ISSN 2234-4438
융합정보논문지 제7권 제1호 pp. 61-66, 2017 DOI : https://doi.org/10.22156/CS4SMB.2017.7.1.061

컴퓨팅 사고 기반의 비전공자 소프트웨어 교육을 위한 앱 인벤터 교육과정 설계

구진희
목원대학교 정보통신융합공학부

Designing an App Inventor Curriculum for Computational Thinking based Non-majors Software Education

Jin-Hee Ku
Department of Information Communication Engineering, Mokwon University

요 약 4차 산업혁명이 본격화되고 인공지능, 사물인터넷 기술 등을 적용한 첨단 서비스들이 대거 상용화되면서 소프트웨어의 중요성에 대한 인식 또한 확산되고 있다. 이러한 배경으로 최근 소프트웨어 교육은 초중등뿐만 아니라 대학에서도 필수로 가르치는 추세이며, 컴퓨팅적 방법론과 모델을 통해 문제를 해결하는데 필요한 컴퓨팅 사고에 대한 관심도 높아지고 있다. 본 연구의 목적은 컴퓨팅 사고에 기반을 두고 비전공자 소프트웨어 교육을 위한 앱 인벤터 교육과정을 설계하는 것이다. 연구의 결과, 컴퓨팅 사고의 세부 역량 6가지를 도출하였고, 6가지 세부 역량은 앱 인벤터 학습요소를 구성하여 맵핑하였다. 또한 컴퓨팅 사고 활용 모델링에 기반하여 대학 교양핵심 교육과정의 IT 교과목에 참여한 학생들에게 적용하기 위한 앱 인벤터 수업 내용을 설계하였다.

키워드 : 컴퓨팅 사고, 비전공자, 소프트웨어 교육, 앱 인벤터, 4차 산업혁명

Abstract As the fourth industrial revolution becomes more popular and advanced services such as artificial intelligence and Internet of Things technology are widely commercialized, awareness of the importance of software is spreading. Recently, software education has been taught not only in elementary school and college but also in college. Also, there is a growing interest in computational thinking needed to solve problems through computing methodology and model. The purpose of this study is to design an app inventor course for non-majors software education based on computational thinking. As a result of the study, six detailed competencies of computational thinking were derived, and six detailed competencies were mapped to the app inventor learning elements. In addition, based on the computational thinking modeling, I designed an app inventor class for students who participated in IT curriculum of university liberal arts curriculum.

Key Words : Computational Thinking, Non-majors, Software Education, App Inventor, The Fourth Industrial Revolution

1. 서론

현대식 컴퓨터가 등장하고 70여년이 지난 오늘날, 컴퓨팅 기술은 한정된 소수의 사람만이 사용했던 기계에서 모든 분야에 걸쳐 유용한 기술로 인식되고 있다. 최근 4차 산업혁명이 본격화되고 전 세계적으로 소프트파워의 중요성을 인식하면서 컴퓨팅 기술은 점차 소프트웨어 중심으로 변화하고 있다. 이와 같이 컴퓨팅 기술이 보편화

Received 2017-01-15 Revised 2017-02-01 Accepted 2017-02-20 Published 2017-02-28
Corresponding author : Jin-Hee Ku (jhku@mokwon.ac.kr)

되면서 컴퓨팅적 방법론과 모델을 통해 문제를 해결하는 데 필요한 컴퓨팅 사고에 대한 관심도 급증하였다.

컴퓨팅 사고(Computational Thinking)는 컴퓨터 과학자들이 문제해결을 위해 사고하는 방법과 그와 관련된 기본적인 문제해결 기술로부터 출발하였다[1]. 최근 컴퓨터 과학자뿐만이 아니라 누구나 갖춰야 하는 기본적인 역량으로 읽기, 쓰기, 셈하기와 같이 인간의 가장 기본적인 역량 항목으로 강조되고 있으며[2], 오늘날에는 모든 분야에 걸쳐 일반적인 문제 해결에 필요한 능력으로 인식되고 있다. 특히, 컴퓨팅 기술이 예술, 인문학, 사회 과학 등 거의 모든 학문 분야에 걸쳐 융합되는 현재 상황에서 소프트웨어 교육은 컴퓨팅 사고를 기반으로 문제해결 능력을 키울 수 있도록 하고 다양한 학문 영역과 융합할 수 있도록 교육과정이 운영될 필요성이 있다[3].

