MineTime

Mirza Ali - Project Manager
Tanay Jaipuria - Language and Tools Guru
Don Yu - System Architect
Patrice Liang - System Integrator
Stephen Zhou - System Tester

Minecraft 101

Sandbox game

- Fighting
- Survival
- Building

(Quick demo)

MineTime Elevator Pitch

"MineTime is a scripting language used to build Minecraft worlds in a fraction of the time"

Buzzwords

Time Saver

Simple

Elegant

Portable

Inspiration to code

Motivation

Save Minecrafters time
Introduce Minecrafters to the wonderful world of programming

With MineTime
time to write program ~ 25 min
time to run program ~ 5 min
Total time ~ 30 min

Without MineTime
Number of blocks ~ 3000
Time to place one block ~ 3.5 sec
Total time ~ 150 min

5X More Productive!!

Cardinal Rules of MineTime

1. Thou shalt use semicolons
2. Thou shalt use curly braces
3. Thou shalt remember that MineTime is dynamically typed
4. Thou shalt initialize a map using the Flatmap method
5. Thou shalt close the map at the end of the program

Hello World

```
map = new Flatmap("testmap.dat" , 100 ,  100, 100); $ create a
map
p = new Point(0,0,0); $ create a point
map.add(block(COBBLESTONE), p); $ place a cobblestone block at
a point
map.close(); $ close the map
```

Syntactic Constructs

For and while loops - standard
Functions - standard
If/Else statements - standard

Built-in:

Map - every program should create one
Blocks - over 100 types of blocks
Point - object that has x,y,z coordinates

A **block** is added at a certain **point** on the **map**

Example to Create an X

```
def even(x) {
    answer = false;
    div2 = x/2;
    times2 = div2*2;
    if (times2 == x) {
        answer = true;
    }
    else {
    answer = false;
    }
    return answer;
}
```

```
def main() {
    map = new Flatmap("xmap2", 300, 127,
100);
    max = 127;

    for (i=1; i < max; i = i+1) {
        $ The second y coord should be the
max coord minus the current index
        y2 = max - i;

CONTINUED...
```

Example continued

```
        ...
    if ( even(i) ) {
            p = new Point(i,i,0);
            map.add(block(COBBLESTONE), p);
            p2 = new Point(i, y2, 0);
            map.add(block(COBBLESTONE), p2);
        }
        else {
            p = new Point(i,i,0);
            map.add(block(BRICK), p);
            p2 = new Point(i, y2, 0);
            map.add(block(BRICK), p2);
        }
    }

    map.close();
}
```

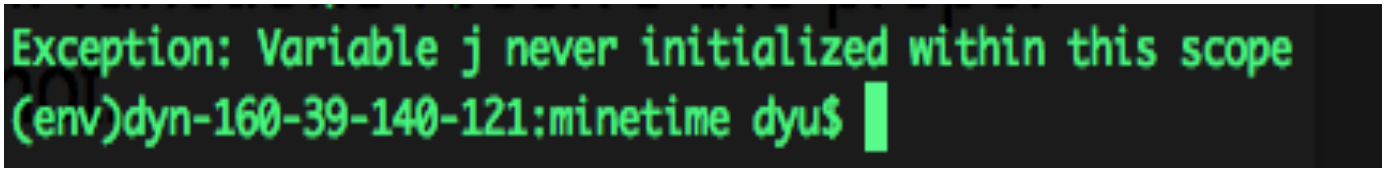
Scope Checking (Variables)

Variable exists in the code block it is defined in

i.e.

```
$ Makes an X
def main() {
    map = new Flatmap("xmap2", 300, 127, 100);
    max = 127;

    for (i=1; i < max; i = i+1) {
        y2 = max - i;
        if ( even(j) ) {
```



Scope Checking (Initializers & Functions)

