Interactive Fiction Language

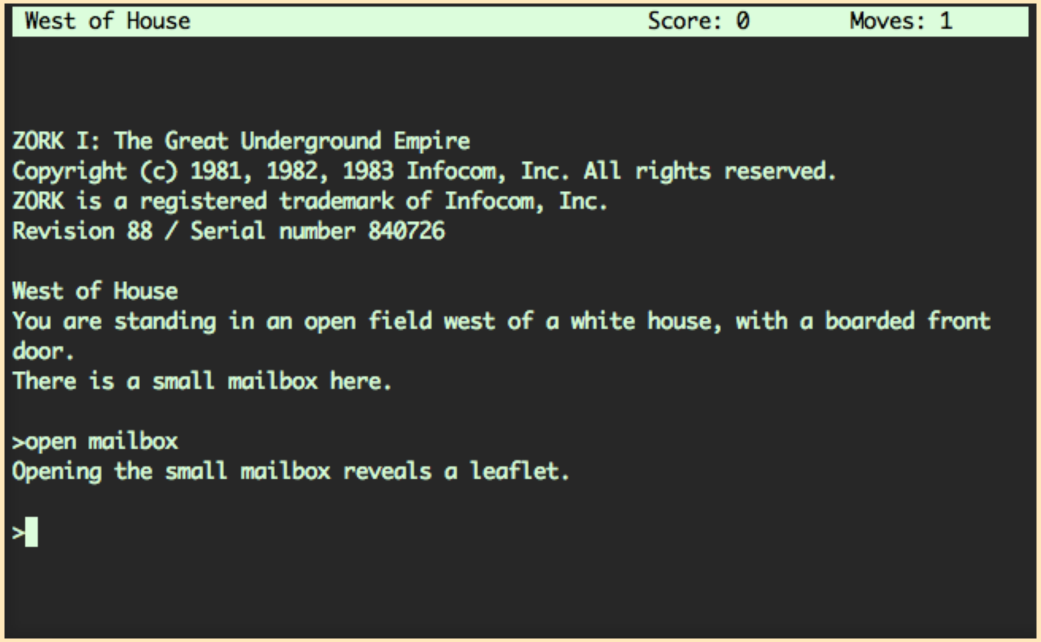
(IFL)

Team Introduction

Project Manager: John Liu
Language Guru: Matthew Suozzo
System Architect: Michael Yan
System Integrator: Qian Yu
System Tester: Heather Fisher

Interactive Fiction = ?

- Text Adventure
- Story-Driven
- Interactive



```
West of House                                    Score: 0        Moves: 1



ZORK I: The Great Underground Empire
Copyright (c) 1981, 1982, 1983 Infocom, Inc. All rights reserved.
ZORK is a registered trademark of Infocom, Inc.
Revision 88 / Serial number 840726

West of House
You are standing in an open field west of a white house, with a boarded front
door.
There is a small mailbox here.

>open mailbox
Opening the small mailbox reveals a leaflet.

>
```

Goals of our IF Language

- Easily Create Interactive Fiction
- Minimal Programming Experience Needed

Our User

- Familiar with programming concepts

- Mostly a **Game Designer**
  - (instead of coder)

