ĐẠI HỌC XÂY DỰNG HÀ NỘI

BM CNPM – KHOA CNTT

Bài 7

Chuỗi

Biểu thức quy tắc

Nội dung

- Lớp String
- Các thao tác với chuỗi
- Lớp StringBuilder
- Các thao tác với StringBuilder
- Biểu thức quy tắc

Lớp String

- Lớp String là một lớp cơ bản trong .NET nằm trong namespace System giúp người dùng thao tác với chuỗi một cách dễ ràng.
- Tất cả các chuỗi trong C# đều là đối tượng xuất phát từ lớp String, do vậy khi một chuỗi được tạo ra bạn có thể gọi các phương thức của lớp String để thực hiện các thao tác với nó
- Các cách tạo chuỗi
  - Khai báo biến thuộc kiểu chuỗi và gán giá trị
  - Tạo chuỗi bằng cách gọi phương thức ToString của đối tượng

Một số thao tác với chuỗi

SE - FIT

Ví dụ

```
// Sử dụng ba cách so sánh hai chuỗi
// Cách 1 sử dụng một chuỗi để so sánh với chuỗi còn lại
Console.WriteLine("S6.Equals(S7) ?: {0}", s6.Equals(s7));
// Cách 2 dùng hàm của lớp string so sánh hai chuỗi
Console.WriteLine("Equals(S6, s7) ?: {0}", string.Equals(s6, s7));
// Cách 3 dùng toán tử so sánh
Console.WriteLine("S6 == S7 ?: {0}", s6 == s7);
// Sử dụng hai thuộc tính hay dùng là chỉ mục và chiều dài của chuỗi
Console.WriteLine("\nChuoi S7 co chieu dai la : {0}", s7.Length);
Console.WriteLine("Ky tu thu 3 cua chuoi S7 la : {0}", s7[2]);
// Kiểm tra xem một chuỗi có kết thúc với một nhóm ký
// tự xác định hay không
Console.WriteLine("S3: {0}\n ket thuc voi chu CNTT ? : {1}\n", s3, s3.EndsWith("CNTT"));
Console.WriteLine("S3: {0}\n ket thuc voi chu Nam ? : {1}\n", s3, s3.EndsWith("Nam"));
// Trả về chỉ mục của một chuỗi con
Console.WriteLine("\nTim vi tri xuat hien dau tien cua chu CNTT ");
Console.WriteLine("trong chuoi S3 là {0}\n", s3.IndexOf("CNTT"));
// Chèn từ nhân lực vào trước CNTT trong chuỗi S3
string s8 = s3.Insert(18, "nhan luc");
Console.WriteLine(" S8 : {0}\n", s8);
// Ngoài ra ta có thể kết hợp như sau
string s9 = s3.Insert(s3.IndexOf("CNTT"), "nhan luc ");
Console.WriteLine(" S9 : {0}\n", s9);
```

Lớp StringBuilder

- Lớp StringBuilder giúp bạn thao tác nhanh các chuỗi với ít tổn hao bộ nhớ hơn so với lớp String.
- Các cách tạo đối tượng StringBuilder

Ví dụ

```
//Tạo đối tượng sb trống
StringBuilder sb = new StringBuilder();
//Khởi tạo với chiều dài là 20 ký tự
StringBuilder sb1 = new StringBuilder(20);
//Khởi tạo với 1 chuỗi xác định
StringBuilder sb2 = new StringBuilder("IT Plus");
//Khởi tạo với 1 chuỗi và chiều dài là 100, tức là chuỗi còn dư khoảng trống
StringBuilder sb3 = new StringBuilder("Welcome to IT Plus",100);
```

Thao tác với StringBuilder 1-2

- **Chèn 1 chuỗi bất kỳ bắt đầu từ chỉ mục trong chuỗi sb**

  sb.Insert(<chỉ mục>,"chuỗi")

- **Xóa chuỗi con trong chuỗi sb**

  sb.Remove(<chỉ mục bắt đầu>,<số ký tự cần xóa>)