We check for user-defined functions and built-in initializers

```
def main() {
    map = new Flatap("xmap2", 300, 127, 100);
    max = 127;
    for (i=1; i < max; i = i+1) {
        y2 = max - i;
        if ( ven(j) ) {
```

```
Exception: Initializer Flatap not defined in this language
```

```
Exception: Function ven is not user-defined nor is it part of the MineTime library
(env)dyn-160-39-140-121:minetime dyu$
```

Type Checking

We will check if functions receive the proper arguments or not.

```
i.e.
$ Makes an X
def main() {
    map = new Flatmap("xmap2", 127, 100);
    max = 127;
    ...
}
```

```
Exception: Initializer Type Check Error for Flatmap, excepted [<type 'str'>, <type 'int
'>, <type 'int'>, <type 'int'>] but got [<type 'str'>, <type 'int'>, <type 'int'>]
```

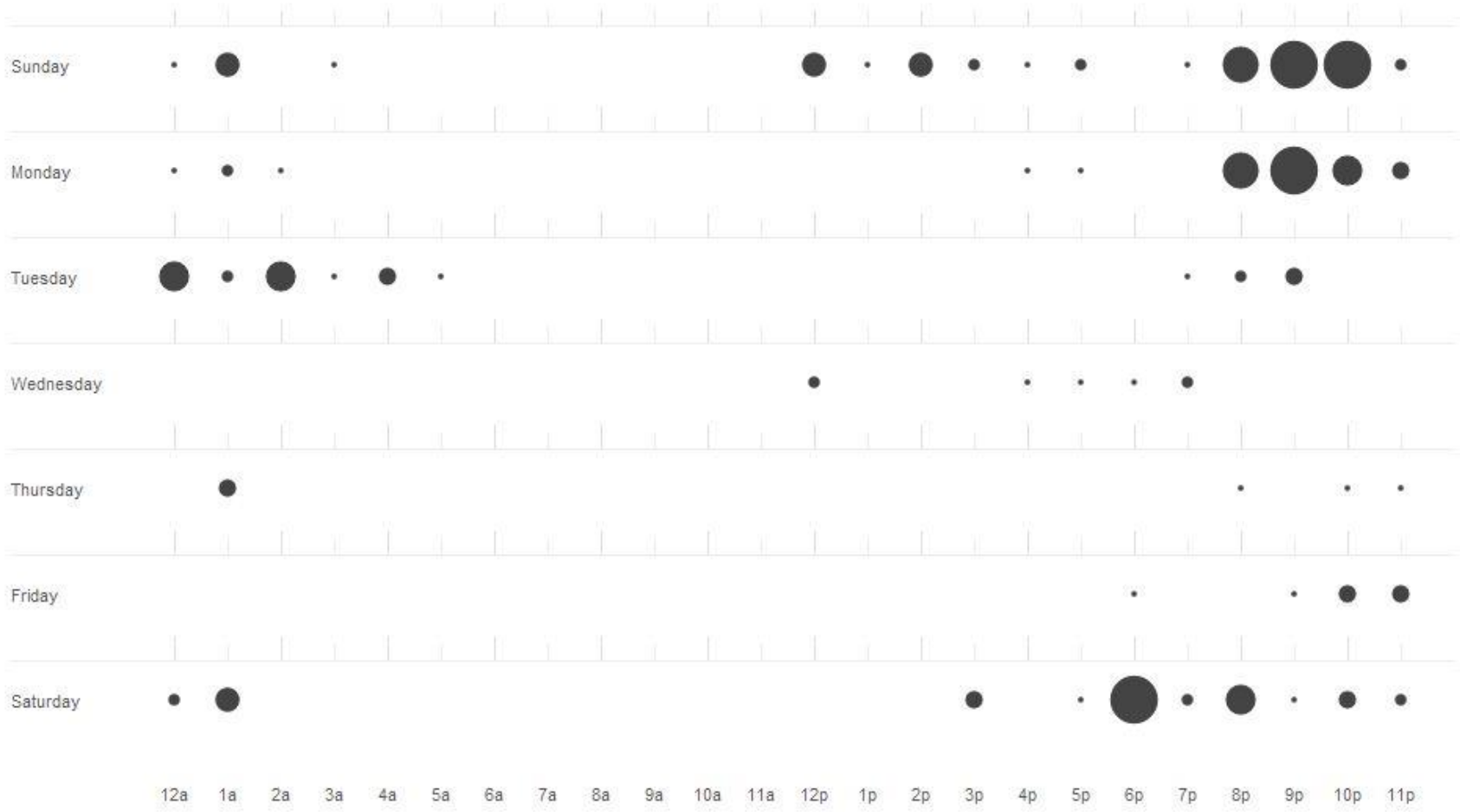
Project Management

- Weekly Meetings with the Team
- Task Management with Asana
- Scheduled meetings with our Advisor Melanie
- Use Google Docs and GitHub for check-ins

