

Corso alternativo di programmazione

Generazione automatica di Pagine per Internet

di Francesco Petroni

In pochi mesi, buona parte degli "informatici" professionisti si sono dovuti riconvertire alle tecnologie, alle numerose tecnologie, interessate dal fenomeno Internet. In particolare, buona parte dei programmatori tradizionali sono stati trasferiti "armi e bagagli" da un mondo in cui esistevano pochi linguaggi affidabili, in quanto consolidati nel corso degli anni ed in cui esistevano precise regole di comportamento, in un nuovo mondo in cui le tecnologie si susseguono, affiancandosi, sostituendosi o sovrapponendosi l'una alle altre ed in cui, per essere ottimisti, si può dire che è ancora presto per fissare regole.

Da alcuni punti di vista Internet rappresenta per loro sicuramente un passo indietro, nel senso che molto del codice che scrivono non può e quindi non deve essere compilato, in quanto viene interpretato solo al momento dell'esecuzione o dal browser oppure dal server, nel senso che non è assolutamente possibile un efficace lavoro di debugging del codice stesso, in quanto non esiste un ambiente specifico che semplifichi questo lavoro, nel senso che il programma può non funzionare o perché il destinatario non dispone di un browser adeguato (della versione prevista dall'incauto programmatore), o perché il destinatario stesso ha attivato dei livelli di sicurezza in grado di bloccare qualsiasi componente, anche il più microscopico, necessario per far funzionare l'applicazione.

C'è poi una difficoltà nella definizione esatta degli output che possono dipendere dalla risoluzione scelta dall'utente, o dalla dimensione dei ca-

ratteri che costui imposta per il proprio browser.

Anche la progettazione dell'applicazione è un'attività che presenta vincoli di tutti i tipi, soprattutto perché un'applicazione per Internet è comunque un'applicazione client/server e quindi vanno regolati i rapporti tra questi due soggetti (chi fa cosa), va ipotizzata la modalità e la velocità di trasmissione dei dati, ecc.

Io penso che, paradossalmente, sono proprio tutte queste limitazioni che rendono la programmazione per Internet più stimolante, in quanto c'è ancora molto spazio per la "creatività" nella ricerca della soluzione "geniale" del problema da risolvere.

Alcune considerazioni sui prodotti per sviluppare applicazioni per Internet

Oggi esistono tanti prodotti di tanti produttori che servono per creare siti Internet, oppure pagine Internet, oppure ancora componenti per pagine Internet.

Alcuni sono più orientati all'estetica della pagina (sia perché, come se non bastassero i problemi, la prima cosa che si nota in un programma per Internet è se le pagine realizzate sono "belle"), altri alla dinamicità della pagina, altri ai rapporti che questa ha con il database in cui sono contenuti i dati da visualizzare, ecc.

La tendenza è quella di realizzare le pagine, anche quelle interattive o quelle prodotte da un programma, con editor di tipo "visuale", in cui l'utente inserisce elementi o imposta proprietà o effetti speciali, che poi sono tradotti in codice (sia HTML, sia DHTML, sia ASP) dall'editor stesso. Anche l'accesso ai dati si realizza inserendo oggetti e impostando come proprietà informazioni sulla base dei dati, sui campi da visualizzare e sulle modalità di gestione di tali dati.

In generale, questi prodotti dispongono di voluminose librerie di funzioni standard, già pronte e scritte nei vari linguaggi (quasi sempre in JavaScript per quelle che lavorano sul client ed in Visual Basic Script per quelle che lavorano sul server). Quando il programmatore prepara la pagina inserendo gli oggetti, il prodotto crea i richiami alle varie librerie, sfruttando il comando Include, che tiene all'esterno dell'applicazione la libreria, ma che ne sfrutta il contenuto. In pratica con il comando Include si effettua un Copia ed Incolla dinamico in fase di esecuzione del programma.

Questo modo di lavorare è facilitato dal fatto che il codice dell'applicazione non deve essere compilato e registrato, e che quindi è solo necessario che le librerie con le funzioni siano disponibili al momento dell'esecuzione dell'applicazione.

Molti programmatori poi, specie quelli più esperti, preferiscono realizzare proprie librerie o modificare quelle disponibili nei vari prodotti per renderle più immediatamente utilizzabili dalla propria applicazione.

Ma perché programmare per Internet

L'obiettivo che ci vogliamo porre in questo articolo è quello di analizzare gli strumenti con i quali è possibile realizzare un programma che produce pagine per Internet. Ma prima di cominciare vediamo perché è un bene sostituire ad un sito fatto di pagine, un programma che produce le pagine.

All'inizio la dimensione del sito si misurava in numero di pagine e se queste erano tante diventava un problema la loro gestione, soprattutto per il fatto che chi preparava le pagine non era lo stesso personaggio che ne definiva i contenuti.

