EVSLabels 1.0: ein Tool zur automatischen Erstellung von SPSS-Setups für Scientific Use Files der EVS 2003

Grund, Thomas

ZUMA Methodenbericht 2004/ 05

EVSLabels 1.0
Ein Tool zur automatischen Erstellung von SPSS-Setups für Scientific Use Files der EVS 2003[1]

Thomas Grund

April 2004
ISSN 1610-9953

[1] Dieses Tool entstand während eines Praktikumaufenthaltes bei Dr. Georg Papastefanou, ZUMA, German Micro Data Lab, Mannheim.

1. Einleitung

Die Einkommens- und Verbrauchsstichprobe (kurz: EVS) wird seit 1962/63 in der Regel alle fünf Jahre erhoben und ist eine wichtige amtliche Statistik über die Lebensverhältnisse privater Haushalte in Deutschland. Sie liefert u.a. statistische Informationen über die Einkommens-, Vermögens- und Schuldensituation, sowie die Konsumausgaben privater Haushalte. Das Statistische Bundesamt bietet der wissenschaftlichen Gemeinde verschiedene faktisch anonymisierte Substichproben (Scientific Use Files) der EVS 93, EVS 98 und EVS 03 an, die sich jeweils im Umfang der Substichprobe und der Anzahl der enthaltenen Merkmale unterscheiden. Durch diese Regelung ist es möglich, die Daten auf Grundlage des Gesetzes über die Statistik der Wirtschaftsrechnungen privater Haushalte und des Bundes-statistikgesetzes für wissenschaftliche Forschung verwenden zu können.

Im Rahmen einer Vereinbarung zwischen dem Statistischen Bundesamt und der GESIS können diese Daten zu Sonderkonditionen (eine Gebühr von jeweils 65 EUR) direkt online unter http://www.destatis.de/shop oder unter folgender Kontaktadresse vom Statistischen Bundesamt bestellt werden:

Statistisches Bundesamt
Zweigstelle Bonn
Gruppe IXC
Graurheindorfer Str. 198
53117 Bonn
Tel.: 01888 / 644 – 8855, Fax: 01888 / 644 – 8970
e-mail: heidrun.wolter@destatis.de

Im Lieferumfang sind die Rohdaten und eine Datensatzbeschreibung enthalten. Die Rohdaten werden ab der EVS 03 im CSV-Format (Comma Separated Value) geliefert und die Datensatzbeschreibung als Word-Dokument. In letzterer sind die Variablen-Labels und Value-Labels aufgelistet. Die Aufbereitung der Daten für SPSS besteht nun darin, die Rohdaten einzulesen und die Labels der Variablen und Values zuzuweisen. Bisher mussten diese Labels manuell eingetragen werden. Bei mehreren hundert Variablen und den dazugehörigen Values dauerte dies sehr lange. Außerdem waren Übertragungsfehler aus der Datensatzbeschreibung nicht auszuschließen.
Dieser Vorgang konnte nun mit dem vorliegenden Tool automatisiert werden. Dabei werden die Informationen, die vorher abgetippt werden mussten, direkt aus der Datensatzbeschreibung des Statistischen Bundesamtes (siehe Abbildung 1) ausgelesen und in einer SPSS- Syntax-Datei gespeichert.

Abb. 1: Beispielseite der Datensatzbeschreibung

Statistisches Bundesamt - Maschinelle Aufbereitung -	Datensatzbeschreibung	
Aufgabengebiet:	Einkommens- und Verbrauchsstichprobe 2003 Grundfile 2 (98% Stichprobe aus Geld- und Sachvermögen)	Blatt 1 von 17 Datum: 10.12.200
Datensatz-Nr./ -Name:	Datensatz-Nr. / - name lt. Ersteller:	Stand:
Materialbezeichnung(en)	EV_GS03A	Bearbeiter: Wolter Land:
ggf. Sortierung: (Archivmaterial)		Berichtszeitraum: 2003
Bemerkungen:	Das Material wird im CSV-Format bereitgestellt Delimiter (;) Der erste Datensatz enthält die Spaltennamen der Datenzeilen	Satzformat F Satztyp 2): Satzlänge in Bytes: 444

