

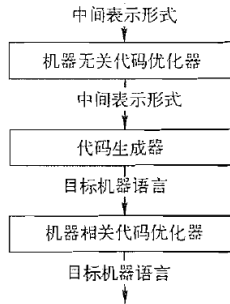
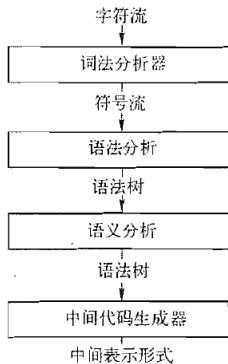
# 符号表

魏恒峰

hfwei@nju.edu.cn

2022 年 12 月 05 日





## 符号表

## Definition (符号表 (Symbol Table))

**符号表**是用于保存**各种信息**的**数据结构**。

## Definition (符号表 (Symbol Table))

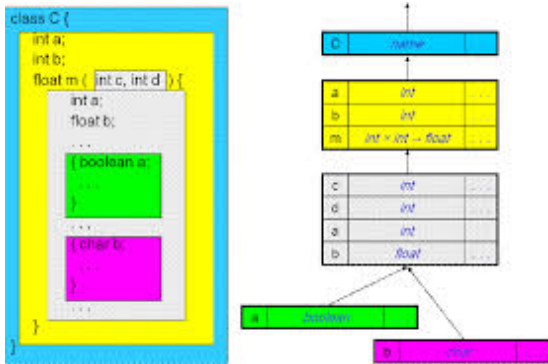
**符号表**是用于保存**各种信息**的**数据结构**。

Name	Type	Size	Dimension	Line of Declaration	Line of Usage	Address	...
<i>count</i>	int	4	0	...	...	...	...
<i>str</i>	char	5	1	...	...	...	...

“领域特定语言” (DSL) 通常只有**单作用域** (全局作用域)

```
host=antlr.org  
port=80  
webmaster=parrt@antlr.org
```

“通用程序设计语言” (GPL) 通常需要**嵌套作用域**

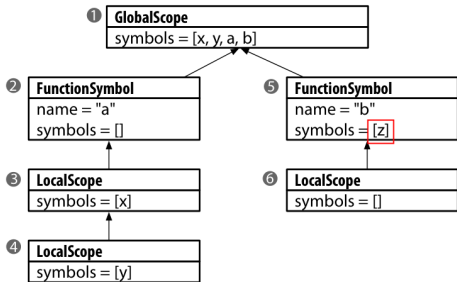


```
1 int x;  
  int y;  
2 void a()  
3 {  
    int x;  
    x = 1;  
    y = 2;  
4    { int y = x; }  
  }  
5 void b(int z)  
6 { }
```

```

1  int x;
   int y;
2  void a()
3  {
       int x;
       x = 1;
       y = 2;
4  { int y = x; }
   }
5  void b(int z)
6  { }

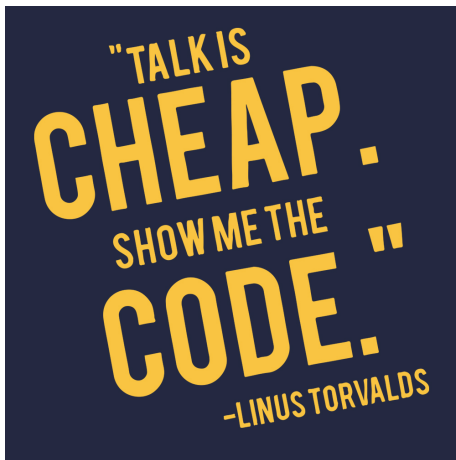
```





```
public interface Scope {  
    public String getScopeName();           // 有名称吗?  
    public Scope getEnclosingScope();       // 有外部作用域吗?  
    public void define(Symbol sym);         // 在作用域中定义符号  
    public Symbol resolve(String name);     // 根据名称查找  
}
```