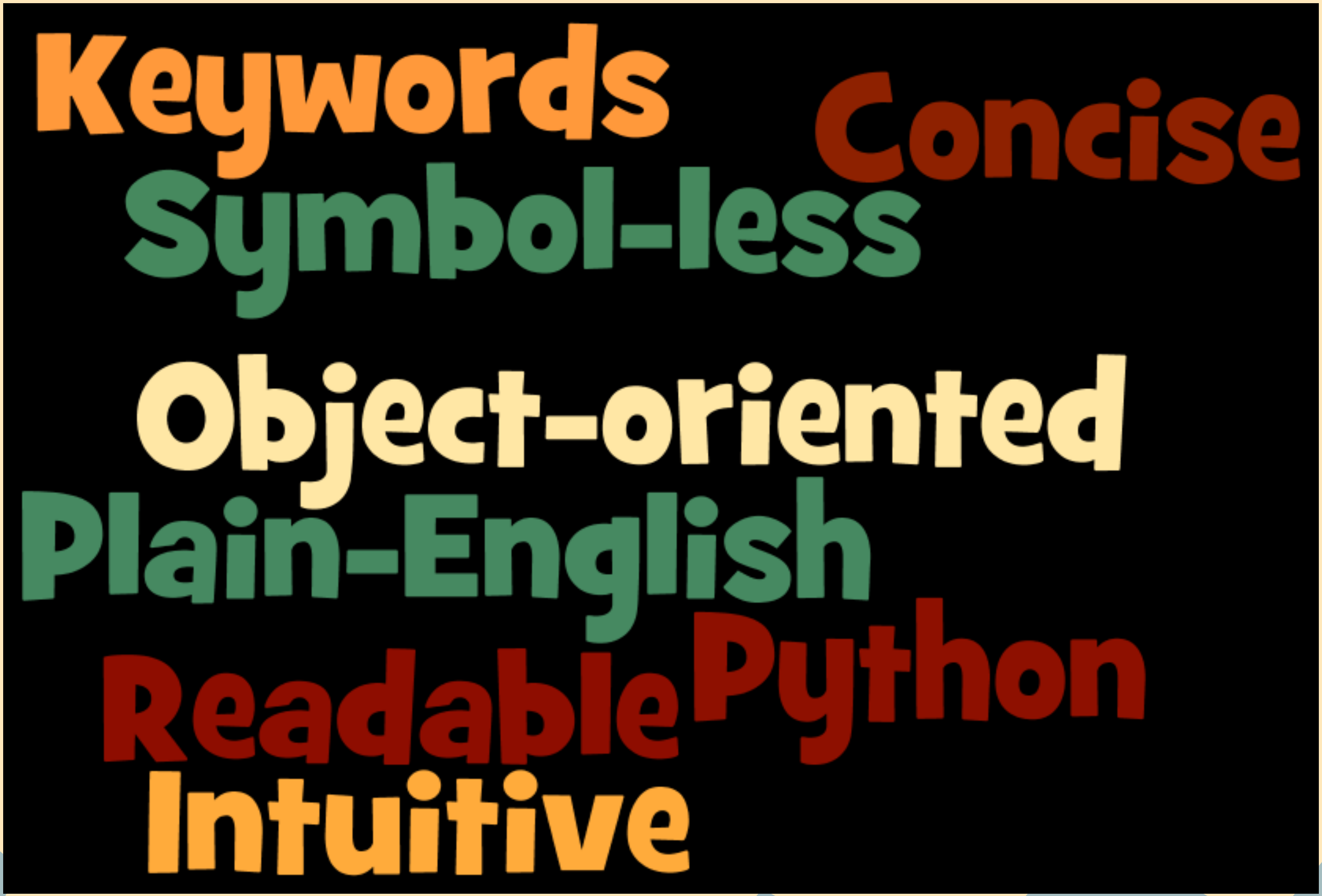
Goals of our IF Language

- Easily Create Interactive Fiction
- Minimal Programming Experience Needed
- Decouple Roles of Writer & Programmer
  - eg. Dialogue Trees

Language Design

Matthew Suozzo

Buzzwords


Keywords
Concise
Symbol-less
Object-oriented
Plain-English
Readable
Python
Intuitive

What we Got

4 Object Types (TLTs)

Item	**Trait**
Character	**Setting**

What we Got

4 Primitive Types

String

TF (bool)

Integer

Decimal

Language Structure

- Program is list of defined types (TLTs)

```
ITEM Apple:
   {block}:
      {statements}
```

`STARTS`	: Constructor
`ACTIONS`	: Actions available in game
`FUNCTIONS`	: Functions available in the code
`DIALOGUE`	: Defines Dialogue

- `PLAYER` Character is like `main` method

Syntax / Statements

```
ADD item TO char
REMOVE item FROM char
SET val TO 3
PRINT str . "str"
MOVE char TO loc
EXECUTE func WITH arg1
INITIATE DIALOGUE AT #LABEL#
INCREASE val BY 3
DECREASE val BY 1
NUMBER OF item IN char
GOTO #LABEL#
USING "diag.txt"
EXIT
```

```
IF case1:
   {statements}
ELSE IF case2:
   {statements}
ELSE:
   {statements}
```

```
ADD {STRING s="hello"}
```

Sample Code Snippet

