ACE Inspiration

# Web Application Development with ASP.Net Core 3.1 MVC 5

ASP.NET Core MVC Web Application Framework– Chapter 3

## Objectives

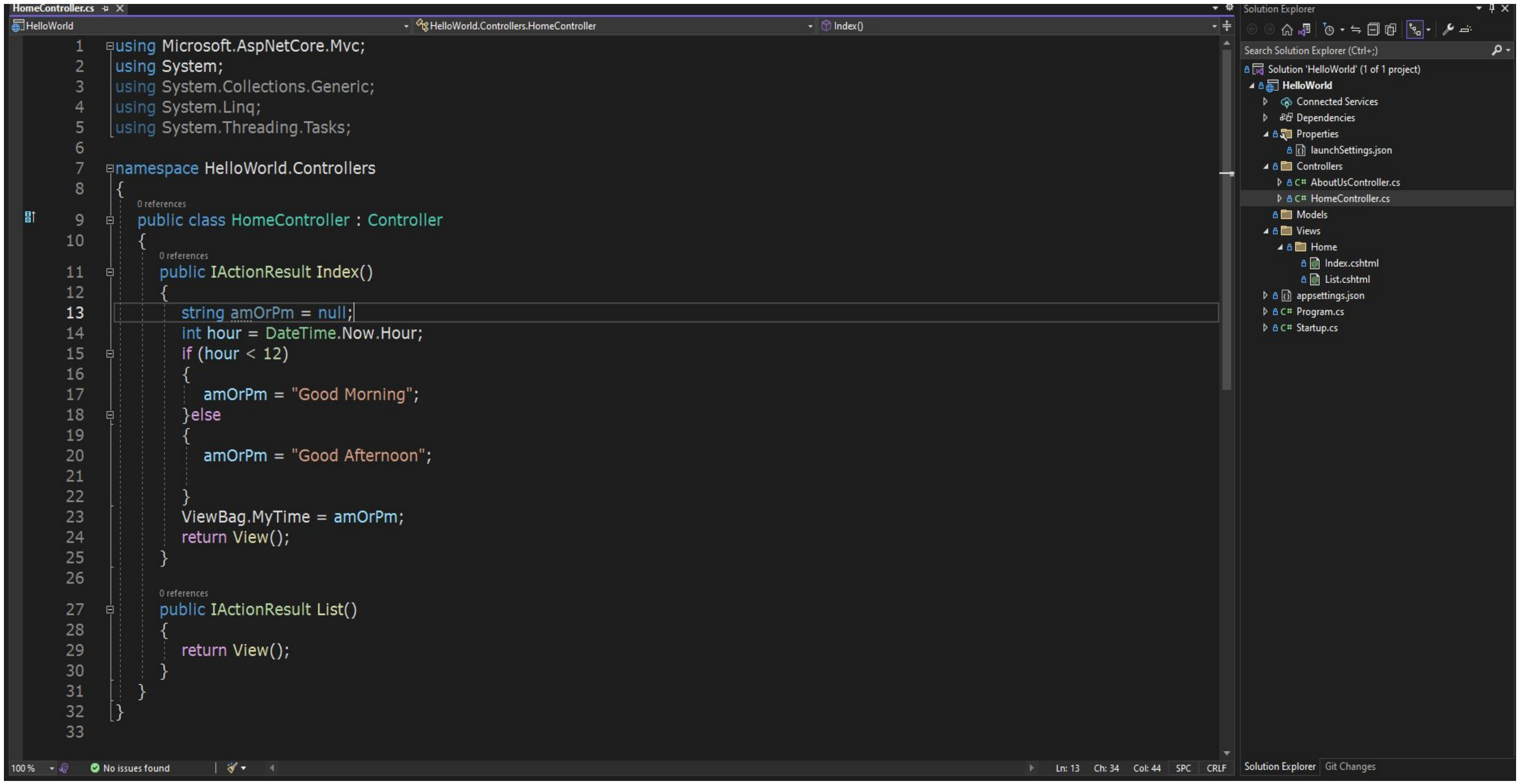
- After this lesson, you should be able to understand
  - Routing
  - Controller
  - Action
  - Action Filter
  - Action Selector
  - Passing data from Controller to View
    - ViewBag
    - ViewData
    - TempData

## ASP.Net Core MVC Routing

- The ASP.NET Routing module is responsible for mapping incoming browser requests to particular MVC controller actions.
- For example, http://servername/Home/Index
- eg http://servername/**{controller}/{action}**)
  - **Home** is the first segment (Controller segment)
  - **Index** is the second segment (Action segment)

