

Thors hjemmeside

En enkel innføring i ASP

Til å begynne med ble CGI – "Common Gateway Interface" – brukt til å kommunisere mellom et HTML skjema og webserveren. I **FORM** knaggen ble navnet på CGI applikasjonen angitt, i attributten **ACTION=**. For hver gang noen sendte informasjon til webserveren startet webserveren en ny utgave av CGI applikasjonen. Dette kan være brukbart for mindre websteder, men når trafikken øker blir belastningen på serveren fort stor. Også JavaScript blir i noen grad brukt, men siden JavaScript tolkes på klientens webleser, og Netscape og Microsoft sine weblesere er noe ulike, må man skrive JavaScript koden slik at den håndterer begge webleserne. Dette medfører mye ekstra arbeid.

Microsofts løsning er **ASP** – "Active Server Pages".

Active Server Pages

Med ASP skriver man koden i et skriptspråk, som regel VBScript eller JavaScript. Skriptene blir tolket av webserveren, og standard HTML koder blir returnert til klientens webleser. Dermed unngår man problemet med at Microsoft og Netscape sine weblesere er ulike. Selve HTML dokumentet består av standard HTML knagger og skriptkommandoer. Dette gir mulighet til å lage dynamiske websider.

Skriptkommandoene plasseres mellom tegnene `<%` og `%>`.

Her er et enkelt eksempel:

```
<% language="VBScript" %>
<HTML>
<HEAD>
<TITLE>Min første ASP side</TITLE>
</HEAD>
<BODY>
<H2>ASP test 1</H3>
<% Response.Write("Hei. Dette er min første ASP side.") %>
</BODY>
</HTML>
```

Den første linjen angir at vi bruker skriptspråket VBScript. Linjen

```
<% Response.Write("Hei. Dette er min første ASP side.") %>
```

forteller webserveren at den skal bruke *Response* objektet sin metoden *Write* til å skrive teksten til klientens webleser. Dersom klienten ser på kildeteksten til websiden vil vedkommende ikke se ASP koden, kun teksten som var plassert mellom de doble hermetegnene. På denne måten kan vi skjule koden for utenforstående.

Hva trenger du for å prøve ASP?

- Først installerer du siste versjon av Internet Explorer
- Så må du installere en webserver som støtter ASP (se nedenfor)
- Til slutt lager du HTML dokumentene med ASP skript og lagrer dem med filnavn som slutter på **.asp**

Webservere

De fleste som benytter ASP bruker Microsoft sin IIS webserver. Den følger med Windows 2000 Server og senere versjoner. Nå har vi imidlertid ikke bruk for 2000 Server e.l. for å teste ASP sidene våre.

Windows 98

Dersom du har Windows 98 har du en "mini utgave" av IIS som Microsoft kaller "Personal Web Server" – PWS. PWS kan ikke håndtere mange brukere, men PWS støtter både VBScript og Microsoft sin versjon av JavaScript – JScript. På Windows 98 installasjons CD leter du etter katalogen *add-ons*, der bør du finne en *pws* katalog. Her kjører du *install.exe*.

Også Windows 2000 Professional og Windows XP Professional installasjons CD inneholder en utgave av PWS.

Windows 95 / Windows NT

Dersom du bruker Windows 95 kan du laste ned en versjon av PWS fra Microsoft sitt [nettsted](#). Plasseringen kan variere så det er best å søke etter "NT 4 Option Pack" (uten hermetegnene " – "). Pass på at du laster ned rett versjon! Foruten versjoner for ulike prosessorer finnes det en versjon for NT 4 Server, en annen for NT 4 Workstation. Og til sist en versjon for Windows 95, denne ser ut til også å fungere med Windows 98. Dersom du har Windows ME vil antagelig denne versjonen fungere, men noen har fått trøbbel med registerfilene (ta sikkerhetskopi først).

I tillegg finnes det flere webservere (gratis og kommersielle) som håndterer ASP. Men de fleste bruker andre skriptspråk enn VBScript. De eksemplene som vises her bruker skriptspråket VBScript.

Vår første test

Etter at du har installert og startet PWS vil du finne en katalog – *inetpub* – på den stasjonen der du installerte PWS. Under katalogen *inetpub* vil du finne katalogen *wwwroot*. Bruk Notisblokk og skriv inn eksemplet som ble vist over. Lagre det som ASPTEST1.ASP i katalogen *wwwroot*.

Start webleseren og skriv inn *http://localhost/ASPTTEST1.ASP* i adressefeltet og trykk på Enter tasten.

Vis kildeteksten i webleseren (Vis, Kilde i Internet Explorer), er ASP koden synlig?

ASP objekter

Microsoft har definert ASP med fem innebygde objekter. Vi har allerede vist bruken til ett av dem – *Response*, der vi brukte metoden *Write* for å skrive tekst til klientens webleser.

De fem objektene er Session, Application, Request, Response og Server. Disse objektene og metodene til objektene er tilgjengelige i VBScript og JavaScript.

Session

Session objektet brukes til å lagre informasjon om en gitt bruker sesjon. Du kan lagre hva en bruker foretrekker i et "Session" objekt, slik:

```
<% Session("Bruker_Pref")=Request.Form("Preferanse") %>
```

og så bruke denne informasjonen senere i ASP applikasjonen.

Et "Session" objekt er unikt for hver bruker, og varer kun 20 minutter, dersom brukeren ikke bruker objektet.

Application

Et "Application" objekt derimot er felles for alle brukere, og varer så lenge applikasjonen kjører.

Request

"Request" objektet brukes til å hente informasjon fra HTML skjema ("Form"), såkalte "cookies", standard CGI variabler o.l.

