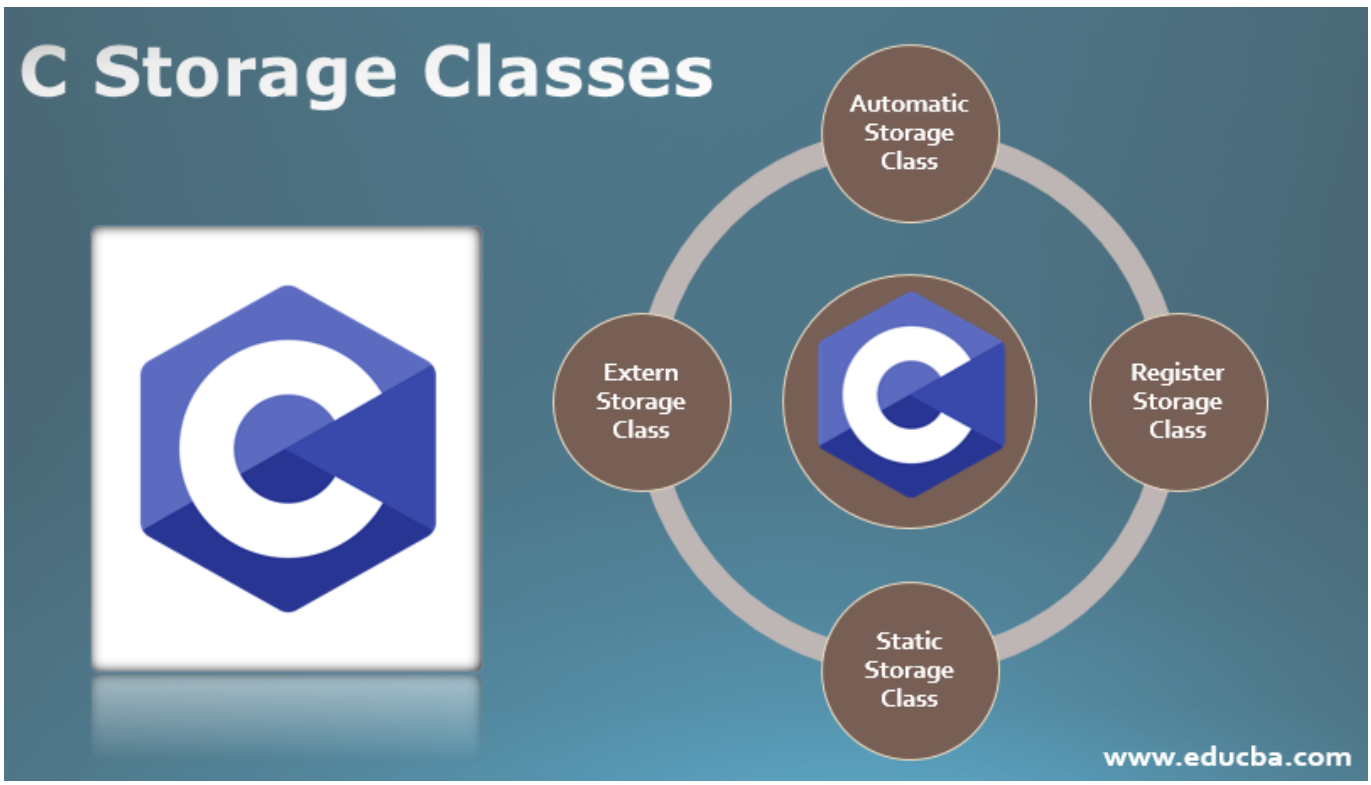
EDUCBA
(https://www.educba.com/software-development/)

← (https://www.educba.com/encapsulation-in-c/)

→ (https://www.educba.com/static-keyword-in-c/)



Introduction to C Storage Classes

Every value or number needs to be stored in some place for later usage, right? This can using variables in C. Variables are storage areas used in our programs. Each variable will be of a specific type like integer, character, and a specific size and layout, depending on their type. Each

(https://www.educba.com/software-development/)

mentioned along with the variable.

Storage Specifier	Storage	Initial value	Scope	Life
auto	Stack	Garbage	Within block	End of block
extern	Data segment	Zero	global Multiple files	Till end of program
static	Data segment	Zero	Within block	Till end of program
register	CPU Register	Garbage	Within block	End of block

Types of Storage Classes in C

Types of Storage Classes in C are as follows.

1. Automatic Storage Class

All variables declared within a function or block will be stored in an auto specifier by default, even if it is not explicitly defined. The specifier for this storage class is 'auto'. The scope or visibility of the variables in the automatic storage class is local to the block or function it is defined. The variable will be destroyed once we come out of the function or the block.

