بسم الله الرحمن الرحیم

آموزشگاه تحلیل داده

تخصصی ترین مرکز برنامه نویسی و دیتابیس در ایران

مدیریت خطاها / ASP.NET – Error Handling

مدرس : مهندس افشین رفوآ

مدیریت خطاها / ASP.NET – Error Handling

مدیریت خطا در ASP.NET دارای سه جنبه ی متفاوت می باشد:

Tracing - قابلیت ثبت وقایع و اطلاعات مربوط به اجرای برنامه در سطح اپلیکیشن یا در سطح page.

Error handling – مدیریت خطاهای standard یا custom در سطح اپلیکیشن یا صفحه.

Debugging – اشکال زدایی که طی آن برنامه نویس مرحله به مرحله برنامه را بررسی کرده و با استفاده از breakpoint ها (نقطه ی انفصال) کد نوشته شده را تجزیه و تحلیل می کند

در مبحث پیش رو به تشریح هر سه جنبه ی نام برده خواهیم پرداخت.

جهت درک مفهوم، یک اپلیکیشن آزمایشی می سازیم. اپلیکیشن مورد نظر دارای یک کنترل label، یک کنترل dropdown list (فهرست کشویی) و یک لینک می باشد.

dropdown list یک لیست آرایه از quote ها را نمایش می دهد. Quote های انتخاب شده در کنترل label نشان داده می شوند.

اپلکیشن مورد نظر همچنین دارای یک لینک می باشد که در صورت کلیک روی آن کاربر به سایت خاصی ارجاع داده نمی شود.

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="ErrorHandling.WebForm1" Trace="true" %>
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">
    <title>Tracing, debugging and error handling
    </title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <asp:Label ID="lblheading" runat="server" Text="Tracing, Debuggin  and Error Handling">
            </asp:Label>
            <br />
            <br />
            <asp:DropDownList ID="ddlquotes" runat="server" AutoPostBack="True"
OnSelectedIndexChanged="ddlquotes_SelectedIndexChanged">
            </asp:DropDownList>
            <br />
            <br />
            <asp:Label ID="lblquotes" runat="server">
            </asp:Label>
            <br />
            <br />
            <asp:HyperLink ID="HyperLink1" runat="server" NavigateUrl="mylink.htm">Link
to:</asp:HyperLink>
        </div>
    </form>
</body>
</html>
```

فایل code behind:

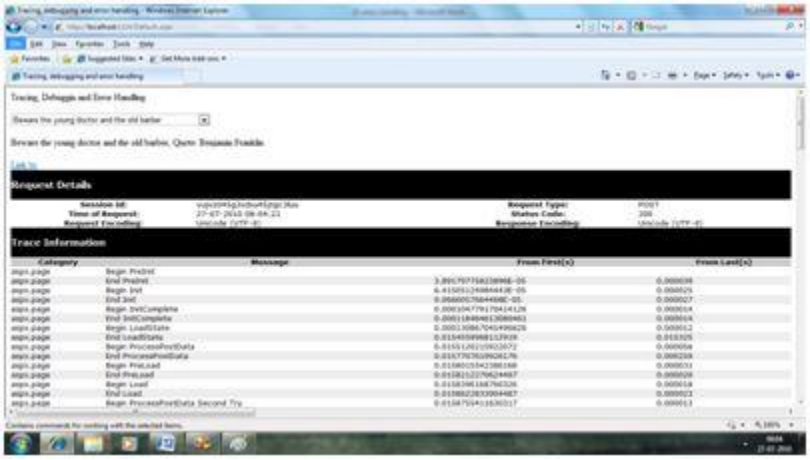
```
    public partial class WebForm1 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

            if (!IsPostBack)
            {
                string[,] quotes =
         {
            {"Imagination is more important than Knowledge.", "Albert Einsten"},
            {"Assume a virtue, if you have it not", "Shakespeare"},
            {"A man cannot be comfortable without his own approval", "Mark Twain"},
            {"Beware the young doctor and the old barber", "Benjamin Franklin"},
            {"Whatever begun in anger ends in shame", "Benjamin Franklin"}
         };
                for (int i = 0; i < quotes.GetLength(0); i++)
                    ddlquotes.Items.Add(new ListItem(quotes[i, 0], quotes[i, 1]));
            }

