ASP.NET 2.0
Web Programming

이 현 정

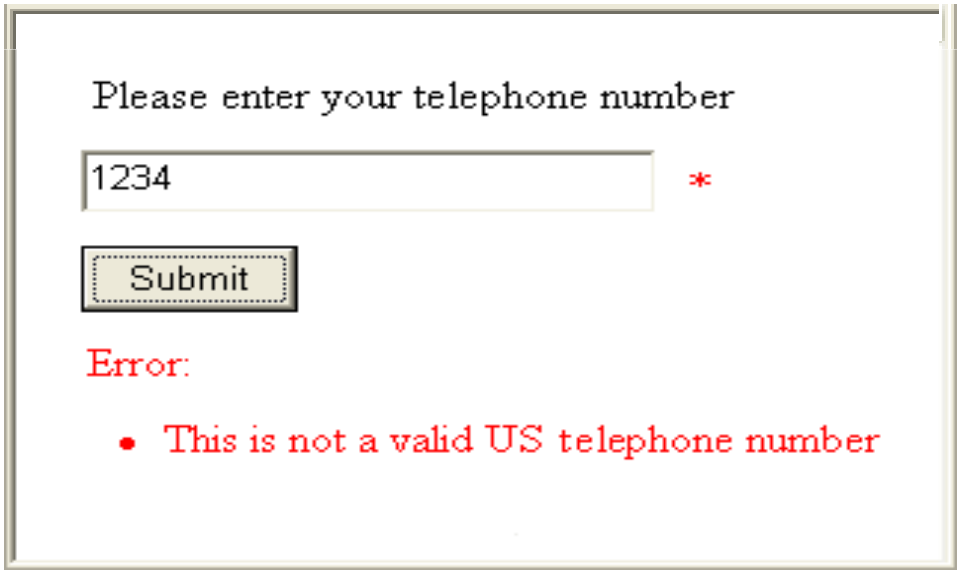
hjyi@hotmail.com

MCT/MCSD/MCAD/MCSD.NET

목차

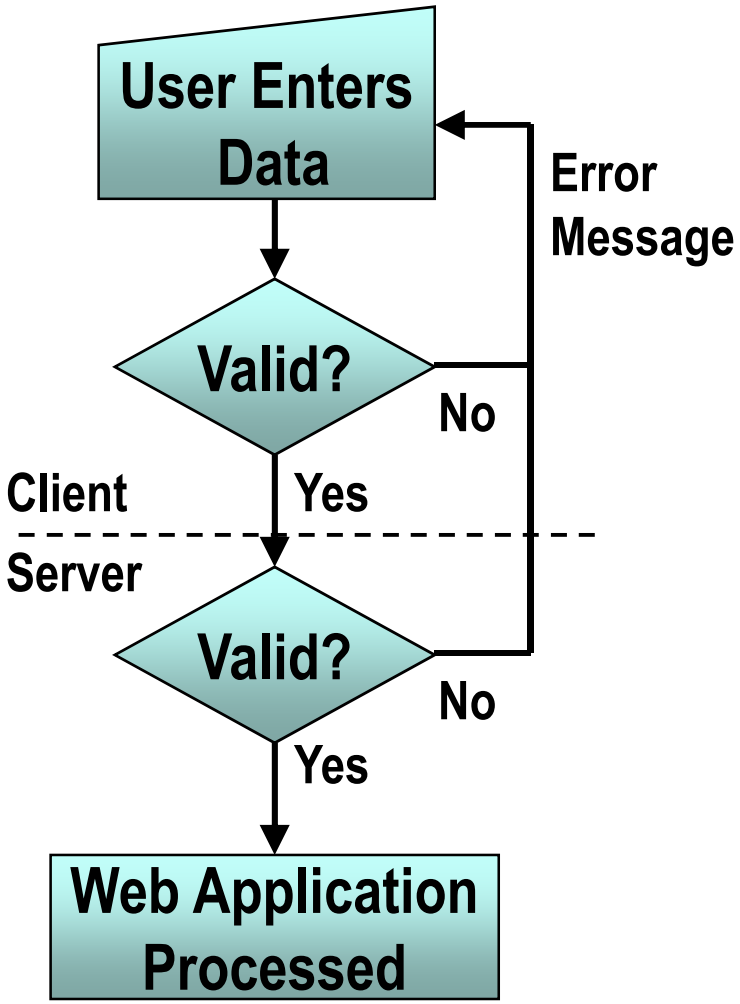
◆유효성 컨트롤(Validation Control) – 302p

- RequiredFieldValidator
- CompareValidator
- RangeValidator
- RegularValidator
- CustomValidator
- ValidationSummary



Client-Side and Server-Side Validation

- **ASP.NET can create both client-side and server-side validation**
- **Client-side validation**
  - Dependent on browser version
  - Instant feedback
  - Reduces postback cycles
- **Server-side validation**
  - Repeats all client-side validation
  - Can validate against stored data



Adding Validation Controls to a Web Form

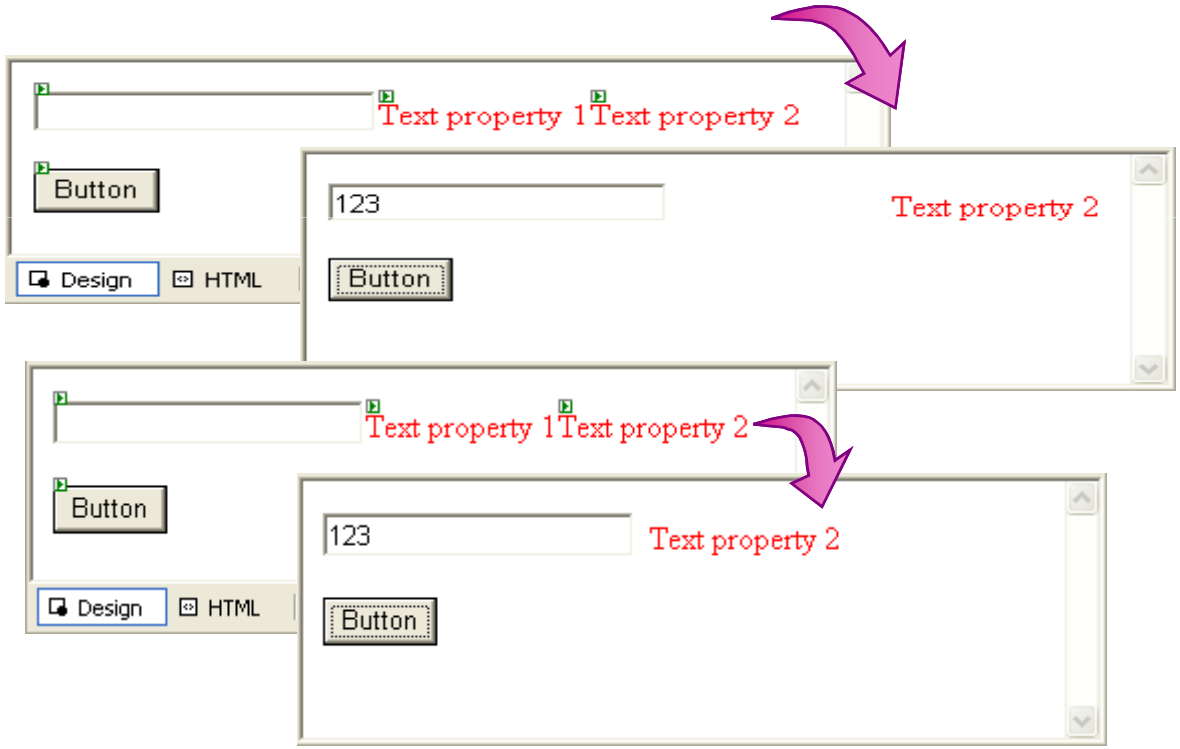
1. Add a validation control
2. Select the input control to validate
3. Set validation properties

```
<asp:TextBox id="txtName" runat="server" />
```

```
<asp:Type_of_Validator
  id="Validator_id"
  runat="server"
  ControlToValidate="txtName"
  ErrorMessage="Message_for_error_summary"
  Display="static|dynamic|none"
  Text="Text_to_display_by_input_control">
</asp:Type_of_Validator>
```

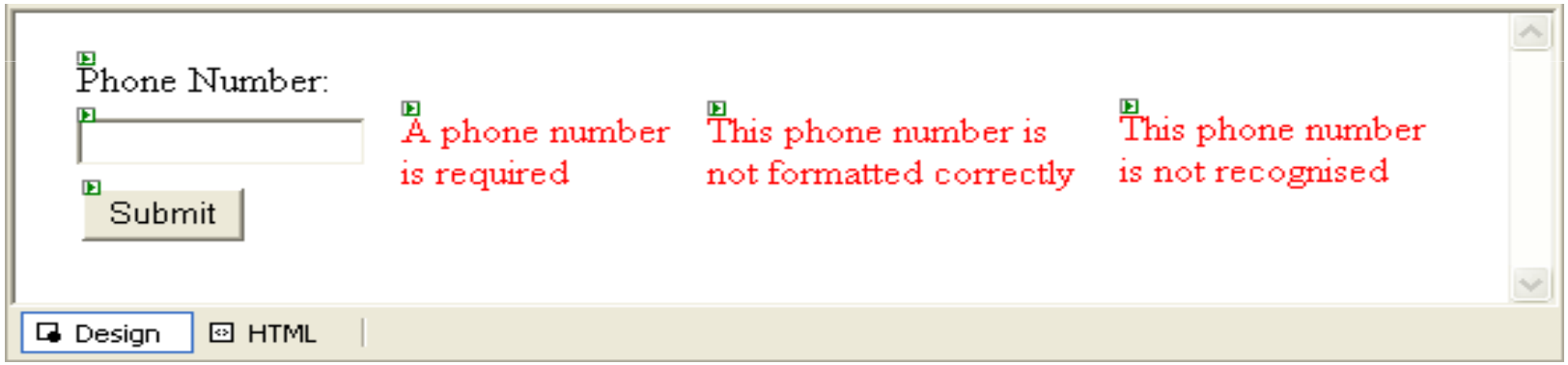
Positioning Validation Controls on a Web Form

- **Create error messages**
- **Select display mode**
  - Static
  - Dynamic



Combining Validation Controls

- Can have multiple validation controls on a single input control
- Only the RequiredFieldValidator checks empty controls



Validation Groups

- **Validation 컨트롤을 새로운 ValidationGroup 속성을 사용하여 그룹으로 묶을 수 있다**
  - Validation 컨트롤에서 구현
  - Button, LinkButton, ImageButton 컨트롤에서도 구현
- **그룹에 있는 모든 Validation 컨트롤들이 승인되어야만 페이지를 post back할 수 있다**
- **ASP.NET 1.x의 결함을 보완**

RegularExpressionValidator

- **Used when input must conform to a pre-defined pattern**
- **Visual Studio .NET includes patterns for:**
  - Telephone numbers
  - Postal codes
  - E-mail addresses

```
<asp:RegularExpressionValidator …
  ControlToValidate="US_PhoneNumber"…
  ValidationExpression="((\(\d{3}\) ?)|(\d{3}-))?\d{3}-\d{4} "
  …>*</asp:RegularExpressionValidator >
```

RegularExpressionValidator -Character Definition

a	Must use the letter a in lower case. Any letter that is not preceded by a backslash (\), or part of a range, is a requirement for that literal value.
1	Must use the number 1. Any number that is not preceded by a backslash (\), or part of a range, is a requirement for that literal value.
?	0 or 1 item.
*	0 to N items.
+	1 to N items (at least 1).
[0-n]	Integer value range from 0 to n.
{n}	Length must be n characters.
|	Separates multiple valid patterns.
\	The following character is a command character.
\w	Must have a character.
\d	Must have a digit.
\.	Must have a period.
@	An at sign (@).

CustomValidator

- Can validate on client-side, server-side, or both
  - ClientValidationFunction – 속성창에서 지정

```
<script language = “javascript">
function MyClientFunction(source, arguments)
 {
   alert("I am running on the client! ");
   var intValue = arguments.Value;
   if (intValue % 2 == 0) {
   arguments.IsValid = true;
   }
   else
   arguments.IsValid = false;
}
</script>
```

CustomValidator

- Can validate on client-side, server-side, or both
  - OnServerValidate – 이벤트로 처리

```
protected void CustomValidator1_ServerValidate(object source,
  ServerValidateEventArgs args)
{
  int intValue = Convert.ToInt16(args.Value);
  if (intValue%2 == 0)
  {
  args.IsValid = true;
  }
  else
  {
  args.IsValid = false;
  }
}
```

Page.IsValid

Polls all validation controls

```
private void cmdSubmit_Click(object s, System.EventArgs e)
{
  if (Page.IsValid)
  {    Message.Text = "Page is Valid!";
       // Perform database updates or other logic here
  }
}
```

ValidationSummary

- Collects error messages from all validation controls on the page
- Can display text and error messages
- Use Text="*" to indicate the location of the error

```
<asp:ValidationSummary id="valSummary"
  runat="server"
  HeaderText="These errors were found:"
  ShowSummary="True"
  DisplayMode="List"/>
```