Cuny, Snyder와 Wing은 컴퓨팅 사고가 일반적으로 코딩 및 컴퓨터 프로그래밍과 관련이 있지만 문제 해결과 시스템 설계 그리고 인간 행동 이해와 관련된 것 이상을 포괄한다고 정의하였다[4]. 또한 Carlisle, Wilson, Humphries와 Hadfield는 비전공자 프로그래밍 학습은 구문을 학습하기 전에 플로우챠트의 알고리즘 시각화를 통해 문제해결을 먼저 학습해야 함을 설명하였다[5]. 이들은 프로그래밍 입문 또는 비전공자의 프로그래밍 학습에 있어서 코딩 및 프로그래밍보다 컴퓨팅 사고 기반의 시각적 소프트웨어 개발 도구가 효과적임을 주장하고 있다[6].

MIT 미디어 랩은 그들의 철학이 담긴 앱 인벤터에 대해서 텍스트 기반 코딩의 복잡한 언어를 시각적인 드래그 앤 드롭 빌딩 블록으로 변환하는 프로그래밍 및 앱 제작에 대한 혁신적인 초보자 입문서라고 소개하였다[7]. 현재 대학 및 K-12 레벨의 입문 레벨 코스를 훌륭하게 지원하고 있으며, 접근하기 쉽고 강력한 플랫폼으로 활용되고 있다[8]. Morelli, Lanerolle, Lake, Limardo, Tamotsu와 Uche 등은 그들의 프로젝트를 통해서 앱 인벤터가 K-12 학생들에게 컴퓨터 사고를 가르치기 위한 좋은 플랫폼임을 제시하였다[9].

이와 같은 배경으로 본 연구에서는 비전공자의 소프트웨어 교육을 하기 위해서 컴퓨팅 사고에 기반하여 앱 인벤터 교육내용을 설계하고자 한다. 본 논문의 구성은 다음과 같다. 2장에서 컴퓨팅 사고에 대한 이론적 고찰과 대학의 소프트웨어 교육과정에서의 컴퓨팅 사고 교육 및 앱 인벤터에 대해 살펴본다. 3장에서는 연구방법에 대해 기술하고 4장에서는 연구결과, 마지막으로 5장에서는 결론을 기술하였다.

2. 관련연구

2.1 컴퓨팅 사고

컴퓨팅 사고(computational thinking)는 2006년 카네기 멜론 대학의 Wing에 의해 처음 소개되었다. Wing에 의하면, 컴퓨팅 사고는 컴퓨터과학자뿐만이 아니라 누구나 갖춰야 하는 기본적인 역량으로 읽기, 쓰기, 셈하기와 같이 인간의 가장 기본적인 역량 항목에 추가해야 함을 강조했다[2]. 영국 BBC 교육 웹사이트 Bitesize는 KS3(Key Stage 3) 컴퓨터과학 주제에서 알고리즘이나 프로그래밍에 우선하여 컴퓨팅 사고를 제시하고 있으며[10], 미국 CSTA의 컴퓨터과학 교육과정에서도 K-12 학습자에게 단계별로 컴퓨팅 사고를 길러주는 것에 목표를 두고 있다[11].

2.2 소프트웨어 교육

4차 산업혁명 시대가 본격화됨에 따라 전 세계적으로 소프트파워에 대한 중요성을 크게 인식하고 초중등 교육과정뿐만 아니라 대학에서도 소프트웨어 교육에 대한 필요성이 제기되었다[12~13]. 인공지능, 사물인터넷 등 IT 혁신을 통한 고도의 지능화된 연결 사회를 의미하는 4차 산업혁명은 우리 일상생활과 산업구조 전반에 많은 변화를 가져올 것으로 예상된다[14]. 소프트웨어 중심 대학은 이러한 산업현장의 요구를 반영하고 대학 소프트웨어 교육을 혁신하기 위해 2015년 8개 대학, 2016년 6개 대학이 선정되어 운영되고 있다. 또한, 최근 소프트웨어 융합 인력 수요가 크게 증가함에 따라 소프트웨어 중심 대학을 포함한 많은 대학에서 비전공자에 대한 소프트웨어 기초 교육을 강화하고 있다[12, 15]. Table 1은 소프트웨어 중심 대학으로 선정된 16개 대학의 소프트웨어 기초 교육을 위한 교육과정을 연구자가 재정리하여 나타낸 것이다. Table 1의 소프트웨어 기초 교육을 위한 필수 이수 교과목을 보면, 거의 모든 학교의 교육과정에서 컴퓨팅 사고와 프로그래밍을 통한 문제해결 관련 과목이 포함되어 있음을 알 수 있다.