```
CHARACTER maid:
     “I am a maid of the house.”
     START:
          USES “Text.txt”
          ADD apple TO SELF
     ACTIONS:
          “talk”
               IF SELF HAS apple:
                    INITIATE DIALOGUE AT #LABEL A#
               ELSE:
                    INITIATE DIALOGUE AT #LABEL D#
     DIALOGUE:
          #LABEL A#:
               IF LAST_INPUT EQUALS “1”:
                    GOTO #LABEL B#
               ELSE IF LAST_INPUT EQUALS “2”:
                    GOTO #LABEL C#
          #LABEL B#:
               IF LAST_INPUT EQUALS “1”:
                    REMOVE apple
                    ADD apple TO PLAYER
                    GOTO #LABEL C
               ELSE IF LAST_INPUT EQUALS “2”:
                    EXIT
```

Sample Code Snippet

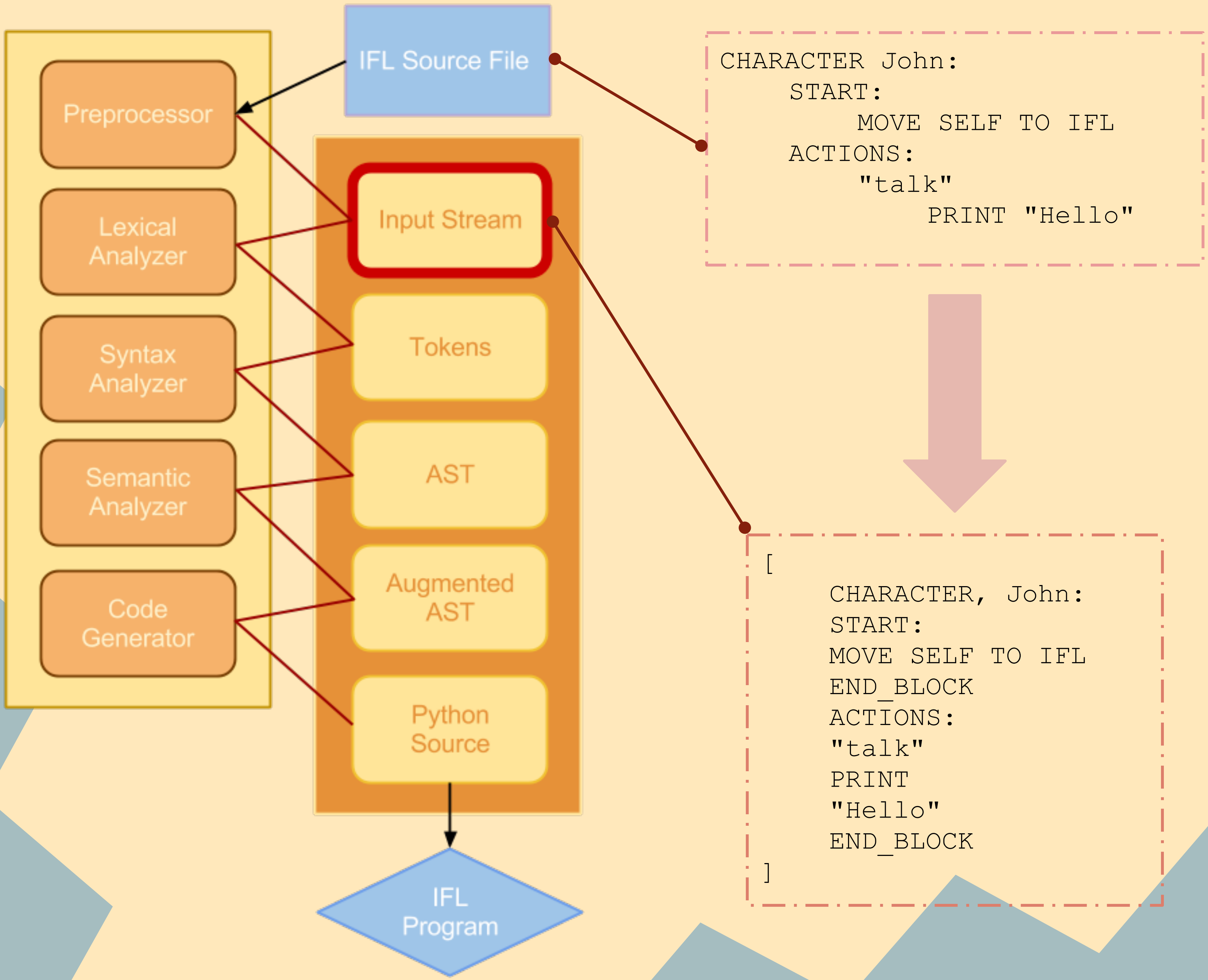
```
#LABEL A#
Maid: Hi, can I help you?
(1) Ask where you are
(2) Ask for an item

#LABEL B#
Maid: You are in the bedroom of a old house.
(1) Ask for an item
(2) Exit

#LABEL C#
The maid gives you an apple
(1) Ask where you are
(2) Exit

#LABEL D#
Maid: Hi, can I help you?
(1) Ask where you are

#LABEL E#
Maid: You are in the bedroom of a old house.
(1) Exit
```

