$LANG Language Proposal

Daniel Cole, Megan Skrypek, Rashedul Haydar, Tim Waterman
dhc2131, ms4985, rh2712, tbw2105

September 30, 2014

Motivation

Most "modern" programing languages trace their origins back decades to before the advent of cheap, general purpose multicore CPUs. They were designed for a distinctly mono-threaded environment. While libraries and enhancements to mainstay languages such as C/C++ and Java have added multithreading capabilities, it remains in many ways bolted on kludge. While newer frameworks such as Node.js provide more integral support for asynchronous operations, they lack the depth of support and power of a fully compiled language. With $Lang, we aim to build a language that has the power and flexibility of a fully compiled C style language, while having native threading support for modern multithreaded applications.

Description

$Lang is inspired by C, which has a very well known syntax, and has been one of the most widely used languages since it was released over forty years ago. $Lang is a general purpose language that supports all standard mathematical and logical operations. In addition to the standard C primitive types (`int, double, char`, etc.), $Lang has native support for the `string` type. This includes concatenation, and a inbuilt length operator.

Examples

```
//comments in Sink are similar to C comments
//this is a single line comment
/*
You can also have multi-line comments
*/

//functions are declared using the 'def' keyword, like Python
def int main() {

        var bool b = true; //booleans are a primitive data type
        var int x = 7; //var keyword declares a variable

        var string s = "This is a String\n";

        var string h = "Sink also supports " + "string concatenation!";

        let double PI = 3.14; //let is used to define constants

        return 0;

}

/*
async keyword: used on function calls and on loop constructs, to
make the loop execute asynchronously
*/

def int main() {

        var int even_sum = 0;
        var int odd_sum = 0;

        //adds up all the even numbers from 0 to 100 million
        async for(var int i = 0; i < 100000000; i+=2) {
                even_sum += i;
        }

        //adds up all the odd numbers from 0 to 100 million
        async for(var int i = 1; i < 100000000; i+=2) {
                odd_sum += i;
        }

        printf("The sum of all values is %d", even_sum + odd_sum);

        return 0;
}
```