

**Tarea Evaluativa 03**

**Curso: 2021/22**

**Programación de servicios y procesos DAM**

**Joseba Sánchez Romero**

**ÍNDICE**

1. INTRODUCCIÓN 1

2. DIFERENTES APARTADOS DE LA TAREA 1

3. AUTOEVALUACIÓN 1

4. BIBLIOGRAFÍA 1

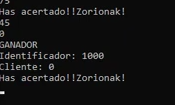
# INTRODUCCIÓN

# DIFERENTES APARTADOS DE LA TAREA

# AUTOEVALUACIÓN

**Principales problemas encontrados:**

**Tarea 1:**

* La posición de la ip address en el array de AddressList (IPAddress ipAddress = ipHostInfo.AddressList[3];). En cada ordenador me aparecía en una posición diferente, hasta que he usado la ip 127.0.0.1 y funciona en todos.
* Los condicionales *if* tienen que ser *else if* para que no se ejecuten todos en el multihilo y solo vaya por la condición que se cumple.
* Que se pase por consola el número 0 como en la explicación de la tarea, lo he hecho igual para hacerlo como el vídeo pero no sé por qué.
* Asegurar con un booleano que solo entre una vez para escribir el nombre del ganador en el servidor cada vez que un cliente sigue metiendo números. Si no, lo escribe un par de veces.

**Tarea 2:**

* Mostrar al cliente la ruta donde ha guardado el fichero: ((System.IO.FileStream)fileStream).Name

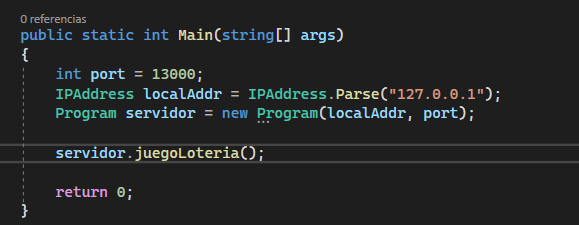
**Funcionalidad:**

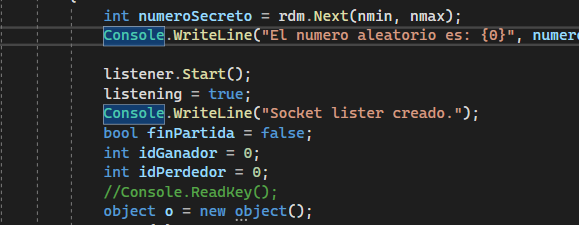
**Tarea 1:**

[**https://www.youtube.com/watch?v=aUXq1vWsg3U**](https://www.youtube.com/watch?v=aUXq1vWsg3U)

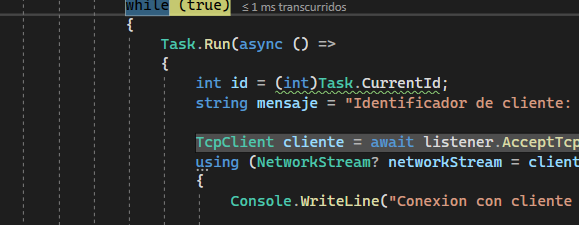
**Lado Servidor**

1. Inicializar variables con datos de conexión y llamada a servidor.juegoLoteria para crear Socket lister y definir número al azar.

