

Princeton University

COS 217: Introduction to Programming Systems

C Variable Declarations and Definitions

Variable **declaration** is a statement that informs the compiler of the name, type, scope, linkage, and duration of the variable. A variable **definition** is a declaration that causes the compiler to allocate memory.

Scope (compile-time concept)

File: The variable is accessible within the file in which it is declared, from the point of declaration to the end of the file.

Block: The variable is accessible within the block in which it is declared, from the point of declaration to the end of the block.

Linkage (link-time concept)

External: The variable is accessible from multiple files.

Internal: The variable is accessible from only the file in which it is declared.

Duration (run-time concept)

Temporary: The variable exists only during the execution of the function or block in which it is declared. Physically, the variable's value is stored in the runtime Stack.

Process: The variable exists throughout the entire process. Physically, the variable's value is stored in the Data Section (if the programmer specifies an initial value) or the BSS Section (if the programmer does not specify an initial value). The variable's value is initialized at program startup. If in the BSS section, its initial value is 0.

C Code	Decl/Def	Scope	Linkage	Duration	Location
int a = 5;	definition	file	external	process	
int b ;	definition*	file	external	process	
extern int c = 5;	definition	file	external	process	
extern int d ;	declaration	file	external	process	???
static int e = 5;	definition	file	internal	process	
static int f ;	definition	file	internal	process	
void fun(int g) {	definition	block	internal	temporary	
int h = 5;	definition	block	internal	temporary	
int i ;	definition	block	internal	temporary	
 extern int j = 5;	ILLEGAL				
 extern int k;	declaration	block	???	process	???
static int l = 5;	definition	block	internal	process	Data
static int m ;	definition	block	internal	process	BSS
...					
}					

* Special rule: If a definition of **b** appears in another .c file, then this becomes a declaration.

Examples of Global Variable Declarations and Definitions

Suppose a program consists of file1.c and file2.c (only). Consider these combinations of global variable declarations and definitions:

	file1.c	file2.c	Result
	Reasonable combinations:		
1	static int i = 5;	static int i = 5;	static def / static def => OK
2	static int i = 5;	static int i;	static def / static def => OK
3	static int i;	static int i;	static def / static def => OK
4	int i = 5;	extern int i;	def / decl => OK
5	int i;	extern int i;	def / decl => OK
	Less reasonable combinations:		
6	int i = 5;	int i;	def / decl => OK (by special rule)
7	int i;	int i;	def / decl => OK (by special rule)
8	int i = 5;	static int i = 5;	def / static def => OK
9	int i = 5;	static int i;	def / static def => OK
10	int i;	static int i = 5;	def / static def => OK
11	int i;	static int i;	def / static def => OK
	Erroneous combinations:		
12	int i = 5;	int i = 5;	def / def => error
13	extern int i;	extern int i;	decl / decl => error
14	extern int i;	static int i = 5;	decl / static def => error
15	extern int i;	static int i;	decl / static def => error

Copyright © 2010 by Robert M. Dondero, Jr.