F.eks. dersom du ønsker å vise navnet på serveren:

```
<% =Request.ServerVariables("SERVER_URL") %>
```

Response

"Response" objektet brukes til å:

- skrive ut informasjon til klienten (inkludert "cookie" verdier)
- sette en tidsbegrensing på hvor lenge et HTML dokument skal lagres i webleserens buffer

F.eks. brukes vi metoden *Response.Write* til å skrive ut informasjon som skal sendes til klientens webleser. *Response.Expires* brukes til å angi i hvor mange minutter HTML dokumentet skal lagres i webleserens lokale buffer før den skal be om en ny versjon av siden.

Server

"Server" objektets viktigste oppgave er å opprette en instans av ActiveX komponenter, f.eks.:
Set minDB=Server.CreateObjekt("ADODB.Connection")

ActiveX komponenter er nøkkelen til å lage nyttige ASP applikasjoner. Eksempelet over viser hvordan man kan få adgang til en database på den maskinen som kjører webserveren.

Flere ASP eksempler

La oss bygge litt videre på den første ASP fila (ASPTEST1.ASP). Her bruker vi fremdeles skriptpråket VBScript, men vi introduserer bruken av variabler. En variabel er et område i minnet som vi får adgang til ved et navn (variabelnavnet). Her kan vi lagre informasjon. Endre ASPTEST1.ASP slik som vist under og lagre den som ASPTEST2.ASP.

```
<% language="VBScript" %>
<% ' Deklarer en variabel, og gi den en verdi.
Dim sBeskjed
sBeskjed = "Hei. Dette er andre ASP eks. Her brukes en variabel."
%>
<HTML>
<HEAD>
<TITLE>Min andre ASP side</TITLE>
```

Thors innføring i ASP

```
</HEAD>
<BODY>
  <H2>ASP test 2</H3>
  <% Response.Write(sBeskjed) %>
</BODY>
</HTML>
```

Første linje er som før, vi angir skriptspråket til VBScript. Andre linje er en kommentar som skripttolken vil overse. I tredje linje deklarerer vi en variabel med det reserverte ordet *Dim*, og vi gir variabelen navnet *sBeskjed*. De som har erfaring med Visual Basic fra før legger merke til at vi ikke angir datatypen til variabelen. Det skyldes at VBScript automatisk tilordner en datatype når vi i neste linje gir variabelen en verdi. (VBScript bruker datatypen Variant, som kan inneholde variabelverdier av alle datatyper.)

I kroppen (BODY) av dokumentet vil linja som skriver ut verdien være litt forskjellig fra første eksempel:

```
<% Response.Write(sBeskjed) %>
```

Vi skriver ikke navnet på variabelen i hermetegn. Dersom vi hadde gjort det ville vi ikke fått skrevet ut beskjedens innhold, men ordet "sBeskjed"!

VBScript har en egen funksjon – *TypeName()* som kan brukes til å finne ut hvilken datatype en variabel har. Endre foregående eksempel slik, og lagre som ASPTEST3.ASP:

```
<% language="VBScript" %>
<% ' Deklarer variabler, og gi dem en verdi.
  Dim sBeskjed, iTall
  sBeskjed = "Hei. Dette er andre ASP eks. Her brukes en variabel"
  iTall = 2
%>
<HTML>
  <HEAD>
    <TITLE>Min tredje ASP side</TITLE>
  </HEAD>
  <BODY>
    <H2>ASP test 3</H3>
    <% Response.Write(sBeskjed) %><BR>
    <% Response.Write("Variabelen sBeskjed er en: " &TypeName(sBeskjed)) %><BR>
    <% Response.Write("Variabelen iTall er en: " &TypeName(iTall)) %>
  </BODY>
</HTML>
```

Hvilke datatyper angis for de to variablene?

Valg

Når vi skal skrive programmer har vi behov for å kunne gjøre valg. Dersom en betingelse er oppfylt gjør vi en ting, dersom betingelsen ikke er oppfylt gjør vi noe annet. La oss se på et eksempel som bruker VBScript funksjonen *Time()* til å skrive enten "God formiddag" dersom klokka er mindre enn 12, ellers skrives "God ettermiddag":

```
<% language="VBScript" %>
<HTML>
  <HEAD>
    <TITLE>ASP If test</TITLE>
  </HEAD>
  <BODY>
```

Thors innføring i ASP

```
<H1>Demo av ASP If test</H1>
<% Response.Write("Klokka er: " & Time()) %><br>
<% If Time() >= #12:00:00# Then
    Response.Write("God ettermiddag")
Else
    Response.Write("God formiddag")
End If%> <BR>
</BODY>
</HTML>
```

Legg merke til at klokkeslettet 12:00:00 er plassert imellom # tegnene, dette gjør at VBScript oppfatter 12:00:00 som et klokkeslett.

Som i Visual Basic må vi avslutte *If* setningen med *End If*.

I tillegg til *If* setningen har også VBScript *Select Case...* setningen:

```
<% language="VBScript" %>
<% Dim sVareType %>
<HTML>
<HEAD>
<TITLE>ASP Select Case test</TITLE>
</HEAD>
<BODY>
<H1>Demo av ASP Select Case test</H1>
<% sVareType = "Mat" %> <BR>
<% Response.Write "Varetype: " &sVareType %><BR>
<% Select Case sVareType
    Case "Mat"
        Response.Write "Halv MVA på matvarer."
        ' Her kan du ha flere CASE.. setninger
    Case Else
        Response.Write "24% MVA"
    End Select %> <BR>
</BODY>
</HTML>
```

Endre verdien til *sVareType* til noe annet enn "Mat", hva vises da som MVA?

Også *Select Case* setningen må avsluttes, med *End Select*. Det bør alltid ha med *Case Else* slik at rutinene håndterer utypiske verdier.