DEMO!

Michael Yan

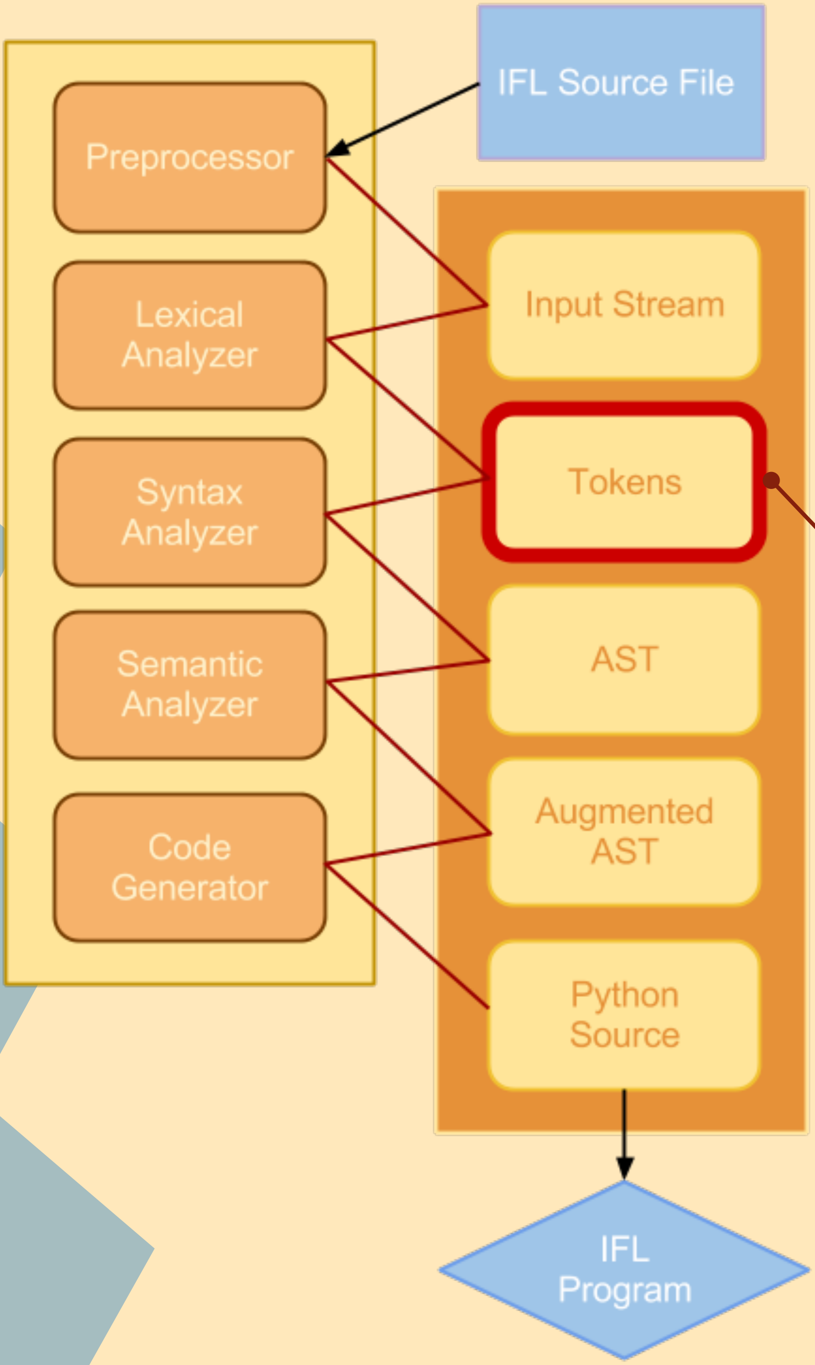
System Architecture

Michael Yan



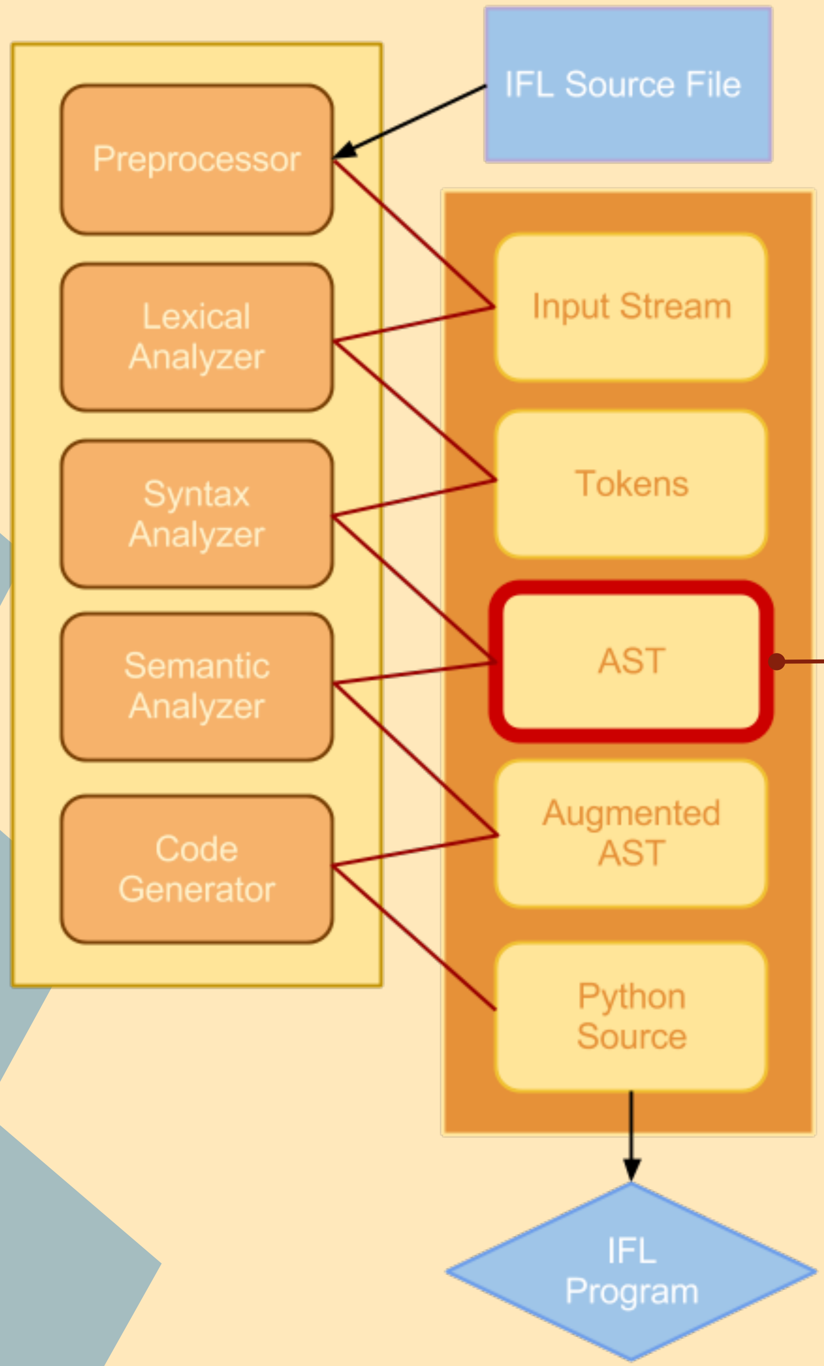
```
CHARACTER John:
    START:
        MOVE SELF TO IFL
    ACTIONS:
        "talk"
            PRINT "Hello"
```

```
[
    CHARACTER, John:
    START:
    MOVE SELF TO IFL
    END_BLOCK
    ACTIONS:
    "talk"
    PRINT
    "Hello"
    END_BLOCK
]
```



