

راهکارهایی برای بهبود طراحی صفحات ASP

در این مقاله سعی می شود که با کلاس های Vbscript آشنا

شده و نحوه استفاده از آنان در طراحی صفحات ASP مورد

بررسی قرار گیرد.

همزمان با ارائه نسخه ۵ Vbscript توسط شرکت

ماکروسافت، پیاده کنندگان نرم افزار این امکان را بدست

آوردند که بتوانند از کلاس هایی نظیر آنچه در ویژوال بیسیک

استفاده می گردد نیز استفاده نمایند. کلاس ها با رویکرد

برنامه نویسی شیء گراء ایجاد و به طراحان صفحات ASP این

امکان را خواهد داد تا قادر به استفاده از کدهای قبلی خود در

سایر صفحات ASP خود باشند. کلاس ها توسط طراحان ذیربط

بعنوان ماژول های جعبه سیاه بگونه ای طراحی می گردند که

سایر استفاده کنندگان بدون نیاز به آگاهی از جزئیات مربوط به

هر کلاس و بهره مندی از متدها و صفات (ویژگی ها) یک

کلاس از توانائی آن در برنامه های خود استفاده نمایند. در ابتدا

لازم است که یک مرور سریع به برنامه نویسی شیء گراء داشته

باشیم.

برنامه نویسی شیء گراء (OOP)، یک متدولوژی برنامه نویسی

است که در آن هر موجودیت با نام یک شیء (Object) شناخته

می شود. هر شیء شامل صفت (Properties) و متدها (

Methods) است. یک صفت ویژگی های یک شیء را تشریح می

کند در حالیکه یک متد عملیات خاصی را بر روی یک شیء انجام

می دهد. شما بعنوان یک طراح صفحات ASP، به دفعات از

اشیاء تعریف شده توسط دیگران استفاده کرده و می کنید. مثلا

ADO یا ActiveX Data Object چیزی بیشتر از یک مجموعه

از اشیاء طراحی شده برای دستیابی به بانک های اطلاعاتی

نیست. زمانیکه از ADO برای دستیابی به بانک های اطلاعاتی

استفاده می شود، استفاده کننده نیاز به دانش و آگاهی لازم

جهت شناخت نوع پروتکل مورد نیاز ارتباطی با یک بانک

اطلاعاتی نبوده و بسادگی با استفاده از متد Open مربوط به

شیء Connection با یک بانک اطلاعاتی مرتبط خواهد شد.

بمنظور شناخت کامل برنامه نویسی شیء گراء، لازم است بین

دو مفهوم یک شیء (Object) و یک نمونه از شیء (Instance

Object) تفاوت قائل شویم. یک شیء نظیر شیء Connection،

مربوط به ADO یک آیتم خیالی است. در حقیقت این شیء

بصورت فیزیکی وجود نداشته و صرفاً بمنزله یک قالب (

Template) برای ایجاد یک نمونه در نظر گرفته می شود. یک

نمونه (Instance)، یک هویت (موجودیت) فیزیکی از یک شیء

است. در مثال زیر ADODB.Connection یک شیء بوده در

حالیکه ObjConn نمونه ای از شیء Connection است.

```
Dim ObjConn
```

```
Set
```

```
ObjConn=Server.CreateObject("ADODB.Connection
```

```
("
```

ما می بایست از هر شیء نمونه ای را داشته باشیم تا بتوانیم با

استفاده از فراخوانی متدهای مربوطه و یا تغییر و استفاده از

صفات ذیربط، زمینه مناسب جهت استفاده عملی از آن را

فراهم نمائیم. در یک مثال واقعی از ارتباط شیء / نمونه، می توان

مثال معروف اتومبیل را در نظر گرفت. یک ماشین یک شیء

است. این شیء یک طرح اولیه (Template) برای تشریح

ویژگی ها و عملکردهای یک اتومبیل خواهد بود. یک "پراید"

نمونه ای از یک اتومبیل است که با بکارگیری متدها و صفات

مربوطه، می توان عملیات دلخواه و از قبل تعریف شده را بر

روی نمونه فوق انجام داد.

