(T18)討論 ValidateInputAttribute(驗證輸入屬性)、CustomActionFilterAttribute(自定操作過濾屬性)
CourseGUID: 8503b39c-5887-4634-8291-facfb3117924
==========
(T18)討論 ValidateInputAttribute(驗證輸入屬性)、CustomActionFilterAttribute(自定操作過濾屬性)
==========

==========

0. Summary

==========
In this tutorial, we will discuss
* Please ensure you fully understand T011 before you continue.
https://ithandyguytutorial.blogspot.com.au/2018/02/t011textareacrosssitescriptingattackxss.html
* Action filters
Reference:
https://docs.microsoft.com/en-us/aspnet/mvc/overview/older-versions-1/controllers-and-routing/understanding-action-filters-cs
An action filter is an attribute that you can apply to a controller action -- or an entire controller -- that modifies the way in which the action is executed.
  * Authorize
  * ChildActionOnly
  * HandleError

  * OutputCache
  * RequireHttps
  * **ValidateInput**
  * **Customised Action Filter**
  * ValidateAntiForgeryToken
* IAuthorizationFilter
* IActionFilter
* IResultFilter
* IExceptionFilter

動作過濾器 Action Filter 4 -關於 ValidateInput 屬性，客製化的 ActionFilter 屬性。
* 初步介紹 ValidateInput 驗證屬性。
* 如果內建的 ActionFilter 屬性不夠你用!?沒問題，我們可以自己寫一個客製化的 ActionFilter 屬性。

Reference:
https://msdn.microsoft.com/en-us/library/windows/desktop/aa386968(v=vs.85).aspx
https://www.iis.net/downloads/microsoft/url-rewrite

==========

1.
"Actions" are controller public methods.
"Action filters" are attributes that can be applied to a controller or controller action.
It allows us to add the extra code of pre and post processing logic to the action methods.

2.
in Asp.net mvc, there are 4 types of filters
Reference:
https://msdn.microsoft.com/en-us/library/gg416513(v=vs.98).aspx
2.1.
Authorization filters
Implements **IAuthorizationFilter**.
E.g.
AuthorizeAttribute and **RequireHttpsAttribute**.
The filters **run before** any other filter.
2.2.
Action filters
Implement **IActionFilter**
2.3. Result filters
Implement **IResultFilter**.
E.g.
OutputCacheAttribute.
2.4.
Exception filters
Implement **IExceptionFilter**.
E.g.
HandleErrorAttribute

