Übungseinheit: # 2

Schiebepuzzle

Entwicklungsumgebung: Excel 2007

Kursunterlagen erstell von:

Ing. Harald Mitsch
Josefgasse 6
A 3380 Pöchlarn

TECHNISCHES BÜRO MITSCH
Ihr IT-Betreuer in Pöchlarn
seit 1990 - 0676/588 09 16
http://www.tb-mitsch.at

letzte Aktualisierung: 11. Jänner 2019

INHALTSVERZEICHNIS

1.) Allgemeines

Ziel dieser Übung ist es, die in der vorherigen Übungseinheit #1 erworbenen Kenntnisse zur Spieleprogrammierung in Excel, in einem ersten Spiel anzuwenden. Ein Schiebepuzzle lässt sich relativ einfach umsetzen und bietet trotzdem ein hohes Maß an Spielspaß.

2.) Spielregeln

Das Spielbrett besteht aus einem Quadrat mit 4 mal 4 Feldern. Auf den einzelnen Feldern befinden sich die 15 Puzzleteile welche mit den Ziffern 1 bis 15 beschriftet sind. Das 16. Feld ist leer. Die Puzzleteile können nun aus allen Richtungen auf das leere Feld gezogen werden. Damit ergibt sich die Möglichkeit ein anderes Teil auf das neue leere Feld zu verschieben. Der Spielablauf besteht darin, zuerst alle Teile zu mischen und dann, mit möglichst wenigen Zügen, die korrekte Reihenfolge der Ziffern von 1 bis 15 wieder herzustellen. Zusätzlich kann man versuchen die Ziffern in die Reigenfolge 15 bis 1 zu bringen oder von 1 bis 15 - jedoch in senkrechten Spalten.

3.) Überlegungen zum Spielablauf

Dem Anwender sollen folgende Funktionen zur Verfügung gestellt werden um das Spiel spielen zu können. Er soll auf eine Schaltfläche klicken können um die Puzzleteile zu mischen. Eine weitere um das Puzzle aufzulösen, also die Steine wieder in die richtige Reihenfolge zu bringen. Das Verschieben eines Puzzleteils soll durch einen Doppelklick auf einen Spielstein realisiert werden. Dabei wird der Stein auf das angrenzende leere Feld (wenn vorhanden) verschoben. Gleichzeitig wollen wir dem Spieler anzeigen, wie viele Züge er bereits gemacht hat. Das Spiel wird beendet, sobald die richtige Reigenfolge aller Puzzleteile wieder hergestellt worden ist.

4.) Programmieren des Spiels

4.1.) Vorbereitungen

Zuerst legen wir uns einen neuen Ordner an in welchem wir alle für das Spiel benötigte Dateien ablegen. Dann kopieren wir die Datei: **Muster_Vorlage.xlsm** (die fertige Excel Datei aus der Übungseinheit #1) in diesen Ordner. Von derselben Datei legen wir uns eine weitere Kopie an und benennen sie um z.B. auf: **Schiebepuzzle.xlsm**. In dieser Datei werden wir das Spiel programmieren. Sollten wir dabei auf Funktionen oder andere Elemente stoßen, welche wir bei anderen Spielen ebenfalls gebrauchen könnten, dann werden wir diese zusätzlich auch in der Mustervorlage hinterlegen.

4.2.) Gestaltung des Spielbretts

Unser Spiel soll im Tabellenblatt "**Spielfeld**" ablaufen. Benennen wir es jetzt um auf "**Schiebepuzzle**". Sollten noch weitere Tabellenblätter vorhanden sein, dann diese einfach löschen.

Dann passen wir noch die bereits vorhandenen globalen Konstanten im Modul Globales an:

```
'--------------------------------------------------------------------------/
'-  Globale Konstanten und Variablen - fuer alle Spiele benoetigt          -/
'--------------------------------------------------------------------------/
```

```
Global Const PROGRAMMNAME = "Schiebe-Puzzle"          'Name des Programmes.
                                                      'Name des Programmierers:
Global Const PROGRAMMAUTOR = "© 2019 by TBM - Technisches Büro Mitsch" '(c) = Asc(184)
Global Const TABELLENBLATT = "Schiebepuzzle"          'Tabellenblatt fuer Spielfeld.
Global oTabellenblatt                    As Object    'Variable fuer Tabellenblatt.

                                                      'Verhalten beim STARTEN:
Global Const SPIELANSICHT = NEIN                      'Waehrend der Spielentwicklung.
'Global Const SPIELANSICHT = JA                       'Nach Fertigstellung des Spiels.
                                                      'Verhalten beim BENDEN:
'Global Const AUTOSPEICHERN = JA                      'Beim Beenden autom. alles speichern.
Global Const AUTOSPEICHERN = NEIN                     'Aenderungen automatisch Verwerfen
'Global Const FRAGESPEICHERN = NEIN                   'wenn auch FRAGESPEICHERN = NEIN oder
Global Const FRAGESPEICHERN = JA                      'zusaetzliche Nachfrage ob Speichern.
```

Bei der Gestaltung des Spielbretts können Sie Ihrer Phantasie freien Lauf lassen. Hier als Beispiel mein Vorschlag:

Spielstein 16 muß während man spielt natürlich ein Leerfeld sein. Man kann jedoch bei Spielende die 16 eintragen lassen, damit der Spieler keine weiteren Züge mehr machen kann, da dann kein angrenzendes Leerfeld mehr vorhanden ist.

Für die Schaltflächen verwende ich keine **Steuerelemente** sondern **Formen** (aus dem Excel-Menüband: **Einfügen**) die man beliebig gestalten kann und später lediglich ein Makro zugewiesen bekommen.

Um die aktuelle Anzahl der Spielzüge einfacher ausgeben zu können, vergeben wir für diese Zelle (egal ob die Zelle nur einen Zahlenwert beinhaltet oder zusammen mit einem beschreibenden Text) den Namen: **ZUEGE**. (Siehe dazu auch Punkt 8.) Zellen mit Namen versehen (Namens-Manager) in der Dokumentation für die Übungseinheit #1.)

Bevor wir mit dem eigentlichen Programmieren beginnen, machen wir uns noch einige Gedanken darüber, wie wir den Programmablauf gestalten könnten, welche Funktionen wir benötigen und in welchen Modulen wir die einzelnen Programmcode hinterlegen können.

Spielbrett automatisch gestalten:

Wer sein Spielbrett genauso gestalten möchte wie in den Kursunterlagen kann nachfolgende Funktion verwenden. Auch wenn Sie eine andere Aufteilung der Puzzleteile verwenden möchten, z.B. 8 x 6, können Sie diese Funktion leicht anpassen.

```
Public Function SpielfeldZeichnen() As Boolean
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION                        */
'*************************************************************************/
'*-----------------------------------------------------------------------*/
'*                                                                       */
'*      FUNKTION:       SpielfeldZeichnen                                */
'*                                                                       */
'*      BESCHREIBUNG:   Gestaltet das Spielbrett fuer 4 x 4 Spielsteine  */
'*                      fuer das Spiel: Schiebe-Puzzle.                  */
'*                                                                       */
'*      RETURN          Boolean ..... True = Funktion in Ordnung         */
'*                                    False = Fehler in Funktion         */
'*      PARAMETER:      keine                                            */
'*                                                                       */
'*-----------------------------------------------------------------------*/
On Error GoTo Fehler                            'Bei Fehler zur Fehlermarke.
SpielfeldZeichnen = True                        'Returnwert = alles OK setzen.

Range("A1:G1").Select:  Selection.RowHeight = 30    'Rahmenhoehe oben.
Range("A2:G5").Select:  Selection.RowHeight = 100   'Hoehe 4 Spielsteine.
Range("A6:G6").Select:  Selection.RowHeight = 30    'Rahmenhoehe unten.

Columns("A:A").Select:  Selection.ColumnWidth = 5   'Rahmenbreite links.
Columns("B:E").Select:  Selection.ColumnWidth = 19  'Breite 4 Spielsteine.
Columns("F:F").Select:  Selection.ColumnWidth = 5   'Rahmenbreite rechts.

Columns("G:G").Select:  Selection.ColumnWidth = 10  'Abstand zum Steuerblock.
Columns("H:H").Select:  Selection.ColumnWidth = 60  'Breite Steuerblock.

Range("A1:F6").Select                           'Spielbrett mit Rahmen:
Selection.Interior.Color = SCHWARZ              'Hintergrundfarbe.
Selection.Font.Color = SCHWARZ                  'Schriftfarbe.

Range("B2:E5").Select                           'Spielbrett 4 x 4.
Selection.Borders.LineStyle = xlContinuous      'Durchgezogene Linie.
Selection.Borders.Weight = xlThick              'Dicke Linie.
Selection.Borders.Color = SCHWARZ               'Linienfarbe.
Selection.Font.Color = GOLD                     'Schriftfarbe.
Selection.Font.Name = "Arial"                   'Schriftart.
Selection.Font.Size = 72                        'Schriftgroesse.
Selection.Font.Bold = True                      'Fettschrift aktivieren.
Selection.HorizontalAlignment = xlCenter        'In beide Richtungen die
Selection.VerticalAlignment = xlCenter          'Ausrichtung zentrieren.

Range("B2,D2,C3,E3,B4,D4,C5,E5").Select         'Ungerade Zahlen:
Selection.Interior.Color = ROT                  'Hintergrundfarbe.

Range("C2,E2,D3,B3,C4,E4,D5,B5").Select         'Gerade Zahlen:
Selection.Interior.Color = WEISS                'Hintergrundfarbe.

Range("B2").Select:  ActiveCell.FormulaR1C1 = "1"   'Die Ziffern eintragen:
Range("C2").Select:  ActiveCell.FormulaR1C1 = "2"
Range("D2").Select:  ActiveCell.FormulaR1C1 = "3"
Range("E2").Select:  ActiveCell.FormulaR1C1 = "4"
Range("B3").Select:  ActiveCell.FormulaR1C1 = "5"
Range("C3").Select:  ActiveCell.FormulaR1C1 = "6"
Range("D3").Select:  ActiveCell.FormulaR1C1 = "7"
Range("E3").Select:  ActiveCell.FormulaR1C1 = "8"
Range("B4").Select:  ActiveCell.FormulaR1C1 = "9"
```

```
Range("C4").Select:  ActiveCell.FormulaR1C1 = "10"
Range("D4").Select:  ActiveCell.FormulaR1C1 = "11"
Range("E4").Select:  ActiveCell.FormulaR1C1 = "12"
Range("B5").Select:  ActiveCell.FormulaR1C1 = "13"
Range("C5").Select:  ActiveCell.FormulaR1C1 = "14"
Range("D5").Select:  ActiveCell.FormulaR1C1 = "15"
Range("E5").Select:  ActiveCell.FormulaR1C1 = "16"

Range("A1").Select                                  'Zelle A1 selektieren.

                                                    'Fehlerbehandlungsroutine:
Beenden:                                            '========================
  Application.Cursor = xlDefault                    'Sanduhr ausblenden.
  Exit Function                                     'Funktion abbrechen.

Fehler:                                             'Fehlermarke - Fehlerbehandlung:
  SpielfeldZeichnen = False                         'Returnwert = Fehler setzen.
  MsgBox Err.Description                            'Fehlermeldung ausgeben.
  Resume Beenden                                    'Bei Funktionsende weitermachen.
  Resume                                            'Nur fuer Testzwecke hinterlegt.
End Function                                        'Funktionsende.
```

Zu obigen Programmcode ist folgendes anzumerken:

Wenn Sie das Schiebepuzzle so programmieren möchten, daß der Spieler selbst festlegen kann, wie sein Spielfeld aufgeteilt sein soll, dann müssen globale Variablen definiert werden in welchen die Anzahl der Puzzleteile waagrecht und senkrecht hinterlegt sind.

Das Eintragen der Zahlen sollte nicht in einzelnen Programmzeilen erfolgen sondern über eine For-Schleife abgehandelt werden, in welcher mit Hilfe des **Mod** Operators unterschieden wird, ob es sich um eine gerade oder ungerade Zahl handelt (10 Mod 2 liefert den Restwert der Division 10 / 2, also 0 = gerade Zahl, 5 Mod 2 ergibt 1 = ungerade Zahl).

```
Dim i                   As Integer                  'Allgemeine Zaehlvariable.
Dim iZeileStart         As Integer                  'Zeile aktueller Puzzleteil.
Dim iSpalteStart        As Integer                  'Spalte aktueller Puzzleteil.

Set oTabellenblatt = Sheets(TABELLENBLATT)          'Akt. Tabellenblatt festlegen.
oTabellenblatt.Select                               'Tabellenblatt auswaehlen.
Range("A1").Select                                  'Zelle A1 selektieren.

iZeileStart = 2                                     'Startposition fuer ersten
iSpalteStart = 2                                    'Puzzleteil festlegen.
For i = 1 To 16
  If (i Mod 2) + ((Int((i - 1) / 4)) Mod 2) = 1 Then '1, 3, 6, 8, 9, 11, 14, 16
    oTabellenblatt.Cells(iZeileStart, iSpalteStart).Interior.Color = ROT
  Else                                              '2, 4, 5, 7, 10, 12, 13, 15:
    oTabellenblatt.Cells(iZeileStart, iSpalteStart).Interior.Color = WEISS
  End If
  oTabellenblatt.Cells(iZeileStart, iSpalteStart) = i   'Zahl eintragen und weiter
  iSpalteStart = iSpalteStart + 1                   'zur naechste Position:
  If iSpalteStart > 5 Then iSpalteStart = 2: iZeileStart = iZeileStart + 1
Next i
```

Weiters empfiehlt es sich die Spalten A bis Z für das Spielfeld zu reservieren und die Spalten ab AA für die Ausgabe zusätzlicher Texte, Schaltflächen, Spielbeschreibung, etc. heranzuziehen. Die für das Spielfeld nicht benötigten Spalten können dann ausgeblendet werden.

Wer sich über weitere Gestaltungsmöglichkeiten für ein Schiebepuzzle interessiert sei auf die Dateien: Schiebe_Puzzle.mdb bzw. Schiebe_Puzzle.mde verwiesen (erstellt mit Access 2007).

