Fixing CTAD for aggregates

Timur Doumler (papers@timur.audio)

Document #: P2082R0
Date: 2020-01-13
Project: Programming Language C++
Audience: Core Working Group

Abstract

This paper proposes to fix a remaining bug in the wording for CTAD for aggregates which unintentionally breaks existing C++17 code.

1 Motivation

C++20 introduces CTAD for aggregate types (motivation see [P1021R5]; wording see [P1816R0]). However, a problem still remains with the current wording, which should be fixed before C++20 is finalised.

As Jason Merrill pointed out, the current wording in the C++20 working paper breaks the existing deduction guide for `std::array`. In C++17, this works:

```
std:array a = {1, 2}; // deduces std::array<int, 2>
```

However, in C++20, an aggregate deduction candidate would be added and would fail, because `std::array` has only one aggregate element but the *braced-init-list* has two initialisers. With the existing wording, this would make the program ill-formed.

The fix, as suggested by Richard Smith, is to simply remove the aggregate deduction candidate from the overload set, such that existing code keeps working as before.

2 Proposed wording

The proposed changes are relative to the C++ working paper [N4842].

Modify [over.match.class.deduct] as follows:

> If there is no such element e_i, ~~the program is ill-formed~~the aggregate deduction candidate is not added to the set.

References

[N4842] Richard Smith. Working Draft, Standard for Programming Language C++. http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2019/n4842.pdf, 2019-11-27.

[P1021R5] Mike Spertus, Timur Doumler, and Richard Smith. Filling holes in Class Template Argument Deduction. http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2019/p1021r5.html, 2019-08-15.

[P1816R0] Timur Doumler. Wording for class template argument deduction for aggregates. http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2019/p1816r0.pdf, 2019-07-70.