Feldbez. EF - Nr.	Satzstellen von - bis	Anzahl	Feldformat 1) allg.	Intern	Inhalt / Bemerkungen
EF1	1	1	C	ALN	leer
EF2	2 - 8	7			
EF2U1	2 - 3	2	C	ALN	Bundesland 01 = Schleswig-Holstein 02 = Hamburg 03 = Niedersachsen 04 = Bremen 05 = Nordrhein-Westfalen 06 = Hessen 07 = Rheinland-Pfalz 08 = Baden-Württemberg 09 = Bayern 10 = Saarland 11 = Berlin-West 12 = Brandenburg 13 = Mecklenburg-Vorpommern 14 = Sachsen 15 = Sachsen-Anhalt 16 = Thüringen 22 = Berlin-Ost
EF2U2	4 - 8	5	C	ALN	Laufende Nummer
EF3	9 - 10	2	C	NOV02K00	Anzahl der Personen im Haushalt (01 - 09) 01 - 09 = Anzahl 09 = 9 Personen und mehr I. Angaben über Haushaltsmitglieder
EF4	11 - 27	17			Angaben zur 1. Person im Haushalt (HEB)
EF4U1	11	1	C	ALN	Stellung innerhalb des Haushalts 1 = Haupteinkommensbezieher(in) - HEB -
EF4U2	12	1	C	ALN	Geschlecht 1 = männlich 2 = weiblich
EF4U3	13 - 16	4	C	ALN	Geburtsjahr 1983 = 18 bis 20 Jahre 1982 = 21 Jahre 1981 = 22 Jahre usw. bis 1918 = 85 Jahre und älter
EF4U4	17	1	C	ALN	Familienstand 1 = ledig 2 = verheiratet 3 = verwitwet

2. Ein Tool zur automatischen Erstellung von SPSS-Setups für die Einkommens- und Verbrauchsstichprobe 2003: EVSLabels 1.0

Das Tool EVSLabels 1.0 vereinfacht nun die Erstellung eines SPSS-Setups für die Einkommens- und Verbrauchsstichprobe 2003. Die mit EVSLabels 1.0 erzeugbare SPSS-Syntax-Datei erfüllt dabei die Aufgaben: Hinzufügen der Variablen- und Value-Labels und Umbenennung der Variablen nach wissenschaftlichen Standards.
Bisher musste man eine solche SPSS-Syntax-Datei manuell erstellen und auf den eigenen Datensatz anpassen. EVSLabels 1.0 stellt nun eine Methode zur Verfügung, die Informationen zu den Variablen-Labels und Value-Labels direkt aus den Datensatzbeschreibungen des Statistischen Bundesamtes für die Einkommens- und Verbrauchsstichprobe 2003 auszulesen. Sie müssen zunächst lediglich die Rohdaten im CVS-Format über File → Open → Data einlesen. Hier ist darauf zu achten, dass Sie die Option „variable names included at the top of your file“ auf „yes“ setzen. Alle weiteren Einstellungen können so belassen werden.

EVSLabels 1.0 nutzt feste Strukturen in den Datensatzbeschreibungen des Statistischen Bundesamtes und wurde als Word-VBA Makro realisiert.

Im Folgenden soll der Quellcode von EVSLabels 1.0 offengelegt und kommentiert werden. EVSLabels 1.0 besteht im Wesentlichen aus drei Elementen:

1. Dem Dokument EVSLabels: Hier finden sich alle wesentlichen Methoden zum Auslesen der Informationen aus der Datensatzbeschreibung. Die einzige aufrufbare Methode ist createSpssSetup. Als Parameter müssen bei dieser Methode ein String mit dem Pfad und Namen der Datensatzbeschreibung und ein String mit dem gewünschten Pfad und Namen der zu erstellenden Syntax-Datei übergeben werden. (EVSLabels.cls)

2. Den Klassen ValueEntry und VariableEntry: Sie werden benötigt, um die Labels in einer internen Datenstruktur zu speichern. Eine Instanz der Klasse ValueEntry repräsentiert hier einen bestimmten Value-Eintrag und hat als Eigenschaft den Code und das Label des Values. Für jeden neuen Value-Eintrag wird ein solches Objekt angelegt.

Eine Instanz der Klasse VariableEntry speichert alle Informationen einer Variablen, d.h. ihren Code, ihr Label und die für sie gültigen Value-Einträge. Der Code entspricht der Feldbezeichnung der Variablen in der Datensatzbeschreibung des Statistischen Bundesamtes. Dieser Schlüssel dient als eindeutiges Identifizierungsmerkmal einer Variablen (ValueEntry.cls und VariableEntry.cls)

