Further refinements to the attempt to create a type-dependent expression that is always false

devblogs.microsoft.com/oldnewthing/20200319-00

March 19, 2020

Raymond Chen

A little while ago, I discussed creating a type-dependent expression that is always false, and I settled upon this:

```
    static_assert(!sizeof(Op*), "message");
```

This covers the cases where `Op` is an incomplete type or `void`. Billy O'Neal pointed out to me that there are a few cases where this doesn't work.

One case is if `Op` is a reference type. You are not allowed to create pointers to reference types, so the attempt to generate a `false` expression will fail with

```
// MSVC
error C2528: 'abstract declarator': pointer to reference is illegal

// gcc
error: forming pointer to reference type

// clang
error: 'type name' declared as pointer to a reference
```

This can be repaired by wrapping `Op` in a `std::` `decay_t`:

```
 static_assert(!sizeof(std::decay_t<Op>*),
               "Don't know what you are asking me to do.");
```

But the next part is worse: The `Op` could be an *abominable function*.

I was previously not aware of abominable functions, but upon reading up on them, I've concluded that they are fully deserving of their name. It's like hot lava, fatal poison, and supernatural malfeasance all rolled up into one.

Read up on abominable function types and see if you agree.

The only way to win the game with abominable functions is not to play, so let’s just hand it off to `std::void_t` to be neutralized into a `void`. This also solves the problem with references, since `std::void_t` simply sucks up everything it is given and spits out a `void`.

That leaves us with this:

```
static_assert(!sizeof(std::void_t<Op>*),
              "Don't know what you are asking me to do.");
```

At this point, since we know that the only thing that can come out of `std::void_t` is `void` itself, we can tweak the expression to make a false statement a bit more directly:

```
static_assert(std::is_same_v<std::void_t<Op>, int>,
              "Don't know what you are asking me to do.");
```

Raymond Chen

Follow