# Control Structures and Functions

1. Conditional Expressions

an if/else has a value, namely the value of the expression that follows the if or else. For example, if (x > 0) 1 else -1

val s = if (x > 0) 1 else -1

In Scala, every expression has a type.

such as

if (x > 0) "positive" else -1

is the common supertype of both branches.

If the else part is omitted, for example in

if (x > 0) 1

This is finessed by introducing a class Unit that has one value, written as ().

equivalent to

if (x > 0) 1 else ()

1. Statement Termination

If you want to continue a long statement over two lines, make sure that the first

line ends in a symbol that cannot be the end of a statement. An operator is often

a good choice:

s = s0 + (v - v0) \* t + // The + tells the parser that this is not the end

0.5 \* (a - a0) \* t \* t

1. Block Expressions and Assignments

In Scala, a { } block contains a sequence of expressions, and the result is also an

expression. The value of the block is the value of the last expression.

**This feature can be useful if the initialization of a val takes more than one step.**

For example,

val distance = { val dx = x - x0; val dy = y - y0; **sqrt(dx \* dx + dy \* dy)** }

A block that ends with an assignment, such as

{ r = r \* n; n -= 1 }

has a Unit value.

Since assignments have Unit value, don’t chain them together.

x = y = 1 // No

The value of y = 1 is ()

1. Input and Output

There is also a printf function with a C-style format string:

printf("Hello, %s! You are %d years old.%n", name, age)

Or better, use string interpolation

print(f"Hello, $name! In six months, you'll be ${age + 0.5}%7.2f years old.%n")

For example, raw"\n is a newline" starts with a backslash and the

letter n, not a newline character

1. Loops

The difference of method and function

http://blog.csdn.net/u013063153/article/details/53433724