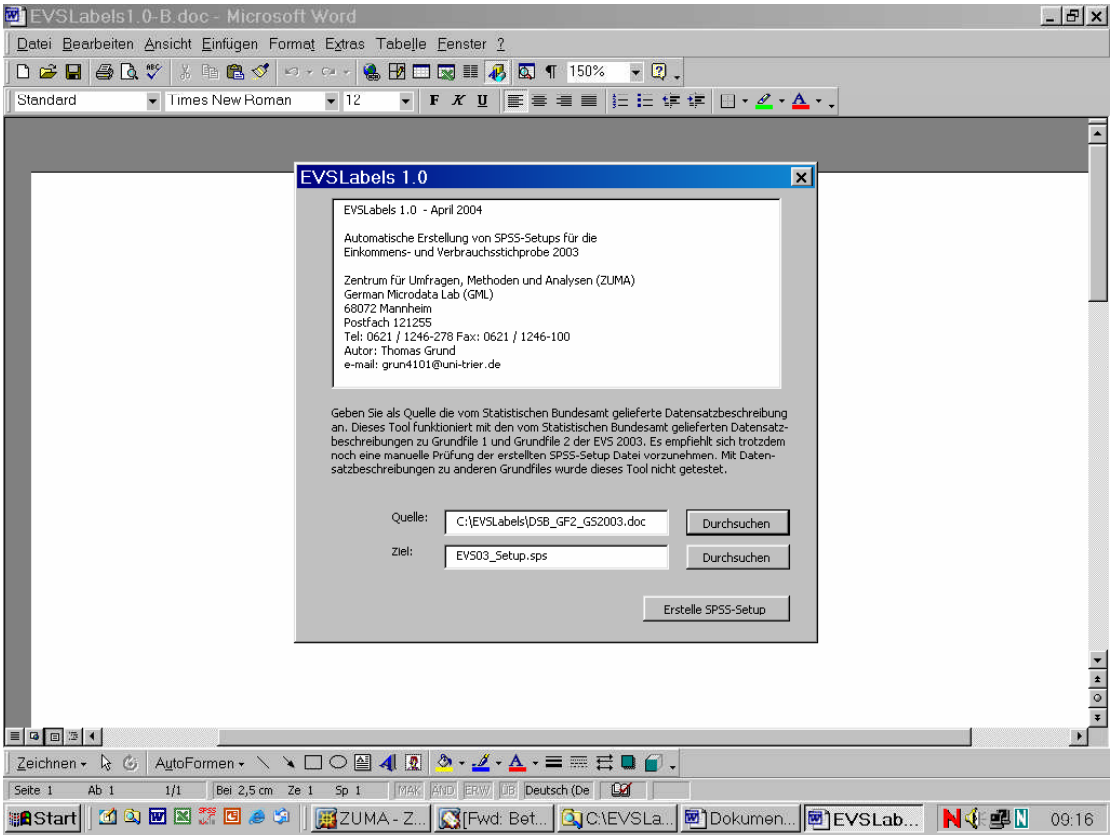
3. Der Form EVSLabelsForm. In der Benutzeroberfläche lässt sich der Name und der Pfad der Datensatzbeschreibung eintragen, die als Quelle zur Erstellung eines SPSS-Setups benutzt werden soll. Hier lässt sich auch der gewünschte Name und Pfad der zu erstellenden Syntax-Datei angeben. Nach Drücken des „Erstelle SPSS-Setup“ Knopfes ruft die Form die Methode EVSLabels.createSpssSetup mit den Parametern Name der Datensatzbeschreibung und Name der zu erstellenden SPSS-Syntax Datei auf. Falls die Datensatzbeschreibung nicht zum Auslesen der Informationen geeignet ist oder keine Angaben zu Quelle und Ziel gemacht wurden, werden Fehlermeldungen angezeigt. (EVSLabelsForm.frm)

3. Anwendung von EVSLabels 1.0

Um die Anwendung zu vereinfachen, wurde EVSLabels als Word-VBA Makro in eine eigene Word-Datei (EVSLabels 1.0.doc) gekapselt und mit einer Benutzeroberfläche versehen. Damit das Makro ausgeführt werden kann, müssen Sie in Word zunächst unter „Extras, Makro, Sicherheit“ die Sicherheitsstufe Mittel auswählen. Wenn hier „hoch“ eingestellt wurde, verhindert Word aus Sicherheitsgründen das Ausführen aller Makros. Beim Starten der Datei EVSLabels 1.0.doc stellt Word nun fest, dass sich in diesem Dokument ein Makro befindet und fragt, ob es aktiviert werden soll. Mit dem Bejahen dieser Nachfrage öffnet sich beim Starten der Word-Datei EVSLabels 1.0.doc ein Dialog (siehe Abbildung 2), in dem Sie die Quelle der Datensatzbeschreibung im Word-Format und den gewünschten Pfad und Namen der zu erstellenden Syntax-Datei eintragen müssen. Das Auslesen der Daten aus der Datensatzbeschreibung können Sie nun mit dem Knopf „Erstelle SPSS-Setup“ starten.

Das Programm erstellt jetzt automatisch eine neue SPSS-Syntax-Datei mit dem eingetragenen Namen.

Abb. 2: Screenshot EVSLabels 1.0



Die erstellte Syntax-Datei kann nun in SPSS ausgeführt werden, um die bereits geladenen Rohdaten mit Variablen-Labels und Value-Labels zu versehen.
EVSLabels 1.0 wurde mit den Datensatzbeschreibungen zu Grundfile 1 (DSB_GF1_EI2003.doc) und Grundfile 2 (DSB_GF2_GS2003.doc) vom Statistischen Bundesamt getestet. Bei anderen Datensatzbeschreibungen müssen eventuell Teile des Makros angepasst werden (siehe hierzu die Kommentare im Quellcode).