이와 같이 소프트웨어 기초 교육을 대학의 신입생 전체로 확대하고 소프트웨어 융합 전공을 강화하는 배경으

Table 1. Software oriented university's non-majors software basic education

Univ.	SW Fundamentals Course	Description
Gachon	Computational Thinking and Coding	1 course or more required, SW basic and SW convergence education
Kyungpook	SW Problem Solving Basics, SW courses by college	SW basic education (3-6 credits required)
Korea	Computational Thinking	Scratch, Python, HomePage, Physical Computing, etc.
Sogang	Computational Thinking(3), Courses by college(3)	All students up to 6 credits SW education with basic literacy
Sungkyungkwan	Computational Thinking and SW Coding(2), Problem Solving and Algorithm(2)	All students SW basic course liberal arts required
Sejong	SW Basic Coding, Introduction to Programming	Programming education for all of the non-major (C, Python, etc.)
Ajou	C Programming and Matlab Python Excel, R, etc.	Required 3 credits for each college + 3 credits for each department
Chungnam	Computational Thinking, Programming Fundamentals	Strengthening SW courses for all students
Kookmin	Computer Programming Ⅰ · Ⅱ	Excel, Scratch, Python, etc.
Dongguk	Computational Thinking (before enrollment), SW and Future sSociety	Overall trend of IT SW convergence technology
Busan	Computational Thinking, Computer Programming	6 hours 3 credits required, SW understanding & problem solving, SW production
SeoulWomen's	Software and creative thinking	SW education for each major specialty(6 courses)
Kaist	Basics of Programming	Required for all students, strengthening software convergence capabilities of non-majors
Hanyang	Creative Computing, Creative Programming	Computational thinking, Problem solving by programming, Computer Science Fundamentals, Advanced Software Skills

로는 인문·경영 등 타 전공과 소프트웨어의 새로운 융합 교육을 통한 지능정보사회를 주도할 다양한 분야의 소프트웨어 역량을 겸비한 융합인재 양성에 있다고 볼 수 있다[15~16].

2.3 앱 인벤터와 컴퓨팅 사고

Morelli, Lanerolle, Lake, Limardo, Tamotsu와 Uche 등은 그들의 연구에서 앱 인벤터가 고등학생을 포함한 컴퓨터 비전공 학생들을 컴퓨터 사고와 컴퓨터 과학 연구에 끌어들이는 엄청난 잠재력을 가지고 있다고 주장하였다[9]. 컴퓨팅 사고가 21 세기를 위한 근본적인 기술이 되면서 Yadav, Zhou, Mayfield, Hambrusch와 Korb는 K-12 교사들이 컴퓨팅 원리에 익숙해 져야 한다고 주장하였다. 컴퓨팅 사고 모듈의 구현 및 평가를 구현한 그들의 연구에서 컴퓨팅 사고에 관한 관련 정보가 주어지면 컴퓨터 과학에 대한 교육 학생의 태도가 더욱 유리 해지고 미래의 가르침에서 컴퓨팅 원리를 통합 할 가능성이 높아질 것이라고 제안하였다[19].

앱 인벤터는 웹 페이지와 자바 인터페이스를 이용해서 안드로이드 애플리케이션을 디자인하고 프로그래밍할 수 있는 시스템이다. 2009년 구글에서 처음 개발하여 무료로 제공되었으며 현재는 2011년 MIT 미디어 랩에서 발표한 블록 기반 프로그래밍 언어 앱 인벤터가 사용되고 있다. 클라우드 플랫폼 서비스 방식으로 배포하기 때문에 웹 브라우저를 통해 실행되는 앱 인벤터 도구들을 이용하여 자바와 같은 프로그래밍 언어에 대한 지식이 없이도 간편하게 안드로이드 앱을 만들 수 있는 것이 특징이다. 컴포넌트와 프로그래밍 로직을 추가해가며 애플리케이션을 완성해가는 비주얼 프로그래밍 플랫폼이다.