Project Management

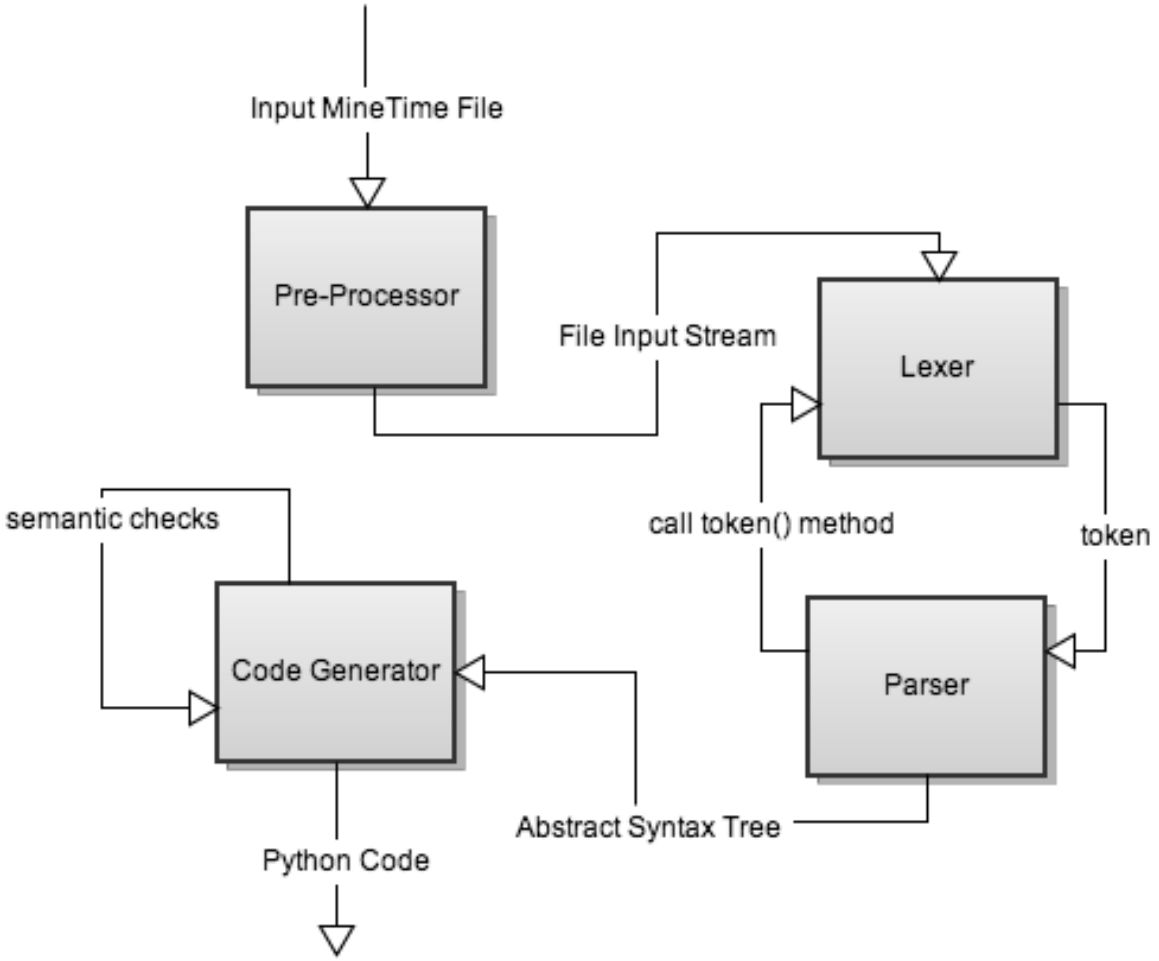
Git Commits over Time

Project Management

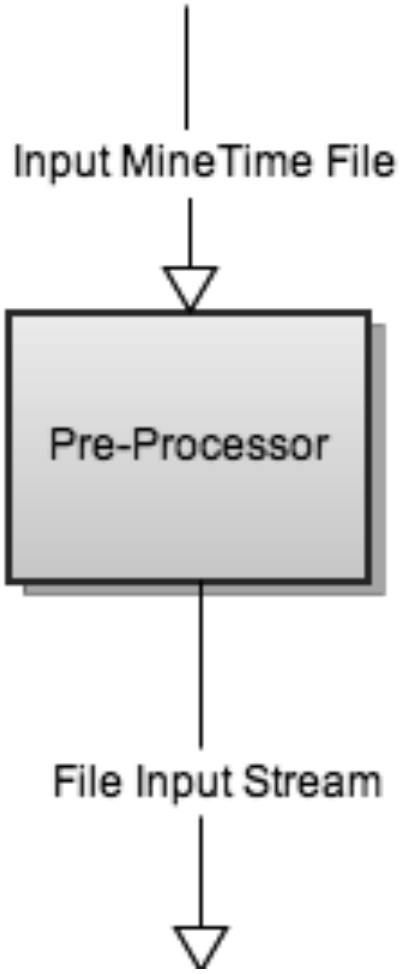


Architecture Overview

- Pre-Processor (preprocessor.py)
- Lexer (lexer.py)
- Parser (yaccing.py)
- Code Generator (traverse.py)
- Bringing Everything Together (minetime.py)



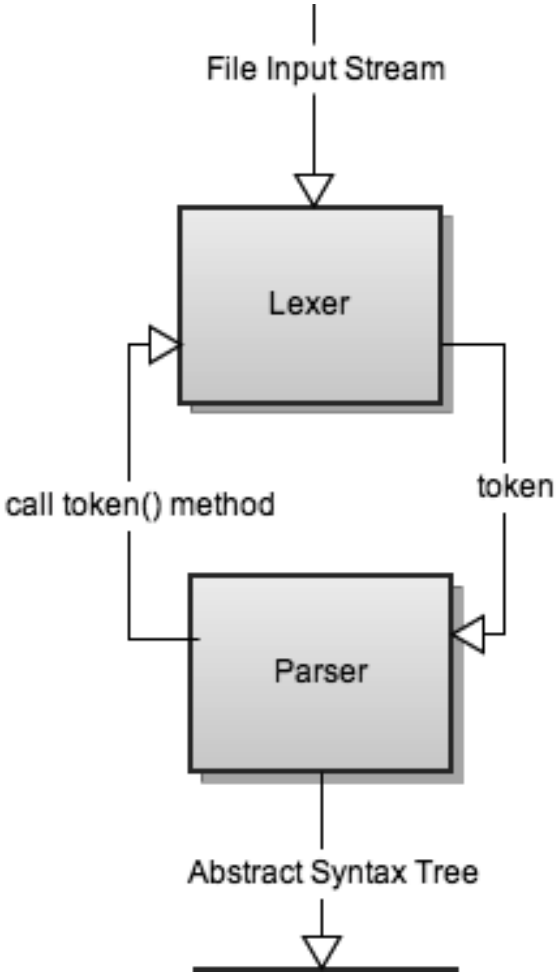
Architecture (Pre-Processor)




```
import tools.mt;
def main() {
map = new Flatmap("testmap.dat" , 100 ,  100, 100);
p = new Point(0,0,0);
map.add(block(COBBLESTONE), p);
map.close();
}

$$
$$ Source MineTime code from tools.mt placed here
$$
def main() {
map = new Flatmap("testmap.dat" , 100 ,  100, 100);
p = new Point(0,0,0);
map.add(block(COBBLESTONE), p);
map.close();
}
```

