

maze

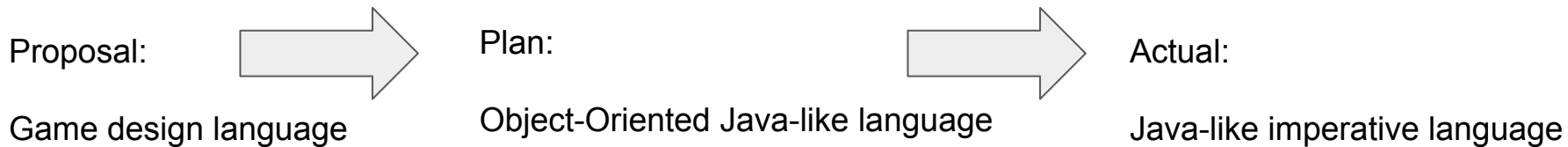


Alexander Brown (aab2212)
Alexander Freemantle (asf2161)
Michelle Navarro (mn2614)
Lindsay Schiminske (ls3245)

Introduction

What is maze?

Language Evolution





An Imperative Language with Java-like syntax

Tools



Escribiendo Co... #actuallyimportant

Channels: #actuallyimportant, #classes, #coolapps, #general, #github, #interviewquestions, #linkedin, #maze, #memes, #networks, #nostalgialtra, #notes, #random, #spiritanimals, #t-shirt, #vim, #winenights, #workworkworkwork...

Direct Messages: slackbot, michellenavarro (you), abrown, afreemantle, linsk3, Invite people

Message #actuallyimportant

October 24th

afreemantle Dec 7th
<https://lvm.moe/ocaml/Lvm.html>

afreemantle Oct 20th
https://drive.google.com/open?id=0BY_PBYEDL6UIRIRuS3IwCG1za2s

michellenavarro Oct 21th
<http://www.cs.columbia.edu/~sedwards/classes/2015/413-fall/rlm/Dice.pdf>

LRM Document from Google Drive

1/4 Members
 michellenavarro (you), abrown, afreemantle, linsk3

Shared Files

Notification Preferences

afreemantle 10:38 PM
 added a Pin in Todo (pinpoint: [todo_maze.txt](#))

- Object creation (New)
-
- Beerly analyzer
- presentation
- paper
- interesting program

afreemantle 5:46 PM
 shared a file

todo Document from Google Drive

afreemantle 5:46 PM
 todo

```
Terminal
plt4115@plt4115: ~/maze
1 1 Codegen for MAZE - based off of MicroC *
2 module L = Lvm
3 module A = Ast
4
5
6 module StringMap = Map.Make(String)
7
8
9 let translate (classes) =
10
11   let grab_body someClass =
12     someClass.A.dbody
13   in
14
15   let rec grab_fcn_lists = function
16     [] -> []
17   | [X] -> let y = x.A.methods in y
18   | head :: tail -> let r = head.A.methods in r @ grab_fcn_lists tail
19   in
20
21   let dbodies = List.map grab_body classes in
22   let functions = grab_fcn_lists dbodies in
23
24   let context = L.global_context () in
25   let the_module = L.create_module context "maze"
26
27   and i32_t = L.i32_type context (* int *)
28   and i8_t = L.i8_type context (* printf *)
29   and i1_t = L.i1_type context (* bool *)
30   and f_t = L.double_type context (* float *)
31   and void_t = L.void_type context (* void *)
32   and str_t = L.pointer_type (L.i8_type context) in
```

Oct 2, 2016 – Dec 19, 2016

Contributions to master, excluding merge commits

Contributions: Commits



Files

- todo
- PLT ROUGH propos...
- maze Final Present...
- LRM
- Final Report

MAZE Architecture

program.maze



Scanner



Parser



AST



Analyzer



Codegen



LLVM IR

Syntax

Basics

Primitives

```
int  
float  
bool  
char
```

```
string
```

Comments

```
(* comment *)
```

Binary Ops

```
+ - * /  
< <= > >=  
== != && ||
```

Unary Ops

```
! -
```

Methods

```
class test {  
  
    void dummy(){  
        int x;  
        x = 1;  
        print(x);  
    }  
  
    void main(){  
        dummy();  
    }  
}
```

Statements

If / else

```
class test{
    void main(){
        int x;
        x = 0;

        if(x>1){
            x = x - 1;
        }
        else if(x == 1){
            x = x * 3;
        }
        else{
            x = x + 5;
            print(x);
        }
    }
}
```

Return

```
class test {

    int computeValue(int x, int y){
        return x + y;
    }

    void main(){

        int a;
        a = computeValue(10,5);
        print(a);
    }
}
```

While Loop

```
class test {
    void main () {

        int i;
        i = 0;

        while(i < 5){
            print("while loop");
            i = i + 1;
        }
    }
}
```

Testing

Menhir

\$ menhir --interpret --interpret-show-cst parser.mly

--explain

```
CLASS ID LBRACE INT ID LPAREN RPAREN LBRACE RBRACE RBRACE
ACCEPT
[program:
  [decls:
    [class_decl_list:
      [class_decl:
        CLASS
        ID
        LBRACE
        [dbody:
          [dbody:]
          [fdecl:
            [typ: INT]
            [fname: ID]
            LPAREN
            [formals_opt:]
            RPAREN
            LBRACE
            [vdecl_list:]
            [stmt_list:]
            RBRACE
          ]
        ]
      ]
    ]
  ]
]
]
EOF
```