앱 인벤터는 컴퓨터 과학 교육을 변화시킬 수 있는 큰 가능성을 가지고 있으며, 학생들로부터 컴퓨팅에 관심을 갖게 하는 동기 부여에 가장 좋은 도구이다. 샌프란시스코 대학(University of San Francisco)의 사례에서 대부분 수학에 대한 두려움을 갖고 있거나 프로그래밍 경험이 없는 역사, 문학, 정치, 디자인, 비즈니스 등 다양한 전공의 학생들을 대상으로 한 교육에서 컴퓨터 과학에 대한 문제 해결 능력과 지식을 믿을 수 없을 만큼 향상되었음을 보고하였다[8].

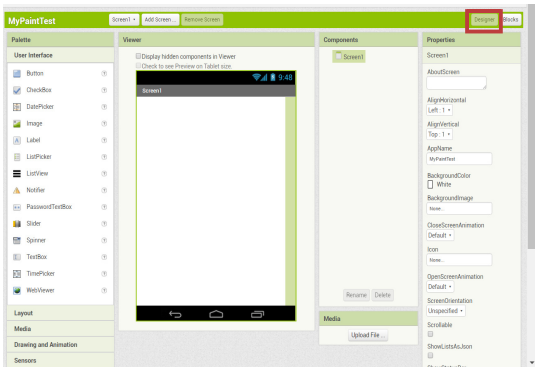


Fig. 1. App Inventor2 Designer

Fig. 1은 앱 인벤터의 디자이너(Designer)를 보여주고 있다. 앱 인벤터 디자이너는 안드로이드 앱을 위한 컴포넌트들을 선택하고 사용자 인터페이스를 디자인하는데 사용되는 도구이다.

Fig. 2는 앱 인벤터의 블록(Blocks) 에디터 화면이다. 블록 에디터는 디자이너에서 화면 디자인을 한 다음, 앱이 동작하도록 하려면 블록 에디터 프로그램을 실행시켜야만 한다. 블록에디터는 앱의 행동(behavior)을 설정하는데 사용되는 독립적인 응용 프로그램이다.앱을 작성하기 위해서 2개의 윈도우를 전환하면서 프로그래밍 해야 한다.

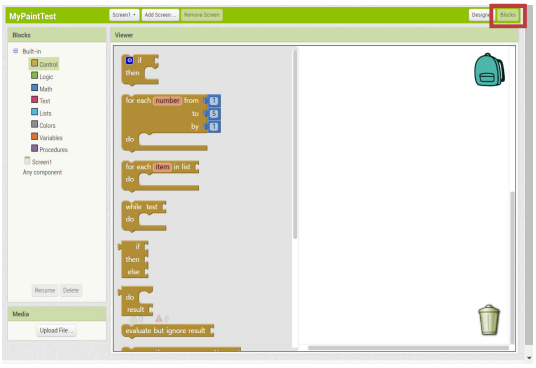


Fig. 2. App Inventor2 Blocks

3. 연구방법

3.1 연구절차

본 연구에서는 대학의 교양 교육과정을 이수하는 컴퓨터 비전공자 학생들을 대상으로 컴퓨팅 사고 활용 모델링에 기반하여 7주간의 앱 인벤터 수업 내용을 설계하였다. 컴퓨팅 사고의 세부 역량은 K-12[11]에서 제시한 9가지 세부 역량과 샌프란시스코 대학의 Wolber 교수의 컴퓨팅, 모바일 앱 및 웹(CS 107) 교육과정[20~21]에 기초하여 연구자가 6가지로 재구성하였다.

