

## Section 2.1 C++11

## enum class

```
enum class SysPort { e_INPUT = 27, e_OUTPUT = 29, e_ERROR = 32, e_CTRL = 6 };
// enumerated port values used to configure our systems
```

Now suppose we want to call the function `setPort` using one of these enumerated values:

```
void setCurrentPortToCtrl()
{
    setPort(SysPort::e_CTRL); // Error, cannot convert SysPort to int
}
```

Unlike the situation for a *classic* `enum`, no implicit conversion occurs from an `enum class` to its underlying integral type, so anyone using this enumeration will be forced to somehow explicitly `cast` the enumerator to some arithmetic type. There are, however, multiple choices for performing this cast:

```
#include <type_traits> // std::underlying_type

void test()
{
    setPort(int(SysPort::e_CTRL)); // (1)
    setPort((int)SysPort::e_CTRL); // (2)
    setPort(static_cast<int>(SysPort::e_CTRL)); // (3)
    setPort(static_cast<std::underlying_type<SysPort>::type>(
        SysPort::e_CTRL)); // (4)
    setPort(static_cast<int>(
        static_cast<std::underlying_type<SysPort>::type>(SysPort::e_CTRL))); // (5)
}
```

Any of the above casts would work in this case, but consider a future where a platform changed `setPort` to take a `long` and the control port was changed to a value that cannot be represented as an `int`:

```
int setPort(long portNumber);
enum class SysPort : unsigned { e_INPUT = 27, e_OUTPUT = 29, e_ERROR = 32,
    e_CTRL = 0x80000000 };
// enumerated port values used to configure our systems
```

Only the casting method, line (4) in the example on this page, will pass the correct value for `e_CTRL` to this new `setPort` implementation. The other variations will all pass a negative number for the port, which would certainly not be the intention of the user writing this code. A classic, C-style `enum` would have avoided any manually written cast entirely, and the proper value would propagate into `setPort` even as the range of values used for ports changes:

```
struct SysPort // explicit scoping for a classic, C-style enum
{
    enum Enum { e_INPUT = 27, e_OUTPUT = 29, e_ERROR = 32,
        e_CTRL = 0x80000000 };
};
```