

맥락 정보를 활용한 모바일 접근성 진단

및 코드 수정 제안 시스템

오종현¹, 한지훈¹, 정유진², 오찬세¹, 박상근¹

¹경희대학교 소프트웨어융합학과, ²산업경영공학과

{sjaiz1717, jhhan0208, yujin010917, chance0727, sk.park}@khu.ac.kr

A Context-Aware System for

Mobile Accessibility Diagnosis and Code Revision Suggestions

Jonghyun Oh¹, Jihoon Han¹, Yujin Chung², Chanse Oh¹, Sangkeun Park¹

¹Department of Software Convergence, Kyung Hee University

²Department of Industrial Management Engineering, Kyung Hee University

요약

모바일 앱 접근성은 일상생활의 필수 요소이자 기업의 사회적 책임으로 대두되고 있다. 특히 시각장애인은 스마트폰 스크린리더를 통해 정보를 인지하므로 이들의 접근성 보장은 필수적이다. 그러나 기존 관련 연구들은 코드 구문이나 시각적 정보 중 하나에만 의존하여, 풍부하고 정확한 대체 텍스트를 생성하는 데 한계가 있다. 본 연구는 이러한 문제를 해결하기 위해 LLM을 활용하여 Flutter 개발 환경에서 코드와 모바일 화면의 문맥을 동시에 심층 분석하는 접근성 문제 탐지 및 수정 시스템을 제안한다. Visual Studio Code 확장 프로그램으로 구현된 이 시스템은 모바일 UI 요소의 기능적 역할과 사용자의 상호작용 흐름까지 고려하여 최적의 대체 텍스트와 코드 수정안을 실시간으로 제공한다. 본 시스템은 개발자가 작업 흐름의 방해 없이 개발 초기 단계부터 앱 접근성 품질을 체계적으로 확보할 수 있도록 지원하여, 시각장애인의 모바일 사용 경험을 실질적으로 향상시키는 데 기여하고자 한다.

1. 서론

모바일 앱은 일상 생활을 영위하기 위한 필수 도구가 되었다. 쇼핑, 교통, 정보 검색 등 많은 부분이 앱을 통해 이뤄진다. 이러한 디지털 서비스에 대한 접근성(Accessibility)은 단순한 기술 문제가 아니라 기업의 사회적 책임이자 브랜드 평판에 직결되는 핵심 요소이다[1].

앞을 볼 수 없는 시각장애인에게 모바일 접근성은 단순 편의를 넘어 사회 참여와 직결되는 권리이다. 이들은 스마트폰에 기본 탑재된 스크린리더를 통해 앱의 콘텐츠를 음성으로 변환하여 정보를 인지하고 상호작용한다. 그러나 많은 모바일 앱이 시각장애인을 위한 접근성을 충분히 고려하지 않아, 시각장애인이 스크린리더를 통해 받아들이는 정보가 제한적이고 빈약한 문제가 발생한다. 접근성 개선에 관심이 있는 모바일 개발자도 있지만, 인식 부족, 전문성 부족, 적절한 도구의 부족으로 개선 작업이 쉽지 않은 상황이다 [2].

모바일 앱 개발 시 시각장애인을 위한 접근성 구현을 지원하는 다양한 연구가 수행되고 있다. 스크린리더가 모바일 콘텐츠를 정확하게 읽어줄 수 있도록 모바일 화면 이미지를 기반으로 적절한 대체 텍스트를

생성하거나[3, 4, 5], 접근성 문제를 탐지하고 문제 해결을 위한 코드까지 제안하는 연구 등[2, 6, 7]이 있다. 하지만 기존 연구는 시각적 정보에 의존하거나 개발 코드에 크게 의존하여 보다 풍부하고 정확한 대체 텍스트를 생성하는데 한계가 존재한다.

본 연구는 개발자가 복잡한 접근성 기능을 쉽고 정확하게 구현할 수 있도록 LLM을 활용하여 개발 환경 내에서 접근성 문제를 탐지하고, 코드와 모바일 화면의 문맥을 깊이 이해하여 최적의 대체 텍스트 및 코드 수정안을 제공하는 Visual Studio Code 확장 프로그램을 제안한다.

