

## Software-Anleitung

# Excel-Projektdaten wirkungsvoll schützen

## Teil 2: Schutzfunktionen für Fortgeschrittene

Der erste Teil dieses Beitrags zeigte, welche bordeigenen Mittel Excel zum Schutz von Dateien und Tabellenblättern zur Verfügung stellt. In diesem zweiten Teil erfahren Sie, welche fortgeschrittenen Techniken es darüber hinaus gibt, um Arbeitsmappen und Tabellen vor Fehlern zu schützen. Eine einfache, aber wirkungsvolle Alternative zu den bereits vorgestellten Werkzeugen bietet die Datenüberprüfung. Die darauf basierenden Techniken sind weniger "restriktiv" als der Blatt- oder Arbeitsmappenschutz, der die Excel-Werkzeuge in Menüband und Symbolleiste ausschaltet. Die hier vorgestellten Lösungen können Sie zur Absicherung der Dateneingabe und zur Vermeidung von Falscheingaben verwenden. Um Daten vor dem Benutzer zu verbergen oder ein Kopieren der Daten zu verhindern, sind sie weniger geeignet.

## Einzelne Bereiche schützen (Datenüberprüfung)

### Überschreiben von Zellinhalten verhindern

Mit Hilfe von benutzerdefinierten Formeln können Sie einzelne Bereiche schützen oder für ausgewählte Benutzer freigeben. Das Prinzip ist einfach: Die Datenüberprüfung lässt eine Eingabe in eine Zelle nur zu, wenn das vom Benutzer vorgegebene Gültigkeitskriterium erfüllt ist. Ein solches Gültigkeitskriterium kann z.B. das Ergebnis einer Formelberechnung sein: Liefert die Formel den Wert "wahr", ist die Eingabe möglich, andernfalls nicht. Definiert der Benutzer als Gültigkeitskriterium eine Formel, die bei jeder Eingabe stets den Wert "falsch" liefert, verhindert er somit, dass Zellinhalte überschrieben werden.

Markieren Sie den Bereich, den Sie schützen wollen und öffnen Sie anschließend den Dialog zur Datenüberprüfung mit *Daten / Datentools / Datenüberprüfung*. Klicken Sie unter "Zulassen" im Register "Einstellungen" die Auswahl "Benutzerdefiniert" an, deaktivieren Sie die Option "Leere Zellen ignorieren" und geben Sie folgende Formel ein:

<>" "

Die Formel sorgt dafür, dass Excel alle Eingaben im ausgewählten Bereich verhindert. Den Zellinhalt zu löschen ist jedoch weiterhin möglich.

### Autor



#### Ignatz Schels

Seit 1986 selbst. DV-  
Dozent und Journalist,  
Leitet u. konzipiert MS-

Project-Seminare u.a. für  
DaimlerChrysler, Siemens, T-Systems

Kontakt: [info@schels.de](mailto:info@schels.de)

Mehr Informationen unter:  
[projektmagazin.de/autoren](http://projektmagazin.de/autoren)

### ähnliche Artikel

in der Rubrik:

[Einsatz von Software](#)

Zur Erläuterung tragen Sie anschließend noch eine passende Fehlermeldung für den Benutzer in der Registerkarte "Fehlermeldung" ein (Bild 1), bevor Sie die Datenüberprüfung mit "OK" bestätigen. Versucht der Benutzer eine Zelle in diesem Bereich zu ändern, erhält er einen entsprechenden Hinweis und muss die Aktion abbrechen.

! Der Typ der Fehlermeldung definiert die "Schärfe" des Schutzes: "Stopp" verhindert alle Eingaben. Die Typen "Warnung" und "Information" eignen sich für Warnmeldungen, der Benutzer kann die Zellen durch Klick auf OK überschreiben.