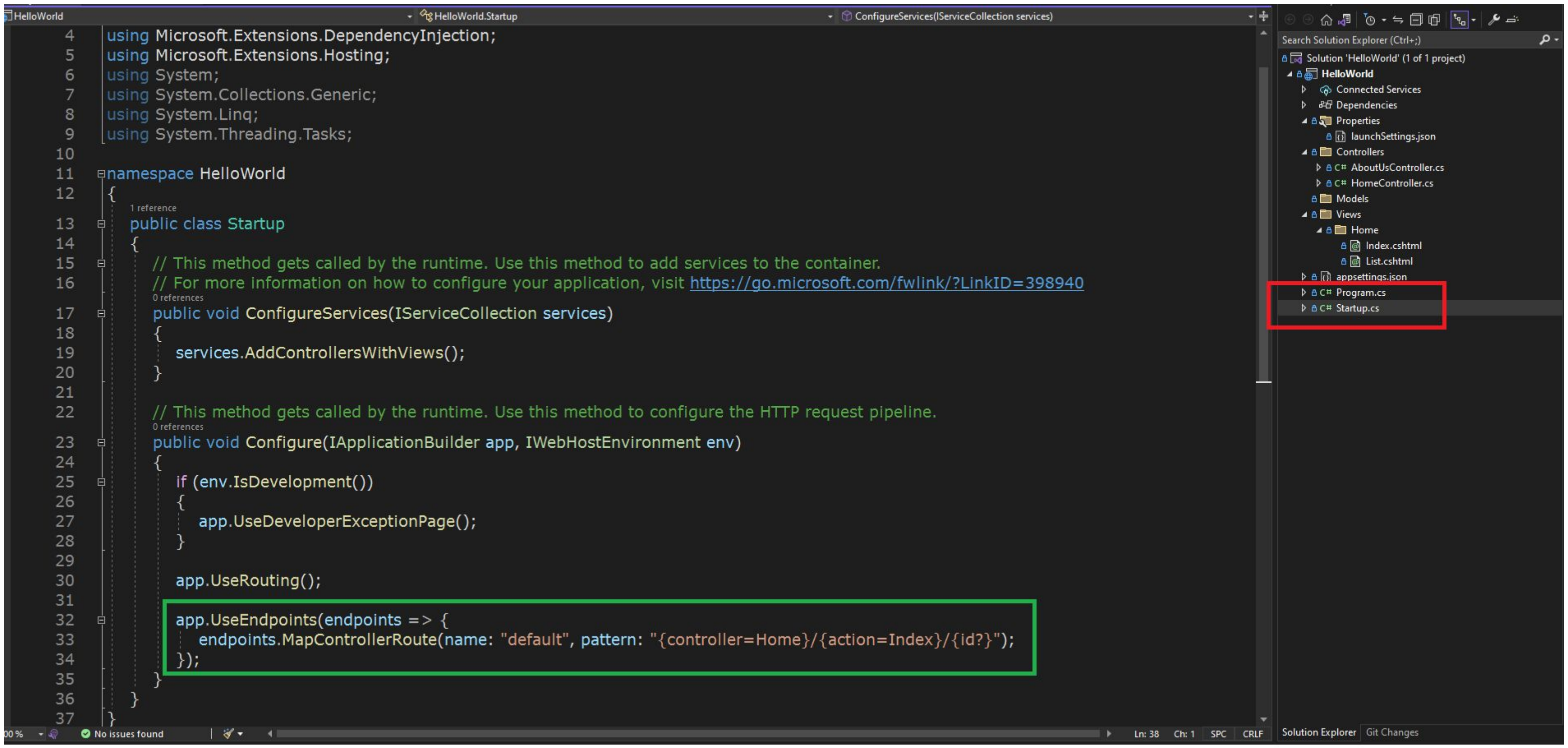
- **Startup. cs(v3.1) or Program.cs(v5+)** file is that part of your application.
- It can called **middleware of your application**.

# ASP.Net Core MVC Routing



```
using Microsoft.AspNetCore.Mvc;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace HelloWorld.Controllers
{
    public class HomeController : Controller
    {
        public IActionResult Index()
        {
            string amOrPm = null;
            int hour = DateTime.Now.Hour;
            if (hour < 12)
            {
                amOrPm = "Good Morning";
            }else
            {
                amOrPm = "Good Afternoon";

            }
            ViewBag.MyTime = amOrPm;
            return View();
        }

        public IActionResult List()
        {
            return View();
        }
    }
}
```

# ASP.Net Core(v3.1) MVC Routing

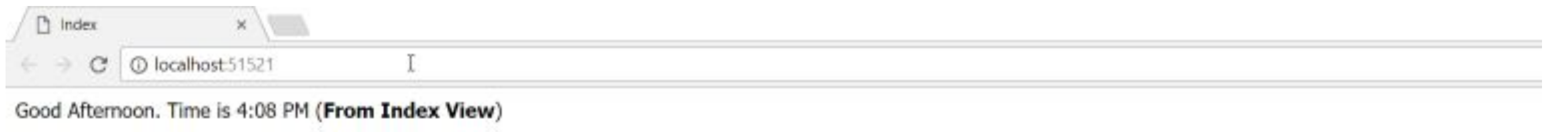


```
using Microsoft.Extensions.DependencyInjection;
using Microsoft.Extensions.Hosting;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace HelloWorld
{
    public class Startup
    {
        // This method gets called by the runtime. Use this method to add services to the container.
        // For more information on how to configure your application, visit https://go.microsoft.com/fwlink/?LinkID=398940
        public void ConfigureServices(IServiceCollection services)
        {
            services.AddControllersWithViews();
        }

        // This method gets called by the runtime. Use this method to configure the HTTP request pipeline.
        public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
        {
            if (env.IsDevelopment())
            {
                app.UseDeveloperExceptionPage();
            }

            app.UseRouting();

            app.UseEndpoints(endpoints => {
                endpoints.MapControllerRoute(name: "default", pattern: "{controller=Home}/{action=Index}/{id?}");
            });
        }
    }
}
```

```
Solution 'HelloWorld' (1 of 1 project)
  HelloWorld
    Connected Services
    Dependencies
    Properties
      launchSettings.json
    Controllers
      AboutUsController.cs
      HomeController.cs
    Models
    Views
      Home
        Index.cshtml
        List.cshtml
    appsettings.json
    Program.cs
    Startup.cs
```

# ASP.Net Core MVC Routing

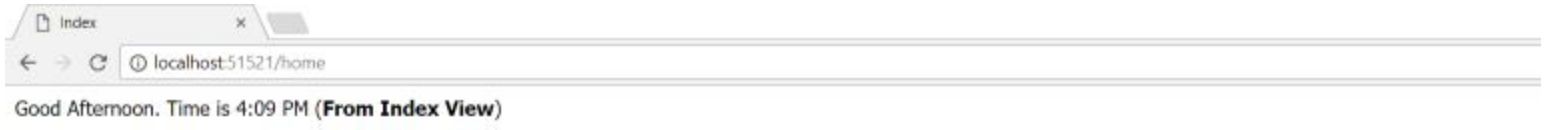
URL - http://localhost:51521 – http://ipaddress:port



Good Afternoon. Time is 4:08 PM (**From Index View**)

## ASP.Net Core MVC Routing

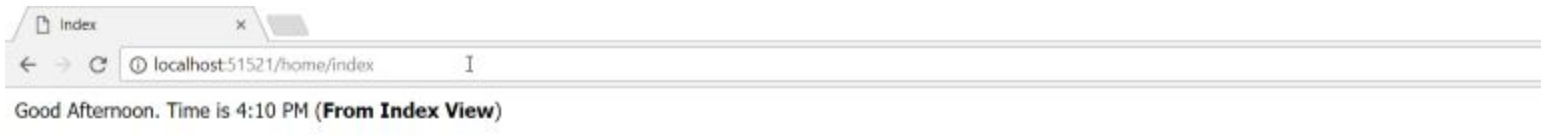
URL - http://localhost:51521/home – http://ipaddress:port/controller



Good Afternoon. Time is 4:09 PM (**From Index View**)

# ASP.Net Core MVC Routing

URL - http://localhost:51521/home/index – http://ipaddress:port/controller/action



Good Afternoon. Time is 4:10 PM (**From Index View**)

## What does Controller do?

- Central unit of your ASP.NET Core MVC application
- 1st front recipient, which interacts with incoming HTTP Request
- Decides which model will be selected, and then it takes the data from the model and passes the same to the respective view, after that, view is rendered
- Controlling the overall flow of the application taking the input and rendering the proper output

## What does Controller do?

- C# classes inheriting from System.Web.Mvc.Controller, which is the built-in controller base class
- Each public method in a controller is known as an action method, meaning you can invoke it from the Web via some URL to perform an action
- ASP.NET MVC convention is to put controllers in the Controllers folder that Visual Studio created when the project was set up.

