Introduzione al Framework ASP.NET 2.0 per lo sviluppo di applicazioni WEB

Mauro Minella
Microsoft
mauro.minella@microsoft.com

Agenda (Mattina)

- Introduzione al .Net Framework
- Introduzione a ASP.NET 2.0
- Membership e controlli per il Log-in
- Le pagine Master, i Temi e gli Skin
- Sorgenti dati e Controlli data-bound
- Gestione della Stato e Caching

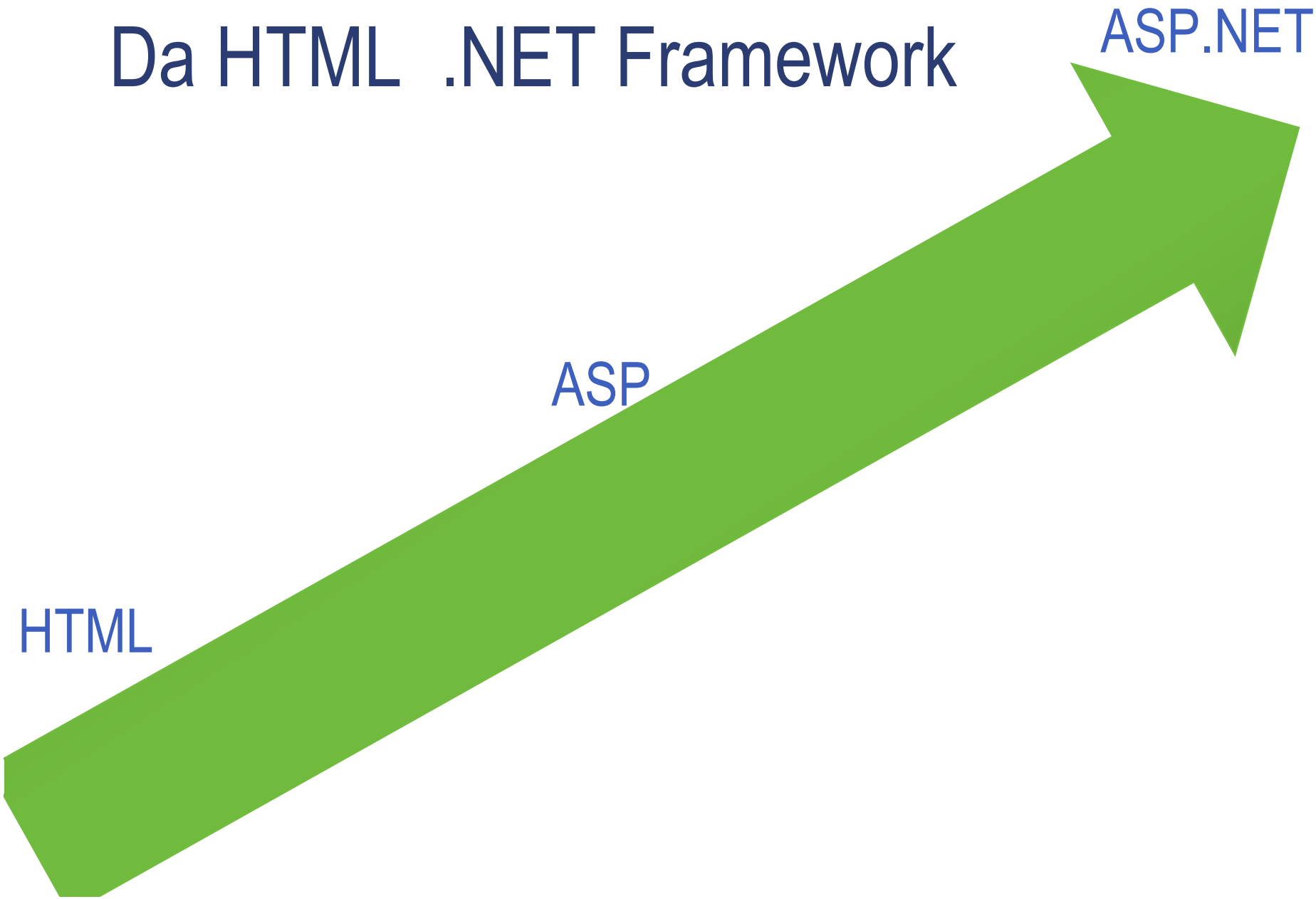
Microsoft

Agenda (Pomeriggio)

