Look familiar?

svelTest: A testing language

Kaitlin Huben (Project Manager)
Emily Hsia (Language Guru)
Josh Lieberman (System Architect)
Chris So (System Integrator)
Mandy Swinton (Verifier and Validator)

What is svelTest?

Object-oriented language with a simple approach to tackle testing code

[1]

Why svelTest?

- Testing is important
- Other libraries are **verbose** and convoluted
  - Java: JUnit, Mauve
  - C/C++: CUnit, CppUnit
  - Python: unittest (formerly PyUnit)
- svelTest is **concise** and flexible
  - Can test Java, C*, and Python**

*Almost done
**In progress

Speaking of buzzwords…

Modular

Flexible

Simple

Portable

Robust

Turing-complete

Speaking of buzzwords…

svelte |svelt, sfelt|
(adjective): slender and elegant.

sveltest |svelt·est, sfelt·est|
(adjective): the most svelte.

JUnit vs svelTest

Code snippet:

```
import static org.junit.Assert.assertEquals;
import org.junit.Test;
import org.junit.Ignore; ... [more imports]
@RunWith(JUnit4.class)
public class FooTest {
   @Test
   public void thisAlwaysPasses() {...}
   @Test
   @Ignore
   public void thisIsIgnored() {...}
}
```

Long

Call:

```
$ java -cp .:/usr/share/java/junit.jar
  org.junit.runner.JUnitCore FooTest
```

Code snippet:

```
main() {
   file trueFile = “AlwaysTrue.java”;
   funct trueFunct = {__main__, (j_String[]), trueFile};
   if(trueFunct.assert((), “true”))
      print(“I’m true!”);
}
```

Short

Call:

```
$ ./compile.sh trueTest.svel
$ python trueTest.py
```

Syntactic constructs

The usual suspects

int, double, boolean,
string, arrays

if, while, for

*Python reserved words are reserved

svelTest

file

funct

input

output

funct.assert()

svelTest idiom

1. Declare the `file` containing the method you want to test

   ```
   file testFile = “path/to/filename-to-test.java”;
   ```

2. Declare the method you want to test

   ```
   funct testFunct = {“name”, (paramtypes), testFile};
   ```

3. Set up `input`(s) and corresponding `output`(s)

4. Call `assert` with combinations of `input`s and `output`s

   ```
   funct.assert(input, output);
   ```

Or, do it all at once!

HelloWorld.java

```
public class HelloWorld {
  public static void main(String[] args) {
    String name = args[0];
    System.out.println(“Hello ” + name);
  }
}
```

helloWorldTest.svel

```
lang = Java;
main() {
    file f = "HelloWorld.java";
    funct hello = {__main__, (j_String[]), f};
    input name = “world”;
    output out = “Hello world”;
    if(hello.assert(name, out)) { print("Test passed."); }
    else { print("Test failed."); }
}
```

Side-by-side

helloWorldTest.svel

```
lang = Java;
main() {
    file f = "HelloWorld.java";
    funct hello = {__main__, (j_String[]), f};
    input name = “world”;
    output out = “Hello world”;
    if(hello.assert(name, out)) {
        print("Test passed.");
    }
    else { print("Test failed."); }
}
```

HelloWorld.java

```
public class HelloWorld {
  public static void main(String[] args)
  {
      String name = args[0];
      System.out.println(“Hello ” + name);
  }
}
```

helloWorldTest.svel

```
lang = Java;
main() {
    file f = "HelloWorld.java";
    funct hello = {__main__, (j_String[]), f};
    input name = “world”;
    output out = “Hello world”;
    if(hello.assert(name, out)) { print("Test passed."); }
    else { print("Test failed."); }
}
```

Fibonacci.java

```java
public class Fibonacci {
  public static int fib(int n){
    if (n == 0){  return 0;  }
    else if (n == 1){  return 1;  }
    else {  return fib(n-1) + fib(n-2);  }
  }
}
```

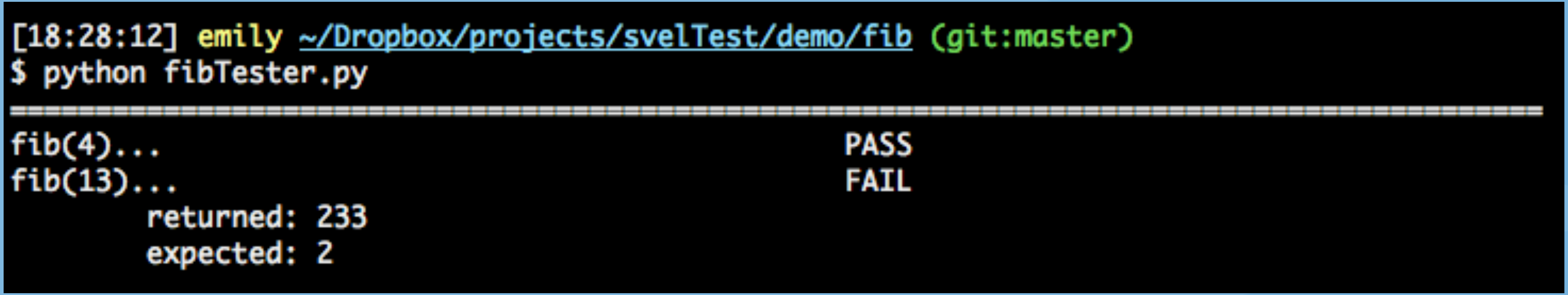
fibTester.svel

```
lang = Java;
main() {
  funct fibFunct = {“fib”, (j_int), “Fibonacci.java”};
  fibFunct.assert(4, 3, verbose);
}
```

