Dipl.-Ing. H. Chris Gast

# Erweiterung von Word 2000 zur Erzeugung von Schriftzeichen höherer Unicode-Ebenen sowie Erstellung eines Makros für Word 2000 bis Word 2010 zur Analyse von Schrift- und Sonderzeichen mit Ausgabe des Schriftzeichen-Namens nach ISO/IEC 10646

Seite

# Vorwort

Mit dieser Arbeit wollte ich herausfinden, wie sich Schrift- und Sonderzeichen höherer Unicode-Ebenen nach ISO/IEC 10646 bzw. nach den Listen des Unicode-Consortium [2], wie antike Zahlzeichen, byzantinische Musiknotenzeichen oder Hieroglyphen, in meinem Computer mit Word 2000 und Windows XP erzeugen lassen.

Ursprünglich sollte diese Arbeit nur ein Kapitel für die Neuauflage meines Buches über die Griechische Schrift [15] werden. Da diese Arbeit aber zu umfangreich geworden ist und außerdem auch für andere Schriften wie kyrillische, hebräische, arabische bis hin zu chinesischen Schriftzeichen anwendbar ist, veröffentliche ich sie jetzt (Sommer 2010) als separaten Aufsatz.

Die beiden hier beschriebenen Word-Makros für die **Synthese von Schriftzeichen** aus einem eingegebenen Code (siehe Abschnitte 5.2 und 5.3) können sämtliche zulässigen Schrift- und Sonderzeichen erzeugen, aber auch Steuerzeichen. Die beiden Synthese-Makros erweitern die Funktionen von Word 2000, entsprechen ansonsten der Funktion in Word 2003 bis Word 2010. Schwierig bleibt jedoch weiterhin das Finden einer geeigneten Schriftart (Font). Darauf gehe ich in Abschnitt 4.1 ein.

Bei der Analyse von vorhandenen Schriftzeichen in einem Word-Text kann das in dieser Arbeit entwickelte **Analyse-Makro** (siehe Abschnitt 6.2) sämtliche normgerechten Schriftzeichen (Unicode) analysieren und (außer bei chinesischen, japanischen und koreanischen Silbenzeichen) deren englischen Namen nach ISO/IEC 10646 ausgeben, einschließlich von Schriftzeichen höherer Unicode-Ebenen. Daneben kann das Analyse-Makro aber auch die Codes von nicht-normgerechten Sonderschriftarten ermitteln sowie bei den Schriftarten "Symbol" und "Wingdings" die Bedeutungen und, soweit vorhanden, die entsprechenden Unicode-Schriftzeichen angeben.

Das Analyse-Makro bietet die Möglichkeit, über 100 Zeichen einer Zeichenkette gleichzeitig zu analysieren. Damit können auch Schriftzeichen mit mehreren diakritischen Zeichen gut analysiert werden, die sonst nur umständlich zu analysieren sind, oder etwa eine Kombination von Schriftzeichen aus Schriftsystemen, die von rechts nach links geschrieben werden, mit Schriftzeichen, die von links nach rechts geschrieben werden. Dies würde sich mit dem Befehl "Alt + C" in Word 2003 und höher nicht analysieren lassen (siehe zum Beispiel Bild 6.3). Das Analyse-Makro funktioniert auch dann, wenn im Text nur Platzhalter ("leere Kästchen") zu sehen sind.

Das Analyse-Makro erweitert nicht nur die vorhandenen Funktionen von Word 2000, sondern bietet auch Vorteile für **Word 2003 bis Word 2010!**

## Verbesserungen seit 2011

Inzwischen funktioniert das Analysemakro auch bei komplizierteren Feldfunktionen, Tabellenfeldern und bei Shapes fehlerfrei.
Es werden die vorhandenen Textmarken aufgelistet (seit 2012).
In Anhang B ergänzte ich weitere für Word 2000 nützliche Makros.
Ferner ergänzte ich eine **englische** Beschreibung des Analyse-Makros ("Abstract"), siehe Anhang C.

Das Analysemakro aus diesem Aufsatz biete ich in einer fertigen **Dokumentvorlage** an, bei der die Makros und die Liste der Namen aller Unicode-Zeichen sowie die Tastenkürzel bereits enthalten sind. Diese Dokumentvorlage enthält auch meine Makros aus [24].
Download: http://www.siebener-kurier.de/chris-aufsaetze/Zeichen-Analyse-kompakt.zip.

Dipl.-Ing. H. Chris Gast

# 1 Grundlagen der Codierung von Schriftzeichen im Computer

Die verbreiteteste Codierung für Schriftzeichen ist heutzutage neben den Normen der Reihe **ISO/IEC 8859** bzw. den davon abgeleiteten Windows-Code das Unicode-System (UCS) nach ISO/IEC 10646. Alle **anderen** Systeme, wie etwa spezielle Schriftfonts, sind heutzutage **veraltet**.

## 1.1 Ältere Zeichencodierungen

**ASCII** [1] ist ein inzwischen veralteter 7-Bit-Zeichensatz (128 Codes), die ersten 32 Zeichen sind (heute noch) Steuerzeichen. Das reichte für amerikanische Fernschreiber völlig aus.

Durch Einführung der Personal Computer (PC) wurde der ASCII-Zeichensatz um 1 Bit auf 8 Bit erweitert (8 Bit = 1 Byte = 256 Codes = zwei Hex-Zahlen). Während ASCII weltweit einheitlich benutzt wurde, gab es je nach Land oder Ländergruppe unterschiedliche Varianten mit 256 Codes (**DOS-Codepages** und später die Normen der Reihe **ISO/IEC 8859**). Auch bei den jetzigen Windows-Codes gibt es immer noch Unterschiede. Der bekannteste ist das in Westeuropa und in den USA verwendete **Windows 1252** (CP1252, "Westeuropäisch"). Letzteres wird fälschlich auch "ANSI"-Code genannt. Windows 1252 unterscheidet sich von **ISO/IEC 8859-1** durch zusätzliche Buchstaben im Bereich dez. 128 bis dez. 159 (hex.80 bis hex. 9F), wo die ISO/IEC-Norm zusätzliche unter Windows nicht verwendete Steuerzeichen hat.

## 1.2 Das Unicode-System

Die verbreitetste Codierung für die griechische Schrift ist heutzutage **Unicode** (UCS [2]). Diese ist in **ISO/IEC 10646** [3] und beim **Unicode-Consortium** [2] festgelegt. In Unicode sind heutzutage die Schriftzeichen oder Textelemente aller bekannten Schriftkulturen und Zeichensysteme in Form eines digitalen Codes enthalten, der hexadezimal oder dezimal angegeben werden kann (zum Beispiel beim Eingeben mit der Alt-Taste in Word). In Unicode wird nur der jeweilige Buchstabe (oder Sonderzeichen) festgelegt, nicht aber seine konkrete Gestalt auf dem Papier. Dafür sind die Schriftarten (Fonts) zuständig. Es ist allerdings nicht immer einfach, bei Sonderzeichen oder Zeichen antiker Schriften wie den Hieroglyphen eine passende kostenlose Schriftart zu finden, siehe Abschnitt 4. Die Bedeutung und Aussprache eines Buchstaben hängt von der jeweiligen Sprache ab. Das Unicode-System benennt Buchstaben deshalb möglichst Anwendungs-unabhängig.

Unicode ist untergliedert in Ebenen (*englisch "planes"*). Am wichtigsten ist die unterste Ebene, die **Ebene 0** mit dem grundlegenden mehrsprachigen Codebereich (**BMP**, englisch: "Basic Multilingual Plane"). In dieser Ebene liegen die Zeichen mit den Unicodes von hexadezimal 0000 bis FFFF bzw. dezimal von 0 bis 65535 [4]. Dies reicht für alle heutigen europäischen Sprachen aus, aber auch für Altgriechisch und Altkirchenslawisch .

Darauf folgt die **Ebene 1** mit dem ergänzenden mehrsprachigen Bereich**,** (**SMP**, englisch: "**Supplementary Multilingual Plane",** SMP, auch als *Plane 1* bezeichnet). Sie enthält vor allem historische Schriftsysteme bis hin zu Hieroglyphen und weitere chinesisch/ japanisch/koreanische Zeichen, aber auch größere Ansammlungen an Zeichen, die selten in

---

1 ASCII = American Standard Code for Information Interchange.

2 englisch: Universal Character Set.

3 Die Fassung von 2011 ist nur noch in Englisch. In der vorhergehenden Ausgabe ISO/IEC 10646 von 2003 (einschließlich Änderung 2von 2006) gab es die Benennungen der Zeichen in Englisch und in Französisch, seitdem gibt es diese nur noch in Englisch.

4 0 bis 65 535 ergeben (inklusive der Null) zwei hoch sechzehn theoretisch mögliche Zeichen.

Gebrauch sind, zum Beispiel Musiknotenzeichen. Weitere Ebenen sind für Europäische Sprachen noch nicht relevant.

Die Unicode-Ebenen werden wiederum in **Blöcke** (englisch "**blocks**") unterteilt, die jeweils einzelne Schriftsysteme wie Lateinisch, Griechisch usw. enthalten. Die ersten beiden Blöcke von hexadezimal 0000 bis 00FF (dezimal 0 bis 255) enthalten lateinische Schriftzeichen und entsprechen der ISO/IEC 8859-1. Ist ein Block voll, müssen spätere Ergänzungen in anderen Blocks untergebracht werden, da kein festgelegtes Unicode-Zeichen wieder gestrichen werden darf. Dies führt teilweise zu einer gewissen Zerstückelung des Systems.

Unicode kann auf verschiedene Arten codiert werden. Üblich sind:

a) UTF-16 bei Computern mit Windows-Betriebssystem. Dieses ist in der für europäischen Sprachen relevanten Unicode-Ebene 0 mit dem einfachen Unicode identisch. Bei Schriftzeichen mit Codes größer hex. FFFF wird der Code in sogenannte Surrogate nach ISO/IEC 10646:2003, Anhang C zerlegt [5]. Siehe hierzu Abschnitte 2.3 und 5.1 ff.
b) UTF-8 bei der (seriellen) Übertragung im Internet. Die Berechnung von UTF-8 aus den Unicode-Werten ist in ISO/IEC 10646:2003, Anhang D, genormt[6]. Hier erfolgt die Umcodierung derart, dass erstens die Codes bei den lateinischen Grundbuchstaben (ASCII-Bereich) kleiner werden als bei der einfachen Unicode-Codierung und dass zweitens bei der Übertragung weniger Fehler entstehen als bei der einfachen Unicode-Codierung. Auf UTF-8 gehe ich in diesem Aufsatz nicht weiter ein.

Die folgende Tabelle 1.1 gibt eine kurze Übersicht über die Unicode-Ebenen 0 (BMP) und 1 (SMP) mit einer Auswahl an Blöcken. Neben den wichtigsten Blöcken mit lateinischen Buchstaben, den Surrogaten und der Private-Use-Area (PUA) enthält sie die Unicode-Blöcke, die in den folgenden Abschnitten erwähnt werden, siehe vor allem Abschnitt 4.2.

[5] Inzwischen Version ISO/IEC 10646:2012, Anhang C.
[6] Inzwischen Version ISO/IEC 10646:2012, Anhang D.

