Advancing Mobile App Development and Generative AI Education through MIT App Inventor

David Y.J. Kim[1], Anqi Zhou[1], Yasuhiro Sudo[2], Kosuke Takano[2]
[1]Massachusetts Institute of Technology
[2]Kanagawa Institute of Technology

ABSTRACT

This study evaluates MIT App Inventor's efficacy in teaching Computational Thinking and Generative AI to a diverse international student cohort. Centered on a five-day workshop, we focus on teaching students to create mobile applications that harness the power of Generative AI. The use of App Inventor, known for its block-based coding, made programming more accessible and engaging, particularly for novices. Participants' feedback indicated a significant shift in their perception of programming. They reported increased confidence and motivation to integrate these skills into daily life. The student-developed applications during the workshop demonstrated practical applications of their learning, aligning with the concept of Computational Action – the application of computational thinking in real-world scenarios. The research highlights App Inventor's strengths as an educational tool and suggests enhancements for its interface and features. It sheds light on the tool's role in encouraging technological proficiency and creativity among global student populations.

KEYWORDS

Computational Thinking, Generative AI, MIT App Inventor, Computational Action

1. INTRODUCTION & BACKGROUND

Technology exerts a profound transformative impact on society, altering our lives on an unparalleled scale. Rather than merely being passive consumers, we envisage a future where individuals actively contribute to technological progress. However, the path to this vision of democratizing technology is frequently hindered by its intricate nature. Numerous efforts from researchers, practitioners, and educators have been made to address this challenge through various approaches. Among these is the implementation of block-based programming, exemplified by platforms like Scratch (Maloney et al., 2010). Another example is MIT App Inventor (Wolber, Abelson, and Friedman, 2015), which enables anyone to craft unique applications for smartphones and tablets. Users of App Inventor develop apps by arranging and connecting geometric, tinker-toy-like blocks through a drag-and-drop interface on their browser screen. The platform then translates these block assemblies into executable apps compatible with Android or iOS devices.

MIT App Inventor has demonstrated its effectiveness in sparking interest among numerous students in creating mobile applications (Perdikuri, 2014), even for young middle school students (Grover and Pea, 2013). However, most of these curricula have been evaluated primarily with students in the United States, leaving it uncertain whether they hold the same effectiveness for students abroad.

This paper explores this question through an intensive one-week workshop designed for Japanese university students, many of whom had no prior experience in coding or even using tools like App Inventor. The selection of participants was notably diverse, encompassing students from various academic backgrounds, thus presenting a unique opportunity to evaluate the efficacy of block-based coding as a universal tool outside of the States for imparting computational thinking education. This paper details the structure of the workshop, the pedagogical strategies employed, and the outcomes observed, shedding light on the transformative potential of block-based coding in education abroad.

1.1. Computational Thinking Education

Computational Thinking (CT), popularized by Jeannette Wing in her seminal 2006 paper (Wing, 2006), represents a fundamental paradigm in modern education, emerging as a critical skill set akin to reading, writing, and arithmetic. At its core, Computational Thinking involves problem-solving methods and techniques that draw from the domain of computer science, yet its application transcends far beyond the confines of programming. The implications of CT education are profound. By fostering computational thinking skills, educators are preparing students for a future where digital literacy is paramount. Moreover, CT education promotes problem-solving skills, logical reasoning, and creativity, which are valuable in various fields (Kong and Abelson, 2022).

1.2. Computational Action

Computational action represents the practical implementation of computational thinking concepts. Computational action is characterized by its application in real-world scenarios, iterative process of refinement, emphasis on collaboration and communication, outcome-oriented nature, and direct engagement with technology (Tissenbaum, Sheldon, and Abelson, 2019). It goes beyond theoretical understanding, involving the creation of tangible solutions like software applications, algorithms, or systems. This transition is critical, particularly in educational contexts, as it enables learners to apply abstract principles to real-world tasks, thereby solidifying their understanding and enhancing their problem-solving skills. This practical approach is essential in the educational process, helping students not only reinforce their computational thinking but also gain confidence and skills in technological innovation and creation (Du et al., 2023).

