Parameters

Denne artikel beskriver hvorfor parameters er gode.

Den forudsætter lidt kendskab til VB.NET og ADO.NET.

Der findes en tilsvarende artikel med C#.

Skrevet den **18. Feb 2010** af **arne_v** I kategorien **Programmering / Visual Basic .NET** | ⭐⭐⭐☆☆

Historie:
V1.0 - 12/11/2005 - original
V1.1 - 14/11/2005 - fix manglende ' i eksempel
V1.2 - 18/10/2010 - smårettelser

Problemerne

Alle der har programmeret op mod en database kender et eller flere af problemerne.

1) uheldige single quotes

Eksempel:

sql = "INSERT INTO tt VALUES(" + id + ",'" + name + "')"

hvis navnet er Hansen så virker det fint:

INSERT INTO tt VALUES(123,'Hansen')

men hvis navnet er O'Toole så giver det fejl:

INSERT INTO tt VALUES(123,'O'Toole')

2) single quotes med vilje (kendt som SQL injection)

Eksempel:

sql = "SELECT * FROM myusers WHERE un = '" + username + "' AND pw = '" + password + "'"

(efterfulgt af et test på om der blev fundet nogle records)

det virker fint for den pæne bruger som indtaster:
arne
hemmeligt

SELECT * FROM myusers WHERE un = 'arne' AND pw = 'hemmeligt'

men det returnerer forkert OK for den ondsinded cracker som indtaster:
arne

x' OR 'x' = 'x

SELECT * FROM myusers WHERE un = 'arne' AND pw = 'x' OR 'x' = 'x'

3) dato formater

Den giver altid problemer med input til databasen.

Hvilket format skal man bruge:
dd mm yyyy (dansk)
mm dd yyyy (US)
yyyy mm dd (sorterings rigtigt)
?

Skal værdierne:
sættes i ''
sættes i ##
konverteres med en funktion
?

Styres formatet af:
operativ system sprog version
database sprog version
styre system sprog indstilling
database sprog indstilling
?

Løsningen

En begynder løsning på de 2 første problemer er at fordoble
alle single quotes:

sql = "INSERT INTO tt VALUES(" + id + ",'" + name.Replace("'","''") + "')"

INSERT INTO tt VALUES(123,'Hansen')

INSERT INTO tt VALUES(123,'O''Toole')

sql = "SELECT * FROM myusers WHERE un = '" + username.Replace("'","''") + "' AND pw = '" +
password.Replace("'","''") + "'"

SELECT * FROM myusers WHERE un = 'arne' AND pw = 'hemmeligt'

SELECT * FROM myusers WHERE un = 'arne' AND pw = 'x'' OR ''x'' = ''x'

og det virker, men det er ikke super godt:
* det er ikke kønt
* det er nemt at glemme
* forskellige databaser kan have forskellige andre tegn som også kan misbruges
* det løser ikke dato problemet

Dato problemet undlader man ofte helt at løse. Man hardkoder SQL sætningerne
med formatet til det system man udvikler på. Enten med en DateTime ToString eller
ved noget banal string manipulation.

Og så får man problemet når man skal have det til at køre på en anden maskine.

Den rigtige løsning som løser alle problemerne er at bruge parameters fremfor at sætte værdier ind i selve SQL strengen.

Med parameters skriver man bare en placeholder alle de steder i ens SQL hvor der skal indsættes værdier og så sætter man de værdier og ADO.BNET håndterer alle problemerne.

Det lyder måske lidt mystisk, men lad os tage nogle eksempler.

Kode eksempler

Eksemplerne vil bruge Access som database, men alle eksemplerne virker lige så godt med SQLServer eller MySQL, man skal bare rette fra OleDbXxxx til SqlXxxx eller MySqlXxxx (og måske tilrette nogle data typer som kan hedde noget forskelligt).

De data som køres på er:

```
CREATE TABLE tt (
    id INTEGER PRIMARY KEY,
    name VARCHAR(50)
);

CREATE TABLE myusers (
    un VARCHAR(32) PRIMARY KEY,
    pw VARCHAR(32)
);

INSERT INTO myusers VALUES('arne', 'hemmeligt');

CREATE TABLE dtest (
    i INTEGER PRIMARY KEY,
    d DATETIME
);
```

Først INSERT med single quotes:

TestPrep1.vb

```
Imports System
Imports System.Data.OleDb

Namespace TestParam
    Public Class TestClass
        Public Shared Sub Main(ByVal args As String())
            Dim con As OleDbConnection = New
OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;Data
```

```
Source=C:\Databases\MSAccess\Test.mdb")
            con.Open
            Dim cmd As OleDbCommand = New OleDBCommand("INSERT INTO tt VALUES
(@id, @name)", con)
            cmd.Parameters.Add("@id", OleDbType.Integer)
            cmd.Parameters.Add("@name", OleDbType.VarChar, 50)
            cmd.Parameters("@id").Value = 123
            cmd.Parameters("@name").Value = "Hansen"
            cmd.ExecuteNonQuery
            cmd.Parameters("@id").Value = 124
            cmd.Parameters("@name").Value = "O'Toole"
            cmd.ExecuteNonQuery
            con.Close
        End Sub
    End Class
End Namespace
```

Bemærk at vi ikke sætter '' omkring placeholder når det er en String.