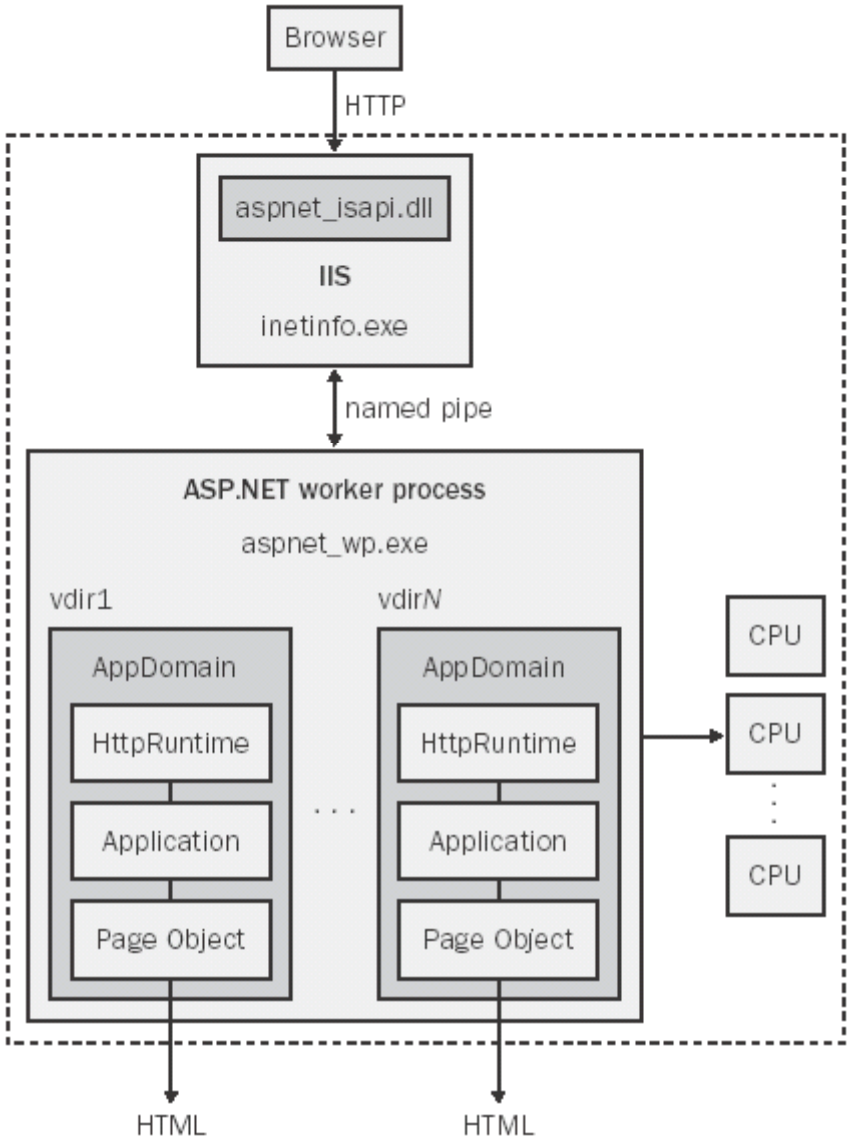
- Hands-on Lab

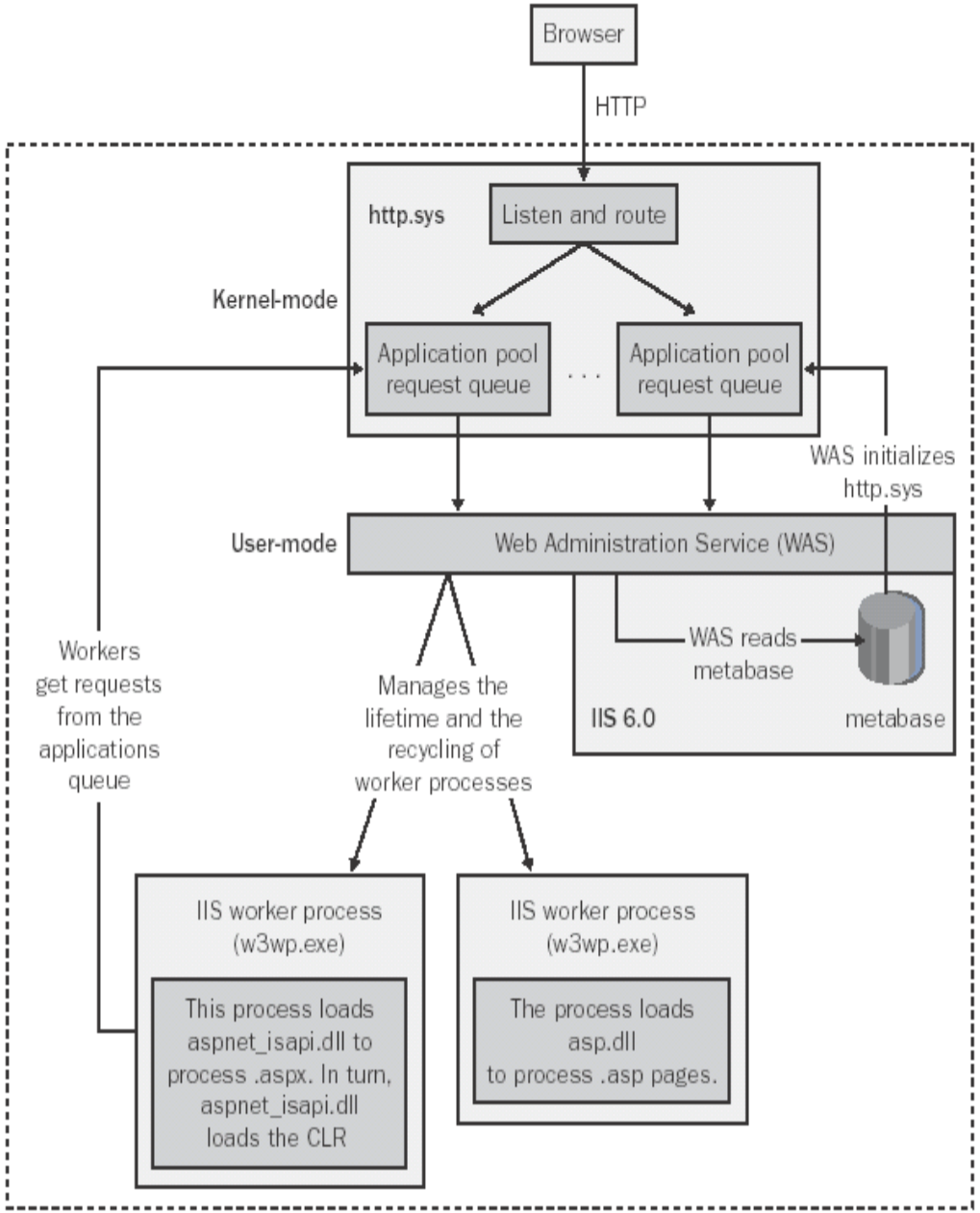
Microsoft

Da HTML .NET Framework

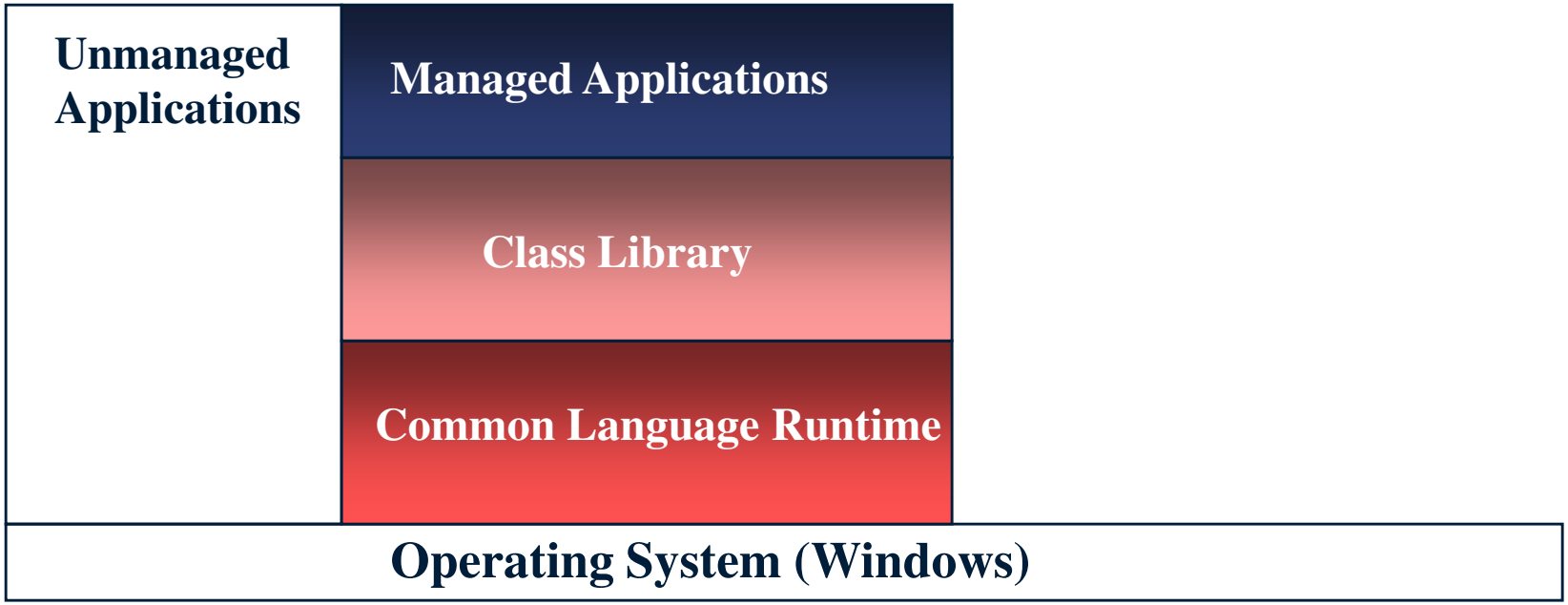


Architettura di IIS 5 vs. 6





Cos'è il .Net Framework

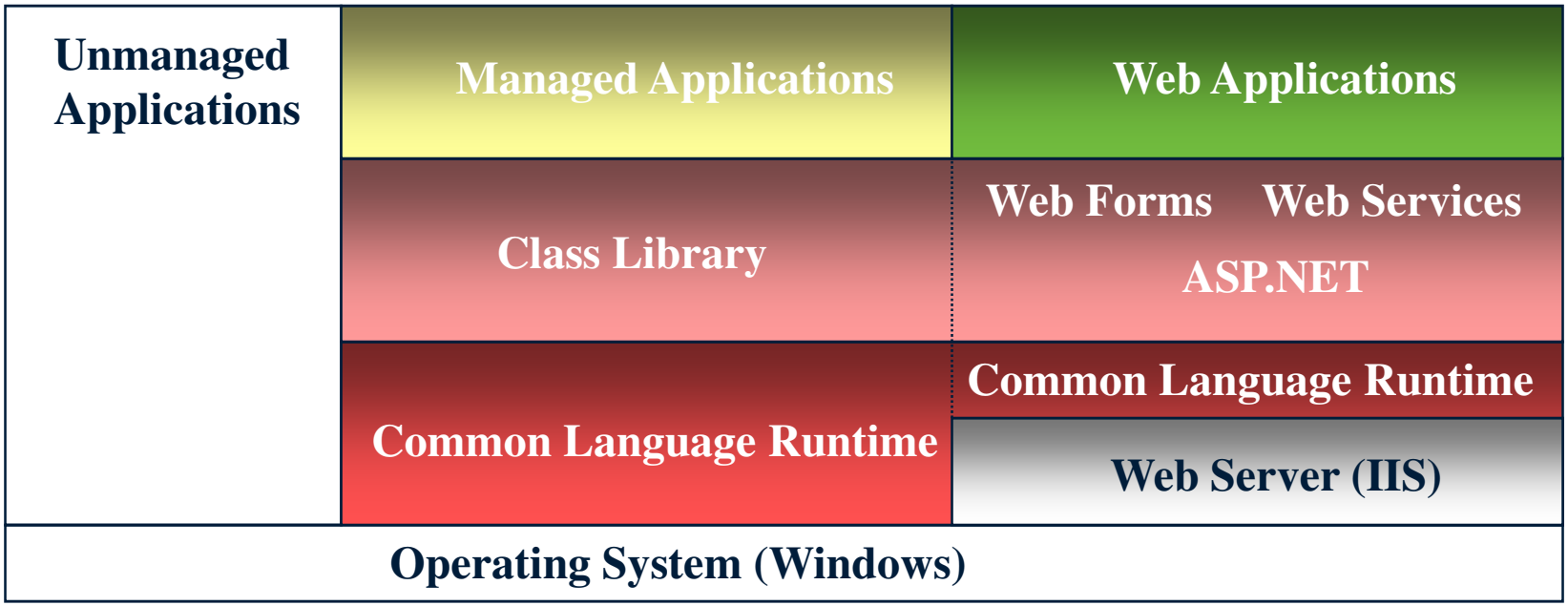


Common Language Runtime — **interoperability, security, garbage collection, versioning, ..**

Class Library — **GUI, collections, threads, networking, reflection, XML, ...**

Microsoft

Cos'è il ASP.NET

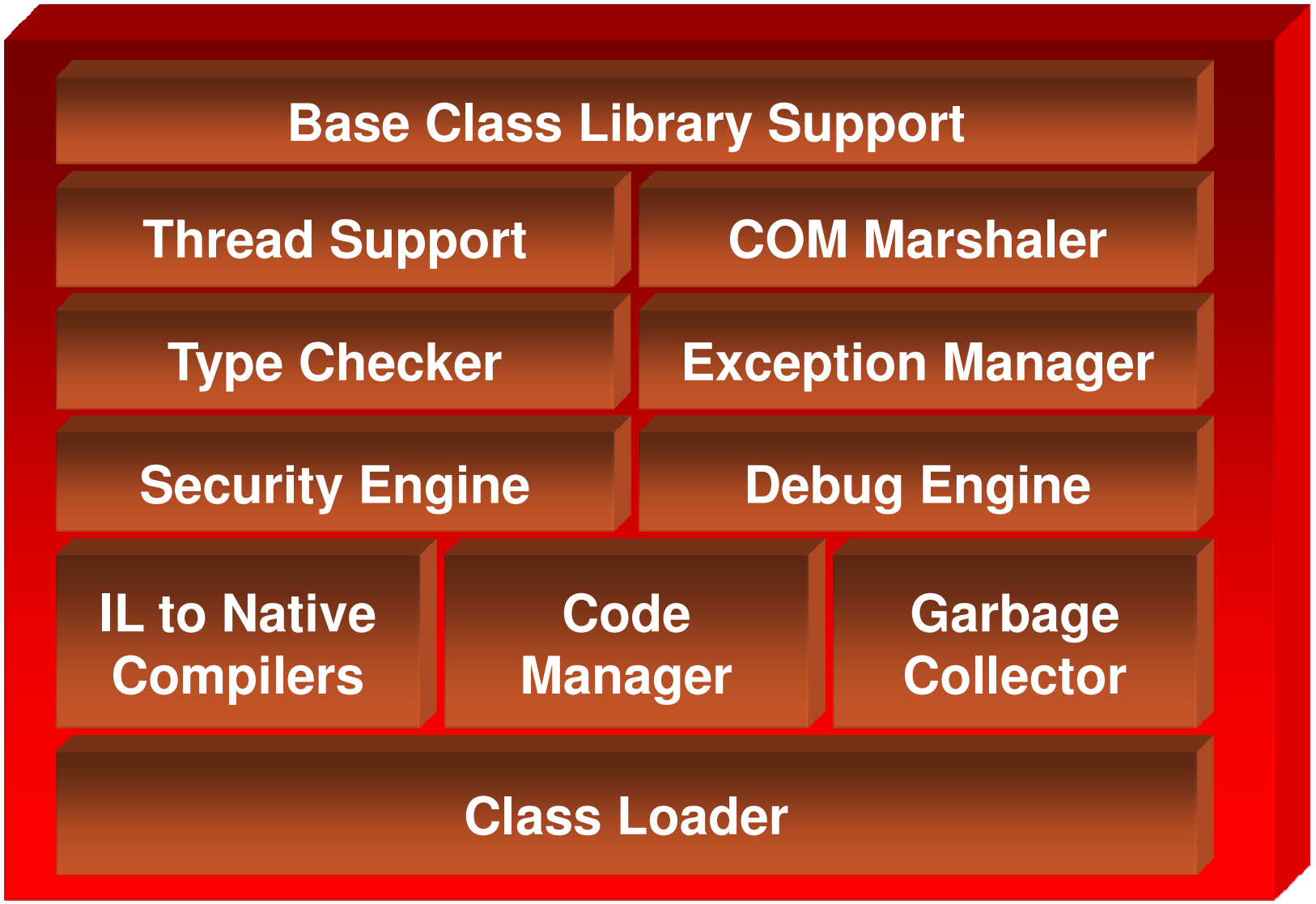


ASP.NET, Web Forms — Web GUI (object-oriented, event-based, browser-independent)

Web Services — distributed services over RPC (SOAP, HTTP)

