Упражнение 2 - Достъп до бази данни.

Задача

Създайте нова уеб страница, чрез File -> New -> Web Site (C# language). Изпълнете следните стъпки и промени:

- Добавете база данни към проекта, с десен бутон върху App_Data -> Add New Item -> SQL Server Database.
- Добавете таблица People към базата (От Server Explorer -> Tables->Add New Table). В таблицата на базата данни да се съхраняват две имена и години на служители/клиенти. Добавете фиктивни данни в таблицата (десен бутон върху името на таблицата -> Show Table Data).
- В aspx станицата (режим Design) добавете SQLDataSource контрола и настройте връзката към базата данни. Трябва да се настрои SELECT операция за визуализиране данните от базата.
- Добавете GridView контрола към ващата страница и я свържете с базата данни.
- Да се включи възможността за разглеждане по страници и да се промени изгледа на таблицата.
- В уеб страницата добавете текстови полета за въвеждане на нова информация в базата (двете имена и години).
- Добавете бутон и поле за извеждане на информация за средната възраст на хората в базата.
- Опишете метода за добавяне на новите данни в базата.

Съдържание:

Почти всяко динамично уеб приложение осъществява някакъв вид достъп до данни и ASP.NET 2.0 прави това изключително лесно за изпълнение. За разлика от ASP.NET 1.0, при който разработчиците трябва да пишат ръчно код за да извличат и свързват данни с контроли, ASP.NET 2.0 решава това без да изисква допълнителен код за най-честите сценарии, като:

- Избор и извеждане на данни
- Сортиране, разделяне на страници и кеширане на данни
- Модификация, добавяне и изтриване на данни
- Филтриране и допълнителни детайли чрез параметри

ASP.NET 2.0 представя два типа сървърни контроли, които участват в декларативния модел на свързване с данни. Тези два типа контроли се справят със сложността на уеб модела за различни сценарии, така че разработчиците не е нужно да разбират събитията от цикъла заявки на страницата само за да изпълнят връзка с данните. Друга полза от този базиран на контроли модел е това, че той лесно може да бъде разширен за да поддържа допълнителни доставчици за съхранение и достъп да данни.

Контроли – източници на данни (Data Source)

Data source контролите нямат изглед, а представляват определен заден източник на данни, като например база данни, бизнес обект, XML файл или XML уеб услуга. Data source контролите също така предоставят богати възможности за операции над данните – сортиране, разделяне на страници, филтриране, модифициране, изтриване и добавяне – които свързани с данните интерфейсни контроли могат автоматично да използват. ASP.NET включва следните сървърни контроли:

Име	Описание
SqlDataSource	Прави връзка към SQL база данни, предоставена от ADO.NET доставчик, като Microsoft™ SQL Server, OLEDB, ODBC или Oracle.
ObjectDataSource	Дава връзка към обект от среден слой като слой за достъп до данни или бизнес компонент.
AccessDataSource	Прави връзка към Microsoft™ Access (Jet) база данни.
SiteMapDataSource	Дава възможност за връзка към йерархия представена от ASP.NET 2.0 навигация на сайта.
XmlDataSource	Прави връзка към XML файл или документ.

Обвързани с данните контроли (Data-bound Controls)

Обвързаните с данни контроли са интерфейсни и зареждат данните като маркировка след заявка на клиентско устройство или браузър. Те могат автоматично да свързват с данните от източника в необходимия момент от време. Тези контроли могат да се възползват от свойствата на контролите – източници: сортиране, модифициране, изтриване, добавяне и други. Един data-bound контрол се свързва към data source контрол чрез неговото `DataSourceID` свойство. Някои контроли познати от ASP.NET v1.x са DataGrid, DataList, Repeater списъчни контроли като DropDownList. ASP.NET 2.0 предоставя и следните нови контроли:

Име	Описание
GridView	Извежда данните в грид формат. Този контрол е усъвършенстван DataGrid и може автоматично да работи с функциите на data source.
DetailsView	Извежда единствен елемент от данните в таблица с име и стойност двойки, подобно на изгледа в Microsoft™ Access. Този контрол може автоматично да работи с функциите на data source.
FormView	Извежда само по един елемент на веднъж в предварително дефиниран шаблон. Извежда единствен елемент от данните в таблица с име и стойност двойки, подобно на изгледа в Microsoft™ Access. Този контрол може автоматично да работи с функциите на data source.
TreeView	Извежда данните в йерархичен дървовиден изглед от вложени възли.
Menu	Извежда данните в йерархично динамично меню.

Връзка към бази данни

Един от най-често срещаните типове данни използвани в уеб приложения идват от SQL бази данни като Microsoft SQL Server, Oracle или други OLEDB или ODBC хранилища. **SqlDataSource** контролът представлява директна връзка към базата данни, която data-bound контроли могат да използват за автоматично получаване на информация. SqlDataSource замества ADO.NET кода, който обикновено се пише за да се създаде връзка и да се командват заявки към база данни. Тъй като заявките са уточнени директно като свойства на data source контролът, това понякога се нарича двуслоен модел, защото заявките се поддържат в кода на страницата. По тази причина SqlDataSource контролът се използва при непрофесионални и лични сайтове, които не изискват силно капсулиран обект от средния слой.