## What does Actions do?

- ASP.NET MVC Action (public) Methods are responsible to execute requests and generate responses to it

- By default, it generates a response in the form of IActionResult.

- Controllers define action methods and these action methods generally have a **one-to-one mapping** with user interaction such as clicking a button or a link, etc.

# Actions – Type of ActionResults

- Actions basically return different types of action results.
- The ActionResult class is the base for all action results.

| **Result Class** | **Description** |
|---|---|
| **ViewResult** | Represents HTML and markup. |
| EmptyResult | Represents No response. |
| **ContentResult** | Represents string literal. |
| FileContentResult/ FilePathResult/ FileStreamResult/FileResult | Represents the content of a file |
| JavaScriptResult | Represent a JavaScript script. |
| JsonResult | Represent JSON that can be used in AJAX |
| RedirectResult | Represents a redirection to a new URL |
| RedirectToRouteResult | Represent another action of same or other controller |
| PartialViewResult | Returns HTML from Partial view |
| HttpUnauthorizedResult | Returns HTTP 403 status |

## Actions

- **Let's exercise a little.**
  - Return view
  - Return static string
  - Return calculated/dynamic string
  - Number Multiplication with **Action Parameters**
  - String Concatenation with **Action Parameters**
  - FileResult with File Download

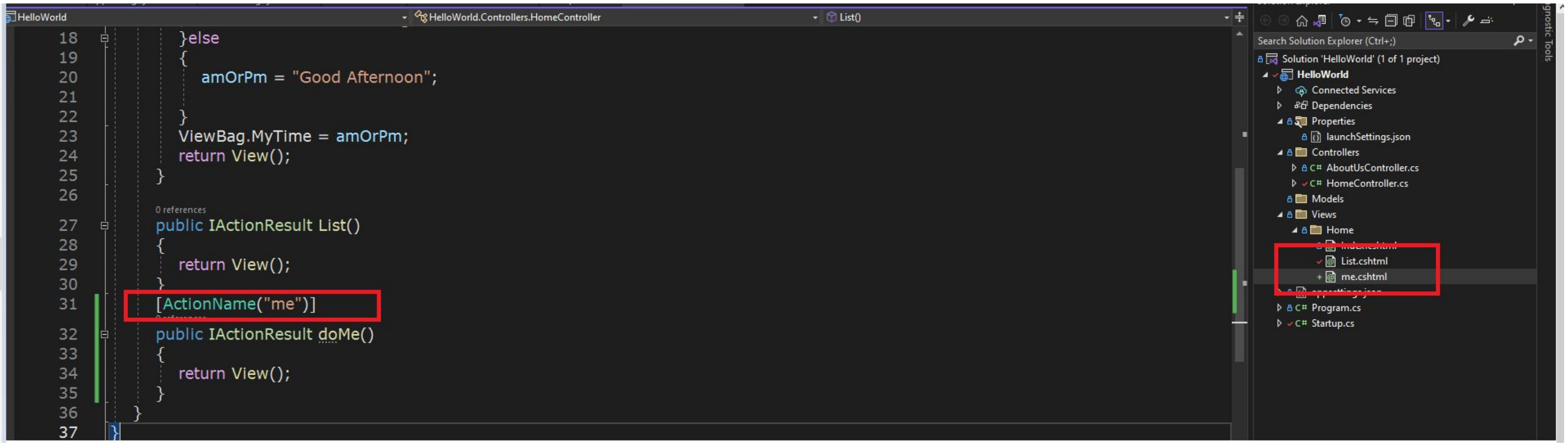
https://www.c-sharpcorner.com/article/fileresult-in-asp-net-core-mvc2/

## Action Selector

- Attributes that can be applied to action methods.
- Helps engine to select the correct action method to handle a particular request.
- There are **three types of action selector attributes** –
  - ActionName
  - NonAction
  - ActionVerbs
    - HttpGet
    - HttpPost
    - HttpDelete
    - HttpPut
    - **…more verbs**

# Action Selector - ActionName

- **Allows us to specify a different action name (OR) method name**



HelloWorld | HelloWorld.Controllers.HomeController | List()

```
                }else
                {
                    amOrPm = "Good Afternoon";

                }
                ViewBag.MyTime = amOrPm;
                return View();
            }

            0 references
            public IActionResult List()
            {
                return View();
            }
            [ActionName("me")]
            public IActionResult doMe()
            {
                return View();
            }
        }
}
```

