(T15)討論ActionFilter的Authorize、ChildActionOnly、HandleError  
CourseGUID: 8503b39c-5887-4634-8291-facfb3117924  
=======================================================================  
(T15)討論ActionFilter的Authorize、ChildActionOnly、HandleError

(T15-1)討論ActionFilter的Authorize (1. to 2.)

(T15-2)討論ActionFilter的ChildActionOnly (3.)

(T15-3)討論ActionFilter的HandleError、CustomErrorView(自訂錯誤頁) (4. to 5.)  
=======================================================================  
0. Introduction

-----------

1. New Project - OnlineGame

1.1. New Project - OnlineGame.Web

1.1.1. Global.asax.cs

1.1.2. App\_Start/RouteConfig.cs

1.1.3. Web.config

-----------

2. OnlineGame.Web - [Authorize]

2.1. Run with IIS

2.2. IIS Authentication

2.3. Controllers/HomeController.cs

2.4. Views/Home/NonSecureIndex.cshtml

2.5. Views/Home/SecureIndex.cshtml

-----------

3. OnlineGame.Web - [childactiononly]

3.1. Controllers/Home2Controller.cs

3.2. Views/Home2/GamersName.cshtml

3.3. Views/Home2/GamersName2.cshtml

3.4. Views/Home2/Index.cshtml

-----------

4. OnlineGame.Web - [HandleError]

4.1. Controllers/Home3Controller.cs

4.2. Add Customized Error View

4.2.1. App\_Start/FilterConfig.cs

4.2.2. Global.asax.cs

4.2.3. Web.config

4.2.4. Views/Shared/Error.cshtml

4.2.5. Test It

4.3. Add Customized Error View and Error Controller

4.3.1. Controllers/ErrorController.cs

4.3.2. Views/Shared/DefaultError.cshtml

4.3.3. Views/Shared/UnauthorizedError.cshtml

4.3.4. Views/Shared/NotFound.cshtml

4.3.5. Views/Shared/InternalServerError.cshtml

4.3.6. Web.config

4.3.7. Test It

4.4. Add Customized Error View and Error Controller (Correct Way)

4.4.1. Web.config

4.4.2. Controllers/ErrorController.cs , and delete Views/Shared/DefaultError.cshtml

4.4.3. Test It

4.5. Test 404 Not Found

4.6. Test 500 Internal Server Error

4.7. [HandleError] attribute

4.7.1. Views/Shared/Error.cshtml

4.7.2. App\_Start/FilterConfig.cs