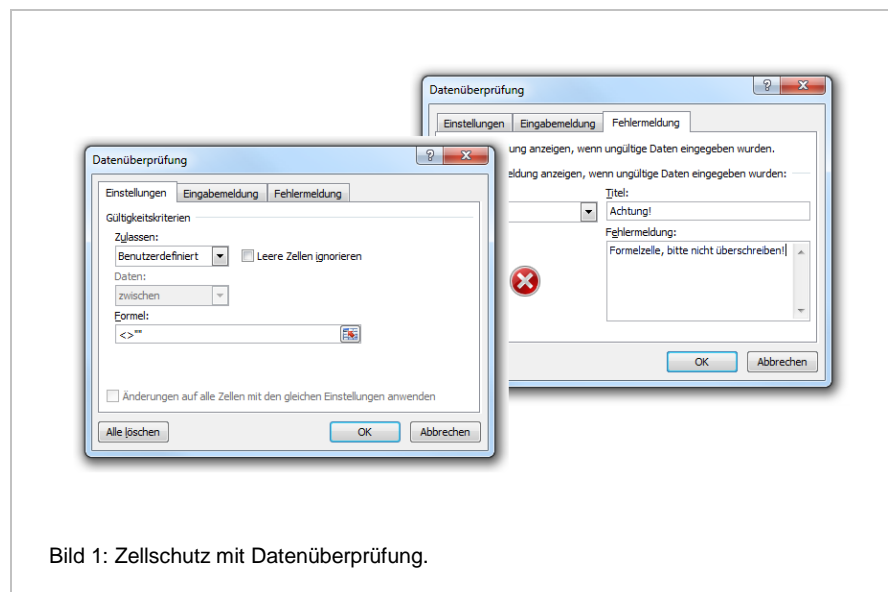


Bild 1: Zellschutz mit Datenüberprüfung.

### Bereiche durch Kennwort schützen (Variante 1)

Die benutzerdefinierte Datenüberprüfung können Sie ebenfalls nutzen, um bestimmte Bereiche der Tabelle für einzelne Benutzer freizuschalten. Definieren Sie dazu ein Kennwort, das Sie entweder in eine **versteckte Zelle** schreiben oder einem **Bereichsnamen** zuweisen.

Schützen Sie anschließend die gewünschten Tabellenbereiche mit einer Datenüberprüfungsformel, die abfragt, ob das richtige Kennwort

eingetragen ist.

Damit ein Anwender die geschützten Bereiche bearbeiten kann, muss ihm das Kennwort mitgeteilt werden. Diese Technik schützt nur vor versehentlicher Bearbeitung von Zellbereichen, da erfahrene Anwender die Datenüberprüfung einfach entfernen können.

! Das Löschen der geschützten Bereiche ist in allen Fällen weiterhin möglich, dagegen bietet die Datenüberprüfung in keinem Fall Schutz.

### Benutzerkennwörter in versteckten Zellen

Wählen Sie eine Zelle weitab von den Bereichen, die mit Daten und Formeln gefüllt sind. Öffnen Sie mit F5 das Gehezu-Fenster, geben Sie eine Zelladresse (z. B. X5000) ein und drücken Sie die Eingabe-Taste. Tragen Sie in diese Zelle Ihr Kennwort ein, z.B.:

X5000: "Kennwort"

und blenden Sie die Spalte anschließend aus.

**!** Verwenden Sie für das Kennwort nicht die letzte Spalte des Tabellenblattes, damit verhindern Sie, dass neue Spalten eingefügt werden können.

Markieren Sie als nächstes den Bereich, der per Kennwort vor dem Überschreiben geschützt werden soll. Wählen Sie *Daten / Datentools / Datenüberprüfung*, schalten Sie im Reiter "Einstellungen" unter "Zulassen" um auf "Benutzerdefiniert" und geben Sie in das Formelfeld die nach dem Schema  $=\$Spalte\$Zeile="Kennwort"$  aufgebaute Formel ein, die den Bereich schützt. Steht z.B. das Benutzerkennwort in Zelle X5000 und lautet es "Kennwort", heißt die Formel:

$=\$X\$5000="Kennwort"$

Entfernen Sie dann noch das Häkchen vor der Option "Leere Zellen ignorieren". Wechseln Sie anschließend in die Registerkarte "Fehlermeldung" und fügen Sie dort einen entsprechenden Text ein, um dem Benutzer zu erklären, dass die Zellen kennwortgeschützt sind.