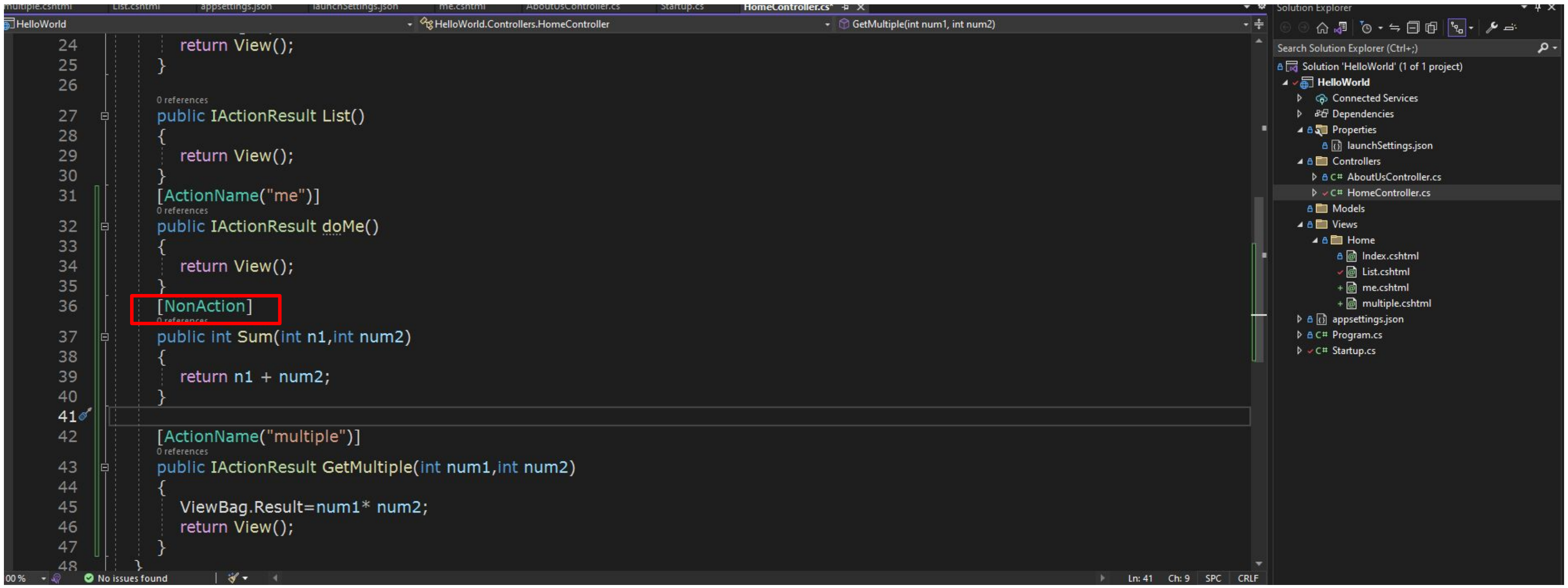
Search Solution Explorer (Ctrl+;)

```
Solution 'HelloWorld' (1 of 1 project)
  HelloWorld
    Connected Services
    Dependencies
    Properties
      launchSettings.json
    Controllers
      AboutUsController.cs
      HomeController.cs
    Models
    Views
      Home
        Index.cshtml
        List.cshtml
        me.cshtml
    appsettings.json
    Program.cs
    Startup.cs
```

## Action Selector - **NonAction**

- Indicates that a public method of a Controller is not an action method
- Use **NonAction** attribute when you want public method in a controller but do not want to treat it as an action method

# Action Selector - NonAction



multiple.cshtml | List.cshtml | appsettings.json | launchSettings.json | me.cshtml | AboutUsController.cs | Startup.cs | HomeController.cs*

HelloWorld | HelloWorld.Controllers.HomeController | GetMultiple(int num1, int num2)

```
            return View();
        }

        0 references
        public IActionResult List()
        {
            return View();
        }
        [ActionName("me")]
        0 references
        public IActionResult doMe()
        {
            return View();
        }
        [NonAction]
        0 references
        public int Sum(int n1,int num2)
        {
            return n1 + num2;
        }

        [ActionName("multiple")]
        0 references
        public IActionResult GetMultiple(int num1,int num2)
        {
            ViewBag.Result=num1* num2;
            return View();
        }
    }
```

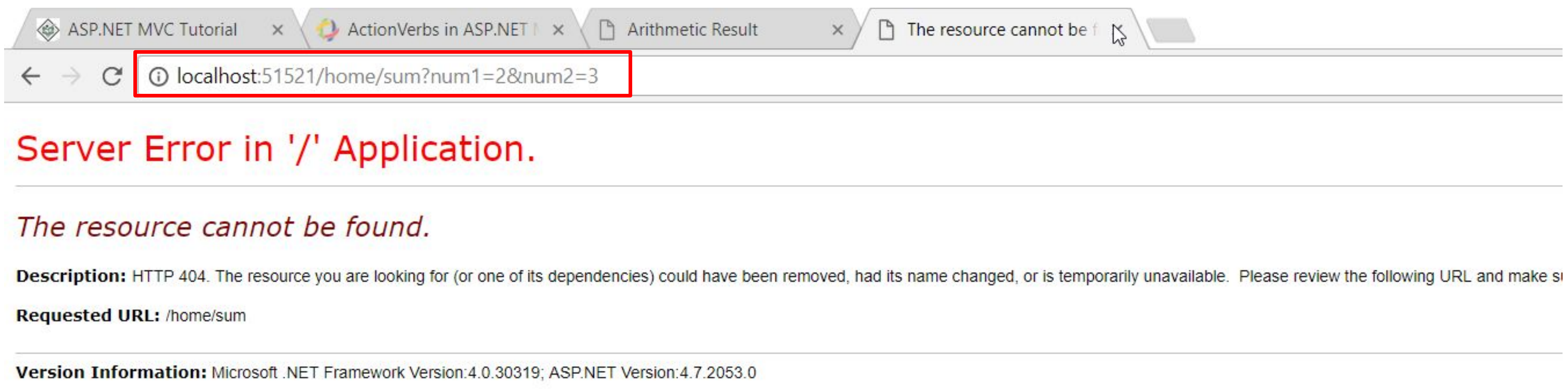
00 % | No issues found | Ln: 41 | Ch: 9 | SPC | CRLF

Solution Explorer

Search Solution Explorer (Ctrl+;)

```
Solution 'HelloWorld' (1 of 1 project)
  HelloWorld
    Connected Services
    Dependencies
    Properties
      launchSettings.json
    Controllers
      AboutUsController.cs
      HomeController.cs
    Models
    Views
      Home
        Index.cshtml
        List.cshtml
        me.cshtml
        multiple.cshtml
    appsettings.json
    Program.cs
    Startup.cs
```

# Action Selector - NonAction



localhost:51521/home/sum?num1=2&num2=3

Server Error in '/' Application.