4.7.3. Controllers/Home3Controller.cs

4.7.4. Test it

-----------

5. Add Customized Error View and Error Controller (Correct Way in Summary)

5.1. Controllers/ErrorController.cs

5.2. Views/Shared/Error.cshtml

5.3. Views/Shared/UnauthorizedError.cshtml

5.4. Views/Shared/NotFound.cshtml

5.5. Views/Shared/InternalServerError.cshtml

5.6. App\_Start/FilterConfig.cs

5.7. Global.asax.cs

5.8. Web.config

5.9. Controllers/Home3Controller.cs (For Test)

5.10. Test It

5.11. App\_Start/FilterConfig.cs  
=======================================================================

0. Introduction

=========================================

In this tutorial, we will discuss

\* 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

    \* ValidateAntiForgeryToken

動作過濾器Action Filter 1 - 常見的Authorize，ChildActionOnly，HandleError

常見的 動作過濾器Action Filter，關於Authorize，ChildActionOnly，HandleError。

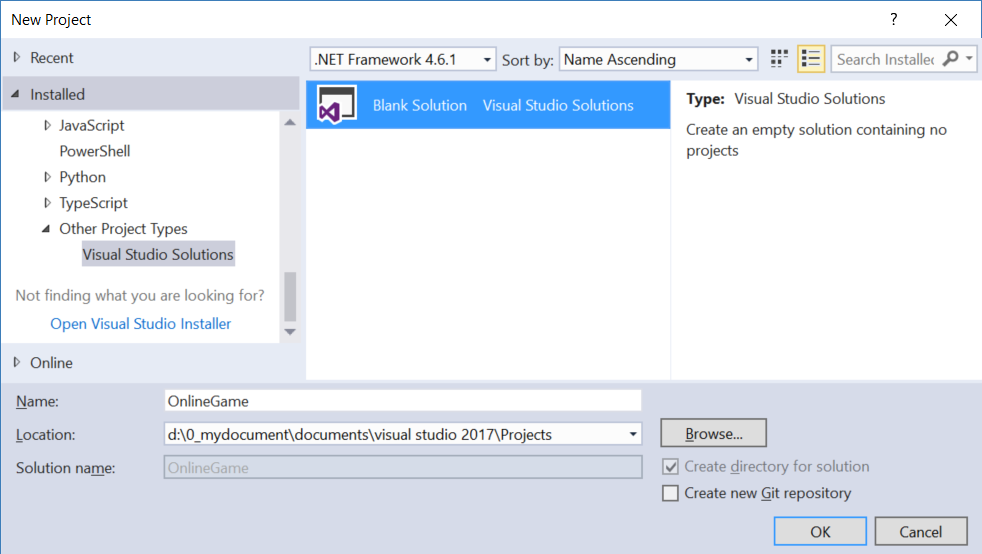
1. New Project - OnlineGame

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

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

-->

Name: **OnlineGame**



1.1. New Project - OnlineGame.Web

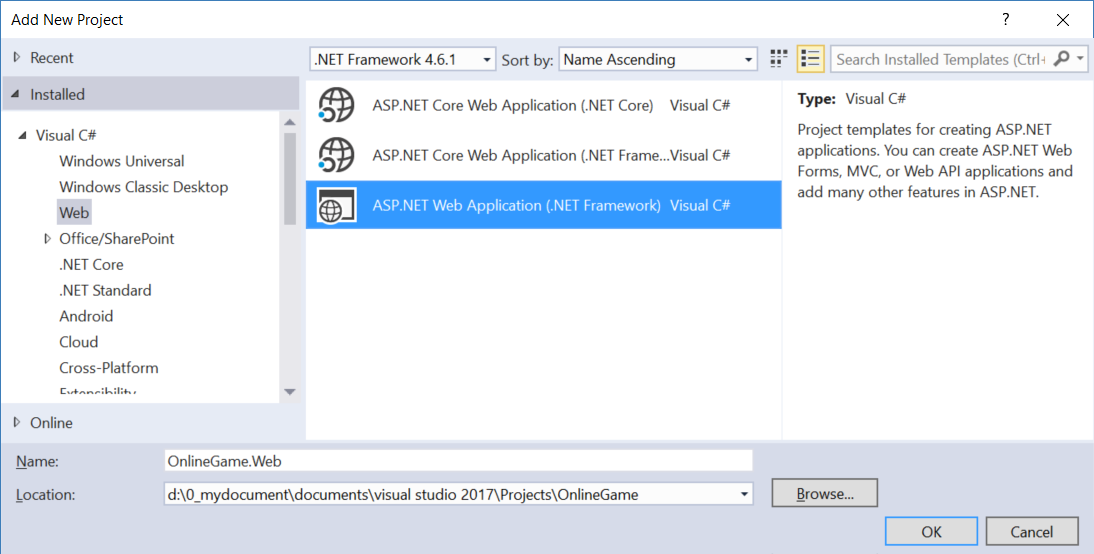
Solutions Name --> Add --> New Project -->

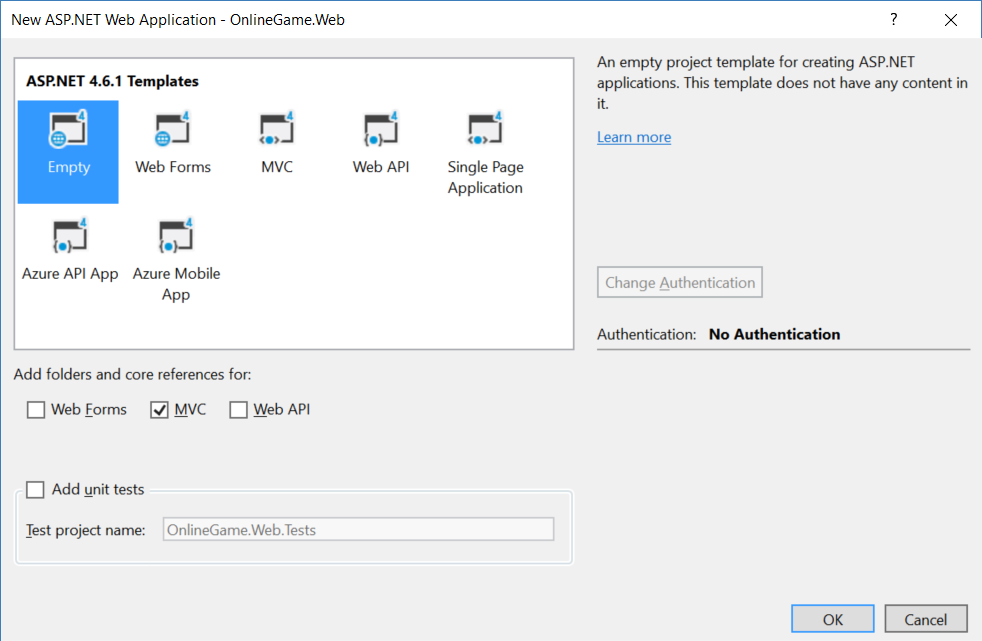
Visual C# --> Web --> [ASP.NET](http://asp.net/)Web Application (.Net Framework)

-->

Name: **OnlineGame.Web**

Empty --> Select "MVC" --> OK





1.1.1. Global.asax.cs

using System.Web.Mvc;

using System.Web.Routing;

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();

            //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);

        }

    }

}

1.1.2. 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](http://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](http://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](http://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](http://asp.net/) trace, trace.axd.