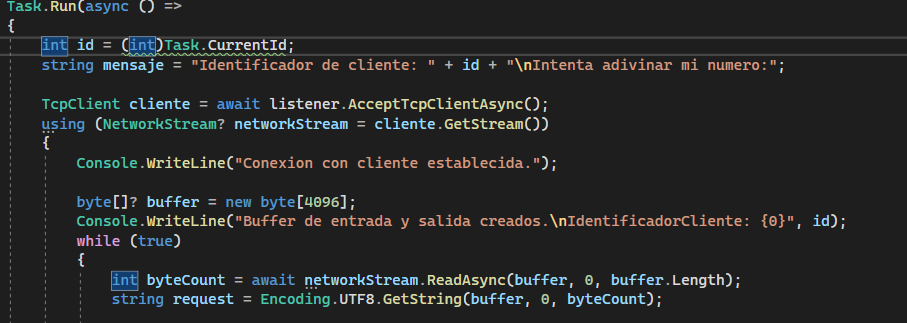




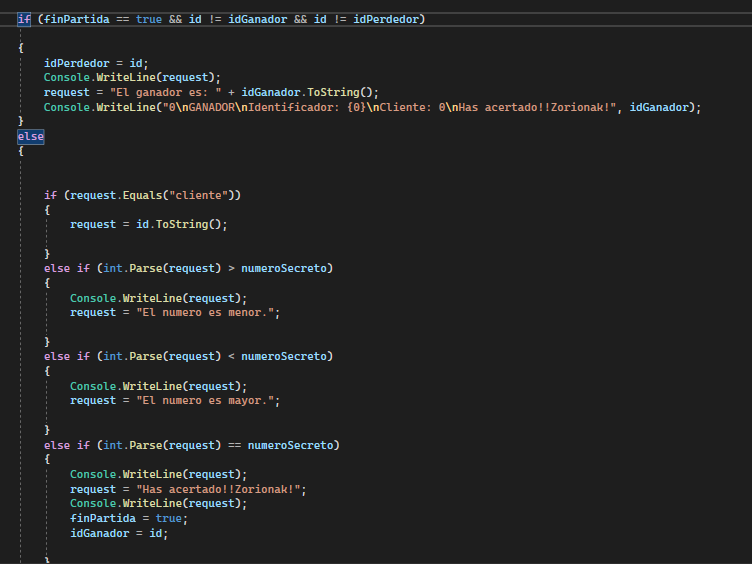
1. Servidor en espera de clientes conectados con while(true).



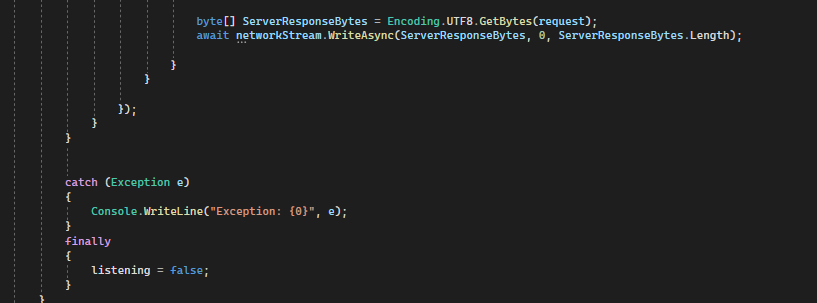
1. Guardar id de la tarea para asignárselo al cliente que se conecte y preparar el texto de bienvenida en el string *mensaje*. Esperar a que se conecte con await listener AccepTcpClientAsync. Usar la conexión creada con el cliente a partir de ahora mediante using NetworkStream. Preparar buffer de intercambio de mensajes y esperar mensaje de cliente para identificarlo.



1. Batería de condicionales if para identificar en qué momento de la partida se encuentran. Lo primero que enviará el cliente será el string con la palabra “cliente” para identificarle y empezar la partida.

****

1. Envío mensajes de servidor a cliente después de pasar por los condicionales y cierre de listener cuando haya ganador.

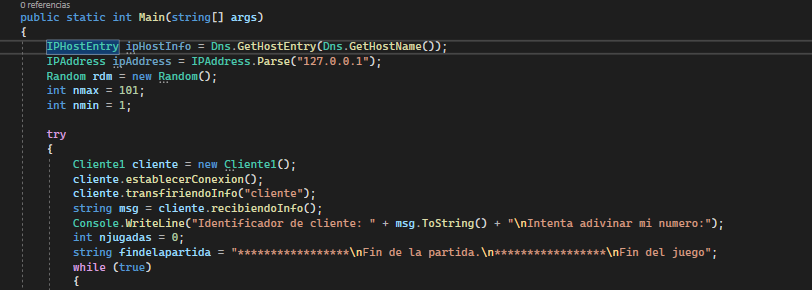


**Lado Cliente**

1. Archivo *Cliente1.cs* donde se encuentran todas las funciones que se llaman desde *Program.cs* para hacer las conexiones en respuesta a lo que hemos visto en el lado Servidor.



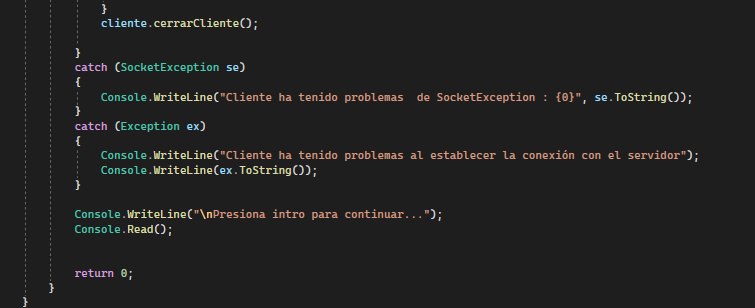
1. Archivo *Program.cs* donde se definen todas las variables a utilizar y se hacen las llamadas a las funciones guardadas en *Cliente1.cs* para hacer las conexiones.



1. Elección de número al azar para proponer al Servidor según las pistas que nos ha dado anteriormente y todos los condicionales preparados para el intercambio de mensajes entre Servidor y Cliente, igual que hemos visto en lado Servidor.



