clip between your fingers. You are now "grounded."

Ever played Mario on Play-Doh or Piano on Bananas? Alligator clip the Internet to Your World.

ORDER NOW

In stock! Ships immediately!

Get Awesome Updates!

Subscribe

Reviews

"four-year-old daughter has managed to connect the kit" ~BBC

"Rejoice!" ~Mashable

"by far the coolest Kickstarter project" ~Kotaku

"turns the whole world into a keybaord" ~Engadget

"a lot of enthusiasm and love" ~Wired

"crazy, inventive experiments" ~PC World

"We love a good diy project" ~LIfehacker

"So small, so quirky, so simple, so awesome." ~Contiki

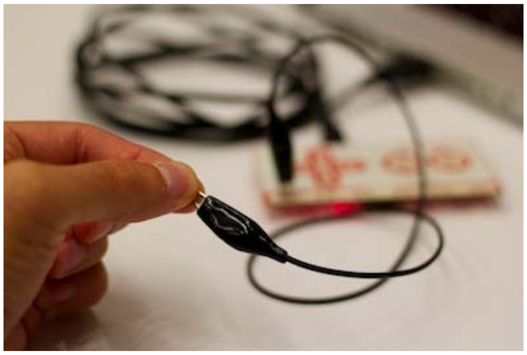
"Mind explosion in progress." ~Indie Cookie

"turns your alphabet soup into a keyboard" ~New Scientist

"Edison meets OK Go" ~Cool Material

Order Your Kit

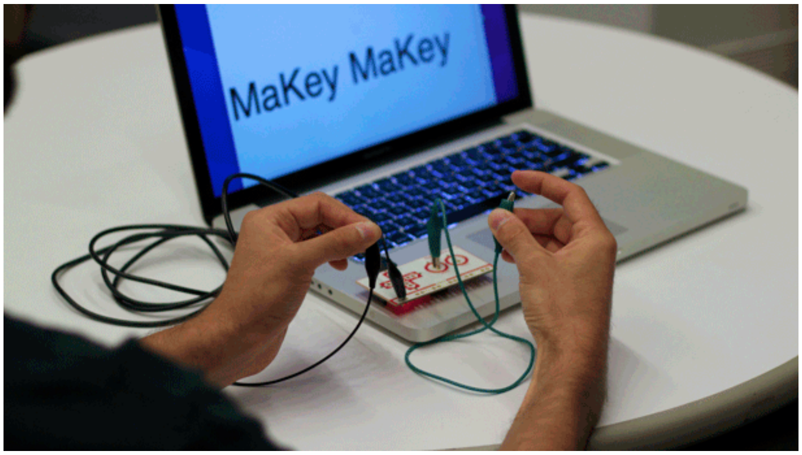
Includes MaKey MaKey, Red USB Cable, 7 Alligator Clips, 6 Connector Wires



5) Connect to "Space" and Try It

While you're still grounded, touch the round "Space" pad on the MaKey MaKey. You should see a green light on the MaKey MaKey, and your computer will think the space bar was pressed. If you click in the text area below, you can make the cursor move. You can also complete the circuit by connecting another alligator clip to "Space."

```
I Luv MaKey MaKey
```

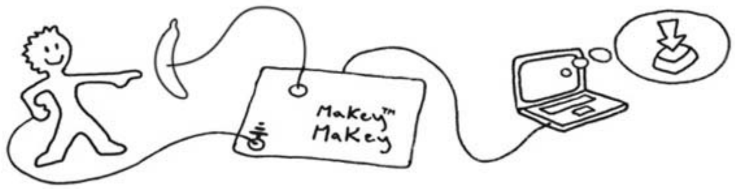


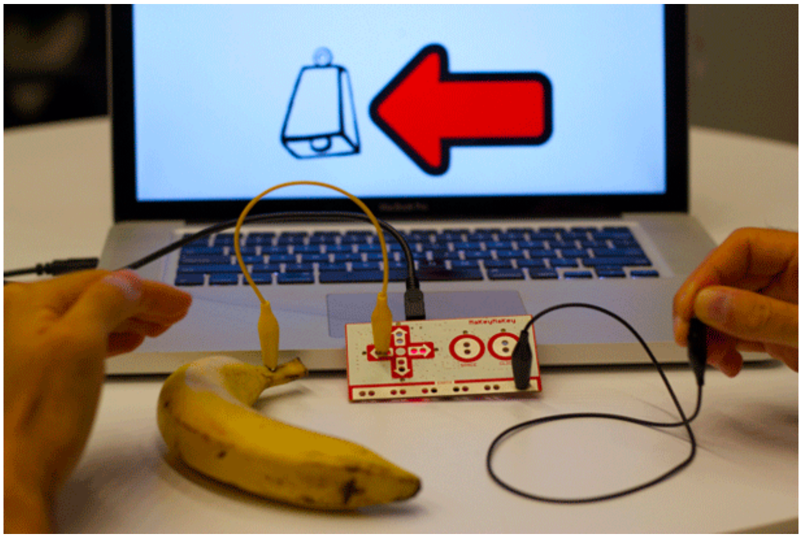
5) Play some MaKey MaKey drums!

Click below to play some drum sounds using the arrow keys and space on your MaKey MaKey.

6) Connect Stuff

Now you can try making your own drums out of anything. How about a banana cowbell, a watermelon bass drum, or a crash cymbal sound when you high-five your friend?





Try Out Different Materials

Make anything into a key! You can make a connection through anything that's even a little bit conductive. You can also create inventions that combine conductive and non-conductive parts.

Conductive Materials
Here are a few things to try:

- Most fruits and vegetables work great.
- Lots of other foods work too. We've tried marshmallows, gummy candies, macaroni and cheese, cupcakes, shrimp, and lots of other things.
- Plants can work too. Try some leaves or flowers, but nothing too dry.
- Play-Doh, Model Magic and other clays work very well as long as they stay moist.
- People are conductive! Connect one person to ground, and another to an input, and you can trigger sounds when they high-five.
- Graphite from a pencil can work. Make thick, dark lines, and be sure to draw on a smooth surface.
- Foil and other metal objects will work. Try out coins, magnets, nuts and bolts, forks and knives, or pots and pans.

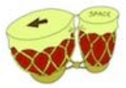
Craft Materials
When you're inventing, anything goes! Here are a few things we have enjoyed using:

Inflatable beach balls, paper plates, cardboard boxes, various types of soft and rigid foam, lego bricks, plastic storage boxes, stuffed animals and other toys, funny hats and other pieces of clothing, sheets of fabric, string, yarn, elastic, and paper.

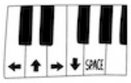
It's also important to have around some tools for cutting, like scissors and exacto knives, and ways to stick things together, like hot glue, superglue, various kinds of tape, and clips or clamps.

Try Out Software

MaKey MaKey works with any software that uses the keyboard or mouse. Here are some ideas to get you started:

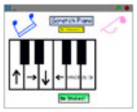


Bongos
Play some bongo drums with the space bar and left arrow. Turn anything into a drum!



Piano
A piano designed for MaKey MaKey. Play a melody with the arrow keys and space bar (and click, too).

Canabalt
This is a running game, with just one button: press the space bar to jump, or invent your own way.



Scratch Piano
This is a piano you can play with the arrow keys. It was created using Scratch, an easy-to-use graphical programming language, so you can remix it and make your own version.

Tetris
You can play this Tetris game with the arrow keys... or make your own custom controller.

Sound Effects
This page turns your keyboard into a sound effects machine. By connecting to the letters on the back of your MaKey MaKey, you can create your own way to trigger sounds.

Flash Flash Revolution
Make your own pads so you can play this dancing game with your feet instead of the arrow keys... or invent a totally new way to play.

ArtCopyCode Connect
Designed just for MaKey MaKey: games, art, music, and some surprises! In order to use each webpage you can ignore their instructions and try the arrows, space, click, or w, a, s, d, f, g to see what works

Chamber Music Piano
You can play 10 different notes on this piano (using w, a, s, d, f, left, up, right, down and space). Play along with the video for a piano duet!

That's just 5 of millions of software apps out there. Go find them, browse the internet, browse Scratch, go make them. Here is a collection of Scratch projects made just for MaKey MaKey. Here is a two-page PDF intro to Scratch and MaKey MaKey.

You can also try out interacting with:

- Powerpoint
- Photobooth
- Music Players

- Video Players (including YouTube)

Here are some good music apps we know of:

- Soundplant
  - Drag and drop sounds onto keys on a keyboard, then press the keys to play them. Try it with the example keymap we made with drum and marimba sounds for mac or windows.
- Garage Band
- Virtual Piano

Here are some good one button games we know of:

- Blog Post with 1-button games
- Flabby Physics

And some awesome multi-button games:

- Nintendo Emulator Site (Remember to customize the controllers)
- Crazy Taxi
- Pac Man

Troubleshooting

I can't get a key to press!

- Make sure your MaKey MaKey is plugged into the computer.
- The MaKey MaKey should have a red light on the back showing that the power is on. Is it lit? If not something is wrong with the computer, the USB cable, or the circuit board.
- If the USB connector on your computer is blue, it's USB 3.0, and may not work with MaKey MaKey. Try using a USB 2.0 port.
- Try making a connection in the simplest way you can. One way is to connect an alligator clip to "earth" and then touch other end to "space."
- When you make a connection, you should see an LED light up on the front of the MaKey MaKey
- When you are making a connection using everyday materials they need to be at least a little bit conductive. For example, play-doh, a banana, your skin, or aluminum foil should work, but plastic, most fabrics or paper will not work directly. You can always combine materials, for example by wetting the paper or putting play-doh onto the plastic.

One of the keys won't stop pressing over and over! What should I do?

- Try unplugging the MaKey MaKey from your computer, then plug it back in again
- Disconnect all of you alligator clips from the MaKey MaKey, then start reattaching them one by one
- If your stuck key is still pressing, have a look at your connections. They might be touching accidentally somehow. Try taking things apart and putting them back together again.
- Perhaps one of the things that you are using as an insulator (or non-conductor) isn't insulating enough. Try using a different material.
- If one of the objects connected to the MaKey MaKey is your own body, then perhaps you are "grounded" to the earth via touching your computer's metal case, or by not wearing shoes. Take a step back and see what you are touching.
- Perhaps one of the objects connected to the MaKey MaKey that you think is well insulated is not. For example, if you connect a banana to your MaKey MaKey and it's sitting on a wooden cutting board, is that cutting board moist or dry? If it's moist then perhaps all of your bananas on the cutting board are connected to each other through the cutting board. Try a dry table instead.
- Is it raining? Is it extremely humid- are you in a rainforest? This can sometimes

cause porous materials, such as paper or clothing, to become conductive.

It works sometimes, but not other times. : (

- Your materials might not be conductive enough. For example, if you are making a connection with your fingertip, it can help to moisten your skin by licking it.
- If you are using a drawn graphite line from a pencil, make sure the line is heavy and dark. Draw your line on a hard smooth surface, such as a table without any grain in it. Take care in folding the paper, because it can sometimes break the graphite connection. Once you are expert at drawing the lines, you don't have to follow any rules, but if you can't get it to work try the tips mentioned.

Submission No. 1

Submission Time Stamp: 3/29/2012 5:08:37 PM

Student Name: {Name removed per IRB regulations}
Partner Name: {Name removed per IRB regulations}
Third Partner: None

Percent of Work Claimed to be Done by This Student: 60 %

Response to Question 1:
Please document what you did to complete this assignment as if you were writing directions for another student to duplicate your work.

1. Under "Variables", hit "Make a List". Name the list "Our Song".

2. Code your melody in the list using offset values, tonic note being 0, so that we may be able to transpose the whole melody with the simple alteration of one parameter. Do not code using absolute note values. If you plan on basing your melody around say, middle C, which is 60, there is a way we can sort of reduce all the number values down to 0, in this case, down by 60, and have them all still sound the same pitch. What we will be getting our code to do is to play our melody based on interval relativity rather than absolute pitch values.

3. Make a new variable. Call it "Starting Note". This will be used to represent tonic.

4. Now add your first block onto the program. Add a "when space is pressed" starting block. The first block in the code will be a "set (variable) to _" block, found under the "Variables" page. Click and drag the new "Starting Note" variable into the (variable) slot and set the value to 60, starting us off in the key of C. The "set variable" block should look like this: set (Starting Note) to 60.

5. Under "Control" find and add a repeat under the "set variable" block. In the space where it asks you how many times you want it to loop, click and drag the "length of (list)" block under the list section of the "Variables" page. Set this list to "Our Song". This will make it so the amount of repeats will be proportionate to the length of the designated list, in this case "Our Song".

6. Make a new variable. Call it "Player".

7. Insert a "play note for _ beats" in between the repeat brackets. Set its beat value to 0.5.

8. Now, insert a "change (variable) by _" under the "play note". Set the variable to "Player", and the value to 1. This block should look like this: change (Player) by (1). Make sure this lies directly under the "play note", still within the repeat brackets.

9. Under "Operators", grab the _+_ block, and drag it into the note value parameter of the "play note" block.

10. Now, under the list section of the "Variables" page, grab the "item _ of (list)" block, and drag it into the first space of our operator. Set the list to "Our Song", and then drag the variable "Player" into the space where it asks us to specify the item number. The whole block should look like this: item (Player) of (Our Song).

11. In the second space of our operator, drag the variable "Starting Note". The whole thing should look like this: play note - item (Player) of (Our Song) + (Starting Note) for 0.5 beats.

12. One final touch that will make this all complete: Above the repeat, can be above or below the first "set variable" block, it doesn't matter, add another "set variable". Set this variable to "Player", and its value to 1. It should look like this: set (Player) to (1). This will make it so when we play our code, your melody will start from the first item in your list, where it should.

13. Now when we press space, it should play each item of your list a half a beat each. If you want you transpose your whole melody up or down a certain degree, all you have to do it simply change the value of the "set (Starting Note)" variable block at the top of the code. It's 60 now, meaning your melody will play in the key of C, but if you want it to play in say, the key of D, change the value to 62.

Response to Question 2:
What do you think about the various aspects of this assignment and their relation to musical composition?

I thought this assignment was a good exercise in recognizing pitch relativity. This was largely attributed to how we were to compose according to the intervallic relationships between the notes, rather than according to absolute note values. It further demonstrates the direct link between music and mathematics, as we see that if we take any monophonic melodic line, and decide to shift every note up one whole-tone, or more generally speaking, by the same amount, we will still recognize it as the same exact melody, just in a different key. This is because the distance between the notes in succession of the melody will remain the same. This relativity of pitches is a very important concept to have mastered if one is to be a serious composer of music.

Response to Question 3:
What did you learn about MUSIC from doing this assignment?

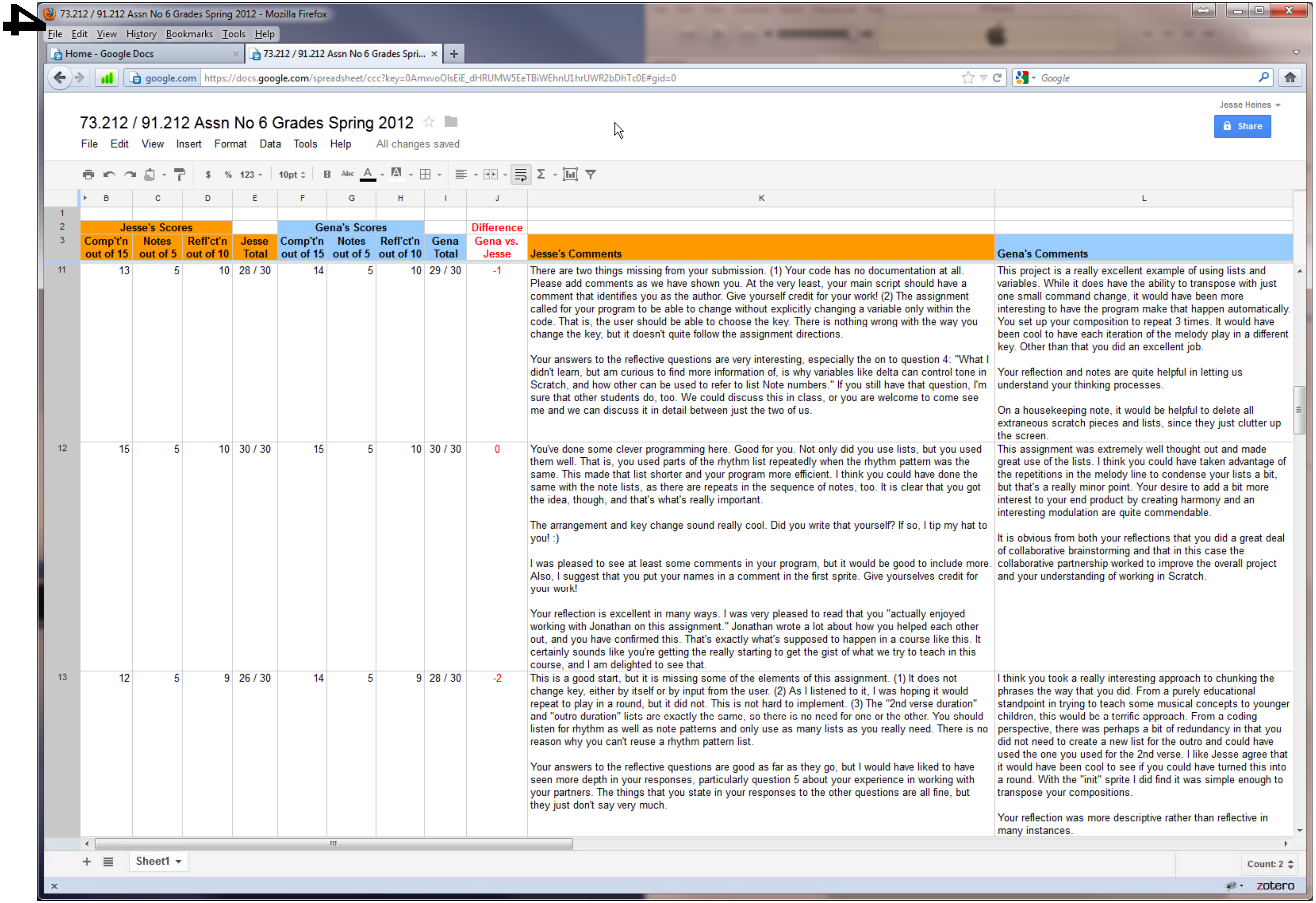
I'm sorry to disappoint you but this section of the reflection is going to be very brief as it usually is. In terms of music and music theory, this course touches on the basic concepts that I have long since learned in my early stages of musical development. I already know most of this stuff. I know much behind how music works. How to get it to work in Scratch however, that is something else entirely, so on that note...

Response to Question 4:
What did you learn about COMPUTING from doing this assignment?

You know, this software is actually starting to grow on me. Scratch, I mean. I think its appeal lies in the sheer mental stimulation that arises from having to apply the utmost critical thinking in the interest of problem solving. Translating a musical idea into a programmed mathematical code is no easy task, and it's certainly something that I have very little experience in. Scratch sure does provide an interesting breakdown of musical ideas, visually and computationally. What I'm seeing more and more throughout these assignments, is the capacity for accomplishing considerably large things with such a minimal amount of code, a quality that I can tell so many advanced programming softwares share. The practicality of having such a feature would of course the simplification. To be most concise and better organized so to not clutter the frame up with an unnecessary amount of junk, making it harder on us. I mean what we did here is program an entire melody, mine spanning 31 notes in length I might add, which then ended up being controlled by a code no larger in size than maybe a sixth of the overall template. And we were able to make it so that by altering just but one parameter, we can change the key of the entire composition. Technology never ceases to amaze me.

Response to Question 5:
What did you learn from the experience of working with this partner?

For the first time this semester, both my partner and I were non-computer science majors working together. My partner, Rachel, is an art major, and as you know, I am a music major. So in essence, we were two art majors banding together to work with computer code, something that we have never bothered getting good at up until now. Normally, to make things easier on me, I have the aid of a computer science major as a counterpart that can provide me with valuable insight as to the computing end of the process. Now, neither of us had any formal training in the art of computer programming, so we had to do all the digging ourselves and go searching far and wide to find what we needed to know. Needless to say, this process called for a lot more independency and self-reliance than any of the other assignments before it. I liked that a lot. I always feel good about myself after demonstrating that, through sheer effort, determination, and strategy, I am perfectly capable of accomplishing something on my own. Of course I had the aid of my partner, who no doubt must have had a similar developmental experience to mine, but still I think the principle applies, even though technically I did not do it 100% on my own. One might think that in a situation like this, us two being just so utterly stumped for so long, that two young college kids might have just given up and called it a night after less than an hour. But we didn't. It took us over three hours at least, but we were determined as hell, and we figured it out. Even though I had a helping hand, I still felt rewarded with that gratifying sense of pride and accomplishment.


73.212 / 91.212 Assn No 6 Grades Spring 2012 - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Home - Google Docs
73.212 / 91.212 Assn No 6 Grades Spri...
google.com https://docs.google.com/spreadsheet/ccc?key=0AmxvoOlsEiE_dHRUMW5EeTBiWEhnU1hrUWR2bDhTc0E#gid=0
Google
Jesse Heines
Share
File Edit View Insert Format Data Tools Help All changes saved
$ % 123 10pt B Abc A
Sheet1
Count: 2
zotero

73.212 / 91.212 Assn No 6 Grades Spring 2012

	Jesse's Scores				Gena's Scores				Difference		
	Comp't'n out of 15	Notes out of 5	Refl'ct'n out of 10	Jesse Total	Comp't'n out of 15	Notes out of 5	Refl'ct'n out of 10	Gena Total	Gena vs. Jesse	Jesse's Comments	Gena's Comments
11	13	5	10	28 / 30	14	5	10	29 / 30	-1	There are two things missing from your submission. (1) Your code has no documentation at all. Please add comments as we have shown you. At the very least, your main script should have a comment that identifies you as the author. Give yourself credit for your work! (2) The assignment called for your program to be able to change without explicitly changing a variable only within the code. That is, the user should be able to choose the key. There is nothing wrong with the way you change the key, but it doesn't quite follow the assignment directions. Your answers to the reflective questions are very interesting, especially the on to question 4: "What I didn't learn, but am curious to find more information of, is why variables like delta can control tone in Scratch, and how other can be used to refer to list Note numbers." If you still have that question, I'm sure that other students do, too. We could discuss this in class, or you are welcome to come see me and we can discuss it in detail between just the two of us.	This project is a really excellent example of using lists and variables. While it does have the ability to transpose with just one small command change, it would have been more interesting to have the program make that happen automatically. You set up your composition to repeat 3 times. It would have been cool to have each iteration of the melody play in a different key. Other than that you did an excellent job. Your reflection and notes are quite helpful in letting us understand your thinking processes. On a housekeeping note, it would be helpful to delete all extraneous scratch pieces and lists, since they just clutter up the screen.
12	15	5	10	30 / 30	15	5	10	30 / 30	0	You've done some clever programming here. Good for you. Not only did you use lists, but you used them well. That is, you used parts of the rhythm list repeatedly when the rhythm pattern was the same. This made that list shorter and your program more efficient. I think you could have done the same with the note lists, as there are repeats in the sequence of notes, too. It is clear that you got the idea, though, and that's what's really important. The arrangement and key change sound really cool. Did you write that yourself? If so, I tip my hat to you! :) I was pleased to see at least some comments in your program, but it would be good to include more. Also, I suggest that you put your names in a comment in the first sprite. Give yourselves credit for your work! Your reflection is excellent in many ways. I was very pleased to read that you "actually enjoyed working with Jonathan on this assignment." Jonathan wrote a lot about how you helped each other out, and you have confirmed this. That's exactly what's supposed to happen in a course like this. It certainly sounds like you're getting the really starting to get the gist of what we try to teach in this course, and I am delighted to see that.	This assignment was extremely well thought out and made great use of the lists. I think you could have taken advantage of the repetitions in the melody line to condense your lists a bit, but that's a really minor point. Your desire to add a bit more interest to your end product by creating harmony and an interesting modulation are quite commendable. It is obvious from both your reflections that you did a great deal of collaborative brainstorming and that in this case the collaborative partnership worked to improve the overall project and your understanding of working in Scratch.
13	12	5	9	26 / 30	14	5	9	28 / 30	-2	This is a good start, but it is missing some of the elements of this assignment. (1) It does not change key, either by itself or by input from the user. (2) As I listened to it, I was hoping it would repeat to play in a round, but it did not. This is not hard to implement. (3) The "2nd verse duration" and "outro duration" lists are exactly the same, so there is no need for one or the other. You should listen for rhythm as well as note patterns and only use as many lists as you really need. There is no reason why you can't reuse a rhythm pattern list. Your answers to the reflective questions are good as far as they go, but I would have liked to have seen more depth in your responses, particularly question 5 about your experience in working with your partners. The things that you state in your responses to the other questions are all fine, but they just don't say very much.	I think you took a really interesting approach to chunking the phrases the way that you did. From a purely educational standpoint in trying to teach some musical concepts to younger children, this would be a terrific approach. From a coding perspective, there was perhaps a bit of redundancy in that you did not need to create a new list for the outro and could have used the one you used for the 2nd verse. I like Jesse agree that it would have been cool to see if you could have turned this into a round. With the "init" sprite I did find it was simple enough to transpose your compositions. Your reflection was more descriptive rather than reflective in many instances.

This message is being sent from Prof. Heines's email account, but it is from Prof. Greher as well as Prof. Heines. If you reply to this message, please be sure to send your reply to both of us.

Subject: 73.212 / 91.212 Grade Report

Assn. No. 6: TRANSPOSING WITH SCRATCH

   Date Due: April 12, 2012

Student: RAN GUO

GRADES

```
  Scores from Jesse
    Composition: 13
          Notes:  5
     Reflection: 10

  Scores from Gena
    Composition: 14
          Notes:  5
     Reflection: 10

    Total Grade: 57 out of 60  for  {Name removed per IRB regulations}
```

COMMENTS

From Jesse:

There are two things missing from your submission. (1) Your code has no documentation at all. Please add comments as we have shown you. At the very least, your main script should have a comment that identifies you as the author. Give yourself credit for your work! (2) The assignment called for your program to be able to change without explicitly changing a variable only within the code. That is, the user should be able to choose the key. There is nothing wrong with the way you change the key, but it doesn't quite follow the assignment directions.

Your answers to the reflective questions are very interesting, especially the on to question 4: "What I didn't learn, but am curious to find more information of, is why variables like delta can control tone in Scratch, and how other can be used to refer to list Note numbers." If you still have that question, I'm sure that other students do, too. We could discuss this in class, or you are welcome to come see me and we can discuss it in detail between just the two of us.

From Gena:

This project is a really excellent example of using lists and variables. While it does have the ability to transpose with just one small command change, it would have been more interesting to have the program make that happen automatically. You set up your composition to repeat 3 times. It would have been cool to have each iteration of the melody play in a different key. Other than that you did an excellent job.

Your reflection and notes are quite helpful in letting us understand your thinking processes.

