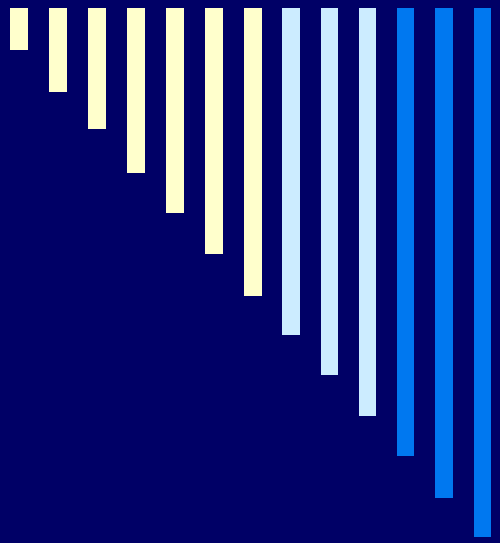
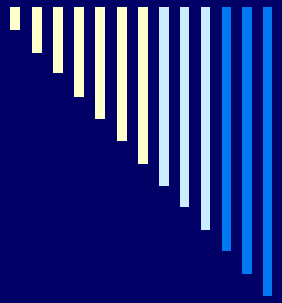


Access und die andern (Office) Programme

Armin Hess

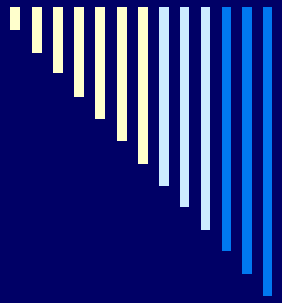
Was wir wollen

- Excel, Word, Powerpoint und andere Programme von Access aus „fernsteuern“



Grundlagen

- Excel, Word, Powerpoint und andere Programme präsentieren sich gegenüber Access als **Objecte**
- Wir brauchen also ein Objekt