## Tabelle 1.1: Übersicht der Unicode-Ebenen 0 und 1 mit einer Auswahl an Blöcken

| | Unicode-Blöcke | | | Unicodes | |
|---|---|---|---|---|---|
| **Ebene** | **Deutsche Bezeichnung**[7] | **Englische Bezeichnung** nach ISO/IEC 10646 | **Französische Bezeichnung** nach ISO/IEC 10646 | (hexadezimal) | (dezimal) |
| **0** | **Unicode-Ebene 0** | **Basic Multilingual Plane (BMP)** | **Plan multilingue de base (PMB)** | **00 00** bis **FF FF** | **0** bis **65 535** |
| | Basis-Lateinisch | Basic Latin (ASCII) | Latin de base | 0000 bis 007F | 0 bis 0127 |
| | Lateinisch-1, Ergänzung | Latin-1 supplement | Supplément latin-1 | 0080 bis 00FF | 0128 bis 0255 |
| | | | | | |
| | Kombinierende diakritische Zeichen | Combining Diacritical Marks | Diacritiques | 03 00 bis 03 6F | 768 bis 879 |
| | Griechisch und Koptisch | Greek and Coptic | Grec et copte | 03 70 bis 03 FF | 880 bis 1023 |
| | | | | | |
| | Griechisch, Erweiterung[8] | Greek Extended | Grec étendu | 1F 00 bis 1F FF | 7 936 bis 8 191 |
| | | | | | |
| | Surrogat 1 (high surrogates) | High Surrogates | Demi-zone haute | D8 00 bis DB FF | 55 296 bis 56 319 |
| | Surrogat 2 (low surrogates) | Low Surrogates | Demi-zone basse | DC 00 bis DF FF | 56 320 bis 57 343 |
| | Private Use Area; PUA | Private Use Area (**PUA**) | Zone à usage privé | E000 bis F8FF | 57344 bis 63743 |
| **1** | **Unicode-Ebene 1:** SMP (Supplementary Multilingual Plane) | Supplementary Multilingual Plane (SMP) | Plan multilingue complémentaire (PMC) | **1 00 00** bis **1 FF FF** | **65 536** bis **131 071** |
| | | | | | |
| | Ägäische Zahlzeichen | Aegean Numbers | Nombres égéens | 1 01 00 bis 1 01 3F | 65 792 bis 65 855 |
| | Altgriechische Notenschriftzeichen | Ancient Greek Numbers | Nombres grecs anciens | 1 01 40 bis 1 01 8A | 65 856 bis 65 930 |
| | | | | | |
| | Ägyptische Hieroglyphen | Egyptian Hieroglyphs | (Hiéroglyphes égyptiens)[9] | 1 30 00 bis 1 34 2F | 77 824 bis 78 895 |
| | | | | | |
| | Byzantinische Notenschriftzeichen | Byzantine Musical Symbols | Symboles musicaux byzantins | 1 D0 00 bis 1 D0 F5 | 118 784 bis 119 029 |
| | | | | | |
| | Altgriechische Zahlzeichen | Ancient Greek Musical Notation | Musique grecque ancienne | 1D2 00 bis 1 D2 45 | 119 296 bis 119 365 |

[7] Weitgehend nach Wikipedia [2]
[8] In Wikipedia heißt es stattdessen "Griechisch, Zusatz" [3].
[9] ISO IEC 10646/Amd5:2008, wo die Hieroglyphen genormt sind, existiert leider nur auf Englisch.

# 2 Anpassen von Windows XP an Schriftzeichen höherer Unicode-Ebenen

## 2.1 Problem mit Windows XP (Stand Juni 2010)

Bei Schriftzeichen (Characters) mit Codes (Codepoints) über hex. FFFF wie im folgendem Beispiel für den Violinschlüssel gibt es bei standardmäßig[10] eingestelltem **Windows XP** doppelte Kästchen, selbst wenn man die passende Schriftart (siehe Abschnitt 4) gefunden hat:

| | | | |
|---|---|---|---|
| U+1D11E (119070) | □□ | Notenschriftzeichen Violinschlüssel | MUSICAL SYMBOL G CLEF |

Es kann auch passieren, dass man Schriftzeichen der Ebene Null ( "Basic Multilingual Plan") statt der erwünschten Schriftzeichen der Ebene 1 angezeigt bekommt.

| | | | |
|---|---|---|---|
| U+10141 (65857) | **Ł** | Griechische akrophonische attische Hälfte | GREEK ACROPHONIC ATTIC ONE HALF |

Hier ist offensichtlich ein Fehler, da man in Griechenland kein polnisches durchgestrichenes "L" verwendet.

**Ursache:** Windows kann nur Schriftzeichen mit Codes bis hexadezimal FFFF bzw. bis dezimal 65 535 darstellen ("Unicode-Ebene 0"). Das sind vermutlich mehr Schriftzeichen, als sich die Konstrukteure der Computer seinerzeit überhaupt vorstellen konnten. Doch allein das Chinesische braucht inzwischen mehr Schriftzeichen. Windows zerlegt deshalb jedes Zeichen mit Codes größer hex. FFFF in zwei "**Surrogate".** Diese sind in ISO/IEC 10646:2003, Anhang C in der Codierung "UTF-16 festgelegt [11].

## 2.2 Lösung

Um nun statt der zwei Kästchen das gewünschten Schriftzeichen zu erhalten, muss man in der Registry von Windows XP einige Einträge vornehmen, damit Windows die Surrogaten als solche erkennt und aus je zwei Surrogaten wieder ein Schriftzeichen macht. Im Internet fand ich zwei Lösungswege:

### 1. Lösung [12]

**"Warum werden bei Unicode-Charakteren über U+10 000 trotz installierter Schriften zwei Rechtecke angezeigt?**
In diesem Falle müssen Surrogate aktiviert werden, die nur von Windows 2000, XP und Vista unterstützt werden:

1. Über Start, Ausführen, und Eintippen von "regedit" den Registrierungseditor starten.
2. Sich durchhangeln zu HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\LanguagePack
3. In die rechte Fensterhälfte klicken und über Neu → DWORD-Wert einen neuen Eintrag anlegen und SURROGATE benennen (Groß/Kleinschreibung beachten!).
4. Dem neuen Eintrag den Wert 2 zuweisen.

Nach einem anschließenden Neustart erscheinen nun auch Unicode-Charactere über U+10000 korrekt auf dem Bildschirm."

---

[10] **Stand Juni 2010**. Dies kann sich inzwischen mit einem der vielen Windows-Updates geändert haben.

[11] Inzwischen Version ISO/IEC 10646:2012, Anhang C.

[12] Quelle: http://de.wikipedia.org/wiki/Wikipedia:UTF-8-Probleme#Warum_werden_bei_Unicode-Charakteren_.C3.BCber_U.2B10000_trotz_installierter_Schriften_zwei_Rechtecke_angezeigt.3F, Download 13. März 2010. Danke hiermit an A. Gumtow, der mir diese Lösung aus dem Internet suchte.

**2. Lösung**

Man kann eine Reg-Datei erstellen, die den obigen Lösungsweg ausführt, ohne dass man als Laie die Registry anrühren muss.[13] Diese Datei schreibt man am besten als ".txt"-Datei und benennt sie dann mit dem Windows-Explorer in "xxx.reg" um, mit einem beliebigen Namen vor der Endung ".reg":

**Erweiterung.reg**

```
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\LanguagePack]
"SURROGATE"=dword:000002
```

