Atitit 编程语言作用域

### **Scoping**

即作用域, 作用域是名称(比如变量的声明)和其实体(entity, 比如变量的定义)的绑定规则。作用域约束了实体的作用范围，保证程序是无歧义的。

### **Expression scope**

实体仅在表达式内可用。

// c

({ int x = f(); x \* x; })

临时变量 *x* 接受函数的返回值并平方，这样避免两次调用函数 *f* .

### **Block scope**

通常编程语言都会使用花括号 {} 来将代码包裹成块(block), 在block内声明的实体，仅在block内有效。

// golang

{

var a int

b := a

}

b := a // compile error

// golang

{

var a int

b := a

}

b := 1 // ok

### **Function scope**

在函数内声明的实体，仅在函数内有效。

def square(n):

return n \* n

def sum\_of\_squares(n):

total = 0

i = 0

while i <= n:

total += square(i)

i += 1

return total

为了不和 *block scope* 混淆，这里用 *python* 的例子。

### **File scope**

在代码文件内声明的全局变量，仅在当前文件内有效。

### **Module scope**

在某些现代语言中，一个实体可以在一个模块内的各个文件内有效，比如 *golang* . 部分语言，一个文件就是一个独立的module，此时，也属于 *file scope* .

### **Global scope**