◆사용자 정의 컨트롤(User Control) – 317p

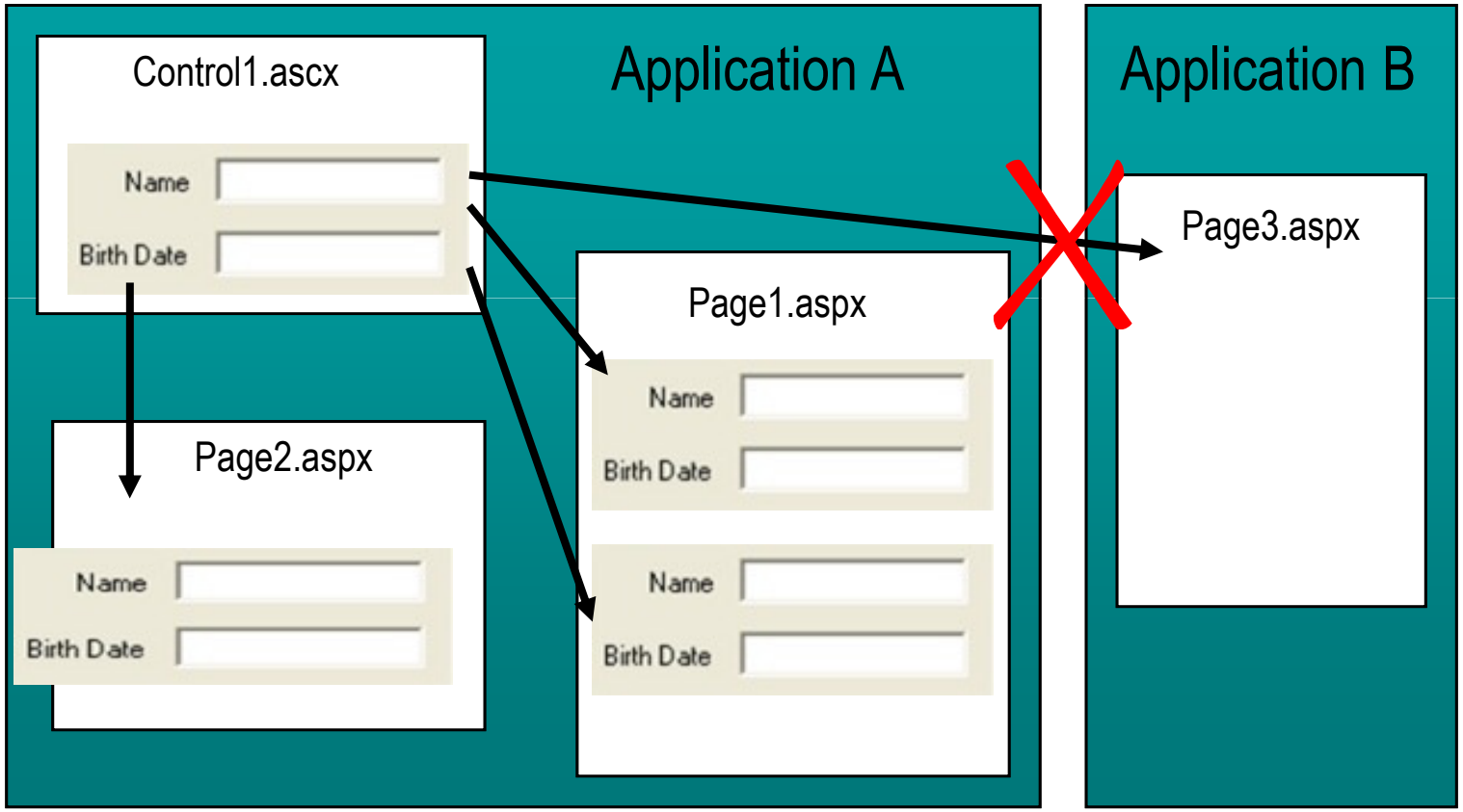
- User controls simplify the reuse of code and UI components within a Web application
- A user control is a user-defined Web server control with an .ascx extension
- Contains HTML, but not the <HTML>, <BODY>, or <FORM> tags

```
<%@ Control Language="c#" %>
```

- Contains code to handle its own events

Why Use User Controls?

- **Reuse user interface and code**



사용자 정의 컨트롤 만들기

- **Two methods for user control creation:**
  - Create a new user control using Visual Studio2005
    - 새 항목 추가 -> 웹 사용자 정의 컨트롤
  - Convert an existing ASP.NET page to a user control
- **Host page interacts with the control using properties**

```
public int pNum
{
  get
  {
    return
      Convert.ToInt32(TextBox1.Text);
  }
  set
  {
    TextBox1.Text =
      Convert.ToString(value);
  }
}
```

실습7 : NumberBox 컨트롤 만들기

- **BeforeUserControl.aspx 확인하기**
  - 솔루션 탐색기에서 기존 항목 추가로 BeforeUserControl.aspx를 추가
  - 디자인 모드에서 열어서 확인하고 브라우저에서 실행해서 테스트
- **NumberBox.ascx 만들기**
  - 솔루션 탐색기에서 새 항목 추가 – 웹 사용자 정의 컨트롤을 선택
    - 이름 : NumberBox.ascx
  - BeforeUserControl.aspx를 디자인 모드에서 열어서 Ctrl를 누르고 마우스를 클릭하여 TextBox1, Validation 컨틀롤들을 한꺼번에 선택하여 복사해둔다.
  - NumberBox.ascx를 디자인 모드에서 열어서 앞에서 복사한 컨트롤들을 붙이기를 수행한다.
  - 외부에서 컨트롤에 접근할 수 있게 public 속성 (pNum)을 구현한다.

```
public int pNum
{
  get
  {
    return  Convert.ToInt32(TextBox1.Text);
  }
  set
  {
    TextBox1.Text =  Convert.ToString(value);
  }
}
```

사용자 정의 컨트롤 사용하기

- **Use the @ Register directive to include a user control in an ASP.NET Page**

```
<%@ Register TagPrefix="demo"
TagName="validNum" Src="numberbox.ascx" %>
```

- **Insert the user control in a Web Form**

```
<demo:validNum id="num1" runat="server"/>
```

- **Use Get and Set properties of the user control**

```
num1.pNum = 5; //uses Set
x = num1.pNum; //uses Get
```

실습7 : NumberBox 컨트롤 사용하기

- ***BeforeUserControl.aspx* 만들기**
  - BeforeUserControl.aspx를 디자인 모드에서 열어서 편집메뉴에서 Ctrl를 누르고 마우스를 클릭하여 TextBox1, Validation 컨틀롤들을 한꺼번에 선택하여 삭제한다.
  - 솔루션 탐색기에서 NumberBox.ascx를 선택하여 BeforeUserControl.aspx 위로 드래깅한다.
  - 다음 이벤트 핸들러를 구현한다.

```
protected void Button1_Click(object sender, EventArgs e)
{
        int sum = NumberBox1.pNum + NumberBox2 .pNum;
        LabelSum.Text = sum.ToString();
}
```

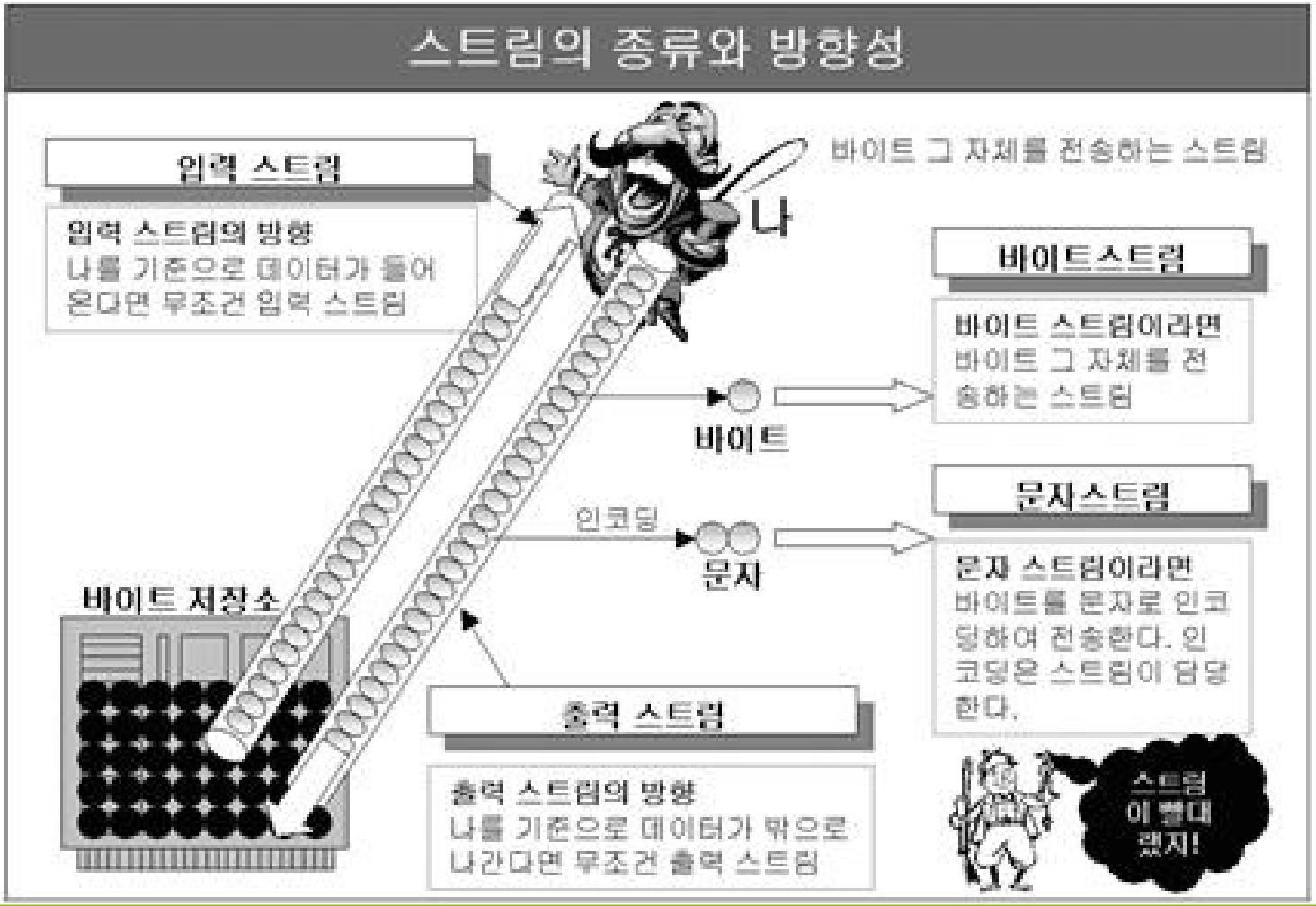
8장 파일 다루기

- .NET Framework의 파일 I/O 시스템
  - Stream
  - System.IO 네임스페이스
  - 바이트 스트림
  - 문자 스트림
  - 디렉토리 및 파일 제어

Streams

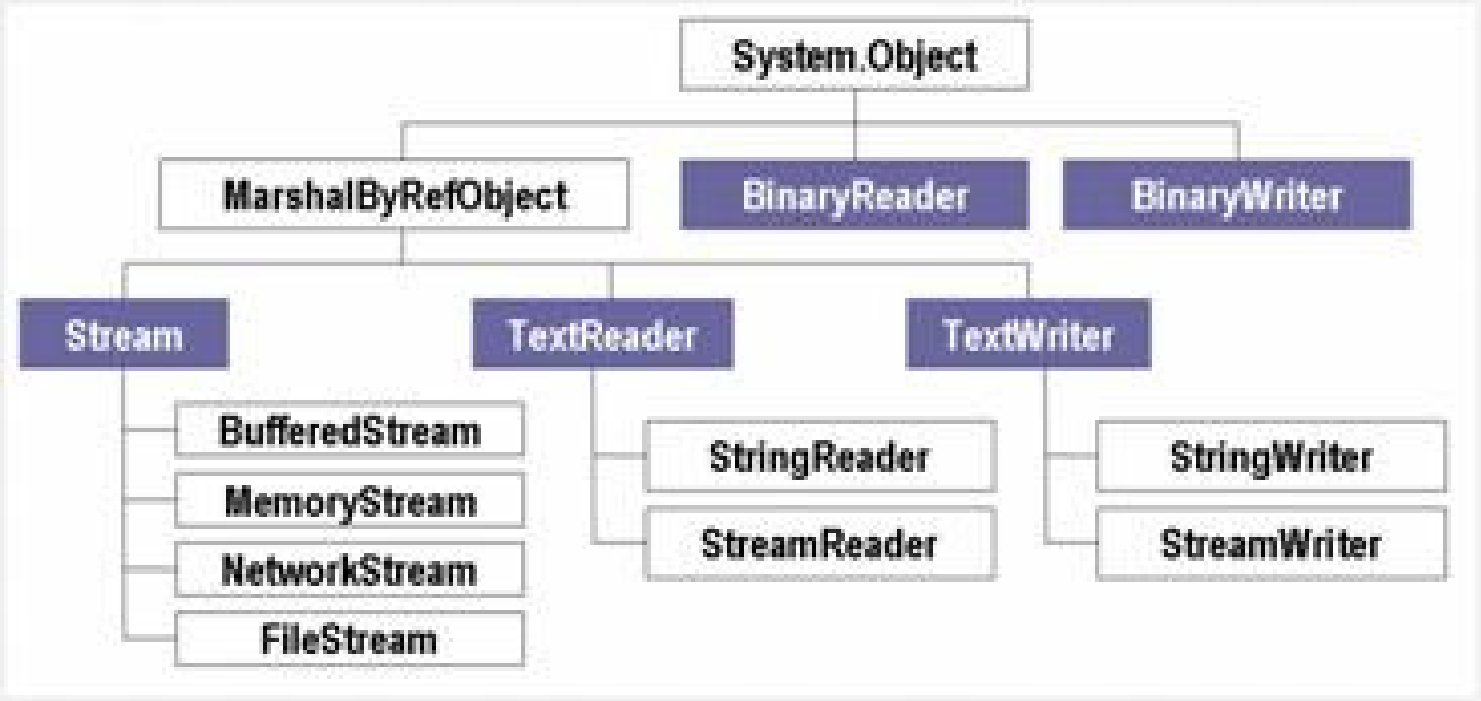
- **A Way to Read and Write Bytes from and to a Backing Store**
  - Stream classes inherit from **System.IO.Stream**
- **Fundamental Stream Operations: Read, Write, and Seek**
  - **CanRead**, **CanWrite**, and **CanSeek** properties
- **Some Streams Support Buffering for Performance**
  - **Flush** method outputs and clears internal buffers
- **Close Method Frees Resources**
  - **Close** method performs an implicit **Flush** for buffered streams
- **Stream Classes Provided by the .NET Framework**
  - **NetworkStream**, **BufferedStream**, **MemoryStream**, **FileStream**, **CryptoStream**

Stream 종류


스트림의 종류와 방향성
입력 스트림
입력 스트림의 방향
나를 기준으로 데이터가 들어
온다면 무조건 입력 스트림
나
바이트 그 자체를 전송하는 스트림
바이트스트림
바이트 스트림이라면
바이트 그 자체를 전
송하는 스트림
바이트
문자스트림
인코딩
문자
문자 스트림이라면
바이트를 문자로 인코
딩하여 전송한다. 인
코딩은 스트림이 담당
한다.
바이트 저장소
출력 스트림
출력 스트림의 방향
나를 기준으로 데이터가 밖으로
나간다면 무조건 출력 스트림
스트림
이 뭔대
했지!

