6.S096 Lecture 10 – Course Recap, Interviews, Advanced Topics

Grab Bag & Perspective

Andre Kessler

Outline

When do you want to use C/C++?

Need a tiny short script?

No, use Python or something else instead.

Need extreme portability with little effort?

No, use an cross-platform interpreted language or Java.

Need the absolute best performance?

Yes.

Need a powerful language for a large software project, integrated with many different libraries?

Yes, C++.

Coding Interviews

C Interviews

Would most likely be focused on the low-level things.

- Security concerns: buffer overflows
- Floating-point subtleties
- Performance (cache efficiency, etc)
- Detecting a memory leak
- Pointers and data structures involving them
- Declaration v definition, compilation

Coding Interviews

C++ Interviews

Would most likely be focused on the concerns of large software projects.

- Questions about design patterns
- How is inheritance implemented? (vtable)
- Properly writing copy constructors
- Knowing important langauge "quirks" or features
- Knowledge of the STL
- Differences between C++ and Java
- Read Effective C++ as prep!

What are the differences between C++ and Java?

Let's list some.

- Java has garbage collection, C++ does not.
- Java passes by value or implicitly by reference.
- C++ can be by value, pointer, or reference.
- C++ allows operator overloading.
- C++ allows multiple inheritance.
- Java runs on the JVM, C++ is compiled to the architecture.

How does this structure look?

Our code is

```
class Base {
  int _a;
public:
  void func();
};

int main() {
  return 0;
}
```

and we compile it with `g++ -onovtable novtable.cpp -fdump-class-hierarchy`.

The Result

```
Class Base
   size=4 align=4
   base size=4 base align=4
Base (0x7f05145d34e0) 0
```

This tells us the the size of class Base is 4, and it should be aligned on word boundaries (locations in memory that are a multiple of 4 bytes).

How does this structure look?

Our code is

```
class Base {
  int _a;
public:
  virtual void func();
};

int main() {
  return 0;
}
```

and we compile it with `g++ -ovtable vtable.cpp -fdump-class-hierarchy`.

The Result

```
Vtable for Base
Base::_ZTV4Base: 3u entries
0     (int (*)(...))0
8     (int (*)(...))(& _ZTI4Base)
16    (int (*)(...))Base::func

Class Base
   size=16 align=8
   base size=12 base align=8
Base (0x7ff9385d44e0) 0
    vptr=((& Base::_ZTV4Base) + 16u)
```

- Notice that class Base now has size 16! (+ 8 byte pointer)
- Should be aligned on multiples of 16 bytes in memory.
- What's a Vtable?

What's a Virtual Table (vtable)?

How C++ really implements inheritance

And more involved...

Our code is

```
class Base {
  int _a;
public:
  virtual void func();
};

class Derived : public Base {
public:
};
```

The Result

```
Vtable for Base
Base::_ZTV4Base: 3u entries
0     (int (*)(...))0
8     (int (*)(...))(& _ZTI4Base)
16    (int (*)(...))Base::func
//...
Vtable for Derived
Derived::_ZTV7Derived: 3u entries
0     (int (*)(...))0
8     (int (*)(...))(& _ZTI7Derived)
16    (int (*)(...))Base::func // points to Base::func!
```

Name Mangling

You'll notice _ZTV7Derived and _ZTI4Base

This will be important for overloading functions: generate a unique symbol identifier for the function.

- For example: `_ZN4Base4funcERi`
- Parse as: _ZN reserved identifier
- 4 Base: 4 character name
- 4 func: 4 character name
- ERi: taking reference to int

There are many different schemes!

In the case of ZTV and ZTI above, ZTV means we're talking about a vtable and ZTI indicates some type info.

Threading and Parallelism

<thread>

OpenMP

MPI

CUDA

Components

Requirements

25%	**Physics Engine** - quality and extensibility of simulation code
25%	**Visualization** - OpenGL; getting a good visualization working
15%	**Unit testing** - gtest, quality and coverage of tests
15%	**Software Process** - code reviews, overall integration of project
10%	**Interactive** - user interactivity with simulation (keyboard, mouse, etc)
10%	**Do something cool** - make it look cool, add a useful feature, do something interesting!

Remember: Extra 5% available in all areas for exceptional effort.

Your Questions

What have you always wanted to know about C or C++?

C++ is a BIG language!

Write more code!

Sharpen your saw with books:

- *Effective C++*, *More Effective C++*, and *Effective STL* by Scott Meyers
- *The C++ Programming Language* by Bjarne Stroustrop
- *C++ Templates: The Complete Guide* by D. Vandervoorde and N. Josuttis
- *Design Patterns* by the Gang of Four
- *Exceptional C++* by Herb Sutter
- *Thinking in C++* by B. Eckel (can find free online)
- *API Design for C++* by Martin Reddy

Wrap-up & Friday

Final project due Sunday

Send me your 2nd code review Saturday please!

Questions?

Let me know what you end up doing with C/C++!