Bei Fragen oder Anregungen zum Code oder zur Anwendung des Programms wenden Sie sich bitte an:

Georg Papastefanou
Zentrum für Umfragen, Methoden und Analysen (ZUMA)
German Microdata Lab
Postfach 12 21 55
68072 Mannheim
e-mail: papastefanou@zuma-mannheim.de

Thomas Grund
Universität Trier, FB IV
Lehrstuhl Prof. Dr. Jäckel
e-mail: grun4101@uni-trier.de

Anhang: Quellcode

Dokument EVSLabels

```
' ************************************************************************
' *
' *                      EVSLabels
' *                     Version 1.0
' *                    vom 01.04.2004
' *
' *          Zentrum fuer Umfragen, Methoden und Analysen (ZUMA)
' *          German Microdata Lab (GML)
' *          Postfach 12 21 55
' *          68072 Mannheim
' *          Tel: 0621 / 1246-278 Fax: 0621 / 1246-100
' *          Autor: Thomas Grund
' *          e-mail: grun4101@uni-trier.de
' *
' *          Dokument: EVSLabels
' *
' ************************************************************************


Option Explicit
Option Base 1

' Diese Konstante gibt an, wie viele Zusätze (z.B. „ – 1. Person im Haushalt“)
' vergeben werden sollen. Diese Behandlung ist notwendig, da in der
' Datensatzbeschreibung die Variablen zu den Angaben der verschiedenen Personen
' eines Haushaltes (z.B. Stellung im Haushalt, Geschlecht, usw.) ohne einen solchen
' Zusatz ausgestattet sind. Die Zuweisung eines Zusatzes erfolgt über die
' Feldbezeichnung in der Datensatzbeschreibung (z.B.: "EFU4*" => " - 1. Person im
' Haushalt" usw.) Bei Grundfile 1 und Grundfile 2 werden neun Zusätze vergeben.
Const additionsDim = 9

' Diese Konstante gibt einen Maximalwert an Variablen an, denen zwar bestimmte
' Value-Labels zugeordnet sind, die aber in der Datensatzbeschreibung nur mit einem
' Vermerk aufgenommen sind, dass sie diesbezüglich wie andere Variablen behandelt
' werden sollen. So sind nämlich ab der dritten Person in einem Haushalt die Value-
' Labels in der Datensatzbeschreibung des Statistischen Bundesamtes nicht mehr
' explizit aufgeführt.
Const similarValuesDim = 90

' Die Datensatzbeschreibungen des Statistischen Bundesamtes weisen eine feste
' Struktur auf. Die Informationen sind in Tabellen abgelegt. Nach einem gleich-
' bleibenden Kopf finden sich die Einträge für die Variablen-Labels und Value-Labels
' in der Spalte „Inhalt/Bemerkungen“. Diese Konstante gibt den Index der Zelle an, ab
' der mit solchen Einträgen gerechnet werden kann.
Const inhaltStart = 81
```

```
' Diese Konstante gibt die Anzahl an Zellen zwischen zwei aufeinanderfolgenden
' Zellen der Spalte „Inhalt/Bemerkungen“ an.
Const nextCell = 9

' Diese Konstante gibt an wie viele Zellen vor der Spalte „Inhalt/Bemerkungen“ die
' Spalte „Feldbezeichnung“ in der Datensatzbeschreibung zu finden ist. Die Werte in
' dieser Spalte werden benutzt um die Einträge in der Spalte Inhalt/Bemerkungen in
' Variablen-Labels und Value-Labels zu unterscheiden.
Const feldbezBeforeInhalt = 7

' In dieser Datenstruktur werden alle erkannten Variablen gespeichert. Für jede
' Variable wird ein VariableEntry-Objekt angelegt und mit dem Code, Label und der
' Feldbezeichnung (Key) dieser Variable versehen. Jedes VariableEntry enthält eine
' Liste mit Einträgen für die Value-Labels dieser Variablen.
Dim vars As New Collection

' In diesem Feld werden die Informationen zu den anzufügenden Zusätzen
' eingetragen. In der ersten Dimension werden Bausteine von Feldbezeichnungen
' abgelegt, die eine Gruppe von Variablen eindeutig identifiziert (z.B. alle Angaben
' zur 1. Person im Haushalt haben eine Feldbezeichnung (key) der mit „EF4U“
' beginnt). In der zweiten Dimension wird ein String mit dem anzufügenden Zusatz
' abgespeichert (z.B. „1. Person im Haushalt“).
Dim additions(additionsDim, 2) As String

' In diesem Feld werden Informationen zu den Variablen gespeichert, die bezüglich
' ihrer Value-Labels genauso wie bestimmte andere Variablen behandelt werden
' sollen. In der ersten Dimension werden die Feldbezeichnungen (keys) von allen
' Variablen eingefügt, deren Value-Labels nicht explizit aufgeführt sind. In der
' zweiten Dimension werden die keys  der Variablen eingetragen, die bezüglich der
' Value-Labels als Vorbild dienen sollen.
Dim similarValues(similarValuesDim, 2) As String

' Diese Prozedur initialisiert die Variable additions. Bei der Datensatzbeschreibung zu
' Grundfile 1 und Grundfile 2 muss den Variablen mit den Feldbezeichnungen EF4U*
' bis EF12U* ein Zusatz mitgegeben werden. Hier werden die Angaben zu den
' verschiedenen Personen eines Haushaltes abgelegt. Um die Angaben der Personen
' voneinander unterscheiden zu können werden die Zusätze „ – 1.Person im Haushalt“
' bis „ – 9.Person im Haushalt“ initialisiert.  Bei anderen Grundfiles müsste die
' Variable additions und diese Prozedur eventuell angepasst werden.
Private Sub initAdditions()
   Dim i

   For i = 1 To additionsDim
      additions(i, 1) = "F" & (i + 3) & "U"
      additions(i, 2) = " - " & i & ". Person im HH."
   Next
End Sub
```

```
' Diese Prozedur initialisiert die Variable similarValues. Bei der Datensatzbe-
' schreibung zu Grundfile 1 und Grundfile 2 werden bei den Angaben ab der dritten
' Person bis zur neunten Person in einem Haushalt die Value-Labels nicht
' mehr explizit aufgeführt. Stattdessen werden Verweise angegeben, dass diese
' Variabeln wie die entsprechenden Variablen bei der zweiten Person in einem
' Haushalt behandelt werden sollen. Deswegen wird hier als keyModel immer der
' entsprechende Schlüssel (EF5U*) der Variablen der zweiten Person eingetragen und
' als keyCopy der Schlüssel der Variablen der weiteren Personen in einem Haushalt
' (EF6U* bis EF12U*) deren Value-Labels nicht mehr explizit aufgeführt sind. Bei
' anderen Grundfiles müsste diese Prozedur eventuell angepasst werden.
Private Sub initSimilarValues()
   Dim i, j, k As Integer
   Dim keyModel, keyCopy As String

   k = 1
   For i = 1 To 12
     keyModel = "EF5U" & i
     For j = 6 To 12
       keyCopy = "EF" & j & "U" & i
       similarValues(k, 1) = keyCopy
       similarValues(k, 2) = keyModel
       k = k + 1
     Next
   Next
End Sub

' Diese Funktion überprüft ob eine Variable einen Zusatz bekommen soll oder nicht,
' d.h. es wird nachgeschaut ob der Key einer Variable in dem Feld additions
' eingetragen ist. Falls ja wird dieser Zusatz zurückgegeben, ansonsten ein leerer
' String.
Private Function getVariableLabelAddition(ByVal key As String) As String
  Dim temp As String
  Dim i As Integer

  temp = ""
  For i = 1 To additionsDim
   If InStr(key, additions(i, 1)) <> 0 Then
      temp = additions(i, 2)
      Exit For
   End If
  Next
  getVariableLabelAddition = temp
End Function
```

```
