Talking With Kids on Game Design, Computer Programming, and Taking Over the World With Dragons

Amanda Ochsner, University of Wisconsin-Madison
Gabriella Anton, Games+Learning+Society

Introduction

Computer programming is among the most in-demand skillsets of the 21st Century. Unfortunately, many K-12 schools are unable to offer classes on computer programming due to lack of resources or qualified staff and because of the strong emphasis placed on curricula that adhere to Common Core standards. After-school programs that focus on game design to engage students in introductory programming and computational thinking skills have emerged as a popular alternative to formal classes during the school day. This paper examines one of these after-school groups. Called the Cyberlearning Club, students at a rural middle school in the Midwest design games and practice programming using programs like *Scratch*, *Kodu*, and *Code Academy*. This research explores how middle school students in an informal after-school programming and game design club think about their game projects, their ability to learn programming, and their trajectories as future designers. In this paper, we report on the results of semi-structured qualitative interviews conducted with students and outline profiles of students who exhibit distinct goals and trajectories for learning programming.

Why young people should learn to program

In his 2014 State of the Union speech, President Obama argued that among the most pressing issues to address over the next few years are the trends of deepening inequality and stalled upward mobility (Associated Press, 2014). Hiring statistics suggest that computer-programming skills can position young people to attain that upward mobility. STEM and computing jobs almost universally dominate lists of most in-demand college majors and top-paying degrees. A Forbes article explains, “Not only are computer scientists and computer engineers the most sought after candidates on the market but–fittingly, understandably–they’re among the highest paid entry level hires that we know of” (Adams, 2013). The National Association of College Employers recently listed computer engineering and computer science in the top four for average starting salary across all college majors (National Association of College and Employers, 2014). Those skilled in computer science and programming are in high demand and there is no reason to expect that demand to decrease soon (Grover & Pea, 2013). Recent estimates project that by the year 2020, of the 9 million STEM jobs in the US (science, technology, engineering, and mathematics), more than half will require computer science skills (Lang et. al, 2013). Proponents of computer science education recognize that students will need computer science skills to thrive in the global information-driven society, yet the K-12 system has not been able to keep up. Many students, particularly in low-income and rural communities, do not have access to the classes they need to acquire programming skills (Lang et. al, 2013; Margolis et al., 2008).

Beyond providing youth with reliable career options, one of the popular arguments for supporting children in learning computing skills comes from Papert (1980). He claims that the iterative process of learning to program and debug code enables children to reflect on their learning process more critically. This constructionist approach to learning also emphasizes engaging learners in the process of constructing or creating artifacts that have personal and social significance (Papert, 1980; Kafai, 2006). Inspired by constructionism, the Computer Clubhouse after-school program (Kafai, 2009) was conceived as an informal learning space for at-risk youth to design their own media and gain valuable experiences with computing. The Computer Clubhouses were not specifically intended for teaching game design or programming, but many of them utilized *Scratch*, a programming tool for kids. The Cyberlearning Club that served as the setting for the research presented on in this paper is not a direct descendent of the Computer Clubhouses, but the Cyberlearning Club shares many of the Computer Clubhouse values while focusing more specifically on programming digital games.

Games matter

The prominence of video games in the lives of young people has made them one of the more popular pathways for teaching programming. Ninety-seven percent of youth report playing videogames on a regular basis (Lenhart et. al, 2008). There have been a number of after-school programs for engaging kids in game design and programming, including but not limited to: *ToonTalk* and The Playground Project (Kahn, 1996; Hayes & Games, 2008), the Computer Clubhouse’s use of *Scratch* (Maloney et al., 2008), the Girls Creating Games Program (Denner, Bean, & Martinez, 2009), the Imaginary World Campus using *Alice* (Adams, 2007), the use of *Storytelling Alice* with middle school girls (Kelleher, Pausch, & Kiesler, 2007), and the Studio K curriculum using *Kodu* (Anton et al., 2013). For a more comprehensive review of game design tools and their educational applications with kids through

2008, see Hayes and Games, 2008. Studies of youth designing and developing their own games have showed gains in a wealth of literacy and critical thinking practices (Peppler, Diazgranados, & Warschauer, 2010; Peppler & Kafai, 2007; Carbonaro et al., 2010), and suggest that these practices facilitate the development of computational thinking skills like logic, debugging, and algorithm design (Berland & Lee, 2011). Overwhelmingly, these tools and programs have been shown to provide an ever-evolving environment for forming creative representations, building expertise and mentorships, and learning without fear of harassment or judgment (Resnick et al., 2009).

