

ACTIVE SERVER PAGES

1	ELEMENTI DI PROGRAMMAZIONE LATO SERVER.....	2
1.1	OGGETTO RESPONSE.....	3
2	COMUNICAZIONE CON L'UTENTE	5
2.1	MODULI.....	5
2.2	OGGETTO REQUEST.....	5
2.3	VARIABILI D'AMBIENTE E INTESTAZIONI http.....	6
2.4	COOKIES	6
3	LA PERSISTENZA DEI DATI	7
3.1	OGGETTO SESSION.....	7
3.2	OGGETTO APPLICATION	7
3.3	INIZIALIZZAZIONE DEGLI OGGETTI SESSION E APPLICATION: FILE GLOBAL.ASA	8
4	COMPONENTI ASP	9
4.1	AD ROTATOR	9
4.2	CONTENT LINKER	10
5	CONTROLLO DEL CLIENT E DEL FLUSSO DEGLI ERRORI.....	12
5.1	OGGETTO BROWSERTYPE	12
5.2	OGGETTO ASPERROR	12
6	LEGGERE E SCRIVERE NEL SERVER WEB	13
6.1	PASSAGGIO O INCLUSIONE DI ALTRE PAGINE	13
6.2	OGGETTO FILESYSTEM	13
6.3	OGGETTO SERVER	15
7	GESTIONE DEI DATABASE LATO SERVER – ADO.....	16
7.1	OGGETTO CONNECTION	16
7.2	OGGETTO RECORDSET.....	17

1 ELEMENTI DI PROGRAMMAZIONE LATO SERVER

Con “Programmazione lato server” si intende, in termini estremamente semplificati, l’esecuzione sul server (web) di programmi o script richiesti da un client. Generalmente il client riceve i risultati di tale elaborazione. La necessità della programmazione lato server sta nel fatto che la gestione delle informazioni sul server è gestita da programmi unicamente residenti sulla macchina (o comunque su macchine per cui è necessario avere i privilegi del server per operare) e non dai singoli client.

Vengono riportati tre listati. Una prima pagina HTML lato client; successivamente la stessa pagina lato server e, infine la pagina che l’utente riceve in seguito all’esecuzione della pagina lato server.

PAGINA LATO CLIENT (ESEMPIO.HTM)

```
<HTML>
<BODY>
  <B>
    <SCRIPT LANGUAGE="JavaScript">
      d = new Date();
      document.writeln("The current time is: " + d);
      if (d.getHours(>12)
        document.write("POMERIGGIO");
      else
        document.write("MATTINO");
    </SCRIPT>
  </B>
</BODY>
</HTML>
```

PAGINA LATO SERVER (ESEMPIO.ASP)

```
<% @Language=VBScript %>
<% Option Explicit %>
<HTML>
<BODY>
<B>
The current time is: <%=Time()%> <P>
<%
IF DatePart("h", Time())>=12 THEN
  Response.Write "POMERIGGIO"
ELSE
  Response.Write "MATTINO"
END IF
%>
</B>
</BODY>
</HTML>
```

PAGINA RICEVUTA DAL CLIENT (ESEMPIO.ASP)

```
<HTML>
<BODY>
<B>
The current time is: 15.21.54 <P>
POMERIGGIO
</B>
</BODY>
</HTML>
```

ACTIVE SERVER PAGES

Esiste un altro modo di far eseguire del codice sul server e consiste nell'impostare l'attributo RUNAT al valore "SERVER" nell'intestazione del tag <SCRIPT>. Questo metodo, come si potrà osservare nel prossimo esempio, comporta dei meccanismi di esecuzione particolari e, quindi, differenti dalla semplice sequenzialità delle istruzioni.

PAGINA LATO CLIENT CON SCRIPT ESEGUITO SUL SERVER (ESEMPIO2.ASP)

```
<HTML>
<BODY>
  <B>
  The current Time is:
  <SCRIPT LANGUAGE="VBScript" RUNAT="SERVER">

      Response.Write time() & " <P>"

      IF DatePart("h", Time())>=12 THEN
        Response.Write "POMERIGGIO"
      ELSE
        Response.Write "MATTINO"
      END IF

  </SCRIPT>
  <P>
  End of Script
</B>
</BODY>
</HTML>
```

PAGINA RICEVUTA LATO CLIENT CON SCRIPT ESEGUITO SUL SERVER (ESEMPIO2.ASP)

```
<HTML>
<BODY>
  <B>
  The current Time is:

  <P>
  End of script
</B>
</BODY>
</HTML>

15.38.11 <P>POMERIGGIO
```

1.1 OGGETTO RESPONSE

Lo scopo di questo oggetto è principalmente quello di inviare dati al client.

