

## Appendice B

# Errori più comuni

Questa appendice illustra gli errori più comuni, gran parte dei quali sono molto semplici da risolvere, commessi dagli sviluppatori che lavorano con ASP.NET con una conoscenza approfondita o meno della programmazione sul server o della precedente tecnologia ASP. Questa appendice può essere utilizzata per fare riferimento a tali errori direttamente, consentendo agli sviluppatori di trovare facilmente una soluzione e di apprendere le tecniche che sono necessarie per affrontare diversi problemi.

L'appendice attuale è suddivisa in due sezioni. La prima sezione descrive gli errori che si verificano più spesso con ASP.NET e ai quali generalmente non si presta la dovuta attenzione. La seconda sezione illustra i cambiamenti che si devono osservare quando si passa da ASP ad ASP.NET, i quali includono le modifiche sintattiche tra VBScript e VB.NET.

## Problemi più comuni con ASP.NET

Molte delle particolarità di ASP.NET possono mettere in difficoltà anche gli sviluppatori più attenti. Ciascuna delle sezioni seguenti contiene un elenco dei messaggi di errore unitamente a una descrizione dettagliata del problema e a una possibile soluzione (nei messaggi di errore, le parole in corsivo possono essere sostituite con nomi specifici).

### Problemi con i form Web

**Problema:** Unexpected end of file looking for `</asp:Control>` tag

**Descrizione:** Avete dimenticato di aggiungere il tag di chiusura di un controllo del server ASP.NET.

**Soluzione:** Tutti i controlli del server devono avere i relativi tag di chiusura. Aggiungete un tag di chiusura utilizzando una delle sintassi di esempio che segue:

```
<asp:Label id="lblMessage" runat="server" />  
<asp:Label id="lblMessage" runat="server"> </asp:Label>
```

**Problema:** Literal content ("html") is not allowed within a '*control*'

**Descrizione:** Avete tentato di utilizzare un controllo del server senza tag di chiusura.

**Soluzione:** Questo problema si verifica solitamente con il controllo `DataList` o `DataGrid` quando si dimentica di includere il tag server di chiusura. Senza questo tag, ASP.NET considera tutto il codice HTML che segue come parte del controllo del server e genera un errore. Per risolvere il problema è sufficiente aggiungere il tag di chiusura per questo controllo.

**Problema:** Il controllo del server non viene visualizzato o non funziona correttamente.

**Descrizione:** ASP.NET non sta interagendo correttamente con il controllo del server: i dati del form non vengono riconosciuti, le proprietà di visualizzazione non sono visualizzati e così via.

**Soluzione:** La ragione più probabile che ha determinato questo problema è la mancanza dell'attributo `runat="server"` sul controllo del server. Aggiungete allora questo attributo al controllo del server.

**Problema:** I dati non vengono visualizzati correttamente in un controllo del server.

**Descrizione:** ASP.NET non visualizza alcun dato o non aggiorna i dati in modo corretto.

**Soluzione:** Avete dimenticato di chiamare il metodo `DataBind`.

**Problema:** An exception has been thrown by the class constructor for `System.Drawing.Internal.SystemColorTracker`

**Descrizione:** ASP.NET non è in grado di disegnare correttamente il controllo specificato (generalmente un `DataGrid`).

**Soluzione:** Riavviate l'applicazione in uno dei modi seguenti:

- chiamate `iisreset` dalla richiesta del comando;
- modificate e salvate il file `global.asax` o il file `web.config`;
- ricompilate un oggetto commerciale nella directory `\bin` dell'applicazione.

**Problema:** Le routine di gestione degli eventi non producono i risultati previsti.

**Descrizione:** Mediante i form Web e i controlli server state tentando di generare l'output desiderato ma non ottenete i risultati previsti. Per esempio, state tentando di modificare il valore contenuto in una casella di testo o in un'etichetta, ma il valore non cambia o non cambia nel modo che voi desiderate.

**Soluzione:** Non dimenticate che l'evento `Page_Load` viene sempre eseguito prima di qualsiasi gestore di eventi e che i gestori di eventi non vengono eseguiti in alcun ordine particolare. L'evento `Page_Load` potrebbe modificare o reimpostare il valore di destinazione prima che il gestore dell'evento abbia la possibilità di utilizzarlo. Una situazione di questo tipo viene illustrata dal codice seguente:

```
sub Page_Load(obj as Object, e as EventArgs)
    tbMessage.Text = "Hello world!"
end sub

sub HandleSubmit(obj as Object, e as EventArgs)
    Response.Write(tbMessage.Text)
end sub
...
...
<asp:TextBox id="tbMessage" runat="server"
    OnTextChanged="HandleSubmit"
    AutoPostBack=true />
...
```

Il form viene inviato quando il testo contenuto nella casella di testo è stato modificato. Tuttavia, il metodo `Page_Load` si esegue per primo e reimposta comunque il valore nella casella di testo su "Hello World!". Quindi il gestore dell'evento visualizza "Hello World!" anziché il nuovo testo.

