Linguagens e Ambientes de Programação (Aula Teórica 2)

LEI - Licenciatura em Engenharia Informática

João Costa Seco (joao.seco@fct.unl.pt)

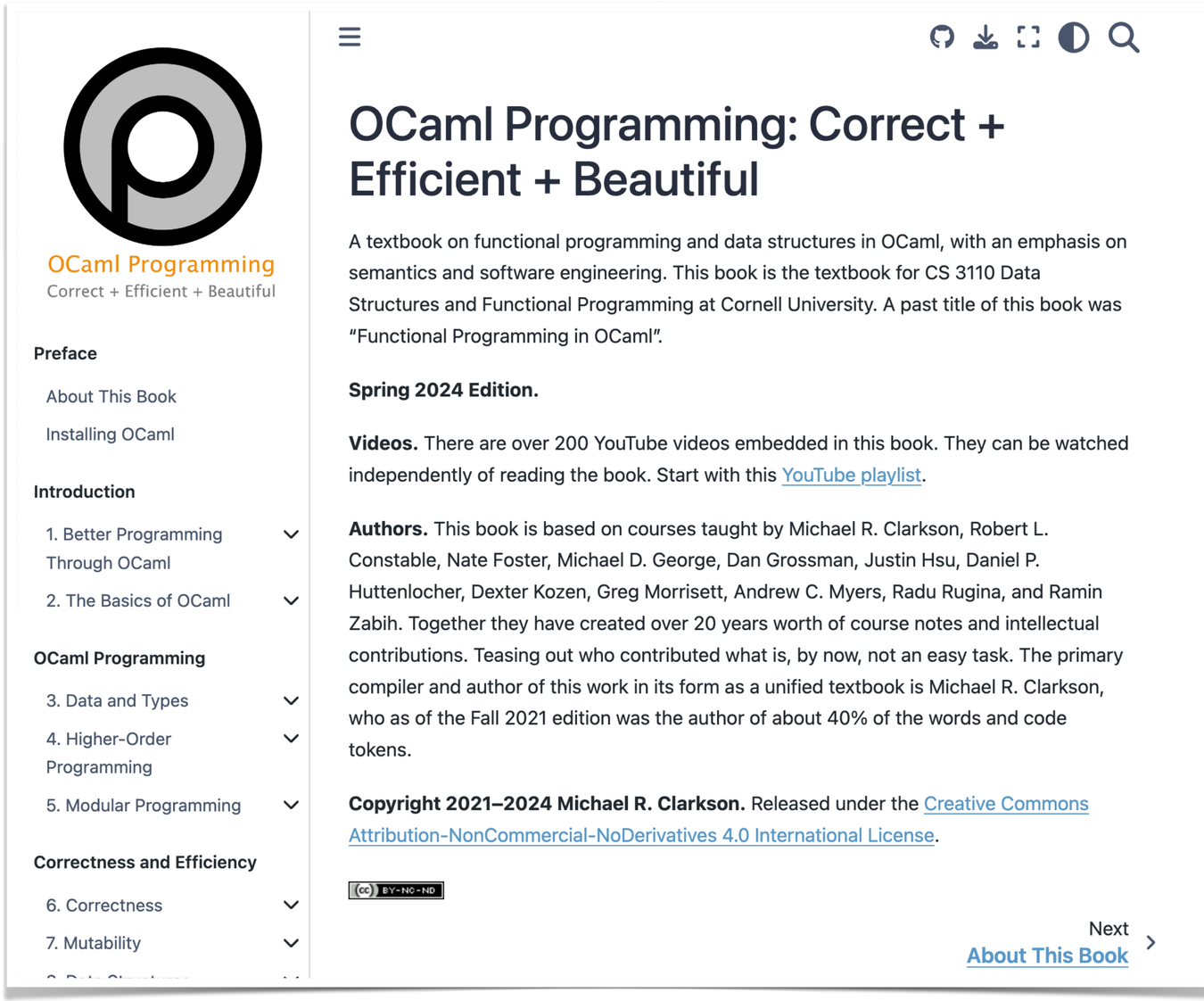
NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

Agenda para hoje

- Introdução à programação funcional.
- A linguagem OCaml.
- Expressões, variáveis e tipos.

Livro de texto

- OCaml Programming:
  Correct + Efficient + Beautiful
- Cornell University
- Michael R. Clarkson et al.
- Online resources
  (book, exercises, videos)



Principais Características da Programação Funcional

- É uma programação declarativa (especificar “o que é", em vez de “como é”)
- As construções típicas são:
  - Todas as construções são **expressões**
  - Funções como valores da linguagem
  - Composicionalidade
  - Valores imutáveis
  - Sistema de tipos forte
- Construções usadas noutras linguagens.