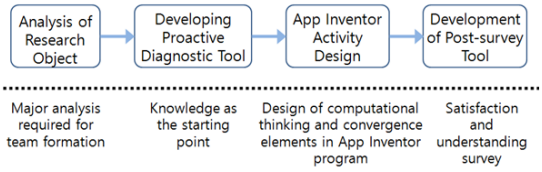


Fig. 3. Research Process

4. 연구결과

4.1 컴퓨팅 사고의 세부 역량과 앱 인벤터 학습요소 구성

본 연구에서는 컴퓨팅 사고 세부 역량을 6가지로 구성하고 이를 기반으로 앱 인벤터를 활용한 학습요소를 도출하였다. 다음 Table 2는 컴퓨팅 사고의 세부 역량과 앱 인벤터 학습요소와의 관계를 나타낸 것이다.

4.2 컴퓨팅 사고 활용 모델링 및 수업 전략

Table 2에서 제시한 6개의 세부 역량에 따라 1주에 3시간 수업을 7주 동안 진행하도록 수업 내용을 설계하였다. Table 3은 페인트앱 프로젝트의 앱 인벤터 학습요소에 대해 컴퓨팅 사고를 활용 모델링 한 예시이다.

수업은 클라우드 기반인 앱 인벤터를 이해하기 위해 1주는 이론 수업이 진행되었고 협력학습과 문제해결학습에 초점을 두고 진행하였다.

Table 2. The detailed competence of computational thinking

The detail competence of computational thinking	Concepts	App Inventor Learning Elements
Decomposition	Disassemble complex problems or systems into small problems that are easy to understand and manage	Components, Variables, Operations
Pattern Recognition	Finding similarities or patterns between small dis-aggregated problems	Condition, Iteration, Procedure
Representation	Understand the problem and organize it into appropriate graphs, texts and pictures	Event
Abstraction	Steps to set the definition of the primary key concept to reduce the complexity of the problem	UI, Designer
Algorithms	A step-by-step plan, a set of instructions to solve the problem	Condition, Iteration
Automation	The process of implementing and seeing the results in a computing tool (language)	Blocks, Emulator

Table 3. Example of Computational Thinking Utilization Modeling (Paint App Project)

App Inventor Lesson Contents	Computational Thinking Utilization Modeling
Visible, Non-visible Component, Data, Calculate	[Decomposition] Functions of Paint are divided into functions such as color selection, drawing mode selection, function selection, and drawing. Disassemble to the function of a smaller unit if possible.
Condition, Iteration, Procedure	[Pattern Recognition] Look for similarities or associations between disassembled problems (functions).
Event	[Representation] Small unit problems, associations and similarities should be expressed in text and pictures.
UI learning, Designer	[Abstraction] Use your designer screens to design your own paint apps. Do not specify specific functional specifications.
Condition, Iteration	[Algorithms] Create a specific functional specification for the pattern abstraction.
Block learning, Running on emulator or smartphone	[Automation] Complete the block screen based on the designer screen and the specific functional specification of the problem.

Fig. 4는 Table 3의 교육과정을 컴퓨팅 사고 활용 모델링에 기반하여 제작한 페인트 앱의 디자인과 실행화면이다.

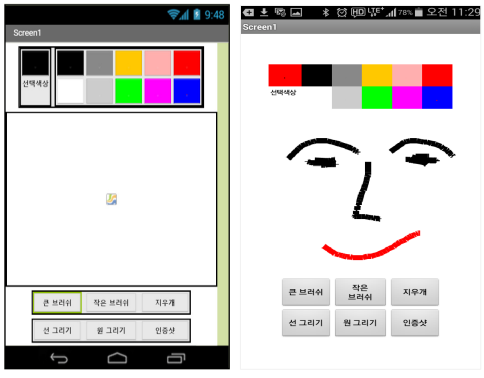


Fig. 4. Paint app design and launch screen

5. 결론 및 논의

본 연구에서는 컴퓨터 비전공 학생들을 위한 컴퓨팅 사고 기반의 앱 인벤터 교육과정을 설계하고 적용하는 방안을 제안하였다.