Methods

The Cyberlearning Club

For this study, we conducted our research with a group called the Cyberlearning Club, an after-school technology club for engaging middle school students in game design and programming. Meeting at the school library in a rural town in the upper Midwest, about 15-20 students attend the Cyberlearning Club each Friday afternoon. The majority of students attending the club are male, a topic that we will discuss further when we discuss future directions and initiatives. In addition to the middle school students, the facilitator of the club—the lead Media Specialist at the school—has enlisted the help of four high school students who act as teaching assistants. The facilitator is not a proficient programmer herself, but feels passionately about supporting kids in learning what she considers to be increasingly important skills. She formed the club in the summer of 2013 to foster these practices and it was an immediate hit with students. At Cyberlearning Club, students are introduced to programs like *Scratch*, *Kodu*, and *Alice*, books like *Python for Kids*, and online recourses like *Code Academy.* For the uninitiated, *Scratch* (Maloney et. al, 2008), designed by the MIT Media Lab, is a programming language and tool designed for kids ages 8 to 16. While many *Scratch* projects are games, it is also designed for making interactive stories and animations. Designed by Microsoft, *Kodu* utilizes a visual programming language to teach game design and computational thinking. *Alice* is a programming environment first conceived at the University of Virginia and then further developed at Carnegie Mellon. *Alice* incorporates narrative aspects into the teaching of programming. Each of these tools are freely available and are utilized by the Cyberlearning Club with varying frequency. Code Academy is a free online tool where users can learn programming languages like html, JavaScript, and Python directly from their web browser. During meetings, students explore the provided resources and programs as their interests guide them, interacting independently, in pairs/trios, and sometimes in larger groups to learn collaboratively. The students who attend the club are primarily, though not exclusively, drawn to the club because of their interest in videogames and spending time with other kids with like interests.

The research outlined in this paper is not a report on use of a specific tool or curriculum. Rather, the students in the Cyberlearning Club use a variety of different software programs and resources. Some of the more advanced students have branched out to make use of complex tools to learn new programming languages. Invited by the club's facilitator to consult on ways to engage a wider student audience, our research team at UW-Madison began attending Cyberlearning Club meetings in the fall of 2013. Before suggesting specific interventions or curricula, we conducted ethnographic research to observe how students' learning trajectories naturally evolved. This paper describes interviews to determine what is personally meaningful to the students in the club. While we initially intended capture students' understanding of programming related to *Kodu*, the interviews evolved to capture a broader landscape of environments, goals, aspirations relating to learning game design and programming.

Interviews with Cyberlearners

Originally attending the club meetings to introduce students with new resources, we quickly discovered that the structure and interactions within the club were already—at least to some extent—supporting advanced practices and the development of a unique culture. Rather than disrupting the group with interventions, we took the opportunity to understand the interests and needs of the kids who attend the club, seeking to identify features within the space that support exploratory and advanced practices among the participants. To gain a deeper understanding of the individual students in the club, we conducted semi-structured interviews with students who regularly attend the Cyberlearning Club. Through these interviews we attempt to capture the students' conceptual understanding and personal landscape relating to the club, programming, and game design activities. Questions in the interview protocol encourage the students to talk about the games they have designed and to describe their experiences with and opinions of programming tools and resources. We also asked about how they enlist friends and family as resources, what their personal goals are, and how/whether they think of technology and programming as valuable workplace and career skills.

Figure 1: Five boys collaborating on an *Alice* game at the Cyberlearning Club.

Currently, we have conducted interviews with eight of the club's most dedicated students, all boys. The interviews ranged from 10 to 45 minutes depending on the depth of the boys' responses. After completion of the interviews, we first developed an emergent coding scheme and used those codes to identify patterns and themes in the students' responses in order to identify emerging learning trajectories or practices. From there we decided to focus on four different cases for this paper—twins with a strong interest in game design and intermediate programming experience, and two other boys with very different profiles. These four cases demonstrate the diversity of attitudes, experiences, and skill levels among the students in the Cyberlearning Club, hinting at the different types of learning trajectories that the students are engaged in. Below, we discuss each of the boys and make recommendations about what learning supports and resources they would benefit from based on their current programming abilities and interests.