1.1.1 Metodi per l'invio dei dati

Response.Write "espressione" oppure <%= "espressione"%>
Consente di inviare stringhe di testo alla pagina client

Response.End
Termina l'esecuzione dello script sul client

ACTIVE SERVER PAGES

`Response.AddHeader NomiIntestazione, ValoreIntestazione`
Consente di inviare i valori di intestazione di una pagina HTML al client

1.1.2 Gestione del buffer

`Response.Buffer (IIS 5.0) = true/false` (subito dopo “optino Explicit”)
Consente di impostare il buffer di invio.

`.Buffer=true` → bufferizza la pagina e l’invio avviene solo a completamento della creazione

`.Buffer=false` → invia la pagina progressivamente alla ricezione dei dati

`Response.Clear`

Rimuove i dati dal buffer (svuota il buffer)

`Response.Flush`

E’ simile a `.Clear` ma prima di svuotare il buffer lo invia al client; cioè visualizza il più possibile della pagina

1.1.3 Permanenza nella cache del client

`Response.Expires = numero`

Imposta il numero di minuti durante i quali la pagina inviata rimane nella cache del client

`Response.ExpiresAbsolute = dataora`

Imposta la data precisa fino alla quale la pagina sarà disponibile nella cache del client

1.1.4 Ridirezionamento verso un’altra pagina

In HTML:

```
| <META http-EQUIV=REFRESH CONTENT="2;URL=www.esempio.com">
```

In JavaScript

```
| <SCRIPT LANGUAGE="JavaScript">  
| <!--  
|     window.location="http://www.esempio.com";  
| // -->
```

In ASP (oggetto Response)

```
| Response.Redirect "www.esempio.com"  
| Response.Redirect = "www.esempio.com"
```

1.1.5 Cookies

Scrittura

`Response.Cookies("NomeCookie")=valore`

Lettura

(Si veda l’oggetto Request)

2 COMUNICAZIONE CON L'UTENTE

2.1 MODULI

ASP è in grado di ricevere e rielaborare i dati inviati da un modulo sia col metodo POST che con il metodo GET.

Esempio di modulo

```
<FORM METHOD="POST" ACTION="www.miapagina.it/esempio.asp">
  <INPUT NAME="nome" TYPE=TEXT|CHECKBOX|RADIO|PASSWORD|... VALUE="valore">
  <SELECT NAME="mese">
    <OPTION VALUE="gennaio">gennaio
    <OPTION VALUE="febbraio">febbraio
    ...
  </SELECT>
</FORM>
```

Il metodo POST trasmette le informazioni impostate nel modulo sotto forma di collezione di dati codificati; al contrario GET invia la query di interrogazione direttamente accodandola all'URL di destinazione. Una stringa inviata con il metodo GET ha, di solito, la forma seguente:

```
| www.miapagina.it/esempio.asp?var1=val1&var2=val2&var3=val3...
```

2.2 OGGETTO REQUEST

Lettura dei dati inviati

A seconda della metodologia utilizzata per inviare i dati (GET o POST) l'oggetto Request mette a disposizione diversi strumenti per leggerne i valori.

Metodo GET

Valore=Request.QueryString(NomeVariabile) (QueryString è una collezione)

Metodo POST

Valore=Request.Form(NomeVariabile) (Anche Form è una collezione)

Se la variabile "NomeVariabile" viene omessa verrà restituita l'intera stringa di query.

2.3 VARIABILI D'AMBIENTE E INTESTAZIONI http

E' possibile richiamare i valori delle intestazioni http e delle variabili d'ambiente tramite il metodo "ServerVariables" dell'oggetto "Request".

La sintassi è la seguente: `Valore = Request.ServerVariables(Intestazione)`
In cui "Intestazione" è una stringa che può assumere uno dei seguenti valori.

