STRINGS

Strings are used to store text. They can be used to display text on an LCD or in the Arduino IDE Serial Monitor window. Strings are also useful for storing the user input.
There are two types of strings in Arduino programming:

- Arrays of characters, which are the same as the strings used in C
- The Arduino String, which let us use a string object in a sketch

String Character Arrays

- A string is an array of char variables
- A string is a special array that has one extra element at the end of the string, which always has the value of 0.

EXAMPLE

```
void setup() {
   char my_str[6]; // an array big enough for a 5 character string
   Serial.begin(9600);
   my_str[0] = 'H'; // the string consists of 5 characters
   my_str[1] = 'e';
   my_str[2] = 'l';
   my_str[3] = 'l';
   my_str[4] = 'o';
   my_str[5] = 0; // 6th array element is a null terminator
   Serial.println(my_str);
}

void loop() {

}
```

The string can be printed out to the Arduino IDE Serial Monitor window by using **Serial.println()** and passing the name of the string.

This same example can be written in a more convenient way as shown below -

```
void setup() {
   char my_str[] = "Hello";
   Serial.begin(9600);
   Serial.println(my_str);
}

void loop() {

}
```

In this sketch, the compiler calculates the size of the string array and also automatically null terminates the string with a zero.

Manipulating String Arrays

We can alter a string array within a sketch as shown in the following sketch:

EXAMPLE

```
void setup() {
   char like[] = "I like coffee and cake"; // create a string
   Serial.begin(9600);
   // (1) print the string
   Serial.println(like);
   // (2) delete part of the string
   like[13] = 0;
   Serial.println(like);
   // (3) substitute a word into the string
   like[13] = ' '; // replace the null terminator with a space
   like[18] = 't'; // insert the new word
   like[19] = 'e';
   like[20] = 'a';
   like[21] = 0; // terminate the string
   Serial.println(like);
}

void loop() {

}
```

RESULT

I like coffee and cake
I like coffee
I like coffee and tea

The sketch works in the following way:

- Creating and Printing the String
- Shortening the String
- Changing a Word in the String

Functions to Manipulate String Arrays

The previous sketch manipulated the string in a manual way by accessing individual characters in the string. To make it easier to manipulate string arrays, you can write your own functions to do so, or use some of the string functions from the **C** language library.

EXAMPLE

sketch_nov03a §

```
void setup() {
   char str[] = "This is my string"; // create a string
   char out_str[40]; // output from string functions placed here
   int num; // general purpose integer
   Serial.begin(9600);

   // (1) print the string
   Serial.println(str);

   // (2) get the length of the string (excludes null terminator)
   num = strlen(str);
   Serial.print("String length is: ");
   Serial.println(num);

   // (3) get the length of the array (includes null terminator)
   num = sizeof(str); // sizeof() is not a C string function
   Serial.print("Size of the array: ");
   Serial.println(num);

   // (4) copy a string
   strcpy(out_str, str);
   Serial.println(out_str);

   // (5) add a string to the end of a string (append)
   strcat(out_str, " sketch.");
   Serial.println(out_str);
   num = strlen(out_str);
   Serial.print("String length is: ");
   Serial.println(num);
   num = sizeof(out_str);
   Serial.print("Size of the array out_str[]: ");
   Serial.println(num);
}

void loop() {

}
```

RESULT

```
This is my string
String length is: 17
Size of the array: 18
This is my string
This is my string sketch.
String length is: 25
Size of the array out_str[ ]: 40
```

The sketch works in the following way:

- Print the String
- Get the Length of the String
  strlen() function
- Get the Length of the Array
  sizeof() operator
- Copy a String
  strcpy() function
- Append a String to a String (Concatenate)
  strcat() function

Array Bounds

When working with strings and arrays, it is very important to work within the bounds of strings or arrays. In the example sketch, an array was created, which was 40 characters long, in order to allocate the memory that could be used to manipulate strings.

If the array was made too small and we tried to copy a string that is bigger than the array to it, the string would be copied over the end of the array. The memory beyond the end of the array could contain other important data used in the sketch, which would then be overwritten by our string. If the memory beyond the end of the string is overrun, it could crash the sketch or cause unexpected behavior.