Cyberlearner Profiles

Adam (Age 11): Choosing the power of dragons over the power of Python.

Programming in *Scratch*: "I don't know what 15 degrees is, so it's kind of hard to program with it."

Tending to shy away from overly intimidating challenges, Adam seems happiest when engaged in activities that he finds to be both fun and easy. He is more content to stick with *Kodu* than many of the other guys, not expressing interest in reading books on web programming and advanced languages or venturing into learning text-based languages. While many of the boys seem to find *Kodu* and *Scratch* to comparably easy to learn, Adam does not fully understand the programming concepts in *Scratch*, such as rotating objects a certain number of degrees to guide a character on the screen. He notes, "It don't know what 15 degrees is so it's kind of hard to program with it [Scratch]." However, Adam is quite proficient with *Kodu*, describing that he felt comfortable making games within a day and now adds increasingly more Kodus [moveable objects] and code into his games. This same tendency shows up when Adam talks about school. He claims that spelling is his favorite subject in school "because it's easy and fun"—the same terms he uses to describe how he feels about *Kodu*. Adam's hesitancy to try out new tools or learn new programming languages seems at least in part because he does not see programming as essential to his future career plans. Hoping to be an artist or geneticist, he expresses that programming probably wouldn't help much with these professional paths. As a geneticist, "I could bring back a dragon and then take over the world," he says. For Adam, proficiency in Python seems less essential than the power of a fire-breathing dragon in a world takeover.

Mac (Age 11): Programming is just for game design. And building a time machine.

What do programmers do? "Fix bugs, program new things, get stressed out mostly." Having discovered *Scratch* at the age of seven with his twin brother, Mac shows a high level of engagement with game making and programming. He is a self-motivated learner and frequently takes the initiative to expand his skillset. He describes having

made roughly 50 games in *Scratch* over the years. While he occasionally makes a game in *Kodu* the tool is a bit simple for his tastes and typically only takes him "about five minutes." He juxtaposes programs like *Kodu* with real programming, which "has if-statements and functions and stuff." Mac has recently begun learning Python and JavaScript with his brother. He says he personally prefers books like *Python for Kids* and *Invent Your Own Computer Games with Python* to online tools like Code Academy, though he notes that different people have different learning styles. Mac and his brother both aspire to be independent game developers. After having just watched *Indie Game the Movie,* he says he got the impression that programming a game professionally is a stressful job, but that he is not deterred. Games are the primary motivator for Mac to learn programming. He has little interest in making websites because that is "not as fun as making games" and says he can't imagine using programming for anything but game development except maybe "building a time machine." He adds, "Possibly. If it's possible."

Mitch (Age 11): Preferring *actual* programming to *Kodu* and *Scratch*.

"I like doing the *actual* programming more than like where they do *Kodu* or *Scratch*."

Mitch is very articulate about the differences between the various game design and programing tools. While he recognizes that *Kodu* provides new learners with a structure, users of *Scratch* learn more of what he calls programming "fundamentals." Mitch and his brother Mac have thrived under early exposure to game design and programming tools. The boys have been making games for four years and they both seem to be in a critical learning period now where they are branching out to learn more complex programming languages like Python and JavaScript. Unlike his brother who never plans to make a website, Mitch admits he may consider making one, but "only to put some games on it." It seems like they are on the threshold of a new level of expertise with programming, starting to be able to compare and critique languages and tools they encounter. The boys enjoy a number of benefits from their home lives. While their parents are not programmers, they both have jobs in science fields and seem to promote a tech-savvy lifestyle. The twins and their sister share a desktop computer at home and each member of the family has their own laptop. In addition to spending about an hour-and-a-half each day collaboratively studying programming, the boys enjoy the added benefits of support from the Cyberlearning Club. Here they learn about new tools like Code Academy and gain access to some of the books on programming that they use. Furthermore, the club adds a more playful-minded, social facet to their current interaction with programming. At this stage in their programming trajectory, it seems that the twins would really benefit from some more advanced mentoring and more formal education on programming. It is unfortunate that there are no formal computer science education courses for them to take at their school.