Microsoft

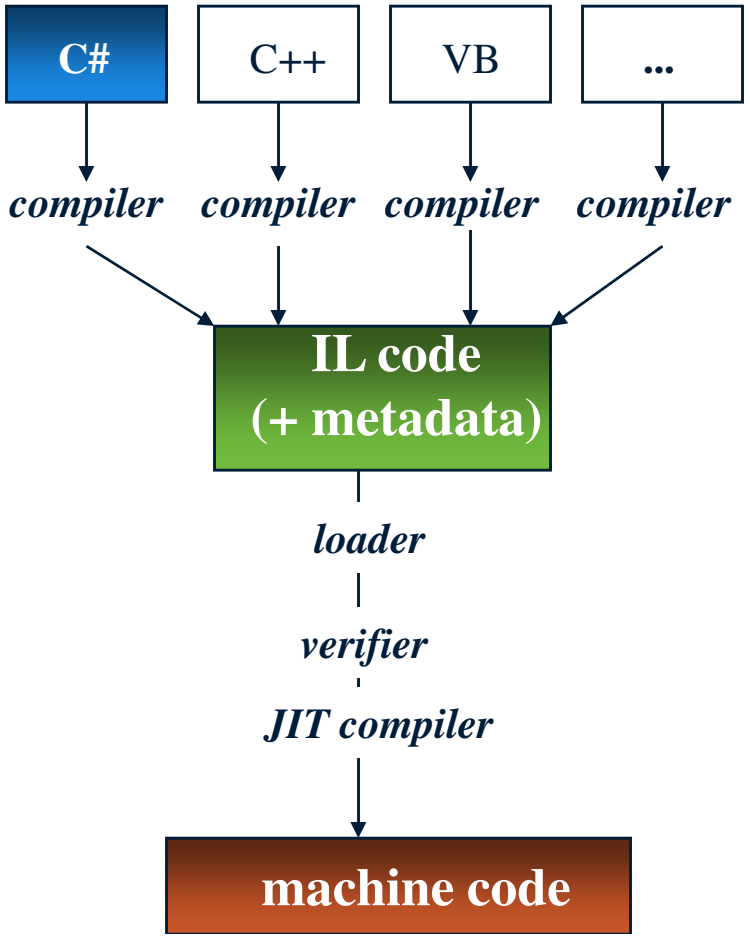
Common Language Runtime

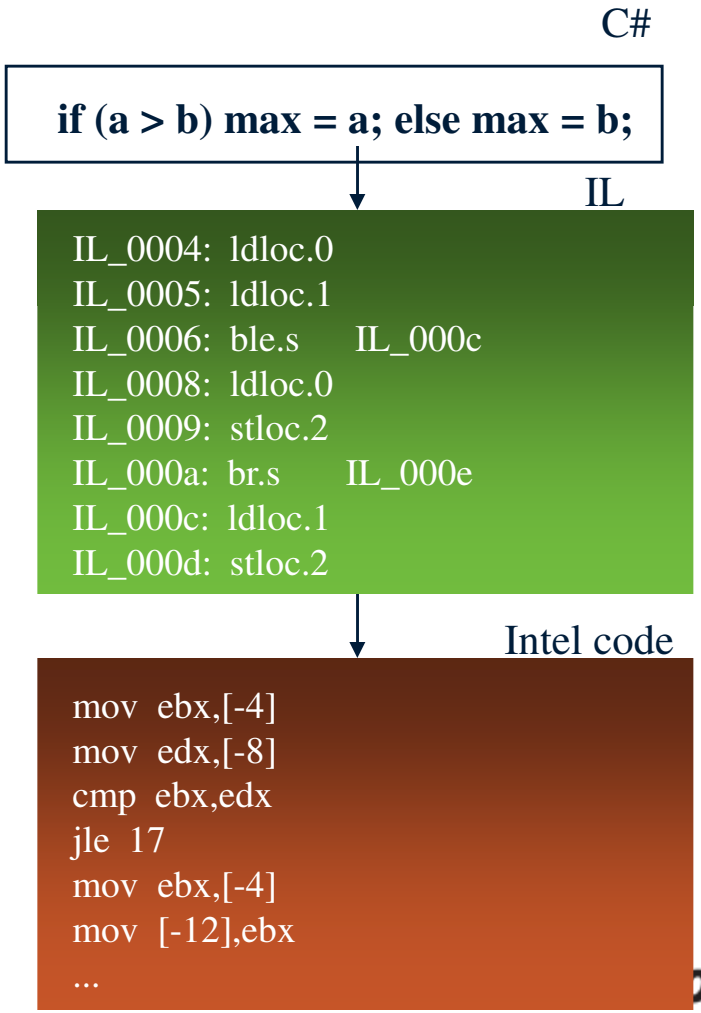


Microsoft

Architettura

- Compatibilità binaria tra linguaggi
- Codice Isolato (accede solo alla memoria permessa)
- Codice type-safe (no buffer overrun)
- Just-in-time compiler





Indipendenza dalla piattaforma e dal linguaggio

- .NET è un'implementazione di CLI
  - Common Language Infrastructure
- CLI è uno standard ECMA, definito con C#
  - ECMA-334, ECMA-335
- Esistono già altre implementazioni di CLI:
  - SSCLI (Microsoft per Windows, FreeBSD e Macintosh)
  - Mono (per Linux)
  - DotGNU
  - Intel OCL (Open CLI Library)
  - …

Microsoft

DEMO
Creiamo la nostra prima applicazione

- Strumento di sviluppo: Microsoft Notepad
- Una pagina .aspx ha tipicamente 3 sezioni:
  - **Page directives**: impostazione dell'ambiente, registrazione controlli, caricamento assembly non ancora nella GAC, registrazione namespace, indicazione del linguaggio utilizzato
  - **Code section**: gestori dei controlli server side della pagina; può essere inline o separato
  - **Page layout**: lo scheletro della pagina, inclusi controlli lato server, testo, e tab HTML

Microsoft

Creiamo la nostra prima applicazione (cont.)

- Creiamo un file .ASP vuoto
- Prepariamo lo scheletro della pagina
- Aggiungiamo i controlli
- Rendiamo i controlli server-side
  - Il runtime ASP.NET esegue il parsing della pagina che viene caricata, e crea istanze di classi del .NET framework per ogni controllo con attributo **runat "server"**
- Inseriamo:

  Funzione di risposta al click: OnServerClick="MakeItUpper"

  Direttiva di linguaggio scelto: <% @Page Language="C#"%>

  Codice C#, VB o C++ (che non viene incluso nella risposta!)

Microsoft

Introduzione a Visual Studio 2005 e ASP.NET 2.0

Esploriamo l'ambiente ...

Visual Web Developer 2005 Express Edition

http://www.microsoft.com/italy/msdn/prodotti/vs2005/editions/download/wdd.mspx

- Download gratuito
- Include versioni free di SQL Server 2005 e MSDN
- La registrazione (non obbligatoria) offre molti vantaggi, tra cui l'hosting gratuito del vostro sito

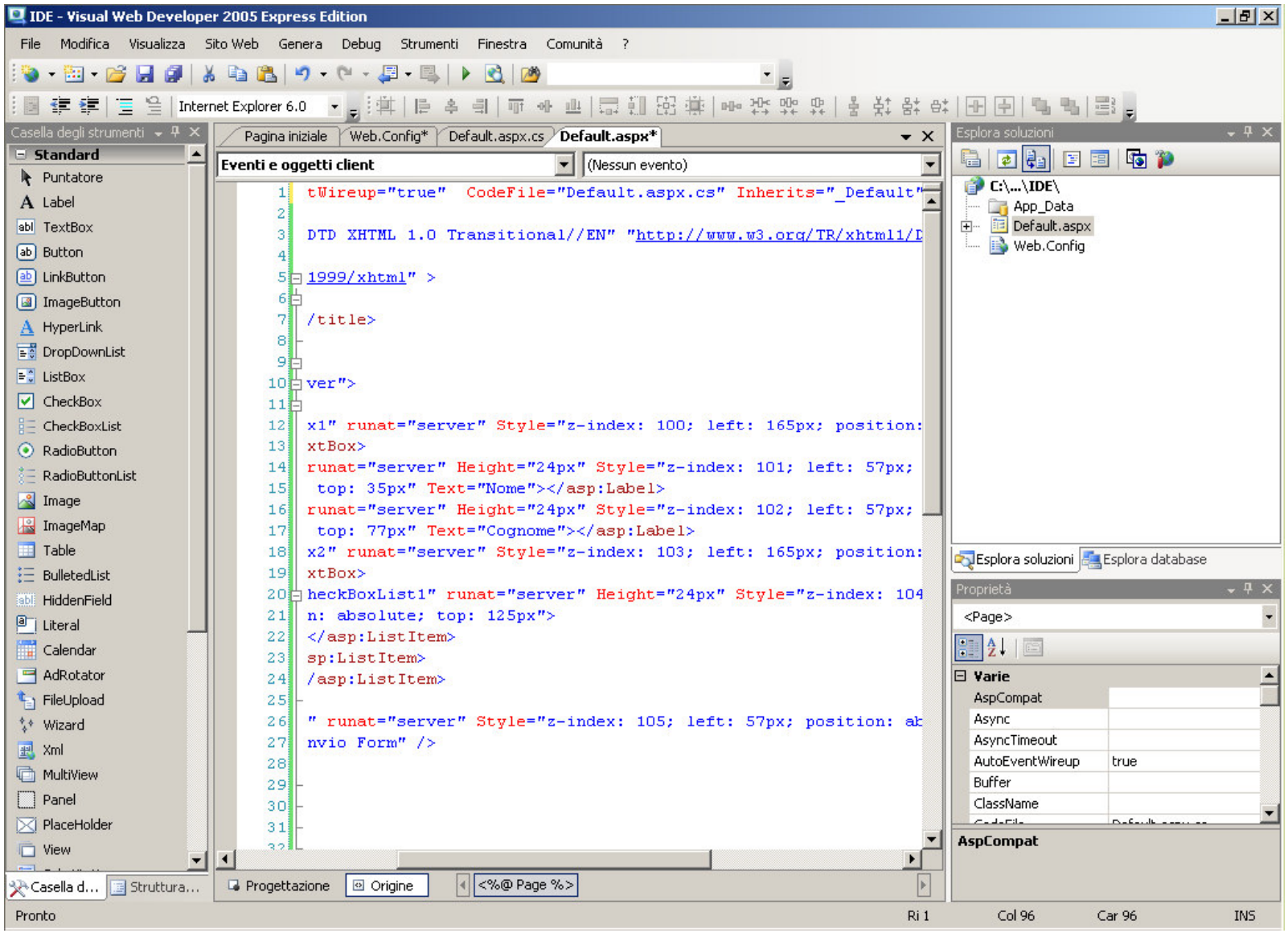
Microsoft

Introduzione a ASP.NET 2.0

- Esploriamo l'IDE
  - Posizionamento assoluto dei controlli
- Creazione di un Web Site
- Controlli e funzionalità di base
  - Controlli server e output dipendente dal dispositivo
  - Gestione del ViewState
  - Validatori
  - Gestione del Post Back
- Diagnostica e Ciclo di Vita della pagina

Microsoft

IDE

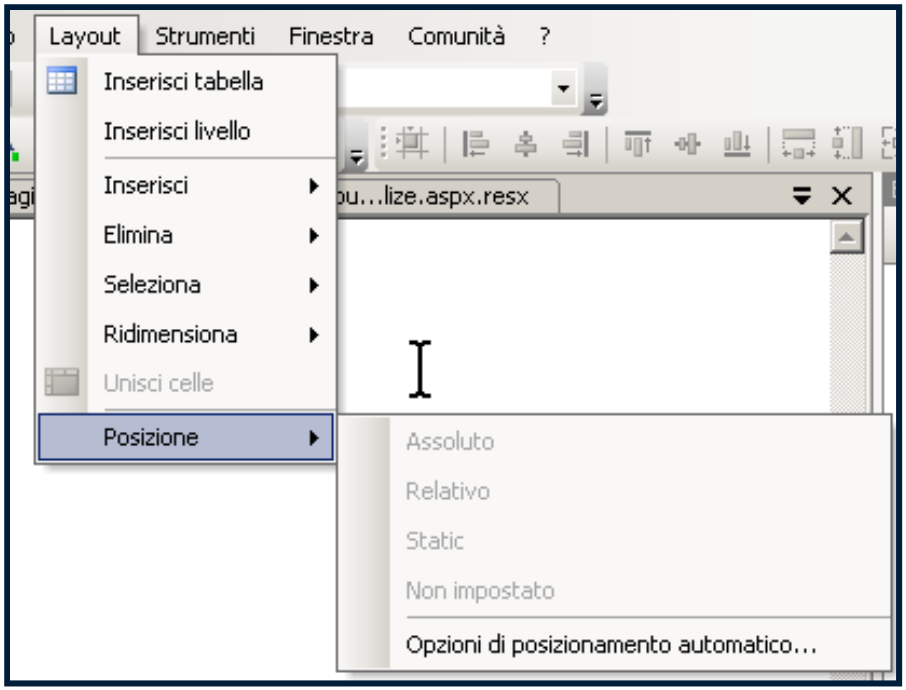


Microsoft

Controlli

- Controlli HTML:
  - è l'HTML standard
    - <a >..., <label>...
- Lato-Server
  - Generano HTML in modo **dipendente** dal browser
  - Ce ne sono tantissimi !! Vediamone alcuni

Posizionamento assoluto dei controlli



Microsoft

ViewState

- Mantiene lo stato a livello di **Pagina**
  - È un dizionario nome/valore
- È un campo Hidden della pagina
- Può essere usato anche programmaticamente
  - ViewState.Add(“NomeUtente”, “Mauro”)
- Può essere disabilitato a livello di pagina
  - <%@ Page ... EnableViewState="false"%>
  - Attenzione che i controlli che usano il view state possono non funzionare più!