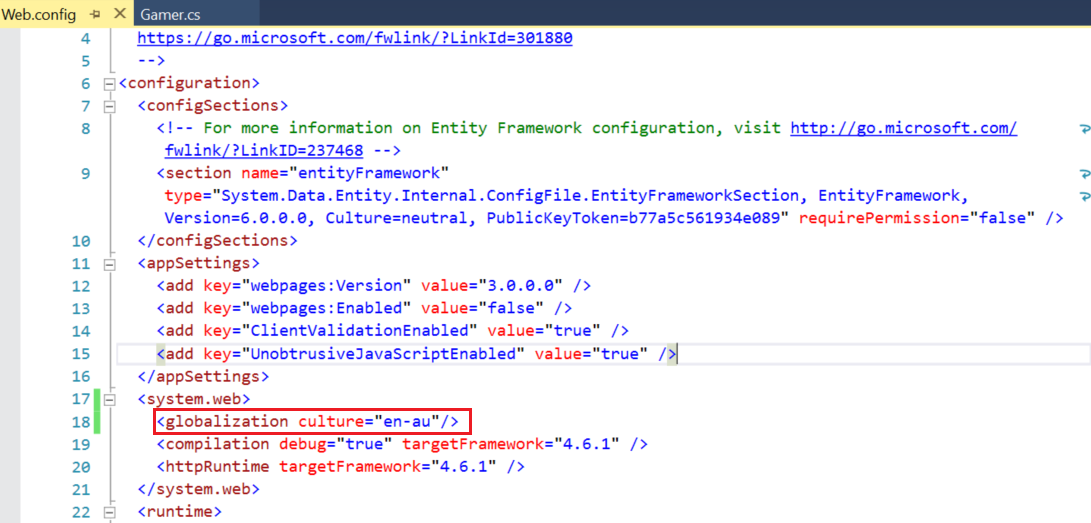
If you do not have

// routes.IgnoreRoute("{resource}.axd/{\*pathInfo}");

then you can not enable the trace.axd.