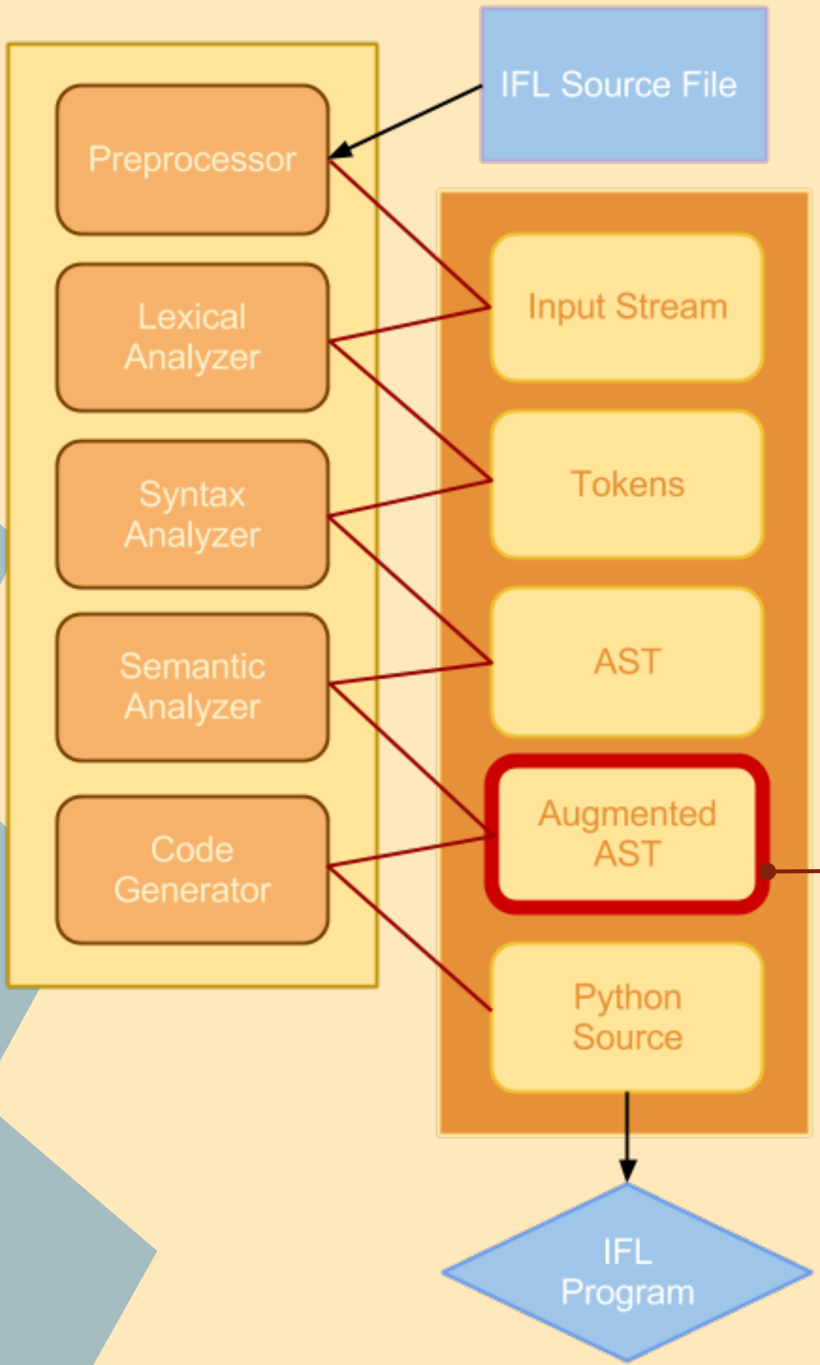
```
[
	CHARACTER, John:
	START:
	MOVE SELF TO IFL
	END_BLOCK
	ACTIONS:
	"talk"
	PRINT
	"Hello"
	END_BLOCK
]
```

```
LexToken(CHARACTER,'CHARACTER',1,0)
LexToken(ID,'John',1,10)
LexToken(COLON,':',1,14)
LexToken(START,'START',2,16)
LexToken(COLON,':',2,21)
LexToken(MOVE,'MOVE',3,23)
LexToken(ID,'SELF',3,28)
LexToken(TO,'TO',3,33)
LexToken(ID,'IFL',3,36)
LexToken(END_BLOCK,'END_BLOCK',4,40)
```



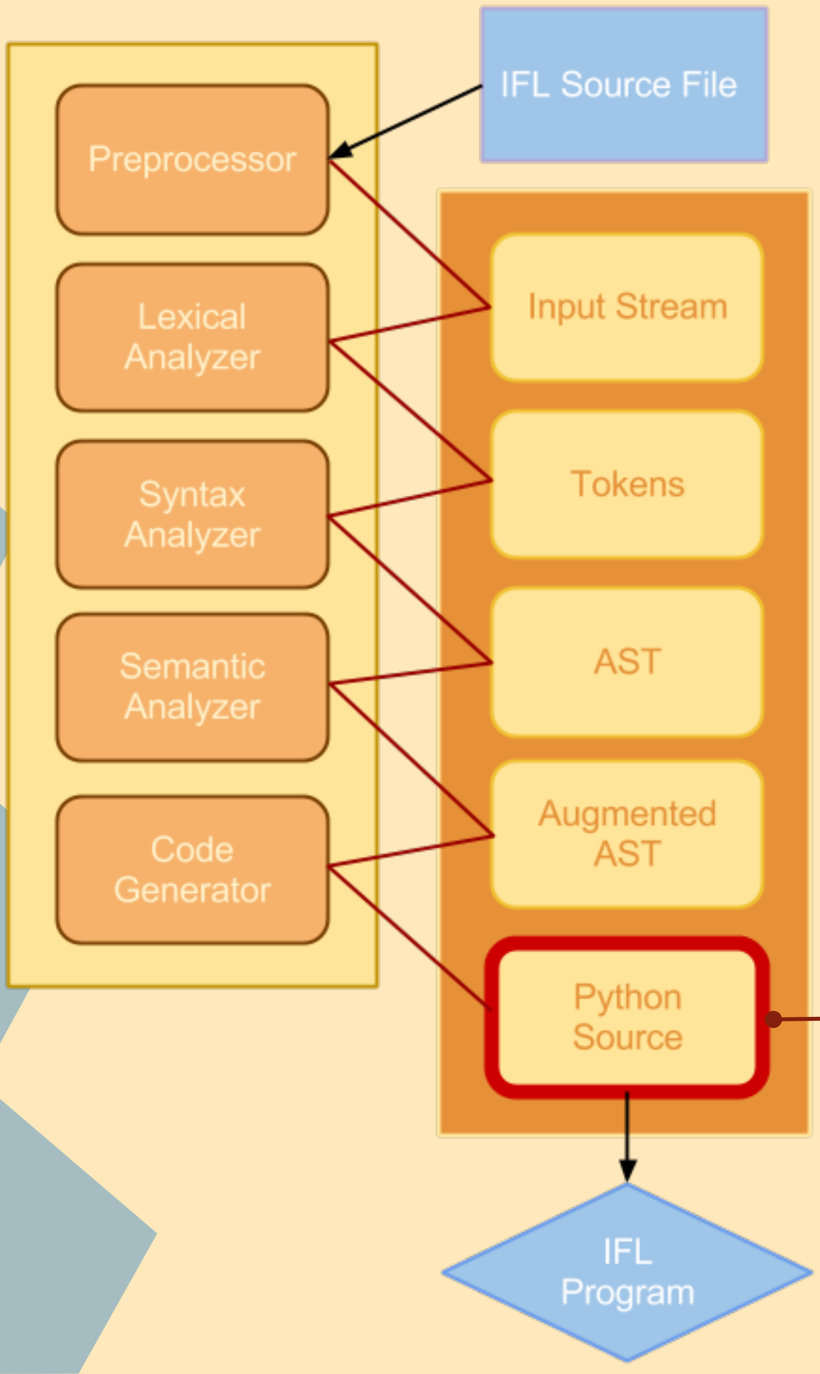
```
def p_move(p):
     'move : MOVE object_chain TO
      object_chain'

p[0] = (p[1], p[2], p[4])
```

```
('John',
  None,
     ('START',
          ('MOVE',
                   ('OBJ', 'SELF'),
                   ('OBJ','IFL')
          ),
     ('ACTIONS',
          ('talk',
               ('PRINT',
                    ('Hello')
               )
          ),
  None,
  None
)
```



```
('John',
  None,
     ('START',
          ('MOVE',
                    ('OBJ', 'SELF'),
                    ('OBJ','IFL')
          ),
     ('ACTIONS',
          ('talk',
               ('PRINT',
                    ('Hello')
               )
          ),
  None,
  None
)
```

```
Program
     TLT (John)
          start
               Statement
                    (MOVE)
                         ('OBJ', SELF)
                         ('LOC', IFL)
          actions
               Action (talk)
                    Statement
                         PRINT ('Hello')
```