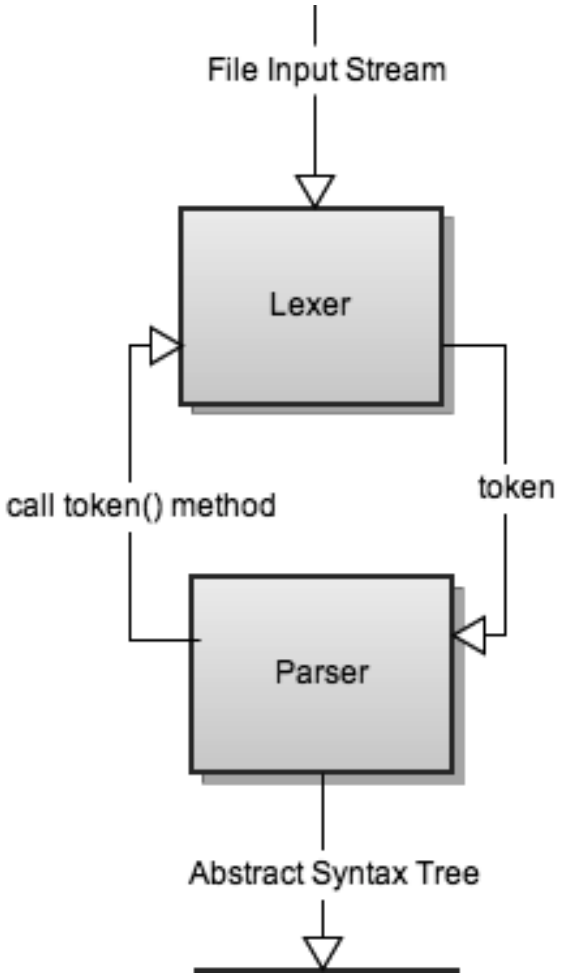
Architecture (Lexing & Parsing)




```
import tools.mt;
def main() {
map = new Flatmap("testmap.dat" , 100 ,  100,
100);
p = new Point(0,0,0);
map.add(block(COBBLESTONE), p);
map.close();
}
```

```
LexToken(DEF,'def',2,1),LexToken(ID,'main',2,5)
LexToken(LPAREN,'(',2,9),LexToken(RPAREN,')',
2,10),LexToken(LCURL,'{',2,12),LexToken
(ID,'map',3,14),LexToken(ASSIGN,'=',3,18),
LexToken(ID,'Flatmap',3,20),LexToken
(LPAREN,'(',3,27),LexToken(STRING,'"testmap.
dat"',3,28),LexToken(COMMA,',',3,41), .....
```