Zeek (Age 13): You fail and people laugh, but you just have to deal with it as a programmer.

"Because, just, programming is fun. It's me, you know? It's not something I can go, "okay, I've learned this to my full extent. I can't go past this point." I like to break through, keep going. ...I want to become a programmer. I want to do things like that my whole life."

Zeek is very passionate and thoughtful in his understanding of programming. He describes being around technology his whole life, listing off the many game systems he has had over the years, and explaining that he has been interested in coding for quite some time. He notes that watching the movie *The Matrix* was his first exposure to the idea of programming and it stuck with him. Zeek seems to be especially well informed about what programmers and game developers do: he talks a lot about how programmers incorporate feedback and criticism, showing that he understands programming as an iterative process that requires constant debugging and testing. However, he is a relatively new programmer, only having about six months of experience starting from when he joined the Cyberlearning Club. He primarily uses *Kodu*, and mostly during his time within the club, though he admits to trying and disliking *Scratch*. Zeek seems especially sensitive to the feedback and criticism inherent in design. He describes that his dream of designing games will be hard because "you always have someone who goes, 'this is wrong, this is wrong, this is wrong – change it." At another point he says, "When you try, you fail and people laugh at you, which is the thing you have to live through as a programmer." Zeek admits to dealing with failure and negative feedback in his life, describing a tendency to poor grades and struggles with being teased. However, he recognizes that these experiences have primed him to be more successful as a programmer, dealing with feedback.

What Zeek would most benefit from is someone to encourage him to be persistent and follow through with things. Seeming to equate tasks that take a long time with being difficult, he seems to lose interest in projects very easily— he describes abandoned games, unfinished writing projects, giving up on figuring out how to make things work on his computer. When asked about his knowledge of programming languages he suggests that he knows JavaScript, but when probed on it he reveals that he only spent about "ten minutes" on Code Academy, but that he is "planning on learning more." He knows a good amount *about* JavaScript, describing it as the base for many other programming languages, but he lacks experience in learning how to program *with* the language. It seems as

though Zeek has a great grasp on his end goals but may not have the support or motivation to reach those goals. Coming from a working class family and having a dad who he describes as not being good with computers, Zeek would benefit immensely from a class on programming at the high school level.

	Adam	Mac	Mitch	Zeek
Enjoys Kodu	Yes	Too easy	Too easy	Yes
Enjoys Scratch	No, too challenging	Yes	Yes, but prefers Python	Not interested
Learning other prog. languages	No, not interested	Python, JavaScript	Python, JavaScript	A bit of JavaScript
Prefers to be challenged	No	Yes	Yes	Depends on context
Likes school	Yes, especially easy classes	Yes, when it's not too easy	Yes, when it's not too easy	No, struggles with his grades
Wants to be a game developer	No	Yes	Yes	Yes
Wants to be a programmer	No	Yes	Yes	Yes
Resources needed for next future steps	Gentle guidance on increasing his code complexity	Mentoring & access to more advanced peers	Mentoring & access to more advanced peers	A formal intro to CS class, encouragement on persistence

Table 1: Overview of the Cyberlearners' likes, dislikes, and future plans.

Analysis

As we completed the first round of interviews with the students in the Cyberlearning Club, we were excited and impressed by the creativity, enthusiasm, and drive evident in these self-directed, engaged students. They *love* games and they *love* designing them. We cannot help but feel inspired listening to these young students talk about how are learning complex programming languages like Python on their own and sharing their ambitious plans about being indie game designers and making only the best games which no one will criticize because they included only the best design elements. Of course, we also anticipate road blocks that the students are likely to encounter along their way, and want to see support structures in place to help them get past upcoming barriers.