2. 관련 연구

2.1 요소의 시각적 특성 기반 대체 텍스트 생성 연구

스크린리더가 모바일의 UI 요소를 정확하게 이해하기 위해서는 접근성 표준에 따라 각 요소에 대체 텍스트 지정이 필수적이다. 개발자가 각 UI요소에 필요한 대체 텍스트를 입력하지 않았더라도 앱 이미지나 아이콘을 정확하게 이해하고 적절한 대체 텍스트를 자동 생성하기 위한 다양한 연구가 수행되었다. Chen et al.[3]은 아이콘 버튼의 시각적 특징을 분석하여 대체 텍스트를 생성하고, Srivatsan et al.[4]은 트위터 게시물의 이미지와 텍스트를 기반으로 의미있는 대체 텍스트 생성을 위한 멀티모달 모델을 개발했다. Zhong et al.[5]는 특정 요소의 보다 정확한 의미를 추론하기 위해 현재 화면의 이미지와 연결된 다른

*본 연구는 과학기술정보통신부 및 정보통신기획평가원의 2025년도 SW중심대학사업의 결과로 수행되었음(2023-0-00042)

화면의 이미지를 함께 학습에 활용하여 해당 요소를 보다 깊이 이해하고 이에 맞는 적절한 대체 텍스트를 생성하는 연구를 수행했다. 기존 연구는 모바일 요소의 시각적 특성에 크게 의존하므로, 생성된 대체 텍스트가 너무 단순한 정보만을 제공하거나 부정확한 한계를 보일 수 있다.

2.2 접근성 오류 탐지 및 코드 수정 제안

개발 환경에서 바로 코드를 분석하고 접근성 문제를 탐지하기 위한 다양한 시도도 존재한다. Mowar et al.[6]은 GitHub Copilot에서 실시간으로 코드를 분석하고 접근성 문제를 탐지하는 도구를 개발했다. 접근성 탐지에서 나아가, LLM을 활용해 적절한 접근성 문제 해결 코드를 제안하는 연구도 수행되었다[2, 7]. 이러한 도구는 접근성 문제 탐지 및 코드 제안이 코드의 문법 및 속성에 크게 의존하여, 모바일 화면 내용과 맥락을 반영하지 않아 대체 텍스트의 수준이 높지 않은 한계가 있다.

3. 접근성 문제 탐지 및 코드 수정 제안 시스템 개발

본 연구에서는 시각장애인 사용자의 모바일 앱 접근성을 실시간으로 점검하고, 문제 발생시 이를 개선할 수 있는 코드를 제안할 수 있는 Visual Studio Code 확장 프로그램¹을 개발했다. 본 시스템은 코드 분석 및 접근성 문제 탐지, 코드 수정 제안 생성, 코드 반영 및 검증의 세 단계를 순차적으로 수행하며 시각 장애인을 위한 접근성 문제를 해결한다. 이를 통해 개발자는 앱 개발 과정에서 접근성 오류를 쉽게 점검하고 수정하며 개발 초기 단계부터 접근성 품질을 체계적으로 확보할 수 있다.

3.1 코드 분석 및 접근성 문제 탐지

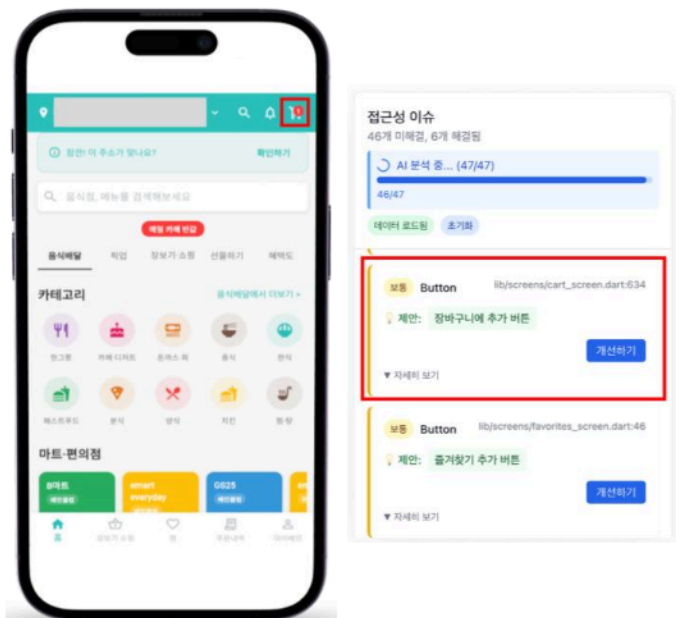


그림 1. 모바일 UI 요소의 접근성 문제 탐지 화면 (장바구니 아이콘 UI 요소의 접근성 문제 탐지 예시)

