1 Introduction

The paper p0189 that introduced the [[nodiscard]] attribute did not consider constructors. However, gcc for example implements the checking for constructors, even so it warns about putting [[nodiscard]] on a constructor definition. Here I propose to allow [[nodiscard]] also on constructors (which it implicitly is allowed by the current wording) and suggest checking it for cast expressions so that we can put it on things like scoped_lock etc.

The need is more obvious in C++ 17 and later, where CTAD allows for fewer factory functions and thus the easy to make mistake by just typing the type and constructor arguments instead of defining a local variable.

Since this change is editorial only, it might be considered to be applied for the current working paper.

2 Impact on the standard

The change is IMHO editorial only, since the semantics of warnings is only in a note. Change section [dcl.attr.nodiscard] as follows. Note that a constructor declaration is a function declaration.

2.0.1 Nodiscard attribute [dcl.attr.nodiscard]

The attribute-token nodiscard may be applied to the declarator-id in a function declaration or to the declaration of a class or enumeration. It shall appear at most once in each attribute-list and no attribute-argument-clause shall be present.

[Note: A nodiscard call is a function call expression or an explicit type conversion that calls a function or constructs an object through a constructor previously declared nodiscard, or whose return type

---

1

---

2
or type is a possibly cv-qualified class or enumeration type marked `nodiscard`. Appearance of a `nodiscard` call as a potentially-evaluated discarded-value expression (7.2) is discouraged unless explicitly cast to `void`. Implementations should issue a warning in such cases. This is typically because discarding the return value of a `nodiscard` call has surprising consequences. — end note}

3  

```cpp
struct [[nodiscard]] error_info { /* ... */ };  
error_info enable_missile_safety_mode();  
void launch_missiles();  
void test_missiles() {  
  enable_missile_safety_mode(); // warning encouraged  
  launch_missiles();  
}  
error_info &foo();  
void f() { foo(); } // warning not encouraged: not a nodiscard call, because neither  
// the (reference) return type nor the function is declared nodiscard  

— end example
```