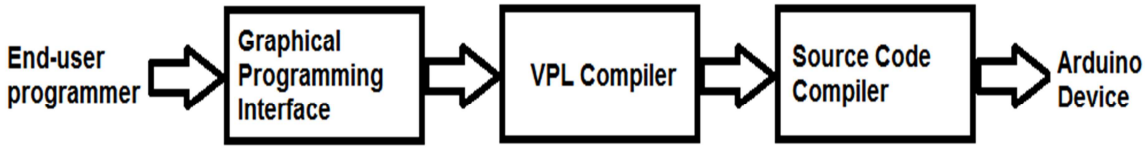
CodeSketch: A Visual Programming Language for Arduino

Eric Su and Bhavana Harrilal

Revised: 10 July, 2013

Project Description

The Arduino prototyping platform is open to a variety of users, enabling users to create usable hardware and software. However, most users of Arduino are novice programmers and struggle with learning how to program for the Arduino platform. The steep learning curve faced by novice Arduino users can be managed if they are introduced to a way of programming that was designed to their specification. This project aims to create a visual programming language for Arduino and will serve as a very helpful tool for creative users who use Arduino but are limited by their inability to code a program. The diagram below shows what the proposed system is intended to do.



1. Problem Statement

The problem we face is how to design and develop a system that would benefit users who want to create programs for Arduino, but lack sufficient programming skills to do so. The problem can be addressed in two ways: *1)* A top-down HCI approach using participatory design methods to design a user interface, and *2)* A bottom-up Graphical Implementation approach using Arduino's API to build visual interactive components based on functions and classes. They will both form an interactive system which aims to deliver a unique user experience as well as functional output capabilities. The problem is important because these two solutions make use of different research methods and design strategies. The two solutions can also be merged to create a more complete system and deliver a better user experience.

HCI design: Can participatory design effectively formulate a visual programming language for Arduino?

This solution to the problem focuses on who the users are and what they want for the system. Participatory design methods will be implemented to attain the kind of programming language that users want. HCI and participatory design have been known to create great results. Utilising technology probes along with ethnographic study to create a visual programming language. These design methods hold the user as an important component to formulating the solution to the problem. The problem addresses the inability of Arduino users to program for Arduino; this is a core function of the Arduino technology. By involving users to be part of the solution yields potential for a visual programming language to be developed for all users, spanning many backgrounds.

Graphical implementation: How to design and build a functional Visual Programming Language.

This solution to the problem focuses on building a system that will effectively translate visual data created by the user into functional Arduino C code. Although Visual Programming usually involves how visual approaches are applied to traditional programming languages (such as executable flowcharts), the aim for this project is to develop a simple visual approach to programming that can be improved by the user through usability testing.

2. Procedures and Methods

The design and development process will be an iterative one. We will follow a general software development model which consists of four basic activities: (1) Identifying user needs and establishing the software requirements. (2) User testing with interactive prototypes. (3) Analysing user data and evaluating the design. (4) Building and testing the final system.

HCI design approach:

The HCI approach to providing effective programming education involves the following steps and methods.

Technology probes

Technology probes used to gauge the knowledge and understanding of the users. Probes are used to gain information about the user group. Probes encourage users to reinterpret them and use them in different ways. Another goal of technology probes is its ability to raise participants' curiosity and attention.

Co-design
Participatory design techniques include the use of technology probes and user testing of prototypes. Due to users being unfamiliar with the technology, contextual inquiries can be made by interviewing users before a prototype is introduced. The initial technology probe will be prototyped by users, this probe allows for users to be part of a design process where they can communicate their preferences and weaknesses in the design. Through users input, the visual programming language's design will be formed.

Ethnographic methods help provide insight to users. First-hand investigations help to understand the particular group of users. This is vital to designing a visual programming language for these users. A more intrusive technique such as direct questioning; provides validation of findings, helps combine observations and gain additional information.

These techniques give understanding of the user's mental models in controlling the hardware. From the knowledge gained from the technology probes and through the design sessions done with the users in co-design; applying what is learnt from users to design the interface.

Prototypes
After gathering user understanding, a low-fidelity prototype can be produced. This will most likely be a paper-based prototype. This will be evaluated by the user group and feedback gathered will be used as key information for the next prototype i.e. high-fidelity prototype. The high-fidelity prototype will build from knowledge from previous designs done with the user group. This will be an interactive system that does not have functionality. This is then taken back to the users to gather feedback again.

Implementation
After the prototype phase, an implementation of the interface will be done. This will have functionality and follow from the design in the previous sessions and knowledge gathered from the user group. This implementation will address a subset of Arduino behaviours.

