Namespaces/enumerators attributes

This lessons discusses attributes when they were applied on namespaces/enumerators, the problems with it in C++ 11 and the fix C++ 17 offers.

WE'LL COVER THE FOLLOWING

A look at attributes

The idea for attributes in C++11 was to be able to apply them to all sensible places like classes, functions, variables, typedefs, templates, enumerations... But there was an issue in the specification that blocked attributes when they were applied on namespaces or enumerators.

This is now fixed in C++17. We can now write:

```
#include <iostream>

namespace [[deprecated("use BetterUtils")]] GoodUtils
{
  void DoStuff() { }
}

namespace BetterUtils
{
  void DoStuff() { }
}

int main()
{
  GoodUtils::DoStuff();
}
```

Clang reports:

warning: 'GoodUtils' is deprecated: use BetterUtils [-Wdeprecated-

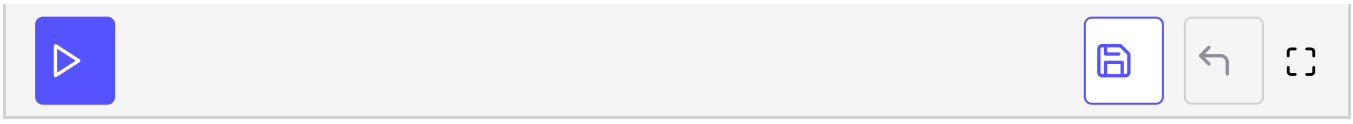
declarations]

Another example is the use of deprecated attribute on enumerators:

```
#include <iostream>
using namespace std;

enum class ColorModes
{
  RGB [[deprecated("use RGB8")]],
  RGBA [[deprecated("use RGBA8")]],
  RGBA16F,
  RGB8,
  RGBA8
};

// use:
auto colMode = ColorModes::RGBA;
```



Under GCC we'll get:

warning: 'RGBA' is deprecated: use RGBA8 [-Wdeprecated-declarations]

> *Extra Info:* The change was described in N4266(wording) and N4196(reasoning).

The next section will now discuss a few more features of attributes in C++ 17.