Other Smaller Removed or Deprecated Items

This lesson will describe the deprecated features.

WE'LL COVER THE FOLLOWING

- Deprecated Features
  - Deprecating `shared_ptr::unique()`
  - Deprecating `<codecvt>`
  - Removing Deprecated Iostreams Aliases
  - Deprecate C library headers
  - Deprecate `std::result_of`
  - Deprecate `std::memory_order_consume` Temporarily
  - Remove allocator support from `std::function`

Deprecated Features

Deprecating `shared_ptr::unique()`

In C++14 `shared_ptr::unique()` was defined as `use_count() == 1`. But since `use_count()` is only approximation in multithreaded environments (as it doesn't imply any synchronisation access) then `unique()` it not reliable.

See more information in P0521.

Deprecating `<codecvt>`

The `<codecvt>` header declares several conversion utilities: `codecvt_utf8`, `codecvt_utf16` and `codecvt_utf8_utf16`. But those classes are hard to use, unsafe and not well specified.

Note: the class `std::codesvt` is not deprecated, as it's located in another header `<locale>`. So you can still use that.

See more information in P0618R0.

See more information in P0618R0

Removing Deprecated Iostreams Aliases

Since C++11 the following iostream types and methods were deprecated, and now they are removed from the Library.

```
typedef T1 io_state;  // T1 is integer
typedef T2 open_mode; // T2 is integer
typedef T3 seek_dir;  // T3 is integer
typedef implementation-defined streamoff;
typedef implementation-defined streampos;
basic_streambuf::stossc()
```

Also, the methods and overrides that depend on the above types were removed.

Previously they were all declared in Annex D of the Standard: [depr.ios.members].

See more information in P0004R1.

Deprecate C library headers

The following headers are now deprecated:

- `<ccomplex>`
- `<cstdalign>`
- `<cstdbool>`
- `<ctgmath>`

This is a result of cleaning up places that depend on the C99 specification. In C++17 the Standard relates to C11 rather than C99.

See more information in P0063R3.

Deprecate `std::result_of`

The type trait `std::result_of` used a non-variadic template declaration which limited its uses. It's advised to use enhanced traits, for example `std::invoke_result`.

See more information in P0604R0.

Deprecate `std::memory_order_consume` Temporarily

Deprecate `std::memory_order_consume` Temporarily

The memory model of `memory_order_consume` is hard to implement and not well specified in the Standard. The model is now temporarily deprecated and may reappear in the future.

See more information in P0371R1.

Remove allocator support from `std::function`

`std::function` uses type-erasure to handle callable objects. It’s very complicated (or not implementable efficiently) to have allocator support for this type, so it was decided to remove this from the Standard.

See more information in P0302R1.

We will conclude with compiler support in the next lesson.