System.IO

- FileStream
- StreamReader / StreamWriter
- DirectoryInfo
- FileInfo



바이트스트림 -327p

- **FileStream Class**
  - Is Used for Reading from and Writing to Files (byte로)
- **FileStream Constructor Parameter Classes**
  - **FileMode** – Open, Append, Create,CreateNew,OpenOrCreate
  - **FileAccess** – Read, ReadWrite, Write
  - **FileShare** – None, Read, ReadWrite, Write

```
FileStream f = new FileStream("data.txt", FileMode.Open,
        FileAccess.Read, FileShare.Read);
```

```
FileStream f = File.OpenRead("data.txt");
```

```
FileInfo aFile = new FileInfo("data.txt");
FileStream f =  aFile.OpenRead();
```

FileStream- File 위치

- **Random Access to Files by Using the Seek Method**
  - Specified by byte offset
  - Offset is relative to seek reference point: Begin, Current, End

```
public long Seek(long offset,SeekOrigin origin);
```

```
aFile.Seek(2,SeekOrigin.Current);
aFile.Seek(-5,SeekOrigin.End);
```

FileStream- 데이터 읽기

- **Byte 데이터만 지원, 이미지나 사운드 파일**

```
public int Read(byte[] array, int offset, int count);
```

- **System.Text.Encoding**

Encoding - I

- System.Text.Encoding

```
using System.IO;
using System.Text ;
//…
string req = "Hello 안녕";
Console.WriteLine ("기본문장: {0}\n",req);

//byte배열로 UTF8형식으로 encoding한 배열 반환
byte[] results1 = Encoding.UTF8 .GetBytes (req);
Console.WriteLine ("UTF8:");
PrintByteArray(results1);
Console.WriteLine (Encoding.UTF8 .GetString (results1));
Console.WriteLine ();

//byte배열로 Default형식으로 encoding한 배열 반환
byte[] results3 = Encoding.Default  .GetBytes (req);
Console.WriteLine ("Default:");
PrintByteArray(results3);
Console.WriteLine (Encoding.Default .GetString (results3));
Console.WriteLine ();

Console.WriteLine ("Encoding Name: {0}",Encoding.Default .EncodingName );
Console.WriteLine ("Encoding Body: {0}",Encoding.Default .BodyName );
Console.WriteLine ("Encoding CodePage: {0}",Encoding.Default .CodePage );
```

Encoding-II


```
public static void PrintByteArray(byte[] input)
{
      for (int i = 0; i< input.Length ;i++)
      {
            Console.Write (input[i]);
            if (input.Length -1 > i ) Console.Write (",");
      }
      Console.WriteLine ();
}
```

임의 접근 파일로부터 데이터 읽기 예제


```
using System.IO;
using system.Text;
 …
byte[]  byData = new byte[100];
char[] charData = new char[100];

try
{
   FileStream aFile = new FileStream("../../Class1.cs",FileMode.Open);
   aFile.Seek(55,SeekOrigin.Begin);
   aFile.Read(byData,0,100);
}
catch(IOException e)
{
  Console.WriteLine(e.Message);
}

//byte배열을 화면에 보여주기 위해 char배열로 변환
Decoder d = Encoding.Default.GetDecoder(); //System.Text.Decorder
d.GetChars(byData,0,byData.Length,charData,0);

Console.WriteLine(charData);
```

임의 접근 파일로부터 데이터 쓰기 예제


```
using System.IO;
using system.Text;
 …
byte[]  byData = new byte[100];
char[] charData = new char[100];

FileStream aFile = new
  FileStream("temp.txt",FileMode.OpenOrCreate);

charData = "Hello world".ToCharArray();

//UTF8 encoding 기반으로 encoder개체를 생성
Encoder  e = Encoding.UTF8.GetEncoder();
// char배열을 byte배열로 변환
e.GetBytes(charData,0,charData.Length,byData,0,true);

aFile.Seek(0,SeekOrigin.Begin);
aFile.Write(byData,0,byData.Length);
```

◆문자스트림 - 330p

- **StreamWriter**
  - 텍스트 파일을 생성
- **String, Integer , Floating Point Number등을 Write**
  - Write()
  - WriteLine()
- 파일 닫기(Close)

```
//...
StreamWriter sw = File.CreateText("MyFile.txt");
sw.WriteLine ("This is my file");
sw.WriteLine (
   "I can write ints {0} or floats {1}", 1, 4.2);
sw.Close();
//...
```

StreamReader - Text File 읽기

- 생성

```
FileStream f = File.Open("data.txt",FileMode.Open);
StreamReader sr = new StreamReader(f);
```

```
FileStream f = new FileStream("data.txt", FileMode.Open);
StreamReader sr = new StreamReader(f);
```

```
StreamReader sr = new StreamReader("data.txt",
        System.Text.Encoding.Default); // 한글
```

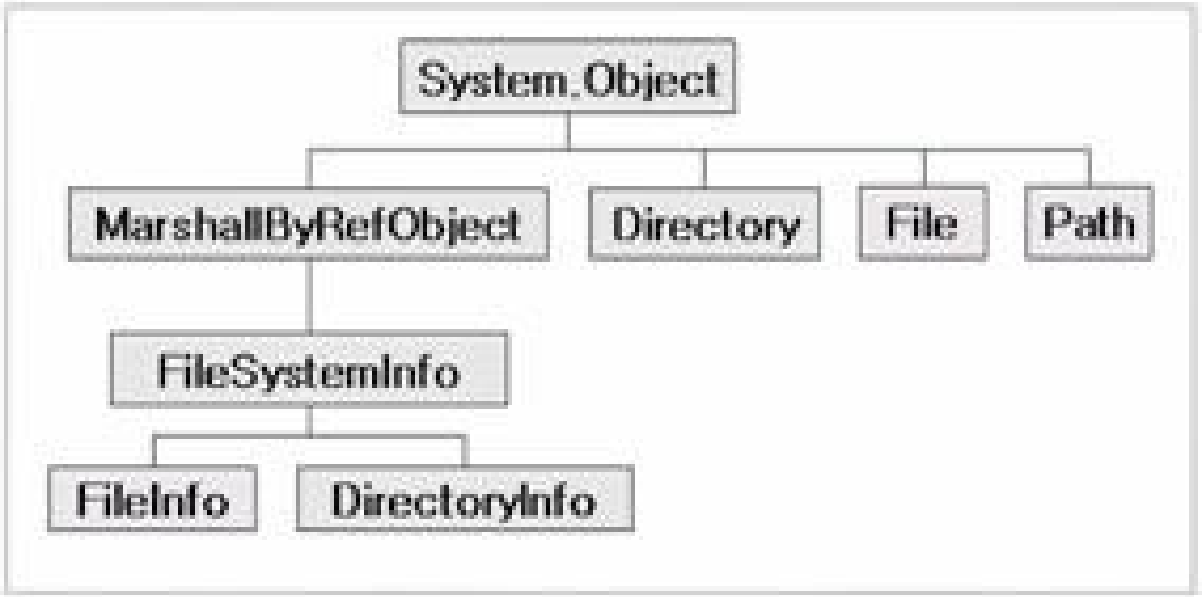
- **Read Text from a File and Output It to the Console**
  - Read() : 한 글자씩 읽기
  - ReadLine() : 한 라인씩 읽기
  - ReadToEnd() : 파일 끝까지 읽기

StreamReader - Text File 읽기


```
//...

StreamReader sr = new StreamReader("c:\\test.txt",
      System.Text.Encoding.Default); // 한글

String input;
while ((input=sr.ReadLine())!=null) {
   Console.WriteLine(input);
}
Console.WriteLine (
   "The end of the stream has been reached.");
sr.Close();
//...
```

◆디렉토리 및 파일 제어-333p

- File 과 FileInfo 클래스
- Directory 와 DirectoryInfo 클래스



File과 FileInfo 클래스

- **File- Static Method만**
  - 파일 생성 ,복사,삭제,이동 , 열기에 관련된 메서드들 제공
- **FileInfo - Instance Methods 만**
  - 파일 생성 ,복사,삭제,이동 , 열기에 관련된 메서드들 제공
  - 객체를 재사용시 보안 체크안함

```
FileStream aStream = File.Create("foo.txt");
```

FileInfo 사용 예제


```
using System.IO;
….
FileInfo fileInfo = new FileInfo(“c:\\test.txt”);
if (fileInfo.Exists)
{
   Console.WriteLine(“폴더이름:{0}”,fileInfo.Directory);
   Console.WriteLine(“전체이름:{0}”,fileInfo.FullName);
   Console.WriteLine(“파일이름:{0}”,fileInfo.Name);
   Console.WriteLine(“확장자이름:{0}”,fileInfo.Extension);
   Console.WriteLine(“생성일:{0}”,fileInfo.CreationTime);
   Console.WriteLine(“크기[바이트]:{0}”,fileInfo.Length);
   Console.WriteLine(“파일속성{0}”,fileInfo.Attribute);

}
else
   Console.WriteLine(“파일이 없습니다.”);
```

파일 관리 예제


```
using System;
using System.IO;

class MainClass
{
   public static void Main()
   {
         //File Handling
         FileInfo sourceFile = new FileInfo ("c:\\test.txt");
         if (sourceFile.Exists )
         {
            FileInfo copyFile =
                  sourceFile.CopyTo("c:\\temp\\copytest.txt",true);
             copyFile.MoveTo(@"c:\move.text");
            if (copyFile.Exists )
                  copyFile.Delete ();
         }
         else
         {
                  Console.WriteLine ("파일이 없습니다");
         }
   }
}
```

Directory와 DirectoryInfo 클래스

- **Directory - Static 메서드들 만**
  - 디렉토리 생성, 이동 및 삭제 등 메서드 제공
- **DirectoryInfo- Instance 메서드들만**
  - 디렉토리 생성, 이동 및 삭제 등 메서드 제공
  - 객체 재사용시 보안 체크 안함

```
DirectoryInfo dir = new DirectoryInfo(".");
foreach (FileInfo f in dir.GetFiles("*.cs"))
{
  String name = f.FullName;
}
```

DirectoryInfo 사용 예제


```
using System.IO;
….
DirectoryInfo dirInfo = new DirectoryInfo(@“c:\winnt\system”);
if (dirInfo.Exists)
{
   Console.WriteLine(“디렉토리이름:{0}”,dirInfo.Name);
   Console.WriteLine(“생성일:{0}”,dirInfo.CreationTime);
   Console.WriteLine(“루트”,dirInfo.Root);
   Console.WriteLine(“부모”,dirInfo.Parent);
   Console.WriteLine(“디렉토리속성{0}”,dirInfo.Attribute);

}
else
   Console.WriteLine(“디렉토리가 없습니다.”);
```

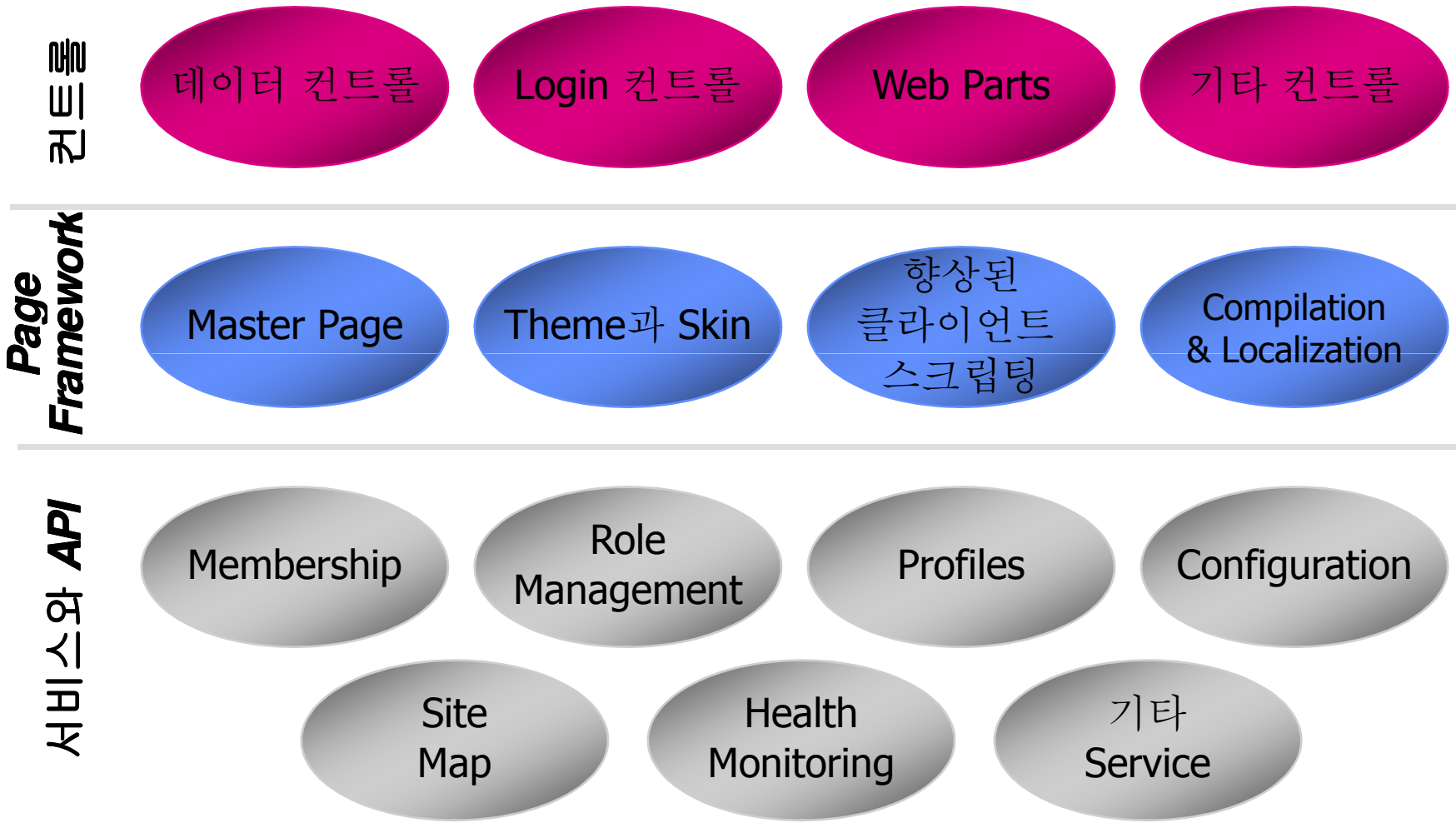
디렉토리 관리 예제


```
using System.IO;
….
DircetoryInfo dir= new DirectoryInfo(“c:\\temp”);
if (!dir.Exists)
{
   Console.WriteLine(“{0} 디렉토리를  생성합니다.”,dir);
   dir.Create();

   Console.WriteLine(“하위디렉토리를 생성합니다”);
   dir.CreateSubDirectory(@”temp”);
   dir.CreateSubDirectory(@”nested\temp”);

   Console.WriteLine(“{0} 디렉토리 하위까지 모두 지웁니다.”,dir);
   dir.Delete(true);
}
```