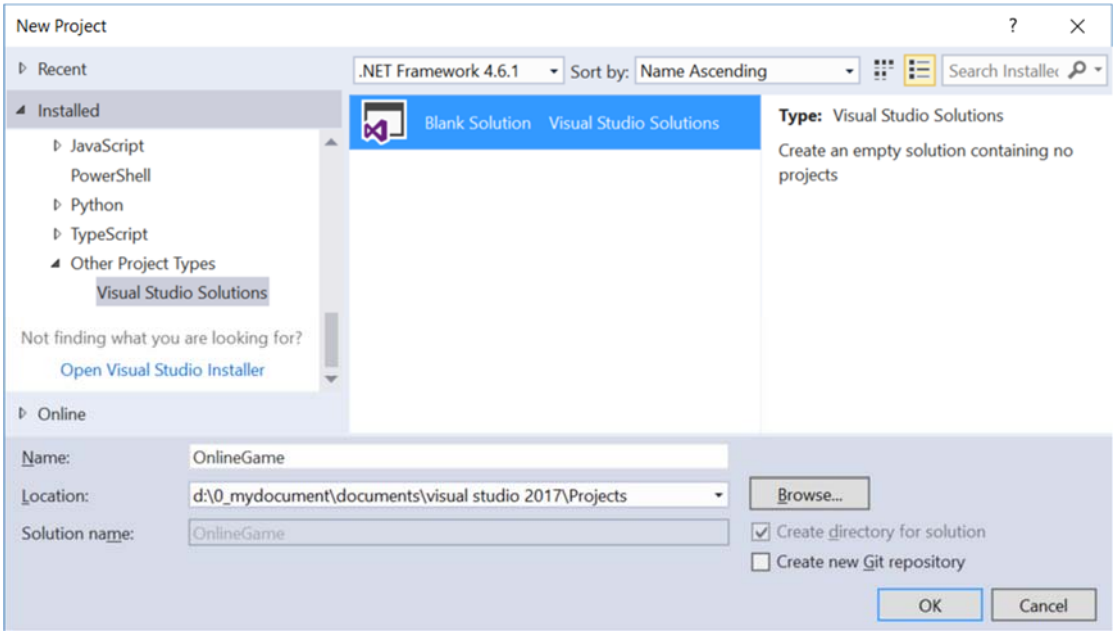
==========

1. New Project - OnlineGame

File --> New --> Project... -->

Other Project Types --> Visual Studio Solutions --> Blank Solution
-->

Name: **OnlineGame**

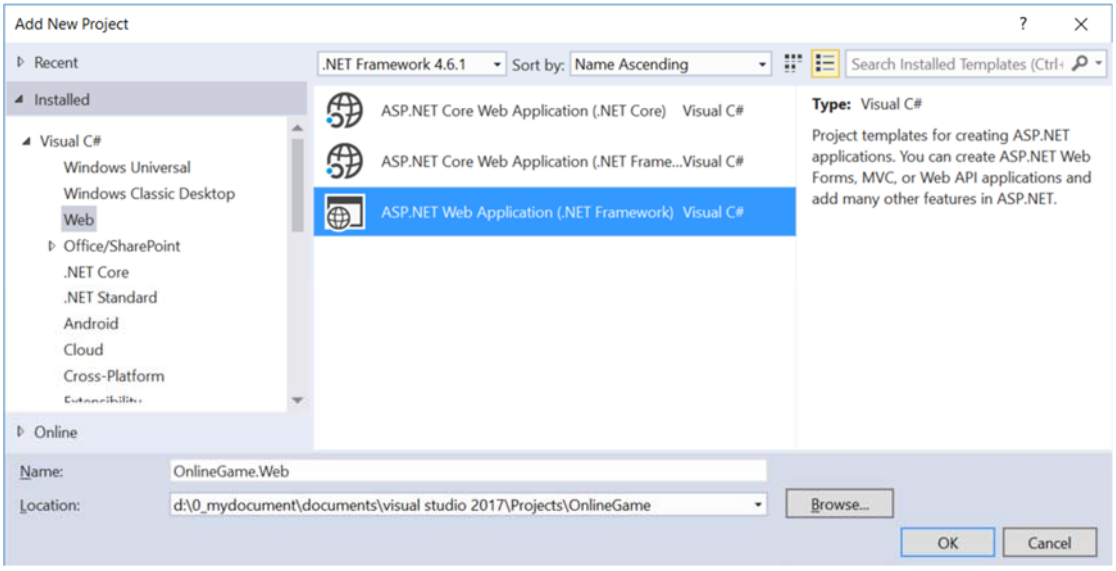


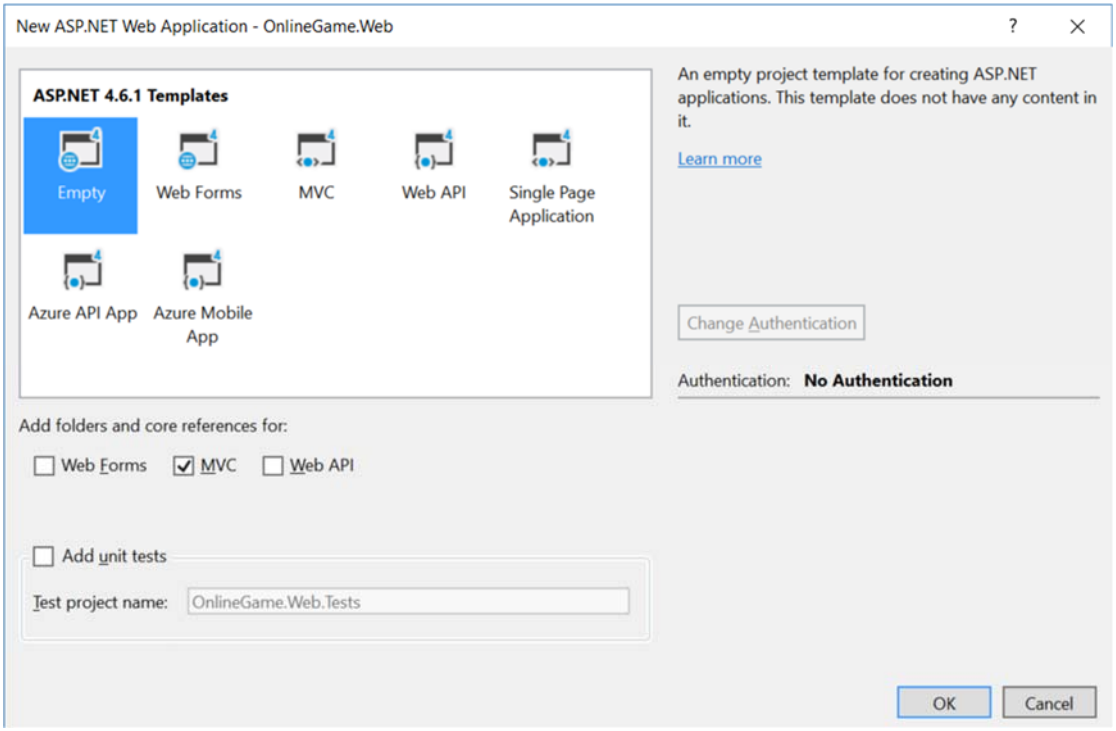
1.1. New Project - OnlineGame.Web

Solutions Name --> Add --> New Project -->

Visual C# --> Web --> ASP.NETWeb Application (.Net Framework)
-->

Name: **OnlineGame.Web**
Empty --> Select "MVC" --> OK





1.1.1. App_Start/FilterConfig.cs

```
using System.Web.Mvc;

namespace WebApplication1
{
    public class FilterConfig
    {
        public static void RegisterGlobalFilters(GlobalFilterCollection filters)
        {
            filters.Add(new HandleErrorAttribute());
        }
    }
}
/*
1.
Register Customized Error View
1.1.
Register HandleErrorAttribute to global filter
In Global.asax,
//FilterConfig.RegisterGlobalFilters(GlobalFilters.Filters);
We pass the GlobalFilters.Filters to
//public static void RegisterGlobalFilters(GlobalFilterCollection filters)
Here, we register "HandleErrorAttribute" to global filter.
1.2.
In Web.Config, add the customErrors mode="On"
//<system.web>
//    <customErrors mode="On">
//    </customErrors>
1.3.
Create error view, Views/Shared/Error.cshtml
*/
```