1.3. MIT App Inventor

The App Inventor's design philosophy is centered around democratizing software development by enabling users of varying programming expertise to create mobile

applications. It utilizes a block-based programming language (Patton, Tissenbaum, Harunani, 2019).

The block-based language allows users to "snap" together command blocks to create a program, eliminating the need for syntax and reducing common coding errors. In the App Inventor environment, the app creation process is divided into two main parts: the Designer and the Blocks Editor.

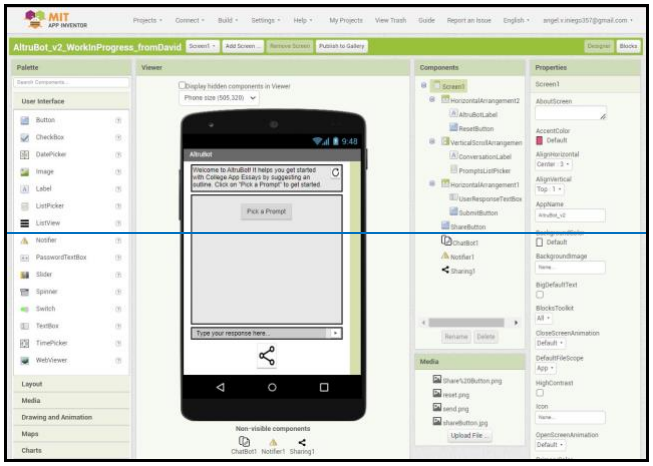

Figure 1. Designer section of App Inventor

Designer (Figure 1): The Designer is used to build the layout of the application. Users can drag and drop components, such as buttons, images, or sliders, onto a visual representation of a phone screen. This way, users can build the app's user interface without writing a single line of code.

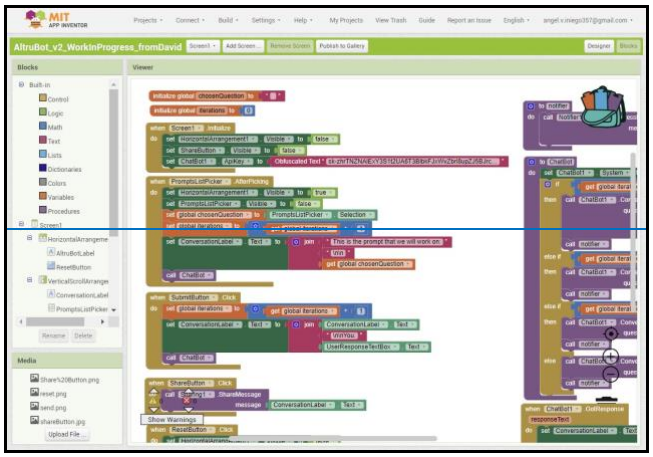

Figure 2. Blocks Editor in App Inventor

Blocks Editor (Figure 2): The Blocks Editor is where the app's behavior is defined. Users can select from a pallet of blocks that represent different functions or variables, and drag them into the workspace. By connecting different blocks, users define the app's responses to user inputs or other events.

1.4. Educating Students about Generative AI

Generative AI refers to artificial intelligence systems that can generate new content, ideas, or data that are novel and not merely a reshuffling of existing information. This field has seen a significant surge in both interest and development in recent years, primarily due to advances in machine learning and neural network technologies (OpenAI, 2016). Generative AI holds the capacity to profoundly transform numerous facets of human society, bringing with it a spectrum of both positive and negative impacts. It is crucial for an increasing number of people to not only become aware of this transformative technology but also to possess the skills and understanding necessary to integrate it into their daily lives effectively. The importance of educating about these technologies becomes increasingly critical. Education in generative AI not only involves understanding the technical workings of these systems but also encompasses a broader comprehension of their ethical, societal, and practical implications (Sharples, 2023). Recently, the MIT App Inventor acquired an innovative addition to its platform - a chatbot/imagebot component. This new feature abstracted the integration of advanced generative AI models, like OpenAI's ChatGPT and Dall-E (Shi et al. 2020), into mobile applications built with App Inventor. With just a few programming blocks, developers can now tap into the power of these AI models, opening up a wide range of possibilities for app functionality.