On a housekeeping note, it would be helpful to delete all extraneous scratch pieces and lists, since they just clutter up the screen.

```
Option Explicit

' 91.212 GradingSpreadsheetMacros.xls
' updated by JMH on April 21, 2012


' set to number of students in class
Dim arrNames(34, 3) As String
Dim lf As String        ' the line feed character, abbreviated for convenience


' this subroutine initialize a 2D array of student names, email addresses, and
'   report file names.
Private Sub Initialize()

    lf = Chr(10)    ' the line feed character, abbreviated for convenience

    ' Spring 2012

    ' array format
    ' arrNames(1, 1) = "LastName, FirstName"
    ' arrNames(1, 2) = "FirstName_LastName@student.uml.edu"
    ' arrNames(1, 3) = "LastName.txt"

    ' actual names removed per IRB regulations

End Sub


' this subroutine writes the last update date in cell A1
' created by JMH on February 9, 2006
Private Sub SetLastUpdateDate()
    Range("A1").Font.Italic = True
    Range("A1").Value = "last updated on " & _
        Format(FileDateTime(ActiveWorkbook.FullName), "mmmm d, yyyy") & _
        " at " & _
        Format(FileDateTime(ActiveWorkbook.FullName), "h:mm AM/PM")
End Sub


' this function generates and returns an ISO8601 date from a short date string
'   (vbShortDate)
Private Function ISO8601(d As String)
    If InStr(d, "/") < 3 Then
        d = "0" & d
    End If
    If InStr(InStr(d, "/") + 1, d, "/") < 6 Then
        d = Left(d, 3) & "0" & Right(d, 6)
    End If
    ISO8601 = Right(d, 4) & "-" & Left(d, 2) & "-" & Mid(d, 4, 2)
End Function


' this function performs a super Trim by removing newline characters that the
'   built-in Trim function does not
' updated by JMH on February 25, 2010 at 2:14 PM
Private Function jTrim(str)
    Dim ret As String   ' string to return
    Dim c As Integer    ' character to test

    'initializations
    ret = Trim(str)     ' do a normal Trim to remove leading & trailing spaces
    c = 0
    If (Len(ret) > 0) Then
        c = Asc(Mid(ret, Len(ret), 1))  ' initialize character to test
    End If

    ' remove spaces (char 32) and all characters below a space from the end of the
    '   string
    While c < 33 And Len(ret) > 0
        ret = Mid(ret, 1, Len(ret) - 1)
        c = Asc(Mid(ret, Len(ret), 1))
    Wend

    ' return result
    jTrim = ret
End Function


' this function takes a comma-separated "last name, first name" string and returns
'   the names in "first name last name" order
Private Function FirstNameFirst(ByVal str As String) As String
    Dim ret As String       ' string to return
    Dim pComma As Integer   ' position of comma in input string

    If Len(str) = 0 Then
        FirstNameFirst = ""
    Else
        'find position of comma and reconstruct string
        pComma = InStr(1, str, ",")
        ret = Mid(str, pComma + 2) & " " & Mid(str, 1, pComma - 1)
        FirstNameFirst = ret
    End If

End Function


' this function wraps a string to a given line width, using a given string
'   (the header) to indent each line if specified
Private Function WrapText4(ByVal str As String, ByVal width As Integer, _
        ByVal header As String) As String

    Dim result As String    ' string to display
    result = ""             ' initialization

    Dim k As Integer        ' loop index

    Dim paras() As String   ' array of paragraphs
    str = Replace(str, "  ", "~ ")
    paras = GetParas(str)   ' split text on Chr(10) = line feeds

    ' for debugging purposes only
    If False Then
        For k = 0 To UBound(paras)
            result = result + ProcessPara1((k + 1), Trim(paras(k)), width, header)
        Next k
        result = "Number of paragraphs = " & (UBound(paras) + 1) & result & _
            lf & "-----------" & lf
    End If

    ' process each paragraph in turn
    For k = 0 To UBound(paras)
        result = result + ProcessPara2((k + 1), Trim(paras(k)), width, header)
        If k <> UBound(paras) Then
            result = result & lf
        End If
    Next k

    WrapText4 = result
End Function


' this is Version 2 of the ProcessPara function
' it formats a single paragraph by wrapping it to a given width, using a given
'   string (the header) to indent each line if specified
Private Function ProcessPara2(ByVal nPara As Integer, ByVal para As String, _
        ByVal width As String, ByVal header As String) As String

    Dim tokens() As String  ' individual words
    tokens = Split(para, Chr(32))

    Dim ret As String       ' return value
    ret = ""

    Dim line As String      ' a single line of output
    line = header           ' initialize first line

    Dim k As Integer        ' loop index
    For k = 0 To UBound(tokens)
        ' the Replace decodes tilde characters back to spaces
        'ret = ret & Chr(10) & (k + 1) & ". |" & tokens(k) & "|"

        If Len(line) = Len(header) Then
            line = line & tokens(k)
        ElseIf Len(line) + Len(tokens(k)) + 1 <= width Then
            line = line & " " & tokens(k)
        Else
            ret = ret & line
            line = lf & header & tokens(k)
        End If
    Next k

    ' add last line
    ret = ret & line

    ' the Replace decodes tilde characters back to spaces
    ret = Replace(ret, "~", " ")

    ProcessPara2 = ret
End Function


' this function returns an array of all the paragraphs in a string
Private Function GetParas(ByVal str As String)
    Dim paras() As String
    Dim k1, k2 As Integer

    paras = Split(str, Chr(10))
    ' test each paragraph to see if it begins with a space,
    ' and if it does, replace all spaces with tildes to preserve them
    For k1 = 0 To UBound(paras)
        If Mid(paras(k1), 1, 1) = Chr(32) Then
            paras(k1) = Replace(paras(k1), Chr(32), "~")
        End If
    Next k1
    GetParas = paras
End Function


' this function takes a student name as a string and returns his or her file name
' updated by JMH on February 25, 2010 at 2:14 PM
Private Function GetStudentFileName(ByVal strStudentName As String) As String
    Dim k As Integer                        ' counter variable
    Dim ret As Variant                      ' MsgBox return value
    strStudentName = jTrim(strStudentName)  ' student name to search for

    ' sequential search
    For k = 1 To UBound(arrNames, 1)
        If (arrNames(k, 1) = strStudentName) Then
            GetStudentFileName = arrNames(k, 3)
            Exit Function
        End If
    Next k

    ' skip students who have withdrawn
    If (strStudentName = "No. of Submissions Graded So Far") Then
        ret = "Withdrawn.txt"
        Exit Function
    End If


     ' if reached here, student name was not found
    GetStudentFileName = "NotFound.txt"
    ret = MsgBox("Student name """ & strStudentName & """ was not found.", vbOKCancel)
    If (ret = vbCancel) Then
        End
    End If
End Function


' this subroutine determines the appropriate drive for my systems
'   and then writes grade reports to individual student files on that disk
' updated by JMH on August 24, 2009 at 10:58 AM
Public Sub WriteReportstoDisk()
    Initialize

    Dim strSys As String
    strSys = Environ("COMPUTERNAME")
    If strSys = "DEXTER" Or strSys = "XCELSIOR" Or strSys = "ACER" Then
        WriteReports ("C")
    ElseIf strSys = "JUDY" Or strSys = "PHILIP" Then
        WriteReports ("D")
    ElseIf strSys = "ABRAHAM" Then
        WriteReports ("D")
    End If
End Sub


' this subroutine determines the appropriate drive for my systems
'   and then calls WriteReflections to write students' reflections to a file
'   on the appropriate drive for my systems
' updated by JMH on March 21, 2012 at 4:05 PM
Public Sub WriteReflectionstoDisk()
    Initialize

    Dim strSys As String
    strSys = Environ("COMPUTERNAME")
    If strSys = "DEXTER" Or strSys = "XCELSIOR" Or strSys = "ACER" Then
        WriteReflections ("C")
    ElseIf strSys = "JUDY" Or strSys = "PHILIP" Then
        WriteReflections ("D")
    ElseIf strSys = "ABRAHAM" Then
        WriteReflections ("D")
    End If
End Sub


' this subroutine writes reports to files on the C Drive
' updated by JMH on November 29, 2008 at 10:39 AM
Private Sub WriteReportstoC()
    WriteReports ("C")
End Sub


' this subroutine writes reports to files on the D Drive
' updated by JMH on November 29, 2008 at 10:39 AM
Private Sub WriteReportstoD()
    WriteReports ("D")
End Sub


' this subroutine does the actual report writing
' updated by JMH on February 25, 2010 at 2:17 PM
' updated by JMH on February 22, 2012 at 8:28 AM
Private Sub WriteReports(DriveLetter As String)

    Const AssnNo As Integer = 6         ' set this number to the assignment to be output
    Const MaxScore As Integer = 60      ' set this number to the maximum score for the
                                        '   current assignment

    Const Prof1 = "Jesse"               ' co-professor #1
    Const Prof2 = "Gena"                ' co-professor #2
    Const bShowGrades = True            ' set to false to suppress printing of grades
    Const bShowProf2Comments = True     ' set to false to suppress printing of Professor
```

```
                                        '   #2's comments

    Dim AssnName As String              ' name of assignment
    Dim AssnDate As String              ' Month Day, Year that assignment is due

    Select Case AssnNo
        Case 1
            AssnName = "Creating a Composition for a Found Objects Instrument"
            AssnDate = "January 31, 2012"
        Case 2
            AssnName = "Creating a Composition from Digitized Found Sounds"
            AssnDate = "February 14, 2012"
        Case 3
            AssnName = "Creating a Song Flowchart"
            AssnDate = "February 21, 2012"
        Case 4
            AssnName = "Sequencing Sounds with Scratch"
            AssnDate = "March 1, 2012"
        Case 5
            AssnName = "Creating a Composition Based on Major Seconds and Perfect Fifths"
            AssnDate = "March 8, 2012"
        Case 6
            AssnName = "Transposing with Scratch"
            AssnDate = "April 12, 2012"
        Case 7
            AssnName = "Using IchiBoards and Sensors"
            AssnDate = "April 21, 2012"
        Case 8
            AssnName = "Final Sound Thinking Project and Performance"
            AssnDate = "May 3, 2012"
    End Select

    Const nStartRow As Integer = 4  ' row containing first student name
    Const ForReading = 1, ForWriting = 2, ForAppending = 3

    Dim counter As Integer          ' counter variable
    Dim fso As Object               ' file system object
    Dim f As Object                 ' file stream object
    Dim row, col As Integer         ' loop control variables
    Dim strFileName As String       ' name of file to write
    Dim strPath As String           ' path to write files to
    Dim strStudentName As String    ' student name read from spreadsheet
    Dim strMsg As String            ' message to print at top of report
    Dim intScore As Integer         ' student's total score

    ' initialization
    strPath = ActiveWorkbook.Path & "\"

    ' loop through all students
    counter = 0

    For row = nStartRow To nStartRow + UBound(arrNames, 1) - 1
        strStudentName = jTrim(ActiveSheet.Cells(row, 1).Value)
        'strFileName = GetStudentFileName(ActiveSheet.Cells(row, 1).Value)

        ' skip students who have withdrawn
        If (strStudentName = "" Or strStudentName = "No. of Submissions Graded So Far") Then
            GoTo EndOfLoop
        End If

        strFileName = GetStudentFileName(strStudentName)
        'MsgBox row & ". " & jTrim(strStudentName) & " -> " & strFileName

        ' open output file for writing
        Set fso = CreateObject("Scripting.FileSystemObject")
        Set f = fso.OpenTextFile(strPath + strFileName, ForWriting, True)

        ' additional notes to students
        strMsg = "This message is being sent from Prof. Heines's email account, " & _
            "but it is from Prof. Greher as well as Prof. Heines.  If you reply " & _
            "to this message, please be sure to send your reply to both of us." ' & _
            Chr(10) & Chr(10) & _
            ' "Regarding reflections on past assignments posted at the end of " & _
            ' "the semester, please note that as discussed in class, we did not " & _
            ' "accept any that were older than Assignment No. 7.  Reflections " & _
            ' "were due at the same time as the assignment submission."
        f.writeline WrapText4(strMsg, 70, "")
        f.writeline
        f.writeline

        ' write page header with course number
        f.writeline "Subject: 73.212 / 91.212 Grade Report"
        f.writeline

        ' write assignment title and due date
        f.writeline "Assn. No. " & AssnNo & ": " + UCase(AssnName)
        f.writeline
        f.writeline "   Date Due: " & AssnDate
        f.writeline
        f.writeline

        ' write student name in uppercase, first name first, then last name
        f.writeline "Student:  " + FirstNameFirst(UCase(strStudentName))
        f.writeline
        f.writeline

        ' gate printing of grades
        If bShowGrades Then

            ' write grades section title
            f.writeline "GRADES"
            f.writeline

            ' special note for Assignment No. 5 in the 2010 Spring semester
            'If AssnNo = 5 Then
            '    f.writeline "  Please Note:"
            '    f.writeline WrapText("If you performed your piece in class, you " & _
            '        "will have three grades from Jesse and three from Alex, each " & _
            '        "worth 8 points, for a total of 24.  If you did not perform " & _
            '        "your piece in class, you will have two grades from each of us, " & _
            '        "each worth 12 points, for the same total of 24.", _
            '        70, "    ")
            '    f.writeline
            'End If

            ' write scores from Professor #1
            f.writeline "  Scores from " & Prof1
            If AssnNo = 1 Then
                f.writeline "     Instrument: " & Format(ActiveSheet.Cells(row, 2).Value & "", "@@")
                f.writeline "    Composition: " & Format(ActiveSheet.Cells(row, 3).Value & "", "@@")
                f.writeline "       Notation: " & Format(ActiveSheet.Cells(row, 4).Value & "", "@@")
                f.writeline "     Reflection: " & Format(ActiveSheet.Cells(row, 5).Value & "", "@@")
            ElseIf AssnNo = 2 Then
                f.writeline "    Composition: " & Format(ActiveSheet.Cells(row, 2).Value & "", "@@")
                f.writeline "          Notes: " & Format(ActiveSheet.Cells(row, 3).Value & "", "@@")
                f.writeline "     Reflection: " & Format(ActiveSheet.Cells(row, 4).Value & "", "@@")
            ElseIf AssnNo = 3 Then
                f.writeline "      Flowchart: " & Format(ActiveSheet.Cells(row, 2).Value & "", "@@")
                f.writeline "          Notes: " & Format(ActiveSheet.Cells(row, 3).Value & "", "@@")
                f.writeline "     Reflection: " & Format(ActiveSheet.Cells(row, 4).Value & "", "@@")
            ElseIf AssnNo = 4 Or AssnNo = 6 Then
                f.writeline "    Composition: " & Format(ActiveSheet.Cells(row, 2).Value & "", "@@")
                f.writeline "          Notes: " & Format(ActiveSheet.Cells(row, 3).Value & "", "@@")
                f.writeline "     Reflection: " & Format(ActiveSheet.Cells(row, 4).Value & "", "@@")
            ElseIf AssnNo = 5 Or AssnNo = 7 Then
                f.writeline "    Composition: " & Format(ActiveSheet.Cells(row, 2).Value & "", "@@")
                f.writeline "     Reflection: " & Format(ActiveSheet.Cells(row, 3).Value & "", "@@")
            ElseIf AssnNo = 8 Then
                f.writeline "     Submission: " & Format(ActiveSheet.Cells(row, 2).Value & "", "@@")
                f.writeline "     Reflection: " & Format(ActiveSheet.Cells(row, 3).Value & "", "@@")
            End If
            f.writeline

            ' write scores from Professor #2
            f.writeline "  Scores from " & Prof2
            If AssnNo = 1 Then
                f.writeline "     Instrument: " & Format(ActiveSheet.Cells(row, 7).Value & "", "@@")
                f.writeline "    Composition: " & Format(ActiveSheet.Cells(row, 8).Value & "", "@@")
                f.writeline "       Notation: " & Format(ActiveSheet.Cells(row, 9).Value & "", "@@")
                f.writeline "     Reflection: " & Format(ActiveSheet.Cells(row, 10).Value & "", "@@")
            ElseIf AssnNo = 2 Then
                f.writeline "    Composition: " & Format(ActiveSheet.Cells(row, 6).Value & "", "@@")
                f.writeline "          Notes: " & Format(ActiveSheet.Cells(row, 7).Value & "", "@@")
                f.writeline "     Reflection: " & Format(ActiveSheet.Cells(row, 8).Value & "", "@@")
            ElseIf AssnNo = 3 Then
                f.writeline "      Flowchart: " & Format(ActiveSheet.Cells(row, 6).Value & "", "@@")
                f.writeline "          Notes: " & Format(ActiveSheet.Cells(row, 7).Value & "", "@@")
                f.writeline "     Reflection: " & Format(ActiveSheet.Cells(row, 8).Value & "", "@@")
            ElseIf AssnNo = 4 Or AssnNo = 6 Then
                f.writeline "    Composition: " & Format(ActiveSheet.Cells(row, 6).Value & "", "@@")
                f.writeline "          Notes: " & Format(ActiveSheet.Cells(row, 7).Value & "", "@@")
                f.writeline "     Reflection: " & Format(ActiveSheet.Cells(row, 8).Value & "", "@@")
            ElseIf AssnNo = 5 Or AssnNo = 7 Then
                f.writeline "    Composition: " & Format(ActiveSheet.Cells(row, 5).Value & "", "@@")
                f.writeline "     Reflection: " & Format(ActiveSheet.Cells(row, 6).Value & "", "@@")
            ElseIf AssnNo = 8 Then
                f.writeline "     Submission: " & Format(ActiveSheet.Cells(row, 2).Value & "", "@@")
                f.writeline "     Reflection: " & Format(ActiveSheet.Cells(row, 3).Value & "", "@@")
            End If
            f.writeline

            ' write total and maximum scores and student name for extraction by makeg batch file
            intScore = 0
            If AssnNo = 1 Then
                ' f.writeline "    Total Grade: " & Format(ActiveSheet.Cells(row, 5).Value + _
                '     ActiveSheet.Cells(row, 11).Value & "", "@@") & " out of " & MaxScore & _
                '     "  for  " & strStudentName
                For col = 2 To 10
                    If (col <> 6) Then
                        intScore = intScore + ActiveSheet.Cells(row, col).Value
                    End If
                Next
                f.writeline "    Total Grade: " & Format(intScore & "", "@@") & " out of " & _
                    MaxScore & "  for  " & strStudentName
            ElseIf AssnNo = 2 Or AssnNo = 3 Or AssnNo = 4 Or AssnNo = 6 Then
                ' f.writeline "    Total Grade: " & _
                ' Format(ActiveSheet.Cells(row, 5).Value + ActiveSheet.Cells(row, 9).Value & "", "@@"
                ' " out of " & MaxScore & "  for  " & strStudentName
                For col = 2 To 8
                    If (col <> 5) Then
                        On Error Resume Next
                        intScore = intScore + ActiveSheet.Cells(row, col).Value
                    End If
                Next
                f.writeline "    Total Grade: " & Format(intScore & "", "@@") & " out of " & _
                    MaxScore & "  for  " & strStudentName
            ElseIf AssnNo = 5 Then
                For col = 2 To 6
                    If (col <> 4) Then
                        On Error Resume Next
                        intScore = intScore + ActiveSheet.Cells(row, col).Value
                    End If
                Next
                f.writeline "    Total Grade: " & Format(intScore & "", "@@") & " out of " & _
                    MaxScore & "  for  " & strStudentName
            ElseIf AssnNo = 7 Or AssnNo = 8 Then
                f.writeline "    Total Grade: " & Format(ActiveSheet.Cells(row, 4).Value + _
                ActiveSheet.Cells(row, 7).Value & "", "@@") & " out of " & MaxScore & "  for  " & _
                strStudentName
            End If
            f.writeline
            f.writeline

        End If  ' If bShowGrades Then

        ' write comments from Professor #1
        f.writeline "COMMENTS"
        f.writeline
        f.writeline "  From " & Prof1 & ":"
        If AssnNo = 1 Then
            f.writeline WrapText4(ActiveSheet.Cells(row, 13).Value, 70, "    ")
        ElseIf AssnNo = 2 Or AssnNo = 3 Or AssnNo = 4 Or AssnNo = 6 Then
            f.writeline WrapText4(ActiveSheet.Cells(row, 11).Value, 70, "    ")
        ElseIf AssnNo = 5 Then
            f.writeline WrapText4(ActiveSheet.Cells(row, 9).Value, 70, "    ")
        ElseIf AssnNo = 7 Or AssnNo = 8 Then
            f.writeline WrapText4(ActiveSheet.Cells(row, 10).Value, 70, "    ")
        End If
        f.writeline

        ' gate printing of Prof #2's comments
        If bShowProf2Comments Then
            ' write comments from Professor #2
            f.writeline "  From " & Prof2 & ":"
            If AssnNo = 1 Then
                f.writeline WrapText4(ActiveSheet.Cells(row, 14).Value, 70, "    ")
            ElseIf AssnNo = 2 Or AssnNo = 3 Or AssnNo = 4 Or AssnNo = 6 Then
                f.writeline WrapText4(ActiveSheet.Cells(row, 12).Value, 70, "    ")
            ElseIf AssnNo = 5 Then
                f.writeline WrapText4(ActiveSheet.Cells(row, 10).Value, 70, "    ")
            ElseIf AssnNo = 7 Or AssnNo = 8 Then
                f.writeline WrapText4(ActiveSheet.Cells(row, 11).Value, 70, "    ")
            ElseIf AssnNo = 99 Then
                f.writeline WrapText4("(to be added)", 70, "    ")
            End If
        End If

        ' close file
        f.Close
        counter = counter + 1

EndOfLoop:
    Next

    MsgBox counter & " files written to:  " & Chr(10) & "   " & Mid(strPath, 1, Len(strPath) - 1)
End Sub


' this subroutine performs a QuickSort on an array
' source:  http://stackoverflow.com/questions/152319/vba-array-sort-function
' downloaded:  April 10, 2012

Private Sub QuickSort(vArray As Variant, inLow As Long, inHi As Long)

    Dim pivot   As Variant
    Dim tmpSwap As Variant
    Dim tmpLow  As Long
    Dim tmpHi   As Long

    tmpLow = inLow
    tmpHi = inHi

    pivot = vArray((inLow + inHi) \ 2)

    While (tmpLow <= tmpHi)

       While (vArray(tmpLow) < pivot And tmpLow < inHi)
          tmpLow = tmpLow + 1
       Wend

       While (pivot < vArray(tmpHi) And tmpHi > inLow)
          tmpHi = tmpHi - 1
       Wend

       If (tmpLow <= tmpHi) Then
          tmpSwap = vArray(tmpLow)
          vArray(tmpLow) = vArray(tmpHi)
          vArray(tmpHi) = tmpSwap
          tmpLow = tmpLow + 1
          tmpHi = tmpHi - 1
       End If

    Wend

    If (inLow < tmpHi) Then QuickSort vArray, inLow, tmpHi
    If (tmpLow < inHi) Then QuickSort vArray, tmpLow, inHi

End Sub


' this subroutine performs a QuickSort on a 2D array
' http://stackoverflow.com/questions/152319/vba-array-sort-function
' http://en.allexperts.com/q/Visual-Basic-1048/string-manipulation.htm
' modifed by JMH to handle 2D arrays
' April 10, 2012

Private Sub QuickSort2(vArray As Variant, inLow As Long, inHi As Long)

    Dim pivot    As Variant
    Dim tmpSwap1 As Variant
    Dim tmpSwap2 As Variant
    Dim tmpLow   As Long
    Dim tmpHi    As Long

    tmpLow = inLow
    tmpHi = inHi

    pivot = vArray((inLow + inHi) \ 2, 1)

    While (tmpLow <= tmpHi)

       While (vArray(tmpLow, 1) < pivot And tmpLow < inHi)
          tmpLow = tmpLow + 1
       Wend

       While (pivot < vArray(tmpHi, 1) And tmpHi > inLow)
          tmpHi = tmpHi - 1
       Wend

       If (tmpLow <= tmpHi) Then
          tmpSwap1 = vArray(tmpLow, 1)
          tmpSwap2 = vArray(tmpLow, 2)
          vArray(tmpLow, 1) = vArray(tmpHi, 1)
          vArray(tmpLow, 2) = vArray(tmpHi, 2)
          vArray(tmpHi, 1) = tmpSwap1
          vArray(tmpHi, 2) = tmpSwap2
          tmpLow = tmpLow + 1
          tmpHi = tmpHi - 1
       End If

    Wend

    If (inLow < tmpHi) Then QuickSort2 vArray, inLow, tmpHi
    If (tmpLow < inHi) Then QuickSort2 vArray, tmpLow, inHi

End Sub



' this function does the actual writing of reflections to disk
' updated by JMH on March 21, 2012 at 4:07 PM
Private Sub WriteReflections(DriveLetter As String)

    Const AssnNo As Integer = 6         ' set this to the assignment number to be output

    Const MaxScore As Integer = 40      ' set this to the maximum score for the current assignment
    Const Prof1 = "Jesse"               ' co-professor #1
    Const Prof2 = "Gena"                ' co-professor #2
    Const bShowGrades = True            ' set to false to suppress printing of grades
    Const bShowProf2Comments = True     ' set to false to suppress printing of Professor #2's
                                        '     comments

    Dim AssnName As String              ' name of assignment
    Dim AssnDate As String              ' Month Day, Year that assignment is due
    Dim nFirstQuestionColumn As Integer ' number of first column containing a reflection response
    Dim nLastQuestionColumn As Integer  ' number of last column containing a reflection response

    Select Case AssnNo
        Case 1
            AssnName = "Creating a Composition for a Found Objects Instrument"
            AssnDate = "January 31, 2012"
        Case 2
            AssnName = "Creating a Composition from Digitized Found Sounds"
            AssnDate = "February 14, 2012"
        Case 3
            AssnName = "Creating a Song Flowchart"
            AssnDate = "February 21, 2012"
        Case 4
            AssnName = "Sequencing Sounds with Scratch"
            AssnDate = "March 1, 2012"
        Case 5
            AssnName = "Creating a Composition Based on Major Seconds and Perfect Fifths"
            AssnDate = "March 8, 2012"
            nFirstQuestionColumn = 5
            nLastQuestionColumn = 8
        Case 6
            AssnName = "Transposing with Scratch"
            AssnDate = "April 12, 2012"
            nFirstQuestionColumn = 5
            nLastQuestionColumn = 10
        Case 7
            AssnName = "Using IchiBoards and Sensors"
            AssnDate = "April 21, 2012"
            nFirstQuestionColumn = 5
            nLastQuestionColumn = 10
        Case 8
            AssnName = "Final Sound Thinking Project and Performance"
            AssnDate = "May 3, 2012"
    End Select

    Const nStartRow As Integer = 2  ' row containing first student name
    Const ForReading = 1, ForWriting = 2, ForAppending = 3

    Dim counter As Integer          ' counter variable
    Dim fso As Object               ' file system object
    Dim f As Object                 ' file stream object
    Dim row, col As Integer         ' loop control variables
    Dim strFileName As String       ' name of file to write
    Dim strPath As String           ' path to write files to
    Dim strStudentName As String    ' student name read from spreadsheet
    Dim strMsg As String            ' message to print at top of report
    Dim intScore As Integer         ' student's total score
    Dim str As String               ' temporary string to output
    Dim nStudents As Integer        ' number of student submissions

    Dim USDate As String            ' date in mm/dd/yyyy format
    Dim SimpleTime As String        ' time in hh:mm:ss format
    Dim k1 As Integer               ' result of call to Mid function

    ' initialization
    USDate = FormatDateTime(Now(), vbShortDate)
    SimpleTime = FormatDateTime(Now(), vbLongTime)
    k1 = InStr(SimpleTime, ":")
    SimpleTime = Mid(SimpleTime, 1, k1 + 2) + Right(SimpleTime, 3)

    strPath = ActiveWorkbook.Path & "\"
    strFileName = "73212_91212 Reflections for Assignment No. " & AssnNo & " as of " & _
        ISO8601(USDate) & ".txt"
    nStudents = 0

    ' open output file for writing
    Set fso = CreateObject("Scripting.FileSystemObject")
    Set f = fso.OpenTextFile(strPath + strFileName, ForWriting, True)

    ' write page header with course number
    f.writeline "73.212 / 91.212 Reflections for Assignment No. " & AssnNo & " as of " & _
        USDate & " at " & SimpleTime
    f.writeline

    ' collect names for index
    Dim Names(100, 2) As String
    row = nStartRow
    While (ActiveSheet.Cells(row, 1).Value <> "")
        Names(row - nStartRow + 1, 1) = UCase(ActiveSheet.Cells(row, 2).Value)
        Names(row - nStartRow + 1, 2) = "" & (row - nStartRow + 1)
        row = row + 1
    Wend

    ''' write index
    f.writeline "================================================================================"
    f.writeline
    f.writeline "Numbers of Student Submissions Sorted by First Name"
    f.writeline

    ' loop through all submissions to write index
    ' writing in original order
    If (False) Then
        For k1 = 1 To row - nStartRow
            If (Names(k1, 2) < 10) Then f.write (" ")
            f.writeline Names(k1, 2) & ". " & Names(k1, 1)
        Next k1
        f.writeline
        f.writeline "================================================================================
        f.writeline
    End If

    ' sort in alphabetical orcer
    QuickSort2 Names, 1, row - nStartRow

    ' write in sorted order
    For k1 = 1 To row - nStartRow
        If (Names(k1, 2) < 10) Then f.write (" ")
        f.writeline Names(k1, 2) & ". " & UCase(Names(k1, 1))
    Next k1

    ''' loop through all submissions to write reflections
    row = nStartRow

    While (ActiveSheet.Cells(row, 1).Value <> "")
        f.writeline
        f.writeline "================================================================================
        f.writeline
        f.writeline "Submission No. " & (nStudents + 1)
        f.writeline
        f.writeline "Submission Time Stamp:" & WrapText4(ActiveSheet.Cells(row, 1).Value, 80, "  ")
        f.writeline
        f.writeline " Student Name:" & WrapText4(UCase(ActiveSheet.Cells(row, 2).Value), 80, "  ")
        f.writeline " Partner Name:" & WrapText4(ActiveSheet.Cells(row, 3).Value, 80, "  ")

        str = ActiveSheet.Cells(row, 4).Value
        If (str = "") Then str = "None"
        f.writeline "Third Partner:" + WrapText4(str, 80, "  ")

        f.writeline
        f.writeline "Percent of Work Claimed to be Done by This Student:" & _
            WrapText4(ActiveSheet.Cells(row, nLastQuestionColumn).Value & "0 %", 80, "  ")

        counter = 1
        For col = nFirstQuestionColumn To nLastQuestionColumn - 1
            f.writeline Chr(10) & "Response to Question " & counter & ":"
            f.writeline WrapText4(Mid(ActiveSheet.Cells(1, col).Value, 5), 80, "")
            f.writeline
            f.writeline WrapText4(ActiveSheet.Cells(row, col).Value, 80, "    ")
            counter = counter + 1
        Next col

        nStudents = nStudents + 1
        row = row + 1
    Wend

    'If True Then
        f.writeline
        f.writeline "================================================================================
        f.Close
        'GoTo EndOfSub
    'End If

'EndOfSub:
    MsgBox nStudents & " records written to:  " & Chr(10) & "   " & _
        Mid(strPath, 1, Len(strPath) - 1)
End Sub

```

CREATE YOUR WORLD
SCRATCH AT MIT 2012

Making Music with Scratch

a workshop presented at

Scratch@MIT 2012

Friday, July 27, 2012

Jesse M. Heines
Dept. of Computer Science
Jesse_Heines@uml.edu

Gena R. Greher
Dept. of Music
Gena_Greher@uml.edu

University of Massachusetts Lowell

www.performamatics.org

PERFORMAMATICS

SCRATCH

Copyright Notice

A full color version of this handout in PDF format may be downloaded from:

http://bit.ly/scratch2012music

Contents

continued on next page

Contents (cont'd)

Workshop Description

> “There is nothing like making music and messing with sound to inspire people to learn how to program.”
>
> — Prof. Dan Trueman, co-founder of the Princeton Laptop Orchestra

We couldn’t agree more. However, the cost of “getting into the digital music game” can be prohibitive on many fronts. Our work strives to make the excitement of music technology accessible to students of all levels.

This workshop presents a subset of the techniques we’ve developed to allow students to explore the intersection of computing and music using the sound capabilities built into Scratch. In addition, it gives participants the opportunity to create music-generating programs themselves using a variety of Scratch constructs. The workshop will conclude with a mini concert in which some participants will play the music they created using these techniques.

One of the distinctive characteristics of the workshop is that it is presented in true interdisciplinary fashion: by a CS and a Music professor working together. This is a hallmark of our work. Participants will be exposed to a

Workshop Description (cont'd)

fresh approach to teaching that they may be able to implement to a greater or lesser degree in their own schools and institutions.

Background

Music applications are incredibly powerful and engaging tools for getting students interested in learning about technology. They can range from music cataloging applications such as the popular musicbrainz.org site, to music playing applications such as the ubiquitous Apple and Adobe products, to music transcribing and composing applications such as Finale, Sibelius, and Noteflight.

We are interested in harnessing the engaging power of music to stimulate student interest in computing in general and computational thinking in particular. Toward that end we have designed an interdisciplinary, general education course that teaches computing *and* music to undergraduates in novel ways and that is open to all students in all majors. Our project is called "Performamatics," and it has been funded by two grants from the National Science

Workshop Description (cont'd)

Foundation. (Please see www.performamatics.org for further information on this project.)

The centerpiece of Performamatics is "Sound Thinking," an interdisciplinary course taught with a music and a computer science professor in the room for all class meetings. (Please see soundthinking.uml.edu for the course website.) This approach models the interdisciplinary environments that students will encounter when they graduate and provides valuable lessons for life in the world of work.

The computational thinking component of our approach not only helps arts majors learn to think analytically, but also helps technical majors understand computing concepts at a deeper level through applications that employ the engaging power of music. We take advantage of the "low floor and high ceiling" of Scratch to appeal to students at both ends of the curricular spectrum. We believe that this is one of the real powers of this media-rich visual programming system. We have seen that harnessing its music capabilities is a tremendous "hook" for making programming appealing to a wide range of students, from those in middle school to those in undergraduate programs.

Workshop Topics

1. Demonstration of Scratch music capabilities

2. Playing MP3 files from Scratch
    - Synching music to animations
    - Manipulation of MP3 files using Audacity

3. Playing MIDI notes from Scratch
    - Creating and playing simple melodies
    - Using loops and broadcasts to structure music

4. Playing MIDI notes using lists
    - Creating and populating lists
    - Working with rhythm and note lists

5. Synchronizing multiple parts
    - Techniques that do not work, and those that do

6. Introduction to external sensor devices
    - The Scratch Board and PicoBoard
    - The IchiBoard

7. Sharing what you've created
    - Live performances by participants ☺

Additional Workshop Info and URLs

Participants should download and install:

- **Scratch 1.4**
  scratch.mit.edu/download
- **Audacity 2.0.1**
  audacity.sourceforge.net/download

Participants should also download and install the appropriate sensor board drivers for their systems.

- for PicoBoards
  www.picocricket.com/whichpicoboard.html
- for IchiBoards
  www.cs.uml.edu/ecg/index.php/IchiBoard

Important Note: Scratch does not have access to a MIDI synthesizer on systems running Linux, Ubuntu, etc. Scratch does synthesize notes on these systems, but you only get one instrument.