디렉토리 탐색하기(dir)


```
using System.IO;
….
Console.WriteLine(“디렉토리명을 입력하시오”);
String reqDir = Console.ReadLine();

DircetoryInfo dir= new DirectoryInfo(reqDir);
if (dir.Exists)
{
   FileSystemInfo[] results = dir.GetFileSystemInfos();
   foreach(FileSystemInfo fsi  in results)
      Console.WrietLine(“{0}\t{1}fsi.Attribute,fsi.FullName);
   Console.WriteLine(“모두 {0}개의 파일이 있습니다.”,results.Length);
}
else //System.IO.DirectoryNotFoundException으로도 처리가능
   Console.WriteLine(“해당 디렉토리가 존재하기 않습니다.”);
```

ASP.NET 2.0의 새로운 기능



9장 데이터베이스와 연동하기

- 데이터 베이스 들어가기
- 데이터 소스 컨트롤을 사용한 데이터베이스 연동
- **ADO.NET**을 사용한 데이터베이스 연동

- 기본 **SQL문 – DDL(Data Definition Language)**
  - CREATE *object_name*
  - ALTER *object_name*
  - DROP *object_name*

```
CREATE DATABASE Northwind ON default = 256
```

```
USE northwind
CREATE TABLE customer
(cust_id int, company varchar(40),
contact varchar(30), phone char(12) )
GO
```

- 기본 **SQL문 – DML(Data Manipulation Language)**
  - SELECT
  - INSERT
  - UPDATE
  - DELETE

```
USE northwind
SELECT categoryid, productname, productid, unitprice
FROM products
```

```
INSERT INTO customer (cust_id, company, contact, phone)
VALUES (1,'HP','Manager','34702114')
```

```
UPDATE customer
SET company = 'AAA'
WHERE cust_id = 1
```

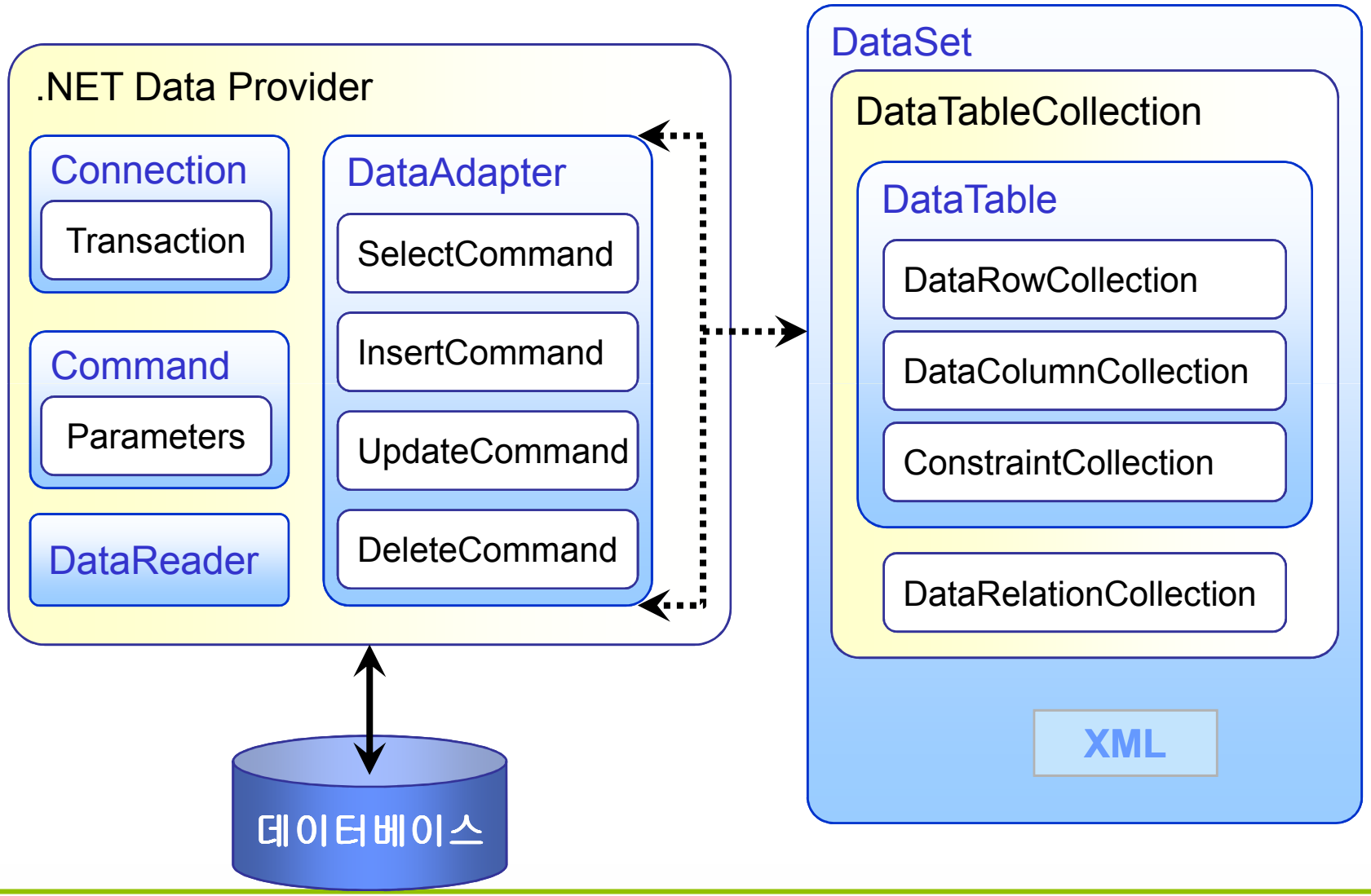
```
DELETE FROM sales
WHERE DATEDIFF(Year,ord_date,GETDATE()) >= 3
```

ADO.NET 개요

- 데이터베이스 연결하는 사용하는 클래스들
- ADO(Active Data Object) 의 다음 버전
- Disconnected 환경에 초점

네임스페이스	설명
System.Data	DataSet, DataTable, DataRow 등의 중요 클래스들을 가지고 있다.
System.Data.SqlClient	NET Framework Data Provider for SQL Server는 SQL Server에 있는 데이터를 사용하는 데 필요한 클래스들을 가지고 있다.
System.Data.OleDb	NET Framework Data Provider for OLE DB는 OLE DB Provider를 사용하여 OLE DB 데이터를 액세스하는 데 필요한 클래스를 가지고 있다.
System.Data.OracleClient	.NET Framework Data Provider for Oracle은 Oracle 8.1.7이상의 Oracle 데이터 소스를 사용하는 데 필요한 클래스들을 가지고 있다.
System.Data.Odbc	.NET Framework Data Provider for ODBC는 관리되는 공간의 Driver를 사용하여 데이터 에 액세스하는 데 사용되는 클래스를 가지고 있다.
System.Common	.NET Data Provider에 의해 공유된 클래스가 있다.
System.Data.SqlTypes	SQL Server에서 사용되는 고유 데이터 타입들을 가지고 있다.

ADO.NET 객체 모델


.NET Data Provider
Connection
Transaction
Command
Parameters
DataReader
DataAdapter
SelectCommand
InsertCommand
UpdateCommand
DeleteCommand
데이터베이스
DataSet
DataTableCollection
DataTable
DataRowCollection
DataColumnCollection
ConstraintCollection
DataRelationCollection
XML

◆간단해진 데이터 바인딩

- 데이터 바인딩 표현식이 간단해지고 **XML** 데이터 바인딩이 지원됨

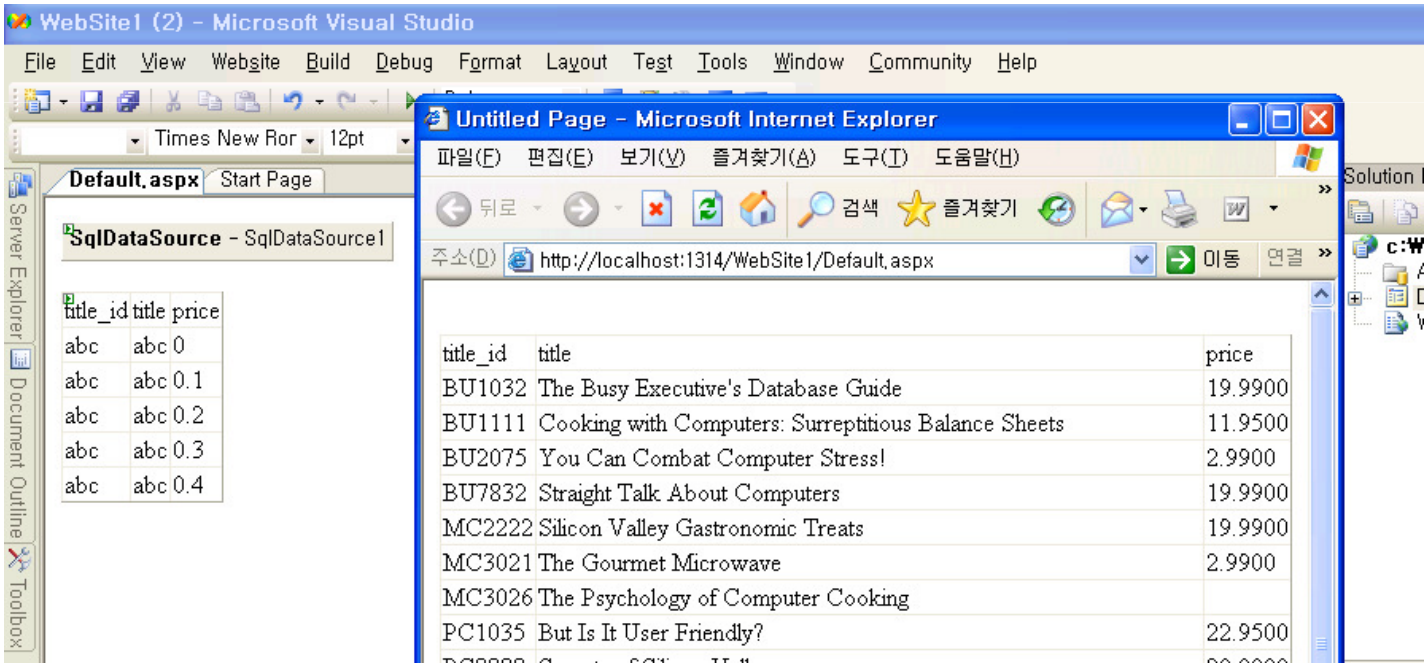
```
<!-- ASP.NET 1.x data binding expression -->
<%# DataBinder.Eval (Container.DataItem, "Price") %>

<!-- Equivalent ASP.NET 2.0 data binding expression -->
<%# Eval ("Price") %>

<!-- XML data binding -->
<%# XPath ("Price") %>
```

ASP.NET 1.x 데이터 바인딩


```
<asp:DataGrid ID="MyDataGrid" RunAt="server" />
  ...
<script language="C#" runat="server">
void Page_Load (Object sender, EventArgs e)
{
    SqlConnection connection = new SqlConnection
        ("server=localhost;database=pubs;integrated security=true");
    try {
        connection.Open ();
        SqlCommand command = new SqlCommand
            ("select title_id, titles, price from titles");
        MyDataGrid.DataSource = command.ExecuteReader ();
        MyDataGrid.DataBind ();
    }
    finally {
        connection.Close ();
    }
}
</script>
```

ASP.NET 2.0 데이터 바인딩


```
<asp:SqlDataSource ID="Titles" RunAt="server"
  ConnectionString="server=localhost;database=pubs;integrated security=true"
  SelectCommand="select title_id, title, price from titles" />
<asp:DataGrid DataSourceID="Titles" RunAt="server" />
```

