The Drift Script Cheat Sheet

Syntax

The source code file is a file with .ft suffix. Use the # symbol for line comments.

Five basic data types, integer, floating point, string, character, and boolean. Other data types are nil, array, tuple, map, enumeration, function, class, interface, module.

In addition to the five basic types of compiled types, there are function type, user type, any type, and generic type.

Use arbitrary size indentation for block statements, eg 4, 2 etc.

Keyword

Twelve compile-time keywords.

```
def  ret  for  aop  if  ef  nf  new  out  go  use  nil
```

Contains six types of compiler checks.

```
int  char  string  float  bool  any
```

Operator

```
+  -  *  /  %  +=  -=  *=  /=  %=  ->  <-  .  ,  :  =  ;  >  <  >=  <=

&  |  !  !=  ==  {  }  (  )  [  ]  _  \
```

Structure

```
E(Expression):

Number = 0 -> 9 | num.num | num.

Char = '<ASCII character>'

String = "<ANY character>"

Boolean = true | false

True = num > 0 | true | bang false | noteq nil

False = num <= 0 | false | bang true | eq nil

Nil = #nil

Id = a -> z | A -> Z | _

Or = E | E

And = E & E

Equal = E == E | E != E

Compare = E > | >= | < | <= E

Term = E + | - E

Factor = E * | / | % E

New = #new E{id: E..} | {}

Not = -E

Bang = !E

Call = id(E..)

Index = id[E]

Get = id.E

Group = (E)

Assign = id.E = E

A-Assign = id += | -= | *= | /= | %= E

G-Assign = get += | -= | *= | /= | %= E

R-Assign = index += | -= | *= | /= | %= E

Array = [E..]

Tuple = (E..)

Map = {id: E..}


T(Type):

Int = #int

Float = #float

Char = #char

String = #string

Boolean = #bool

Array = []T

Tuple = ()T

Map = {}<T1, T2>

Function = |T..| | -> T

User = id


S(statement):

~ = new interval

Block = same indentation interval

Expression = E

Var = #def id T = E

Enumeration = #def id ~ block <- id..

Function = #def (id.. T.. | id1, id2.. T..[<-T]) id | -> T ~ block

Class = #def id ~ block <- S

Interface = #def id ~ block <- method

Method = \T..\ id | -> T

Return = #ret -> | E

For-Loop = #for E; E; E ~ block

A-Loop = #aop E ~ block

If = #if E ~ block |> ef..

Ef = #ef E ~ block |> nf

Nf = #nf ~ block

Out = #out -> | E

Go = #go -> | E

Use = #use id | <-id


G(Generic):

G-Function = #def <G T.. | G.. T1 | T2..> =Function

G-Inteface = #def id<G..> =Interface

G-Class = #def id<G T.. | G.. T1 | T2..> =Class
```

Modules

Drift supports external modular programming. A single drift source file is a module, which can be imported through the `use` keyword and uses its internally defined symbols.

Modules under the standard library need to specify the environment variable `FTPATH` to the system, pointing to the location of the standard library folder.

Exceptions

Drift uses the built-in standard library to declare exceptions for statements.

This is a simple way to handle exceptions, please refer to the link below for sample code.

Language Label

Script Language, Byte code, Interpreter, TDOP parser, Indented grammar, Minimalist grammar, Classes, Generics, Duck method, Module support. GC automatic garbage collection, Toy language.

Example

The relevant sample code can be obtained in the playground and run online. https://play.drift-lang.fun/

End