

OBJECTIVE-C STRINGS

http://www.tutorialspoint.com/objective_c/objective_c_strings.htm

Copyright © tutorialspoint.com

The string in Objective-C programming language is represented using NSString and its subclass NSMutableString provides several ways for creating string objects. The simplest way to create a string object is to use the Objective-C @"..." construct:

```
NSString *greeting = @"Hello";
```

A simple example for creating and printing a string is shown below.

```
#import <Foundation/Foundation.h>

int main ()
{
    NSString *greeting = @"Hello";

    NSLog(@"Greeting message: %@\n", greeting );

    return 0;
}
```

When the above code is compiled and executed, it produces result something as follows:

```
2013-09-11 01:21:39.922 demo[23926] Greeting message: Hello
```

Objective-C supports a wide range of methods for manipulate strings:

S.N.	Method & Purpose
1	- NSString * capitalizedString; Returns a capitalized representation of the receiver.
2	- unichar characterAtIndex:NSUInteger index; Returns the character at a given array position.
3	- double doubleValue; Returns the floating-point value of the receiver's text as a double.
4	- float floatValue; Returns the floating-point value of the receiver's text as a float.
5	- BOOL hasPrefix:NSString * aString; Returns a Boolean value that indicates whether a given string matches the beginning characters of the receiver.
6	- BOOL hasSuffix:NSString * aString; Returns a Boolean value that indicates whether a given string matches the ending characters of the receiver.
7	- id initWithFormat:NSString * format ...;

Returns an NSString object initialized by using a given format string as a template into which the remaining argument values are substituted.

8 - **NSIntegerintegerValue;**

Returns the NSInteger value of the receiver's text.

9 - **BOOLisEqualToString:NSString * aString;**

Returns a Boolean value that indicates whether a given string is equal to the receiver using a literal Unicode-based comparison.

10 - **NSUIntegerlength;**

Returns the number of Unicode characters in the receiver.

11 - **NSString * lowercaseString;**

Returns lowercased representation of the receiver.

12 - **NSRangerangeOfString:NSString * aString;**

Finds and returns the range of the first occurrence of a given string within the receiver.

13 - **NSString * stringByAppendingFormat:NSString * format ...;**

Returns a string made by appending to the receiver a string constructed from a given format string and the following arguments.

14 - **NSString * stringByTrimmingCharactersInSet:NSCharacterSet * set;**

Returns a new string made by removing from both ends of the receiver characters contained in a given character set.

15 - **NSString * substringFromIndex:NSUIntegeranIndex;**

Returns a new string containing the characters of the receiver from the one at a given index to the end.

Following example makes use of few of the above-mentioned functions:

```
#import <Foundation/Foundation.h>

int main ()
{
    NSString *str1 = @"Hello";
    NSString *str2 = @"World";
    NSString *str3;
    int len ;

    NSAutoreleasePool * pool = [[NSAutoreleasePool alloc] init];

    /* uppercase string */
    str3 = [str2 uppercaseString];
    NSLog(@"Uppercase String : %@\n", str3 );

    /* concatenates str1 and str2 */
    str3 = [str1 stringByAppendingFormat:@"World"];
```

```

    NSLog(@"Concatenated string:  %@\n", str3 );

    /* total length of str3 after concatenation */
    len = [str3 length];
    NSLog(@"Length of Str3 :  %d\n", len );

    /* initWithFormat */
    str3 = [[NSString alloc] initWithFormat:@"%s %s", str1, str2];
    NSLog(@"Using initWithFormat:  %@\n", str3 );
    [pool drain];

    return 0;
}

```

When the above code is compiled and executed, it produces result something as follows:

```

2013-09-11 01:15:45.069 demo[30378] Uppercase String :  WORLD
2013-09-11 01:15:45.070 demo[30378] Concatenated string:  HelloWorld
2013-09-11 01:15:45.070 demo[30378] Length of Str3 :  10
2013-09-11 01:15:45.070 demo[30378] Using initWithFormat:  Hello World

```

You can find a complete list of Objective-C NSString related methods in [NSString Class Reference](#).

Loading [MathJax]/jax/output/HTML-CSS/jax.js