سلسلة تعلم البرمجة بلغة C++ الحديثة

Learn Modern C++ Programming Course

إعداد المهندس أحمد الديب

#14: Enumerations

Enumerations

- A type that can hold a set of integer values specified by the user. Some of an enumeration’s possible values are named and called enumerators.
- enum classes, for which the enumerator names are local to the enum and their values do not implicitly convert to other types.
- Plain enums, for which the enumerator names are in the same scope as the enum and their values implicitly convert to integers.

Source: The C++ Programming Language (4th Edition), Bjarne Stroustrup

enum classes

```
enum class Traffic_light { red, yellow, green };
enum class Warning { green, yellow, orange, red };

Warning a1 = 7;                     // error : no int->Warning conversion
int a2 = green;                     // error: green not in scope
int a3 = Warning::green;            // error : no Warning->int conversion
Warning a4 = Traffic_light::green;  // error : different types
Warning a5 = Warning::green;        // OK
```

Source: The C++ Programming Language (4th Edition), Bjarne Stroustrup

enum underlying type

- Must be one of the signed or unsigned integer types, the default is int. We could be explicit about that.

```
enum class Warning : int { green, yellow, orange, red };
enum class Warning : char { green, yellow, orange, red };

void print_warning_enum() {
  std::cout << static_cast<int>(Warning::green) << std::endl;
  std::cout << static_cast<int>(Warning::yellow) << std::endl;
  std::cout << static_cast<int>(Warning::orange) << std::endl;
  std::cout << static_cast<int>(Warning::red) << std::endl;
}
```

Source: The C++ Programming Language (4th Edition), Bjarne Stroustrup

enum Operators

```
enum class StatusFlags {
  data_received = 0x1,
  rx_buffer_full = 0x2,
  rx_error = 0x4,
  data_transmitted = 0x8,
  tx_buffer_empty = 0x10,
  tx_error = 0x20,
};

constexpr StatusFlags operator|(StatusFlags a, StatusFlags b) {
  return static_cast<StatusFlags>(static_cast<int>(a) | static_cast<int>(b));
}
constexpr StatusFlags operator&(StatusFlags a, StatusFlags b) {
  return static_cast<StatusFlags>(static_cast<int>(a) & static_cast<int>(b));
}
```

By default, an enum class has only assignment, initialization, and comparisons (e.g., == and <;) defined.

Source: The C++ Programming Language (4th Edition), Bjarne Stroustrup

enum Operators

```
void check_status_flags(StatusFlags status) {
  StatusFlags ready_to_read{StatusFlags::data_received |
                            StatusFlags::rx_buffer_full};
  if ((status & ready_to_read) == ready_to_read) {
    std::cout << "Read Data" << std::endl;
  }

  StatusFlags ready_to_write{StatusFlags::data_transmitted |
                             StatusFlags::tx_buffer_empty};
  if ((status & ready_to_write) == ready_to_write) {
    std::cout << "Write Data" << std::endl;
  }

  StatusFlags any_error{StatusFlags::rx_error | StatusFlags::tx_error};
  StatusFlags no_error{static_cast<StatusFlags>(0)};
  if ((status & any_error) != no_error) {
    std::cout << "RX and/or TX Error" << std::endl;
  }
}

int main() {
  //
  StatusFlags status{static_cast<StatusFlags>(0x3F)};
  check_status_flags(status);
}
```

Source: The C++ Programming Language (4th Edition), Bjarne Stroustrup

Thank you