Solange das Kennwort in Zelle X5000 steht, lässt sich der zuvor markierte Bereich ohne Einschränkungen bearbeiten. Löschen Sie jedoch das Kennwort, lassen sich die Zellen in diesem Bereich nicht mehr überschreiben.

## Benutzerkennwörter in berechneten Bereichsnamen

Ein Benutzerkennwort lässt sich nicht nur in einer Zelle unterbringen, wie zuvor gezeigt, sondern auch in einem Bereichsnamen. Aktivieren Sie dazu mit Strg+F3 den NamensManager. Erstellen Sie einen neuen Bereichsnamen (z.B. "KW"), wählen Sie unter "Bereich" das aktive Tabellenblatt und geben Sie unter "Bezieht sich auf" die Formel ein, die dem Bereichsnamen den Text des Benutzerkennworts zuweist, in unserem Beispiel

Bezieht sich auf:  $=\text{"Kennwort"}$

Im nächsten Schritt markieren Sie wieder, wie oben beschrieben, den Bereich, der kennwortgeschützt werden soll. Diesmal geben Sie unter *Datenüberprüfung / Zulassen / Benutzerdefiniert* folgende Formel ein, um den Bereich vor Überschreibung zu schützen:

$=KW="Kennwort"$

Vergessen Sie nicht, das Häkchen vor der Option "Leere Zellen ignorieren" zu entfernen und fügen Sie wieder auf der Registerkarte "Fehlermeldung" für den Benutzer eine Erklärung ein, dass die Zellen kennwortgeschützt sind (Bild 2).

Sobald Sie dem Bereichsnamen statt "Kennwort" einen anderen Text zuweisen, ist der Bereich geschützt und kann nicht mehr überschrieben werden.

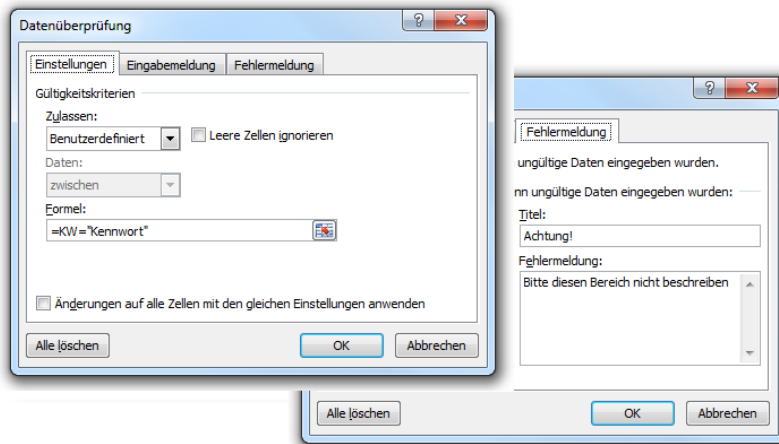


Bild 2: Datenüberprüfungsformel für den berechneten Bereichsnamen.

### Bereiche durch Kennwort schützen (Variante 2)

Eine komfortable Variante der eben vorgestellten Technik verwendet den Benutzernamen als Kennwort. Das Vorgehen eignet sich, um einzelne Tabellenbereiche oder Tabellen nur für bestimmte Benutzer freizuschalten. Sie ersparen sich damit jeweils die Kennworteingabe, wenn Sie eine geschützte Mappe bearbeiten wollen, denn Excel kann den Benutzernamen, der gleichzeitig als Kennwort dient, automatisch auslesen.