An assessment of the workshop's effectiveness was primarily based on the feedback provided by the students and the evaluation of the projects presented on the final day. These projects served as a practical indicator of the students' grasp of the concepts and skills imparted during the workshop.

2. METHOD

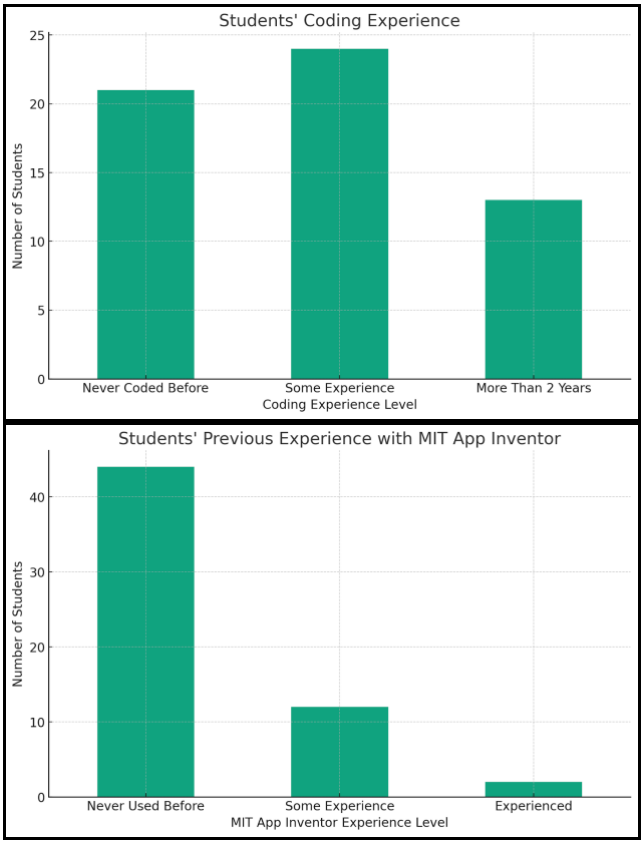

Figure 3. Experience in coding and in App Inventor

The workshop titled “Harnessing Generative AI in Mobile Application Development” was conducted at the

Kanagawa Institute of Technology. It spanned five days, with each session lasting three hours. The participant group comprised 23 in-person students at the Kanagawa Institute of Technology and 60 to 100 remote students, primarily students from Malaysia and Indonesia.

The workshop's primary objective was to introduce students, many of whom had minimal to no experience in coding, as you can see in Figure 3, to the basics of mobile application development using App Inventor. A special emphasis was placed on the integration of generative AI components, showcasing the potential of block-based coding in teaching computational thinking and practical application skills.

<table>
<tr><th></th><th>DAY 1</th><th>DAY 2</th><th>DAY 3</th><th>DAY 4</th><th>DAY 5</th></tr>
<tr><td>0 min ~ 30 mins</td><td rowspan="2">Introduction to MIT App Inventor</td><td>Talk2Me Tutorial</td><td rowspan="2">Digital Doodle Tutorial</td><td rowspan="6">Generative AI class: Learning Intutiion and the design of generative AI neural networks</td><td rowspan="6">Student Presentation</td></tr>
<tr><td>30 mins ~ 1 hour</td><td rowspan="2">TodoList Tutorial</td></tr>
<tr><td>1 hour ~ 1 hour 30 mins</td><td>Hello Purr Tutorial</td><td>Introduction to Dall-E</td></tr>
<tr><td>1 hour 30 mins ~ 2 hours</td><td rowspan="2">MoleMash Tutorial</td><td>Introduction to ChatGPT</td><td rowspan="3">Dall-E with App Inventor</td></tr>
<tr><td>2 hours ~ 2 hours 30 mins</td><td rowspan="2">ChatGPT with App Inventor</td></tr>
<tr><td>2 hours 30 mins ~ 3 hours</td><td>Extra Questions</td></tr>
</table>

Figure 4. Curriculum of the workshop

As shown in Figure 4, the first three days of the workshop were dedicated to hands-on tutorials in App Inventor, focusing particularly on utilizing its new chatbot and imagebot components. These sessions were designed to provide step-by-step guidance, enabling students to become familiar with block-based coding and the essentials of mobile app creation.