4.3.) Überlegungen zum Programmablauf

a) Beim Öffnen der Datei soll das Spielbrett entweder mit dem zuletzt gespeicherten Spielstand starten oder mit einem gelösten Puzzle. Das legen wir beim Schließen der Excel-Datei fest, indem wir die Datei automatisch speichern lassen und in einer globalen Konstante **SPIELSTART_NEU** = JA (gelöstes Spielbrett wird angezeigt) oder = NEIN (beim letzten Spielstand weitermachen). Unsere Prozedur **Workbook_Open** ruft am Ende die Funktion **SpielStarten** im Modul **Funktionen** auf. In dieser Funktion entscheiden wir dann ob das Spiel zu Beginn gelöst sein soll oder nicht. Abhängig davon müssen wir auch die Anzahl der Spielzüge rücksetzen oder nicht.

b) Ein Klick auf die Schaltflächen soll die entsprechende Funktion ausführen. Dazu legen wir uns im Modul **Funktionen** die Funktion **SpielMischen** und **SpielAufloesen** an. Weiters werden wir zwei Makros erstellen, welche diese Funktionen aufrufen. Diese Makros weisen wir dann den Schaltflächen bzw. den "Formen" zu.

c) Bei Doppeklick auf einen Spielstein soll dieser, falls möglich, auf das leere Spielfeld verschoben werden. Dazu erstellen wir im Klassenmodul **Tabelle1 (Schiebepuzzle)** die Prozedur **Worksheet_BeforeDoubleClick**. Darin lassen wir dann den Spielstein verschieben.

d) Die Anzahl die Spielzüge muß aktualisiert und ausgegeben werden. Wir merken uns die Anzahl in der globalen Variable **iAnzahlZuege**. Am Besten gleich in der Prozedur **Worksheet_BeforeDoubleClick** nach jedem Verschieben des Steines die Anzahl der Spielzüge erhöhen und im Tabellenblatt ausgeben.

e) Bei Spielende entweder den 16. Spielstein ausgeben oder nichts machen. Dazu legen wir uns eine Funktion **SpielEnde** an, welche überprüft, ob sich alle Steine an der richtigen Position befinden. Wenn ja, dann setzen wir entweder den 16. Stein oder nicht. Das können wir über eine weitere globale Konstante **SPIELSTEIN_16** = JA oder = NEIN festlegen. Augerufen wird die Funktion **SpielEnde** jedesmal nachdem ein Stein verschoben wurde (in der Prozedur **Worksheet_BeforeDoubleClick**).

f) Beim Beenden des Spiels aktuellen Zustand speichern oder nicht - abhängig davon, was beim erneuten Öffnen der Excel-Datei passieren soll. Wir werden die Excel-Datei automatisch speichern lassen und entscheiden erst beim nächsten Öffnen der Spieldatei was tatsächlich geschehen soll (abhängig von der globalen Konstante: **SPIELSTART_NEU**).

4.4.) Globale Konstanten

Zunächst legen wir uns im Modul **Globales** die globalen Konstanten anhand unserer Überlegungen an:

```
'--------------------------------------------------------------------------/
'-  Globale Konstanten und Variablen - fuer das aktuelle Spiel benoetigt    -/
'--------------------------------------------------------------------------/
                                                  'Verhalten beim STARTEN:
Global Const SPIELSTART_NEU = JA                  'Beim Oeffnen geloestes Spiel anzeigen.
'Global Const SPIELSTART_NEU = NEIN               'Mit letztem Spielstand weiterspielen.
Global Const SPIELSTEIN_16 = JA                   'Bei Spielende 16. Stein anzeigen.
'Global Const SPIELSTEIN_16 = NEIN                'Bei Spielende 16. Stein nicht anzeigen.
Global iAnzahlZuege                    As Integer 'Anzahl der aktuellen Spielzuege.
```

Dann kontrollieren wir noch folgende globalen Konstanten, um das Öffnen und Schließen der Datei nach unseren Wünschen ablaufen zu lassen:

```
                                                  'Verhalten beim BEENDEN:
Global Const AUTOSPEICHERN = JA                   'Beim Beenden autom. alles speichern.
'Global Const AUTOSPEICHERN = NEIN                'Aenderungen automatisch Verwerfen
Global Const FRAGESPEICHERN = NEIN                'wenn auch FRAGESPEICHERN = NEIN oder
'Global Const FRAGESPEICHERN = JA                 'zusaetzliche Nachfrage ob Speichern.
```

4.5.) Funktionen

Legen wir uns nun vorab im Modul **Funktionen** folgenden Funktionen an:

```
Public Function SpielStarten()
  MsgBox "SpielStarten"
End Function                                          'Funktionsende.

Public Function SpielAufloesen()
  MsgBox "SpielAufloesen"
End Function                                          'Funktionsende.

Public Function SpielMischen()
  MsgBox "SpielMischen"
End Function                                          'Funktionsende.

Public Function SpielEnde()
  MsgBox "SpielEnde"
End Function                                          'Funktionsende.
```

Sich solche einfache Funktionen gleich am Anfang zu hinterlegen bietet den Vorteil, daß man diese Funktionen beim Programmieren einer Funktion bereits aufrufen kann, auch wenn diese noch nicht fertig ausprogrammiert wurde.

4.5.1.) Funktion: SpielStarten

In der Funktion SpielStarten bestimmen wir nun den Anfangszustand des Spieles nach dem Öffnen der Excel-Datei. Wenn wir die globale Konstante **SPIELSTART_NEU** auf JA gesetzt haben, dann würde ein gelöstes Spielbrett angezeigt werden, andernfalls wird das Spielbrett wie beim letzten Schließen der Datei angezeigt.

```
If SPIELSTART_NEU = JA Then                           'Geloestes Spiel anzeigen lassen:
  Call SpielAufloesen                                 'Spiel aufloesen lassen.
End If
```

Nachdem wir die Funktion mit Rückgabewert und Fehlerbehandlung erweitert und dokumentiert haben könnte die fertige Funktion in etwa so aussehen:

```
Public Function SpielStarten() As Boolean
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION                        */
'*************************************************************************/
'*-----------------------------------------------------------------------*/
'*                                                                       */
'*      FUNKTION:       SpielStarten                                     */
'*                                                                       */
'*      BESCHREIBUNG:   Wurde die globale Konstante SPIELSTART_NEU im Modul */
'*                      Globales auf JA gesetzt, dann wird beim Oeffnen der */
'*                      Datei ein geloestes Schiebepuzzle angezeigt. Bei */
'*                      NEIN kann der Spieler weitermachen wo er aufgehoert */
'*                      hat. Oder man koennte das Spielbrett neu mischen. */
'*                                                                       */
'*      RETURN          Boolean ..... True = Funktion in Ordnung         */
'*                                    False = Fehler in Funktion         */
'*      PARAMETER:      keine                                            */
'*                                                                       */
'*-----------------------------------------------------------------------*/
```

```
On Error GoTo Fehler                          'Bei Fehler zur Fehlermarke.
SpielStarten = True                           'Returnwert = alles OK setzen.

If SPIELSTART_NEU = JA Then                   'Geloestes Spiel anzeigen lassen:
  Call SpielAufloesen                         'Spiel aufloesen lassen.
Else                                          'Weiter beim letzten Spielstand:
  'nix machen.                                'Normalerweise nichts machen, vielleicht
  'Call SpielMischen                          'aber, je nach Belieben, Mischen.
End If

                                              'Fehlerbehandlungsroutine:
Beenden:                                      '=========================
  Application.Cursor = xlDefault              'Sanduhr ausblenden.
  Exit Function                               'Funktion abbrechen.

Fehler:                                       'Fehlermarke - Fehlerbehandlung:
  SpielStarten = False                        'Returnwert = Fehler setzen.
  MsgBox Err.Description                      'Fehlermeldung ausgeben.
  Resume Beenden                              'Bei Funktionsende weitermachen.
  Resume                                      'Nur fuer Testzwecke hinterlegt.
End Function                                  'Funktionsende.
```

4.5.2.) Funktion: SpielAufloesen

Mit dieser Funktion soll das Spiel gelöst werden – also die Ziffern 1 bis 16 in die richtige Reihenfolge gebracht werden. Dabei ist darauf zu achten, dass auch die korrekte Hintergrundfarbe eingetragen wird. Bei einem Spielbrett von 3x3 oder 5x5 Steinen, bzw. wenn die Anzahl der waagrechten Spielsteine ungerade ist, dann erhalten alle geraden und ungeraden Zahlen dieselbe Hintergrundfarbe. Da wir aber ein Spielbrett mit 4x4 Steinen haben müssen wir zusätzlich feststellen, in welcher Zeile ein Spielstein liegt. Das können wir mit **Int((i - 1) / 4)) Mod 2** (ergibt Zeile 0 bis 3) berechnen. Addieren wir dazu den Wert **i Mod 2** (gerade oder ungerade Zahl) dann erhalten wir für die Farbe des ersten Steins immer 1, die dazwischen liegenden Werte betragen 0 oder 2.

Hier nun die fertige Funktion:

```
Public Function SpielAufloesen() As Boolean
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION                        */
'*************************************************************************/
'*----------------------------------------------------------------------*/
'*                                                                      */
'*      FUNKTION:       SpielAufloesen                                  */
'*                                                                      */
'*      BESCHREIBUNG:   Gibt die Ziffern 1 bis 16 in der richtigen Reihen-  */
'*                      folge aus und setzt die entsprechende Hintergrund-  */
'*                      farbe. Zum Schluss wird der letzte Spielstein in    */
'*                      Abhaengigkeit der globalen Konstante SPIELSTEIN_16  */
'*                      als Spielstein oder als Leerfeld eingetragen.       */
'*                                                                      */
'*      RETURN          Boolean ..... True = Funktion in Ordnung        */
'*                                    False = Fehler in Funktion        */
'*      PARAMETER:      keine                                           */
'*                                                                      */
'*----------------------------------------------------------------------*/
Dim i                   As Integer              'Allgemeine Zaehlvariable.
Dim iZeileStart         As Integer              'Zeile aktueller Puzzleteil.
Dim iSpalteStart        As Integer              'Spalte aktueller Puzzleteil.
Dim lFarbeAktuell       As Long
```

```
On Error GoTo Fehler                                    'Bei Fehler zur Fehlermarke.
SpielAufloesen = True                                   'Returnwert = alles OK setzen.

Set oTabellenblatt = Sheets(TABELLENBLATT)              'Akt. Tabellenblatt festlegen.
oTabellenblatt.Select                                   'Tabellenblatt auswaehlen.
iZeileStart = 2                                         'Startposition fuer ersten
iSpalteStart = 2                                        'Puzzleteil festlegen.

For i = 1 To 16                                         'Die 16 Puzzelteile abarbeiten:
  'If (i Mod 2) = 1 Then                                'Nur wenn Anzahl waagrechte
                                                        'Steine ungerade ist, sonst:
  If (i Mod 2) + ((Int((i - 1) / 4)) Mod 2) = 1 Then '1, 3, 6, 8, 9, 11, 14, 16
    oTabellenblatt.Cells(iZeileStart, iSpalteStart).Interior.Color = ROT
  Else                                                  '2, 4, 5, 7, 10, 12, 13, 15:
    oTabellenblatt.Cells(iZeileStart, iSpalteStart).Interior.Color = WEISS
  End If

  oTabellenblatt.Cells(iZeileStart, iSpalteStart) = i 'Die Zahl eintragen und
  iSpalteStart = iSpalteStart + 1                       'zur naechste Position:
  If iSpalteStart > 5 Then iSpalteStart = 2: iZeileStart = iZeileStart + 1
Next i

If SPIELSTEIN_16 = JA Then                              '16. Stein einblenden lassen:
  Range("E5").Select:  ActiveCell.FormulaR1C1 = "16" 'Wenn JA, dann 16 eintragen
  Selection.Interior.Color = ROT                        'und Hintergrund ROT machen.
Else                                                    '16. Spielstein nicht einblenden:
  Range("E5").Select:  ActiveCell.FormulaR1C1 = Null 'dann NULL-Wert eintragen
  Selection.Interior.Color = SCHWARZ                    'und Hintergrund SCHWARZ machen.
End If
Range("A1").Select                                      'Zelle A1 selektieren.


                                                        'Fehlerbehandlungsroutine:
Beenden:                                                '=========================
  Application.Cursor = xlDefault                        'Sanduhr ausblenden.
  Exit Function                                         'Funktion abbrechen.

Fehler:                                                 'Fehlermarke - Fehlerbehandlung:
  SpielAufloesen = False                                'Returnwert = Fehler setzen.
  MsgBox Err.Description                                'Fehlermeldung ausgeben.
  Resume Beenden                                        'Bei Funktionsende weitermachen.
  Resume                                                'Nur fuer Testzwecke hinterlegt.
End Function                                            'Funktionsende.
```

Am Ende der Funktion überprüfen wir noch die globale Konstante **SPIELSTEIN_16**, welche festlegt, ob der letzte Spielstein eingetragen werden soll oder als Leerfeld darzustellen ist.

4.5.3.) Funktion: SpielMischen

Wer nun denkt man könnte das Mischen einfach per Zufallsgenerator erledigen, indem man die Zahlen in eine zufällige Reihenfolge bringt und dann neu anzeigen lässt, der wird später beim Spielen sein blaues Wunder erleben. Es sind nämlich nicht alle Zahlenverteilungen lösbar. Vertauscht man zum Beispiel beim Aufbau des Spielbrettes die Zahlen 14 und 15, dann kann das Puzzle nicht gelöst werden. Genauso ist es unmöglich bei korrektem Anfang die Zahlen in die Reihenfolge von 15 bis 1 zu bringen.

Probieren Sie es später einmal aus – nachdem wir das Spiel fertig programmiert haben!

Daher müssen wir uns einen anderen Programmablauf zurechtlegen – trotzdem soll das Mischen zufällig gestaltet werden.

Das gestalten wir am besten so, wie man es auch manuell mit dem Spiel in der Hand machen würde: Immer wieder wahllos, ohne zu Überlegen, einen Puzzleteil nach dem anderen auf das neue leere Feld verschieben bis man denkt, jetzt ist das Puzzle gut genug durchgemischt und man mit dem Lösen beginnen will.

Wie könnte nun der Ablauf dieser Mischfunktion aussehen. Zuerst müssen wir sicherstellen, dass das Spielbrett sich im Ausgangszustand befindet. Das erledigen wir über die Funktion SpielAufloesen. Danach machen wir den 16. Spielstein zum Leerfeld. Nun bestimmen wir einen zufälligen Spielstein und kontrollieren ob er sich neben dem Leerfeld befindet. Wenn ja, dann verschieben wir den Stein, wenn nicht, dann eben nicht. Danach bestimmen wir wieder zufällig einen neuen Stein. Um das Mischen auch abschließen zu können, zählen wir die Anzahl der Steine die verschoben wurden. Ist die vorgegebene Anzahl der Verschiebungen erreicht, dann ist auch das Mischen beendet.

