http://dx.doi.org/10.5392/JKCA.2013.13.02.019

프로그래밍 초보자를 위한 스타일직소의 구현과 실험

Implementation and Experimentation of StyleJigsaw for Programming Beginners

이윤정*, 정인준**, 우균***
부산대학교 컴퓨터 및 정보통신연구소*, eBay 코리아**,
부산대학교 컴퓨터공학과/컴퓨터 및 정보통신연구소/LG전자 스마트 제어센터***

Yun-Jung Lee(leeyj01@pusan.ac.kr)*, In-Joon Jung(spd1335@pusan.ac.kr)**,
Gyun Woo(woogyun@pusan.ac.kr)***

요약

가독성 있는 소스코드는 이해하기 쉽고 수정하기 편하기 때문에 손쉽게 유지보수할 수 있다. 소스코드의 가독성은 프로그램의 제어 구조와 같은 알고리즘의 복잡도뿐만 아니라 함수명, 들여쓰기 등과 같은 코딩스타일에 의해서도 많은 영향을 받는다. 지금까지 소스코드의 가독성을 높이기 위해 다양한 코딩 표준들이 제안되었으나 프로그래밍 교과목에서는 코딩스타일을 다루지 않거나 무시하는 경우가 많았다. 그 이유는 코딩스타일이 프로그램의 효율에는 영향을 주지 않기 때문에 강제하기 어렵기 때문이다. 이 논문에서는 프로그램 소스코드의 코딩스타일을 분석하고 그 결과를 시각화하는 스타일직소(StyleJigsaw) 시스템을 제안한다. 스타일직소 시스템은 C/C++나 Java 언어로 작성된 소스코드의 코딩스타일을 분석하여 이를 정량화하고 그 결과를 퍼즐화된 이미지로 시각화한다. 스타일직소 시스템의 교육적 효과를 입증하기 위해 C++ 프로그래밍 수업을 듣는 학생들을 대상으로 스타일직소 사용 실험을 진행하였다. 실험 결과 스타일직소 시스템을 사용한 경우 코딩스타일 평균 점수가 약 8.0점(10.9%) 가량 향상된 것으로 나타났다. 또한, 프로그래밍 수업의 수강생들을 대상으로 한 설문조사에서 약 88.5%의 학생이 스타일직소 시스템이 코딩스타일 학습에 도움이 되었다고 응답하였다. 프로그래밍 수업에서 스타일직소 시스템을 활용함으로써 학생들이 가독성 있는 프로그램 작성 능력을 기르는 데 도움을 줄 수 있을 것이다.

■ 중심어 : | 코딩표준 | 코딩스타일 | 프로그램 가독성 | 시각화 |

Abstract

Since the high readable source codes help us to understand and modify the program, it is much easy to maintain them. The readability of source code is not only affected by the complexity of algorithms such as control structures but also affected by the coding styles such as naming and indentation. Although various coding standards have been presented for promoting the readability of source codes, it has been usually lost or ignored in a programming course. One of the reasons is that the coding standard is not a hard-and-false rule since it does not contribute to the performance of software. In this paper, we propose a simple automatic system, namely StyleJigsaw, which checks the style of the source codes written by C/C++ or Java. In this system, the coding style score is calculated and visualized as a jigsaw puzzle. To measure the educational effectiveness of StyleJigsaw, several experiments have been conducted on a class students in C++ programming course. According to the experimental results, the coding style score increased about 8.0 points(10.9%) on average using StyleJigsaw. Further, according to a questionnaire survey targeting the students who attended the programming course, about 88.5% of the students responded that StyleJigsaw was of help to learn the coding standards. We expect that the StyleJigsaw can be effectively used to encourage the students to obey the coding standards, resulting in high readable programs.

■ keyword : | Coding Standard | Coding Style | Pragram Readability | Visualization |

* 이 논문은 부산대학교 자유과제 학술연구비(2년)에 의하여 연구되었음.
접수번호 : #121218-003
접수일자 : 2012년 12월 18일
심사완료일 : 2013년 01월 30일
교신저자 : 우균, e-mail : woogyun@pusan.ac.kr

I. 서 론

오늘날 소프트웨어 산업은 과거의 폐쇄적인 소프트웨어 개발 환경에서 벗어나 오픈 소스형 공개 소프트웨어의 확산으로 패러다임이 전환되고 있다. 오픈 소스는 누구라도 소스코드를 사용할 수 있으며, 소스코드를 개선하여 새로운 기능을 추가할 수도 있다. 따라서 개발 과정과 결과물이 공개되어 있기 때문에, 기술 능력이 상대적으로 취약한 개발자들과 사용자들이 공개 소프트웨어를 깊이 있게 살펴봄으로써 소프트웨어 개발 능력을 향상시킬 수 있다[1].

이러한 오픈 소스 개발 환경에서는 타인이 작성한 소스코드를 보고 프로그램의 구조 및 기능을 이해하는 과정이 점점 더 중요하게 인식되며 따라서 코드 가독성이 매우 중요하게 된다. 가독성이 높은 소스코드는 이해하기 쉽고 수정하기 편하기 때문에 유지보수에 도움을 준다[2]. 그뿐만 아니라 프로그램 개발 단계에서 개발자들 간의 코드 검토와 의사소통을 원활하게 해줌으로써 프로그램의 개발 및 유지보수에 드는 비용을 절감하는데 도움을 준다.

소스코드의 가독성은 프로그램의 제어 구조와 같은 알고리즘의 복잡도뿐만 아니라 함수명, 들여쓰기 등과 같은 코딩스타일에 많은 영향을 받는다. 지금까지 소프트웨어 개발 단체와 여러 연구자들은 소스코드의 가독성을 높이기 위해 여러 가지 코딩 표준들을 제안하였다[3-5]. 소프트웨어 개발에서 코딩 표준을 준수하는 것은 팀의 소통을 원활하게 하고, 프로그램 오류를 줄이며, 코드 품질을 개선시키는데 중요한 역할을 한다[6-7].

대부분의 개발자가 코딩 표준을 준수하는 것에 대한 중요성은 인식하고 있지만 실제 프로그램 개발에서는 코딩 표준을 소홀히 다루는 경향이 있다. 그 이유는 여러 가지로 생각해 볼 수 있겠지만 무엇보다도 코딩스타일은 프로그램의 실행에 직접적인 영향을 주지 않기 때문에 프로그램을 개발할 때 쉽게 무시되는 경우가 많고, 코딩스타일은 프로그래머의 오래된 코딩 습관에 많은 영향을 받기 때문이다. 따라서 가독성 있는 소스코드 작성을 위해서는 프로그램을 처음 배우는 단계에서 코딩 표준을 지키는 좋은 코딩 습관을 길러주는 교육이 필요하다[8].

