# Introduction to Functional Programming in *OCaml*

**Roberto Di Cosmo, Yann Régis-Gianas, Ralf Treinen**

**Week 0 - Sequence 4:**

**The *OCaml system* : a bird's eye view**





OCaml PRO

# A mature system

A rich set of development tools

- modern package manager
  for installing the libraries you need
  and speed up development
  see `http://opam.ocaml.org`
- hundreds of packaged libraries
  see `http://opam.ocaml.org/packages/`
- debugger
- profiler
- ...

# A mature system

A rich set of compiler tools

- REPL (Read-Evaluate-Print Loop)
  for fast development
- `ocamlc` bytecode compiler
  for portable code, see
  `http://caml.inria.fr/ocaml/portability.en.html`
- `ocamlopt` native compiler (AMD64,IA32,Power PC,ARM)
  for very fast executables
- `js_of_ocaml` compiler to JavaScript
  for building Web applications

# A *REPL* is cool

*OCaml* has a full-fledged Read-Evaluate-Print Loop, called *toplevel* by *OCaml* programmers, that

- reads your program, *phrase* by *phrase*
- compiles it on the fly, reporting any error found,
- evaluates it
- prints the results

This means that you can see the results produced by your program, in the toplevel, without writing a printer.

# Meet the *OCaml toplevel*

A typical interaction looks like the following one

```
>ocaml
        OCaml version 4.02.0

# List.map (fun x -> x+1) [1;2;3;4;5;6];;
- : int list = [2; 3; 4; 5; 6; 7]
```

The result is right in front of our eyes.

# The *OCaml toplevel* for the course

For this course

You will run the *OCaml* toplevel right in your browser!

- no need to install anything
- same interface for everybody
- fully integrated in the learning system
- no need to depend on an external server

Looks like magic?

- toplevel written in *OCaml*
- compiled into bytecode using `ocamlc`
- compiled into JavaScript using `js_of_ocaml`
- loaded into your browser when accessing the web page

# Time to try this out