PostBack

- È un evento che scatta la seconda volta che si arriva su una pagina
  - In seguito ad una POST HTTP che si verifica
    - Submit di un bottone
    - Controlli server-side possono avere la proprietà **AutoPostBack** abilitata
      - Può servire per popolare altri controlli o disabilitarli

PostBack: Uso tipico

- Posso ottimizzare il codice eseguito nella pagina
  - Accedo una sola volta alle risorse costose (database)

```
protected void Page_Load(..)
{
        if(Page.IsPostBack == false)
        {
                // E' il primo accesso alla pagina
                // Accesso al database
        }
}
```

Microsoft

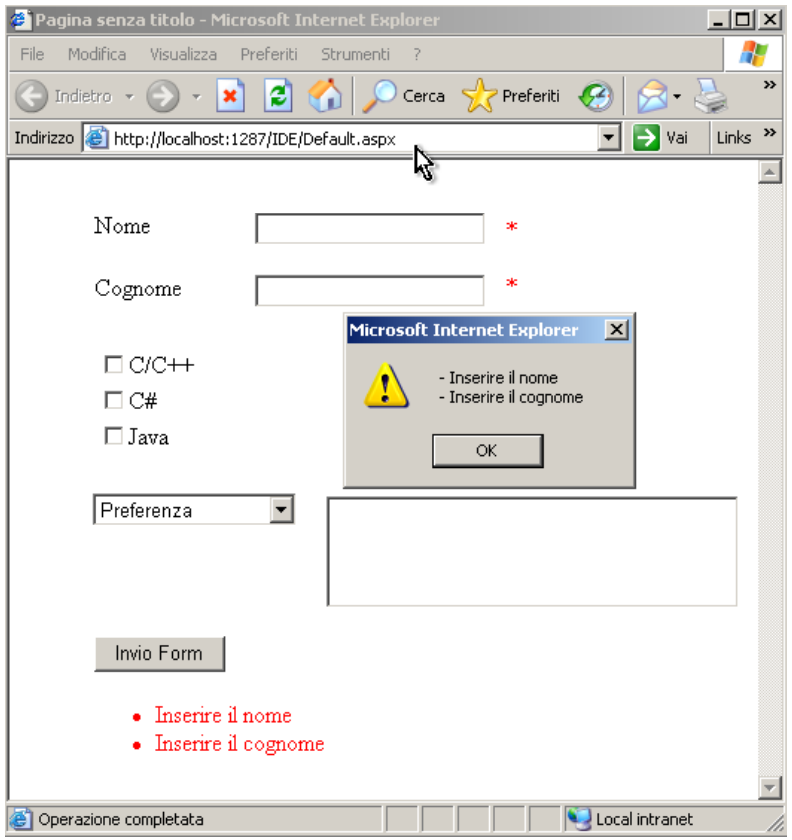
Demo

- Creazione di un sito web
- Posizionamento dei controlli
- View State
- Code-behind
- Post-back

Microsoft

Validatori

- Controlli per la validazione dei controlli **lato server**
- Rilevano se il browser supporta la validazione **lato client**
  - RequiredFieldValidator
  - CustomValidator (richiamo funzione JS)
  - ValidatorSummary (message box)
  - RangeValidator
  - RegularExpressionValidator
  - CompareValidator



Microsoft

Demo

- Validatori

Microsoft

Debugging (Server-side)

- Il debugging viene abilitato nel web.config

```
<configuration>
    <appSettings/>
    <connectionStrings/>
    <system.web>
        <!--
            Impostare compilation debug="true" per inserire i
            simboli di debug nella pagina compilata. Poiché tale operazione ha effetto
            sulle prestazioni, impostare questo valore su true
            solo durante lo sviluppo.
        -->
        <compilation debug="true"/>

        <!--
            La sezione <authentication> consente di configurare
            la modalità di autenticazione della protezione utilizzata da
            ASP.NET per identificare un utente in ingresso.
        -->
        <authentication mode="Windows"/>
        <!--
            La sezione <customErrors> consente di configurare
            l'operazione da eseguire in caso di errore non gestito
            durante l'esecuzione di una richiesta. In particolare,
            consente agli sviluppatori di configurare le pagine di errore HTML
            in modo che vengano visualizzate al posto dell'analisi dello stack dell'errore.

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

Debugging (Javascript)

- Più complicato da impostare
  - Abilitare il browser

  - Quindi o si fa partire il debugger da IE e poi si mette il breakpoint sul javascript
  - o da Visual Studio ci si "attacca" al processo IE
    - Questa funzionalità non è supportata nella versione "Express"
  - "Trucco": istruzione "debugger;" nel codice Javascrip
    - Attenzione a non lasciarla in produzione!

Microsoft

Tracing

- Si può abilitare a livello di web.config e di Pagina
  - pageOutput abilita l'output sulla pagina o richiamando **trace.axd**

```
<trace enabled="true" pageOutput="false" />
```

```
<%@ Page Language="C#" ... Trace="true"%>
```

- Per scrivere

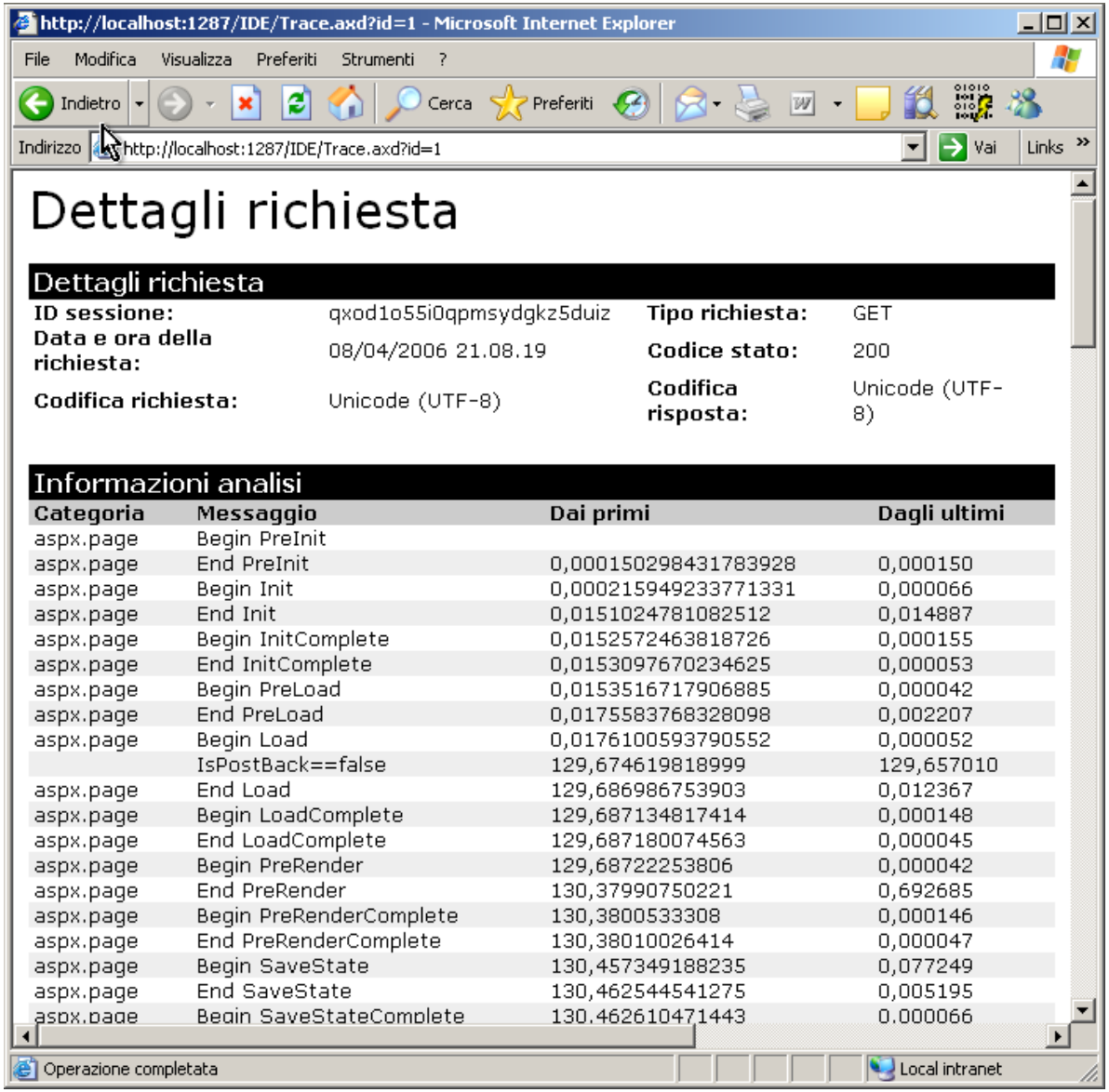
```
Trace.Write (categoria, messaggio, eccezione);

Trace.Warn (categoria, messaggio, eccezione);