Dieser Registry-Eintrag findet sich auf der Internetseite der **Microsoft Corporation** [14] in hexadezimaler Schreibweise wie folgt:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\LanguagePack]
SURROGATE=(REG_DWORD)0x00000002
```

Da ich meinen Computer schon umgestellt hatte, konnte ich diese Variante nicht mehr testen.

Damit bekam ich endlich Schriftzeichen der Unicode-Ebene 1 angezeigt:

| | | | |
|---|---|---|---|
| U+1D11E (119070) | 𝄞 | Notenschriftzeichen Violinschlüssel | MUSICAL SYMBOL G CLEF |

bzw. "nur noch" einfache Kästchen, wenn lediglich die passende Schriftart fehlt:

| | | | |
|---|---|---|---|
| U+1D11E (119070) | □ | Notenschriftzeichen Violinschlüssel | MUSICAL SYMBOL G CLEF |

## 2.3 Zum Prinzip der Surrogate

Die Hohen und Niedrigen Surrogate sind in speziell dafür reservierten Unicode-Blöcken untergebracht, siehe folgende Tabelle 2.1. Zusätzlich sind in dieser Tabelle die Bereiche der Ebenen 0 und 1 aufgeführt. Eine gute Übersicht über weitere Blöcke und weitere Ebenen gibt es in Wikipedia [3].

**Tabelle 2.1: Die Unicode-Blöcke der Surrogate**

| Ebene | **Unicode Blöcke** | **Unicode** (hexadezimal) | **Unicode** (dezimal) |
|---|---|---|---|
| 0 | Ebene 0: **BMP** (Basic Multilingual Plane) | 00 00 bis FF FF | 0 bis 65535 |
| | ... | | |
| | Surrogat 1 (High Surrogates ) | D8 00 bis DB FF | 55 296 bis 56 319 |
| | Surrogat 2 (Low Surrogates ) | DC 00 bis DF FF | 56 320 bis 57 343 |
| | ... | | |
| 1 | Ebene 1: SMP (Supplementary Multilingual Plane) | 1 00 00 bis 1 FF FF | 65 536 bis 131 071 |

[13] Dieses Programm gibt es ungefähr sechzehn Mal in Google [13] (Stand: 23. März 2010). Welches davon die ursprüngliche Quelle ist, weiß ich nicht. Ich selbst bekam sie von A. Gumtow.

# 3 Probleme bei Word 2000 mit Zeichen höherer Unicode-Ebenen

Word 2000 kann Schriftzeichen der Unicode-Ebene 1 darstellen, wenn Windows entsprechend eingestellt ist (siehe Abschnitt 2) und die entsprechende Schriftart installiert ist (siehe Abschnitt 4.1).

Allerdings konnte ich in Word 2000 weder mittels Tastaturbefehlen Alt + Zahlencode noch mit der Maus über "Einfügen → Symbol" Schriftzeichen mit Unicodewerten über hex. FFFF einfügen. Auch mit der Zeichentabelle von Windows ist dies nicht möglich. Schließlich gelang mir das Einfügen von Schriftzeichen der Unicode-Ebene 1 in **Open-Office**. Dort können problemlos mit der Maus über "Einfügen → Sonderzeichen" auch Zeichen der Unicode-Ebene 1 eingefügt werden. Wenn man will, kann man dann die Datei wieder als Word-Datei speichern.

Ich habe für Word inzwischen mein Makro zur Zeichenerzeugung passend erweitert, so dass nun damit auch Schriftzeichen der höheren Unicode-Schriftarten eingefügt werden können, siehe Abschnitt 5.

Es gelang mir auch nicht, aus **Mozilla Firefox** [14] direkt über die Zwischenablage Zeichen der Unicode-Ebene 1 aus einer Webseite von Wikipedia [3] in eine Worddatei zu übertragen. Ich musste die html-Datei als Textdatei (.txt) abspeichern, und dann aus dieser Textdatei über die Zwischenablage die Schriftzeichen der Unicode-Ebene 1 in ein Word-Dokument übertragen. Dies hat damit zu tun, dass Wikipedia (wie im Internet üblich) die Codierung UTF 8 verwendet, Windows aber die Codierung UTF-16. Und diese beiden Codierungen machen im Bereich der höheren Unicode-Ebenen wegen der Surrogate in UTF-16 häufig Probleme beim Umcodieren.

**Achtung:** Der Befehlt "Bearbeiten → Rückgängig" funktioniert bei Zeichen der Ebene 1 in Word 2000 fehlerhaft!

Auch die Zwischenablage ist bei diesen Zeichen höherer Unicode-Ebenen tückisch. Es sind Computer-intern wegen der Surrogate immer zwei Zeichen, die kopiert werden, auch wenn das eine davon unsichtbar ist. Es fühlt sich ein bisschen so an, als ob man mit einem alten Leiterwagen Klaviere transportieren will... ☺ .

---

[14] Um den Internet-Explorer für Zeichen der Unicode-Ebene 1 zu nutzen, ist ein weiterer Eingriff in die Registry nötig [14]:
[HKEY_CURRENT_USER\Software\Microsoft\Internet Explorer\International\Scripts\42]
IEFixedFontName=[Surrogate Font Face Name]
IEPropFontName=[Surrogate Font Face Name].
Dieses probierte ich aber nicht aus, da man in der Unicode-Ebene 1 keinesfalls mit nur einer Schriftart auskommt, siehe Tabelle 4.1. Außerdem gilt für mich die Devise: "*Never touch a running system*", wenn es nicht unbedingt nötig ist.

# 4 Das Problem der Schriftartensuche (Fonts) bei höheren Unicode-Ebenen

## 4.1 Schriftartensuche

Die Unicode-Ebene 0 (Basic Multilingual Plane) enthält im Prinzip alle in europäischen Sprachen nötigen Schriftzeichen. Inzwischen hat man jedoch weitere Schriftzeichen genormt, die in der ursprünglichen Ebene 0 nicht mehr unterzubringen waren, wie etwa zusätzliche antike Zahl- und Notenschriftzeichen[15]. In ISO/IEC 10646 oder Unicode.org [2]), bzw. einfacher bei Wikipedia [3], findet man alle genormten Schriftzeichen-Codes, aber leider nicht die dafür jeweils nötigen Schriftarten.

Während bei der Unicode-Ebene 0 die auf dem Computer vorhandene Schriftarten meistens ausreichen, sind für die höhere Unicode-Ebenen grundsätzlich spezielle Schriftarten nötig, da keine für Windows geeignete Schriftart mehr als 65 536 Zeichen enthalten kann. Mit dem Schriften-Suchprogramm **FileFormat.Info** [4] fand ich die Schriften **Cardo.ttf** [5] und **Code 2001.ttf** [7]. Weitere Schriften für den griechischen Sprachraum finden sich auf der Webseite "**Unicode Fonts for Ancient Scripts"** [8]. Dort fand ich neben der Schriftart "Aegean.otf" [10] auch die die für byzantinischen und modernen Notenzeichen geeignete Schriftart "**Musica.otf**" [9], siehe Tabelle 4.1. Zum Vergleich testete ich auch den Unicode-Block "Notenschriftzeichen". Für Hieroglyphen fand ich die Schriftart "Gardiner" [11] geeignet.

Die Schriftarten Cardo und Aegean sind für die Wiedergabe griechischer Schriftzeichen am vielseitigsten geeignet.

Auch die Schriftart "Musica ist recht vielseitig einsetzbar. Für byzantinische Musiknoten-zeichen ist sie als einzige brauchbar.

Nachdem ich die Schriftart Musica.otf gefunden hatte, verzichtete ich auf die Suche nach eventuell existierenden weiteren geeigneten Schriftarten. Mit etwas Geschick kann man weitere Schriftarten auch direkt mittels Google oder anderen Suchmaschinen finden.

Die folgende Tabelle 4.1 gibt eine Übersicht über die geeigneten Schriftarten für die gewählten Unicode-Blöcke. Zur Übersicht ergänzte ich auch die Ergebnisse für den Unicode-Block "Griechisch und Koptisch" und den Unicode-Block **"Greek Extended"** mit den polytonischen Akzenten von der Unicode-Ebene 0.

Die Schriftarten Arial-Unicode und Code 2000 habe ich nur kurz für die Erstellung dieser Tabelle 4.1 installiert und anschließend gleich wieder entfernt (und hatte sie nicht mehr zur Verfügung, als ich Hieroglyphen testete).

---

[15] Die heute noch gebräuchlichen griechischen Zahlzeichen nach dem milesischen Prinzip sind im Unicode-Block "Griechisch und Koptisch" in der Unicode-Ebene 0 enthalten, siehe [15].

**Tabelle 4.1: Unicode-Blöcke und Schriftarten**

| Schrift | Unicode-Block Ebene 0 | | Unicode-Block Ebene 1 | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Griechisch + Koptisch | Griechisch - Erweiterung | Ägäische Zahl-zeichen | Altgriech. Zahl-zeichen | Altgriech. Noten-schrift-zeichen | Byzanti-nische Noten-schrift-zeichen | Noten-schrift-zeichen | Hiero-glyphen |
| **DejaVu Sans** | **100 %** | 100 % | 0 | 0 | 0 | 0 | 0 | 0 |
| **DejaVu Serif** | **87 %** (115/134) (brauchbar) | **100 %** | 0 | 0 | 0 | 0 | 0 | 0 |
| **Palatino Linotype** | 63 % (86/135) (brauchbar) | **100 %** | 0 | 0 | 0 | 0 | 0 | 0 |
| Arial Unicode | 80 % | 100 % | 0 | 0 | 0 | 0 | 0 | – |
| Microsoft Sans Serif | 84 % | 100 % | 0 | 0 | 0 | 0 | 0 | 0 |
| **Aegean** | 100 % | 100 % | **100 %** | **100 %** | 0 | 0 | 0 | 0 |
| **Cardo** | **93 %** (brauchbar) | 100 % | 2 % (1/57) | **100 %** | **100 %** | 0 | 0 | 0 |
| Code 2000 | 100 % | 100 % | 0 | 0 | 0 | 0 | 0 | – |
| Code 2001 | <50 % (unbrauch-bar) | < 50 % (unbrauch-bar) | **100 %** | 0 | 0 | 13 % | 20 % (43/219) | 0 |
| **Musica** | 0 | 0 | 0 | 0 | **100 %** | **100 %** | **100 %** | 0 |
| **Gardiner** | 0 | 0 | | 0 | 0 | 0 | 0 | **100 %** |

## 4.2 Beispiele für Zeichen der Unicode-Ebene 1 wie antike Zahlzeichen, Notenschriftzeichen und Hieroglyphen

In den folgenden Tabellen stellte ich Beispiel dar für Schriftzeichen der Unicode-Ebene 1. Die UTF-8-Codes entnahm ich den Tabellenwerken von T. Schild [24], die UTF-16-Codes ermittelte ich selber, siehe Abschnitt 6.2, die deutschen Bezeichnungen entnahm ich weitgehend Wikipedia, die englischen Benennungen sind in ISO 10646:2012 und die französischen Benennungen entnahm ich den inzwischen zurückgezogenen Versionen ISO 10646:2003 und ISO/IEC 10646/AMD 1:2005 sowie [2].

Als geeignete Schriftarten für die Ägäischen Zahlzeichen fand ich **Code2001.ttf** und **Aegean.otf**, siehe obige Tabelle 4.1 und die folgende Tabelle 4.2:

**Tabelle 4.2: Beispiele für Ägäische Zahlzeichen (hex. 101 00 ff)**[16]

| Unicode | Zeichen | UTF-8 | UTF-16 | Beschreibung | Englische Bezeichnung | Französische Bezeichnung |
|---|---|---|---|---|---|---|
| U+10100 (65792) | 𐄀 | F0 90 84 80 | D800 DD00 | Ägäischer Worttrennstrich | AEGEAN WORD SEPARATOR LINE | TRAIT ÉGÉEN SÉPARATEUR DE MOTS |
| U+10101 (65793) | 𐄁 | F0 90 84 81 | D800 DD01 | Ägäischer Worttrennpunkt | AEGEAN WORD SEPARATOR DOT | POINT ÉGÉEN SÉPARATEUR DE MOTS |
| U+10102 (65794) | 𐄂 | F0 90 84 82 | D800 DD02 | Ägäisches X-Zeichen | AEGEAN CHECK MARK | COCHE ÉGÉENNE |

Als geeignete Schriftarten für Altgriechische Zahlzeichen fand ich **Cardo.ttf** und **Aegean.otf,** siehe Tabelle 4.1 und die folgende Tabelle 4.3:

**Tabelle 4.3: Beispiele für Altgriechische Zahlzeichen (hex. 101 40 ff)**[17]

| Unicode | Zeichen | UTF-8 | UTF-16 | Beschreibung | Englische Bezeichnung | Französische Bezeichnung |
|---|---|---|---|---|---|---|
| U+10141 (65857) | 𐅁 | F0 90 85 81 | D800 DD41 | Griechische akrophonische attische Hälfte | GREEK ACROPHONIC ATTIC ONE HALF | UNE MOITIÉ ACROPHONIQUE GREC ATTIQUE |
| U+10142 (65858) | 𐅂 | F0 90 85 82 | D800 DD42 | Griechische akrophonische attische Drachme | GREEK ACROPHONIC ATTIC ONE DRACHMA | UN DRACHME ACROPHONIQUE GREC ATTIQUE |

Als geeignete Schriftarten für Altgriechische Notenschriftzeichen fand ich **Cardo.ttf** und **Musica.otf**, siehe Tabelle 4.1 und die folgende Tabelle 4.4:

**Tabelle 4.4: Beispiele für Altgriechische Notenschriftzeichen (hex. 1D2 00ff)** [18]

| Unicode | Zeichen | UTF-8 | UTF-16 | Beschreibung | Englische Bezeichnung | Französische Bezeichnung |
|---|---|---|---|---|---|---|
| U+1D200 (119296) | 𝈀 | F0 9D 88 80 | D834 DE00 | Griechisches Vokalmusikzeichen 1 (E) | GREEK VOCAL NOTATION SYMBOL-1 | SYMBOLE NOTATION GRECQUE VOCALE 1 |
| U+1D201 (119297) | 𝈁 | F0 9D 88 81 | D834 DE01 | Griechisches Vokalmusikzeichen 2 (tieferes Eis) | GREEK VOCAL NOTATION SYMBOL-2 | SYMBOLE NOTATION GRECQUE VOCALE 2 |
| U+1D202 (119298) | 𝈂 | F0 9D 88 82 | D834 DE02 | Griechisches Vokalmusikzeichen 3 (höheres Eis) | GREEK VOCAL NOTATION SYMBOL-3 | SYMBOLE NOTATION GRECQUE VOCALE 3 |

Die Byzantinischen Notenschriftzeichen sind vollständig darstellbar mit dem Font (Schriftart) "**Muscia.otf**", siehe Tabelle 4.5:

[16] aus Wikipedia, http://de.wikipedia.org/wiki/Unicode-Block_Ägäische_Zahlzeichen [3].
[17] aus Wikipedia, http://de.wikipedia.org/wiki/Unicode-Block_Altgriechische_Zahlzeichen, [3].
[18] aus Wikipedia: http://de.wikipedia.org/wiki/Unicode-Block_Altgriechische_Notenschriftzeichen, [3].

**Tabelle 4.5: Beispiele für Byzantinische Notenschriftzeichen (hex. 1D0 00ff)**[19]

| Unicode | Zeichen | UTF-8 | UTF-16 | Beschreibung | Englische Bezeichnung | Französische Bezeichnung |
|---|---|---|---|---|---|---|
| U+1D000 (118784) | 𝀀 | F0 9D 80 80 | D834 DC00 | Byzantinisches Notenschriftzeichen Psili | BYZANTINE MUSICAL SYMBOL PSILI | SYMBOLE MUSICAL BYZANTIN PSILI |
| U+1D001 (118785) | 𝀁 | F0 9D 80 81 | D834 DC01 | Byzantinisches Notenschriftzeichen Daseia | BYZANTINE MUSICAL SYMBOL DASEIA | SYMBOLE MUSICAL BYZANTIN DASEIA |
| U+1D002 (118786) | 𝀂 | F0 9D 80 82 | D834 DC02 | Byzantinisches Notenschriftzeichen Perispomeni | BYZANTINE MUSICAL SYMBOL PERISPOMENI | SYMBOLE MUSICAL BYZANTIN PERISPOMENI |

Nicht zum Thema "Griechische Schrift" gehören die Notenschriftzeichen der Westlichen Musik. Diese sind ebenfalls Zeichen der Unicode-Ebene 1. Eine dafür geeignete Schriftart ist "**Musica.otf",** siehe Tabelle 4.6:

**Tabelle 4.6: Beispiele für Notenschriftzeichen (hex. 1D1 00 ff)**[20]

| Unicode. | Zeichen | UTF-8 | UTF-16 | Beschreibung | Englische Bezeichnung | Französische Bezeichnung |
|---|---|---|---|---|---|---|
| U+1D100 (119040) | 𝄀 | F0 9D 84 80 | D834 DD00 | Notenschriftzeichen Taktstrich | MUSICAL SYMBOL SINGLE BARLINE | SYMBOLE MUSICAL BARRE DE MESURE SIMPLE |
| | | | | | | |
| U+1D11E (119070) | 𝄞 | F0 9D 84 9D | D834 DD1E | Notenschriftzeichen Violinschlüssel | MUSICAL SYMBOL G CLEF | SYMBOLE MUSICAL CLÉ DE SOL |
| U+1D121 (119073) | 𝄡 | F0 9D 84 A1 | D834 DD21 | Notenschriftzeichen Bratschenschlüssel; C-Schlüssel; | MUSICAL SYMBOL C CLEF | SYMBOLE MUSICAL CLÉ D'UT |
| U+1D122 (119074) | 𝄢 | F0 9D 84 A2 | D834 DD22 | Notenschriftzeichen Bassschlüssel | MUSICAL SYMBOL F CLEF | SYMBOLE MUSICAL CLÉ DE FA |

Ein weiteres Beispiel für Zeichen der Unicode-Ebene 1 sind die Hieroglyphen. Ich nahm hierfür die Schriftart "Gardiner.otf", siehe Tabelle 4.7.

**Tabelle 4.7: Beispiele für Hieroglyphen (hex. 130 00 ff)**[21]**:**

| Hexcode | Zeichen | Surrogate (UTF-16) | Erklärung |
|---|---|---|---|
| U+13000 (77824) | 𓀀 | D80C und DC00 | Ägyptische Hieroglyphe A1 (Gardiner A1) |
| U+13050 (77904) | 𓁐 | D80C und DC50 | Ägyptische Hieroglyphe B1 (Gardiner B1) |

[19] aus Wikipedia, http://de.wikipedia.org/wiki/Unicode-Block_Byzantinische_Notenschriftzeichen [3].
[20] aus Wikipedia, http://de.wikipedia.org/wiki/Unicode-Block_Notenschriftzeichen [3].
[21] Zeichencode der Hieroglyphen siehe ISO/IEC 10646 oder [2].

# 5 Makro zum Erzeugen von Schriftzeichen aus dem hexadezimalen Unicode, auch für höhere Unicode-Ebenen (UCS > $FFFF_{hex}$)

## 5.1 Aufgabenstellung

Ich wollte ein Makro schreiben für Word 2000 zur Erzeugung von Schriftzeichen aus dem Unicode. Dabei kann der Unicode in dezimaler oder in hexadezimaler Schreibweise vorliegen, was im Endeffekt zu zwei sehr ähnlichen Makros führte. Diese Makros sollen im Gegensatz zu den Funktionen von Word 2000 ausnahmslos alle Schriftzeichen erzeugen können, für die auf dem Computer eine passende Schriftart (Font) vorhanden ist, auch für Schriftzeichen mit Unicodes größer als hex. FFFF, was für mein bisheriges Makro von 2009 [22] die obere Grenze war.[22]

Hierbei ist zu berücksichtigen, dass Windows-Computer nicht Unicode direkt verwenden, sondern die (für 16-Bit-Rechner günstigere) Codierung UTF-16. Für Unicode-Werte bis hex. FFFF (das sind 16 Bit) sind Unicode und UTF-16 identisch. Für Werte darüber weicht **UTF-16** von Unicode ab. Der Unicode-Wert wird in zwei 16-Bit-Codewerte, sogenannte "Surrogate", zerlegt. Dieses ist in **ISO/IEC 10646:2003, Anhang C**, festgelegt. Nach ISO/IEC 10646:2003, Anhang C, Abschnitt C.3 gilt für die Umrechnung von UCS in UTF-16 (siehe folgende Tabelle):

**Tabelle 5.1: Codierung nach ISO/IEC 10646:2003, Anhang C** [23]

| **hexadezimaler Unicode (UCS-4 (4-octet))** | **Codierung nach UTF-16 (UTF-16, 2-octet elements)** |
|---|---|
| x = 0000 0000 bis 0000 FFFF | x (bis hex. FFFF) |
| x = 0001 0000 bis 0010 FFFF | **Surrogate y und z**;<br>mit y = ((x – 0001 0000) \ 400) + D800<br>und z = (Mod (x - 0001 0000)/400) + DC00 |
| x = 0011 0000 bis 7FFF FFFF | keine UTF-16-Werte definiert |

Hierbei bedeutet "\" eine sogenannte "Ganzzahlige Division", nämlich den Integerwert der Division, und "Mod()" der "Ganzzahlige Rest einer Division", eine Modulo-Funktion.

**1. Problem:** Einige Rechenoperationen funktionieren bei mir nur mit Dezimalzeichen, nicht mit Hexadezimalzeichen. Deshalb rechnete ich einige wichtige Zahlenwert in Dezimalwerte um, siehe Tabelle 5.2.

**2. Problem**: Beim Umwandeln von Hexzahlen in Dezimalzahlen gibt die Val-Funktion "Integer"-Werte aus, die aber bei VBA nur bis (dezimal) "+32 767" erlaubt sind, danach werden die "Ergebnisse" negativ. Aus hex. FFFF wird "-1"! Um höhere Zahlen als (dezimal) 32 767 zu ermöglichen, muss "&" angehängt werden. Dies erzwingt eine Ausgabe der Val-Funktion als Zeichen vom Typ "Long" anstelle vom Typ "Integer". Mit dem Typ "Long" sind laut Hilfefunktion dezimale Zahlen bis 2 147 483 647 möglich, ohne das es Vorzeichen-probleme gibt, was für Unicode völlig ausreicht.

---

[22] Nachtrag: Inzwischen (21. Oktober 2010) kann Word 2003 auch Zeichen höherer Unicode-Ebenen erzeugen und analysieren, was aber beim Schreiben dieses Kapitels im Mai/Juni 2010 noch nicht der Fall war.

[23] In dieser ISO-Tabelle bedeutet "x" der hexadezimale Code, und die Surrogate sind "y" und "z". In meinem Makro dagegen verwende ich andere Formelzeichen!

**3. Problem:** Word 2000 sucht nicht automatisch eine passende Schriftart aus, und Word 2003 nicht immer. Je nach Unicode-Bereich muss ich deshalb in meinem Makro eine Schriftart vorschlagen. Eine nachträgliche Zuordnung von Hand funktioniert unzuverlässig. Einige Schriftarten habe ich auch vermieden, weil sie Probleme bei der Konvertierung in eine pdf-Datei machten.

**Tabelle 5.2: Für das Makro wichtige Unicode-Werte, umgerechnet in Dezimalwerte:**

| Bereich bzw. Grenzwert | Hexadezimal | Dezimal |
|---|---|---|
| Absolute untere Grenze | 00 20 | 32 |
| Surrogat 1 (High Surrogates) | D8 00 bis DB FF | 55 296 bis 56 319 |
| Surrogat 2 (Low Surrogates) | DC 00 bis DF FF | 56 320 bis 57 343 |
| Grenze für Ebene 0 | FF FF | 65 535 |
| Ab hier UTF-16 ungleich UCS | 1 00 00 | 65 536 |
| Obergrenze für Unicode | 10 FF FF | 1 114 111 |

## 5.2 Makro zur Erzeugung eines Schriftzeichens aus dem Hexcode

Dieses Makro ist ab Word 2003 überflüssig.