Dersom du vil teste for flere verdier i en *Case* setning, plasseres verdiene etter *Case* skilt med komma:

```
<% Dim Alder
Alder = 12
Select Case Alder
    Case 1, 2, 3, 4, 5:
        Response.Write("Under skolepliktig alder.")
    Case 6,7,8,9,10,11,12,13,14,15,16
        Response.Write("I grunnskolen.")
    Case 17
        Response.Write("Snart voksen.")
    Case Else
        Response.Write("Voksen.")
End Select %>
```

Løkker

For løkken brukes nå vi på forhånd vet hvor mange ganger en operasjon skal gjentas. Vi bruker en tellevariabel og vi angir start og sluttverdi:

```
<%
  Dim i
  For i = 1 To 5
    Response.Write "Linje nr. " &i &"<BR>"
  Next
%>
```

La oss se på et annet eksempel der vi bruker *Request* objektet sin samling *ServerVariables*. *ServerVariables* er en samling av egenskaper, som bl.a. forteller navn og IP adresse på serveren og mye annet. En slik samling ("Collection") kan vi få adgang til med operatoren *Each*:

```
<% language="VBScript" %>
<% Dim Tmp, ServerVar %>
<HTML>
  <HEAD>
  <TITLE>ASP For løkke</TITLE>
  </HEAD>
  <BODY>
  <H1>Demo av ASP For løkke</H1>
  <B>Server variables:</B><br>
  <% For Each ServerVar In Request.ServerVariables
    Tmp = Request.ServerVariables(ServerVar)
    Response.Write ServerVar &" = " &Tmp &"<BR>"
  Next %> <BR>
  </BODY>
</HTML>
```

Vi starter med å deklarere to variabler (Tmp og ServerVar). Uttrykket *For Each ServerVar ...* vil hente ut navnet til hver eneste verdi i samlingen *ServerVariables*. I neste linje brukes den verdi som variabelen *ServerVar* fikk til å hente ut verdien til denne egenskapen.

VBScript har flere løkkestrukturer som kan brukes når vi ikke på forhånd vet hvor mange gjentakelser som behøves. F.eks. *While ... Wend*:

```
<% Dim i
  i = 1
  While i <= 5
    i = i + 1
  Wend %>
```

"Cookies"

"Cookies" er små tekstfiler som webserveren lagrer på klientens datamaskin. En slik "cookie" kan inneholde opplysninger om den aktuelle brukeren, for eksempel slik at nettstedet kan skreddersy visningen for den aktuelle klienten.

En "cookie" opprettes med kommandoen:

Thors innføring i ASP

```
Response.Cookies("CookieNavn")=1
```

Denne kommandoen må plasseres foran knaggen <HTML> i .ASP fila! Det gjelder alle *Response.Cookies* kommandoer.

Det er mulig å lagre flere opplysninger i samme "cookie" ved å føye til elementene (nøklerne) slik:

```
Response.Cookies("CookieNavn") ("Opprettet")=Now()  
Response.Cookies("CookieNavn") ("AntallBesok")=1
```

Her vil "cookie" ha navnet *CookieNavn*, med nøklene *Opprettet* og *AntallBesok*.

Vi kan angi levetiden til en "cookie" med *Expires*. Dersom vi vil at en "cookie" skal vare et år:

```
Response.Cookies("CookieNavn").Expires=Date() + 365
```

Vi kan bruke samme teknikk til å slette en "cookie":

```
Response.Cookies("CookieNavn").Expires=Now()
```

Vi setter *Expires* til dagaens dato og klokkeslett, dermed slettes den.

Du kan hente informasjon fra en "cookie" med kommandoen

```
Request.Cookies("CookieNavn")
```

Denne kommandoen kan plasseres hvor som helst i .ASP fila. Her er et eksempel, "cookie" får her navnet *Cookie1*:

```
<% Language="VBScript" %>  
<%  
Dim sCookie, AntBesok           ' Deklarer variabler  
Dim sClientIP, sOpprettet       ' Informasjon til cookie  
Dim CookieFinnes  
sClientIP = Request.ServerVariables("REMOTE_ADDR")  
sOpprettet = Now()  
AntBesok = 0  
' --- Finnes Cookie fra før?  
sCookie=Request.Cookies("Cookie1")  
If sCookie = "" Then  
' -- Cookie finnes ikke, gi mld. og opprett den.  
AntBesok = 1  
CookieFinnes = False  
Else  
AntBesok = Request.Cookies("Cookie1") ("AntallBesok")  
AntBesok = AntBesok + 1  
Response.Cookies("Cookie1") ("AntallBesok")=AntBesok  
CookieFinnes = True  
End If  
'Response.Cookies("Cookie1").Expires=Now() ' Dersom cookie skal slettes  
>%  
<HTML>  
<HEAD>  
<TITLE>ASP Cookie test 1</TITLE>  
<!-------  
Demonstrere hvordan man kan lage en "Cookie"  
----->  
</HEAD>
```

Thors innføring i ASP

```
<BODY>
<H2>ASP Cookie test I</H2>
<P>
  Dette ASP eksemplet viser hvordan man oppretter en "Cookie".
</P>
<%
  If Not CookieFinnes Then
    Response.Write("Finner ikke 'Cookie'! Oppretter ny.<BR>")
    Response.Write("Dette er ditt første besøk til denne siden.")
  Else
    Response.Write("Cookie funnet.<BR>")
    Response.Write "Cookie først opprettet: " &_
    Request.Cookies("Cookie1")("Opprettet") &"<BR>"
    Response.Write "Du har besøkt denne siden: " &AntBesok - 1 &" ganger."
  End If
%> <BR>
</BODY>
</HTML>
```

Hente informasjon fra et HTML skjema

Normalt brukes et HTML skjema ("Form") for å gi brukeren mulighet til å sende informasjon til webserveren. Vi kan bruke ASP til å ta imot og behandle denne informasjonen. Vi har da behov for to websider (filer); en fil for siden som inneholder selve skjemaet (kan godt være en helt normal HTML side), og en fil som inneholder den ASP koden som behandler informasjonen fra skjemaet.