대학의 프로그래밍 수업은 코딩 표준을 가르칠 좋은 시작점이라고 할 수 있다. 그러나 앞서 언급한 바와 같이 코딩스타일은 프로그램의 실행에 직접적인 영향을 주지 않으므로 실제 프로그래밍 수업에서 프로그램 언어의 문법이나 알고리즘에 비해 코딩스타일에 대한 비중은 그다지 크지 않으며, 대부분 평가에 반영되지 않고 있다. 이로 인해 학생들은 코딩 표준을 준수하는 것에 대해 큰 의미를 부여하지 않고 있으며, 코딩 표준을 지키는 것에 대해 어렵게 느끼는 경향이 있다.

Li는 프로그래밍 수업의 수강생들을 대상으로 한 설문조사를 바탕으로 문서로 코딩 규칙을 제시하는 것 보다는 예제나 샘플 코드를 통한 학습이 더 효율이라고 제시하였다[8]. 또한 효과적으로 코딩 규칙을 가르치기 위해서 학생들의 수준에 맞는 코딩 규칙을 제공하고, 코딩 규칙 학습에 대한 지속적인 피드백이 필요하다고 제안하였다.

본 연구자들은 이전 논문에서 소스코드의 코딩 규칙의 준수 정도를 정량화하고 이를 얼굴 이미지를 이용하여 시각화하는 StyleVisualizer를 제안하였다[9]. 실제로 프로그래밍 수업에서 StyleVisualizer를 활용하여 학생들에게 코딩스타일에 대한 관심을 유도하고, 학생들 스스로 코딩스타일을 검사하게 할 수 있었다. 그러나 StyleVisualizer는 에러 없이 실행 가능한 소스코드만 코딩스타일 체크가 가능하므로 프로그램을 완성하지 않은 경우에는 코딩스타일 점수를 계산할 수 없는 단점이 있었다. 또한 코딩스타일 검사 결과를 얼굴 이미지의 표정 변화를 통해서만 확인할 수 있기 때문에 전체적인 정확한 코딩스타일 점수나 소스코드의 어느 부분이 코딩규칙에 위배되는지 등의 정보를 알 수 없었다.

이 논문에서는 StyleVisualizer에서 나타난 이러한 단점을 보완한 스타일직소(StyleJigsaw) 시스템을 제안하고자 한다. 스타일직소는 C/C++와 Java 프로그래밍 언어를 위한 자동화된 소스코드 스타일 분석 시스템이다. 스타일직소 시스템은 사용자가 미리 정한 코딩 규칙을 적용하여 소스코드의 코딩스타일을 분석하고 그 결과를 정량화하고, 그 결과를 직소퍼즐 형태의 이미지로 시각화한다.

기존에 제안되었던 StyleVisualizer는 에러 없이 실행 가능한 완성된 소스코드에만 적용가능했던 것에 반해 스타일직소 시스템은 이미 작성되어 실행 가능한 소스코드뿐만 아니라 작성중인 소스코드에도 적용이 가능하다. 따라서 사용자는 소스코드 작성 도중에 자신의 코딩스타일을 지속적으로 점검할 수 있다.

코딩스타일 분석 결과는 코딩 규칙 준수 정도에 따라 수치로 정량화되고, 그 결과는 퍼즐로 변환된 이미지로 시각화된다. 이때 코딩스타일 점수가 높을수록 이미지의 많은 부분을 보여주어 학생들의 관심을 유도하고, 학생들 스스로 코딩스타일에 대한 지속적인 평가와 피드백을 제공받을 수 있다.

이 논문은 이전 논문[11]의 기본 아이디어를 바탕으로 하여 C 언어로 작성된 소스코드뿐만 아니라 C++와 Java 언어로 작성된 소스코드도 적용가능하게 확장하였다. 또한 직소퍼즐 생성 모듈과 사용자 인터페이스를 새롭게 개선하여 구현하였다. 스타일직소를 이용하여 사용자는 자신이 검사할 규칙을 선택할 수 있으며 또한 자신만의 퍼즐 이미지를 선택할 수 있다.

이 논문의 구성은 다음과 같다. 2장에서는 프로그래밍 언어의 코딩 표준과 코딩스타일 평가에 관한 기존 연구들을 살펴본다. 3장에는 스타일직소 시스템의 구성과 기능에 대해 자세히 설명한다. 4장에서는 실제 프로그래밍 수업에서 스타일직소 시스템을 활용한 실험결과를 제시한다. 실험은 크게 스타일직소 사용 전후의 학생들 과제물에 대한 코딩스타일 점수 비교와 스타일직소의 사용에 대한 학생들의 설문응답으로 구성된다. 마지막으로 5장에서 결론과 향후연구를 제시한다.

II. 관련 연구

1. C/C++와 Java를 위한 코딩 표준

C/C++와 Java는 현재까지 소프트웨어 개발을 위해 가장 많이 사용되고 있는 프로그래밍 언어로서 여러 단체에서 이 언어들을 위한 코딩 표준을 제시하였다. 1990년에 AT&T사의 벨 연구소에서 Indian Hill C 스타일을 제안하였다[12]. Indian Hill C 스타일은 프로그램 스타일 표준 확립과 프로그램 가이드를 제공하기 위한 것으로 코딩스타일을 C 언어의 문법적 구성 요소를 기준으로 총 11개의 범주(파일, 주석, 함수선언, 일반선언, 공백, 단순문, 복합문, 연산자, 이름규칙, 상수, 매크로)로 분류하고 있다.

1992년에는 GNU 코딩 표준이 발표되었는데, 이것은 공개 소스를 기반으로 하는 GNU 개발 프로젝트를 진행할 때 준수해야 할 사항들을 정의하고 있다[4]. GNU 코딩 표준 문서는 1992년에 최초로 공개되었으며 현재까지 지속적으로 업데이트되고 있다. 가장 최신 문서는 2012년 6월 30일에 업데이트된 것이다. GNU 코딩 표준에서는 코딩스타일을 명시적으로 분류하고 있지는 않지만, 프로그램 설계와 프로그램 문서화 등의 요소에 대해 세부 규칙들을 정의하고 있다.

MISRA C 가이드라인은 자동차 산업용 소프트웨어 개발을 위한 표준 지침으로 정의되었으며, 현재는 자동차뿐만 아니라 모든 산업 분야에 적용할 수 있도록 재정의되었다[13-15]. MISRA-C:1998 가이드라인은 93개의 필수 규칙과 34개의 권고 규칙으로 구성되어 있으며, 이는 MISRA-C:2004 가이드라인에서 121개의 필수 규칙과 20개의 권고 규칙으로 확장되었다.

C++의 코딩 규칙으로는 구글에서 정의한 C++ 스타일 가이드를 들 수 있다[16]. C++는 구글의 많은 오픈소스 프로젝트에 사용되는 주요 개발 언어로 아주 많은 강력한 특징을 가지고 있으나 이러한 특징들은 프로그램을 상당히 복잡하게 만들며, 더 많은 버그를 발생시켜 코드를 읽기 어렵고 유지 보수하기 힘들게 만든다. 구글의 C++ 스타일 가이드는 C++의 많은 고급 문법 중 특정 문법들의 사용을 권고하거나 제한함으로써 코드를 단순하게 유지하고, 이런 문법들을 사용함으로써 발생될 수 있는 일반적인 다양한 오류와 문제들을 피하도록 하고 있다. 구글의 C++ 스타일 가이드는 헤더 파일, 클래스, 이름짓기, 주석을 포함한 총 9개의 범주로 구성되어 있다.

