**What is a storage class in C?**

In C language, each variable has a **storage class** which decides scope, visibility and lifetime of that variable. The following storage classes are most oftenly used in C programming,

1. **Automatic variables**
2. **External variables**
3. **Static variables**
4. **Register variables**

**Automatic variables**

A variable declared inside a function without any storage class specification, is by default an **automatic variable**. They are created when a function is called and are destroyed **automatically** when the function exits. Automatic variables can also be called local variables because they are local to a function. By default they are assigned **garbage value** by the compiler.

1. **void** main()
2. {
3. **int** detail;
4. **or**
5. **auto** **int** detail; //Both are same
6. }

**External or Global variable**

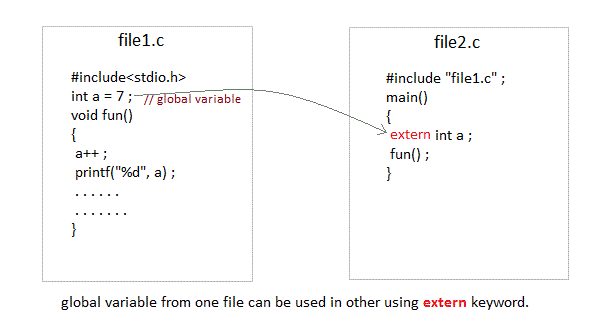
A variable that is declared outside any function is a **Global variable**. **Global** variables remain available throughout the entire program. One important thing to remember about global variable is that their values can be changed by any function in the program.

1. **int** number;
2. **void** main()
3. {
4. number=10;
5. }
6. fun1()
7. {
8. number=20;
9. }
10. fun2()
11. {
12. number=30;
13. }

Here the global variable **number** is available to all three functions.

**extern keyword**

The **extern** keyword is used before a variable to inform the compiler that this variable is declared somewhere else. The **extern** declaration does not allocate storage for variables.



**Problem when extern is not used**

1. main()
2. {
3. a = 10; //Error:cannot find variable a
4. printf("%d",a);
5. }

**Example Using extern in same file**

1. main()
2. {
3. **extern** **int** x; //Tells compiler that it is defined somewhere else
4. x = 10;
5. printf("%d",x);
6. }
8. **int** x; //Global variable x

**Static variables**

A **static** variable tells the compiler to persist the variable until the end of program. Instead of creating and destroying a variable every time when it comes into and goes out of scope, **static** is initialized only once and remains into existence till the end of program. A static variable can either be internal or external depending upon the place of declaraction. Scope of **internal static** variable remains inside the function in which it is defined. **External static** variables remain restricted to scope of file in each they are declared.

They are assigned **0 (zero)** as default value by the compiler.

1. **void** test(); //Function declaration (discussed in next topic)
3. main()
4. {
5. test();
6. test();
7. test();
8. }
9. **void** test()
10. {
11. **static** **int** a = 0; //Static variable
12. a = a+1;
13. printf("%d\t",a);
14. }
15. output :
16. 1 2 3

**Register variable**

**Register** variable inform the compiler to store the variable in register instead of memory. **Register** variable has faster access than normal variable. Frequently used variables are kept in register. Only few variables can be placed inside register.

**NOTE :**We can never get the address of such variables.

**Syntax :**

1. **register** **int** number;

**Storage class:** It describes the persistence of the objects accessed by the scope, linkage, and visibility of the variable.

"storage class" on its own doesn't mean much, it’s just the term used for the group of five keywords known as "storage class specifiers": **auto**, **register**, **static**, **extern**, thread\_local\*. When something is described as having “storage class xyz”, all it means is that “xyz” was the keyword used when this something was declared.

These five keywords specify two unrelated properties of variables in C: “storage duration” and “linkage”. Plus, **register** has a special semantic meaning: it prohibits the address-of operator.