About the Workshop Leaders

Jesse Heines is a Professor of Computer Science at UMass Lowell. He has a keen interest in CS education and computer applications in the arts, particularly those in music. He teaches courses in object-oriented and graphical user interface programming. Jesse grew up in a musical household and currently enjoys singing in a barbershop chorus.

Gena Greher spent 20 years as a music producer/director in advertising before moving to UMass Lowell. She has published on the influence of multimedia technology in the general music classroom, middle school music curricula, and music teacher education curricula. Gena teaches courses in music methods, world music for the classroom, technology in music ed., and socio-cultural impacts on teaching music.

Jesse's and Gena's work has been supported by two **National Science Foundation** awards: "Performamatics: Connecting Computer Science to the Performing, Fine, and Design Arts" and "Computational Thinking through Computing and Music." Please see www.performamatics.org for more information on these projects and the opportunity to attend one of our two-day, NSF-sponsored interdisciplinary workshops.

Important Note on Turbo Speed

The timing of virtually all music scripts can be improved by setting Turbo Speed. To do this, select:

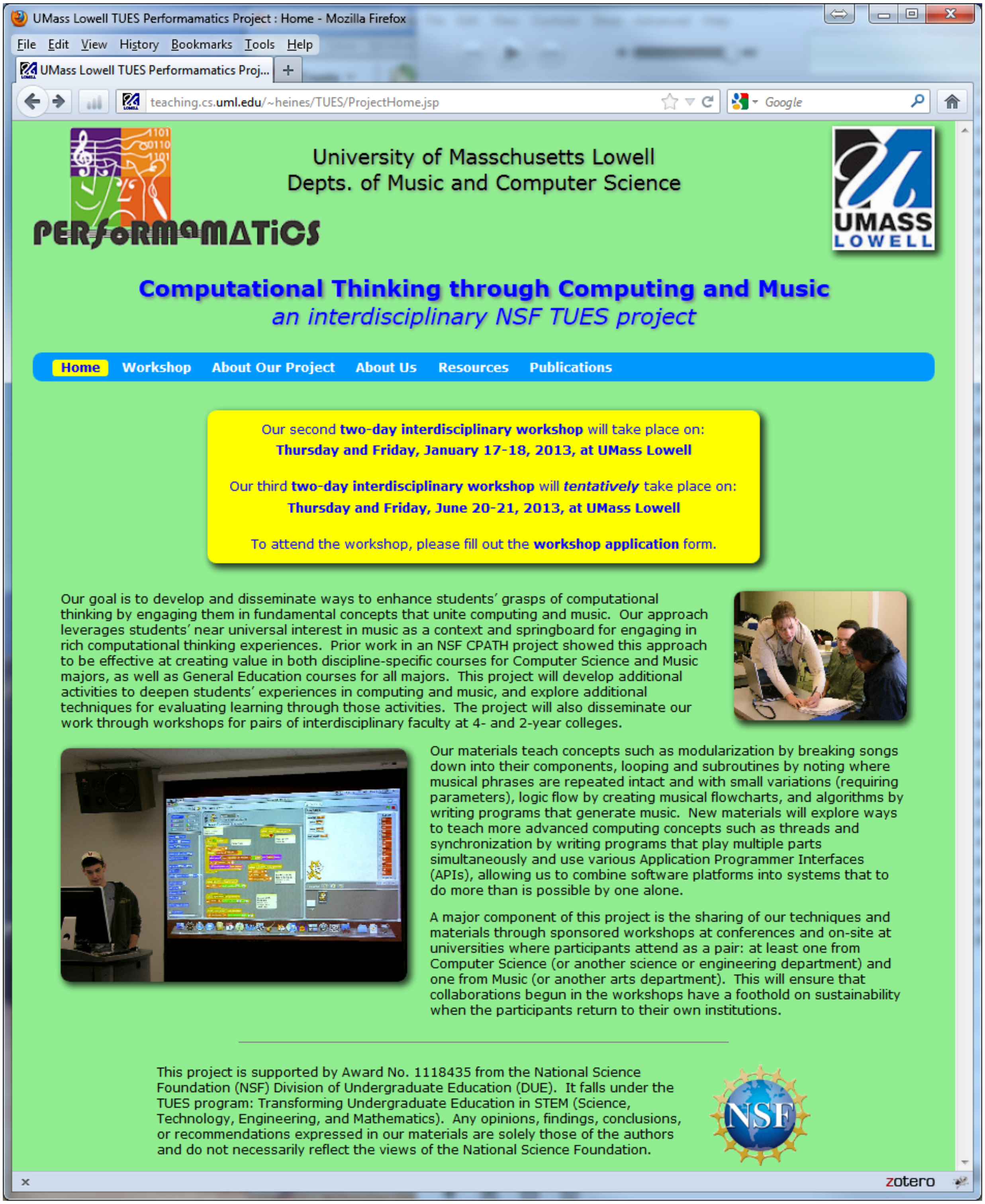
Edit ➔ Set Single Stepping... ➔ Turbo Speed

Acknowledgements

Additional contributors to this work include UMass Lowell Profs. Gena Greher, S. Alex Ruthmann, and Fred Martin, graduate student Mark Sherman, recent undergraduates Paul Laidler and Charles Saulters, and current undergraduates Brendan Reilly, Angelo Gamarra, Matt Vaughan, and Nathan Goss.

The materials presented in this workshop are based on work supported by the National Science Foundation under Award Nos. 0722161 and 1118435. Any opinions, findings, and conclusions or recommendations expressed or implied in these materials or the workshop discussion are those of the authors alone and do not necessarily reflect the views of the National Science Foundation.

Performamatics Project Website


UMass Lowell TUES Performamatics Project : Home - Mozilla Firefox
File Edit View History Bookmarks Tools Help
UMass Lowell TUES Performamatics Proj...
teaching.cs.uml.edu/~heines/TUES/ProjectHome.jsp
Google
PERFoRMaMATiCS
UMASS LOWELL
Home
Workshop
About Our Project
About Us
Resources
Publications
NSF
zotero

University of Masschusetts Lowell
Depts. of Music and Computer Science

Computational Thinking through Computing and Music
an interdisciplinary NSF TUES project

Our second **two-day interdisciplinary workshop** will take place on:
Thursday and Friday, January 17-18, 2013, at UMass Lowell

Our third **two-day interdisciplinary workshop** will ***tentatively*** take place on:
Thursday and Friday, June 20-21, 2013, at UMass Lowell

To attend the workshop, please fill out the **workshop application** form.

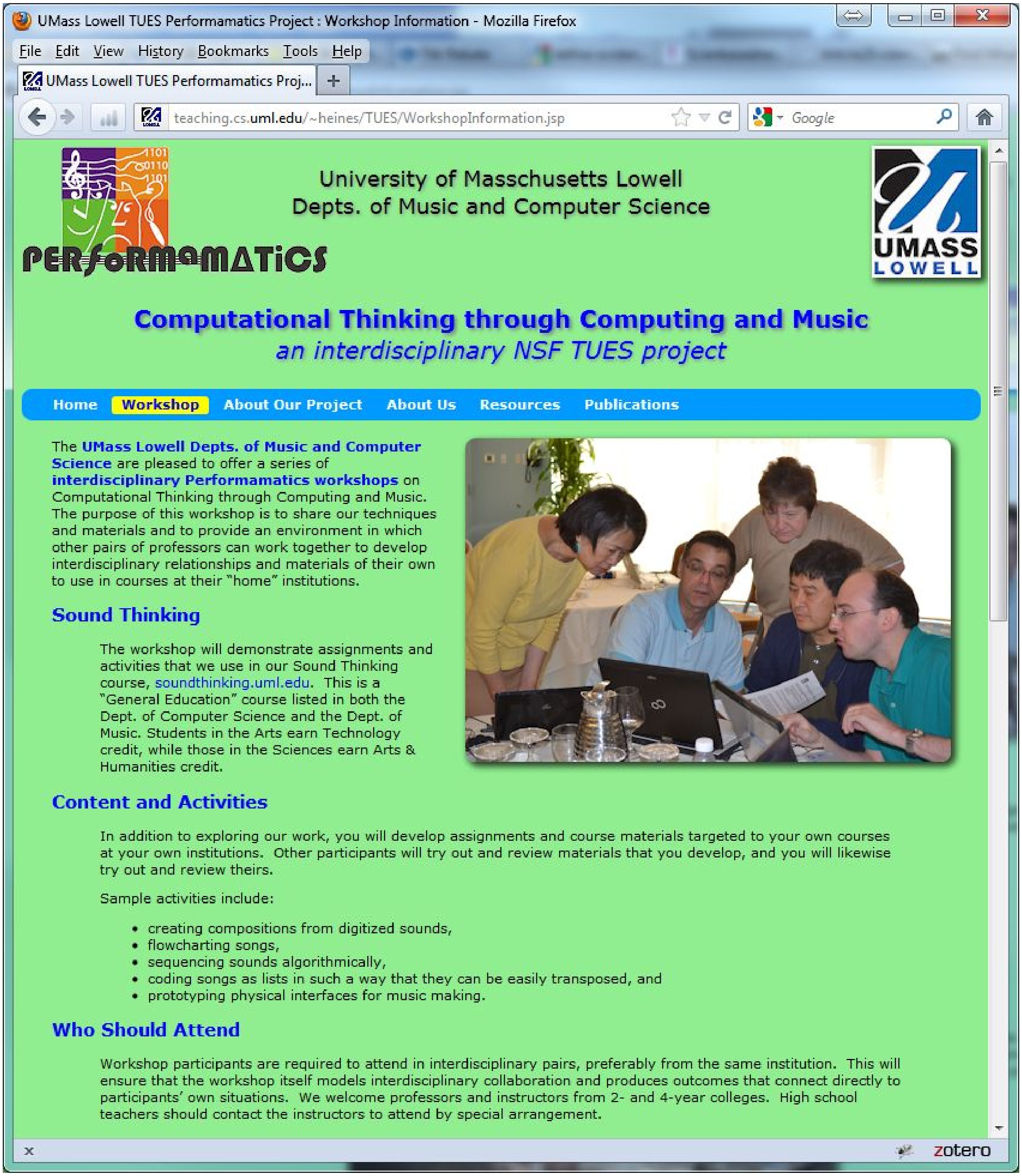
Our goal is to develop and disseminate ways to enhance students' grasps of computational thinking by engaging them in fundamental concepts that unite computing and music. Our approach leverages students' near universal interest in music as a context and springboard for engaging in rich computational thinking experiences. Prior work in an NSF CPATH project showed this approach to be effective at creating value in both discipline-specific courses for Computer Science and Music majors, as well as General Education courses for all majors. This project will develop additional activities to deepen students' experiences in computing and music, and explore additional techniques for evaluating learning through those activities. The project will also disseminate our work through workshops for pairs of interdisciplinary faculty at 4- and 2-year colleges.

Our materials teach concepts such as modularization by breaking songs down into their components, looping and subroutines by noting where musical phrases are repeated intact and with small variations (requiring parameters), logic flow by creating musical flowcharts, and algorithms by writing programs that generate music. New materials will explore ways to teach more advanced computing concepts such as threads and synchronization by writing programs that play multiple parts simultaneously and use various Application Programmer Interfaces (APIs), allowing us to combine software platforms into systems that to do more than is possible by one alone.

A major component of this project is the sharing of our techniques and materials through sponsored workshops at conferences and on-site at universities where participants attend as a pair: at least one from Computer Science (or another science or engineering department) and one from Music (or another arts department). This will ensure that collaborations begun in the workshops have a foothold on sustainability when the participants return to their own institutions.

This project is supported by Award No. 1118435 from the National Science Foundation (NSF) Division of Undergraduate Education (DUE). It falls under the TUES program: Transforming Undergraduate Education in STEM (Science, Technology, Engineering, and Mathematics). Any opinions, findings, conclusions, or recommendations expressed in our materials are solely those of the authors and do not necessarily reflect the views of the National Science Foundation.

Performamatics Workshop Info


UMass Lowell TUES Performamatics Project : Workshop Information - Mozilla Firefox
File Edit View History Bookmarks Tools Help
UMass Lowell TUES Performamatics Proj...
teaching.cs.uml.edu/~heines/TUES/WorkshopInformation.jsp
Google
PERformamATiCS
UMASS LOWELL
zotero

University of Masschusetts Lowell
Depts. of Music and Computer Science

Computational Thinking through Computing and Music
an interdisciplinary NSF TUES project

Home | **Workshop** | About Our Project | About Us | Resources | Publications

The **UMass Lowell Depts. of Music and Computer Science** are pleased to offer a series of **interdisciplinary Performamatics workshops** on Computational Thinking through Computing and Music. The purpose of this workshop is to share our techniques and materials and to provide an environment in which other pairs of professors can work together to develop interdisciplinary relationships and materials of their own to use in courses at their "home" institutions.

Sound Thinking

The workshop will demonstrate assignments and activities that we use in our Sound Thinking course, soundthinking.uml.edu. This is a "General Education" course listed in both the Dept. of Computer Science and the Dept. of Music. Students in the Arts earn Technology credit, while those in the Sciences earn Arts & Humanities credit.

Content and Activities

In addition to exploring our work, you will develop assignments and course materials targeted to your own courses at your own institutions. Other participants will try out and review materials that you develop, and you will likewise try out and review theirs.

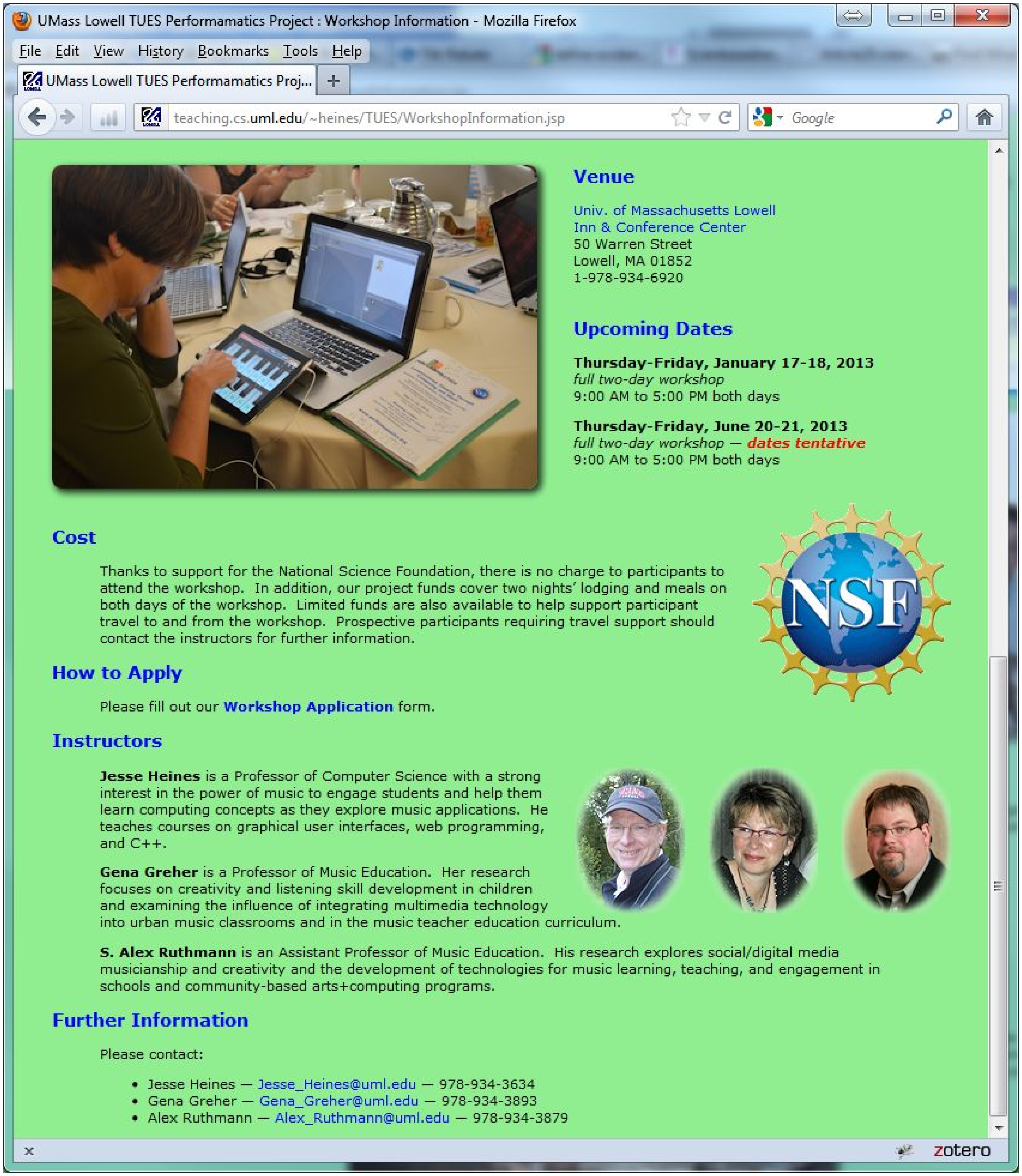
Sample activities include:

- creating compositions from digitized sounds,
- flowcharting songs,
- sequencing sounds algorithmically,
- coding songs as lists in such a way that they can be easily transposed, and
- prototyping physical interfaces for music making.

Who Should Attend

Workshop participants are required to attend in interdisciplinary pairs, preferably from the same institution. This will ensure that the workshop itself models interdisciplinary collaboration and produces outcomes that connect directly to participants' own situations. We welcome professors and instructors from 2- and 4-year colleges. High school teachers should contact the instructors to attend by special arrangement.

Performamatics Workshop Info (cont'd)


UMass Lowell TUES Performamatics Project : Workshop Information - Mozilla Firefox
File Edit View History Bookmarks Tools Help
UMass Lowell TUES Performamatics Proj...
Google
zotero

teaching.cs.uml.edu/~heines/TUES/WorkshopInformation.jsp

Venue

Univ. of Massachusetts Lowell
Inn & Conference Center
50 Warren Street
Lowell, MA 01852
1-978-934-6920

Upcoming Dates

Thursday-Friday, January 17-18, 2013
full two-day workshop
9:00 AM to 5:00 PM both days

Thursday-Friday, June 20-21, 2013
full two-day workshop — ***dates tentative***
9:00 AM to 5:00 PM both days

Cost

Thanks to support for the National Science Foundation, there is no charge to participants to attend the workshop. In addition, our project funds cover two nights' lodging and meals on both days of the workshop. Limited funds are also available to help support participant travel to and from the workshop. Prospective participants requiring travel support should contact the instructors for further information.

How to Apply

Please fill out our **Workshop Application** form.

Instructors

Jesse Heines is a Professor of Computer Science with a strong interest in the power of music to engage students and help them learn computing concepts as they explore music applications. He teaches courses on graphical user interfaces, web programming, and C++.

Gena Greher is a Professor of Music Education. Her research focuses on creativity and listening skill development in children and examining the influence of integrating multimedia technology into urban music classrooms and in the music teacher education curriculum.

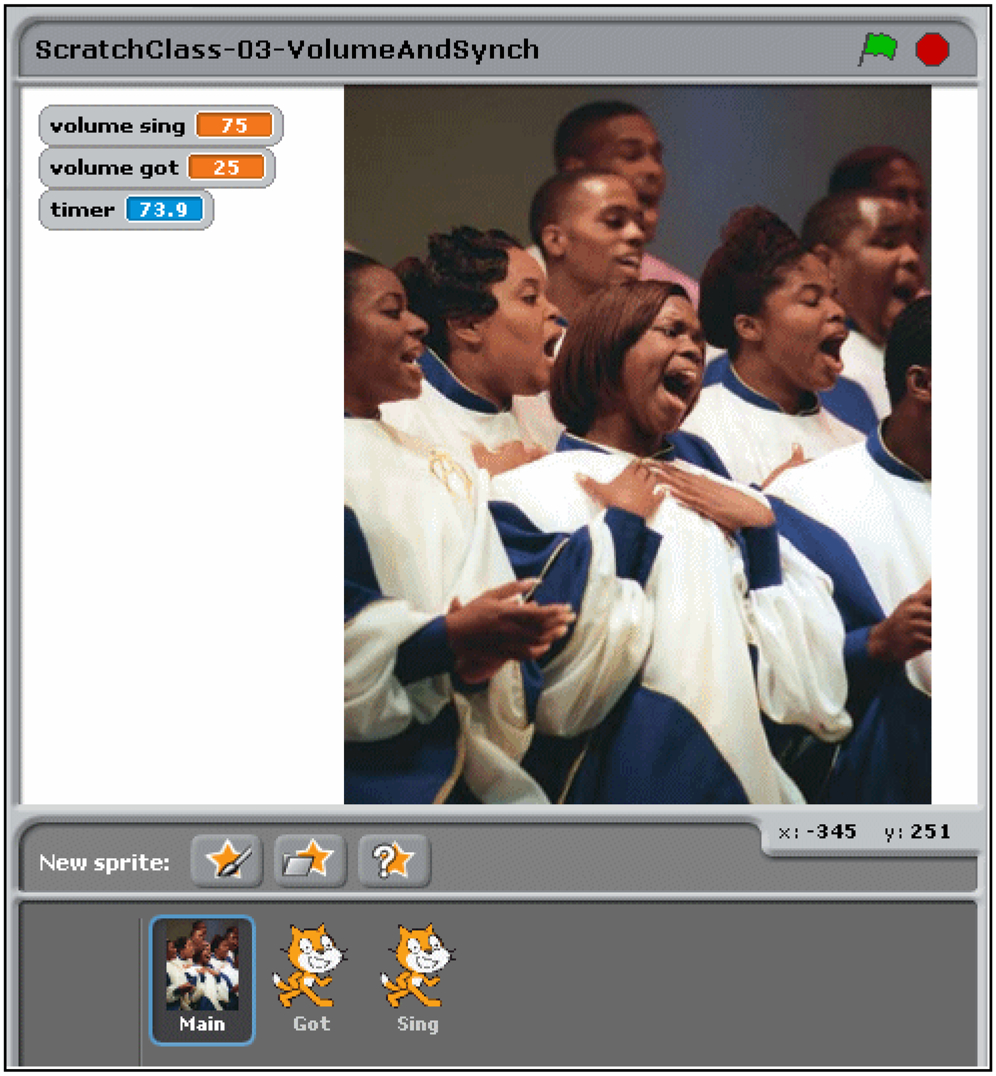
S. Alex Ruthmann is an Assistant Professor of Music Education. His research explores social/digital media musicianship and creativity and the development of technologies for music learning, teaching, and engagement in schools and community-based arts+computing programs.

Further Information

Please contact:

- Jesse Heines — Jesse_Heines@uml.edu — 978-934-3634
- Gena Greher — Gena_Greher@uml.edu — 978-934-3893
- Alex Ruthmann — Alex_Ruthmann@uml.edu — 978-934-3879

Playing and Synchronizing MIDI Files



volume sing 75

volume got 25

timer 73.9

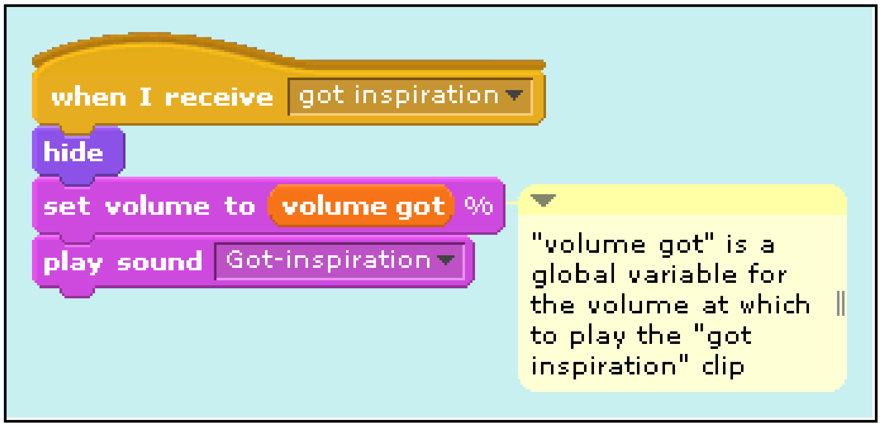
Volume and Synchronization Concepts

- use of variables when setting the volume
- local vs. global attributes, specifically volume
- use of a control script and broadcasts
- use of the Scratch timer for synchronization

Playing and Synching MIDI Files (cont'd)

MP3 Player Scripts

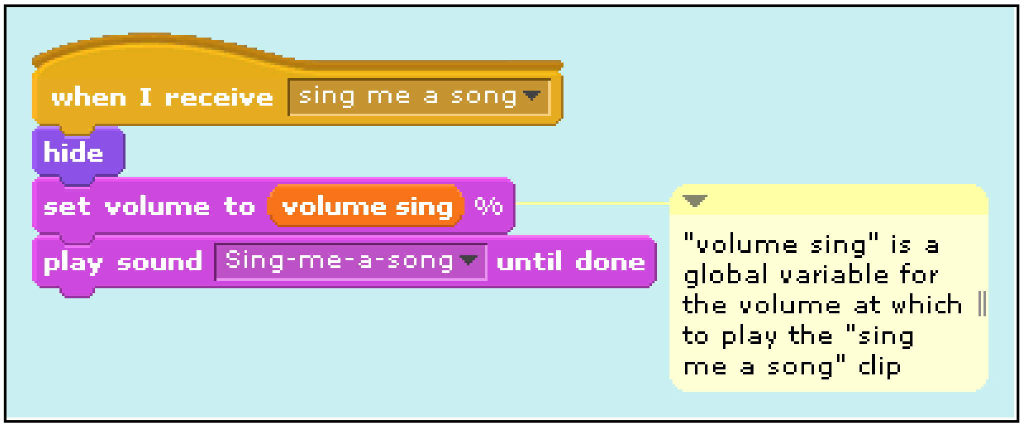
Script in Sprite "Got"



```
when I receive [got inspiration ▾]
hide
set volume to (volume got) %
play sound [Got-inspiration ▾]
```

"volume got" is a global variable for the volume at which to play the "got inspiration" clip

Script in Sprite "Sing"



```
when I receive [sing me a song ▾]
hide
set volume to (volume sing) %
play sound [Sing-me-a-song ▾] until done
```

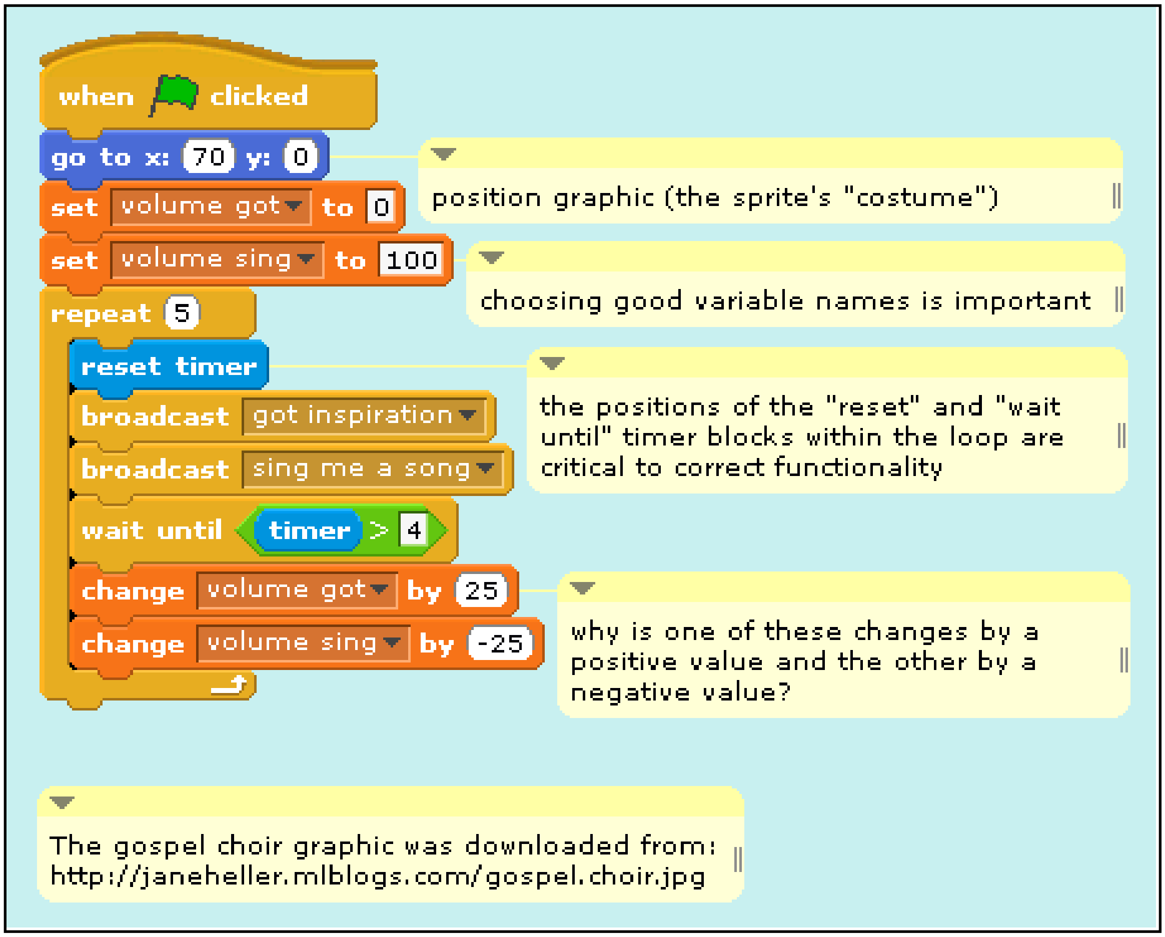
"volume sing" is a global variable for the volume at which to play the "sing me a song" clip

Each script must be in its own sprite to allow volume to be controlled independently.

