Require a non-throwing default contract-violation handler

John Lakos (jlakos@bloomberg.net)

Document #:	P3577R0
Date:	2025-01-13
Project:	Programming Language C++
Audience:	SG21, EWG

Abstract

This paper proposes a one-line addition to require that the system-supplied default violation handler not throw.

1 Introduction

Some people are concerned that a new codepath might result if an *implicit* precondition — a precondition attached by the implementation to a core language expression, as proposed in [P3081R1], [P3100R1], and [P3329R0] — is violated and the contract-violation handler exits via an exception.

This paper addresses that problem by requiring that the *default* contract-violation handler does not exit via an exception. Thus, the new code path will not happen without an explicit replacement of the contract-violation handler by the user, which means it cannot happen inadvertently by accident or as the result of a port to a conforming implementation.

Note that this restriction can always be relaxed in a later iteration of C++ as a non-breaking change, but it cannot be so added.

2 Proposed wording

The proposed wording is relative to [P2900R13].

Modify [basic.contract.handler], paragraph 1 as follows:

> The *contract-violation handler* of a program is a function named `::handle_contract_violation`. The contract-violation handler shall take a single argument of type "lvalue reference to `const std::contracts::contract_violation`" and shall return `void`. The contract-violation handler may have a non-throwing exception specification. The implementation shall provide a definition of the contract-violation handler, called the *default contract-violation handler*. The default contract-violation handler shall not exit via an exception. [*Note:* No declaration for the default contract-violation handler is provided by any standard library header. — *end note*]

3 Summary

By adding this one line of wording, we eliminate a stated concern, and leave open all options moving forward. It is anticipated that this proposal will increase consensus, as it has no obvious or likely negative consequences.

Acknowledgements

Thanks to Timur Doumler for providing the wording.

Bibliography

[P2900R13] Joshua Berne, Timur Doumler, and Andrzej Krzemieński. Contracts for C++. https://wg21.link/p2900r13, 2025-01-13.

[P3081R1] Herb Sutter. Core safety profiles for C++26. https://wg21.link/p3081R1, 2025-01-06.

[P3100R1] Timur Doumler, Gašper Ažman, and Joshua Berne. Undefined and erroneous behaviour is a contract violation. https://wg21.link/p3100r1, 2024-10-16.

[P3329R0] Timur Doumler and Joshua Berne. Making erroneous behaviour consistent with Contracts. https://wg21.link/p3329r0, 2025-01-13.