Recall:

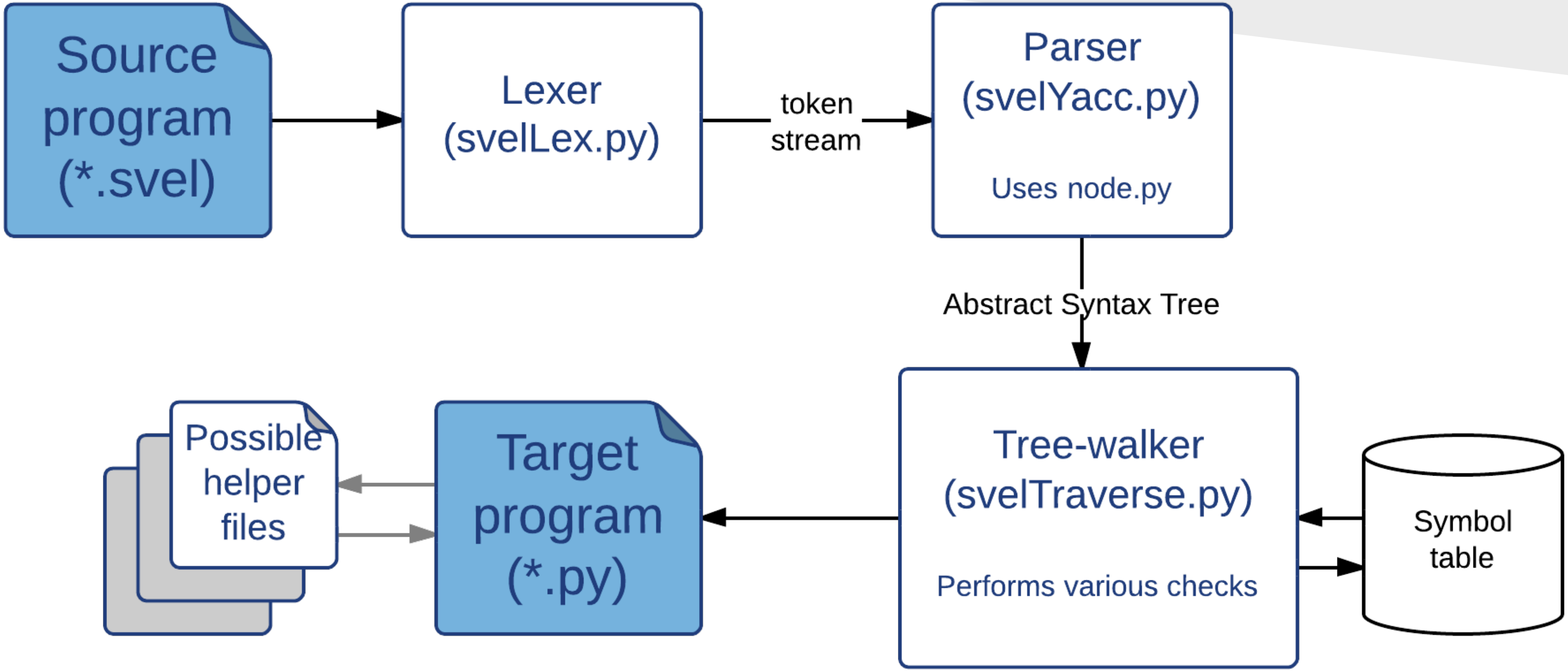
```
public static int fib(int n)
```

The verbose flag

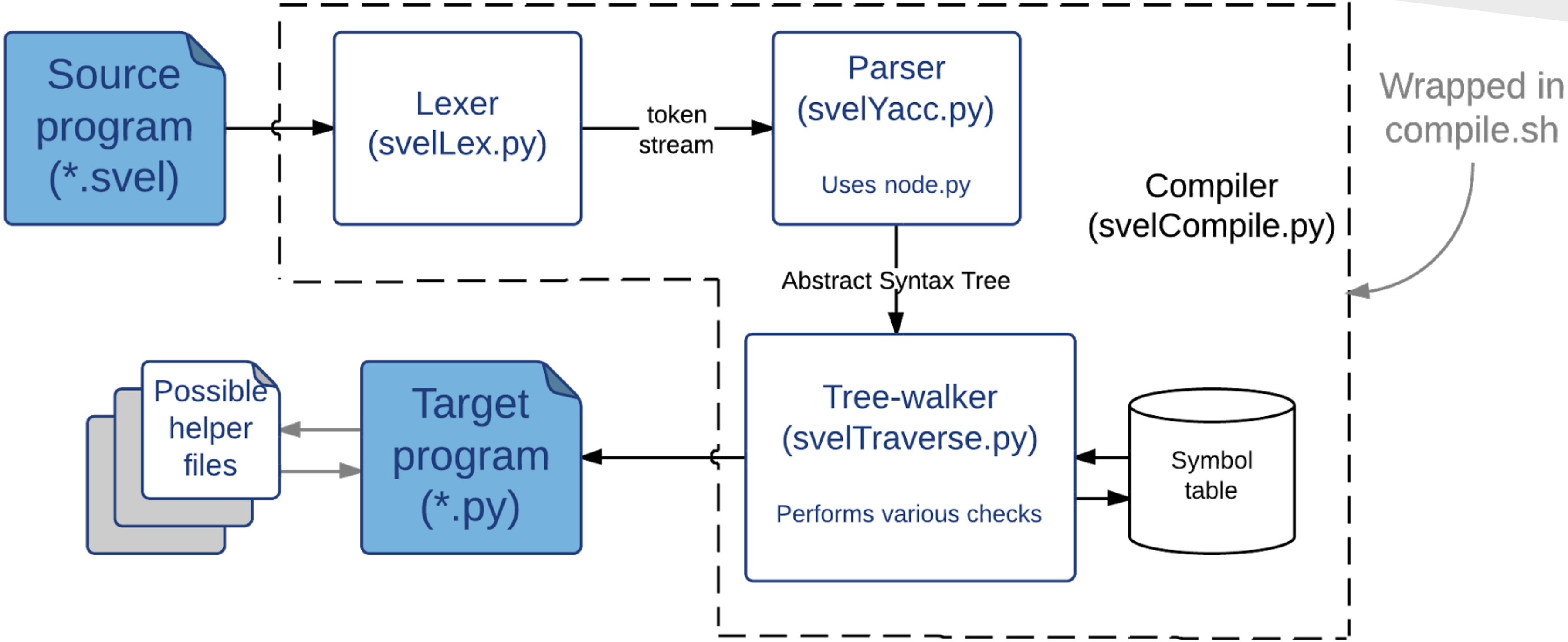


```
[18:28:12] emily ~/Dropbox/projects/svelTest/demo/fib (git:master)
$ python fibTester.py
============================================================================================
fib(4)...                                         PASS
fib(13)...                                        FAIL
        returned: 233
        expected: 2
```

