AVL: Algorithm Visualization language

Jiuyang Zhao	System Tester
Qianxi Zhang	Project Manager
Qinfan Wu	System Integrator
Shining Sun	Language Guru
Yu Zheng	System Architect

Motivation

“A picture is worth a thousand words.”

“Then how about a video?”

Algorithm Visualization

- Algorithms are hard
- A better way to teach and learn
- To see to learn!

Demo

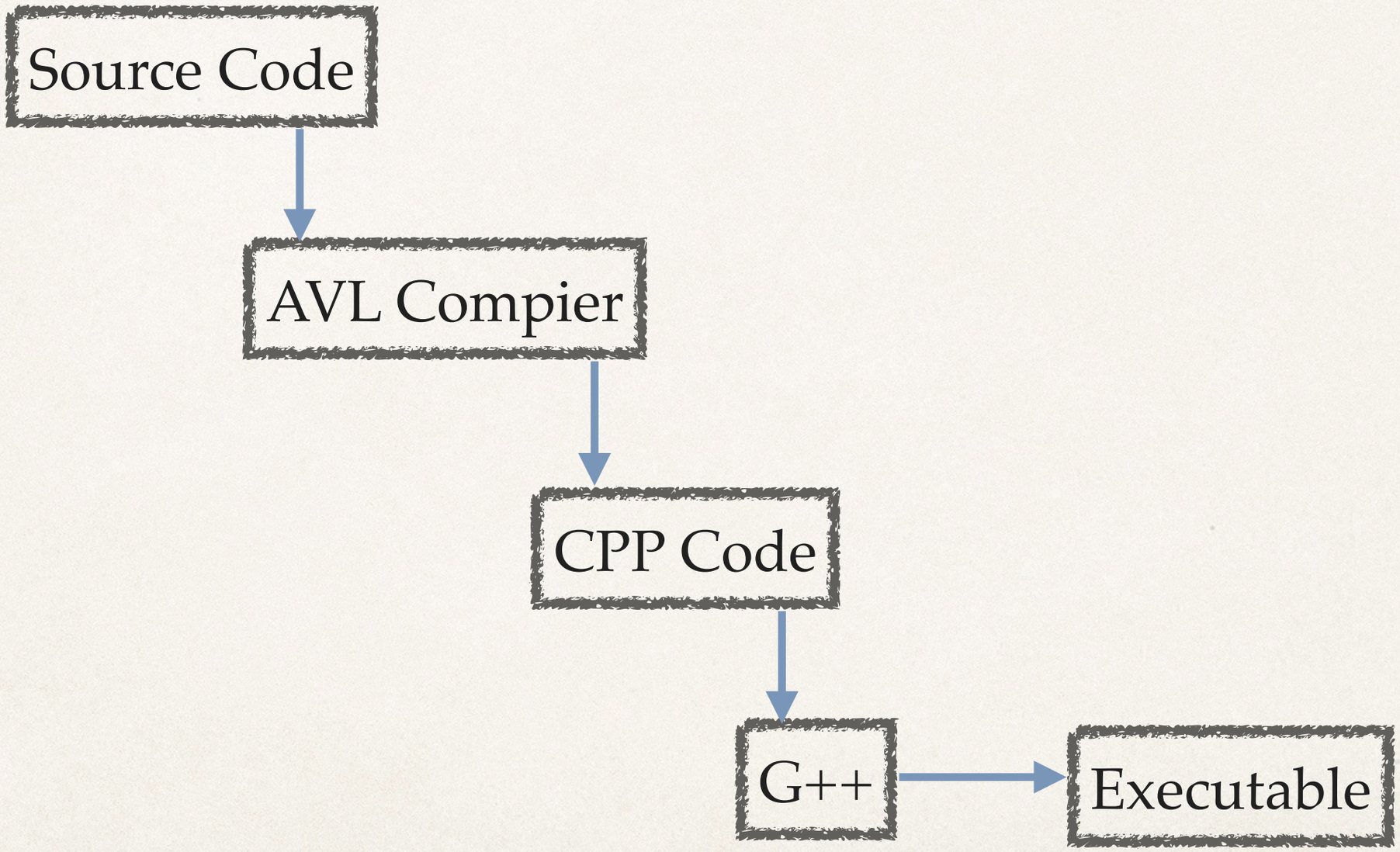
Compile and Run

- Save source code (plain text) into some .avl file

- Compile:
  - avl -o exe sample.avl

- ./exe

Compile and Run



Properties

- Visual
- Educational
- Easy to learn

Syntactic Constructs

- C Style language
- New features
- Display and hide keywords
- <begin_display>, <end_display> block
- Subarray & index
- Swap function

