Rename `affine_on`

Document Number: P4151R1
Date: 2026-03-27
Reply-to: Robert Leahy <rleahy@rleahy.ca>
Audience: LEWG

Abstract

`std::execution::affine_on` should be named `std::execution::affine`.

Background

When `std::execution::affine_on` was added [1] it was a binary-invocable object accepting two parameters, respectively:

- The sender to render affine, and
- The scheduler on which the aforementioned sender is to be affine

Later [2] it was redesigned to be unary accepting only a sender (the first bullet from above).

During LEWG discussion of the above (2026-03-24 in Croydon) it was mentioned (by the author of this paper) that changing the shape of the interface warranted changing the name of the algorithm, but this change was not polled nor was it executed in the paper under discussion.

Discussion

Before the addition of `std::execution::task` [1] the `std::execution` namespace contained three entities whose name was intended to contain the ordinary English word “on:”

- `std::execution::on`,
- `std::execution::continues_on`, and
- `std::execution::starts_on`

All three of these accept a scheduler as a parameter, the scheduler on which the operation starts, or on which the operation continues. This is expected if “on” is being used as a preposition: It expresses a relationship between two things, both of which must be stated.

Note also the use of “on” in the ordinary English description of binary `std::execution::affine_on` earlier in this paper:

“[A]ccepting two parameters [...] [t]he sender [...] and [t]he scheduler on which [...]”

Again consistent with use of “on” as a preposition.

Rendering `std::execution::affine_on` unary [2] destroys the rationale for including “on” in the name thereof. Affine on what, exactly?

Conclusion

Remove “on” from the name of `std::execution::affine_on` as it no longer makes sense.

Wording

Rename:

- `std::execution::affine_on` to `std::execution::affine`
- `std::execution::affine_on_t` to `std::execution::affine_t`
- The member function used to customize the `affine_on` algorithm from `affine_on` to `affine`

Apply the above after all papers moved at Croydon 2026 plenary with the exception of:

- P4159R0, and
- P4154R0

Review History

R0

Presented to LEWG in Croydon 2026-03-26. The following polls were taken:

POLL: Forward P4151R0 (rename affine_on to affine) to LWG for C++26.

SF	F	N	A	SA
7	7	0	0	0

Attendance: 22 IP, 4 online
# of Authors: 1
Author’s Position: SF
Outcome: Consensus in favor

Revision History

R1

- Adopted LWG feedback
- Specified application order

References

[1] D. Kühl et al. Add a Coroutine Task Type P3552R3
[2] D. Kühl. Scheduler Affinity P3941R2