On the fourth day, the workshop shifted its focus to the foundational concepts of generative AI. This segment included both theoretical and practical elements, aiming to enhance students' understanding of how generative AI operates and how it can be incorporated into mobile applications. This was particularly relevant given the use of AI components in the App Inventor activities.

The workshop culminated on the fifth day with student presentations. Each participant or group was tasked with presenting a simple mobile application they had developed using App Inventor, which incorporated elements of generative AI. This session provided an opportunity for students to demonstrate their understanding and creative application of the skills acquired during the workshop.

An assessment of the workshop's effectiveness was primarily based on the feedback provided by the students and the evaluation of the projects presented on the final day. These projects served as a practical indicator of the student's grasp of the concepts and skills imparted during the workshop. The IRB approval was obtained from Kanagawa Institute of Technology, ensuring that all research methods, participant recruitment, and data handling procedures complied with ethical standards and regulatory guidelines.

3. RESULTS

The students' final presentations were particularly impressive, considering that most of them had never heard of MIT App Inventor before the workshop and for many, English was not their primary language. Despite these challenges, they showcased remarkable ingenuity in integrating generative AI with mobile application development into their everyday lives. For instance, highlighted in Figure 5, a standout project was an app developed by a student using a chatbot to determine a random ‘lucky color’. This color then inspired the generation of images of items in that hue, along with information on where to find these items. The student noted, “This app helps me choose the color of my shirt each day,” brilliantly demonstrating the practical use of chatbot and imagebot functionalities. This example underscores the students' capacity to creatively utilize AI tools, significantly enhancing their daily routines and decision-making processes, all achieved within the context of navigating a new programming language and working in a non-native language.

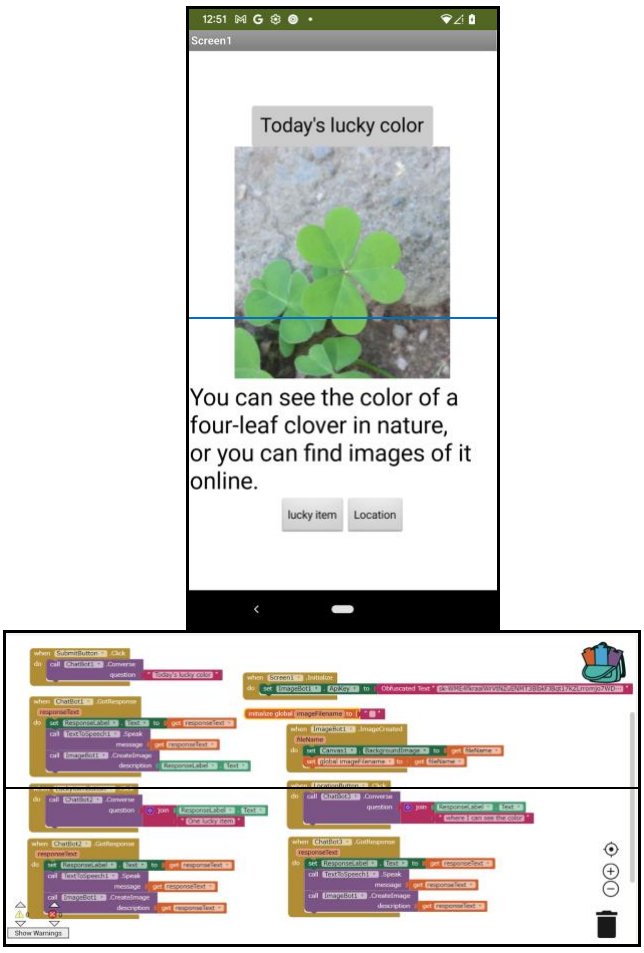

Figure 5. Example of an app a student created

We also show both qualitative and quantitative results based on the student survey after the workshop ended.