Trace.Write (messaggio);
...
```

```
13      protected void Page_Load(object sender, EventArgs e)
14      {
15          if (Page.IsPostBack == false)
16          {
17              // E' la prima volta che accedo alla pagina
18              // Accesso a risorse costose (database)
19              Trace.Write("IsPostBack==false");
20
21              ViewState.Add("Mypro", "pietro");
22          }
23          else
24          {
25              string msg = "vuoto";
26              try
27              {
28                  msg = ViewState["Mypro"].ToString();
29              }
30              catch (Exception)
```

Esempio di Trace



Dettagli richiesta

Dettagli richiesta

ID sessione:	qxod1o55i0qpmsydgkz5duiz	**Tipo richiesta:**	GET
Data e ora della richiesta:	08/04/2006 21.08.19	**Codice stato:**	200
Codifica richiesta:	Unicode (UTF-8)	**Codifica risposta:**	Unicode (UTF-8)

Informazioni analisi

Categoria	Messaggio	Dai primi	Dagli ultimi
aspx.page	Begin PreInit		
aspx.page	End PreInit	0,000150298431783928	0,000150
aspx.page	Begin Init	0,000215949233771331	0,000066
aspx.page	End Init	0,0151024781082512	0,014887
aspx.page	Begin InitComplete	0,0152572463818726	0,000155
aspx.page	End InitComplete	0,0153097670234625	0,000053
aspx.page	Begin PreLoad	0,0153516717906885	0,000042
aspx.page	End PreLoad	0,0175583768328098	0,002207
aspx.page	Begin Load	0,0176100593790552	0,000052
	IsPostBack==false	129,674619818999	129,657010
aspx.page	End Load	129,686986753903	0,012367
aspx.page	Begin LoadComplete	129,687134817414	0,000148
aspx.page	End LoadComplete	129,687180074563	0,000045
aspx.page	Begin PreRender	129,68722253806	0,000042
aspx.page	End PreRender	130,37990750221	0,692685
aspx.page	Begin PreRenderComplete	130,3800533308	0,000146
aspx.page	End PreRenderComplete	130,38010026414	0,000047
aspx.page	Begin SaveState	130,457349188235	0,077249
aspx.page	End SaveState	130,462544541275	0,005195
aspx.page	Begin SaveStateComplete	130,462610471443	0,000066

Microsoft

Ciclo di Vita di una Pagina (Cenni)

- **PreInit**: serve per
  - Usare la proprietà IsPostBack
  - Creare controlli dinamici
  - Applicare temi e pagine master dinamicamente
  - Leggere e scrivere profili utente
- **Init**: leggere e inizializzare le proprietà dei controlli
- **Load**: leggere e aggiornare le proprietà dei controlli
- **PreRender**: apportare modifiche ai contenuti della pagina
- **UnLoad**: operazioni di chiusura finale

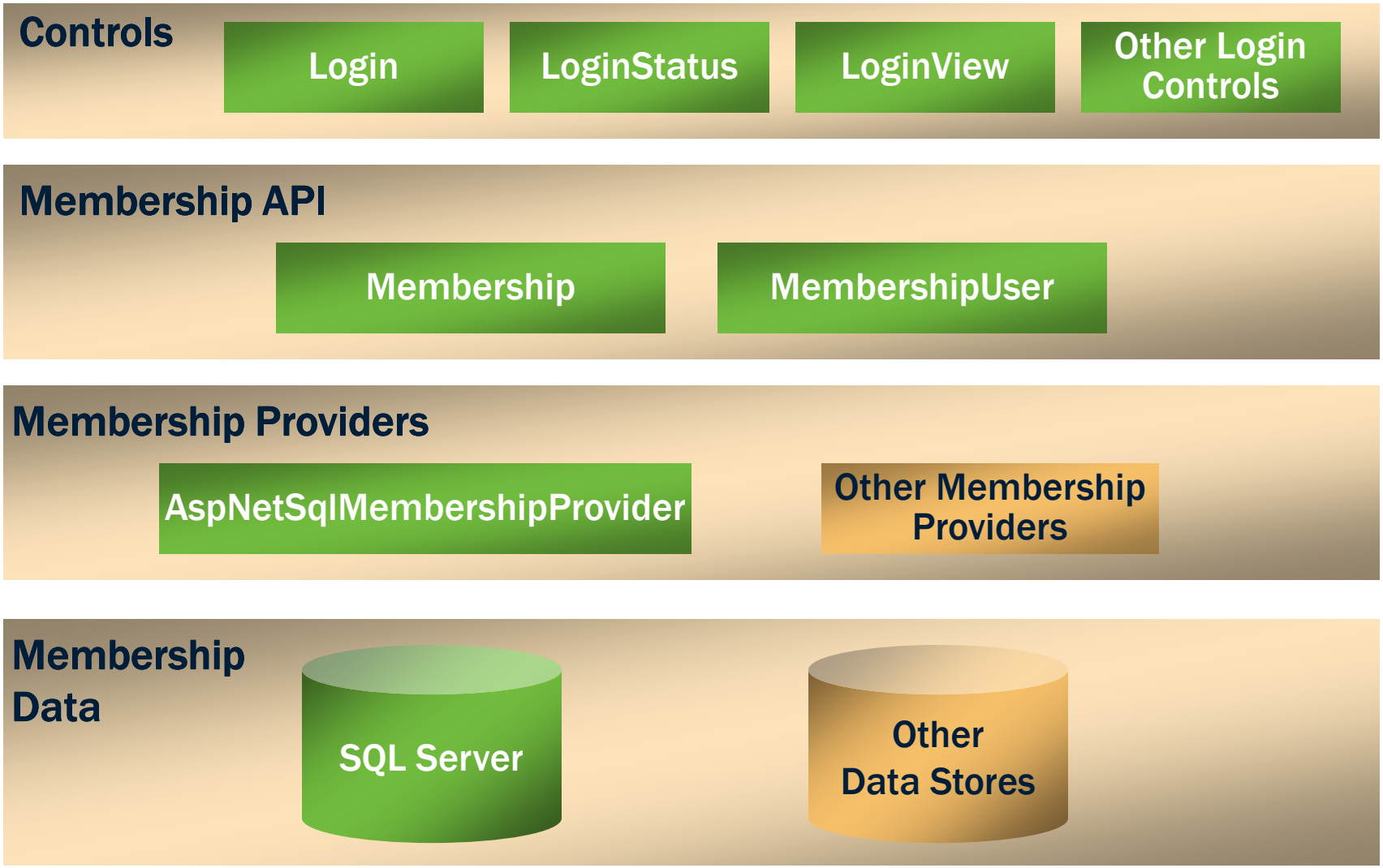
Membership, Ruoli e controlli per il log-in

Autenticazione e autorizzazione

Autenticazione: scenari

- Riconoscere **chi** si sta loggando al nostro sito web
- Due scenari tipici per l'autenticazione:
  - **Intranet**: si appoggia su sistemi di autenticazioni della intranet aziendale
    - Internet Information Server (IIS) usa la Integrated Authentication (ad esempio)
    - Tipicamente gli utenti sono su Active Directory.
  - **Internet**: può appoggiarsi su un database per la gestioni degli utenti

Un modello estendibile



Microsoft

Controlli per il log-in

- Interagiscono con un provider per la gestione delle funzionalità di membership
- **Login**: permette di effettuare la login usando nickname e password
- **LoginView**: permette di inserire contenuto diverso per utenti autenticati e non
- **PasswordRecovery**: posso recuperare la password (mail) rispondendo ad una domanda
- **LoginStatus**: dice se l'utente è loggato o no
- **LoginName**: nome dell'utente in logon
- **ChangePassword**: per cambiare password
- **CreateUserWizard**: molto codice risparmiato!

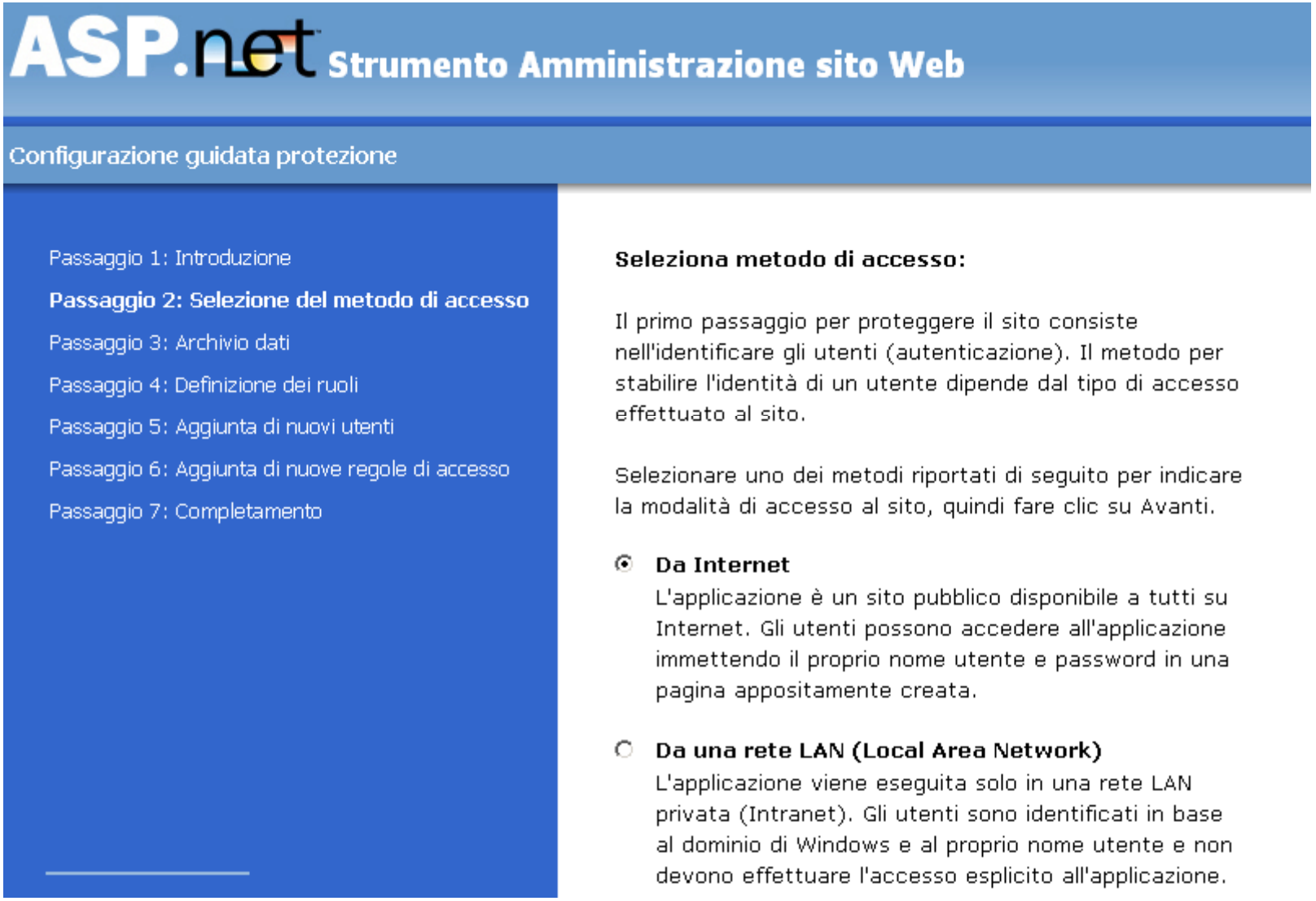
Microsoft

Usare il sito di amministrazione (WSA)

- Imposta parametri dell'applicazione (web.config)
  - tipo di autenticazione
  - SMTP server, tracing, debugging, errori, ...
- Gestisce gli utenti
  - Creazione, cancellazione, modifica ...
- Gestisce ruoli e regole di accesso
  - Per distinguere l'autorizzazione
- Configura i provider di accesso al database
- Disponibile anche tramite Wizard in 7 passi

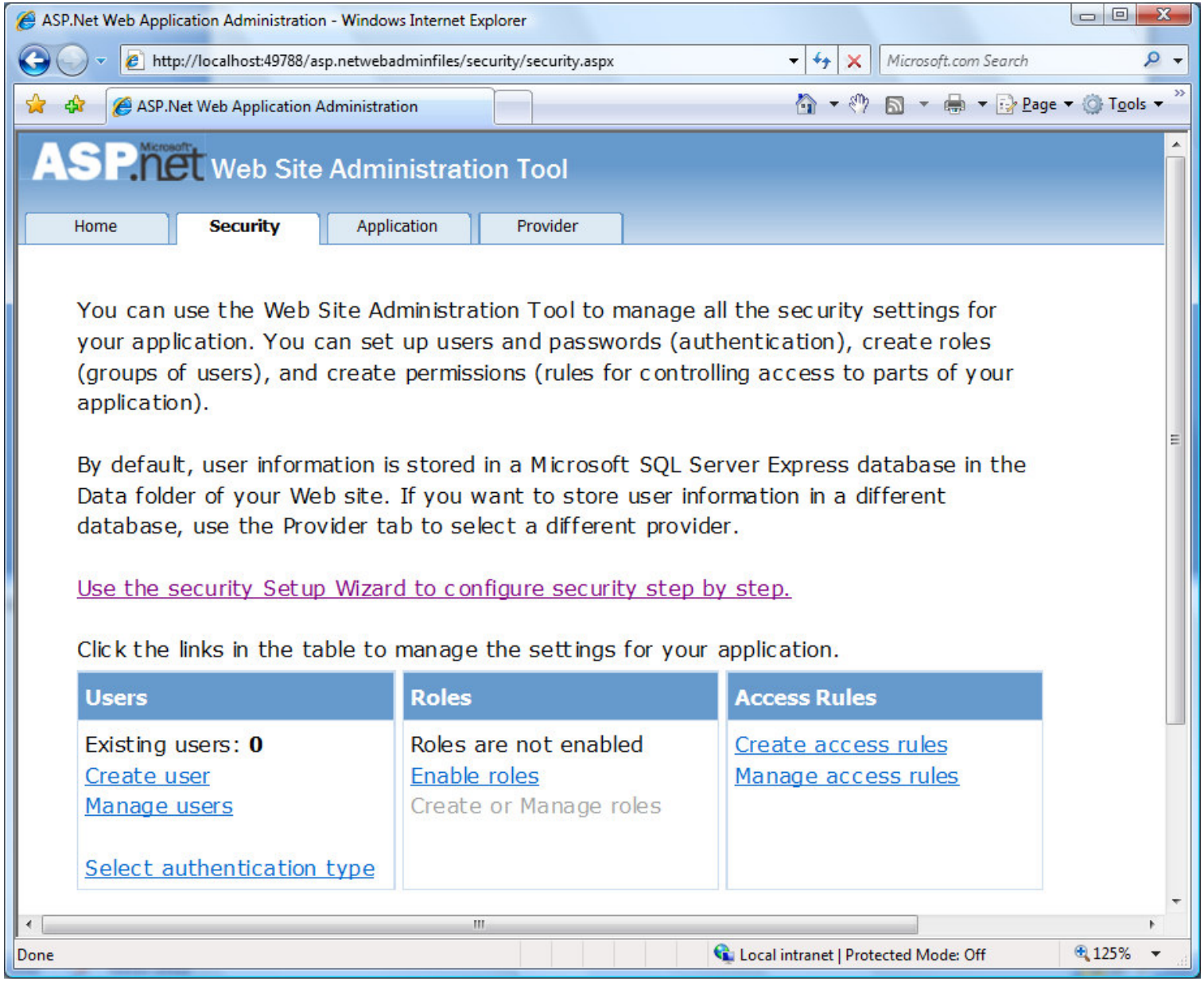
Microsoft

WSA: wizard in 7 passi



Microsoft

WSA: sezione Security



Microsoft

Esempio di uso API Membership