Display and Hide

- Indicate whether the variable is displayable on screen
- Can be used in declaration & expression
- Default: hide

```
display int a[] = {1, 2};
hide int b[] = {3, 4};

hide a;
display b;
```

<begin_display> and <end_display>

- Operations between the two tags will be displayed
- Only displayable variables will be shown

```
display int a[] = {1, 2};
hide int b[] = {3, 4};
<begin_display>
a[0] = b[0];
a[1] = b[1];
<end_display>
```

Index & Subarray

- Index
  - Similar to int type
  - Helps to highlight elements inside an array
- Subarray
  - Convenient expression

```
display int a[] = {1, 2};
hide int b[] = {3, 4};
index i;
<begin_display>
for (i=0; i<2; i++)
      a[i] = b[i];
<end_display>


quicksort(a[0:i]);
quicksort(a[i+1:n]);
```

Swap

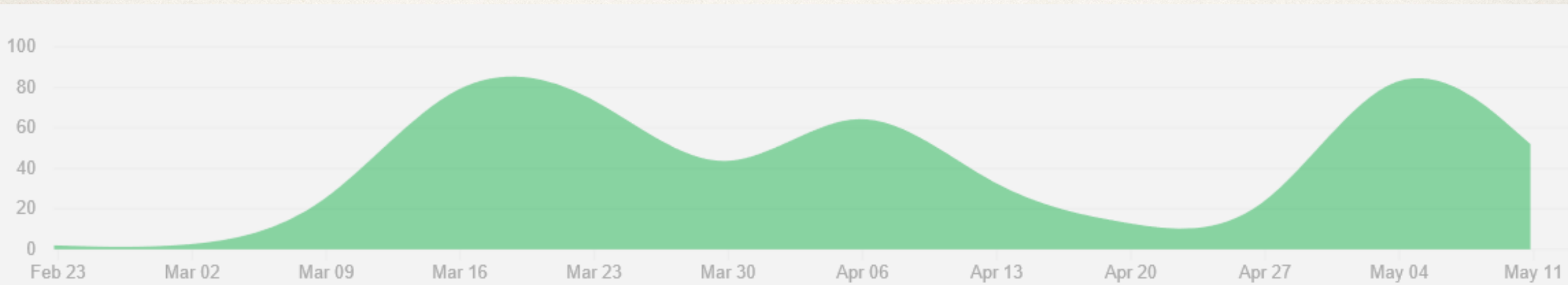
- Create animation for swapping two elements in one array

```
temp = a[0];
a[0] = a[1];
a[1] = temp;
```