“A language that doesn’t affect the way you think about programming is not worth knowing.”

—Alan J. Perlis (1922-1990), first recipient of the Turing Award

OCaml Programming

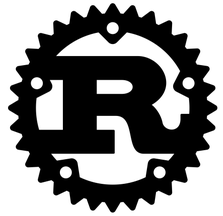
Correct + Efficient + Beautiful

Gradiente do paradigma funcional em linguagens

- Linguagens Funcionais Puras (Haskell)

- Linguagens Funcionais com construções Imperativas (OCaml)

- Linguagens Imperativas com mecanismos Funcionais (Rust)



- Linguagens Imperativas (procedimentais ou object oriented) (Java <17)

Linguagens Funcionais (logo game)

ML (OCaml) is logic in programming

$$\frac{A \longrightarrow B \qquad A}{B}$$

ML (OCaml) is logic in programming

$$\frac{f : A \longrightarrow B \quad x : A}{f\ x : B}$$

Example in Java

```
B f1(A a) {...}

C f2(B b) {...}

D f3(C c) {...}

D d = f3(f2(f1(new A())))
```

$$f1 : A \longrightarrow B$$
$$f2 : B \longrightarrow C$$
$$f3 : C \longrightarrow D$$

$$\dfrac{\dfrac{\dfrac{A \quad A \longrightarrow B}{B} \quad B \longrightarrow C}{C} \quad C \longrightarrow D}{D}$$

OCaml

- É um dialecto da família ML
- É uma linguagem de expressões que:
  - Ou denotam um valor,
  - Ou terminam com uma exceção
  - Ou não terminam…
- É fortemente tipificada com inferência de tipos

facebook Microsoft docker

Jane Street Bloomberg ahrefs

Xavier Leroy

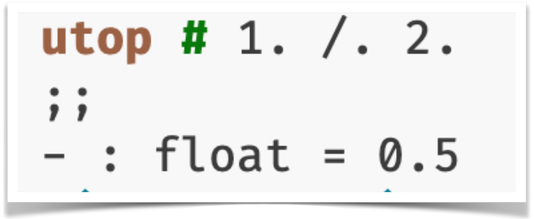
Didier Rémy

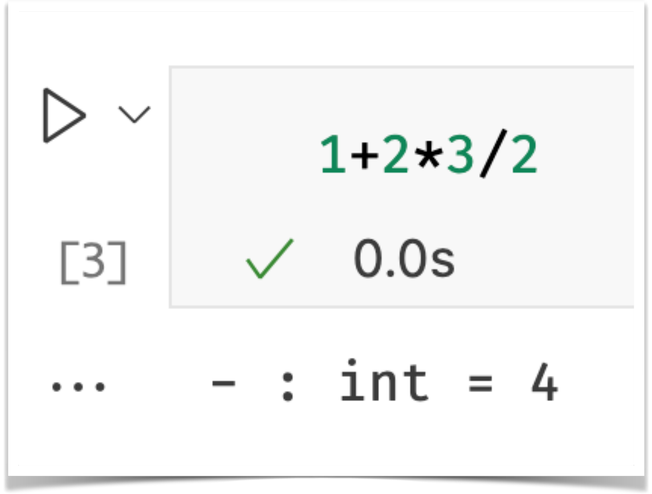
Damien Doligez

Jérôme Vouillon

Ambientes de programação

- Interpretador: `ocaml` / `utop`
- Compilador: `ocamlc` + `make/dune`
- Visual Studio Code + OCaml plugin
- Jupyter Notebook + OCaml kernel





Tipos básicos, literais, operadores e funções

Tipo	Literal	Operador	Função
int	1 2 3 0b101 0x42	+ - * /	
float	1. 2. 3.5 4e10	+. -. *. /.	float_of_int
bool	true false	&& ||	
char	'a' 'b'		char_of_int, int_of_char
strings	"hello" ""	^	string_of_int
unit	()		

```
[4]  3.14 * 2.
     0.0s
...  File "[4]", line 1, characters 0-4:
     1 | 3.14 * 2.
         ^^^^
     Error: This expression has type float but an expression was expected of type
              int
```

Tipos básicos, literais, operadores e funções

Tipo	Literal	Operador	Função
int	1 2 3 0b101 0x42	+ - * /	
float	1. 2. 3.5 4e10	+. -. *. /.	float_of_int
bool	true false	&& ||	
char	'a' 'b'		char_of_int, int_of_char
strings	"hello" ""	^	string_of_int
unit	()		

```
[5]  3.14 *. (float_of_int 2)
     ✓ 0.0s
...  - : float = 6.28
```