Sie haben dabei die Wahl, ob Sie den Windows-Anmeldennamen als Kennwort verwenden wollen oder den Namen, den der Benutzer in den Excel-Optionen eingetragen hat. Um diese Namen zu ermitteln, schreiben Sie zwei kleine VBA-Makros: Schalten Sie mit Alt+F11 in den VBA-Editor. Markieren Sie im Projekt-Explorer das Projekt der Arbeitsmappe und wählen Sie *Einfügen / Modul*. Geben Sie folgende Codezeilen ein:

```
Declare Function GetUserName Lib "advapi32.dll" _
```

```
Alias "GetUserNameA" (ByVal lpBuffer As String, _
```

```
ByRef nSize As Long) As Long
```

```
Function BenutzerNameWindows()
```

```
Dim IBuffLen As Long
```

```
Dim IAPIResult As Long, stBuff As String * 255
```

```
Application.Volatile
```

```
IBuffLen = 255
```

```
IAPIResult = GetUserName(stBuff, IBuffLen)
```

```
If IBuffLen > 0 Then BenutzerNameWindows = Left$(stBuff, IBuffLen - 1)
```

```
End Function
```

```
Function BenutzerNameExcel()
```

*Application.Volatile*

*BenutzerNameExcel = Application.UserName*

*End Function*

Den Benutzernamen holen Sie anschließend über die soeben definierten Funktionen in eine beliebige Zelle. Wollen Sie z.B. den Windows-Anmeldenamen in Zelle X5000 einfügen, geben Sie dort ein:

*X5000: =BenutzerNameWindows()*

*Um den Excel-Benutzernamen einzufügen, lautet die Formel:*

*X5000: =BenutzerNameExcel()*

Markieren Sie anschließend den zu schützenden Bereich und wählen Sie *Daten / Datenüberprüfung*. Deaktivieren Sie die Option *Leere Zellen ignorieren*. Schalten Sie unter "Zulassen" um auf "Benutzerdefiniert" und tragen Sie folgende Formel ein:

*=X\$1="Max Mustermann"* ("Max Mustermann" steht hier für den Excel-Benutzernamen bzw. den Windows-Anmeldenamen).

### Praxisbeispiel: Kostenerfassung

In folgendem Praxisbeispiel, das Sie in der Beispieldatei zum Artikel finden, sind den einzelnen Projektleitern bestimmte Projekte sowie die zugehörigen Plan- und Ist-Daten der Projektkosten zugewiesen. Die Projektliste ist in den Spalten A-D mit einer Gültigkeitsprüfung vor Überschreibung geschützt, wobei jeder Projektleiter nur die Werte des ihm zugewiesenen Projekts bearbeiten kann. Die Formeln für die Gültigkeitsprüfung lauten:

*E2: =\$H\$1=B2*

*E3: =\$H\$1=B3*

*E4: =\$H\$1=B4*

*usw.*

Bild 3: Windows-Benutzername in der Datenüberprüfung.

## Fehleingaben verhindern

Die Datenüberprüfung lässt sich nicht nur verwenden, um ein versehentliches Überschreiben zu verhindern, sondern auch, um Zellen gezielt vor Fehleingaben zu schützen. Tabelle 1 zeigt, für welchen Zweck Sie welche Formel verwenden können. Markieren Sie immer als erstes den Bereich, auf den sich die Datenüberprüfung beziehen soll und geben Sie dann die jeweilige Formel unter *Datenüberprüfung* / *Zulassen* / *Benutzerdefiniert* ein. Verwenden Sie dabei in der Formel den relativen Bezug zur aktiven Zelle des markierten Bereichs (Die Beispiele in Tabelle 1 gehen davon aus, dass es sich bei Zelle A1 um die aktive Zelle handelt.)

Formel für Gültigkeitskriterium	Zweck
=UND(NICHT(ISTTEXT(A1);A1>0))	Stellen Sie sicher, dass der Benutzer keinen Text, keine Null und keine Minuszahlen einträgt
=ZÄHLENWENN(\$A:\$A;A1)=1	Sorgen Sie dafür, dass der Benutzer keine doppelten Daten in Spalte A einträgt
=NICHT(ISTNV(VERGLEICH(A1;Projektnummern;0)))	Prüfen Sie, ob die Eingabe des Benutzers einem Eintrag im Bereich "Projektnummern" entspricht
=ZELLE("Dateiname")<>"	Lassen Sie keine Eingaben zu, solange die Mappe nicht gespeichert ist (bei neuen Mappen mit dem Pseudonamen Mappe1, Mappe2 ...)

Tabelle 1: Zellschutz mit benutzerdefinierten Gültigkeitskriterien.