\*/

1.1.3. Web.config



  <system.web>

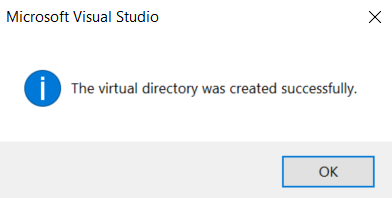
    <globalization culture="en-au"/>

2. OnlineGame.Web - [Authorize]

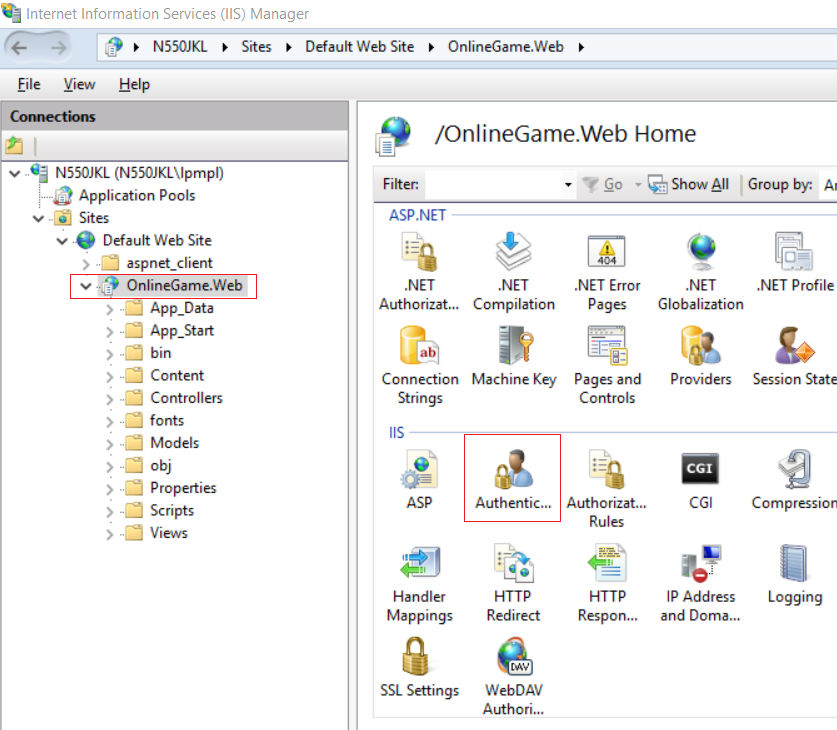
2.1. Run with IIS

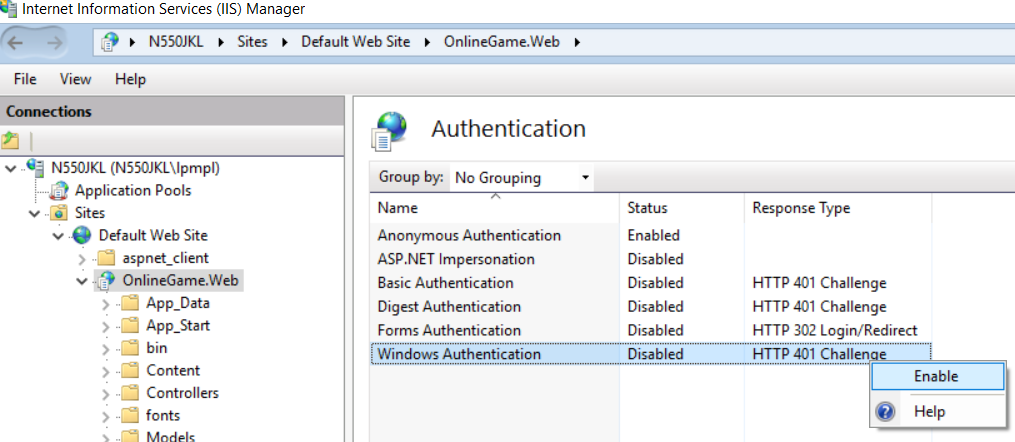
Graphical user interface, text, application, email

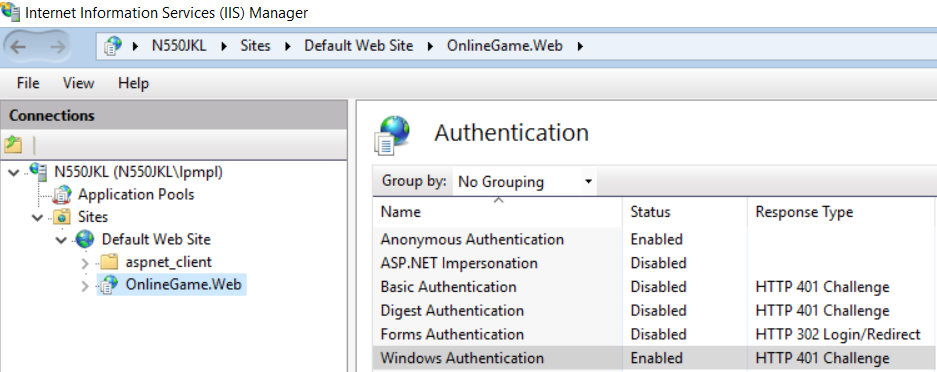
Description automatically generated



2.2. IIS Authentication







2.3. Controllers/HomeController.cs

using System.Web.Mvc;

namespace OnlineGame.Web.Controllers

{

    ////[Authorize]

    //1.

    //[Authorize] Need authentication to access

    //2.

    //[Authorize] can apply to controller level.

    [Authorize]

    public class HomeController : Controller

    {

        //1.

        ////[AllowAnonymous]

        //If [Authorize] apply to controller level,

        //then you need [AllowAnonymous] to apply to NonSecureIndex()

        [HttpGet]   //Home/NonSecureIndex

        [AllowAnonymous]

        public ActionResult NonSecureIndex()

        {

            return View();

        }

        //[Authorize] //Need authentication to access

        [HttpGet]   //Home/SecureIndex

        public ActionResult SecureIndex()

        {

            return View();

        }

    }

}

2.4. Views/Home/NonSecureIndex.cshtml

@{

    ViewBag.Title = "NonSecureIndex";

}

<h2>Home NonSecureIndex</h2>

2.5. Views/Home/SecureIndex.cshtml

@{

    ViewBag.Title = "SecureIndex";

}

<h2>Home SecureIndex</h2>

3. OnlineGame.Web - [childactiononly]

3.1. Controllers/Home2Controller.cs

using System;

using System.Collections.Generic;

using System.Web.Mvc;

namespace OnlineGame.Web.Controllers

{

    public class Home2Controller : Controller

    {

        // GET: Home2

        [HttpGet]   //Home2/Index

        public ActionResult Index()

        {

            return View();

        }

        [ChildActionOnly]

        [HttpGet]   //Home2/GamersName

        public ActionResult GamersName(List<String> gamersName)

        {

            return View(gamersName);

        }

        [HttpGet]   //Home2/GamersName2

        public ActionResult GamersName2(List<String> gamersName)

        {

            return View(gamersName);

        }

    }

}

/\*

1.

//[ChildActionOnly]

1.1.

Background:

In the Views/Home2/Index.cshtml

//@Html.Action("GamersName", new { gamersName = new List<string> { "Name2", "Name1", "Name3" } })

//@{ Html.RenderAction("GamersName", new { gamersName = new List<string> { "Name2", "Name1", "Name3" } }); }

//@Html.Action("GamersName2", new { gamersName = new List<string> { "Name2", "Name1", "Name3" } })

//@{Html.RenderAction("GamersName2", new { gamersName = new List<string> { "Name2", "Name1", "Name3" } });}

We call "GamersName" and "GamersName2" action in the index view.

By default of MVC convention,

users can use URL "Home2/GamersName" and "Home2/GamersName2"

to access the GamersName and GamersName2 actions.

We want to prevent the users to access directly to GamersName and GamersName2 actions.

[NoAction] attribute can do so.

