Crazy Easy Scripting with ChaiScript

Jason Turner

- http://chaiscript.com
- http://cppcast.com
- http://cppbestpractices.com
- http://github.com/lefticus
- Independent Contractor

ChaiScript

- Header only
- 0 external dependencies
- Mature: 6 years old
- Stable: every release is tested against
  - clang / MSVC / g++
  - MacOS / Windows / Linux
  - 64bit / 32bit
  - AddressSanitizer / ThreadSanitizer
  - MSVC's Static Analyzer / cppcheck
- Thread safe by default (20% perf boost if disabled)
- Crazy easy to use

ChaiScript History

- Developed with SWIG but the build process overhead was frustrating for small projects
- At the same time I started wondering what it would take to do multimethod dispatch with C++
- And my cousin, a language geek, was interested in working on a language with me.

Then ChaiScript was born. First as a toolkit to build your own scripting language with ChaiScript being just one reference implementation.

You can still see some of this in the file layout with the `dispatchkit` and `language` sub folders.

ChaiScript Goals

- Syntax feel natural to C++ developers
- Seemless integration with C++
- Not get in the way
- No pre-processor or complex build process
- Be "fast enough" (~1.5M C++ callbacks / second possible currently)

ChaiScript is Not as Fast as Lua

- But it doesn't need to be
- If you need performance, simply call into C++

Basics

```cpp
#include <chaiscript/chaiscript.hpp>

int main()
{
  chaiscript::ChaiScript chai;

  chai.eval(R"(
    print("Hello ChaiScript")
  )");
}
```

Adding a Function

```cpp
#include <chaiscript/chaiscript.hpp>
#include <iostream>

void say_hi()
{
  std::cout << "Hello C++\n";
}

int main()
{
  chaiscript::ChaiScript chai;

  chai.add(chaiscript::fun(&say_hi), "say_hi");

  chai.eval(R"(
    say_hi();
  )");
}
```

Sharing a Value

```
#include <chaiscript/chaiscript.hpp>

int main()
{
  chaiscript::ChaiScript chai;

  int &i = chai.eval<int &>(R"(
    var i = 5;
    i;
  )");

  i = 20;

  chai.eval(R"(
    print(i); // prints 20
  )");
}
```

Passing a Function

```
#include <chaiscript/chaiscript.hpp>

void do_something(const std::function<double (double, double)> &f)
{
  std::cout << "Calculated: " << f(2.6, 3.5) << '\n';
}

int main()
{
  chaiscript::ChaiScript chai;
  chai.add(chaiscript::fun(&do_something), "do_something");

  chai.eval(R"(
    do_something(`+`); // prints "Calculated 6.1"
    do_something(`-`); // prints "Calculated -.9"
    do_something(`*`); // prints "Calculated 9.1"
    do_something(fun(x, y) { x + y / 3 + y} ); // prints "Calculated 7.2666667"
  )");
}
```

ChaiScript as a Configuration Language

- Why use an INI?

Jason Turner

- http://chaiscript.com
- http://cppcast.com
- http://cppbestpractices.com
- http://github.com/lefticus
- @chaiscript
- @lefticus
- Independent Contractor - *Always interested in meeting new clients*