Cygni 1.0
A Short Reference

He Yanjie

December 31, 2016

Abstract

1 What is Cygni?

Cygni is a script programming language implemented in C#. It is easy to use, has neat grammar and can interacts with C#. It is convenient to wrap C# classes as Cygni class, namely the Cygni libraries are based on the C# class libraries.

Cygni is designed by me. I spent a lot of spare time on it, and I love it very much. I hope you will like it too!

2 Core Language

2.1 Reserved words

- and
- or
- not
- true
- false
- nil
- if
- else
- elif
- while

- for
- in
- def
- lambda
- class
- var
- unpack
- break
- continue
- return

2.2 Reserved Symbols

- Add:+
- Subtract:-
- Multiply:*
- Divide:/
- Integer Divide: //
- Modulo:%
- Power: ˆ
- Concatenate: &
- assign: =
- Equals:==
- Not Equals: !=
- Greater than: >
- Less than: <
- Greater than or Equals: >=
- Less than or Equals: <=
- Goes to: =>
- Parentheses: ( )

- Brackets: [ ]
- Braces: { }
- Colon: :
- Comma: ,

2.3 Identifiers

The first character of identifiers should be underline or letters, the rest can be underlines, letters or numbers. Note that the identifiers should not be the same as the reserved words.

2.4 Comments

Line comment start with #.

2.5 Strings

String should be enclosed by ” or ””. If there is symbol at the start of string, then the escaped characters in the string will be ignored, and the ’ or ” in the string shoule be written twice.

2.6 Types

Variables in Cygni don’t have type. Only the values have.

- integer
- number
- boolean: true or false
- string
- list: lists contain elements from various types.
- dictionary: key-value pairs
- function
- native function: wrapper for C# native functions
- tuple
- range
- struct
- class
- userdata: wrapper for C# native classes.
- nil

2.7 Control Statements

2.7.1 If

'if' statement can have one or more braches.

```
if condition {
        # Do something
} else {
        # Do something
}

if condition1 {
        # Branch 1
} elif condition2 {
        # Branch 2
} else {
        # Branch 3
}
```

2.7.2 For

'for' statement iterates a collection. You may use range syntax to declare a iterable collection. The iterator shouldn't be changed during the iteration.

```
for i = start: end {
        # Do something
}

for i = start: end: step {
        # Do something
}

for item in collection {
        # Do something
}
```

While 'while' statement will keep running unless the condition is false.

```
while condition {
        # Do something
}
```

2.7.3 Break, Continue, Return

break, return exit the loop. continue stays in the loop.

2.8 Range Constructor

```
start: end
start: end: step
```

2.9 List Constructor

```
[item1, item2, ...]
```

2.10 Dictionary Constructor

Note that dictionary only takes values of integer, boolean, string as keys.

```
{key1: value1, key2:  value2, ...}
```

2.11 Function Definition

'def' statement declares a function with a name. 'lambda' statement declares an anonymous function, which contains a statement or a block.

```
def FunctionName (arg1, arg2, ...) {
        # Do something
}

a = lambda(arg1, arg2, ...) => # Expression

a = lambda(arg1, arg2,...) => {
        # Do something
}
```

2.12 Invoke Function

```
f(arg1, arg2, ...)
```

2.13 Tuple Constructor

```
a = tuple(10, 20)
unpack x, y = a
```

2.14 Struct Constructor

```
a = struct('key1', value1, 'key2', value2, ...)
```

2.15 Class Definition

```
class MyClass {
        # body
}

class DerivedClass: MyClass {
        # body
}
```

2.15.1 Reserved Fields

- __init
- __add
- __sub
- __mul
- __div
- __mod
- __pow
- __unp
- __unm
- __cmp
- __eq
- __getItem
- __setItem
- __toStr

3 Basic Library

3.1 Executing

- source(fileName [,encoding])

  description: execute a script file.
- require(fileName [,encoding])

  description: execute a script file and return it as a module.
- import(fileName [,encoding])

  description: execute a script file in the current global scope.

3.2 Console Output and input

- print(args) print arguments in the console, separated by tab.
- printf(content, args) print format string in the console. The arguments can be indexed by {0},{1}... in the string.
- input([content]) write the content in the console and waiting for user to input. The content can be omitted.

3.3 Conversion

- int(a) convert certain value into integer.
- number(a) convert certain value into number.
- str(a) convert certain value into string.
- list(a) convert an iterable object into list.

4 The Math Library

To simplify, all the arguments for the functions in math library only takes number as parameters. If the argument is an integer, it will be converted into number.

- math.sqrt(x)
- math.abs(x)
- math.log(x [,base])
- math.log10(x)
- math.max(args)
- math.min(args)
- math.exp(x)
- math.sign(x)
- math.sin(x)
- math.cos(x)
- math.tan(x)
- math.asin(x)
- math.acos(x)

- math.atan(x)
- math.sinh(x)
- math.cosh(x)
- math.tanh(x)
- math.ceiling(x)
- math.floor(x)
- math.round(x)
- math.truncate(x)