This can be explained better with an example. Consider the example given below:

Code:

(https://www.educba.com/software-development/)

```
printf("%d\n", i);
}
printf("%d\n", i);
}
```

Output:

Here, a variable I of type integer is declared first with value 2 assigned to it. Next, inside a loop or block again, variable I of the same integer type is declared with value 4 assigned to it. If the storage specifier is not mentioned, by default, it will be taken as auto. The first printf statement mentioned inside the block will print 4 on printing the value of I. Whereas in the second printf statement, which is mentioned outside the block, will print the value of I as 2, the value which is mentioned outside the block. It is better to initial some value to auto variables because there are chances of getting any garbage value sometimes if initialization is not done. This example gives a clear picture of auto variables and about local scope.

2. Register Storage Class

The variables stored in the register storage class will also have local scope, which mean accessible or visible only in the block in which it is declared. This storage is similar to auto, but the main difference is that auto variables are stored in memory, whereas the register variables

example, variables which are used for counters or similar usage types are stored using a register specifier.

3. Static Storage Class

Variable, may it be global or local, are stored using static specifier in static storage class the variable needs to be declared once, and the value needs to be retained. When a variable is declared as static, the value will be saved or retained between the function calls. Permanent

(https://www.educba.com/software-development/)

permanent storage is created, and it is declared only once. But even though it is global, these variables are only visible within the file in which it is defined.

Static variables can be clearly pictured using the below example:

Code:

```
#include<stdio.h>
int samplefunc() {
static int a = 0;
a = a+2;
return a;
}
int main() {
int result1 = samplefunc();
int result2 = samplefunc();
printf("%d\n", result1);
printf("%d\n", result2);
}
```

Output:

```
2
4
```

(https://www.educba.com/software-development/)

value and continues with the operation, making the final result as 4. This is the main usage and advantage of static variables.

4. Extern Storage Class

Variable declared as extern depicts that the variable is defined elsewhere in another program. These extern variables are used when we want any variable or function defined in one program to be used in another file too. The variables with the extern specifier are stored in the extern storage class. When the variable is declared as an extern is a program, it specifies the external linkage, and hence it is not defined or initialized again. Storage is allocated only one and also initialized only once. If extern variables are initialized again with another value in the external program, we will get an error stating 'Redefinition of the variable'.

Extern variables are explained using the below example:

Code:

Prg1.c

```
int count;
int main() {
count = 10;
}
```

Prg2.c

```
extern int count;
```

(https://www.educba.com/software-development/)

Output:

Here, the integer variable count is declared in the first C program (Prg1.c), and inside the main function, it is initialized to value 10. In the second C program, the same count variable is declared using an extern specifier, which specifies that there is external linkage and the value is fetched from the storage, and the value 10 is given to the value count when we print it in the second program. This is the usage of extern variables. Thus, depending on the different purpose, each storage classes are used (https://www.educba.com/storage-class-in-c-plus-plus/) for appropriate variables, and it is declared with the corresponding specifiers.

Recommended Articles

This is a guide to the C Storage Classes. Here we discuss the basic concept, four different types of Storage Classes in C, respectively. You may also look at the following articles to learn more –

1. ES6 vs ES5 (https://www.educba.com/es6-vs-es5/)
2. C++ vs Visual C++ (https://www.educba.com/c-plus-plus-vs-visual-c-plus-plus/)
3. C vs C++ Performance (https://www.educba.com/c-vs-c-plus-plus-performance/)
4. C vs C++ | Top Differences (https://www.educba.com/c-vs-c-plus-plus/)

(https://www.educba.com/software-development/)

Android (https://play.google.com/store/apps/details?id=com.educba.www)

Resources

Free Courses (https://www.educba.com/software-development/free-courses/?source=footer)

Java Tutorials (https://www.educba.com/software-development/software-development-tutorials/java-tutorial/?source=footer)

Python Tutorials (https://www.educba.com/software-development/software-development-tutorials/python-tutorial/?source=footer)

All Tutorials (https://www.educba.com/software-development/software-development-tutorials/?source=footer)

Certification Courses

All Courses (https://www.educba.com/software-development/courses/?source=footer)

Software Development Course - All in One Bundle (https://www.educba.com/software-development/courses/software-development-course/?source=footer)

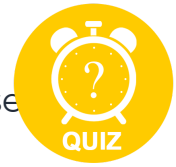
Become a Python Developer (https://www.educba.com/software-development/courses/python-certification-course/?source=footer)

Java Course (https://www.educba.com/software-development/courses/java-course/?source=footer)

Become a Selenium Automation Tester (https://www.educba.com/software-development/courses/selenium-training-certification/?source=footer)

Become an IoT Developer (https://www.educba.com/software-development/courses/iot-course/?source=footer)

ASP.NET Course (https://www.educba.com/software-development/course net-course/?source=footer)



VB.NET Course (https://www.educba.com/software-development/courses/vb-net-course/?source=footer)

(https://www.educba.com/software-development/)