La oss se på et enkelt eksempel der bruker taster inn navn og alder i HTML skjemaet, denne informasjonen sendes til webserveren når brukeren trykker på skjemaets "Submit" knapp. Det eneste ASP delen gjør er å returnere informasjonen med en takk til klienten. Lag først denne HTML fila (FORM1.HTM):

```
<HTML>
<HEAD>
  <META NAME="AUTHOR" Content="TH">
  <TITLE>ASP Form demo I</TITLE>
</HEAD>
<BODY>
<H2>ASP Form demo I</H2>
<FORM ACTION="aspform1.asp" METHOD=POST Name="Form1">
  Tast inn navnet ditt:
  <INPUT NAME="Navn" TYPE="Text" Size="20"><BR>
  Tast inn alderen din:
  <INPUT NAME="Alder" TYPE="Text" Size="3"><BR>
  <INPUT NAME="Submit" TYPE=Submit Value="Send info">
</FORM>
</BODY>
</HTML>
```

Deretter lager du fila ASPFORM1.ASP:

```
<% Language="VBScript" %>
<% Dim sNavn, iAlder ' Deklarer variabler. %>
<HTML>
<HEAD>
  <META NAME="AUTHOR" Content="TH">
  <TITLE>ASP Form1</TITLE>
```

Thors innføring i ASP

```
</HEAD>
<BODY>
<!-- Her henter vi inn info. fra HTML skjemaet FORM1.HTM -->
<%
  sNavn = Request.Form("Navn")    ' Hent inn navn og alder fra skjema
  iAlder = Request.Form("Alder")
  Response.Write "<H2>Informasjon mottatt</H2>"
  Response.Write "Hei " &sNavn &"<BR>"
  Response.Write "Takk for at du tok deg tid til å svare."
%>
</BODY>
</HTML>
```

Filen Global.asa

I ASP finnes filen GLOBAL.ASA (plassert i rotkatalogen til webserveren, *wwwroot* for en standard installasjon av PWS el. IIS). Global.asa fila brukes til å lagre skript som skal være tilgjengelige for alle ASP sidene, og til å deklare sesjons ("session") og applikasjons ("application") objekter. Merk at i Global.asa så brukes ikke knaggene <% og %>, men det som skrives må stå mellom knaggene <SCRIPT Language=...> og </SCRIPT>. Her kan vi fortelle hva sesjons og applikasjons objektene skal gjøre når en applikasjon el. sesjon startes el. avsluttes. Hendelsene ("event") OnStart og OnEnd er definert for disse to objektene.

Dersom vi ønsker å lage en teller som teller opp hvor mange samtidige besøkende en side har, kan vi lage denne i Global.asa filen:

```
<SCRIPT language="VBScript" runat="server">
' ----- GLOBAL.ASA for IIS / PWS -----
'
' -----
' "Besokende" er en variabel som forteller hvor mange
' samtidige brukere webstedet har.
' Vi initialiserer telleren "Besokende" i Start hendelsen
' til Applikasjons objektet:
'
Sub Application_OnStart
  Application("Besokende") = 0
End Sub

' -----
' Hver gang en ny Sesjon starter, økes telleren Besokende
'
Sub Session_OnStart
  Session("start") = Now()
  ' ---- Dette er IIS / PWS sin måte for å oppdatere ant. besøkende:
  Application.Lock          ' Lås Appl. mens teller oppdateres
  Application("Besokende") = Application("Besokende") + 1
  Application.Unlock        ' Frigi Appl. objektet etterpå.
End Sub

Sub Session_OnEnd
  Application.Lock          ' Lås Appl. mens teller oppdateres.
  Application("Besokende") = Application("Besokende") - 1
  Application.Unlock        ' Frigi Appl. objektet etterpå.
End Sub
</SCRIPT>
```

Legg merke til at før Application("Besokende") oppdateres, så låses applikasjonsobjektet. Dette betyr at ingen andre får tilgang til dette objektet før det blir frigitt igjen (med "UnLock"). Dermed unngår vi at to sesjoner samtidig prøver å oppdatere telleren vår.

Bruke ASP og ADO til å vise postene i en Access database

ASP og ADO – ActiveX Data Objects – kan brukes til å vise informasjon fra en database (f.eks. en Access database) som ligger på webserveren. La oss se litt på teorien bak.

Vi må da først opprette et koblingsobjekt ("Connection object"), som lager en kobling mellom datafila på web-serverens harddisk og ASP. Det er flere måter å opprette en slik kobling på. Man kan benytte et "ODBC Data Source Name – DNS", men "JET OLEDB" er raskere og gir flere muligheter. Vi benytter den siste metoden i våre eksempler.

For å opprette en kobling til en databasefil (vi skal bruke en Access 97 .mdb fil), må vi først opprette koblingsobjektet, og datamaskinen må selvsagt ha ADODB kontrollen installert (installer Microsoft Data Access Components, Personal Web Server el. Access).

Koblingsobjektet opprettes med et kall til Server objektet, slik:

```
Dim Kobling
Set Kobling = Server.CreateObject("ADODB.Connection")
```

I tillegg må vi angi hvilken kontroll som tilbyr datafila (Kobling.Provider), og navnet på databasefila, og hvor den befinner seg (stasjon:\bane). Dersom du flytter fila må du selv endre stasjon: og bane, det er mye enklere å la Server objektet sin metode "MapPath" håndtere dette:

```
Dim sBane
sBane = Server.MapPath("test.mdb") ' Datafila vår heter "test.mdb"
```

Etter at Koblingsobjektet er opprettet, og vi har fått stasjon:\bane og databasefilnavn tilordnet variabelen sBane, kan vi gi koblingsobjektet de nødvendige opplysninger:

```
Kobling.Provider = "Microsoft.Jet.OLEDB.4.0"
Kobling.ConnectionString = "Data Source=" &sBane
```

Strengen ...Jet.OLEDB.4.0 brukes for versjon 4.0 av ADO kontrollen, eldre versjoner kan kreve strengen ..Jet.OLEDB.3.5

Det eneste som gjenstår nå er å åpne koblingen:

```
Kobling.Open
```

For å få adgang til postene i datafila brukes et postobjekt ("RecordSet"). Vi oppretter det slik:

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

Når vi åpner postobjektet kan vi angi som parameter hvilken tabell vi vil åpne (f. eks. Navneliste), hvilke felter som skal være med etc. Se eksemplet nedenfor.

