Project: ISO JTC1/SC22/WG21: Programming Language C++
Doc No: WG21 **P3828R0**
Date: 2025-10-06
Reply to: Nicolai Josuttis (nico@josuttis.de)
Co-authors:
Audience: LEWG, LWG
Issues:
Previous:

Rename the to_input view to as_isput

The current name of the "to_input" view is misleading, because this sounds like the elements of the underlying range are actively processed (like with std::ranges::to). Instead it only ensures that the underlying range is seen ***as*** something different for those that later process the elements using this view. For views like this we usually use the name “as_...” (e.g. as_const or as_rvalue).

This paper proposes to rename the to_input view to as_input. This is both more intuitive and consistent.

Note that in this case this might include to also change the name of the section in the C++ standard and the name of the feature test macro.

Proposed Wording

(All against N4944)

In principle simply rename all occurences of “to_input” to “as_input” and “to input view” to “as input view.”

In **17.3.2 Header <version> synopsis [version.syn]**

Rename as follows:

```
#define __cpp_lib_ranges_toas_input    202502L    // freestanding, also in <ranges>
```

In 25.2 Header <ranges> synopsis [ranges.syn]

Rename as follows:

```
// 25.7.35, to as input view
template<input_range V>
    requires view<V>
class toas_input_view;
template<class V>
  constexpr bool enable_borrowed_range<toas_input_view<V>> =
    enable_borrowed_range<V>;
namespace views { inline constexpr unspecified toas_input = unspecified ; }
```

Rename

25.7.35 To input view [range.to.input]

to

25.7.35 As input view [range.as.input]

Rename

25.7.35.1 Overview [range.to.input.overview]

to

25.7.35.1 Overview [range.to.input.overview]

and rename inside:

1 ~~to~~as_input_view presents a view of an underlying sequence as an input-only non-common range. [*Note 1* : This is useful to avoid overhead that can be necessary to provide support for the operations needed for greater iterator strength. —*end note*]

2 The name views::~~to~~as_input denotes a range adaptor object (25.7.2). Let E be an expression and let T be decltype((E)). The expression views::~~to~~as_input(E) is expression-equivalent to:

(2.1) — views::all(E) if T models input_range, does not satisfy common_range, and does not satisfy forward_range.

(2.2) — Otherwise, ~~to~~as_input_view(E).

Rename

25.7.35.2 Class template ~~to~~_input_view [range.~~to~~.input.view]

to

25.7.35.2 Class template as_input_view [range.as.input.view]

and rename inside:

```
namespace std::ranges {
template<input_range V>
 requires view<V>
class ~~to~~as_input_view : public view_interface<~~to~~as_input_view<V>> {
   V base_ = V(); // exposition only
   // 25.7.35.3, class template ~~to~~as_input_view::iterator
   template<bool Const> class iterator ; // exposition only
public:
   ~~to~~as_input_view() requires default_initializable<V> = default;
   constexpr explicit ~~to~~as_input_view(V base);
   …
 template<class R>
   ~~to~~as_input_view(R&&) -> ~~to~~as_input_view<views::all_t<R>>;
}
constexpr explicit ~~to~~as_input_view(V base);
```

1 *Effects*: Initializes *base_* with std::move(base).

Rename

25.7.35.3 Class template ~~to~~_input_view::*iterator* [range.~~to~~.input.iterator]

to

25.7.35.3 Class template as_input_view::*iterator* [range.as.input.iterator]

and rename inside:

```
namespace std::ranges {
 template<input_range V>
   requires view<V>
 template<bool Const>
 class ~~to~~as_input_view<V>::iterator {
   using Base = maybe-const <Const, V>; // exposition only
…
```

Acknowledgements

Thanks to a lot to everybody who helped and gave support to come to finally get this proposal done.

Rev0:

First initial version.