WEB COUNTER API

I am sending you two different Visual Studio Solutions:

1. Counter API which was developed in .Net Core 2.1
2. Calling application made by two projects:
3. cordova cross platform, and
4. MVC Web Project.
5. API

The task for Counter API was clear to me. However, no further instructions were provided after the second call. Therefore, I deliberated an assumption that there may be many possibilities but I limit myself to only three options I explain below.

1. The Counter repeats itself as from the beginning. In other words, on a third call it would increase by 1; on a fourth call twice in a row by 1, and so on. The outcome is below:

1st call 0 + 1 = 1

2nd call 1 + 2 = 3

3rd call 3 + 1 = 4

4th call 4 + 8 = 12

5th call 12 + 1= 13

And so on and so forth.

1. The counter always increments with a new value. Thus, when called third, time the new value becomes relevant and increases to 9. In other words, at each subsequent call increment current value is increased by 3. The illustration is as follows:

1st call 0 + 1 = 1 3^0

2nd call 1 + 2 = 3 3^1

3rd call 3 + 6 = 9 3^2

4th call 9 + 18= 27 3^3

5th call 27 + 54= 81 3^4

And so on and so forth.

1. The increment on the Counter is always 1. This means that after the second call the increment always rise by 1, so that upon the third call the increment current value is tripled, or on the fourth call quadrupled. The result is the following:

1st call 0 + 1 = 1

2nd call 1 + 2 = 3

3rd call 3 + 9 = 12

4th call 12 + 48 = 60

5th call 60 + 300= 360

And so forth and so on.

I have presented different options but it is my option that the option one would be the most advantageous to adopt. My detailed explanation and the computation of all three options is further elaborated below.

1. MVC/Mobile Web Project

MVC/Mobile Web Project consists of two parts. In fact, both are identical projects but the one is written in Cordova Cross Platform, whist the other one is written in ASP MVC. I used Material Design as a base for mobile (https://material.angularjs.org) and web controls (<https://materializecss.com>); if requested these could be done Bootstrap controls. I did not wish to make Windows Desktop Project because I wanted to keep it in web – mobile environment. However, it could be easily done if desired (WebResponse and HttpWebRequest Classes in System.Net namespace would be sufficient to get data from the service).

SET UP

1. In Counter API folder you will find publish folder. In that folder you will find all necessary files to run API service. I used local IIS and set it up to work on port 5000 that is why in application you may see this address as a default address.
2. Web Counter is MVC application and also can be hosted on a local IIS or it can be run directly from project. Just in a web.config file there is <add key="ServiceURL" value="http://localhost:5000/"/> which can be changed according to your service address.
3. Mobile application can be run from the project and can be tested on Ripple emulators of Chrome Browser. It can also be run directly on android device if you wish to see how it looks like on mobile phone.

USAGE

1. Counter API is consisted of three main functions.
   1. ReadCounter() ([Route(“api/GetCounter”)]).

This function reads counter value. It is public function and accessible for all calls. It returns

public class CounterResult

{

private int countervalue;

private int incrementcalls;

private string memoaddress;

public int CounterValue {

get { return countervalue; }

set { countervalue = value; }

}

public int IncrementCalls {

get { return increasmentcalls; }

set { incrementcalls = value; }

}

public string MemoryAddress {

get { return memoaddress.NullIF(); }

set { memoaddress = value; } }

}

* 1. SetCounter() ([Route(“api/GetCounter”)]).

This function increases counter value.

* 1. ResetCounter() ([Route(“api/ResetCounter”)]).

This function reset counter.

1. Once opened any of the application you will see 3 buttons.
   1. Button to read counter value
   2. Button to increase counter value
   3. Button to reset counter

Any time one of these functions is called application leaves option for a user to retrieve counter value. On screen there are three values shown

* Counter Value
* Numbers Of Increment Of Counter
* Address Of Heap Memory that counter value is stored.