Eksempel I. Vise alle postene i en Access datafil

Først starter du Access (97 el. senere) og oppretter en ny databasefil – TEST.MDB. Databasen skal inneholde kun en tabell – Navneliste. Opprett disse feltene:

Id Nøkkelfelt som automatisk oppdateres av Access

Thors innføring i ASP

Navn Tekst (std. lengde på 50 tegn)
Alder Tall (Byte)

Legg inn fire – fem poster i fila. Kopier databasefilen (Test.mdb) til den katalogen på webserveren (IIS el. PWS el. annen webserver som støtter ASP) der du har .ASP filene dine. I dette eksempelet skal vi vise alle postene i databasefila i en HTML tabell. Dette går greit så lenge datafila bare har noen få poster.

Bruk en teksteditor til å lage denne fila (db1.asp):

```
<% Language="VBScript" %>
<HTML>
<HEAD>
<TITLE>ASP Access DataBase test 1</TITLE>
<!-------
Demonstrere hvordan man kan lese feltene i en Access (97) database.
Databasen inneholder kun en tabell - Navneliste.
Tabellen har et nøkkelfelt: Id, og to datafelt:
Navn (Tekst, std.)
Alder (Tall, byte)
----->
</HEAD>
<BODY>
<H2>Access DataBase test I</H2>
<P>
Dette ASP eksemplet viser hvordan man kan lese feltene i en
Access database. Kun en tabell - Navneliste - med et nøkkelfelt
kalt <EM>Id</EM>, og to datafelte: <EM>Navn</EM> og
<EM>Alder</EM>. Innhold:
</P>
<%
Dim Kobling, RecSet ' Deklarer variabler, Koblingsobjektet brukes
                    ' til å koble oss opp mot .mdb, RecSet tar feltene
Dim sBane           ' For stasjon og bane til datafil
' Først brukes Koblings objektet til å åpne forbindelse til ADO:
Set Kobling = Server.CreateObject("ADODB.Connection")
sBane = Server.MapPath("test.mdb")           ' Server gir oss bane til datafil
' Angi hvem som tilbyr koblingen. "... OLEDB.4.0" for
' Office 2K, bruk "...OLEDB.3.5" for eldre ver.
Kobling.Provider = "Microsoft.Jet.OLEDB.4.0"
' Angi stasjon:\bane\databasefilnavn:
Kobling.ConnectionString = "Data Source=" &sBane
Kobling.Open           ' Åpne
' Opprett et postobjekt (Recordset)
Set RecSet = Server.CreateObject("ADODB.Recordset")
' Velg alle poster (*) fra tabellen 'Navneliste'
RecSet.Open "SELECT * FROM Navneliste ORDER BY Id", Kobling, 1, 1
' Lag en HTML tabell der informasjonen vises (OK for lite ant. poster)
Response.Write "<TABLE COLS=2 BORDER=1 CELLPADDING=2 CELLSPACING=0>"
Response.Write "<TR><TD COLSPAN=2>Tabellnavn: Navneliste</TD></TR>"
Response.Write "<TR><TD><B>Navn</B></TD><TD><B>Alder</B></TD></TR>"
' Gå gjennom alle postene inntil slutt (EOF)
Do While Not RecSet.EOF
  ' Hent ut data fra feltene 'Navn' og 'Alder'
  Response.Write "<TR><TD>" &RecSet("Navn") &"</TD>" &_
                "<TD>" &RecSet("Alder") &"</TD></TR>"
  RecSet.MoveNext           ' Gå til neste post
Loop
Response.Write "</TABLE>"           ' Avslutt tabell def.
```

Thors innføring i ASP

```
RecSet.Close           ' Lukk postobjektet
Kobling.Close          ' Lukk koblingsobjektet
Set RecSet = Nothing
Set Kobling = Nothing
%>
</BODY>
</HTML>
```

Her bruker vi en Do While løkke (Do While Not RecSet.EOF) til å gå gjennom alle posten i datafila (inntil vi når siste post – EOF). Vi bruker Response.Write setninger til å bygge opp en tabell, der hver post vises i en av tabellens rader, første kolonne viser innholdet i feltet 'Navn', andre kolonne viser innholdet i feltet 'Alder'. Metoden *MoveNext* brukes til å flytte postpekeren til neste post i datafila.

Til slutt lukkes postobjektet og koblingsobjektet (med metoden *Close*).

Eksempel II. Legg poster til en Access fil

La oss fortsette med datafila *Test.mdb*, og vise hvordan vi fra et HTML skjema kan legge til nye poster bakerst i datafila. Her bruker vi to filer, en HTML fil som inneholder skjemaet (*Leggtil.htm*), og en ASP fil (*leggtil.asp*) som er den fil som kalles fra ACTION=... i HTML skjemaet.

I HTML skjemaet kan bruker skrive inn informasjon i inndatafeltene for 'Navn' og 'Alder'. Når så bruker trykker på "Send" knappen kalles ASP fila *leggtil.asp* og informasjonen sendes til ASP fila.