' Diese Prozedur überprüft, ob eine Variable bezüglich der Value-Labels wie eine
' andere bereits existierende Variable behandelt werden soll, d.h. es wird in dem Feld
' similarValues nachgeschaut ob der Schlüssel der zu prüfenden Variablen eingetragen
' wurde. Fall ja, werden der untersuchten Variablen die entsprechenden Value-Labels
' hinzugefügt.
Private Sub checkSimilarValues(ByRef variableToCheck As VariableEntry)
  Dim i As Integer
  Dim v As VariableEntry

  For i = 1 To similarValuesDim
     ' falls die Variable mit diesem key wie eine andere behandelt werden soll
     If StrComp(variableToCheck.key, similarValues(i, 1)) = 0 Then
        Set v = vars.Item(similarValues(i, 2))
        variableToCheck.copyValueList v.valueList
     End If
   Next
End Sub

' Diese Prozedur erzeugt aus der Datensatzbeschreibung die Datenstruktur vars in der
' sämtliche Variablen mit ihren Feldbezeichnungen (keys), ihrem Code, ihren Labels
' und ihren Values gespeichert werden.
Private Sub getLabels(filename As String)
   Dim doc As Document
   Dim cellIndex, varIndex, page, feldbezIndex, pos As Integer
   Dim lastVariable, newVariable As VariableEntry
   Dim lastValue, newValue As valueEntry
   Dim rng, bezRng, codeRng, valueRng As Range

   Set doc = Application.Documents.Open(filename)
   Call initAdditions
   Call initSimilarValues
   varIndex = 0
   ' für alle Seiten der Datensatzbeschreibung
   For page = 1 To ActiveDocument.Tables.Count
      cellIndex = inhaltStart
      ' solange es noch nicht betrachtete Zellen gibt
      While (cellIndex < doc.Tables(page).Range.Cells.Count)
         Set rng = doc.Tables(page).Range.Cells(cellIndex).Range
         rng.MoveEnd Unit:=wdCharacter, Count:=-1
         ' wenn diese Zelle nicht leer ist
         If Not rng.Start = rng.End Then
            feldbezIndex = cellIndex - feldbezBeforeInhalt
            Set bezRng = doc.Tables(page).Range.Cells(feldbezIndex).Range
            bezRng.MoveEnd Unit:=wdCharacter, Count:=-1
            ' wenn es sich bei dieser Zelle um die Bezeichnung einer Variablen
            ' handelt
            If Not bezRng.Start = bezRng.End Then
               ' überprüfe ob die zuletzt eingefügte Variable eine richtige Variable war
               ' oder nur eine Überschrift für eine weitere Gruppe von Variablen
               If (InStr(bezRng, "U") <> 0) Then
```

```
                If (InStr(lastVariable.key, "U") = 0) Then
                  vars.Remove (vars.Count)
                  varIndex = varIndex - 1
                End If
              End If
              '  anlegen einer neuen Variablen
              If (InStr(rng, "leer") = 0) Then
                Set newVariable = New VariableEntry
                rng = rng & getVariableLabelAddition(bezRng)
                newVariable.init varIndex, rng, bezRng
                Set lastVariable = newVariable
                ' eintragen einer neuen Variablen in die collection vars
                vars.Add Item:=newVariable, key:=bezRng
                varIndex = varIndex + 1
                checkSimilarValues newVariable
              End If
            ' potentielle Kandidaten für Value-Bezeichnungen
            Else
              ' prüfen ob Kandidat einen neuen Value bezeichnet, d.h.
              ' ob ein "=" Zeichen in rng vorkommt
              pos = InStr(1, rng, "=", 1)
              ' wenn Kandidat neues Value-Label definiert
              If Not pos = 0 Then
                ' trennen von Code und Bezeichnung
                Set codeRng = doc.Range(rng.Start, rng.Start + pos - 1)
                Set valueRng = doc.Range(rng.Start + pos, rng.End)
                ' erzeuge neuen Value Eintrag für die letzte Variable
                Set newValue = New valueEntry
                newValue.init codeRng, valueRng
                Set lastValue = newValue
                lastVariable.addValueEntry newValue
              ' wenn Kandidat eine weitere Zeile der Value Beschreibung repräsentiert
              Else
                ' prüfe ob überhaupt schon ein Value für diese Variable
                ' eingetragen ist
                If lastVariable.valueList.Count > 0 Then
                  lastValue.expandLabel Trim(rng)
                ' wenn es sich um eine weitere Zeile des Variablen Namens handelt
                Else
                  lastVariable.expandLabel Trim(rng)
                End If
              End If
            End If
          End If
          cellIndex = cellIndex + nextCell
        Wend
      Next
      doc.Close
End Sub
```

```
' Diese Prozedur ist die einzige, die von außen aufgerufen werden kann. Sie bekommt
' als Parameter den Ort der Datensatzbeschreibung und den Namen der zu
' erzeugenden Syntax-Datei als String übergeben. Mit dem Aufruf von getLabels wird
' die Datenstruktur vars aus der Datensatzbeschreibung erzeugt. Danach erstellt die
' Prozedur eine neue SPSS-Sysntax-Datei.
Public Sub createSpssSetup(ByVal dbsFile As String, ByVal spsFile As String)
   Dim v As VariableEntry
   Dim l As valueEntry
   Dim str As String
   Dim newdoc As New Document

   Set vars = Nothing
   Call getLabels(dbsFile)
   ' aktiviere das neuerstellte Dokument
   newdoc.Activate
   ' Ausgabe des Kopfes
   str = "*Automatisch erstellt mit: EVSLabels 1.0" & vbLf _
      & "*erstellt aus Datei: " & dbsFile & vbLf & vbLf _
      & "*Zentrum fuer Umfragen, Methoden und Analysen (ZUMA)" & vbLf _
      & "*German Microdata Lab (GML)" & vbLf _
      & "*Postfach 12 21 55" & vbLf _
      & "*68072 Mannheim" & vbLf _
      & "*Tel: 0621 / 1246-278 Fax: 0621 / 1246-100" & vbLf _
      & "*e-mail: papastefanou@zuma-mannheim.de" & vbLf _
      & "*Georg Papastefanou, Thomas Grund" & vbLf & vbLf _
      & "* I. Variable Labels" & vbLf & vbLf
   Selection.TypeText Text:=str
   ' Ausgabe der Variablen-Labels
   Selection.TypeText Text:="variable labels" & vbLf
   For Each v In vars
      Selection.TypeText Text:=" / " & v.key & " '" & v.label & "'" & vbLf
   Next
   ' Ausgabe der Value-Labels
   Selection.TypeText Text:=vbLf & "* II. Value Labels" & vbLf & vbLf &
"value labels" & vbLf
   For Each v In vars
      If v.valueList.Count > 0 Then
         Selection.TypeText Text:="/ " & v.key & " "
         For Each l In v.valueList
            Selection.TypeText Text:=l.code & " '" & l.label & "'" & vbLf
         Next
      End If
   Next
   Selection.TypeText Text:="Execute." & vbLf
   ' Abändern der Variablennamen
   For Each v In vars
      Selection.TypeText Text:="rename variable " & v.key & " = v" & v.code &
"." & vbLf
   Next
   Selection.TypeText Text:="Execute."
```

```
   newdoc.SaveAs spsFile, wdFormatTextLineBreaks
   newdoc.Close
   Set vars = Nothing
End Sub
```

