

# static operator[]

---

Document number: P2589R0

Date: 2022-05-12

Project: Programming Language C++, Evolution Working Group

Reply-to: Nevin “☺” Liber, [nliber@anl.gov](mailto:nliber@anl.gov)

## Table of Contents

<b>Introduction</b> .....	<b>1</b>
<b>Motivation and Scope</b> .....	<b>1</b>
<b>Impact On the Standard</b> .....	<b>2</b>
<b>Design Decisions</b> .....	<b>2</b>
<b>Technical Specifications</b> .....	<b>2</b>
<b>Acknowledgements</b> .....	<b>3</b>
<b>References</b> .....	<b>3</b>

## Introduction

Both the `static operator()` proposal and the multidimensional `operator[]` proposal mention that `operator[]` should also be `static`. This paper proposes to make `operator[]` consistent with `operator()` in this regard.

## Motivation and Scope

[P2128 Multidimensional subscript operator](#) had this to say on the subject of a `static operator[]`:

```
static operator[]
```

Our proposal does not support `static operator[]` declaration, but we would not oppose such a proposal

We do recommend it should be consistent with the call operator, as is explored in [P1169R0](#).

This author agrees these should be consistent and uniform, both from a useability and a teachability point of view.

Looking at the latest revision of [P1169 static operator\(\)](#), it mainly talks about [EWG88](#):

This idea was previously referenced in [\[EWG88\]](#), which reads:

In `c++std-core-14770`, `Dos Reis` suggests that `operator[]` and `operator()` should both be allowed to be static. In addition to that, he suggests that both should allow multiple parameters. It's well known that there's a possibility that this breaks existing code (`foo[1, 2]` is valid, the thing in brackets is a comma-expression) but there are possibilities to fix such cases (by requiring parens if a comma-expression is desired). EWG should discuss whether such unification is to be strived for.

Discussed in Rapperswil 2014. EWG points out that there are more issues to consider here, in terms of other operators, motivations, connections with captureless lambdas, who knows what else, so an analysis paper is requested.

There is a separate paper proposing multi-argument subscripting [\[P2128R3\]](#) already, with preexisting code such as `foo[1, 2]` already having been deprecated.

Again, this author agrees that `operator[]` and `operator()` should both be static.

## Impact On the Standard

The impact is minimal. Most of the changes added for `static operator()` involve lambdas and library wording, neither of which is applicable to `static operator[]`. Specifically, the only library feature at this time having multidimensional subscript operator support is [P0009 MDSPAN](#), and that library does not take advantage of a `static operator[]`.

## Design Decisions

Stay as close as possible to the behavior of member function `operator()`.

## Technical Specifications

Wording relative to [N4910 Working Draft, Standard for Programming Language C++](#), in [\[over.sub\]p1](#):

A *subscripting operator function* is a [member](#) function named `operator[]` ~~that is a non-static member function~~.

## Acknowledgements

Thanks to Mark Hoemmen, Christian Trott and Corentin Jabot for their comments.

## References

[P2128R6](#) Multidimensional subscript operator, Mark Hoemmen, Daisy Hollman, Corentin Jabot, Isabella Muerte, Christian Trott

[P1169R4](#) static operator(), Barry Revzin, Casey Carter

[EWG88](#) [tiny] Uniform handling of operator[] and operator()

[P0009R16](#) MDSPAN, Christian Trott, D.S. Hollman, Damien Lebrun-Grandie, Mark Hoemmen, Daniel Sunderland, H. Carter Edwards, Bryce Adelstein Lelbach, Mauro Bianco, Ben Sander, Athanasios Iliopoulos, John Michopoulos, Nevin Liber

[N4910](#) Working Draft, Standard for Programming Language C++, Thomas Köppe