Una possibile soluzione consiste nel verificare la proprietà `Page.IsPostBack` nell'evento `Page_Load` per determinare se il form sia stato appena inviato o meno, quindi il valore non dovrebbe essere reimpostato:

```
sub Page_Load(obj as Object, e as EventArgs)
    if not Page.IsPostBack then
        tbMessage.Text = "Hello World!"
    end if
end sub
...
```

Un'altra soluzione consiste semplicemente nel ristrutturare le routine.

## Altri problemi

**Problema:** `MissingMethodException: Member not found.`

**Descrizione:** Avete tentato di fare riferimento a un metodo o a una proprietà di un oggetto, generalmente quando viene passata da un metodo.

**Soluzione:** Nonostante questo problema possa avere diverse cause, la più frequente è relativa al casting del tipo di dato. Osservate per esempio il gestore di eventi seguente:

```
sub MyHandler(obj as Object, e as EventArgs)
    Response.Write(obj.Text)
end sub
```

Supponendo che un evento di un'etichetta determini l'esecuzione di questo metodo, la variabile `obj` dovrebbe rappresentare quell'etichetta e voi dovrete essere in grado di accedere alla proprietà `Text` dell'etichetta. Tuttavia, è buona norma non fare mai affidamento su quello che vi aspettate da una variabile; non partite dal presupposto che un oggetto sia ciò che dovrebbe essere. Dovreste invece eseguire direttamente il casting dell'oggetto:

```
Response.Write(CType(obj, Label).Text)
```

Spesso le variabili di tipo `Object` di ASP.NET sono associate in precedenza; ciò significa che vengono valutate quando la pagina viene compilata (non al momento della sua richiesta). Per questo motivo, ASP.NET ostacola la richiesta per la proprietà `Text` (un `Object` non ha alcuna proprietà `Text`). Al momento della richiesta della pagina, però, verrà impostato il riferimento appropriato e sarà possibile chiamare la proprietà `Text`, ma a quel punto sarà troppo tardi. Eseguite sempre il casting delle variabili in modo corretto.

**Problema:** The type *object* in Assembly *name*, Version=*version*, Culture=*culture*, PublicKeyToken=*token* is not marked as serializable.

**Descrizione:** Avete tentato di mantenere un oggetto in stato di sessione o in stato di applicazione.

**Soluzione:** Alcuni oggetti, come `DataSet`, non possono essere mantenuti automaticamente in una variabile di stato. Tentate di utilizzare il metodo `ShouldSerializeObject` dell'oggetto in questione (si veda Appendici C e D).

**Problema:** Type not defined: *type*

**Descrizione:** State tentando di dichiarare un'istanza di un oggetto noto.

**Soluzione:** Assicuratevi di aver digitato il tipo correttamente. Verificate inoltre che gli spazi dei nomi necessari siano stati importati. Non dimenticate che le classi VB.NET (come i form code behind e gli oggetti commerciali) non importano automaticamente gli spazi dei nomi, a differenza di quanto avviene per le pagine ASP.NET.

## Differenze rispetto ad ASP

Gli sviluppatori che passano da ASP e VBScript ad ASP.NET e VB.NET commettono spesso errori di sintassi, oppure incontrano altri problemi dovuti alla migrazione. I paragrafi che seguono descrivono gli errori più frequenti di questo tipo e riportano le soluzioni possibili.

## Problemi con VBScript

**Errore:** `Wend` is no longer supported; use `End While` instead.

**Descrizione:** Avete tentato di chiudere un'istruzione `while` con la parola chiave `wend`.

**Soluzione:** In VB.NET, la parola chiave `wend` non è supportata. Utilizzate `End While` al posto di `wend`.

**Errore:** The syntax `<lower bound> To <upper bound>` is no longer supported for specifying array bounds.

**Descrizione:** L'array sta tentando di dichiarare una dimensione fissa.

**Soluzione:** In VBScript potete utilizzare `dim MyArray(0 to 5)` per dichiarare un array a dimensione fissa. Questo aspetto non è però supportato in VB.NET.

**Errore:** The name '*myArray*' is not declared. Dove *myArray* è una variabile di matrice.

**Descrizione:** Non avete ancora dichiarato la variabile array.

**Soluzione:** State tentando di dichiarare un array utilizzando la parola chiave `ReDim`. In VB.NET dovete dichiarare un array utilizzando la parola chiave `Dim` prima di utilizzare `ReDim`, a differenza di quanto richiesto da VBScript.

**Errore:** A value of type '*type*' cannot be converted to '*object*'.

**Descrizione:** Generalmente ciò significa che state tentando di assegnare una proprietà a un oggetto senza specificare il nome della proprietà.

**Soluzione:** In VBScript gli oggetti supportavano proprietà predefinite. In altri termini, non era necessario specificare il nome di una proprietà per impostarla.