Con l'evoluzione delle tecnologie un fondamentale passo in avanti è stata la nascita del concetto di stile (a proposito, prima o poi ne parleremo), che permette di isolare l'aspetto estetico da quello contenutistico e quindi permette di preparare tante pagine di contenuti "puliti" che fanno riferimento ad un unico foglio di stile, che risiede su file esterno.

Il passo successivo è stato quello di preparare le pagine con i contenuti di un database. Un esempio evidente è costituito dai cataloghi on-line, centinaia di pagine ricche di informazioni, di tipo strutturato, di tipo testuale, di tipo grafico, ma in definitiva identiche come organizzazione. La soluzione è quella di fare un'unica pagina prototipo da riempire con i campi di un singolo record di un database, magari prodotta "a richiesta" da parte dell'utente.

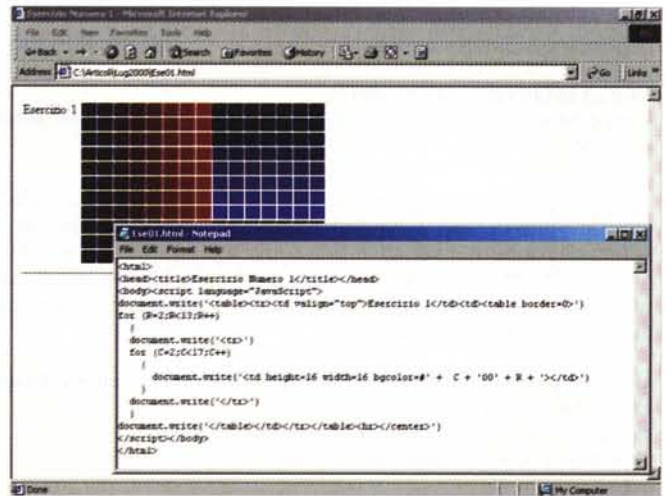
Un ulteriore passo può essere quello che prevede una vera e propria procedura tradizionale, con tanto di maschere di acquisizione e gestione dei dati, la cui finalità è quella di produrre in automatico i contenuti del sito.

Come esempio si pensi ad un "portale" che contiene decine di elementi che cambiano "minuto per minuto". Il responsabile di tale portale non agisce certo sulle pagine HTML, ma lavora con procedure in cui inserirà, ad esempio, le News, con indicazione del loro periodo di validità, dell'importanza che devono avere all'interno della pagina, del link a cui si riferiscono e così la produzione della pagina sarà quindi automatica.

Ebbene, chiarito cosa significa produzione automatica di pagine, occorre parlare un po' della strumentazione a di-

Figura 1 - Esercizio 1 - Uso dell'istruzione Document.Write in un codice JavaScript.

Un appassionato di programmazione, sia esso un programmatore professionista oppure un utilizzatore evoluto, sicuramente non ama i prodotti che producono elementi "statici", ad esempio non ama il word processor, anche se evidentemente lo usa per scrivere i suoi documenti, e non ama gli editor che producono semplice codice HTML. Per un programmatore una pagina HTML DEVE essere prodotta via codice e quindi deve essere il risultato di una elaborazione. Quando si scrive un programma che produce una pagina Internet si presentano sempre tre alternative: far lavorare il server, far lavorare il client, oppure far lavorare ciascuno dei due per quanto di sua competenza. In questo primo esercizio, descritto nel testo, usiamo l'istruzione JavaScript Document.Write, il cui compito è proprio quello di generare codice HTML. Nella figura vediamo codice e risultato.



sposizione per ottenere tale obiettivo.

Ci dedicheremo in particolare ai due linguaggi "per Internet" che vanno per la maggiore e poi proporremo alcuni esperimenti di media complessità, facilmente realizzabili su qualsiasi macchina.

Primi esperimenti con il JavaScript

Una pagina HTML può contenere del codice JavaScript opportunamente

scritto ed individuato all'interno del codice HTML.

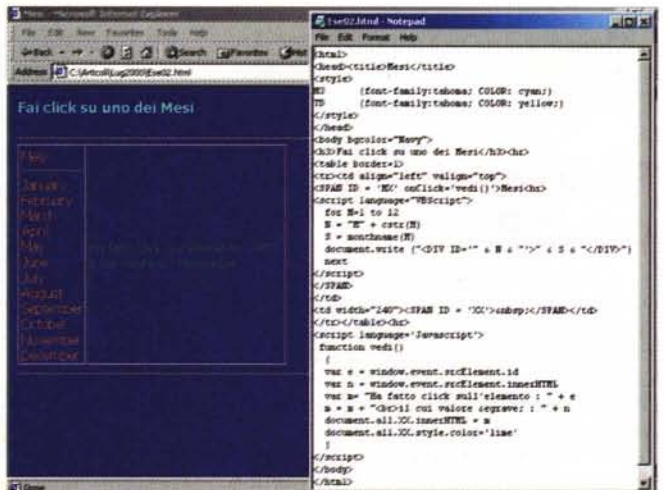
Questo codice JS viene inviato, ancora sotto forma di codice, dal server al client, in quanto è il browser che si incarica di interpretarlo.

