ArcherCalc Guide

Aaron Hodgen

October 17, 2015

Contents

Chapter 1

Introduction

Archer-calc provides a way to express Archer calculations in re-usable and manageable pieces that can be type-checked, documented, tested, and compiled into a format suitable for Archer.

1.1 Quick Example

Here is an example. The following is a file named "`test.ac`":

```
//Archer fields that will be used in our calculations.
let incid_date   = field "Date of Incident" : Date as ISODate "2015-04-05";
let incid_status = field "Incident Status"  : Text as "Open";

// Calculate the number of days elapsed since a given date.
let daysAgo x = datedif x now Day;

// Given a date, return a color based on the age.
let date_color date =
    if ((daysAgo date) < 30) then
        "Green"
    else if ((daysAgo date) < 60) then
        "Yellow"
    else if ((daysAgo date) < 90) then
        "Orange"
    else
        "Red";

// Color items with a status of "Open"
let incid_color stat dt =
    if (stat == "Open") then
```

```
        (date_color dt)
    else
        "";
```

Note that we have not used the fields in the body of our functions, although it is possible to do so. This is considered a best practice for re-usability and testability. Functions with the field embedded can only be used on that particular field, but a function *taking* a field can work with any field of the same type.

Let's start ArcherCalc and load the file:

```
     _              _                 ____      _
    / \   _ __ ___| |__   ___ _ __ / ___|__ _| | ___
   / _ \ | '__/ __| '_ \ / _ \ '__| |   / _` | |/ __| Version 0.0.1.0
  / ___ \| | | (__| | | |  __/ |  | |__| (_| | | (__  Type :help for help
 /_/   \_\_|  \___|_| |_|\___|_|   \____\__,_|_|\___|

ArcherCalc is provided with ABSOLUTELY NO WARRANTY.
ArcherCalc> :load test.ac
```

Now lets test the date_color function:

```
ArcherCalc> date_color ISODate "2015-05-05"
"Red" : Text
ArcherCalc> date_color ISODate "2015-08-05"
"Orange" : Text
ArcherCalc> date_color ISODate "2015-09-05"
"Yellow" : Text
ArcherCalc> date_color ISODate "2015-10-05"
"Green" : Text
```

We can also run it with our defined fields:

```
ArcherCalc> date_color incid_date
"Red" : Text
```

Now let's compile the previous in a format suitable for Acher:

```
ArcherCalc> :emit date_color incid_date
IF(DATEDIF([Date of Incident],NOW(),DAY)<30,"Green",IF(DATEDIF([Date of Incident],
NOW(),DAY)<60,"Yellow",IF(DATEDIF([Date of Incident],NOW(),DAY)<90,"Orange",
"Red")))
```

Chapter 2

Installation

TODO

Chapter 3

Using ArcherCalc

3.1 Introduction

3.2 Command Reference

This section documents commands available in archer-calc. These commands can be abbreviated if they don't cause a conflict (e.g. type :l for :load). The commands also respond to tab-completion.

The commands documented here are for interacting and manipulating the language itself, but not part of the language specification.

3.2.1 :browse

Browse will list the names and type signatures of *all* variables, fields, and functions available at the current scope, including built-in functions. The listing is ordered alphabetically.

3.2.2 :emit

This commad will compile an expression into a format suitable for Archer. The result can be pasted into a calculated field formula in Archer.

3.2.3 :help

If :help is given no arguments, then usage information for commands will result.

If :help is given an argument that matches a built-in function, then some basic help information will be returned. Currently this information documents the arguments that the function accepts.

Here is an example:

```
ArcherCalc> :help datedif
Usage: datedif start_date end_date datetime_unit
ArcherCalc>
```

3.2.4 :load

This command will load a file into the current environment. If a function in the new file conflicts with one already defined in the environment it will be replaced with the function in the loaded file.

3.2.5 :quit

This command will exit archer-calc.

3.2.6 :type

The :type command will list the type signature of any expression given.

Here is an example to illustrate the usefulness. Let's check the type of a built-in function:

```
ArcherCalc> :t left
left : Text -> Num -> Text
```

This means that left takes a text value and a number, and returns a text value.

Chapter 4

Syntax Reference

4.1 Comments

```
// Single-line comment

/* Comment
   that spans
   multiple lines. */
```

4.2 Literals

4.2.1 Numbers

Numbers can be entered verbatim, with the exception that negative numbers need parenthesis.

```
// Normal
1
3.14159

// Hexadecimal
0x90
0xAB
0xfe

// Octal
0o72
0O65

// Scientific notation
```

```
1e12
1E-5

// Negative numbers
(-0.3)
(-3.14159)
```

4.2.2 Text

Text is surrounded by double quotes. Double quotes are escaped with a backslash (\).

```
"This is a test."
"a"
"They said \"All good!\""
```

4.2.3 Boolean

```
True
False
```

4.2.4 Dates

If a time is not specified, midnight UTC is assumed.

```
// ISO Style
ISODate "2015-04-02T02:00:00Z"
ISODate "2015-04-02"
ISODate "2015-04-02T04:00:00PDT"

// Archer Style
ArDate "04/02/2015 02:00:00"
ArDate "04/02/2015"
```

4.3 Functions

```
let double x = x*2;

let circ_area r = pi*r^2;

let rect_area w h = w * h;
```

4.4 Control Flow

```
if (temp > 80) then "hot" else "cold";

if (risk >= 4) then
    "red"
else if (risk >= 3) then
    "orange"
else
    "green";
```

4.5 Fields

The format for defining a field is:

```
field "Field name" : Type as Value;
```

"Field name" is the actual name of the field in Archer. The Type is the type of field in Archer (Date, Text, Num, etc.) and is required. The "as Value" is optional, but it lets one evaluate and test the field in ArcherCalc.

Here are some examples:

```
field "Number of Incidents" : Num as 132;
field "Date of Incident" : Date as ISODate "2015-04-05";
field "Incident Status" : Text as "Open";
field "Risk Rating" : Text;
```

4.6 Built-in Functions

4.7 Type Signatures