1.1.2. App_Start/RouteConfig.cs

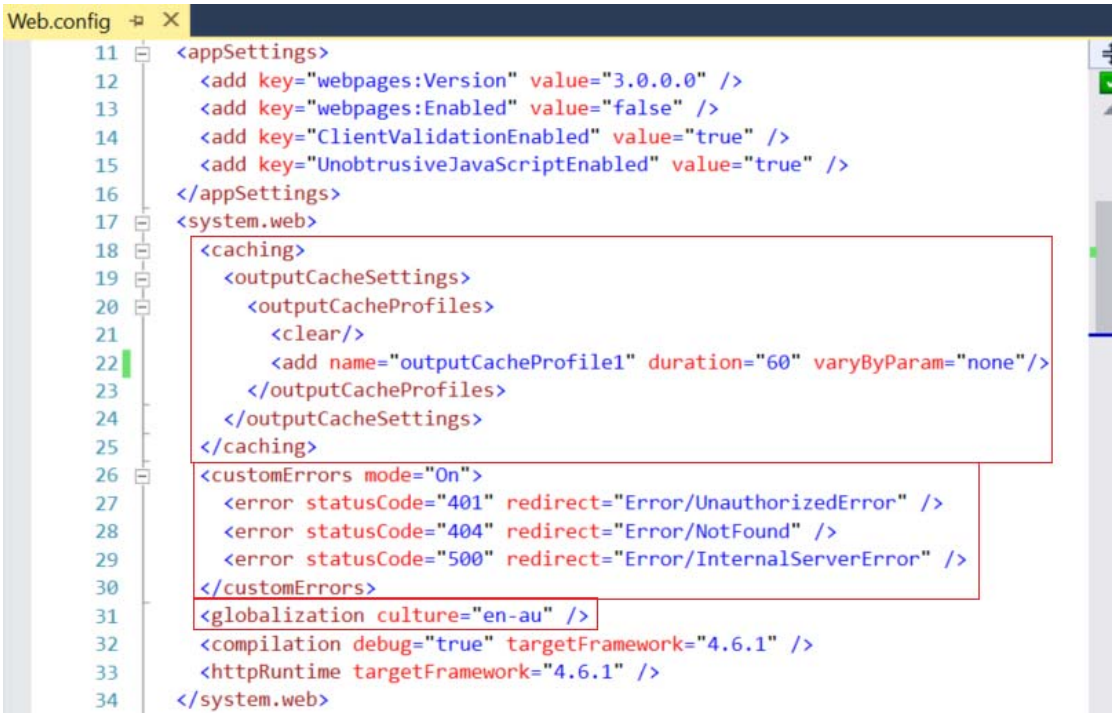
```
using System.Web.Mvc;
using System.Web.Routing;
namespace OnlineGame.Web
{
    public class RouteConfig
    {
        public static void RegisterRoutes(RouteCollection routes)
        {
            //Handle the Route of the axd request file.
            //E.g. ASP.Net Tracing
            routes.IgnoreRoute("{resource}.axd/{*pathInfo}");
            //Handle the Route called "Default".
            //The mapping URL is "{controller}/{action}/{id}"
            //Set the default value of Controller, action, and id.
            routes.MapRoute(
                name: "Default",
                url: "{controller}/{action}/{id}",
                defaults: new { controller = "Home", action = "Index", id = UrlParameter.Optional }
            );
        }
    }
}
/*
1.
//routes.MapRoute(
//    name: "Default",
//    url: "{controller}/{action}/{id}",
//    defaults: new { controller = "Home", action = "Index", id = UrlParameter.Optional }
//);
1.1.
When a request comes in,
it's trying to do a pattern match based on
all the templates it sees in these mapped routes.
A route is some instructions for
how to take a URI coming into a request
and map it to some code,
normally a controller.
In this case,
look at defaults parameter,
when user request http://localhost:PortNumber/
IIS Express will run
HomeController Index action.
It will map to Controllers/HomeController.cs
and   map to Index Method
1.2.
By convention in MVC.
All controllers will have Controller suffix.
This suffix is not required in the URL.
So, if you want to invoke Home controller,
you specify /Home and not /HomeController.
-----------------------------------
2.
//routes.IgnoreRoute("{resource}.axd/{*pathInfo}");
2.1.
Reference:
https://stackoverflow.com/questions/9016650/what-is-routes-ignorerouteresource-axd-pathinfo
```

```
This line can handle the axd file request route,
E.g. trace.axd
.axd files don't exist physically.

ASP.NET uses URLs with .axd extensions
(ScriptResource.axd and WebResource.axd) internally,
and they are handled by an HttpHandler.
Therefore, you should keep this rule,

to prevent ASP.NET MVC from trying to handle the request
instead of letting the dedicated HttpHandler do it.
2.2.
trace.axd
Reference:
https://msdn.microsoft.com/en-us/library/wwh16c6c.aspx
trace.axd trace details for a specific request.
If you want to enable trace.axd,
then you have to go to Web.config
Add <trace enabled="true" pageOutput="false"/> under <system.web>
Then run the project, type the following URL
http://localhost/OnlineGame.Web/trace.axd

This will return ASP.NET trace, trace.axd.
If you do not have
// routes.IgnoreRoute("{resource}.axd/{*pathInfo}");
then you can not enable the trace.axd.
*/
```