E' quindi possibile creare un programma in JavaScript che fa sostanzialmente due cose: esegue dei calcoli intesi in senso lato e produce il risultato sotto forma di codice HTML all'interno di una pagina Internet. Un esempio di comando JS che si usa per produrre codice HTML è il seguente:

```
document.write("<html><head><title>Zero</title></head><body>Ciao</body></html>")
```

Figura 2 - Esercizio 2 - Esempio di programmazione Object.

Una delle caratteristiche più stimolanti della programmazione per Internet consiste nel poter gestire in un'ottica Object il documento prodotto. E' possibile identificare tutti gli elementi della pagina, gestirne le proprietà, intercettare gli eventi. Nell'esercizio mostriamo appunto come sia possibile identificare gli elementi della pagina ed intercettare gli eventi su di essi. Vediamo anche come sia possibile utilizzare nella stessa pagina due differenti linguaggi, nel nostro caso il Visual Basic Script, per sfruttare le sue comode funzioni sulle date, ed il JavaScript al quale si delega la gestione degli Object.



Questo piccolo comando JS può essere eseguito in due modi, o all'interno di un semplice tag <script>:

```
<script language="JavaScript">
<document.write ("<html><head><title>Zero</title></head><body>Ciao</body></html>")
</script>
```

ed in questa maniera viene eseguito direttamente in fase di caricamento della pagina, oppure inserito in una funzione JavaScript eseguita al verificarsi di un evento, ad esempio al click su un pulsante o al load della pagina:

```
<input type="button" name="P1" value="Premi" onclick="Esegui()">
<body onload="Esegui()">
<script language="JavaScript">
function Esegui()
{
document.write ("<html><head><title>Zero</title></head><body>Ciao</body></html>")
}
</script>
```

In questo caso il codice viene eseguito e quindi la pagina viene prodotta quando qualcuno fa click sul pulsante.

E' evidente e stimolante il fatto che in pratica l'intero codice della pagina può essere prodotto da un programma, non solo il codice HTML, ma anche quello DHTML, o quello JavaScript. Nel caso mostrato tra qualche riga vediamo un programma JavaScript che genera un programma JavaScript completo, nel caso specifico produce un pulsante con il suo evento onClick:

```
<script language="JavaScript">
document.write ("<input type=button value=Red name=P onClick=document.bgColor='#ff0000'>")
</script>
```

Questi tre pezzettini di codice mostrano tre possibili modalità di esecuzione di un codice JavaScript.

Primi esperimenti con ASP

Il codice JavaScript viene interpretato dal client; in pratica è il browser che contiene un "motorino" in grado di riconoscere ed eseguire il programma. La tecnologia Active Server Pages (ASP) invece prevede che il programma venga eseguito dal "motorino" presente sul server, ma questo è disponibile per ora solo nell'Internet Information Server della Microsoft.

Il linguaggio può essere ancora una

volta JavaScript oppure il Visual Basic Script.

Semplificando, potremo dire che un

programma ASP contiene istruzioni, funzioni assolutamente in linea con gli standard VB, VBA e VBS. Per produrre il codice HTML per la pagina che viene generata si utilizza l'istruzione Response.Write.

Un semplice esempio di ASP potrebbe essere questo:

```
<HTML>
<%
Response.Write "Inizio della Parte ASP<hr>"
For R=1 to 20
Response.Write "Questa è la Riga Numero " & R & "<br>"
Next
Response.Write "<hr>Fine della Parte ASP"
%>
<HTML>
```

In pratica il programma ASP produce stringhe contenenti codice HTML, che viene inviato al client in forma di codice HTML (quindi viene inviato il risultato dell'esecuzione del programma, non il suo listato). I tag "<% ... %>" individuano la parte di codice ASP che va scritto al suo posto all'interno del normale codice HTML. Lo stesso programma di prima lo possiamo realizzare con un codice meno elegante, ma che produce lo stesso effetto:

```
<HTML>
Inizio della Parte ASP<hr>
<% For R=1 to 20 %>
Questa è la Riga Numero <%= R %><br>
<% Next %>
<hr>Fine della Parte ASP
<HTML>
```

Per eseguire il codice ASP è indispensabile disporre di un server Internet, ma per i nostri esperimenti è sufficiente il Personal Web Server (PWS), anche quello per Windows 95 o Windows 98 (Windows 2000 Professional lo ha in dotazione). E' indispensabile creare una cartella virtuale nella quale posizionare il file, che a sua volta deve essere salvato con il suffisso ASP.

