Enumerated Types

Single value types are called scalar types. All of the built-in, primitive data types—`int`, `char`, `bool` and `double`—are **scalar** data types. The `Date` type, which you met in the last lesson, was a user-defined **structured** type.

The `Weekday` type shown here above the `Date` is a **user-defined scalar** type which contains only a single simple value.

You may define your own **new scalar types** by listing the **elements in their domain**. Such types are called **enumerated types**. Enumerated types are based upon another type, in this case the integers. Such types are called **derived** or **compound types**.

The syntax for defining an enumerated type is:

```
enum class type-name { name-list };   // C++ 11 scoped enums
enum type-name { name-list };         // traditional "plain" enums
```

Here ***type-name*** is the **name** of the type and ***name-list*** is a list of literals representing the values in the domain, separated by commas. The ***name-list*** does not need any semicolons, unlike regular variable definitions. However, you **must end the definition with a semicolon**.

The **scoped enumeration** was added in C++11, while the older type is called an **un-scoped or plain enumeration**. In this class we'll use the newer, scoped enumerations, since they will help you avoid all kinds of scope bugs.