Architecture



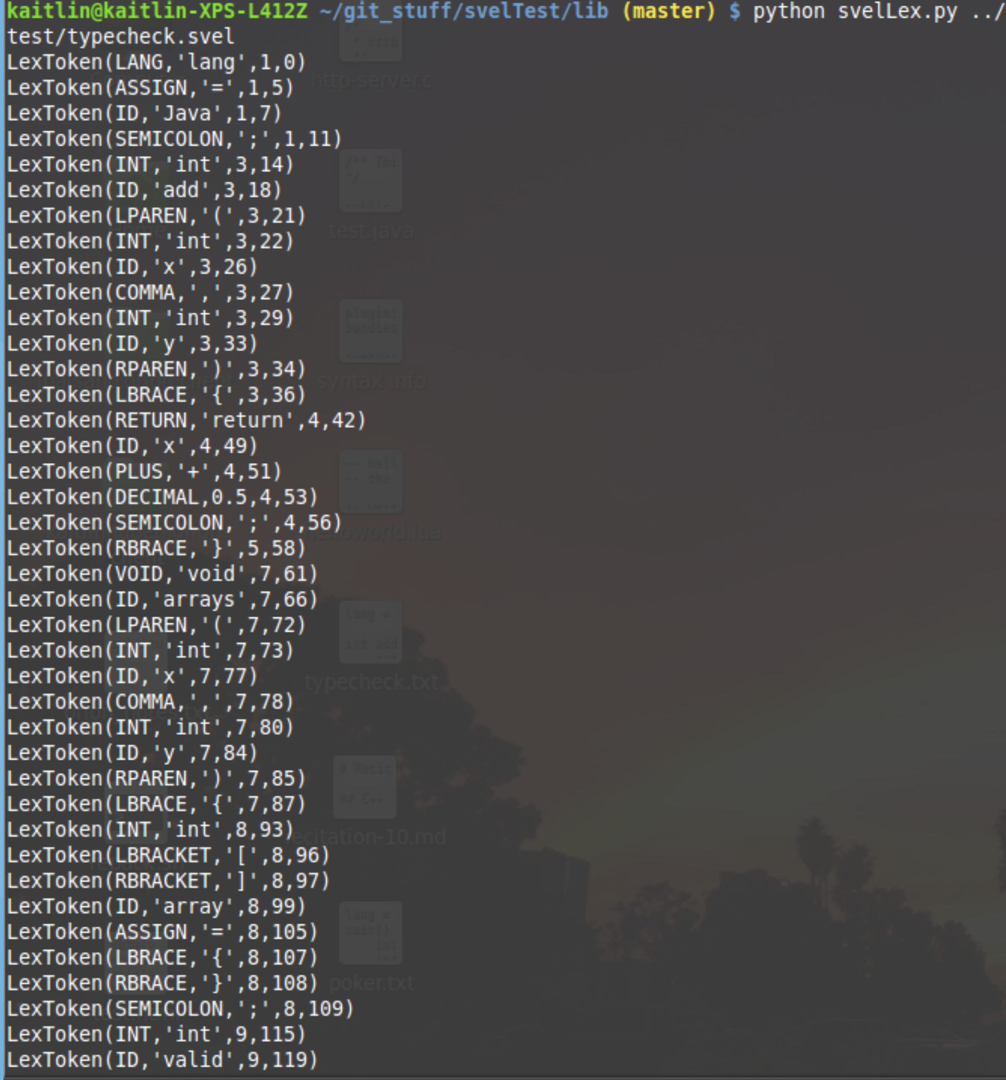
Architecture



Testing

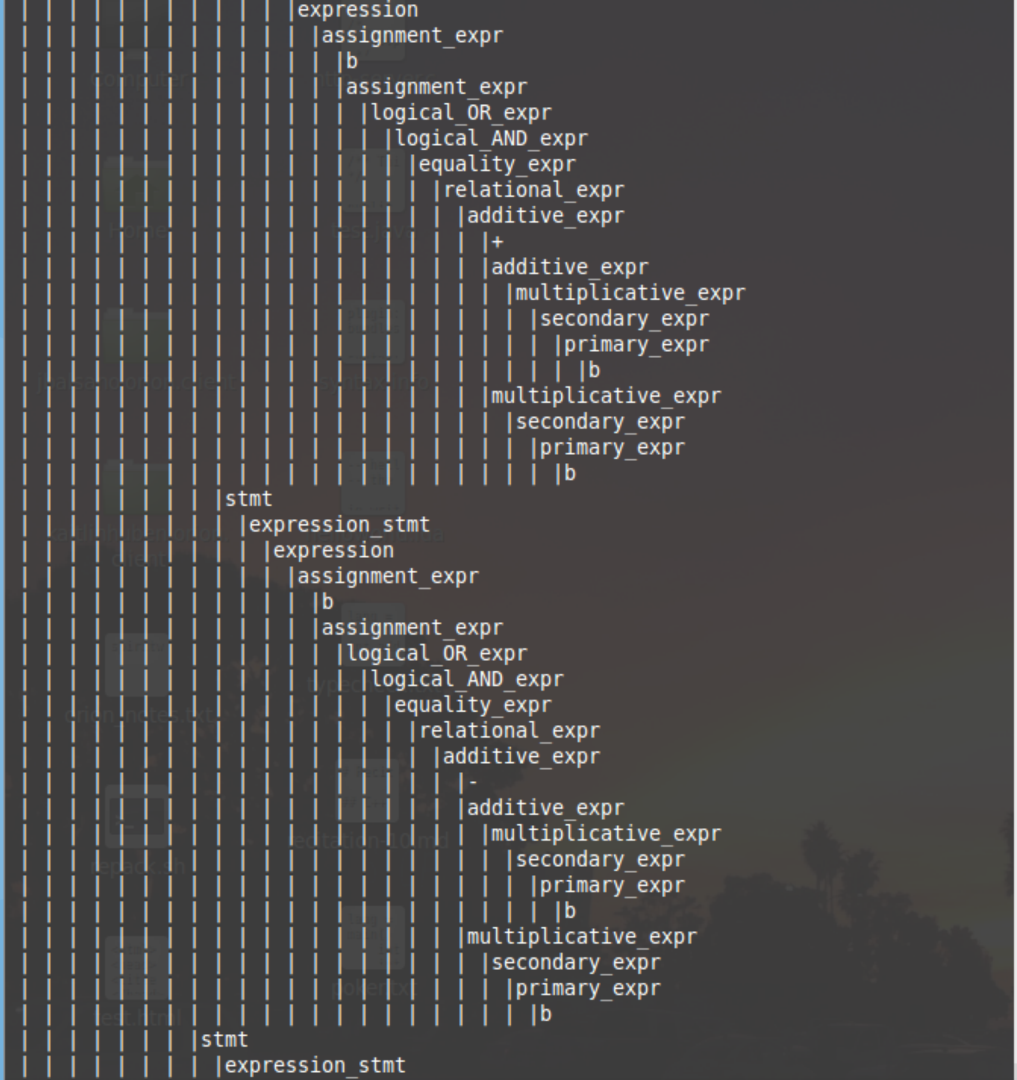
- Comprehensive test suite
- Lexer, parser, and compiler each implemented for on-the-fly testing
  - Can take single lines of code
  - Can take a program at any step (e.g. only lex)
- The meta-test: testing a svelTest program with another svelTest program!
  - (No, we haven’t gotten there yet)