Lag *Leggtil.htm* slik:

```
<HTML>
<HEAD>
<TITLE>ASP Access DataBase legg til post</TITLE>
</HEAD>
<BODY>
<H2> ASP Access DataBase legg til post </H2>
<P>
Vise hvordan man legger til en ny post bakerst i en Access database.
</P>
<Form Action="leggtil.asp" Method="POST">
  Navn : <Input Type="Text" Name="Navn" Value="" Size="20"> <BR>
  Alder: <Input Type="Text" Name="Alder" Value="0" Size="3"> <BR>
      <INPUT TYPE="Submit" Name="Submit" Value="Send">
</Form>
</BODY>
</HTML>
```

ASP fila – *leggtil.asp* – er slik:

```
<% Language="VBScript" %>
<HTML>
<HEAD>
<TITLE>ASP Access DataBase test 4</TITLE>
<!-------
Demonstrere hvordan man kan legge til en post i en Access (97) database.
Databasen inneholder kun en tabell - Navneliste.
Tabellen har et nøkkelfelt: Id, og to datafelt:
Navn (Tekst, std.)
Alder (Tall, byte)
Denne .ASP fila kalles fra ACTION=... i leggtil.htm
```

Thors innføring i ASP

```
----->
</HEAD>
<BODY>
<H2>Access DataBase test IV</H2>
<P>
  Dette ASP eksemplet viser hvordan man kan legge til en post bakerst
  i en Access database. Kun en tabell - Navneliste -
  med et nøkkelfelt kalt <EM>Id</EM>, og to datafelter:
  <EM>Navn</EM> og <EM>Alder</EM>.<BR>
</P>
<%
  Dim Kobling, RecSet ' Deklarer variabler, Koblingsobjektet brukes
                        ' til å koble oss opp mot .mdb, RecSet tar feltene
  Dim sBane           ' Stasjon:\bane til .MDB
  Dim sNavn, s, Tmp

  ' Først brukes Koblings objektet til å åpne forbindelse til ADO:
  Set Kobling = Server.CreateObject("ADODB.Connection")
  Kobling.Mode = adModeWrite           ' Åpne datafila med skriveadgang.
  sBane = Server.MapPath("test.mdb")   ' Server gir oss bane til datafil
  ' Angi hvem som tilbyr koblingen. "...OLEDB.4.0" for
  ' Office 2K, bruk "...OLEDB.3.5" for eldre ver.
  Kobling.Provider = "Microsoft.Jet.OLEDB.4.0"
  ' Angi stasjon:\bane\databasefilnavn
  Kobling.ConnectionString = "Data Source=" &sBane
  Kobling.Open                 ' Åpne datafila
  ' ----- Hent info fra HTML skjema
  sNavn = Request.Form("Navn")      ' Hent info fra Alder feltet
  ' Opprett et postobjekt (Recordset)
  Set RecSet = Server.CreateObject("ADODB.Recordset")
  ' Bygg opp strengen s med nødvendige parametre
  s = "INSERT INTO Navneliste (Navn, Alder) VALUES(" &"'" &_
      sNavn &"'" &", " &"'" &Request.Form("Alder") &"'" &)"
  Tmp = Kobling.Execute(s)
  Kobling.Close                ' Lukk koblingsobjektet
  RecSet.Close                 ' Lukk postobjektet
  Set RecSet = Nothing
  Set Kobling = Nothing
%>
</BODY>
</HTML>
```

Først deklarerer nødvendige variabler for koblingsobjekt og postobjekt. I tillegg deklarerer variabler for feltet 'Navn', en variabel (s) som skal inneholde SQL kommandoen som brukes for å legge til posten og en midlertidig variabel (Tmp).

Koblingsobjekt og postobjekt åpnes som før. Deretter hentes verdien som ble tastet inn i HTML skjemaets felt 'Navn' slik:

```
sNavn = Request.Form("Navn")
```

Siden METHOD=POST i HTML skjemaet brukes RequestForm... i ASP fila. SQL kommandoen som brukes til å legge til posten bygges opp slik:

```
s = "INSERT INTO Navneliste (Navn, Alder) VALUES(" &"'" &_
    sNavn &"'" &", " &"'" &Request.Form("Alder") &"'" &)"
```

Legg merke til at variabelen *sNavn* må omgis av apostrof ('), det samme gjelder *Alder* som

hentes direkte med Request.Form...

Selve SQL kommandoen utføres med:

```
Tmp = Kobling.Execute(s)
```

Til slutt lukkes koblings- og postobjektene, dette er god programmeringsskikk.

Eksempel III. Vise en og en post fra en Access fil

Så lenge datafila kun inneholder noen få poster, er det ikke noe problem å vise alle postene i en HTML tabell. Dersom datafila har mange poster kan HTML tabellen fort bli svært lang, og da blir det vanskelig å orientere seg. En annen løsning kan da være å vise en og en post, og la brukeren bla seg frem til neste post med å trykke på en knapp (og la en annen knapp vise forrige post).

Her brukes kun en fil – ASP fila, og vi lar den kalle seg selv. For å få fram knappene kan vi lage et skjema ("Form") nederst i ASP fila vår. Vi må passe på å angi navnet på ASP fila i *ACTION*=... attributten if <FORM ... > seksjonen.