Geosoft사에서는 Java 개발자 커뮤니티에서 보편적인 사용되는 Java 코딩 권고안들을 바탕으로 Java 코딩 지침을 발표하였다[17]. 2011년 1월 발표된 버전 6.3 문서가 가장 최신의 것으로 각 권고안들은 주제별로 그룹

화하여 적용할 수 있는 경우 및 적용사례들을 제시하고 있다. 이 지침에는 이름짓기, 주석, 들여쓰기를 포함한 총 9개의 범주로 구성되어 있다.

2. 코딩 스타일 지원 도구

지금까지 코딩스타일의 교육 및 검사를 위한 여러 가지 지원도구들이 개발되었다. 먼저 Li의 연구에서는 대학의 프로그래밍 수업에서 코딩 표준을 학습하고 수용하는 것에 대한 학생들의 설문 조사를 통해 코딩 표준에 대한 인식과 교육이 중요함을 언급하였다[8]. 코딩스타일에 대한 설문조사 결과 학생들은 소스코드의 가독성에 대해 중요하다고 인식은 하고 있으나 가독성 높은 프로그램을 작성하기 위해 코딩 표준을 준수하는 것은 어렵게 느끼고 있으며 잘 지키지 않는 것으로 조사되었다. 이 연구에서 Li는 프로그래밍 수업에서 전문적인 프로그래밍까지 전형적으로 사용되는 코딩 표준의 대표적인 지침을 소개하고 코딩 표준에 대한 숙련도를 높이기 위해서 학생들의 수준에 맞는 코딩 표준을 제시해야 한다고 주장하였다. 또한, 학생들의 소스코드에 대한 개별적 피드백을 제공하는 것이 코딩스타일 학습에 효과적이며 이를 위해 자동화된 도구 사용의 필요성을 주장하였다.

문양선의 연구에서는 인지심리 이론에 근거한 객체지향 설계 기법 및 코딩스타일 지침을 제안하였다[18]. 이 연구에서는 인지심리 이론 중 7±2 모듬(chunk)이론과 학생들을 대상으로 한 인지 실험을 통해 유도된 내포구조(nested structure) 한계 수 3을 바탕으로 객체지향 소프트웨어 성분(메소드, 클래스, 메시지, 클래스 상속구조)들의 구성 및 관계에 대한 지침들을 정의하고 이를 지원하는 도구를 개발한 바 있다. 7±2 모듬이론은 G. Greeno[19]와 R. Mayer[20]가 주장한 프로그램 의미의 인지 과정에 바탕을 두고 있다. 이 주장은 프로그램 이해과정을 모듬화(chunking)로 보고 있다.

황준하의 연구에서는 C 언어 교육과 정적 분석 도구 개발을 위해 Indian Hill C Style과 GNU Coding Standard, MISRA C Guideline을 종합하여 새로운 C 코딩스타일을 제안하고 CStyler라는 자동화된 C 코딩 스타일 검증기를 lex와 yacc를 이용해 개발하였다[21]. 이 연구에서 코딩 규칙의 분류 기준으로 문법적 요소뿐만 아니라 기능적 요소를 추가적으로 고려하여 1차적으로 문법적 요소를 기준으로 10개의 범주를 나누었으며, 2차적으로 기능에 따라 배치, 이름, 제어 흐름, 데이터 흐름과 같이 4가지 범주로 분류하였다. CStyler는 새롭게 정의된 규칙을 포함하여 총 137개의 코딩 규칙들을 검사한다. [그림 1]은 CStyler를 이용한 소스코드의 코딩 규칙 사용을 검증한 결과를 보여준다.

그림 1. CStyler의 코딩 스타일 검증 예[21]

그 외에도 지정훈 등은 C/C++언어로 작성된 소스코드의 코딩스타일을 검사하고 코딩스타일 점수를 얼굴 표정을 이용해 시각화하는 시스템인 StyleVisualizer를 개발하였고[9], 이를 확장하여 스타일아바타 시스템[10]을 제안하였다. StyleVisualizer와 스타일아바타 시스템은 프로그래밍 수업의 보조 도구로 활용하여 프로그래밍 수업을 듣는 학생들이 스스로 자신이 만든 소스코드의 코딩스타일을 검사할 수 있도록 해줌으로써 코딩스타일에 대한 흥미를 불러일으키고 프로그래밍 수업에서 교수자가 코딩스타일에 대한 개별적인 피드백을 해주지 않아도 자신의 코딩스타일을 평가할 수 있도록 해준다. 그러나 아바타(가상 캐릭터)의 얼굴 표정을 학생들이 오해할 수지가 있다는 단점이 있다.

III. 스타일직소 설계

스타일직소 시스템은 C/C++ 또는 Java 언어로 된 소스코드를 입력받아 주어진 코딩 규칙에 따른 준수 여부를 검사하여 정량화하고 그 결과를 퍼즐로 변환된 이미지로 시각화하는 자동화된 코딩스타일 검사 시스템이다. [그림 2]는 스타일직소 시스템의 구조를 보여준다.

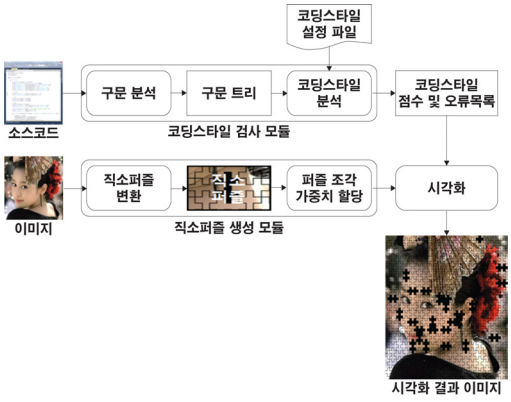


그림 2. 스타일직소 시스템 구성

스타일직소 시스템은 크게 코딩스타일 검사 모듈과 직소퍼즐 생성 모듈로 구성된다. 코딩스타일 검사 모듈은 사용자의 소스코드를 입력받아 미리 정의한 코딩스타일 설정파일을 이용하여 코딩 규칙을 얼마나 잘 준수하고 있는지를 검사하여 정량화한다. 그리고 직소퍼즐 생성 모듈은 시스템에서 제공하는 임의의 이미지를 퍼즐 조각으로 만들어 정량화된 코딩스타일 점수에 따라 시각화하여 사용자에게 보여준다.