Playing and Synching MIDI Files (cont'd)

Control Script

Script in Sprite "Main"



```
when ⚑ clicked
go to x: 70 y: 0
set volume got to 0
set volume sing to 100
repeat 5
    reset timer
    broadcast got inspiration
    broadcast sing me a song
    wait until timer > 4
    change volume got by 25
    change volume sing by -25
```

position graphic (the sprite's "costume")

choosing good variable names is important

the positions of the "reset" and "wait until" timer blocks within the loop are critical to correct functionality

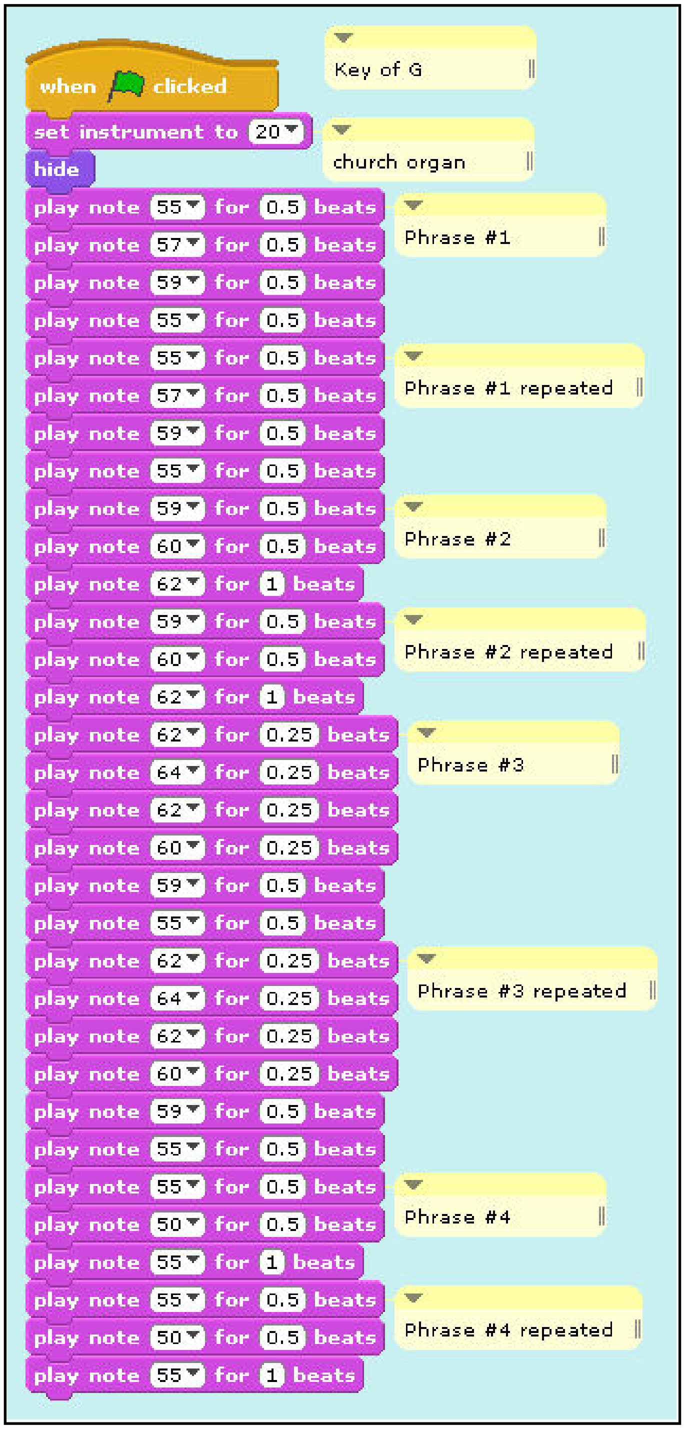
why is one of these changes by a positive value and the other by a negative value?

The gospel choir graphic was downloaded from: http://janeheller.mlblogs.com/gospel.choir.jpg

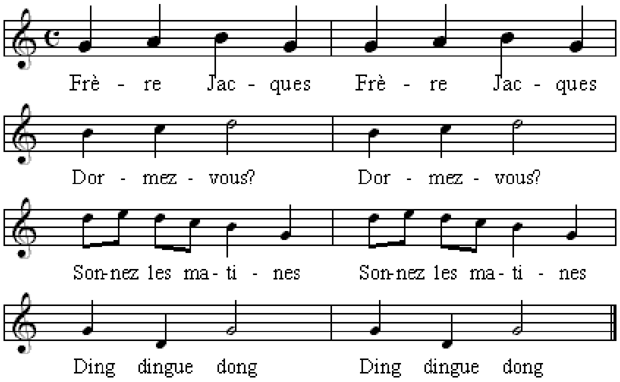
Note the order of the blocks and the critical position of the change blocks. Changing the volume parameter before the wait until block will cause the volume to be changed while the MP3 is playing. Such behavior may be desirable in other programs, but not this one.

Frère Jacques

Version 1: Playing Notes



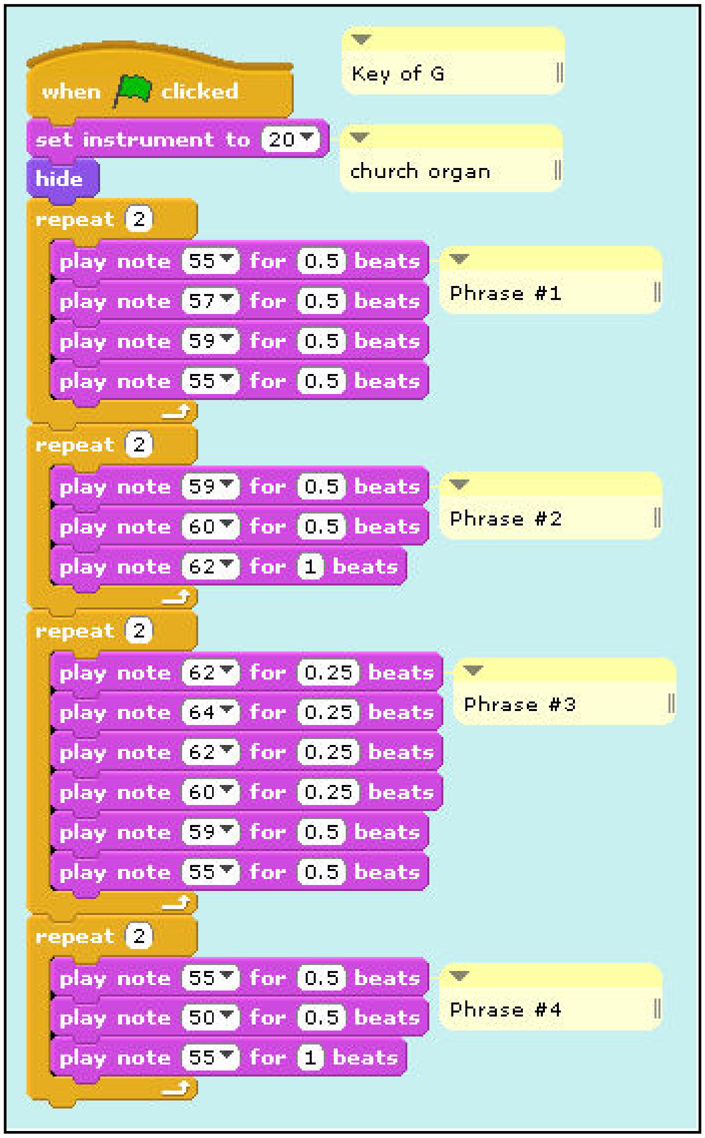
```
when [flag] clicked                  Key of G
set instrument to (20)               church organ
hide
play note (55) for (0.5) beats       Phrase #1
play note (57) for (0.5) beats
play note (59) for (0.5) beats
play note (55) for (0.5) beats
play note (55) for (0.5) beats       Phrase #1 repeated
play note (57) for (0.5) beats
play note (59) for (0.5) beats
play note (55) for (0.5) beats
play note (59) for (0.5) beats       Phrase #2
play note (60) for (0.5) beats
play note (62) for (1) beats
play note (59) for (0.5) beats       Phrase #2 repeated
play note (60) for (0.5) beats
play note (62) for (1) beats
play note (62) for (0.25) beats      Phrase #3
play note (64) for (0.25) beats
play note (62) for (0.25) beats
play note (60) for (0.25) beats
play note (59) for (0.5) beats
play note (55) for (0.5) beats
play note (62) for (0.25) beats      Phrase #3 repeated
play note (64) for (0.25) beats
play note (62) for (0.25) beats
play note (60) for (0.25) beats
play note (59) for (0.5) beats
play note (55) for (0.5) beats
play note (55) for (0.5) beats       Phrase #4
play note (50) for (0.5) beats
play note (55) for (1) beats
play note (55) for (0.5) beats       Phrase #4 repeated
play note (50) for (0.5) beats
play note (55) for (1) beats
```


Frè - re Jac - ques Frè - re Jac - ques
Dor - mez - vous? Dor - mez - vous?
Son-nez les ma - ti - nes Son-nez les ma - ti - nes
Ding dingue dong Ding dingue dong

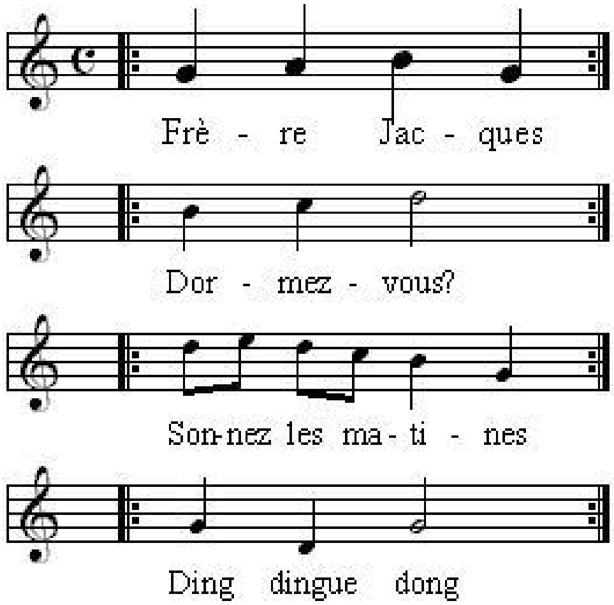
Remember Turbo Speed!

Frère Jacques

Version 2: Using Loops



```
when [flag] clicked                              Key of G
set instrument to 20                             church organ
hide
repeat 2
    play note 55 for 0.5 beats                   Phrase #1
    play note 57 for 0.5 beats
    play note 59 for 0.5 beats
    play note 55 for 0.5 beats
repeat 2
    play note 59 for 0.5 beats                   Phrase #2
    play note 60 for 0.5 beats
    play note 62 for 1 beats
repeat 2
    play note 62 for 0.25 beats                  Phrase #3
    play note 64 for 0.25 beats
    play note 62 for 0.25 beats
    play note 60 for 0.25 beats
    play note 59 for 0.5 beats
    play note 55 for 0.5 beats
repeat 2
    play note 55 for 0.5 beats                   Phrase #4
    play note 50 for 0.5 beats
    play note 55 for 1 beats
```



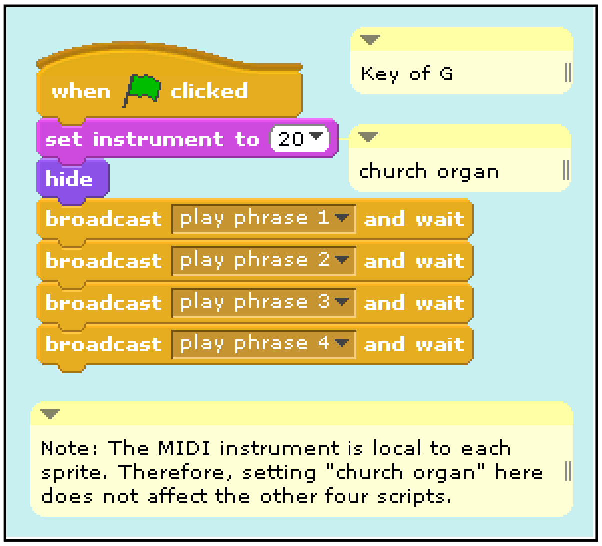
Remember to set Turbo Speed to improve performance.

Acknowledgement: The scores on this and the previous page were adapted from `www.csdraveurs.qc.ca/musique/flutalors/images/frere.gif` and `www.mamalisa.com/images/scores/frerejacques.jpg`, respectively, but as of July 12, 2012, these URLs no longer appear to be valid.

Frère Jacques

Version 3: Separating Phrases

Main Script

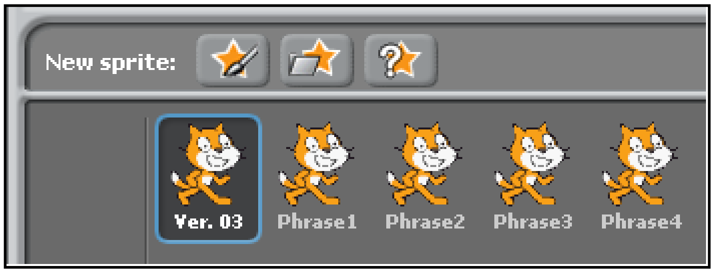


```
when [flag] clicked
set instrument to 20        church organ
hide
broadcast play phrase 1 and wait
broadcast play phrase 2 and wait
broadcast play phrase 3 and wait
broadcast play phrase 4 and wait
```

Key of G

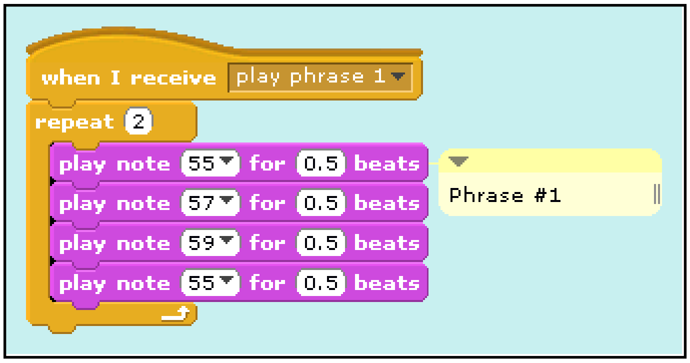
church organ

Note: The MIDI instrument is local to each sprite. Therefore, setting "church organ" here does not affect the other four scripts.



Phrases Scripts (4, cont'd on next page)

#1



```
when I receive play phrase 1
repeat 2
    play note 55 for 0.5 beats
    play note 57 for 0.5 beats
    play note 59 for 0.5 beats
    play note 55 for 0.5 beats
```

Phrase #1

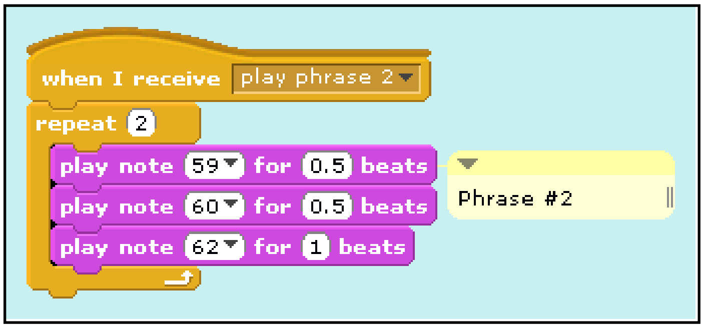
Thought: We could set the instrument in each script, but that would contradict the **DRY** programming principle: "Don't Repeat Yourself."

Frère Jacques

Version 3: Separating Phrases (cont'd)

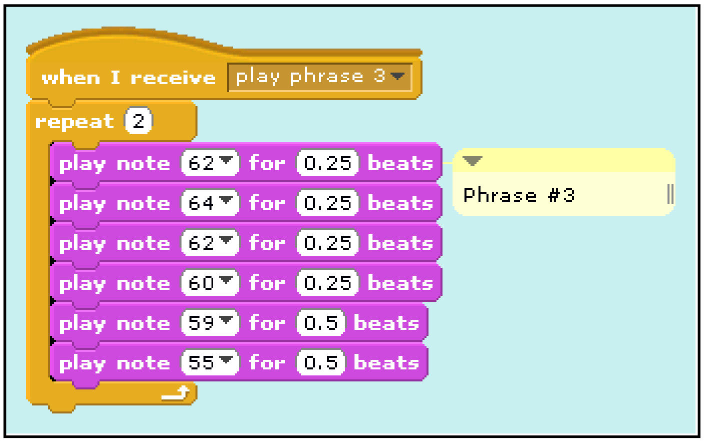
Phrases Scripts (cont'd)

#2



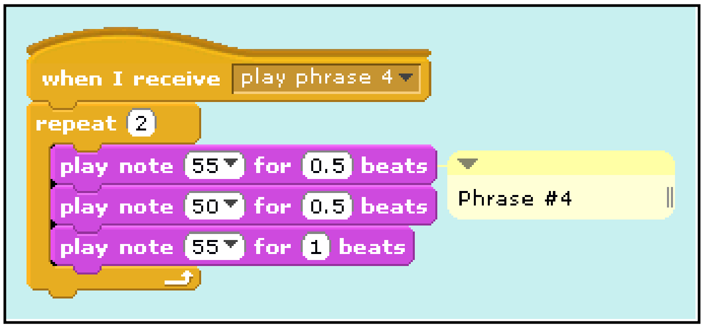
```
when I receive [play phrase 2 ▾]
repeat (2)
    play note (59 ▾) for (0.5) beats    Phrase #2
    play note (60 ▾) for (0.5) beats
    play note (62 ▾) for (1) beats
```

#3



```
when I receive [play phrase 3 ▾]
repeat (2)
    play note (62 ▾) for (0.25) beats    Phrase #3
    play note (64 ▾) for (0.25) beats
    play note (62 ▾) for (0.25) beats
    play note (60 ▾) for (0.25) beats
    play note (59 ▾) for (0.5) beats
    play note (55 ▾) for (0.5) beats
```

#4



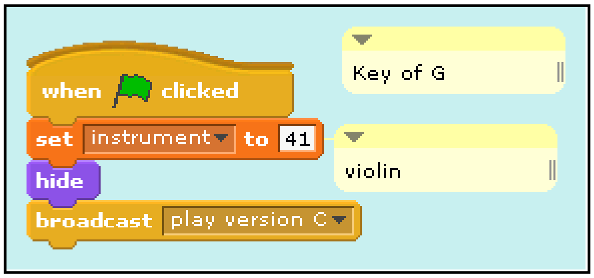
```
when I receive [play phrase 4 ▾]
repeat (2)
    play note (55 ▾) for (0.5) beats    Phrase #4
    play note (50 ▾) for (0.5) beats
    play note (55 ▾) for (1) beats
```

Challenge: How can we set the instrument **JUST ONCE** and have that setting apply to all scripts?

Frère Jacques

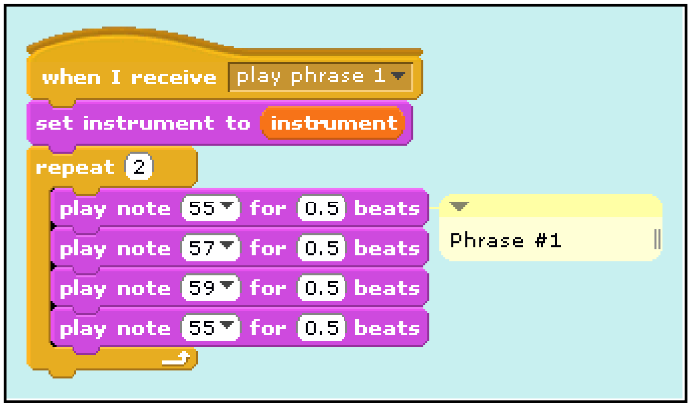
Version 4: Playing a Round

Main Script



```
when ⚑ clicked                 Key of G
set instrument ▾ to 41         violin
hide
broadcast play version C ▾
```

Phrases Scripts



```
when I receive play phrase 1 ▾
set instrument to instrument
repeat 2
    play note 55 ▾ for 0.5 beats    Phrase #1
    play note 57 ▾ for 0.5 beats
    play note 59 ▾ for 0.5 beats
    play note 55 ▾ for 0.5 beats
```

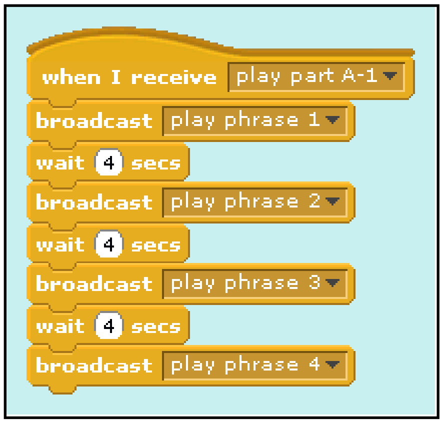
Note the addition of the set instrument block and the use of the instrument variable (set in the Main script) as the value to set. Other phrase scripts similarly contain this one revision.

continued on next page

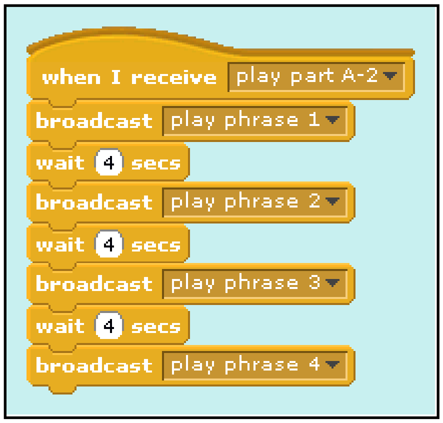
Frère Jacques

Version 4: Playing a Round (cont'd)

Scripts A-1 through A-4



```
when I receive play part A-1
broadcast play phrase 1
wait 4 secs
broadcast play phrase 2
wait 4 secs
broadcast play phrase 3
wait 4 secs
broadcast play phrase 4
```

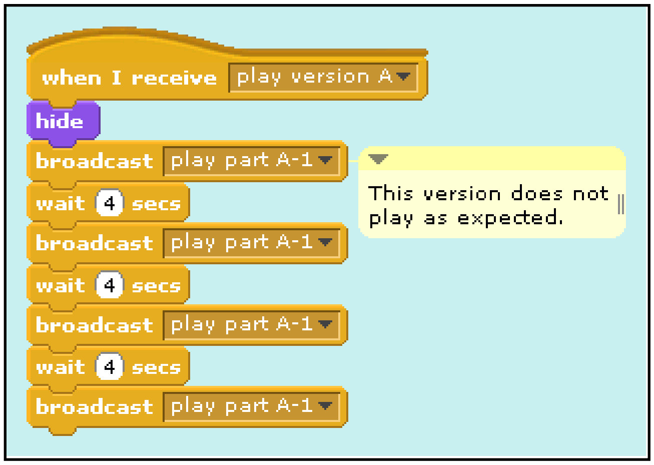


```
when I receive play part A-2
broadcast play phrase 1
wait 4 secs
broadcast play phrase 2
wait 4 secs
broadcast play phrase 3
wait 4 secs
broadcast play phrase 4
```

...

Others scripts are similar, differing only in **when I receive**.

Control Script A – single threaded



```
when I receive play version A
hide
broadcast play part A-1
wait 4 secs
broadcast play part A-1
wait 4 secs
broadcast play part A-1
wait 4 secs
broadcast play part A-1
```

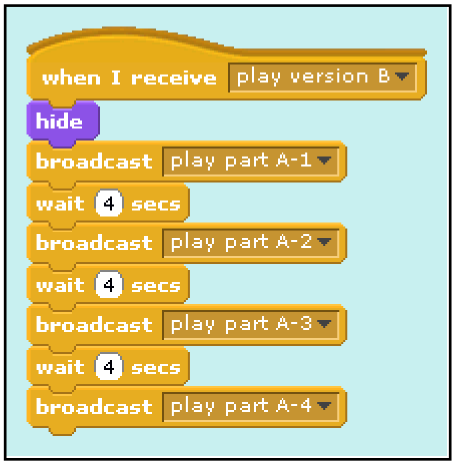
This version does not play as expected.

continued on next page

Frère Jacques

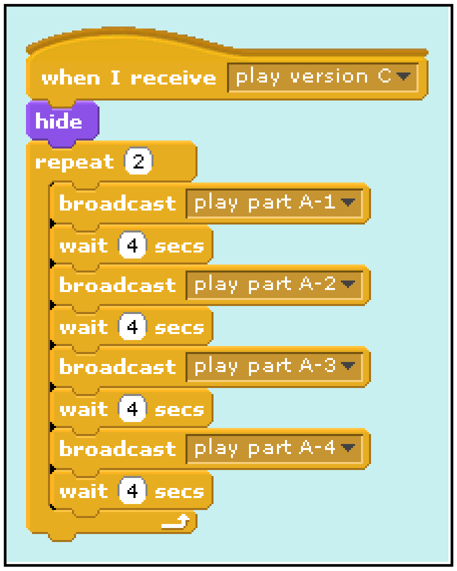
Version 4: Playing a Round (cont'd)

Control Script B – multi-threaded



```
when I receive [play version B ▾]
hide
broadcast [play part A-1 ▾]
wait (4) secs
broadcast [play part A-2 ▾]
wait (4) secs
broadcast [play part A-3 ▾]
wait (4) secs
broadcast [play part A-4 ▾]
```

Control Script C – multi-threaded repeat



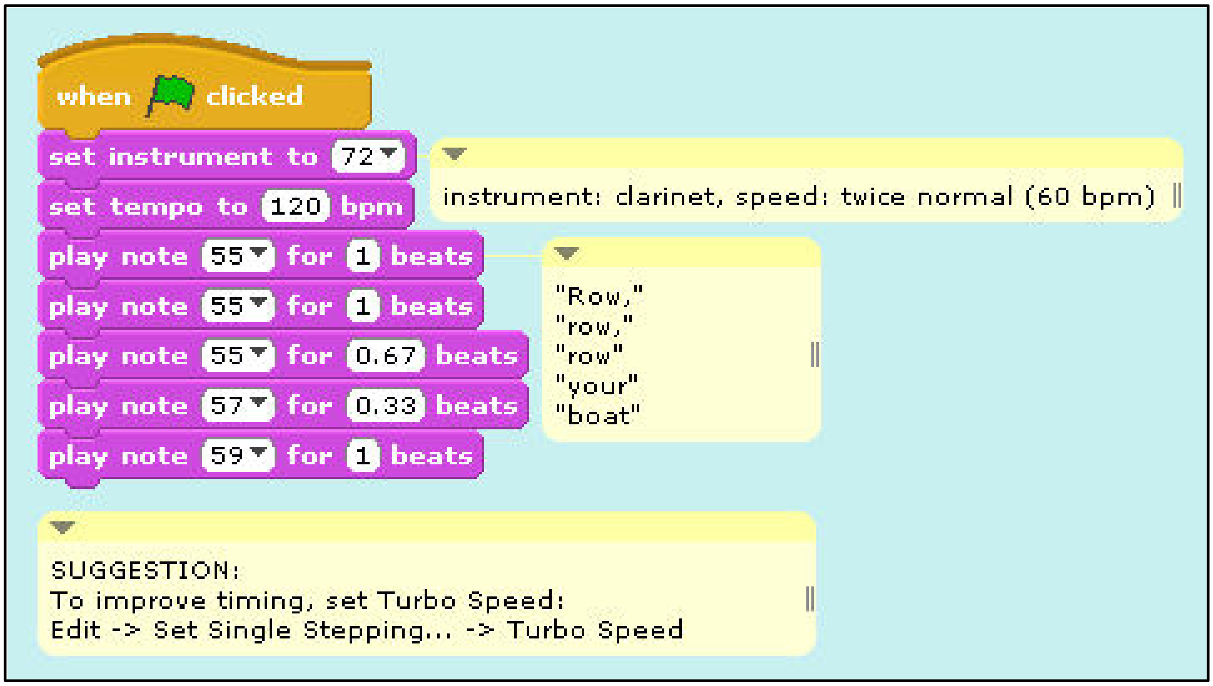
```
when I receive [play version C ▾]
hide
repeat (2)
    broadcast [play part A-1 ▾]
    wait (4) secs
    broadcast [play part A-2 ▾]
    wait (4) secs
    broadcast [play part A-3 ▾]
    wait (4) secs
    broadcast [play part A-4 ▾]
    wait (4) secs
```

end of Version 4

Row, Row, Row Your Boat

Version 1: Playing Notes

Single Script



```
when [flag] clicked
set instrument to 72
set tempo to 120 bpm
play note 55 for 1 beats
play note 55 for 1 beats
play note 55 for 0.67 beats
play note 57 for 0.33 beats
play note 59 for 1 beats
```

instrument: clarinet, speed: twice normal (60 bpm)

"Row,"
"row,"
"row"
"your"
"boat"

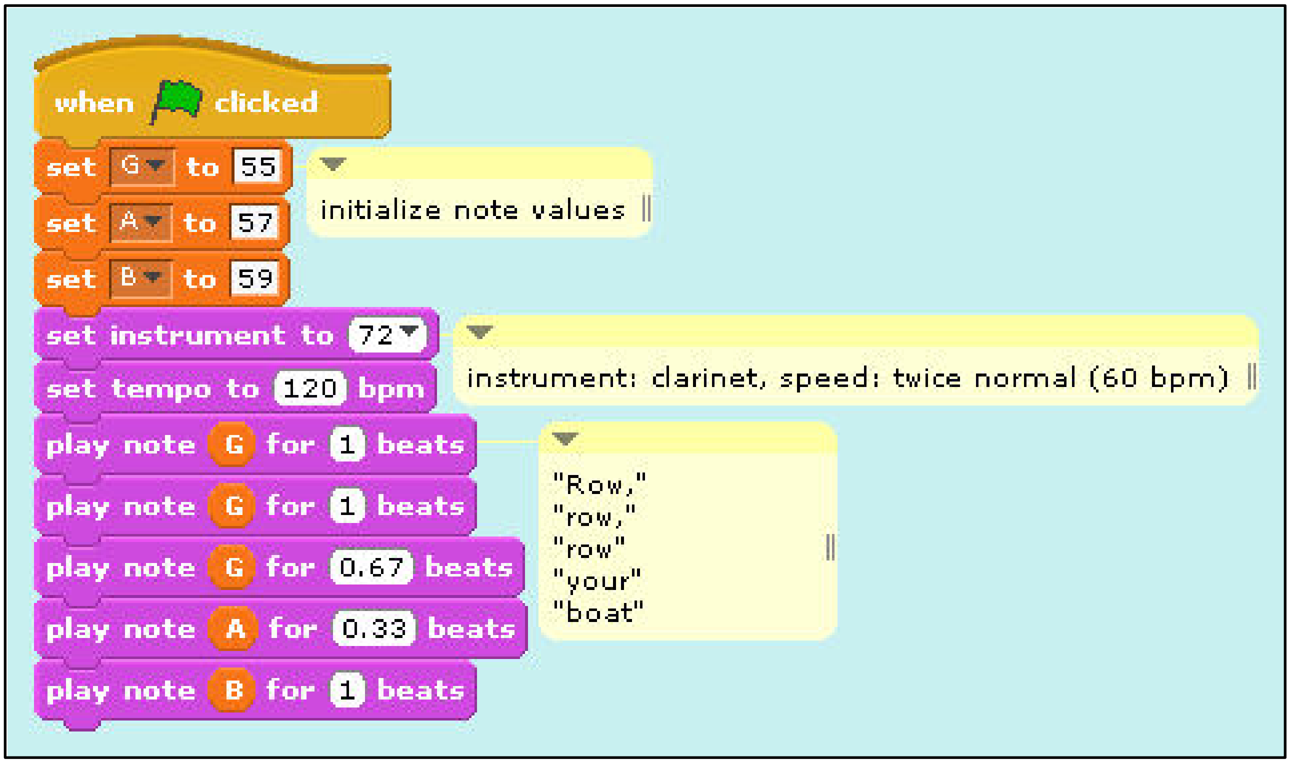
SUGGESTION:
To improve timing, set Turbo Speed:
Edit -> Set Single Stepping... -> Turbo Speed