Lexer



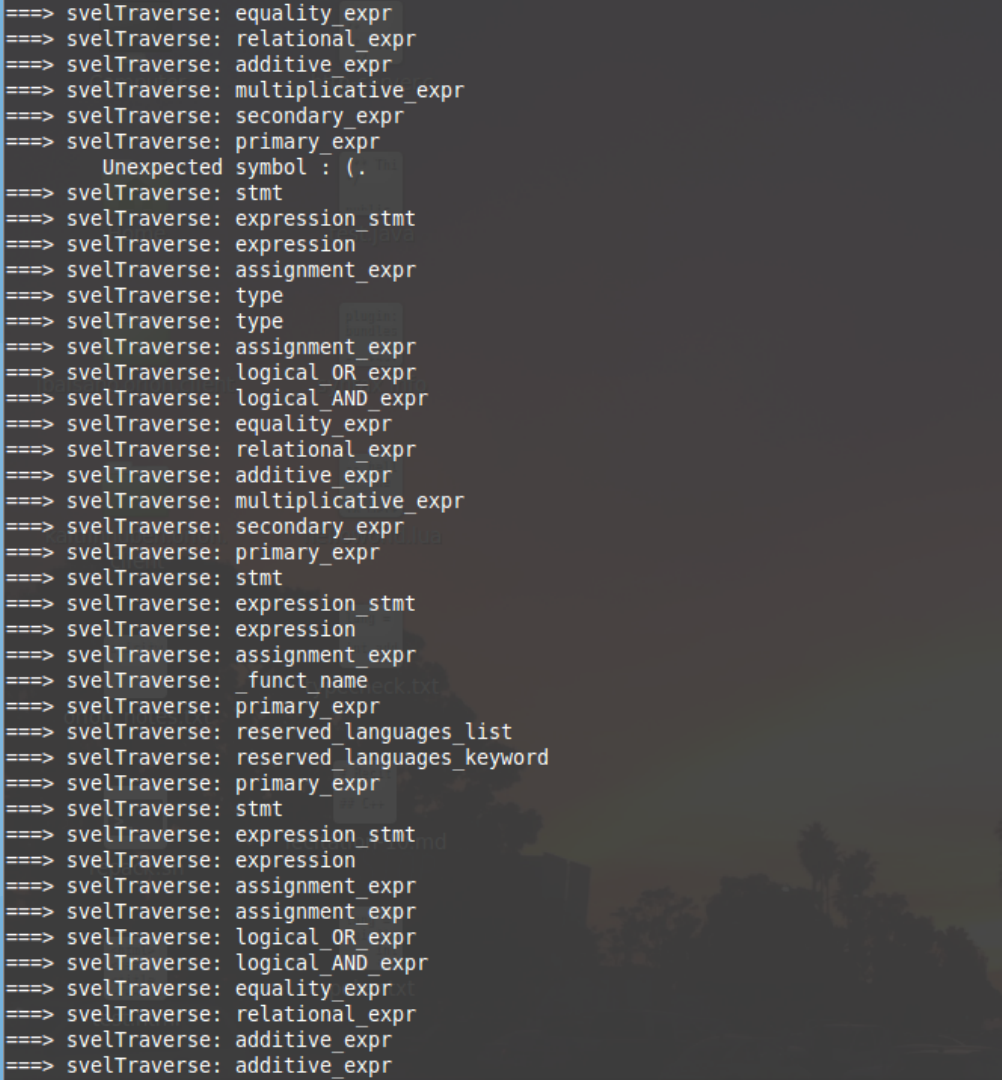
```
kaitlin@kaitlin-XPS-L412Z ~/git_stuff/svelTest/lib (master) $ python svelLex.py ../
test/typecheck.svel
LexToken(LANG,'lang',1,0)
LexToken(ASSIGN,'=',1,5)
LexToken(ID,'Java',1,7)
LexToken(SEMICOLON,';',1,11)
LexToken(INT,'int',3,14)
LexToken(ID,'add',3,18)
LexToken(LPAREN,'(',3,21)
LexToken(INT,'int',3,22)
LexToken(ID,'x',3,26)
LexToken(COMMA,',',3,27)
LexToken(INT,'int',3,29)
LexToken(ID,'y',3,33)
LexToken(RPAREN,')',3,34)
LexToken(LBRACE,'{',3,36)
LexToken(RETURN,'return',4,42)
LexToken(ID,'x',4,49)
LexToken(PLUS,'+',4,51)
LexToken(DECIMAL,0.5,4,53)
LexToken(SEMICOLON,';',4,56)
LexToken(RBRACE,'}',5,58)
LexToken(VOID,'void',7,61)
LexToken(ID,'arrays',7,66)
LexToken(LPAREN,'(',7,72)
LexToken(INT,'int',7,73)
LexToken(ID,'x',7,77)
LexToken(COMMA,',',7,78)
LexToken(INT,'int',7,80)
LexToken(ID,'y',7,84)
LexToken(RPAREN,')',7,85)
LexToken(LBRACE,'{',7,87)
LexToken(INT,'int',8,93)
LexToken(LBRACKET,'[',8,96)
LexToken(RBRACKET,']',8,97)
LexToken(ID,'array',8,99)
LexToken(ASSIGN,'=',8,105)
LexToken(LBRACE,'{',8,107)
LexToken(RBRACE,'}',8,108)
LexToken(SEMICOLON,';',8,109)
LexToken(INT,'int',9,115)
LexToken(ID,'valid',9,119)
```