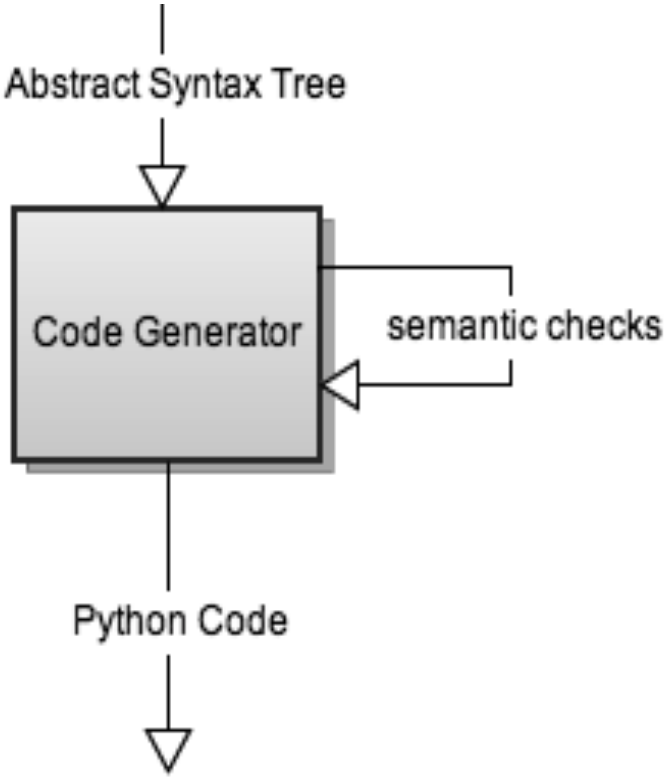
Architecture (Lexing & Parsing)




```
LexToken(DEF,'def',2,1),LexToken(ID,'main',2,5)
LexToken(LPAREN,'(',2,9),LexToken(RPAREN,')',
2,10),LexToken(LCURL,'{',2,12),LexToken
(ID,'map',3,14),LexToken(ASSIGN,'=',3,18),
LexToken(ID,'Flatmap',3,20),LexToken
(LPAREN,'(',3,27),LexToken(STRING,'"testmap.
dat"',3,28),LexToken(COMMA,',',3,41), .....

translation_unit
 |external_declaration
 | |function_definition
 | | |main
 | | |parameter_list
 | | |statement_list
 | | | |statement_list
 | | | | |statement_list
......
```

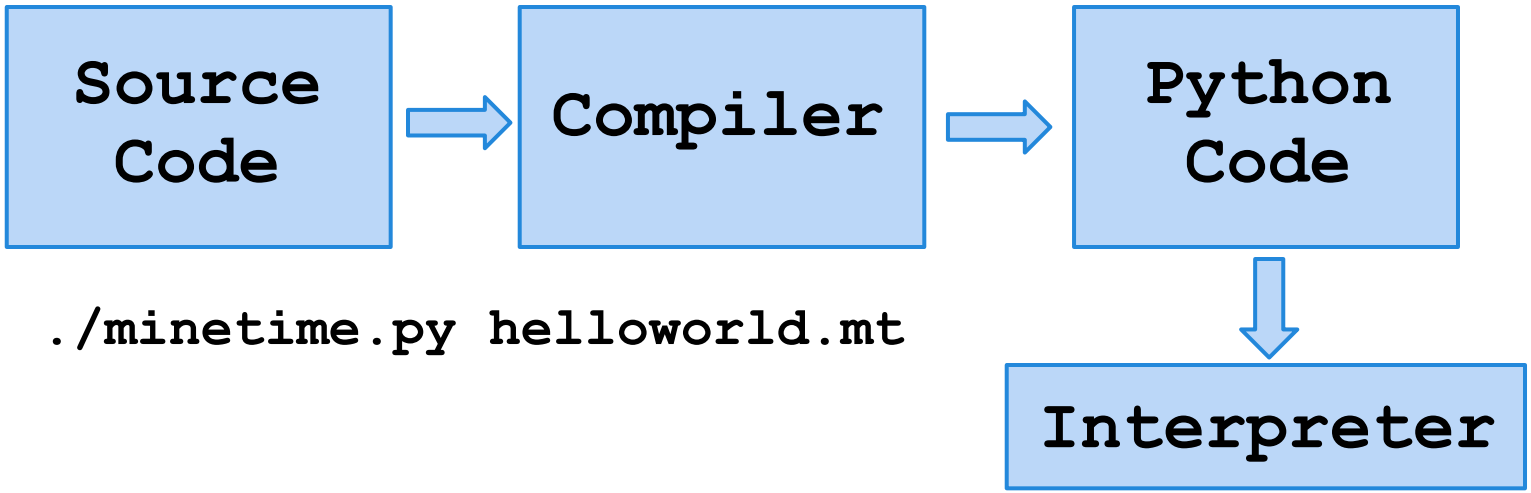
Architecture (Code Generation)




```
translation_unit
 |external_declaration
 | |function_definition
 | | |main
 | | |parameter_list
 | | |statement_list
 | | | |statement_list
 | | | | |statement_list
```

```
import logging
import os
import sys
from pymclevel import mclevel
from pymclevel.box import BoundingBox
def main():
    map=mclevel.MCInfdevOldLevel("testmap.at",
create=True)
    map.createChunksInBox(BoundingBox((-50,0,
-50),(100,100,100)))
    p=(0,0,0)
    map.fillBlocks(BoundingBox(origin=p,size=
(1,1,1)),map.materials.blockWithID(4))
    pass
```