Vorab wieder die vollständige Funktion mit der erweiterten Möglichkeit, dem Spieler beim Mischen zuschauen zu lassen falls die Variable **bZuschauen=JA**. Es werden **200** Verschiebungen durchgeführt.

```
Public Function SpielMischen()
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION                        */
'*************************************************************************/
'*-----------------------------------------------------------------------*/
'*                                                                       */
'*      FUNKTION:       SpielMischen                                     */
'*                                                                       */
'*      BESCHREIBUNG:   Mischt die Teile des Schiebepuzzle durch, gibt das  */
'*                      Leerfeld aus und setzt die Anzahl der Spielzuege */
'*                      auf 0.                                           */
'*                                                                       */
'*      RETURN          Boolean ..... True = Funktion in Ordnung         */
'*                                    False = Fehler in Funktion         */
'*      PARAMETER:      keine                                            */
'*                                                                       */
'*-----------------------------------------------------------------------*/
Dim iZeile              As Integer                  'Aktuelle Zeile.
Dim iSpalte             As Integer                  'Aktuelle Spalte
Dim iZeileNeu           As Integer                  'Neue Zeile.
Dim iSpalteNeu          As Integer                  'Neue Spalte.
Dim bZuschauen          As Boolean                  'Wenn JA, dann kann der Spieler
                                                    'das Mischen optisch verfolgen,
                                                    'sonst nicht.
On Error GoTo Fehler                                'Bei Fehler zur Fehlermarke.
SpielMischen = True                                 'Returnwert = alles OK setzen.
bZuschauen = JA                                     'Zuschauen lassen oder nicht:

Set oTabellenblatt = Sheets(TABELLENBLATT)          'Akt. Tabellenblatt festlegen.
oTabellenblatt.Select                               'Tabellenblatt auswaehlen.

If bZuschauen = NEIN Then                           'Je nachdem wie es beliebt:
  Application.ScreenUpdating = False                'Bildschirmaktualisierung
End If                                              'ausschalten oder nicht.
                                                    'Zuerst Spiel aufloesen lassen
Call SpielAufloesen                                 'damit der 16. Stein auf E5 ist.
Range("E5").Select:  ActiveCell.FormulaR1C1 = Null  'Dort einen NULL-Wert eintragen
Selection.Interior.Color = SCHWARZ                  'und Hintergrund SCHWARZ machen.
Range("A1").Select                                  'Zelle A1 selektieren.
```

```
While iAnzahlZuege < 200                                '200 Verschiebungen durchfuehren:
  iZeile = 1 + Int(4 * Rnd) + 1                         'Zufaellige Zeile und
  iSpalte = 1 + Int(4 * Rnd) + 1                        'zufaellige Spalte festlegen.
  If Not (IsNull(oTabellenblatt.Cells(iZeile, iSpalte))) Then 'Nur wenn zufaellige
    iZeileNeu = 0                                       'Zelle nicht das Leerfeld ist:
    iSpalteNeu = 0                                      'Neue Position vordefinieren.

    'Wenn sich das Leerfeld in einer der vier Richtungen direkt neben dem zufaelligen
    'Spielstein befindet, dann die neue Position des Steines festlegen.
    'Normalerweise wuerde es genuegen, die Hintergrundfarbe abzufragen.
    'Da hier jedoch sowohl das Leerfeld als auch der Rahmen um das Spielfeld die
    'selbe Hintergrundfarbe haben, ist es notwendig zusaetzlich die Schriftfarbe
    'abzufragen. Daher zu Beginn sicherstellen, dass die Schriftfarbe im Rahmen
    '(aktuell SCHWARZ) ungleich der Schriftfarbe im Spielbrett (aktuell GOLD) ist!
    If oTabellenblatt.Cells(iZeile + 0, iSpalte + 1).Interior.Color = SCHWARZ And
oTabellenblatt.Cells(iZeile + 0, iSpalte + 1).Font.Color = GOLD Then iZeileNeu =
iZeile + 0: iSpalteNeu = iSpalte + 1
    If oTabellenblatt.Cells(iZeile + 0, iSpalte - 1).Interior.Color = SCHWARZ And
oTabellenblatt.Cells(iZeile + 0, iSpalte - 1).Font.Color = GOLD Then iZeileNeu =
iZeile + 0: iSpalteNeu = iSpalte - 1
    If oTabellenblatt.Cells(iZeile + 1, iSpalte + 0).Interior.Color = SCHWARZ And
oTabellenblatt.Cells(iZeile + 1, iSpalte + 0).Font.Color = GOLD Then iZeileNeu =
iZeile + 1: iSpalteNeu = iSpalte + 0
    If oTabellenblatt.Cells(iZeile - 1, iSpalte + 0).Interior.Color = SCHWARZ And
oTabellenblatt.Cells(iZeile - 1, iSpalte + 0).Font.Color = GOLD Then iZeileNeu =
iZeile - 1: iSpalteNeu = iSpalte + 0

    If iZeileNeu <> 0 And iSpalteNeu <> 0 Then          'Nur wenn Leerfeld neben Stein:
                                                        'Nummer des Steines eintragen:
      oTabellenblatt.Cells(iZeileNeu, iSpalteNeu) = oTabellenblatt.Cells(iZeile,
iSpalte)
                                                        'Hintergrundfarbe uebernehmen:
      oTabellenblatt.Cells(iZeileNeu, iSpalteNeu).Interior.Color =
oTabellenblatt.Cells(iZeile, iSpalte).Interior.Color
      oTabellenblatt.Cells(iZeile, iSpalte) = Null      'Neues Leerfeld aktualisieren:
      oTabellenblatt.Cells(iZeile, iSpalte).Interior.Color = SCHWARZ
      If bZuschauen = JA Then Sleep (100)               'Beim Zuschauen etwas warten.
      iAnzahlZuege = iAnzahlZuege + 1                   'Anzahl der zufaelligen
    End If                                              'Verschiebungen erhoehen.
  End If
Wend
iAnzahlZuege = 0                                        'Anzahl Spielzuege ruecksetzen.
oTabellenblatt.Range("ZUEGE") = "Anzahl Züge bisher:  0000" 'und ausgeben lassen.

Application.ScreenUpdating = True                       'Bildschirmaktualisierung auf
                                                        'alle Faelle wieder einschalten.


                                                        'Fehlerbehandlungsroutine:
Beenden:                                                '=========================
  Application.Cursor = xlDefault                        'Sanduhr ausblenden.
  Exit Function                                         'Funktion abbrechen.

Fehler:                                                 'Fehlermarke - Fehlerbehandlung:
  SpielMischen = False                                  'Returnwert = Fehler setzen.
  MsgBox Err.Description                                'Fehlermeldung ausgeben.
  Resume Beenden                                        'Bei Funktionsende weitermachen.
  Resume                                                'Nur fuer Testzwecke hinterlegt.
End Function                                            'Funktionsende.
```

4.5.4.) Funktion: SpielEnde

In dieser Funktion überprüfen wir, ob sich alle Zahlen in der richtigen Reihenfolge befinden. Dazu gehen wir die 15 Positionen der Puzzleteile durch und kontrollieren ob die darin enthaltene Zahl korrekt ist. Sobald eine Zahl nicht übereinstimmt wird die Funktion abgebrochen und liefert als Rückgabewert **False** – das Spiel ist noch nicht beendet. Wenn die richtige Reihenfolge vorhanden, dann wird am Ende der Funktion noch der 16. Spielstein eingeblendet, falls das in der globalen Konstante hinterlegt ist.

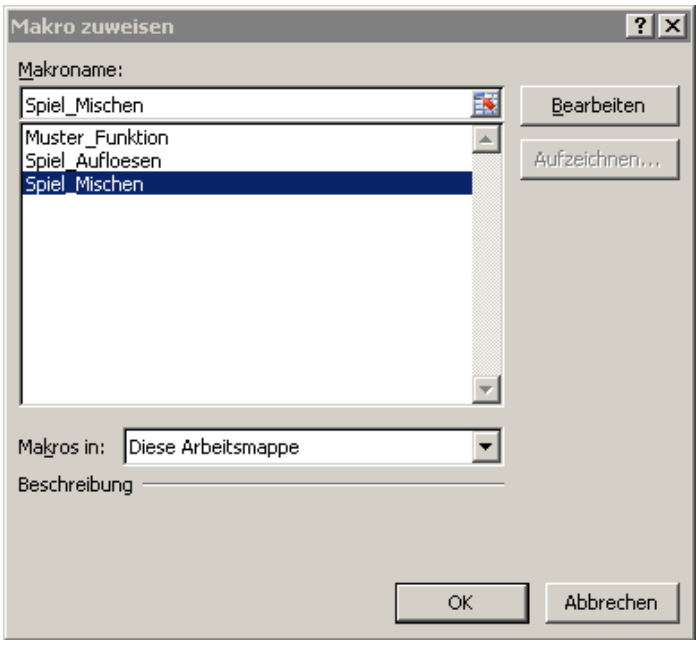
```
Public Function SpielEnde() As Boolean
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION                        */
'*************************************************************************/
'*-----------------------------------------------------------------------*/
'*                                                                       */
'*      FUNKTION:       SpielEnde                                        */
'*                                                                       */
'*      BESCHREIBUNG:   Ueberprueft ob sich alle Puzzleteile an der      */
'*                      richtigen Stelle befinden. Wenn ja, dann wird am */
'*                      Ende noch ueberprueft ob der 16. Spielstein      */
'*                      einzublenden ist, wenn ja dann wird er eingeblendet.*/
'*                                                                       */
'*      RETURN          Boolean ..... True = Spiel ist zu Ende           */
'*                                    False = Spiel noch nicht zu Ende   */
'*      PARAMETER:      keine                                            */
'*                                                                       */
'*-----------------------------------------------------------------------*/
Dim i                   As Integer                  'Allgemeine Zaehlvariable.
Dim iZeileStart         As Integer                  'Zeile aktueller Puzzleteil.
Dim iSpalteStart        As Integer                  'Spalte aktueller Puzzleteil.

On Error GoTo Fehler                                'Bei Fehler zur Fehlermarke.
SpielEnde = True                                    'Returnwert = alles OK setzen.

Set oTabellenblatt = Sheets(TABELLENBLATT)          'Akt. Tabellenblatt festlegen.
oTabellenblatt.Select                               'Tabellenblatt auswaehlen.
iZeileStart = 2                                     'Startposition fuer ersten
iSpalteStart = 2                                    'Puzzleteil festlegen.
For i = 1 To 15                                     '15 Puzzleteile kontrollieren:
  If oTabellenblatt.Cells(iZeileStart, iSpalteStart) <> i Then
    SpielEnde = False                               'Returnwert=Spiel nicht beendet.
    Exit Function                                   'Funktion kann vorzeitig
  End If                                            'beendet werden.
  iSpalteStart = iSpalteStart + 1                   'Zur naechste Position:
  If iSpalteStart > 5 Then iSpalteStart = 2: iZeileStart = iZeileStart + 1
Next i
If SPIELSTEIN_16 = JA Then                          'Wenn 16. Stein einblenden:
  Range("E5").Select:  ActiveCell.FormulaR1C1 = "16" 'dann Zahl 16 eintragen
  Selection.Interior.Color = ROT                    'und Hintergrund ROT machen.
End If

                                                    'Fehlerbehandlungsroutine:
Beenden:                                            '========================
  Application.Cursor = xlDefault                    'Sanduhr ausblenden.
  Exit Function                                     'Funktion abbrechen.
Fehler:                                             'Fehlermarke - Fehlerbehandlung:
  SpielEnde = False                                 'Returnwert = Fehler setzen.
  MsgBox Err.Description                            'Fehlermeldung ausgeben.
  Resume Beenden                                    'Bei Funktionsende weitermachen.
  Resume                                            'Nur fuer Testzwecke hinterlegt.
End Function                                        'Funktionsende.
```

4.6.) Makros

Als nächstes erstellen wir die Makros, welche wir dann den Formen für die Schaltflächen zuweisen werden. Die Makros könnten so aussehen:

```
Public Sub Spiel_Mischen()
'************************************************************************/
'*  Makro wird ausgefuehrt bei Klick auf Schaltflaeche: mischen         */
'************************************************************************/
If SpielMischen = False Then                        'Wenn Fehler in Funktion aufge-
                                                    'treten: Fehlermeldung ausgeben.
  MsgBox "Fehler beim Ausführen der Funktion SpielMischen!", _
         vbOKOnly + vbExclamation + vbDefaultButton1, PROGRAMMNAME
Else                                                'Funktion korrekt ausgefuehrt:
                                                    'Erfolgsmeldung ausgeben.
  MsgBox "Funktion SpielMischen erfolgreich ausgeführt.", _
         vbOKOnly + vbInformation + vbDefaultButton1, PROGRAMMNAME
End If
End Sub                                             'Prozedurende.


Public Sub Spiel_Aufloesen()
'************************************************************************/
'*  Makro wird ausgefuehrt bei Klick auf Schaltflaeche: aufloesen       */
'************************************************************************/
If SpielAufloesen = False Then                      'Wenn Fehler in Funktion aufge-
                                                    'treten: Fehlermeldung ausgeben.
  MsgBox "Fehler beim Ausführen der Funktion SpielAufloesen!", _
         vbOKOnly + vbExclamation + vbDefaultButton1, PROGRAMMNAME
Else                                                'Funktion korrekt ausgefuehrt:
                                                    'Erfolgsmeldung ausgeben.
  MsgBox "Funktion SpielAufloesen erfolgreich ausgeführt.", _
         vbOKOnly + vbInformation + vbDefaultButton1, PROGRAMMNAME
End If
End Sub                                             'Prozedurende.
```

Nachdem die Makros getestet wurden kann man die Ausgabe der erfolgreichen Funktionsausführung wieder unter Kommentar stellen.



Zum Zuweisen der Makros wechseln wir auf das Tabellenblatt **Schiebepuzzle** und Rechtsklicken die Form **mischen**.

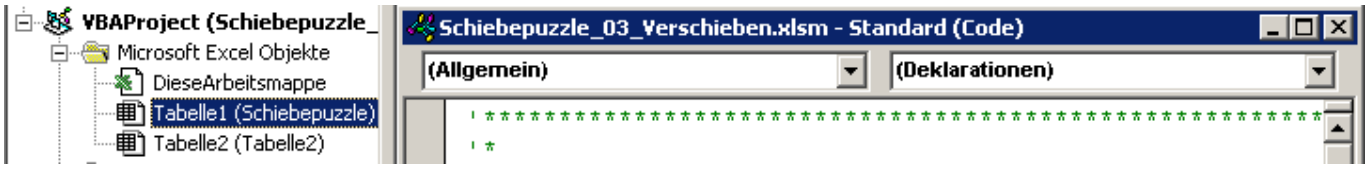
Im aufgehenden Menü wählen wir **Makro zuweisen…** aus, selektieren das Makro **Spiel_Mischen** und klicken abschließend auf OK.

Dasselbe machen wir dann mit der Form **auflösen** und weisen das Makro **Spiel_Aufloesen** zu.

Und das wäre auch schon wieder alles zum Thema Makros bei unserem Schiebepuzzle.