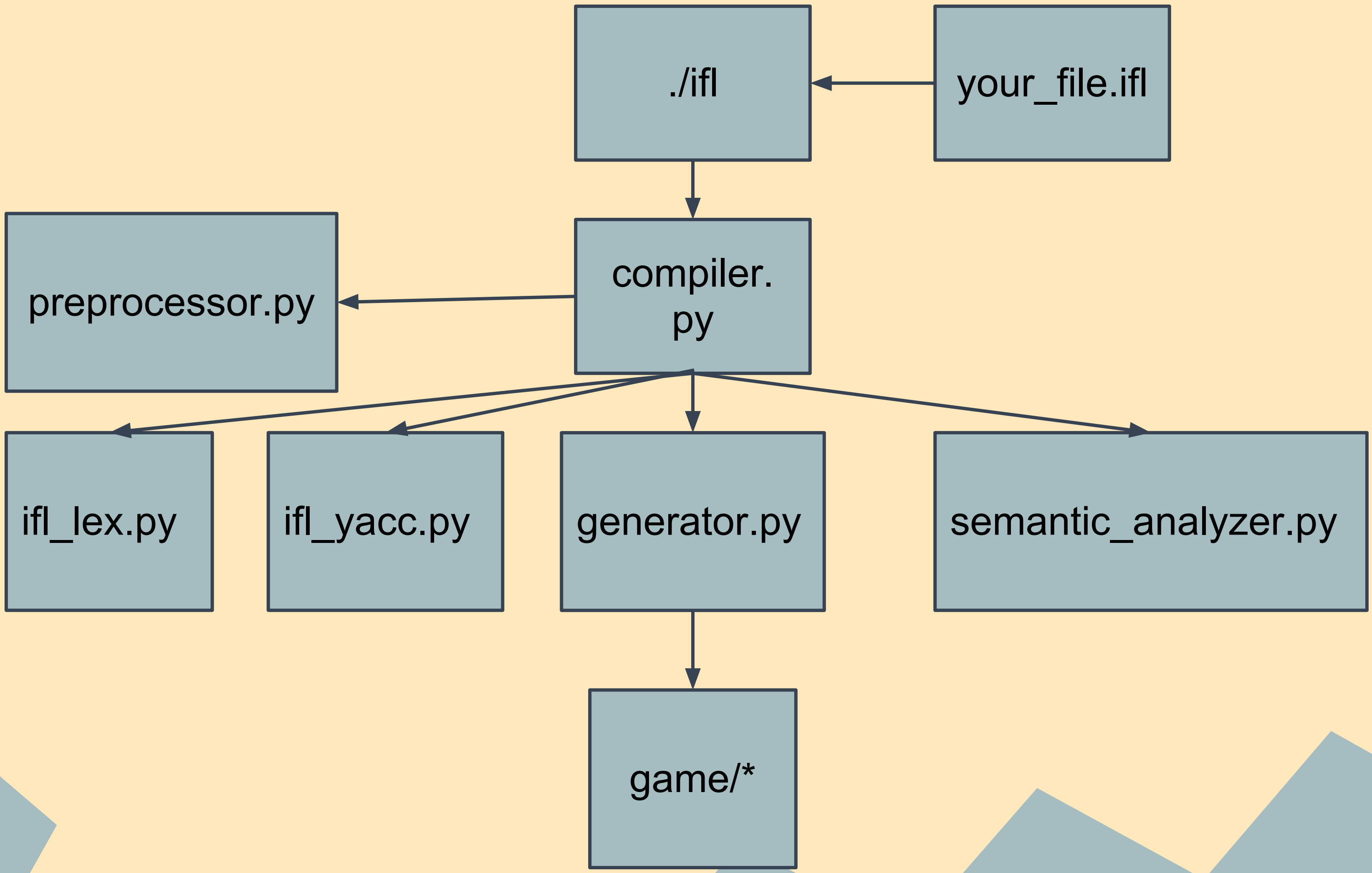
```
CHARACTER John:
     START:
          MOVE SELF TO IFL
     ACTIONS:
          "talk"
               PRINT "Hello"
```

```
class John:
     def __init__(self):
          self.location = 'IFL'
     def talk():
          print 'Hello'
```

System Integration

Qian Yu

How The Pieces Fit



What's Actually Generated?

