

1. <customErrors> Element in web.config

- Route to custom error page when error occurred(Error.cshtml)
- Enable the <customErrors> in web.config, as shown below.

```
<system.web>  
  <customErrors mode="On"></customErrors>  
</system.web>
```

- We can configure default page for specific error

```
<customErrors mode="On">  
  <error statusCode="404" redirect="~/ErrorPage"/>  
</customErrors>
```

2. HandleErrorAttribute

- must add `HandleErrorAttribute` filter in the `FilterConfig.RegisterGlobalFilters()`
- set the mode attribute to On `<customErrors mode="On">` in web.config.
- apply `[HandleError]` attribute to the action method
- It will display Error.cshtml view as default view for any exceptions.
- can also be used to configure different pages for different types of exceptions,

Default Exception
Filter

```
[HandleError]  
[HandleError(ExceptionType =typeof(NullReferenceException), View  
="~/Views/Error/NullReference.cshtml")]  
public ActionResult Contact()  
{  
    string msg = null;  
    ViewBag.Message = msg.Length;  
  
    return View();  
}
```

3. Overriding Controller.OnException Method

- handle controller level exceptions
- handles all your unhandled errors with error code 500.
- It allows to log an exception and redirect to the specific view.
- It does not require to enable the <customErrors> config in web.config
- The below code is added in controller file

```
protected override void OnException(ExceptionContext
filterContext)
{
    filterContext.ExceptionHandled = true;

    //Log the error!!


    //Redirect to action
    filterContext.Result = RedirectToAction("Error",
"InternalError");

    // OR return specific view
    filterContext.Result = new ViewResult
    {
        ViewName = "~/Views/Error/InternalError.cshtml"
    };
}
```

```
public class MyExceptionHandler : HandleErrorAttribute
{
    public override void OnException(ExceptionContext
filterContext)
    {
        Exception e = filterContext.Exception;
        filterContext.ExceptionHandled = true;
        filterContext.Result = new ViewResult()
        {
            ViewName = "Error4"
        };
    }
}
```

Can be added globally
to any error in any
controller in the app
By adding this code in
global.asax

then we have MyExceptionHandler method in global.asax, we can use it as an attribute on controller level, or method level as :



```
namespace ExceptionHandling.Controllers
{
    [MyExceptionHandler]
    0 references | 0 changes | 0 authors, 0 changes
    public class HomeController : Controller
    {
        0 references | 0 changes | 0 authors, 0 changes
        public ActionResult Index()
    }
}
```

4. Application_Error event of HttpApplication

log exception occurred in any part of MVC application is to handle it in the Application_Error event in the global.asax file.

```
protected void Application_Error()
{
    var ex = Server.GetLastError();
    //log an exception
}
```