How to: Search Strings Using Regular Expressions (C# Programming Guide)

Article02/04/2013

The System.Text.RegularExpressions.Regex class can be used to search strings. These searches can range in complexity from very simple to making full use of regular expressions. The following are two examples of string searching by using the Regex class. For more information, see .NET Framework Regular Expressions .

Example

The following code is a console application that performs a simple case-insensitive search of the strings in an array. The static method Regex.IsMatch performs the search given the string to search and a string that contains the search pattern. In this case, a third argument is used to indicate that case should be ignored. For more information, see System.Text.RegularExpressions.RegexOptions .

C#

```
class TestRegularExpressions
{
    static void Main()
    {
        string[] sentences =
        {
            "C# code",
            "Chapter 2: Writing Code",
            "Unicode",
            "no match here"
        };

        string sPattern = "code";

        foreach (string s in sentences)
        {
            System.Console.Write("{0,24}", s);

            if (System.Text.RegularExpressions.Regex.IsMatch(s, sPattern,
System.Text.RegularExpressions.RegexOptions.IgnoreCase))
            {
                System.Console.WriteLine("  (match for '{0}' found)", sPat-
tern);
```

```
            }
            else
            {
                System.Console.WriteLine();
            }
        }

        // Keep the console window open in debug mode.
        System.Console.WriteLine("Press any key to exit.");
        System.Console.ReadKey();

    }
}
/* Output:
           C# code  (match for 'code' found)
           Chapter 2: Writing Code  (match for 'code' found)
           Unicode  (match for 'code' found)
           no match here
*/
```

The following code is a console application that uses regular expressions to validate the format of each string in an array. The validation requires that each string take the form of a telephone number in which three groups of digits are separated by dashes, the first two groups contain three digits, and the third group contains four digits. This is done by using the regular expression ^\\d{3}-\\d{3}-\\d{4}$. For more information, see Regular Expression Language - Quick Reference .

C#

```
class TestRegularExpressionValidation
{
    static void Main()
    {
        string[] numbers =
        {
            "123-555-0190",
            "444-234-22450",
            "690-555-0178",
            "146-893-232",
            "146-555-0122",
            "4007-555-0111",
            "407-555-0111",
            "407-2-5555",
        };

        string sPattern = "^\\d{3}-\\d{3}-\\d{4}$";

        foreach (string s in numbers)
```

```
        {
            System.Console.Write("{0,14}", s);

            if (System.Text.RegularExpressions.Regex.IsMatch(s, sPattern))
            {
                System.Console.WriteLine(" - valid");
            }
            else
            {
                System.Console.WriteLine(" - invalid");
            }
        }

        // Keep the console window open in debug mode.
        System.Console.WriteLine("Press any key to exit.");
        System.Console.ReadKey();
    }
}
/* Output:
       123-555-0190 - valid
     444-234-22450 - invalid
      690-555-0178 - valid
       146-893-232 - invalid
      146-555-0122 - valid
     4007-555-0111 - invalid
      407-555-0111 - valid
        407-2-5555 - invalid
*/
```

See Also

Concepts

C# Programming Guide

Other Resources

Strings (C# Programming Guide)