**Customizing the GridView**

|  |  |
| --- | --- |
| **Edit / Cancel / Page – Field with ascx** | **Field with Link** |
|  |  |

**When editing we want to be sure the Quantity field has a value, if not a message is displayed**

**Task 1: Create a new Entity Framework Data Model**

Add an **ADO.NET Entity Data Model** project item named **AdventureWorks.edmx**. The data model should be generated from a database, by using the **AdventureWorksLT2008** connection string from the Web.config file. All tables from the database, with the exception of the **BuildVersion** table, should be added to the data model.

1. In Solution Explorer, right-click **AdventureWorks**, point

**Task 2: Create a Shopping Cart Quantity user control ShoppingCard.ascx**

1. Create a new user control named **Quantity.ascx**. (Add / New Item / Web / Web User Control ) In the **Name** box, type **Quantity.ascx**, and then click **Add**.

2. Add a **TextBox** control named **QuantityTextBox** to the user control.

<asp:TextBox ID="QuantityTextBox" runat="server"></asp:TextBox>

3. Add a **RequiredFieldValidator** control named **QuantityRequiredFieldValidator** which validate the **QuantityTextBox**, and display the text “You must fill in the Quantity.” as the error message, if the user leaves the text box empty. The display mode must be dynamic.

1. **HTML**

<%@ Control Language="C#" AutoEventWireup="true" CodeBehind="Quantity.ascx.cs" Inherits="AdventureWorks.Quantity" %>

<asp:TextBox ID="QuantityTextBox" runat="server"></asp:TextBox>

<asp:RequiredFieldValidator ID="QuantityRequiredFieldValidator" runat="server"

ErrorMessage="You must fill in the Quantity."

ControlToValidate="QuantityTextBox"

Display="Dynamic">

</asp:RequiredFieldValidator>

1. **CODE BEHIND**

In the code behind, add a public property named **Text** to the user control. The property must be of type **String**/**string**, and get and set the **Text** property of the **QuantityTextBox** control.

public partial class Quantity : System.Web.UI.UserControl

{

public string Text

{

get

{

return QuantityTextBox.Text;

}

set

{

QuantityTextBox.Text = value;

}

}

protected void Page\_Load(object sender, EventArgs e)

{

}

}

**The ShoppingCard.cs**

**ShoppingCard has 5 fields**

namespace AdventureWorks

{

public class ShoppingCart

{

public int ProductId { get; set; }

public string Name { get; set; }

|  |
| --- |
| private int productQuantity;  public int Quantity  {  get  {  return productQuantity;  }  set  {  productQuantity = value;  TotalPrice = productQuantity \* ListPrice;  }  } |

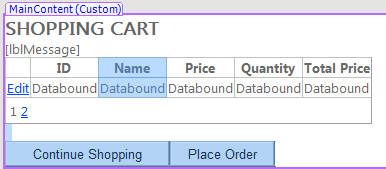
public decimal ListPrice { get; set; }

public decimal TotalPrice { get; set; }

}

}

**Task 3: Create a Shopping Cart Quantity user control ShoppingCard.ascx**



1. **HTML**

**Register the ascx control**

<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master" AutoEventWireup="true"

CodeBehind="ShoppingCart.aspx.cs" Inherits="AdventureWorks.ShoppingCart1" %>

<%@ Register Src="Quantity.ascx" TagName="Quantity" TagPrefix="uc1" %>

**Add label and 2 buttons**

<h2>

Shopping Cart

</h2>

<asp:Label ID="lblMessage" runat="server" Text=""></asp:Label>

<br />

<asp:Button ID="btnContinue" runat="server" OnClick="btnContinue\_Click" Text="Continue Shopping" />

<asp:Button ID="btnPlaceOrder" runat="server" OnClick="btnPlaceOrder\_Click" Text="Place Order" />

**Add Grid**

* Make the grid editable AutoGenerateEditButton="true"
* Change quantity to template for edit
* Create computed field

<asp:GridView ID="gvCart" runat="server" AutoGenerateColumns="False" AllowPaging="True"