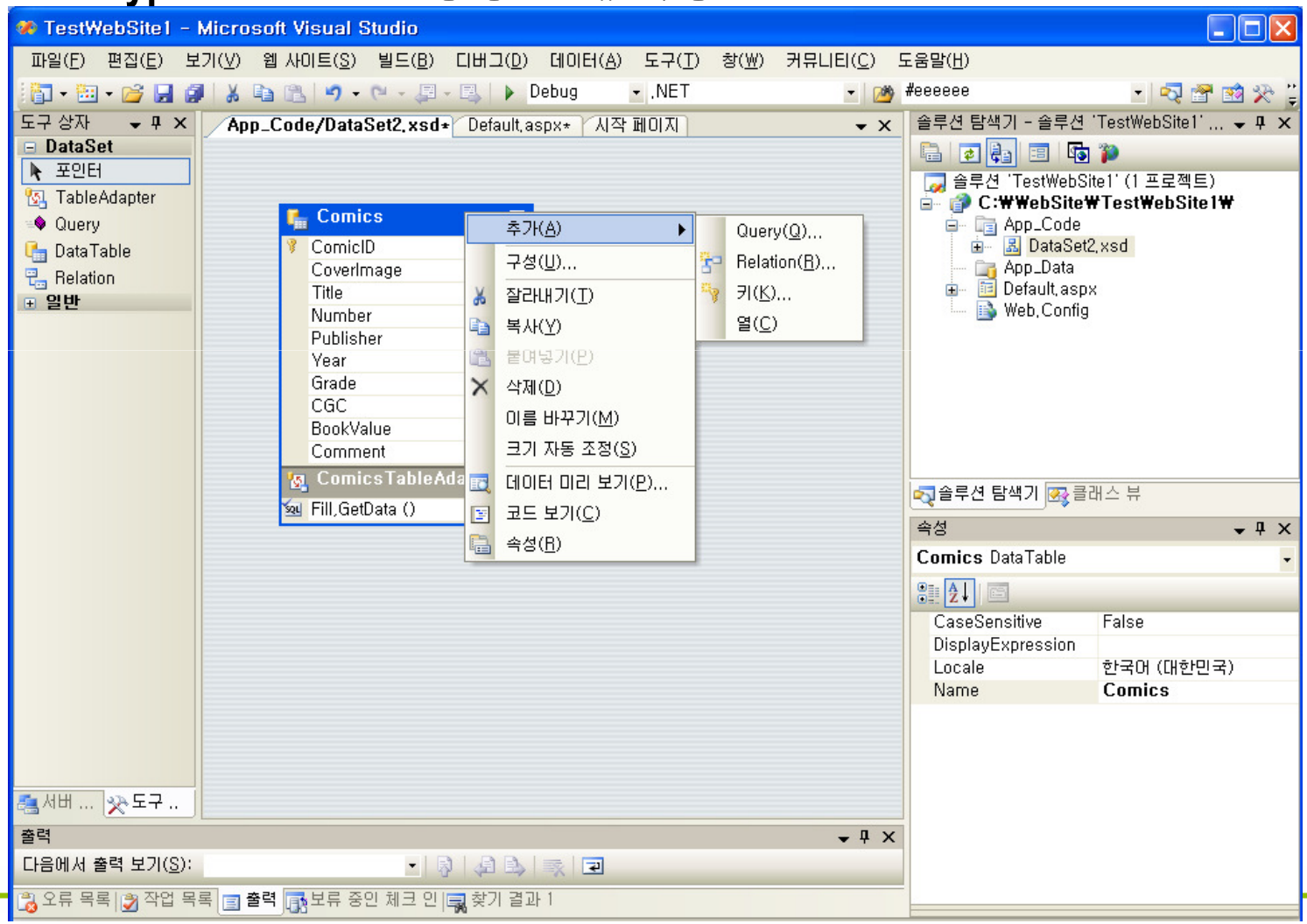


◆Visual Studio 2005 Wizards

- **데이터 집합 디자이너(DataSet Designer)**
  - 솔루션 탐색기에서 새 항목 추가 - 데이터 집합 추가
  - 새로운 Typed DataSet을 생성할 때 사용
- **TableAdapter 구성 마법사**
  - Typed DataSet에 있는 TableAdapter를 생성하고 편집할 때 사용
  - TableAdapter는 데이터 테이블에 있는 데이터를 가져오고 저장하기 위해 DataAdapter를 사용
- **데이터 소스 구성 마법사**
  - 데이터 소스 컨트롤을 사용하여 데이터 소스를 구성할 때 사용

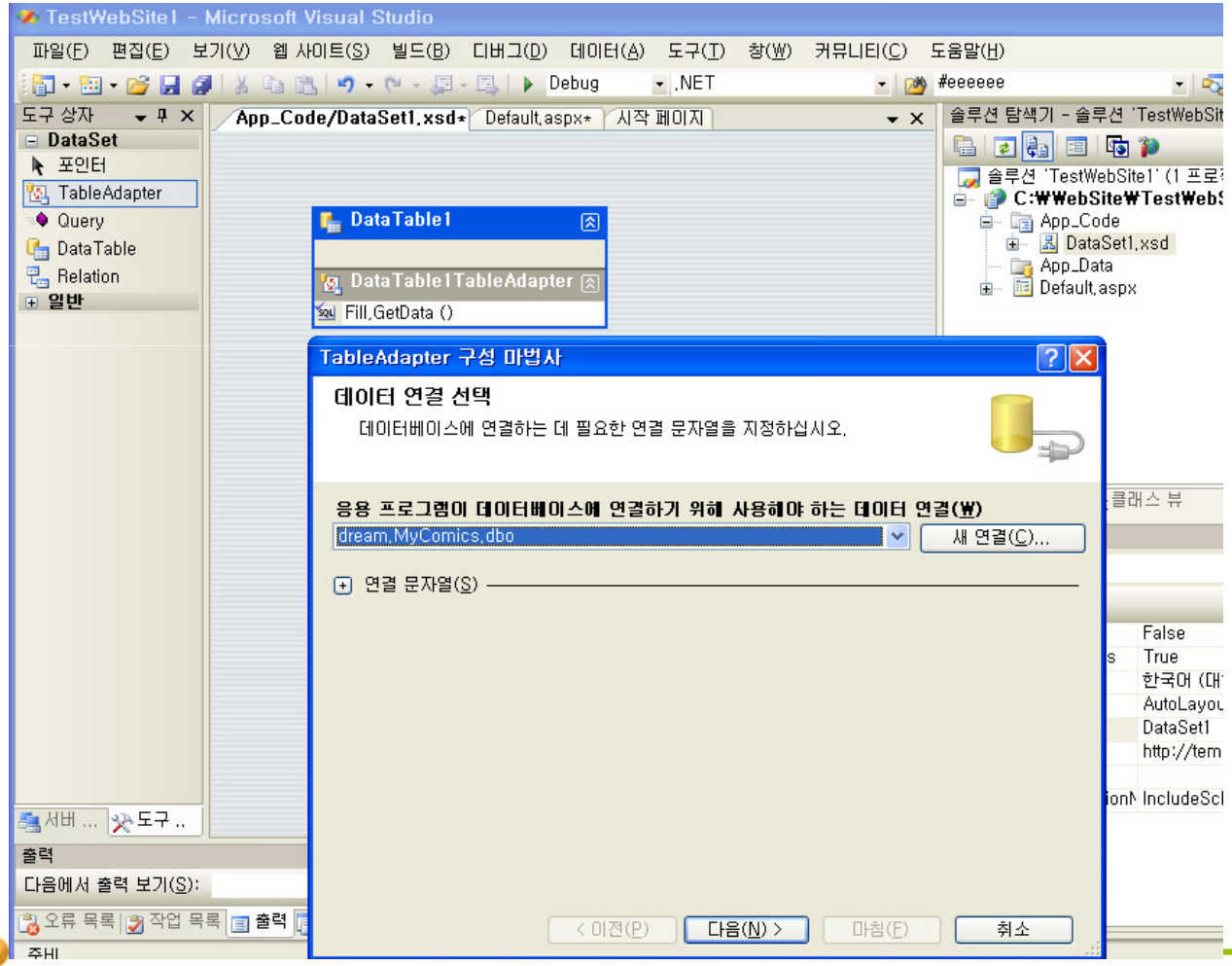
데이터 집합 디자이너

- 새로운 **Typed DataSet**을 생성할 때 사용



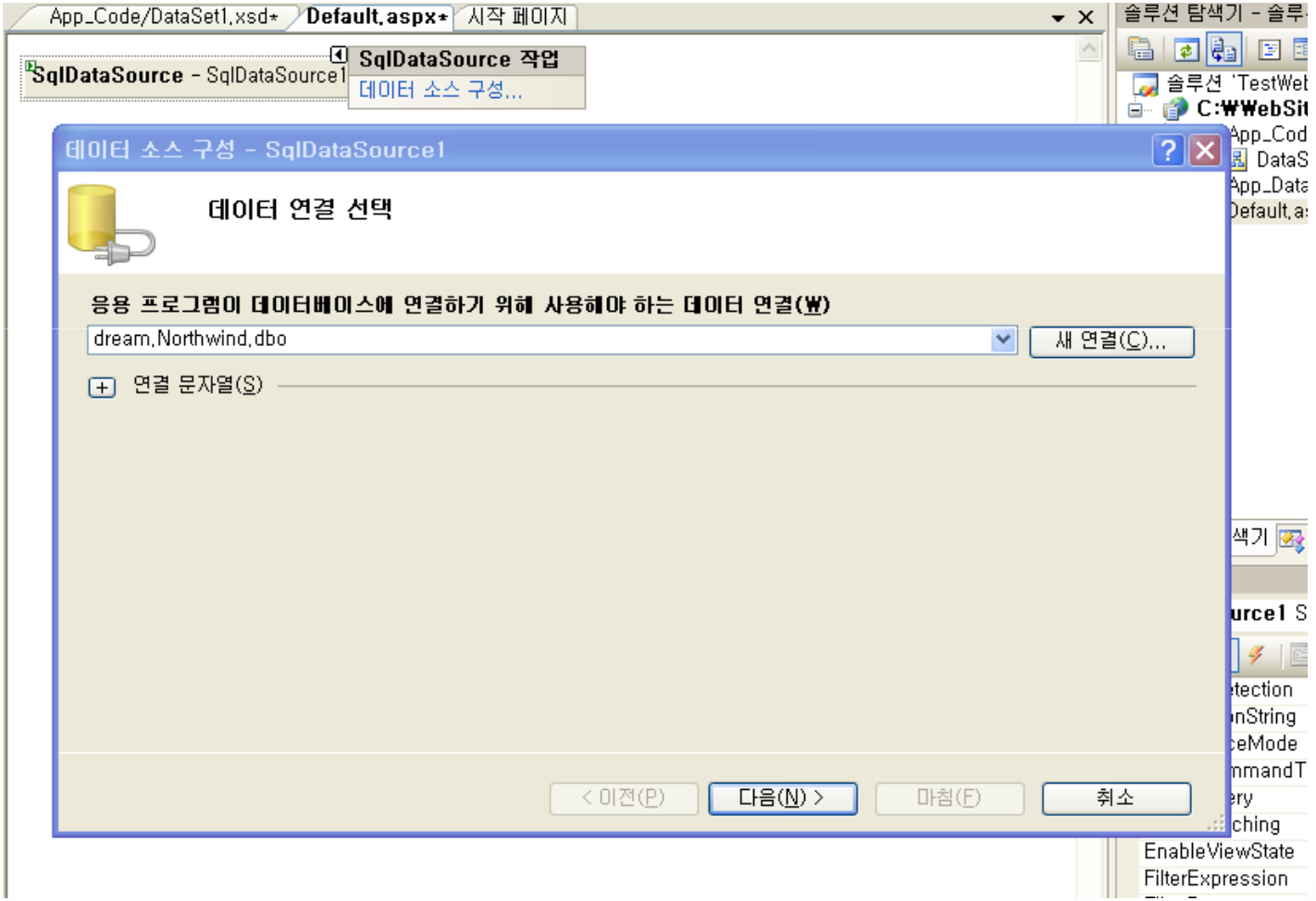
TableAdapter 구성 마법사

- Typed DataSet에 있는 TableAdapter를 생성하고 편집할 때 사용



데이터 소스 구성 마법사

- 데이터 소스 컨트롤을 사용하여 데이터 소스를 구성할 때 사용



데이터 소스 컨트롤- 347p

- 선언적인 (no-code) 데이터 바인딩
- 데이터 소스 구성 마법사 사용

이름	설명
SqlDataSource	SQL 데이터베이스를 연결하는 데이터바인딩 컨트롤
AccessDataSource	Access 데이터베이스를 연결하는 데이터바인딩 컨트롤
XmlDataSource	XML 데이터를 연결하는 데이터바인딩 컨트롤
ObjectDataSource	데이터 컴포넌트를 연결하는 데이터바인딩 컨트롤
SiteMapDataSource	Sitemap 데이터를 연결하는 데이터바인딩 컨트롤

◆데이터 소스 컨트롤을 사용한 데이터베이스 연동 – 365p

- **SqlDataSource** 사용하기
- 데이터 바인딩 컨트롤
  - 리스트 컨트롤들
    - BulletedLIst, RadioButtonList,DropDownList 등)
  - Repeater
  - DataList
  - GridView
  - DetailsView

◆SqlDataSource 사용하기

- SQL 데이터베이스를 선언적 데이터 바인딩으로 연결
  - Managed Provider에 의해 임의의 데이터베이스에 연결
- 양방향 데이터 바인딩
  - SelectCommand는 쿼리를 정의
  - InsertCommand, UpdateCommand,DeleteCommand는 업데이트를 정의
- 쿼리 결과를 캐싱할 수도 있다
- 매개변수 처리

```
<asp:SqlDataSource ID="Titles" RunAt="server"
  ConnectionString="server=localhost;database=pubs;integrated security=true"
  SelectCommand="select title_id, title, price from titles" />
<asp:DataGrid DataSourceID="Titles" RunAt="server" />
```

```
<asp:SqlDataSource ID="Titles" RunAt="server"
  ConnectionString="..."
  SelectCommand="select title_id, title, price from titles"
  ProviderName="System.Data.OracleClient" />
```

SqlDataSource 속성

이름	설명
ConnectionString	Data Source를 연결할 때 사용되는 연결 문자열
SelectCommand	쿼리를 수행할 때 사용되는 명령
InsertCommand	삽입할 때 사용되는 명령
UpdateCommand	업데이트할 때 사용되는 명령
DeleteCommand	삭제할 때 사용되는 명령
DataSourceMode	DataSet이 사용되는 지 DataReader 가 사용되는 지를 명시 (디폴트 = DataSet)
ProviderName	Provider를 명시 (디폴트 = SQL Server .NET provider)