Per esempio, dato un controllo `Label`, la riga seguente sarebbe possibile in VBScript:

```
dim Label as Label
Label = "Hello World"
```

In tal modo, la proprietà `Text` verrebbe impostata sulla stringa `"Hello World"`. In VB.NET, al contrario, le proprietà predefinite non sono più supportate, a meno che non assumano parametri. Dovete utilizzare la riga seguente:

```
Label.Text = "Hello World"
```

**Problema:** Let and Set are no longer supported on assignment statements.

**Descrizione:** State utilizzando la parola chiave `Let` o `Set` per assegnare un elemento a un oggetto.

**Soluzione:** Queste parole chiave non sono supportate in VB.NET. Rimuovendole, il codice dovrebbe funzionare correttamente.

**Problema:** The name 'N' is not declared (anche se lo avete dichiarato).

**Descrizione:** La dichiarazione per una variabile in una parte della pagina non è accessibile da una parte diversa della pagina.

**Soluzione:** In VBScript le variabili dichiarate all'interno dei blocchi (qualsiasi insieme di istruzioni che terminano con `End`, `Next` o `Loop`) possono essere visualizzate all'esterno del blocco. Per esempio, il codice seguente scriverà 11 sul browser:

```
dim I as integer
For I = 1 To 10
    Dim N As Double
    N = N + I
Next
Response.write(N)
```

In VB.NET la variabile `N` del codice che precede ha visibilità solo nell'ambito del blocco, ovvero non è accessibile all'esterno del blocco di codice (il ciclo `for` in questo caso), pertanto questo codice genererà un errore. Dovete quindi dichiarare la variabile all'esterno del ciclo.

**Problema:** Optional parameters must always specify a default value.

**Descrizione:** Avete dichiarato una funzione con la parola chiave `optional`, specificando che un parametro è facoltativo.

**Soluzione:** In VB.NET i parametri facoltativi devono specificare un valore predefinito. Per esempio, trasformate il codice VBScript seguente:

```
sub MySub(Optional MyParam as String)

in

sub MySub(Optional MyParam as String = "HI")
```

Inoltre, la funzione `IsMissing`, utilizzata per rilevare se in VBScript è stato specificato un parametro facoltativo, non è supportata in VB.NET.

**Problema:** Argument lists in all call statements must now be enclosed in parentheses.

**Descrizione:** Avete tentato di chiamare una funzione o una subroutine senza utilizzare le parentesi. Per esempio:

```
Response.Write "Hello World"
```

**Soluzione:** In VB.NET tutte le chiamate di funzione e di subroutine devono racchiudere i parametri tra parentesi, sia quando restituiscono sia quando non restituiscono un valore. Al posto del precedente, utilizzate il codice seguente:

```
Response.Write("Hello World")
```

**Problema:** Type-declaration character & does not match declared data type type.

**Descrizione:** State tentando di utilizzare la `&` commerciale per la concatenazione di stringhe.

**Soluzione:** In VBScript era possibile concatenare stringhe utilizzando la `&` commerciale senza inserire alcuno spazio tra i nomi delle variabili. Per esempio, il codice seguente in VBScript avrebbe scritto *"HI there everyone"* nel browser:

```
dim a as string = "HI "  
dim b as string = "there "  
dim c as string = "everyone"  
Response.Write(a&b&c)
```

Questo causerà un errore in ASP.NET. Dovete invece utilizzare la stringa che segue:

```
Response.Write(a & b & c)
```

## Problemi con ASP

**Problema:** Syntax Error o Expected variable, constant, Enum, Type, or procedural declaration.

**Descrizione:** Le dichiarazioni di metodi e variabili globali non funzionano correttamente. State tentando di dichiarare un metodo o una variabile all'interno di un blocco di rendering del codice. Per esempio:

```
<%  
    dim I as Integer  
    sub HelloWorld  
        ...  
    end sub  
>%
```

**Soluzione:** In VB.NET tutte le dichiarazioni di metodi e variabili globali devono essere contenute all'interno di blocchi di dichiarazione del codice (tra i tag `<script>`). Inoltre, tutte le istruzioni di non assegnazione devono essere inserite in un altro metodo, come `Page_Load`.

**Problema:** `Request` non restituisce quanto previsto.

**Descrizione:** State utilizzando l'oggetto `Request` per restituire dati dalla richiesta HTTP: per esempio, `Request.Form` o `Request.QueryString`.

**Soluzione:** In ASP, l'oggetto `Request` restituirebbe stringhe contenenti l'intero insieme di variabili. Per esempio, dato l'URL `http://localhost/test/Test.asp?values=45&values=600` e il codice seguente:

```
Response.Write(Request.QueryString(values))
```

ASP visualizzerebbe una sola stringa, "`45, 600`", nel browser. In ASP.NET, invece l'oggetto `Request` restituisce array di stringhe anziché una stringa concatenata. Per esempio, con l'URL precedente e il codice seguente:

```
Response.Write(Request.QueryString(values)(0))
```

ASP.NET visualizzerebbe `45`, contrariamente alla riga seguente:

```
Response.Write(Request.QueryString(values)(1))
```

che visualizzerebbe `600`. Questa funzionalità è la stessa per `Request`, `Request.QueryString` e `Request.Form`.