AutoGenerateEditButton="true"

OnRowEditing="gvCart\_RowEditing"

OnRowCancelingEdit="gvCart\_RowCancelingEdit"

OnRowUpdating="gvCart\_RowUpdating"

OnPageIndexChanging="gvCart\_PageIndexChanging"

PageSize="1">

<Columns>

<asp:BoundField DataField="ProductId" HeaderText="ID" ReadOnly="true" />

<asp:BoundField DataField="Name" HeaderText="Name" />

<asp:BoundField DataField="ListPrice" HeaderText="Price" DataFormatString="{0:c}" />

|  |
| --- |
| **CHANGE FROM FIELD TO TEMPLATE, SO IT IS POSSIBLE TO EDIT**  <asp:TemplateField HeaderText="Quantity">  <EditItemTemplate>  <uc1:Quantity ID="Quantity1" runat="server" Text='<%# Bind("Quantity") %>' />  </EditItemTemplate>  <ItemTemplate>  <asp:Label ID="Label1" runat="server" Text='<%# Bind("Quantity") %>'></asp:Label>  </ItemTemplate>  </asp:TemplateField> |
| **CREATE COMPUTED FIELD**    <asp:TemplateField HeaderText="Total Price">  <ItemTemplate>  <asp:Label ID="Label1" runat="server" Text='<%# Eval("TotalPrice") %>' DataFormatString="{0:c}"></asp:Label>  </ItemTemplate>  </asp:TemplateField> |

</Columns>

</asp:GridView>

1. **CODE BEHIND**

namespace AdventureWorks

{

public partial class ShoppingCart1 : System.Web.UI.Page

{

|  |
| --- |
| // When the page is called, it brings the information that is in Session[\_cart]  // At the outset the grid is not populated from a database but form a Session variable  // if it is empty then a massage is shown  protected void Page\_Load(object sender, EventArgs e)  {  if (!IsPostBack)  {  if (Session["\_cart"] != null)  {  List<ShoppingCart> cart = (List<ShoppingCart>) Session["\_cart"];  gvCart.DataSource = cart;  gvCart.DataBind();  }  else  {  lblMessage.Text = "Your cart is empty";  btnPlaceOrder.Visible = false;  }  }  } |

**EDIT / UPDATE / CANCEL**

|  |
| --- |
| protected void **gvCart\_RowEditing**(object sender, GridViewEditEventArgs e)  {  gvCart.EditIndex = e.NewEditIndex;  BindData();  }    **// The data is not updated in a database, but in the Session variable**  protected void **gvCart\_RowUpdating**(object sender, GridViewUpdateEventArgs e)  {  // Retrieve existing cart from session  List<ShoppingCart> cart = (List<ShoppingCart>) Session["\_cart"];  // Get current row  GridViewRow row = gvCart.Rows[e.RowIndex];  // Find matching row in stored cart  ShoppingCart cartToEdit = cart.Find((cartToFind) => cartToFind.ProductId == int.Parse(row.Cells[1].Text));  // Update values  Quantity qty = gvCart.Rows[e.RowIndex].FindControl("Quantity1") as Quantity;  cartToEdit.Quantity = int.Parse(qty.Text);  // Save updated cart to session  Session["\_cart"] = cart;  // Reset edit index  gvCart.EditIndex = -1;  BindData();  }  protected void **gvCart\_RowCancelingEdit**(object sender, GridViewCancelEditEventArgs e)  {  gvCart.EditIndex = -1;  BindData();  }  private void **BindData**()  {  List<ShoppingCart> cart = (List<ShoppingCart>) Session["\_cart"];  gvCart.DataSource = cart;  gvCart.DataBind();  } |

|  |
| --- |
| protected void btnContinue\_Click(object sender, EventArgs e)  {  Response.Redirect("~/Default.aspx");  }  protected void btnPlaceOrder\_Click(object sender, EventArgs e)  {  Session["\_cart"] = null;  Response.Redirect("~/ThankYou.aspx");  }  protected void **gvCart\_PageIndexChanging**(object sender, GridViewPageEventArgs e)  {  // Cancel paging operation if user attempts to navigate to another page while in edit mode  if (((GridView) sender).EditIndex != -1)  {  e.Cancel = true; // Cancel paging operation  int newPageIndex = e.NewPageIndex + 1;  lblMessage.Text = "Update the item before moving to page " + newPageIndex.ToString();  }  else  {  // Clear message text  lblMessage.Text = "";  // Set page index  ((GridView) sender).PageIndex = e.NewPageIndex;  BindData();  }  } |

