C-style strings

Matt Warner

1 Overview

A C string (also known as a *null-terminated string*) is usually declared as an array of `char`. However, an array of `char` is not by itself a C string.

A valid C string requires the presence of a terminating “null character”.

Note:-

A **null character has an ASCII value 0, usually represented by the character literal** `'\0'`

Since `char` is a built-in data type, no header file needs to be included to create a C string. The C library header file <cstring>contains a number of utility functions that operate on C strings

Here are some examples of declaring C strings as arrays of `char`

```
char s1[20]; // character array - can hold C string, not yet a valid string

char s2[20] = {'h', 'e', 'l', 'l', 'o', '\0'}; // Array init

char s3[20] = "Hello"; // Shortcut array init

char s4[20] =  "";// Empty or "null" C string of length 0, equal to the string literal ""
```

It is also possible to declare a C string as a pointer to a char.

```
const char* s3 = "Hello";
```

This creates an unnamed character array just large enough to hold the string (including the null character) and places the address of the first element of the aray in the `char` pointer `s3`

This is a somewhat advanced method of manipulating C strings that should probaly be avoided by inexperienced programmers who dont understand pointers yet.