9.5. Llamadas al sistema

Si hay algo que no sepamos o podamos hacer, pero que alguna utilidad del sistema operativo sí es capaz de hacer por nosotros, podemos hacer que ella realice ese trabajo. La forma de llamar a otras órdenes del sistema operativo (e incluso programas externos de casi cualquier tipo) es creando un nuevo proceso con "Process.Start". Por ejemplo, podríamos lanzar el bloc de notas de Windows con:

```
Process proc = Process.Start("notepad.exe");
```

(que necesitará System.Diagnostics en la lista de "using").

En los actuales sistemas operativos multitarea se da por sentado que no es necesario esperar a que termine otra la tarea, sino que nuestro programa puede proseguir. Si aun así, queremos esperar a que se complete la otra tarea (por ejemplo, porque deseemos comprobar si ha existido algún error durante la ejecución), lo conseguiríamos con "WaitForExit", añadiendo esta segunda línea:

```
proc.WaitForExit();
```

Un programa completo que lanzara el bloc de notas y que esperase a que se cerrase podría ser:

```
// Ejemplo_09_05a.cs
// Lanzar otro proceso y esperar
// Introducción a C#, por Nacho Cabanes

using System;
using System.Diagnostics;

class Ejemplo_09_05a
{
    static void Main()
    {
        Console.WriteLine("Lanzando el bloc de notas...");
        Process proc = Process.Start("notepad.exe");
        Console.WriteLine("Esperando a que se cierre...");
        proc.WaitForExit();
        Console.WriteLine("Terminado!");
    }
}
```

Si se desea añadir algún parámetro al programa que se lanza, se puede hacer usando una versión sobrecargada de Process.Start:

```
Process proc = Process.Start("notepad.exe","fichero.txt");
```

De igual modo, si no deseamos esperar indefinidamente, sino una cierta cantidad máxima de segundos, se puede indicar como parámetro opcional de WaitForExit:

```
proc.WaitForExit(5000);
```

Así, una variante del programa anterior podría ser:

```
// Ejemplo_09_05b.cs
// Lanzar otro proceso y esperar (2)
// Introducción a C#, por Nacho Cabanes

using System;
using System.Diagnostics;

class Ejemplo_09_05b
{
    static void Main()
    {
        Console.WriteLine("Lanzando el bloc de notas...");
        Process proc = Process.Start("notepad.exe", "fichero.txt");
        Console.WriteLine("Esperando 5 segundos...");
        proc.WaitForExit(5000);
        Console.WriteLine("Terminado!");
    }
}
```

Se puede saber si el proceso ha salido con algún código de error mirando el valor de "proc.ExitCode". También se puede afinar un poco más el comportamiento, por ejemplo forzando a que la ventana del nuevo proceso esté inicialmente minimizada (Minimized) o incluso oculta (Hidden) con la ayuda de la clase "ProcessStartInfo", así:

```
ProcessStartInfo proc = new ProcessStartInfo("miPrograma.exe");
proc.WindowStyle = ProcessWindowStyle.Minimized;
Process.Start(proc);
```

Ejercicios propuestos:

(9.5.1) Crea un programa que muestre un menú, que te permita lanzar ciertas aplicaciones que uses con frecuencia pulsando sólo una tecla (p.ej., del 0 al 9).

(9.5.2) Haz un programa que mida el tiempo que tarda en ejecutarse un cierto proceso. Este proceso se le indicará como parámetro en la línea de comandos.

(9.5.3) Crea un programa que lance un compresor de línea de comandos (como 7-zip o RAR, por ejemplo) para comprimir un fichero cuyo nombre escoja el usuario, usando la contraseña "1234".

(9.5.4) Crea un programa que trate de descomprimir el fichero comprimido que has creado en el ejercicio anterior, probando todas las contraseñas numéricas de

4 cifras (desde "0000" hasta "9999"). Podrás saber si has encontrado la contraseña correcta porque en ese intento proc.ExitCode valdrá 0 (para indicar que el proceso anterior se ha podido lanzar sin problemas), mientras que en el resto de intentos será un número distinto de cero (puedes mirar la documentación del compresor que hayas utilizado, si quieres saber qué códigos de error concretos devuelve al sistema operativo). Quizá te interese lanzar el (des)compresor como ventana oculta o al menos como ventana minimizada, para que las 10.000 ventanas que pueden llegar a aparecer no te impidan la visión del escritorio y, por tanto, el manejo normal del sistema.

(9.5.5) Mejora el ejercicio 9.4.3 (la versión básica del programa "tipo Comandante Norton") para que, si se pulsa Intro sobre un cierto fichero, lance el correspondiente proceso.

(9.5.6) Aplica esta misma mejora al ejercicio 9.4.5 (la versión con dos paneles del programa "tipo Comandante Norton").