```
Sub Hexcode_in_Schriftzeichen_umwandeln()
' Erstellt von Dipl.-Ing. H. Chris Gast,  2. bis 8. Mai 2010, Stand 25. November 2014.
' Diese Prozedur kann sämtliche Unicode-Schriftzeichen  aus dem Hexcode erzeugen,
'   auch bei höheren Unicode-Ebenen, was in Word 2000 und Word 2003 sonst nicht möglich ist.
' Zu beachten ist, dass Windows UTF-16 nach ISO/IEC 10646 verwendet.
' Empfohlenes Tastenkürzel: Alt + C
'
  Dim z1 As String, z2 As String, y1 As Long, y2 As Long
  Dim sur1 As String, sur2 As String, Font1 As String
  Dim Abfrage As Integer
  Dim Rng1 As Range, Rng2 As Range
  ' z1  ist der einzugebende hexadezimale Unicodewert, z2 der Kontrollwert für die Eingabe,
  ' y1  ist der daraus berechnete dezimale Unicodewert,
  ' y2  ist der um hex. 10000 (dez. 65536) verminderte dez. Unicodewert (= Zwischenschritt),
  ' sur1 ist der gesuchte hexadezimale Wert des High Surrogate (Surrogat 1),
  ' sur2 ist der gesuchte hexadezimale Wert des Low Surrogate (Surrogat 2),
  ' Font 1 ist die zu wählende Schriftart
  '
  ' Wenn nichts markiert ist, links vom Cursor "Wort" markieren:
    If Selection.Type <> wdSelectionNormal Then
        Selection.MoveLeft Unit:=wdWord, Count:=1, Extend:=wdExtend
  '     Markierbereich nach vorne um "U+" verlängern, sofern zutreffend:
        Set Rng1 = Selection.Range
        Set Rng2 = Selection.Range
        Rng2.SetRange Start:=Rng2.Start - 2, End:=Rng2.Start
        If Rng2.Text = "U+" Then
            Rng2.SetRange Start:=Rng2.Start, End:=Rng1.End
            Rng2.Select
        End If
     End If
  '
  ' Markierte Textstelle soll der Eingabewert sein:
    z1 = Selection
  '
  ' Für Hexcode-Angabe besser Großbuchstaben nehmen:
    z1 = UCase(z1)
```

```
' ********************************
' versehentlich mitkopierte Leerzeichen beseitigen:
  z1 = Trim(z1)
' vorangestelltes "U+" bzw. "0x" löschen:
  If Left(z1, 2) = "U+" Or Left(z1, 2) = "0X" Then
     z1 = Right(z1, Len(z1) - 2)
  End If
'
' *********************************
' Grobe Prüfung des Wertebereichs:
  If Len(z1) > 6 Then
     MsgBox "Es sind nur Unicodes bis hex. 10FFFF zulässig."
     Exit Sub
  End If
'
  y1 = Val("&H" & z1 & "&")
'    Erläuterung:   Um höhere Zahlen als (dezimal) 32767 zu ermöglichen,
'    muss  [ & "&"] angehängt werden. Dies erzwingt eine Ausgabe als Zeichen
'    vom Typ "Long" anstelle vom Typ "Integer". Aber auch "Long" lässt nur 7 Stellen zu,
'    ohne Vorzeichenfehler zu machen, deshalb obige Prüfung.
'
' ******************
' Prüfung, ob Eingabe eine Hexzahl ist:
  z2 = z1
' Führende  Nullen beseitigen (auch in der Politik sinnvoll):
  Do While Left(z2, 1) = 0
     If Left(z2, 1) = 0 Then
        z2 = Right(z2, Len(z2) - 1)
     Else
       Exit Do
     End If
  Loop
'
  If z2 <> Hex(y1) Then
    ' MsgBox ("Eingabe war " & z1 & ", Dezwert ist " & y1 & ", Rückwandlung ist " & Hex(y1))
     MsgBox "Eingabe ist keine richtige Hexzahl (nur 0 bis 9 und A bis F)"
     Exit Sub
  End If
'
' ******************
' Fallunterscheidung:
  If y1 < 30 Then
      Abfrage = MsgBox("Es handelt sich um ein Steuerzeichen." _
            & vbCrLf & vbCrLf & "Wollen Sie dieses trotzdem erzeugen?", _
             vbYesNo + vbQuestion + vbDefaultButton2, "Windows-Steuerzeichen")
      If Abfrage = vbNo Then Exit Sub
  End If
'
  If y1 > 1114111 Then
     MsgBox "Unicodewerte über hex. 10FFFF bzw. dez. 1'114'111 sind nicht zulässig."
     Exit Sub
  End If
'
' Fall: Hexcode zwischen D800 und DFFF:
  If y1 > 55295 And y1 < 57344 Then
      Abfrage = MsgBox("Es handelt sich um ein Surrogat, kein zulässiges Zeichen!" _
            & vbCrLf & vbCrLf & "Wollen Sie dieses trotzdem erzeugen?", _
            vbYesNo + vbQuestion + vbDefaultButton2, "Windows-Steuerzeichen")
      If Abfrage = vbNo Then Exit Sub
  End If
'
' für Zeichen bis hex. FFFF gilt: Ergebnis ist "ChrW(y1)"
' ********************************************************
```