Det vil si at vi har behov for to "Submit" knapper. Dette vil fungere bra så lenge vi bruker ulike navn på de to knappene. Bruk ASP fila fra eksempel I som utgangspunkt og endre den slik (lagre som: db3.asp):

```
<% Language="VBScript" %>
<HTML>
<HEAD>
<TITLE>ASP Access DataBase test 3</TITLE>
<!-------
Demonstrere hvordan man kan bla fra en til den neste i en Access (97)
database. Databasen inneholder kun en tabell - Navneliste.
Tabellen har et nøkkelfelt: Id, og to datafelt:
Navn (Tekst, std.)
Alder (Tall, byte)
----->
</HEAD>
<BODY>
<H2>Access DataBase test III</H2>
<P>
  Dette ASP eksemplet viser hvordan man kan bla fra en post til den
  Neste i en Access database. Kun en tabell - Navneliste -
  med et nøkkelfelt kalt <EM>Id</EM>, og to datafelter:
  <EM>Navn</EM> og <EM>Alder</EM>. Innhold:
</P>
<%
  Dim Kobling, RecSet ' Deklarer variabler, Koblingsobjektet brukes
                    ' til å koble oss opp mot .mdb, RecSet tar feltene
  Dim sBane          ' Stasjon:\bane til .MDB
  Dim PostNr         ' Hold rede på hvilken post som vises
  Dim sFlytt         ' Flytte til Neste el. Forrige post?
  Dim sTmp

  ' Først brukes Koblings objektet til å åpne forbindelse til ADO:
  Set Kobling = Server.CreateObject("ADODB.Connection")
  sBane = Server.MapPath("test.mdb")          ' Server gir oss bane
  ' Angi hvem som tilbyr koblingen. "... OLEDB.4.0" for Office 2K,
  ' bruk "...OLEDB.3.5" for eldre ver.
  Kobling.Provider = "Microsoft.Jet.OLEDB.4.0"
  ' Angi stasjon:\bane\databasefilnavn
  Kobling.ConnectionString = "Data Source=" &sBane
```

Thors innføring i ASP

```
Kobling.Open                ' Åpne databasen
' Opprett et postobjekt (Recordset)
Set RecSet = Server.CreateObject("ADODB.Recordset")
' Velg alle poster (*) fra tabellen 'Navneliste'
RecSet.Open "SELECT * FROM Navneliste ORDER BY Id", Kobling, 1, 1
' Her henter vi info. fra skjema (Form) til slutt i denne ASP fila.
sTmp = Request.Form("Neste")      ' Klikk på "Neste" knappen?
If sTmp = "->" Then                ' Ja, sTmp får knappens "value"
    sFlytt = "Neste"              ' Flytt til neste post
Else
    sTmp = Request.Form("Forrige") ' Ellers sjekk "Forrige" knapp
    If sTmp = "<-" Then
        sFlytt = "Forrige"
    End If
End If
PostNr = CLng(Request.Form("PostNr")) ' Hent lagret postnr
If (sFlytt = "Neste") And (PostNr < RecSet.RecordCount - 1) Then
    PostNr = PostNr + 1          ' Vis neste post
End If
If (sFlytt = "Forrige") And (PostNr > 0) Then
    PostNr = PostNr - 1        ' Vis forrige post
End If
' Vis angitt post
s=RecSet.Move(PostNr, 1)      ' Hent angitt post
' Hent ut data fra feltene 'Navn' og 'Alder'
Response.Write "Navn: " &RecSet("Navn") &", " &_
                RecSet("Alder") &" år.<BR>"

RecSet.Close                ' Lukk postobjektet
Kobling.Close              ' Lukk koblingsobjektet
Set RecSet = Nothing
Set Kobling = Nothing
%>
<!-- Pass på at ACTION=... angir navnet på denne .ASP fila. ----->
<Form Action="DB3.asp" Method="POST">
    <Input Type="Hidden" Name="PostNr"
    <% Response.Write "Value=" &PostNr %> >
    <BR>
    <INPUT TYPE="Submit" Name="Forrige" Value="<-">
    <INPUT TYPE="Submit" Name="Neste" Value="->">
</Form>
</BODY>
</HTML>
```

Her er variabelen *sTmp* lagt til variabeldeklarasjonene. Denne variabelen brukes til å hente inn verdien til de to "Submit" knappene som er lagt inn i FORM delen av denne ASP fila.

Vi henter verdien til "Submit" knappen Neste (->) med instruksjonen:

```
sTmp = Request.Form("Neste")
```

Dersom *sTmp* har verdien "->", får variabelen *sFlytt* verdien "Neste".

I FORM delen er det to "Submit" knapper. Den ene har navnet "Neste", den andre har navnet "Forrige". Så lenge de har ulike navn kan vi skille dem i ASP koden; `Request.Form("Neste")` eller `Request.Form("Forrige")`. Den verdien som `Request.Form` returnerer er den verdien som vil legger inn i attributten "Value":

```
<INPUT TYPE="Submit" Name="Neste" Value="->">
```

eller:

```
<INPUT TYPE="Submit" Name="Forrige" Value="<-">
```

Variablene sTmp vil få verdien "->" dersom "Neste" knappen ble trykket, eller verdien "<-" dersom "Forrige" knappen ble trykket. Dersom "Forrige" eller "Neste" knappen er trykket kontrollerer vi at vi ikke går foran første post eller bak siste post før postpekeren flyttes til forrige eller neste post.