- a game/ directory
- a .py class file for each object type (Item, Character, Trait, Setting)
- a main game file (game.py) that actually runs the game

Execution Environment

- game.py
- instantiate and setup characters and settings
- Enter into read-evaluate-print-loop (REPL)
    - Get User Command
    - Search for Command in Action List of Current Setting
    - Call Appropriate Functions
    - Print Results
    - Update Availables Actions
    - Repeat

Development Environment

- Python Lex/Yacc (PLY)
- Git and Github for Version Control
- IDE/Debugger: PyCharm
- Terminal for Running and Testing

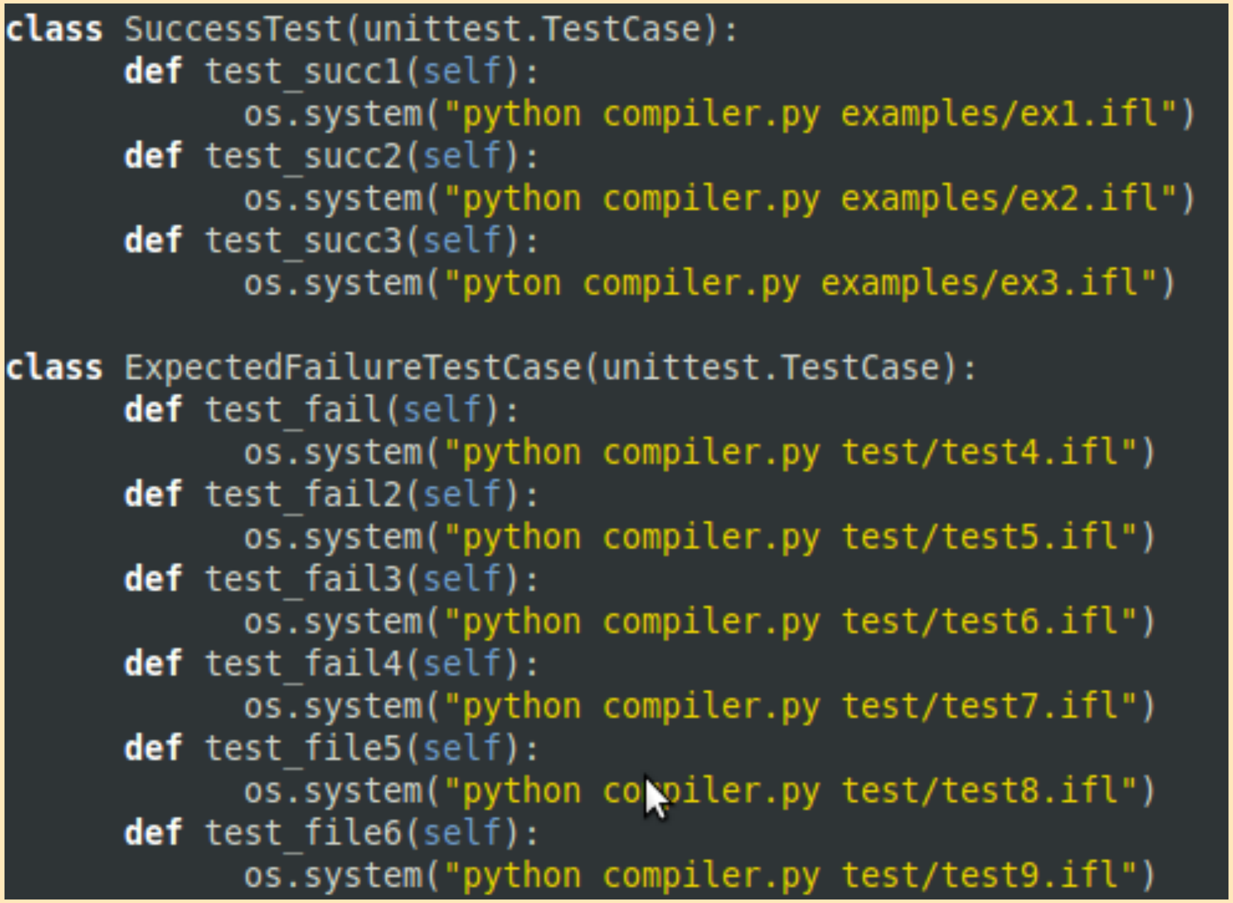
Testing

Heather Fisher

Testing Process

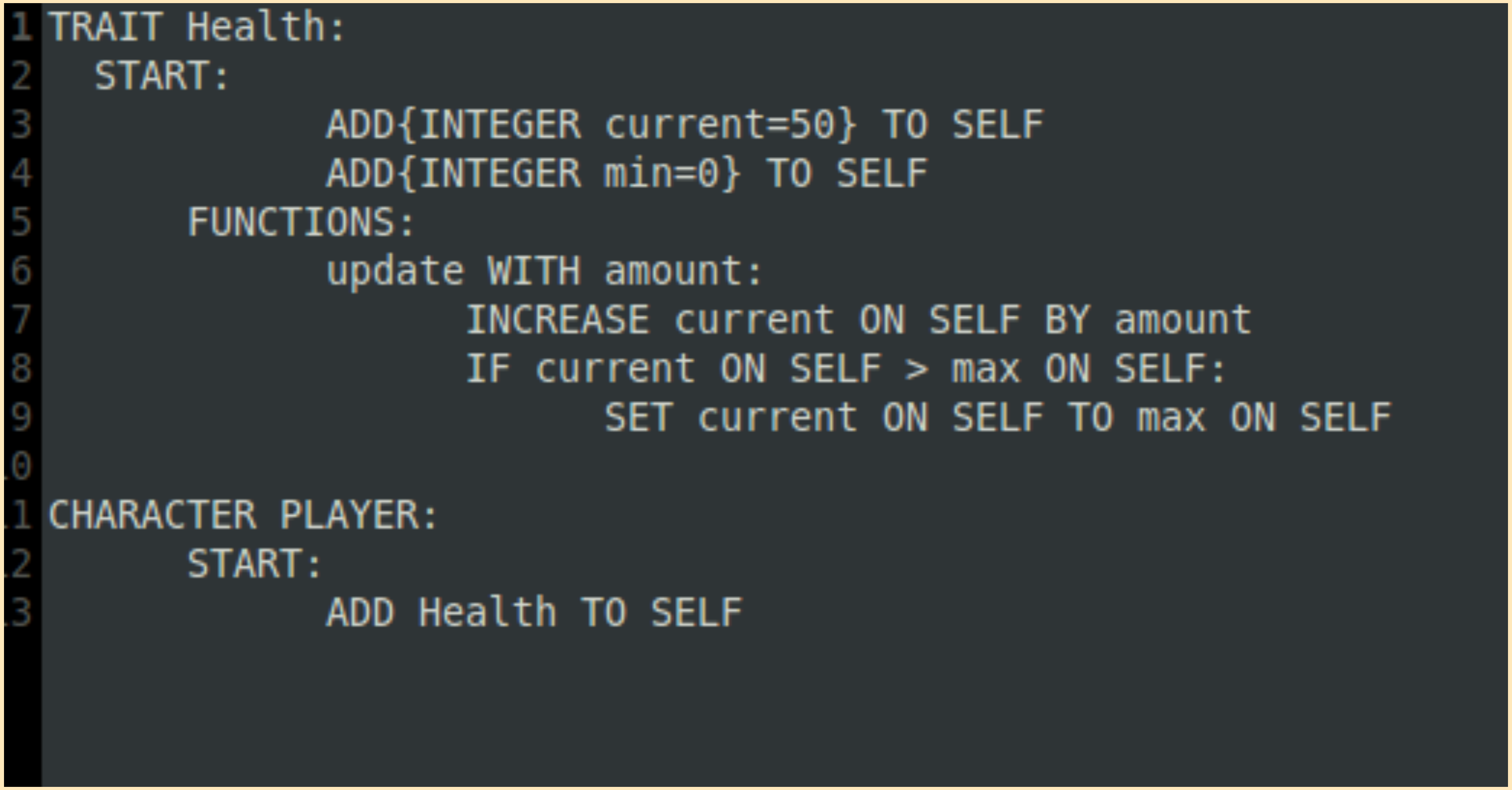
- Unit Test
- Test as we go

Test Suite



```
class SuccessTest(unittest.TestCase):
	def test_succ1(self):
		os.system("python compiler.py examples/ex1.ifl")
	def test_succ2(self):
		os.system("python compiler.py examples/ex2.ifl")
	def test_succ3(self):
		os.system("pyton compiler.py examples/ex3.ifl")

class ExpectedFailureTestCase(unittest.TestCase):
	def test_fail(self):
		os.system("python compiler.py test/test4.ifl")
	def test_fail2(self):
		os.system("python compiler.py test/test5.ifl")
	def test_fail3(self):
		os.system("python compiler.py test/test6.ifl")
	def test_fail4(self):
		os.system("python compiler.py test/test7.ifl")
	def test_file5(self):
		os.system("python compiler.py test/test8.ifl")
	def test_file6(self):