Klasse ValueEntry

```
' ************************************************************************
' *
' *                      EVSLabels
' *                     Version 1.0
' *                    vom 01.04.2004
' *
' *          Zentrum fuer Umfragen, Methoden und Analysen (ZUMA)
' *          German Microdata Lab (GML)
' *          Postfach 12 21 55
' *          68072 Mannheim
' *          Tel: 0621 / 1246-278 Fax: 0621 / 1246-100
' *          Autor: Thomas Grund
' *          e-mail: grun4101@uni-trier.de
' *
' *          Klasse: ValueEntry
' ************************************************************************

' Hier wird  der Code eines Value intern gespeichert.
Private codeIntern As String

' Hier wird  das Label eines Value intern gespeichert.
Private labelIntern As String

' Setzt den Code eines Values.
Property Let code(ByVal newCode As String)
   codeIntern = newCode
End Property

' Liefert den Code eines Values.
Property Get code() As String
   code = codeIntern
End Property

' Setzt das Label eines Values.
Property Let label(ByVal newLabel As String)
   labelIntern = newLabel
End Property

' Liefert das Label eines Values.
Property Get label() As String
   label = labelIntern
End Property
```

```
' Initialisiert einen Value mit einem Code und einem Label.
Function init(ByVal newCode As String, ByVal newLabel As String)
   code = newCode
   label = newLabel
End Function

' Diese Methode erweitert das Label eines Values um den gegebenen String. Sie wird
' aufgerufen, wenn sich ein Value Label über mehrere Zellen in der Datensatz-
' beschreibung erstreckt.
Sub expandLabel(ByVal expLabel As String)
   label = label & " " & expLabel
End Sub
```

