

## Odds and Ends

---

# *Odds and Ends*

CS193D, 3/15/06

## Where We've Been

- Covered more of the C++ language than many professional C++ programmers know.
- Discussed/Reviewed the principles and theories of Object-Oriented Design.
- Learned reusable design and language patterns.
- Built non-trivial, real-world C++ applications with unit tests and documentation.
- Added several buzzwords to our resumes: XML, SOAP, Extreme Programming, Unit Testing, Test-Driven Development.

## What We Haven't Covered

- Overloading new and delete – Chapter 16
  - Efficiency – Chapter 17
  - Cross-Platform and Cross-Language Development – Chapter 18
  - Debugging Techniques – Chapter 20
  - Advanced STL – Chapter 23
  - Distributed Objects (other than SOAP) – Chapter 24
  - Use of cppunit and xerces libraries
- 
- OS-specific features (threads, GUIs, etc.)
  - GUI Frameworks (MFC, etc.)
  - Networking (sockets)

## Evaluating C++ (the good)

- C++ is familiar (it's C-based)
- C++ is general purpose (like C and Java) and supports generics
- C++ is fast and compile-time oriented
- C++ is extremely popular, and a large amount of code exists
- C++ is flexible (operator overloading, type casting, etc.)
- C++ is standardized, nobody owns it
- C++ permits low-level operations (new/delete) and is compiled
- C++ has built-in functionality, but allows you to use libraries

## Evaluating C++ (the bad)

- C++ is dangerous (buffer overflows, bus errors)
- C++ lacks memory management
- C++ isn't as hip as it once was (no regex???)
- C++ is more rigid than Python, Perl, C#, etc.
- C++ is less runtime-oriented than Java, Smalltalk, Objective-C
- C++ has a steep learning curve (but we're so over that)
- C++ is huge!
- C++ has backwards compatibility baggage

## The Future of C++

The working name for the next major version of the C++ language is C++0x. Here are some *likely* additions:

- *Concepts*, a "type of a type". A way to specify the requirements of a class for use in a template. Results in better error messages.
- Less obnoxious iterator syntax:  
`for (auto p = v.begin(); p != v.end(); p++) ...`
- New built-in functionality, including hash tables, regex, better smart pointers, new math-related functions. See also: Boost
- Optional garbage collection
- A way of wrapping references so they can be used in the STL?
- Sockets and threads?
- A standard GUI library? Swing for C++? Not very likely...
- New features that make the language easier to learn (huh?)

## Other C++ Resources

*The C++ Programming Language* (Stroustrup, 2000)

*Thinking in C++, Volume 2* (Eckel, 2003)

*Effective C++* (Meyers, 2005)

*More Effective C++* (Meyers, 1995)

*Effective STL* (Meyers, 2001)

*Design Patterns* (Gamma, 1995)

*Extreme Programming Explained* (Beck, 2004)

## Some Highlights From Effective C++

- 1) Use const instead of #define
- 11) Declare a copy ctor and op= if you have dynamic memory
- 20) Avoid data members in the public interface
- 24) Choose carefully between function overloading and default args
- 33) How to use inlining the right way
- 45) Know what functions C++ silently writes and calls
- 48) Pay attention to compiler warnings
- 50) Improve your understanding of C++

## Where to Go From Here

I want to write Windows apps:

- Download the free Microsoft tools
- Give C# a try
- Take CS193W (CS193E?)

I want to be more hard-core:

- Take CS244A (networking)
- Take CS93SI (Modern C++ Techniques)

I want to learn more about OOP design:

- Take CS108 (Object-Oriented Systems Design)
- Take CS249 (OOP: A Modeling and Simulation Perspective)

I want to learn about other stuff:

- Keep taking CS193's

## The Final

The final is on Wednesday, March 22<sup>nd</sup> from 8:30am – 11:30am in Gates B01 (the regular class room, for those who regularly come to class)

An alternate final will be offered one day earlier, Tuesday, March 21<sup>st</sup> from 3:30pm – 6:30pm, room TBA.

Since the alternate is at an otherwise unscheduled time, nobody should have any conflicts. You may attend either final and you don't need to tell me ahead of time which one you're coming to.

Grade breakdown (from Handout 1):

Assignments 40%

Midterm 25%

Final 35%