Intestazioni http

<code>http_accept</code>	<code>http_connection</code>	<code>http_user_agent</code>	<code>http_cookie</code>
<code>http_accept_language</code>	<code>http_host</code>	<code>http_referrer</code>	<code>all_http</code>

Variabili di ambiente

<code>URL</code>	<code>PATH_TRANSLATED</code>	<code>QUERY_STRING</code>	<code>SERVER_SOFTWARE</code>
<code>PATH_INFO</code>	<code>APPL_PHYSICAL_PATH</code>	<code>SERVER_NAME</code>	

2.4 COOKIES

Come si è già illustrato:

Scrittura

```
Response.Cookies("NomeCookie")=valore
```

Lettura

```
<%=Request.ServerVariables("COOKIE")%> `tutti i cookies  
<%=Request.Cookies("NomeCookie", "chiave")
```

3 LA PERSISTENZA DEI DATI

3.1 OGGETTO SESSION

Questo oggetto mantiene lo stato nell'ambito della durata della visibilità del sito. Questo "mantenimento" avviene attraverso l'impostazione delle "Variabili di Sessione".

La sintassi è la seguente.

Scrittura:

```
Session(NomeVariabileDiSessione)=Valore
```

Lettura:

```
Variabile=Session(NomeVariabileDiSessione)
```

Se si vuole "uccidere" la sessione corrente è sufficiente richiamare il metodo "Abandon" come illustrato qui di seguito.

```
Session.Abandon
```

Esistono due proprietà importanti dell'oggetto "Session":

Session.SessionID	'Restituisce un numero unico caratteristico 'di una sessione
Session.Timeout	'E' il numero di minuti di inattività dopo i 'quali la sessione scade

Eventi

L'evento OnStart si verifica quando un nuovo visitatore (cioè un utente che non possiede ancora un'istanza Session) arriva al sito.

3.2 OGGETTO APPLICATION

Impostazione di variabili Application

```
Application(NomeVariabileApplicazione)=Valore|Array
```

Lettura di variabili Application

```
Variabile=Application(NomeVariabileApplicazione)
```

Eventi

L'evento OnStart si verifica solo una volta, prima che venga creato il primo Session.

3.3 INIZIALIZZAZIONE DEGLI OGGETTI SESSION E APPLICATION: FILE GLOBAL.ASA

Gli eventi di inizializzazione e chiusura degli oggetti Session e Application affinché siano eseguiti correttamente devono essere inseriti in un file denominato GLOBAL.ASA posizionato nella radice del Web.

FILE GLOBAL.ASA

```
<SCRIPT LANGUAGE="VBScript" RUNAT="Server">  
  
  Sub Application_OnStart()  
    ...  
  End Sub  
  
  Sub Session_OnStart()  
    ...  
  End Sub  
  
  Sub Application_OnEnd()  
    ...  
  End Sub  
  
  Sub Session_OnEnd()  
    ...  
  End Sub
```

4 COMPONENTI ASP

ASP mette a disposizione dello sviluppatore lato server degli oggetti che implementano le più comuni funzioni oggi utilizzabili in Internet.

4.1 AD ROTATOR

“Ad Rotator” è un componente che consente di inserire in una pagina web un banner scelto a caso tra una serie di banner specificati. Ad ogni banner viene specificato un “peso” che ne determina la frequenza di scelta rispetto a tutti gli altri banner. L’utilizzo di questo oggetto avviene attraverso due passi fondamentali. Il terzo lo compirà il server nel momento in cui il client richiederà la pagina in cui sono inseriti i banner (passo 2).

1) Creazione del ROTATOR SCHEDULE FILE (sia esso il file *AdList.txt* nella root)

Il file è composto da due parti. La prima parte riguarda le prime quattro righe del file che specificano le dimensioni, la posizione e l’aspetto del banner; segue un asterisco, segue una lista di righe. Ogni riga specifica una caratteristica di un banner. Per identificare correttamente un banner sono necessarie quattro righe, quindi la lista è divisa in pacchetti di quattro righe successive. Ogni pacchetto riguarda un unico banner. Segue lo schema del file. La somma dei pesi di tutti i banner non deve superare 10000

AdList.txt

```
REDIRECT url_se_si_clicca_sul_banner
WIDTH larghezza_in_pixel
HEIGHT altezza_in_pixel
BORDER spessore_bordo
*
URLPrimaImmagine
URLHomePagePrimaImmagine
TestoAlternativoPrimaImmagine
PesoPrimaImmagine
URLSecondaImmagine
URLHomePageSecondaImmagine
TestoAlternativoSecondaImmagine
PesoSecondaImmagine
...
```

2) Creazione della pagina lato server che implementa l’oggetto Ad Rotator (ESEMPIO.ASP)

E’ la pagina ASP che contiene uno dei banner che si desidera inserire. E’ necessario creare l’oggetto e richiamare il suo metodo “GetAdvertisement” laddove si desidera inserire il banner.

