Operator Overloading

Dustin Chase

C++ Lightning Talks

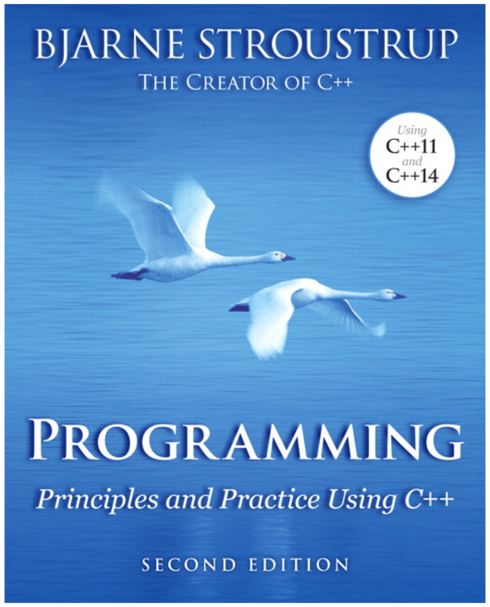
December 14, 2015

Outline of Presentation

- Introduction
- Enumeration
- Operator Overloading
- Operator Overloading In Action
- Closing Thoughts

Introduction

- Who am I?
- Expected Background/Audience
- Source Material



Enumeration

- Enumeration (enum): A very simple user-defined type, specifying its set of values (its enumerators) as symbolic constants.

```
enum class Card {copper, silver, gold, council_room, …};
   if (card_played == Card::copper) {
          copperAction();
     }
```

Operator Overloading

- Operator Overloading: Define almost any C++ operator for class or enumeration operands.
- Provide conventional notation for a type we design
    - “+” means add, “=“ means assignment
    - Convenience, Efficiency, Clarity

Enumeration and Overloading Together

- Example enumeration:

```
enum class Month {
        Jan=1, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec
};
```

- Next month?

A Function!

```
Month nextMonth(const Month
&m) {
        switch (m)
  {
     case(Month::Jan):
        return Month::Feb;
        break;
     case(Month::Feb):
        return Month::Mar;
        break;
```

- Lots of code to write – error prone
- Not as convenient for users of your code
- What does nextMonth(const Month &m) do?

Operator Overload!

```
 Month& operator++(Month &m) {
   m = (m==Month::Dec) ? Month::Jan : Month(int(m) + 1);
   return m;
}

Month end = Month::Jan;
++end;
//vs. end = nextMonth(end);
//More natural to read than a series of function calls
```

++ Operator Usage

```
for (Month n = Month::Jan; n < Month::Dec; ++n) {
    cout << int(n) << endl;
}
```

Another example: ‘[]’ operator

- C++ strings

```
string name = “William”;
name[3] = ‘x’;
//name is now “Wilxiam”;
```

- Saves us from iterating through the string object, or using some combination of other functions

Closing

- Enum helps make your code readable
  - Card::copper vs. 7 (magic number)
- Operator overloading as well
- Define operators only with conventional meaning
  - "+" should mean addition, not "tooblekane"
- Don't define operators for a type unless you're sure it will make positive changes to your code
- The most interesting operators aren't +, -, * and /, but =, ==, !=, [] and ().