Eksempel IV. Slette en post fra en Access fil

La oss fortsette med datafila Test.mdb, og vise hvordan vi kan slette den aktive posten i datafila. Ta utgangspunkt i Eks.III og endre slik (lagre som db4.asp):

```
<% Language="VBScript" %>
<HTML>
<HEAD>
<TITLE>ASP Access DataBase test 4</TITLE>
<!-------
Demonstrere hvordan man kan vise en og en post i en
Access (97) database. Databasen inneholder kun en
tabell - Navneliste.
Tabellen har et nøkkelfelt: Id, og to datafelt:
Navn (Tekst, std.)
Alder (Tall, byte)
----->
</HEAD>
<BODY>
<H2>Access DataBase test IV</H2>
<P>
Dette ASP eksemplet viser hvordan man kan vise en og en post i en
Access database, og slette en post. Kun en tabell - Navneliste -
med et nøkkelfelt kalt <EM>Id</EM>, og to datafelder:
<EM>Navn</EM> og <EM>Alder</EM>. Innhold:
</P>
<%
Dim Kobling, RecSet ' Deklarer variabler, Koblingsobjektet brukes
                    ' til å koble oss opp mot .mdb, RecSet tar feltene
Dim sBane           ' Stasjon:\bane til .MDB
Dim PostNr          ' Hold rede på hvilken post som vises
Dim PostID          ' Verdi til postidentifikator (nøkkelfelt)
Dim sFlytt          ' Flytte til Neste el. Forrige post?
Dim sSQL            ' SQL streng
Dim sTmp
' Først brukes Koblings objektet til å åpne forbindelse til ADO:
Set Kobling = Server.CreateObject("ADODB.Connection")
sBane = Server.MapPath("test.mdb")           ' Server gir oss bane
' Angi hvem som tilbyr koblingen. "... OLEDB.4.0" for
' Office 2K, bruk "...OLEDB.3.5" for eldre ver.
Kobling.Provider = "Microsoft.Jet.OLEDB.4.0"
' Angi stasjon:\bane\databasefilnavn
Kobling.ConnectionString = "Data Source=" &sBane
Kobling.Open           ' Åpne databasen
' Opprett et postobjekt (Recordset)
Set RecSet = Server.CreateObject("ADODB.Recordset")
' Velg alle poster (*) fra tabellen 'Navneliste'
RecSet.Open "SELECT * FROM Navneliste ORDER BY Id", Kobling, 1, 1
' Her henter vi info. fra skjema (Form) til slutt i denne ASP fila.
```

Thors innføring i ASP

```
sTmp = Request.Form("Neste")      ' Klikk på "Neste" knappen?
If sTmp = "->" Then                ' Ja, sTmp får knappens "value"
    sFlytt = "Neste"              ' Flytt til neste post
Else
    sTmp = Request.Form("Forrige")
    If sTmp = "<-" Then
        sFlytt = "Forrige"
    Else
        sTmp = Request.Form("Slett")
        If sTmp = "Slett" Then
            sFlytt = "Slett"      ' Slette aktiv post
        End If
    End If
End If
PostNr = CLng(Request.Form("PostNr")) ' Hent lagret postnr
If (sFlytt = "Neste") And (PostNr < RecSet.RecordCount - 1) Then
    PostNr = PostNr + 1          ' Vis neste post
End If
If (sFlytt = "Forrige") And (PostNr > 0) Then
    PostNr = PostNr - 1          ' Vis forrige post
End If
' Vis angitt post
s = RecSet.Move(PostNr, 1)      ' Hent angitt post
' Hent ut data fra feltene 'Navn' og 'Alder'
Response.Write "Navn: " & RecSet("Navn") & ", " & _
                RecSet("Alder") & " år.<BR>"
PostID = CLng(RecSet("Id"))      ' Hent verdi til aktiv posts nøkkelfelt

If sFlytt = "Slett" Then
    Response.Write "Sletter aktiv post nr. " & PostID
    ' Bygg opp strengen sSQL med nødvendige parametre:
    sSQL = "DELETE * FROM Navneliste WHERE Id = " & PostID
    sTmp = Kobling.Execute(sSQL)
End If
RecSet.Close                    ' Lukk postobjektet
Kobling.Close                   ' Lukk koblingsobjektet
Set RecSet = Nothing
Set Kobling = Nothing
%>
<!-- Pass på at ACTION=... angir navnet på denne .ASP fila. ---->
<Form Action="DB4.asp" Method="POST">
    <Input Type="Hidden" Name="PostNr"
    <% Response.Write "Value=" & PostNr %> >
<BR>

    <INPUT TYPE="Submit" Name="Forrige" Value="<-">
<INPUT TYPE="Submit" Name="Neste" Value="->">
<INPUT TYPE="Submit" Name="Slett" Value="Slett">
</Form>
</BODY>
</HTML>
```

Første del er som tidligere, men vi har deklartert en ny variabel sSQL. Denne variabelen (sSQL) bruker vi til å bygge opp den SQL setningen som sletter den aktive posten (den posten som vises på skjermen).

Som tidligere bruker vi variabelen sTmp til å sjekke hvilken av de tre "Submit" knappene som brukeren valgte ("Neste", "Forrige" eller "Slett").

Instruksjonen

Thors innføring i ASP

```
PostId = CLng("RecSet.Id")
```

henter nøkkelfeltet til den aktive posten. Denne verdien blir brukt dersom posten skal slettes. Vi bruker CLng funksjonen til å omforme den tekstverdien som RecSet("Id") returnerer til datatypen Long, som er kompatibel med Access sin nøkkelfelt (identifikator) datatype.

Vi viser posten på skjermen som i forrige eksempel, men vi har like etter føyd til noen instruksjoner som sjekker om variabelen sFlytt har verdien "Slett". Dersom det er tilfelle skriver vi en beskjed (med Response.Write) om at aktiv post slettes. Deretter bygger vi opp SQL strengen (i variabelen sSQL) som angir hvilken post som skal slettes:

```
sSQL = "DELETE * FROM Navneliste WHERE Id = " & PostID
```

Siden feltet Id er et nøkkelfelt blir bare aktiv post slettet.

Vi bruker objektet Kobling sin metode Execute til å utføre SQL instruksjonen:

```
sTmp = Kobling.Execute(sSQL)
```

Siden Execute returnerer en verdi lar vi den midlertidige hjelpevariabelen sTmp ta imot returverdien. Til slutt lukkes Koblings og RecSet objektene som vanlig.

I FORM delen har vi lagt til en linje:

```
<INPUT TYPE="Submit" Name="Slett" Value="Slett">
```

som oppretter en ny "Submit" knapp med navnet "Slett". En If setning i skript delen kontrollerer om variabelen sFlytt har verdien "Slett", dersom det er tilfelle slettes den aktive posten (den som vises).