```
MembershipCreateStatus ms;
MembershipUser user=Membership.CreateUser(
    “mauro",
    “Password1!",
    “mauro.minella@microsoft.com",
    “Colore preferito?",
    “Blu",
    true,out ms);

if (user==null)
  Label1.Text = "Non è possibile creare un utente";
else
  Label1.Text = "Utente creato";
```

```
if (Membership.ValidateUser(username.Text, password.Text))
  FormsAuthentication.RedirectFromLoginPage(username.Text, false);
else
  Label1.Text = "Username e password non corretti";
```

Microsoft

Demo

- Creazione di un sito:
  - Sfruttare i meccanismi di sicurezza integrati nel tool di sviluppo
    - Controlli Login, LoginName e LoginStatus
  - Usare il sito di amministrazione per creare utenti e gruppi
  - Usare IIS Manager per configurare il server WEB

Localizzazione, Pagine Master, Temi e Skin

Come creare una grafica del sito omogenea e consistente

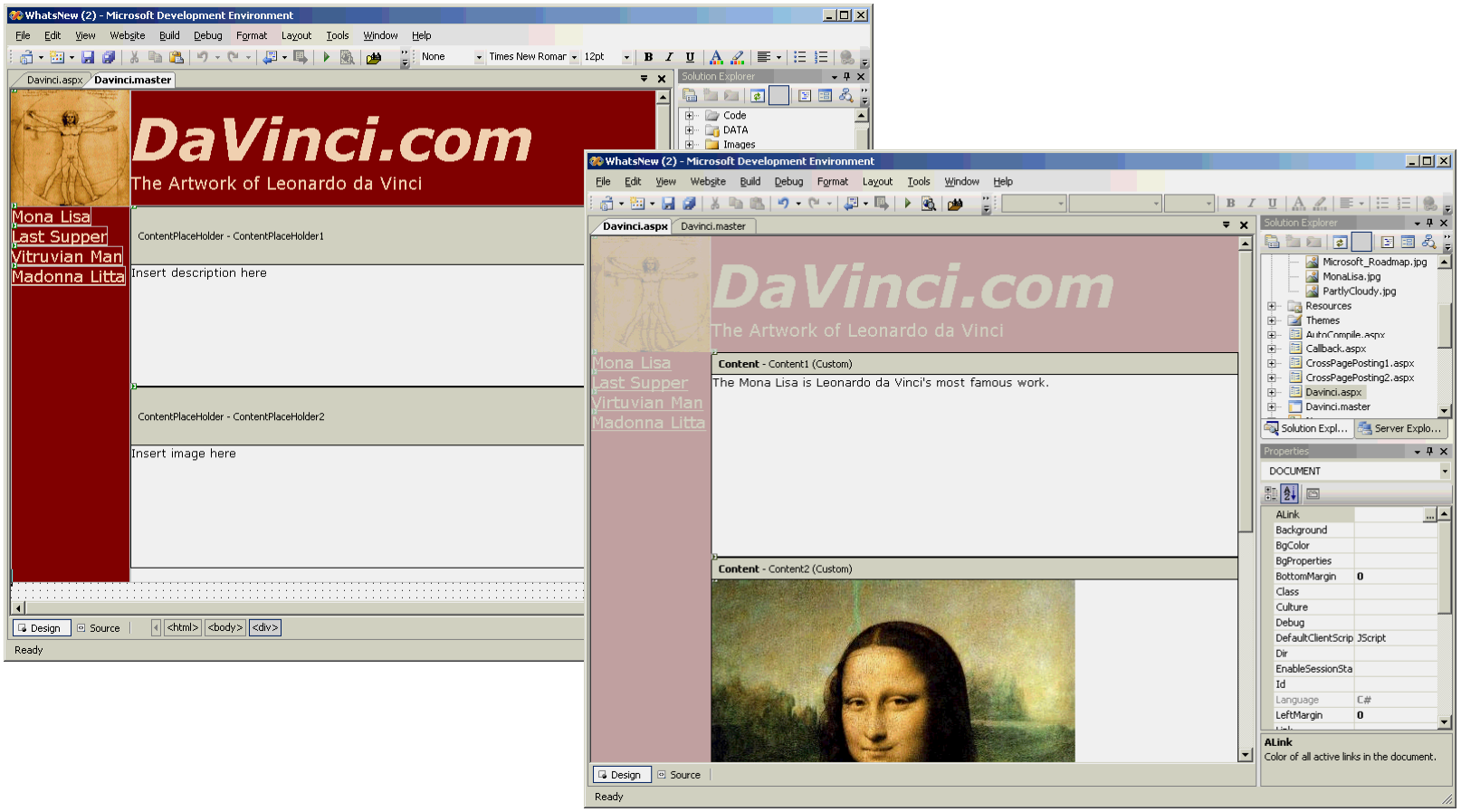
Localizzazione

- Usa file di risorse selezionati a run-time da ASP.NET
- Due cartelle:
  - *App_LocalResources*: i file contengono risorse per una singola pagina
  - *App_GlobalResources*: i file possono essere letti da qualsiasi pagina del sito Web
- Indicazione della localizzazione nelle direttive di pagina
  - UICulture="auto"
- Naming convention per la pagina:
  - linguaggio neutrale: pagina.aspx.**resx**
  - linguaggio specifico (es. italiano): pagina.aspx.**it.resx**
- Naming convention per le risorse:
  - meta:resourcekey="ResourceXYZ" nelle proprietà controllo (tag HTML)
  - Name=ResourceXYZ.<Proprietà> e Value=Valore nei file <page>.resx e <page>.<language>.resx
- Impostare nel browser del client lingua e priorità
- Tipi di risorse: stringhe, immagini, audio, file, icone, altro

DEMO

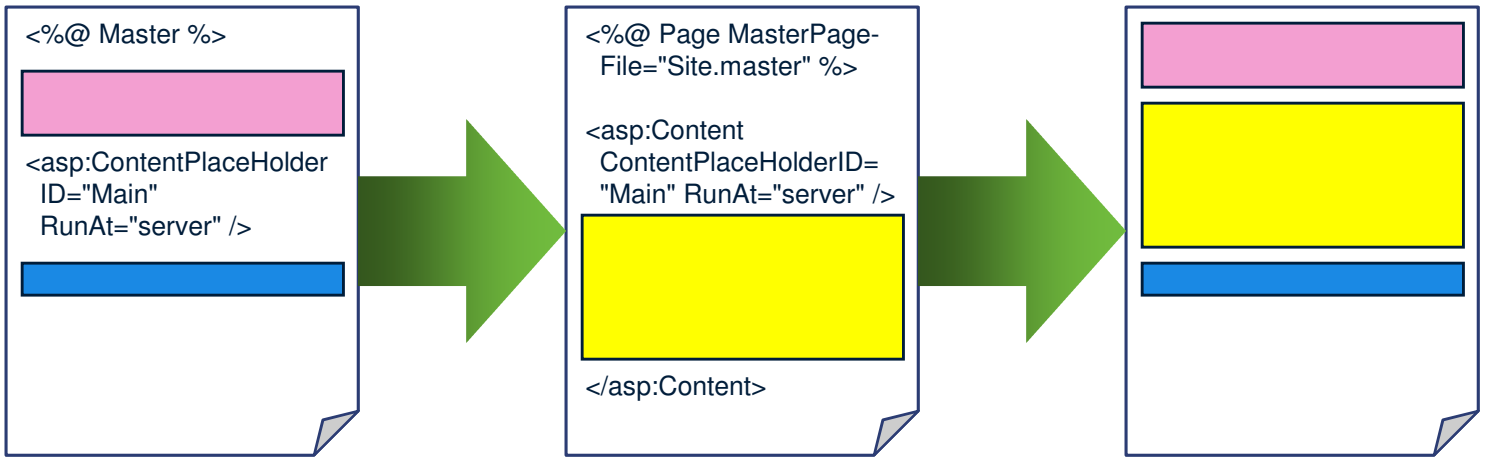
Microsoft

Pagine Master



Pagine master

- Le pagine master definiscono la struttura e dei place holder (<asp:ContentPlaceHolder>)
- Le pagine “figlie” referenziano la master e creano il contenuto (<asp:Content>)
- Le pagine master sono trasparenti all’utente, il quale invoca solo le pagine content



Microsoft

La proprietà Page.Master

- Ottiene un riferimento alla pagina master dalla pagina figlia
- Usata per avere accesso programmatico al contenuto della pagina master
  - Usare FindControl per weak typing
  - Usere public property nella master page per strong typing

Microsoft

Accedere ad un Controllo della Master Page
Weak Typing

Nella master page...

```
<asp:Label ID="Title" RunAt="server" />
```

Nella pagina figlia...

```
((Label) Master.FindControl ("Title")).Text = "Orders";
```

Microsoft

Accedere ad un Controllo della Master Page
Strong Typing

Nella master page...

