# Require `span` & `basic_string_view` to be Trivially Copyable

Document number: P2251R1
Date: 2021-03-19
Project: Programming Language C++, Library Evolution Working Group
Reply-to: Nevin "☺" Liber, nliber@anl.gov

## Table of Contents

Introduction ........ 1
Revision History ........ 1
R1 ........ 1
Motivation and Scope ........ 2
Poll ........ 2
Impact on the Standard ........ 3
Technical Specifications ........ 3
Acknowledgements ........ 3
References ........ 3

## Introduction

Given its definition, it is strongly implied that `span` & `basic_string_view` are trivially copyable, but that is not yet a requirement.

## Revision History

### R1

- Expanded on the motivation.
- Added poll results from LEWG telecon discussion on 2021-Mar-16.
- Corrected description about Compiler Explorer link (it refers to three, not two, standard library implementations).
- Note: No change to the technical specification.

## Motivation and Scope

It is desirable that `span` and `basic_string_view` be trivially copyable for the following reasons:

- In heterogeneous computing, trivially copyable is heavily used as a proxy for types which can be copied between a host (such as a CPU) and a device (such as a GPU) or between two devices by just copying the bytes which make up the object representation of the type. If they are not trivially copyable, heterogeneous computing would not be able to use these types as vocabulary types.

- These types are already trivially copyable in all the major standard library implementations and developers may already be dependent upon it. That very question was asked about `string_view` on Stack Overflow over two years ago.

Both `span` and `basic_string_view` have:

- Defaulted copy constructors
- Defaulted copy assignment operators
- Defaulted destructors
- Exposition-only types consisting of a raw pointer and a `size_t`.
- Many member functions that are `constexpr`, and in C++17 would have required trivial destruction (for `basic_string_view` as `span` was added in C++20).

Because of the above, it is strongly implied that these are trivially copyable types. However, that is not a stated requirement.

Furthermore, libstdc++, libc++ and MSVC implement them as trivially copyable types: https://godbolt.org/z/nWY3dv.

I ran this by LWG and there was support for it with no objections.

## Poll

We took the following poll at the LEWG Telecon discussion on 2021-Mar-16:

***POLL***: Send P22510R0 (Require span & basic_string_view to be Trivially copyable) to LEWG Electronic Balloting for C++23, with priority B2: Improvement.

| SF | F | N | A | SA |
|---|---|---|---|---|
| 16 | 5 | 0 | 0 | 0 |

***# of Authors***: 1

***# of Participants***: 30

***Author Position:*** SF

***Outcome***: Unanimous consent

## Impact on the Standard

This is purely additive to the standard.

## Technical Specifications

These changes are relative to C++20:

In [span.overview], add:

**`span<ElementType, Extent>` is a trivially copyable type.**

`ElementType` is required to be a complete object type that is not an abstract class type.

In [string.view.template.general], add:

The complexity of `basic_string_view` member functions is O(1) unless otherwise specified.

**`basic_string_view<charT,traits>` is a trivially copyable type.**

## Acknowledgements

Thanks to Barry Revzin for pointing out that `basic_string_view` is strongly implied but not strictly required to be trivially copyable. Thanks to Jonathan Coe, Casey Carter for supporting this direction and encouraging me to write this paper, and an additional thanks to Jonathan Wakely for that as well as reviewing the wording.

This was supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of two U.S. Department of Energy organizations (Office of Science and the National Nuclear Security Administration) responsible for the planning and preparation of a capable exascale ecosystem, including software, applications, hardware, advanced system engineering, and early testbed platforms, in support of the nation's exascale computing imperative. Additionally, this research used resources of the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC02-06CH11357.

## References

C++20: Programming Languages – C++, International Standard ISO/IEC 14882, Sixth edition 2020-10

Stack Overflow: Is std::string_view trivially copyable?