1.1.3. Global.asax.cs

```
using System.Web.Mvc;

using System.Web.Routing;

using WebApplication1;

namespace OnlineGame.Web
{
    public class MvcApplication : System.Web.HttpApplication
    {
        //Application_Start() is the magic start point of this application

        protected void Application_Start()
        {
            AreaRegistration.RegisterAllAreas();

            //Register HandleErrorAttribute to global filter
            FilterConfig.RegisterGlobalFilters(GlobalFilters.Filters);

            //1.

            //Register Route Configure in RouteConfig.cs

            //If you want to see route configuration,

            //you may find it in RouteConfig.cs

            //2.

            //System.Web.Routing.RouteCollection Routes { get; }

            //Gets a collection of objects that derive from the System.Web.Routing.RouteBase class.
            RouteConfig.RegisterRoutes(RouteTable.Routes);
        }
    }
}
```

1.1.4. Web.config




```
<system.web>
  <caching>
    <outputCacheSettings>
      <outputCacheProfiles>
        <clear/>
        <add name="outputCacheProfile1" duration="60" varyByParam="none"/>
      </outputCacheProfiles>
    </outputCacheSettings>
  </caching>
  <customErrors mode="On">
    <error statusCode="401" redirect="Error/UnauthorizedError" />
    <error statusCode="404" redirect="Error/NotFound" />
    <error statusCode="500" redirect="Error/InternalServerError" />
  </customErrors>
  <globalization culture="en-au" />
  <compilation debug="true" targetFramework="4.6.1" />
  <httpRuntime targetFramework="4.6.1" />
</system.web>
```

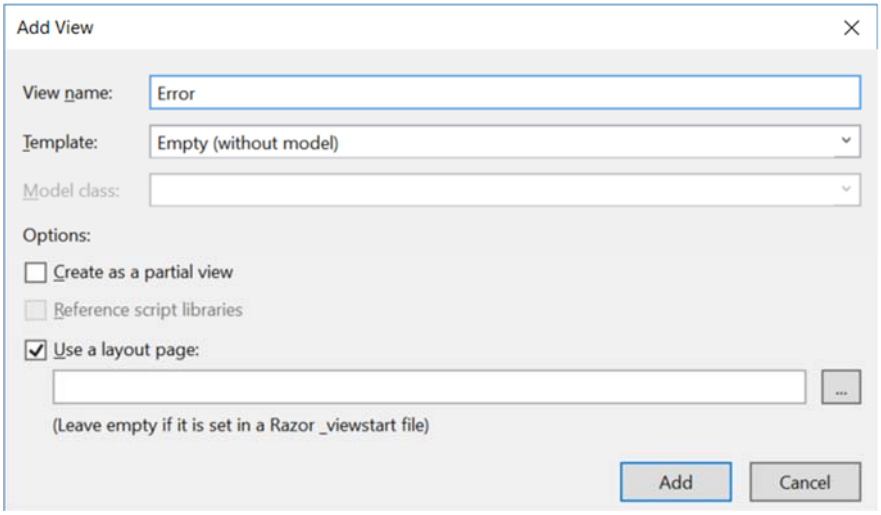
1.1.5. Add Customized Error View and Error Controller