해당 시스템은 Flutter² 기반 모바일 앱 개발 환경에서 동작한다. 본 연구에서 개발한 시스템은 OpenAI의 GPT-4o LLM을 활용해 Flutter로 작성된 코드의 UI 위젯 패턴과 모바일 화면의 맥락을 함께 고려해서 각 요소가 어떤 기능을 수행해야 하는지 판단한다 [그림 1 (왼쪽)]. 이를 통해, 단순한 Role-Label 속성 누락 여부 판단이 아니라, 화면 흐름을 고려했을 때 기능이 코드에 적절히 반영되어 있는지 함께 평가한다. 예를 들어 [그림 1 (왼쪽)]의 장바구니 아이콘이 단순 이미지로 작성되어 Role과 대체 텍스트가 지정되지 않은 경우 스크린리더가 이를 버튼으로 인식할 수 없다. 본 시스템은 해당 요소와 관련된 코드를 LLM으로 분석하여 사용자가 해당 요소를 선택하면 '장바구니 페이지'가 활성화된다는 위젯 패턴을 추출한다. 이를 통해 해당 요소가 장바구니 버튼 역할을 수행함을 파악하고, "Button" 역할과 "장바구니에 추가 버튼"과 같은 적절한 대체 텍스트를 제안한다 [그림 1 (오른쪽)]. "개선하기" 버튼을 누르면 이러한 분석 결과는 구조화된 JSON으로 기록되며, 이후 코드 수정 제안 단계의 입력으로 활용된다.

3.2 접근성 문제 해결을 위한 코드 수정 제안

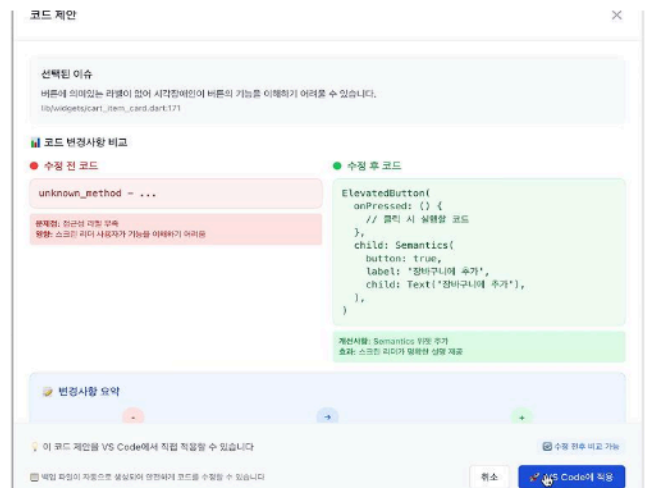


그림 2. Flutter 코드 수정 제안

3.1에서 추출된 접근성 이슈와 해당 UI 요소가 정의된 코드 일부, 그리고 이 요소와 연관된 주변 코드 맥락을 함께 입력으로 사용한다. 이러한 정보는 JSON 형태로 정리되어 모델에 전달되며, LLM은 이를 기반으로 해당 UI 요소가 화면에서 수행해야 할 기능을 해석한다. 이 과정에서 모델은 단순 속성 누락 여부를 넘어서, 요소가 어떤 상호작용 흐름 속에서 사용되는지를 고려하여 적절한 대체 텍스트와 역할 정보를 반영한 수정 코드를 제안한다[그림 2]. 이때 제안된 코드는 기존 UI 구조나 개발자의 작성 스타일을 최대한 보존하면서도 즉시 삽입 가능한 형태로 제공된다. 개발자가 해당 제안을 승인하면, 제안 내용이 실제 코드에 반영된다. 이를 통해 개발자는 별도의 코드 탐색이나 수정 작업 없이도, 시각장애인을 위한 모바일 접근성 품질을 체계적이고 효율적으로 개선할 수 있다.