Cyberlearning benefits

Though the boys we interviewed were ranging in their interests and experience with programming, it was evident that the structure and culture within the club was significant in facilitating interest in programming for these students. The club coordinator provides an open, creative space and myriad resources to tempt students into programming. Though not a programmer herself, she's designed a self-supporting culture, in which students gain expertise and support each other as they interact with the new resources she supplies. For each boy we interviewed, the club provided a different, meaningful experience for their interactions with programming and game design. For Adam, the club provides him with motivated, experienced programmers who he partners with to design games in *Kodu.* While he continues to gain experience and skills with programming, he primarily wants to design games rather than program them. For the twins, the club offers a space in which to interact with more dedicated programmers. Their time within the club is generally spent working with one or two other comparably experienced club members making games. It is a space for them to tackle challenging problems in a social, fun way. Finally, for Zeek, the club gives him exposure to the ways in which other students are working towards their programming goals in addition to the many resources to get their. Zeek seems to gain the most from the tangible and social resources offered within the club, mentioning that he was exposed to *Kodu* and *CodeAcademy* through the club and only started working with *Scratch* when it was reintroduced within that space.

Cyberlearning needs

Many of the students in the Cyberlearning Club are at critical learning points, ready to leverage their enthusiasm and curiosity toward more substantive knowledge about computer programming and game design. These students are lucky to have a club coordinator who devotes her time to the club and seeks out resources like researchers at the University of Wisconsin-Madison who can help support her by recommending resources and curricula, or with writing grants to acquire better technology for the school library. However, the Cyberlearning Club coordinator is only one person and has a limited knowledge of game design and programming herself, and as we got to know the students in the club better, we realize that some of these students have needs that go beyond the current scope of the club. Many of them could benefit from classes in the formal school curricula, expert peers and mentors, or more focused scaffolding and encouragement.

Adam is content with things as they are, liking the tools he already uses and not wanting to branch out to learn more complex programming languages. Perhaps that's enough, as this is an interest-driven after-school club, but we wonder what he might do with a mentor to gently push him to challenge himself and expand his abilities. Mac and Mitch have tools like Code Academy and a few books, but their knowledge seems to be quickly exceeding the current club activities. We believe they could benefit from access to a more advanced group of peers, people who have made more complex programs and games who can offer more personalized instruction than a book or an on-rails web resource. The twins seem almost ready for a high school or college level introductory course on programming. Zeek has the same ambitious drive toward game design and passion as Mac and Mitch, but like Adam, he gets deterred when he encounters a really tough challenge. Many of his projects are left unfinished and he seems to lack persistence in his efforts to learn more complex programming skills. A structured class in his middle school, with specific objectives and assignment due dates might help him keep on track with his learning trajectory. Fortunately for Zeek, the club coordinator is helping other teachers to incorporate programming activities to better engage him, as he is dangerously close to disengaging from his core classes. Unfortunately, the school district that these students live in does not have computer science or programming classes at any level, inhibiting the progress they will be able to make until they reach the next steps in their education.

Diversifying Cyberlearning

One of the reasons that the Cyberlearning Club facilitator reached out to researchers at UW-Madison was that she was concerned that the club was not attracting female students. Girls occasionally come to the club meetings, but they tend not to attend regularly or engage as deeply in the club's activities as the majority of the boys. In late 2013, the researchers conducted focus group testing with middle and high school girls in the school district in order to ascertain how the club might foster a culture more interesting to the girls. As a result, the Cyberlearning Club facilitator and UW-Madison researchers will be piloting a summer course in wearable computing, connecting fabric and crafts design with electronics and computational literacies. Additionally, the facilitator has been working to specifically invite and encourage female students with an interest in game design and programming, a tactic that has been shown to be effective in recruiting girls to programming classes (Goode, 2008).

Conclusion

Our interactions with the Cyberlearning Club illuminate several meaningful factors in understanding programming education for youth. While formal learning environments provide a structured base of fundamentals, informal spaces, especially those that offer a wealth of resources and environments in which to explore programming, may prove more engaging and effective for a wider audience of learners. For the students in the Cyberlearning Club, the variety of tools and resources offered is key to keeping them interested, confident, and creating. Furthermore, the social environment within the club supports students gaining expertise in a variety of areas, with students focusing on design, game programming, website development, and even art. This culture is extremely important for supporting a creative environment but also in introducing students to new practices. Consequently, the learning goals and practices for each boy we interview may become more meaningful as we continue to conduct our interviews and study of the Cyberlearning Club. We summarized the each boy's programming landscape, and patterns throughout these may develop as we interview a larger number of club participants. As we continue to work with the Cyberlearning Club in the upcoming year, we plan to continue to introduce new tools and programs to the students as a way to support continued growth and learning for the kids in the club. While we hope that the district offers a programming course in the formal curricula, until that happens, the informal programming environment is providing a space to support students' engaged learning practices.