But if the [NoAction] attribute applies to GamersName and GamersName2 actions,

then the Views/Home2/Index.cshtml view can not access these actions either.

Thus, we need [ChildActionOnly] attribute.

1.2.

[ChildActionOnly] make GamersName action to be accessible only by a child request.

so no one can make a direct URL request to this action

That means Index.cshtml view can still access this action,

but the users can not use url "Home2/GamersName" to directly access the GamersName action.

[ChildActionOnly] is usually associated with partial views.

\*/

3.2. Views/Home2/GamersName.cshtml

@model List<string>

@{

    ViewBag.Title = "GamersName";

}

<h2>Home2 GamersName</h2>

@foreach (string gamerName in Model)

{

    @gamerName<br />

}

--------------------------------

Home2/GamersName

Graphical user interface

Description automatically generated with medium confidence

3.3. Views/Home2/GamersName2.cshtml

@model List<string>

@{

    ViewBag.Title = "GamersName";

}

<h2>Home2 GamersName2</h2>

@foreach (string gamerName in Model)

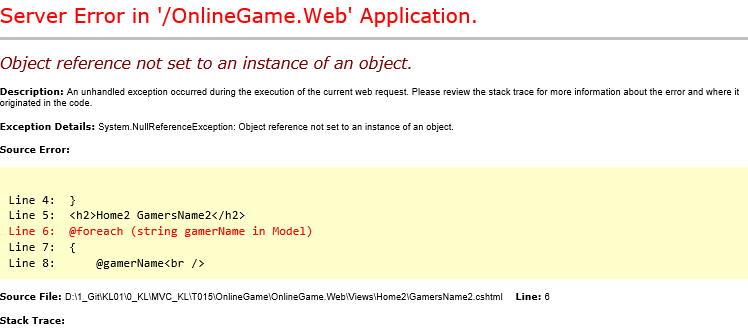
{

    @gamerName<br />

}

--------------------------------

Home2/GamersName2



3.4. Views/Home2/Index.cshtml

@{

    ViewBag.Title = "Index";

}

<h2>Home2 Index</h2>

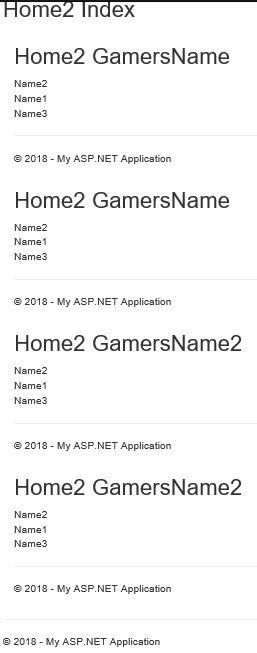
@Html.Action("GamersName", new { gamersName = new List<string> { "Name2", "Name1", "Name3" } })

@{Html.RenderAction("GamersName", new { gamersName = new List<string> { "Name2", "Name1", "Name3" } });}

@Html.Action("GamersName2", new { gamersName = new List<string> { "Name2", "Name1", "Name3" } })

@{Html.RenderAction("GamersName2", new { gamersName = new List<string> { "Name2", "Name1", "Name3" } });}

Home2/Index



4. OnlineGame.Web - [HandleError]

4.1. Controllers/Home3Controller.cs

using System;

using System.Web.Mvc;

namespace OnlineGame.Web.Controllers

{

    public class Home3Controller : Controller

    {

        // GET: Home3

        [HttpGet]   //Home3/Index

        public ActionResult Index()

        {

            throw new Exception("Exception occurs");

        }

    }

}

<http://localhost/OnlineGame.Web/home3/index>



4.2. Add Customized Error View

4.2.1. App\_Start/FilterConfig.cs

using System.Web;

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

\*/

4.2.2. 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.

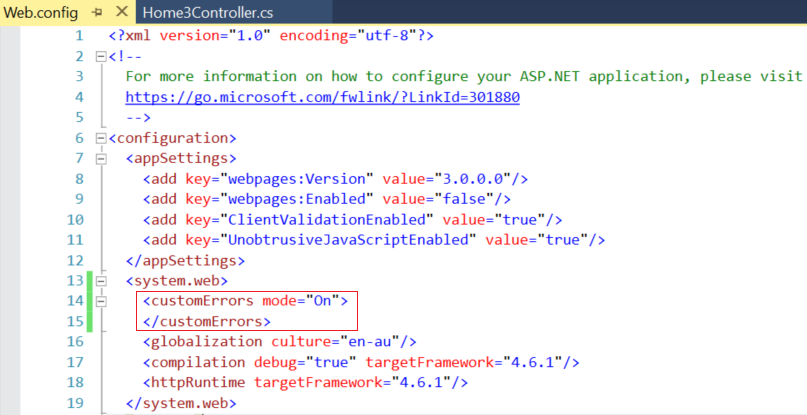
            RouteConfig.RegisterRoutes(RouteTable.Routes);