¹ <https://www.youtube.com/watch?v=pdvzpfMuuQQ>

² <https://flutter.dev/>

3.3 제안 코드 적용 및 검증

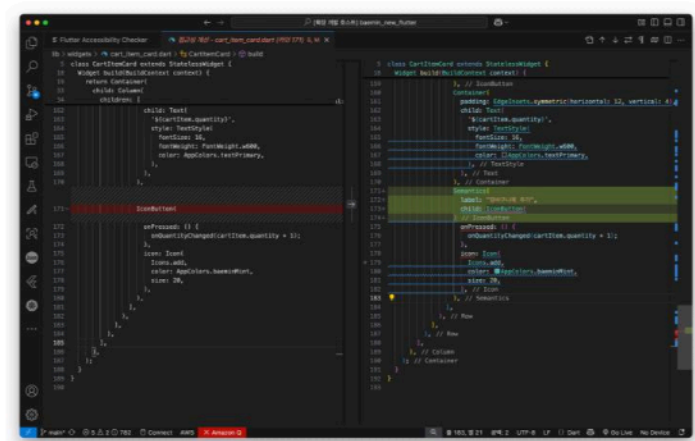


그림 3. 실시간 코드 수정

본 단계의 핵심은 LLM이 생성한 수정안을 원본 코드에 정확하게 삽입하는 것이다. 이를 위해 코드 위치를 찾는 과정을 수행하며 UI 위젯의 형태나 속성 패턴을 기반으로 주변 문맥을 탐색한 뒤, 원본 코드와 수정안의 문자열 유사도를 레벤슈타인 거리[8]로 계산해 가장 유사한 위치를 선택하는 방식을 채택하였다.

수정 위치가 확정되면, IDE는 원본 코드를 별도의 백업 파일로 저장한 뒤 수정안을 적용한다. 그 후 백업본과 수정된 파일을 VS Code 내부 문서 비교 방식인 diff 모드로 제시함으로써, 개발자가 변경 내용을 직관적으로 검토할 수 있도록 한다. 제안한 접근성 문제 해결 코드를 개발자가 승인하면, [그림3]과 같이 제안 코드를 적용하기 전(왼쪽)과 후(오른쪽)를 비교해 볼 수 있다.

4. 결론

본 연구에서는 LLM을 활용하여 모바일 앱 개발 과정에서 발생하는 접근성 문제를 실시간으로 탐지하고 최적의 코드 수정안을 제안하는 Visual Studio Code 확장 프로그램을 개발했다. Flutter 코드 분석과 모바일 화면의 문맥을 함께 고려함으로써, 단순한 속성 누락 검사를 넘어 사용자의 상호작용 흐름에 기반한 고품질의 대체 텍스트와 코드 개선안을 제안한다. 코드 수정 전후 비교 및 검증 기능까지 개발 환경에 자연스럽게 통합함으로써, 본 시스템은 개발자의 작업 흐름을 방해하지 않고 개발 초기 단계부터 접근성 품질을 체계적으로 확보하도록 지원한다. 이를 통해, 모바일 개발자의 접근성에 대한 인식 함양 및 장기적으로는 시각장애인의 모바일 접근성 향상 효과를 기대할 수 있다.

참고문헌

[1] Krok, Ewa. "Digital Accessibility as an Essential Element of an Inclusive Organization." European Research Studies Journal. Volume 27, Special Issue A (2024): 857-876.

[2] Mehralian et al. "Automated Code Fix Suggestions for Accessibility Issues in Mobile Apps." arXiv preprint arXiv:2408.03827 (2024).

[3] Chen et al. "Unblind your apps: Predicting natural-language labels for mobile gui components by deep learning." Proceedings of the ACM/IEEE 42nd international conference on software engineering. 2020.

[4] Srivatsan et al. "Alt-Text with Context: Improving Accessibility for Images on Twitter." ICLR 2024 Conference Paper, 2024.

[5] Zhong et al. "Caption: Generating Informative Content Labels for Image Buttons Using Next-Screen Context." Adjunct Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology. 2025.

[6] Mowar et al. "CodeA11y: Making AI Coding Assistants Useful for Accessible Web Development." Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems. 2025.

[7] Cali et al. "A Prototype VS Code Extension to Improve Web Accessible Development." 2025 IEEE/ACM Second IDE Workshop (IDE). IEEE, 2025.

[8] Sankoff, D., and J. B. Kruskal. "Binary Codes Capable of Correcting Deletions, Insertions, and Reversals." (1983).