Pretty-Printer

Made sure input tokens = output tokens

```
Terminal
pt4115@pt4115: ~/Desktop/maze
pt4115@pt4115:~/Desktop/maze$ make
ocamlfind ocamlopt -c -package llvm str.cmxa ast.ml
ocamlfind ocamlopt -c -package llvm str.cmxa codegen.ml
ocaml yacc parser.mly
ocamlc -c ast.ml
ocamlc -c parser.ml
ocamlfind ocamlopt -c -package llvm str.cmxa parser.ml
ocamllex scanner.mll
107 states, 5588 transitions, table size 22994 bytes
ocamlfind ocamlopt -c -package llvm str.cmxa scanner.ml
ocamlfind ocamlopt -c -package llvm str.cmxa analyzer.ml
ocamlfind ocamlopt -c -package llvm str.cmxa maze.ml
ocamlfind ocamlopt -linkpkg -package llvm str.cmxa -package llvm.analysis ast.cmxa codegen.cmxa parser.cmxa
cmx scanner.cmx analyzer.cmx maze.cmx -o maze
pt4115@pt4115:~/Desktop/maze$ ./testall.sh
test-addition...OK
test-binop...OK
test-binop2...OK
test-binopmult...OK
test-bool...OK
test-classes...OK
test-equality...OK
test-fbinop...OK
test-func-call...OK
test-gcd...OK
test-hello...OK
test-if2...OK
test-ifEmpty...OK
test-ifNested...OK
test-ifestmt...OK
test-ops...OK
test-printfloat...OK
test-printid-bool...OK
test-printid-char...OK
test-printid-float...OK
test-printid-int...OK
test-printid-simple...OK
test-printid-string...OK
test-printint...OK
test-return...OK
test-subtraction...OK
test-unop...OK
test-whilestmt...OK
fail-binopOperands...OK
fail-defprint...OK
fail-dupClass...OK
fail-dupClassvar...OK
fail-dupFormal...OK
fail-dupFun...OK
fail-dupFun...OK
fail-dupvar...OK
fail-undeclaredID...OK
fail-voidClassvar...OK
fail-voidFormal...OK
fail-voidvar...OK
pt4115@pt4115:~/Desktop/maze$
```

```
Ubuntu Desktop
pt4115@pt4115: ~/Desktop/maze/tests
pt4115@pt4115:~/Desktop/maze/tests$ ls
fail-binopOperands.err fail-voidvar.err test-gcd.maze test-printid-char.maze
fail-binopOperands.maze fail-voidvar.maze test-gcd.out test-printid-char.out
fail-defprint.err test-addition.maze test-hello.maze test-printid-float.maze
fail-defprint.maze test-addition.out test-hello.out test-printid-float.out
fail-dupClass.err test-binop2.maze test-if2.maze test-printid-int.maze
fail-dupClass.maze test-binop2.out test-if2.out test-printid-int.out
fail-dupClassvar.err test-binopmult.maze test-ifEmpty.maze test-printid-simple.maze
fail-dupClassvar.maze test-binopmult.out test-ifEmpty.out test-printid-simple.out
fail-dupFormal.err test-binop.out test-ifNested.maze test-printid-string.maze
fail-dupFormal.maze test-binop.out test-ifNested.out test-printid-string.out
fail-dupFun.err test-binop.maze test-ifestmt.maze test-printint.maze
fail-dupFun.maze test-bool.out test-ifestmt.out test-printint.out
fail-dupvar.err test-classes.maze test-ops.out test-return.out
fail-dupvar.maze test-classes.out test-ops.out test-return.out
fail-undeclaredID.err test-equality.maze test-printchar.maze test-subtraction.maze
fail-undeclaredID.maze test-equality.out test-printchar.out test-subtraction.out
fail-voidClassvar.err test-fbinop.maze test-printfloat.maze test-unop.maze
fail-voidClassvar.maze test-fbinop.out test-printfloat.out test-unop.out
fail-voidFormal.err test-func-call.maze test-printid-bool.maze test-whilestmt.maze
fail-voidFormal.maze test-func-call.out test-printid-bool.out test-whilestmt.out
pt4115@pt4115:~/Desktop/maze/tests$ ls | wc -l
80
pt4115@pt4115:~/Desktop/maze/tests$
```

- Add feature → Add test
- Run ./testall.sh
- Ensure all tests pass

Tests that should pass:

```
pl4115@pl4115: ~/maze/tests
1 class test {
2     void main () {
3
4         int x;
5         x = 3 + 7;
6
7         if(x==10){
8             print("correct");
9         }
10    }
11 }
```

Tests that should fail (with appropriate error message)

```
pl4115@pl4115: ~/maze/tests
1 class test {
2     void main () {
3         int x;
4         int x;
5     }
6 }
```



```
pl4115@pl4115:~/maze$ ./maze -c tests/fail-dupvar.maze
Fatal error: exception Failure("duplicate variable x")
```


GCD

```
plt4115@plt4115: ~/maze/tests
1 class test {
2     void main () {
3         int x;
4         int y;
5         x = 15;
6         y = 20;
7
8         if (x == 0) {
9             print(x);
10        }
11
12        while(x != y)
13        {
14            if ( x > y) {
15                x= x-y;
16            } else {
17                y= y-x;
18            }
19        }
20        print(x);
21    }
22 }

"test-gcd.maze" 22L, 334C 1,1
```

Passing print an identifier:

```
plt4115@plt4115: ~/maze/tests
1 class test {
2
3     void main(){
4         string x;
5         x = "Hello";
6         print(x);
7     }
8
9 }
```

DEMO

Fibonacci is cool

plt4115@plt4115: ~/maze/tests

```
1 class test {
2
3     int fib(int n){
4
5         if(n <= 1){
6             return n;
7         }
8         return fib(n-1) + fib(n-2);
9     }
10
11     void main() {
12         int n;
13         int answer;
14
15         n = 9;
16         answer = fib(n);
17         print(answer);
18     }
19
20 }
```