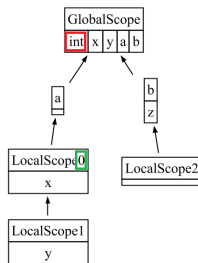
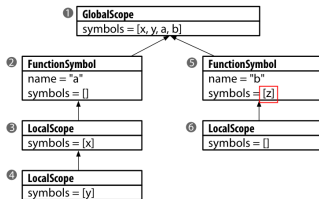
全局作用域、函数/方法作用域、局部作用域

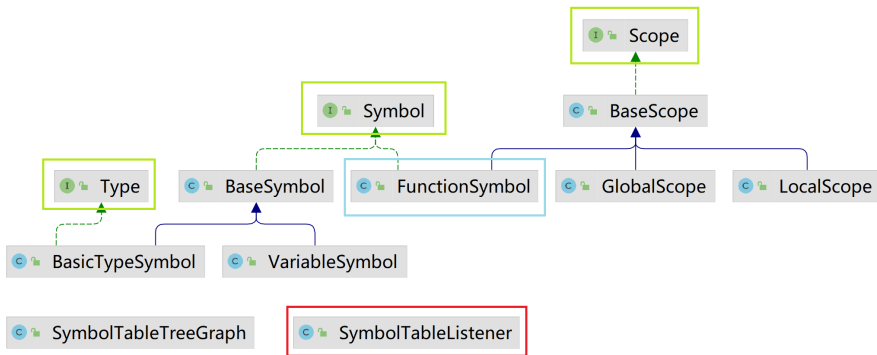


```

1 int x;
  int y;
2 void a()
3 {
    int x;
    x = 1;
    y = 2;
4     { int y = x; }
5 }
6 void b(int z)
7 { }



```



































Scope	
(m)	<code>setName(String)</code> <code>void</code>
(m)	<code>getSymbols()</code> <code>Map&lt;String, Symbol&gt;</code>
(m)	<code>getEnclosingScope()</code> <code>Scope</code>
(m)	<code>define(Symbol)</code> <code>void</code>
(m)	<code>getName()</code> <code>String</code>
(m)	<code>resolve(String)</code> <code>Symbol</code>

  SymbolTableListener

SymbolTableListener

SymbolTableListener		
f	currentScope	Scope
f	globalScope	GlobalScope
f	graph	SymbolTableTreeGraph
f	localScopeCounter	int

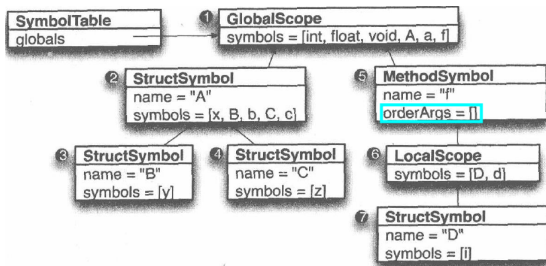
## SymbolTableListener

 	currentScope	Scope
 	globalScope	GlobalScope
 	graph	SymbolTableTreeGraph
 	localScopeCounter	int
 	enterBlock(BlockContext)	void
 	enterFunctionDecl(FunctionDeclContext)	void
 	enterProg(ProgContext)	void
 	exitBlock(BlockContext)	void
 	exitFormalParameter(FormalParameterContext)	void
 	exitFunctionDecl(FunctionDeclContext)	void
 	exitId(IdContext)	void
 	exitProg(ProgContext)	void
 	exitVarDecl(VarDeclContext)	void
 	getGraph()	SymbolTableTreeGraph



## struct: 类型作用域

```
❶  
❷ struct A {  
❸     int x;  
❹     struct B { int y; };  
❺     B b;  
❻ struct C {int z; };  
❼     C c;  
❽ };  
❾ A a;  
  
❶ void f()  
❷ {  
❸     struct D {  
❹         int i;  
❺     };  
❻     D d;  
❼     d.i = a.b.y;  
❽ }
```



*d.i*      *a.b.y*

symtab @ antlr by parrt

symtab @ cs652 by parrt

Thank  
You!



Office 926

hfwei@nju.edu.cn