```
<asp:Label ID="Title" RunAt="server" />
  .
  .
  .
<script language="C#" runat="server">
public string TitleText
{
    get { return Title.Text; }
    set { Title.Text = value; }
}
</script>
```

Nella pagina figlia…

```
Master.TitleText = "Orders";
```

Microsoft

Demo Master Pages

- Creazione di un sito ex-novo
- Aggiunta una pagina master
- Aggiunta di una pagina che usa il template della pagina master
- Controllo della pagina master dalla pagina content con weak typing

Microsoft

Temi (CSS e Skin)

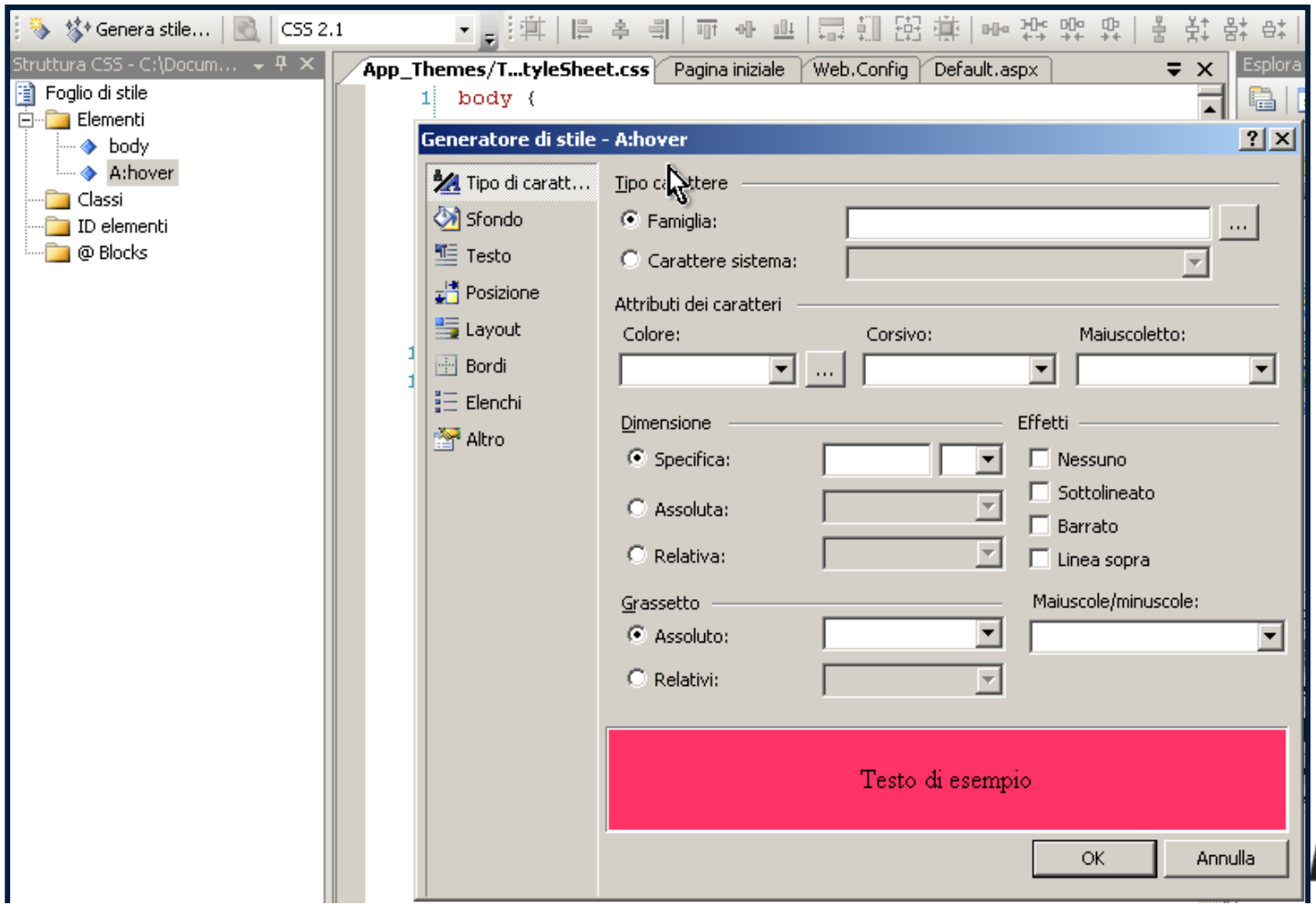
- Definiscono la grafica della pagina
- Sono un superset dei CSS (Cascading Style Sheets)
- Sono disponibili solo in ASP.NET 2.0
- Assegnano un insieme di stili e attributi visuali agli elementi personalizzabili del sito
- Sono strettamente legati ai temi di XP: impostare un tema è rapido e facile come impostare una proprietà
- Si applicano controlli individuali, pagine o siti

Temi Locali e Globali

- Temi Locali: sono nella directory App_Themes
- Temi Globali: sono visti da tutte le applicazioni web della macchina e si trovano sotto la cartella App_Themes
- Un tema può includere:
  - File CSS: definizioni di stili da applicare agli elementi di un file HTML (richiedono HTML 3.2>)
  - Skin: set di proprietà e template che si possono applicare a uno o più controlli server side
  - Immagini
  - Template

Creazione di CSS

- Si usa un tool integrato nell' ambiente, che si attiva dal menu "Stili" dopo avere aggiunto un file CSS

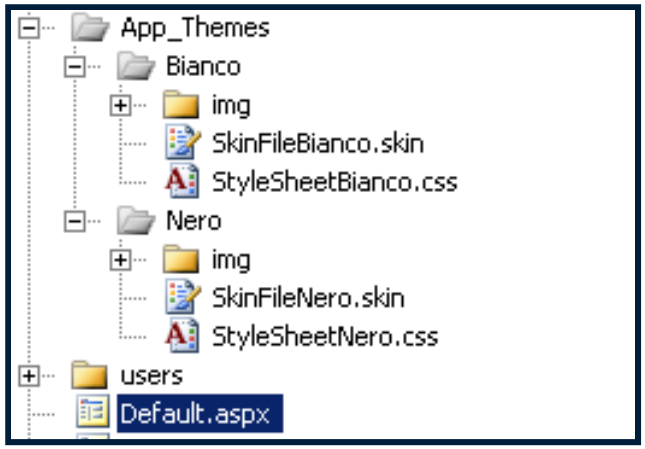


Microsoft

File di Skin

- Definisce la grafica dei controlli server
- Creo una cartella per ogni skin sotto App_Themes
- Imposto lo skin nella direttiva di pagina *StylesheetTheme*
- <asp:label runat="server" backColor="Yellow" SkinId="lblTitolo" />
- Definisco uno SkinID altrimenti lo skin viene applicato a tutti i controlli di quel tipo, e lo seleziono nelle proprietà

- DEMO



Microsoft

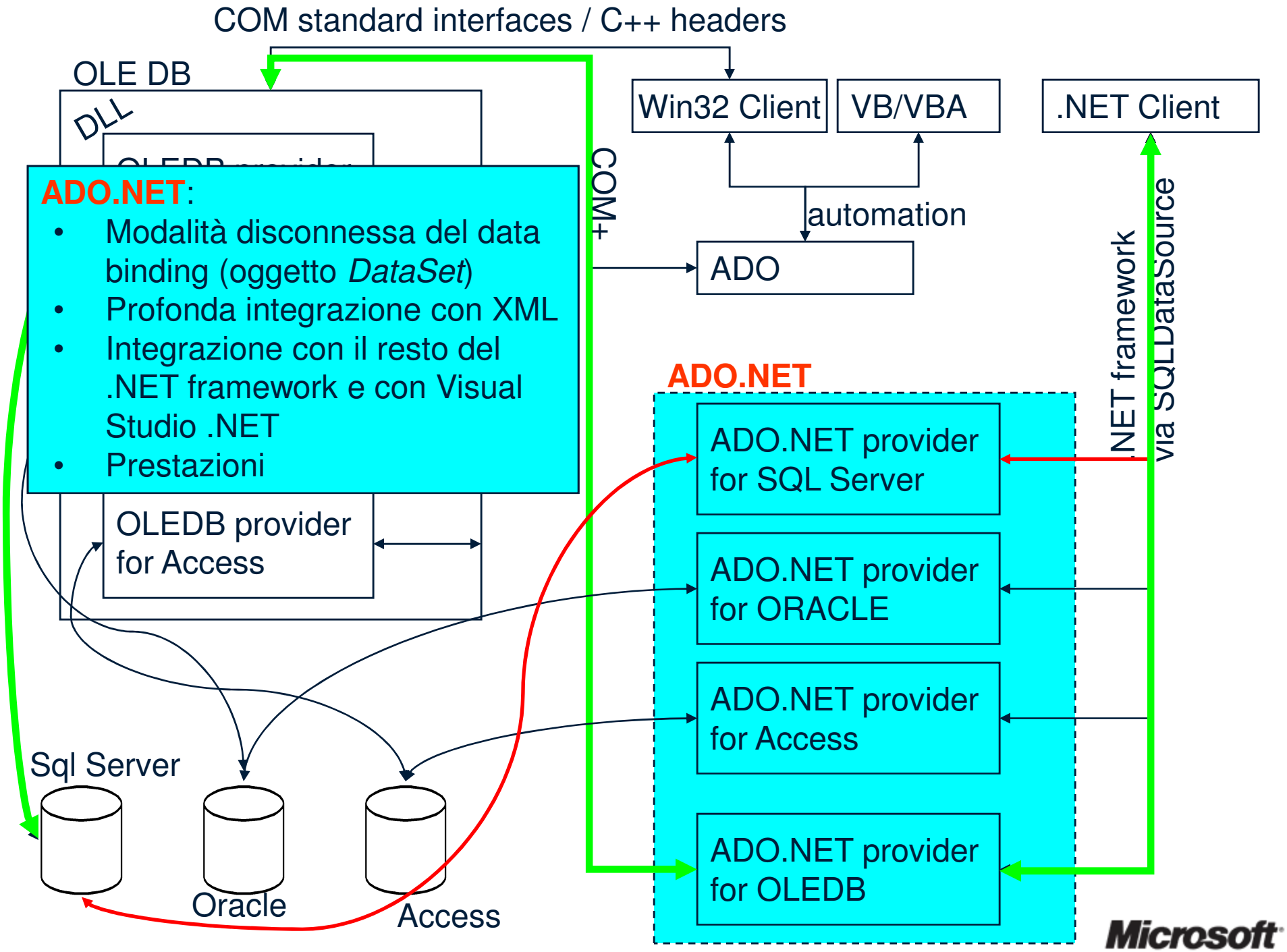
Sorgenti dati e controlli data-bound

- Evoluzione delle tecnologie di Data Access
- ADO.NET: lo stato dell'arte per l'accesso ai dati con il .NET framework
- Dataset: accesso a dati disconnessi
- DEMO: ADO.NET e SQL Server 2005

Evoluzione di accesso ai dati

- 1993 - ODBC (Open Database Connectivity): API uniforme per chiamate SQL a database server differenti
- 1997 - OLE DB: COM-based API per sorgenti dati esprimibili in forma tabellare
  - il consumer e il provider comunicano attraverso COM
  - Principale svantaggio: primariamente disegnato per C++
- 1999 –ADO: automazione verso OLE DB
  - vantaggio: usufruibile da via automation (VB5/6, VBA, ...)
  - svantaggio: ridondanza ed efficienza
- 2001: ADO.NET:
  - Modalità disconnessa del data binding (oggetto *DataSet*)
  - Profonda integrazione con XML
  - Integrazione con il resto del .NET framework e con Visual Studio .NET
  - Prestazioni

Microsoft


COM standard interfaces / C++ headers
OLE DB
DLL
ADO.NET:
• Modalità disconnessa del data binding (oggetto *DataSet*)
• Profonda integrazione con XML
• Integrazione con il resto del .NET framework e con Visual Studio .NET
• Prestazioni
OLEDB provider for Access
COM+
Win32 Client
VB/VBA
.NET Client
automation
ADO
.NET framework via SQLDataSource
ADO.NET
ADO.NET provider for SQL Server
ADO.NET provider for ORACLE
ADO.NET provider for Access
ADO.NET provider for OLEDB
Sql Server
Oracle
Access
Microsoft

Provider OLE DB vs. Managed

- **Implementazione interna**: i provider .NET managed offrono un subset di interfacce rispetto a OLE DB
- **Integrazione**: i provider .NET managed usano tipi di dati del Framework .NET (no COM interop layer)
- **Interazione diretta**: i provider .NET managed parlano direttamente con i client, mentre i provider OLE DB sono wrappati da OLE DB
- **Data source** per ADO.NET:
  - SQL Server
  - OLE DB
  - ODBC
  - Oracle

Microsoft

Controlli DataSource

- Approccio dichiarativo per ottenere i dati

Nome	Descrizione
SQLDataSource	Connects data-binding controls to SQL databases through ADO.NET data providers
AccessDataSource	Connects data-binding controls to Access databases
XmlDataSource	Connects data-binding controls to XML data
ObjectDataSource	Connects data-binding controls to data components
SiteMapDataSource	Connects site navigation controls to site map data

Come accedere ai dati in modalità disconnessa

- DataSet
  - Insieme **disconnesso** di viste associate ad un nome
- DataTable
  - Rapppresenta una tabella di dati in memoria
- Data Adapter
  - Ponte fra il data source e l'oggetto DataSet
  - Serve per le operazioni di creazione del DataSet e aggiornamento del data source

Microsoft

Esempio: usare DataSet disconnesso

- Creiamo il DataAdapter per aprire la connessione al database

