03/03/15

Bugsquashing Seminar
C++11
nullptr pointer literal

Eric Heim

Bugsquashing – nullptr pointer literal

- Before C++11:
  - 0 or the preprocessor macro NULL is used for pointers
  - 0 can be interpreted as a pointer value or an integer
  - Can cause problems, e.g. function overloading:

```
1 void foo(char*);
2 void foo(int);
```

Which one should be used?

- C++11 **nullptr**:
  - Convertible / comparable to every pointer type
  - **Not** Convertible to integral types, except bool
  - Allows forwarding via template functions

Bugsquashing – nullptr pointer literal

- Example nullptr:

```
 5 MyObject *o = nullptr; // OK
 6 int   *pi = nullptr;     // OK
 7 bool    b = nullptr;     // OK. b is false.
 8 int     i = nullptr;     // error
 9
10 foo(nullptr);// calls foo(nullptr_t),
11                // not foo(int);
```

Bugsquashing – nullptr pointer literal

- Forwarding in template function:

```
17 template<class F, class A>
18 void Fwd(F f, A a) {
19   f(a);
20 }
21
22 void g(int* i) {
23   std::cout << "Function g called\n";
24 }
25
26 int main() {
27   g(NULL);            // Fine
28   g(0);               // Fine
29
30   Fwd(g, nullptr);    // Fine
31   Fwd(g, NULL);  // ERROR: No function g(int)
32 }
```

http://en.cppreference.com/w/cpp/language/nullptr

Summary

Simply use nullptr whenever you would have used NULL in the past.

Questions?