Parser



```
| | | | | | | | | | | |expression
| | | | | | | | | | | | |assignment_expr
| | | | | | | | | | | | | |b
| | | | | | | | | | | | | |assignment_expr
| | | | | | | | | | | | | | |logical_OR_expr
| | | | | | | | | | | | | | | |logical_AND_expr
| | | | | | | | | | | | | | | | |equality_expr
| | | | | | | | | | | | | | | | | |relational_expr
| | | | | | | | | | | | | | | | | | |additive_expr
| | | | | | | | | | | | | | | | | | | |+
| | | | | | | | | | | | | | | | | | | |additive_expr
| | | | | | | | | | | | | | | | | | | | |multiplicative_expr
| | | | | | | | | | | | | | | | | | | | | |secondary_expr
| | | | | | | | | | | | | | | | | | | | | | |primary_expr
| | | | | | | | | | | | | | | | | | | | | | | |b
| | | | | | | | | | | | | | | | | | | |multiplicative_expr
| | | | | | | | | | | | | | | | | | | | |secondary_expr
| | | | | | | | | | | | | | | | | | | | | |primary_expr
| | | | | | | | | | | | | | | | | | | | | | |b
| | | | | | | | |stmt
| | | | | | | | | |expression_stmt
| | | | | | | | | | |expression
| | | | | | | | | | | |assignment_expr
| | | | | | | | | | | | |b
| | | | | | | | | | | | |assignment_expr
| | | | | | | | | | | | | |logical_OR_expr
| | | | | | | | | | | | | | |logical_AND_expr
| | | | | | | | | | | | | | | |equality_expr
| | | | | | | | | | | | | | | | |relational_expr
| | | | | | | | | | | | | | | | | |additive_expr
| | | | | | | | | | | | | | | | | | |-
| | | | | | | | | | | | | | | | | | |additive_expr
| | | | | | | | | | | | | | | | | | | |multiplicative_expr
| | | | | | | | | | | | | | | | | | | | |secondary_expr
| | | | | | | | | | | | | | | | | | | | | |primary_expr
| | | | | | | | | | | | | | | | | | | | | | |b
| | | | | | | | | | | | | | | | | | |multiplicative_expr
| | | | | | | | | | | | | | | | | | | |secondary_expr
| | | | | | | | | | | | | | | | | | | | |primary_expr
| | | | | | | | | | | | | | | | | | | | | |b
| | | | | | | |stmt
| | | | | | | | |expression_stmt
```

Compiler (verbose)



```
===> svelTraverse: equality_expr
===> svelTraverse: relational_expr
===> svelTraverse: additive_expr
===> svelTraverse: multiplicative_expr
===> svelTraverse: secondary_expr
===> svelTraverse: primary_expr
      Unexpected symbol : (.
===> svelTraverse: stmt
===> svelTraverse: expression_stmt
===> svelTraverse: expression
===> svelTraverse: assignment_expr
===> svelTraverse: type
===> svelTraverse: type
===> svelTraverse: assignment_expr
===> svelTraverse: logical_OR_expr
===> svelTraverse: logical_AND_expr
===> svelTraverse: equality_expr
===> svelTraverse: relational_expr
===> svelTraverse: additive_expr
===> svelTraverse: multiplicative_expr
===> svelTraverse: secondary_expr
===> svelTraverse: primary_expr
===> svelTraverse: stmt
===> svelTraverse: expression_stmt
===> svelTraverse: expression
===> svelTraverse: assignment_expr
===> svelTraverse: _funct_name
===> svelTraverse: primary_expr
===> svelTraverse: reserved_languages_list
===> svelTraverse: reserved_languages_keyword
===> svelTraverse: primary_expr
===> svelTraverse: stmt
===> svelTraverse: expression_stmt
===> svelTraverse: expression
===> svelTraverse: assignment_expr
===> svelTraverse: assignment_expr
===> svelTraverse: logical_OR_expr
===> svelTraverse: logical_AND_expr
===> svelTraverse: equality_expr
===> svelTraverse: relational_expr
===> svelTraverse: additive_expr
===> svelTraverse: additive_expr
```

Compiler-generator tools

- PLY: Python Lex-Yacc

[3]

Development environment

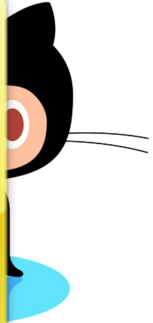
- Python 2.7.3
- PLY 3.4-2build1
- Java SE 6 (1.6.0)
- C99
- CLIC Linux Machines/Local Mac/Windows

