Document: N4229
Date: 2014-10-13
Working Group: EWG
Reply-To: gdr@microsoft.com

Pointer Ordering

Gabriel Dos Reis

Abstract

This paper suggests a simple fix to an embarrassing glaring hole in the standard library section regarding pointer comparison function objects: that they yield the same result as the built-in comparison operators when the result is defined for the latter.

Introduction

The standard library section on function objects provides facilities for creating relational comparison function objects. In particular, it requires that these comparisons objects define a total ordering on pointers even when the corresponding built-in operators lack such properties. However, it remains loudly silent on the actual semantics relationship between these library facilities and the built-in operators. This paper fixes that semantics gap.

Proposed Wording

Append the following sentence to paragraph 20.9.5/14:

> **For specializations on pointer types, the call operator for each of these templates shall yield the same value as its corresponding built-in operator (5.9) when the result is defined by this International Standard.**

This simple fix matches the programmer's expectation, and is entirely compatible with existing practice and the fundamentally simple memory model of the C++ since its inception.

Conclusion

Without giving up on the core *(object, offset)* address space model, the standard library provides function objects that are consistent with the built-in relational operators.

Acknowledgement

Thanks to Mike Miller for suggesting this fix, Herb Sutter for raising pointer comparison issues to the committee's attention, and to the countless contributors to the numerous pointer comparison debates.