# - Example

Let's look at an example of how scoped enumerations can be used.

WE'LL COVER THE FOLLOWING

- Example

## Example #

We can explicitly specify the type of enumerators. By default, it's `int`.

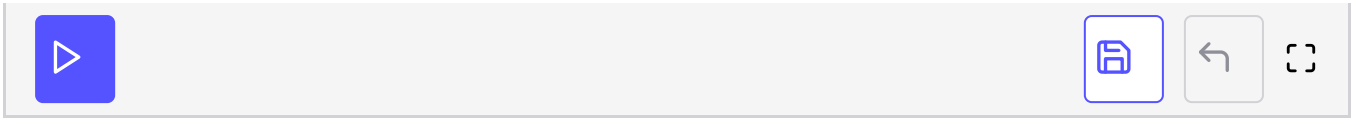
But that does not have to be. We can use integral types like `bool`, `char`, `short int`, `long int`, or, `long long int`. Read msdn.microsoft.com for the details. See the post Check types for information on how to check if a type is integral at compile time.

We can use the scoped property and the explicit type specification of an enumeration independently. Depending on the base types, the enumerations can have different sizes.

Below is an example of the implementations of scoped enumerations using `struct` and `class`.

```
#include <iostream>

enum class Color1{
  red,
  blue,
  green
};

enum struct Color2: char{
  red= 100,
  blue, // 101
  green // 102
};

void useMe(Color2 color2){
```

```
  switch(color2){
  case Color2::red:
    std::cout << "Color2::red" << std::endl;

    break;
  case Color2::blue:
    std::cout << "Color2::blue" << std::endl;
    break;
  case Color2::green:
    std::cout << "Color2::green" << std::endl;
    break;
  }

}

int main(){

  std::cout << std::endl;

  std::cout <<  "static_cast<int>(Color1::red): " <<  static_cast<int>(Color1::red) << std::e
  std::cout <<  "static_cast<int>(Color2::red): " <<  static_cast<int>(Color2::red) << std::e

  std::cout << std::endl;

  std::cout << "sizeof(Color1) = "  << sizeof(Color1) << std::endl;
  std::cout << "sizeof(Color2) = "  << sizeof(Color2) << std::endl;

  std::cout << std::endl;

  Color2 color2Red{Color2::red};
  useMe(color2Red);

  std::cout << std::endl;

}
```



For further information, see enum.

---

In the next lesson, we'll move on to **pointers**.