SqlDataSource와 캐싱

- SqlDataSource는 다음 속성들을 통해 선언적인 캐싱을 지원한다

이름	설명
EnableCaching	캐시를 활성화할 것인지를 명시(디폴트 = false)
CacheDuration	결과가 몇 초 동안이나 캐시되는 지를 나타냄
CacheExpirationPolicy	캐시 지속 기간이 sliding인지 absolute인지를 명시
CacheKeyDependency	캐시 키에 종속적인 캐시를 생성
SqlCacheDependency	지정된 데이터베이스 엔터티에 종속성을 생성

- EnableCaching="true"는 DataSourceMode="DataSet"인 경우에만 유효

쿼리 결과를 캐싱하기


```
<asp:SqlDataSource ID="Countries" RunAt="server"
  ConnectionString="server=localhost;database=northwind;..."
  SelectCommand="select distinct country from customers order by country"
  EnableCaching="true" CacheDuration="60" />
<asp:DropDownList ID="MyDropDownList" DataSourceID="Countries"
  DataTextField="country" AutoPostBack="true" RunAt="server" />
```

- **EnableCaching=True인 경우에만 캐싱됨**
- **위의 예제는 ASP.NET Application Cache에 SqlDataSource 쿼리 결과를 60초 동안 캐시함**

◆ 데이터 컨트롤

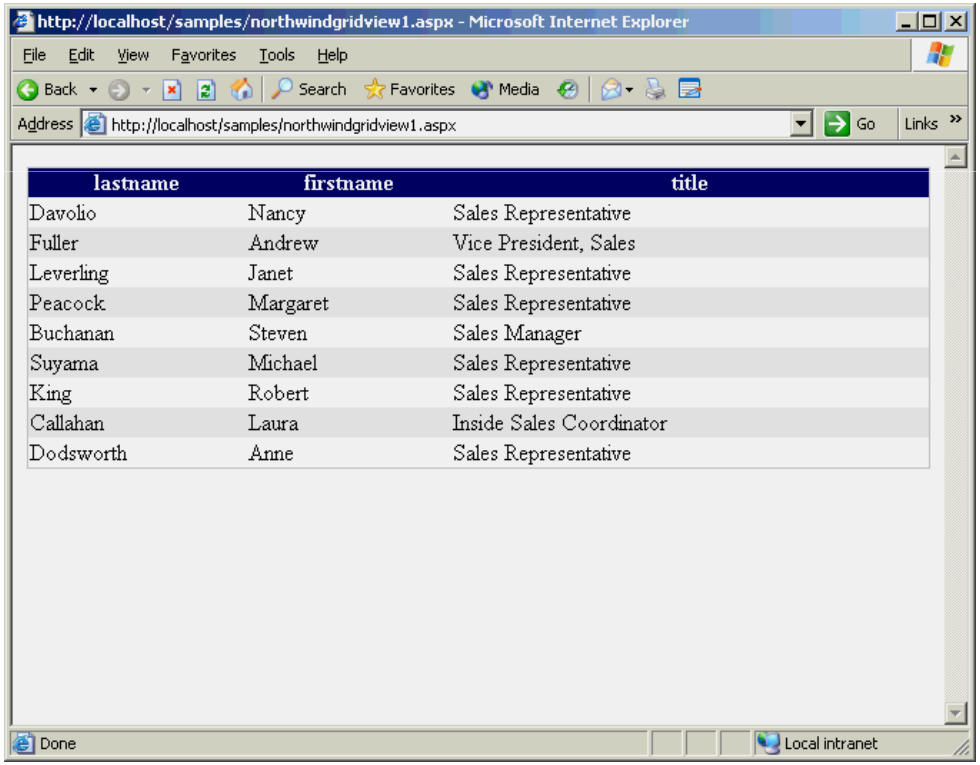
- 리스트 컨트롤들
  - BulletedLIst, RadioButtonList,DropDownList 등
- **Repeater 컨트롤**
- **DataList 컨트롤**
- **GridView 컨트롤**
- **DetailsView 컨트롤**

◆ GridView 컨트롤 (p380)

- **향상된 DataGrid 컨트롤**
  - 레코드들의 집합을 HTML 테이블형태로 보여줌
- **정렬, 페이징, 검색, 업데이트, 삭제, 추가 등의 기능을 제공**
- **CheckBoxFields 같은 다양한 형태의 필드 타입을 지원**
  - <Columns> 요소에 선언되어야 함
- **UI를 변경할 수 있다**

GridView 예제


```
<asp:SqlDataSource ID="Employees" RunAt="server"
  ConnectionString="server=localhost;database=northwind;..."
  SelectCommand="select lastname, firstname, title from employees" />
<asp:GridView DataSourceID="Employees" Width="100%" RunAt="server" />
```



lastname	firstname	title
Davolio	Nancy	Sales Representative
Fuller	Andrew	Vice President, Sales
Leverling	Janet	Sales Representative
Peacock	Margaret	Sales Representative
Buchanan	Steven	Sales Manager
Suyama	Michael	Sales Representative
King	Robert	Sales Representative
Callahan	Laura	Inside Sales Coordinator
Dodsworth	Anne	Sales Representative

-위 그림은 <% Page Theme="BasicBlue" %>를 사용한 예제임

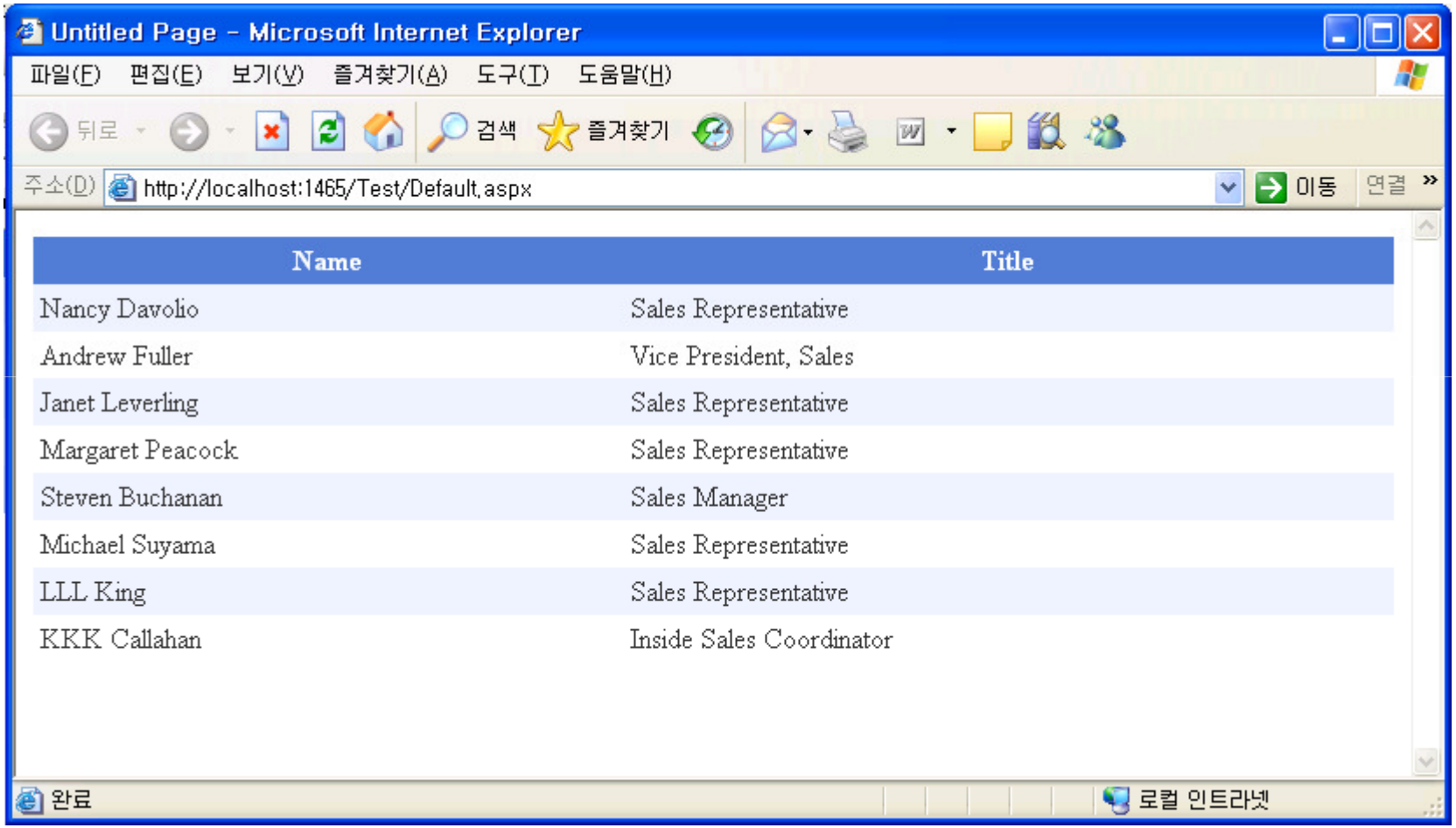
GridView 필드 타입(p383)

이름	설명
BoundField	데이터 소스에 있는 필드를 텍스트 컬럼으로 표현
ButtonField	Buttons 컬럼 (push button, image 또는 link)
CheckBoxField	Bool를 Check boxes 컬럼으로 표현
CommandField	GridView 데이터를 선택하고 편집하기 위한 컨트롤을 표현
HyperLinkField	하이이퍼링크 컬럼
TemplateField	HTML 템플릿을 사용하는 컬럼

필드 타입 명시


```
<asp:SqlDataSource ID="Employees" RunAt="server"
  ConnectionString="server=localhost;database=northwind;..."
  SelectCommand="select lastname, firstname, title from employees" />
<asp:GridView DataSourceID="Employees" Width="100%" RunAt="server"
  AutoGenerateColumns="false" >
  <Columns>
    <asp:TemplateField HeaderText="Name">
      <ItemTemplate>
        <%# Eval ("firstname") + " " + Eval ("lastname") %>
      </ItemTemplate>
    </asp:TemplateField>
    <asp:BoundField HeaderText="Title" DataField="title" />
  </Columns>
</asp:GridView>
```

결과


Untitled Page - Microsoft Internet Explorer
파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말(H)
뒤로
검색
즐겨찾기
주소(D)
http://localhost:1465/Test/Default.aspx
이동
연결
Name
Title
Nancy Davolio
Sales Representative
Andrew Fuller
Vice President, Sales
Janet Leverling
Sales Representative
Margaret Peacock
Sales Representative
Steven Buchanan
Sales Manager
Michael Suyama
Sales Representative
LLL King
Sales Representative
KKK Callahan
Inside Sales Coordinator
완료
로컬 인트라넷

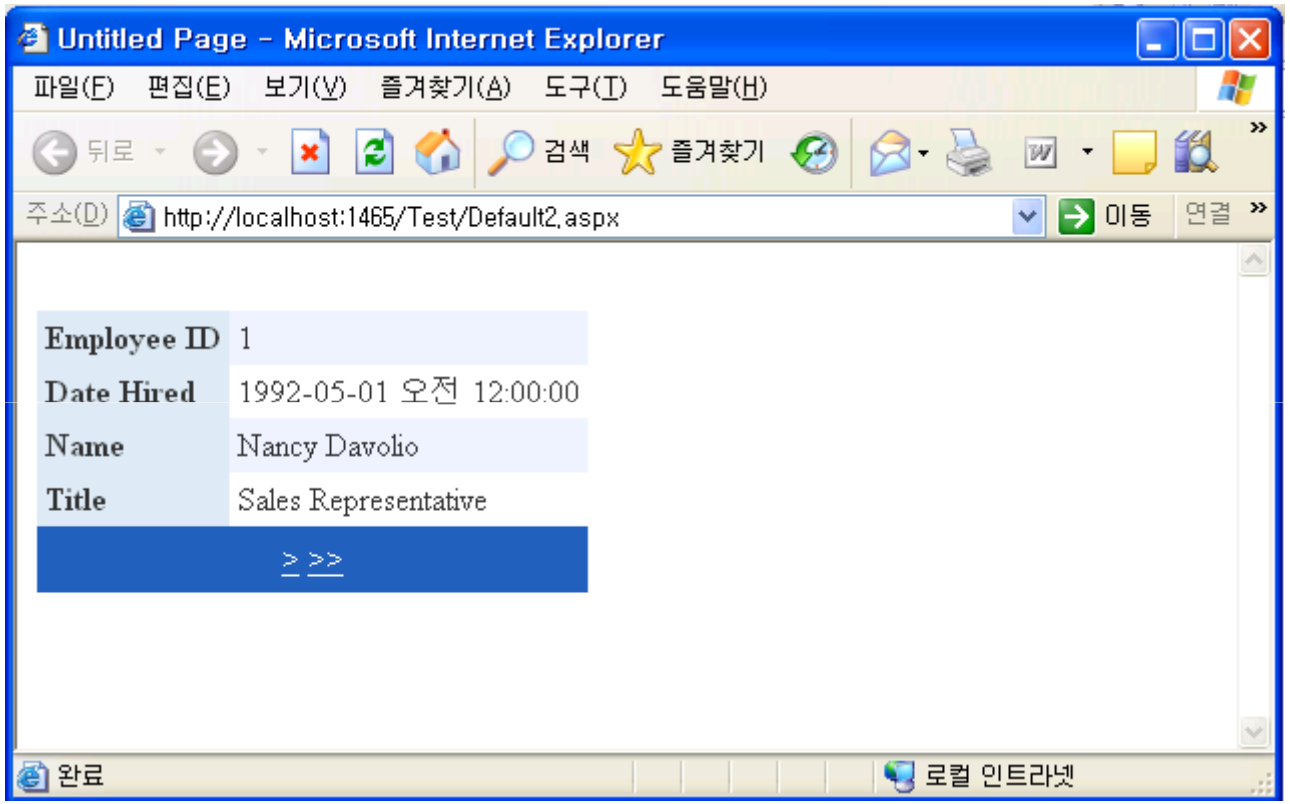
DetailsView 컨트롤

- 개별적인 레코드를 보여줌
  - Master-detail views를 위해 GridView와 같이 사용
  - 또는 GridView없이 개별적인 레코드를 보여주기 위해 사용
- 페이징, 삽입, 업데이트, 삭제 기능을 제공
- GridView에서 선언된 다양한 형태의 필드 타입 사용 가능
  - <Fields> 요소에 선언되어야 함
- UI를 변경할 수 있다

