Revenge of the Nerds

Yanan Xiao

Masdar Institute of Science and Technology

Software Engineering Course Presentation

“Programming languages have almost caught up with 1958.”

Paul Graham

A Bit of History

Question about *computation*.

If we had machines that have **infinite** computational power, what problems would we be able to solve?

Lambda calculus.

- A formal system developed by Alonzo Church.
- Essentially a programming language for one of those imaginary machines.
- Equivalent in power with Turing Machine.

Lisp.

- Invented by John McCarthy as an implementation of Alonzo's lambda calculus, in 1958.
- Lisp machine developed by programmers from MIT AI lab, as a native hardware implementation.

Functional Programming ABC

Alonzo Church

- A practical implementation of Alonzo Church's ideas.
- A set of **ideas**, not a set of strict guidelines.
- **A function is a very basic unit in functional programming.**

Functional or Object-Oriented?

Objects are little capsules, containing . . .

- Some internal states.
- A collection of method calls.

Functional programming tries to . . .

- Avoid state changes.
- Works with data flowing between **functions**.

In this manner, functional programming can be considered the opposite of object-oriented programming.

Theoretical and Practical Advantages

Functional design may seem like an odd constraint to work under[1]. Why should you avoid objects and side effects? Some sharp benefits are:

- Formal provability.
- Modularity.
- Composability.
- Ease of debugging and testing.

[1] And it is indeed.

- Formal provability. It’s easier to construct a **mathematical proof** that a functional program is correct.
- Modularity. It forces you to break apart your problem into **small pieces**.
- Composability. Over time you will form a personal library of utilities. It’s because of the modularity benefit.
- Ease of debugging and testing. For debugging: functions are generally small and clearly specified. For testing: each function is a potential subject fir a unit test.

Concurrency

A functional program is ready for concurrency without any further modifications.

Toy Code in Concurrency

```
String s1 = somewhatLongOperation1();
String s2 = somewhatLongOperation2();
String s3 = concatenate(s1, s2);
```

As shown above, even if your application is inherently single threaded, the **compiler**[2] can still optimize functional programs to run on multiple CPUs.

[2] The compiler plays a vital role.

Hot Code Deployment

In