

Entendiendo Regular Expressions con C#

por Martín Márquez <xomalli@gmail.com>

Las expresiones regulares (regular expressions) han sido utilizadas con éxito desde hace tiempo como una solución avanzada mucho más compleja y eficiente para el procesamiento y la validación de texto en herramientas como grep, sed, AWK, bash y en lenguajes de programación del tipo scripting como Perl, Python y PHP.

Una expresión regular (regular expression o regexp) es un patrón de cadenas de caracteres formado por una combinación de caracteres especiales llamados metacaracteres o cuantificadores y de caracteres alfanuméricos llamados literales, este patrón representa un lenguaje regular o un conjunto regular de cadenas para tres operaciones básicas: adyacencia, repetición y alteración.

En el sitio <http://www.regular-expressions.info/> se da una mayor referencia acerca de los cuantificadores y su significado.

Existen dos implementaciones de expresiones regulares POSIX y PERL, en el caso de .NET el motor de expresiones regulares utiliza la implementación compatible con Perl 5. Las expresiones regulares en .NET se encuentran integradas como clases dentro del ensamblado System.Text.RegularExpressions estas clases utilizan un motor implementado como un autómata finito no determinístico (NFA) similar al que emplean Perl, Python y Emacs con algunas características propias de .NET.

Para ejemplificar el uso de expresiones regulares en C#, mostraremos un programa sencillo que tenga una funcionalidad similar al comando grep o egrep, como sabemos este comando en su funcionamiento básico recibe como argumentos una expresión regular y uno o varios archivos en donde buscar e imprime las líneas que coincidan con esa expresión regular. El código del programa es el siguiente:

```
using System;  
using System.Text;  
using System.Text.RegularExpressions;  
using System.IO;
```

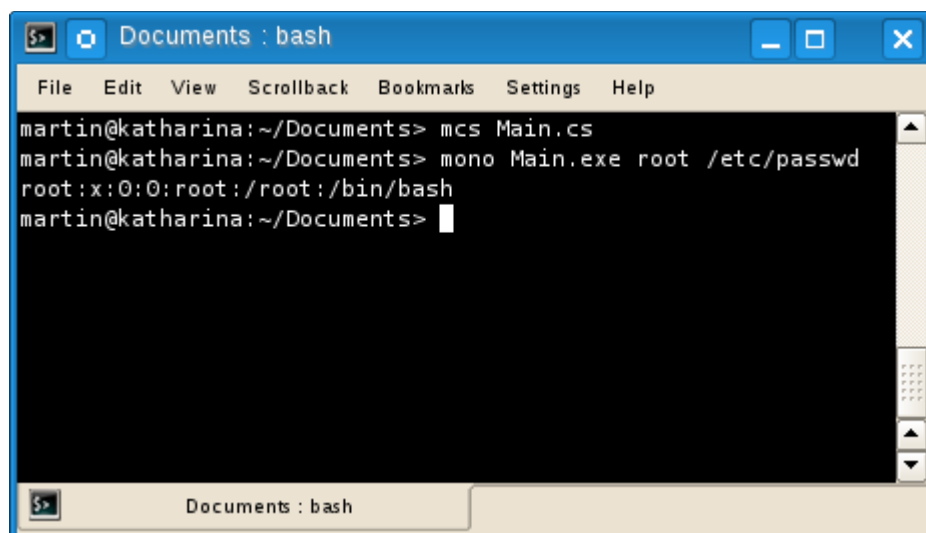
```

namespace Samples.TestGrep
{
    class Program
    {
        public static void Main(string[] args)
        {
            if(args.Length == 2)
            {
                string pattern = args[0];
                string file = args[1];
                Regex regexp = new Regex(@pattern, RegexOptions.Multiline);
                FileInfo fi = new FileInfo(file);
                if(File.Exists(fi.FullName))
                {
                    using(StreamReader sr = new StreamReader(file))
                    {
                        string line;
                        while((line = sr.ReadLine()) != null)
                        {
                            Match m = regexp.Match(line);
                            if(m.Success)
                                Console.WriteLine("{0}", line);
                        }
                    }
                }
                else
                    Console.WriteLine("File not found");
            }
            else
                Console.WriteLine("Usage: mono TestGrep [pattern] [file]");
        }
    }
}

