Mr. Bellavia

LESSON 07 - MORE ON STRINGS

In this lesson we will dive more into strings and explore some of the built-in string functions and properties that are available to us in C# for string usage/manipulation. Understanding how to use strings and how to manipulate them is a fundamental process when learning how to code.

Sections:

I. STRING LENGTH:

Strings have a property that lets you calculate the length. For example:

```
string myString = "";
myString = "Hello World";
Console.WriteLine("'myString' has a length of " + myString.Length);
```

Output will be:

```
'myString' has a length of 11
```

Notice that we accessed the **.Length** property on the string variable 'myString'. **Note:** A **property** is like a **function** but there are no brackets or parameters.

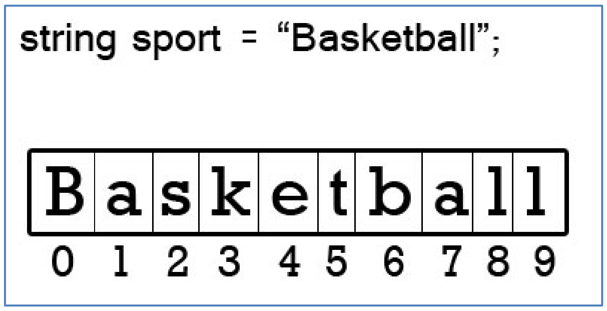
You can use the **.Length** string property on any **string** variable. You may also want to store the length inside an **int** variable. For example:

```
string myString = "";
myString = "Hello World";
int myStringLength = myString.Length;
Console.WriteLine("'myString' has a length of " + myStringLength);
```

The above code will give you the same **output.**

II. STRING POSITIONS:

Strings are basically a sequence of characters stored in memory:



This is called an **array**, but we won't get into that just yet. You can access each character in a string directly. Consider the following:

```
string myString = "";
myString = "Hello World";
Console.WriteLine("The variable 'myString' equals '" + myString + "'");
Console.WriteLine("The seventh character of 'myString' is: " + myString[6]);
```

Output will be:

```
The variable 'myString' equals 'Hello World'
The seventh character of 'myString' is: W
```

The **number** inside the **square brackets [ ]** in the code above beside the **'myString'** variable is called the **index**, which is the position of a character in a string. It is important to remember that the string position **starts at 0**, so if you want position 7 then you must say *myString[6]*.

You can also store a specific character from a string into a **char** variable:

```
string myString = "";
char myChar = '\0';
myString = "Hello World";
myChar = myString[6];
Console.WriteLine("The variable 'myString' equals '" + myString + '\'');
Console.WriteLine("The seventh character of 'myString' is: " + myChar);
```

The above code will give you the same **output.**

Also know that you can also access use an **integer variable** inside the **square brackets,** for example:

```
string myString = "Hello World";
int myIndex = 6;
char myChar = myString[myIndex];
```

III. SUBSTRINGS:

You may also want to take only part of a string. Consider the following:

```
string myString = "";
myString = "Hello World";
Console.WriteLine("Substring from position 2 and length 3: " + myString.Substring(2, 3));
```

Output will be:

```
Substring from position 2 and length 3: llo
```

The code above makes use of the built-in **Substring()** string function in which the first parameter ('2' in this case) is the starting **(index)** position where we start the substring, and the second parameter ('3' in this case) is the **length** of the string you want to extract starting from the index you chose. Always remember that the index starts at zero! So, in the above code, index 2 is the character 'l' and a length of '3' starting at that index will give you the substring **'llo'** from the string 'Hello World'.

You can also store your substring into another string:

```
string myString = "";
string mySubString = "";
myString = "Hello World";
mySubString = myString.Substring(2, 3);
Console.WriteLine("Substring from position 2 and length 3: " + mySubString);
```

The above code will give you the same **output.**

We can even ask the user for the parameters of substring, for example:

```
string myString = "", mySubString = "";
myString = "Hello World";
int startIndex = 0, length = 0;
Console.Write("Enter a start index: ");
startIndex = Convert.ToInt32(Console.ReadLine());
Console.Write("Enter a length: ");
length = Convert.ToInt32(Console.ReadLine());
mySubString = myString.Substring(startIndex, length);
Console.WriteLine("Substring from position 2 and length 3: " + mySubString);
```

IV: SEARCHING STRINGS:

It is also possible to search for parts of a string. Consider the following:

```
string myString = "";
myString = "Hello World";
Console.WriteLine(myString.IndexOf("World"));
```

Output will be:

```
6
```

What the built-in **.IndexOf()** string function is doing is searching for the word **"World"** in the **myString** variable and returning the starting index of where "World" was found . You must be careful; however, strings are **case sensitive**. Consider the following:

```
string myString = "";
myString = "Hello World";
Console.WriteLine(myString.IndexOf("world"));
```

Output will be:

```
-1
```

When a search is not successful the **.IndexOf()** function will return **'-1'.** In this case, the word **'world'** could not be found because the 'w' was not capitalized since strings are case sensitive.