DetailsView 예제

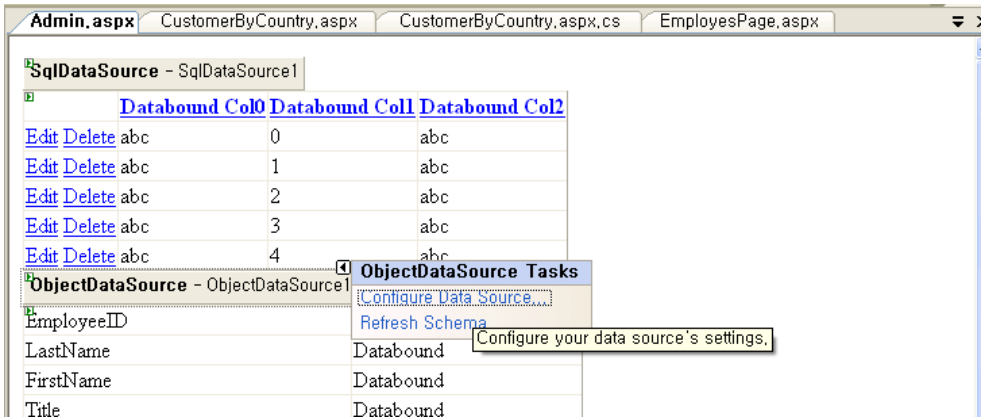

```
<asp:SqlDataSource ID="Employees" RunAt="server"
  ConnectionString="server=localhost;database=northwind;..."
  SelectCommand="select employeeid, ... from employees" />
<asp:DetailsView DataSourceID="Employees" RunAt="server"
  AllowPaging="true" AutoGenerateRows="false"
  PagerSettings-Mode="NextPreviousFirstLast">
  <Fields>
   <asp:BoundField HeaderText="Employee ID" DataField="employeeid" />
    <asp:BoundField HeaderText="Date Hired" DataField="hiredate" />
    <asp:TemplateField HeaderText="Name">
      <ItemTemplate>
        <%# Eval ("firstname") + " " + Eval ("lastname") %>
      </ItemTemplate>
    </asp:TemplateField>
    <asp:BoundField HeaderText="Title" DataField="title" />
  </Fields>
</asp:DetailsView>
```

결과

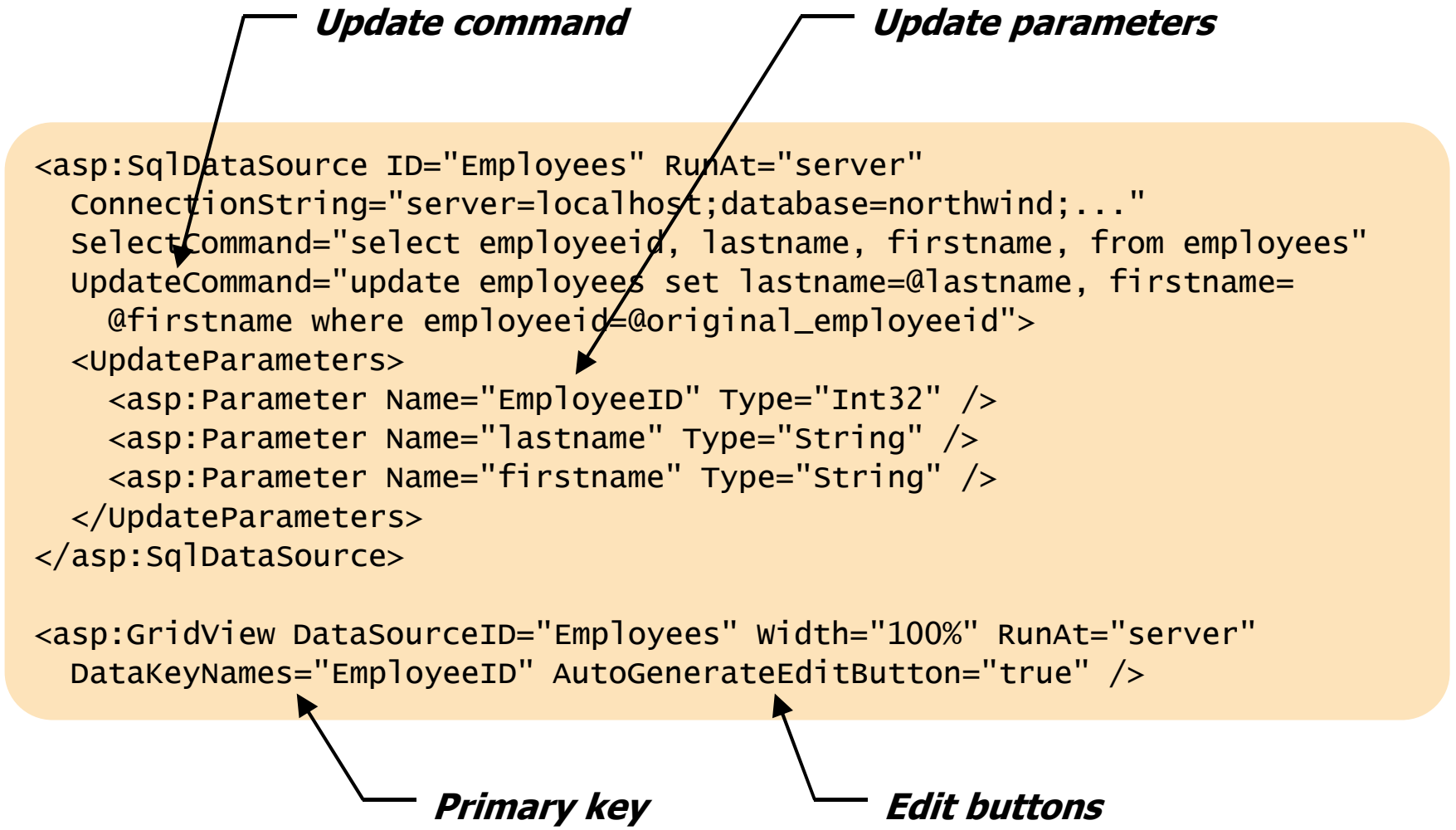

Untitled Page - Microsoft Internet Explorer
파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말(H)
뒤로
검색
즐겨찾기
주소(D) http://localhost:1465/Test/Default2.aspx
이동
연결
Employee ID 1
Date Hired 1992-05-01 오전 12:00:00
Name Nancy Davolio
Title Sales Representative
> >>
완료
로컬 인트라넷

삽입, 업데이트, 삭제

- **Data 컨트롤: 편집할 수 있는 UI를 제공**
  - AutoGenerateXxxButton 속성
  - Insert/EditRowStyle 속성
- **Data source 컨트롤: 편집 로직을 제공**
  - Insert/Update/DeleteCommand 속성
  - Insert/Update/DeleteParameters 속성
  - Inserting/ed, Updating/ed, Deleting/ed 이벤트
- **Visual Studio의 데이터 소스 구성 사용**

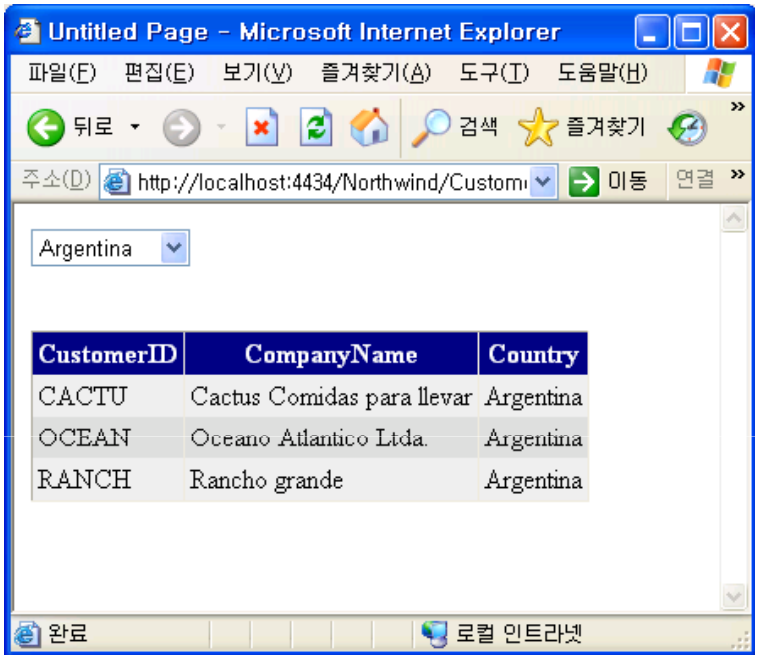


GridViews에서 편집하기



Parameterized Commands

- **XxxParameters 속성으로 매개변수화된 데이터 베이스 명령을 처리할 수 있다**
  - 예: Drop-down list에서 항목을 선택하거나 쿼리 문자열(Query String) 매개변수로 SelectCommand의 WHERE절 값을 얻는다
  - 예: GridView로부터 DeleteCommand의 WHERE절 값을 얻는다
- **XxxParameter 타입으로 매개변수 값의 근원지를 알 수 있다**



XxxParameters 속성

이름	설명
SelectParameters	SelectCommand에 사용되는 매개변수를 지정
InsertParameters	InsertCommand 에 사용되는 매개변수를 지정
UpdateParameters	UpdateCommand 에 사용되는 매개변수를 지정
DeleteParameters	DeleteCommand 에 사용되는 매개변수를 지정
FilterParameters	FilterExpression 에 사용되는 매개변수를 지정

-FilterExpression은 SqlDataSource가 쿼리 결과를 캐싱하는 경우에만 동작
SelectCommand에 의해 리턴되는 데이터를 필터

XxxParameter 타입

이름	설명
Parameter	데이터 필드를 매개변수로 바인드
ControlParameter	컨트롤 속성를 매개변수로 바인드
CookieParameter	쿠키 값를 매개변수로 바인드
FormParameter	폼 필드를 매개변수로 바인드
QueryStringParameter	쿼리 매개변수를 매개변수로 바인드
SessionParameter	세션 변수(Session Variable)를 매개변수로 바인드

-위 타입은 SelectParameters, InsertParameters, UpdateParameters, DeleteParameters, FilterParameters에서 사용됨

ControlParameter 사용하기


```
<asp:SqlDataSource ID="Countries" RunAt="server"
  ConnectionString="server=localhost;database=northwind;..."
  SelectCommand="select distinct country from customers order by country" />
<asp:SqlDataSource ID="Customers" RunAt="server"
  ConnectionString="server=localhost;database=northwind;..."
  SelectCommand="select * from customers where country=@Country">
  <SelectParameters>
    <asp:ControlParameter Name="Country" ControlID="MyDropDownList"
      PropertyName="SelectedValue" />
  </SelectParameters>
</asp:SqlDataSource>
<asp:DropDownList ID="MyDropDownList" DataSourceID="Countries"
  DataTextField="country" AutoPostBack="true" RunAt="server" />
<asp:DataGrid DataSourceID="Customers" RunAt="server" />
```

-**DropDowList**에서 선택한 항목이 두번째 **SqlDataSource**의 **Select**문의 **where**절 매개변수로 사용됨

-**<SelectParameters>**의 **<ControlParameter>**가 **Select**문의 **@Country**에 해당한다.

SqlDataSource로 저장 프로시저 호출하기


```
<asp:SqlDataSource ID="Countries" RunAt="server"
  ConnectionString="server=localhost;database=northwind;..."
  SelectCommand="proc_GetCountries" />
<asp:SqlDataSource ID="Customers" RunAt="server"
  ConnectionString="server=localhost;database=northwind;..."
  SelectCommand="proc_GetCustomers">
  <SelectParameters>
    <asp:ControlParameter Name="Country" ControlID="MyDropDownList"
      PropertyName="SelectedValue" />
  </SelectParameters>
</asp:SqlDataSource>
<asp:DropDownList ID="MyDropDownList" DataSourceID="Countries"
  DataTextField="country" AutoPostBack="true" RunAt="server" />
<asp:DataGrid DataSourceID="Customers" RunAt="server" />
```

```
CREATE PROCEDURE proc_GetCustomers
@Country nvarchar (32) AS
  SELECT * FROM Customers
  WHERE Country = @Country
GO
```

```
CREATE PROCEDURE proc_GetCountries AS
  SELECT DISTINCT Country
  FROM Customers
  ORDER BY Country
GO
```

실습 9.1: SqlDataSource로 저장프로시저 호출하기 I

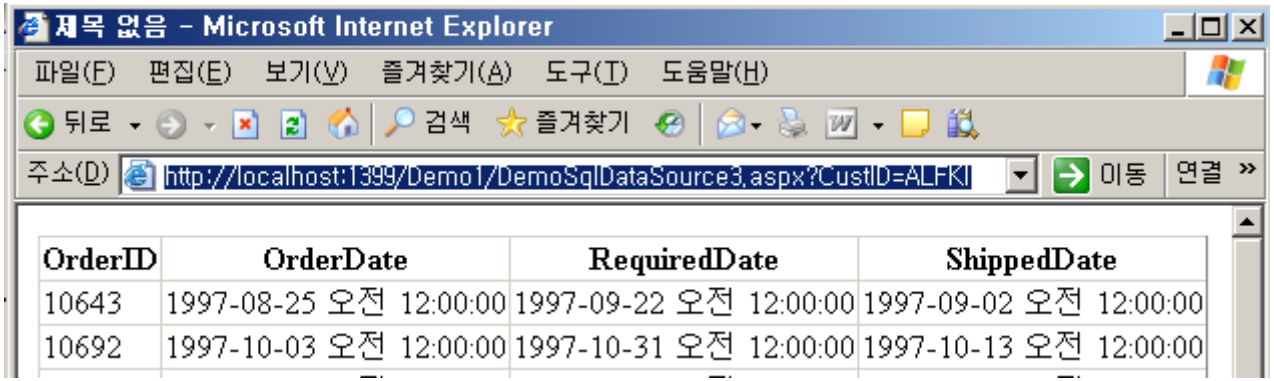
- **SqlDataSource 사용**
  - 솔루션 탐색기에서 새 항목 추가로 DemoSqlDataSource3.aspx 추가
  - SqlDataSource컨트롤을 DemoSqlDataSource3.aspx 위로 드래깅해서 추가
  - 데이터 소스 구성 하기
    - 데이터 연결 선택 : NorthwindConnectionString
    - Select문 구성에서 다음 항목 선택
      - 사용자 지정 SQL문 또는 저장 프로시저 지정 선택
      - SELECT문에서 저장 프로시저 선택
        : CustOrdersOrders 선택
      - 매개변수 정의
      - 매개변수 소스 : Query String
      - QueryStringField : CustID
  - 도구 상자에서 GridView 컨트롤을 SqlDataSource1밑에 추가
    - 데이터 소스 선택 : SqlDataSource1
    - 자동서식 지정하기