        }

    }

}

4.2.3. Web.config



<system.web>

    <customErrors mode="On">

    </customErrors>

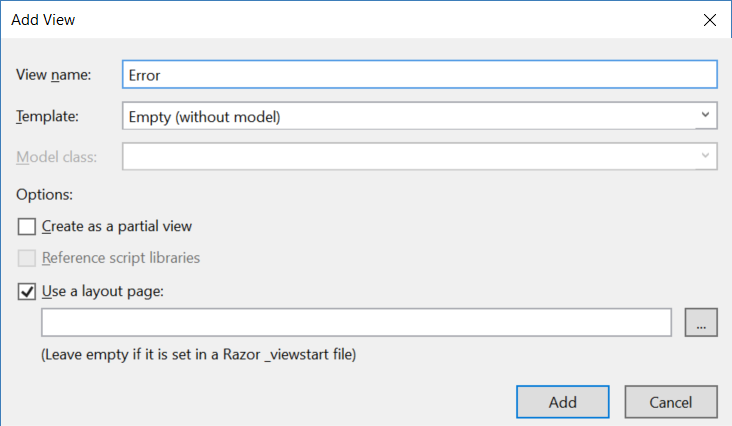
    <globalization culture="en-au"/>

    <compilation debug="true" targetFramework="4.6.1"/>

    <httpRuntime targetFramework="4.6.1"/>

  </system.web>

4.2.4. Views/Shared/Error.cshtml



@{

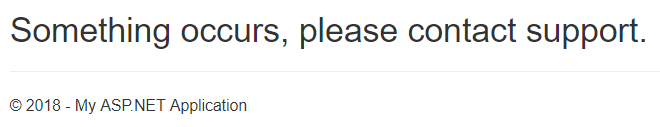
    ViewBag.Title = "Error";

}

<h2>Something occurs, please contact support.</h2>

4.2.5. Test It

<http://localhost/OnlineGame.Web/home3/index>



4.3. Add Customized Error View and Error Controller

4.3.1. Controllers/ErrorController.cs

using System.Web.Mvc;

namespace OnlineGame.Web.Controllers

{

    public class ErrorController : Controller

    {

        //This method can be deleted, because it is not useful.

        [HttpGet]

        public ActionResult DefaultError()

        {

            return View();

        }

        //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>

\*/

4.3.2. Views/Shared/DefaultError.cshtml

@{

    ViewBag.Title = "DefaultError";

}

<h2>Error DefaultError</h2>

<http://localhost/onlinegame.web/Error/DefaultError>



4.3.3. 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>



4.3.4. 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>

Text

Description automatically generated with medium confidence

4.3.5. Views/Shared/InternalServerError.cshtml

@{

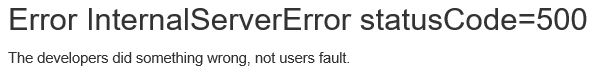
    ViewBag.Title = "InternalServerError";

}

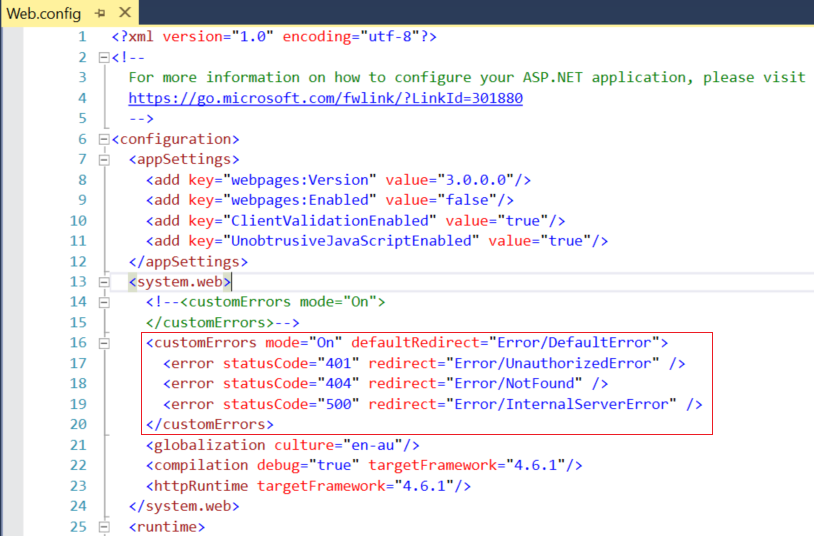
<h2>Error InternalServerError statusCode=500</h2>

The developers did something wrong, not users fault.