Output Window

Row, Row, Row Your Boat

Version 2: Playing Notes Using Variables

Single Script



```
when ⚑ clicked
set G to 55
set A to 57
set B to 59
set instrument to 72
set tempo to 120 bpm
play note G for 1 beats
play note G for 1 beats
play note G for 0.67 beats
play note A for 0.33 beats
play note B for 1 beats
```

initialize note values

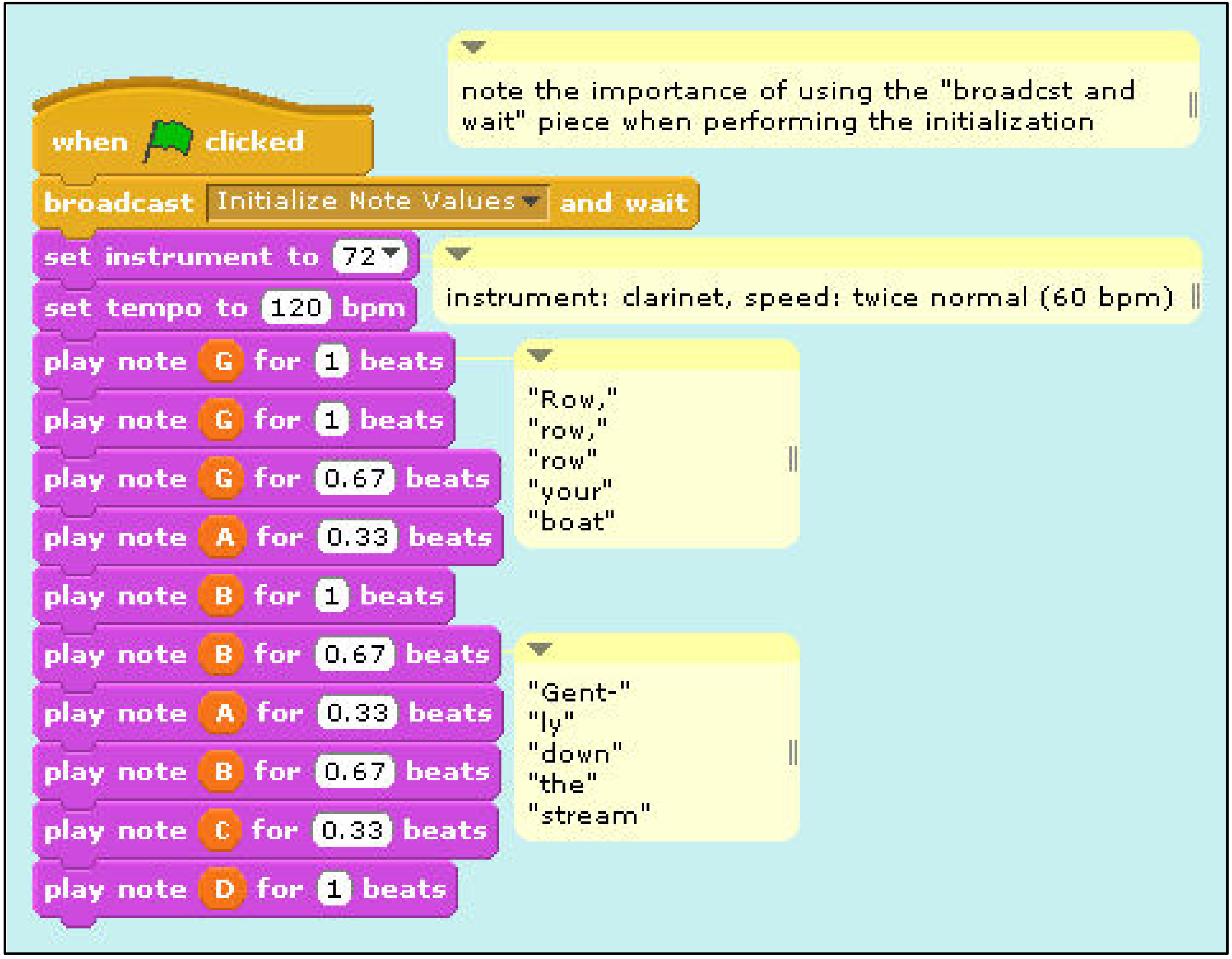
instrument: clarinet, speed: twice normal (60 bpm)

"Row,"
"row,"
"row"
"your"
"boat"

Version 3: Separating Initialization

Two Scripts

(3a) **Main** Script



```
when [flag] clicked
broadcast [Initialize Note Values] and wait
set instrument to 72
set tempo to 120 bpm
play note G for 1 beats
play note G for 1 beats
play note G for 0.67 beats
play note A for 0.33 beats
play note B for 1 beats
play note B for 0.67 beats
play note A for 0.33 beats
play note B for 0.67 beats
play note C for 0.33 beats
play note D for 1 beats
```

note the importance of using the "broadcst and wait" piece when performing the initialization

instrument: clarinet, speed: twice normal (60 bpm)

"Row,"
"row,"
"row"
"your"
"boat"

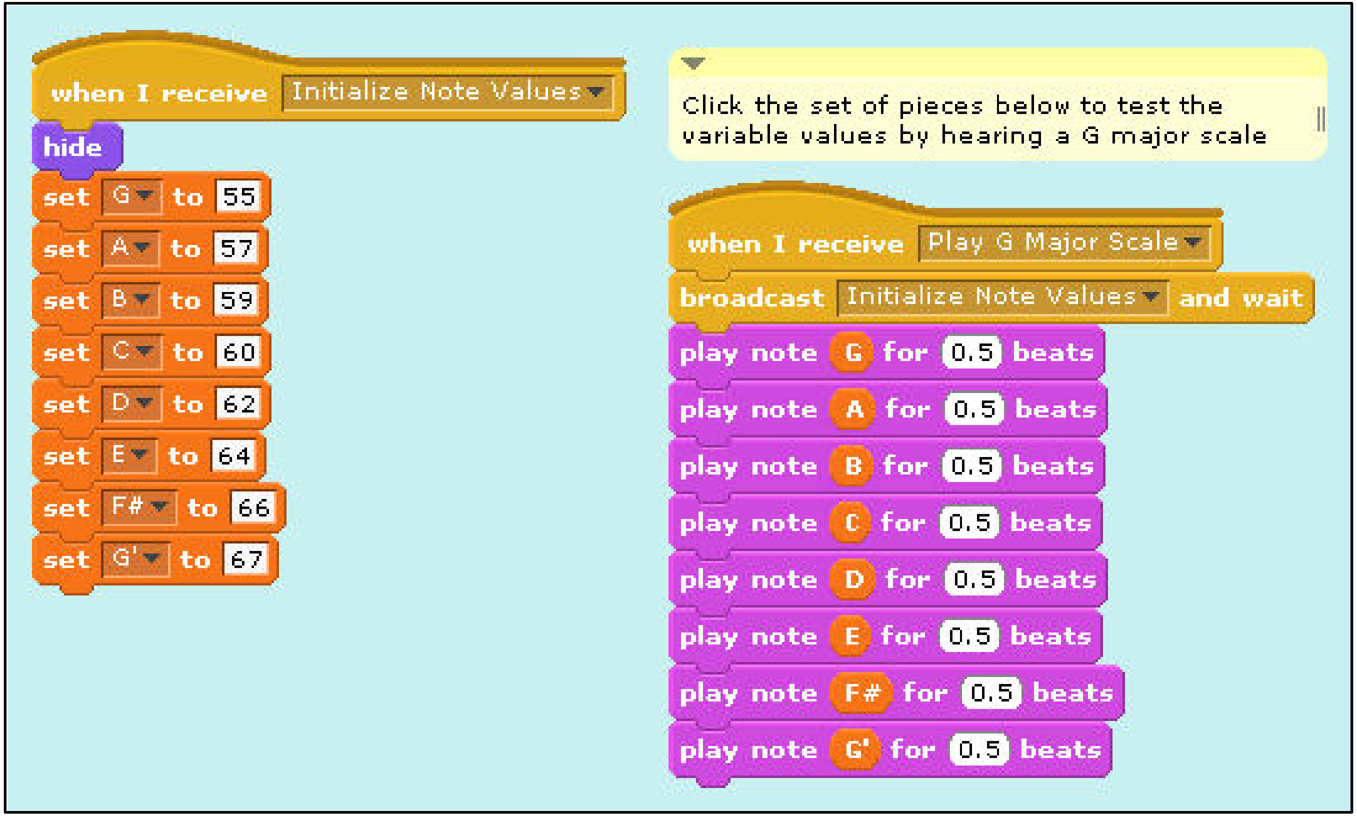
"Gent-"
"ly"
"down"
"the"
"stream"

continued on next page

Row, Row, Row Your Boat

Version 3: Separating Initialization (cont'd)

(3b) **Initialization ("Init")** Script



```
when I receive [Initialize Note Values ▾]
hide
set [G ▾] to 55
set [A ▾] to 57
set [B ▾] to 59
set [C ▾] to 60
set [D ▾] to 62
set [E ▾] to 64
set [F# ▾] to 66
set [G' ▾] to 67
```

Click the set of pieces below to test the variable values by hearing a G major scale

```
when I receive [Play G Major Scale ▾]
broadcast [Initialize Note Values ▾] and wait
play note (G) for 0.5 beats
play note (A) for 0.5 beats
play note (B) for 0.5 beats
play note (C) for 0.5 beats
play note (D) for 0.5 beats
play note (E) for 0.5 beats
play note (F#) for 0.5 beats
play note (G') for 0.5 beats
```

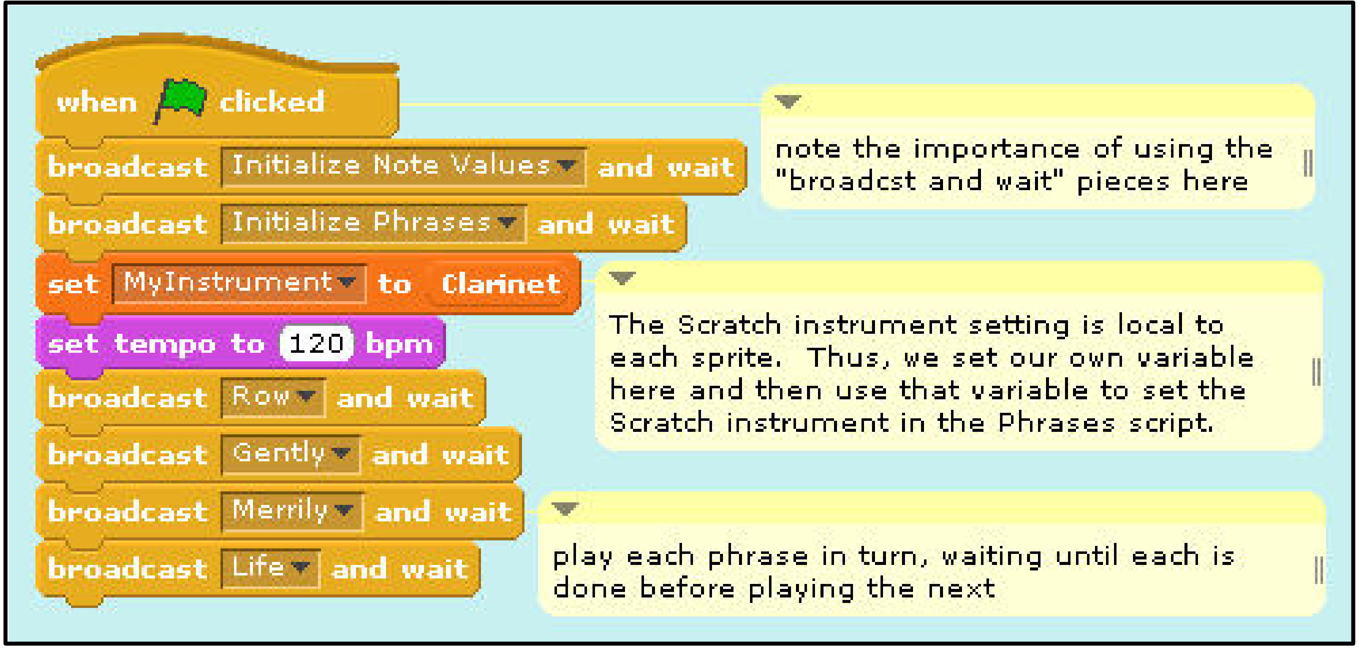
end of Version 3

Row, Row, Row Your Boat

Version 4: Separating Phrases

Three Scripts

(4a) **Main** Script



```
when ⚑ clicked
broadcast [Initialize Note Values ▾] and wait
broadcast [Initialize Phrases ▾] and wait
set [MyInstrument ▾] to Clarinet
set tempo to (120) bpm
broadcast [Row ▾] and wait
broadcast [Gently ▾] and wait
broadcast [Merrily ▾] and wait
broadcast [Life ▾] and wait
```

note the importance of using the "broadcst and wait" pieces here

The Scratch instrument setting is local to each sprite. Thus, we set our own variable here and then use that variable to set the Scratch instrument in the Phrases script.

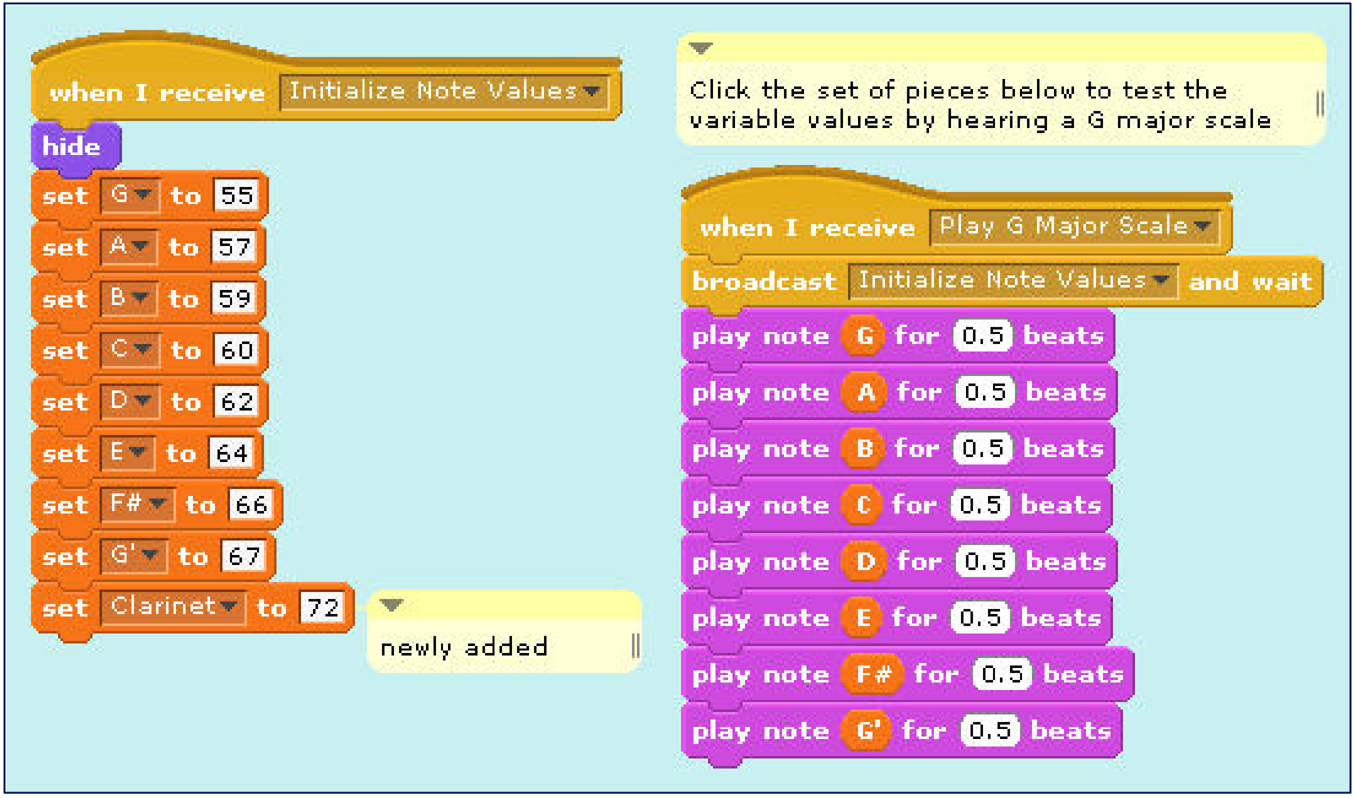
play each phrase in turn, waiting until each is done before playing the next

continued on next page

Row, Row, Row Your Boat

Version 4: Separating Phrases (cont'd)

(4b) **Initialization ("Init")** Script



```
when I receive [Initialize Note Values ▾]
hide
set [G ▾] to (55)
set [A ▾] to (57)
set [B ▾] to (59)
set [C ▾] to (60)
set [D ▾] to (62)
set [E ▾] to (64)
set [F# ▾] to (66)
set [G' ▾] to (67)
set [Clarinet ▾] to (72)
```

newly added

Click the set of pieces below to test the variable values by hearing a G major scale

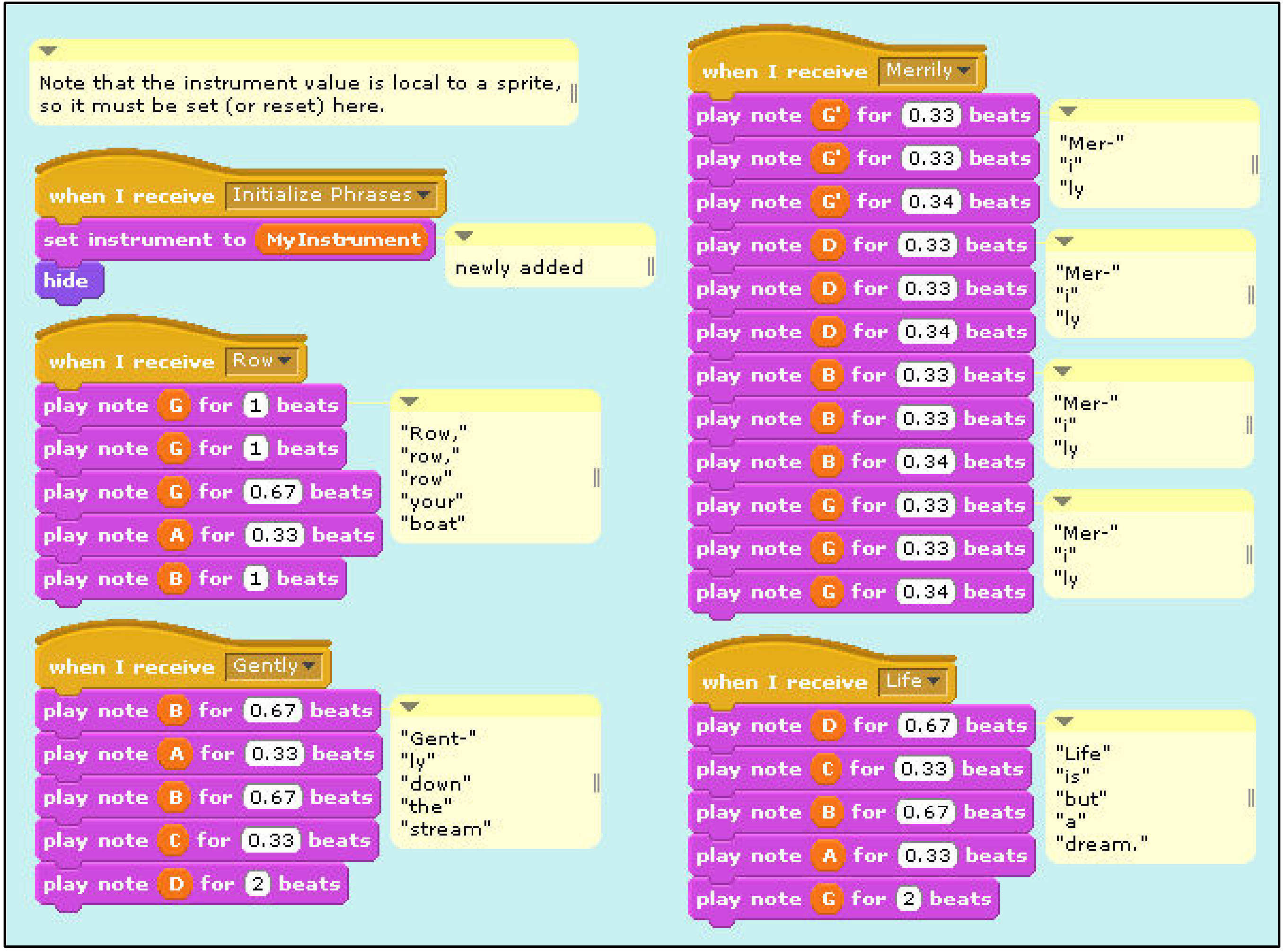
```
when I receive [Play G Major Scale ▾]
broadcast [Initialize Note Values ▾] and wait
play note (G) for (0.5) beats
play note (A) for (0.5) beats
play note (B) for (0.5) beats
play note (C) for (0.5) beats
play note (D) for (0.5) beats
play note (E) for (0.5) beats
play note (F#) for (0.5) beats
play note (G') for (0.5) beats
```

continued on next page

Row, Row, Row Your Boat

Version 4: Separating Phrases (cont'd)

(4c) **Phrases** Script



Note that the instrument value is local to a sprite, so it must be set (or reset) here.

```
when I receive Initialize Phrases
set instrument to MyInstrument      newly added
hide
```

```
when I receive Row
play note G for 1 beats             "Row,"
play note G for 1 beats             "row,"
play note G for 0.67 beats          "row"
play note A for 0.33 beats          "your"
play note B for 1 beats             "boat"
```

```
when I receive Gently
play note B for 0.67 beats          "Gent-"
play note A for 0.33 beats          "ly"
play note B for 0.67 beats          "down"
play note C for 0.33 beats          "the"
play note D for 2 beats             "stream"
```

```
when I receive Merrily
play note G' for 0.33 beats         "Mer-"
play note G' for 0.33 beats         "i"
play note G' for 0.34 beats         "ly
play note D for 0.33 beats          "Mer-"
play note D for 0.33 beats          "i"
play note D for 0.34 beats          "ly
play note B for 0.33 beats          "Mer-"
play note B for 0.33 beats          "i"
play note B for 0.34 beats          "ly
play note G for 0.33 beats          "Mer-"
play note G for 0.33 beats          "i"
play note G for 0.34 beats          "ly
```

```
when I receive Life
play note D for 0.67 beats          "Life"
play note C for 0.33 beats          "is"
play note B for 0.67 beats          "but"
play note A for 0.33 beats          "a"
play note G for 2 beats             "dream."
```

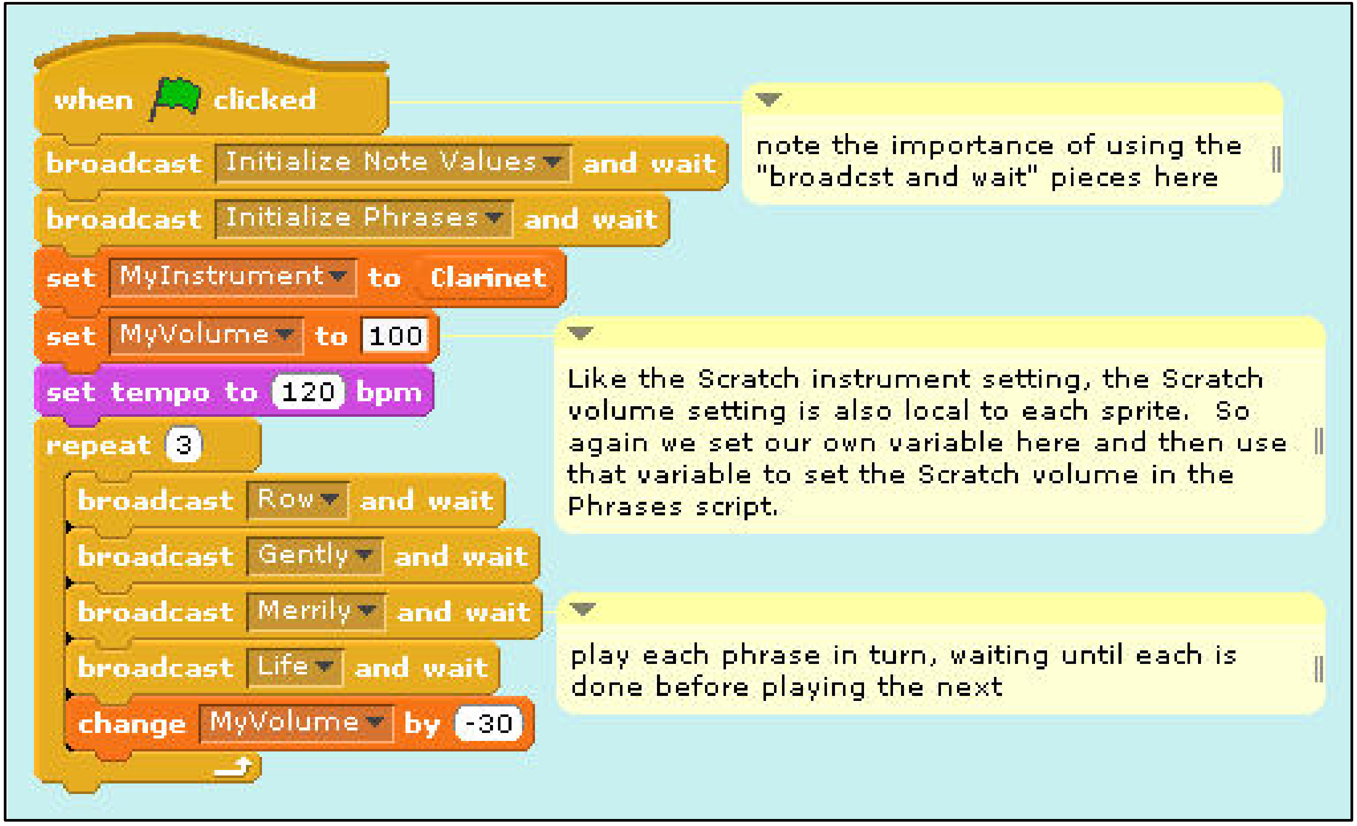
end of Version 4

Row, Row, Row Your Boat

Version 5: Looping and Fading

Three Scripts

(5a) **Main** Script



```
when [green flag] clicked
broadcast [Initialize Note Values] and wait
broadcast [Initialize Phrases] and wait
set [MyInstrument] to Clarinet
set [MyVolume] to 100
set tempo to 120 bpm
repeat 3
    broadcast [Row] and wait
    broadcast [Gently] and wait
    broadcast [Merrily] and wait
    broadcast [Life] and wait
    change [MyVolume] by -30
```

note the importance of using the "broadcst and wait" pieces here

Like the Scratch instrument setting, the Scratch volume setting is also local to each sprite. So again we set our own variable here and then use that variable to set the Scratch volume in the Phrases script.

play each phrase in turn, waiting until each is done before playing the next

(5b) **Initialization ("Init")** Script

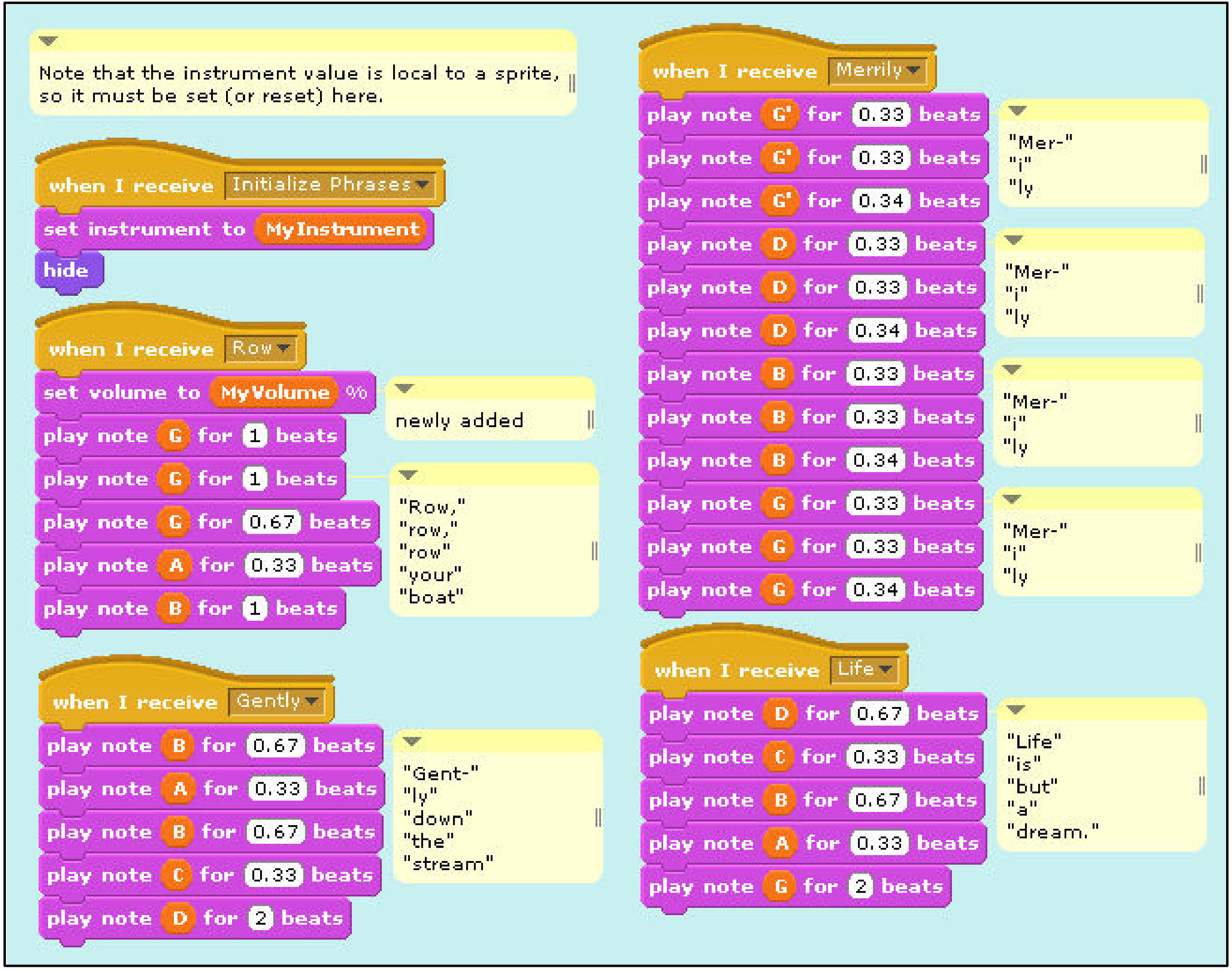
(*same as on page 34*)

continued on next page

Row, Row, Row Your Boat

Version 5: Looping and Fading (cont'd)

(5c) **Phrases** Script



```
Note that the instrument value is local to a sprite,
so it must be set (or reset) here.

when I receive Initialize Phrases
set instrument to MyInstrument
hide

when I receive Row
set volume to MyVolume %          newly added
play note G for 1 beats
play note G for 1 beats           "Row,"
play note G for 0.67 beats        "row,"
play note A for 0.33 beats        "row"
play note B for 1 beats           "your"
                                  "boat"

when I receive Gently
play note B for 0.67 beats        "Gent-"
play note A for 0.33 beats        "ly"
play note B for 0.67 beats        "down"
play note C for 0.33 beats        "the"
play note D for 2 beats           "stream"

when I receive Merrily
play note G' for 0.33 beats       "Mer-"
play note G' for 0.33 beats       "i"
play note G' for 0.34 beats       "ly
play note D for 0.33 beats        "Mer-"
play note D for 0.33 beats        "i"
play note D for 0.34 beats        "ly
play note B for 0.33 beats        "Mer-"
play note B for 0.33 beats        "i"
play note B for 0.34 beats        "ly
play note G for 0.33 beats        "Mer-"
play note G for 0.33 beats        "i"
play note G for 0.34 beats        "ly

when I receive Life
play note D for 0.67 beats        "Life"
play note C for 0.33 beats        "is"
play note B for 0.67 beats        "but"
play note A for 0.33 beats        "a"
play note G for 2 beats           "dream."
```

