This is a C# code that utilizes the Windows.Devices.Bluetooth library to interact with Bluetooth Low Energy (BLE) devices. Specifically, this program is designed to connect to a BLE device that implements the Fitness Machine Service (UUID: 0x1826) and control its resistance remotely.

The program begins by defining the UUIDs of the Fitness Machine Service and the BLE device to connect to. Then, it creates a DeviceWatcher object that listens for nearby BLE devices that are not yet paired. The program registers event handlers for the device watcher that get called when a new device is found, an existing device is updated, or a device is removed.

After starting the device watcher, the program enters a while loop and waits for a device with the specified UUID to be found. Once the device is found, the program connects to it using BluetoothLEDevice.FromIdAsync() method and gets a list of GATT services that the device offers by calling GetGattServicesAsync() method. The program then iterates through the list of services and checks if the Fitness Machine Service is supported by comparing the UUIDs.

If the Fitness Machine Service is supported, the program gets a list of characteristics that the service offers and iterates through them to find the characteristic that allows for changing the resistance level. Once the correct characteristic is found, the program checks if it supports the "Indicate" property, which means it can be subscribed to for notifications. If it does, the program subscribes to the characteristic by calling WriteClientCharacteristicConfigurationDescriptorAsync() method and passing GattClientCharacteristicConfigurationDescriptorValue.Indicate as a parameter.

The program also checks if the characteristic supports the "Write" property, which means it can be written to. If it does, the program writes a value of 0x00 to the characteristic to unlock access control. After this, the program enters a while loop that waits for user input. If the user inputs a number between 1-100, the program writes the corresponding value to the characteristic to set the resistance level. If the user inputs "X", the program exits.

**More Specific.**

1. The program starts by defining two service IDs as constants. These are the IDs for the services provided by the BLE device that the program is going to communicate with.
2. The program then creates a DeviceWatcher object to scan for nearby BLE devices that are not currently paired with the host device. When a nearby device is detected, the program subscribes to the Added, Updated, and Removed events of the DeviceWatcher.
3. The main thread then goes into a loop where it waits for a device to be found by the DeviceWatcher. When a device is found, the program pairs with the device using its DeviceInformation object, which was obtained from the DeviceWatcher.
4. After successfully pairing with the device, the program proceeds to obtain the list of GATT (Generic Attribute Profile) services offered by the device. It then searches for the Fitness Machine Service with the UUID of "00001826-0000-1000-8000-00805f9b34fb", and the Indoor Cycling Service with the UUID of "ee0c".
5. Once the program finds the Indoor Cycling Service, it searches for a specific characteristic within that service with the UUID of "00002ad9-0000-1000-8000-00805f9b34fb". It then checks if this characteristic supports both indicating and writing properties.
6. If the characteristic supports indicating, the program subscribes to the ValueChanged event of the characteristic and writes the GattClientCharacteristicConfigurationDescriptorValue.Indicate to the CCCD (Client Characteristic Configuration Descriptor) of the characteristic.
7. If the characteristic supports writing, the program writes a value of 0x00 to the characteristic using a DataWriter object.
8. After successfully configuring the characteristic, the program waits for user input from the console. If the user enters a number between 1-100, the program converts the input to a byte array and writes it to the characteristic using another DataWriter object.
9. The values that user want to enter will be converted to byte by the program.
10. If the user enters "X", the program breaks out of the loop and ends.

That's the basic flow of the program! It's essentially a BLE client that connects to a BLE device and writes values to a specific characteristic on that device.

**NOTE**

* The device has the flag of indicate and write which means the before writing the values to the device the indication has to be turned on so the device knows that someone is attempting to write
* And follow the procedure to write the values.
* The OP code the device requires to write values is
  + 0x00 to get access of the control
  + 0x01 to reset the values
  + 0x04 to write the resistance value, followed by the value that the user wants to write for example,
    - Writebyte 0x04
    - Writebyte 0x00, or 0x64 (byte values 64 is 100 and 00 is 0)
* Follow the FTMS documentation provided to get more information about other characteristic and services incase if you want to write values to other properties too.
  + That documentation explains the procedure and the OP code that has to be followed.

**(GATT Document)**

Read the document named as 16bit Uuid if you want to work with other characteristics too. The document has all the SIG adopted characteristics and the services.

You can follow this video guide

<https://www.youtube.com/watch?v=AokDN6r4iz8>