```

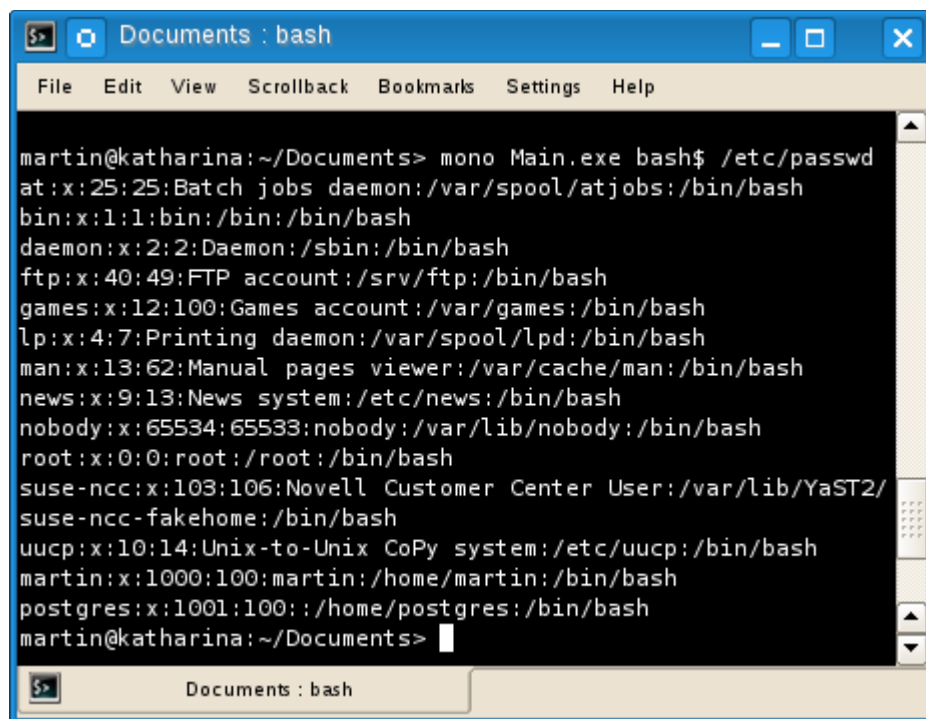
Al ejecutar el programa podemos observar el resultado como se muestra en las siguientes imágenes probando con diferentes patrones con los archivos etc/password y /etc/group respectivamente.

Fig 1 Una primera prueba con el archivo /etc/password



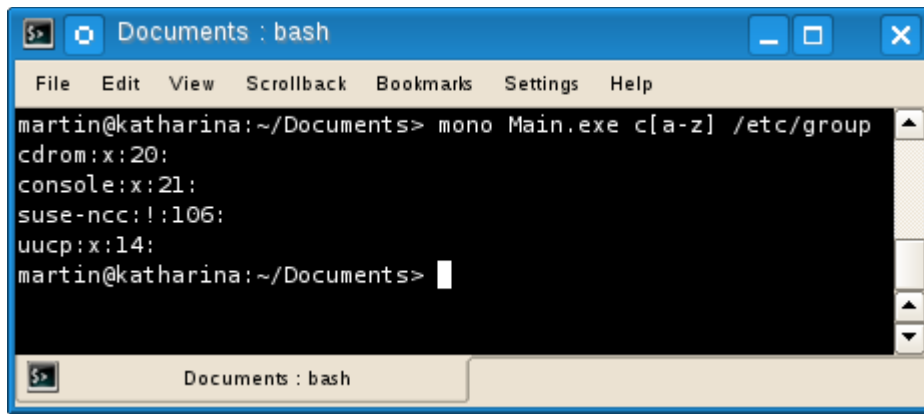
```
Documents : bash
File Edit View Scrollback Bookmarks Settings Help
martin@katharina:~/Documents> mcs Main.cs
martin@katharina:~/Documents> mono Main.exe root /etc/passwd
root:x:0:0:root:/root:/bin/bash
martin@katharina:~/Documents> 
```

Fig 2 Una segunda prueba con el archivo /etc/password



```
Documents : bash
File Edit View Scrollback Bookmarks Settings Help
martin@katharina:~/Documents> mono Main.exe bash$ /etc/passwd
at:x:25:25:Batch jobs daemon:/var/spool/atjobs:/bin/bash
bin:x:1:1:bin:/bin:/bin/bash
daemon:x:2:2:Daemon:/sbin:/bin/bash
ftp:x:40:49:FTP account:/srv/ftp:/bin/bash
games:x:12:100:Games account:/var/games:/bin/bash
lp:x:4:7:Printing daemon:/var/spool/lpd:/bin/bash
man:x:13:62:Manual pages viewer:/var/cache/man:/bin/bash
news:x:9:13:News system:/etc/news:/bin/bash
nobody:x:65534:65533:nobody:/var/lib/nobody:/bin/bash
root:x:0:0:root:/root:/bin/bash
suse-ncc:x:103:106:Novell Customer Center User:/var/lib/YaST2/
suse-ncc-fakehome:/bin/bash
uucp:x:10:14:Unix-to-Unix CoPy system:/etc/uucp:/bin/bash
martin:x:1000:100:martin:/home/martin:/bin/bash
postgres:x:1001:100::/home/postgres:/bin/bash
martin@katharina:~/Documents> 
```

Fig 3 Probando con el archivo /etc/group



```
Documents : bash
File Edit View Scrollback Bookmarks Settings Help
martin@katharina:~/Documents> mono Main.exe c[a-z] /etc/group
cdrom:x:20:
console:x:21:
suse-ncc:!:106:
uucp:x:14:
martin@katharina:~/Documents>
```



[Download el código fuente para Xamarin Studio o Visual Studio](#)

Este documento está protegido bajo la licencia de documentación libre *Free Documentacion License* del Proyecto GNU, para consulta ver el sitio <http://www.gnu.org/licenses/fdl.txt> , toda persona que lo desee está autorizada a usar, copiar y modificar este documento según los puntos establecidos en la «Licencia FDL»