실습 9.1: SqlDataSource로 저장프로시저 호출하기II

- 실행 결과 확인하기 : 솔루션 탐색기에서 브라우저 보기로 확인, QueryString 다음과 같이 입력하기

  http://localhost:1399/Demo1/DemoSqlDataSource3.aspx?CustID=ALFKI

- DemoSqlDataSource3.aspx를 소스 보기로 열어서 추가된 태그 확인해보기



◆ObjectDataSource

- 데이터 컴포넌트를 선언적 바인딩으로 연결
  - 중간 계층의 데이터 액세스 컴포넌트
  - 데이터 액세스 코드를 UI Layer로부터 분리하여 관리
- 양방향 데이터 바인딩
  - SelectCommand,InsertCommand, UpdateCommand,DeleteCommand
- 쿼리 결과를 캐싱할 수도 있다
- 매개변수 처리

ObjectDataSource 속성

이름	설명
TypeName	데이터 컴포넌트 타입 이름 (클래스 이름)
SelectMethod	쿼리를 수행하기 위해 데이터 컴포넌트에서 호출되는 메소드
InsertMethod	삽인을 수행하기 위해 데이터 컴포넌트에서 호출되는 메소드
UpdateMethod	엡데이드을 수행하기 위해 데이터 컴포넌트에서 호출되는 메소드
DeleteMethod	삭제을 수행하기 위해 데이터 컴포넌트에서 호출되는 메소드
EnableCaching	캐시를 활성화할 것인지를 명시(디폴트 = false)

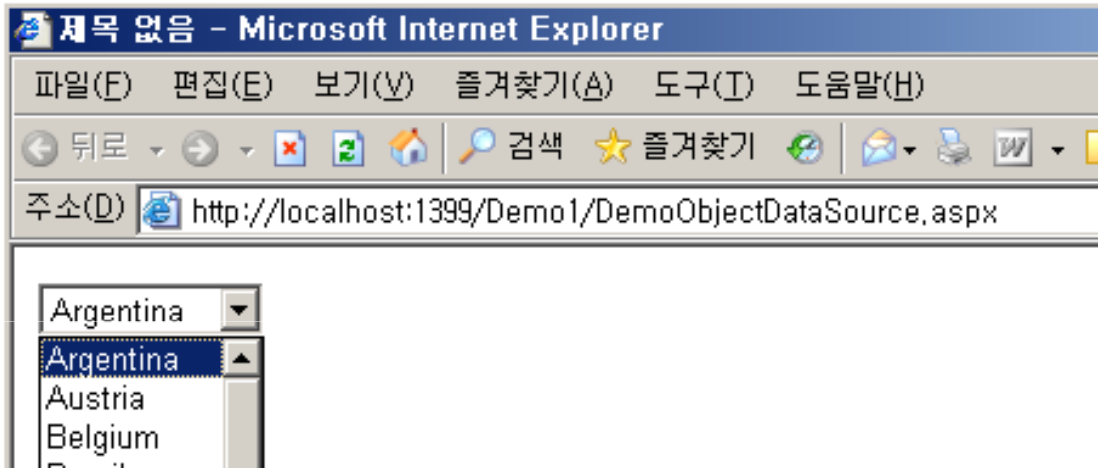
ObjectDataSource 속성

이름	설명
CacheDuration	데이터가 몇 초 동안이나 캐시되는 지를 나타냄
SqlCacheDependency	명시된 데이터베이스 엔터티에 종속성을 생성
SelectParameters	SelectMethod의 매개변수들을 지정
InsertParameters	InsertMethod의 매개변수들을 지정
UpdateParameters	UpdateMethod의 매개변수들을 지정
DeleteParameters	DeleteMethod의 매개변수들을 지정

실습 9.2 : ObjectDataSource:Data Component -I

- **SqlDataSource 사용**
  - 솔루션 탐색기에서 새 항목 추가로 DemoObjectDataSource.aspx 추가
  - SqlDataSource컨트롤을 DemoObjectDataSource.aspx 위로 드래깅
  - 데이터 소스 구성 하기
    -데이터 연결 선택 : NorthwindConnectionString
    - Select문 구성에서 다음 항목 선택
      테이블 또는 뷰의 열 지정 : Customers 선택
      열 : Country 선택
      고유한 행만 반환 선택
      Select문 확인 :SELECT DISTINCT [Country] FROM [Customers]
    - 쿼리 테스트
  - 도구 상자에서 DropDownList 컨트롤을 SqlDataSource1밑에 추가
    - 데이터 소스 선택 : SqlDataSource1
    - AutoPostBack 선택
  - 실행 결과 확인하기 : 솔루션 탐색기에서 브라우저 보기로
  - DemoObjectDataSource.aspx 를 소스 보기로 열어서 추가된 태그 확인해보기

실습 9.2: ObjectDataSource:Data Component -II



실습 9.2: ObjectDataSource:Data Component- III

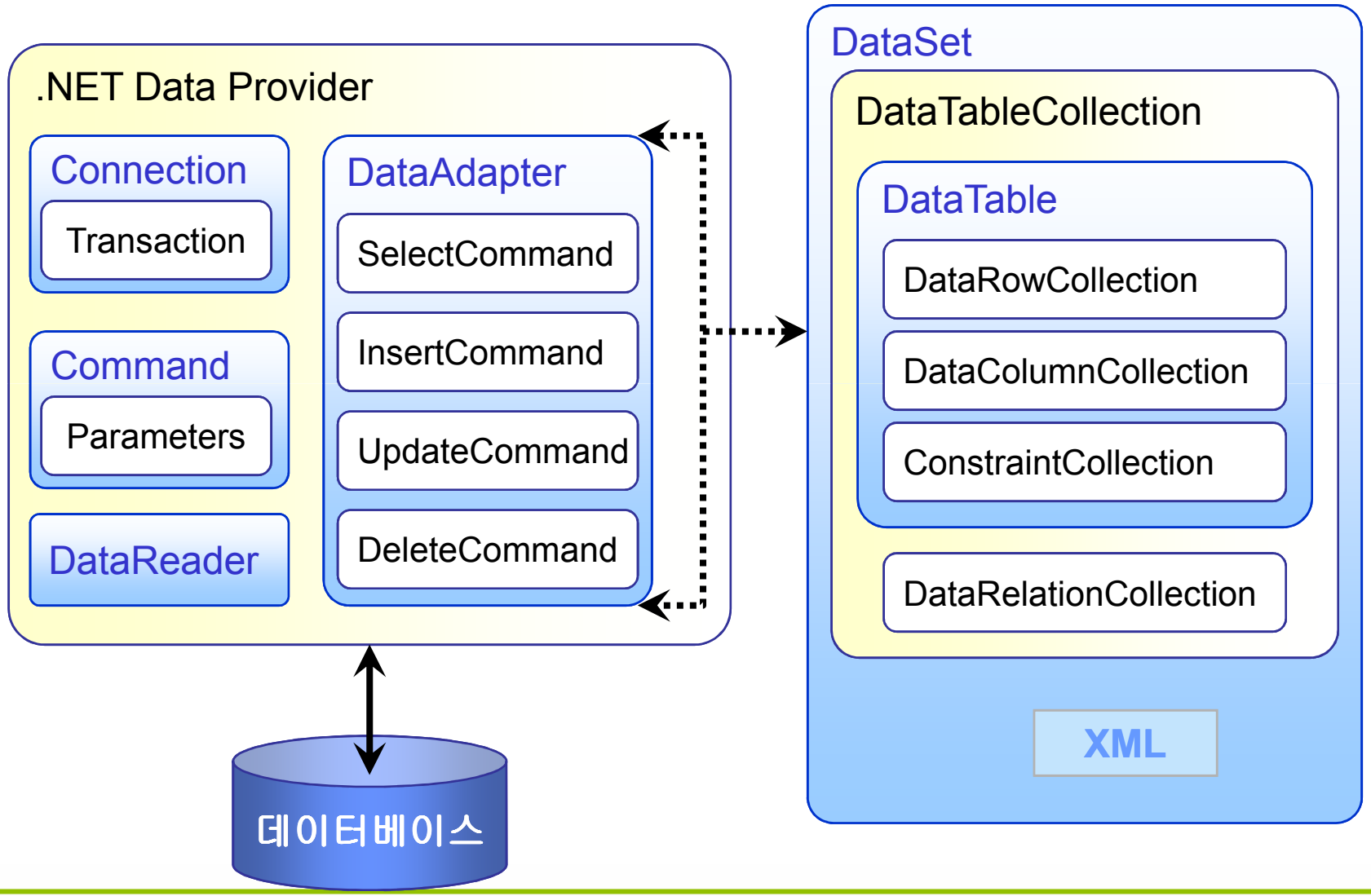
- **ObjectDataSource 사용 – Data Component 사용**
  - Data Component 추가하기
    - 솔루션 탐색기에서 ASP.NET 폴더 추가를 선택하여 App_Code를 추가
    - App_Code에 기존 항목 추가를 선택하여 c:\WebSites\Demo\DataComponent\Customers.cs를 추가
  - 새로운 ObjectDataSource컨트롤을 DropDownList1밑으로 드래깅하여 추가
  - ObjectDataSource1 의 데이터 소스 구성 하기
    - 비즈니스 개체 선택 : Customers
    - 데이터 메서드 정의:
      - SELECT문에 GetCustomersByCountry()를 선택
    - 매개변수 정의:
      - 매개변수 소스 :Control 선택
      - ControlID :DropDownList1
  - 도구 상자에서 GridView 컨트롤을 ObjectDataSource1밑에 추가
    - 데이터 소스 선택 : ObjectDataSource1
    - 자동서식 지정하기
  - 실행 결과 확인하기 : 솔루션 탐색기에서 브라우저 보기로
  - DemoObjectDataSource.aspx를 소스 보기로 열어서 추가된 태그 확인해보기

실습9.2 : ObjectDataSource:Data Component- IV

◆ADO.NET을 사용한 데이터베이스 연동-399p

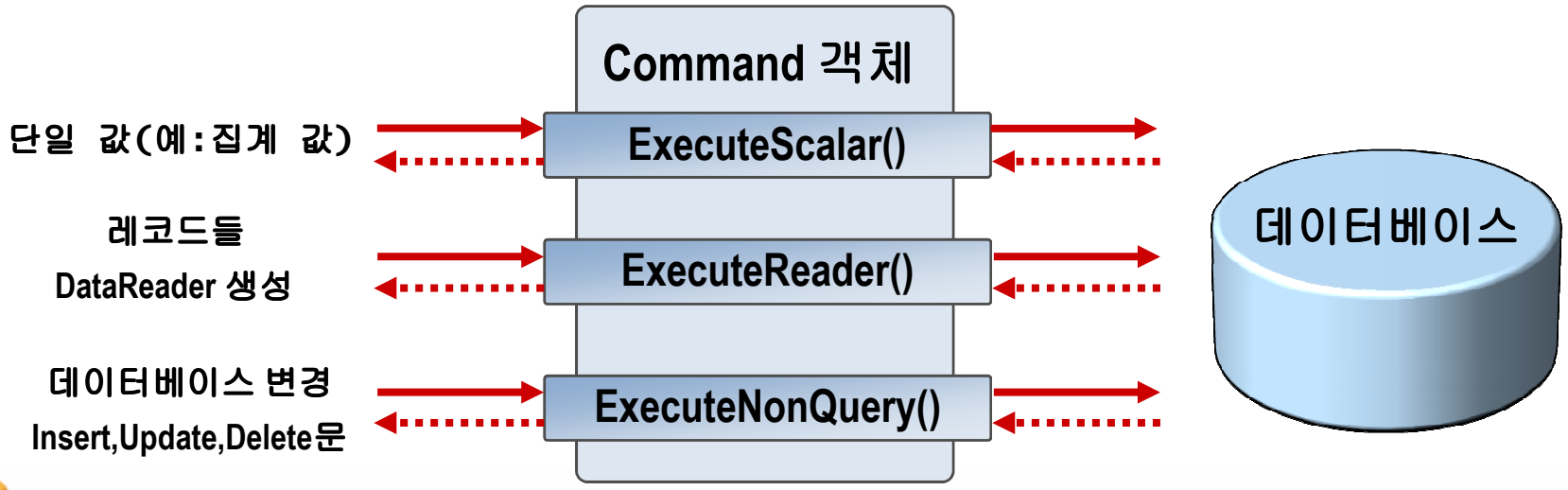
- 연결기반 데이터베이스 연동
- 비연결기반 데이터베이스 연동

ADO.NET 객체 모델


.NET Data Provider
Connection
Transaction
Command
Parameters
DataReader
DataAdapter
SelectCommand
InsertCommand
UpdateCommand
DeleteCommand
데이터베이스
DataSet
DataTableCollection
DataTable
DataRowCollection
DataColumnCollection
ConstraintCollection
DataRelationCollection
XML

◆연결기반 데이터베이스 연동

- DataReader 사용
- Select – Command.ExecuteReader()
- Insert – Command.ExecuteNonQuery()
- Update- Command.ExecuteNonQuery()
- Delete-Command.ExecuteNonQuery()



◆비연결 기반 데이터베이스 연동- 408p

- DataSet 사용