        }
        protected void ddlquotes_SelectedIndexChanged(object sender, EventArgs e)
        {
            if (ddlquotes.SelectedIndex != -1)
            {
                lblquotes.Text = String.Format("{0}, Quote: {1}", ddlquotes.SelectedItem.Text,
ddlquotes.SelectedValue);
            }

        }
    }
```

Tracing

به منظور ایجاد امکان tracing در سطح page، لازم است Page directive را اصلاح کرده و یک خصیصه ی Trace به آن اضافه کنید:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="ErrorHandling.WebForm1" Trace="true" %>
```

پس از اجرای فایل، اطلاعات مربوط به tracing (ردگیری و ثبت اطلاعات مربوط به اجرای برنامه) در اختیار شما قرار داده می شود:



اطلاعات زیر را ارائه می دهد:

- Session ID (session ی شناسه)
- Status Code (کد وضعیت)
- Time of Request (زمان درخواست)
- Type of Request (نوع درخواست)
- Request and Response Encoding (رمز گذاری درخواست و پاسخ)

Status code ای که با هر بار درخواست صفحه (page request) از سمت سرور ارسال می گردد، اسم خطا و زمان رخداد آن را نشان می دهد. جدول زیر status code های متداول HTTP را در اختیار شما قرار می دهد:

عدد	شرح

Informational (100 - 199)	
100	Continue
101	Switching protocols
Successful (200 - 299)	
200	OK
204	No content
Redirection (300 - 399)	
301	Moved permanently
305	Use proxy
307	Temporary redirect
Client Errors (400 - 499)	
400	Bad request
402	Payment required
404	Not found
408	Request timeout
417	Expectation failed

Server Errors (500 - 599)	
500	Internal server error
503	Service unavailable
505	HTTP version not supported

زیر بخش اطلاعات سطح بالا (top level info)، با یک Trace log (فایل ثبت وقایع و اطلاعات مر بوط به اجرای برنامه) مواجه می شویم که اطلاعاتی را درباره ی چرخه ی حیات (life cycle) صفحه در اختیار ما قرار می دهد. Trace log همچنین مدت زمان سپری شده از زمان مقداردهی اولیه ی صفحه را بر حسب ثانیه در اختیار ما قرار می دهد.

Trace Information

Category	
aspx.page	Begin PreInit
aspx.page	End PreInit
aspx.page	Begin Init
aspx.page	End Init
aspx.page	Begin InitComplete
aspx.page	End InitComplete
aspx.page	Begin LoadState
aspx.page	End LoadState
aspx.page	Begin ProcessPostData
aspx.page	End ProcessPostData
aspx.page	Begin PreLoad
aspx.page	End PreLoad
aspx.page	Begin Load
aspx.page	End Load
aspx.page	Begin ProcessPostData Second Try
aspx.page	End ProcessPostData Second Try
aspx.page	Begin Raise ChangedEvents
aspx.page	End Raise ChangedEvents

بخش بعدی control tree می باشد که کلیه کنترل های موجود در صفحه را به صورت سلسله مراتبی فهرست می کند:

Control Tree

Control UniqueID	Type	Render Size
__Page	ASP.default_aspx	2850
ctl02	System.Web.UI.LiteralControl	175
ctl00	System.Web.UI.HtmlControls.HtmlHead	70
ctl01	System.Web.UI.HtmlControls.HtmlTitle	57
ctl03	System.Web.UI.LiteralControl	14
form1	System.Web.UI.HtmlControls.HtmlForm	2571
ctl04	System.Web.UI.LiteralControl	27
lblheading	System.Web.UI.WebControls.Label	65
ctl05	System.Web.UI.LiteralControl	42
ddlquotes	System.Web.UI.WebControls.DropDownList	570
ctl06	System.Web.UI.LiteralControl	42
lblquotes	System.Web.UI.WebControls.Label	96
ctl07	System.Web.UI.LiteralControl	42
HyperLink1	System.Web.UI.WebControls.HyperLink	49
ctl08	System.Web.UI.LiteralControl	24
ctl09	System.Web.UI.LiteralControl	20

در پایان خلاصه یا چکیده ای از وضعیت های Application و Session، همچنین cookie ها و header ها (سرآیند) و به دنبال آن فهرستی از متغیرهای server را داریم.

