HAWK (HTML is All We Know)
Language Proposal

Jonathan Adelson, Ethan Benjamin, Justin Chang,Graham Gobieski, George Yu
jma2215, jc4137, eb2947, gsg2120, gy2206

Syntax Notation

For all parts of this manual relating to syntax, the following conventions will be used. Syntactic categories will be shown in italic (e.g. *expr* or *stmt*) and literal characters and strings will be shown in Courier fornt (e.g. `if (1==1){a=b;}`).

Meaning of Identifiers

Identifiers are names which can refer to functions, variables, and table fields. Each identifier is a string consisting of digits, letters, and underscores which does not begin with a digit.

Variables are storage locations that contain values. Depending on where in a program variables are initialized, they are either **global** or **local** to a particular scope. See **Storage Scope** for more details.

DYNAMICALLY TYPED VERSION: HAWK is dynamically typed, which means that the value stored by a variable has a type, but variables themselves have no type.

STATICALLY TYPED VERSION:

HAWK is statically typed, which means that every variable has a type. The type of a variable determines the meaning and behavior of its values, and also the nature of storage needed for those values.

Storage Scope

The visibility of an identifier and liftetime of a variable's storage depends on where a variable is initialized. If a variable is initialized within a `BEGIN` or `END` block, it is a **global** variable. A global variable can be accessed by any part of the program below the global variable's initialization. It's storage stays alive throughout the entire execution of the program.

If a variable is initialized within any block other than a `BEGIN` or `END` block, it is a **local** variable. A local variable can be accessed within the scope it is initialized, at or below its initialization. It's storage will be destroyed at the end of the scope.

Basic Types

There are four basic types in HAWK: **integers**, **decimals**, **strings**, and **tables**.

Integers are signed integer values which can span an unlimited range. Similarly, decimals are signed decimal values which can span an unlimited range and precision. We will refer to decimals and integers as **arithmetic** type. Because arithmetic types can encompass an infinite range of values, they necessarily require a flexible amount of storage space. HAWK language implementations have the freedom to allocate memory to these values however they see fit.

Strings are a sequence of 0 or more unicode characters. They are guaranteed to occupy $O(n)$ space in memory, where n is the number of characters in the string.

Both arithmetic types and strings are **immutable**, which means that their value cannot be changed once they are created. When a variable with an immutable type (NOTE TO SELF: CHANGE FOR DYNAMIC) is assigned a new value, the old value and underlying storage are destroyed.

Tables, unlike the immutable types, are **objects**. This means that variables do not contain tables, but rather contain **references** to tables. Assigning a table to a variable results in that variable storing a reference to that table. Similarly, when tables are passed as parameters to functions or returned from functions, the respective parameters and return values are references. In each of these operations, there is no copying of internal table data, only copying of references.

HAWK uses reference counting to keep track of how many variables store references to the same table. When a table no longer has any variables referencing it, the underlying storage for the table is destroyed.

Automatic Conversions

Promotion of Integers in Mixed Arithmetic Expressions

In mathematical binary expressions where one operand is an integer and the other operand is a decimal, the integer will be automatically converted to a decimal value. The conversion will be performed using the the built-in function `int_to_string`.

String Conversion in String Concatentation Expressions

In binary addition expressions where one operand is a string, and the other operand is not a string, the non-string will be automatically converted to a string value. Tables will be converted using `table_to_string`, integers using `int_to_string`, and `decimal_to_string`.