Also know, similar to before, we can store the result of **.IndexOf()** into a separate **int** variable, for example:

```
string myString = "";
myString = "Hello World";
int myStringSearch = myString.IndexOf("World");
Console.WriteLine(myStringSearch);
```

Output will be:

```
6
```

We can even ask the user for the word to search for, for example:

```
string myString = "";
string? searchWord = "";
myString = "Hello World";
Console.Write("Enter a word to search: ");
searchWord = Console.ReadLine();
int myStringSearch = myString.IndexOf(searchWord);
Console.WriteLine(myStringSearch);
```

Note: You may get a **warning** here for using the string variable inputted by the user, but that's ok!

V: STRING CONCATENATION:

We have seen how we can use the plus sign **'+'** in **Console.WriteLine():**

```
Console.WriteLine("hello" + ' ' + "goodbye");
```

Output will be:

```
hello goodbye
```

Similarly, we can do the same with string variables. When you add strings together it is called string **concatenation:**

```
string s1 = "hello";
string s2 = "goodbye";
string s3 = s1 + ' ' + s2;
Console.WriteLine(s3);
```

The above code will give you the same **output.**

VI. MANY BUILT-IN STRING FUNCTIONS:

There are many built-in string functions to choose from, and there are many ways you can use them. Consider the following example:

```
1   string s = "";  //holds the string
2   int sLength = 0;    //length of string
3   int sStartIndex = 0;    //position of the current space in the string
4   int sEndIndex = 0;  //position of the next space in the string
5   string sFirstWord = ""; //holds the first word
6   string sMiddleWord = "";    //holds the middle word
7   string sLastWord = "";  //holds the last word
8
9   Console.Write("Enter a 3 word sentence: ");
10  s = Console.ReadLine();
11  sLength = s.Length;
12  sEndIndex = s.IndexOf(' ');
13  sFirstWord = s.Substring(sStartIndex, sEndIndex);
14  sStartIndex = sEndIndex + 1;
15  sEndIndex = s.IndexOf(' ', sStartIndex);
16  sMiddleWord = s.Substring(sStartIndex, sEndIndex - sStartIndex);
17  sStartIndex = sEndIndex + 1;
18  sLastWord = s.Substring(sStartIndex);
19
20  Console.WriteLine("Our string is: " + s);
21  Console.WriteLine("\nIt's length is: " + sLength);
22  Console.WriteLine("\nThe first word is: " + sFirstWord);
23  Console.WriteLine("\nThe middle word is: " + sMiddleWord);
24  Console.WriteLine("\nThe last word is: " + sLastWord);
```

Sample Input (line 10):

```
Enter a 3 word sentence: hello there world!
```

Output will be:

```
Our string is: Hello there world!

It's length is: 18

The first word is: Hello

The middle word is: there

The last word is: world!
```

Notice on **line 9** we are getting the string from the user. Finding the length is straightforward on **line 11,** so lets look ahead at how we use the two int variables **'sStartIndex'** & **'sEndIndex'** as parameters for **.Substring().** Here is what our string **'s'** looks like:

H	e	l	l	o		t	h	e	r	e		w	o	r	l	d	!
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17

Let's look at **lines 12 & 13** in our code:

```
12  sEndIndex = s.IndexOf(' ');
13  sFirstWord = s.Substring(sStartIndex, sEndIndex);
```

At this point, **sStartIndex** equals **'0',** and **sEndIndex** equals **'5'.** We can use both these values in our **.Substring()** function to extract the first word **'hello'.**

Let's look at **lines 14 & 15** in our code:

```
14  sStartIndex = sEndIndex + 1;
15  sEndIndex = s.IndexOf(' ', sStartIndex);
```

Now we set the value of **sStartIndex** equal to **sEndIndex** (5) plus 1. This sets our **sStartIndex** equal to **'6'.** On **line 15** we see that **s.IndexOf()** is taking two parameters this time (not just one like before). When the **.IndexOf()** function is given two parameters, the first parameter is still what we want to **search** for, however the second parameter is the **index** we wish to start at. Therefore, we are searching for the space character, but starting at index **'6'** (i.e., value of **sStartIndex**). This will result in **sEndIndex** equaling **'11'.**

On **line 16** we have:

```
16  sMiddleWord = s.Substring(sStartIndex, sEndIndex - sStartIndex);
```

In this case for **.Substring(),** our **start index** is set to **sStartIndex** which equals **'6',** and our **end index** is set to **sEndIndex** (11) minus **sStartIndex** (6) which will equal **'5'.** Now we have successfully extracted our middle word **'there'.**

Finally, we have lines **17 & 18:**

```
17  sStartIndex = sEndIndex + 1;
18  sLastWord = s.Substring(sStartIndex);
```

We now set **sStartIndex** to **sEndIndex** (11) plus 1 which equals **'12'.** We then use the **.SubString()** function and supply only one parameter this time (unlike before). When you pass only one parameter to **.SubString()** then it will start from the **index** given and go to the end of the string automatically. In this case, we are taking the substring from index '11' to the end of the string which results in our last word **'world!'.**

As you can see, there are many ways you can use/manipulate strings!