Klasse VariableEntry

```
' ************************************************************************
' *
' *                      EVSLabels
' *                     Version 1.0
' *                    vom 01.04.2004
' *
' *          Zentrum fuer Umfragen, Methoden und Analysen (ZUMA)
' *          German Microdata Lab (GML)
' *          Postfach 12 21 55
' *          68072 Mannheim
' *          Tel: 0621 / 1246-278 Fax: 0621 / 1246-100
' *          Autor: Thomas Grund
' *          e-mail: grun4101@uni-trier.de
' *
' *          Klasse: VariablEntry
' *
' ************************************************************************

' Hier wird  der Code einer Variablen intern gespeichert.
Private codeIntern As String

' Hier wird  das Label einer Variablen intern gespeichert.
Private labelIntern As String

' Hier wird  die Liste mit allen zu dieser Variablen gehörenden Values gespeichert.
Private valueListIntern As New Collection

' Hier wird  die Feldbezeichnung (Key) einer Variablen intern gespeichert.
Private keyIntern As String

' Setzt den Code einer Variablen.
Property Let code(ByVal newCode As String)
   codeIntern = newCode
End Property
```

```
' Liefert den Code einer Variablen.
Property Get code() As String
   code = codeIntern
End Property

' Setzt die Feldbezeichnung (Key) einer Variablen.
Property Let key(ByVal newKey As String)
   keyIntern = newKey
End Property

' Liefert die Feldbezeichnung (Key) einer Variablen.
Property Get key() As String
   key = keyIntern
End Property

' Setzt das Label einer Variablen.
Property Let label(ByVal newLabel As String)
   labelIntern = newLabel
End Property

' Liefert das Label einer Variablen.
Property Get label() As String
   label = labelIntern
End Property

' Setzt die Liste mit allen Values einer Variablen.
Property Let valueList(ByVal newValueList As Collection)
   Set valueListIntern = newValueList
End Property

' Liefert die Liste mit allen Values einer Variablen.
Property Get valueList() As Collection
   Set valueList = valueListIntern
End Property

' Diese Methode kopiert die angegebene Liste mit allen Values in diese Variable.
Sub copyValueList(ByRef newValueList As Collection)
   For Each v In newValueList
      addValueLabel v.code, v.label
   Next
End Sub

' Initialisiert eine Variable mit einem Code, einem Label und einem Key.
Function init(ByVal newCode As String, ByVal newLabel As String, ByVal
newKey As String)
   code = newCode
   label = newLabel
   key = newKey
End Function
```

```
' Diese Methode erstellt einen neuen ValueEntry und fügt diesen der Liste der Values
' hinzu.
Sub addValueLabel(ByVal newCode As String, ByVal newLabel As String)
   Dim tempValue As New valueEntry
   tempValue.init newCode, newLabel
   valueListIntern.Add Item:=tempValue
End Sub

' Diese Methode fügt der Liste der Values einen weiteren Value hinzu.
Sub addValueEntry(ByVal newValueEntry As valueEntry)
   valueListIntern.Add Item:=newValueEntry
End Sub

' Diese Methode erweitert das Label einer Variablen um den gegebenen String. Sie
' wird aufgerufen, wenn sich ein Variablen-Label über mehrere Zellen in der
' Datensatzbeschreibung erstreckt.
Sub expandLabel(ByVal expLabel As String)
   label = label & " " & expLabel
End Sub
```