3.1. Did students become more confident in programming?

The five-day workshop utilizing MIT App Inventor for mobile application development significantly influenced the participants' views on programming. Attendees who already had an interest in programming noted less change in their perspective. In contrast, for many others, the workshop was an eye-opening experience, revealing its simplicity and accessibility.

The workshop boosted the participants' confidence in programming. Individuals who previously found programming challenging or had struggled with mobile app development reported that the workshop rendered these tasks more manageable and enjoyable. Someone mentioned that “Before diving into programming, I was overwhelmed and doubted my ability to master it. However, once I began to learn, my confidence grew, sparking a genuine interest in application development.”

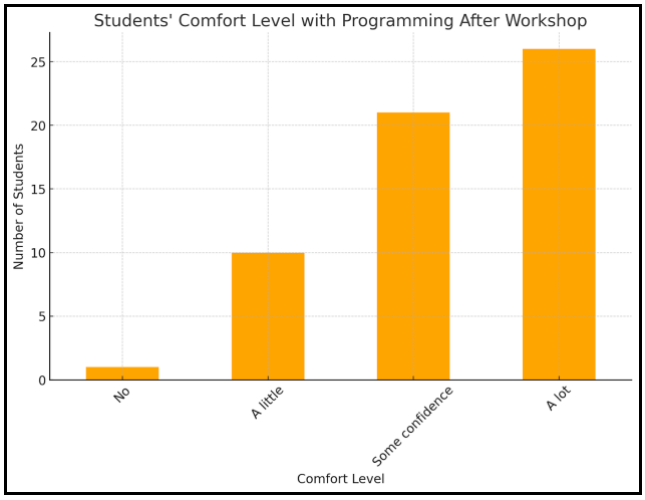

Figure 6.

Additionally, A considerable number of participants discovered a new enthusiasm for programming, largely attributed to the user-friendly and less intimidating nature of the block-based approach, as opposed to traditional line coding. As one student noted “I had the impression that programming would be difficult, but this lecture made me realize the freedom and ease of programming. Thanks to this, I became interested in programming.” Our workshop effectively made software development more tangible and engaging, particularly for beginners, by demystifying the process.

The practicality of the workshop was another key aspect highlighted by attendees. One student mentioned “This workshop introduced us to block programming which is straightforward, requiring no prior coding experience.” Others mention that it helped clarify fundamental programming concepts such as event handling, data storage, and the overall logic of programming languages. The workshop proved to be revelatory for those initially skeptical or unfamiliar with block-based programming, demonstrating how this kind of programming can streamline and elevate the development process.

Furthermore, the workshop showcased the exciting possibilities of integrating AI into app development. It not only sparked an interest in programming and AI among participants but also shed light on alternative approaches to programming, such as visual programming and the use of pre-built components, highlighting the diverse applications and versatility of AI.

3.2. Was App Inventor an effective tool to learn?

Participants unanimously lauded MIT App Inventor for its user-centric, accessible interface, highlighting its particular appeal to novices and those with minimal coding background. Its simplicity, a stark contrast to traditional coding approaches, stood out as a significant benefit. The platform's block-based, drag-and-drop interface was celebrated for demystifying the app development process, as encapsulated by one participant's remark, “It is easy to tinker around with blocks, making programming far less daunting than traditional line coding.”

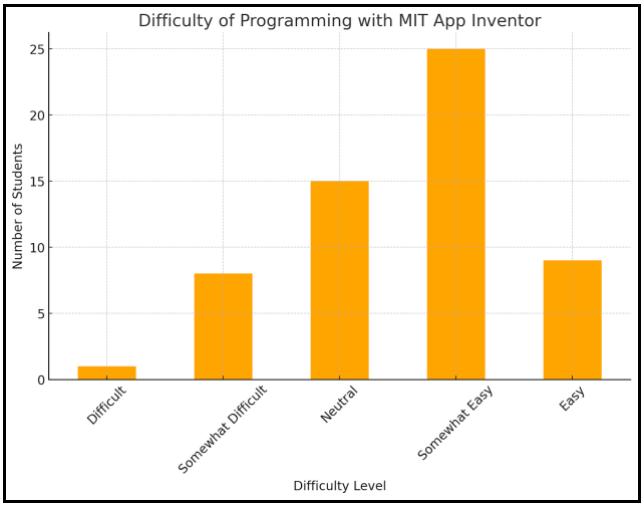

Figure 7.