*The resource cannot be found.*

**Description:** HTTP 404. The resource you are looking for (or one of its dependencies) could have been removed, had its name changed, or is temporarily unavailable. Please review the following URL and make su

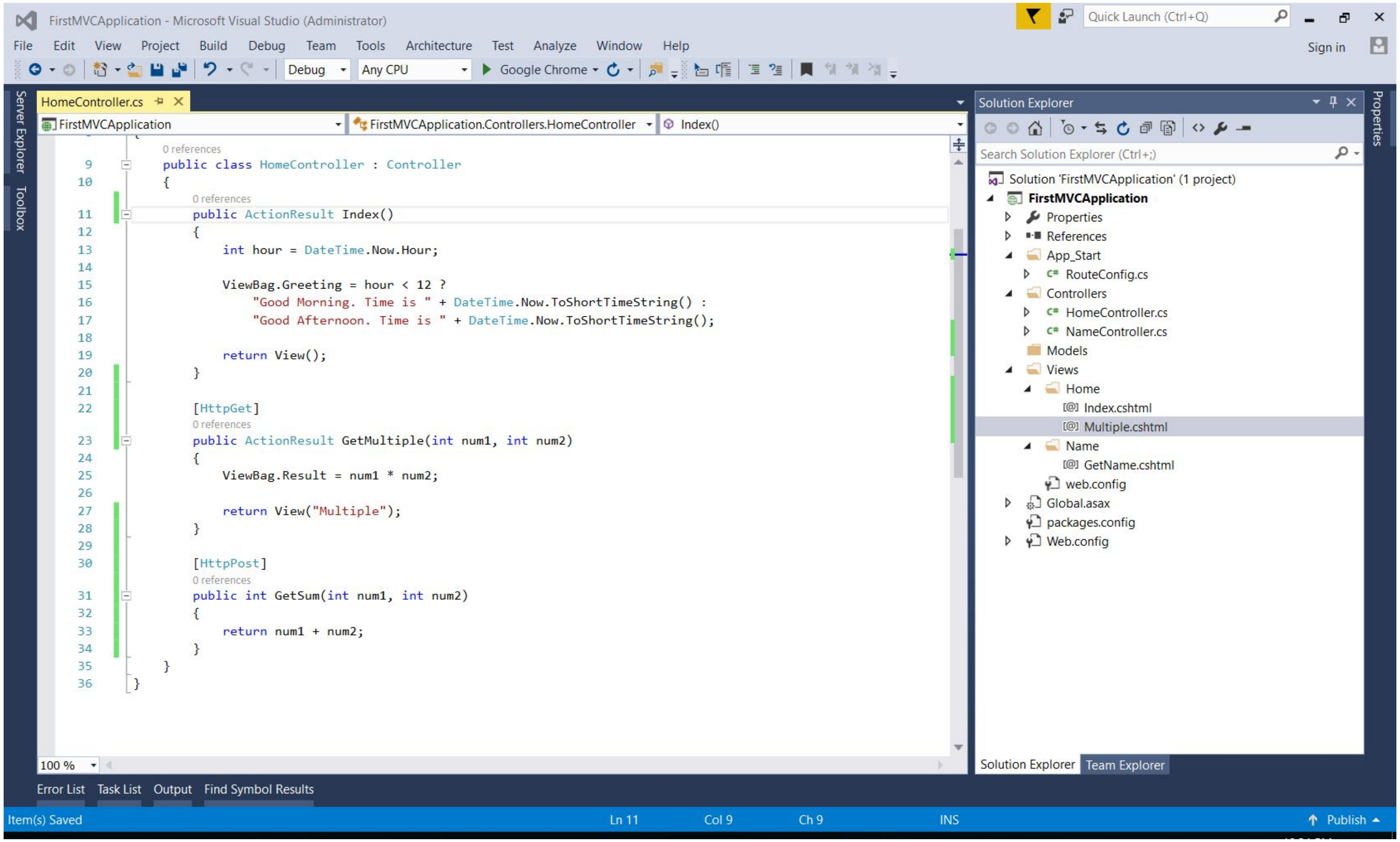
**Requested URL:** /home/sum

**Version Information:** Microsoft .NET Framework Version:4.0.30319; ASP.NET Version:4.7.2053.0

# Action Selector - ActionVerbs

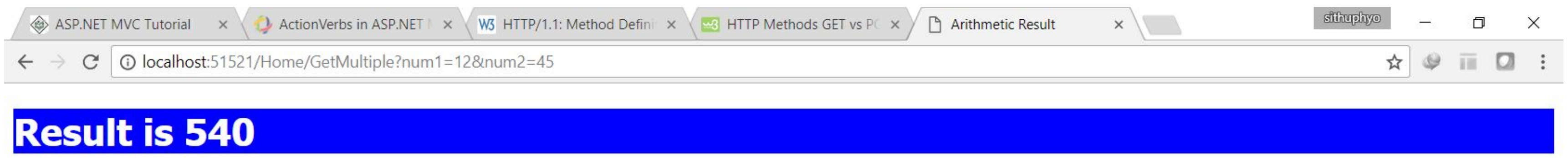
- Http Request Method
  - HttpGet
    - To retrieve the information from the server. Parameters will be appended in the query string
    - Requests data from a specified resource
  - HttpPost
    - To create or update a new resource
    - Submits data to be processed to a specified resource
  - …
- ActionVerbs selector is used when you want to control the request method of an action
- 
- If you do not apply any attribute then it considers it **a GET request by default.**

# Action Selector - ActionVerbs



```
0 references
public class HomeController : Controller
{
    0 references
    public ActionResult Index()
    {
        int hour = DateTime.Now.Hour;

        ViewBag.Greeting = hour < 12 ?
            "Good Morning. Time is " + DateTime.Now.ToShortTimeString() :
            "Good Afternoon. Time is " + DateTime.Now.ToShortTimeString();

        return View();
    }

    [HttpGet]
    0 references
    public ActionResult GetMultiple(int num1, int num2)
    {
        ViewBag.Result = num1 * num2;

        return View("Multiple");
    }

    [HttpPost]
    0 references
    public int GetSum(int num1, int num2)
    {
        return num1 + num2;
    }
}
}
```

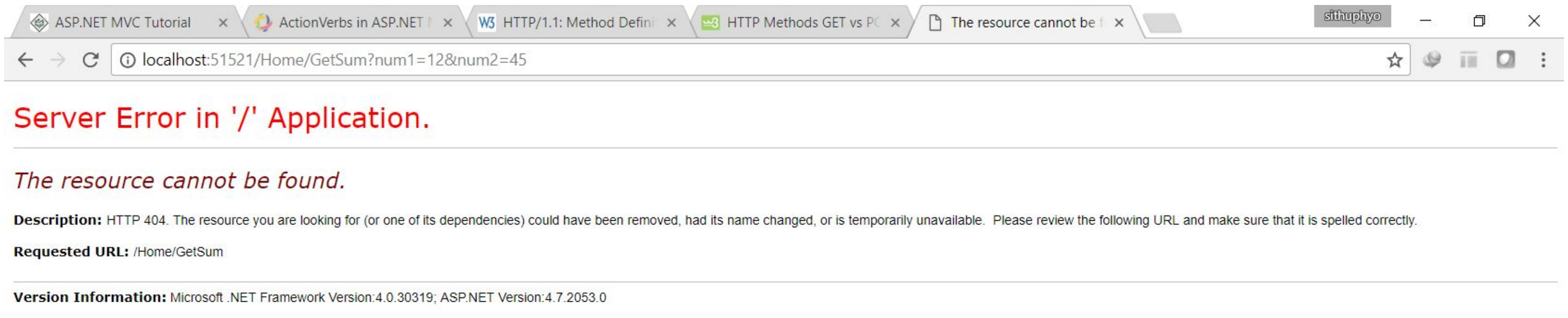
## Action Selector - ActionVerbs


ASP.NET MVC Tutorial | ActionVerbs in ASP.NET | HTTP/1.1: Method Defin | HTTP Methods GET vs P | Arithmetic Result