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



4.3.6. Web.config



<system.web>

    <!--<customErrors mode="On">

    </customErrors>-->

    <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>

    <globalization culture="en-au"/>

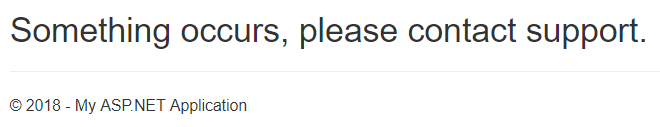
    <compilation debug="true" targetFramework="4.6.1"/>

    <httpRuntime targetFramework="4.6.1"/>

  </system.web>

4.3.7. Test It

<http://localhost/OnlineGame.Web/home3/index>



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" />

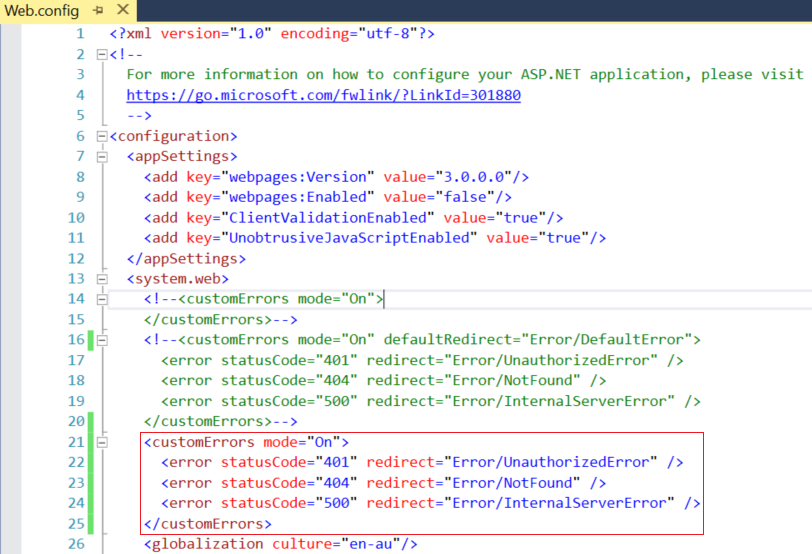
//    <error statusCode="404" redirect="Error/NotFound" />

//    <error statusCode="500" redirect="Error/InternalServerError" />

//</customErrors>

4.4. Add Customized Error View and Error Controller (Correct Way)

4.4.1. Web.config



<system.web>

    <!--<customErrors mode="On">

    </customErrors>-->

    <!--<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>-->

    <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>

4.4.2. Controllers/ErrorController.cs , and delete Views/Shared/DefaultError.cshtml

Delete **Views/Shared/DefaultError.cshtml**

Delete **DefaultError()** in **ErrorController.cs**

using System.Web.Mvc;

namespace OnlineGame.Web.Controllers

{

    public class ErrorController : Controller

    {

        ////This method can be deleted, because it is not useful.

        //[HttpGet]

        //public ActionResult DefaultError()

        //{

        //    return View();

        //}

        //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" />

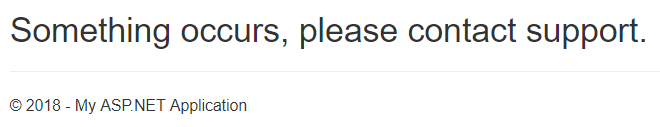
//    <error statusCode="500" redirect="Error/InternalServerError" />

//</customErrors>

\*/

4.4.3. Test It

<http://localhost/OnlineGame.Web/home3/index>



4.5. Test 404 Not Found

[http://localhost/OnlineGame.Web/home3/index2](http://localhost/OnlineGame.Web/home3/index)

it will redirect to

<http://localhost/onlinegame.web/Error/NotFound?aspxerrorpath=/onlinegame.web/home3/index2>

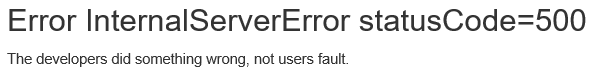


4.6. Test 500 Internal Server Error

Delete **Views/Shared/Error.cshtml**

Go to

<http://localhost/OnlineGame.Web/home3/index>



It supposed to go to **Views/Shared/Error.cshtml**,

but we have just deleted **Views/Shared/Error.cshtml**

Thus, it will return an internal server error.

It means the developers did something wrong.

4.7. [HandleError] attribute

4.7.1. Views/Shared/Error.cshtml

We add the view back.

@{

    ViewBag.Title = "Error";

}

<h2>Something occurs, please contact support.</h2>

4.7.2. App\_Start/FilterConfig.cs

We delete the code which will register the HandleErrorAttribute to global filter.

using System.Web;

using System.Web.Mvc;

namespace WebApplication1

{

    public class FilterConfig

    {

        public static void RegisterGlobalFilters(GlobalFilterCollection filters)

        {

            //filters.Add(new HandleErrorAttribute());

        }

    }

}

4.7.3. Controllers/Home3Controller.cs

using System;

using System.Web.Mvc;

namespace OnlineGame.Web.Controllers

{

    public class Home3Controller : Controller

    {

        // GET: Home3

        [HttpGet]   //Home3/Index

        [HandleError]

        public ActionResult Index()

        {

            throw new Exception("Exception occurs");

        }

        public ActionResult Index3()

        {

            throw new Exception("Exception occurs");

        }

    }

}

4.7.4. Test it

<http://localhost/OnlineGame.Web/home3/index>

it will redirect to

<http://localhost/onlinegame.web/Error/InternalServerError?aspxerrorpath=/OnlineGame.Web/home3/index3>

Graphical user interface, application

Description automatically generated

----------------------------------------------

[http://localhost/OnlineGame.Web/home3/index3](http://localhost/OnlineGame.Web/home3/index)

it will redirect to

<http://localhost/onlinegame.web/Error/InternalServerError?aspxerrorpath=/OnlineGame.Web/home3/index3>

Text

Description automatically generated

5. Add Customized Error View and Error Controller (Correct Way in Summary)

5.1. Controllers/ErrorController.cs

using System.Web.Mvc;

namespace OnlineGame.Web.Controllers

{

    public class ErrorController : Controller

    {

        ////This method can be deleted, because it is not useful.

