CS 696 Functional Programming and Design
Fall Semester, 2015
Doc 1 Introduction
Aug 25, 2015

Course Issues

http://www.eli.sdsu.edu/courses/index.html

Crashing
Course Web Site
Wiki
Screencasts
Prerequisites
Grading
Clojure

Crash Policy

By seniority

- Measured by SDSU credits on transcript
- Alternate undergrad and graduate students
- 2 undegrad students for each graduate student

Provide unofficial transcript

- Hardcopy or softcopy via email
- Available for free in SDSU portal
- Need at least two hours before start of class

Crash Policy

SDSU students have priority over Open University students

CS majors have priority over non-cs majors

Crash Policy

Start adding students in class Thursday

So don't ask for add code an end of class today

No transcript by Thursday assume you have 0 SDSU units

Must be present to get add code

If miss a class then I drop you from crash list

Crash List FAQ

Why not get a bigger room and admit everyone?

No first hard assignment to scare people

No Grader

Do you really want a 500 level class of 120 people?

Crash List FAQ

Sept 4

Last day for regular students to add/drop classes

Open University students have lower priority than SDSU students

CS 596 Functional & CS 696 Functional

You can not count both courses toward your Masters Degree

Will this be a difficult course?

You need to change how you solve problems in code

Yes very hard

```
public float average(ArrayList data)
    if (data.length == 0) return 0;
    sum = 0;
    for (int k = 0; data.length; k++)
        sum = sum + data[k]
    return sum/data.length;
}
```

```
(defn average
  [data]
  (if (empty? data)
    0
    (/ (reduce + data) (count data))))
```

For Thursday

Watch "Simplicity Matters" by Rich Hickey

http://www.youtube.com/watch?v=rI8tNMsozo0

Functional Thinking with Neal Ford

https://www.youtube.com/watch?v=JeK979aqqqc

ClojureScript Koans

Download Light Table

Goal

Understand the common features of functional programming

Know how to use features of functional programming

Become comfortable using functional programming

Build well designed functional programs

John Carmack

No matter what language you work in, programming in a functional style provides benefits.

You should do it whenever it is convenient, and you should think hard about the decision when it isn’t convenient.

http://www.altdev.co/2012/04/26/functional-programming-in-c/

Some Users of Clojure

Amazon
Apple
Citigroup
CircleCL
Deutche Bank
eBay
Facebook
Groupon
Netflix
Spotify
Staples
Walmart

http://leonid.shevtsov.me/en/companies-that-are-using-clojure

http://dev.clojure.org/display/community/Clojure+Success+Stories

Why the Interest in Functional Programming

Concurrency

One style of programming does not fit all situations

Object-Oriented programming has matured

Problems with Object-oriented programming

Why Clojure

Practical language

Wanted pure functional language

- No imperative/oo programming

Clojure has access to Java

- But controlled access

Clojure, Java & Assignments

Clojure has special syntax to call Java code

You can

- Create Java objects
- Call methods on objects

So you can avoid Functional programming

Point of course is to learn Functional programming

You are not allowed to use Java code in assignments

Problems that use Java receive zero points

What is Functional Programming

Elements of Functional Programming

Pure Functions

First Class Functions

Higher-Order Functions

Immutability

Lazy Evaluation

Recursion

Currying

Memoization

Destructuring

Collection Pipelines

List Compressions

Raw Data + functions

Raw Data + functions

```
class Person {
    private String firstName;
    private String lastName;
    private int age;
```

```
{:first-name “Roger”
 :last-name “Whitney”
 :age 21 }
```

filter (select), remove
map (fold)
reduce
transducers (Clojure 7)

Pure Functions

Functions with no side-effects

Only depend on arguments

Don't change state

Why important

Easier to

debug

test

understand program

```
class Foo {
    int bar

    public int notPure(int y) {
        return bar + y
    }

    public void alsoNotPure(int y) {
        bar = y
    }
```

> OO makes code understandable by encapsulating moving parts.
>
> FP makes code understandable by minimizing moving parts.
>
> Michael Feathers

First Class Functions

Functions can be

- Assigned to variables
- Passed as arguments
- Returned from functions

Why important

- Flexibility
- Generality

Anonymous functions

- Lambdas
- Closures

Higher-Order Functions

Functions that operate on functions

Why important

Fewer details/
higher level logic

Concurrency

Immutability

Data structures can not be modified

Like Java's Strings

> OO makes code understandable by encapsulating moving parts.
>
> FP makes code understandable by minimizing moving parts.
>
> Michael Feathers

Why important

Concurrency

No need for private data

Easier to

- debug
- test
- understand program

Lazy Evaluation

Operations & functions evaluated

- When used
- Not when called

Why important

Simplifies logic