

State

Dr. Mattox Beckman

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
DEPARTMENT OF COMPUTER SCIENCE

Objectives

- ▶ Explain how to use `let` and function declarations to control when a variable is created.
- ▶ Use functions to encapsulate state in a safe manner.

Local Variable Example

```
1 # let foo x =  
2     let a = 10 + 20 in  
3         a + x;;  
4 val foo : int -> int = <fun>  
5 # foo 15;;  
6 - : int = 45  
7 # foo 30;;  
8 - : int = 60
```

How many times does the `10 + 20` get computed?

Global Variable Example

```
1 # let a = 10 + 20;;
2 val a : int = 30
3 # let foo x =
4     a + x;;
5 val foo : int -> int = <fun>
6 # foo 15;;
7 - : int = 45
8 # foo 30;;
9 - : int = 60
```

How many times does the `10 + 20` get computed?

Encapsulated Variable Example

```
1 # let foo =  
2     let a = 10 + 20 in  
3         fun x -> a + x;;  
4 val foo : int -> int = <fun>  
5 # foo 15;;  
6 - : int = 45  
7 # foo 30;;  
8 - : int = 60
```

How many times does the `10 + 20` get computed?

Using Local State

```
1 # let counter =  
2   let ct = ref 0 in  
3   fun () -> ct := !ct + 1; !ct;;  
4 val counter : unit -> int = <fun>  
5 # counter ();;  
6 - : int = 1  
7 # counter ();;  
8 - : int = 2
```

- ▶ This protects `ct`, making it available only to `counter`.

Bad Pun

```
1 # fun twice f x = f (f x)
2 # twice counter () + twice counter ();
3 res4 : Int = 6
4 # twice counter () + twice counter ();
5 res4 : Int = 14
```

- ▶ Function `twice` is the Church numeral for 2.
- ▶ You know what this means, right?

Random Number Generators

```
1 # let mkRandom s =  
2     let s = ref s in  
3     fun () -> s := (!s * 541 + 5) mod 1024; !s;;  
4 val mkRandom : int ref -> unit -> int = <fun>  
5 # let rnd0 = mkRandom (ref 1);;  
6 val rnd0 : unit -> int = <fun>  
7 # rnd0 ();;  
8 - : int = 546  
9 # rnd0 ();;  
10 - : int = 479  
11 # rnd0 ();;  
12 - : int = 72
```


Function Tuples

```
1 # let (counter, reset) =
2     let ct = ref 0 in
3         (fun () -> ct := !ct + 1; !ct),
4         (fun nv -> ct := nv);;
5 val counter : unit -> int = <fun>
6 val reset : int -> unit = <fun>
7 # counter ();;
8 - : int = 1
9 # reset 5;;
10 - : unit = ()
11 # counter ();;
12 - : int = 6
```