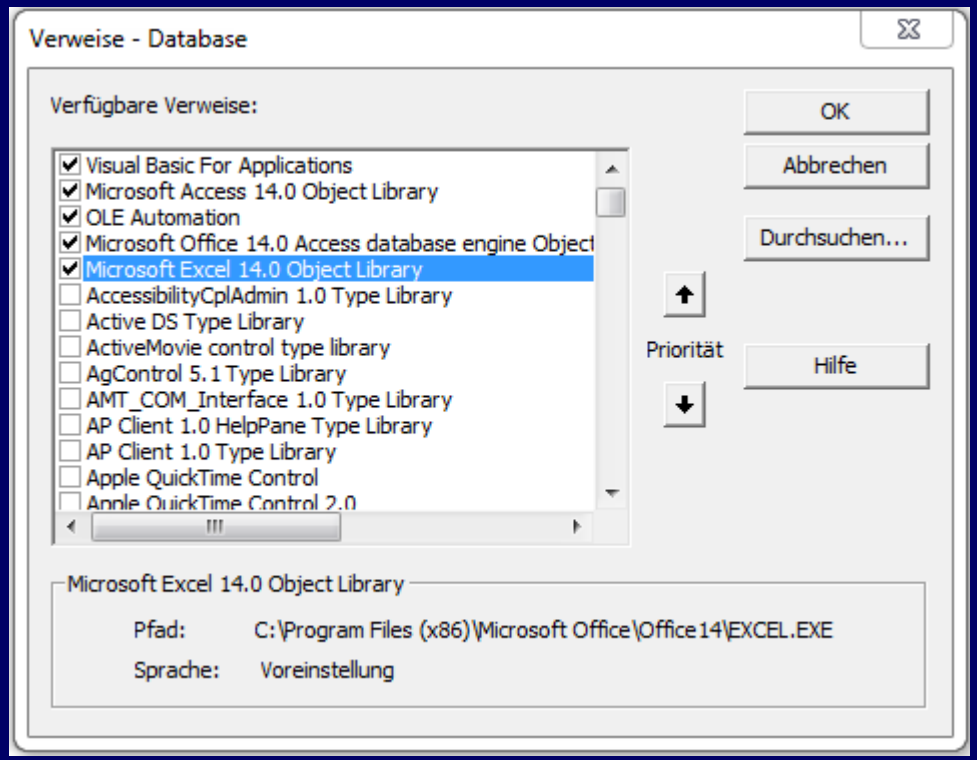


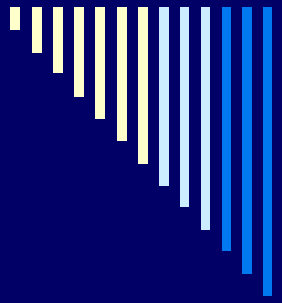
Woher bekommen wir ein Objekt?

```
Dim oXLS as Excel.Application
Set oXLS = new Excel.Application
```

- Die letzte Zeile startet ein unsichtbares Excel
- Das nennt man Early Binding.
- Es wird zur Compilezeit gebunden.

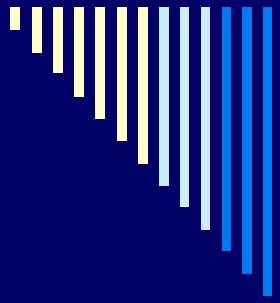
Randbedingung





Das Kreuz mit den Verweisen

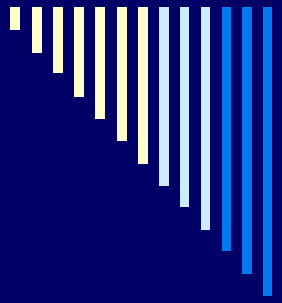
- Welche Version hat mein Zielsystem?
- Unter welchem Pfad?
- …?



Eine Alternative

```
Dim oXLS as object
Set oXLS = createObject(“Excel.Application“)
```

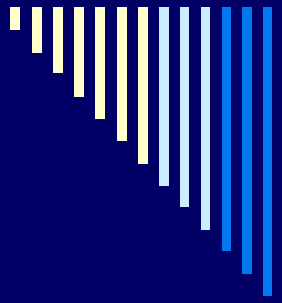
- Leider haben wir dann kein Intelisense ☹
- Das nennt man Late Binding
- Es wird zur Laufzeit gebunden



Eine Lösung

- Wir nehmen zum Entwickeln Early Binding.
- Wir nehmen zur Laufzeit Late Binding.

- Und wie macht man das am geschicktesten?

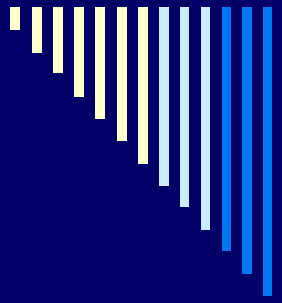


Meine Lösung

- Alle Zugriffe auf die Office Programme werden in einer Klasse gekappselt.
- Die Entwicklungsversion der Klasse arbeitet mit Early Binding.
- Die Produktionsversion mit Late Binding.
- Die entsprechenden Variablen werden an einer Stelle deklariert.
- Dann kann das einfach geändert werden.

Excel erzeugen

```
Dim oXLS as Object
On Error Resume Next
Set getExcel = GetObject(, "Excel.Application")
If getExcel Is Nothing Then
   Set getExcel = createObject("Excel.Application")
End If
```

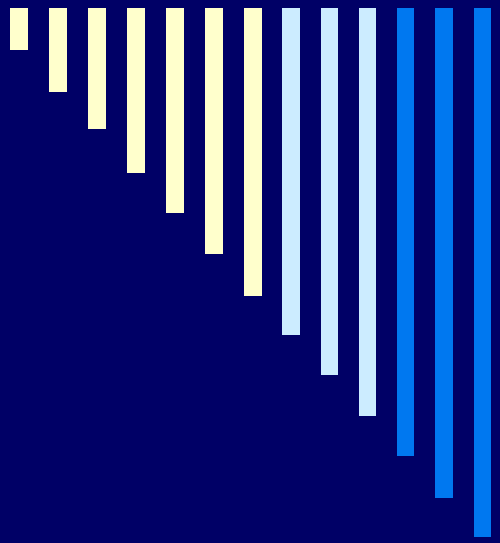


Und was machen wir jetzt damit?