```
  ' Abfrage: Ist der Eingabewert eine hexadezimale Zahl kleiner gleich FFFF?
  ' If Val("&H" & z1) < Val("&H" & "10000") Then
  If y1 < 65536 Then
     '  "UTF-16 ist hier mit Unicode identisch"
     If y1 <= 1103 Then
        Selection.TypeText Text:=ChrW(y1)
     Else
        If y1 > 61471 And y1 < 61696 Then
           Font1 = InputBox("Bitte passende Schriftart wählen", "Hexcode = " & z1, "Symbol")
        Else
           Font1 = InputBox("Bitte passende Schriftart wählen", "Hexcode = " & z1, "DejaVu Serif")
        End If
        Selection.Font.Name = Font1
        Selection.TypeText Text:=ChrW(y1)
        Selection.Font.Name = Font1
     End If
     Exit Sub
  End If
  '
  ' ****************************************
  '  Für Schriftzeichen mit Unicode höherer Ebenen (hex.10000 bis hex.10FFFF) müssen Surrogate
  '  nach ISO/IEC 10646:2003, Anhang C angewendet werden.
  '  Für die Surrogate verwende ich die Formelzeichen "sur1" und "sur2".
  '     und "y2" als Zwischenwert für "Eingabewert minus hex. 00010000":
  ' y2 = Val("&H" & z1) - Val("&H" & "10000")
  y2 = y1 - 65536
  '
  ' Die Umrechnung von hex. D800 mittel "Val(&HD800)" führte bei hohen Werten
  '    zu negativen Zahlen, deshalb statt "hex.D800" besser dezimal "55296".
  ' Statt "Val(&HD400)" schreibe ich einfacher direkt dezimal "1024":
  '
  ' Daraus ergeben sich als Surrogate:
   sur1 = Hex(Int(y2 / 1024) + 55296)
   sur2 = Hex(y2 - (Int(y2 / 1024) * 1024) + 56320)
'
'  Anzeige = MsgBox(" Der Hexcode ist " & z1 & vbCrLf & " Der y1 ist: " _
'     & y1 & vbCrLf & " Das erste Surrogat ist: " & sur1 & vbCrLf & "Das zweite Surrogat ist: " _
'     & sur2, 0, "Zeichen aus Unicode")
'
'   ******************************
'   passende Schriftart festlegen (Font):
   If y1 >= 65792 And y1 <= 65930 Then
      Font1 = InputBox("Bitte passende Schriftart wählen!", "hex. Unicode = " & z1, "Aegean")
   Else
      If y1 >= 77824 And y1 <= 78894 Then
         Font1 = InputBox("Bitte passende Schriftart wählen!", "hex. Unicode = " & z1, "Gardiner")
      Else
         ' Besser Font "Symbola" verwenden, 24. November 2014.
         Font1 = InputBox("Bitte passende Schriftart wählen!", "hex. Unicode = " & Z1, "Symbola")
      End If
   End If
'
'  ******************************
'  Ergebnis an Stelle des Codes einfügen:
   Selection.Font.Name = Font1
   Selection.Font.Bold = True
   Selection.Font.Size = 14
   Selection.TypeText Text:=ChrW("&H" & sur1) & ChrW("&H" & sur2)
End Sub
```

## 5.3 Makro zur Erzeugung eines Zeichens aus dem Dezimalcode

Das folgende Makro wandelt den markierten dezimalen Zahlencode bzw. den dezimalen Zahlencode links vom Cursor in das zugehörige Unicode-Zeichen um. Im Bereich dezimal 129 bis 159 werden die Zeichen des jeweiligen Windows-Code (bei mir Windows 1252) erzeugt.

Zeichen aus dem DOS-Code erzeugt das folgende Makro nicht; Eine Unterscheidung zwischen Dezimalcodes mit einer Null davor und ohne Null davor wie bei Windows-Computern wird hier nicht gemacht.[24]

Das folgende Makro ist meines Erachtens auch noch in Word 2010 bequemer als das Eintippen des Codes über den Nummernblock.

```
Sub Dezimalcode_in_Schriftzeichen_umwandeln()
'  Erstellt von H. Chris Gast, Berlin, Mai 2009 und August 2010, 16. Februar 2011, 22. November
2014.
'  Makro zum Erzeugen eines Schriftzeichens aus dem Dezimalcode,
'  funktioniert auch im Windows-Sonderbereich  dezimal  0128 bis 0159
'  und für höhere Unicode-Ebenen.
'  Aufrufen  mit Alt + k
'
  Dim y1 As Variant, y2 As Long
  Dim sur1 As String, sur2 As String
  Dim Font1 As String
  Dim Abfrage As Integer
  '
  '  y1  ist der markierte dezimale Code,
  '  y2  ist der um hex. 10000 (dez. 65536) verminderte dez. Unicodewert (= Zwischenschritt),
  '  sur1 ist der gesuchte hexadezimale Wert des High Surrogate (Surrogat 1),
  '  sur2 ist der gesuchte hexadezimale Wert des Low Surrogate (Surrogat 2),
  '  Font 1 ist die zu wählende Schriftart
  '
  '  Wenn nichts markiert ist, links vom Cursor "Wort" markieren (dies kann bis zu 6-stellig sein):
  If Selection.Type <> wdSelectionNormal Then
    Selection.MoveLeft Unit:=wdWord, Count:=1, Extend:=wdExtend
  End If
  y1 = Selection
  '
  ' Prüfung, ob eine Zahl markiert wurde:
  If IsNumeric(y1) = False Then
       MsgBox "Markierter Wert muss eine reine dezimale Zahl sein!"
       Exit Sub
  End If
  '
  If y1 > 1114111 Then
      MsgBox "Unicodewerte über hex. 10FFFF bzw. dez. 1'114'111 sind nicht zulässig."
      Exit Sub
  End If
  '
  ' *********************
  '  Fallunterscheidung:
  If y1 < 30 Then
     Abfrage = MsgBox("Wollen Sie ein Steuerzeichen erzeugen?", 292, "Windows-Steuerzeichen")
  '  Erklärung der Abfragebox: (VbYesNo = 4) + (VbQuestion = 32) + (VbDefaultButton2 = 256)
     If Abfrage = vbNo Then
        Exit Sub
     End If
```

[24] Auf Computern mit Windows gilt zum Beispiel: Alt + Ziffer 9 ergibt ○ (weißer Kreis), dagegen Alt + Ziffern 09 ergibt Steuerzeichen "Tabulator".

```
  End If
  '
  ' Fall: Hexcode zwischen D800 und DFFF:
  If y1 > 55295 And y1 < 57344 Then
       Abfrage = MsgBox("Es handelt sich um ein Surrogat, kein zulässiges Zeichen!" _
             & vbCrLf & vbCrLf & "Wollen Sie dieses trotzdem erzeugen?", _
             vbYesNo + vbQuestion + vbDefaultButton2, "Windows-Steuerzeichen")
       If Abfrage = vbNo Then Exit Sub
  End If
  '
  If y1 < 65536 Then
      ' (Bis hex. FFFF ist UTF-16 mit dem einfachen Unicode identisch)
      '
      If y1 > 127 And y1 < 160 Then
         MsgBox "Achtung, Windows-Sonderbereich!"
         Selection.TypeText Text:=Chr(y1)
      Else
         Selection.TypeText Text:=ChrW(y1)
      End If
      '
      If y1 > 61471 And y1 < 61696 Then
          Font1 = InputBox("Bitte Sonderschriftart eingeben", "Achtung Sonderschrift!", "Symbol")
          Selection.MoveLeft Unit:=wdCharacter, Count:=1, Extend:=wdExtend
          Selection.Font.Name = Font1
      End If
      Exit Sub
  End If
  '
  ' ****************************************
  '  Für Schriftzeichen mit Unicode höherer Ebenen (hex.10000 bis hex.10FFFF) müssen Surrogate
  '  nach ISO/IEC 10646:2003, Anhang C angewendet werden.
  '  Für die Surrogate verwende ich die Formelzeichen "sur1" und "sur2".
  '     und "y2" als Zwischenwert für "Eingabewert minus hex. 00010000":
  ' y2 = Val("&H" & z1) - Val("&H" & "10000")
  y2 = y1 - 65536
  '
  ' Die Umrechnung von hex. D800 mittel "Val(&HD800)" führte bei hohen Werten
  '    zu negativen Zahlen, deshalb statt "hex.D800" besser dezimal "55296".
  ' Statt "Val(&HD400)" schreibe ich einfacher direkt dezimal "1024":
  '
  ' Daraus ergeben sich als Surrogate:
   sur1 = Hex(Int(y2 / 1024) + 55296)
   sur2 = Hex(y2 - (Int(y2 / 1024) * 1024) + 56320)
'
'  Anzeige = MsgBox(" Der Hexcode ist " & z1 & vbCrLf & " Der Dezimalcode1 ist: " _
'     & y1 & vbCrLf & " Das erste Surrogat ist: " & sur1 & vbCrLf & "Das zweite Surrogat ist: " _
'     & sur2, 0, "Zeichen aus Unicode")
'
'  ******************************
'  passende Schriftart festlegen (Font):
   If y1 >= 65792 And y1 <= 65930 Then
      Font1 = InputBox("Bitte passende Schriftart wählen!", "Dez. Unicode = " & y1, "Aegean")
   Else
      If y1 >= 77824 And y1 <= 78894 Then
         Font1 = InputBox("Bitte passende Schriftart wählen!", "Dez. Unicode = " & y1, "Gardiner")
      Else
        " Font1 = InputBox("Bitte passende Schriftart wählen!", "Dez. Unicode = " & y1, "Musica")
         ' besser "Symbola" vorschlagen, 22. November 2014.
         Font1 = InputBox("Bitte passende Schriftart wählen!", "Dez. Unicode = " & y1, "Symbola")
      End If
   End If
'
'  ******************************
```

```
' Ergebnis an Stelle des Codes einfügen:
   Selection.Font.Name = Font1
   Selection.Font.Bold = True
   Selection.Font.Size = 14
   Selection.TypeText Text:=ChrW("&H" & sur1) & ChrW("&H" & sur2)
End Sub
```

## 5.4 Auswertung und Beurteilung der beiden Makros in 5.2 und 5.3

Die beiden Makros zur Erzeugung von Schriftzeichen erwiesen sich (nach einigen Verbesserungen, auf die ich hier nicht im Einzelnen eingehen will) geeignet zur Erzeugung aller Unicode-Schriftzeichen. Die Makros eignen sich auch für die Erzeugung von Zeichen höherer Unicode-Ebenen (>hex. FFFF), von Steuerzeichen und von Zeichen der Windows-Sonderschriftarten.

Mit dem Makro für die Erzeugung von Zeichen aus dem dezimalen Code können auch Schriftzeichen des Windows-Sonderbereichs von (dezimal) 0128 bis 0159 erzeugt werden.

Bei Eingabe des Codes im "Privat Use"-Bereich wird die Windows-Sonderschriftart "Symbol" als Default-Schrift vorgeschlagen, bei Schriftzeichen der höheren Unicode-Ebenen habe ich als Default-Schrift "Musica" gewählt, weil ich mit dieser die meisten Schriftzeichen darstellen konnte, siehe Tabelle 4.1.

Diese beiden Makros sind nützlich für die Erzeugung von Schrift- und Sonderzeichen in Word 2000 und Word 2003, ist aber nicht mehr nötig für Word 2007.

# 6 Makro zur Analyse auch von Schriftzeichen höherer Unicode-Ebenen (UCS > $FFFF_{hex}$)

## 6.1 Die einzelnen Probleme und Lösungen

### 6.1.1 Unicode-Zeichen höherer Ebenen

Zeichen aus höheren Unicode-Ebenen können im Word 2000 mit den vorhandenen Funktionen nicht analysiert werden. Deshalb wollte ich dafür ein Makro schreiben. Diese Zeichen werden in Word (Windows) mittels **zwei** Surrogate nach ISO/IEC 10646:2003, Anhang C dargestellt, das Windows mit UTF-16 arbeitet. Dabei sind im Unicode-Bereich hex. D800 bis hex. DBFF das obere Surrogat (in Schreibrichtung gesehen das erste Zeichen) und im Bereich hex. DC00 bis hex. DFFF das untere Surrogat (rechts davon). Nur wenn beide Surrogate vorhanden sind und richtig angeordnet, ergibt sich ein zulässiges Unicode-Zeichen höherer Ebenen. Das folgende Makro (siehe Abschnitt 6.2) erkennt, ob ein zulässiges Surrogatpaar vorliegt und berechnet daraus auch die Unicodes höherer Ebenen[25].

Nachdem die zwei UTF-16-Surrogate ermittelt sind, muss daraus der gesuchte Unicode wie folgt nach ISO 10646 berechnet werden:

[25] seit Sonntag, 13. Juni 2010.

**Auszug aus ISO 10646:2003, Anhang C:**

| UTF-16, 2-octet elements | | UCS-4 (4-octet) |
|---|---|---|
| x = | 0000; ... D7FF; | x |
| x = | E000; ... FFFF; | x |
| pair (x, y) such that | | |
| x = | D800; ... DBFF; | ((x - D800) * 400 |
| y = | DC00; ... DFFF; | + (y - DC00)) |
| | | + 0001 0000 |

Ich verwende im Makro statt x und y die Formelzeichen z1 und z2 für die Surrogate.

Weil das Rechnen bei meinen Makros meist nur mit dezimalen Zahlen funktioniert, rechnete ich vorher die Konstanten um, siehe die folgende Tabelle:

**Tabelle 6.1: Hexwerte umgerechnet in Dezimalwerte:**

| Hexadezimal | Dezimal |
|---|---|
| 0020 | 32 |
| 0400 | 1024 |
| D8 00 | 55 296 |
| DC 00 | 56 320 |
| 1 00 00 | 65 536 |

Die obige Gleichung nach ISO 10646:2003, Anhang C als VBA-Makro:

```
y(1) = Val("&H" & z(1) & "&")
y(2) = Val("&H" & z(2) & "&")
y(0) = ((y(1) - 55296) * 1024) + (y(2) - 56320) + 65536
z(0) = Hex(y(0))
MsgBox "Der Unicode des Zeichens höherer Ebene ist: " & z(0) &" !"
```

Bei Unicodes höherer Ebenen war für mich überraschenderweise nicht die Umrechnung der Surrogate nach der ISO-Norm das Problem, sondern das genaue Auffinden der beiden Surrogate im Text bei Word 2003. Mehrere Makro-Ansätze funktionierten in Word 2000, aber nicht in Word 2003. Ein anderer Ansatz funktionierte in Word 2003, aber nicht in Word 2000[26]. Schließlich blieben nur die beiden Analyse-Varianten:

a) mit der Mid-Funktion:

```
x(i) = Mid(Selection, i, 1)
```

b) die Zerlegung des markierten Bereichs in **einzelne Ranges**

```
Set Rg(i) = Selection.Range
Rg(i).SetRange Start:=Rg(i).Start + i - 1, End:=Rg(i).Start + i
```

Nur diese erkannten sowohl in Word 2000 als auch in Word 2003 gleichermaßen die einzelnen Surrogate richtig. Nachdem das Makro schließlich sowohl in Word 2000 als auch in Word 2003 lief, funktionierten die beiden Varianten auch problemlos unter Word 2007[27]. Da aber bei der Mid-Funktion die Windows-Sonderschriftarten (siehe 6.1.5) nur sicher erkannt werden, wenn sie an erster Stelle der Zeichenkette (String) stehen, wählte ich die Variante mit der Zerteilung in Ranges, obwohl dieses wiederum Probleme bei Steuerzeichen für Tabellen und Felder macht.

---

[26] Der Trim-Befehl zerstörte mir Surrogate, der Move- und der Next-Befehl übersprangen in Word 2003 die Einzelbytes (die Surrogate), eine anderer Befehl funktionierte dafür zwar in Word 2003, aber nicht in Word 2000, usw., bis ich als Lösung die Mid-Funktion fand, die sich in beiden Word-Versionen auch bei Surrogaten gleichartig verhielt.

[27] Die Tests mit Word 2007 erfolgten in einem Copyshop, da ich dieses nicht habe. (Inzwischen stellte sich heraus, dass das fertige Analyse-Programm auch problemlos unter Word 2010 läuft).

### 6.1.2 Probleme mit hohen Zahlen beim VBA-Datentyp "Integer"

Probleme gab es auch bei der Funktion der Code-Ermittlung mit "AscW(Zeichen)". Da der Code in VBA als Datentyp "Integer" ausgegeben wird, der nur bis 32 767 positive Zahlenwerte ermöglicht, waren einige "Verrenkungen" in der Programmierung nötig. Ich ermittelte zuerst die hexadezimalen Werte mit "Hex(AscW(Zeichen)" und dann mit der Val-Funktion wie in 5.1 die dezimalen Werte. Bei der Angabe des Dezimalcodes im Bereich 128 bis 255 fügte ich in der Ergebnis-Anzeige eine voranzustellende Null ein, die man davorsetzen muss, wenn man unter Windows den Dezimalcode mit Alt + Zahlenwert auf der Nummerntastatur eingibt, damit Windows nicht den alten DOS-Code statt des Windows-Codes verwendet.

### 6.1.3 Erkennen von kombinierenden diakritischen Zeichen

Ferner gibt es Kombinierende Zeichen, die mit dem Buchstaben davor optisch verschmelzen, hauptsächlich kombinierende diakritische Zeichen (en: COMBINING DIACRITICAL MARKS) aus dem Unicode-Bereich hex. 0300 bis hex. 036F. Weitere kombinierende Zeichen siehe ISO/IEC 10646:2003, Anhang B [28]. Das folgende Makro kann bei der "Autofokus-Funktion" bis zu vier diakritische Zeichen feststellen, und auch, wenn diese mit einem Schriftzeichen der Windows-Sonderschriftart Symbol kombiniert sind. So ist sichergestellt, dass von einem Buchstaben mit diakritischen Zeichen letztere vollständig mit-analysiert werden.

### 6.1.4 Sonderzeichen in der Messagebox

Ein weiteres Problem ist die Ergebnisanzeige in einer Messagebox. Die MsgBox-Funktion von Visual Basic (bzw. VBA) ist nicht Unicode-tauglich. Teilweise wurden fehlerhaft "a" statt "α" und "d" statt "δ" angezeigt, während das kleine Beta wiederum richtig aussah. Um trotzdem die einzelnen Zeichen erklären zu können, verwendete ich eine API-Funktion, nämlich die MessageBoxW von Windows, wobei mit der Funktion StrPtr (Stringpointer) nur die Speicheradresse des Strings "MsgBoxAnzeige" übergeben wird, was verhindert, dass sonst der Unicode beschnitten wird und fehlerhafte Anzeigen erfolgen [15]. Am Anfang des Makromoduls (vor der ersten Prozedur) muss allerdings der Aufruf der MessageBoxW deklariert werden. Damit ist das Makro allerdings nur in Windows-Umgebung lauffähig, und da auch in der MessageBoxW nur die Systemschrift von Windows verwendet werden kann, sind auch weiterhin nicht alle Zeichen anzeigbar. Hier half nur die Ausgabe als separates Word-Dokument, die ich in meiner Messagebox in der letzten Zeile unter der Anklickvariante "Ja" anbiete, siehe Ergebnis-Beispiele in Abschnitt 6.3.2.

In der Ergebnisanzeige verzichtete ich bei der Angabe der einzelnen Zeichen auf Anführungszeichen, weil sonst einige kombinierenden Zeichen oder Schriftzeichen mit Strichen nicht immer eindeutig lesbar sind.

---

[28] Inzwischen Version ISO/IEC 10646:2012, Anhang B.

### 6.1.5 Probleme mit den Windows-Sonderschriftarten

Das größte Problem stellten bei der Erstellung des Analyse-Makros die Zeichen der **Windows-Sonderschriftarten** wie Symbol und Wingdings dar. Für Zeichen der Windows-Sonderschriftarten funktionieren nämlich die normalen Regeln der Zeichencodebestimmung mit den Befehlen "Asc(Zeichen)" bzw. "AscW(Zeichen)" und der Schriftartbestimmung mit "Selection.Font.Name" nicht, wenn diese über "Einfügen → Symbol" eingegeben worden waren. Word setzt nämlich im Quelltext an dieser Stelle dann eine "*Runde Klammer links*" (englisch: "LEFT PARENTHESIS"):

αβχ (Buchstaben in Symbol-Schrift umgewandelt) → Analyse funktionierte richtig.
αβχ (über Einfügen → Symbol → Schriftart Symbol) → Analyse ergab fälschlich "(".

Hier hilft lediglich die folgende Analyse-Methode mit einem "Dialog" [16]:

```
   If x(i) = "(" Or x(i) Like "[" & ChrW(61472) & "-" & ChrW(61695) & "]"
Then
      Rg(i).Select
      With Dialogs(wdDialogInsertSymbol)
            f1 = .Font
            Zeichennummer1 = .CharNum
      End With
```

Diese Abfrage mit dem "Dialog" funktioniert allerdings nur bei Einzelzeichen-Analyse beziehungsweise nur beim ersten Zeichen einer Zeichenkette. Deshalb konnte ich mit dem einfachen Analyse-Makro in A.2 nicht weiterarbeiten, sondern musste die Markierung in Ranges mit einzelnen Zeichen zerlegen.

Windows-Sonderzeichen sind nicht in den Unicode-Tabellen der ISO 10646 enthalten. Ihre Codes liegen im dem frei verfügbaren "PUA"-Bereich (PUA = "Private Use Area"). Für die beiden häufigsten Windows-Sonderschriftarten "Symbol" und "Wingdings" ergänzte ich spezielle Unterfunktionen, die diese Zeichen mit ihrer Bedeutung und, soweit vorhanden, mit den Unicode-Entsprechungen enthalten. Details dazu finden sich unter [24].

Ein Problem kann sich in seltenen Fällen ergeben, wenn die Umwandlung normaler Buchstaben in Symbol-Zeichen nicht korrekt erfolgte. Wird einem Zeichen mit einem Code größer als dezimal 255 nachträglich die Schriftart "Symbol" zugeordnet, werden für dessen Anzeige auf dem Bildschirm manchmal nur die letzten zwei hexadezimale Stellen des Codes berücksichtig. Aus dem Ohmzeichen "Ω" (Unicode hex. 2126) wird dann zum Beispiel fälschlich "&", während die Computer-interne Codierung der Code für Ohmzeichen bleibt! Ob der hexadezimale Code nun "26", "126", "226", "426", "1026" oder richtig "F026" lautet ("&", "Ħ", "À" usw.), das Zeichen wird bei Wechsel in die Schriftart "Symbol" optisch immer zum "Ampersand"!

### 6.1.6 Probleme mit Steuerzeichen für Feldfunktionen und für Tabellen

Am problematischsten waren (abgesehen von den Windows-Sonderschriften) die **Steuerzeichen für Tabellen- und Feldfunktionen** von Word. Diese können zwar mit der Funktion "Mid(Selection, i, 1)" gut ermittelt werden, führen aber bei der Zerlegung des markierten Textes in Ranges zu Problemen. Der Befehl zur Windows-Sonderzeichen-Erkennung führt bei diesen Steuerzeichen zum Absturz des Makros. Deshalb musste ich die Steuerzeichen für Feldfunktionen vorher separat ermitteln und vor der eigentlichen Analyse aus dem zu analysierenden Bereich entfernen.

a) Bei sichtbarem Feldcode wird die Markierung auf die Zeichen innerhalb des Feldes beschränkt (bei mehreren Feldern auf den Feldcode des ersten Feldes).

b) Bei Tabellen bzw. Tabellenausschnitte werden die markierten Tabellenfelder in einer separaten (nicht sichtbaren) Datei in Text umgewandelt. Anschließend wird dieser Text so aufbereitet, dass er dem Quelltext der markierten Tabellenfelder entsprechen würde. Hierbei stellt sich die Frage, ob auch bei Tabellenausschnitten das doppelte Tabellenfeld-Steuerzeichen ChrW(7) am Ende der Tabellen-Reihe angezeigt werden sollte, oder nur dann, wenn die ganze Tabellen-Reihe markiert war. Ich entschied mich, Tabellenfelder in der Markierung wie eine selbstständige Tabelle zu analyiseren.
Ferner wird bei der Analyse eines Strings (siehe Makro in Abschnitt A.2) eine etwas abweichende Codierung ermittelt (nämlich 13 + 7) als im Quelltext (nur "7"). Mein fertiges Analyse-Makro orientiert sich am Quelltext.

c) Bei nichtsichtbarem Feldcode wird der markierte Text in einer separaten Datei untersucht, wobei erst die Feldcode-Ansicht für die Anzeige im Ergebnis in einer Zeichenkette[29] gespeichert wird, dann werden die Feldfunktionen aufgehoben (*Selection.Fields.Unlink*) und das Feldergebnis analysiert. Unbekannte Schriftzeichen befinden sich nie im Feldcode, sondern stets nur im Feldergebnis. Sollte wider Erwarten doch noch der Feldcode analysiert werden, kann man das Ergebnis als Datei ausgeben lassen und dort mit demselben Analysemakro die Zeile mit der Feldcode-Ansicht analysieren, z. B. "*▯ REF _Ref293239865 \h ▯* ".

d) Bei durch-Feldfunktion geschützten-Sonderzeichen, bei denen der Unlink-Befehl nicht funktioniert, musste ich die Word-Feldfunktion für die Analyse durch das Makro erst aufwendig nachbilden, und für sonstige Funktionen, bei denen der Unlink-Befehl nicht funktioniert wie bei der Overstrike-Funktion, durch Kopieren und Inhalte-Einfügen die Feldfunktionen beseitigen.

Die folgende Tabelle zeigt eine Übersicht verschiedener Feldfunktionen.

**Tabelle 6.2: Beispiele für Feldfunktionen**

| Art des Feldes | Feldergebnis | Feldcode *(Screenshots)* | Quelltext in Word 2000 |
|---|---|---|---|
| Seitenzahl | 23 | {PAGE } | ▯ PAGE ▯ - 1 -▯ |
| Speicherdatum | 14. Januar 2015 | { SAVEDATE \@ "d. MMMM yyyy" } | ▯ SAVEDATE \@ "d. MMMM yyyy" ▯ 13. April 2012▯ |
| Index-Eintrag | *(ist ausgeblendet)* | { XE "Sonderzeichen" } | (ist ausgeblendet)<br>▯ XE "Sonderzeichen" ▯ |
| Unsichtbare Titel für das Inhalts-verzeichnis | *(ist ausgeblendet)* | { TC "Überschrift Ebene 3" \l 3} | (ist ausgeblendet)<br>▯ TC "Überschrift Ebene 3" \l 3▯ |
| Overstrike-Feld | d | {eq \o(d;_)} | ▯ eq \o (d;_)▯ |

[29] Achtung: Windows-Sonderschriftarten sind in dieser Ergebniszeile nicht erkennbar.

**Tabelle 6.2: Beispiele für Feldfunktionen** *(fortgesetzt)*

| Art des Feldes | Feldergebnis | Feldcode *(Screenshots)* | Quelltext in Word 2000 |
|---|---|---|---|
| Geschütztes Sonderzeichen | Ω | SYMBOL 61527 \f "Symbol" \s 10 | SYMBOL 61527 \f "Symbol" \s 10 W |
| Querverweis auf einen Abschnitt | Siehe Abschnitt 5.2 | Siehe Abschnitt REF _Ref261203182 \r \h * MERGEFORMAT | Siehe Abschnitt REF _Ref261203182 \r \h * MERGEFORMAT 5.2 |
| Querverweis auf Überschrift mit Sonderzeichen | Siehe Testüberschrift Farbmarkierung ω | Siehe Testüberschrift ↵ REF _Ref322083838 \h * MERGEFORMAT | Siehe Testüberschrift EF _Ref322083838 \h Farbmarkierung wð |

= Steuerzeichen 19
= Steuerzeichen 20
= Steuerzeichen 21
= Steuerzeichen 1

Der Quelltext der eingefügten Screenshots: EMBED MSPhotoEd.3

Der Querverweis auf eine Überschrift mit Sonderzeichen lautet im Quelltext bei **Word 2010:**

Siehe Testüberschrift REF _Ref322083838 \h Farbmarkierung w

Hier ist der Unterschied zwischen Word 2000 und Word 2010 (im "Kompatibilitätsmodus") gut zu sehen. Sobald ein Zeichen der Sonderschriftarten Symbol oder Wingdings in der Datei vorkommt, werden im Quelltext zwischen alle normalen Unicode-kompatiblen Zeichen im gesamten Dokument je ein Leerzeichen gesetzt, und nur beim Zeichen der Sonderschriftart "Symbol" das Zeichen "ð".

### 6.1.7 Ermittlung der (englischen) Namen der Unicode-Schriftzeichen

In der wahlweisen Anzeige des Ergebnisses in einer separaten Datei wird auch der englische Name des Zeichens aus einer umfangreichen Unicode-Liste [2] angezeigt, sofern das Zeichen als Unicode-Zeichen genormt ist. Bei Symbol-Zeichen wird das entsprechende Unicode-Zeichen angegeben. Chinesische, japanische und ein Teil der koreanischen Unicode-Zeichen sind in dieser Liste nicht aufgeführt. In meinem Makro wird stattdessen auf die jeweilige Quelle im Internet verwiesen, die nur als recht große Pdf-Dateien vorliegen. Deshalb habe ich hier auf ein automatisches Nachschlagen verzichtet.

Die Textdatei mit der Unicode-Liste habe ich für mein Makro im Ordner mit den Word-Vorlagen gespeichert (bei den ".dot"-Dateien), da ich keinen ständigen und auch keinen schnellen Internetzugang habe. Nur wenn das Makro diese Datei dort nicht findet, und auch nicht im Ordner "Word\STARTUP", wird im Internet nachgesehen.

Zuhause mit nicht sehr schnellem Internetzugang und Word 2000 braucht das Makro zur Zeichenanalyse mit direktem Internetzugriff beim ersten Aufruf nach dem Löschen aller temporären Dateien je nach Qualität der Internetverbindung zwischen 11 und 70 Sekunden und bei Folgeaufrufen 3 bis 4,8 Sekunden. Beim Zugriff auf die Kopie der Unicode-Liste auf der Festplatte (also ohne Internet) dauert der Zugriff dagegen lediglich zwischen 0,8 und 1,2 Sekunden.[30]

Im Büro mit schnellerem Internetzugang und Word 2003 dauerte der Internetzugriff beim ersten Mal nur etwa 6 bis 7 Sekunden, bei Folgezugriffen war der Internetzugriff nur

[30] Als Test für die Rechenzeit-Messung wählte ich ein (unsinniges) Zeichen der Unicode-Ebene 1 mit drei kombinierenden diakritischen Zeichen (× ˘).

unwesentlich langsamer als der Zugriff auf die Kopie der Unicode-Liste auf der Festplatte (0,62 bis 0,75 Sekunden gegenüber 0,44 bis 0,47 Sekunden).

Im Copyshop mit Word 2007 dauert das Makro beim ersten Internetzugriff etwas über 10 Sekunden und Folgezugriffe etwa 1 Sekunde. Ob ich dort auf dem fremden Computer die Unicode-Liste auf der Festplatte speichern könnte, probierte ich nicht aus.

Für die Weitergabe an Kollegen hat die Möglichkeit des direkten Internetzugriffs den Vorteil, dass die Kollegen die Unicode-Liste nicht erst auf der Festplatte speichern müssen, und das Programm außer beim ersten Mal trotzdem nicht wesentlich langsamer läuft.

### 6.1.8 Weitere Steuerzeichen von Word

Word verwendet nicht nur für Feldfunktionen und für Tabellen, sondern auch für Absatzmarken, Feste Leerzeichen, Geschützte Bindestriche, Bedingte Trennstriche, Tabulatoren usw. Steuerzeichen im Codebereich von (dezimal) 01 bis 31.

Diese werden vom Makro problemlos analysiert. In der Ergebnis-Anzeige habe ich einige davon zur besseren Verständlichkeit durch ihre deutsche Beschreibung ersetzt. Weitere Word-Steuerzeichen wie etwa das für Fußnotenzeichen finden sich unter [23]. Zu dem Problem mit Tabellenfeld-Steuerzeichen und Feld-Steuerzeichen siehe oben 6.1.6.

### 6.1.9 Grafiken, Textfelder, Zeichnungen und ähnliches im Tex

Wenn Bilder oder Formeleditor-Objekte in der Markierung enthalten sind, erscheint in der Analyse der Code "01".

Wenn ein Grafik-Anker markiert ist, wird dies vorab durch die Abfrage des Markiertyps abgefangen ("Selection.Type").

Für die seltenen Fällen, in denen ein Grafik-Anker, ein Zeichenobjekt ("Autoform") oder die Linien eines Textfeldes ("Textbox") in der Markierung enthalten sind, stürzte das Makro ab. Es stellte sich heraus, dass die störenden Objekte alle "Shapes" waren, während "Frames" und "InlineShapes" die Analyse nicht behinderten. Will man trotz enthaltener "Shapes" den Text analysieren, muss man die markierte Stelle mitsamt diesen "Shapes" in eine Zwischendatei kopieren und alle "Shapes löschen":

```
Dim i As Integer
For i = ActiveDocument.Shapes.Count To 1 Step -1
  ActiveDocument.Shapes(i).Delete
Next i
```

Dabei ging allerdings der Inhalt von Textfeldern verloren. Dies umgeht das Makro durch die Umwandlung von Textfeldern (Textbox) in Textrahmen (Frame).

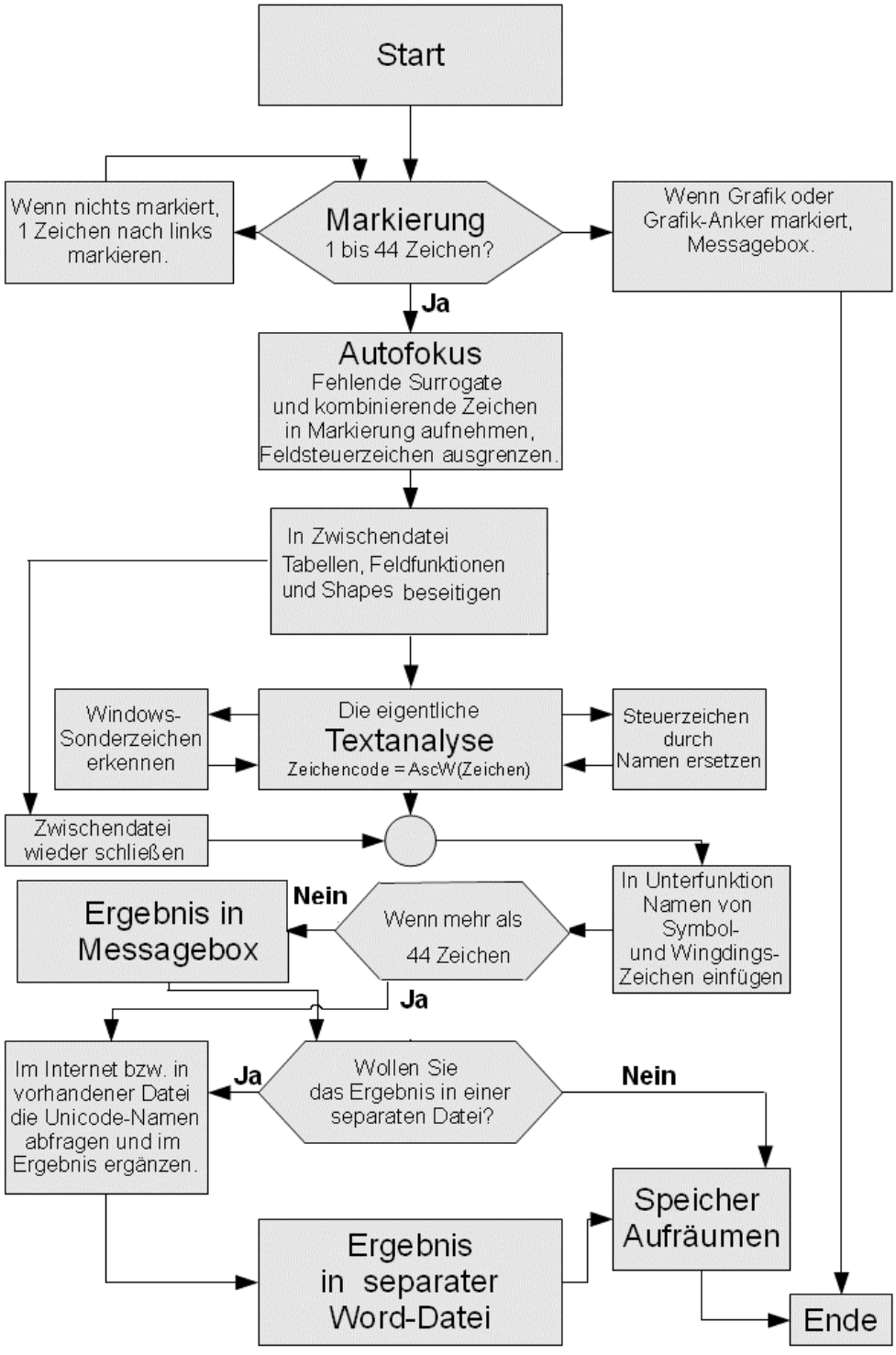
Shapes, die frei positionierte Bilder sind, lassen sich in InlineShapes umwandeln (normale Bilder in der Zeile). Dies bringt aber für die Analyse nichts. Deshalb werden sie in der Zwischendatei wie auch die AutoShapes gelöscht.

Die Abfrage `If Selection.Range.ShapeRange.Count > 0 Then`
brachte einen Absturz in Word 2000 (aber nicht in Word 2010), wenn die Markierung in einer Fußnote oder in einem Textfeld ist. Dort sind keine Shapes zulässig.
Deshalb wird im Analysemakro jetzt (seit Oktober 2012) auf die Abfrage verzichtet und dafür immer eine Zwischendatei verwendet. Dies kostet keine nennenswert Rechenzeit.

## 6.2 Das fertige Makro zur Analyse von Schrift- und Sonderzeichen, auch von Windows-Sonderzeichen, von höheren Unicode-Ebenen und von Schriftzeichen mit kombinierenden diakritischen Zeichen

Die Hauptprozedur des Makros befindet sich in Abschnitt 6.2.3 auf Seite 27 ff. Die untergeordnete Funktion zur Ermittlung der Unicode-Zeichennamen siehe Abschnitt 6.2.4, die Funktion zur Angabe der Unicode-Entsprechungen für Symbol-Zeichen siehe 6.2.5, und die Funktion zur Angabe der Bedeutung von Wingdingszeichen siehe 6.2.6. Zur Installation des Makros auf dem Computer siehe Abschnitt 7.

### 6.2.1 Funktionsablauf des Analyse-Makro



### 6.2.2 Die Deklarationen am Anfang des Makro-Moduls