4.7.) Spielstein verschieben mit Doppelklick

Excel bietet uns eine eigene Methode an um ein Doppelklick Ereignis abfragen zu können: die Prozedur **Worksheet_BeforeDoubleClick**. Um die Prozedur zu erstellen doppelklicken Sie in **Visual Basic** im **Projekt-Explorer** das Microsoft Excel Objekt: **Tabelle1 (Schiebepuzzle)**:



Wählen Sie dann rechts im Code-Fenster der Combobox (Allgemein) "Workbook" aus und daneben in der Combobox (Deklarationen) "BeforeDoubleClick" aus. Folgender Code wird automatisch generiert:

```
Private Sub Worksheet_BeforeDoubleClick(ByVal Target As Range, Cancel As Boolean)

End Sub
```

Dabei beinhaltet der Parameter Target jene Zelle (als Range-Objekt), auf welche der Doppelklick gemacht wurde. Den Parameter Cancel können wir verwenden, um das Ereignis **BeimDoppelklicken** abzubrechen. Normalerweise bewirkt ein Doppelklick auf eine Zelle, das der Cursor in diese gesetzt wird und der Inhalt der Zelle bearbeitet werden kann. Da wir die Zelle nicht bearbeiten, sondern "verschieben" wollen, setzen wir am Ende unserer Funktion den Cancel-Parameter auf **True**.

Bevor wir das eigentliche Verschieben programmieren, kümmern wir uns noch um den verfügbaren Parameter **Target**. Er beinhaltet Informationen auf jene Zelle welche doppelgeklickt wurde. Mit den Eigenschaften **Row** und **Column** können wir die Zeile und Spaltennummer der Zelle ermitteln.

Wir wollen nur Doppelklicks auf Zellen, welche Puzzelteile im Spielbrett sind, behandeln. Daher fragen wir zu Beginn der Prozedur den Parameter Target ab und wenn die Zelle nicht zum Spielfeld gehört, dann wird die Prozedur einfach wieder beendet und lassen den Cursor zuvor noch in die linke obere Ecke setzen. Die Prozedur könnte dann folgend aussehen:

```
Private Sub Worksheet_BeforeDoubleClick(ByVal Target As Range, Cancel As Boolean)
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION - PROZEDUR              */
'*************************************************************************/
'*-----------------------------------------------------------------------*/
'*                                                                       */
'*      FUNKTION:       Worksheet_BeforeDoubleClick                      */
'*                                                                       */
'*      BESCHREIBUNG:   Diese Prozedur wird bei einem Doppelklick, irgendwo */
'*                      im Tabellenblatt, aufgerufen.                    */
'*                      Wird in dieser Prozedur der Parameter: Cancel    */
'*                      auf True gesetzt, so wird die Prozedur abgebrochen. */
'*                                                                       */
'*      RETURN          nichts                                           */
'*                                                                       */
'*      PARAMETER:      Range ............ Doppelt geklickte Zelle       */
'*                      Boolean .......... True = Funktion abbrechen     */
'*                                                                       */
'*-----------------------------------------------------------------------*/
On Error GoTo Fehler                                  'Bei Fehler zur Fehlerbehandlung.
```

```
If Target.Row > 1 And Target.Row < 6 Then           'Nur Zellen, welche Teil des
  If Target.Column > 1 And Target.Column < 6 Then   'Spielbrettes sind abhandeln:
    MsgBox "Puzzleteil verschieben..."              'Teil verschieben lassen und
  End If                                            'Kontrolle auf Spielende:
End If

Range("A1").Select                                  'Zelle A1 selektieren.
Cancel = True                                       'Prozedur abbrechen.


                                                    'Fehlerbehandlungsroutine:
Beenden:                                            '=========================
  Application.Cursor = xlDefault                    'Sanduhr ausblenden.
  Exit Sub                                          'Prozedur abbrechen.

Fehler:                                             'Fehlermarke - Fehlerbehandlung:
  MsgBox Err.Description                            'Fehlermeldung ausgeben.
  Resume Beenden                                    'Bei Funktionsende weitermachen.
  Resume                                            'Nur fuer Testzwecke hinterlegt.
End Sub                                             'Prozedurende.
```

Beim Verschieben des Puzzleteils soll folgendes erledigt werden: Zuerst den Teil verschieben lassen. Konnte der Teil verschoben werden, dann erhöhen wir die Anzahl die Spielzüge und geben sie im Tabellenblatt aus. Zusätzlich überprüfen wir ob das Puzzle fertig gelöst wurde, mit der zuvor erstellten Funktion **SpielEnde**.

Genaugenommen haben wir die Funktion zum Verschieben eines Puzzleteils bereits programmiert – nämlich in der Funktion **Spiel_Mischen**. Dort verschieben wir aber rein zufällig ein Puzzleteil nach dem anderen – jetzt wollen wir aber nur einen bestimmten Teil verschieben. Um für dieselbe Funktionalität nicht zweimal denselben Programmcode zu hinterlegen, lagern wir den Code für das Verschieben in eine eigene Funktion aus.

Basiswerte für das Verschieben sind die Zeilen- und Spaltennummer des zu verschiebenden Teiles. Daher übergeben wir die Zeile und Spalte als Parameter an diese Funktion. Als Rückgabewert der Funktion übergeben wir **True** oder **False** – Puzzleteil wurde verschoben oder nicht.

Die neue Funktion könnte dann so aussehen.

```
Public Function SpielSteinVerschieben(iZeile As Integer, iSpalte As Integer) As
Boolean
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION                        */
'*************************************************************************/
'*----------------------------------------------------------------------*/
'*                                                                      */
'*      FUNKTION:       SpielSteinVerschieben                           */
'*                                                                      */
'*      BESCHREIBUNG:   Verschiebt einen Puzzleteil auf ein angrenzendes*/
'*                      Leerfeld falls vorhanden.                       */
'*                                                                      */
'*      RETURN          Boolean ......... True = Funktion in Ordnung    */
'*                                        False = Fehler in Funktion    */
'*                                                                      */
'*      PARAMETER:      Integer ......... Zeilennummer des Puzzleteiles */
'*                      Integer ......... Spaltennummer des Puzzleteiles*/
'*                                                                      */
'*----------------------------------------------------------------------*/
```

```
Dim iZeileNeu           As Integer                  'Neue Zeile.
Dim iSpalteNeu          As Integer                  'Neue Spalte.

On Error GoTo Fehler                                'Bei Fehler zur Fehlermarke.
SpielSteinVerschieben = False                       'Returnwert = Stein wurde
                                                    'nicht verschoben festlegen.
If Not (IsNull(oTabellenblatt.Cells(iZeile, iSpalte))) Then   'Nur wenn der zu
  iZeileNeu = 0                                     'verschiebende Stein nicht
  iSpalteNeu = 0                                    'das Leerfeld ist:
                                                    'Neue Position vordefinieren.
  'Wenn sich das Leerfeld in einer der vier Richtungen direkt neben dem zufaelligen
  'Spielstein befindet, dann die neue Position des Steines festlegen.
  'Normalerweise wuerde es genuegen, die Hintergrundfarbe abzufragen.
  'Da hier jedoch sowohl das Leerfeld als auch der Rahmen um das Spielfeld die
  'selbe Hintergrundfarbe haben, ist es notwendig zusaetzlich die Schriftfarbe
  'abzufragen. Daher zu Beginn sicherstellen, dass die Schriftfarbe im Rahmen
  '(aktuell SCHWARZ) ungleich der Schriftfarbe im Spielbrett (aktuell GOLD) ist!
  If oTabellenblatt.Cells(iZeile + 0, iSpalte + 1).Interior.Color = SCHWARZ And
oTabellenblatt.Cells(iZeile + 0, iSpalte + 1).Font.Color = GOLD Then iZeileNeu =
iZeile + 0: iSpalteNeu = iSpalte + 1

  If oTabellenblatt.Cells(iZeile + 0, iSpalte - 1).Interior.Color = SCHWARZ And
oTabellenblatt.Cells(iZeile + 0, iSpalte - 1).Font.Color = GOLD Then iZeileNeu =
iZeile + 0: iSpalteNeu = iSpalte - 1

  If oTabellenblatt.Cells(iZeile + 1, iSpalte + 0).Interior.Color = SCHWARZ And
oTabellenblatt.Cells(iZeile + 1, iSpalte + 0).Font.Color = GOLD Then iZeileNeu =
iZeile + 1: iSpalteNeu = iSpalte + 0

  If oTabellenblatt.Cells(iZeile - 1, iSpalte + 0).Interior.Color = SCHWARZ And
oTabellenblatt.Cells(iZeile - 1, iSpalte + 0).Font.Color = GOLD Then iZeileNeu =
iZeile - 1: iSpalteNeu = iSpalte + 0

  If iZeileNeu <> 0 And iSpalteNeu <> 0 Then        'Nur wenn Leerfeld neben Stein:
                                                    'Nummer des Steines eintragen:
    oTabellenblatt.Cells(iZeileNeu, iSpalteNeu) = oTabellenblatt.Cells(iZeile,
iSpalte)
                                                    'Hintergrundfarbe uebernehmen:
    oTabellenblatt.Cells(iZeileNeu, iSpalteNeu).Interior.Color =
oTabellenblatt.Cells(iZeile, iSpalte).Interior.Color
    oTabellenblatt.Cells(iZeile, iSpalte) = Null    'Neues Leerfeld aktualisieren:
    oTabellenblatt.Cells(iZeile, iSpalte).Interior.Color = SCHWARZ
    SpielSteinVerschieben = True                    'Returnwert = Teil verschoben.
  End If
End If
                                                    'Fehlerbehandlungsroutine:
Beenden:                                            '========================
  Application.Cursor = xlDefault                    'Sanduhr ausblenden.
  Exit Function                                     'Funktion abbrechen.

Fehler:                                             'Fehlermarke - Fehlerbehandlung:
  SpielSteinVerschieben = False                     'Returnwert = Fehler setzen.
  MsgBox Err.Description                            'Fehlermeldung ausgeben.
  Resume Beenden                                    'Bei Funktionsende weitermachen.
  Resume                                            'Nur fuer Testzwecke hinterlegt.
End Function                                        'Funktionsende.
```

Die Funktion Spiel Mischen sieht jetzt so aus:

```
Public Function SpielMischen() As Boolean
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION                        */
'*************************************************************************/
'*-----------------------------------------------------------------------*/
'*                                                                       */
'*      FUNKTION:       SpielMischen                                     */
'*                                                                       */
'*      BESCHREIBUNG:   Mischt die Teile des Schiebepuzzle durch, gibt das  */
'*                      Leerfeld aus und setzt die Anzahl der Spielzuege */
'*                      auf 0.                                           */
'*                                                                       */
'*      RETURN          Boolean ..... True = Funktion in Ordnung         */
'*                                    False = Fehler in Funktion         */
'*      PARAMETER:      keine                                            */
'*                                                                       */
'*-----------------------------------------------------------------------*/
Dim iZeile              As Integer                  'Aktuelle Zeile.
Dim iSpalte             As Integer                  'Aktuelle Spalte
Dim bZuschauen          As Boolean                  'Wenn JA, dann kann der Spieler
                                                    'das Mischen optisch verfolgen,
                                                    'sonst nicht.

On Error GoTo Fehler                                'Bei Fehler zur Fehlermarke.
SpielMischen = True                                 'Returnwert = alles OK setzen.
bZuschauen = NEIN                                   'Zuschauen lassen oder nicht:

Set oTabellenblatt = Sheets(TABELLENBLATT)          'Akt. Tabellenblatt festlegen.
oTabellenblatt.Select                               'Tabellenblatt auswaehlen.

If bZuschauen = NEIN Then                           'Je nachdem wie es beliebt:
  Application.ScreenUpdating = False                'Bildschirmaktualisierung
End If                                              'ausschalten.
                                                    'Zuerst das Spiel aufloesen lassen
Call SpielAufloesen                                 'damit der 16. Stein auf E5 ist.
Range("E5").Select:  ActiveCell.FormulaR1C1 = Null  'Dort einen NULL-Wert eintragen
Selection.Interior.Color = SCHWARZ                  'und Hintergrund SCHWARZ machen.
Range("A1").Select                                  'Zelle A1 selektieren.

While iAnzahlZuege < 200                            '200 Verschiebungen durchfuehren:
  iZeile = 1 + Int(4 * Rnd) + 1                     'Zufaellige Zeile und
  iSpalte = 1 + Int(4 * Rnd) + 1                    'zufaellige Spalte festlegen.
  If SpielSteinVerschieben(iZeile, iSpalte) = True Then
    If bZuschauen = JA Then Sleep (50)              'Beim Zuschauen etwas warten.
    iAnzahlZuege = iAnzahlZuege + 1                 'Anzahl der zufaelligen
  End If                                            'Verschiebungen erhoehen.
Wend
iAnzahlZuege = 0                                    'Anzahl der Spielzuege ruecksetzen.
oTabellenblatt.Range("ZUEGE") = "Anzahl Züge bisher:  0000" 'und im Tabellenblatt ausgeben.

Application.ScreenUpdating = True                   'Bildschirmaktualisierung auf
                                                    'alle Faelle wieder einschalten.

                                                    'Fehlerbehandlungsroutine:
Beenden:                                            '========================
  Application.Cursor = xlDefault                    'Sanduhr ausblenden.
  Exit Function                                     'Funktion abbrechen.

Fehler:                                             'Fehlermarke - Fehlerbehandlung:
  SpielMischen = False                              'Returnwert = Fehler setzen.
  MsgBox Err.Description                            'Fehlermeldung ausgeben.
  Resume Beenden                                    'Bei Funktionsende weitermachen.
  Resume                                            'Nur fuer Testzwecke hinterlegt.
End Function                                        'Funktionsende.
```

Und die fertige Prozedur **Worksheet_BeforeDoubleClick** könnte so aussehen:

```
Private Sub Worksheet_BeforeDoubleClick(ByVal Target As Range, Cancel As Boolean)
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION - PROZEDUR              */
'*************************************************************************/
'*-----------------------------------------------------------------------*/
'*                                                                       */
'*      FUNKTION:       Worksheet_BeforeDoubleClick                      */
'*                                                                       */
'*      BESCHREIBUNG:   Diese Prozedur wird bei einem Doppelklick, irgendwo */
'*                      im Tabellenblatt, aufgerufen.                    */
'*                      Wird in dieser Prozedur der Parameter: Cancel    */
'*                      auf True gesetzt, so wird die Prozedur abgebrochen. */
'*                                                                       */
'*      RETURN          nichts                                           */
'*                                                                       */
'*      PARAMETER:      Range ............ Doppelt geklickte Zelle       */
'*                      Boolean .......... True = Funktion abbrechen     */
'*                                                                       */
'*-----------------------------------------------------------------------*/
Dim stAnzahlZuege       As String
On Error GoTo Fehler                                'Bei Fehler zur Fehlerbehandlung.

If Target.Row > 1 And Target.Row < 6 Then           'Nur Zellen, welche Teil des
  If Target.Column > 1 And Target.Column < 6 Then   'Spielbrettes sind abhandeln:
    Set oTabellenblatt = Sheets(TABELLENBLATT)      'Akt. Tabellenblatt festlegen.

    If SpielSteinVerschieben(Target.Row, Target.Column) = True Then 'Wenn verschoben,
      iAnzahlZuege = iAnzahlZuege + 1               'Anzahl der Spielzuege erhoehen.
      stAnzahlZuege = Right("0000" & Trim(Str(iAnzahlZuege)), 4) 'Anzahl als String.
      oTabellenblatt.Range("ZUEGE") = "Anzahl Züge bisher:  " & stAnzahlZuege
    End If                                          'und im Tabellenblatt ausgeben.

    If SpielEnde = True Then                        'Kontrolle auf Spielende:
      MsgBox "GRATULATION! Sie haben das Puzzle gelöst!", _
             vbOKOnly + vbInformation + vbDefaultButton1, PROGRAMMNAME
    End If                                          'Falls Puzzle geloest wurde,
  End If                                            'dann entsprechende Meldung
End If                                              'ausgeben.

Range("A1").Select                                  'Zelle A1 selektieren.
Cancel = True                                       'Prozedur abbrechen.

                                                    'Fehlerbehandlungsroutine:
Beenden:                                            '========================
  Application.Cursor = xlDefault                    'Sanduhr ausblenden.
  Exit Sub                                          'Prozedur abbrechen.

Fehler:                                             'Fehlermarke - Fehlerbehandlung:
  MsgBox Err.Description                            'Fehlermeldung ausgeben.
  Resume Beenden                                    'Bei Funktionsende weitermachen.
  Resume                                            'Nur fuer Testzwecke hinterlegt.
End Sub                                             'Prozedurende.
```

Damit hätten wir alle benötigten Programmteile ausprogrammiert und wir können gleich mal versuchen das Schiebepuzzle zu lösen. Versuchen Sie diverse Parameter und Konstanten zu ändern, bis Sie mit dem Ablauf des Spieles zufrieden sind.