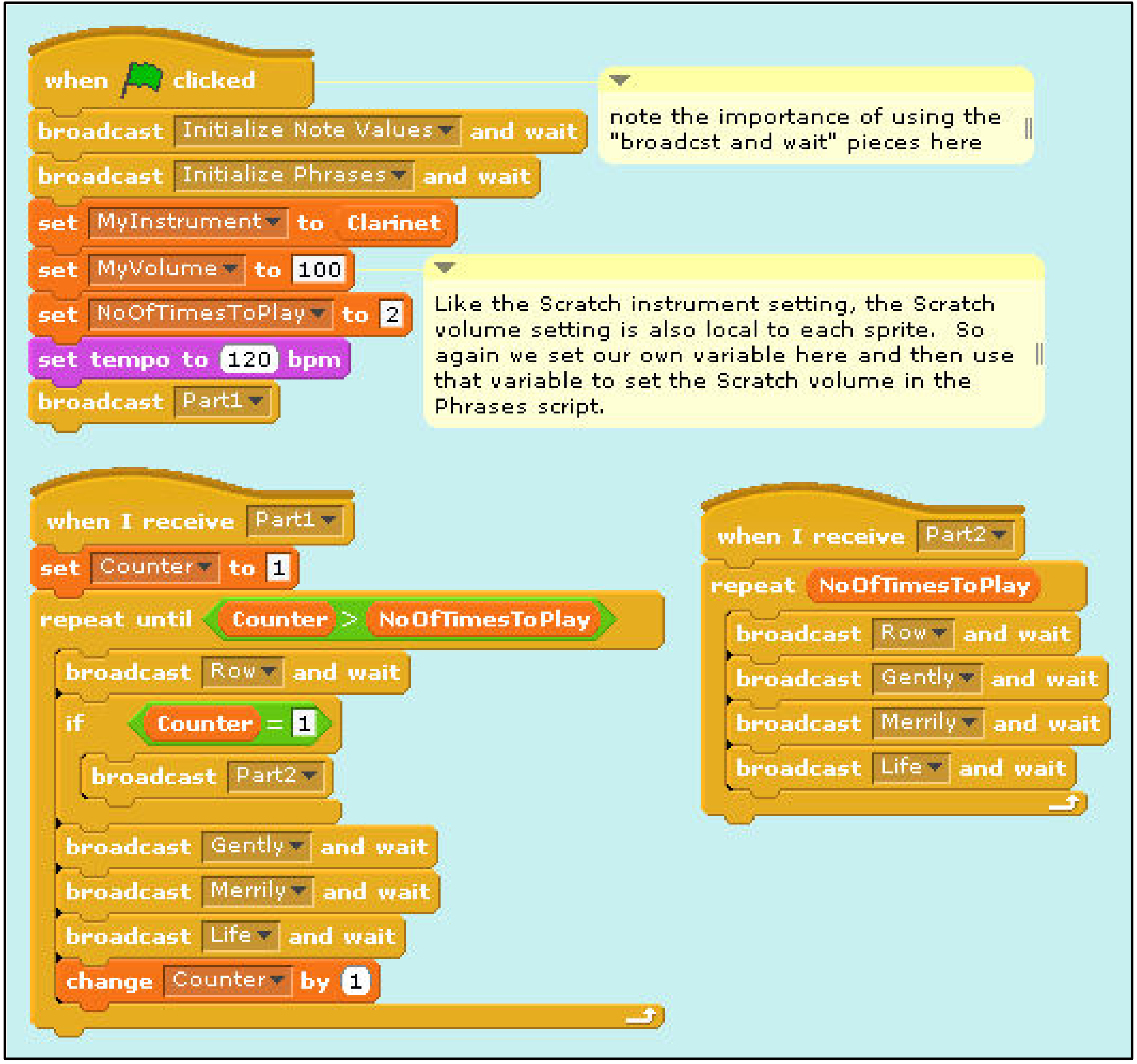
end of Version 5

Row, Row, Row Your Boat

Version 6: Playing a Round with One Instrument

Three Scripts

(6a) **Main** Script



```
when green flag clicked
broadcast [Initialize Note Values] and wait
broadcast [Initialize Phrases] and wait
set [MyInstrument] to (Clarinet)
set [MyVolume] to (100)
set [NoOfTimesToPlay] to (2)
set tempo to (120) bpm
broadcast [Part1]
```

note the importance of using the "broadcst and wait" pieces here

Like the Scratch instrument setting, the Scratch volume setting is also local to each sprite. So again we set our own variable here and then use that variable to set the Scratch volume in the Phrases script.

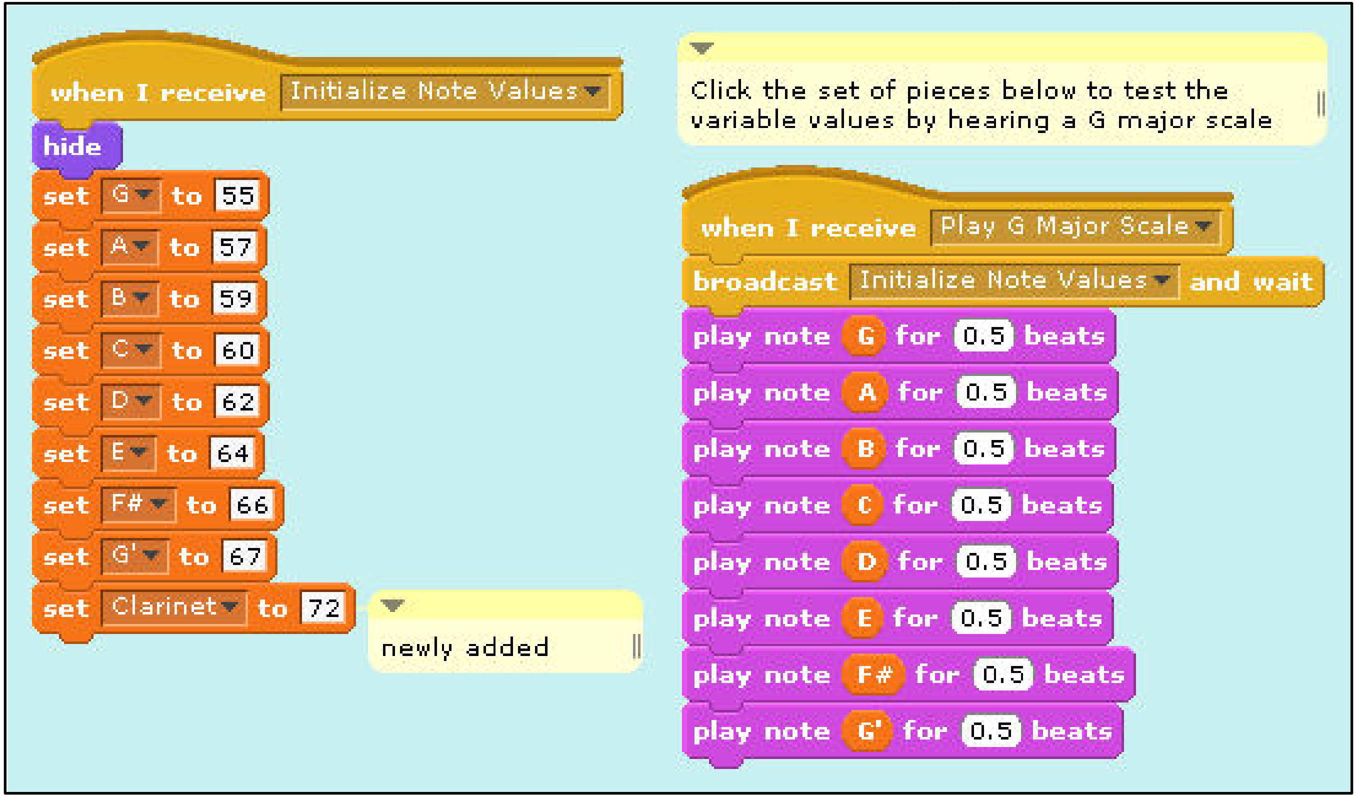
```
when I receive [Part1]
set [Counter] to (1)
repeat until <(Counter) > (NoOfTimesToPlay)>
    broadcast [Row] and wait
    if <(Counter) = (1)>
        broadcast [Part2]
    broadcast [Gently] and wait
    broadcast [Merrily] and wait
    broadcast [Life] and wait
    change [Counter] by (1)
```

```
when I receive [Part2]
repeat (NoOfTimesToPlay)
    broadcast [Row] and wait
    broadcast [Gently] and wait
    broadcast [Merrily] and wait
    broadcast [Life] and wait
```

Row, Row, Row Your Boat

Version 6: Playing a Round with One Instrument (cont'd)

(6b) **Initialization ("Init")** Script



```
when I receive [Initialize Note Values ▾]
hide
set [G ▾] to (55)
set [A ▾] to (57)
set [B ▾] to (59)
set [C ▾] to (60)
set [D ▾] to (62)
set [E ▾] to (64)
set [F# ▾] to (66)
set [G' ▾] to (67)
set [Clarinet ▾] to (72)
```

newly added

Click the set of pieces below to test the variable values by hearing a G major scale

```
when I receive [Play G Major Scale ▾]
broadcast [Initialize Note Values ▾] and wait
play note (G) for (0.5) beats
play note (A) for (0.5) beats
play note (B) for (0.5) beats
play note (C) for (0.5) beats
play note (D) for (0.5) beats
play note (E) for (0.5) beats
play note (F#) for (0.5) beats
play note (G') for (0.5) beats
```

(6c) **Phrases** Script

(*same as on page 37*)

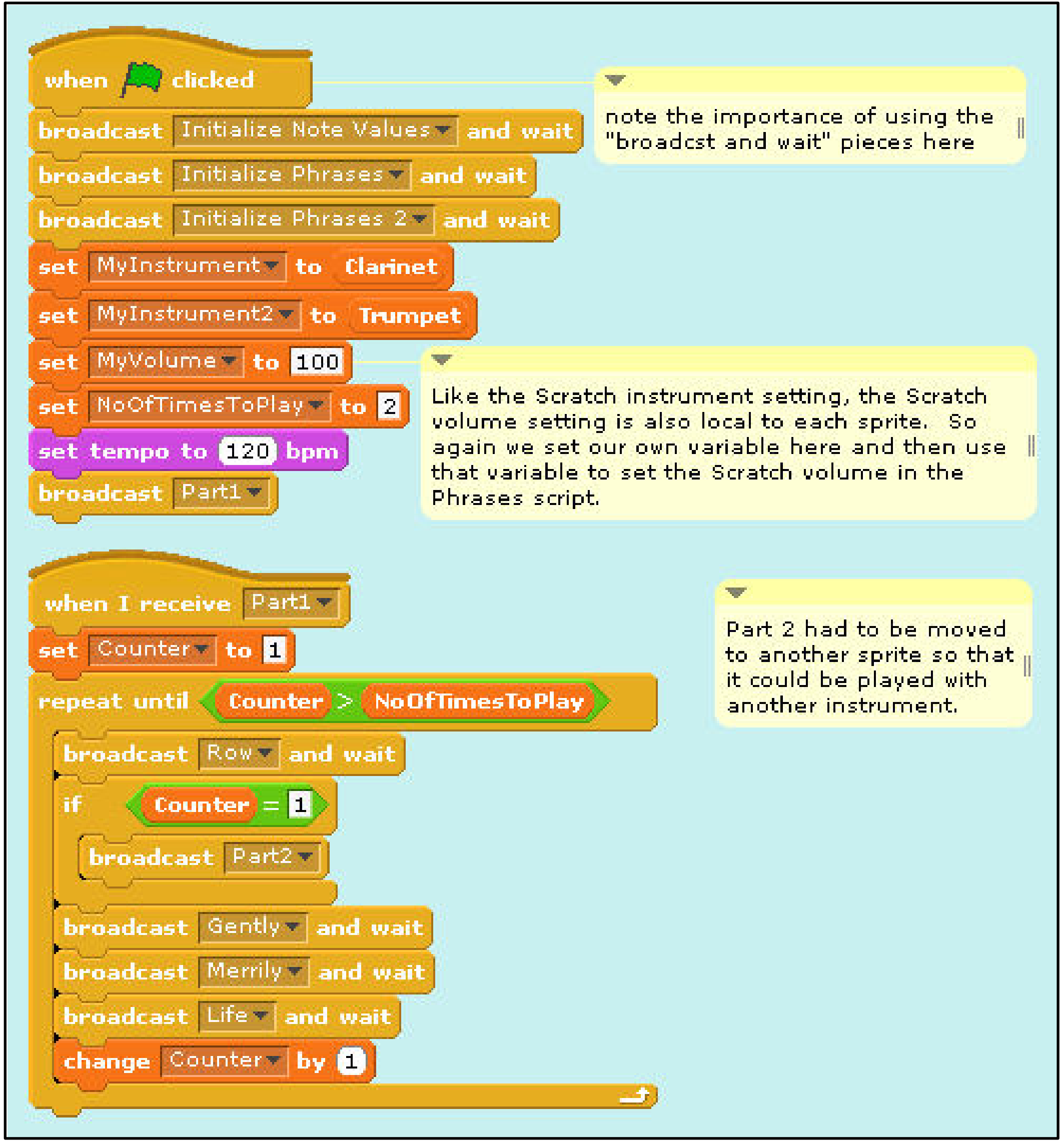
end of Version 6

Row, Row, Row Your Boat

Version 7: Playing a Round with Two Instruments

Five Scripts

(7a) **Main** Script



```
when ⚑ clicked
broadcast [Initialize Note Values] and wait
broadcast [Initialize Phrases] and wait
broadcast [Initialize Phrases 2] and wait
set [MyInstrument] to (Clarinet)
set [MyInstrument2] to (Trumpet)
set [MyVolume] to 100
set [NoOfTimesToPlay] to 2
set tempo to 120 bpm
broadcast [Part1]
```

note the importance of using the "broadcst and wait" pieces here

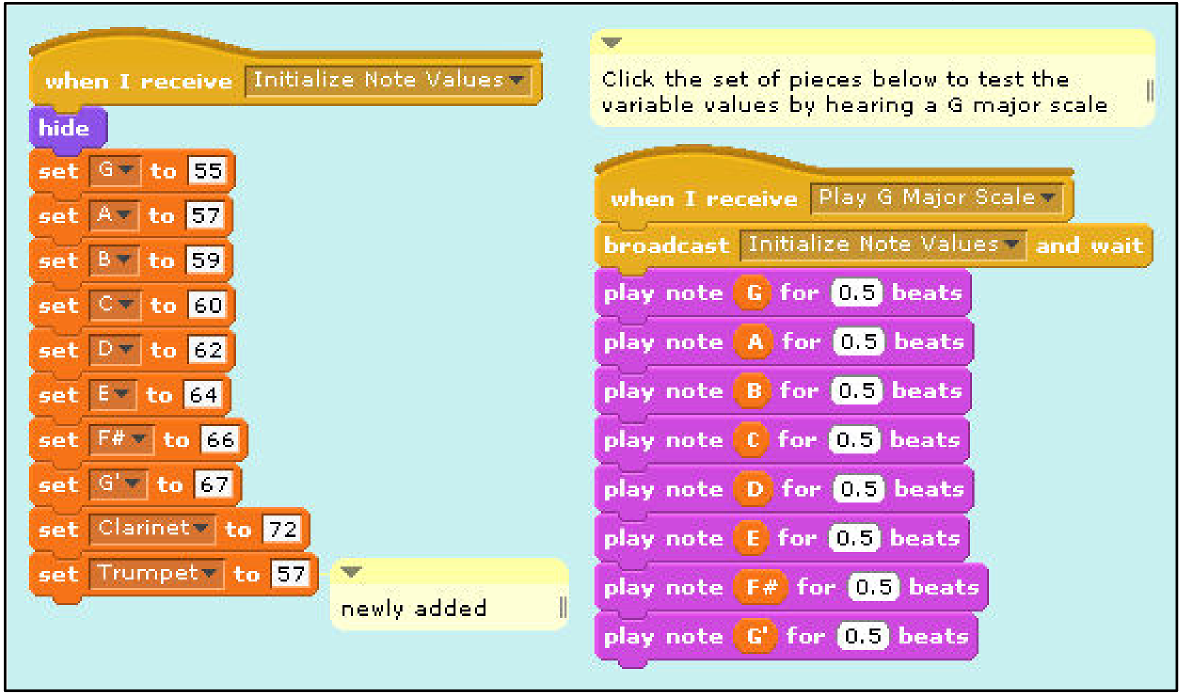
Like the Scratch instrument setting, the Scratch volume setting is also local to each sprite. So again we set our own variable here and then use that variable to set the Scratch volume in the Phrases script.

```
when I receive [Part1]
set [Counter] to 1
repeat until <(Counter) > (NoOfTimesToPlay)>
    broadcast [Row] and wait
    if <(Counter) = 1>
        broadcast [Part2]
    broadcast [Gently] and wait
    broadcast [Merrily] and wait
    broadcast [Life] and wait
    change [Counter] by 1
```

Part 2 had to be moved to another sprite so that it could be played with another instrument.

Version 7: Playing a Round with Two Instruments (cont'd)

(7b) **Initialization ("Init")** Script



```
when I receive [Initialize Note Values]
hide
set [G] to 55
set [A] to 57
set [B] to 59
set [C] to 60
set [D] to 62
set [E] to 64
set [F#] to 66
set [G'] to 67
set [Clarinet] to 72
set [Trumpet] to 57
```

newly added

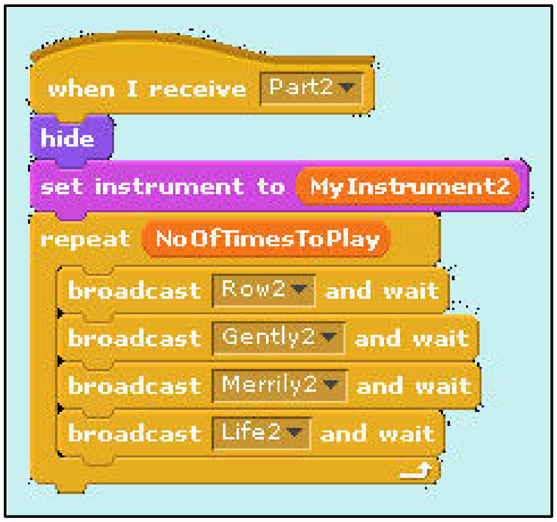
Click the set of pieces below to test the variable values by hearing a G major scale

```
when I receive [Play G Major Scale]
broadcast [Initialize Note Values] and wait
play note (G) for 0.5 beats
play note (A) for 0.5 beats
play note (B) for 0.5 beats
play note (C) for 0.5 beats
play note (D) for 0.5 beats
play note (E) for 0.5 beats
play note (F#) for 0.5 beats
play note (G') for 0.5 beats
```

(7c) **Phrases** Script

(*same as on page 37*)

(7d) **Part2** Script ➔



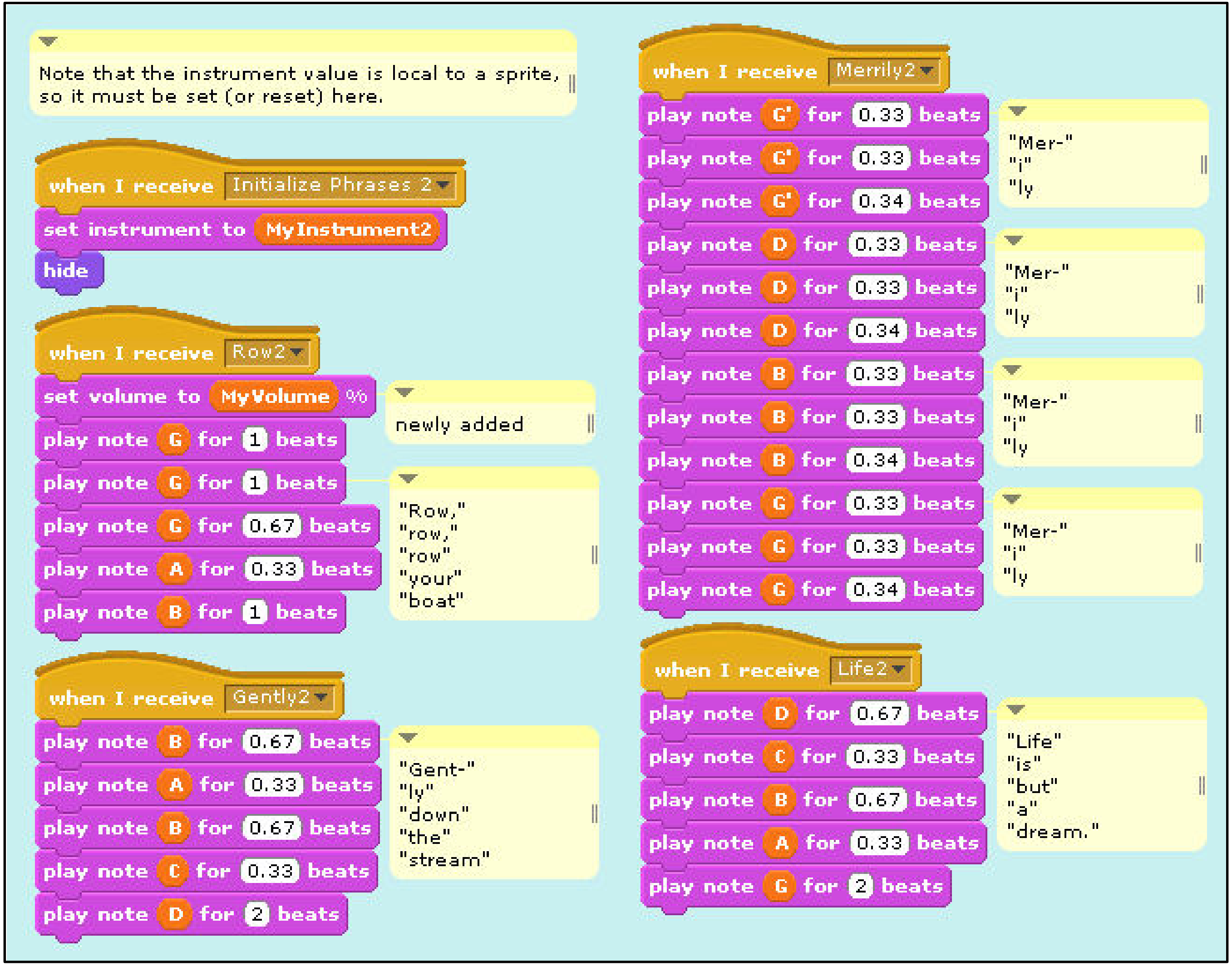
```
when I receive [Part2]
hide
set instrument to (MyInstrument2)
repeat (NoOfTimesToPlay)
    broadcast [Row2] and wait
    broadcast [Gently2] and wait
    broadcast [Merrily2] and wait
    broadcast [Life2] and wait
```

continued on next page

Row, Row, Row Your Boat

Version 7: Playing a Round with Two Instruments (cont'd)

(7e) **Instrument2 ("Instru2")** Script



Note that the instrument value is local to a sprite, so it must be set (or reset) here.

```
when I receive [Initialize Phrases 2]
set instrument to (MyInstrument2)
hide
```

```
when I receive [Row2]
set volume to (MyVolume) %
play note (G) for (1) beats
play note (G) for (1) beats
play note (G) for (0.67) beats
play note (A) for (0.33) beats
play note (B) for (1) beats
```

newly added

"Row,"
"row,"
"row"
"your"
"boat"

```
when I receive [Gently2]
play note (B) for (0.67) beats
play note (A) for (0.33) beats
play note (B) for (0.67) beats
play note (C) for (0.33) beats
play note (D) for (2) beats
```

"Gent-"
"ly"
"down"
"the"
"stream"

```
when I receive [Merrily2]
play note (G') for (0.33) beats
play note (G') for (0.33) beats
play note (G') for (0.34) beats
play note (D) for (0.33) beats
play note (D) for (0.33) beats
play note (D) for (0.34) beats
play note (B) for (0.33) beats
play note (B) for (0.33) beats
play note (B) for (0.34) beats
play note (G) for (0.33) beats
play note (G) for (0.33) beats
play note (G) for (0.34) beats
```

"Mer-"
"i"
"ly

"Mer-"
"i"
"ly

"Mer-"
"i"
"ly

"Mer-"
"i"
"ly

```
when I receive [Life2]
play note (D) for (0.67) beats
play note (C) for (0.33) beats
play note (B) for (0.67) beats
play note (A) for (0.33) beats
play note (G) for (2) beats
```

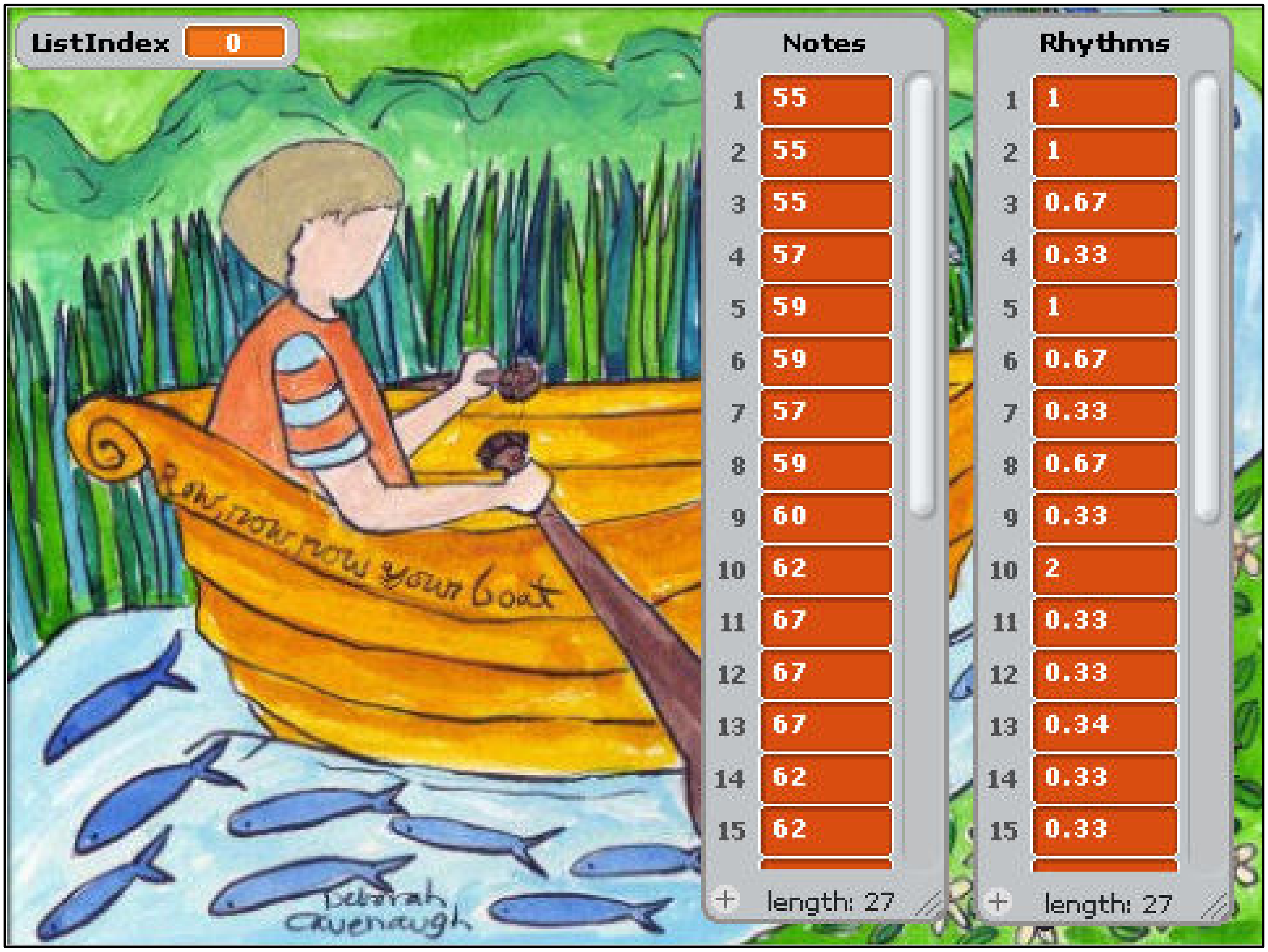
"Life"
"is"
"but"
"a"
"dream."

end of Version 7

Row, Row, Row Your Boat

Version 8: Storing Notes and Rhythms in Lists

Output Window



ListIndex 0

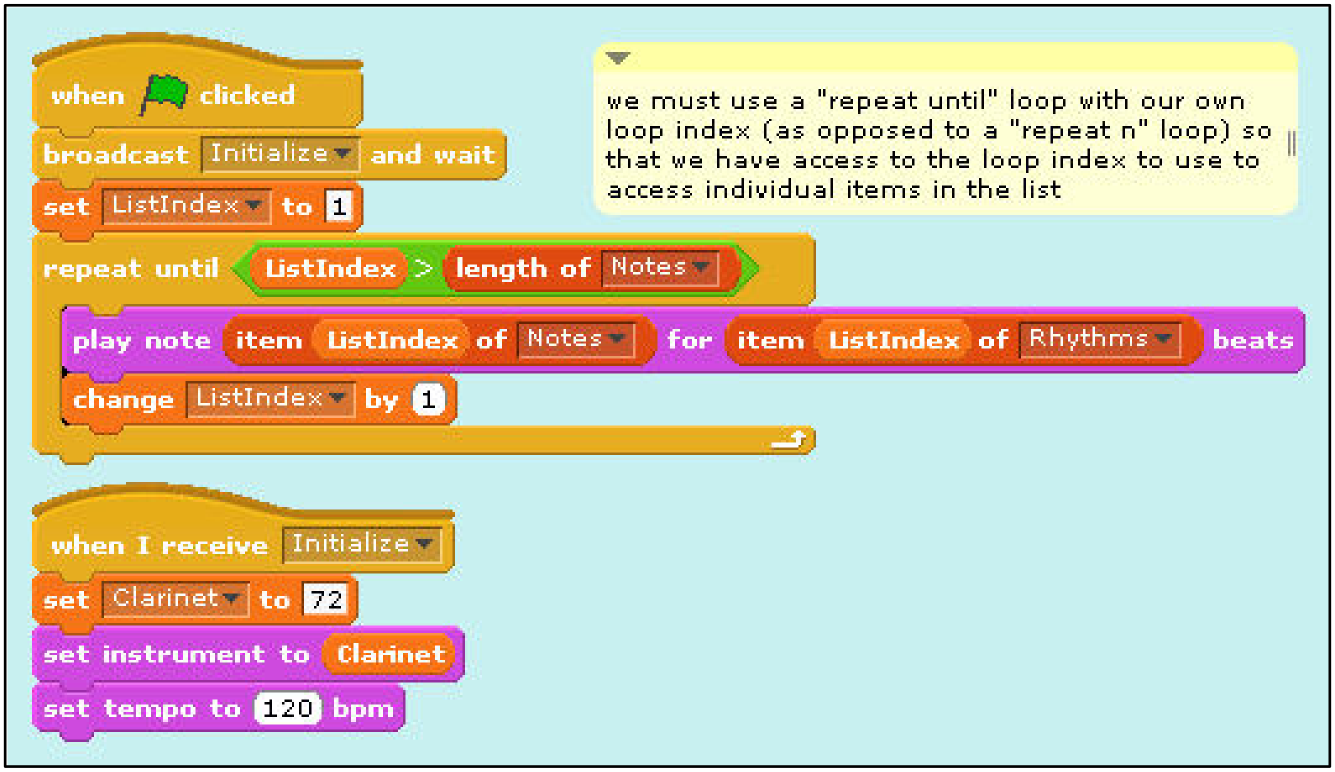
	Notes	Rhythms
1	55	1
2	55	1
3	55	0.67
4	57	0.33
5	59	1
6	59	0.67
7	57	0.33
8	59	0.67
9	60	0.33
10	62	2
11	67	0.33
12	67	0.33
13	67	0.34
14	62	0.33
15	62	0.33
	length: 27	length: 27

continued on next page

Row, Row, Row Your Boat

Version 8: Storing Notes and Rhythms in Lists (cont'd)