- **Thay thế 1 chuỗi con trong sb với 1 chuỗi mới**

  sb.Replace(<chuỗi con cần thay thế>,<chuỗi con mới>)

- **Bổ sung 1 chuỗi vào chuỗi sb**

  sb.Append(<chuỗi cần bổ sung>)

Thao tác với StringBuilder 2-2

SE - FIT

Ví dụ

```
//thao tác với chuỗi
StringBuilder sbd = new StringBuilder("Con kien bo tren lung con bo");
sbd.Insert(12, "nhanh ");
//kết quả: Con kien bo nhanh tren lung con bo
sbd.Remove(23, 5);
//kết quả: Con kien bo nhanh tren con bo
sbd.Replace("con bo", "con trau");
//kết quả: Con kien bo nhanh tren con trau
sbd.AppendFormat(" voi van toc {0} m/s", 1);
//kết quả: Con kien bo nhanh tren con trau voi van toc 1 m/s

int vitri = sb.ToString().IndexOf("kien");
if (vitri > 0)
    // chèn dấu phẩy ở vị trí tìm thấy + chiều dài của chuỗi con "kien"
    sb.Insert(vitri + "kien".Length, ",");
```

Biểu thức quy tắc

- Biểu thức quy tắc(Regular Expression-RE) là một ngôn ngữ cực mạnh dùng mô tả văn bản cũng như thao tác trên văn bản. Một RE thường được ứng dụng lên một chuỗi, nghĩa là lên một nhóm ký tự. Chẳng hạn, ta có chuỗi sau: **Mot, Hai, Ba, Bon, NEVERLAND.**
- Một RE là một kiểu mẫu văn bản gồm 2 phần:
  - **literal** (phần chữ). Một literal đơn thuần chỉ là một ký tự (a-z) mà bạn muốn đem so khớp với chuỗi đích.
  - **Metacharacters** (phần siêu ký tự). là một ký tự đặc biệt hoạt động như là 1 mệnh lệnh đối với bộ phận phân tích ngữ nghĩa (parser) của RE

Các MetaCharacter

Các MetaCharacter 2-2

[abc] : khớp với 1 ký tự nằm trong nhóm là a hay b hay c.
[a-z] so trùng với 1 ký tự nằm trong phạm vi a-z, dùng dấu - làm dấu ngăn cách.
[^abc] sẽ không so trùng với 1 ký tự nằm trong nhóm, ví dụ không so trùng với a hay b hay c.
() : Xác định 1 group (biểu thức con) xem như nó là một yếu tố đơn lẻ trong pattern .ví dụ ((a(b))c) sẽ khớp với b, ab, abc.
? : khớp với đứng trước từ 0 hay 1 lần. Ví dụ A?B sẽ khớp với B hay AB.
***** : khớp với đứng trước từ 0 lần trở lên . A*B khớp với B, AB, AAB
+ : khớp với đứng trước từ 1 lần trở lên. A+B khớp với AB, AAB.
{n} : n là con số, Khớp đúng với n ký tự đúng trước nó . Ví dụ A{2}) khớp đúng với 2 chữ A.
{n, } : khớp đúng với n ký tự trở lên đứng trước nó , A{2,} khớp vói AA, AAA ...
{m,n} : khớp đùng với từ m->n ký tự đứng trước nó, A{2,4} khớp vói AA,AAA,AAAA

Lớp Regex 1-2

- **Phương thức:**
  - **GetGroupNames:** trả về mảng gồm toàn tên nhóm thu lượm đối với RE.
  - **GetGroupNumbers:** trả về mảng gồm toàn số nhóm thu lượm tương ứng với tên nhóm trên 1 mảng.
  - **GroupNameFromNumber:** đi lấy tên nhóm tương ứng với số nhóm được khai báo.
  - **IsMatch:** trả về trị bool cho biết liệu xem RE có tìm thấy một so khớp hay không trên pattern
  - b dò tìm trên pattern xem có xuất hiện một RE hay không rồi trả về kết quả chính xác như là một đối tượng Match duy nhất.
  - **Matches:** dò tìm trên pattern xem tất cả các xuất hiện của một RE có hay không rồi trả về tất cả những so khớp thành công xem như Match được gọi nhiều lần.
  - **Replace:** cho thay thế những xuất hiện của một pattern được định nghĩa bởi một RE bởi một chuỗi ký tự thay thế được chỉ định.
  - **Split:** chẻ một pattern thành một mảng gồm những chuỗi con ở những vị trí được chỉ định bởi một so khớp trên RE
  - **Unescape:** cho unescape bất cứ những ký tự nào được escape trên pattern.