کپسوله نمودن پیچیدگی ها

برنامه نویسی شیء گراء می تواند بعنوان یک راهکار موفق جهت

کپسوله نمودن پیچیدگی های مرتبط با فعالیت های خاص نیز

مورد توجه جدی قرار گیرد. مثلاً فرض نمائید که بر روی وب

سایت خود نیاز داشته باشیم که یک فایل متنی را باز کرده)

Open)، تعداد خطوط آن را محاسبه (شمارش) و ماحصل کار

را در یک فایل دیگر ثبت (Log) نمائیم. در چنین حالتی در

صورتیکه از برنامه نویسی شیء گراء استفاده نکنیم، هر زمان که

قصد انجام مجموعه فعالیت های گفته شده را داشته باشیم، می

بایست عملیات زیر را مرحله به مرحله انجام داد:

• باز نمودن یک فایل متنی خاص

• شمارش تعداد خطوط موجود در فایل

• بستن فایل متنی

• باز کردن فایل مورد نظر برای ثبت (Logging File)

• ثبت تعداد خطوط شمارش شده

• بستن فایل Log

با اینکه مراحل فوق پیچیده بنظر نمی آیند ولی تصور کنید که
بخواهیم این عملیات را در چندین صفحه ASP انجام دهیم. در

چنین حالتی می توان بسادگی تمامی مراحل فوق را کپسوله و در

یک شیء خاص قرار داد. ببینیم چگونه می توان این شیء را ایجاد

کرد. فرض کنید نام این شیء را LogTextFileLines در نظر

بگیریم. شیء فوق صرفا دارای یک صفت با نام TextFilePath

بوده که مسیر فیزیکی فایل Log را مشخص خواهد کرد. شیء

فوق همچنین دارای یک متد با نام UpdateLog بوده که

مسئول شمارش تعداد خطوط در فایل مورد نظر خواهد بود.

متد فوق می تواند دارای تعریفی مشابه زیر باشد:

StrFileName(Function UpdateLog

StrFileName، مسیر و نام فایل متنی است که می بایست

تعداد خطوط آن شمارش و ثبت گردد. متد فوق یک ارزش

منطقی (Boolean)، را برمیگرداند. مقدار فوق موفقیت آمیز

بودن عملیات را نشان خواهد داد. پس از ایجاد شیء فوق، مراحل

شش گانه گفته شده بسادگی انجام خواهند شد:

Dim ObjLog

LogTextFileLines Set ObjLog = New

ObjLog.LogTextFileLine =

"C:\MyLogFiles\Linecount.log

(ObjLog.UpdateLog("C:\MyTextFiles\Test.Txt If

Then

```
Response.Write("C:\MyTextFiles\Test.txt was  
(successfully logged
```

Else

```
Response.Write("There are an error when  
to log C:\MyTextFiles\Test.Txt attempting
```

If End

همانطور که مشاهده می شود، استفاده از شی فوق تمام

پردازش روی فایل را بصورت یک جعبه سیاه در آورده است.