1. 코딩스타일 검사 모듈

관련 연구에서 살펴본 여러 코딩 규칙들은 대부분 특정 언어에 대한 구체적인 코딩 규칙을 제시하고 있다. 하지만 프로그래밍을 처음 배우게 되는 단계에서는 특정 언어에 대한 코딩 규칙보다는 코딩스타일에 대한 개념 및 중요성, 언어와 관계없이 적용될 수 있는 코딩 요소에 관한 내용을 다룰 필요가 있다. Oman과 Cook은 코딩스타일을 구성하는 요소들을 기능적인 범주로 분류하는 분류법을 제시하였다[22]. 이 분류법에서는 코딩스타일 요소를 일반 범주(General Practices), 편집 스타일(Typographic Style), 제어구조 스타일(Control Structure Style), 정보구조 스타일(Information Structure Style)의 4가지 범주로 나누고, 각각의 범주에 대한 정의, 분류 기준, 하위클래스 목록 그리고 스타일 요소의 응용 예제들을 제시하였다. 각 범주에 대한 정의는 [표 1]과 같다.

표 1. Oman과 Cook의 스타일 범주 정의

범주	정의 및 세부 적용 항목
일반	프로그램의 스타일에 직접적으로 영향을 주는 프로그래밍 프로세스와 관련된 규칙(설계 기법, 언어 선택 및 제약, 테스트 및 디버깅 실행, 유지보수 등)
편집	프로그램 실행에 영향을 주지 않고, 코드 레이아웃과 주석에 영향을 주는 스타일 특성(전체 프로그램 형식, 전체 및 모듈 간 주석, 식별자 및 레이블 이름, 문장 형식, 들여쓰기 등)
제어 구조	제어구조의 선택 및 프로그램 또는 시스템을 구성하는 알고리즘 구현 방법에 관련된 스타일 특성(모듈 분할 및 결합, 모듈 재사용과 은닉, 구조화, 제어흐름 및 예외처리 등)
정보 구조	데이터구조와 데이터흐름 기법의 선택과 사용에 관련된 스타일 특성(전역 데이터 구조, 입출력 구조, 모듈 간 통신, 변수 초기화, 변수의 사용 제약 등)

일반 범주에 속하는 스타일 요소는 프로그램이나 시스템의 개발 전 과정에 걸친 것으로 코딩 전에 충분히 문제를 이해하고 정의해야 할 것, 디버깅과 데이터 테스트 기법을 수립할 것 등의 지침을 포함한다. 편집 범주는 프로그램의 실행에 영향을 주지 않는 스타일 요소로 한 줄에 하나 이상의 제어문을 쓰지 말 것, 의미가 있는 식별자 이름을 사용할 것, 명료성과 가독성을 높이기 위해 괄호를 사용할 것 등의 지침을 포함한다. 제어구조 범주는 프로그램의 제어구조 사용에 관한 것으로 세부 지침으로 라이브러리 함수를 사용할 것, 불필요한 goto문 사용을 제한할 것 등이 있다. 마지막으로 정보구조 범주는 데이터 구조와 흐름에 관한 것으로 전역 변수 사용을 피할 것, 변수 사용 전 초기화 할 것 등의 세부 치침을 포함한다.

지금까지 제시된 코딩스타일 표준들은 들여쓰기, 명명법, 주석과 같은 편집스타일 규칙뿐만 아니라 제어흐름이나 데이터 흐름과 같은 고수준의 프로그래밍 개념이 필요한 규칙들도 많이 포함하고 있다. 특히 [표 1]에 설명된 일반 범주에 속하는 지침들은 프로그램 설계

기법이나 언어 선택 등과 같이 실제 소스 코드 작성 외의 요소도 포함되어 있으므로 특정 프로그래밍 언어의 교육과 학습의 범위를 벗어난다고 할 수 있다. 앞서 언급한 Li의 연구에서처럼 이러한 고수준의 규칙들은 프로그래밍 언어를 처음 시작하는 학생들의 수준에는 다소 어려울 수 있으며 이 때문에 코딩 규칙을 준수하는 것 자체를 꺼리게 만들 수도 있다. 따라서 이 논문에서는 코딩스타일 분석 항목을 편집스타일에 관련된 요소로 제한한다. [표 2]는 이 논문에서 정의한 코딩스타일 지침을 정리한 것이다.

표 2. 스타일직소 시스템이 지원하는 코딩 지침

분류	코딩 지침
배치 규칙	복합문의 내부코드 들여쓰기 catch문은 들여쓰기 규칙 예외 do-while문의 while문은 들여쓰기 규칙 예외 제어문과 반복문의 시작 중괄호는 명령어와 같은 들여쓰기로 명령어의 다음 줄에 배치 제어문과 반복문의 종료 중괄호는 명령어와 같은 들여쓰기로 제어문과 반복문의 마지막 문장 다음 줄에 배치
문장 규칙	if문의 조건식에 대입문 사용하지 않기 goto문 사용 금지 제어문, 반복문에서 중괄호 반드시 사용
명명 규칙	함수와 변수명의 길이는 3-31 글자로 제한 지역변수와 매개변수의 경우, 변수 명의 첫 문자는 소문자
공백 규칙	대입문 '=' 양쪽에 공백 추가 이항 연산자는 피 연산자와 공백으로 분리 중괄호 앞 공백 문자 사용 쉼표 다음 공백 분자 사용 다중 템플릿 선언시 공백문자 사용

스타일직소에서는 [표 2]에 나타난 코딩지침의 여러 항목 중에서 사용자가 검사항목을 선택할 수 있도록 [그림 3]과 같은 인터페이스를 제공한다. 들여쓰기 단위나 중괄호 위치와 같은 세부 항목들을 사용자가 선택할 수 있도록 함으로써 프로그래밍 수업에서 좀 더 유연하게 코딩지침을 적용할 수 있도록 하였다.

구문분석 모듈은 C/C++ 또는 Java 프로그램 소스코드를 입력받아 지정된 코딩 지침에 따라 소스코드의 구문을 해석하고 코딩 규칙 위배 여부를 판별한다. 배치 규칙이나 공백규칙을 검사하기 위해 공백이나 탭의 개수를 세야 하므로 일반적인 파서와는 다르다. 이때 코딩스타일 점수는 코딩 지침을 준수한 항목 수와 전체 검사 항목 수의 비율로 계산된다. 코딩스타일 검사 모듈은 최종적으로 소스코드의 코딩스타일 점수와 코딩스타일 위반 정보를 출력한다.

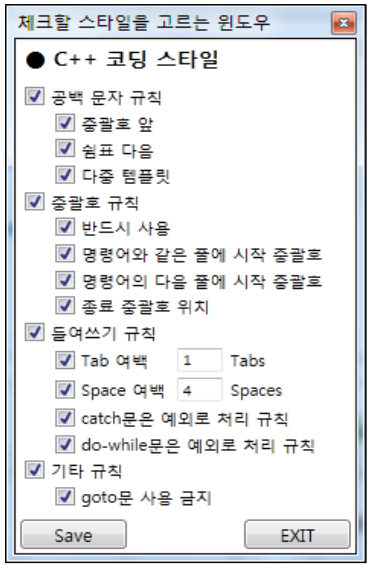


그림 3. 실제 적용할 코딩 지침을 선택할 수 있는 스타일직소 인터페이스

2. 직소퍼즐 생성 모듈

스타일직소 시스템은 소스코드의 코딩스타일 준수 여부를 코딩스타일 위반정보와 함께 코딩스타일 점수를 퍼즐로 변환된 이미지로 시각화한다. 이때 코딩스타일 점수가 높을수록 많은 퍼즐 조각이 나타나게 된다. [그림 4]는 퍼즐로 변환된 전체 이미지를 보여준다.

