

Ejercicios Eventos

[Descargar estos ejercicios](#)

Índice


1. [Ejercicio 1](#)
2. ☒ [Ejercicio 2](#)
3. [Ejercicio 3](#)
4. [Ejercicio 4](#)

Ejercicio 1

Vamos realizar un programa que detecte **mediante eventos** la modificación del nombre de cualquier archivo perteneciente a determinado directorio. Para ello, vamos a utilizar la clase `FileSystemWatcher` que define un evento `Renamed` el cual se desencadena cuando un usuario cambia el nombre de un archivo que se encuentre en la carpeta especificada en la ruta que le indicamos al crear el objeto `FileSystemWatcher`. Puedes obtener más información de su uso en la [documentación de C#](#).

El evento se desencadenará siempre que se renombre un archivo que esté dentro del directorio indicado y además esté activado el escuchador del evento, esta activación se hará mediante la propiedad `EnableRaisingEvents` de la clase `FileSystemWatcher`.

Deberemos suscribir al **evento** `Renamed` un método estático que muestre por consola si ha sido renombrado un archivo, el nombre que tenía antes del cambio y el nuevo nombre.

 **Importante:** recuerda que para que nuestro objeto `FileSystemWatcher` desencadene los eventos deberemos establecer la propiedad `EnableRaisingEvents` a `true`.

✓ Ejercicio 2

Supongamos que un banco quiere que cada vez que se retire dinero de sus cajeros se envíe un **eMail** y un **SMS** al usuario que ha realizado la transacción. Desde el equipo informático se ha diseñado una clase **Cajero** que tendrá:

- Un campo entero que será el número de cajero.
- Un evento al que podremos suscribirnos denominado **RetiradaDeEfectivo** y cuya signatura enviará a los suscriptores el cajero donde se ha producido el evento, el dni del usuario que está realizando la retirada y la cantidad.
- Un método público **RetiraEfectivo** que recibirá el dni del usuario que hace la operación y la cantidad. Además, internamente el método mostrará por la pantalla del cajero (la consola) el mensaje **"Realizando operación..."**, se esperará 2 segundos y mostrará **"Operación realizada con éxito"** y a continuación desencadenará el evento informando a todos los suscriptores de la operación realizada.

Crea dos clases **EnvioEMail** y **EnvioSMS** que implementen el método

EnviarAvisoRetiradaDeEfectivo mediante el cual se suscriban a la publicación del evento por parte del objeto cajero y muestren por pantalla.

- **"Buscando datos usuario <dni> ..."**
- **"Enviando <eMain / SMS> al usuario <dni> de retirada de <cantidad> el día <fecha> a las <hora> horas."**

Por último, crea un programa principal donde instancias objetos de las tres clases anteriores y pruebes que el evento es enviado correctamente.

Ejercicio 3

Cuando en un programa de consola pulsamos **Control + C** se produce un evento de cancelación y un programa finaliza. Además, esta combinación de teclas no es detectada por un **Console.ReadKey(...)** o un **Console.ReadLine()**, etc. Sin embargo, existe un evento especial de Consola llamado **CancelKeyPress**, al que podremos suscribirnos para detectar dicha pulsación.

- Vamos a crear un clase denominada **ConsoleKeyLog** que contendrá una **lista de cadenas privada**. Donde cada cadena contendrá la descripción de una tecla presionada.
- Además definirá un evento **GestionTeclasPulsadasTrasFinalizar** al que podremos suscribirnos y que se generará cuando el usuario pulse **Control + C** para finalizar el

programa. Dicho evento **pasará como parámetro al suscriptor** la lista de cadenas con las pulsaciones de tecla hasta que se pulso **Control + C** .

- La clase definirá un método público denominado **BucleLogConsola** que se suscribirá al evento de cancelación de la consola mencionado al principio (**CancelKeyPress**) y que irá leyendo pulsaciones de tecla en la consola y las añadirá a la lista, a través del siguiente método de utilidad.

```
static string TextoTecla(ConsoleModifiers m, ConsoleKey t)
{
    string txt = "";
    if ((m & ConsoleModifiers.Alt) == ConsoleModifiers.Alt) txt += "Alt + ";
    if ((m & ConsoleModifiers.Control) == ConsoleModifiers.Control) txt += "Control ";
    if ((m & ConsoleModifiers.Shift) == ConsoleModifiers.Shift) txt += "Shift + ";
    txt += t.ToString();
    return txt;
}
```

Crea un programa principal que, a través de un objeto instanciado de **ConsoleKeyLog** ...

1. Leerá la teclas pulsadas del teclado mediante el método **BucleLogConsola** .
2. Defina un manejador para el evento **GestionTeclasPulsadasTrasFinalizar** que guarde en un fichero de texto denominado **teclas.log** la lista con las teclas pulsadas. (Una por línea).

👉 **Importante:** Para que funcione correctamente el evento de cancelación deberás ejecutar el programa desde la línea de comandos y no desde el termina del VS Code. Por tanto, debes sustituir en **launch.json** **integratedTerminal** por **externalTerminal** .

Ejercicio 4

Inventa una situación similar a la del **ejercicio 2** donde varios objetos suscriptores se suscriban a otro publicador, realizando una acción determinada cuando este último genere un evento. Por ejemplo, el disparo de un evento por parte de un sensor de movimiento que ponga en marcha una sirena y genere una llamada telefónica a la policía (simulada con una salida por consola).