Ad esempio, se la macchina si chiama TEST, la cartella PROVE e il file ESEMPIO.ASP, l'indirizzo per eseguirlo deve essere:

<http://test/prove/esempio.asp>

Il PWS lo trovate nell'Option Pack di Windows NT, nel secondo CD del pacchetto Visual Basic ed in buona parte dei CD della Microsoft Press che hanno a che fare con lo sviluppo per Internet.

Dopo aver parlato di JavaScript e di ASP e tornando al tema della generazione automatica delle pagine HTML, risulta evidente il fatto che JavaScript sarà utilizzato per risolvere i problemi lato client (come il controllo dei dati in fase di digitazione), mentre ASP sarà utilizzato lato server (ad esempio quando occorre accedere ad un database). Rimangono comunque tante le problematiche risolubili, sia con il JavaScript che con le

pagine ASP e quindi può essere utile darsi ulteriori regole di comportamento.

Vediamo gli esercizi

A corredo degli articoli abbiamo preparato 8 esercizi di cui mostriamo, in 9 figure, sia gli output che tutto il codice. Trovate comunque tutti i file relativi all'articolo, figure comprese, nel CD allegato e nel nostro sito.

Il codice è impaginato in modo che "venga bene" nella foto e conseguentemente appare un po' disordinato ai puristi.

Nel primo esercizio (in figura 1) verifichiamo come sia possibile produrre tutto il contenuto della pagina con un codice JavaScript. Nel nostro caso abbiamo realizzato due loop, quello esterno per riga e quello interno per colonna, che producono una tabella. Il colore delle celle della tabella viene determinato da

calcoli numerici che coinvolgono il contatore delle righe e quello delle colonne.

Il programma svolge due lavori ben individuabili: esegue i calcoli che servono per preparare, sotto forma di stringa, il codice HTML e visualizza il codice prodotto con il comando Document.Write.

Il secondo esercizio (figura 2) ci permette di verificare numerose cosette. Prima di tutto la possibilità di utilizzare due linguaggi nella stessa pagina/programma. E' evidente che trattandosi ancora di programmazione lato client è necessario che il browser sia in grado di interpretare ambedue i linguaggi.

Poi notiamo la potenza del tag DIV, che permette di dare un nome ad un elemento della pagina, nel nostro caso

Figura 3 - Esercizio 3 - E' necessaria una totale padronanza degli elementi in gioco.

Il buon programmatore deve saper maneggiare tutti gli oggetti inseribili in un form. Nel nostro caso vediamo come si inseriscono, da codice, elementi in un oggetto di tipo Select, come si possono anche togliere e come, dell'elemento scelto, sia possibile conoscere ogni proprietà, come progressivo, identificativo e valore. L'esercizio non tratta le numerose altre varianti, la più stimolante delle quali è quella che prevede la selezione multipla degli elementi.

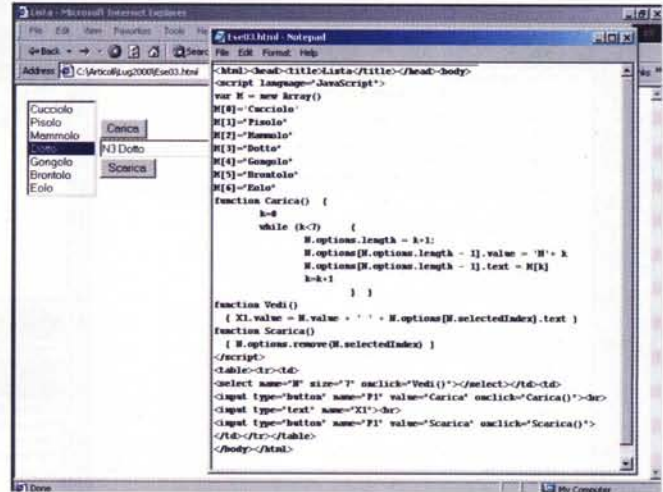
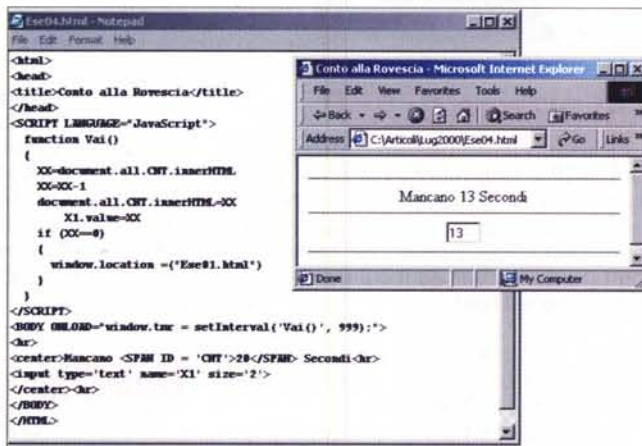


Figura 4 - Esercizio 4 - Utilizzo del Timer.