Runtime Environment



- Virtual environment: pymclevel, PyYaml, numpy, PLY
- minetime.py
  - imports preprocess.py, lexing.py, yaccing.py, traverse.py
  - input: .mt file; output: .py file

Development Environment

- PLY (Python Lex-Yacc)
- pymclevel
- Python
- Git
- Vim (or Sublime Text)

Compiler Tools

- PLY (Python Lex-Yacc)

  Parse file with 1000 random expressions (805KB) and build an abstract syntax tree

  - PLY-2.3 : 2.95 sec, 10.2 MB (Python)
  - YAPPS2 : 6.57 sec, 32.5 MB (Python)
  - PyParsing : 13.11 sec, 15.6 MB (Python)

- pymclevel (Python library to read Minecraft worlds)

Testing

- Python unittest module
- Approach

  Shared test cases

  Test suites for each part of the compiler

  Assertions and prints

Test #1: Hello World

```
helloworld = """
def main() {
    x = new flatmap
("testfilesgitestmap",
500,500,500);
    b = new Point(10,20,30);
    x.add(block(STONE), b);
    x.close();
}"""
```

Tests Part 2: Grammar Exs

```
while_loop = """
def main() {
    while (i=1)
{i=1;i=1;i=1;}
    i = 0;
}"""


empty_function = """
def main(1,2) {
}"""
```

```
if_elseif_else = """
def main() {
    i = 2;
    if (i == 1)
       i = 2;
    else if (i==2) {
       i = 3;
    } else
       i = 4;
}"""
```

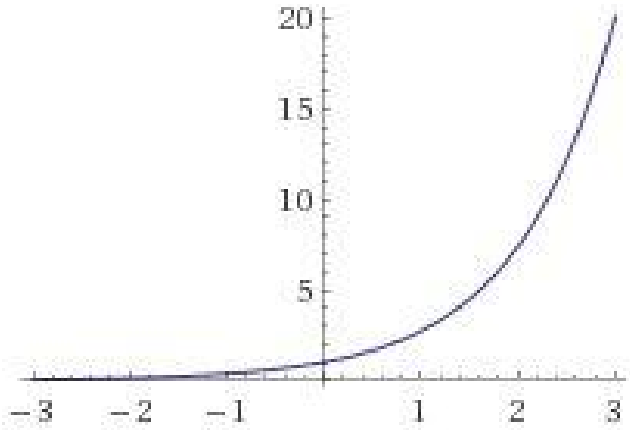
Tests Part 3: Semantics

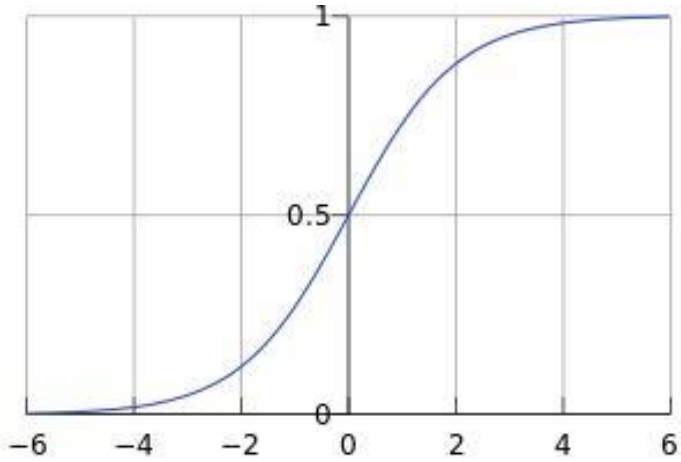
```
make_blocks = '''
def makeblocks(start, end, x) {
    while (start < end) {
        c = new Point(0,0,start);
        x.add(block(COBBLESTONE), c);
        start = start + 1;
    }
    for (;start<end;start=start+1)
    {
        c = new Point(0,0,start);
        x.add(block(COBBLESTONE), c);
    }
}

def main() {
x = new Flatmap("testfiles/testmap",500,500,500);
makeblocks(0,"hi", x);
x.close();
}
'''
```