GridView контрол

За целите на демонстрацията как да се свържат данни от база, примерите използват контролът **GridView**. Той е нов контрол в ASP.NET 2.0 за извеждане на данни в табличен формат. Всеки ред в грида съответства на един запис от данни, а колоните представляват полетата му. Ако сте запознати с DataGrid контролът, GridView е негов усъвършенстван вариант, поддържащ подобен обектен модел.

GridView контролът поддържа следните възможности:

- Връзка към data source контроли.
- Вградени възможности за сортиране.
- Вградени възможности за модифициране и изтриване.
- Вградени възможности за изглед по страници.
- Вградени възможности за избор на ред.
- Програмируем достъп до GridView обектния модел за динамична настройка на свойства и обработка на събития.
- Нови типове колони като CheckBoxField и ImageField.

- Множество полета данни за колоните хипервръзка.
- Множество ключови полета за избор, модификация и изтриване.
- Възможност за промяна на изгледа чрез теми и стилове.

Изготвяне на доклад от данни

Най-простият тип страница с данни е такава, в която данните се предоставят само за четене в доклад, който извежда данните без да дава възможност на потребителя да променя начина, по който те се виждат или самите тях. За да се състави такъв доклад от SQL база данни, първо се конфигурира SqlDataSource на страницата и след това се свързва data-bound контрол като GridView към този източник на данни като се уточни неговото `DataSourceID` свойство. Примерът по-долу показва GridView контрол асоцииран с SqlDataSource контрол.

```
<form runat="server">
  <asp:GridView ID="GridView1" DataSourceID="SqlDataSource1"
runat="server"/>
  <asp:SqlDataSource ID="SqlDataSource1" runat="server"
    SelectCommand="SELECT [au_id], [au_lname], [au_fname] FROM [authors]"
    ConnectionString="<%$ ConnectionStrings:Pubs %>" />
</form>
```

ConnectionString свойството на SqlDataSource определя стрингът за връзка към базата данни и свойството **SelectCommand** посочва заявката, която ще се изтегли данните. Низът за връзка може буквално да се посочи в страницата, но в този случай свойството е определено използвайки нов синтаксис вземащ взема стойността от Web.config. В примера по-долу GridView контрол се свързва с SqlDataSource контрол свързан с Microsoft SQL Server база данни.

Пример: GridView-SqlDataSource

SqlDataSource контролът не е ограничен за връзка само с Microsoft™ SQL Server бази данни. Той може да се свърже с всеки ADO.NET доставчик конфигуриран като **System.Data.Common.DbProviderFactory**. По подразбиране има четири провайдера включени в .NET Framework machine.config:

```
<configuration>

  <system.data>
    <DbProviderFactories>
      <add name="Odbc Data Provider" invariant="System.Data.Odbc"
type="System.Data.Odbc.OdbcFactory, ..." />
      <add name="OleDb Data Provider" invariant="System.Data.OleDb"
type="System.Data.OleDb.OleDbFactory, ..." />
      <add name="OracleClient Data Provider"
invariant="System.Data.OracleClient"
type="System.Data.OracleClient.OracleClientFactory, ..." />
      <add name="SqlClient Data Provider"
invariant="System.Data.SqlClient"
type="System.Data.SqlClient.SqlClientFactory, ..." />
```

```
    </DbProviderFactories>
  </system.data>
</configuration>
```

ProviderName свойството на SqlDataSource може да бъде установено като инвариантно име на произволен валиден провайдер (по подразбиране е System.Data.SqlClient). Когато се промени името на провайдъра трябва да се провери дали ConnectionString и SelectCommand свойствата използват правилен синтаксис за избрания източник.

В горния пример GridView контрол отразява полетата на записите върнати от SqlDataSource, като динамично генерира колоните на грида. Могат допълнително да се добавят **DataControlField** обекти към множеството колони на GridView. Това позволява да се оказва точно кои колони да се показват и техния относителен ред. Следващият пример демонстрира множество от **BoundField** и **CheckBoxField** обекти в GridView Columns. Други типове полета които могат да бъдат оказани са ImageField, HyperLinkField, CommandField, ButtonField и TemplateField.

Пример: GridView-SqlDataSource (BoundFields)

Свойството SelectCommand на SqlDataSource може да бъде настроено като име на съхранена процедура вместо SQL команден текст. За да се разреши това свойството **SelectCommandType** трябва да има стойност "StoredProcedure". Долният пример показва SqlDataSource контрол конфигуриран да селектира данни от съхранена процедура в примерна база данни Northwind.