Una pagina "dinamica" è in grado di reagire agli eventi ovvero alle azioni che l'utente esegue sugli elementi della pagina. Importantissimo è anche l'evento Timer, che produce degli eventi temporizzati a cui si possono collegare delle routine che eseguono con tale periodicità "qualcosa" sulla pagina senza che l'utente debba fare nulla (può anche solo guardare stando a braccia conserte). Ad esempio si può prevedere una navigazione automatica e temporizzata da una pagina alle successive.



proprietà e metodi. Chi già conosce questo modo di programmare dovrà solo trovare le istruzioni JavaScript, che già conosce dal punto di vista concettuale.

Per caricare l'oggetto usiamo un array JavaScript opportunamente caricato di dati, che poi viene via via letto per alimentare l'oggetto Select. Se l'oggetto si chiama N utilizzeremo:

N.selectedIndex numerodell'Item, ovvero il progressivo

N.value proprietà Value, ad esempio il codice dell'articolo

N.options[n].text suo valore esterno, ad esempio la descrizione dell'articolo.

Il quarto esercizio, il più semplice fino ad ora, utilizza il comodissimo Timer

al paragrafo che mostra il nome del mese, e poi di intercettare l'evento click sull'elemento stesso. Queste tecniche si basano sull'uso del modello ad oggetti della pagina, fondamentale non tanto nella produzione automatica della pagina, quanto nella sua interazione con l'utente.

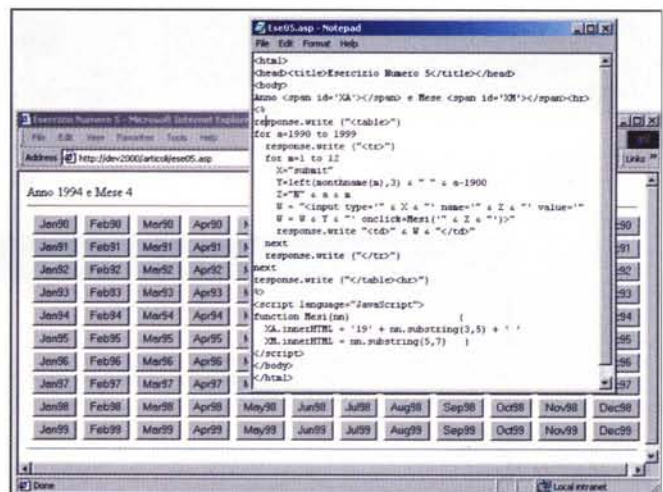
Vale la pena di ricordare che nel primissimo HTML l'unica interazione possibile erano i link per la navigazione tra le pagine, mentre ora, ad esempio, la navigazione può essere gestita con il DOM: Document Object Model.

Riguarda l'interazione con l'utente anche il successivo esercizio 3 (figura 3). In particolare ci occupiamo dell'uso dinamico dell'oggetto Select (la combo box di Visual Basic). Per dinamico intendiamo il suo caricamento da programma e la visualizzazione delle informazioni che contiene.

Anche in questo caso si tratta di programmazione Object, in cui si usano

Figura 5 - Esercizio 5 - Programma ASP che genera 120 pulsanti.

Il programma ASP viene eseguito dal server che invia il risultato della sua esecuzione al client. Un primo vantaggio sta nel fatto che dal browser non è possibile leggere il codice, che in tal modo risulta protetto da "sguardi indiscreti". Anche un programma ASP può servire per generare codice HTML, che viene prodotto come sequenza di stringhe che vengono inviate al client. All'interno della pagina ASP, oltre al codice ASP può essere presente normale codice HTML e normali brani di script JavaScript, che non vengono interpretati dal server, ma inviati direttamente al client che si occuperà di interpretarli. In questo esempio abbiamo prodotto, con ASP, 120 pulsanti che richiamano tutti una stessa funzione JavaScript parametrizzata in modo da individuare da quale pulsante sia stata richiamata.



