CC3002 Presentation

Alexandre Bergel
http://bergel.eu
12/03/2018

CC3002 Team

Professor

Alexandre Bergel: abergel@dcc.uchile.cl

Office 308 (DCC, 3rd floor, old building)

Monday 12:00 - 13:30 (G306)

Wednesday 14:30 - 16:00 (B204)

Auxiliar (begins next week)

Juan-Pablo Cristián Silva Peña

Friday 12:00 - 13:30 (G306)

Ayudantes

Camacho Orlic, José Tomás, Slater Muñoz, Ignacio Andrés, Caballero Guillén, Marco Antonio

Objectives

Become comfortable with *object oriented programming*: Learn to *think with objects*

Learn the Java programming language: its syntax, idioms, patterns, and styles

Learn the *essentials of the Java class library*, and *learn how to learn* about other parts of the library when you need them

Objectives

Remember that this lecture is *not* a lecture on learning how to program and how to make software

It is an *introductory* course of programming methodology

Objectives

At the end of the semester, you will *not* be an experimented software engineer or even a good programmer

> Years of training and complementary lectures are necessary

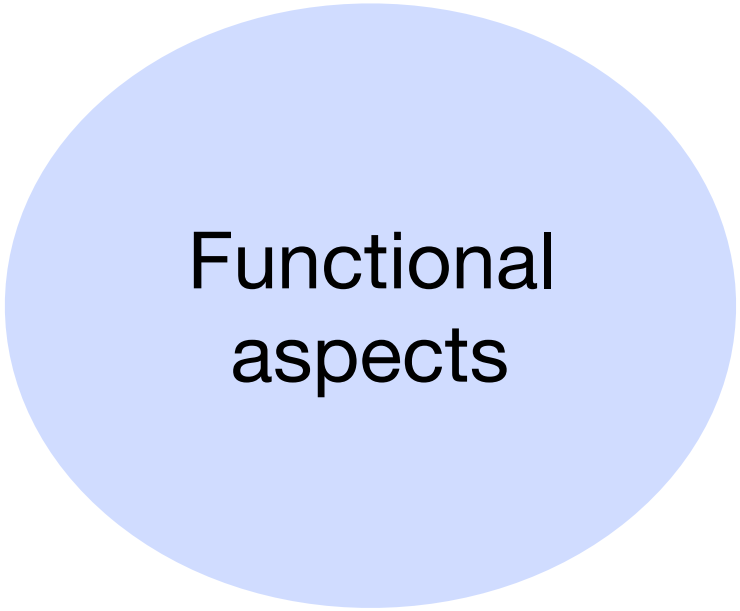
... but at the end of the semester, you will probably be able to realize small (in times, effort, and resources) tasks... and you shouldn’t be ashamed of the code you write

From a previous semester

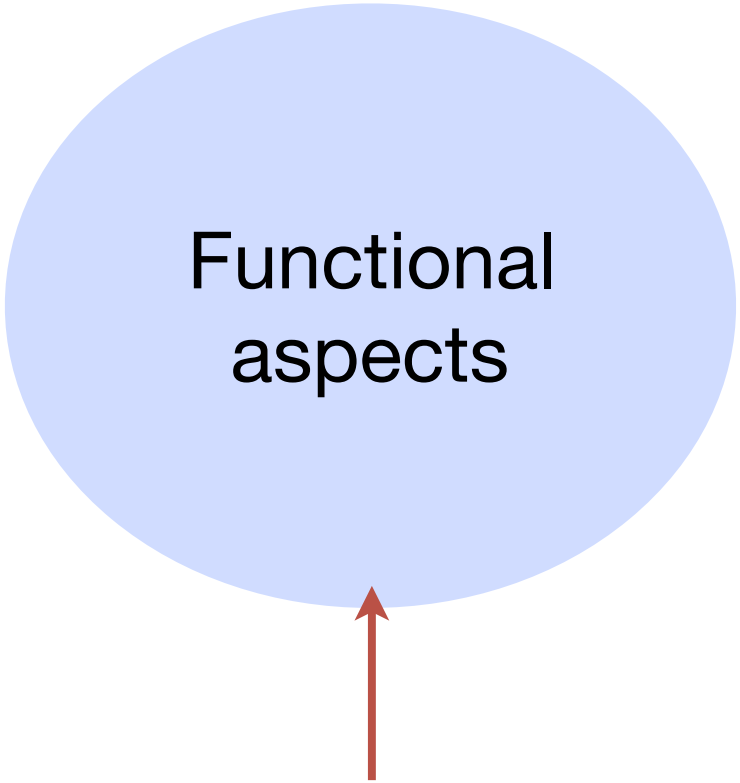
Un amigo que dio el ramo conmigo me dijo que Usted el profesor Alex le "subio su precio de mercado"