The ease with which users could navigate and utilize App Inventor was a recurring theme among feedback. Its direct, no-frills functionality facilitated a seamless and swift app creation experience, devoid of the complexities often associated with coding. The platform’s design, inherently accommodating to those without a coding pedigree, enables the swift and straightforward development of mobile applications. This accessibility is pivotal, positioning App Inventor as an invaluable resource across a broad spectrum of users, particularly those venturing into programming for the first time. Moreover, the platform’s intuitive structure allows users to quickly comprehend both the logic behind app development and its design aspects. This feature was especially attractive to participants who, despite finding traditional coding barriers, were keen on venturing into mobile app development.

Also, some praised the geometric tinkering process of App Inventor. One student noted “the workshop ignited my interest in programming, particularly because I tend to avoid tasks that require extensive memorization, like learning a programming language. The transformation of programming into a puzzle-like format simplified the learning process for me, allowing me to grasp the underlying concepts of the project more intuitively.”. The visual nature of App Inventor, where coding is akin to

solving puzzles, was highlighted as a feature that enhances learning and retention, especially for those who struggle with writing code from memory. The platform was also lauded for its ability to facilitate understanding of technological developments and for making programming a more approachable and enjoyable experience.

Furthermore, the platform was recognized for its efficiency in frontend development and its broad functionality, supporting various features needed for smartphone application creation as one mentioned "The breadth of functionality that allows for the implementation of a complete set of functions needed to create a smartphone application, as well as support for external hardware such as pose estimation, ChatBot, cloud, Lego, etc." The convenience of real-time programming checks and the reduced need for high-end equipment were also mentioned.

3.3. What can App Inventor improve?

The feedback from participants on MIT App Inventor was varied, focusing on enhancements in user interface (UI) design, additional features, and educational resources.

UI/UX Design Improvements: Several respondents suggested more flexibility and customization options in the UI design of the platform. This included a desire for more UI components and the ability to edit code directly for customizing UI and logic. Improvements in UI/UX design were a recurring theme, with suggestions like a more user-friendly interface and the introduction of features like dark mode.

Enhanced Features: Participants expressed interest in seeing more advanced features in App Inventor. Specific suggestions included improved functionality for the chatbot and imagebot components, image recognition AI, and an in-built emulator for quick app testing. Some users also requested more variety in components for editing user interfaces and a desire for the platform to support hardcoding.

Educational Resources: Requests for more comprehensive educational resources were common. This included more advanced tutorials, both in video and PDF formats, complete documentation about the blocks, and additional tutorials on diverse topics, including game development. The idea of making tutorials more accessible and inclusive for various learning environments was also highlighted.

Accessibility and Language Support: Enhancements in accessibility features, such as Japanese language support and a clearer display of warnings and commands, were mentioned. Suggestions for an offline mode and improvements in the website's UI/UX were also proposed.

Performance and Bug Fixes: Addressing performance issues and fixing bugs were noted as areas for improvement. This includes dealing with issues where blocks do not display or the display freezes.

Community and Collaboration Features: Some participants suggested features to facilitate sharing and collaboration directly within the app, such as enabling multiple users to work on an app simultaneously and hosting activities to promote App Inventor's growth globally.

Transparency in Coding: A few responses indicated an interest in seeing the block-based code translated into standard programming language notation, which could help those interested in transitioning to traditional coding.