Runtime Environment

- svelTest utilizes the user’s environment
  - Compiled programs use direct system calls
  - All svelTest programs need Python 2.x to run
- How to compile and run:
  -
    ```
    $ cd path/to/lib
    $ ./compile.sh [-v] path/to/your_svel_program.svel
      compiler output
    $ cd path/to/your_svel_program
    $ python your_svel_program.py
    ```

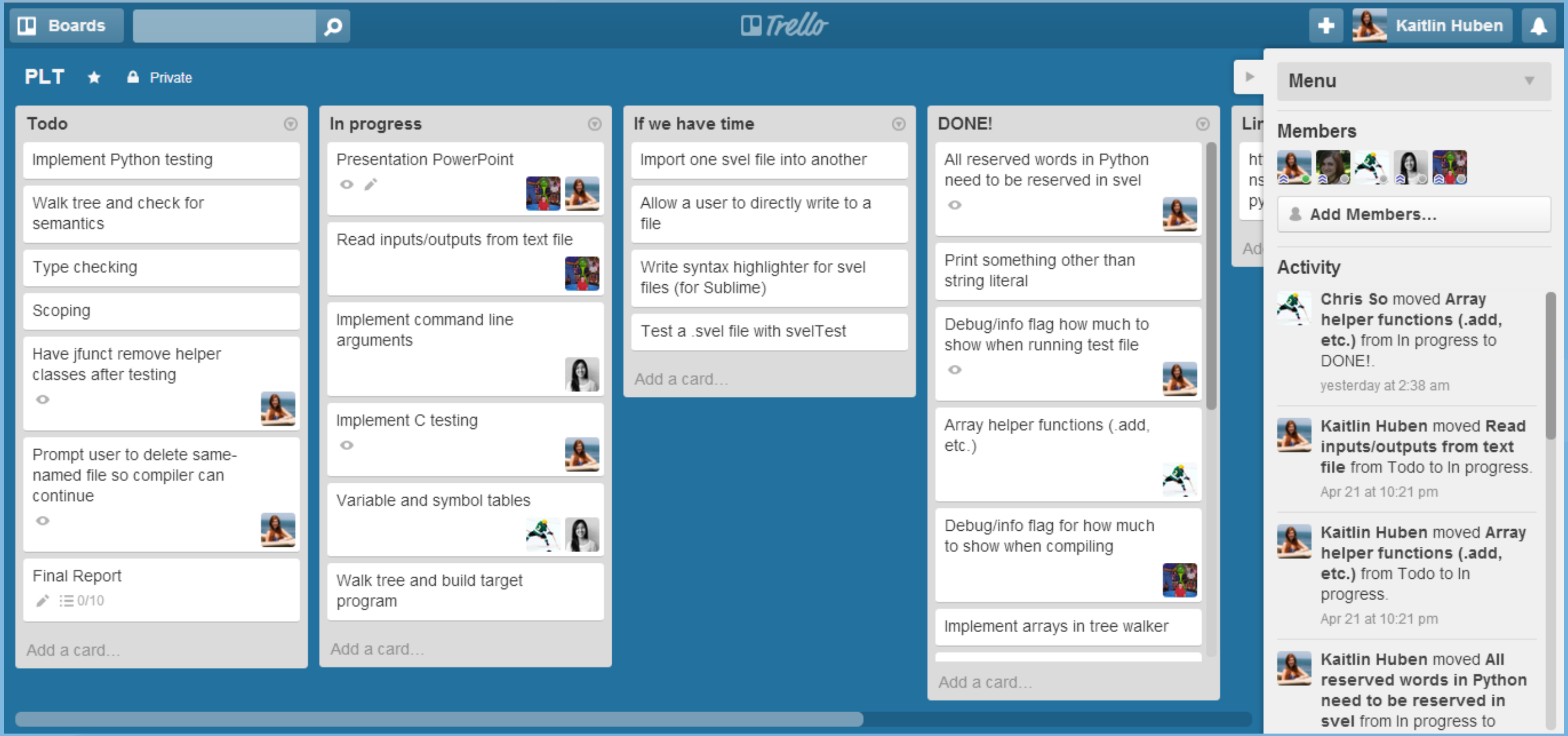
Demo!

Project management



[4]

Trello



Todo
- Implement Python testing
- Walk tree and check for semantics
- Type checking
- Scoping
- Have jfunct remove helper classes after testing
- Prompt user to delete same-named file so compiler can continue
- Final Report 0/10

In progress
- Presentation PowerPoint
- Read inputs/outputs from text file
- Implement command line arguments
- Implement C testing
- Variable and symbol tables
- Walk tree and build target program

If we have time
- Import one svel file into another
- Allow a user to directly write to a file
- Write syntax highlighter for svel files (for Sublime)
- Test a .svel file with svelTest

DONE!
- All reserved words in Python need to be reserved in svel
- Print something other than string literal
- Debug/info flag how much to show when running test file
- Array helper functions (.add, etc.)
- Debug/info flag for how much to show when compiling
- Implement arrays in tree walker

Lir

ht
ns
py

Activity

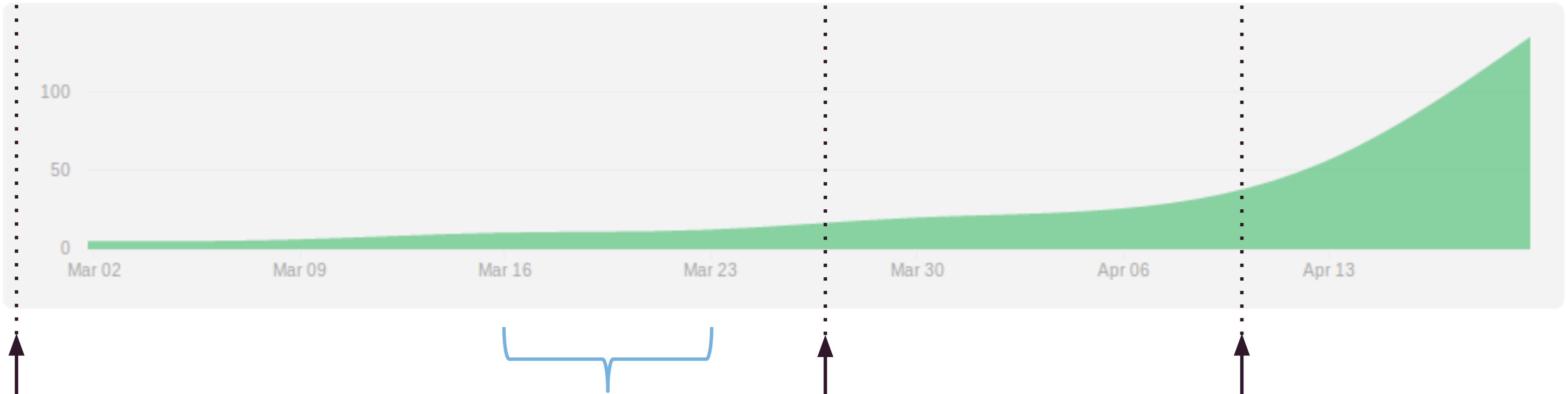
Chris So moved **Array helper functions (.add, etc.)** from In progress to DONE!.
yesterday at 2:38 am

Kaitlin Huben moved **Read inputs/outputs from text file** from Todo to In progress.
Apr 21 at 10:21 pm

Kaitlin Huben moved **Array helper functions (.add, etc.)** from Todo to In progress.
Apr 21 at 10:21 pm

Kaitlin Huben moved **All reserved words in Python need to be reserved in svel** from In progress to

Git: Commits



Whitepaper due

Spring break

LRM/Tutorial due

Realized we were presenting in class

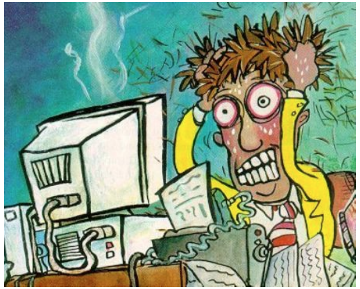