1.1.5.1. Controllers/ErrorController.cs

```
using System.Web.Mvc;
```

```
namespace OnlineGame.Web.Controllers
{
    public class ErrorController : Controller
    {
        //error statusCode="401"
        [HttpGet]
        public ActionResult UnauthorizedError()
        {
            return View();
        }
        //error statusCode="404"
        [HttpGet]
        public ActionResult NotFound()
        {
            return View();
        }
        //error statusCode="500"
        [HttpGet]
        public ActionResult InternalServerError()
        {
            return View();
        }
    }
}
/*
1.
In the Web.config
//<customErrors mode="On" defaultRedirect="Error/DefaultError">
//    <error statusCode="401" redirect="Error/UnauthorizedError" />
//    <error statusCode="404" redirect="Error/NotFound" />
//    <error statusCode="500" redirect="Error/InternalServerError" />
//</customErrors>
We notice that it will still show the Views/Shared/Error.cshtml
when exception occurs.
Thus, we can delete Views/Shared/DefaultError.cshtml.
We also can delete DefaultError() in ErrorController.cs
In the Web.config, we can set as the following.
//<customErrors mode="On">
//    <error statusCode="401" redirect="Error/UnauthorizedError" />
//    <error statusCode="404" redirect="Error/NotFound" />
//    <error statusCode="500" redirect="Error/InternalServerError" />
//</customErrors>
*/
```

1.1.5.2. Views/Shared/Error.cshtml




```
@{
    ViewBag.Title = "Error";
}
<h2>Something occurs, please contact support.</h2>
```

1.1.5.3. Views/Shared/UnauthorizedError.cshtml

```
@{
    ViewBag.Title = "UnauthorizedError";
}
<h2>Error UnauthorizedError statusCode=401</h2>
You are trying to access something which you are not allowed to access.
```

http://localhost/onlinegame.web/Error/UnauthorizedError

Error UnauthorizedError statusCode=401

You are trying to access something which you are not allowed to access.

1.1.5.4. Views/Shared/NotFound.cshtml

```
@{
    ViewBag.Title = "NotFound";
}
<h2>Error NotFound statusCode=404</h2>
The request can not be found.
```

http://localhost/onlinegame.web/Error/NotFound

Error NotFound statusCode=404

The request can not be found.

1.1.5.5. Views/Shared/InternalServerError.cshtml

```
@{
    ViewBag.Title = "InternalServerError";
}
<h2>Error InternalServerError statusCode=500</h2>
The developers did something wrong, not users fault.
```

http://localhost/onlinegame.web/Error/InternalServerError

Error InternalServerError statusCode=500

The developers did something wrong, not users fault.

1.1.6. WebShared/CustomizeCacheAttribute.cs

1.1.6.1. WebShared/CustomizeCacheAttribute.cs

```
using System.Web.Mvc;
using System.Web.Configuration;
namespace OnlineGame.Web.WebShared
{
    public class CustomizeCacheAttribute : OutputCacheAttribute
    {
        public CustomizeCacheAttribute(string cacheProfileName)
        {
            OutputCacheSettingsSection cacheSettings =
                (OutputCacheSettingsSection)WebConfigurationManager
                .GetSection("system.web/caching/outputCacheSettings");
            OutputCacheProfile cacheProfile = cacheSettings.OutputCacheProfiles[cacheProfileName];
            Duration = cacheProfile.Duration;
            VaryByParam = cacheProfile.VaryByParam;
            VaryByCustom = cacheProfile.VaryByCustom;
        }
    }
}
/*
In Web.config
//<system.web>
//    <caching>
//        <outputCacheSettings>
//        <outputCacheProfiles>
//            <clear/>
//            <add name="outputCacheProfile1" duration="60" varyByParam="none"/>
//        </outputCacheProfiles>
//        </outputCacheSettings>
//    </caching>
//    <customErrors mode="On">
//        <error statusCode="401" redirect="Error/UnauthorizedError" />
//        <error statusCode="404" redirect="Error/NotFound" />
//        <error statusCode="500" redirect="Error/InternalServerError" />
//    </customErrors>
//    <globalization culture="en-au" />
```

```
//    <compilation debug="true" targetFramework="4.6.1" />
//    <httpRuntime targetFramework="4.6.1" />
//</system.web>
*/
```