ESEMPIO.ASP

```
<% @ LANGUAGE="VBScript">
<% Optino Explicit%>
<% Dim ObjAd
    Set ObjAd = Server.CreateObject("MSWC.AdRotator")%>
<HTML>
<BODY>
    Qui sotto inserisco il banner<BR>
    <%= ObjAd.GetAdvertisement("AdList.txt")% & "<BR>">
    Qui sopra c’è il banner
</BODY>
</HTML>
```

ACTIVE SERVER PAGES

3) Pagina ricevuta dal client (risultato dello script lato server) (ESEMPIO.ASP)

Ovviamente il client, visitando la pagina ESEMPIO.ASP riceve il codice HTML completo affinché sia visualizzabile il banner desiderato. Ne segue il codice esatto.

ESEMPIO.ASP

```
<HTML>
<BODY>
  Qui sotto inserisco il banner<BR>
  <A HREF="URLHomePageXesimaImmagine">
    <IMG SRC="URLXesimaImmagine"
      ALT="TestoAlternativoXesimaImmagine"
      HEIGHT="altezza_in_pixel"
      WIDTH="larghezza_in_pixel"
      BORDER="spessore_bordo"
    >
  </A>
  Qui sopra c'è il banner
</BODY>
</HTML>
```

4.2 CONTENT LINKER

Questo oggetto consente di parametrizzare i link contenuti nelle diverse pagine così da poterli modificare facilmente mantenendo un unico file, il "Content Linking List File". Si procede in due passi consecutivi. Prima di tutto si crea il CLLF, successivamente si inseriscono nelle diverse pagine, i collegamenti a tale file.

1) Struttura del CLLF

FILE CLLF01.TXT

URLPaginaWeb1	DescrizionePagina1	Commento1
URLPaginaWeb2	DescrizionePagina2	Commento2
URLPaginaWeb3	DescrizionePagina3	Commento3
...		

L'oggetto Content Ginger (MSWC.NextLink) possiede diversi metodi che possono essere utilizzati nelle pagine ASP.

.GetListCount(FileCLLF)	'il numero di link nel file
.GetListIndex(FileCLLF)	'l'indice corrente nel file
.GetNextURL(FileCLLF)	'l'URL successivo nella lista
.GetNextDescription(FileCLLF)	'descrizione della prossima pagina
.GetPreviousURL(FileCLLF)	'l'URL precedente nella lista
.GetPreviousDescription(FileCLLF)	'descrizione della pagina precedente
.GetNthURL(FileCLLF,i)	'I-esimo URL
.getNthDescription(FileCLLF,i)	'I-esima descrizione

2) Inclusione dell'URL in una pagina ASP

Immaginiamo che precedentemente avessimo inserito un collegamento alla pagina ESEMPIO.ASP e, successivamente all'adozione dell'oggetto Content Linker, tale collegamento sia stato inserito nel file CLLF.TXT nella dodicesima riga. Precedentemente il file aveva questa forma.

Vecchio File PAGINA.ASP

```
...  
<A HREF="ESEMPIO.ASP">  
    ESEMPIO  
</A>  
...
```

Nuovo File PAGINA.ASP

```
...  
<% Set ObjNL=Server.CreateObject("MSWC.NextLink")%>  
<A HREF="<%=Response.Write(ObjNL.GetNthURL("CLLF.TXT",12))%">  
    <%=ObjNL.GetNthDescription("CLLF.TXT",12)%>  
</A>  
...
```

5 CONTROLLO DEL CLIENT E DEL FLUSSO DEGLI ERRORI

5.1 OGGETTO BROWSERTYPE

L'oggetto `BrowserType` consente di identificare il tipo e le capacità del browser del client.

Definizione:

```
Dim ObjBT
Set ObjBT=Server.CreateObject("MSWC.BrowserType")
```

Proprietà principali

```
.Browser
.Version
.Frames (booleana)
.Tables (booleana)
.BackgroundSounds (booleana)
.VBScript (booleana)
.JavaScript (booleana)
```

5.2 OGGETTO ASPERROR

Esiste, infine, un oggetto specifico per analizzare ed identificare gli errori .

Definizione:

```
Dim ObjAE
Set ObjAE=Server.GetLastError()
```

Proprietà principali

```
.ASPCode
.Number
.Source
.Category
.File
.Line
.Column
.Description
.ASPDescription
```

La pagina di reindirizzamento dell'errore (.File) può essere inserita nelle impostazioni degli ERRORI PERSONALIZZATI di IIS.