4. DISCUSSION

In this study, we evaluated the efficacy of MIT App Inventor as an educational tool for imparting Computational Thinking and Generative AI skills to students globally, extending beyond the confines of the United States. The workshop, encompassing a blend of theoretical learning and hands-on practice over five days, effectively guided students through the core principles and practical applications of programming. The strategic use of App Inventor was instrumental in this process, enabling students to engage with coding constructs visually. This approach not only facilitated a deeper understanding of programming concepts but also made the learning journey more accessible and less daunting for beginners.

Our analysis reveals that students overwhelmingly favored the block-based programming approach offered by App Inventor, appreciating its intuitiveness and ease of access. The feedback underscored a significant enhancement in their confidence in programming, coupled with a newfound inspiration to integrate these skills into their daily lives. Importantly, the various applications developed by the students during the workshop are a testament to their creative engagement with the tool. These applications reflect not just a grasp of programming concepts but also a broader vision of using technology as a means of personal and community development.

This aligns closely with the essence of Computational Action, where the application of learned skills in real-world scenarios is as crucial as the learning itself. The successful implementation of App Inventor in this context showcases its potential as a powerful catalyst in the realm of educational technology, particularly in fostering a deeper, more practical understanding of Computational Thinking and Generative AI across diverse student populations. The study thereby contributes valuable insights into the scalability and adaptability of such educational tools in a global educational landscape, highlighting their role in shaping a technologically adept and innovative future generation.

5. REFERENCES

John Maloney, Mitchel Resnick, Natalie Rusk, Brian Silverman, and Evelyn Eastmond. 2010. The Scratch Programming Language and Environment. ACM Trans. Comput. Educ. 10, 4, Article 16 (November 2010), 15 pages. https://doi.org/10.1145/1868358.1868363

David Wolber, Harold Abelson, and Mark Friedman. 2015. Democratizing Computing with App Inventor. GetMobile: Mobile Comp. and Comm. 18, 4 (October 2014), 53–58. https://doi.org/10.1145/2721914.2721935

Katerina Perdikuri. 201). Students' Experiences from the use of MIT App Inventor in classroom. In Proceedings of

the 18th Panhellenic Conference on Informatics (PCI '14). Association for Computing Machinery, New York, NY, USA, 1–6. https://doi.org/10.1145/2645791.2645835

Shuchi Grover and Roy Pea. 2013. Using a discourse-intensive pedagogy and android's app inventor for introducing computational concepts to middle school students. In Proceeding of the 44th ACM technical symposium on Computer science education (SIGCSE '13). Association for Computing Machinery, New York, NY, USA, 723–728. https://doi.org/10.1145/2445196.2445404

Jeannette M. Wing. 2006. Computational thinking. Commun. ACM 49, 3 (March 2006), 33–35. https://doi.org/10.1145/1118178.1118215

Siu-Cheung Kong, Harold Abelson. (2022). Computational Thinking Education in K–12: Artificial Intelligence Literacy and Physical Computing. The MIT Press

Mike Tissenbaum, Josh Sheldon, and Hal Abelson. 2019. From computational thinking to computational action. Commun. ACM 62, 3 (March 2019), 34–36. https://doi.org/10.1145/3265747

Xiaoxue Du, Robert Parks, Selim Tezel, Jeff Freilich, H. Nicole Pang, Hal Abelson, and Cynthia Breazeal. 2023. Designing a Computational Action Program to Tackle Global Challenges. In Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 2 (SIGCSE 2023). Association for Computing Machinery, New York, NY, USA, 1320. https://doi.org/10.1145/3545947.3576267

Patton, E.W., Tissenbaum, M., Harunani, F. (2019). MIT App Inventor: Objectives, Design, and Development. In: Kong, SC., Abelson, H. (eds) Computational Thinking Education. Springer, Singapore. https://doi.org/10.1007/978-981-13-6528-7_3

OpenAI. (2016) Generative Models. Generative models (openai.com)

Sharples, M. (2023). Towards social generative AI for education: theory, practices and ethics. Learning: Research and Practice, 9, 159 - 167.

Shi, Z., Zhou, X., Qiu, X., & Zhu, X. (2020). Improving image captioning with better use of captions. arXiv preprint arXiv:2006.11807