- Hinter Excel steckt ein umfangreiches Objektmodell.
- Details gibt es hier: http://msdn.microsoft.com/de-de/library/wss56bz7%28v=vs.80%29.aspx
- Das kann man durcharbeiten und die nötigen Eigenschaften und Funktionen suchen.

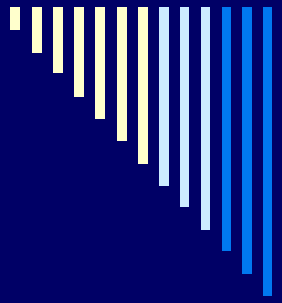
Es geht allerdings auch einfacher

1. Excel starten
2. Entwicklertools -> Makro aufzchn.
3. Das tun, das man machen möchte
4. Entwicklertools -> Aufzeichnung beenden
5. Entwicklertools -> Visual Basic



Demo

Excel Macro Recorder



Ein Beispiel

- Wir öffenen eine Datei und wählen ein Arbeitsblatt aus

```
Sub Makro5()
    Workbooks.Open Filename:="D:\data\test.xls"
    Sheets("Sheet1").Select
End Sub
```

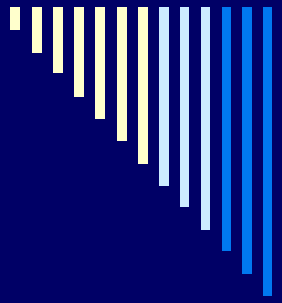
Noch ein Beispiel

- Wir klicken in die Zelle B2 und schreiben eine Zahl hinein.

```
Sub Makro1()
    Range("B2").Select
    ActiveCell.FormulaR1C1 = "123"
    Range("B3").Select
End Sub
```

Und jetzt nach Access

```
With oXLS.ActiveSheet
     .Range("B2").Select
     .ActiveCell.FormulaR1C1 = "123"
End with
```

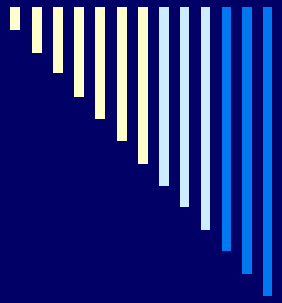


Und wie geht's weiter?

- Mit diesen Teilstücken können nun Funktionen gebaut werden
- Dabei werden konstante Teile aus den Makros durch Variablen ersetzt

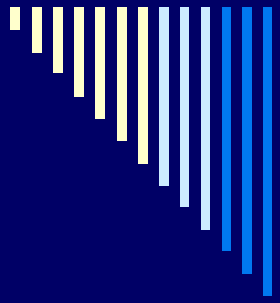
Die wichtigsten Objekte

- Application
  - Excel selbst
- Workbook
  - Eine Exceldatei
- Worksheet
  - Ein Tabellenblatt
- Selection
  - Eine oder mehrere ausgewählte Zellen



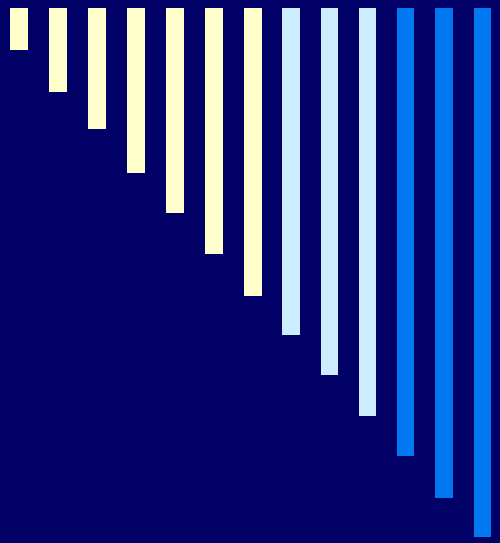
Noch eine Möglichkeit

- Ich hab mir eine Klasse geschrieben
- Mit der können Exceldateien bearbeitet werden. Z.B:
- Import
- Ergänzen von Spalten
- Komplexer Export