شئ Trace به برنامه نویس امکان می دهد اطلاعات سفارشی یا custom به خروجی trace بیافزاید. شئ مذکور برای این منظور دو متد Write و Warn را ارائه می دهد.

Event handler (مدیریت کننده ی رخداد) Page_Load را برای چک کردن متد Write اصلاح کنید:

```
protected void Page_Load(object sender, EventArgs e)
{
    Trace.Write("Page Load");

    if (!IsPostBack)
      {
       Trace.Write("Not Post Back, Page Load");

      }
}
```

{حال کد را اجرا کرده تا نتیجه ی آن را مشاهده کنید:

aspx.page	Begin PreLoad
aspx.page	End PreLoad
aspx.page	Begin Load
	Page Load
	Not Post Back, Page Load
aspx.page	End Load
aspx.page	Begin LoadComplete
aspx.page	End LoadComplete

برای چک کردن متد Warn، تعدادی کد نادرست را به طور عمد در اندیس انتخابی event handler اصلاح شده وارد می کنیم:

```
try
 {
```

```
  int a = 0;
  int b = 9 / a;
 }
catch (exception e)
 {
  trace.warn("useraction", "processing 9/a", e);
 }
```

Try-Catch در واقع یک سازه ی برنامه نویسی برگرفته از زبان C# است. بلوک try کدهایی را دربرمی گیرد که ممکن است باعث ایجاد خطا (error) شوند و بلوک catch نیز خطا را گرفته و ضبط می کند. پس از اجرای برنامه، هشدار به داخل trace log فرستاده شده و نمایش داده می شود.

Tracing در سطح اپلیکیشن در واقع به تمامی صفحات موجود در وب سایت اعمال می شود. tracing در سطح اپلیکیشن با وارد کردن خط های کد زیر در فایل web.config پیاده سازی می شود:

```
    <system.web>
        <trace enabled="true" />
    </system.web>
```

Error Handling (مدیریت خطا)

اگرچه خود تکنولوژی ASP.NET قادر به پیدا کردن خطاهای زمان اجرا (runtime error) می باشد، با این وجود ممکن است تعدادی خطای جزئی از دید آن پنهان بماند. پیدا کردن و نظارت بر خطاها در واقع کار برنامه نویسان است و نه کاربرها. از این رو به منظور جلوگیری از وقوع چنین رخدادهایی می توان تنظیماتی مبنی بر مدیریت خطا (error handling settings) به فایل web.config اضافه کنید. برای مثال در صورتی که قرار است مدیریت خطا در کل برنامه اجرا شود، کافی است کدهای زیر را به فایل web.config خود اضافه کنید:

```
<configuration>
  <system.web>
    <customErrors mode="RemoteOnly" defaultRedirect="GenericErrorPage.htm">
      <error statusCode="403" redirect="NoAccess.htm"     />
      <error statusCode="404" redirect="FileNotFound.htm" />
    </customErrors>
  </system.web>
  <configuration>
```

بخش <customErrors> دارای خصیصه های ممکن زیر می باشد:

Mode : custom error page (صفحات خطا customize شده) را فعال یا غیر فعال می کند.

On: در این حالت صفحات سفارشی (custom) نمایش داده می شوند.

Off: صفحات خطای پیش فرض ASP.NET را نمایش می دهد (صفحات زرد رنگ)

RemoteOnly: خطاهای سفارشی را به کلاینت نشان داده و خطاهای ASP.NET را به صورت محلی را نمایش می دهد

defaultRedirect: دربرداندره ی URL صفحه ای است که در صورت برخورد با خطاهای مدیریت نشده (unhandled error) نمایش داده می شود.

به منظور قرار دادن صفحات خطای customized شده برای انواع مختلف خطا، از subTag های <error> بهره می گیریم که در آن انواع مختلف صفحات خطا بر اساس کد وضعیت (status code) خطاها مشخص می شود.

به منظور پیاده سازی مدیریت خطا که در سطح صفحه اجرا می شود (page level error handling)، باید page directive را اصلاح کرد:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
   Inherits="errorhandling._Default" Trace ="true" ErrorPage="PageError.htm" %>
```