Пример: GridView-SqlDataSource (Съхранени процедури)

По подразбиране SqlDataSource контролът връща DataView от DataSet обект, който съдържа резултатите от заявката. Вместо това SqlDataSource контролът може да бъде настроен да връща данните като DataReader като свойството **SqlDataSourceMode** приеме стойност "DataReader". Ползването на DataReader е обикновено по-ефективно в сравнение с DataSet когато просто се нуждаете от forward-only, read-only достъп до данните. Въпреки това трябва да се има предвид, че възможността за сортиране на SqlDataSource в този режим не е позволена. Следващият пример показва DataReader режима на SqlDataSource.

Пример: GridView-SqlDataSource (DataReader)

Настройки на низа за свързване

В горните примери SqlDataSource се свързва с низа за свързване посредством декларативен синтаксис, който определя стринга по време на изпълнение. Низа за свързване от своя страна е съхранен в Web.config файл в `<connectionStrings>` конфигурационна секция, така че да може лесно да се поддържа на едно място за всички страници в приложението.

```
<configuration>
  <connectionStrings>
    <add name="Pubs" connectionString="Server=(local);Integrated
Security=True;Database=pubs;"
      providerName="System.Data.SqlClient" />
  </connectionStrings>
</configuration>
```

Следващият пример показва Web.config файл използван в горните SqlDataSource примери.

Пример: Настройка на низовете за свързване

Съхранението на такива низове в Web.config е препоръчителна практика за всяко ASP.NET приложение не само за централизирано управление, а и за подсигуряване на сигурността им. Съществува инструмент, който да криптира тази секция за по-голяма сигурност. Тук няма да се спираме над тази възможност. В долния пример може да се наблудава как изглежда Web.config файл с криптирана `<connectionStrings/>` секция.

Пример: Настройка на низовете за свързване (криптиран)

Сортиране и разделяне по страници на данните

Едно от главните предимства на GridView контрола е възможността му автоматично да работи с data source свойствата. Вместо да се разчита на код в страницата за ръчно сортиране и индексация на данните, GridView контролът може да изпълни тези операции веднага щом източникът на данни е конфигуриран и поддържа тези операции.

Модифициране и изтриване на данни

Също както при сортирането и пейджинга, GridView контролът може автоматично да зарежда интерфейс за промяна на данните чрез Update и Delete операции. SqlDataSource контролът поддържа Update и Delete операции, когато съответно неговите **UpdateCommand** и **DeleteCommand** свойства са установено с валидни процедури за модификация или изтриване или пък съхранена процедура. UpdateCommand или DeleteCommand трябва да съдържат полета за параметър за всяка стойност, която ще бъде подадена от GridView контрол. Също така могат да се определят и

UpdateParameters или **DeleteParameters** множества, определящи свойствата на всеки един параметър, като тип на данните, посока или стойност по подразбиране.

```
<asp:SqlDataSource ID="SqlDataSource1" runat="server"
  ConnectionString="<%$ ConnectionStrings:Pubs %>"
  SelectCommand="SELECT [au_id], [au_lname], [au_fname], [state] FROM
[authors]"
  UpdateCommand="UPDATE [authors] SET [au_lname] = @au_lname, [au_fname] =
@au_fname, [state] = @state WHERE [au_id] = @au_id"
  DeleteCommand="DELETE FROM [authors] WHERE [au_id] = @au_id"/>
```

За да се включи интерфейса на GridView за модификация и изтриване може или да се настроят свойствата **AutoGenerateEditButton** и **AutoGenerateDeleteButton** като истина, или пък да се добави **CommandField** поле и да се активират свойствата му **ShowEditButton** и **ShowDeleteButton**. GridView поддържа модификация или изтриване на един ред на веднъж. За да започне модификация потребителят трябва да Edit бутона, а за да потвърди промените Update бутона. Потребителят може да натисне Cancel бутона за да отхвърли направените промени и да се върне в режим само на четене. Следващия пример показва GridView и SqlDataSource конфигурирани за модификация на редове данни.

Пример: GridView Updating

Добавяне на данни

Подобно на GridView контрола, DetailsView също поддържа модифициране и изтриване на данни чрез техния data source. DetailsView поддържа и добавяне на данни за разлика от GridView. DetailsView може лесно да бъде свързан с GridView за да се разреши добавяне на записи към изобразения GridView контрол.

За да може SqlDataSource да поддържа добавяне **InsertCommand** свойството трябва да има подходяща команда за добавяне, с параметри съответстващи на полетата от записите. Допълнително могат да се определи множество **InsertParameters** параметри.

За да е възможно добавяне чрез интерфейса трябва свойството **AutoGenerateInsertButton** да бъде истина или да се добави CommandField с **ShowInsertButton** истина. За да се влезе в режим на добавяне трябва да се натисне бутона "New". DetailsView съставя входен контрол за всяко едно поле. За да се изключи поле от режима на добавяне, трябва стойността на свойството **InsertVisible** да е лъжа. За да се извърши операцията по добавяне, се натиска бутона "Insert", а за отмяна – "Cancel".

Когато се изпълнява операцията по добавяне, DetailsView събира стойностите от входните полета и ги препраща към източника на данни. SqlDataSource прилага тези стойности към параметрите на InsertCommand преди да се изпълни реално командата.

Пример: Master-Details Insert