پس از اصول اولیه فوق، در ادامه با نحوه استفاده از کلاس های

Vbscript تشریح می گردد. کلاس های Vbscript صرفاً شامل

یک Constructor است. یک Constructor، روئینی است که

بلافاصله پس از ایجاد یک نمونه از شیء بصورت اتوماتیک اجرا

خواهد شد. متاسفانه Constructor ها در Vbscript نمی

توانند پارامتری را بعنوان ورودی اخذ نمایند. در ضمن کلاس

های Vbscript، توارث را نیز حمایت نمی کنند. در زمان ایجاد

یک کلاس، می بایست به این نکته توجه شود که شما در حال

تولید و ساخت ابزاری هستید که قرار است توسط سایر

طراحان مورد استفاده قرار گیرد. پیاده کنندگانی را که برای

سایر پیاده کنندگان کلاس هائی را ایجاد می کنند Class

Developer و پیاده کنندگانی را که از این کلاس ها استفاده می

نمایند End Developer می گویند. کلاس هائی که توسط

Vbscript ایجاد می شود را نمی توان مانند عناصر COM،

کامپایل نمود. بنابراین یک End Developer که قصد استفاده

از کلاس فوق را در صفحات ASP داشته باشد، می بایست

تعریف کلاس را به همراه کدهای ASP داشته باشد. بنابراین

روش مناسب این است که تعاریف مربوط به کلاس ها را در

یک فایل متنی قرار داد و بدین ترتیب استفاده کنندگان از

کلاس فوق با استفاده از SSI (استفاده از فایل های ضمیمه)

قادر به استفاده از کلاس های فوق در صفحات ASP خود

خواهند بود. پس از ایجاد یک کلاس، پیاده کنندگان کلاس ها

می بایست به این نکته توجه نمایند که اعمال هر گونه تغییر در

یک کلاس می بایست بگونه ای باشد که باعث بروز اشکال در

متدها و صفات قبلی تعریف شده نگردد.

parsi e-book
WWW.PARSIBOOK.4T.COM

ایجاد یک کلاس

برای ایجاد یک کلاس از دستورالعمل Class که دارای گرامری

مشابه زیر است، استفاده می شود: *کپی برداری بدون ذکر نام منبع مجاز نیست*

Class Name_Of_Class

'تعریف متدها و صفات مربوطه

Class End

رویدادهای آغاز و خاتمه

زمانیکه کلاس ها ایجاد می گردند، دو رویداد مهم در این

زمینه نقش مهمی را برعهده خواهند گرفت : رویداد

مقداردهی اولیه (Initilize) و رویداد خاتمه (Terminate).

رویداد مقداردهی اولیه زمانیکه توسط دستور New یک نمونه

یا مصداق از کلاسی ایجاد می گردد، فعال خواهد شد. مثال ۱-

دستورات زیر یک نمونه از کلاسی با نام SomeObject را ایجاد می کنند.

Dim ObjSomeObjectInstance

SomeObject ObjSomeObjectInstance = New Set

رویداد خاتمه زمانیکه به حیات نمونه ای از یک شیء خاتمه داده

می شود، فعال خواهد شد. حیات نمونه ای از کلاس یا با استفاده

از عبارتی نظیر **ObjSomeObjectInstance = Nothing** با

صراحت اعلان می گردد و یا زمانیکه از حوزه ای که شیء را

تعریف کرده ایم، خارج شویم، بدین ترتیب زمینه اجرای رویداد

فوق فراهم خواهد شد. رویدادهای دوگانه فوق را می توان در

زمان طراحی یک کلاس، تعریف نمود. رویداد مقداردهی اولیه

را برای مقداردهی اولیه صفات یک کلاس یا انجام عملیات خاص

مورد نیاز برای شروع استفاده از یک کلاس می توان استفاده

کرد. رویداد مقداردهی اولیه را Constructor می نامند.

رویداد خاتمه را می توان جهت انجام عملیات خاصی که مورد

نیاز اتمام فعالیت یک کلاس است، استفاده کرد. رویداد فوق را

Deconstructor می نامند.

مثال ۲- در مثال زیر یک کلاس به همراه رویدادهای آغاز و خاتمه

تعریف شده است.

```
Class SomeObject
```

```
( )Private Sub Class_Initialize
```

```
Sub End
```

```
( )Private Sub Class_Terminate
```

```
End Sub
```

Class End

رویداد `Class_Initialize` زمانی که یک نمونه از یک شیء ایجاد

می گردد، بصورت اتوماتیک اجراء خواهد شد. رویداد

`Class_Terminate` زمانی که نمونه تعریف شده از یک شیء از

بین می رود، بصورت اتوماتیک اجراء خواهد شد.