## Tabellenblätter schützen

Wenn Ihre Projektkalkulation Tabellenblätter enthält, die nicht für jedermanns Augen bestimmt sind, können Sie diese ausblenden. Verknüpfungen auf Bereiche dieser unsichtbaren Tabellen funktionieren weiterhin.

Die einfachste Methode, ein Tabellenblatt vor dem Benutzer zu verbergen, ist das Ausblenden. Ein Klick mit der rechten Maustaste auf das Tabellenregister öffnet das Kontextmenü, das die Option "Ausblenden" anbietet. Allerdings ist diese Methode nicht besonders effektiv, da sich ausgeblendete Blätter über die Option "Einblenden" im Kontextmenü wieder sichtbar machen lassen.

Wesentlich wirkungsvoller ist der nachfolgend beschriebene Trick, mit dem Sie ein Tabellenblatt sicherer verstecken.

### Tabellen verstecken mit xlVeryHidden

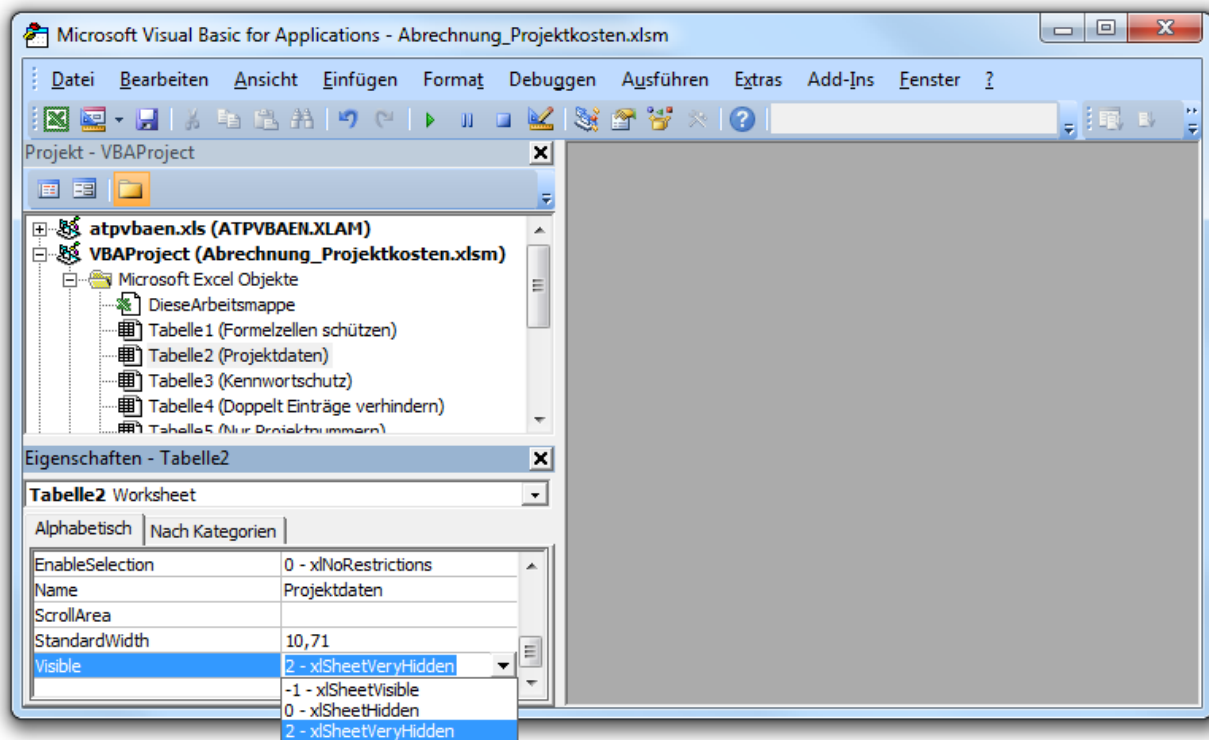


Bild 4: Tabellenblätter verbergen im VBA-Editor.

