> "In some ways, programming is like painting. You start with a blank canvas and certain basic raw materials. You use a combination of science, art, and craft to determine what to do with them. You sketch out an overall shape, paint the underlying environment, then fill in the details. You constantly step back with a critical eye to view what you've done. Every now and then you'll throw a canvas away and start again." - The Pragmatic Programmer

An Exploration to the *Complex* World of Software Development

Mini workshop for final informatic students University of Lampung
November 2017

Ringkasan

The World as of Now

Multicore CPU saat ini ada di mana-mana, kita sedang menyambut era IoT dimana aplikasi dituntut menjadi terintegrasi dengan aplikasi lain via internet secara `realtime`, `ad-hoc` dan atau `intelligently`. Artinya, bagi pengembang software, domain solusi dan masalah nya kini kian bertambah kompleks, tidak mungkin satu orang bisa menguasai satu sistem secara menyeluruh. Menghadapi hal ini, bahasa-bahasa pemrograman, baik yang lama dan yang baru, sudah banyak menyediakan abstraksi dan pustaka yang memudahkan, namun tidak semua bahasa pemrograman bisa atau mampu memberikan dukungan yang baik untuk membuat proses desain program menjadi lebih sederhana.

> "Simplicity is a prerequisite for reliability." - E.W Dijkstra

Object Oriented Programming

Banyak orang ternyata kemudian mulai menyadari, bahwa, ada banyak software baru yang dibuat dengan bahasa pemrograman berorientasi objek (OOP) menjadi sangat rumit, kompleks dan sulit dipelihara.

> "Object-oriented programming, whose essence is nothing more than programming using data with associated behaviors, is a powerful idea. It truly is. But it's not always the best idea. … Sometimes data is just data and functions are just functions." - Rob Pike

> "I think that large objected-oriented programs struggle with increasing complexity as you build this large object graph of mutable objects. You know, trying to understand and keep in your mind what will happen when you call a method and what will the side effects be." - Rich Hickey

> "Object-oriented programming offers a sustainable way to write spaghetti code." - Paul Graham

Functional Programming

Functional programming menggunakan paradigma yang berbeda dibanding bahasa pemrograman imperative dan object-oriented. Yang paling mendasar adalah ia menggunakan *side-effect free* **function** sebagai building-block nya. Hal ini membuat banyak hal seperti *concurrency* menjadi mudah namun pada sisi lain banyak pula kemudahan yang dikorbankan untuk alasan *safety*, *correctness* dan *long maintainability*. Keunggulan utama lainnya dari functional programming adalah bahwa eksekusi dari side-effect free function bisa tidak bergantung pada urutan pemanggilannya, hal ini mengijinkan compiler atau virtual machine pengeksekusinya melakukan optimasi, parallelization secara transparan.

Namun demikian, nyatanya, tidak bisa pula secara serta merta, atau memang belum saat nya, paradigma functional programming ini menggantikan imperative programming secara total. Jadi, untuk saat ini, baiknya bagi kita untuk memprediksi dan merasa bahwa bahasa pemrograman *hybrd* yang mengusung keunggulan functional programming namun mendukung pula pendekatan object-oriented secara terkontrol menjadi lebih **praktis** untuk digunakan dibanding bahasa pemrograman yang **strict**, *purely* functional.

> "Programming is not about typing, it's about thinking." - Rich Hickey

By definition, jika suatu bahasa pemrograman tidak memperkenankan *side effect free* function bercampur dengan *side effect full* function, maka dia kita sebut sebagai **pure** functional. `Haskell`, `Purescript`, `Idris` adalah beberapa diantaranya.

Bahasa pemrograman functional *hybrid* yang lebih praktis diantaranya adalah `Ocaml`, `Scala`, `Clojure` dan banyak lagi, bahkan hampir semua bahasa pemrograman *main stream* saat ini sudah memberikan dukungan penerapan paradigma functional.

What we will do?

Dalam workshop ini kita akan menggunakan `Clojure` untuk mendemonstrasikan bagaimana menghadapi tantangan pengembangan software yang kian kompleks dengan cara, konsep yang **simple** dan **elegant**.

> "A language that doesn't affect the way you think about programming, is not worth knowing." - Alan Perlis

Agenda

- Intro, Into `Emacs` the powerful text editor (30 minutes)

- Going functional with `Clojure` (60 minutes)
- `Clojurescript`, the Web and `ReactJS` (40 minutes)
- React Native with `Clojurescript` (30 minutes)
- Closing, Clj & Cljs Goodies (20 minutes)

Prasyarat

- Partisipan adalah programmer atau siap berprofesi di dunia rekayasa software
- Mempunyai komputer/laptop Linux sendiri sangat dianjurkan
- Mengetahui atau sudah pernah mengerjakan proyek pengembangan software
- Mampu menggunakan dan pelakukan penyesuaian sistem operasi, text editor atau IDE dalam bekerja, bisa menggunakan `Emacs` atau `Vim` sangat dianjurkan
- Sudah membaca dan sedianya mengamalkan dokumen-dokumen yang dilampirkan, dan dianjurkan untuk mencari sendiri sumber-sumber lain yang berkaitan dengan bahasa pemrograman `Clojure`, diantaranya:
  - Clojure tutorial in Bahasa Indonesia (video version)
  - Clojure for the Brave and True

Lampiran

Linux & Emacs:

1. "`IlmuKomputer_Cepat Mahir Linux.pdf`"
2. `dokumen.tips_tutorial-emacs-bahasa-indonesia-by-sopier.pdf`

Clojure & Functional Programming: *(Sebaiknya dibaca secara berurutan!)*

1. `RichHickey_SimpleMadeEasy.pdf`
2. `Clojure.pdf`
3. `ClojureSlides.pdf`
4. "`Clojure tutorial by Rich Hickey.pdf`"