localhost:51521/Home/GetMultiple?num1=12&num2=45

Result is 540

# Action Selector - ActionVerbs



Server Error in '/' Application.

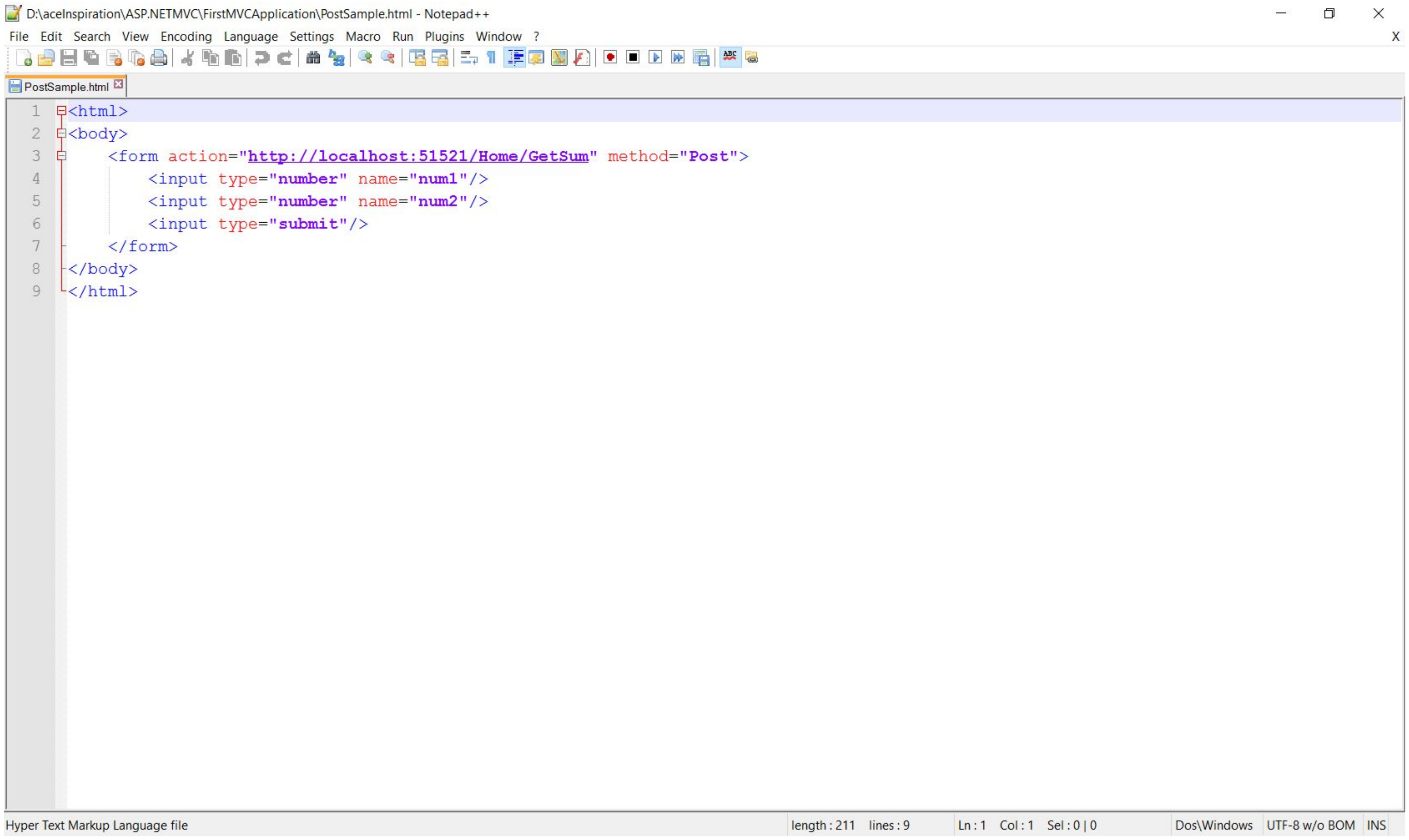
*The resource cannot be found.*

**Description:** HTTP 404. The resource you are looking for (or one of its dependencies) could have been removed, had its name changed, or is temporarily unavailable.  Please review the following URL and make sure that it is spelled correctly.

**Requested URL:** /Home/GetSum

**Version Information:** Microsoft .NET Framework Version:4.0.30319; ASP.NET Version:4.7.2053.0

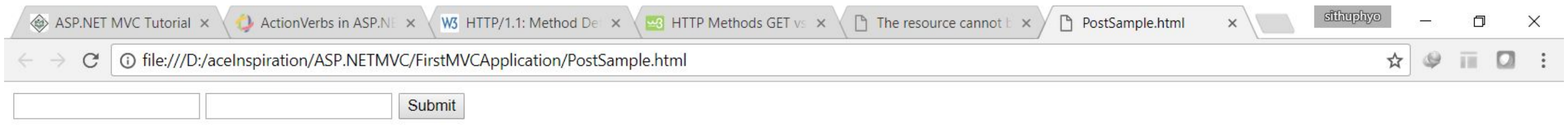
## Action Selector - ActionVerbs


D:\aceInspiration\ASP.NETMVC\FirstMVCApplication\PostSample.html - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
PostSample.html