Variables , Member Properties , Methods , Member Function

از دیدگاه استفاده کننده از یک کلاس، یک کلاس شامل

مجموعه ای از صفات و متدها است. صفات، متغیرهای هستند

که پیاده کنندگان را قادر به تغییر و یا استفاده از آنان بمنظور

مشخص نمودن وضعیت یک کلاس می کنند. متدها توابعی

هستند که پیاده کنندگان را قادر به انجام عملیات خاصی را بر

روی یک کلاس می کنند. مثلاً شیء **ADODB.Connection**

شامل صفاتی است که وضعیت یک نمونه از شیء ایجاد شده را

تشریح خواهد کرد. (نظیر
کپی برداری بدون ذکر نام منبع مجاز نیست)

متدهائی **ConnectionString, ConnectionTimeout**،

نظیر **Open, Close** عملیات خاصی را در رابطه با نمونه شیء

ایجاد شده، انجام خواهند داد. یک کلاس می تواند دارای صفات

و یا متدهائی باشد که یک استفاده کننده از کلاس، نمی تواند

آنها را مستقیماً استفاده و فعال نماید. این متغیرها و توابع مخفی،

Member Variables و Member Function نامیده می

شوند. بنابراین در صورتیکه یک استفاده کننده قادر باشد

مستقیماً تابعی را فعال نماید به آن متد و در غیراینصورت به آن

Member Function می گویند. این وضعیت در رابطه با

صفات نیز صدق می کند. در صورتیکه یک استفاده کننده از

کلاسی قادر به صدا زدن و استفاده از یک متغیر کلاس باشد از

آن با نام **Property** نام برده می شود، در غیر اینصورت به آن

Member Variable می گویند.

عبارات **Private** , **Public**

همانطور که قبلاً گفته شد، کپسوله نمودن پیچیدگی ها، یکی از

اهداف برنامه نویسی شیء گراء است. در این حالت جزئیات

پیاده سازی از دید استفاده کنندگان یک کلاس مخفی نگه

داشته می شود. **Vbscript** این امکان را فراهم می سازد که

بتوان در زمان طراحی یک کلاس، برخی از متدها و صفات یک

شیء را از دید استفاده کنندگان آن کلاس، مخفی نگه داشت.

برای ایجاد Member Function و Variable Member از واژه

Private قبل از نام یک تابع و یا یک صفت استفاده می شود.

یک Member Function صرفاً توسط متد دیگری از کلاس

مورد نظر و یا یک Member Function دیگر صدا زده می

شود. یک Member Variable را می توان توسط یک متد و یا

یک Function Member تغییر و استفاده نمود. با استفاده از

امکانات Property Set , Property Get , Property Let نیز

امکان تغییر و استفاده از این نوع متغیرها وجود خواهد داشت.

برای ایجاد یک صفت و یا یک متد کافی است قبل از نام آنها از

واژه Public استفاده کرد. در صورتیکه قبل از نام یک متد و یا

یک صفت با صراحت نوع آنها (Public, Private) مشخص

نگردد، بصورت پیش فرض Public در نظر گرفته خواهند شد.

مثال ۳- در مثال زیر یک کلاس ایجاد و صفات و متدهای

Public و Private برای آن تعریف شده است.

Class myclass

()A_Public_Method Public Sub

در این محل می توان متدهای Private را صدا زد.

A_Private_Method

End Sub

Public A_Public_Property

()A_Private_Method Private Sub

...