Form EVSLabelsForm

```
' ************************************************************************
' *
' *                      EVSLabels
' *                     Version 1.0
' *                    vom 01.04.2004
' *
' *          Zentrum fuer Umfragen, Methoden und Analysen (ZUMA)
' *          German Microdata Lab (GML)
' *          Postfach 12 21 55
' *          68072 Mannheim
' *          Tel: 0621 / 1246-278 Fax: 0621 / 1246-100
' *          Autor: Thomas Grund
' *          e-mail: grun4101@uni-trier.de
' *
' *          Form: EVSLabelsForm
' *
' ************************************************************************


' Die Prozedur wird ausgeführt beim Drücken des Knopfes „Erstelle SPSS-Setup“ Sie
' überprüft ob die gemachten Angaben zu Quelle und Ziel gültig sind und ruft dann
' createSpssSetup auf.
Private Sub createSpssButton_Click()
   Dim dbsFile, spsFile As String
```

```
   dbsFile = ""
   spsFile = ""
   ' liest die gemachten Angaben
   dbsFile = EVSLabelsForm.fileToOpen.Value
   spsFile = EVSLabelsForm.fileToSave.Value
   ' stellt sicher, dass sowohl für Quelle als auch für Ziel Werte
   ' eingetragen werden.
   If dbsFile = "" Or spsFile = "" Then
      MsgBox "Bitte geben Sie Quelle und Ziel an.", vbCritical
   Else
      ' Fehlerbehandlung falls mit der angegebenen Quelle keine
      ' Syntax Datei erstellt werden kann.
      On Error GoTo InvalidSource
      EVSLabels.createSpssSetup dbsFile, spsFile
   End If
   Exit Sub
InvalidSource:
   MsgBox "Die Quelle ist ungültig.", vbCritical
End Sub

' Diese Prozedur wird ausgeführt beim Drücken des Knopfes „Durchsuchen“ für die
' Auswahl der Datensatzbeschreibung als Quelle. Sie öffnet einen Datei-Öffnen-
' Dialog und weist den Namen der ausgewählten Datei dem Textfeld fileToOpen zu.
' Bei erneuter Kompilierung des Codes muss die Microsoft Excel 9.0 Object Library
' dazu geladen werden, falls Excel nicht installiert ist.
Private Sub openFileButton_Click()
   EVSLabelsForm.fileToOpen.Value =
Excel.Application.GetOpenFilename("Word-Dateien(*.doc),*.doc")
End Sub

' Diese Prozedur wird ausgeführt beim Drücken des Knopfes „Durchsuchen“ für die
' Auswahl des Ziels. Sie öffnet einen Speichern-Unter Dialog und weist den Namen
' der ausgewählten oder neu eingetragenen Datei dem Textfeld fileToSave zu. Bei
' erneuter Kompilierung des Codes muss die Microsoft Excel 9.0 Object Library dazu
' geladen werden, falls Excel nicht installiert ist.
Private Sub saveFileButton_Click()
   EVSLabelsForm.fileToSave.Value =
Excel.Application.GetSaveAsFilename("EVS03_SpssSetup.sps", "SPSS-Setup
Datei (*.sps),*.sps")
End Sub
```