4.8.) Spiel mit Sounds unterlegen

Zuerst müssen wir uns überlegen bei welchen Aktionen ein Sound wiedergegeben werden soll und vor allem welcher Sound. Bei diesem Schiebepuzzle würde ich folgendes vorschlagen:

Hintergrundmusik: entweder Hintergrundmusik.wav (sanft) oder Hintergrundmusik_hard.wav (Rock).

Beim Starten (wenn Mischen beendet) – Sound ähnlich der Windows Anmeldung – Starten.mp3
Beim Verschieben (wenn möglich) – ein zischendes Geräusch – Verschieben.mp3
Beim Verschieben (wenn nicht möglich) – Windows-Sound: Kritischer Fehler – Fehler.mp3
Beim Mischen (wenn möglich) – Schreibmaschinen Geklapper – Mischen.mp3
Wenn Fertig (Puzzle gelöst) – Tusch mit Applaus – Fertig.mp3
Wenn Fertig (Puzzle auflösen lassen) – Wie beim Verschieben – Aufloesen.mp3

Windows Sounds findet man als Wave-Dateien im Verzeichnis: **C:\Windows\Media**
Beispiele für Hintergrundmusik habe ich im Verzeichnis: **\Gemafreie_Musik** abgelegt.
Oder das Internet durchsuchen nach "Geräusche mp3" und herunterladen - dabei die Rechte beachten!

Hat man sich die gewünschten Musikdateien zusammengestellt, dann mit einem beliebigen Audio-Bearbeitungsprogramm die gewünschte Lautstärke einstellen (sollten für alle Geräusche in etwas gleich laut sein, die Hintergrundmusik entsprechend leiser) und die Spieldauer anpassen (Stille am Anfang und Ende entfernen oder einen Musikabschnitt herausfiltern). Danach noch als mp3- oder wav-Datei abspeichern. Wave-Datei für Hintergrundmusik, Rest als mp3-Dateien.

Bevor wir jedoch beginnen das Spiel mit den Sounds zu hinterlegen sollten wir eine zusätzliche Schaltfläche erstellen, mit welcher wir den Ton ein- bzw. ausschalten können. Dabei empfehle ich beim Starten des Spiels (also beim Öffnen der Datei) den Ton und die Musik grundsätzlich auszuschalten. Dann kann der Spieler, wenn er es wirklich will, die Sounds einschalten und auch die gewünschte Lautstärke über das Betriebssystem (oder Audiotreiber) selbst einstellen.

4.8.1.) Die Schaltfläche: Ton EIN / Ton AUS

Dazu legen wir und eine neue Form an (Kopieren einer bestehenden) und beschriften sie mit **Ton EIN**:

Beim Öffnen der Excel-Datei definieren wir über eine Globale Variable, daß der Ton ausgeschaltet ist. Ein Klick auf die Schaltfläche **Ton EIN** schaltet dann die Ausgabe der Sounds ein. Gleichzeitig ändern wir dabei die Beschriftung von **Ton EIN** auf **Ton AUS**. Ein Klick auf dieselbe Form, welche nun **Ton AUS** heißt, schaltet die Musikwiedergabe aus und setzt die Beschriftung wieder auf **Ton EIN**.

Und schon geht´s an die Programmierung: Zuerst ergänzen wir im Modul **Globales** die Variable in der wir uns merken ob Sounds wiedergegeben werden sollen oder nicht.

```
Global bSoundsAbspielen                 As Boolean  'Sounds abspielen lassen.
```

Dann schreiben wir uns eine Funktion zum Umschalten der Musikwiedergabe, am besten im Modul **Funktionen** ablegen.

```
Public Function TonUmschalten()
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION                        */
'*************************************************************************/
'*-----------------------------------------------------------------------*/
'*                                                                       */
'*      FUNKTION:       TonUmschalten                                    */
'*                                                                       */
'*      BESCHREIBUNG:   Schaltet die Wiedergabe von Hintergrundmusik und */
'*                      Geraeuschen aus bzw. ein.                        */
'*                                                                       */
'*      RETURN          nichts                                           */
'*                                                                       */
'*      PARAMETER:      keine                                            */
'*                                                                       */
'*-----------------------------------------------------------------------*/
Dim i                   As Integer                  'Allgemeine Zaehlvariable.

Set oTabellenblatt = Sheets(TABELLENBLATT)          'Akt. Tabellenblatt festlegen.
oTabellenblatt.Select                               'Tabellenblatt auswaehlen.

'Sollte ein Shape (Form) nicht die Eigenschaft: TextFrame.Characters.Text
'haben, dann tritt ein Laufdzeitfehler auf. Um dieses zu verhindern wird
'der Programmcode bei einem Fehler in der naechsten Programmzeile fortgesetzt.
'Genaugenommen wird das naechste Shape abgehandelt.

On Error Resume Next                                'Bei Fehler einfach weitermachen:
For i = 0 To oTabellenblatt.Shapes.Count            'Alle Formen im Tabellenblatt
  Err = 0                                           'durcharbeiten. Fehler 0 setzen.
  If oTabellenblatt.Shapes(i).TextFrame.Characters.Text = "Ton AUS" Then
    If Err = 0 Then                                 'Wenn Ton ausschalten:
      bSoundsAbspielen = False                      'Merken: Keine Tonwiedergabe.
      oTabellenblatt.Shapes(i).TextFrame.Characters.Text = "Ton EIN"
      Exit For                                      'Beschriftung aendern und
    End If                                          'Funktion beenden.

  ElseIf oTabellenblatt.Shapes(i).TextFrame.Characters.Text = "Ton EIN" Then
    If Err = 0 Then                                 'Wenn Ton einschalten:
      bSoundsAbspielen = True                       'Merken: Ton wiedergeben.
      oTabellenblatt.Shapes(i).TextFrame.Characters.Text = "Ton AUS"
      Exit For                                      'Beschriftung aendern und
    End If                                          'Funktion beenden.
  End If
Next i
End Function                                        'Funktionsende.
```

Nun erstellen wir noch ein Makro, welches wir dann der Schaltfläche **Ton EIN** zuweisen müssen:

```
Public Sub Ton_Umschalten()
'*************************************************************************/
'*  Makro wird ausgefuehrt bei Klick auf Schaltflaeche: Ton AUS bzw. Ton EIN*/
'*************************************************************************/
Call TonUmschalten
End Sub                                             'Prozedurende.
```

4.8.2.) Definition der Geräusche und Hintergrundmusik

Dazu erweitern wir zuerst das Modul **Globales** mit folgenden Definitionen:

```
Global bSoundStartenOK                As Boolean   'Sound Start kann abgespielt werden.
Global bSoundFertigOK                 As Boolean   'Sound Fertig kann abgespielt werden.
Global bSoundMischenOK                As Boolean   'Sound Mischen kann abgespielt werden.
Global bSoundSchiebenOK               As Boolean   'Sound Schieben kann abgespielt werden.
Global bSoundAufloesenOK              As Boolean   'Sound Aufloesen kann abgespielt werden.
Global bSoundFehlerOK                 As Boolean   'Sound Fehler kann abgespielt werden.
Global stDateiHintergrundMusik        As String    'Aktueller Dateiname fuer Hintergrundmusik.

Global Const DATEI_SOUND_STARTEN = "\Starten.mp3"
Global Const DATEI_SOUND_FERTIG = "\Fertig.mp3"
Global Const DATEI_SOUND_MISCHEN = "\Mischen.mp3"
Global Const DATEI_SOUND_SCHIEBEN = "\Schieben.mp3"
Global Const DATEI_SOUND_AUFLOESEN = "\Aufloesen.mp3"
Global Const DATEI_SOUND_FEHLER = "\Fehler.mp3"
Global Const DATEI_SOUND_HINTERGRUND = "\Hintergrundmusik.wav"
'Global Const DATEI_SOUND_HINTERGRUND = "\Hintergrundmusik_hard.wav"

Global Const ALIAS_SOUND_STARTEN = "Alias_Sound_Starten"
Global Const ALIAS_SOUND_FERTIG = "Alias_Sound_Fertig"
Global Const ALIAS_SOUND_MISCHEN = "Alias_Sound_Mischen"
Global Const ALIAS_SOUND_SCHIEBEN = "Alias_Sound_Schieben"
Global Const ALIAS_SOUND_AUFLOESEN = "Alias_Sound_Aufloesen"
Global Const ALIAS_SOUND_FEHLER = "Alias_Sound_Fehler"
Global Const ALIAS_SOUND_HINTERGRUND = "Alias_Hintergrundmusik"
```

In den globalen Variablen vermerken wir, ob die einzelnen Dateien überhaupt abgespielt werden können. Wenn nicht, dann wird später auch gar nicht erst versucht die Datei wiederzugeben. In den globalen Konstanten definieren wir die einzelnen Musikdateien und die dazugehörigen Alias-Namen.

Als nächstes versuchen wir die Musikdateien für alle Geräusche und die Hintergrundmusik einzulesen und die entsprechenden Aliasnamen zuzuordnen um die einzelnen Sounds später gezielt abspielen zu können. Das erledigen wir in einer neuen Funktion welche am besten im Modul **Sounds** hinterlegt wird:

```
Public Function SpielSoundsDefinieren() As Boolean
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION                        */
'*************************************************************************/
'*-----------------------------------------------------------------------*/
'*                                                                       */
'*      FUNKTION:       SpielSoundsDefinieren                            */
'*                                                                       */
'*      BESCHREIBUNG:   Definiert die Aliasnamen von MP3-Sounddateien.   */
'*                                                                       */
'*      RETURN          Boolean ..... True = Funktion in Ordnung         */
'*                                    False = Fehler in Funktion         */
'*      PARAMETER:      keine                                            */
'*                                                                       */
'*-----------------------------------------------------------------------*/
Dim stDateiname         As String                   'Sound-Datei.
Dim stAlias             As String                   'Aliasname fuer Sound-Datei.
Dim stBuffer            As String                   'Buffer fuer DOS 8.3 Dateinamen.

On Error GoTo Fehler                                'Bei Fehler zur Fehlerbehandlung.
SpielSoundsDefinieren = True                        'Returnwert = alles OK setzen.
Call SpielSoundsRuecksetzen                         'Zuerst alles ruecksetzen.

'*-----------------------------------------------------------------------*/
'*- Sound nach Starten des Spieles (nach dem Mischen, beim Starten)     -*/
'*-----------------------------------------------------------------------*/
stDateiname = ActiveWorkbook.Path & DATEI_SOUND_STARTEN 'Dateiname für Sound und
```

```
stAlias = ALIAS_SOUND_STARTEN                          'Namen fuer Alias festlegen.
stBuffer = Space$(255)                                 'DOS 8.3 Dateinamen ermitteln:
If GetShortPathName(stDateiname, stBuffer, Len(stBuffer)) <> 0 Then
  stDateiname = Left$(stBuffer, InStr(stBuffer, vbNullChar) - 1)
End If                                                 'Leerzeichen am Ende entfernen.
If PlayMP3Sound("open " & stDateiname & " type MPEGVideo alias " & stAlias, 0, 0, 0) = 0 Then
  bSoundStartenOK = True                               'Sound kann abgespielt werden.
Else
  bSoundStartenOK = False                              'Sound nicht abspielbar.
End If

'*-------------------------------------------------------------------------*/
'*- Sound nachdem Spiel fertig (nachdem Puzzle geloest wurde)              -*/
'*-------------------------------------------------------------------------*/
stDateiname = ActiveWorkbook.Path & DATEI_SOUND_FERTIG 'Dateiname für Sound und
stAlias = ALIAS_SOUND_FERTIG                           'Namen fuer Alias festlegen.
stBuffer = Space$(255)                                 'DOS 8.3 Dateinamen ermitteln:
If GetShortPathName(stDateiname, stBuffer, Len(stBuffer)) <> 0 Then
  stDateiname = Left$(stBuffer, InStr(stBuffer, vbNullChar) - 1)
End If                                                 'Leerzeichen am Ende entfernen.
If PlayMP3Sound("open " & stDateiname & " type MPEGVideo alias " & stAlias, 0, 0, 0) = 0 Then
  bSoundFertigOK = True                                'Sound kann abgespielt werden.
Else
  bSoundFertigOK = False                               'Sound nicht abspielbar.
End If

'*-------------------------------------------------------------------------*/
'*- Sound beim Verschieben eines Puzzleteiles (nach dem Verschieben)       -*/
'*-------------------------------------------------------------------------*/
stDateiname = ActiveWorkbook.Path & DATEI_SOUND_SCHIEBEN 'Dateiname für Sound und
stAlias = ALIAS_SOUND_SCHIEBEN                         'Namen fuer Alias festlegen.
stBuffer = Space$(255)                                 'DOS 8.3 Dateinamen ermitteln:
If GetShortPathName(stDateiname, stBuffer, Len(stBuffer)) <> 0 Then
  stDateiname = Left$(stBuffer, InStr(stBuffer, vbNullChar) - 1)
End If                                                 'Leerzeichen am Ende entfernen.
If PlayMP3Sound("open " & stDateiname & " type MPEGVideo alias " & stAlias, 0, 0, 0) = 0 Then
  bSoundSchiebenOK = True                              'Sound kann abgespielt werden.
Else
  bSoundSchiebenOK = False                             'Sound nicht abspielbar.
End If

'*-------------------------------------------------------------------------*/
'*- Sound beim Verschieben eines Puzzleteiles (Verschieben nicht moeglich) -*/
'*-------------------------------------------------------------------------*/
stDateiname = ActiveWorkbook.Path & DATEI_SOUND_FEHLER 'Dateiname für Sound und
stAlias = ALIAS_SOUND_FEHLER                           'Namen fuer Alias festlegen.
stBuffer = Space$(255)                                 'DOS 8.3 Dateinamen ermitteln:
If GetShortPathName(stDateiname, stBuffer, Len(stBuffer)) <> 0 Then
  stDateiname = Left$(stBuffer, InStr(stBuffer, vbNullChar) - 1)
End If                                                 'Leerzeichen am Ende entfernen.
If PlayMP3Sound("open " & stDateiname & " type MPEGVideo alias " & stAlias, 0, 0, 0) = 0 Then
  bSoundFehlerOK = True                                'Sound kann abgespielt werden.
Else
  bSoundFehlerOK = False                               'Sound nicht abspielbar.
End If

'*-------------------------------------------------------------------------*/
'*- Sound beim Mischen                                                    -*/
'*-------------------------------------------------------------------------*/
stDateiname = ActiveWorkbook.Path & DATEI_SOUND_MISCHEN 'Dateiname für Sound und
stAlias = ALIAS_SOUND_MISCHEN                          'Namen fuer Alias festlegen.
stBuffer = Space$(255)                                 'DOS 8.3 Dateinamen ermitteln:
If GetShortPathName(stDateiname, stBuffer, Len(stBuffer)) <> 0 Then
  stDateiname = Left$(stBuffer, InStr(stBuffer, vbNullChar) - 1)
End If                                                 'Leerzeichen am Ende entfernen.
If PlayMP3Sound("open " & stDateiname & " type MPEGVideo alias " & stAlias, 0, 0, 0) = 0 Then
  bSoundMischenOK = True                               'Sound kann abgespielt werden.
```

```
Else
  bSoundMischenOK = False                          'Sound nicht abspielbar.
End If

'*-------------------------------------------------------------------------*/
'*- Sound nach dem Aufloesen                                              -*/
'*-------------------------------------------------------------------------*/
stDateiname = ActiveWorkbook.Path & DATEI_SOUND_AUFLOESEN 'Dateiname für Sound und
stAlias = ALIAS_SOUND_AUFLOESEN                  'Namen fuer Alias festlegen.
stBuffer = Space$(255)                           'DOS 8.3 Dateinamen ermitteln:
If GetShortPathName(stDateiname, stBuffer, Len(stBuffer)) <> 0 Then
  stDateiname = Left$(stBuffer, InStr(stBuffer, vbNullChar) - 1)
End If                                           'Leerzeichen am Ende entfernen.
If PlayMP3Sound("open " & stDateiname & " type MPEGVideo alias " & stAlias, 0, 0, 0) = 0 Then
  bSoundAufloesenOK = True                       'Sound kann abgespielt werden.
Else
  bSoundAufloesenOK = False                      'Sound nicht abspielbar.
End If

'*-------------------------------------------------------------------------*/
'*- Hintergrundmusik                                                      -*/
'*-------------------------------------------------------------------------*/
stDateiHintergrundMusik = ActiveWorkbook.Path & DATEI_SOUND_HINTERGRUND 'Dateinamen für
                                                 'Hintergrundmusik 2 festlegen.
stBuffer = Space$(255)                           'DOS 8.3 Dateinamen ermitteln:
If GetShortPathName(stDateiHintergrundMusik, stBuffer, Len(stBuffer)) <> 0 Then
  stDateiHintergrundMusik = Left$(stBuffer, InStr(stBuffer, vbNullChar) - 1)
Else                                             'Die Leerzeichen am Ende entfernen.
  stDateiHintergrundMusik = ""                   'Hintergrundmusik nicht abspielbar.
End If
                                                 'Fehlerbehandlungsroutine:
Beenden:                                         '========================
  Exit Function                                  'Funktion abbrechen.

Fehler:                                          'Fehlermarke - Fehlerbehandlung:
  SpielSoundsDefinieren = False                  'Returnwert = Fehler setzen.
  MsgBox Err.Description                         'Fehlermeldung ausgeben.
  Resume Beenden                                 'Bei Funktionsende weitermachen.
  Resume                                         'Nur fuer Testzwecke hinterlegt.
End Function                                     'Funktionsende.
```