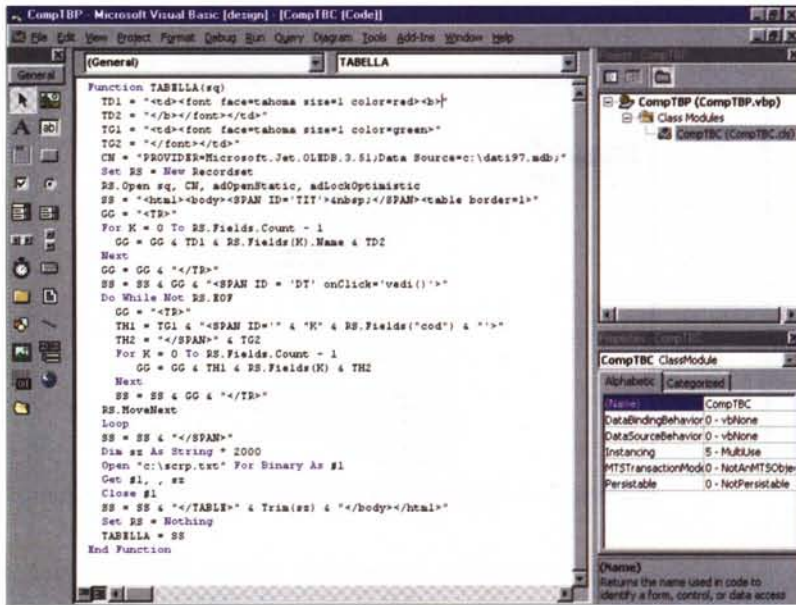


Figura 6 - Esercizio 6 - Creazione di una DLL da usare solo per Internet - Codice della DLL.

Una delle caratteristiche più stimolanti delle tecnologie Microsoft è quella di essere tutte allineate tra di loro e di essere allineate alle tecnologie di base, ad esempio COM, DCOM. L'esperimento più semplice da realizzare consiste nel costruire una libreria, compilata sotto forma di DLL, che contenga funzioni da usare in pagine ASP. E' inoltre possibile, anche se non è corretto da un punto di vista "filosofia COM", realizzare funzioni che lavorano solo per il WEB. Nel nostro caso abbiamo realizzato una semplice DLL che contiene una sola classe che a sua volta contiene una sola funzione. Questa funzione riceve un'istruzione SQL sotto forma di stringa e restituisce una "stringona" che corrisponde ad una tabella HTML, perfettamente confezionata, che visualizza i dati richiesti. Qui vediamo il codice nel suo ambiente nativo, che è il Visual Basic.

net della Microsoft, in cui c'è l'interprete del codice, o di un suo surrogato, come il PWS.

Il primo esercizio (figura 5) prevede la realizzazione di 120 pulsanti opportunamente intabellati, ciascuno con un suo nome e un suo valore. La procedura ASP che genera i pulsanti associa a ciascun pulsante un evento onClick, che corrisponde ad un codice JavaScript presente anch'esso nel file, ma che viene inviato direttamente al client, perché sarà il client che lo dovrà eseguire.

JavaScript. Per attivarlo basta impostare nel BODY ONLOAD, ma va bene leggerlo a qualsiasi altro evento, l'intervallo in millisecondi e la routine da chiamare al verificarsi dell'evento.

<BODY ONLOAD="window.tmr = setInterval("vai()",999)"

significa un intervallo di quasi un secondo e che ogni "quasi secondo" viene eseguita la function Vai().

Cosa fare all'interno della funzione Vai dipende da cosa deve fare il programma; nel nostro caso, produciamo un conto alla rovescia che, allo scadere dei quasi 20 secondi, apre un'altra finestra.

Un pizzico di ASP

Riassumiamo quanto detto fino ad ora. ASP è una tecnologia della Microsoft che permette di creare delle pagine attive, in pratica delle pagine che contengono del codice che viene interpretato dal server ed il cui risultato viene inviato al client.

Il client "si vede arrivare" codice HTML o anche codice JavaScript, ma ignora il fatto che tale codice è stato prodotto da un programma.

Per sperimentare il codice ASP occorre quindi disporre di un server Inter-

Evidentemente la funzione Mes(pp), richiamata da ciascuno dei 120 pulsanti, è parametrizzata nel senso che riceve anche l'indicazione di quale sia il mese da cui è stata chiamata. In pratica il parametro è una stringa che contiene il numero dell'anno ed il numero del mese, che vengono visualizzati in cima al documento.

Una DLL per ASP

Una delle caratteristiche più stimolanti che i programmatori trovano nelle pagine ASP consiste nella possibilità di utilizzare direttamente le DLL. E' evidente che poiché ASP lavora sul server, le DLL sono quelle che risiedono sul server.

Molte delle funzionalità standard di ASP si appoggiano su DLL di sistema oppure installate insieme al server, ad esempio la libreria ADO o la libreria Scripting Object.

Si possono ovviamente utilizzare DLL fatte in casa, come quella che vi proponiamo nell'esercizio 6, l'unico che dispone di due figure, quella che mostra l'intero codice della DLL e l'altra che mostra lo scarno codice ASP e il risultato nel browser.