References

Adams, J. C. (2007). Alice, middle schoolers, & the imaginary world camps. *ACM SIGCSE Bulletin*, *39*(1), 307–311.

Adams, S. (2013, Jan. 24). The college degrees with the highest starting salaries. Forbes. Retrieved from: http://www.forbes.com/sites/susanadams/2013/01/24/college-degrees-with-the-highest-starting-salaries-2/

Anton, G., Harris, S., Ochsner, A., Salmon, A., Rothschild, M., Berland, M., & Squire, K. (2013). Patterns of play: Understanding computational thinking through game design. In Williams, C., Ochsner, A., Dietmeier, J., & Steinkuehler, C. (Eds.) *Proceedings of the Games, Learning, and Society Conference: Vol. 3.* Pittsburgh, PA: ETC Press.

Associated Press. (2014, Jan. 28). Obama to say in State of the Union, 'Inequality has deepened. Upward mobility has stalled' [Press release]. Retrieved from: http://blog.apastyle.org/apastyle/2010/09/how-to-cite-a-press-release-in-apa-style.html

Denner, J., Bean, S., & Martinez, J. (2009). The girls game company: Engaging Latina girls in information technology. *Afterschool Matters*, *8*, 26–35.

Goode, J. (2008). Increasing diversity in K-12 computer science: Strategies from the field. In Proceedings of the 39th SIGCSE technical symposium on Computer science education.

Grover, S., & Pea, R. (2013). Computational thinking in K-12: A review of the state of the field. *Educational Researcher*, *42*(1), 38–43. doi:10.3102/0013189X12463051

Hayes, E. R., & Games, I. a. (2008). Making computer games and design thinking: A review of current software and strategies. *Games and Culture*, *3*(3-4), 309–332.

Kafai, Y. B. (2006). Constructionism. In R. K. Sawyer (Ed.), *The Cambridge Handbook of the Learning Sciences* (pp. 35–46). Cambridge, MA: Cambridge University Press.

Kafai, Y. B., Peppler, K. A., & Chapman, R. N. (2009). *The Computer Clubhouse: Constructionism and creativity in youth communities.* New York: Teachers College Press.

Kelleher, C., Pausch, R., & Kiesler, S. (2007). Storytelling alice motivates middle school girls to learn computer programming. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '07*, 1455. doi:10.1145/1240624.1240844

Khan, K. (1996). ToonTalk--An animated programming environment for children. *Journal of Visual Languages and Computing*, 7, 197-217.

Steinkuehler, C., & King, E. (2009). Digital literacies for the disengaged: Creating after school contexts to support boys' game-based literacy skills. *On the Horizon*, *17*(1), 47–59.

Lang, K., Galanos, R., Goode, J., Steehorn, D., Trees, F., Phillips, P., & Stephenson, C. (2013). Bugs in the system: Computer science teacher certification in the U.S. *Immunological Reviews,* 255. (Vol. 255, pp. 256–74). New York. doi:10.1111/imr.12092

Lenhart, A., Kahne, J., Middaugh, E., Macgill, A. R., Evans, C., & Vitak, J. (2008). *Teens, video games, and civics*. Washington DC.

Margolis, J., Estrella, R., Goode, J., Holme, J. J., & Nao, K. (2008). Stuck in the shallow end: Education, race, and computing. Cambridge, MA: MIT Press.

Maloney, J., Peppler, K., Kafai, Y. B., Resnick, M., & Rusk, N. (2008). Programming by choice: Urban youth learning programming with Scratch. In *Proceedings of the 39th SIGCSE technical symposium on Computer science education* (pp. 367–371).

National Association of Colleges and Employers. (2014). Engineering dominates list of top-paid majors for class of 2013 grads. Retrieved from: https://www.naceweb.org/about-us/press/engineering-dominates-top-paid-list-class-2013.aspx

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York: Basic Books.

Resnick, M., Maloney, J., Monroy-Hernandez, A., Rusk, N., Eastmond, E., Brennan, K., … Kafai, Y. (2009). Scratch: Programming for all. *Communications of the ACM*, *52*(11).