특히, 앱 인벤터는 프로그래밍의 기초를 다루는 것뿐만 아니라 센서 및 로봇 제어 앱 등을 제작할 수 있다. 따라서 본 논문에서 제안한 컴퓨터 비전공자를 위한 컴퓨팅 사고 기반의 앱 인벤터 수업 설계는 최근 컴퓨팅 사고 향상을 위해 여러 연구에서 시도되고 있는 로봇 프로그래밍, 피지컬 컴퓨팅 등 다양한 형태의 컴퓨팅 사고에 초점을 둔 교육과정 개발 연구에서 활용될 수 있을 것으로 기대된다.

REFERENCES

[1] D. S. Kim, *Software and Computational Thinking*, Life and Power Press, 2015.

[2] J. M. Wing, "Computational Thinking", *Communication of the ACM*, Vol. 49, No. 3, Mar. 2006.

[3] J. H. Ku, S. B. Shin, M. Y. Kim, "Instructional Design using App Inventor to Improve Computational Thinking of Non-majors Computing", *Proceedings of the Korea Academia-Industrial cooperation Society*, 2016.

[4] Carnegie Melon University, "Center for Computational Thinking," https://www.cs.cmu.edu/~CompThink, 2016. 10.

[5] M. C. Carlisle, T. A. Wilson, J. W. Humphries and S. M. Hadfield, "RAPTOR : Introducing Programming to Non-Majors with Flowcharts," *Journal of Computing Sciences in Colleges*, Vol. 19, Issue 4, pp. 52-60, April, 2004.

[6] M. Urban-Lurain, D. J. Weinshank, "Is There A Role for Programming in Non-Major Computer Science Courses?," *Frontiers in Education Conference*, FIE 2000 30th Annual, 2000.

[7] http://www.appinventor.org/, 2017. 02.

[8] D. Wolber. "Teaching App Inventor", AppInventor.org, http://www.appinventor.org/teaching-android, no date.

[9] R. Morelli, T. de Lanerolle, P. Lake, N. Limardo, E. Tamotsu, C. Uche, "Can Android App Inventor Bring Computational Thinking to K-12?", http://hfoss.org/uploads/docs/appinventor_manuscript.pdf, 2010.

[10] BBC, "Bitesize", http://www.bbc.co.uk/education, no date.

[11] Computational Thinking teacher resources. http://www.csta.acm.org, no date.

[12] Software Oriented Society, Available From: https://www.software.kr, Sep, 2016. 09.

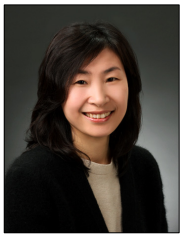
[13] Korea Institute for Curriculum and Evaluation, "Introduction to the revised curriculum in 2015", Research Report CRC 2015-28, 2015.

[14] World Economic Forum, *The Future of Jobs-Employment, Skills and Workforce Strategy for the Fourth Industrial Revolution*, Jan. 2016.

[15] Ministry of Science, ICT and Future Planning, *Software oriented university promotion plan*, 2015.
[16] Ministry of Science, ICT and Future Planning, *Minister of Science, ICT and Future Planning-Software oriented university President Meeting Press release*, Oct. 2016.
[17] MIT App Inventor, http://appinventor.mit.edu/explore, no date.
[18] J. Tyler, *App Inventor for Android : Build Your Own Apps No Experience Required!*, John Wiley & Sons, 2011.
[19] A. Yadav, N. Zhou, C. Mayfield, S. Hambrusch, J. T. Korb, "Introducing Computational Thinking in Education Courses", *Proceedings of the 42nd ACM technical symposium on Computer science education*, pp. 465-470, 2011.
[20] D. Wolber, H. Abelson, E. Spertus, L. Looney, *App Inventor2 (2nd Edn)*, O'Reilly Media, 2015.
[21] Computing, Mobile Apps, and the Web (CS 107) USF, https://sites.google.com/site/appinventorcourse/spring-2013-schedule, no date.

저 자 소 개

구 진 희(Jin-Heel Ku) [정회원]



- 2001년 2월 : 충남대학교 컴퓨터과학교육학과(교육학석사)
- 2010년 2월 : 충남대학교 공업(컴퓨터)교육학과(교육학박사)
- 2010년 9월 ~ 현재 : 목원대학교 정보통신융합공학부 교수

<관심분야> : 컴퓨터과학 교육, 빅데이터, 데이터마이닝, 클라우드 컴퓨팅