Exploring OCaml Basics

Harshil Goyal

January 19, 2025

Example 1

Original Expression and Output

```
let u = [1; 2; 3; 4];;
val u : int list = [1; 2; 3; 4]
```

Changed Expression and Output

```
utop # let u = [1; 3; 1.2; 4];;
Error: The constant 1.2 has type float but an expression was
   expected of type int
```

Explanation

List elements in OCaml must all have the same type.

Example 2

Original Expression and Output

```
let dummy = "hi" = "hello";;
val dummy : bool = false
```

Changed Expression and Output

```
utop # let dummy = "hi" = 5;;
Error: The constant 5 has type int but an expression was expected
   of type string
```

Explanation

While comparing values directly, they must be of the same type in OCaml.

Example 3

Original Expression and Output

```
 let a = 1 in let b = 2 in a + b;;
- : int = 3
```

Changed Expression and Output

```
utop # let a = 1 in let b = 2 in a + "b";;
Error: This constant has type string but an expression was
   expected of type int
```

Explanation

The + operator in OCaml is only for integer addition, so both of the operands must be of type *int*

Example 4

Original Expression and Output

```
utop # let square x = x * x;;
val square : int -> int = <fun>
```

Changed Expression and Output

```
utop # let square x = x * 1.0;;
Error: The constant 1.0 has type float but an expression was
   expected of type
         int
```

Explanation

$x * 1.0$ tries to multiply x of type *int*(by the function's perspective) with 1.0 of type *float* but the $*$ operator expects both the operands to be of type *int*.

Example 5

Original Expression and Output

```
utop # String.ends_with;;
- : suffix:string -> string -> bool = <fun>

utop # String.ends_with ~suffix:"less" "stateless";;
- : bool = true
```

Changed Expression and Output

```
utop # String.ends_with ~sufix:"less" "stateless";;
Error: The function applied to this argument has type suffix:
   string -> bool
This argument cannot be applied with label ~sufix
```

Explanation

Using wrong spelling for the label (in this case "*sufix*" instead "*suffix*"), the label is not recognized as a valid label.

Example 6

Original Expression and Output

```
utop # (fun x -> x * x * x) 10;;
- : int = 1000
```

Changed Expression and Output

```
utop # fun x -> x * x * x;;
- : int -> int = <fun>
utop # fun 10;;
Error: Syntax error
```

Explanation

fun keyword in OCaml is used to define anonymous functions, but not to call functions. Here *fun 10* is invalid because :

- *fun* should be declared and called at the same time i.e (fun x-> x*x*x) 10;
- *fun* cannot directly be applied to the argument defining the body.

Example 7

Original Expression and Output

```
utop # List.map (fun x -> x * x) [0;1;2;3;4;5;6];;
- : int list = [0; 1; 4; 9; 16; 25; 36]
```

Changed Expression and Output

```
utop # List.map [0;1;2;3;4;5;6] (fun x -> x * x);;
Error: This expression should not be a list literal, the expected
   type is
       'a -> 'b
```

Explanation

- *List.map* expects the first argument to be a function.
- However, the list *[0;1;2;3;4;5;6]* was provided first, which according to compiler should be a function, but list cannot be applied as function.