4.8.3.) Hintergrundmusik starten und stoppen

Zum Ein- bzw. Ausschalten der Hintergrundmusik legen wir uns im Modul **Sounds** zwei weitere Funktionen an:

```
Public Function HintergrundMusik_EIN()
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION                        */
'*************************************************************************/
'*-----------------------------------------------------------------------*/
'*                                                                       */
'*      FUNKTION:       HintergrundMusik_EIN                             */
'*                                                                       */
'*      BESCHREIBUNG:   Startet Wiedergabe der Hintergrundmusik in einer */
'*                      Endlos-Schleife.                                 */
'*                                                                       */
'*      RETURN          nichts                                           */
'*                                                                       */
'*      PARAMETER:      keine                                            */
'*                                                                       */
'*-----------------------------------------------------------------------*/
```

```
If bSoundsAbspielen = True And stDateiHintergrundMusik <> "" Then 'Wenn abgespielt
  Call PlayWaveSound(stDateiHintergrundMusik, SND_ASYNC Or SND_LOOP) 'werden soll und
End If                                                              'Datei fuer Wiedergabe vorhanden.
End Function                                                        'Funktionsende.
```

Nachdem die Wiedergabe von Sounds eingeschaltet wurde soll die Hintergrundmusik automatisch gestartet und dann in einer Endlos-Schleife wiedergegeben werden. Wird der Tonwiedergabe ausgeschaltet, dann soll auch die Hintergrundmusik sofort beendet werden. Das werden wir später in unserer Funktion **TonUmschalten** noch erweitern.

```
Public Function HintergrundMusik_AUS()
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION                        */
'*************************************************************************/
'*-----------------------------------------------------------------------*/
'*                                                                       */
'*      FUNKTION:       HintergrundMusik_AUS                             */
'*                                                                       */
'*      BESCHREIBUNG:   Stoppt Wiedergabe der Hintergrundmusik.          */
'*                                                                       */
'*      RETURN          nichts                                           */
'*                                                                       */
'*      PARAMETER:      keine                                            */
'*                                                                       */
'*-----------------------------------------------------------------------*/
Call PlayWaveSound(0, SND_PURGE)                    'Wiedergabe sofort beenden.
End Function                                        'Funktionsende.
```

4.8.4.) Wiedergabe Geräusche starten und stoppen

Für die Wiedergabe der einzelnen Geräusche legen wir uns im Modul **Sounds** eine weitere Funktion an. Je nachdem, welcher Alias-Name während des Spiels übergeben wurde, wird auch die dazugehörige mp3-Datei abgespielt:

```
Public Function SoundAbspielen(stAliasname As String)
If bSoundsAbspielen = True Then                     'Wenn abgespielt werden soll:
  Select Case stAliasname                           'Je nachdem welches Geraeusch:
    Case ALIAS_SOUND_STARTEN                        'Wenn dazugehoerige Datei
      If bSoundStartenOK = True Then                'abgespielt werden kann, dann
        Call PlayMP3Sound("play " & stAliasname & " from 0", 0, 0, 0)
      End If                                        'diese abspielen lassen.

    Case ALIAS_SOUND_FERTIG
      If bSoundFertigOK = True Then
        Call PlayMP3Sound("play " & stAliasname & " from 0", 0, 0, 0)
      End If

    Case ALIAS_SOUND_SCHIEBEN
      If bSoundSchiebenOK = True Then
        Call PlayMP3Sound("play " & stAliasname & " from 0", 0, 0, 0)
      End If

    Case ALIAS_SOUND_MISCHEN
      If bSoundMischenOK = True Then
        Call PlayMP3Sound("play " & stAliasname & " from 0", 0, 0, 0)
      End If
```

```
    Case ALIAS_SOUND_AUFLOESEN
      If bSoundAufloesenOK = True Then
        Call PlayMP3Sound("play " & stAliasname & " from 0", 0, 0, 0)
      End If

    Case ALIAS_SOUND_FEHLER
      If bSoundFehlerOK = True Then
        Call PlayMP3Sound("play " & stAliasname & " from 0", 0, 0, 0)
      End If

    Case Else

  End Select
End If
End Function                                          'Funktionsende.
```

Wurde die Wiedergabe von Geräuschen deaktiviert, dann soll die aktuelle Wiedergabe eines Geräusches sofort gestoppt werden und die Zuweisung der Aliasnamen wird zurückgesetzt. Das erledigen wir das ebenfalls in einer eigenen Funktion im Modul **Sounds**. Diese Funktion werden wir später in unserer Funktion **TonUmschalten** ergänzen.

```
Public Function SpielSoundsRuecksetzen()
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION                        */
'*************************************************************************/
'*----------------------------------------------------------------------*/
'*                                                                      */
'*      FUNKTION:       SpielSoundsRuecksetzen                          */
'*                                                                      */
'*      BESCHREIBUNG:   Stoppt alle Wiedergaben und schließt MCI´s.     */
'*                                                                      */
'*      RETURN          nichts                                          */
'*                                                                      */
'*      PARAMETER:      keine                                           */
'*                                                                      */
'*----------------------------------------------------------------------*/
PlayMP3Sound "stop " & ALIAS_SOUND_STARTEN, 0, 0, 0    'Wiedergabe stoppen.
PlayMP3Sound "close " & ALIAS_SOUND_STARTEN, 0, 0, 0   'MCI schliessen.

PlayMP3Sound "stop " & ALIAS_SOUND_FERTIG, 0, 0, 0     'Wiedergabe stoppen.
PlayMP3Sound "close " & ALIAS_SOUND_FERTIG, 0, 0, 0    'MCI schliessen.

PlayMP3Sound "stop " & ALIAS_SOUND_SCHIEBEN, 0, 0, 0   'Wiedergabe stoppen.
PlayMP3Sound "close " & ALIAS_SOUND_SCHIEBEN, 0, 0, 0  'MCI schliessen.

PlayMP3Sound "stop " & ALIAS_SOUND_MISCHEN, 0, 0, 0    'Wiedergabe stoppen.
PlayMP3Sound "close " & ALIAS_SOUND_MISCHEN, 0, 0, 0   'MCI schliessen.

PlayMP3Sound "stop " & ALIAS_SOUND_AUFLOESEN, 0, 0, 0  'Wiedergabe stoppen.
PlayMP3Sound "close " & ALIAS_SOUND_AUFLOESEN, 0, 0, 0 'MCI schliessen.

PlayMP3Sound "stop " & ALIAS_SOUND_FEHLER, 0, 0, 0     'Wiedergabe stoppen.
PlayMP3Sound "close " & ALIAS_SOUND_FEHLER, 0, 0, 0    'MCI schliessen.

End Function                                           'Funktionsende.
```

4.8.5.) Einbinden der Soundwiedergabe in die Spielfunktionen

Zunächst erledigen wir die Standardeinstellungen nachdem die Excel-Datei geöffnet wurde. Das findet in der Funktion **SpielStarten** statt. Wir ergänzen diese Funktion um die gelb hinterlegten Programmzeilen:

```
Public Function SpielStarten() As Boolean
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION                        */
'*************************************************************************/
'*-----------------------------------------------------------------------*/
'*                                                                       */
'*      FUNKTION:       SpielStarten                                     */
'*                                                                       */
'*      BESCHREIBUNG:   Wurde die globale Konstante SPIELSTART_NEU im Modul */
'*                      Globales auf JA gesetzt, dann wird beim Oeffnen der */
'*                      Datei ein geloestes Schiebepuzzle angezeigt. Bei */
'*                      NEIN kann der Spieler weitermachen wo er aufgehoert */
'*                      hat. Oder man koennte das Spielbrett neu mischen. */
'*                                                                       */
'*      RETURN          Boolean ..... True = Funktion in Ordnung         */
'*                                    False = Fehler in Funktion         */
'*      PARAMETER:      keine                                            */
'*                                                                       */
'*-----------------------------------------------------------------------*/
Dim i                   As Integer                  'Allgemeine Zaehlvariable.

On Error GoTo Fehler                                'Bei Fehler zur Fehlermarke.
SpielStarten = True                                 'Returnwert = alles OK setzen.

Set oTabellenblatt = Sheets(TABELLENBLATT)          'Akt. Tabellenblatt festlegen.
oTabellenblatt.Select                               'Tabellenblatt auswaehlen.

On Error Resume Next                                'Bei Fehler einfach weitermachen,
                                                    'da nicht alle Shapes die
                                                    'Eigenschaft TextFrame haben.
For i = 0 To oTabellenblatt.Shapes.Count            'Alle Shapes (Frames) durchsuchen:
                                                    'Sollte angezeigt werden, den
  If oTabellenblatt.Shapes(i).TextFrame.Characters.Text = "Ton AUS" Then 'Ton NICHT
     oTabellenblatt.Shapes(i).TextFrame.Characters.Text = "Ton EIN" 'wiedergeben,
  End If                                            'dann Text umaendern, damit der
Next i                                              'Spieler Ton selbst aktivieren kann.
On Error GoTo Fehler                                'Bei Fehler wieder zur Fehlermarke.

bSoundsAbspielen = False                            'Musikwiedergabe deaktivieren.

If SPIELSTART_NEU = JA Then                         'Geloestes Spiel anzeigen lassen:
  Call SpielAufloesen                               'Spiel aufloesen lassen.
Else                                                'Weiter beim letzten Spielstand:
  'nix machen.                                      'Normalerweise nichts machen,
  'Call SpielMischen                                'aber, je nach Belieben, Mischen.
End If

                                                    'Fehlerbehandlungsroutine:
Beenden:                                            '=========================
  Application.Cursor = xlDefault                    'Sanduhr ausblenden.
  Exit Function                                     'Funktion abbrechen.

Fehler:                                             'Fehlermarke - Fehlerbehandlung:
  SpielStarten = False                              'Returnwert = Fehler setzen.
  MsgBox Err.Description                            'Fehlermeldung ausgeben.
  Resume Beenden                                    'Bei Funktionsende weitermachen.
  Resume                                            'Nur fuer Testzwecke hinterlegt.
End Function                                        'Funktionsende.
```

