18

[C#] Regex no mundo .NET

Transcrição

O mundo .Net e C# não pode ficar de fora e claro, também dão suporte de primeira para usar as expressões regulares. A classe principal se chama `Regex`, do *namespace* `System.Text.RegularExpressions` e recebe o *pattern* no construtor:

```
Regex regexp = new Regex(@"(\d\d)(\w)");
```

Repare que a string possui um `@` na sua frente. Isso é útil e melhora a legibilidade, pois não é preciso escapar o caractere `\`.

Uma vez o objeto `Regex` criado, podemos chamar o método `match` para executar uma busca simples, e o método `Matches` para buscar todos os *matches* dentro do alvo:

```
string alvo = "12a34b56c";
Regex regexp = new Regex(@"(\d\d)(\w)");

MatchCollection resultados = regexp.Matches(alvo);
```

O objeto do tipo `MatchCollection` possui todas as informações sobre cada *match*, basta iterar:

```
foreach(Match resultado in resultados) {
    Console.WriteLine(resultado.Value);//imprime 12a,34b,56c
    Console.WriteLine(resultado.Groups[1]); //imprime 12, 34, 56
    Console.WriteLine(resultado.Groups[2]); //imprime a, b, c
    Console.WriteLine(resultado.Index); //imprime 0, 3, 6
    Console.WriteLine(resultado.Length)); //imprime 3, 3, 3
}
```

Suporte excelente pelo C#, não? Segue uma vez o código completo:

```
using System.Text.RegularExpressions;
namespace ExemploRegex
{
    class Program
    {
        static void Main(string[] args)
        {
            string alvo = "12a34b56c";
            Regex regexp = new Regex(@"(\d\d)(\w)");

            MatchCollection resultados = regexp.Matches(alvo);
            foreach(Match resultado in resultados)
            {
                Console.WriteLine(string.Format("Resultado: {0}, Grupos: {1} e {2}, Index: [{3},{4}]
                    resultado.Value,
                    resultado.Groups[1],
                    resultado.Groups[2],
```

```
                    resultado.Index,
                    (resultado.Index+resultado.Length)));
                    Console.WriteLine("--------------");
                }
            }
        }
    }
```