Og en gang mere.

TestPrep2.vb:

```
Imports System
Imports System.Data
Imports System.Data.OleDb

Namespace TestParam
    Public Class TestClass
        Public Shared Function IsValid(ByVal un As String, ByVal pw As String)
As Boolean
            Dim con As OleDbConnection = New
OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=C:\Databases\MSAccess\Test.mdb")
            con.Open
            Dim cmd As OleDbCommand = New OleDbCommand("SELECT * FROM myusers
WHERE un = @un AND pw = @pw", con)
            cmd.Parameters.Add("@un", OleDbType.VarChar, 50)
            cmd.Parameters.Add("@pw", OleDbType.VarChar, 50)
            cmd.Parameters("@un").Value = un
            cmd.Parameters("@pw").Value = pw
            Dim rdr As OleDbDataReader = cmd.ExecuteReader
            Dim res As Boolean = rdr.Read
            rdr.Close
            con.Close
            Return res
        End Function
        Public Shared Sub Main(ByVal args As String())
            Console.WriteLine(IsValid("anonymous", ""))
            Console.WriteLine(IsValid("arne", "hemmeligt"))
            Console.WriteLine(IsValid("arne", "x' OR 'x' = 'x"))
```

```
        End Sub
    End Class
End Namespace
```

Og til sidst dato.

TestPrep3.vb:

```
Imports System
Imports System.Threading
Imports System.Data
Imports System.Data.OleDb

Namespace TestParam
    Public Class TestClass
        Public Shared Sub Main(ByVal args As String())
            Dim con As OleDbConnection = New
OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=C:\Databases\MSAccess\Test.mdb")
            con.Open
            Dim ins As OleDbCommand = New OleDbCommand("INSERT INTO dtest
VALUES (@i, @d)", con)
            ins.Parameters.Add("@i", OleDbType.Integer)
            ins.Parameters.Add("@d", OleDbType.Date)
            Dim j As Integer
            For j = 0 To 9
                ins.Parameters("@i").Value = j
                Dim dt As DateTime = DateTime.Now
                ins.Parameters("@d").Value = dt
                ins.ExecuteNonQuery
                Thread.Sleep(1000)
            Next
            Dim sel As OleDBCommand = New OleDBCommand("SELECT * FROM dtest
WHERE d > @d", con)
            sel.Parameters.Add("@d", OleDbType.Date)
            Dim cut As DateTime = DateTime.Now.AddSeconds(-5)
            sel.Parameters("@d").Value = cut
            Dim rdr As OleDbDataReader = sel.ExecuteReader
            While rdr.Read
                Dim i As Integer = CType(rdr(0), Integer)
                Dim dt As DateTime = CType(rdr(1), DateTime)
                Console.WriteLine(i & " " & dt)
            End While
            rdr.Close
            con.Close
        End Sub
    End Class
End Namespace
```

Stored Procedures

Det er ikke nødvendigt at bruge stored procedures for at bruge parameters.

Men hvis man bruger stored procedures så er syntaxen for parameters den samme.

Konklusion

Det er en selvfølge at man bruger parameters når man skal igang med seriøs brug af ADO.NET.

Eksemplerne i denne artikel skulle gerne have givet et indblik i hvordan man bruger parameters.

Kommentar af kieldjen d. 16. Dec 2005 | 1

Kommentar af webcreator d. 13. Nov 2005 | 2

Kommentar af ghetto d. 21. Apr 2006 | 3

Kommentar af capn d. 11. Mar 2007 | 4

Fed artikel. Altid godt med noget best-pratice prædiken. http://www.ogp-consult.dk

Kommentar af larsmeyer d. 21. Feb 2010 | 5

God guide, alle der koder eller begyndet på .NET burde som det første gå i gang med at bruge parameters og få afvænnet alle de dårlige vaner fra tidligere programmering (fx fra ASP).

Og hvis nogen har kørende hjemmeside løsninger derude, så kan jeg klart anbefale at i tester jeres site for sql injection. Der findes gratis tools (mener det er bl.a. IBM der har lavet det) som automatisk tester hele websitet og alle undersider og finder de mest åbenlyse sikkerhedsfejl, primært sql injections.

Bare google efter 'freesql injection tool' så skal i nok finde det :)