		os.system("python compiler.py test/test9.ifl")
```

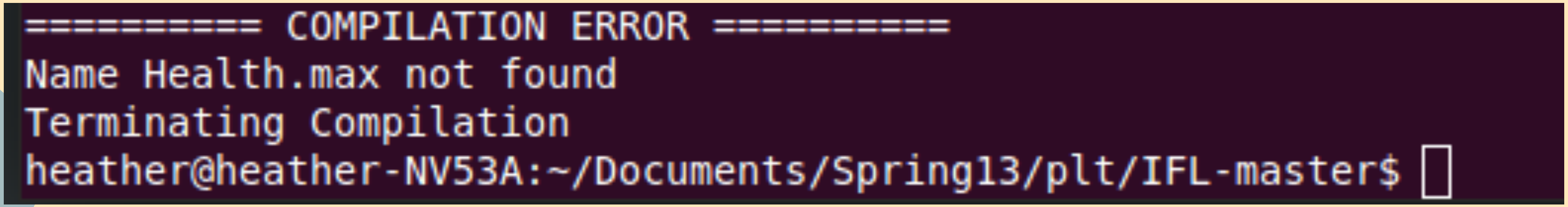
Sample Test Program



```
TRAIT Health:
  START:
            ADD{INTEGER current=50} TO SELF
            ADD{INTEGER min=0} TO SELF
      FUNCTIONS:
            update WITH amount:
                  INCREASE current ON SELF BY amount
                  IF current ON SELF > max ON SELF:
                        SET current ON SELF TO max ON SELF

CHARACTER PLAYER:
      START:
            ADD Health TO SELF
```

- Will this program work?

Results



```
=========== COMPILATION ERROR ===========
Name Health.max not found
Terminating Compilation
heather@heather-NV53A:~/Documents/Spring13/plt/IFL-master$ 
```

- Missing

  ADD{INTEGER max = 100} TO SELF

Why Testing is Important

- Test early and test often!
- You never know what will break your code

Project Management

John Liu

Project Process

- Waterfall Methodology to Agile Development
- Python & Java to Python Only
- Team Organization
    - Weekly Meetings, Google Hangout, Instant Messaging
    - Google Docs, Github

Lessons Learned

- Taking Advantage of Version Control (Git)
- Group Development vs Working Individually
- Communicate, Communicate, Communicate!

Possible Expansion

- Expand Character Encoding
- Support for Libraries
- Dialog Tree markup
- More Customization Options
    - Support for Multiplayer / Networking

The End

IFL: Do you have any questions?

```
(1) Yes
(2) No
>> _
```