6 LEGGERE E SCRIVERE NEL SERVER WEB

6.1 PASSAGGIO O INCLUSIONE DI ALTRE PAGINE

Esistono tre modi fondamentali di collegarsi automaticamente ad altre pagine.

INCLUSIONE DI FILE

```
...
<!--#include file="noefile"-->
...
```

ESECUZIONE DI UNA PAGINA ASP

```
...
<%
    Response.Write("SONO NELLA PAGINA 1 <BR>")
    Server.Execute("pagina2.asp")
    Response.Write("SONO TORNATO NELLA PAGINA 1 <BR>")
%>
```

TRASFERIMENTO DEL CONTROLLO AD UN'ALTRA PAGINA

```
<%Server.Transfer("Pagina2.asp")%>
```

In quest'ultimo caso il contesto rimane quello della pagina chiamante così come valore della variabile URL di ServerVariables.

6.2 OGGETTO FILESYSTEM

Un oggetto molto importante è il File System Object che consente di gestire file e cartelle

Definizione

```
Dim ObjFSO
Set ObjFSO = Server.CreateObject("Scripting.FileSystemObject")
```

Metodi fondamentali

```
.FolderExists("folder_path")      `restituisce True o False
.FileExists("file_path")           `restituisce True o False
.GetFile("file_name")              `restituisce un oggetto di tipo file
.GetFolder("folder_name")          `restituisce un oggetto di tipo folder
```

I percorsi dei file

Affinché l'oggetto FileSystemObject sia in grado di gestire correttamente i nomi dei file e delle cartelle è necessario utilizzare i metodi MapPath dell'oggetto Server. Questo restituisce il path completo locale del file specificato. Ad esempio

```
...
Set ObjFile = ObjFSO.GetFile(Server.MapPath("log.txt"))
...
```

6.2.1 Oggetto File

Una volta ottenuto un oggetto File dal metodo GetFile si possono controllare le caratteristiche fondamentali del file stesso.

Proprietà in lettura-scrittura

.Attributes
.Name

Proprietà in sola lettura

.DateCreated	.ParentFolder	.ShortPath
.DateLastModified	.Path	.Size
.DateLastAccessed	.ShortName	.Type
.Drive		

6.2.2 Oggetto Folder

Anche l'oggetto Folder possiede delle proprietà importanti

Proprietà in lettura-scrittura

.Attributes
.Name

Proprietà in sola lettura

.DateCreated	.IsRootFolder	.ShortName
.DateLastAccessed	.ParentFolder	.Size
.DateLastModified	.Path	.Type
.Drive		

Collezioni particolarmente importanti

.Files	collezione contenente i file della cartella
.Subfolders	collezione contenente le sottocartelle della cartella

6.2.3 File System Object: Operazioni comuni sui file

Creazione di un file sequenziale

```
Dim ObjFSO, ObjFile
Set ObjFSO = Server.CreateObject("Scripting.FileSystemObject")
Set ObjFile = ObjFSO.CreateTextFile(file_name)
```

Apertura di un file sequenziale

```
Dim ObjFSO, ObjFile
Set ObjFSO = Server.CreateObject("Scripting.FileSystemObject")
Set ObjFile = ObjFSO.OpenTextFile(file_name, modo, crea, formato)
`VALORI DEI PARAMETRI
`file_name = nome del file da aprire
`modo      = ForReading(1) ForWriting(2) ForAppending(8)
`crea      = True|False a seconda se si desidera creare il file nel
`           caso in cui non esista
`formato   = -2(default) 0(ASCII) -1(Unicode)
```

ACTIVE SERVER PAGES

Letture di un file sequenziale

```
Dim RigaLetta
Dim ObjFSO, ObjFile
Set ObjFSO = Server.CreateObject("Scripting.FileSystemObject")
Set ObjFile = ObjFSO.OpenTextFile("esempio.txt", 1, True, 0)
While not ObjFile.AtEndOfStream
    RigaLetta = ObjFile.Read(numero_caratteri)
    Response.Write RigaLetta & "<BR>"
Wend
```

Scrittura in un file sequenziale

```
Dim ObjFSO, ObjFile
Set ObjFSO = Server.CreateObject("Scripting.FileSystemObject")
Set ObjFile = ObjFSO.OpenTextFile("esempio.txt", 2, True, 0)
ObjFile.Write("testo di esempio")
ObjFile.WriteLine(" che continua ma dopo vado a capo")
'ora aggiungo tre righe bianche
ObjFile.WriteLine(3)
```