Geräusch beim automatischen Auflösen des Puzzles in Funktion **SpielAufloesen** ergänzen:

```
Public Function SpielAufloesen() As Boolean
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION                        */
'*************************************************************************/
'*-----------------------------------------------------------------------*/
'*                                                                       */
'*      FUNKTION:       SpielAufloesen                                   */
'*                                                                       */
'*      BESCHREIBUNG:   Gibt die Ziffern 1 bis 16 in der richtigen Reihen-  */
'*                      folge aus und setzt die entsprechende Hintergrund-  */
'*                      farbe. Zum Schluss wird der letzte Spielstein in    */
'*                      Abhaengigkeit der globalen Konstante SPIELSTEIN_16  */
'*                      als Spielstein oder als Leerfeld eingetragen.       */
'*                                                                       */
'*      RETURN          Boolean ..... True = Funktion in Ordnung         */
'*                                    False = Fehler in Funktion         */
'*      PARAMETER:      keine                                            */
'*                                                                       */
'*-----------------------------------------------------------------------*/
Dim i                   As Integer                  'Allgemeine Zaehlvariable.
Dim iZeileStart         As Integer                  'Zeile aktueller Puzzleteil.
Dim iSpalteStart        As Integer                  'Spalte aktueller Puzzleteil.
Dim lFarbeAktuell       As Long

On Error GoTo Fehler                                'Bei Fehler zur Fehlermarke.
SpielAufloesen = True                               'Returnwert = alles OK setzen.

Set oTabellenblatt = Sheets(TABELLENBLATT)          'Akt. Tabellenblatt festlegen.
oTabellenblatt.Select                               'Tabellenblatt auswaehlen.
iZeileStart = 2                                     'Startposition fuer ersten
iSpalteStart = 2                                    'Puzzleteil festlegen.
Call SoundAbspielen(ALIAS_SOUND_AUFLOESEN)          'Aufloesesound abspielen.
For i = 1 To 16                                     'Die 16 Puzzelteile abarbeiten:
  'If (i Mod 2) = 1 Then                            'Nur wenn Anzahl waagrechte
                                                    'Steine ungerade ist, sonst:
  If (i Mod 2) + ((Int((i - 1) / 4)) Mod 2) = 1 Then '1, 3, 6, 8, 9, 11, 14, 16
    oTabellenblatt.Cells(iZeileStart, iSpalteStart).Interior.Color = ROT
  Else                                              '2, 4, 5, 7, 10, 12, 13, 15:
    oTabellenblatt.Cells(iZeileStart, iSpalteStart).Interior.Color = WEISS
  End If

  oTabellenblatt.Cells(iZeileStart, iSpalteStart) = i 'Die Zahl eintragen und
  iSpalteStart = iSpalteStart + 1                   'zur naechste Position:
  If iSpalteStart > 5 Then iSpalteStart = 2: iZeileStart = iZeileStart + 1
Next i

If SPIELSTEIN_16 = JA Then                          'Soll 16. Stein eingeblendet werden:
  Range("E5").Select:  ActiveCell.FormulaR1C1 = "16" 'dann Zahl 16 eintragen
  Selection.Interior.Color = ROT                    'und Hintergrund ROT machen.
Else                                                '16. Spielstein nicht einblenden:
  Range("E5").Select:  ActiveCell.FormulaR1C1 = Null 'dann NULL-Wert eintragen
  Selection.Interior.Color = SCHWARZ                'und Hintergrund SCHWARZ machen.
End If
Range("A1").Select                                  'Zelle A1 selektieren.


                                                    'Fehlerbehandlungsroutine:
Beenden:                                            '========================
  Application.Cursor = xlDefault                    'Sanduhr ausblenden.
  Exit Function                                     'Funktion abbrechen.
Fehler:                                             'Fehlermarke - Fehlerbehandlung:
  SpielAufloesen = False                            'Returnwert = Fehler setzen.
  MsgBox Err.Description                            'Fehlermeldung ausgeben.
  Resume Beenden                                    'Bei Funktionsende weitermachen.
  Resume                                            'Nur fuer Testzwecke hinterlegt.
End Function                                        'Funktionsende.
```

Geräusch beim Mischen des Puzzles in Funktion **SpielMischen** erweitern. Dabei sollte man zusätzlich berücksichtigen, daß der Mischvorgang in etwa genauso lange dauert wie das Abspielen des Geräusches. Dies kann man erreichen, indem die Millisekunden in der Sleep-Prozedur angepaßt werden oder die Anzahl der Verschiebungen.

```
Public Function SpielMischen() As Boolean
'************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION                       */
'************************************************************************/
'*----------------------------------------------------------------------*/
'*                                                                      */
'*      FUNKTION:       SpielMischen                                    */
'*                                                                      */
'*      BESCHREIBUNG:   Mischt die Teile des Schiebepuzzle durch, gibt das  */
'*                      Leerfeld aus und setzt die Anzahl der Spielzuege    */
'*                      auf 0.                                          */
'*                                                                      */
'*      RETURN          Boolean ..... True = Funktion in Ordnung        */
'*                                    False = Fehler in Funktion        */
'*      PARAMETER:      keine                                           */
'*                                                                      */
'*----------------------------------------------------------------------*/
Dim iZeile              As Integer                  'Aktuelle Zeile.
Dim iSpalte             As Integer                  'Aktuelle Spalte

On Error GoTo Fehler                                'Bei Fehler zur Fehlermarke.
SpielMischen = True                                 'Returnwert = alles OK setzen.

Call SoundAbspielen(ALIAS_SOUND_MISCHEN)            'Mischsound abspielen.
Set oTabellenblatt = Sheets(TABELLENBLATT)          'Akt. Tabellenblatt festlegen.
oTabellenblatt.Select                               'Tabellenblatt auswaehlen.

If ZUSCHAUEN = NEIN Then                            'Je nachdem wie es beliebt:
  Application.ScreenUpdating = False                'Bildschirmaktualisierung
End If                                              'ausschalten.
                                                    'Zuerst das Spiel aufloesen lassen
Call SpielAufloesen                                 'damit der 16. Stein auf E5 ist.
Range("E5").Select:  ActiveCell.FormulaR1C1 = Null  'Dort einen NULL-Wert eintragen
Selection.Interior.Color = SCHWARZ                  'und Hintergrund SCHWARZ machen.
Range("A1").Select                                  'Zelle A1 selektieren.

While iAnzahlZuege < 200                            '200 Verschiebungen durchfuehren:
  iZeile = 1 + Int(4 * Rnd) + 1                     'Zufaellige Zeile und
  iSpalte = 1 + Int(4 * Rnd) + 1                    'zufaellige Spalte festlegen.
  If SpielSteinVerschieben(iZeile, iSpalte) = True Then
    If ZUSCHAUEN = JA Then Sleep (15)               'Beim Zuschauen etwas warten.
    iAnzahlZuege = iAnzahlZuege + 1                 'Anzahl der zufaelligen
  End If                                            'Verschiebungen erhoehen.
Wend
iAnzahlZuege = 0                                    'Anzahl der Spielzuege ruecksetzen.
oTabellenblatt.Range("ZUEGE") = "Anzahl Züge bisher:  0000" 'und im Tabellenblatt ausgeben.
Call SoundAbspielen(ALIAS_SOUND_STARTEN)            'Startsound abspielen.

Application.ScreenUpdating = True                   'Bildschirmaktualisierung auf
                                                    'alle Faelle wieder einschalten.


                                                    'Fehlerbehandlungsroutine:
Beenden:                                            '=========================
  Application.Cursor = xlDefault                    'Sanduhr ausblenden.
  Exit Function                                     'Funktion abbrechen.
Fehler:                                             'Fehlermarke - Fehlerbehandlung:
  SpielMischen = False                              'Returnwert = Fehler setzen.
  MsgBox Err.Description                            'Fehlermeldung ausgeben.
  Resume Beenden                                    'Bei Funktionsende weitermachen.
  Resume                                            'Nur fuer Testzwecke hinterlegt.
End Function                                        'Funktionsende.
```

In der Funktion **TonUmschalten** müssen wir noch vier Programmzeilen Zeilen ergänzen: Dazu setzen wir die Aliasnamen für die Geräusche zurück bzw. lesen sie neu ein. Auch die Hintergrundmusik wird hier ein- bzw. ausgeschaltet!

```
Public Function TonUmschalten()
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION                        */
'*************************************************************************/
'*----------------------------------------------------------------------*/
'*                                                                      */
'*      FUNKTION:       TonUmschalten                                   */
'*                                                                      */
'*      BESCHREIBUNG:   Schaltet die Wiedergabe von Hintergrundmusik und    */
'*                      Geraeuschen aus bzw. ein.                       */
'*                                                                      */
'*      RETURN          nichts                                          */
'*                                                                      */
'*      PARAMETER:      keine                                           */
'*                                                                      */
'*----------------------------------------------------------------------*/
Dim i                   As Integer                  'Allgemeine Zaehlvariable.

Set oTabellenblatt = Sheets(TABELLENBLATT)          'Akt. Tabellenblatt festlegen.
oTabellenblatt.Select                               'Tabellenblatt auswaehlen.

'Sollte ein Shape (Form) nicht die Eigenschaft: TextFrame.Characters.Text
'haben, dann tritt ein Laufdzeitfehler auf. Um dieses zu verhindern, wird
'der Programmcode bei einem Fehler in der naechsten Programmzeile fortgesetzt.
'Genaugenommen wird das naechste Shape abgehandelt.

On Error Resume Next                                'Bei Fehler einfach weitermachen:
For i = 0 To oTabellenblatt.Shapes.Count            'Alle Formen im Tabellenblatt
  Err = 0                                           'durcharbeiten. Fehler 0 setzen.
  If oTabellenblatt.Shapes(i).TextFrame.Characters.Text = "Ton AUS" Then
    If Err = 0 Then                                 'Wenn Ton ausschalten:
      bSoundsAbspielen = False                      'Merken: Keine Tonwiedergabe.

      Call SpielSoundsRuecksetzen                   'Alle Sounds zuruecksetzen.
      Call HintergrundMusik_AUS                     'Hintergrundmusik stoppen.

      oTabellenblatt.Shapes(i).TextFrame.Characters.Text = "Ton EIN"
      Exit For                                      'Beschriftung aendern und
    End If                                          'Funktion beenden.

  ElseIf oTabellenblatt.Shapes(i).TextFrame.Characters.Text = "Ton EIN" Then
    If Err = 0 Then                                 'Wenn Ton einschalten:
      bSoundsAbspielen = True                       'Merken: Ton wiedergeben.

      Call SpielSoundsDefinieren                    'Alle Sounds neu definieren.
      Call HintergrundMusik_EIN                     'Hintergrundmusik starten.

      oTabellenblatt.Shapes(i).TextFrame.Characters.Text = "Ton AUS"
      Exit For                                      'Beschriftung aendern und
    End If                                          'Funktion beenden.

  Else                                              'Reserve fuer Erweiterungen:
    'nix machen
  End If
Next i
End Function                                        'Funktionsende.
```

Die Geräusche beim Verschieben hinterlegen wir in der Prozedur **Worksheet_BeforeDoubleClick**. Auch der Ton, falls das Puzzle gelöst wurde, kann dort hinterlegt werden:

```
Private Sub Worksheet_BeforeDoubleClick(ByVal Target As Range, Cancel As Boolean)
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION - PROZEDUR              */
'*************************************************************************/
'*-----------------------------------------------------------------------*/
'*                                                                       */
'*      FUNKTION:       Worksheet_BeforeDoubleClick                      */
'*                                                                       */
'*      BESCHREIBUNG:   Diese Prozedur wird bei einem Doppelklick, irgendwo */
'*                      im Tabellenblatt, aufgerufen.                    */
'*                      Wird in dieser Prozedur der Parameter: Cancel    */
'*                      auf True gesetzt, so wird die Prozedur abgebrochen. */
'*                                                                       */
'*      RETURN          nichts                                           */
'*                                                                       */
'*      PARAMETER:      Range ............ Doppelt geklickte Zelle       */
'*                      Boolean .......... True = Funktion abbrechen     */
'*                                                                       */
'*-----------------------------------------------------------------------*/
Dim stAnzahlZuege       As String
On Error GoTo Fehler                            'Bei Fehler zur Fehlerbehandlung.
If SPIELSTEIN_16 = JA Then
  If Spielende = True Then GoTo Spielende       'Kontrolle auf Spielende:
End If

If Target.Row > 1 And Target.Row < 6 Then       'Nur Zellen, welche Teil des
  If Target.Column > 1 And Target.Column < 6 Then 'Spielbrettes sind abhandeln:
    Set oTabellenblatt = Sheets(TABELLENBLATT)  'Akt. Tabellenblatt festlegen.

    If SpielSteinVerschieben(Target.Row, Target.Column) = True Then 'Wenn verschoben,
      Call SoundAbspielen(ALIAS_SOUND_SCHIEBEN)     'Schiebesound abspielen.
      iAnzahlZuege = iAnzahlZuege + 1               'Anzahl der Spielzuege erhoehen.
      stAnzahlZuege = Right("0000" & Trim(Str(iAnzahlZuege)), 4) 'Anzahl als String.
      oTabellenblatt.Range("ZUEGE") = "Anzahl Züge bisher:  " & stAnzahlZuege
    Else                                        'und im Tabellenblatt ausgeben.
      Call SoundAbspielen(ALIAS_SOUND_FEHLER)   'Fehlersound abspielen wenn
    End If                                      'nicht verschiebbar.

    If Spielende = True Then                    'Kontrolle auf Spielende:
      If bSoundsAbspielen = True Then           'Wenn Sounds wiedergeben, dann
        Call SoundAbspielen(ALIAS_SOUND_FERTIG) 'Fertigsound abspielen.
      End If
      MsgBox "GRATULATION! Sie haben das Puzzle gelöst!", _
             vbOKOnly + vbInformation + vbDefaultButton1, PROGRAMMNAME
    End If                                      'Falls Puzzle geloest wurde,
  End If                                        'dann entsprechende Meldung
End If                                          'ausgeben.

Spielende:
Range("A1").Select                              'Zelle A1 selektieren.
Cancel = True                                   'Prozedur abbrechen.

                                                'Fehlerbehandlungsroutine:
Beenden:                                        '=========================
  Application.Cursor = xlDefault                'Sanduhr ausblenden.
  Exit Sub                                      'Prozedur abbrechen.

Fehler:                                         'Fehlermarke - Fehlerbehandlung:
  MsgBox Err.Description                        'Fehlermeldung ausgeben.
  Resume Beenden                                'Bei Funktionsende weitermachen.
  Resume                                        'Nur fuer Testzwecke hinterlegt.
End Sub                                         'Prozedurende.
```

