Технологии программирования

Лабораторная работа №6

Тема: Регулярные выражения

Регулярные выражения являются быстрым, простым и относительно стандартным способом поиска и обработки текстовых данных. Это привело к большой популярности данной технологии, несмотря на ее недостатки. Большинство языков и сред программирования содержат ту или иную библиотеку поддержки регулярных выражений. В C# используется библиотека регулярных выражений, совместимая с Perl 5. Она включена в стандартный набор библиотек и имеет пространство имен System.Text.RegularExpressions.

Для работы с регулярными выражениями служит класс Regex. Один экземпляр класса соответствует одному шаблону регулярных выражений. Это связано с необходимостью поддержания сложных структур данных для работы конечного автомата или формальной грамматики, соответствующих данному выражению. Эти структуры обеспечивают высокую производительность.

Для примера, проверим, есть ли в введенной строке русские буквы:

private void button1\_Click(object sender, EventArgs e)

{

Regex r\_test;

r\_test = new Regex("[а-яА-ЯёЁ]");

if (r\_test.IsMatch(textBox1.Text))

textBox2.AppendText("есть\r\n");

else

textBox2.AppendText("нету\r\n");

}

Метод IsMatch определяет, есть ли в строке совпадения с данным шаблоном (регулярным выражением). В случае, если нам необходимо найти конкретные соответствующие элементы, используется метод Matches:

private void button2\_Click(object sender, EventArgs e)

{

Regex match\_test = new Regex("[0-9]+ \*\\+ \*[0-9]+");

MatchCollection result = match\_test.Matches(textBox1.Text);

foreach (Match m in result)

textBox2.AppendText(m.Value + "\r\n");

}

В этом примере осуществляется поиск операций сложения целых чисел в строке. Также на этом примере вы можете убедиться, что в обычных регулярных выражения не осуществляется поиск назад (совпадения ищутся только впереди), введя строку типа «5 + 6 + 2».

В случаях, если вам необходимо получить строки, являющиеся лишь частью поискового шаблона, используются поисковые группы (именованные или индексированные). Например, выделим имена пользователей из набора строк:

private void button3\_Click(object sender, EventArgs e)

{

Regex pass\_test = new Regex("^Username\\: (?<username>[a-zA-Z0-9]+)", RegexOptions.Multiline);

foreach (Match m in pass\_test.Matches(textBox1.Text))

{

textBox2.AppendText(m.Groups["username"] + "\r\n");

}

}

Вариант 1.

1. Проверить, что строка является корректным шестнадцатеричным числом.
2. Проверить, что строка является корректным доменным именем.
3. Выделить из строки все имена (начинаются с большой буквы, все остальные строчные, только на одном языке)

Вариант 2.

1. Проверить, что строка является корректным восьмеричным числом.
2. Проверить, что строка является корректным адресом электронной почты.
3. Выделить из строки все идентификаторы.

Вариант 3.

1. Проверить, что строка является корректным двоичным числом.
2. Проверить, что строка является корректным абсолютным URL.
3. Выделить из строки все вещественные числа.

Вариант 4.

1. Проверить, что строка является корректным вещественным числом.
2. Проверить, что строка является корректным путем к файлу.
3. Выделить из строки все целые числа.