(a) 원본 이미지 (b) 직소퍼즐 이미지

그림 4. 직소퍼즐 형태로 변환된 이미지

스타일직소 시스템에서는 인물사진을 이용하여 코딩스타일 점수를 시각화한다. 최종 시각화 단계에서 소스코드의 코딩스타일 점수에 따라 퍼즐로 쪼개진 이미지의 각 부분이 나타나게 된다. 이때, 사용자의 흥미를 지

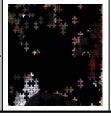
속시키기 위해서는 이미지의 중요 부분이 높은 점수가 나올 때까지 노출되지 않아야 한다. 일반적으로 사진의 중심부분에 중요한 부분이 놓일 확률이 높으며 특히 인물사진은 얼굴 부분이 가장 중요한 부분이라고 할 수 있다. 따라서 이 논문에서는 퍼즐 조각이 노출되는 순서를 정할 때 각 퍼즐 조각의 위치와 색상을 고려하여 가중치를 부여한다. 즉, 퍼즐 조각의 위치가 이미지의 가운데일수록, 픽셀의 색이 피부색에 가까울수록 높은 가중치를 부여한다.

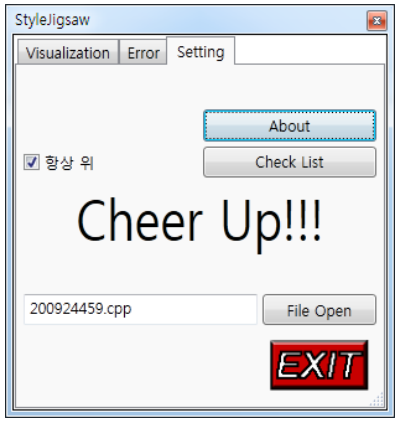
전체 이미지에서 피부색인 픽셀을 검출하기 위하여 기존에 제안된 다양한 피부 색상 모델을 이용할 수 있다[23-24]. 이 논문에서는 일반적으로 많이 사용되는 Chai 피부 색상 모델을 사용하였다[25]. Chai 모델에서는 이미지 픽셀값을 $YCbCr$ 색상 모델로 변환하고 식 1과 같이 피부 픽셀을 판별한다.

$$\begin{aligned} 136 \le Cr \le 156 \\ 110 \le Cb \le 123 \end{aligned} \qquad (1)$$

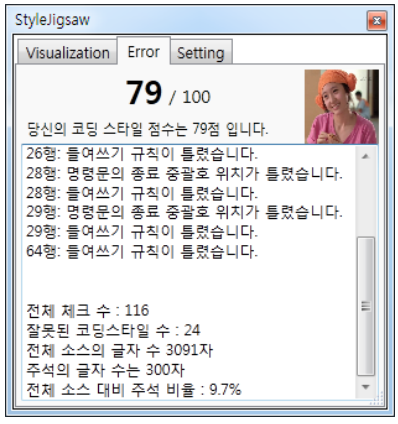
시각화 단계에서 코딩스타일 점수에 따라 나타나는 퍼즐 조각의 수가 결정되고, 가중치가 낮은 퍼즐 조각의 순서대로 시각화된다. 가중치가 같은 조각이 여러 개 있을 경우에는 무작위로 한 조각을 선택한다. [표 3]은 코딩스타일 점수에 따라 시각화된 결과 이미지를 보여준다. 가중치를 적용하지 않은 경우는 낮은 점수에서도 얼굴 부분의 퍼즐 조각이 보이는 것을 알 수 있다. 그에 비해 퍼즐 조각의 가중치에 따라 노출 순서를 달리 한 경우는 높은 점수를 받을 때까지 얼굴 부분의 퍼즐 조각이 노출되지 않음을 알 수 있다.

표 3. 코딩스타일 점수에 따른 결과 이미지 퍼즐 조각의 노출 정도

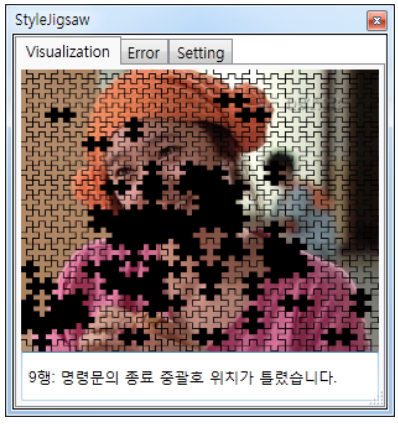
구분	25점	50점	75점	100점
무작위				
가중치 적용				



(a) 소스코드 파일 선택 및 코딩스타일 지침 설정



(b) 코딩스타일 위반 목록



(c) 코딩스타일 점수 시각화

그림 5. 스타일직소 시스템 인터페이스

3. 스타일직소 시스템 구현 및 인터페이스

스타일직소는 자동화 코딩스타일 검사 시스템으로 프로그래밍 수업에서 보조도구로 활용할 수 있도록 간단한 GUI 인터페이스를 제공한다. 스타일직소 시스템은 Visual C# 언어로 Visual Studio 10.0버전으로 구현되었으며 Windows XP와 Windows 7 환경에서 동작한다. 그리고 C/C++, Java 언어로 작성된 소스코드의 코딩스타일 검사를 지원한다.

스타일직소 시스템의 인터페이스는 총 3개의 탭으로 구성되어 있으며, 코딩스타일 분석 결과를 두 가지 방법으로 시각화한다. 첫 번째는 스타일점수에 따라 퍼즐로 변환된 이미지이고, 두 번째는 코딩 규칙에 따르지 않은 항목의 목록이다. [그림 5]는 스타일직소 시스템의 인터페이스를 보여준다.

[그림 5](a)는 '설정' 탭으로 코딩스타일을 검사할 소스코드를 불러오기 위한 탭이다. 또한 'check list' 버튼을 클릭하면 [그림 3]과 같이 적용할 코딩 규칙의 세부사항들을 선택할 수 있다. 이렇게 불러온 소스코드의 코딩스타일 검사 결과는 [그림 5](b)의 '오류' 탭으로 볼 수 있다. 소스코드의 코딩스타일을 검사 후 전체 코딩스타일 점수와 코딩 규칙에 따르지 않은 항목들을 라인별로 목록으로 보여주고, 목록의 마지막에 전체 검사 항목 수와 위반 항목 수 등의 간단한 통계정보를 보여준다. 마지막으로 [그림 5](c)는 계산된 코딩스타일 점수에 따른 퍼즐로 변환된 이미지 조각들을 보여준다. 앞서 설명한 바와 같이 점수가 높을수록 많은 조각들이 노출되며 이미지의 중요 부분들은 높은 점수일 때만 나타난다.