End Sub

Private A_Private_Property

Class End

همانگونه که در مثال فوق مشاهده می نمائید، زمانیکه از واژه

های Public و Private به همراه متغیرها استفاده می گردد،

دیگر نیازی با استفاده از عبارت Dim نخواهد بود. در صورتیکه

با صراحت نوع یک متغیر را مشخص ننمائیم، می بایست قبل از

نام این نوع متغیرها از عبارت Dim استفاده شود. در چنین

وضعیتی این نوع متغیرها بصورت Public در نظر گرفته خواهند

شد. توصیه می گردد که همواره با صراحت نوع یک متغیر و یا

یک تابع را توسط بکارگیری واژه های Private و یا Public

مشخص نمود. در صورتیکه یک استفاده کننده از کلاس قصد

دستیابی به متدها و صفات Private نمونه شیء myclass را در

یک برنامه داشته باشد، با یک پیام خطا مواجه می گردد.

مثال ۴- در مثال زیر نحوه دستیابی به توابع و صفات Private و

Public با تاکید بر شیء myclass که در مثال ۳ تعریف شده

بود، نشان داده می شود.

Dim Objmyclass

New myclass = Set Objmyclass

Calling Objmyclass.A_Public_Method() 'Valid

Calling Objmyclass.A_Public_Oroperty = 12 'Valid

Calling Objmyclass.A_Private_Method 'Invalid

Objmyclass.A_Private_Property = 120 'Invalid

Calling

Ivalue Dim

= Invalid Calling Ivalue'

Objmyclass,A_Private_Property

استفاده از Property Get

تمامی متغیرها در Vbscript از نوع Variant بوده و اگر یک

متغیر Public را تعریف کنیم، هیچگونه کنترلی از بعد نوع

محتوائی که استفاده کننده به آن نسبت خواهد داد، وجود

نخواهد داشت. مثلاً فرض کنید که صفتی با نام

PhoneNumber در یک کلاس تعریف کرده باشیم، بدیهی

است که متغیر فوق قرار است شماره تلفن ها را در خود ذخیره

نماید. استفاده کنندگان از کلاس فوق می توانند بسادگی یک

بردار، نمونه ای از یک شیء و سایر موارد را به آن نسبت دهند.

بمنظور حل مشکلاتی از این نوع یک متغیر Private در کلاس

ایجاد و در ادامه متدی Public را جهت دستیابی و استفاده از

این متغیر تعریف می کنیم. در Vbscript تمامی صفات را می

توان بصورت Private تعریف و در صورتیکه بخواهیم استفاده

کنندگان از کلاس فوق، قادر به خواندن مقدار این صفت باشند،

با استفاده از عبارت Property Get در کلاس، امکان استفاده از

آن را می توان فراهم کرد. عبارت فوق دارای گرامری مطابق

کپی برداری بدون ذکر نام منبع مجاز نیست

narsi e-book

زیر است:

```
Get propertyName [ ( Public | Private ] Property ]
```

```
[ (arglist
```

```
to assign the value of the private property...'
```

```
propertyName
```

```
End Property
```

مثال ۵- در مثال زیر یک صفت با نام strphonebumber

تعریف شده است.

Class Information

StrPhoneNumber Private

book

End class

در ادامه اگر بخواهیم که سایر استفاده کنندگان از شیء فوق،

قادر به خواندن مقدار **StrPhoneNumber** باشند، می توانیم

از عبارت **Property Get** مشابه زیر استفاده نمود:

Class Information

StrPhoneNumber Private

() Public Property Get PhoneNumber

strPhoneNumber = PhoneNumber

End Property

class End

پس از انجام عملیات فوق، سایر استفاده کنندگان از کلاس فوق

می توانند با استفاده از کدهائی مشابه زیر به

StrPhoneNumber دستیابی داشته باشند.

