

## MasC – idea documentation

---

Masc is to be a way to group together the most basic algorithms to be in a transitional state, to be converted to different target languages and systems. It is very basic and is its infancy. There are a lot of ideas and different actions I would like this programming language to take. At this point it is more of an idea. The language is very simple, and can be learned in a matter of twenty minutes.

### Instruction for MasC

creating your own instructions is like a function in C. However it works like a built-in instruction. An example on how to create is the following:

```
object Point {  
  
    section var {  
  
        define word x = 0  
        define word y = 0  
  
    }  
  
    method setpos(&px,&py) {  
  
        mov x, px  
        mov y, py  
  
    }  
  
    method setSize {  
  
        setpos 1,2  
  
    }  
  
}
```

the instruction would be called from within Point's scope:

the – operator would prefix the instruction template so it would be recognized and be differentiated from standard instructions.

If we had another object say: level with a instance of object Point it would be referenced a bit differently.

```
object level {  
  
  section var {  
  
    define type Point ptr  
  
  }  
  
  method setCords {  
  
    ptr->setpos 60,60  
  
  }  
  
}
```

### Instruction Expressions

a instruction expression is a instruction that instead of taking a standard operand for a second argument, can take a expression. A example:

```
add x, $[ y+source_x-(3*4)]
```

this will add to x the result of the expression.

This will work for many instructions, and template's for instructions will not have a limit on arguments.