C++ Programming

~ Introduction ~

Prepared for Ingenix, Inc.

C++ in Three Days!!!

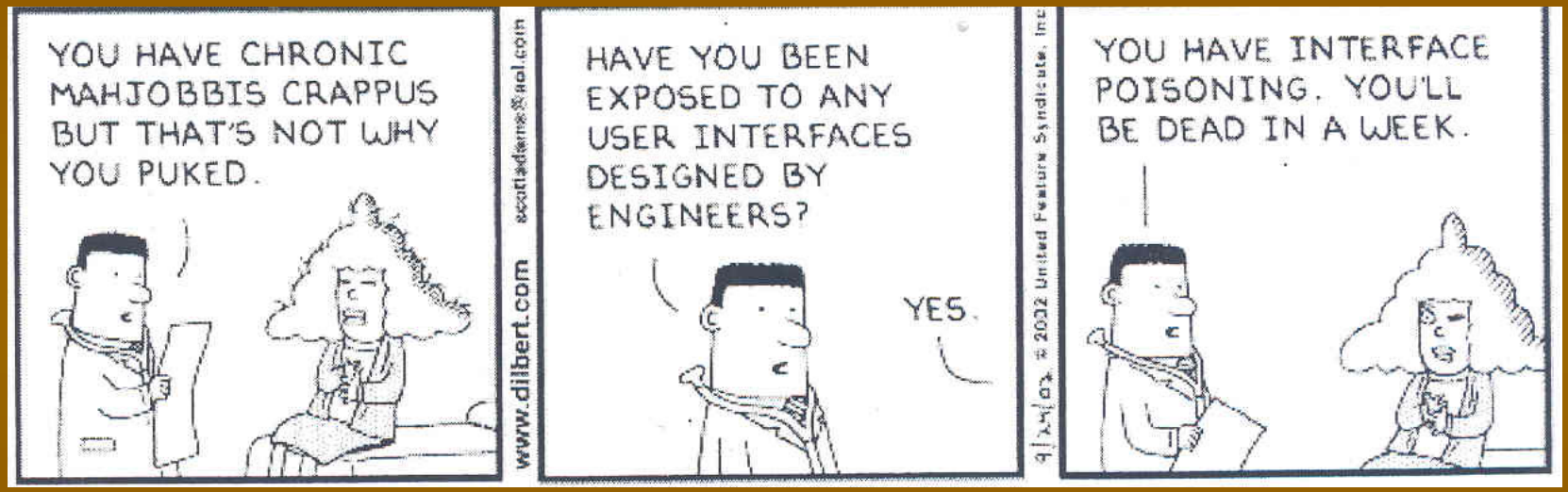
- Three *intense* days
- Ask questions
  - We'll customize as we go
- Do the exercises
  - We'll set time aside for them
- We'll use but not emphasize the Microsoft Visual Studio .NET 2003 IDE
  - You are learning *Standard* C++

About Me

- CS Professor at UVSC since 2001
- Used to work at Ingenix! (Nov 1998 - Mar 2000)
- 3 degrees in math
- 20+ years software engineering experience
  - Defense contractors, LDS Church, Oracle, Novell
- Contributing member of the C++ Standards Committee (designed the *bitset* container)
- Past Editor of the *C/C++ Users Journal* (1992-2003)
- Founding editor of *The C++ Source* (2004 -)
- Author of 2 C++ books published by Prentice-Hall
- "Teaching" since 1977 (7 colleges, corporate)

About You

- Experienced Programmers
- Very smart
  - (otherwise you wouldn't feel comfortable around David North)
- Able to stay interested and alert while sitting in a dark room for extended periods
- Actually, believe it or not, want to learn this Cool Stuff


YOU HAVE CHRONIC MAHJOBBIS CRAPPUS BUT THAT'S NOT WHY YOU PUKED.
www.dilbert.com scottadams@aol.com
HAVE YOU BEEN EXPOSED TO ANY USER INTERFACES DESIGNED BY ENGINEERS?
YES.
YOU HAVE INTERFACE POISONING. YOU'LL BE DEAD IN A WEEK.