Dim Objinfo

Information Set objinfo = New

& " = Response.Write " Phone number

Valid' Objinfo.PhoneNumber

& " = Response.Write " Phone number

Invalid' Objinfo.StrPhoneNumber

در صورتیکه صفت برگردانده شده توسط **Property Get**،

یک شیء باشد، می بایست از عبارت **Set** در زمان مقداردهی

استفاده نمود. مثلاً اگر **StrPhoneNumber** بجای یک رشته

شامل یک شیء دیکشنری باشد، عبارت **Property Get** مورد

نیاز بصورت زیر نوشته خواهد شد:

parsi e-book
WWW.PARSIBOOK.4T.COM

()Public Property Get PhoneNumber

Set PhoneNumber = StrPhoneNumber

Property End

عنصر arglist مربوط به عبارت **Property Get** زمانی مورد

استفاده قرار می گیرد که صفت به یک پارامتر، ایندکس نیاز

داشته باشد. مثلاً در صورتیکه یک بردار را بعنوان یک صفت

Private داشته باشیم و بخواهیم به استفاده کنندگان از کلاس

فوق امکان خواندن یک المان خاص از بردار را بدهیم ، می

بایست از کد مشابه زیر استفاده کرد:

Class Information

()lost Private

```

Private sub Class_Initialize

(ReDim Iostan(27

..."Khoozestan" = Iostan(0

End Sub

Public Property Get Ostan(Index

Iostan(Index = Ostan

End Property

Class End

```

پس از ایجاد نمونه ای از شیء Information، می توان از کدهای

زیر برای خواندن یک مقدار خاص از بردار Iostan، استفاده

کرد.

```
Dim ObjInfo
```

Information Set ObjInfo = New

(Response.write ObjInfo.Ostan(0

Nothing = Set ObjInfo

استفاده از Property Let

Property Get این امکان را برای استفاده کننده از یک کلاس

فراهم می سازد که بتوانند اقدام به بازیابی یک صفت Private

بنمایند. Let Property، این امکان را برای استفاده کننده از یک

کلاس فراهم می سازد که بتوانند اقدام به مقداردهی اولیه یک

صفت Private بنمایند. ویژگی فوق نظیر بیمه نمودن یک

صفت در مقابل هر گونه حوادث احتمالی و خرابی است. اگر

کلاسی به همراه یک صفت Public و با نام StrPhoneNumber

را ایجاد کرده باشیم، استفاده کنندگان از کلاس فوق قادر به

مقدار دهی هر نوع ارزشی به این صفت خواهند بود. بدیهی

است که طراح کلاس فوق قصد ندارد این اجازه را به استفاده

کننده از کلاس بدهند که هر مقداری را به این صفت نسبت

دهند. مقدار نسبت داده شده به صفت فوق می بایست دارای

نوع و ارزش قابل قبول باشد. StrPhoneNumber می بایست

دارای یک نوع داده با فرمت "#####" باشد. عبارت

Let Property این امکان را فراهم خواهد ساخت که یک نوع

داده به همراه فرمت مربوطه را تعریف و بررسی های لازم بر

روی آن در زمان استفاده توسط یک استفاده کننده فراهم می

گردد. گرامر عبارت فوق مشابه زیر است:

```
Let propertyName [ ( Public | Private ) Property ]
```

[arglist,) value

is of the Statements : Check to see if value...

correct datatype

private and format. If it is, assign value to the...

property

End Property

جهت مقداردهی یک مقدار به صفت StrPhoneNumber می

توان عبارت Let Property را مطابق زیر استفاده نمود:

Class Information

StrPhoneNumber Private

(Public Property Let PhoneNumber(StrPhone

StrPhone = StrPhoneNumber

End Property

Class End

عبارت **Property Let** فوق، دارای یک فرمت یا بررسی نوع

داده و ارزش مربوطه در زمان اختصاص یک مقدار توسط

استفاده کنندگان کلاس به **StrPhoneNumber** نخواهد بود.

پارامتر **StrPhoneNumber** از عبارت **Property Let** مقداری

را که توسط استفاده کننده از کلاس وارد می شود در خود

ذخیره خواهد کرد.

مثال ۶- در این مثال مقدار ۹۹۹۹۹۹۹ مقدار پارامتر ورودی بوده

که به صفت **PhoneNumber** نسبت داده می شود.