Chiusura di un file sequenziale

```
ObjFile.Close
ObjFile.Nothing
```

6.3 OGGETTO SERVER

Spesso si è già utilizzato l'oggetto Server o comunque qualcuno dei suoi accetti costituenti. Qui vengono riepilogati i metodi e le proprietà già utilizzati (tra cui molti costruttori).

```
.CreateObject(nome_oggetto) .Execute(path_pagina) .GetLastError()
.MapPath(percorsovirtuale) .Transfer(path_pagina) .ScriptTimeout
```

7 GESTIONE DEI DATABASE LATO SERVER – ADO

Quando si desidera accedere ad un database a partire da una pagina ASP è necessario utilizzare ADO (ActiveX Data Object). In particolare sono necessari due oggetti: Connection e Recordset. Il primo consente di effettuare la connessione al database, il secondo serve principalmente per manipolare i dati del database a cui ci si è connessi.

7.1 OGGETTO CONNECTION

ESEMPIO DI CONNESSIONE (SENZA DSN)

```

<!--#include virtual= "/adovbs.inc"-->
<% Dim ObjConn

    `DEFINIZIONE E APERTURA DELLA CONNESSIONE
    Set ObjConn = Server.CreateObject("ADODB.Connection")
    ObjConn.ConnectionString= "DRIVER="
                            & "{Microsoft Access Driver (*.mdb)};"
                            & "DBQ=C:\LIBRI\LIBRI.MDB"
    ObjConn.Open Info, UserName, Password

    `DEFINIZIONE E APERTURA DEL RECORDSET
    Dim ObjRS
    Set ObjRS = Server.CreateObject("ADODB.Recordset")
    ObjRS.Open StringaSQL, ObjConn, CursorType, LockType, CommandType
    Do While Not ObjRS.EOF
        Response.Write ObjRS("NomeCampo") & "<BR>"
        ObjRS.MoveNext
    Loop
    ObjRS.Close
    Set ObjRS = Nothing

    ObjConn.Close
    Set ObjConn = Nothing
%>

```

Stringhe di connessione

Le stringhe di connessione variano a seconda del modo con cui si desidera interfacciarsi con il database.

1. Per Database Access:
"DRIVER={Microsoft Access Driver (*.mdb)};DBQ=path\nomefile.mdb"
2. Per Access OLEDB:
"Provider=Microsoft.Jet.OLEDB.3.51;DataSource=path\nomefile.mdb"
3. Per connessioni DSN (ODBC)
"DSN=path\nomefile.dsn"

7.2 OGGETTO RECORDSET

Creazione:

```
Dim ObjRS
Set ObjRS=Server.CreateObject("ADODB.Recordset")
```

Apertura

```
ObjRS.Open Source, Connection, CursorType, LockType, CommandType
```

Parametri di .Open

Source :	Nome di una tabella o una stringa SQL di selezione
Connection:	Connessione (oggetto Connection) a cui fare riferimento
CursorType:	adOpenForwardOnly=0 adOpenKeyset=1 adOpenDynamic=2 adOpenStatic=3
LockType:	adLockReadOnly=1 adLockPessimistic=2 adLockOptimistic=3 adLockBatchOptimistic=4
CommandType:	adCmdUnknown=&H0008 adCmdText=&H0001 adCmdTable=&H0002 adCmdStoredProc=&H0004 adCmdFile=&H0100 adCmdTableDirect=&H0200

Alcune proprietà dell'oggetto Recordset

.EOF	.Value	.NumericScale
.BOF	.Type	.DefinedSize
.Name	.Precision	.ActualSize
.RecordCount		

Impostazione e lettura dei valori dei campi del Recordset (metodi equivalenti)

```
ObjRS.Fields("NomeCampo") → ObjRS("NomeCampo") → ObjRS("NomeCampo").Value →  
ObjRS.Fields("NomeCampo").Value
```

Metodi per la modifica, eliminazione e lo spostamento

.AddNew	.Delete	.MoveNext
.Update	.MoveFirst	.MovePrevious
.Sort "elenco campi"	.Filter "cond. where"	

Metacaratteri di Like (SQL)

```
% zero o più caratteri (già *)
_ un carattere (già ?)
[...] tutti i caratteri nell'intervallo
```