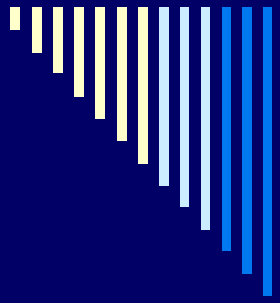
Und so geht‘s

```
Dim cXLS as clsExcel
Set cXLS = new clsExcel
cXLS.File=“D:\data\import.xls“
lngRows=cXLS.lastrow
For lngRow = 1 to lngRows
   strData=cXLS.getCell (1,lngRow)
   cXLS.setCell 2, lngRow, uCase(strData)
Next
cXls.save
```



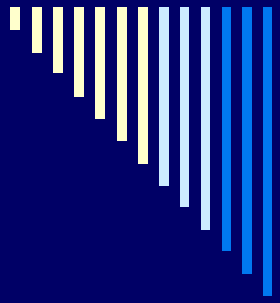
Demo

clsExcel



Word und Powerpoint

- Geht auf die gleiche Art und Weise
- Die entsprechenden Parameter beim createObject heißen eben “Word.Application“ und “Powerpoint.Application“
- Leider ist beim Powerpoint 2012 der Macrorecorder entfallen.
- Da muss man auf Powerpoint 2003 zurückgreifen.



Noch ein paar Tipps

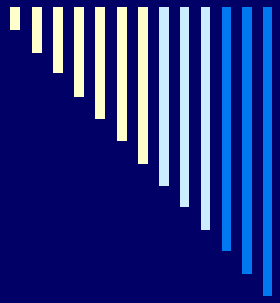
- Das Zerstören des Objekts beendet **nicht** die Anwendung.
- Dafür gibt es eine Funktion (oXLS.Quit)
- Läuft Access auf einen Laufzeitfehler, gibt es u.U. einen Zombie.
- Der muss mit dem Taskmanager gekillt werden.

Und noch was nützliches

- clsMeter
- Ist ein leicht zu benutzender Fortschrittsbalken mit vielen Einstellmöglichkeiten.

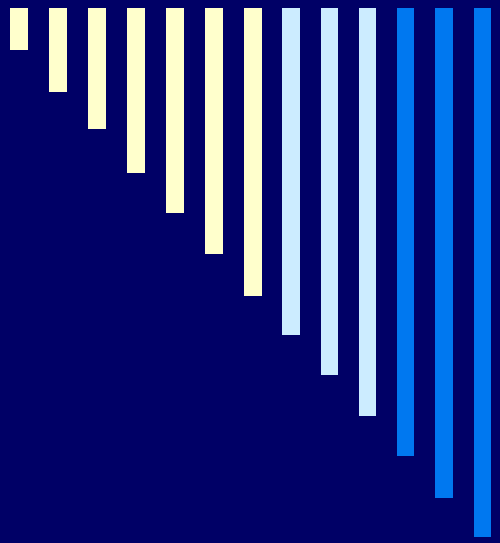
Und so geht‘s

```
Dim cMeter as clsMeter
Set cMeter = new clsMeter
cMeter.init “Import“, lngRows
For lngRow = 1 to lngRows
    ...
    if not cMeter.update() then exit For
Next
cMeter.remove
```



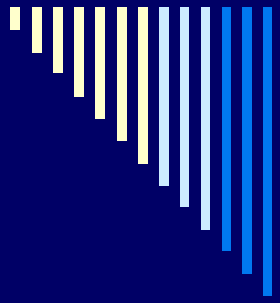
Weitere Features

- Anzeige der Prozente
- Verstrichene Zeit
- Geschätzte Zeit
- Verzögerung für Schätzung
- Text für Schätzung
- Abbruch Knopf
- Balkenfarbe
- …



Demo

clsMeter



Nutzung

- Die Klassen können frei genutzt werden.
- Auch in kommerziellen Anwendungen.
- Der Autorenhinweis darf nicht entfernt werden.
- Wenn Ihr Fehler oder Schwächen findet, bitte Mail an armin@gaeuwetter.de
- Der Code wird ohne jede Gewährleistung überlassen.

Download

http://www.gaeuwetter.de/access

Aber erst ab Donnerstag, 20.6.13

Noch Fragen???