1. Cierre de conexiones cuando haya ganador y catchs.

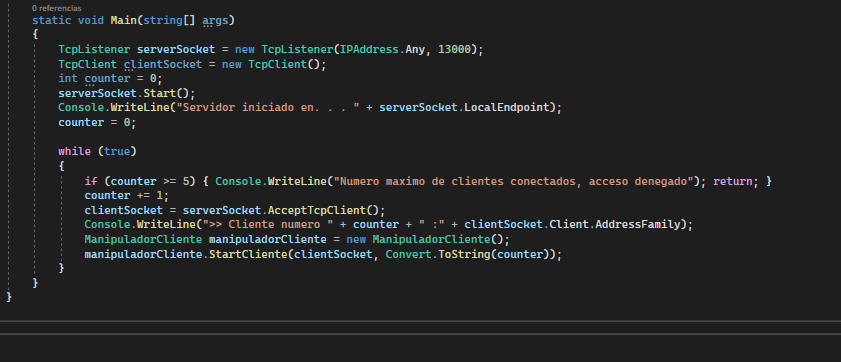
****

**Tarea 2:**

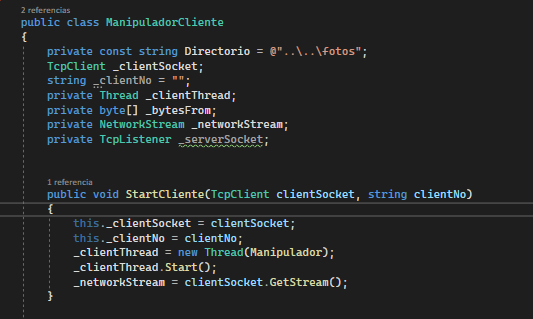
[**https://www.youtube.com/watch?v=WPc06eZO0To**](https://www.youtube.com/watch?v=WPc06eZO0To)

**Lado Servidor**

1. Apertura de socket TcpListener y socket TcpClient. Variable contador para admitir hasta 5 clientes y denegar acceso al resto. Llamada a *ManipuladorCliente* para atender sus peticiones.



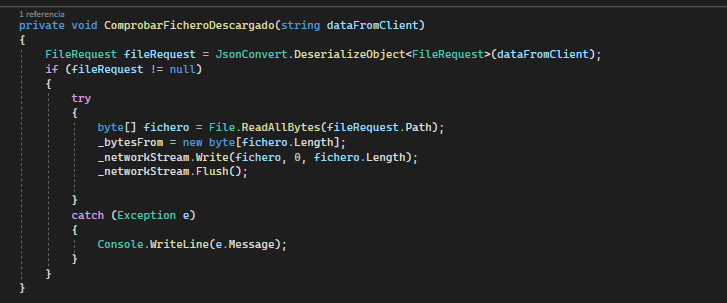
1. Definir directorio donde guardaremos las fotos a ofrecer. Definir hilo, array bytes y resto de variables para la conexión.



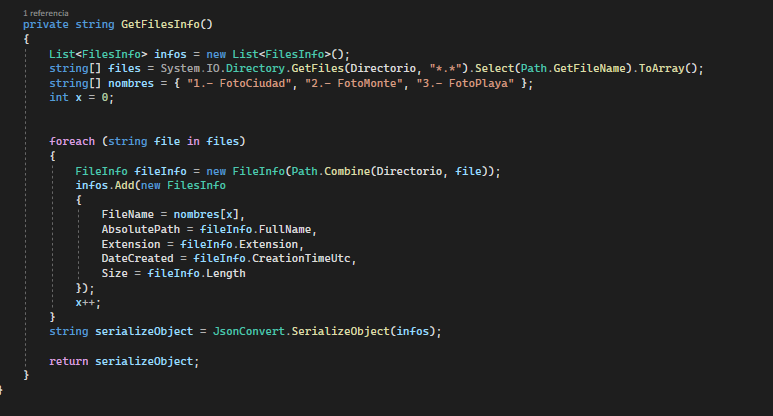
1. Abrir buffer para intercambio de datos, llamada a *ArbolEnvioFichero* para mostrar el árbol con el contenido de los ficheros disponibles y llamada a *ComprobarFicheroDescargado*.



1. Función *ComprobarFicheroDescargado* para abrir buffer con el tamaño de la imagen a descargar.

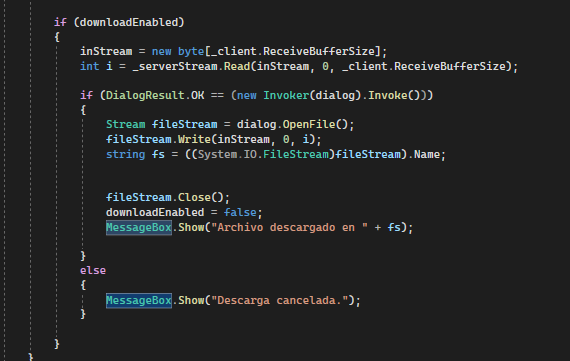


1. Función *GetFilesInfo* que es llamada desde *ArbolFichero* para mostrar los datos aunque solo utilizaremos los nombres definidos en el array string *nombres*.



**Lado Cliente**

1. Aplicación de Windows Forms con los típicos archivos para definir la IU y sus funcionalidades. En una de las funciones guardadas en Form1.cs definimos el mensaje para mostrar al usuario después de guardar el archivo con el path de guardado o informando de la cancelación de la descarga.



# BIBLIOGRAFÍA