        //[HttpGet]

        //public ActionResult DefaultError()

        //{

        //    return View();

        //}

        //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>

\*/

5.2. Views/Shared/Error.cshtml

Graphical user interface, text, application, email

Description automatically generated

@{

    ViewBag.Title = "Error";

}

<h2>Something occurs, please contact support.</h2>

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>



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>

Text

Description automatically generated with medium confidence

5.5. Views/Shared/InternalServerError.cshtml

@{

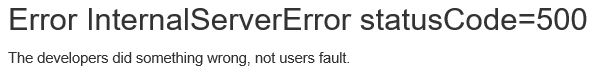
    ViewBag.Title = "InternalServerError";

}

<h2>Error InternalServerError statusCode=500</h2>

The developers did something wrong, not users fault.

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



5.6. App\_Start/FilterConfig.cs

using System.Web;

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

\*/

5.7. 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.

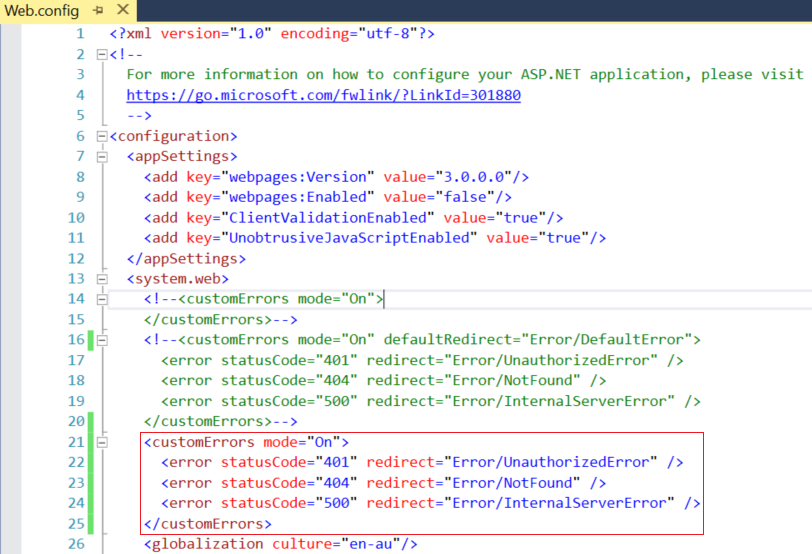
            RouteConfig.RegisterRoutes(RouteTable.Routes);

        }

    }

}

5.8. Web.config



<system.web>

    <!--<customErrors mode="On">

    </customErrors>-->

    <!--<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>-->

    <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>

5.9. Controllers/Home3Controller.cs (For Test)

You have to delete

filters.Add(new HandleErrorAttribute());

in App\_Start/FilterConfig.cs

before you test it.

using System;

using System.Web.Mvc;

namespace OnlineGame.Web.Controllers

{

    public class Home3Controller : Controller

    {

        // GET: Home3

        [HttpGet]   //Home3/Index

        [HandleError]

        public ActionResult Index()

        {

            throw new Exception("Exception occurs");

        }

        public ActionResult Index3()

        {

            throw new Exception("Exception occurs");

        }

    }

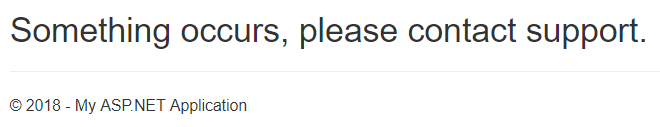
}

5.10. Test It

<http://localhost/OnlineGame.Web/home3/index>

it will redirect to

<http://localhost/onlinegame.web/Error/InternalServerError?aspxerrorpath=/OnlineGame.Web/home3/index3>

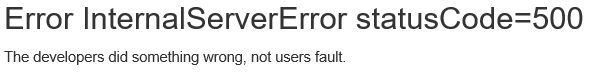


----------------------------------------------

[http://localhost/OnlineGame.Web/home3/index3](http://localhost/OnlineGame.Web/home3/index)

it will redirect to

<http://localhost/onlinegame.web/Error/InternalServerError?aspxerrorpath=/OnlineGame.Web/home3/index3>



5.11. App\_Start/FilterConfig.cs

You have to add

filters.Add(new HandleErrorAttribute());

in App\_Start/FilterConfig.cs

using System.Web;

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

\*/