Proposal for C2y

WG14 N3511

Title: Type qualifiers do not apply to type categories

Author, affiliation: Christopher Bazley, Arm. (WG14 member in individual capacity – GPU expert.)

Date: 2025-03-02

Proposal category: Editorial

Target audience: Committee

Abstract: A proposal to delete one word from a footnote.

Prior art: N3422.

Type qualifiers do not apply to type categories

Reply-to: Christopher Bazley (chris.bazley.wg14@gmail.com)
Document No: N3511
Date: 2025-03-02

Summary of Changes

N3511

- Initial proposal

Rationale

The wording proposed by N3422 '_Optional: a type qualifier to indicate pointer nullability (v2)' [1] included the following:

> *3 The unary `&` operator yields the address of its operand. If the operand has type "type", the result has type "pointer to type", preserving all qualifiers except any `_Optional` qualifier that previously applied to the type category of the operand.*

This wording was based on an existing footnote on 6.5.17.2p1 in the ISO C standard [2]:

> *The asymmetric appearance of these constraints with respect to type qualifiers is due to the conversion (specified in 6.3.3.1) that changes lvalues to "the value of the expression" and thus removes any type qualifiers that were applied to the type category of the expression (for example, it removes `const` but not `volatile` from the type `int volatile * const`).*

6.2.5p30 in the ISO C standard [2] defines type category as follows:

> *A type is characterized by its type category, which is either the outermost derivation of a derived type (as noted previously in this subclause in the construction of derived types), or the type itself if the type consists of no derived types.*

N3422 used 'type category' as a shorthand for the definition given above, with the goal of maintaining consistency in the standard's text.

However, in reflector message [SC22WG14.28618], Joseph Myers opined that:

> *Qualifiers apply to types, not to type categories; type categories are things such as "pointer" ot [sic] "function".*

And then in a following message that:

> *I think that existing wording is wrong.*

He is right, if one accepts that the term type category implies a higher level of abstraction. For example, the type categories ‘array’ and ‘pointer’ include many different types. Either way, the committee need to resolve this issue.

Proposed wording

The proposed wording is a diff from the N3435 working draft [2]. ~~Red~~ text is deleted text.

6.5.17.2 Simple assignment

Constraints

1 One of the following shall hold:[112)]

…

[112)] The asymmetric appearance of these constraints with respect to type qualifiers is due to the conversion (specified in 6.3.3.1) that changes lvalues to "the value of the expression" and thus removes any type qualifiers that were applied to the type ~~category~~ of the expression (for example, it removes `const` but not `volatile` from the type int `volatile * const`).

Acknowledgements

Joseph Myers for raising this issue.

References

[1] N3422 2024/12/17 Bazley, _Optional: a type qualifier to indicate pointer nullability (v2)
https://www.open-std.org/jtc1/sc22/wg14/www/docs/n3422.pdf

[2] N3435 2025/01/03 Meneide, C2y Working Draft
https://www.open-std.org/jtc1/sc22/wg14/www/docs/n3435.pdf