A proposal for a type trait to detect scoped enumerations

Document Number: P1048R1
Date: 2020-10-12
Reply-to: Juan Alday (alday@ieee.org)
Audience: Library Working Group

Introduction
This paper proposes is_scoped_enum, a new trait for the C++ Standard Library, to detect whether a type is a scoped enumeration.

Motivation and Scope
It is useful in certain contexts to know whether an enumeration is scoped or unscoped and apply (via SFINAE) different behavior depending on the type of such enumeration.

One use the author has recently worked on involves creating a set of unit tests to track the progress of a legacy library migration to modern C++. By using this trait, it is possible to define a unit test to track the progress of migration of unscoped to scoped enumerations.

Impact On The Standard
It proposes changes to an existing header, <type_traits>, but it does not require changes to any standard classes or functions and it does not require changes to any of the standard requirement tables.

This proposal does not depend on any other library extensions.

Naming
The existing trait to detect an enum type is is_enum, so variations containing
‘enumeration’ are not considered by the author.
is_scoped_enum as the suggested name was approved by LEWG in Cologne.

**Wording**

All proposed additions (there are no deletions) are relative to the 2020-04 working draft N4861. Editorial notes are displayed against a gray background.

Insert into [meta.type.synop] as shown:

```cpp
template<class T> struct is_member_pointer;
template<class T> struct is_scoped_enum;
```

```cpp
template<class T>
inline constexpr bool is_member_pointer_v = is_member_pointer<T>::value;
```

```cpp
template<class T>
inline constexpr bool is_scoped_enum_v = is_scoped_enum<T>::value;
```

Insert into [tab:meta.unary.prop]

```
<table>
<thead>
<tr>
<th>Template Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>T is a scoped enumeration [dcl.enum]</td>
<td></td>
</tr>
</tbody>
</table>
```

**Feature test macro**

Insert into [version.syn]

```cpp
#define __cpp_lib_is_scoped_enum <DATE OF ADOPTION> // also in <type_traits>
```

**Approvals**

<table>
<thead>
<tr>
<th>Organization</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEWG</td>
<td>2019-07-18</td>
</tr>
<tr>
<td>EWG</td>
<td>2019-07-23</td>
</tr>
<tr>
<td>CWG</td>
<td>2020-08-17</td>
</tr>
</tbody>
</table>
Example implementation

```cpp
template<class _T, bool = is_enum_v<_T>> struct __is_scoped_enum_helper : false_type {};
template<class _T>struct __is_scoped_enum_helper<_T, true>
  : public bool_constant<!is_convertible_v<_T, underlying_type_t<_T>>> {};
template<class _T>struct is_scoped_enum : public __is_scoped_enum_helper<_T> {};
```

Bibliography


Document history

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2018-05-01</td>
<td>Initial draft</td>
</tr>
<tr>
<td>1</td>
<td>2020-10-12</td>
<td>Removed numeric sections and moved to meta.unary.prop Added feature test macro Added dates for the evolution of the paper through the working groups.</td>
</tr>
</tbody>
</table>