A short Itroduction of Skiff Script

What's Skiff Script

Skiff Script is a kind of programming language actived by Bison and Flex.

1)The construction of SkiffScript

There are two parts of a SkiffScript program.

1. Toplevel

Toplevel is a set of statements,the SkiffScript interpreter will execute the program from toplevel.So there's no such thing called "main()" in Skiffscript.

2.Function Definition

Function definition defined a function that can be called from other parts of the program.When execute the program,interpreter will ignore the statements in the function definition.You can place the function definition before calling it or after calling it.

2) The Grammar

The code of the resource file is UTF-8.

2.2 Key Words

These words are used as the key words of SkiffScript,you cannot use it as the name of variable or function.

```
var integer real string boolean byte else elif
while do goto break continue return for if
function
```

Notice:"goto" is only used as the reserved word,not use in SkiffScript.

2.4 Comment

Type "//" to make a context

2.5 Identifier

The interpreter will think of it as the name of variables and functions.

- First Character: A~Z a~z or underline
- Other Character:Latin letters,numbers or underlines

Or this:

```
[A-Za-z_][A-Za-z_0-9]*
```

2.6 Literal

Boolean literal :true or false

Integer literal :0 or a number start with 1~9

Or match this:

([1－9][0－9]*)|"0"

Real literal : 0 ~ 9(repeat more than once) + '.' + 1 ~

9(repeat more than once).

cannot use .7 instead of 0.7

Or match this:

```
[0-9]+\.[0-9]+
```

String literal :

Start with a ' " ',and end with a ' " ' like "Nathan"

There're some special characters:

\n : Linebreak

\\ : a '\'

\a : beep

\t : Tab

\" a ' " '

e.g. "Linebreak\n And let's have a beep\a"

2.7 Operator

These signs are used as operator

```
+ - * / % & | ! && || > >= < <= == != ( )
```

2.8 Punctuator

These signs are used as punctuator

```
() {} ; : ,
```

3) Data Type

There're types in Skiffscript

3.1 Logical type

```
var bool_type : boolean;
```

To create a boolean type.

```
bool_type = true;

//a boolean can only be true or false
```

3.2 Integer type

```
var int_type : integer;
```

To create a integer type.

```
int_type = 22;

int_type = 0x0d;//Don't support hex now!
```

It's the same as int in C environment.

3.3 Real type

```
var real_type : real;
```

To create a real type.

```
real_type = 7.0;

real_type = .5;//Not allowed!
```

It's the same as double in C environment.

3.4 Byte type

```
var byte_type : byte;
```

```
byte_type = 128;

byte_type = 300;//The range of byte is between
//0 and 255 so donnot do this
```

Byte type is the same as unsigned char in C environment

3.5 String type

```
var string_type : string;
```

To make a string type.

```
string_type = "Welcome ";

string_type = string_type + "to SkiffScript
world!";

//The value of string_type is "Welcome to
//SkiffScript world!"
```

It's immutable.

String type is the same as char* in C environment.

4)Expression

4.1 Priority of operator.

Highest

```
-(minus) !

* / %

> >= < <=

== !=
```

```
& |
&& ||
=
```

Lowest

4.2 Expression Stetements

The statements end with a ';'

```
print("Hello, World!\n");//Function call expression


n = 9;//assign expression


5;//meaningless
```

4.3 If Statement

```
if (condition1) {
    // execute when condition1 is true
} elif (condition2) {
    //execute when condition 2 is true
} else {
    //Execute when all are false
}
```

You must write "{}" even if there's only one statement in a

if statement.

4.4 While Statement

```
while(condition) {
    //repeat if condition is true
}
do {
    //at least repeat once
}while (condition)
```

You must write "{}" even if there's only one statement in a while statement.

4.5 For Statement

```
for (statement1 ; statement2; statement3) {
    //Repeat if statement2 is true
}
```

You must write "{}" even if there's only one statement in a for statement

4.6 continue and break

```
while (condition) {
    ...
    if (...) {
        continue;//start a new turn of repeat
```

```
    } elif (...) {
        break;//Halt the loop
    }
}
```

4.7 return

```
return expression;
//it will jump out of a function
```

5) Function

You can define a function like this

```
function func_name (parameter) {
    //statements here
}
```

e.g.

```
function add (a,b) {
    return a + b;
}
```

5.1 Local Variable

Variables that define in a function or a block are called local function,you cannot use it in outer structure.

6) Copyright

Special Thanks:

GNU Project Team. Ubuntu team.