1.1.6.2. The way to use WebShared/CustomizeCacheAttribute.cs

```
        [HttpGet]
        //[OutputCache(Duration = 60)]
        [OutputCache(CacheProfile = "outputCacheProfile1")]
        public async Task<ActionResult> Index4()
        {
            return View(await db.Gamers.ToListAsync());
        }
        //[ChildActionOnly] make the action to be accessible only by a child request,
        //so no one can make a direct URL request to this action.
        [ChildActionOnly]
        [HttpGet]
        //[OutputCache(Duration = 60)]
        //[OutputCache(CacheProfile = "outputCacheProfile1")]   //This will thrwo exception
        [CustomizeCache("outputCacheProfile1")]
        public string GetGamerCount2()
        {
            return $"Gamer Count = {db.Gamers.Count()} At {DateTime.Now}";
        }
```

2. OnlineGame.Web - [ValidateInput(false)]

2.1. Controllers/HomeController.cs

```
using System.Web.Mvc;
namespace OnlineGame.Web.Controllers
{
    public class HomeController : Controller
    {
        // GET: Home
        [HttpGet]
        public ActionResult Index()
        {
            return View();
        }
        [HttpPost]
        //[ValidateInput(false)]
        public string Index(string note)
        {
            return "Note : " + note;
        }
```

```
        // GET: Home
        [HttpGet]
        public ActionResult Index2()
        {
            return View();
        }
        [HttpPost]
        [ValidateInput(false)]
        public string Index2(string note)
        {
            return "Note : " + note;
        }
    }
}
/*
//[HttpPost]
//[ValidateInput(false)]
//public string Index2(string note)
[ValidateInput(true)] is the default setting.
It means we don't take any HTML tag in the input.
When we use [ValidateInput(false)],
it means we allow to have HTML tag input.
This will open the back door for XSS attack.
Please see my previous tutorial for more details
https://ithandyguytutorial.blogspot.com.au/2018/02/t011textareacrosssitescriptingattackxss.html
*/
```

2.2. Views/Home/Index.cshtml

```
@{
    ViewBag.Title = "Home Index";
}
<h2>Home Index</h2>
<div style="font-family:Arial">
    @using (Html.BeginForm("Index", "Home"))
    {
        <b>Note:</b>
        <br />
        @Html.TextArea("Note")
        <br />
        <input type="submit" value="Submit" />
    }
</div>
```

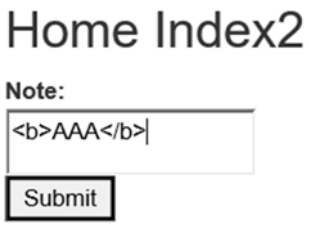
AAA

Go to Views/Shared/Error.cshtml

2.3. Views/Home/Index2.cshtml

```
@{
    ViewBag.Title = "Home Index2";
}
<h2>Home Index2</h2>
<div style="font-family:Arial">
    @using (Html.BeginForm("Index2", "Home"))
    {
        <b>Note:</b>
        <br />
        @Html.TextArea("Note")
        <br />
        <input type="submit" value="Submit" />
    }
</div>
```

