CMSC 330: Organization of Programming Languages

Working with OCaml

1

CMSC330 Summer 2018

## **OCaml Compiler**

- OCaml programs can be compiled using ocamlc
  - Produces .cmo ("compiled object") and .cmi ("compiled interface") files
    - We'll talk about interface files later
  - By default, also links to produce executable a.out
    - Use -o to set output file name
    - Use -c to compile only to .cmo/.cmi and not to link
- Can also compile with ocamlopt
  - Produces .cmx files, which contain native code
  - Faster, but not platform-independent (or as easily debugged)

**OCaml Compiler** 

• Compiling and running the following small program:

```
hello.ml:
  (* A small OCaml program *)
  print_string "Hello world!\n";;
```

```
% ocamlc hello.ml
% ./a.out
Hello world!
%
```

## **OCaml Compiler: Multiple Files**

<u>main.ml</u>:

```
let main () =
    print_int (Util.add 10 20);
    print_string "\n"
let () = main ()
```

<u>util.ml</u>:

let add x y = x+y

- Compile both together (produces a.out) ocamlc util.ml main.ml
- Or compile separately

   ocamlc -c util.ml
   ocamlc util.cmo main.ml
- To execute

## OCamlbuild

- Use **ocamlbuild** to compile larger projects and automatically find dependencies
- Build a bytecode executable out of main.ml and its local dependencies

ocamlbuild main.byte

• The executable main.byte is in \_build folder. To execute:

./main.byte

### **OCaml Top-level**

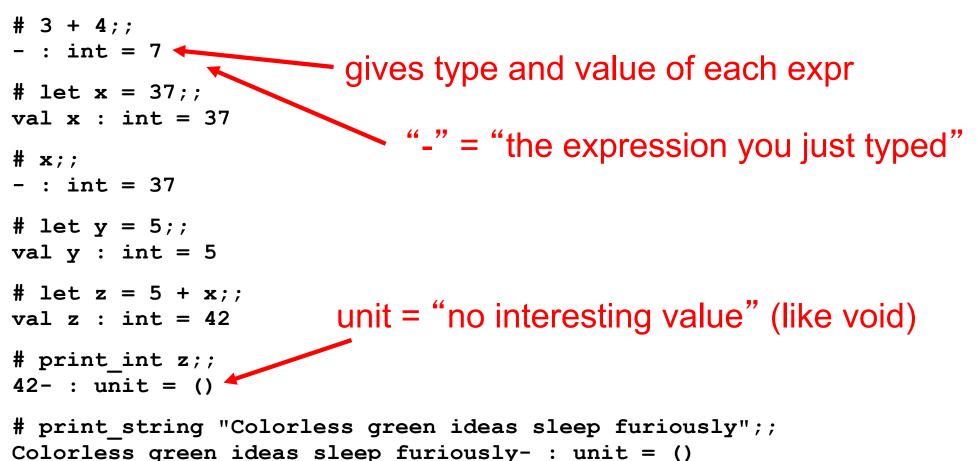
- The *top-level* is a read-eval-print loop (REPL) for OCaml
   Like Ruby's irb
- Start the top-level with the ocaml command: ocaml

```
OCaml version 4.04.0
# print_string "Hello world!\n";;
Hello world!
- : unit = ()
#
```

 To exit the top-level, type ^D (Control D) or call the exit 0 # exit 0;;

# OCaml Top-level (cont'd)

Expressions can also be typed and evaluated at the top-level:



# print\_int "Colorless green ideas sleep furiously";;
This expression has type string but is here used with type int

#### Loading files

File hello.ml:

print\_string "Hello world!\n";;

• Load a file into top-level

#use "filename.ml"

• Example:

#use loads in a file one line at a time

# #use "hello.ml";;

Hello world!

#

### A Note on ;;

- ;; ends an expression in the top-level of OCaml
  - Use it to say: "Give me the value of this expression"
  - Not used in the body of a function
  - Not needed after each function definition
    - Though for now it won't hurt if used there
- There is also a single semi-colon ; in OCaml
  - But we won't need it for now
  - It's only useful when programming imperatively, i.e., with side effects
    - Which we won't do for a while