Evaluation and User Testing
The implementation is to be evaluated by the users through two ways; the first way being through task completion tests. Users will be asked to complete a set of behaviours using the software and success being determined on completion of task. The second way is through the users' evaluation of the overall experience, interviews with the users.

This solution only addresses a subset of Arduino behaviours. Accommodating for all cases would be a huge increase in scope. This would not be feasible given the time constraints.

Design sessions with users will be conducted with Iterative development. This allows for design and the application to be optimised without unnecessary additional prototypes.

Graphical Implementation approach:

Design

A bottom-up approach will be implemented to build interactive visual components using the current Arduino API as a reference. Functions and classes can be visualized as objects, and how these objects interact with each other will determine how the system behaves. Thus, looking at the code structure of current Arduino functions, it is not difficult to visualize them graphically. Now all that is needed is to build an interface that will allow these visual functions to be instantiated and interactive.

Tools

A development environment would be needed to build a Visual Programming Interface (VPI), Squeak is an open source programming language based on Smalltalk. It contains the necessary tools required to build a functional VPI and it is the development environment of choice for this project.

Evaluation

After the system has been built, it will undergo usability tests focused on Learnability, Efficiency, Memorability, and Accuracy. The overall user experience will also be recorded to improve other usability issues. The user test process is planned to take place as soon as ethical clearance has been approved. The aim is to gather at least 10 test users in order to gain statistically sound results for problem generalization.

3. Ethical, Professional and Legal Issues

Participatory design sessions

Using students who have no science or engineering background as users for participatory design sessions requires their consent and participation to this study. Ethical clearance will be needed to conduct these sessions.

User testing

The same users will be used for the different iterations of the visual programming languages design. These users will need ethical clearance as well.

4. Related Work

Arduino has been used to teach child friendly programming languages (Eisenberg et. al, 2009). Confliction on ideas about what exactly children should be able to learn such as recursion, procedure and variables, utilising graphical programming or robots that are programmable. It is shown that these topics are important to children learning of a programming language as it provides meaningful information to programming. One of the implementations was a paper based Arduino prototype. Ambient programming which is informal, moment-to-moment ways of programming was shown to be effective with children (Eisenberg et. al, 2009). There have also been many VPL design projects where similar systems were developed, such as *Kudo* [3].

TurTan uses Logo which is a graphical programming language to encourage its users to be creative and explore the potentials of programming (Gallardo et al., 2008). This example of a tangible programming language proved to be a positive with users, much of this positivity being credited to its tangibility attribute. Users found it to be enjoyable and easy to understand. Gallardo concluded that a tangible programming language yields good learning of basic programming language skills with its users.

One of the very first intuitive visual programming interface was designed in 1990 where Masahito Hirakawa described how an iconic programming system can be successfully implemented by utilizing visual information. It was an important concept as his paper described the fundamentals regarding interaction with machines using icons. The methodologies for this object oriented icon management framework is now a popular visualisation concept for designing VPLs. [5]

5. Anticipated Outcomes

Software Interface and User Experience:

The user must be able to anticipate the behaviour of the software from its visual properties. The software interface must provide adequate feedback to the user to achieve the purpose of allowing novice programmers to code for Arduino. The interface should help the users, who would normally be helpless with Arduino coding. Through user experience of the interface, coding for Arduino will become an understandable and novice programmer friendly process.

Program functionality and Code generation:

The system must be able to abstract the visual data created by the user. It must also perform the necessary control flow and dataflow compilations needed, as well as handle all user events and program exceptions. Finally, it must be able to generate the correct output program code for the Arduino device.

Effective Arduino education

Success of this project pends on the project being able to provide a visual programming language that effectively and understandably teaches novice programmers to be able to code core Arduino functionality with confidence.

6. Project Plan

Risks

Preferences gathered from design sessions are unrealistic to implement

Likelihood: Low

Severity: Medium - High

Information obtained at design sessions and through ethnographic methods yield preferences from users which are impossible to implement as a visual programming language. The language would need to be designed using visual programming language heuristics as opposed to be designed by users for users.

Mitigation: To avoid such a situation, an iterative development model has been implemented.

Users are not willing to fully participate

Likelihood: Medium

Severity: Medium

Users involved in the participatory design sessions and user testing after visual programming language development does not fully participate and give full feedback.

Mitigation: Sessions to be conducted in an environment which promotes users comfort and participation.

Damage to hardware equipment

Likelihood : Low

Severity : Medium

Hardware provided is damaged whilst implementing the application or testing application with users. Novice users to Arduino technology may mishandle the device

Mitigation: Ensure safe and proper usage of hardware devices, ensure backup a hardware device is always available.