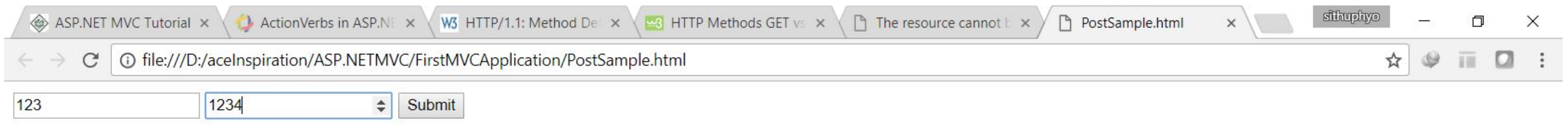
```
<html>
<body>
    <form action="http://localhost:51521/Home/GetSum" method="Post">
        <input type="number" name="num1"/>
        <input type="number" name="num2"/>
        <input type="submit"/>
    </form>
</body>
</html>
```

Hyper Text Markup Language file
length : 211 lines : 9
Ln : 1 Col : 1 Sel : 0 | 0
Dos\Windows UTF-8 w/o BOM INS

## Action Selector - ActionVerbs



## Action Selector - ActionVerbs

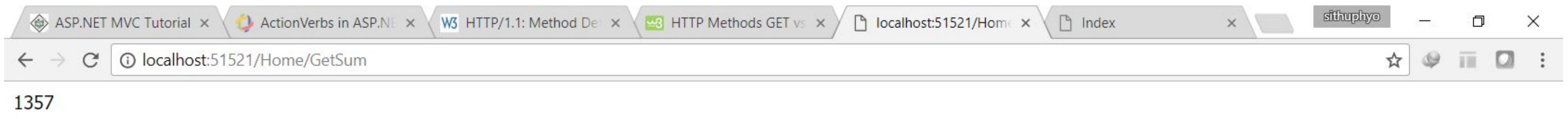


file:///D:/aceInspiration/ASP.NETMVC/FirstMVCApplication/PostSample.html

123

1234

## Action Selector - ActionVerbs



localhost:51521/Home/GetSum

```
1357
```

# Action Selector - ActionVerbs

- HttpGet
  - GET requests **can be cached**
  - GET requests **remain in the browser history**
  - GET requests **can be bookmarked**
  - GET requests **should never be used when dealing with sensitive data**
  - GET requests have **length restrictions**
  - GET requests should be **used only to retrieve data**

- HttpPost
  - POST requests are **never cached**
  - POST requests **do not remain in the browser history**
  - POST requests **cannot be bookmarked**
  - POST requests **have no restrictions on data length**

# HTTP GET Vs POST

ACE Inspiration

| Keys | GET | POST |
|---|---|---|
| BACK button/Reload | Harmless | Data will be re-submitted (the browser should alert the user that the data are about to be re-submitted) |
| Bookmarked | Can be bookmarked | Cannot be bookmarked |
| Cached | Can be cached | Not cached |
| Encoding type | application/x-www-form-urlencoded | application/x-www-form-urlencoded or multipart/form-data. Use multipart encoding for binary data |
| History | Parameters remain in browser history | Parameters are not saved in browser history |
| Restrictions on data length | Yes, when sending data, the GET method adds the data to the URL; and the length of a URL is limited (maximum URL length is 2048 characters) | No restrictions |
| Restrictions on data type | Only ASCII characters allowed. | No restrictions. Binary data is also allowed |
| Security | GET is less secure compared to POST because data sent is part of the URL.<br><br>Never use GET when sending passwords or other sensitive information. | POST is a little safer than GET because the parameters are not stored in browser history or in web server logs |
| Visibility | Data is visible to everyone in the URL. | Data is not displayed in the URL. |

## Action Filter

- We would like to perform some operations before or after a particular action.

- For achieving this functionality, ASP.NET MVC provides feature (**Action Filter**) to add **<u>pre and post action</u>** behaviors on controller's action methods.

ACE Inspiration

## Action Filters

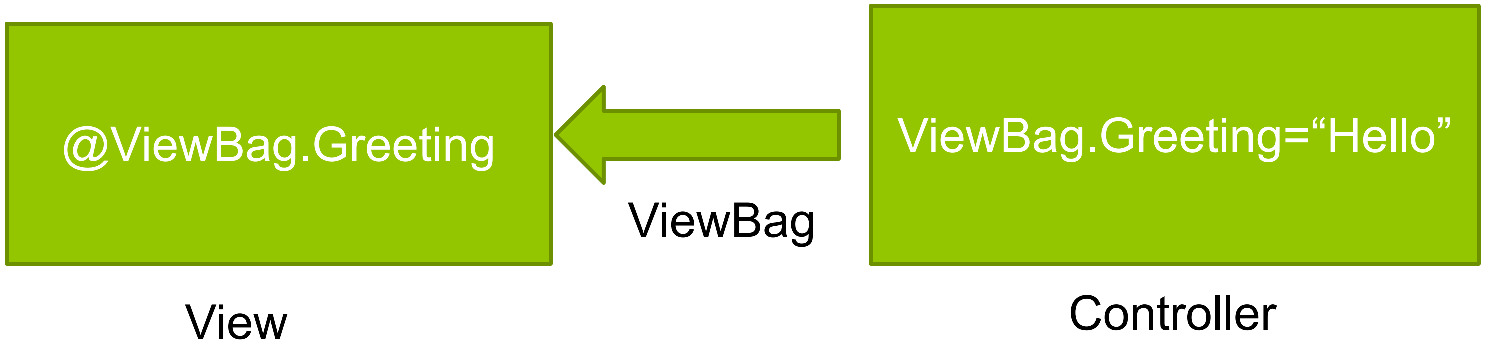
- **Action Filters:** Action filters are used to implement logic that gets executed before and after a controller action executes.
- **Authorization Filters:** Authorization filters are used to implement authentication and authorization for controller actions.
- **Result Filters:** Result filters contain logic that is executed before and after a view result is executed.
- **Exception Filters:** Exception filters are the last type of filter to run. You can use an exception filter to handle errors raised by either your controller actions or controller action results. You also can use exception filters to log errors.