(Note: "storage" is the fancy C standarese term that means "memory" to most people)

There are four kinds of storage duration:

* **automatic**: memory allocated when control flow enters the block where the variable is declared and deallocated when control flow leaves that block by any means.
* **static**: memory allocated at the start of the program and deallocated at the end of the program
* **thread**: memory allocated at the start of a thread and deallocated at the end of the thread
* **allocated**: memory allocated and deallocated manually, using malloc, calloc, aligned\_alloc, realloc, and free

There are three kinds of linkage:

* **no linkage**: the variable is visible only to the block it's in
* **internal**: the variable is visible to the file, but not other files
* **external**: the variable is visible to the entire program

The mapping of storage class specifies to storage durations and linkages is complicated and is popular with C trivia. I'll try to summarize the mapping just to the storage durations:

* function parameters:
  + default (no storage class specifier), **register** -> automatic.
  + No other storage class specifiers permitted.
* local variables:
  + default (no storage class specifier), **auto**, and **register** -> automatic
  + **static**, **extern** -> static
  + thread\_local -> thread
* global variables:
  + default (no storage class specifier), **static**, **extern** -> static
  + thread\_local -> thread

\*the actual keyword is technically \_Thread\_local. thread\_local is a macro that expands to that keyword.

**Storage class** in C decides the part of storage to allocate memory for a variable, it also determines the scope of a variable. All variables defined in a C program get some physical location in memory where variable's value is stored. Memory and CPU registers are types of memory locations where a variable's value can be stored. The storage class of a variable in C determines the life time of the variable if this is 'global' or 'local'. Along with the life time of a variable, storage class also determines variable's storage location (memory or registers), the scope (visibility level) of the variable, and the initial value of the variable.

**Types of Storage Classes** :

1. Automatic Storage Class
2. Register Storage Class
3. Static Storage Class
4. External Storage Class

Every variable or a function has a defined scope (visibility) and life-time and also memory location . the storage class defines the scope , lifetime and memory location of a variable or function.

normally we tend to ignore the storage class and initialise the variables without mentioning the storage class . it is then automatically treated as “auto” storage class .

variable initialisation - auto int n; or register int n;

here auto and register are storage classes .

**AUTO STORAGE CLASS** -

Storage: CPU Memory

Scope : Local or Block Scope

life : Exists as long as Control remains in the block

**REGISTER STORAGE CLASS -**

Storage place: Register memory

Scope: local

Life: Within the function only.

**STATIC REGISTER CLASS -**

Storage place: CPU memory  
  
Scope: local

Life: Retains the value of the variable between different function calls.

**EXTERN STORAGE CLASS -**

Storage place: CPU memory  
  
Scope: Global

Life: Till the end of the main program. Variable definition might be anywhere in the C program.

1.1k views · [View 3 Upvoters](https://www.quora.com/What-is-a-storage-class-in-C)

[Aditi Atray](https://www.quora.com/profile/Aditi-Atray)

[Aditi Atray](https://www.quora.com/profile/Aditi-Atray), B.tech Computer Science, Vidya College of Engineering (2019)

[Answered Jul 19, 2017](https://www.quora.com/What-is-a-storage-class-in-C/answer/Aditi-Atray)

In C language, each variable has a storage class which decides the following things:

* scope i.e in which all functions, the value of the variable would be available.
* default initial value i.e if we do not explicitly initialize that variable, what will be its default initial value.
* lifetime of that variable i.e for how long will that variable exist.
  1. **Automatic variables**
  2. **External variables**
  3. **Static variables**
  4. **Register variables**
  5. And i would like u to recommend [Hackr.io - Find & share the best online programming courses & tutorials](http://hackr.io/) for learning C language.

1.1k views · [View 1 Upvoter](https://www.quora.com/What-is-a-storage-class-in-C)

[](https://www.quora.com/profile/Smit-Prakash)

[Smit Prakash](https://www.quora.com/profile/Smit-Prakash), Asociate Software Engineer at IBM (2016-present)

[Answered Mar 7, 2019](https://www.quora.com/What-is-a-storage-class-in-C/answer/Smit-Prakash) · Author has **1k** answers and **930.4k** answer views

To fully define a variable one needs to mention not only its ‘type’ but also its ‘storage class’. In other words, not only do all variables have a data type, they also have a ‘storage class’.

storage classes have defaults. If we don’t specify the storage class of a variable in its declaration, the compiler will assume a storage class depending on the context in which the variable is used. Thus, variables have certain default storage classes.