스타일직소 시스템은 사용하기 쉬운 GUI 인터페이스를 제공하므로 학생들은 쉽게 자신의 소스코드의 코딩스타일을 검사하고 또 수정할 수 있다.

IV. 실험 및 결과

이 논문에서는 스타일직소를 이용한 코딩스타일 교육의 효과를 평가하기 위하여 실제 프로그래밍 수업에 적용하여 학생들의 과제 결과물을 대상으로 스타일직소 사용 전후의 코딩스타일 점수를 비교하였다.

먼저 스타일직소 사용여부에 따른 스타일점수 향상 정도를 평가하기에 앞서 교수자의 설명과 유인물로만 코딩스타일을 학습한 학생들의 코딩스타일 점수 변화를 살펴보았다. 조사 데이터는 2011학년도 2학기 C++ 프로그래밍 수업을 수강한 29명의 학부 2학년들이 제출한 과제물이며 [표 4]에 정리되어 있다.

표 4. 교수자의 설명과 유인물로만 코딩스타일에 관해 학습한 학생들이 제출한 소스코드 데이터(2011년 2학기 C++ 프로그래밍 수강생의 과제물)

항목	제출인원	코딩스타일 평균 점수
과제 1	23	65.4
과제 2	25	68.4
과제 3	26	68.5
과제 4	26	70.5

한 학기에 걸친 코딩스타일 점수 변화를 살펴보기 위해서 전체 수강생 29명 중에서 총 4회의 과제 중 3회 이상 과제를 제출한 26명이 제출한 소스코드를 실험 데이터로 사용하였다. 제출된 소스코드의 코딩스타일 점수를 측정한 결과 코딩스타일 평균 점수는 약 65점에서 70점 사이로 나타났으며, 학기 초에 제출한 과제 1보다 학기말에 제출한 과제 4의 평균 코딩스타일 점수가 65.4점에서 70.5점으로 약 5.1점 향상된 것으로 나타났다. 이는 학기 초 점수 65.4점 기준 7.8% 향상된 것이다.

다음으로 스타일직소의 코딩스타일 교육 효과를 살펴보기 위해 프로그래밍 수업에서 스타일직소를 적용한 후 코딩스타일 점수 변화를 살펴보았다. 실험은 2012년 1학기 C++ 프로그래밍 수업을 수강한 26명의 컴퓨터공학과 학부 2학년을 대상으로 시행되었다. 중간고사 이전까지는 교수자가 코딩 규칙에 관해 설명과 유인물로만 가르친 후 학생들에게 프로그래밍 과제를 작성하도록 하였으며, 중간고사 이후에는 스타일직소를 사용하여 과제를 제출하기 전에 학생들 스스로 과제의 코딩스타일을 검사하도록 하였다.

실험을 위해 학생들에게 중간고사 이전과 이후에 각각 두 번의 과제를 부과하였다. 실험 데이터는 [표 5]에

정리되어 있다. 앞선 실험과 마찬가지로 총 4회의 과제 중에서 3회 이상 제출한 학생들의 과제물을 대상으로 코딩스타일 점수 변화를 살펴보았다. 2012학년도 1학기의 경우 과제를 제출한 학생의 수가 적었으며, 총 26명의 학생 중 18명이 3회 이상 과제를 제출하였다. 이 학생들이 제출한 과제의 코딩스타일 평균 점수는 학기 초보다 학기 말로 갈수록 점차 상승하였고 학기 초 보다 학기 말에 약 13.1점 상승한 것으로 나타났다. 이는 학기 초 점수 기준 18.7% 상승한 것이다.

표 5. 스타일직소를 사용하여 코딩스타일을 교육하였을 때 학생들이 제출한 소스코드 데이터(2012년 1학기 C++ 프로그래밍 수강생의 과제물)

항목	제출인원	코딩스타일 평균 점수
과제 1	18	70.0
과제 2	14	73.2
과제 3	13	73.3
과제 4	15	83.1

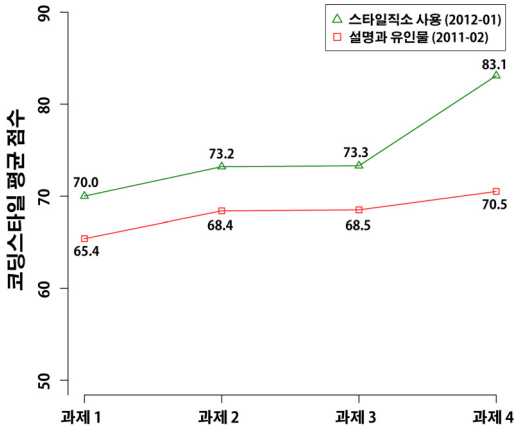


그림 6. 코딩스타일 학습 방법에 따른 코딩스타일 점수 변화

[그림 6]은 코딩스타일 학습 방법에 따른 코딩스타일 점수 변화를 보여준다. 그림에서 코딩스타일 학습에서 설명과 유인물을 이용한 경우와 스타일직소를 사용한 경우 모두 학기 초보다 학기 말에 코딩스타일 점수가 증가한 것을 알 수 있다. 실험 결과를 통해 스타일직소와 같은 코딩스타일 학습 도구를 사용하지 않더라도 프로그래밍 수업시간에 코딩 규칙 준수에 대한 중요성을 지속적으로 설명하고 프로그래밍 경험 증가함에 따라 학생들의 코딩스타일이 어느 정도는 개선되는 것으로 생각할 수 있다. 그러나 수업시간이 한정되어 있어 코딩스타일 교육에 많은 시간을 할애하기 어렵고, 학생들의 과제에 일일이 코딩스타일에 대한 피드백을 해 주는 것 또한 많은 부담이 된다.

스타일직소를 사용한 경우는 학생들의 코딩스타일 평균 점수가 설명과 유인물로 학습한 경우보다 코딩스타일 평균 점수가 더 많이 상승하였다(8.0점, 10.9%). 따라서 스타일직소를 이용한 코딩스타일 교육이 기존의 설명과 유인물을 통한 교육보다 더 효과적이라고 할 수 있다. 또한 교수자가 따로 시간을 할애하여 학생들의 코딩스타일에 대한 피드백을 해주지 않아도 학생들 스스로 코딩스타일을 검사하고 코딩스타일을 개선할 수 있다는 점이 스타일직소의 장점이라고 할 수 있다.

다음으로 프로그래밍 수업의 수강생들을 대상으로 코딩스타일에 대한 인식과 스타일직소 시스템에 대한 평가를 위해 설문조사를 실시하였다. 설문조사 문항은 [표 6]과 같다.

표 6. 코딩스타일에 대한 인식과 스타일직소 시스템에 대한 평가를 위한 설문조사 문항

문항	내용
1	코딩스타일은 얼마나 중요하다고 생각합니까?
2	스타일직소가 코딩스타일을 이해하는데 얼마나 도움이 되었습니까?
3	코딩스타일에 대한 스타일직소의 접근 방법은 얼마나 흥미로웠습니까?