Aktivieren Sie mit "Alt+F11" den Visual Basic-Editor. Im Menü *Ansicht* finden Sie die Einträge "Projekt-Explorer" und "Eigenschaftenfenster", stellen Sie sicher, dass diese beiden Fenster am linken Rand des Editorbereiches sichtbar sind. Suchen Sie im Projekt-Explorer das Projekt der Arbeitsmappe und markieren Sie das Tabellenblatt. Im Eigenschaftenfenster finden Sie die Eigenschaft "Visible"; stellen Sie diese, wie in Bild 4 gezeigt, auf die Opti-

on 2 ("xlSheetVeryHidden"). Das Tabellenblatt ist damit weder im Excel-Fenster sichtbar noch erscheint es in der Liste der ausgeblendeten Blätter.

Um das Tabellenblatt wieder sichtbar zu machen, stellen Sie die Eigenschaft im VBA-Projekt wieder zurück auf die Option "xlSheetVisible".

Um den Schutz perfekt zu machen, weisen Sie dem VBA-Projekt ein Kennwort zu. Klicken Sie dazu mit der rechten Maustaste auf das Projekt im Projekt-Explorer und wählen Sie Eigenschaften von VBAProject. Kreuzen Sie auf der Registerkarte "Schutz" die Option "Projekt für Anzeige sperren" an und tragen Sie ein Kennwort in die beiden Kennwortfelder ein.

### Tabellenblätter nur zur Ansicht freischalten

Falls Ihre Mappe Tabellenblätter enthält, die der Benutzer zwar sehen, aber nicht bearbeiten darf, können Sie diese mit einem Ereignismakro nur zur Ansicht freischalten:

Aktivieren Sie mit "Alt+F11" den VBA-Editor und suchen Sie im Projekt-Explorer das Tabellenblatt im Projekt der Arbeitsmappe. Klicken Sie es doppelt an, um das Codefenster zu aktivieren. Schalten Sie in der Liste links oben von "Allgemein" auf "Worksheet" um und stellen Sie in der Liste rechts das Ereignis "Activate" ein. Das automatisch eingefügte Makro für das Ereignis "SelectionChange" können Sie gleich wieder löschen.

Ergänzen Sie das Makro für das Activate-Ereignis des Tabellenblatts:

```
Private Sub Worksheet_Activate()
```

```
MsgBox "Nur zur Ansicht"
```

```
Sheets(1).Select
```

```
End Sub
```

Klickt der Benutzer auf das Registerblatt der Tabelle, erhält er eine Nachricht, die er nur mit "OK" bestätigen kann. Danach schaltet Excel wieder auf das erste Tabellenblatt "Sheets(1)" zurück. An Stelle von Sheets(1) können Sie auch den Registernamen eines Blatts angeben, z.B. Sheets "Starttabelle".

Leider bietet das Makroereignis keine Möglichkeit festzustellen, aus welcher Tabelle der Benutzer sich zuschaltet. Sie können sich behelfen, indem Sie das Activate-Ereignis der aufrufenden Tabellenblätter nutzen, um damit den Namen des jeweils aktivierten Blatts in einem Bereichsnamen festzuhalten. Diesen Bereichsnamen liest das aufgerufene Tabellenblatt aus und aktiviert mit seinem Activate-Makro das entsprechende Blatt, um den Benutzer wieder an den Ausgangsort zurückzuführen.



### Praxisbeispiel: Codetabelle

Das Tabellenblatt "Codes" in der Beispieldatei enthält Codenummern für verschiedene Aufgabenbereiche, zum Beispiel Kostenstellen, PSP-Elementlisten, SAP-Transaktionscodes oder wichtige Telefonnummern. Es kann von jedem anderen Tabellenblatt per Hyperlink zur Ansicht aufgerufen werden (im Beispiel von den Tabellenblättern "Dateneingabe 1" und "Dateneingabe 2"), nach Bestätigung einer Meldung schaltet Excel automatisch in das aufrufende Blatt zurück.