Comentario en u-cursos

Software Source code

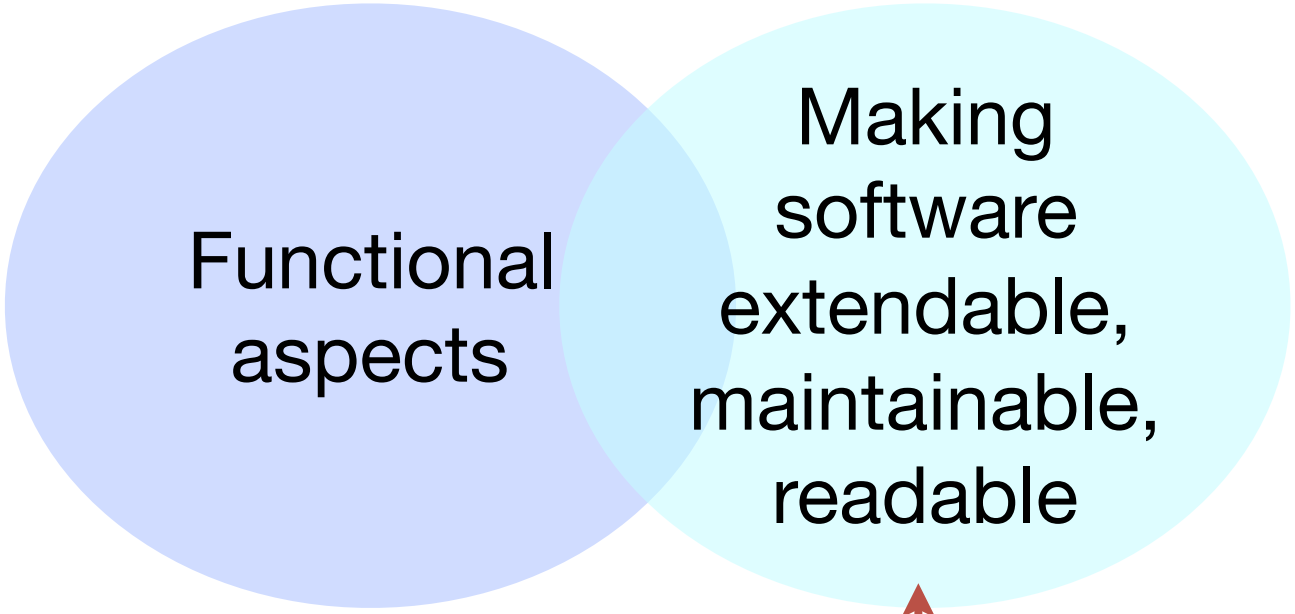


Software Source code



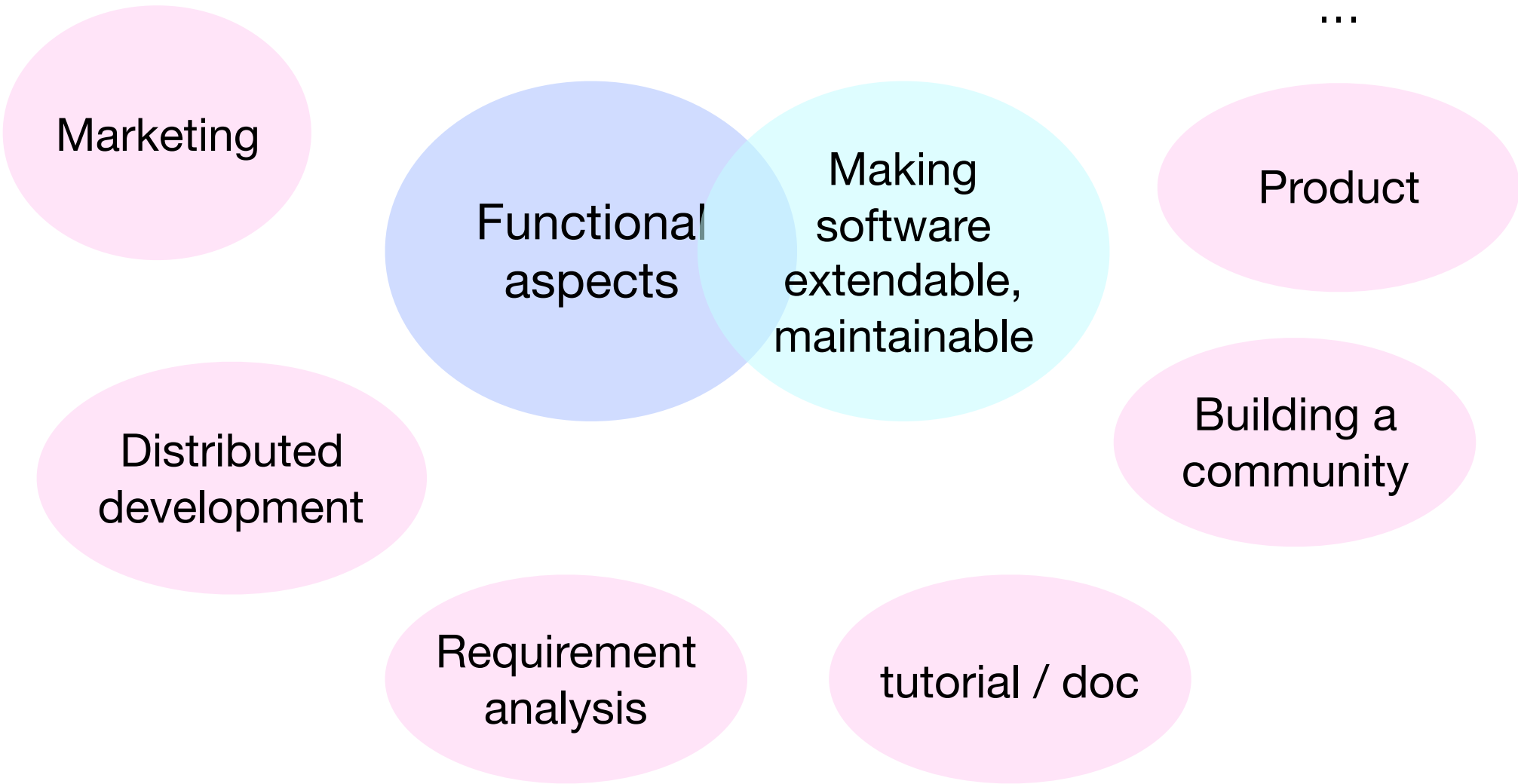
What you have learnt so far
(*e.g.,* writing functions, algorithms,
structure ideas)