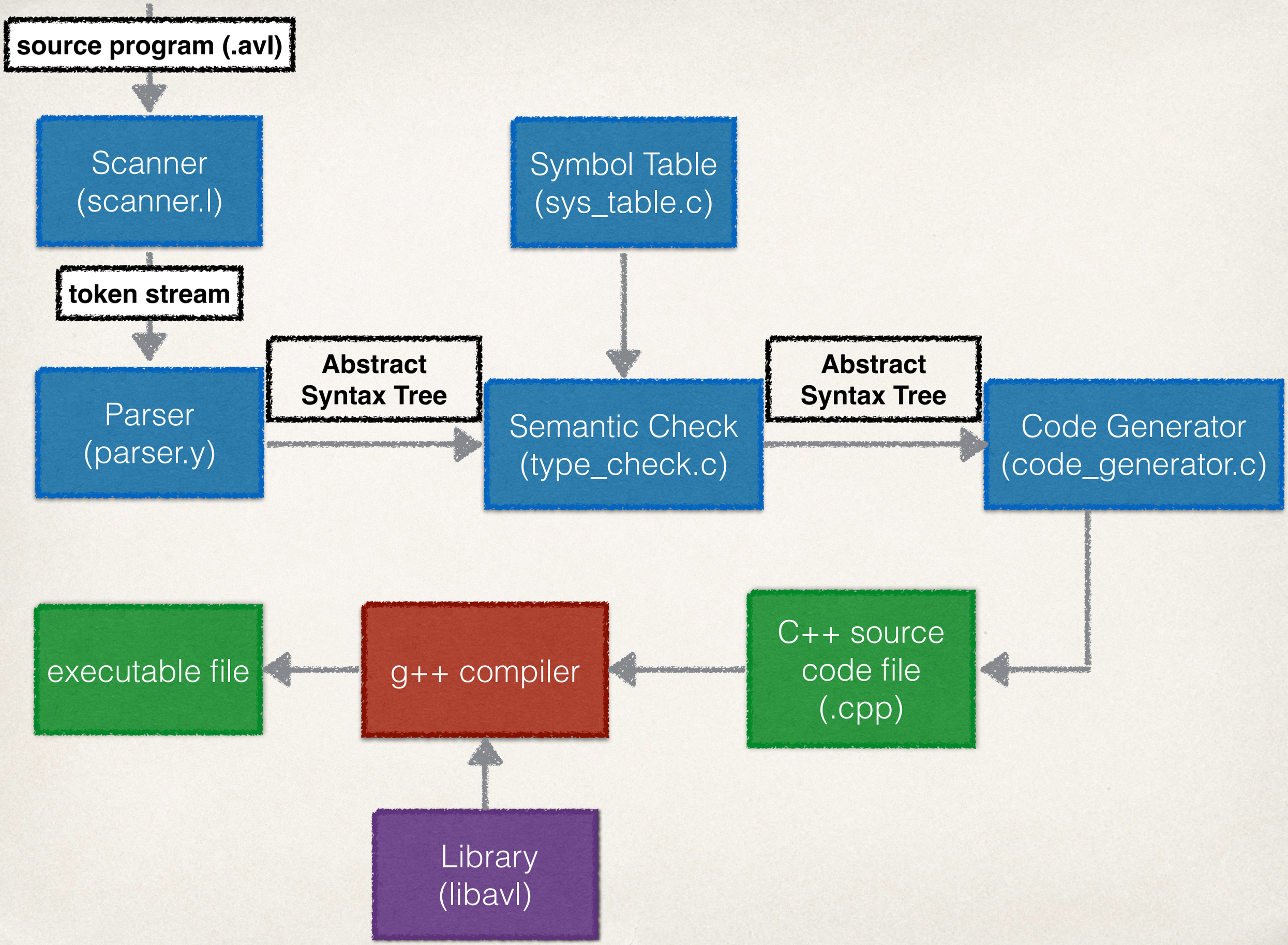
```
swap(a, 0, 1);
```

Project Management

- Weekly meeting
- Github
- Github commits over time:



Translator Architecture


source program (.avl)
Scanner
(scanner.l)
token stream
Parser
(parser.y)
Abstract
Syntax Tree
Symbol Table
(sys_table.c)
Semantic Check
(type_check.c)
Abstract
Syntax Tree
Code Generator
(code_generator.c)
executable file
g++ compiler
C++ source
code file
(.cpp)
Library
(libavl)

array_swap.avl

```
int main(){

    display int array[5] = {1,2,3,4,5};

    <begin_display>
        swap(array,1,2);
    <end_display>

    return 0;
}
```

array_swap.cpp

```
AvlVisualizer *__avl__vi = NULL;
bool __avl__ready()
{
    return __avl__vi != NULL;
}
std::mutex __avl__mtx;
std::condition_variable_any __avl__cv;

void __avl__display(int argc, char **argv)
{
    __avl__mtx.lock();
    __avl__vi = new AvlVisualizer(argc, argv);
    __avl__cv.notify_one();
    __avl__mtx.unlock();
    __avl__vi->show();
}

int main(int argc, char *argv[] )
{
    std::thread __avl__loop(__avl__display, argc, argv);
    __avl__mtx.lock();
    __avl__cv.wait(__avl__mtx, __avl__ready);
    __avl__mtx.unlock();
    avlSleep(0.5);
    {
        AvlArray< AvlInt> array = { 1, 2, 3, 4, 5 };
        array.set_name("array");
        __avl__vi->addObject(&array, "array");

        __avl__vi->start();
        avlSleep(0.5);
        avlSleep(0.15);
        array.swap( 1, 2 );
        avlSleep(0.15);
        avlSleep(0.1);
        __avl__vi->stop();
        __avl__loop.join();
        delete __avl__vi;
        return 0;
    }
}
```

