C++ Style Checker for Visual Studio Code

Proof-of-Concept Implementation of Cevelop's C++ Style Checks in an LSP Language Server

Graduate

Marco Gartmann

Fabian Thurnheer

Examiner
Thomas Corbat

Co-Advisor
Guido Zgraggen, Google Switzerland, Zürich, ZH

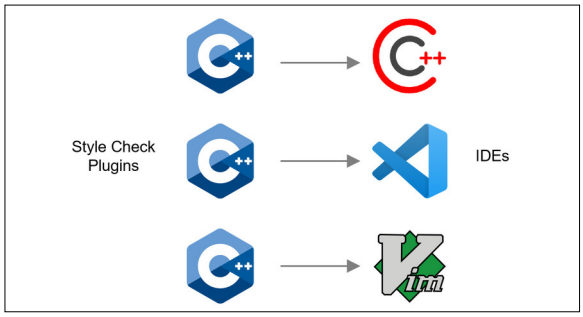
Subject Area
Software Engineering - Core Systems

Introduction: Programming in C++ usually relies on extensive toolchains for building, testing, and deploying applications. The fact that a given C++ source file successfully compiles does not imply that its code is of good quality and style. Moreover, it is not assured that agreed coding guidelines, which can be self-defined or well renowned, are met. Ensuring this by manual code reviews after the code passed through compilation and testing, results in a long feedback loop and negatively affects efficiency. To counteract this problem, Cevelop, a C++ IDE built by OST's Institute for Software (IFS), offers style checks that give programmers instant feedback on the code as they type. While some of these checks are implemented in other IDEs and plugins as well, many of them are exclusive to Cevelop and are not available in other IDEs like Microsoft's Visual Studio Code. However, this would be desirable in the future so that students using other IDEs than Cevelop can profit from these style checks as well.

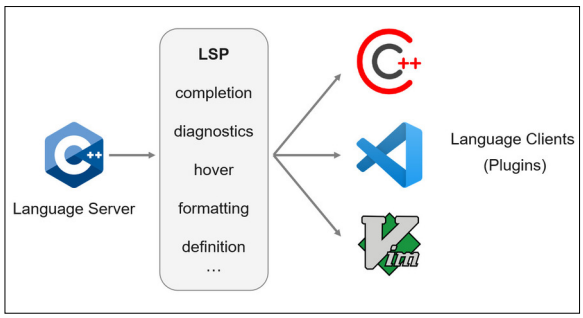
Approach: In this thesis, with the LLVM compiler project and its clang-tidy code analysis component, a feasible infrastructure was elaborated. Using this infrastructure, Cevelop's style checks and new ones can be implemented in the future to make them available in Visual Studio Code. Furthermore, after an analysis of offered style checks in Cevelop, selected checks were implemented as a proof-of-concept for LLVM's clang-tidy component using the C++ programming language. In the chosen approach, all the code analysis intelligence is encapsulated in an IDE-independent language server (LLVM clangd, which includes clang-tidy). To use the implemented style checks in an other IDE than Visual Studio Code, only a small plugin is needed to communicate with the language server through the Language Server Protocol (LSP). Therefore, they can also be offered in other IDEs with minimal additional effort.

Result: This thesis laid the foundation for offering Cevelop's style checking intelligence in IDEs independent of Cevelop. Thus, users of other IDEs can be reached, and more developers can be helped to write clean C++ code. The created checks were presented to the LLVM community to be integrated into the project's code base and to make them public. Until the created style checks are integrated, a self-built executable of LLVM's clangd language server, which includes clang-tidy and the created style checks, can be used with clangd's VS Code plugin. This way, the created checks could help OST students enlisted in a C++ course to write clean C++ code and to comply with taught best practices, without being bound to Cevelop. A created developer's guide assists programmers (e.g., IFS employees) to further extend clang-tidy with style checks that would be beneficial for them or for students.

Without LSP: Each IDE Needs its Own Style Checker Plugin
Own presentment



Using one Language Server with LSP to Support Multiple IDEs
Own presentment



Example of an Implemented Style Check and Corresponding Quick Fixes
Own presentment