Group member falls ill or encounters family emergency

Likelihood : Low

Severity : Low

For unforeseen reasons a team member is indisposed or forced to leave the course. Worst case scenario is that remaining team member(s) will have to complete as much of their allocated work by him(her)self.

Mitigation: Allocate work for each team member so that their research contents are independent of each other but are still closely related.

Milestones

Description	Days	Start Date	End Date
Literature review	17	04/08/13	04/30/13
Project proposal	9	05/01/13	05/13/13
Project proposal presentation	7	05/15/13	05/23/13
Initial Feasibility Demonstration	37	05/24/13	07/15/13
Background chapter	23	06/19/13	07/19/13
Design chapter	26	07/20/13	08/26/13
First implementation + write up	15	08/27/13	09/16/13
Final implementation + write up	8	09/16/13	09/25/13
Chapters on Implementation and Testing. Final implementation	11	09/16/13	09/30/13
Outline of complete report	5	10/01/13	10/07/13
Final Complete Draft of Report	10	10/08/13	10/21/13
Project Report Final Hand in	5	10/22/13	10/28/13
Poster due	3	10/29/13	10/31/13
Web page	2	11/01/13	11/04/13
Reflection paper	4	11/05/13	11/08/13
Final project presentation	4	11/09/13	11/14/13

Deliverables

Project Website	11/06
Project Proposal (Revised)	14/07
Horizontal Prototype	06/08
Final Implementation	30/09
User test results	05/10
Functional test results	12/10
Report draft	21/10
Final report	28/10
Project poster	31/10
Individual Reflection	08/11

Resource Required

- Arduino UNO starter kit (acquired two sets from project supervisor).
- Realistic user population for ethnographic research and usability testing. (Accessible on campus).
- System Development Software. (Dependent on application program requirements, most are available online).

Work Allocation

Eric is in charge of building a functional VPL using graphical implementation and testing its usability.

Bhavana is in charge of using HCI and implementing participatory design methods to attain what kind of programming language users want and designing a suitable interface.

References

1. Eisenberg, M., Elumeze, N., MacFerrin, M., & Buechley, L. (2009, June). Children's programming, reconsidered: settings, stuff, and surfaces. In *Proceedings of the 8th International Conference on Interaction Design and Children* (pp. 1-8). ACM.

2. Burnett, Margaret M. and Marla J. Baker, A Classification System for Visual Programming Languages, Journal of Visual Languages and Computing, 287-300, September 1994.

3. Matt MacLaurin, Kodu: end-user programming and design for games, Proceedings of the 4th International Conference on Foundations of Digital Games, April 26-30, 2009.
4. Gallardo, D., Julia, C. F., & Jorda, S. (2008, October). TurTan: A tangible programming language for creative exploration. In *Horizontal Interactive Human Computer Systems, 2008. TABLETOP 2008. 3rd IEEE International Workshop on* (pp. 89-92). IEEE.

5. Hirakawa, Masahito, Minoru Tanaka, Tadao Ichikawa, An Iconic Programming System, HI-VISUAL. In IEEE Trans. on Software Engineering, October, 1990.

Figure 1: Gantt Chart with milestone

	Task Name	Start Date	End Date	Duration
1	**A Visual Programming Language**	**04/08/13**	**11/11/13**	
2	Literature review	04/08/13	04/30/13	17
3	Project proposal	05/01/13	05/13/13	9
4	Project proposal presentation	05/15/13	05/23/13	7
5	Initial Feasibility Demonstration	05/24/13	07/15/13	37
6	Background chapter	06/19/13	07/19/13	23
7	Design chapter	07/20/13	08/26/13	26
8	first draft	08/27/13	08/05/13	
9	First implementation + write up	08/27/13	09/16/13	15
10	first draft	08/28/13	09/02/13	
11	second draft	09/03/13	09/09/13	
12	Final implementation + write up	09/16/13	09/25/13	8
13	first draft	09/17/13	09/20/13	
14	Chapters on Implementation and Testing. Final implementation	09/16/13	09/30/13	11
15	Outline of complete report	10/01/13	10/07/13	5
16	Final Complete Draft of Report	10/08/13	10/21/13	10
17	first draft	10/09/13	10/15/13	
18	Project Report Final Hand in	10/22/13	10/28/13	5
19	Poster due	10/29/13	10/31/13	3
20	Web page	11/01/13	11/04/13	2
21	Reflection paper	11/05/13	11/08/13	4
22	Final project presentation	11/09/13	11/14/13	4
23