Dim ObjInfo

Information Set ObjInfo = New

`"۹۹۹۹۹۹۹" = ObjInfo.PhoneNumber`

برای بررسی فرمت کد مربوطه، می بایست تصمیم گرفت که

چه نوع فرمتی قابل قبول و چه نوع قالبی غیرپذیرش خواهد

بود. مثلاً در رابطه با مثال فوق فرمت یک تلفن می تواند

بصورت ##### باشد و اگر استفاده کننده از این کلاس

قالبی دیگر را وارد کند، قابل قبول نخواهد بود.

مثال ۷- در این مثال یک کلاس جدید Information تعریف

شده است.

Class Information

StrPhoneNumber Private

(Public Property Let PhoneNumber(StrPhone

```

Isobject(StrPhone) Then If

Err.Raise vbobjectError + 1000 , "Information

.Calss","Invalid format for PhoneNumber

Exit Property

End if

ObjRegExp Dim

Set ObjRegExp = New regexp

"${ObjRegExp.Pattern = "^\\d{7

ObjRegExp.Test(StrPhone) Then If

StrPhoneNumber = StrPhone

Else

vbobjectError + 1000 , "Information Err.Raise

.PhoneNumber Calss","Invalid format for

End if

Set ObjRegExp = Nothing

```

WWW.PAKISTANIPEDIA.COM

End Property

()Property Get PhoneNumber Public

PhoneNumber = StrPhoneNumber

Property End

End Class

اگر استفاده کننده از کلاس، کدی مطابق زیر را اجراء نماید:

Dim ObjInfo

Information Set ObjInfo = New

ObjInfo.PhoneNumber = " This is an Invalid Phone

" Number

یک پیام خطا مبنی بر عدم رعایت فرمت قابل قبول، را دریافت

خواهد کرد. عبارت **Property Let** همچنین دارای یک **arglist**

اختیاری نیز هست. این آرگومان می بایست با arglist عبارت

متناظر در Property Get یکسان باشد (در صورت وجود)

استفاده از Property Set کلاس ها می توانند دارای صفاتی

باشند که خود بمنزله شی باشند. در چنین حالتی می بایست

مراقبت های ویژه ای را در زمان اختصاص (برگرداندن) یک

شی از طریق یک Property Get انجام داد.

در Vbscript با استفاده از عبارت Property Set امکان

مقداردهی یک نمونه از یک شی به یک صفت وجود خواهد

داشت. گرامر عبارت فوق بشکل زیر است:

```
Set propertyName [ ( Public | Private ) Property ]
```

```
[arglist,] reference
```

then use , Perform any needed checks here '

set ObjPrivateProperty = ProperName '

Property End

قالب Property Get تقریبا مشابه Property Let است. تنها

تفاوت عملیاتی بین آنها در این است که عبارت Property Let

مقادیری را که به یک صفت نسبت می دهد از نوع شی نخواهد

بود در حالیکه Property set نمونه ای از یک شی را که به یک

صفت Private نسبت می دهد. مثلا فرض کنید دارای کلاسی

باشیم که دارای یک صفت اختصاصی با نام ObjConn باشد،

تعریف کلاس، عبارات Property Set و Property Get مطابق

زیر خواهد بود.

parsi e-book
WWW.PARSIBOOK.4T.COM

```

Class MyConnectionClass

Private ObjConn

( )Connection Public Property Get

Set Connection = ObjConn

End Property

(Connection( Objconnection Public Property Set

Set ObjConn = ObjConnection

End Property

Class End

```

استفاده کنندگان از کلاس با استفاده Property Set مطابق
زیر از کلاس فوق استفاده خواهند کرد.

```

ObjMyRecordSet ، Dim ObjMyClass

Set ObjMyclass = New MyConnectionClass

= Set ObjConnection

```

```
(Server.Createobject("ADODB.Connection
```

```
objconnection = Set ObjMyclass.connection
```

پارسا در نام منبع مجاز نیست
parsiebook



parsiebook
WWW.PARSIBOOK.4T.COM