Die übrigen Tabellenblätter (hier: "Dateneingabe 1" und "Dateneingabe 2") erhalten Hyperlinks zur Ansteuerung der Codebereiche. Hyperlinks funktionieren in Excel-Tabellen wie auf Internet-Seiten: Ein Klick darauf schaltet auf die verlinkte Seite um. Im Unterschied zu Weblinks können Excel-Hyperlinks auch auf Zellen oder Zellbereiche in Tabellenblättern verweisen (Aktives Dokument). Nach dem Einfügen steht der Hyperlink als Zellbezug in der Zelle, überschreiben Sie ihn einfach mit einem Text Ihrer Wahl.

The screenshot displays two Excel worksheets. The 'Codes' worksheet (top) contains two tables. The first table, 'Kostenstelle', has columns for 'Kostenstelle', 'Beschreibung', 'PSP-Code', and 'Bezeichnung'. The second table, 'Wichtige Telefonnummern', has columns for 'Wichtige Telefonnummern' and 'Bezeichnung'. The 'Datenerfassung' worksheet (bottom) shows a table with columns A through H. In cell G2, there is a blue hyperlink labeled 'Kostenstellen'. In cell G3, there is a blue hyperlink labeled 'PSP-Codes'. In cell G4, there is a blue hyperlink labeled 'Telefonnummern'. A red arrow points to the 'Kostenstellen' hyperlink with the text 'Hyperlink: Codes!A1'.

	A	B	C	D	E
1	Kostenstelle	Beschreibung		PSP-Code	Bezeichnung
2	100-000	Grundstücke und Gebäude		3210-01	Neubau Verwaltungsgebäude Ost
3	100-001	Wasserversorgung		3210-02	Energieversorgung
4	100-002	Stromerzeugung		3210-03	Innenausstattung
5	100-003	Sozialeinrichtungen			
6	100-004	Werkschutz			
7					
8					
9					
10					
11					
12					
13					
14					

	A	B	C	D	E	F	G	H
1								
2							Kostenstellen	
3							PSP-Codes	
4							Telefonnummern	
5								
6								
7								
8								
9								

Bild 5: Codeblatt und Hyperlinks zur Ansteuerung einzelner Bereiche.

Um die Hyperlinks zur Zieltabelle einzufügen, öffnen Sie mit *Einfügen / Hyperlink* den entsprechenden Dialog. Wählen Sie dort unter "Link zu:" die Auswahl "Aktuelles Dokument", geben Sie unter "Zellbezug" den Wert "A1" ein und wählen Sie im Bereich darunter die Tabelle "Codes" aus. Für die Links zu den Tabellen "PSP-Codes" und "Telefonnummern" verfahren Sie entsprechend. Die Zellbezüge lauten hier "D1" bzw. "D8", ausgewählt wird in beiden Fällen wieder die Tabelle "Codes".

Um zu erreichen, dass der Namen des jeweils aktiven Tabellenblattes in Zelle H1 der Tabelle "Code" eingetragen wird, ergänzen Sie das Makro für das Activate-Ereignis mit folgendem Code:

```
Private Sub Worksheet_Activate()
```

```
    Sheets("Codes").Range("H1") = ActiveSheet.name
```

```
End Sub
```

Das Activate-Ereignis mit dem Code wird in das Makroblatt aller Tabellenblätter eingetragen, die einen Link auf das Blatt "Codes" enthalten.

Das Blatt "Codes" erhält ebenfalls ein Ereignismakro, das den Namen der aufrufenden Tabelle aus der Zelle H1 ausliest und diese nach Bestätigung der Meldung mit "OK" wieder ansteuert:

```
Private Sub Worksheet_Activate()
```

```
    MsgBox "Zurück zur Tabelle " & [H1]
```

```
    Sheets(Range("H1").Value).Select
```

```
End Sub
```

### Hat Ihnen dieser Artikel gefallen?

Bewerten Sie ihn im Projekt Magazin online und teilen Sie so Ihre Meinung anderen Lesern mit. Wählen Sie dazu den Artikel im Internet unter [www.projektmagazin.de/ausgaben/2011](http://www.projektmagazin.de/ausgaben/2011) oder klicken Sie [hier](#), um direkt zum Artikel zu gelangen.