Und zum Schluß nicht vergessen: Hintergrundmusik beim Schließen der Datei sofort stoppen:

```
Private Sub Workbook_BeforeClose(Cancel As Boolean)
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION - PROZEDUR              */
'*************************************************************************/
'*--------------------------------------------------------------------------*/
'*                                                                          */
'*      FUNKTION:       Workbook_BeforeClose                                */
'*                                                                          */
'*      BESCHREIBUNG:   Diese Prozedur wird vor den Schliessen der Excel-   */
'*                      Datei bzw. vor dem Beenden von Excel aufgerufen.    */
'*                      Wird in dieser Prozedur der Parameter: Cancel       */
'*                      auf True gesetzt, so wird das Schliessen bzw. das   */
'*                      Beenden abgebrochen - Excel-Datei bleibt geoeffnet. */
'*                                                                          */
'*      RETURN          nichts                                              */
'*                                                                          */
'*      PARAMETER:      Boolean .......... True = Funktion abbrechen        */
'*                                                                          */
'*--------------------------------------------------------------------------*/
On Error GoTo Fehler                            'Bei Fehler zur Fehlerbehandlung.

Call HintergrundMusik_AUS                       'Hintergrundmusik beenden.
Call SpielAnsichtAktualisieren(NEIN)            'Umschalten auf Entwickleransicht.

If AUTOSPEICHERN = JA Then                      'Je nach Definition im Modul Globales
  Application.DisplayAlerts = False             'die aktuelle Arbeitsmappe automatisch
  ActiveWorkbook.Save                           'speichern oder alle Aenderungen
  Application.DisplayAlerts = True              'verwerfen. Dabei die Ausgabe von
                                                'Fehler- & Hinweismeldungen deaktivieren.
ElseIf FRAGESPEICHERN = NEIN Then               'Alle Aenderungen verwerfen:
  ThisWorkbook.Saved = True                     '(Excel mitteilen, dass Datei gespeichert
Else                                            'ist - auch wenn es nicht stimmt - daher
  'In diesem Else-Zweig stellt Excel die Frage  'versucht Excel gar nicht zu speichern.)
  'ob die Aenderungen gespeichert werden sollen.
End If
                                                'Nach Programmabsturz kann dieser
Application.EnableEvents = True                 'Befehl sehr hilfreich sein!


                                                'Fehlerbehandlungsroutine:
Beenden:                                        '=========================
  Application.Cursor = xlDefault                'Sanduhr ausblenden.
  Exit Sub                                      'Prozedur abbrechen.

Fehler:                                         'Fehlermarke - Fehlerbehandlung:
  MsgBox Err.Description                        'Fehlermeldung ausgeben.
  Resume Beenden                                'Bei Funktionsende weitermachen.
  Resume                                        'Nur fuer Testzwecke hinterlegt.
End Sub                                         'Prozedurende.
```

Natürlich die Hintergrundmusik auch dann stoppen, wenn man zu einer anderen Excel-Datei wechselt.

```
Private Sub Workbook_Deactivate()
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION - PROZEDUR              */
'*************************************************************************/
'*--------------------------------------------------------------------------*/
'*                                                                          */
'*      FUNKTION:       Workbook_Deactivate                                 */
'*                                                                          */
'*      BESCHREIBUNG:   Diese Prozedur wird aufgerufen, wenn man aus dieser */
'*                      Excel-Datei zu einer anderen geoeffneten Excel-Datei*/
'*                      wechselt. Egal ob ein Excel gestartet wurde und     */
'*                      damit mehrere Dateien geoeffnet sind oder ob Excel  */
```

```
'*                      mehrmals gestartet wurde und mit jedem Excel nur    */
'*                      eine Datei geoeffnet wurde.                         */
'*                                                                          */
'*                      Dieses Prozedur soll dafuer sorgen, dass die        */
'*                      Anzeigeeinstellungen fuer diese Datei NICHT in die  */
'*                      anderen Excel-Datei uebernommen wird.               */
'*                                                                          */
'*      RETURN          nichts                                              */
'*                                                                          */
'*      PARAMETER:      keine                                               */
'*                                                                          */
'*--------------------------------------------------------------------------*/
On Error GoTo Fehler                            'Bei Fehler zur Fehlerbehandlung.

Call HintergrundMusik_AUS                       'Hintergrundmusik beenden.

Application.DisplayFullScreen = False           'Vollbildmodus ausschalten.
Application.WindowState = lExcelFensterGroesse  'Excel Fenstergroesse wiederherstellen.

                                                'Fehlerbehandlungsroutine:
Beenden:                                        '=========================
  Application.Cursor = xlDefault                'Sanduhr ausblenden.
  Exit Sub                                      'Prozedur abbrechen.

Fehler:                                         'Fehlermarke - Fehlerbehandlung:
  MsgBox Err.Description                        'Fehlermeldung ausgeben.
  Resume Beenden                                'Bei Funktionsende weitermachen.
  Resume                                        'Nur fuer Testzwecke hinterlegt.
End Sub                                         'Prozedurende.
```

Und wenn man wieder zur Spieldatei zurückkehrt, dann eventuell die Hintergrundmusik wieder abspielen lassen:

```
Private Sub Workbook_Activate()
'****************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION - PROZEDUR                */
'****************************************************************************/
'*--------------------------------------------------------------------------*/
'*                                                                          */
'*      FUNKTION:       Workbook_Activate                                   */
'*                                                                          */
'*      BESCHREIBUNG:   Diese Prozedur wird aufgerufen, wenn man aus einer  */
'*                      anderen geoeffneten Excel-Datei zu dieser zurueck-  */
'*                      wechselt. Egal ob ein Excel gestartet wurde und     */
'*                      damit mehrere Dateien geoeffnet sind oder ob Excel  */
'*                      mehrmals gestartet wurde und mit jedem Excel nur    */
'*                      eine Datei geoeffnet wurde.                         */
'*                                                                          */
'*                      Dieses Prozedur soll dafuer sorgen, dass die        */
'*                      Anzeigeeinstellungen fuer diese Datei automatisch   */
'*                      wiederhergestellt werden.                           */
'*                                                                          */
'*      RETURN          nichts                                              */
'*                                                                          */
'*      PARAMETER:      keine                                               */
'*                                                                          */
'*--------------------------------------------------------------------------*/
On Error GoTo Fehler                            'Bei Fehler zur Fehlerbehandlung.


If bSoundsAbspielen = True Then                 'Wenn Soundwiedergabe aktiviert:
  Call HintergrundMusik_EIN                     'Hintergrundmusik wiedergeben.
End If
```

```
Call SpielAnsichtAktualisieren(SPIELANSICHT)        'Umschalten auf Entwickleransicht
                                                    'oder auf Spielansicht, je nach
                                                    'Definition im Modul Globales.
                                                    'Wenn gewuenscht, dann
Application.WindowState = xlMaximized               'Excel Fenster immer maximieren.

                                                    'Fehlerbehandlungsroutine:
Beenden:                                            '=========================
  Application.Cursor = xlDefault                    'Sanduhr ausblenden.
  Exit Sub                                          'Prozedur abbrechen.

Fehler:                                             'Fehlermarke - Fehlerbehandlung:
  MsgBox Err.Description                            'Fehlermeldung ausgeben.
  Resume Beenden                                    'Bei Funktionsende weitermachen.
  Resume                                            'Nur fuer Testzwecke hinterlegt.
End Sub                                             'Prozedurende.
```

4.9.) Zusatzfunktionen

Um das Spiel zu optimieren können noch einige Erweiterungen ergänzt werden. Ich bevorzuge es in jedem Spiel eine sogenannte "Flucht-Taste" zu hinterlegen, vorzugsweise die Escape-Taste (ESC). Drückt man diese Taste, dann soll das Spiel sofort beendet werden, ohne Nachfrage und ohne Speichern, einfach nur alles vom Bildschirm verschwinden zu lassen (wenn der Chef reinkommt). Manchmal beende ich dabei auch gleich die gesamte Applikation (hier Excel), jedoch nicht in dieser Funktion, sondern in der Prozedur **Workbook_BeforeClose**.

Diese Zusatzfunktion werden wir im Modul **Standards** ablegen und könnte so aussehen:

```
Public Function Fluchttaste()
'*************************************************************************/
'*      INTERNES UNTERPROGRAMM - INTERNE FUNKTION                        */
'*************************************************************************/
'*----------------------------------------------------------------------*/
'*                                                                      */
'*      FUNKTION:       Fluchttaste                                     */
'*                                                                      */
'*      BESCHREIBUNG:   Schliesst die aktuelle Excel-Datei sofort und ohne  */
'*                      zu speichern. Zuvor werden noch alle Hintergrund-   */
'*                      prozesse beendet und die Entwickleransicht wird     */
'*                      wieder hergestellt. Excel selbst bleibt geoeffnet.  */
'*                                                                      */
'*      RETURN          nichts                                          */
'*                                                                      */
'*      PARAMETER:      keine                                           */
'*                                                                      */
'*----------------------------------------------------------------------*/
On Error Resume Next                                'Bei Fehler einfach weitermachen.

bFluchttaste = True                                 'ESC wurde jetzt gedrueckt.

Call HintergrundMusik_AUS                           'Hintergrundmusik beenden.
Call SpielSoundsRuecksetzen                         'Sounds ruecksetzen.
Call SpielAnsichtAktualisieren(NEIN)                'Umschalten auf Entwickleransicht.

ActiveWorkbook.Close savechanges:=False             'Excel-Datei sofort schliessen.

End Function                                        'Funktionsende.
```

Wie sie bereits aus obigem Programmcode ersehen können verwenden wir eine neue zusätzliche globale Variable welche wir nun im Modul **Globales** ergänzen:

```
Global bFluchttaste                    As Boolean 'Fluchttaste wurde gedrueckt?
```

Der Variable weisen wir den Wert **False** (Fluchttaste wurde nicht gedrückt) zu, gleich zu Spielbeginn in der Funktion **SpielStarten** im Modul **Funktionen**:

```
bFluchttaste = False                           'ESC wurde noch nicht gedrueckt.
```

Nun müssen wir die Funktion **Fluchttaste** noch der Escape-Taste zuordnen. Das erledigen wir in der Prozedur **Workbook_Activate**. Wir ergänzen dort folgende Programmzeile:

```
Application.OnKey "{ESC}", "Fluchttaste"       'Wenn Escape-Taste gedrueckt, dann
                                               'Funktion: Fluchttaste ausfuehren.
```

Damit die Escape-Taste in anderen geöffneten Excel-Dateien ihre normale Funktion behält, müssen wir sie beim Dateiwechsel wieder deaktivieren. Das erledigen wir in der Prozedur **Workbook_Deactivate**:

```
Application.OnKey "{ESC}"                      'Funktionszuweisung an ESC-Taste
                                               'wieder entfernen.
```

Damit das Spiel wirklich so rasch als möglich vom Bildschirm verschwindet, müssen wir uns noch die Prozeduren **Workbook_BeforeClose** (wird als erstes beim Schließen der Datei ausgeführt) und **Workbook_Deactivate** (wird zusätzlich nach dem Schließen auch noch ausgeführt) anschauen.

Hier unterscheiden wir nun anhand der Variable, ob die Fluchttaste gedrückt wurde oder nicht, ob das Spiel sofort beendet wird oder ob der vordefinierte Ablauf beim Schließen Anwendung finden soll. In der Prozedur **Workbook_BeforeClose** ergänzen wir:

```
If bFluchttaste = True Then                  'Wurde die Escape-Taste gedrueckt,
  Application.Quit                           'dann Excel beenden.
Else
  Call HintergrundMusik_AUS                  'Hintergrundmusik beenden.
  Call SpielAnsichtAktualisieren(NEIN)       'Umschalten auf Entwickleransicht.

  If AUTOSPEICHERN = JA Then                 'Je nach Definition im Modul Globales
    Application.DisplayAlerts = False        'aktuelle Arbeitsmappe automatisch
    ActiveWorkbook.Save                      'speichern oder alle Aenderungen
    Application.DisplayAlerts = True         'verwerfen. Dabei die Ausgabe von
                                             'Fehler- und Hinweismeldungen
                                             'deaktivieren.
  ElseIf FRAGESPEICHERN = NEIN Then          'Alle Aenderungen verwerfen:
    ThisWorkbook.Saved = True                'Excel mitteilen Datei = gespeichert.
  Else                                       'auch wenn es nicht stimmt - daher
  'In diesem Else-Zweig stellt Excel die Frage 'versucht Excel nicht zu speichern.
  'ob die Aenderungen gespeichert werden sollen.
  End If
                                             'Nach Programmabsturz kann dieser
  Application.EnableEvents = True            'Befehl sehr hilfreich sein!
End If
```

Ob man die Programmzeile **Application.Quit** verwendet oder unter Kommentar setzt ist Geschmackssache.

Zum Schluß ergänzen wir noch die Prozedur **Workbook_Deactivate**:

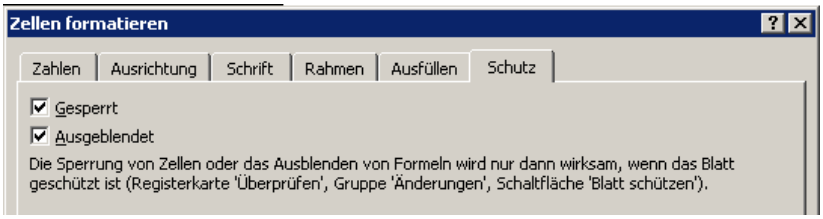
```
Application.OnKey "{ESC}"                          'Funktionszuweisung an ESC-Taste
                                                   'wieder entfernen.
If bFluchttaste = False Then                       'Wenn Fluchttaste nicht gedrückt:
  Call HintergrundMusik_AUS                        'Hintergrundmusik beenden.

  Application.DisplayFullScreen = False            'Vollbildmodus ausschalten.
  Application.WindowState = lExcelFensterGroesse'Excel Fenstergroesse wiederherstel.
End If
```

5.) Schlußbemerkungen

Für ein Spiel sollten Sie auch ein **Icon** erstellen, welches man einer Verknüpfung zuweisen kann. Zum Erstellen eigener Icons verwende ich den kostenlosen **Greenfish Icon Editor Pro**. Damit wurde auch das Icon für das Schiebepuzzle erstellt:

Bei komplexeren Spielen sollte auch eine detaillierte **Spielbeschreibung** erstellt werden.

Definieren Sie den Schutz für die Zellen im Tabellenblatt. Am besten zuerst alle Zellen markieren und dann den Schutz für alle Zellen aktivieren:



Danach jene Zellen auswählen, welche vom Programm beschrieben werden, hier nur die Zelle mit der Punkteanzahl, und dort das Häkchen bei "Gesperrt" entfernen.

Aktivieren Sie den Blattschutz in Excel. Im Register **Überprüfen**, Symbol **Blatt schützen**:

Beim Schiebepuzzle müssen zusätzlich die Berechtigung: **Zellen formatieren** (da wir die Hintergrundfarbe der Steine verändern) und **Objekte bearbeiten** (da wir die Beschriftung in einem Shape ändern) abgehakt werden.

Schützen Sie zusätzlich Ihr **VBA-Projekt** mit einem Paßwort wenn Sie den Programmcode geheim halten wollen. In Visual Basic das VBA-Projekt mit der rechten Maustaste anklicken -> Eigenschaften von VBA-Projekt… -> Registerblatt: Schutz. Dort zweimal das Kennwort eingeben und Klick auf OK.

So das wär´s dann wieder Mal – viel Spaß beim Nachprogrammieren oder Überarbeiten des Spieles.