Single Script



```
when [flag] clicked
broadcast Initialize and wait
set ListIndex to 1
repeat until ListIndex > length of Notes
    play note item ListIndex of Notes for item ListIndex of Rhythms beats
    change ListIndex by 1

when I receive Initialize
set Clarinet to 72
set instrument to Clarinet
set tempo to 120 bpm
```

we must use a "repeat until" loop with our own loop index (as opposed to a "repeat n" loop) so that we have access to the loop index to use to access individual items in the list

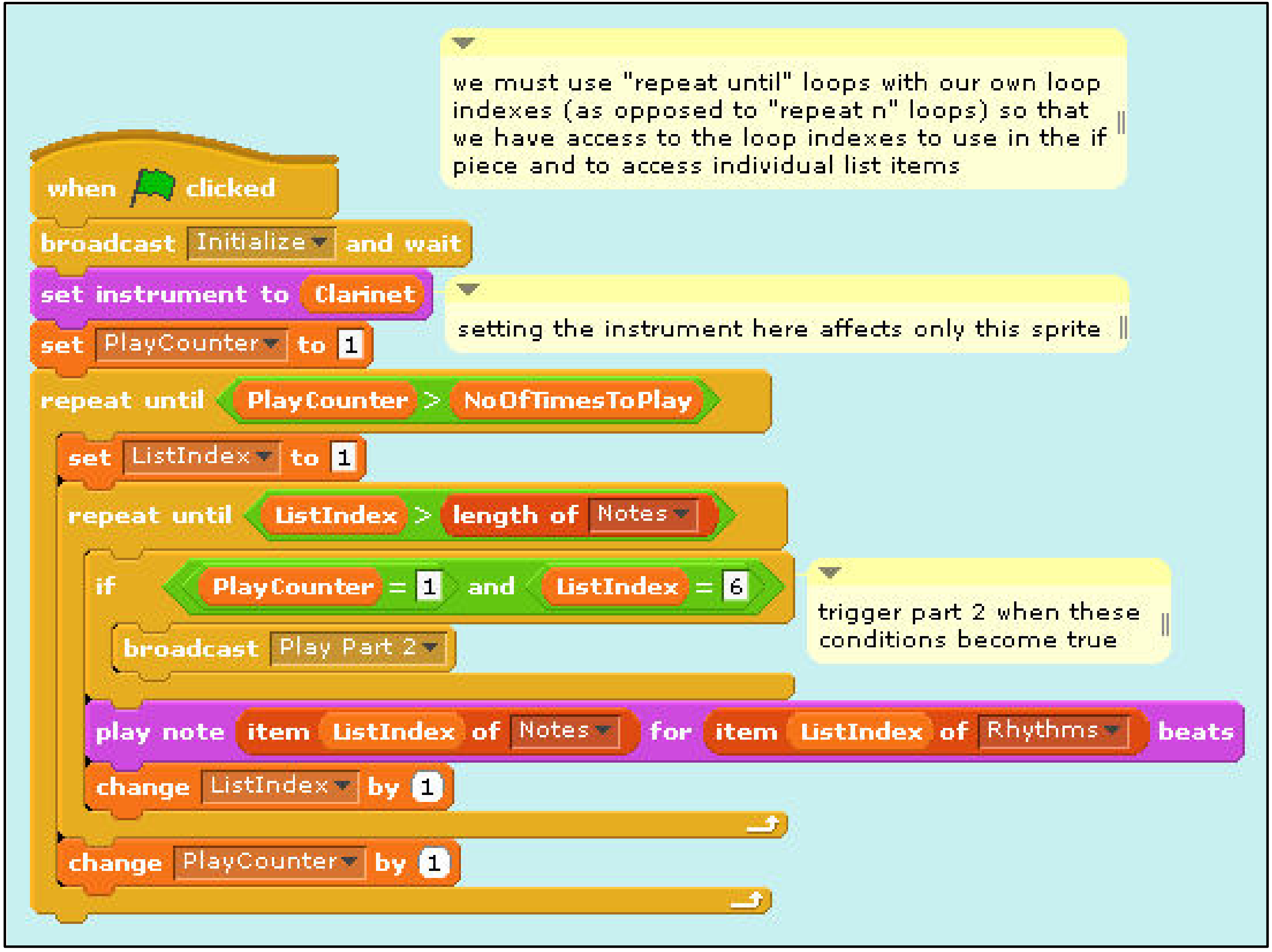
end of Version 8

Row, Row, Row Your Boat

Version 9: Playing a Round Using Lists

Three Scripts

(9a) **Main** Script



we must use "repeat until" loops with our own loop indexes (as opposed to "repeat n" loops) so that we have access to the loop indexes to use in the if piece and to access individual list items

```
when [flag] clicked
broadcast Initialize and wait
set instrument to Clarinet
set PlayCounter to 1
repeat until PlayCounter > NoOfTimesToPlay
    set ListIndex to 1
    repeat until ListIndex > length of Notes
        if PlayCounter = 1 and ListIndex = 6
            broadcast Play Part 2
        play note item ListIndex of Notes for item ListIndex of Rhythms beats
        change ListIndex by 1
    change PlayCounter by 1
```

setting the instrument here affects only this sprite

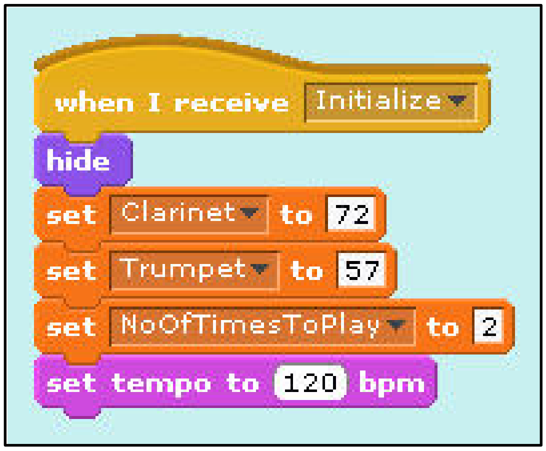
trigger part 2 when these conditions become true

continued on next page

Row, Row, Row Your Boat

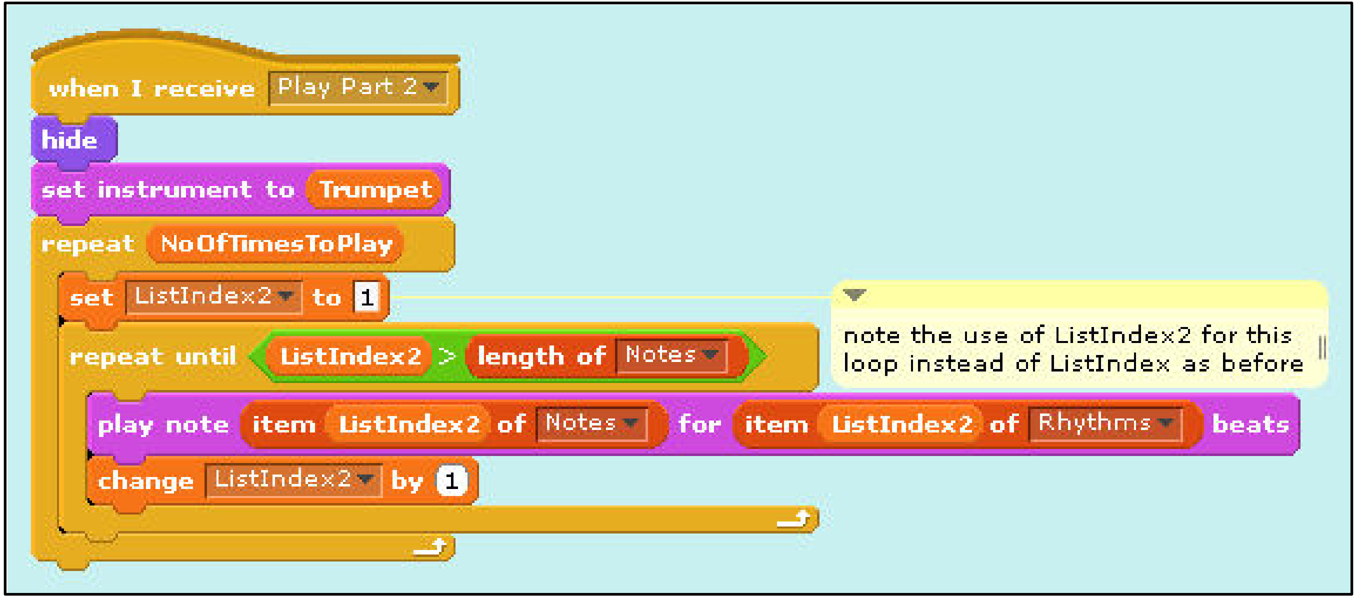
Version 9: Playing a Round Using Lists (cont'd)

(9b) **Initialization ("Init")** Script



```
when I receive Initialize
hide
set Clarinet to 72
set Trumpet to 57
set NoOfTimesToPlay to 2
set tempo to 120 bpm
```

(9c) **Part2** Script



```
when I receive Play Part 2
hide
set instrument to Trumpet
repeat NoOfTimesToPlay
    set ListIndex2 to 1
    repeat until ListIndex2 > length of Notes
        play note item ListIndex2 of Notes for item ListIndex2 of Rhythms beats
        change ListIndex2 by 1
```

note the use of ListIndex2 for this loop instead of ListIndex as before

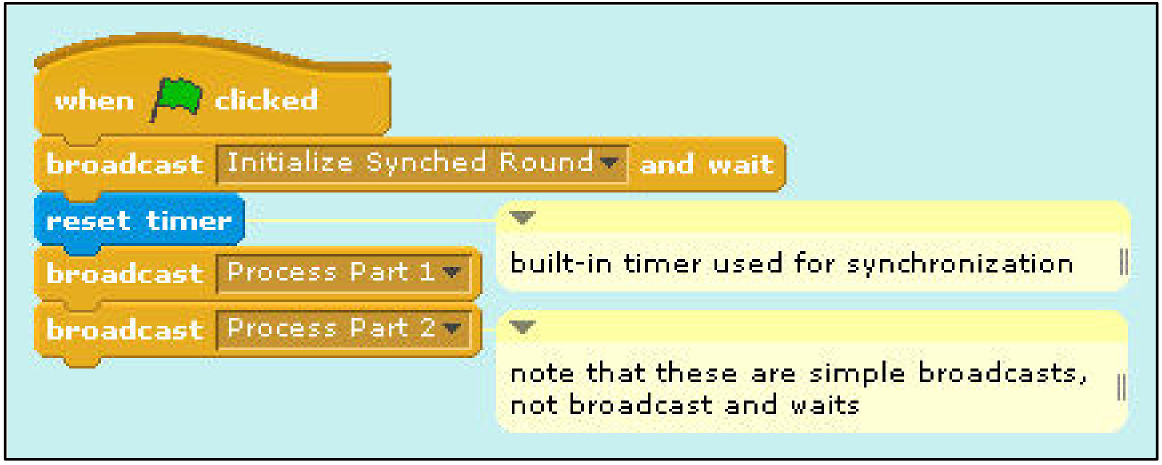
end of Version 9

Row, Row, Row Your Boat

Version 10: Synchronizing Play from Lists

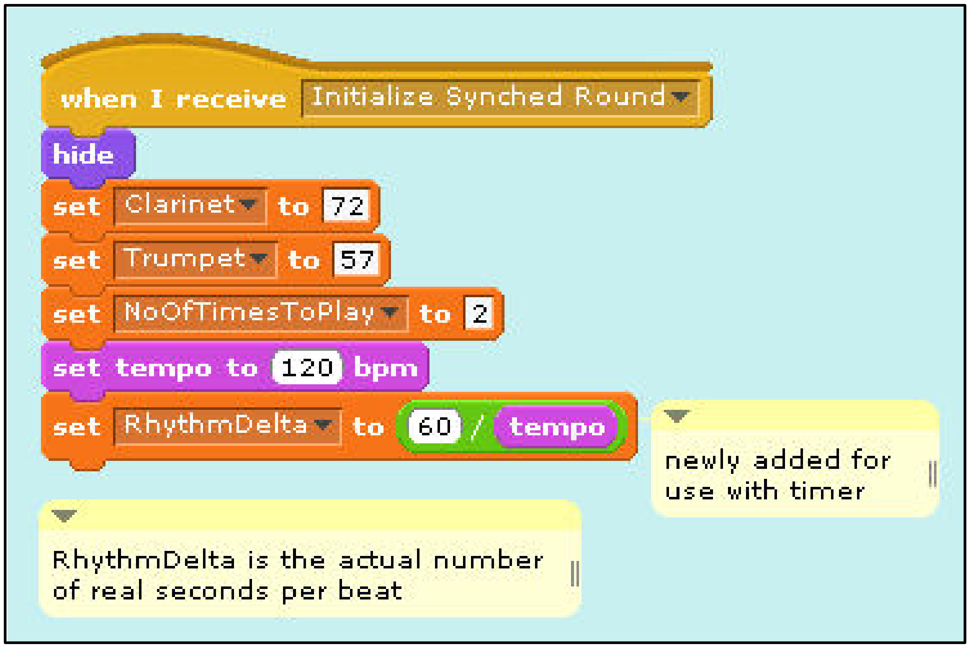
Four Scripts

(10a) **Main** Script



```
when ⚑ clicked
broadcast [Initialize Synched Round ▾] and wait
reset timer                          built-in timer used for synchronization
broadcast [Process Part 1 ▾]
broadcast [Process Part 2 ▾]         note that these are simple broadcasts,
                                     not broadcast and waits
```

(10b) **Initialization ("Init")** Script



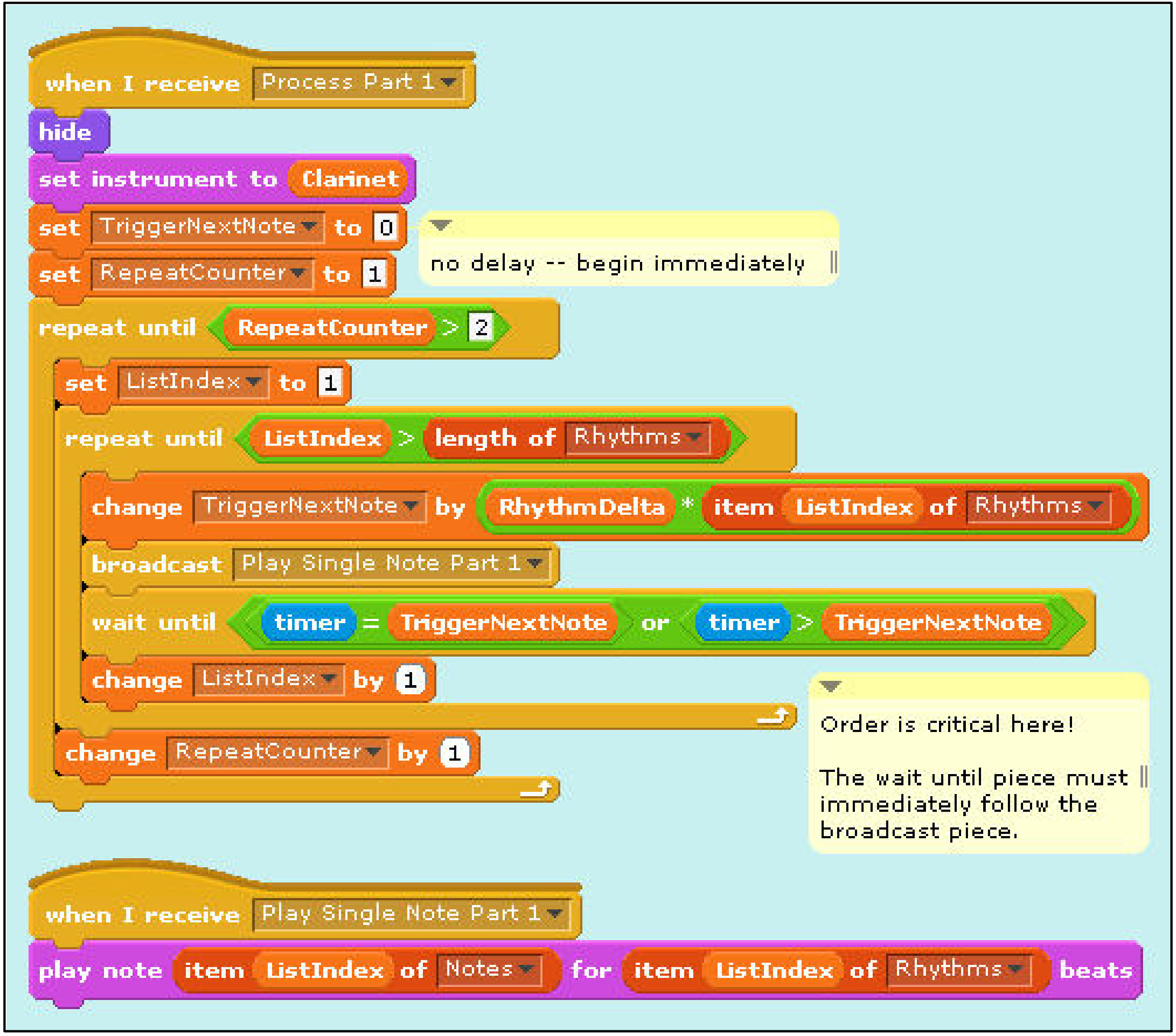
```
when I receive [Initialize Synched Round ▾]
hide
set [Clarinet ▾] to 72
set [Trumpet ▾] to 57
set [NoOfTimesToPlay ▾] to 2
set tempo to 120 bpm
set [RhythmDelta ▾] to (60 / tempo)    newly added for
                                       use with timer

RhythmDelta is the actual number
of real seconds per beat
```

continued on next page

Row, Row, Row Your Boat

Version 10: Synchronizing Play from Lists (cont'd)

(10c) **Part 1** Script



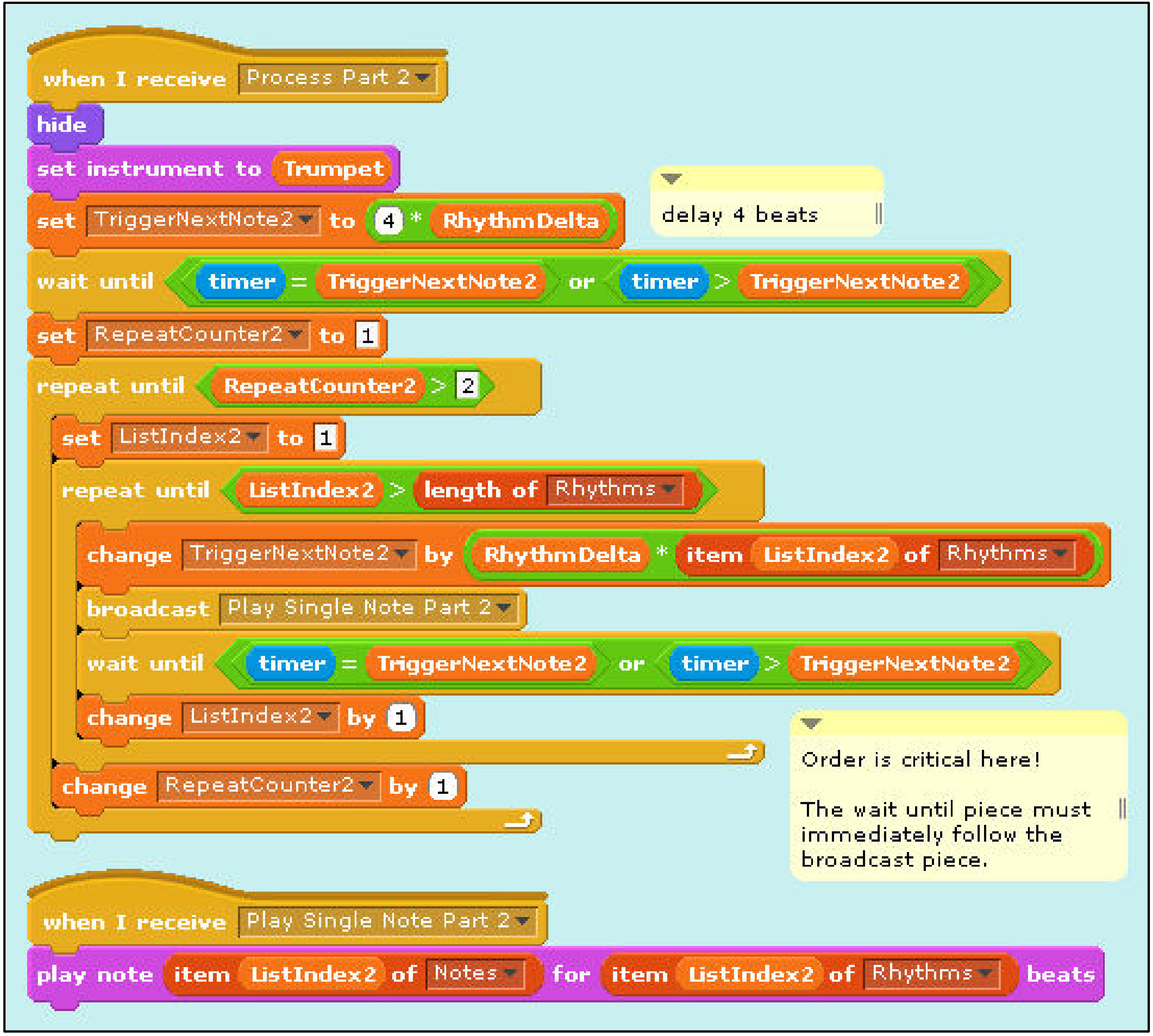
```
when I receive [Process Part 1]
hide
set instrument to (Clarinet)
set [TriggerNextNote] to [0]          no delay -- begin immediately
set [RepeatCounter] to [1]
repeat until < (RepeatCounter) > [2] >
    set [ListIndex] to [1]
    repeat until < (ListIndex) > (length of [Rhythms]) >
        change [TriggerNextNote] by ((RhythmDelta) * (item (ListIndex) of [Rhythms]))
        broadcast [Play Single Note Part 1]
        wait until < < (timer) = (TriggerNextNote) > or < (timer) > (TriggerNextNote) > >
        change [ListIndex] by (1)
    change [RepeatCounter] by (1)
                                      Order is critical here!

                                      The wait until piece must
                                      immediately follow the
                                      broadcast piece.

when I receive [Play Single Note Part 1]
play note (item (ListIndex) of [Notes]) for (item (ListIndex) of [Rhythms]) beats
```

continued on next page

Row, Row, Row Your Boat

Version 10: Synchronizing Play from Lists (cont'd)

(10d) **Part 2** Script



```
when I receive [Process Part 2]
hide
set instrument to (Trumpet)
set [TriggerNextNote2] to ((4) * (RhythmDelta))
wait until <<(timer) = (TriggerNextNote2)> or <(timer) > (TriggerNextNote2)>>
set [RepeatCounter2] to (1)
repeat until <(RepeatCounter2) > (2)>
    set [ListIndex2] to (1)
    repeat until <(ListIndex2) > (length of [Rhythms])>
        change [TriggerNextNote2] by ((RhythmDelta) * (item (ListIndex2) of [Rhythms]))
        broadcast [Play Single Note Part 2]
        wait until <<(timer) = (TriggerNextNote2)> or <(timer) > (TriggerNextNote2)>>
        change [ListIndex2] by (1)
    end
    change [RepeatCounter2] by (1)
end

when I receive [Play Single Note Part 2]
play note (item (ListIndex2) of [Notes]) for (item (ListIndex2) of [Rhythms]) beats
```

delay 4 beats

Order is critical here!

The wait until piece must immediately follow the broadcast piece.

end of Version 10

Extending the Examples

1. **Use a variable to set the tempo.**
   - Add a slider to the variable so that you can change the tempo in real time.
   - Find all the places you need to use the variable to reset the tempo when you change it in real time.
   - Which version of playing the round best stays synchronized when you change the tempo?

2. **Transpose the melody to another key.**
   - Create a variable to hold a pitch offset.
   - Find all the places you need to use that variable to play the melody in the new key.

3. **Increase the number of times that the round repeats.**
   - Do the parts stay in synch?

4. **Increase the number of parts that play simultaneously.** (Be sure to set Turbo Speed to do this!)
   - When should each part "come in"?
   - How much should the first beat of each part be offset?

Extending the Examples (cont'd)

5. **Play the melody backwards.**
   - Can you play multiple parts backwards, too?

6. **Increase the number of times that the round repeats.**
   - Do the parts stay in synch?

7. **Increase the number of parts that play simultaneously.** (Be sure to set Turbo Speed before you try this!)
   - When should each part "come in"?
   - How much should the first beat of each part be offset?

8. **Make a round using the G-major scale.**
   - Put the note values for a G-major scale into a list. See page 32 for code that initializes and plays a G-major scale, but remember that you must use the integer values, not the variable names, to play notes from a list.
   - Start Part 2 when Part 1 plays its third note (B, MIDI note #59).
   - Add Part 3, starting when Part 1 plays its fifth not (D, #62).

Extending the Examples (cont'd)

9. **Play random notes in the G-major scale.**
   - Start with the list created for the previous exercise.
   - Use the "pick random" piece in the Operators group to pick a random note from the list.
   - Play each note for 0.25, 0.50, 0.75, or 1.00 beats, also selected randomly.
   - Does the result sound musical?

10. **Create a program that can play any <u>major</u> scale given any starting note.**
    - Store the starting note in a variable.
    - For a major scale, the number of half-tones between each note is:

      2, 2, 1, 2, 2, 2, 1
    - Another way to think about this is:

      Do + 2 ➔ Re + 2 ➔ Mi + 1 ➔ Fa + 2 ➔
      Sol + 2 ➔ La + 2 ➔Ti + 1 ➔ Do
    - Create a list containing the changes between the notes, and then use a loop to process the list and play the scale.

Extending the Examples (cont'd)

11. **Create a program that can play any <u>harmonic minor</u> scale given any starting note.**
    - For a harmonic minor scale, the number of half-tones between each note is:
      2, 1, 2, 2, 1, 3, 1
    - Create a new list containing these changes, but use the same loop that you created for the previous exercise to play this scale.

12. **Create a program to play a major chord.**
    - A major chord is the 1st, 3rd, and 5th notes of the scale, usually complemented by the octave above the 1st note. Thus, a G-major scale has notes G (#55), B (#59), D (#62), and G' (#67).
    - Another way to think about this is to compute the half-tone difference from the starting note: 0, 4, 7, 12.
    - Set a starting note and then use a "broadcast" to play the four notes simultaneously.

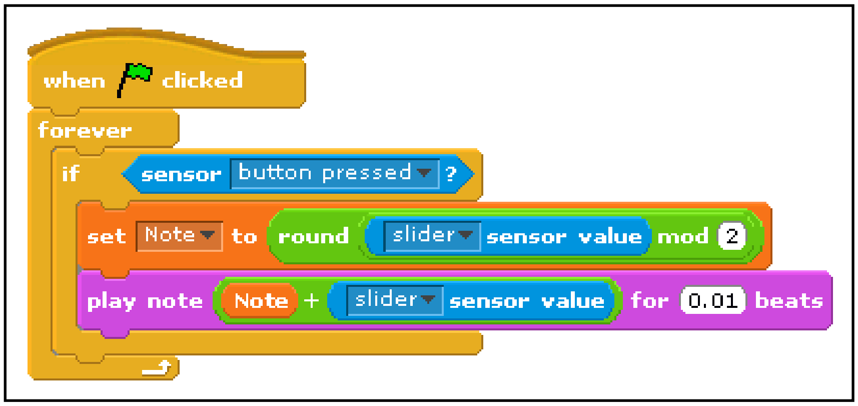
The IchiBoard

Board Layout

(courtesy of Mark Sherman,
UMass Lowell
Computer Science
Engaging Computing Group)

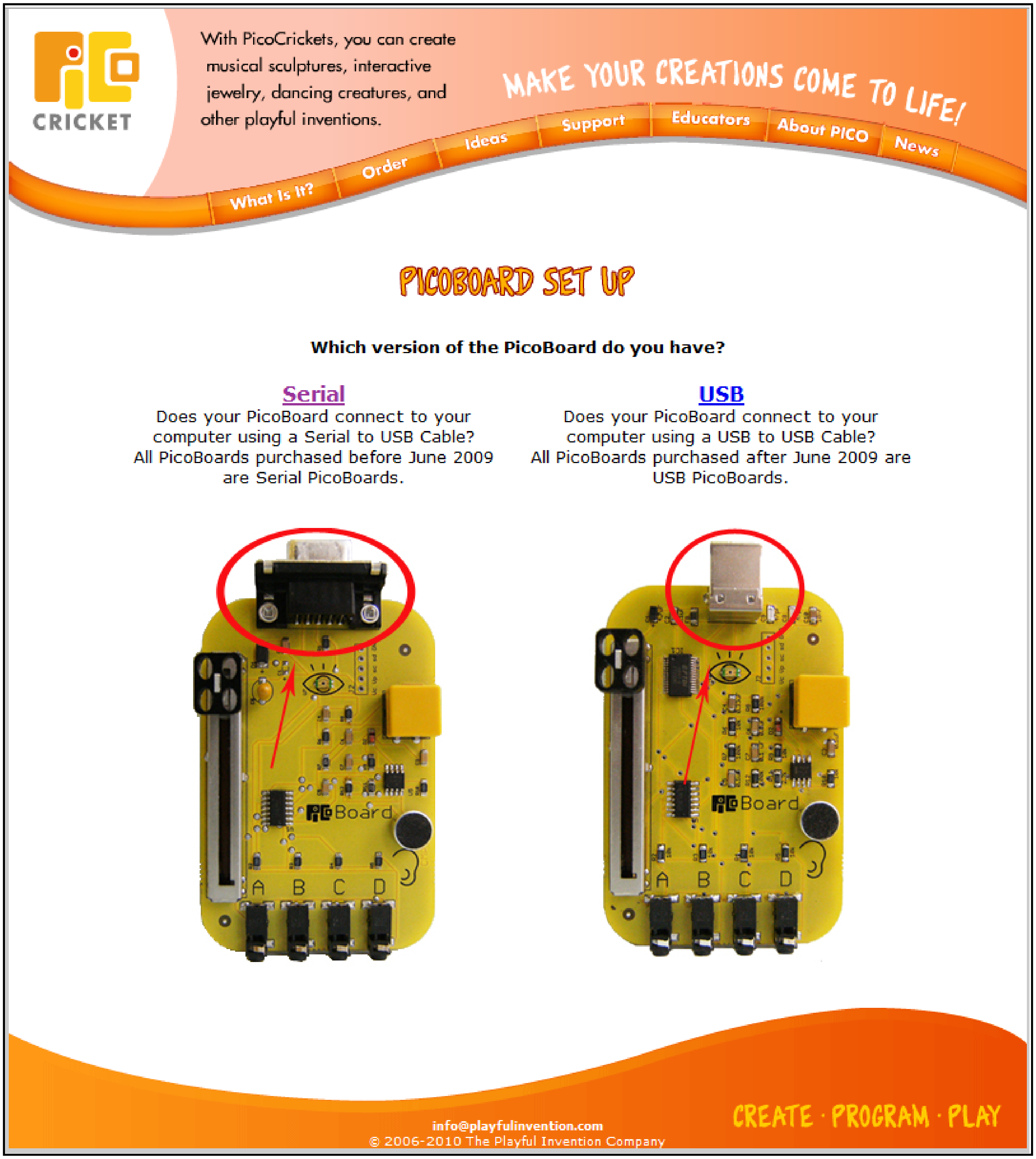
Scratch Code for an IchiBoard Musical Instrument

(courtesy of Alex Ruthmann)



```
when ⚑ clicked
forever
  if  sensor [button pressed ▾] ?
    set [Note ▾] to  round  ([slider ▾] sensor value) mod (2)
    play note (Note) + ([slider ▾] sensor value)  for (0.01) beats
```

The PicoBoard



With PicoCrickets, you can create musical sculptures, interactive jewelry, dancing creatures, and other playful inventions.