https://code-maze.com/action-filters-aspnetcore/

## Passing data from Controller to View

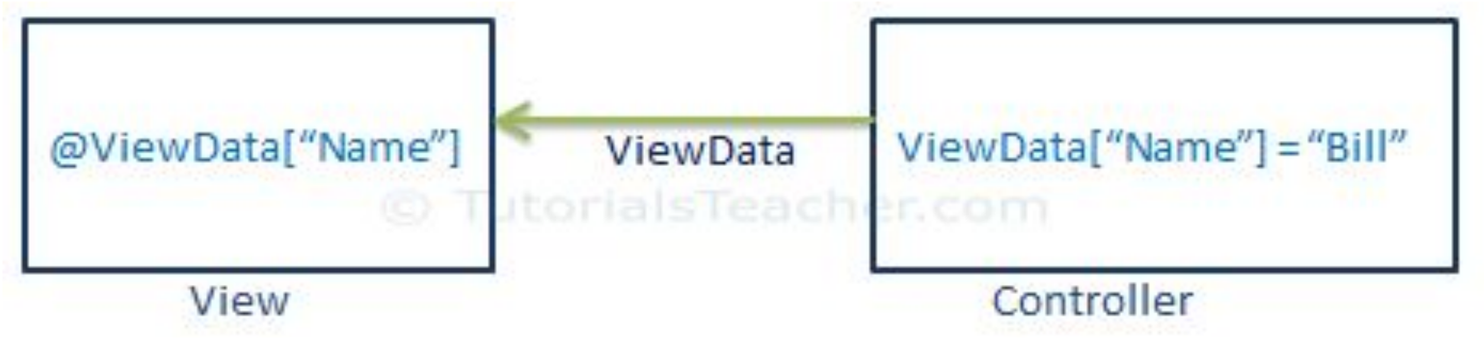
- ViewBag
  - -Transfers data from the controller to the view, ideally temporary data which in not included in a model.
  - -Dynamic property that takes advantage of the new dynamic features in C# 4.0
  - You can assign any number of properties and values to ViewBag
  - ViewBag's life only lasts during the current http request. ViewBag values will be null if redirection occurs
  - - is actually a wrapper around ViewData

  - - is available only for Current Request. It will be destroyed on redirection.



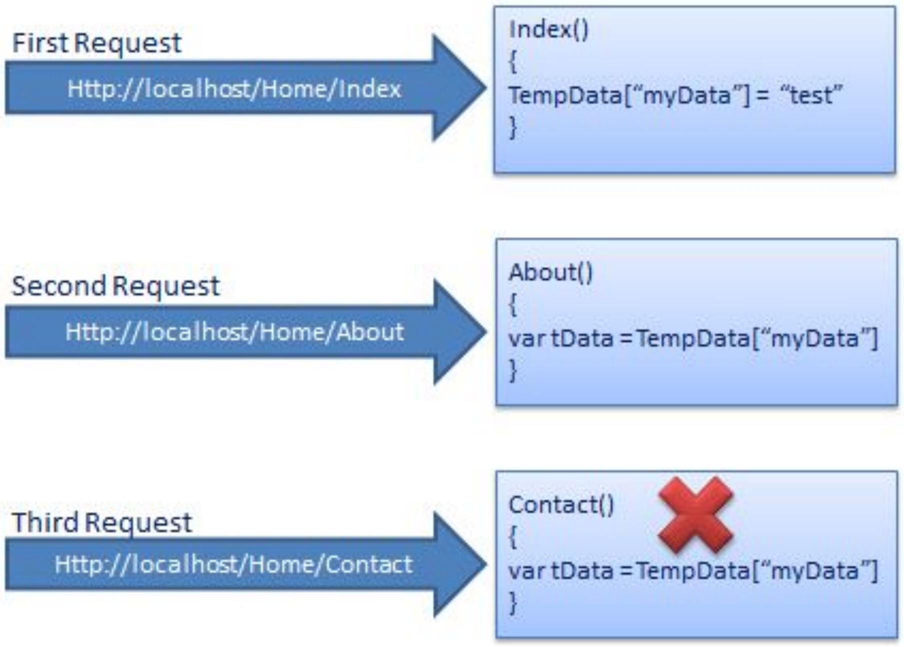
## Passing data from Controller to View

- ViewData
  - - transfers data from the Controller to View, not vice-versa.
  - - is derived from ViewDataDictionary which is a dictionary type.
  - ViewData's life only lasts during current http request. ViewData values will be cleared if redirection occurs.
  - ViewData value must be type cast before use.
  - ViewBag internally inserts data into ViewData dictionary. So the key of ViewData and property of ViewBag must **NOT** match.



## Passing data from Controller to View

- TempData
  - - can be used to store data between **two consecutive requests**. TempData values will be retained during redirection.
  - - is a TempDataDictionary type.
  - - internaly use Session to store the data. So think of it as a short lived session.
  - - can be used for passing value from Controller to View and also from Controller to Controller.


First Request
Http://localhost/Home/Index
Second Request
Http://localhost/Home/About
Third Request
Http://localhost/Home/Contact

```
Index()
{
TempData["myData"] = "test"
}
```

```
About()
{
var tData = TempData["myData"]
}
```

```
Contact()
{
var tData = TempData["myData"]
}
```

## Passing data from Controller to View

| ViewData | ViewBag | TempData |
|---|---|---|
| It is Key-Value Dictionary collection | It is a type object | It is Key-Value Dictionary collection |
| ViewData is a dictionary object and it is property of ControllerBase class | ViewBag is Dynamic property of ControllerBase class. | TempData is a dictionary object and it is property of controllerBase class. |
| ViewData is Faster than ViewBag | ViewBag is slower than ViewData | NA |
| ViewData is introduced in MVC 1.0 and available in MVC 1.0 and above | ViewBag is introduced in MVC 3.0 and available in MVC 3.0 and above | TempData is also introduced in MVC1.0 and available in MVC 1.0 and above. |
| ViewData also works with .net framework 3.5 and above | ViewBag only works with .net framework 4.0 and above | TempData also works with .net framework 3.5 and above |
| Type Conversion code is required while enumerating | In depth, ViewBag is used dynamic, so there is no need to type conversion while enumerating. | Type Conversion code is required while enumerating |
| Its value becomes null if redirection has occurred. | Same as ViewData | TempData is used to pass data between two consecutive requests. |
| It lies only during the current request. | Same as ViewData | TempData only works during the current and subsequent request |

**Note:**

**ViewData and ViewBag are almost similar and it helps us to transfer the data from controller to view whereas TempData also works during the current and subsequent requests.**

Thank you!!
Q&As

## Reference links

- https://www.c-sharpcorner.com/article/fileresult-in-asp-net-core-mvc2/

- https://www.youtube.com/watch?v=5a9sP0EOTxo&list=PLLN1EhtVIKgldpXOXxWw-57pI_NsiPejv