Lớp Regex 2-2

Ví dụ

```
string chuoi = "Mot, Hai, Ba, Bon, NEVERLAND.";
//tạo pattern
//luật: xem chuỗi nào có chứa khoảng trắng hay dấu phẩy
string pattern = " |, ";
Regex myRegex = new Regex(pattern);
string[] sKetQua = myRegex.Split(chuoi);
foreach (string subString in sKetQua)
{
    Console.WriteLine(subString);
}
//kết quả mỗi từ sẽ in ra trên 1 dòng
```

Lớp Match

- Lớp này tượng trưng cho những kết quả duy nhất của một tác vụ so khớp (match) RE. Phương thức Match của lớp Regex để trả về 1 đối tượng kiểu Match để có thể tìm ra so khớp đầu tiên trên chuỗi nhập vào. Sử dụng thuộc tính Match.Access của lớp Match báo cho biết liệu xem đã tìm ra 1 so khớp hay chưa.

Ví dụ

```
string chuoi = "123abcd456bdabc";
string pattern = "abc";
Regex myRegex = new Regex(pattern);
Match m = myRegex.Match(chuoi);
if (m.Success)
{
    Console.WriteLine("Tim thay chuoi con {0} o vi tri thu {1} trong chuoi", m.Value, m.Index);
}
else
    Console.WriteLine("Khong tim thay chi ca");
```

Lớp MatchCollection 1-2

- Lớp này tượng trưng cho 1 loạt những so khớp thành công đè chồng lên nhau tạo thành một tập hợp bất di bất dịch và lớp này không có phương thức khởi tạo. Nhưng đối tượng MatchCollection sẽ do thuộc tính **Regex.Matches** của lớp Regex trả về.

Lớp MatchCollection 2-2

SE - FIT

Ví dụ

```
//tập hợp chứa những so khớp
MatchCollection mc;
//1 chuỗi thử nghiệm
string chuoi = "I like money, like woman and like C#";
//tạo pattern
string pattern = "like";
//khởi tạo 1 đối tượng của Regex
//truyền chuỗi pattern vào constructor
Regex myRegex = new Regex(pattern);
//dùng phương thức Matches của myRegex
//để tìm ra matches và chỉ mục của từng match
mc = myRegex.Matches(chuoi);
foreach (Match m in mc)
{
    Console.WriteLine("Chuoi con '{0}' xuat hien o chi muc {1}", m.Value, m.Index);
}
```

Lớp Group

SE - FIT

Ví dụ

```
string pattern = "(a(b))c";
string chuoi = "abdabc";
//định nghĩa những substring abc,ab,b
Regex myRegex = new Regex(pattern);
Match m = myRegex.Match(chuoi);

for (int i = 0; m.Groups[i].Value != ""; i++)
{
    Console.WriteLine("{0} co chieu dai {1}", m.Groups[i].Value, m.Groups[i].Length);
}
//kết quả
//abc co chieu dai 3
//ab co chieu dai 2
//b co chieu dai 1
```

Lớp GroupCollection

- Là lớp tượng trưng cho 1 tập hợp gồm toàn những nhóm được thu lượm và trả về một lô những nhóm được thu lượm trong một lần so khớp duy nhất. Collection này thuộc loại read-only và không có phương thức khởi tạo. Các thể hiện của lớp GroupCollection được trả về trong tập hợp mà thuộc tính Match.Groups trả về.

Ví dụ

```
Regex myRegex = new Regex("(a(b))c");
Match m = myRegex.Match("abdabc");
Console.WriteLine("So nhom duoc tim thay la: {0}", m.Groups.Count);
```

Question & Answer

SE - FIT