AAA



Go to
http://localhost:57946/Home/Index2

Note : **AAA**

3. OnlineGame.Web - Customised Action Filter Attribute

3.1. LogExecutionTime/LogExecutionTime.txt

Create LogExecutionTime/LogExecutionTime.txt

3.2. WebShared/LogExecutionTimeAttribute.cs

```
using System;
using System.Web;
using System.Web.Mvc;
```

```
using System.IO;
namespace MVCDemo.WebShared
{
    public class LogExecutionTimeAttribute : ActionFilterAttribute, IExceptionFilter
    {
        public override void OnActionExecuting(ActionExecutingContext filterContext)
        {
            string logText
= $"\n[{filterContext.ActionDescriptor.ControllerDescriptor.ControllerName} : {filterContext.ActionDescript
or.ActionName}] -> OnActionExecuting \t- {DateTime.Now} \n";
            LogExecutionTimeIntoFile(logText);
        }
        public override void OnActionExecuted(ActionExecutedContext filterContext)
        {
            string logText
= $"\n[{filterContext.ActionDescriptor.ControllerDescriptor.ControllerName} : {filterContext.ActionDescript
or.ActionName}] -> OnActionExecuted \t- {DateTime.Now} \n";
            LogExecutionTimeIntoFile(logText);
        }
        public override void OnResultExecuting(ResultExecutingContext filterContext)
        {
            string logText
= $"\n[{filterContext.RouteData.Values["controller"]} : {filterContext.RouteData.Values["action"]}] ->
OnResultExecuting \t- {DateTime.Now} \n";
            LogExecutionTimeIntoFile(logText);
        }
        public override void OnResultExecuted(ResultExecutedContext filterContext)
        {
            string logText
= $"\n[{filterContext.RouteData.Values["controller"]} : {filterContext.RouteData.Values["action"]}] ->
OnResultExecuted \t- {DateTime.Now} \n";
            LogExecutionTimeIntoFile(logText);
            LogExecutionTimeIntoFile("-----------------------------------------------------------\n");
        }
        public void OnException(ExceptionContext filterContext)
        {
            string logText
= $"\n[{filterContext.RouteData.Values["controller"]} : {filterContext.RouteData.Values["action"]}] -
> \n OnException Message: {filterContext.Exception.Message} OnResultExecuted \t- {DateTime.Now} \n";
            LogExecutionTimeIntoFile(logText);
            LogExecutionTimeIntoFile("-----------------------------------------------------------\n");
        }
        private void LogExecutionTimeIntoFile(string logText)
        {
            File.AppendAllText(HttpContext.Current.Server.MapPath("~/LogExecutionTime/LogExecutionTime.tx
t"), logText);
        }
    }
}
```

3.3. Controllers/HomeController.cs

```
using System;
using System.Web.Mvc;
using MVCDemo.WebShared;
```

```
namespace OnlineGame.Web.Controllers
{
    public class HomeController : Controller
    {
        // GET: Home
        [HttpGet]
        public ActionResult Index()
        {
            return View();
        }
        [HttpPost]
        //[ValidateInput(false)]
        public string Index(string note)
        {
            return "Note : " + note;
        }
        // GET: Home
        [HttpGet]
        public ActionResult Index2()
        {
            return View();
        }
        [HttpPost]
        [ValidateInput(false)]
        public string Index2(string note)
        {
            return "Note : " + note;
        }
        [LogExecutionTime]
        public string Index3()
        {
            return "Home Index3 action has been called.";
        }
        [LogExecutionTime]
        public string Index4()
        {
            throw new Exception("Something Bad happened.");
        }
    }
}
/*
//[HttpPost]
//[ValidateInput(false)]
//public string Index2(string note)
[ValidateInput(true)] is the default setting.
It means we don't take any HTML tag in the input.
When we use [ValidateInput(false)],
it means we allow to have HTML tag input.
This will open the back door for XSS attack.
Please see my previous tutorial for more details
https://ithandyguytutorial.blogspot.com.au/2018/02/t011textareacrosssitescriptingattackxss.html
*/
```

Navigate to
Home/Index3
Home/Index4
and see what happened in
LogExecutionTime/LogExecutionTime.txt

HomeController.cs | LogExecutionTimeAttribute.cs | LogExecutionTime.txt

```
1
2   [Home : index3] -> OnActionExecuting    - 22/02/2018 4:20:40 PM
3
4   [Home : index3] -> OnActionExecuted     - 22/02/2018 4:20:40 PM
5
6   [home : index3] -> OnResultExecuting    - 22/02/2018 4:20:40 PM
7
8   [home : index3] -> OnResultExecuted     - 22/02/2018 4:20:40 PM
9   ------------------------------------------------------------
10
11  [Home : index4] -> OnActionExecuting    - 22/02/2018 4:20:47 PM
12
13  [Home : index4] -> OnActionExecuted     - 22/02/2018 4:20:51 PM
14
15  [home : index4] ->
16   OnException Message: Something Bad happened. OnResultExecuted  - 22/02/2018 4:20:51 PM
17  ------------------------------------------------------------
18
```