설문지는 총 3개의 문항으로 구성되어 있으며 첫 번째 문항은 수강생들의 코딩스타일에 대한 인식을 조사하기 위한 것이고, 나머지 두 문항은 스타일직소의 목표인 교육적 의도에 부합하는지 알아보기 위한 것이다. 각 문항에 대한 답변은 각 5개의 항목으로 선택하게 하였다.

먼저 첫 번째 문항에 대한 학생들의 답변은 [표 7]과 [그림 7]에 나타나 있다. 코딩스타일의 중요성에 대해 전체 26명 중 5명의 학생은 “매우 중요하다”고 답변하였고, 나머지 21명의 학생은 “중요하다”로 답변하여 모

든 학생이 코딩스타일의 중요성은 인식하고 있는 것으로 나타났다.

표 7. "코딩스타일은 얼마나 중요하다고 생각합니까?" 에 대한 학생들의 답변

응답	학생수(명)	비율(%)
매우 중요하다	5	19.2
중요하다	21	80.8
중요하지 않다	0	0.0
전혀 중요하지 않다	0	0.0
기타	0	0.0
합계	26	100.0

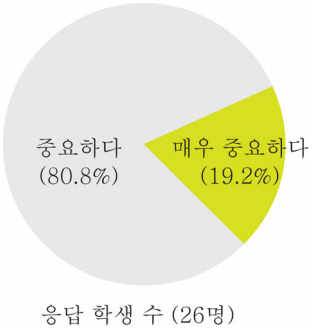


그림 7. 코딩스타일의 중요성에 대한 응답 결과

다음으로 두 번째 문항인 "스타일직소가 코딩스타일을 이해하는데 얼마나 도움이 되었습니까?"에 대한 학생들의 답변이 [표 8]과 [그림 8]에 나타나 있다. 스타일직소의 사용이 코딩스타일을 이해하는 데 도움이 되었다고 응답한 학생은 23명으로 전체의 88.5%로 나타났고, 나머지 3명의 학생은 도움이 되지 않았다고 응답하였다. 따라서 대부분의 학생에게 스타일직소가 코딩스타일 학습에 도움이 되는 것을 알 수 있다.

표 8. "스타일직소가 코딩스타일을 이해하는데 얼마나 도움이 되었습니까?" 에 대한 학생들의 답변

응답	학생수(명)	비율(%)
매우 도움이 되었다.	6	23.1
도움이 되었다.	17	65.4
도움이 되지 않았다.	3	11.5
전혀 도움이 되지 않았다.	0	0.0
기타	0	0.0
합계	26	100.0

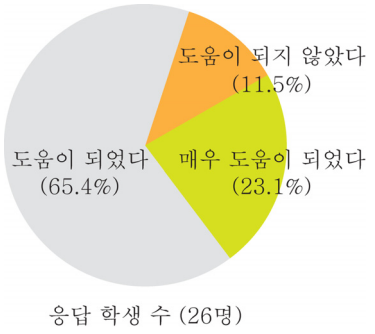


그림 8. 스타일직소의 효과에 대한 응답 결과

마지막으로 "코딩스타일에 대한 스타일직소의 접근 방법은 얼마나 흥미로웠습니까?"라는 질문에 대한 학생들의 응답은 [표 9]와 [그림 9]에 나타나 있다. 스타일직소를 통한 코딩스타일 교육이 흥미로웠다고 응답한 학생은 22명으로 전체의 84.6%를 차지하였고, 흥미롭지 않았다고 응답한 학생이 3명, 기타 응답이 1명이었다. 수강생들의 대부분이 스타일직소 사용에 흥미를 느낀 것으로 응답하여 스타일직소 시스템이 학생들에게 코딩스타일에 관한 관심을 불러일으키는 데 효과적임을 알 수 있다.

표 9. "코딩스타일에 대한 스타일직소의 접근 방법은 얼마나 흥미로웠습니까?" 에 대한 학생들의 답변

응답	학생수(명)	비율(%)
매우 흥미로웠다.	3	11.5
흥미로웠다.	19	73.1
흥미롭지 않았다.	3	11.5
전혀 흥미롭지 않았다.	0	0.0
기타	1	3.9
합계	26	100.0

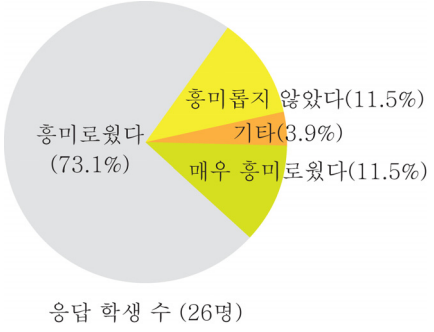


그림 9. 스타일직소 시스템의 흥미도에 대한 응답 결과

설문조사를 통해 학생들의 대부분은 코딩스타일의 중요성을 인식하고 있으며, 스타일직소를 통한 코딩스타일 교육이 유인물이나 설명을 통한 교육보다 더 효과적임을 알 수 있었다. 따라서 스타일직소 시스템을 프로그래밍 수업의 보조도구로 활용한다면 더욱 효과적으로 코딩스타일을 교육할 수 있을 것이라고 생각한다.

V. 결론 및 향후 연구

이 논문에서는 C/C++와 Java로 작성된 소스코드의 코딩스타일을 검사하고 시각화하는 스타일직소 시스템을 제안하였다. 스타일직소 시스템은 기존에 제안된 코딩스타일 시각화 시스템인 StyleVisualizer의 단점을 보완하여 완성되지 않은 소스코드의 코딩스타일 검사가 가능하며, 소스코드의 어느 부분에 어떤 코딩 규칙을 위배했는지에 대한 정보를 보여준다. 따라서 사용자는 프로그램을 작성하는 중간에 자신의 코딩스타일을 점검하고 잘못된 부분을 바로 수정할 수 있다.

스타일직소 시스템이 코딩스타일 교육에 얼마나 효과적인지를 알아보기 위해 실제 컴퓨터공학과 학부 2학년의 C++ 프로그래밍 수업에 보조도구로 활용하여 코딩스타일 교육을 실시하였다. 실험 결과 스타일직소 시스템을 사용한 경우가 사용하지 않고 교육한 경우에 비해 코딩스타일 평균 점수가 약 8.0점(10.9%)점 가량 향상된 것으로 나타나 스타일직소가 코딩스타일 교육에 효과적인 것을 알 수 있었다.

그리고 학생들의 코딩스타일에 대한 인식과 스타일직소 시스템에 대한 반응을 알아보기 위한 설문조사를 하였다. 설문조사 결과 수강생 전원은 코딩스타일의 중요성에 대해 인식하고 있는 것으로 응답하였으며, 약 88.5%의 학생이 스타일직소 시스템이 코딩스타일 교육에 도움이 된다고 응답하였다. 또한, 84.6%의 학생이 스타일직소 시스템의 사용에 흥미를 보인 것으로 조사되었다.