Lots of learning

- Start early, work often
- Programming language != application
- What worked well:
  - Teamwork!
  - (At least) weekly meetings, standups, or scrums
  - Lots of commits
  - Ambitious goals: “if we have time” list on Trello
- What we would have changed:
  - Get full bare-bones compiler kernel running sooner (goal was week of 3/9, didn’t happen until 3/30)
  - Higher comfort level with git (mistakes fixed with rebasing caused trouble)

So...why svelTest, again?

- Testing code should be easy
  - You don’t want to end up like these guys
- You shouldn’t need to learn a new library for every language you want to test

Our logo is awesome Sveltest



[5]

Thank you!

Bonus Pic: Team Bonding!

HAPPY BIRTHDAY - ❤ your team

Aww thanks guys!

Repository

Our code:

https://github.com/svelTest/svelTest

Language tutorial:

http://sveltest.github.io/tutorial/

Image citations

Cover: http://media-cache-ak0.pinimg.com/736x/1c/69/2e/1c692e55506a47fc718a16ee28e8b38b.jpg

1. http://c.tadst.com/gfx/600x400/international-programmer-day.jpg
2. http://sheownsit.com/wp-content/uploads/2013/08/How-Flexible-Is-Your-BI-1.jpg
3. http://media.tumblr.com/tumblr_m5vygksZWP1qhsyf1.jpg (dog)
   http://sd.keepcalm-o-matic.co.uk/i/i-m-so-awesome-that-you-cannot-keep-calm.png
4. Project Management slide:
   a. Aho: http://www1.cs.columbia.edu/~aho/Simons_Aho_1.jpg
   b. Email: http://before-ido.com/wp-content/uploads/2014/01/email-icon.jpg
   c. Message: http://www.iphonehacks.com/wp-content/uploads/2013/10/imessage-icon-ios-7.jpg
   d. Github: http://www.velonuboso.com/made/blog/wp-content/uploads/2013/04/github-logo-transparent.jpg
   e. Trello: http://blog.trello.com/wp-content/uploads/2011/09/03-Trello-300x173.png
   f. Drive: http://fc01.deviantart.net/fs70/i/2012/136/3/a/google_drive_library_by_gsm2k-d500d7l.png
   g. Snapchat: http://upload.wikimedia.org/wikipedia/en/5/5e/Snapchat_logo.png
5. http://www.the-scoop.co.uk/wp-content/uploads/2013/10/angry-computer.jpg