La nostra libreria si chiama CompTBP (il file generato quando compileremo il sor-

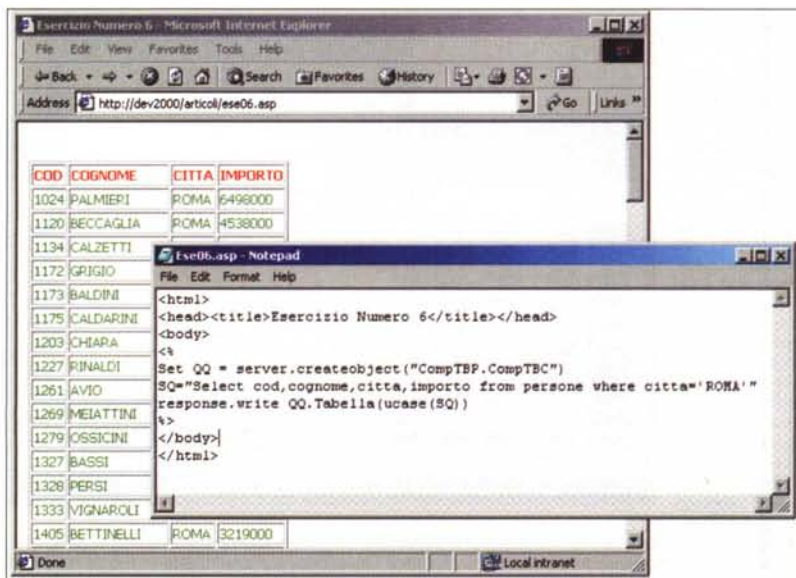


Figura 7 - Esercizio 6 - Creazione di una DLL da usare solo per Internet - Codice ASP e risultato.

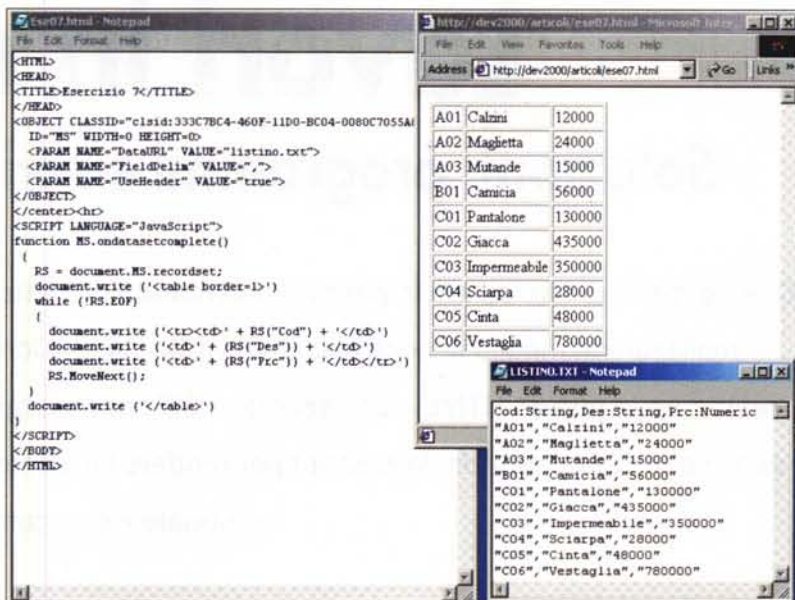
E' evidente, anche dalle nostre figure, che il fatto di trasferire la logica dell'applicazione all'interno della DLL alleggerisce il codice del programma ASP, che deve solo referenziare l'oggetto e usarne metodi e proprietà. Altri vantaggi sono la velocità di esecuzione, dovuta al fatto che il codice DLL è compilato mentre il codice ASP no, e la facilità di manutenzione che permette di aggiornare la DLL senza dover modificare il codice ASP che la utilizza.

gente si chiama CompTBP.DLL), contiene una classe che si chiama CompTBC, che contiene una funzione che abbiamo chiamato tabella. A questa funzione passiamo una stringa SQL:

```
select cod,cognome,citta,importo from persone where citta='ROMA'
```

Il compito della funzione è quello di costruire una stringa (molto lunga) che contiene un codice HTML che a sua volta contiene una tabella con le intestazioni e con i dati. Se serve del codice in più, ad esempio funzioni JavaScript, lo si può inserire leggendo da un file testuale per ag-

Figura 8 - Esercizio 7 - Lettura di un file testuale. Nel prossimo numero parleremo di pagine Internet che "leggono" file e ne visualizzano il contenuto. Per ora "accontentiamoci" di utilizzare il TDC, Tabular Data Control, componente di Internet Explorer 4.0 o superiori (risiede quindi sul client), che permette di vedere sotto forma di Recordset un file testuale opportunamente organizzato. Quello che mostriamo è un uso elementare del TDC, che è peraltro uno dei cento strumenti a disposizione del programmatore per dialogare con un database.