The Deception of Visual, Event-driven Programming

- Only deals with UI
- Seems simple, but encourages poor practice
  - Prototypes become products before their time
  - UI should be *totally separate* from object model
  - Programming is not Mouse Engineering
- <soapbox>
  - Programmers shouldn't design GUIs
  - Let the users do it!
- </soapbox>

Programming is more than Visual

- “The effective exploitation of his powers of abstraction must be regarded as one of the most vital activities of a competent programmer.” -- Dijkstra

About C++

- The most powerful programming language in the history of computing
- It is *low-level*
  - Evolved from C
  - Only assembly language goes lower
- It is *high-level*
  - No language has higher-level constructs
  - (Exception: no Lisp-like **eval** function)
- It is everything else in between

More About C++

- It is not easy to master
- You can hurt yourself
  - Especially if you stay low-level
  - Should use the library components available
- Is being revised
  - C++0x is under construction
  - Will have even more powerful components
  - Will (hopefully) be easier to use
  - Is not owned by a single company!
    - But is the best choice for .NET development!

A Multi-paradigm Language

- Procedural programming
  - if-else, loops, functions, arrays, records (struct)
  - Supports large, multi-file projects
  - Compiles to native code (no virtual machine)
- Object-based programming
  - Classes (data members + member functions)
  - Automatic initialization and clean-up
  - Operator overloading

A Multi-paradigm Language (continued)

- Object-oriented Programming
  - Inheritance
    - Single and multiple
    - Interface inheritance (public), implementation inheritance (private), and in between (protected)
  - Polymorphism
    - OO's great contribution to software engineering
    - The fastest there is
- Generic Programming
  - The next revolution in computing
  - Key to "higher-level" programming
  - C++'s advantage over competitors: type safety

The C++ Type System

- C++ offers many built-in types
  - char, int, float, etc.
  - Library types: string, vector, list, map, etc.
- Statically typed
  - Proper use of types enforced at compile time
  - Very difficult to violate type safety
- User-defined classes are types
  - But you must define them well!

Teasers

- A look at typical Modern C++ usage
  - Should look a little foreign to most of you
  - Our goal for you after these 3 days!
  - Don’t worry: we’ll step back and work our way there
- Hello (6 examples)
  - #include, namespaces, stream objects and operations, string objects, command-line arguments
- STL (3 examples)
  - vectors, maps, sets, common algorithms, iterators

Namespaces

- Simply a way of packaging things
  - Like packages in Java and Common Lisp
  - To avoid name clashes
  - Standard C++ library uses the namespace **std**
- Three ways of accessing members
- 1) myspace::f( )
- 2) A *using declaration*:
  - using myspace::f; … f( )
- 3) A *using directive*:
  - using namespace myspace; … f( ); (+ more)
  - Use sparingly (never in a header)!

Defining a Namespace

- namespace myspace {
  …
  }
- Can span multiple files
  - That's what the standard headers do

Vectors

- An improvement on the array concept
- Grows as needed
- Generic: can hold any type
- Homogeneous: holds only one type at a time
  - Type-safe (checks proper usage)
- Can be used by standard algorithms

Algorithms

- Generic functions
  - Can process a sequence (vector, deque, list, set, map) of any homogeneous type
  - Including arrays!
- Replace loops in most instances
- Already written for you!
  - Over 50
- Flexible and customizable

The Modern C++ Programmer

- Knows and uses standard library components
- Uses *string* instead of arrays of characters
- Uses *vector* or some other sequence container instead of arrays
- Uses algorithms instead of loops
- Rarely has to worry about memory management
  - The standard library takes care of it for you!

Exercise

- Write a program that reads the lines of a text file, and then prints those lines in descending alphabetical order, without repetition (discard duplicate lines). Your program should be able to handle an arbitrarily large number of text lines. Ignore blank lines altogether.