```
' Impostazione dei parametri di connessione al Database.
Dim strConn As String
strConn = "Provider=Microsoft.Jet.OLEDB.4.0; Data Source = C:\Northwind.mdb"
Dim cn As New OleDbConnection(strConn)

' Comando SQL per il recupero dei record.
Dim sql As String
sql = "SELECT * FROM Clienti ORDER BY NomeSocieta"
Dim cmd = New OleDbCommand(sql, cn)

' Assegnamo il comando al DataAdapter.
Dim da As New OleDbDataAdapter()
da.SelectCommand = cmd

' Apriamo la connessione.
cn.Open()
```

Microsoft

Esempio: usare DataSet disconnesso

- Creiamo il DataAdapter per aprire la connessione al database
- Creiamo DataSet e usiamo il DataAdapter per riempire il DataTable

```
Dim ds As New DataSet("dsClienti")   ' Il nome del DataSet è opzionale

' Riempiamo il DataSet specificando il nome della tabella.
 ds.Clear()
 da.Fill(ds, "Clienti")
```

Microsoft

Esempio: usare DataSet disconnesso

- Creiamo il DataAdapter per aprire la connessione al database
- Creaiamo DataSet e usiamo il DataAdapter per riempire il DataTable
- Modifichiamo i dati nelle DataTable usando i DataRow

```
Dim tblClienti As DataTable()
tblClienti = ds.Tables("Clienti") ' Referenzia la tabella Clienti del DataSet

' Esempio di modifica di un record.
tblClienti.Rows(0)("NomeSocieta") = "Società ABC"

' Esempio di inserimento nuovo record.
Dim NewRec As DataRow
NewRec = tblClienti.NewRow
NewRec("IDCliente") = "ID_ABC"
NewRec("NomeSocieta") = "SocietàABC"
tblClienti.Rows.Add(NewRec)

' Esempio di eliminazione record.
tblClienti.Rows(1).Delete()
' oppure...
tblClienti.Rows.Remove(1)
```

Microsoft

Esempio: usare DataSet disconnesso

- Creiamo il DataAdapter per aprire la connessione al database
- Creiamo DataSet e usiamo il DataAdapter per riempire il DataTable
- Modifichiamo i dati nelle DataTable usando i DataRow
- Aggiungiamo i comandi di aggiornamento al DataAdapter

```
Creazione manuale di un comando SQL
Dim sql As String
sql = "INSERT INTO Clienti (IDCliente, NomeSocieta VALUES(@IDCliente, @NomeSocieta)"
Dim cmd As New OleDbCommand(sql, cn)
da.InsertCommand = cmd
da.InsertCommand.Parameters.Add("@IDCliente", OleDbType.VarChar, 5, "ID_ABC")
da.InsertCommand.Parameters.Add("@NomeSocieta", OleDbType.VarChar, 40, "SocietaABC")

Generazione automatica dei comandi SQL.
' I comandi vengono generati sulla base di quanto contenuto nella SelectCommand

Dim cmdBuild As New OleDbCommandBuilder(da)
' Assegniamo i comandi generati al nostro DataAdapter
da.UpdateCommand = cmdBuild.GetUpdateCommand()
da.DeleteCommand = cmdBuild.GetDeleteCommand()
```

Microsoft

Esempio: usare DataSet disconnesso

- Creiamo il DataAdapter per aprire la connessione al database
- Creiamo DataSet e usiamo il DataAdapter per riempire il DataTable
- Modifichiamo i dati nelle DataTable usando i DataRow
- Aggiungiamo i comandi di aggiornamento al DataAdapter
- Aggiorniamo la fonte dati

```
' Il metodo Update() restituisce il numero dei record
' interessati dall'aggiornamento.
da.Update(ds, "Clienti")
```

Microsoft

Esempio: usare DataSet disconnesso

- Creiamo il DataAdapter per aprire la connessione al database
- Creiamo DataSet e usiamo il DataAdapter per riempire il DataTable
- Modifichiamo i dati nelle DataTable usando i DataRow
- Aggiungiamo i comandi di aggiornamento al DataAdapter
- Aggiorniamo la fonte dati
- Aggiorniamo il DataSet

```
ds.AcceptChanges()
```

Microsoft

Controlli Data-bound

- Alcuni controlli hanno la capacità di collegarsi a sorgenti dati e di rappresentarne il contenuto:
  - ListBox, BulletedList, RadioButtonList, CheckBoxList
  - TreeView, Menu, FormView, GridView, DetailsView
  - Datalist, Repeater
- Molte volte basta un semplice Drag & Drop !
  - Zero code !

Microsoft

SqlDataSource

- Collegamento a database SQL in modo dichiarativo
- Maschera l'uso delle classi ADO.Net per l'accesso ai dati (Command, Connection, etc)
- Data binding bi-direzionale
  - SelectCommand
  - InsertCommand, UpdateCommand, and DeleteCommand
- Caching opzionale per il risultato delle query
- Supporto di parametri nei comandi (Select, etc)

Microsoft

Usare SqlDataSource

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

Microsoft

Caching dei risultati

```
<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" />
```

Microsoft

Comandi con Parametri

- Le proprietà XxxParameters consentono di parametrizzare le query fatte al database
  - Esempio: valore della clausola WHERE nella SelectCommand il cui paramtetro è preso dalla query string o da un drop-down box
- XxxParameter specificano la sorgente del parametro

Microsoft

Usare i ControlParameter

```
<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" />
```

Microsoft

Chiamare le Stored Procedures

```
<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
```

Microsoft

Il controllo GridView

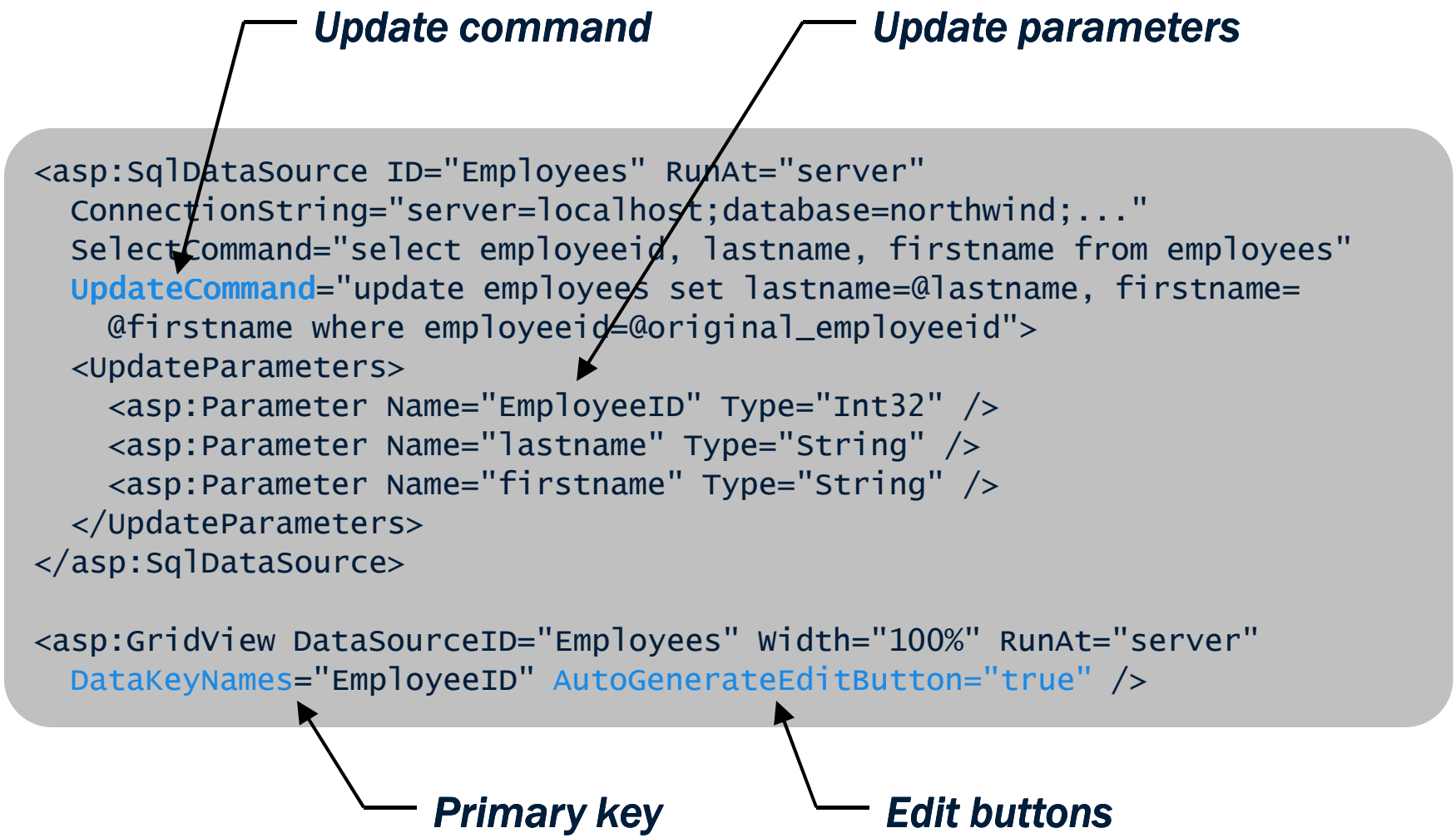
- Permette il sorting, paging, selecting, updating, ed il deleting
- Supporta colonne fatte con molti tipi, compresi i CheckBoxFields
- Interfaccia customizzabile

Specificare il tipo dei campi

```
<asp:SqlDataSource ID="Employees" RunAt="server"
  ConnectionString="server=localhost;database=northwind;..."
  SelectCommand="select photo, 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>
```

Microsoft

Editing con GridViews



Microsoft

Esempio per DetailsView

```
<asp:SqlDataSource ID="Employees" RunAt="server"
  ConnectionString="server=localhost;database=northwind;..."
  SelectCommand="select employeeid, photo, ... 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>
```

Microsoft

ObjectDataSource

- Permette di creare applicazioni con uno strato in più per l'accesso ai dati
  - È possibile inserire della business logic
  - Il codice di accesso ai dati è separato dalla UI
- Binding bidirezionale
  - SelectMethod, InsertMethod, UpdateMethod, and DeleteMethod
- Caching dei risultati opzionale
- Parametri

Microsoft

DEMO
Data Binding

- *SqlDataSource* per popolare *drop down list* e *gridview*
- *Gridview* filtrata da drop down list
- *Eliminazione* dal database: usare i *DataMembers* come *paramentri*
- Controllo *DetailsView*
- *Scrittura* nel DB: uso di *UpdateQuery*
- *EnableCaching* e *CacheDuration*

Microsoft

Q&A

mauro.minella@microsoft.com

Microsoft