Scope and Memory Management

CSCI 334
Stephen Freund

Inline Blocks
{  
  int x = 2;
  int y = 10
  {
    int z = 2;
    int x = 3;
    x = z + y;
  }
  print x;
}

Simplified Machine Model

Control link
x              2
Environment
Pointer

Inline Blocks
{  
  int x = 2;
  int y = 10
  {
    int z = 2;
    int x = 3;
    x = z + y;
  }
  print x;
}

Declarations
val Pi = 3.14;
fun for(lo,hi,f) =
  ...
fun build(...) =
  ...

Control link
Pi              3.14
Environment
Pointer

Function Calls
  1 int sumSquares(int n) {
  2    int i, sum = 0;
  3    for (i = 0; i < n; i++)
  4        sum = sum + i * i;
  5    return sum;
  6  }
  7  ...
  8  {
  9    int x = sumSquares(15);
 10    print x;
 11  }
fun fact(n) = 
  if n <= 1 then 1 
  else fact(n-1)*n;
val y = fact(2);

fun fact(n) = 
  if n <= 1 then 1 
  else fact(n-1)*n;
val y = fact(2);

fun fact(n) = 
  if n <= 1 then 1 
  else fact(n-1)*n;
val y = fact(2);

fun fact(n) = 
  if n <= 1 then 1 
  else fact(n-1)*n;
val y = fact(2);

fun fact(n) = 
  if n <= 1 then 1 
  else fact(n-1)*n;
val y = fact(2);

fun swap(x, y) = 
  let val t = !x in 
  x := !y; y := t 
  end;
val a = ref 1;
val b = ref 2;
swap(a,b);

fun add(x, y) = 
  x + y;
val a = ref 1;
val b = ref 2;
add(!a, !b);

---

**Parameter Passing in ML**

**By Ref**

fun swap(x, y) = 
  let val t = !x in 
  x := !y; y := t 
  end;
val a = ref 1;
val b = ref 2;
swap(a,b);

**By Val**

fun add(x, y) = 
  x + y;
val a = ref 1;
val b = ref 2;
add(!a, !b);

val a = ref 1;
val b = ref 2;
add(a, b);
Why Does it Matter?

- Side Effects
- Aliasing
  ```
  int add(x, y) {
    x = x + 1;
    return x + y;
  }
  z = 5;
  print add(z, z);
  ```
- Efficiency
  add(z, z) by val
  ```
  | x   | 5 |
  | y   | 5 |
  | z   | 5 |
  ```

Accessing Globals

```plaintext
val m = 5;
fun force(a) = m * a;
fun cow(y) =
  let m = y * y in
  force(m)
  end;
cow(10);
```

```plaintext
Accessing Globals

```plaintext
val m = 5;
fun force(a) = m * a;
fun cow(y) =
  let m = y * y in
  force(m)
  end;
cow(10);
```  
Dynamic Scope: follow control links

Examples of Dynamic Scoping

```plaintext
fun formatBuffer(buffer) =
  ... setColor(highlightColor) ...

let highlightColor = Blue in
  formatBuffer(b);
```  
```
fun playGame() =
  ... if strategy(...) = goLeft then ... 

let fun strategy(...) = ...
  in playGame();
```  
```
Stack Inspection

- Permission depends on:
  - permission of calling method
  - permission of all methods above it on stack

void open(String s) {
  SecurityManager.checkRead();
  ...
}
Stack Inspection

- Permission depends on:
  - permission of calling method
  - permission of all methods above it on stack

void open(String s) {
  SecurityManager.checkRead();
  ...
}

Fails if Applet code is not trusted

FileInputStream.open("/etc/passwd")

System.main()
Applet.run()
Applet.sneaky()

Accessing Globals

val m = 5;

fun force(a) = m * a;
fun cow(y) =
  let m = y * y in
  force(m)
end;

fun moo(y) =
  cow(y);
moo(10);

moo(10);
force(100)

m
5
control link

force

m
100
cow

moo

y
10
control link

moo(10)

force

a
100
control link