스타일직소 시스템은 사용하기 쉬운 인터페이스를 제공하므로 학생들 스스로 자신이 만든 소스코드의 코딩스타일을 검사하고 수정할 수 있으며, 따라서 교수자가 많은 시간을 할애하지 않아도 코딩스타일의 지속적 피드백이 가능하다. 또한, 학생들의 수준에 맞는 코딩 지침을 선택할 수 있기 때문에 수준별 코딩스타일 교육도 가능하게 해준다.

현재 스타일직소 시스템은 C/C++와 Java로 작성된 소스코드만 지원하고 있으므로 향후 좀 더 많은 프로그래밍 언어를 지원하도록 확장이 필요하며, 프로그램의 제어 구조에 관한 고수준의 코딩 지침도 검사할 수 있도록 코딩 규칙을 확장할 필요가 있다.

참 고 문 헌

[1] 이덕희, *디지털화와 산업의 양극화*, 삼성경제연구소, 2008.

[2] T. Tenny, “Program Readability: Procedures Versus Comments,” IEEE Trans. on Software Engineering, Vol.14, No.9, pp.1271-1279, 1988.

[3] http://www.cwu.edu/~gellenbe/javastyle/

[4] http://www.gnu.org/prep/standards/standards.html

[5] H. Sutter and A. Alexandrescu, *C++ Coding Standards: 101 Rules, Guidelines, and Best Practices (C++ in Depth Series)*, Addison-Wesley Professional, 2004.

[6] X. Fang, “Using a Coding Standard to Improve Program Quality,” in Proc. of the 2nd Asia-Pacific Conference on Quality Software, pp.73-78, IEEE, 2011.

[7] T. Howles, “Fostering the Growth of a Software Quality Culture,” ACM SIGCSE Bulletin, Vol.35, No.2, pp.45-47, 2003.

[8] X. Li and C. Prasad, “Effectively Teaching Coding Standards in Programming,” in Proc. of the 6th Conference on Information Technology Education, pp.239-244, 2005.

[9] 지정훈, 이윤정, 우균, “얼굴 표정을 이용한 코딩스타일 점수 시각화”, 정보과학회논문지: 소프트웨어 및 응용, 제37권, 제7호, pp.578-583, 2010.

[10] J. S. Lim, J. H. Ji, Y. J. Lee, and G. Woo, "Style Avatar: a Visualization System for Teaching C Coding Style," in Proc. of the 2011 ACM Symposium on Applied Computing, pp.1210-1211, ACM, 2011.

[11] I. J. Jung, Y. J. Lee, B. H. Chun, E. K. Kim, and G. Woo, "Design and Implementation of Coding Style Jigsaw Game for C Learners," in Proc. of 2011 International Digital Design Invitation Exhibition, pp.61-62, 2011.

[12] L. W. Cannon, R. A. Elliott, L. W. Kirchoff, J. H. Miller, R. W. Mitze, E. P. Schan, N. O. Whittington, H. Spencer, D. Keppel, and M. Brader, *Recommended C Style and Coding Standards*, AT&T Bell Labs. 1990.

[13] MISRA Ltd., *Guidelines for the use of the C language in vehicle based software*, The Motor Industry Software Reliability Association, 1998.

[14] MISRA Ltd., *Guidelines for the use of the C language in critical systems*, The Motor Industry Software Reliability Association, 2004.

[15] L. Hatton, "Language Subsetting in an Industrial Context: A Comparison of MISRA C 1998 and MISRA C 2004," J. of Information Software Technology, Vol.49, No.5, pp.475-482, 2007.

[16] http://google-styleguide.googlecode.com/svn/trunk/cppguide.xml

[17] http://geosoft.no/development/javastyle.html

[18] 문양선, 유철중, 장옥배, "인지심리 이론을 반영한 객체지향 설계 및 프로그래밍 스타일 지침", 정보과학회논문지(B) 제25권, 제3호, pp.530-542, 1998.

[19] J. G. Greeno, "The Structure of Memory and the Process of Solving Problem," in Contemporary Issues in Cognitive Psychology, pp.23-45, 1973.

[20] R. Mayer and B. Shnejderman, "Syntactic/Semantic Interactions in Programmer Behavior: Model and Experimental Result," J. of Computer and Information Sciences, Vol.8, No.3, pp.213-238, 1979.

[21] 황준하, "C 코딩스타일 검증기의 설계 및 구현", 한국컴퓨터정보학회논문지, 제13권, 제2호, pp.31-40, 2008.

[22] P. W. Oman and C. R. Cook, "A Taxonomy for Programming Style," in Proc. of the 1990 ACM annual conference Cooperation, pp.244-250, ACM, 1990.

[23] 이윤정, 김영봉, "컬러 정보를 이용한 실시간 표정 데이터 추적 시스템", 한국콘텐츠학회논문지 제9권, 제7호, pp.159-170, 2009.

[24] 윤진성, 김계영, 최형일, "피부색상을 이용한 유해영상 분류기 개발," 한국콘텐츠학회논문지, 제9권, 제4호, pp.1-11, 2009.

[25] D. Chai and K. N. Ngan, "Face Segmentation Using Skin-color Map in Videophone Application," IEEE Trans. on Circuit and Systems for Video Technology, Vol.9, No.4, pp.551-564, 1999.

저 자 소 개

이 윤 정(Yun-Jung Lee) 정회원

- 1995년 2월 : 부경대학교 전자계산학과(이학사)
- 1999년 2월 : 부경대학교 전산정보학과(이학석사)
- 2008년 8월 : 부경대학교 전자계산학과(이학박사)
- 2008년 9월 ~ 2012년 8월 : 부산대학교 U-Port 정보기술사업단 박사후연구원
- 2012년 9월 ~ 현재 : 부산대학교 컴퓨터 및 정보통신연구소 기금교수

<관심분야> : 얼굴 애니메이션, 웹 콘텐츠 시각화, 사회연결망 분석

정 인 준(In-Joon Jung) 준회원

- 2010년 8월 : 부경대학교 컴퓨터멀티미디어공학과(이학사)
- 2012년 8월 : 부산대학교 컴퓨터공학과 공학석사
- 2012년 12월 ~ 현재 : eBay 코리아

<관심분야> : 데이터 클러스터링, 코드 시각화

우 균(Gyun Woo) 정회원

- 1991년 : 한국과학기술원 전산학(학사)
- 1993년 : 한국과학기술원 전산학(석사)
- 2000년 : 한국과학기술원 전산학(박사)
- 2000년 ~ 2002년 : 동아대학교 컴퓨터공학과 전임강사
- 2002년 ~ 2004년 : 동아대학교 컴퓨터공학과 조교수
- 2005년 ~ 2012년 8월 : 부산대학교 컴퓨터공학과 부교수
- 2012년 9월 ~ 현재 : 부산대학교 컴퓨터공학과 교수

<관심분야> : 프로그래밍언어 및 컴파일러, 함수형 언어, 그리드컴퓨팅, 소프트웨어 메트릭, 프로그램 시각화