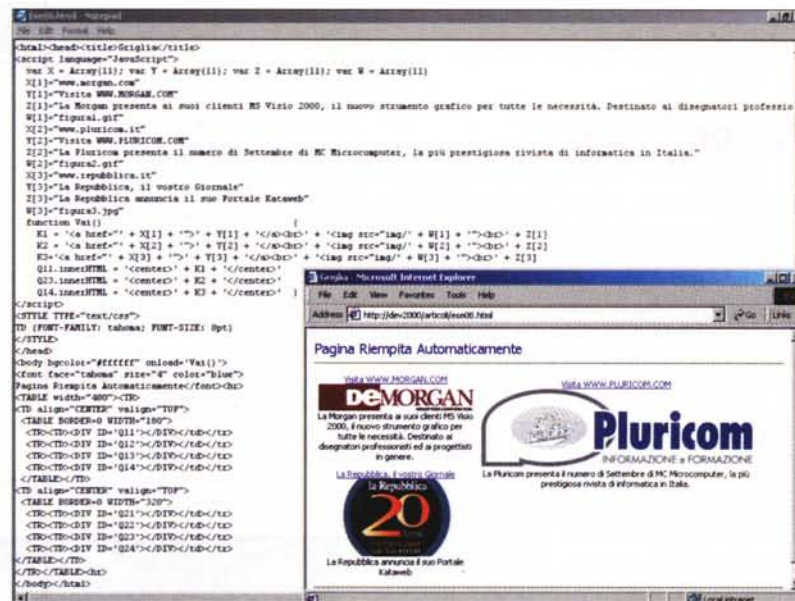
giungerlo alla stringa prodotta. Il trasferimento del codice nella DLL semplifica enormemente il lavoro del codice ASP (evidente dimostrazione nel codice di figura 7).

Questo semplice esercizio, che trovate in "scatola di montaggio" (c'è anche il database), dimostra l'interagibilità delle tecnologie Microsoft, grazie alla quale un programma per Internet può essere sviluppato in gran parte in Visual Basic.

Altro componente interessante per tale interagibilità è costituito da ADO, tecnologia universale e sofisticatissima di accesso ai dati, che dispone di oggetti, metodi, proprietà, eventi "pensati" per le applicazioni Internet.

Il settimo esercizio (figura 8) mostra l'utilizzo della tecnologia TDC, Tabular Data Control, componente (si tratta di un OCX) che viene installato assieme ad Internet Explorer e che permette di definire, caricare e gestire un Recordset. I dati, nel nostro caso si tratta di un semplice file TXT che nella prima riga riporta i nomi ed il tipo dei campi, risiedono sul server, ma vengono caricati sul client, sul quale possono essere manipolati con metodi e proprietà tipici di un Recordset. E' evidente che si tratta di una tecnologia che funziona solo sui browser MS.

Figura 9 - Esercizio 8 - Generazione automatica della pagina. Il nostro obiettivo è di quindi quello di generare una pagina automaticamente. Qui vediamo un esempio supersemplificato di quello che intendiamo per produzione automatica. In pratica realizziamo una pagina con dei segnaposto (i tag DIV nella tabella) che vengono riempiti "da programma" con dati (testi, link, figure, tag HTML, ecc.) provenienti da un database presente sul server. Per cambiare la pagina sarà sufficiente cambiare i dati nel database. Per ragioni di spazio, i nostri dati sono stati semplicemente inseriti a mano in array JavaScript. Nel prossimo numero, come appena detto, ci occuperemo della lettura dei dati e quindi avremo occasione di riparlare di tali tecniche.



che sarà alla base del nostro successivo articolo, relativo all'utilizzo dei database all'interno di pagine Internet.

Conclusioni

L'ultimo esercizio (figura 9) esemplifica il nostro obiettivo, creare una pagina che viene riempita di contenuti grazie ad una procedura che legge i dati da un database.

Nel nostro caso abbiamo una pagina (ad esempio la home di un portale) che

contiene una griglia vuota, in cui le varie celle sono ben identificate da un nome. La procedura legge i dati, nel nostro esempio li abbiamo direttamente scritti in un array JavaScript, poi li elabora ed invia il risultato dell'elaborazione alle varie celle.

L'evoluzione di tale semplice esempio è costituito da una vera e propria procedura che gestisce un database di informazioni, strutturate, testuali, grafiche, e che produce dinamicamente pagine o componenti di pagine, a richiesta dell'utente.