类型定义语句，主要是结构体（类、联合体）的定义。

它们都是以关键字struct / class / union开始的。

我在scf框架里没有支持作用域运算符::
，所以结构体的定义都是全局的，不能在结构体里
定义其他结构体类型（但可以包含其他结构体变量）。

```
struct S0 {

struct S0* next;

struct Data {

int x;

int y;

} d;

};
```

像上面这样的结构体定义，我觉得分析起来太麻烦了[捂脸]

我认为，还是把C++的S0::Data这个类型放到全局里定义比较好。

```
struct S0_Data {

int x;

int y;

};

struct S0 {

struct S0* next;

S0_Data d;
```

};

如果是union的话，我倒是在scf框架里支持了结构体里的匿名联合体。

我设计的结构体里是可以添加成员函数的，这点类似C++。

结构体的语法分析并不复杂，它是以struct关键字加上结构体名字
，再加上{}扩起来的顺序块构成的，最后以分号结尾。

我没有支持这样的结构体定义：struct S { int x; int y; } s0;

我的打算是类型定义和变量声明尽量分开（union除外）。

之所以对union做了跟C语言
一样的处理，是不想让下面的代码也要单独定义一个全局的union类型：

struct S {

struct S* next;

union {

int i;

float f;

double d;

void* p;

};

};

init_module()函数

类型定义模块的节点有6个：

1，class或struct关键字，

这两个在scf框架里没有区别，因为我没有支持public / private的权限控制。

2，类名的标志符，它是个字母或下划线开头的字符串。

3，左中括号，表示类成员（变量或函数）的定义开始，

4，右中括号，表示类成员（变量或函数）的定义结束。

5，分号，表示定义结束。

6，end，类定义结束之后要运行的代码，例如计算所占的字节数。

语法编辑

语法的编辑也很简单，只要把这些节点按照跟源代码一样的顺序连接起来就行。

除了union类型的成员变量之外，其他成员变量都是以类型名字开始的，所以引用了类型模块type的入口节点。

说说怎么计算结构体的字节数：

计算结构体的大小

CPU读写内存时要尽量对齐：

1，如果变量是1个字节可以从任何地址开始，

2，变量是2字节就从能被2整除的地址开始，

3，变量是4字节就从被4整除的地址开始，

4，再大的变量就从被8整除的地址开始。

如果变量的字节数不足，就对齐到最近的2的幂。

例如，3字节的变量也是以4对齐的，5字节的也是以8对齐的。

对齐之后就是变量在结构体里的偏移量
，然后加上变量的大小就可以计算出结构体的大小。

具体的语法分析函数，见下面的几张图，已加注释。