}

}

**In the Default.aspx**

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  | Product Categories  <br />    <asp:ListBox ID="lbCategories"  runat="server"  DataTextField="Name"  DataValueField="ProductCategoryID"  Height="270px"  Width="186px">  </asp:ListBox>    <br />    <asp:Button ID="btnCategory"  runat="server" Text="Submit">  </asp:Button> | protected void Page\_Load(object sender, EventArgs e)  {  if (!Page.IsPostBack)  {  DataSet ds = DataAccessLayer.Products.GetCategories();  lbCategories.DataSource = ds.Tables[0];  lbCategories.DataBind();  }  else  {  if(lbCategories.SelectedIndex != -1)  {  string category = lbCategories.SelectedValue;  Response.Redirect("/Products.aspx?id=" + category);  }  }  } |

**In Product.aspx**

|  |  |
| --- | --- |
|  |  |
|  | <h2>  Products  </h2>  <asp:Label ID="lblCategory" runat="server" Text="Label"></asp:Label>  <p>  <asp:GridView ID="gvProducts" runat="server"  AutoGenerateColumns="False"  CellPadding="4"  ForeColor="#333333" GridLines="None">  <AlternatingRowStyle BackColor="White" ForeColor="#284775" />  <Columns>  <asp:BoundField DataField="ProductID" HeaderText="ID" />     |  | | --- | | <asp:TemplateField>  <ItemTemplate>  <asp:HyperLink ID="HyperLink1" runat="server"  NavigateUrl='<%# Eval("ProductID", "~/ProductDetail.aspx?id={0}") %>'  Text='<%# Eval("Name") %>'></asp:HyperLink>  </ItemTemplate> |     </asp:TemplateField>  <asp:BoundField DataField="Color" HeaderText="Color" />  <asp:BoundField DataField="Size" HeaderText="Size" />  <asp:BoundField DataField="Weight" HeaderText="Weight" />  <asp:BoundField DataField="ListPrice" HeaderText="List Price"  DataFormatString="{0:c}" />  </Columns>    <EditRowStyle BackColor="#999999" />  <FooterStyle BackColor="#5D7B9D" Font-Bold="True" ForeColor="White" />  <HeaderStyle BackColor="#5D7B9D" Font-Bold="True" ForeColor="White" />  <PagerStyle BackColor="#284775" ForeColor="White" HorizontalAlign="Center" />  <RowStyle BackColor="#F7F6F3" ForeColor="#333333" />  <SelectedRowStyle BackColor="#E2DED6" Font-Bold="True" ForeColor="#333333" />  <SortedAscendingCellStyle BackColor="#E9E7E2" />  <SortedAscendingHeaderStyle BackColor="#506C8C" />  <SortedDescendingCellStyle BackColor="#FFFDF8" />  <SortedDescendingHeaderStyle BackColor="#6F8DAE" />  </asp:GridView>  </p> |

**Retrieve ALL products according to the category passed from Default**

protected void Page\_Load(object sender, EventArgs e)

{

if (!Page.IsPostBack)

{

string categoryId = "5";

if (Request["id"] != null) categoryId = Request["id"];

string categoryName = DataAccessLayer.Products.GetCategoryName(categoryId);

lblCategory.Text = categoryName;

DataSet ds = DataAccessLayer.Products.GetProductsByCategory(categoryId);

gvProducts.DataSource = ds.Tables[0];

gvProducts.DataBind();

}

}