Conclusions

How much time does a MineTime mine if a MineTime could mine time?





But first, demo time (and comparison of times)

Lessons Learned

"Don't take a shortcut or shrug off a poorly-written block of code in an attempt at speed. They will come back to haunt you." - Don

"GitHub is bigger and better than life. Version control is a life-saver."
- Stephen

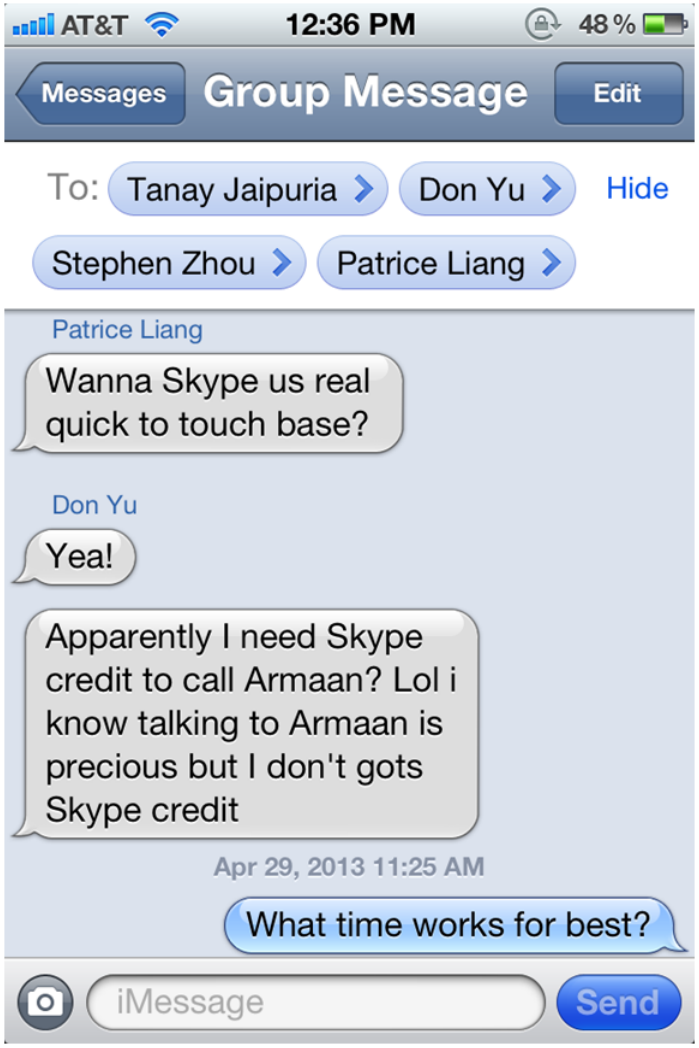
"Meetings, texts, and Asana reminders are all ways to keep momentum going, but nothing beats the camaraderie that comes with having 5 people invested in the same goal." - Patrice

"No matter how early you start, you will be working on your project right up to the deadline."
- Tanay

"Without requirements or design, programming is the art of adding bugs to an empty text file." - Louis Srygley (Armaan)

What Worked Well

- Version control
- Group texting!
- Asynchronous tasks with few dependencies
- Consistent team meetings



What Could We Have Done Differently?

- Determine what the programmer really needs to build Minecraft structures rather than just add everything in the beginning
- Implement type and scope checking from the start rather than add it in the end

Use MineTime!!

Questions?