Software Source code



What this *ramo CC3002* is all about
(*e.g.,* writing well engineered code, unit testing, methodology, design pattern)

What a software product involve



Important point to keep in mind

It is not enough just to write code that works.

It is as important -- perhaps more important -- *to read code and write code well*; not merely code that works, but code that is legible, maintainable, reusable, fast, and efficient

Practice is key

- Just reading the slides and books is not enough.
- It is like trying to learn French by reading a book

Formalities

Régimen de clases

- 3 horas de cátedra
- 1.5 horas de docencia auxiliar
- 5.5 horas de trabajo personal

Evaluación

- 3 Tareas
- 3 Controles
- Examen
- Mini-ejercicios en clase

Requisitos de Aprobación

Promedio de notas de los controles, las tareas y la nota del examen debe ser >= 4.0

$$NF = (0.6 * PNC + 0.4 * E) * 0.7 + (0.3 * T) + \varepsilon$$

NF = Nota final

PNC = Promedio de Notas de control

E = Nota examen

T = Promedio de Notas de tareas

ε = mini-ejercicios y mini-tareas

Requisitos de Aprobación

Podrán eximirse del examen aquillos que tengan un promedio de controles >= 5.5. En ese caso, la nota del examen será el promedio de los controles

El examen no reemplaza a la peor nota de control

A few words about ethics

ethics: *moral principles* that govern a person's or group's behavior

Things you should *not* do:

ask for extra days for a tarea or a control. You can actually, but only for a good reason (e.g., attending a conference, international project)

complain that you did not understand a lecture without asking for explanation (in my office or via email / foro)

say you authored a piece of work if this is not the case

do the work at the last minute

complain for missing 0.1 puntos for control if you do not have done the optional exercises

...

A few words about ethics

Things you are welcome to do

look on internet for inspiration

do your homework early enough

come to my office to ask for help

ask for help to the auxiliar

attend catedras and auxiliar

use u-cursos to interact with other students

send email in advance when you will be late, need to leave earlier

...

StackOverflow

“A language-independent collaboratively edited question and answer site for programmers.”

You can ask questions and look for answers

Agenda

1. Introduction
2. Dealing with objects
3. A testing framework
4. Debugging and tools
5. Iterative development
6. Inheritance and refactoring

Agenda

7. GUI construction
8. Generics and annotation
9. Advanced topics (threads, VM, test coverage, profiling)


LOUIS ST. LOUIS


5

Pomodoro technique

Great methodology!

Help estimating effort, reduce procrastination, be objective-driven

Help to divide large tasks

Reduce amount of bad code

Pomodoro technique

1 - Choose a task you would like to get done

2 - Set the pomodoro for 25 minutes

3 - Work on the task until the pomodoro rings

4 - When the pomodoro rings, put a tick on a paper

5 - Take a short break

6 - Every 4 pomodoros, take a longer break

http://pomodorotechnique.com