PICOBOARD SET UP

Which version of the PicoBoard do you have?

Serial
Does your PicoBoard connect to your computer using a Serial to USB Cable?
All PicoBoards purchased before June 2009 are Serial PicoBoards.

USB
Does your PicoBoard connect to your computer using a USB to USB Cable?
All PicoBoards purchased after June 2009 are USB PicoBoards.

info@playfulinvention.com
© 2006-2010 The Playful Invention Company

CREATE · PROGRAM · PLAY

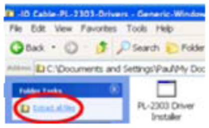
PicoBoard Serial to USB Setup

Windows Instructions

Windows XP (and older) users: Download the PicoBoard **Windows Driver (1471 KB)**

Windows Vista/7 users should just plug the USB to Serial cable in their computer and let the operating system choose the correct driver. Then skip to step #3 below.
If you do not have internet access on your computer, download the PicoBoard **Windows Driver (1471 KB)**.

1 First, open the file that you just downloaded. Click the link that says "Extract all files".


-IO Cable-PL-2303-Drivers - Generic-Window
File Edit View Favorites Tools Help
Back
Search
Folder
Address
C:\Documents and Settings\Paul\My Doc
Folder Tasks
Extract all files
PL-2303 Driver Installer

2 Then double-click on the file that you just extracted, and follow the on-screen instructions.

Mac Instructions

Mac OS X Driver (61 KB)

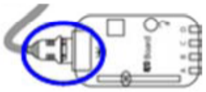
1 First, open the file that you just downloaded by clicking the magnifying glass in the Download panel.

2 Then double-click on the file that you just extracted, and follow the on-screen instructions.

Both Operating Systems:

3 Connect the USB part of your USB-Serial Cable to a USB port on your computer.

4 Connect the serial part of your USB-Serial Cable to the serial port on the PicoBoard.



5 Read **Getting Started with PicoBoards** to start making projects in **Scratch** with your PicoBoard.

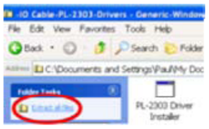
PicoBoard USB to USB Setup

Windows Instructions

Windows XP users: Download the PicoBoard **Windows Driver (1.71 MB)**

Windows Vista/Windows 7 users should just plug the USB cable in their computer and let the operating system choose the correct driver. Then skip to step #3 below.
If you do not have internet access on your computer, download the PicoBoard **Windows Driver (1.71 MB)**.

1 First, open the file that you just downloaded. Click the link that says "Extract all files".


-IO Cable-PL-2303-Drivers - Generic-Window
File Edit View Favorites Tools Help
Back
Search
Folder
Address C:\Documents and Settings\Paul\My Doc
Folder Tasks
Extract all files
PL-2303 Driver Installer

2 Then double-click on the file that you just extracted, and follow the on-screen instructions.

Mac Instructions

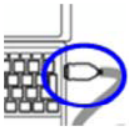
Mac OS X Driver (419 KB)

1 First, open the file that you just downloaded by clicking the magnifying glass in the Download panel.

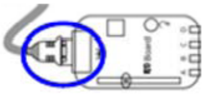
2 Then double-click on the file that you just extracted, and follow the on-screen instructions.

Both Operating Systems:

3 Connect the USB Cable to a USB port on your computer.



4 Connect the other part of the USB Cable to the USB port of the PicoBoard.



5 Read **Getting Started with PicoBoards** to start making projects in **Scratch** with your PicoBoard.

Computing and Music:
What Do They Have in Common?

Computing and music share deep structural similarities. For starters, both rely on notational symbol systems. Programming loops are typically delineated with opening and closing curly brackets { }, parentheses, or levels of indentation. Music loops are delineated with begin and end repeat signs 𝄆 𝄇 or initiated by "D.S." (Italian: *dal segno*), which instructs musicians to "repeat back to the sign," typically designated as 𝄋. As in programming, musical iteration can also make use of loop control variables. For example, Figure 1 shows a loop in which the music changes the second time through.

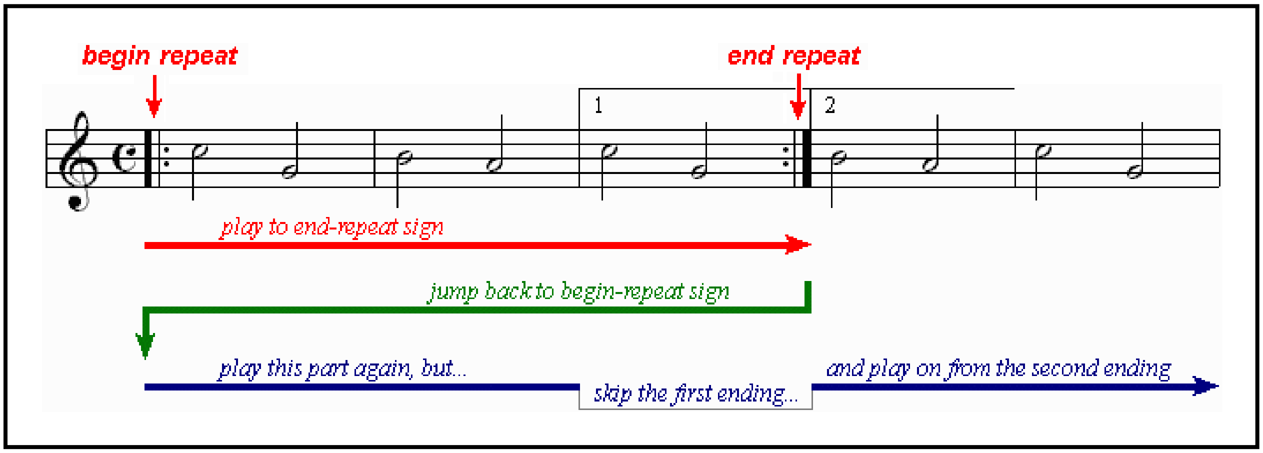

begin repeat
end repeat
1
2
play to end-repeat sign
jump back to begin-repeat sign
play this part again, but...
skip the first ending...
and play on from the second ending

Figure 1. Musical iteration with a loop control variable. [6]

Both computing and music have logic and flow. Figure 2 shows the logic one student saw in The Beatles' All You Need Is Love. If one were to turn this flowchart into a computer program, it would not only contains loops, but if and switch statements as well.

One can also go the other way, converting musical concepts into computer programs. For example, the Scratch [2, 3] program in Figure 3a plays Jimmy Page's famous guitar riff from Led Zeppelin's Kashmir. This code works properly, but consider the many computational thinking (CT) concepts learned by transforming the code in Figure 3a to 3b, which plays exactly the same riff.

Computing and Music (cont'd)

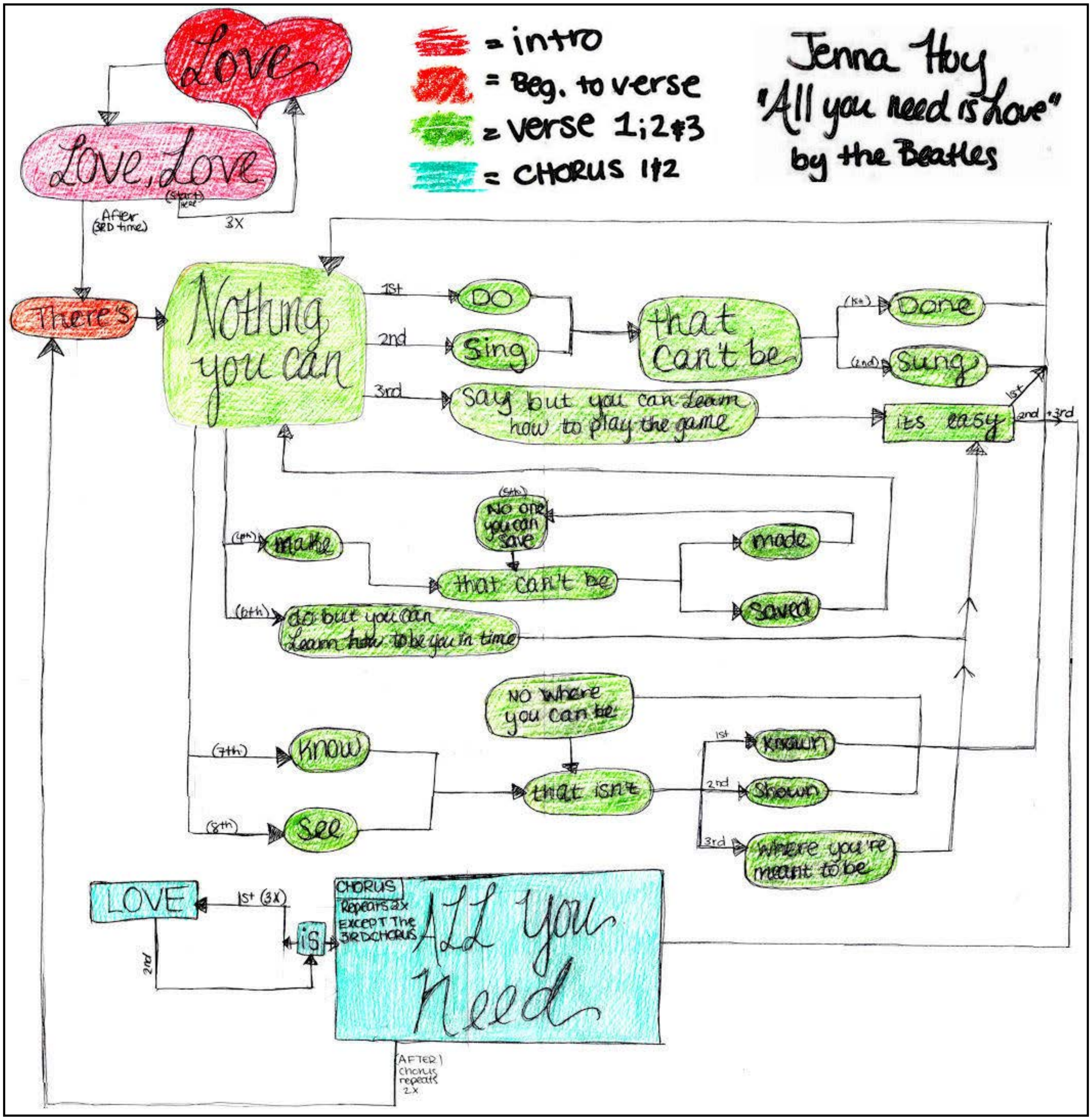
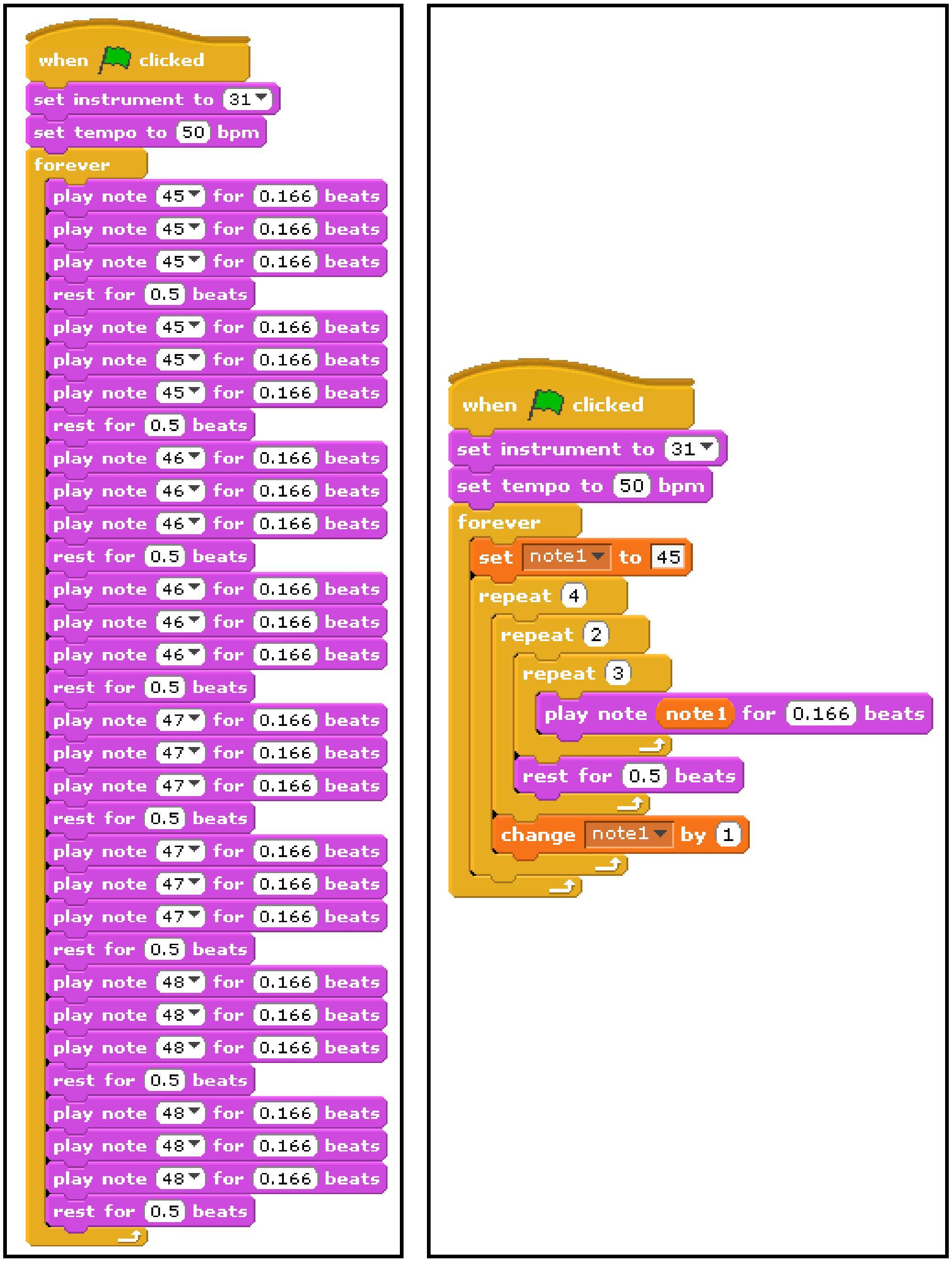


Figure 2. A song flowchart. [1]

Computing and Music (cont'd)



3a

```
when ⚑ clicked
set instrument to 31
set tempo to 50 bpm
forever
    play note 45 for 0.166 beats
    play note 45 for 0.166 beats
    play note 45 for 0.166 beats
    rest for 0.5 beats
    play note 45 for 0.166 beats
    play note 45 for 0.166 beats
    play note 45 for 0.166 beats
    rest for 0.5 beats
    play note 46 for 0.166 beats
    play note 46 for 0.166 beats
    play note 46 for 0.166 beats
    rest for 0.5 beats
    play note 46 for 0.166 beats
    play note 46 for 0.166 beats
    play note 46 for 0.166 beats
    rest for 0.5 beats
    play note 47 for 0.166 beats
    play note 47 for 0.166 beats
    play note 47 for 0.166 beats
    rest for 0.5 beats
    play note 47 for 0.166 beats
    play note 47 for 0.166 beats
    play note 47 for 0.166 beats
    rest for 0.5 beats
    play note 48 for 0.166 beats
    play note 48 for 0.166 beats
    play note 48 for 0.166 beats
    rest for 0.5 beats
    play note 48 for 0.166 beats
    play note 48 for 0.166 beats
    play note 48 for 0.166 beats
    rest for 0.5 beats
```

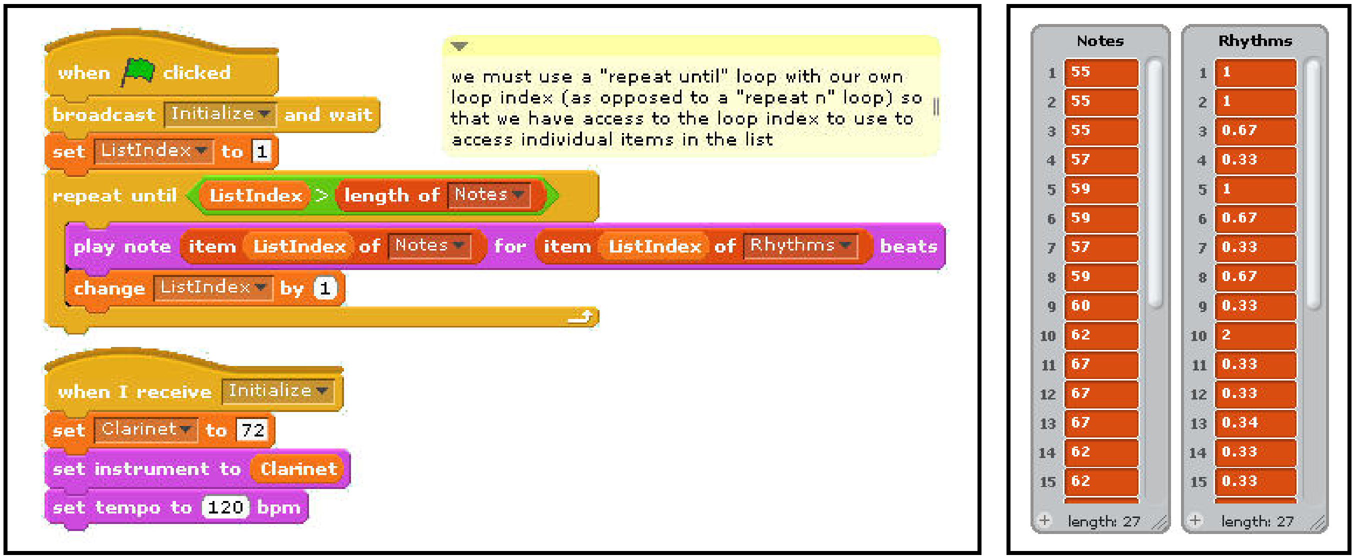
3b

```
when ⚑ clicked
set instrument to 31
set tempo to 50 bpm
forever
    set note1 to 45
    repeat 4
        repeat 2
            repeat 3
                play note note1 for 0.166 beats
            rest for 0.5 beats
        change note1 by 1
```

Figure 3. Two versions of Jimmy Page's *Kashmir* riff programmed in Scratch. [4]

Computing and Music (cont'd)

List and array data structures can be used to represent pitches and durations. Figure 4 shows an array (or indexed list) of MIDI note values paired with an array of note durations (in fractions of beats) that plays part of Row, Row, Row Your Boat. Using such structures, one can explore synchronization when the values are read by multiple threads with entrances staggered in time, resulting in the performance of a canon (or round).



```
when [flag] clicked
broadcast Initialize and wait
set ListIndex to 1
repeat until ListIndex > length of Notes
    play note item ListIndex of Notes for item ListIndex of Rhythms beats
    change ListIndex by 1

when I receive Initialize
set Clarinet to 72
set instrument to Clarinet
set tempo to 120 bpm
```

we must use a "repeat until" loop with our own loop index (as opposed to a "repeat n" loop) so that we have access to the loop index to use to access individual items in the list

4a

	Notes
1	55
2	55
3	55
4	57
5	59
6	59
7	57
8	59
9	60
10	62
11	67
12	67
13	67
14	62
15	62

length: 27

	Rhythms
1	1
2	1
3	0.67
4	0.33
5	1
6	0.67
7	0.33
8	0.67
9	0.33
10	2
11	0.33
12	0.33
13	0.34
14	0.33
15	0.33

length: 27

4b

Figure 4. Processing Scratch lists of notes and rhythms for *Row, Row, Row Your Boat.* [5]

References Cited

[1] Hoy, J. (2010). *Song flowchart for The Beatles' "All You Need is Love."* Created for a course assignment in "Sound Thinking."

[2] MIT Scratch Team (2009). *Scratch.* scratch.mit.edu accessed Dec. 21, 2009.

[3] Resnick, M., Maloney, J., Monroyhernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., & Kafai, Y. (2009). *Scratch Programming for All.* Comm. of the ACM 52(11):60-67.

[4] Ruthmann, S.A. (2009). *Computational Zeppelin.* scratch.mit.edu/projects/alexruthmann/ 736779 accessed Jan. 5, 2010.

[5] Ruthmann, S.A., & Heines, J.M. (2010). *Exploring Musical and Computational Thinking Through Musical Live Coding in Scratch.* Scratch@MIT. Cambridge, MA.

[6] Smith, D.E. (1997). *Repeats, Second Endings, and Codas.* www.scenicnewengland.net/ uitar/ notate/repeat.htm accessed Dec. 25, 2009.

Computer Science, Math, and Music: Concepts Covered in Scratch

Computer Science

- statements
- sequential control flow
- iteration
- conditional execution
- arithmetic operators
- Boolean operators
- objects
- concurrency
- variables
- lists
- event handling
- user interaction
- optimization

Math

- positive and negative numbers
- real numbers
- decimal notation
- built-in functions with inputs
- angles
- Cartesian coordinates
- trigonometric operators
- random numbers

Music

- pitch
- rhythm (as duration)
- melodic fragments
- modes and scales
- polyphony
- synchronization
- harmony
- composing
- performing
- transposition
- balance and dynamics
- digital audio (as sound files)
- MIDI notes and timbres
- tempo
- form and structural analysis

Additional Readings

Heines, J.M., Greher, G.R., Ruthmann, S.A., & Reilly, B. (2011). **Two Approaches to Interdisciplinary Computing+Music Courses.** *IEEE Computer* 44(12):25-32, December 2011.

http://teaching.cs.uml.edu/~heines/academic/papers/2011ieee/ieee2011paper-v33-forWebsite.pdf

The intersection of computing and music can enrich pedagogy in numerous ways, from low-level courses that use music to illustrate practical applications of computing concepts to high-level ones that use sophisticated computer algorithms to process audio signals. This paper explores the ground between these extremes by describing our experiences with two types of interdisciplinary courses. In the first, arts and computing students worked together to tackle a joint project even though they were taking independent courses. In the second, all students enrolled in the same course, but every class was taught by two professors: one from music and the other from computer science. This course was designed to teach computing and music together, rather than one in service to the other. This paper presents the philosophy and motivation behind these courses, describes some of the assignments students do in them, and shows examples of student work.

Ruthmann, S.A., Heines, J.M., Greher, G.R., Laidler, P., & Saulters, C. (2010). **Teaching Computational Thinking through Musical Live Coding in Scratch.** *41st ACM SIGCSE Technical Symposium on CS Education.* Milwaukee, WI, March 12, 2010.

http://teaching.cs.uml.edu/~heines/academic/papers/2010sigcse/SoundThinking-SIGCSE-2010.pdf

This paper discusses our ongoing experiences in developing an interdisciplinary general education course called Sound Thinking that is offered jointly by our Dept. of Computer Science and Dept. of Music. It focuses on the student outcomes we are trying to achieve and the projects we are using to help students realize those outcomes. It explains why we are moving from a web-based environment using HTML and JavaScript to Scratch and discusses the potential for Scratch's "musical live coding" capability to reinforce those concepts even more strongly.

Additional Readings (cont'd)

Maloney, J., Resnick, M., Rusk, N., Silverman, B., and Eastmond, E. (2010). **The Scratch Programming Language and Environment.** ACM Transactions on Computing Education 10(4). Article 16.

http://web.media.mit.edu/~jmaloney/papers/ScratchLangAndEnvironment.pdf

Scratch is a visual programming environment that allows users (primarily ages 8 to 16) to learn computer programming while working on personally meaningful projects such as animated stories and games. A key design goal of Scratch is to support self-directed learning through tinkering and collaboration with peers. This article explores how the Scratch programming language and environment support this goal.

Martin, F., Greher, G.R., Heines, J.M., Jeffers, J., Kim, H.J., Kuhn, S., Roehr, K., Selleck, N., Silka, L., and Yanco, H. (2009). **Joining Computing and the Arts at a Mid-Size University.** *2009 Conference of the Consortium for Computing Sciences in Colleges — Northeastern Region (CCSCNE 2009).* Plattsburgh, NY, April 24, 2009.

http://teaching.cs.uml.edu/~heines/academic/papers/2009ccscne/JoiningComputingAndArts.pdf

This paper describes two NSF-funded collaborations among faculty members in the Computer Science, Art, Music, and English departments at a public university in the Northeast USA. Our goal has been to create undergraduate learning opportunities across the university, focusing on connecting computer science to creative and expressive domains. In past publications, we have focused on student learning outcomes. This paper reports on the motivations, opportunities, and challenges for the faculty members involved.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., and Kafai, B. (2009). **Scratch: Programming for All.** *Communications of the ACM* 52(11):60-67.

http://web.media.mit.edu/~mres/papers/Scratch-CACM-final.pdf

"Digital fluency" should mean designing, creating, and remixing, not just browsing, chatting, and interacting. In this article we discuss the design principles that guided our development of Scratch and our strategies for making programming accessible and engaging for everyone.

Additional Readings (cont'd)

Heines, J.M., Greher, G.R., & Kuhn, S. (2009). **Music Performamatics: Interdisciplinary Interaction.** *40th ACM SIGCSE Technical Symposium on CS Education.* Chattanooga, TN, March 7, 2009.

http://teaching.cs.uml.edu/~heines/academic/papers/2009sigcse/fp119-heines.pdf

This paper describes how a graphical user interface (GUI) programming course offered by the Dept. of Computer Science (CS) was paired with a general teaching methods course offered by the Dept. of Music in an attempt to revitalize undergraduate CS education and to enrich the experiences of both sets of students. The paper provides details on the joint project done in these classes and the evaluation that assessed its effect on the curriculum, students, and professors.

Urban, J. (organizer), Heines, J.M., Fox, E.A., & Taylor, H.G. (2009). **Panel on Revitalized Undergraduate Computing Education.** *40th ACM SIGCSE Technical Symposium on CS Education.* Chattanooga, TN, March 5, 2009.

http://teaching.cs.uml.edu/~heines/academic/papers/2009sigcse/sigcse2009panel-JMH-accepted.pdf

There is an imbalance in the supply and demand for computing professionals that has generated shortages in meeting personnel needs within industry. A major program was developed by the U.S. National Science Foundation to encourage innovations in undergraduate computing education. There are a variety of new projects that are revitalizing undergraduate computing education. One approach to such revitalization is the introduction of interdisciplinary courses to expand the scope of computing education. The basic idea is to have students from various disciplines work together on computing projects to expand their educational horizons and make computing courses more appealing. This panel brings together research managers with educators who have developed and taught interdisciplinary courses with these goals in mind.

Additional Readings (cont'd)

Heines, J.M., Jeffers, J., & Kuhn, S. (2008). **Performamatics: Experiences With Connecting a Computer Science Course to a Design Arts Course.** *The International Journal of Learning* 15(2):9-16.

http://teaching.cs.uml.edu/~heines/academic/papers/2008learning/AsPublished-IntlJrnlLearning.pdf

This paper describes our efforts to stem the tide of declining CS enrollments by introducing innovations into our curriculum to give students more flexibility in course selection, especially in the freshman and sophomore years. Our approach is based on a partnership between the CS and Art, Music, and English departments in the area of exhibition and performance technologies.

In addition to describing our work, this paper provides the results of an evaluation conducted by an independent research. It reports on the impact this work has had on the CS and Art students and their respective projects, as well as on the professors and the way they teach their courses. It also describes steps that are being taken to improve the courses in the future.

Related Websites

Performamatics Website and Scratch Gallery and YouTube Channel

http://www.performamatics.org ➜ http://teaching.cs.uml.edu/~heines/TUES/
http://www.scratchmusic.org ➜ http://scratch.mit.edu/galleries/view/90913
http://www.youtube.com/performamatics

Scratch Projects by Performamatics People

http://scratch.mit.edu/users/alexruthmann (Music Prof. Alex Ruthmann)
http://scratch.mit.edu/users/drjay (CS Prof. Jesse Heines)
http://scratch.mit.edu/users/performamatics (additional collections)

Scratch Software

http://scratch.mit.edu (home page)
http://scratch.mit.edu/download (download page)
http://scratch.mit.edu/forums (discussion forums)

Scratch Resources for Teaching and Teachers

http://scratched.media.mit.edu (learn – share – connect for educators)

Scratch Project Galleries

http://scratch.mit.edu/channel/featured (featured projects)
http://scratch.mit.edu/galleries/browse/newest (members' personal galleries)

Scratch Information and Support

http://info.scratch.mit.edu/Support/Get_Started (getting started instructions)
http://info.scratch.mit.edu/sites/infoscratch.media.mit.edu/files/file/ScratchGettingStartedv14.pdf (Getting Started Guide)
http://info.scratch.mit.edu/Support/Reference_Guide_1.4 (Reference Guide)
http://info.scratch.mit.edu/Support (support page)
http://info.scratch.mit.edu/Video_Tutorials (video tutorials)
http://info.scratch.mit.edu/Support/Scratch_Cards (single-topic lessons)

Lifelong Kindergarten Group and Collaborators' Websites

http://llk.media.mit.edu (John Maloney and Mitchel Resnick)
http://teaching.cs.uml.edu (Jesse Heines)
http://www.alexruthmann.com (Alex Ruthmann)