From C compiler’s point of view, a variable name identifies some physical location within the computer where the string of bits representing the variable’s value is stored. There are basically two kinds of locations in a computer where such a value may be kept— Memory and CPU registers. It is the variable’s storage class that determines in which of these two locations the value is stored.

150 views

[Payal Saxena](https://www.quora.com/profile/Payal-Saxena-38)

[Payal Saxena](https://www.quora.com/profile/Payal-Saxena-38), Manager

[Answered May 24, 2018](https://www.quora.com/What-is-a-storage-class-in-C/answer/Payal-Saxena-38) · Author has **199** answers and **46.2k** answer views

**Storage classes in C**

In [C language](http://www.learnvern.com/unit/c-storage-class), each variable has a storage class which decides the following things:

* **scope** i.e where the value of the variable would be available inside a program.
* **default initial value** i.e if we do not explicitly initialize that variable, what will be its default initial value.
* **lifetime** of that variable i.e for how long will that variable exist.

364 views · [View 1 Upvoter](https://www.quora.com/What-is-a-storage-class-in-C)

[](https://www.quora.com/profile/Himanshu-Chavhan-1)

[Himanshu Chavhan](https://www.quora.com/profile/Himanshu-Chavhan-1), B.E from Yeshwantrao Chavan College of Engineering (2020)

[Answered Jul 24, 2017](https://www.quora.com/What-is-a-storage-class-in-C/answer/Himanshu-Chavhan-1) · Author has **71** answers and **565.5k** answer views

Originally Answered: [What are the storage classes in C?](https://www.quora.com/What-are-the-storage-classes-in-C?no_redirect=1)

Every C variable has a storage class and a scope. The storage class determines the part of memory where storage is allocated for an object and how long the storage allocation continues to exist. It also determines the scope which specifies the part of the program over which a variable name is visible, i.e. the variable is accessible by name. The are four storage classes in C are automatic, register, external, and static.

600 views · [View 2 Upvoters](https://www.quora.com/What-is-a-storage-class-in-C)

[](https://www.quora.com/profile/Maya-Thakkar-1)

[Maya Thakkar](https://www.quora.com/profile/Maya-Thakkar-1), Product Development Manager at LearnVern (2017-present)

[Answered Mar 1, 2018](https://www.quora.com/What-is-a-storage-class-in-C/answer/Maya-Thakkar-1) · Author has **499** answers and **219.9k** answer views

Storage Class In C:

* Accessibility of a declared in a program.
* Visibility of a variable to a function or area.
* As variable declared in main function is visible in main function only but it can be changed.

If you want to more information about storage class in c, so please visit this site.It’s a describe in detail with example.

Every **C** variable has a **storage class** and a scope. It also determines the scope which specifies the part of the program over which a variable name is visible, i.e. the variable is accessible by name. Ther are four **storage classes in C:** automatic, register, external, and static

Every **C** variable has a **storage class** and a scope. It also determines the scope which specifies the part of the program over which a variable name is visible, i.e. the variable is accessible by name. The are four **storage classes in C** are automatic, register, external, and static.

**Storage Classes**. Every **C** variable has a**storage class** and a scope. It also determines the scope which specifies the part of the program over which a variable name is visible, i.e. the variable is accessible by name. The are four **storage classes in C** are automatic, register, external, and static.

To Know more with free videos visit Learnvern.

Every **C** variable has a **storage class** and a scope. ... It also determines the scope which specifies the part of the program over which a variable name is visible, i.e. the variable is accessible by name. The are four **storage classes in C** are automatic, register, external, and static.

To know more with videos you can learn it at Learnvern.

A storage class defines the scope (visibility) and life-time of variables and/or functions within a C Program. They precede the type that they modify. We have four different storage classes in a C program −

* auto
* register
* static
* ex-tern

Author has **231** answers and **62.6k** answer views

Every **C** variable has a **storage class** and a scope. ... It also determines the scope which specifies the part of the program over which a variable name is visible, i.e. the variable is accessible by name. The are four **storage classes in C** are automatic, register, external, and static