Tipos básicos, literais, operadores e funções

Tipo	Literal	Operador	Função
int	1 2 3 0b101 0x42	+ - * /	
float	1. 2. 3.5 4e10	+. -. *. /.	float_of_int
bool	true false	&& ||	
char	'a' 'b'		char_of_int, int_of_char
strings	"hello" ""	^	string_of_int
unit	()		

```
int_of_string "not an int"
```

[6] ⊗ 0.9s

```
Exception: Failure "int_of_string".
Raised by primitive operation at unknown location
Called from Stdlib__fun.protect in file "fun.ml", line 33, characters 8-15
Re-raised at Stdlib__fun.protect in file "fun.ml", line 38, characters 6-52
Called from Toploop.load_lambda in file "toplevel/toploop.ml", line 212, characters 4-150
```

Tipos básicos, literais, operadores e funções

Tipo	Literal	Operador	Função
int	1 2 3 0b101 0x42	+ - * /	
float	1. 2. 3.5 4e10	+. -. *. /.	float_of_int
bool	true false	&& ||	
char	'a' 'b'		char_of_int, int_of_char
strings	"hello" ""	^	string_of_int
unit	()		

```
"abc".[0]
```

[7] ✓ 0.0s

```
- : char = 'a'
```

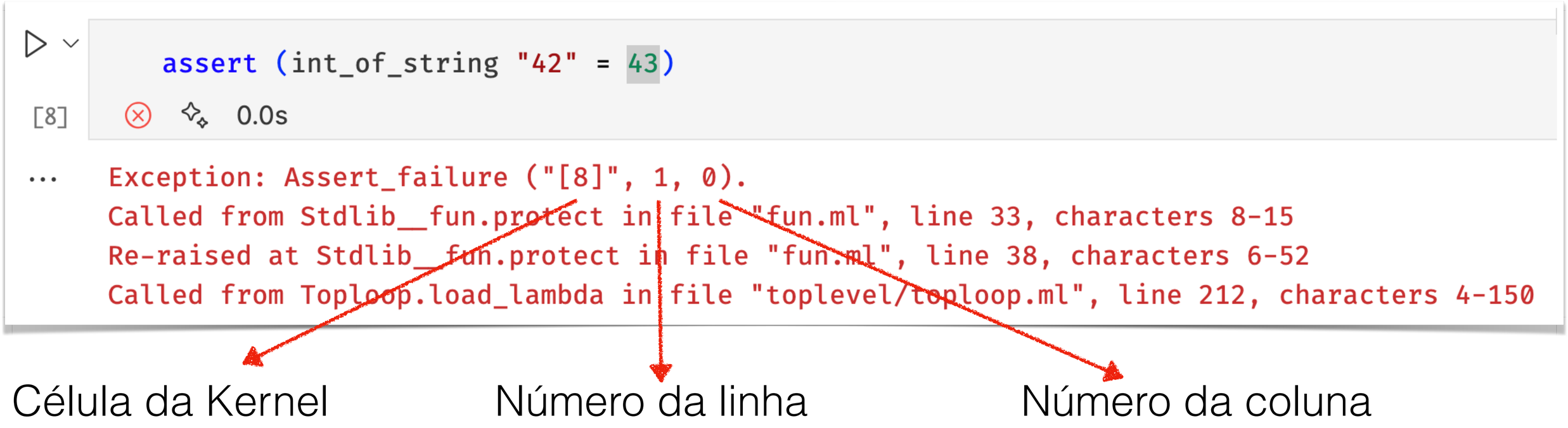
Igual e Igual

- A igualdade em OCaml pode ter inúmeros sentidos, pode ser redefinida.
- No entanto, a linguagem permite duas formas nativas de testar a igualdade.
- Pode ser de forma **estrutural** onde dois valores têm a mesma estrutura e conteúdo.
- Ou então de forma **física**, onde dois valores apontam para o mesmo objeto em memória.

Comparação	Estrutural	Física
Igualdade	=	==
Desigualdade	<>	!=
Compara	Conteúdos	Posições de memória
Exemplo	`[1;2;3] = [1;2;3]`	`[1;2;3] == [1;2;3]`

Asserções

- Asserções permitem o teste de uma expressão, lançando uma excepção no caso **falso**.



Expressões condicionais

- As expressões condicionais são expressões onde os dois ramos têm o mesmo tipo.

```
[9]   4 + (if 'a' = 'b' then 1 else 2)
      ✓ 0.0s
...   - : int = 6
```

- Note: o ramo **`else`** é obrigatório para todos os tipos excepto **`unit`**.

```
[10]  if "hello" = "world" then 1
      ⊗ 0.0s
...   File "[10]", line 1, characters 26-27:
      1 | if "hello" = "world" then 1
                                 ^

      Error: This expression has type int but an expression was expected of type
               unit
             because it is in the result of a conditional with no else branch
```