video

Generator Tools

Yacc

Lex

g++ compiler

LibAVL

AvlTypes.h

AvlUtils.h

AvlVisualizer.h

AvlVisualizer.cpp

+

OpenGL

Software Development Environment

Modular design

Translator

avl/src/
avl/src/avl.c
avl/src/scanner.l
avl/src/parser.y
avl/src/sym_table.c
avl/src/type_check.c
avl/src/code_generator.c
.....

Library

avl/lib/
avl/lib/AvlTypes.h
avl/lib/AvlVisualizer.h
avl/lib/AvlVisualizer.cpp
avl/lib/AvlUtils.h
.....

Tests

avl/tests/
avl/tests/libavl.test
avl/tests/avl.test
.....

Strict compiler flags

```
lib_LTLIBRARIES = libavl.la
libavl_la_SOURCES = AvlVisualizer.cpp
include_HEADERS = AvlVisualizer.h AvlTypes.h AvlUtils.h
AM_CPPFLAGS = -Wall -Wextra -Werror -std=c++11 -I/opt/local/include
```

Run-time environment

configure.ac

Makefile.am
src/Makefile.am
lib/Makefile.am
tests/Makefile.am
tests/avl.test/Makefile.am
.....

autoconf
automake
libtool

configure

./configure

config.h
Makefile
src/Makefile
lib/Makefile
tests/Makefile
tests/avl.test/Makefile
.....

souce code

Flex, Bison, Gcc

FreeGLUT

make && make install

/usr/local/bin/avl
/usr/local/include/AvlVisualizer.h
/usr/local/include/AvlTypes.h
/usr/local/lib/
    libavl.a
    libavl.la
    libavl.so -> libavl.so.0.0.0
    libavl.so.0 -> libavl.so.0.0.0
    libavl.so.0.0.0

How to install avl to your system:

```
$ tar xvzf avl.tar.gz
$ cd avl
$ ./configure
$ make
$ sudo make install
```

Everything is fully GNU standards-compliant

How to use our compiler:

```
$ avl --help
Usage: avl [-h|-o|-t] file
  Options:  -h --help                Display this information
            -o --output=<file>       Compile and place the executable into <file>
            -t --translate           Translate the source files into c++
                                          xxx.avl -> xxx.cpp
Use at most one option at a time.
For bug reporting instructions, please see:<https://github.com/wqfish/avl>.
```

Examples:

```
$ avl test.avl
$ ./a.out
```

```
$ avl -o test test.avl
$ ./test
```

Test Tool

- Automake in GNU will generate automatic test suite harness
- Output
  - result of each test case
  - log file for each test case
  - statistics
- Command: make check

```
PASS: statement_for.sh
FAIL: insertion_sort.sh
PASS: operator_char_mid.sh
PASS: declaration_bool_array.sh
PASS: declaration_char_array.sh
PASS: declaration_int_array.sh
PASS: expression_bool_array.sh
PASS: display_bool_array.sh
PASS: display_char_array.sh
PASS: expression_char_array.sh
PASS: expression_int_array.sh
PASS: display_int_array.sh
make[5]: Nothing to be done for `all'.
============================================================================
Testsuite summary for avl 0.1
============================================================================
# TOTAL: 61
# PASS:  40
# SKIP:  0
# XFAIL: 2
# FAIL:  18
# XPASS: 1
# ERROR: 0
============================================================================
See tests/avl.test/test-suite.log
Please report to avl-plt@googlegroups.com
============================================================================
make[4]: *** [test-suite.log] Error 1
make[3]: *** [check-TESTS] Error 2
make[2]: *** [check-am] Error 2
make[1]: *** [check-recursive] Error 1
make: *** [check-recursive] Error 1
```

Test Plan

- Test cases:
  - Covering every line of grammar
  - Displaying every objects (data types)
  - XFail cases for scope/type checking

Conclusion

- What we have learned
- Why everyone should use your language **fun!**

Thank you!

AVL Team Presents