```
' Modul "Unicode_Analyse"
' Dipl.-Ing. H. Chris Gast, Berlin (c) 2010, 2011, 2012, Stand 2014.
' http://www.siebener-kurier.de/chris-aufsaetze/Word-Erweiterung-Unicode-Makros.pdf
' und
' http://www.siebener-kurier.de/chris-aufsaetze/Zeichen-Analyse-kompakt.zip

Option Explicit
' erzeugt Fehlermeldung, wenn beim Programmieren Dim-Befehle vergessen wurden.

' Die folgende Funktions-Deklaration muss vor dem ersten Makro des Moduls eingefügt werden:
Private Declare Function MessageBoxW Lib "user32.dll" (ByVal Hwnd As Long, _
    ByVal lpText As Long, ByVal lpCaption As Long, ByVal uType As Long) As Long
```

### 6.2.3 Die Hauptprozedur

```
Sub Hexcode_und_Name_ermitteln()
' Erstellt von Dipl.-Ing. Hanna-Chris Gast, 2010, 2011, 2012, März 2014, Stand 19. Dezember 2014.
' Funktioniert in Word 2000 bis Word 2010 sowie unter Windows XP und Windows 7.
'
' Dieses Makro ermittelt die Schriftart und die Unicode-Werte eines markierten Zeichens oder
' einer markierten Zeichenfolge, aber auch die Schriftart und den Code der beiden
' Sonderschriftarten "Symbol" und "Wingdings". Bei diesen werden, soweit vorhanden,
' die entsprechenden Unicodezeichen mit (englischen) Namen angegeben.
'
' Das Analyse-Ergebnis wird bei bis zu 40 Zeichen in einer Messagebox ausgegeben, kann aber
' auch in eine separate Datei ausgegeben werden. Bei über 40 Zeichen wird das Ergebnis direkt
' in einer separaten Datei ausgegeben. In der separaten Datei mit dem Ergebnisprotokoll werden
' auch die jeweiligen (englischen) Namen der Unicode-Zeichen nach ISO/IEC 10646 angegeben,
' außer bei Steuerzeichen ("Control").
'
' Bei Feldern im Word-Text wird nur noch das Feldergebnis analysiert. In der Feldcode-Ansicht
' befinden sich im Feld keine unbekannten Schriftzeichen.
'
' Bei Textmarken in der Markierung werden diese ebenfalls aufgelistet (ergänzt November 2012).
'
' Dieses Makro kann seit 2011 auch in Tabellen mehrere Tabellen-Zellen auf einmal analysieren
' Dieses Makro analysiert auch Felder (seit Herbst 2011 auch bei Double-Strike).
'
' In Word 2010 gehen bei "Selection.Paste" manchmal die Absatz-Schriftart verloren,
' und Leerzeichen werden nicht zwischen Absatzmarken eingefügt. Dies wird gesondert abgefangen
' (ergänzt 2014).
'
' Bei Zeichen, deren Font durch Feldcode geschützt ist, muss das Zeichen auch ermittelt werden
' können, wenn der Code hexadezimal ist (25. bis 27. November 2014).
'
' Gewähltes Tastenkürzel: AltGr + C.
' *****************
  Dim x() As String  ' x(i) ist das zu analysierende Zeichen
  Dim Y() As String  ' z(i) der Hexcode
  Dim z() As String  ' y(i) der Dezimalcode
  Dim f() As String  ' f(i) die jeweilige Schriftart (Font)
  Dim Rg() As Range  ' Rg(i) ist der Range des jeweiligen Zeichens
  Dim fs As String
  Dim Feldsteuerzeichen As Boolean
  Dim zL As String, zR As String
  Dim Ergebnis1 As String    ' Feldcode-Ansicht
  Dim Ergebnis2 As String    ' Analyse-Ergebnis
  Dim i As Integer
  Dim j As Integer
  Dim N As Integer
```

```
  Dim k1 As Integer
  Dim k2 As Integer
  Dim strFontname
  Dim Zeichennummer1 As Integer  ' bei Sonderschriftarten
  Dim rng0 As Range      ' Rng0 ist der zu analysierende Bereich.
  Dim Rng1 As Range
  Dim Rng2 As Range
  Dim MsgBoxAnzeige As String
  Dim Abfrage As Integer
  Dim Windowssonderschrift() As Boolean
  Dim Pfad1 As String
  Dim Unicodeliste1 As String
  Dim Pos1 As Integer
  Dim Pos2 As Integer
  Dim doc1 As Word.Document  '  die Datei mit den zu analysierenden Zeichen
  Dim doc2 As Word.Document  '  temporäre Zwischendatei
  Dim aTable As Table
  '' Dim B_Zwischendatei As Boolean:   B_Zwischendatei = False
  Dim Unsichtbares_zeigen As Boolean
  Dim Shape_Meldung As String: Shape_Meldung = ""
  Dim B_Leerzeichen As Boolean: B_Leerzeichen = False
  Dim B_Font As Boolean: B_Font = False
'  ***********************************
  ' "Ausgeblendete" Zeichen sichtbar machen, wenn sie unsichtbar sind:
  If ActiveWindow.View.ShowAll = False Then
    Unsichtbares_zeigen = True
    ActiveWindow.View.ShowAll = True
  End If
  '
  ' Wenn der Cursor am Textanfang steht und nichts markiert ist:
  If Selection.Range.End = 0 Then
     MsgBox "Bitte Zeichen markieren, der Cursor steht vor dem Text " _
         & "oder es ist kein Text vorhanden!"
     Exit Sub
  End If
  '
  ' Wenn nichts markiert ist, 1 Zeichen links vom Cursor markieren:
  If Selection.Type < 2 Then
     Set rng0 = Selection.Range
     rng0.SetRange Start:=rng0.End - 1, End:=rng0.End
     rng0.Select
  End If
  If Selection.Type = 8 Then
     MsgBox "Es ist eine frei positionierte Grafik oder deren Ankerpunkt markiert."
     Exit Sub
  End If
  ' ************************
  ' Fehlerbehandlung:
    On Error GoTo Zeile_Abbruch_weil_kein_Schriftzeichen
  ' ************************'
  ' Maximale Länge der Zeichenkette wird begrenzt.
  N = Len(Selection)
  If N > 120 Then
     ' Bei größer 44 Zeilen ist keine Messagebox mehr möglich, aber bei über 100 Zeilen
     ' wird auch die Ausgabe in eine Datei zu unübersichtlich. Theoretisch ist hier keine Grenze.
     MsgBox "Maximal 120 Zeichen markieren! Es sind hier " & N & " Zeichen markiert."
     Exit Sub
  End If
'  ***********************************************************************
'   "Autofokus" wegen kombinierenden diakritischen Zeichen
'    und Zeichen höherer Unicode-Ebenen:
'  ***********************************************************************
  '  a) Markierung um kombinierende diakritische Zeichen erweitern
```

```
'    (Unicode-Zeichen von hex. 0300 bis hex. 036F., bzw. dezimal von 768 bis 879):
' a1) Wenn nur das kombinierende diakritische Zeichen (oder eines von bis zu vier) markiert ist:
For i = 1 To 4
  zL = Hex(AscW(Left(Selection, 1)))
  If Val("&H" & zL & "&") > 767 And Val("&H" & zL & "&") < 880 Then
      Set rng0 = Selection.Range
      Set Rng2 = rng0
      Rng2.SetRange Start:=Rng2.Start - 1, End:=Rng2.End
      zL = Hex(AscW(Left(Rng2, 1)))
   '   Zeichen links davon gehört dazu.
      Rng2.Select
  Else
     Exit For
  End If
Next i
' a2) Wenn rechts von der Markierung eins bis vier kombinierende diakritische Zeichen stehen:
Set rng0 = Selection.Range
Set Rng2 = rng0
For i = 1 To 4
   Rng2.SetRange Start:=Rng2.Start, End:=Rng2.End + 1
   zR = Hex(AscW(Right(Rng2, 1)))
   If Val("&H" & zR & "&") > 767 And Val("&H" & zR & "&") < 880 Then
     Rng2.Select
   Else
     Exit For
   End If
Next i
' b) Wenn Surrogate für Unicode-Zeichen höherer Ebenen vorhanden sind:
' b1) Markierung nach rechts erweitern, wenn nur das linke Surrogat vorhanden ist:
zR = Hex(AscW(Right(Selection, 1)))
If Val("&H" & zR & "&") > 55295 And Val("&H" & zR & "&") < 56320 Then
    Set rng0 = Selection.Range
    Set Rng2 = rng0
    Rng2.SetRange Start:=Rng2.Start, End:=Rng2.End + 1
    ' Nur wenn das neue Zeichen das gesuchte rechte Surrogat ist, Markierung erweitern:
    zR = Hex(AscW(Right(Rng2, 1)))
    If Val("&H" & zR & "&") > 56319 And Val("&H" & zR & "&") < 57344 Then
    Rng2.Select
    End If
 End If
 ' b2) Markierung nach links erweitern, wenn nur das rechte Surrogat vorhanden ist:
 zL = Hex(AscW(Left(Selection, 1)))
 If Val("&H" & zL & "&") > 56319 And Val("&H" & zL & "&") < 57344 Then
    Set rng0 = Selection.Range
    Set Rng2 = rng0
    Rng2.SetRange Start:=Rng2.Start - 1, End:=Rng2.End
    ' Nur wenn das neue Zeichen das gesuchte linke Surrogat ist, Markierung erweitern:
    zL = Hex(AscW(Left(Rng2, 1)))
    If Val("&H" & zL & "&") > 55295 And Val("&H" & zL & "&") < 56320 Then
      Rng2.Select
    End If
 End If
' *******
' Durch "Autofokus" veränderte Länge des markierten Bereichs:
N = Len(Selection)
Set rng0 = Selection.Range
' ****************************************************************
' Wenn die Markierung sichtbaren Feldcode mit den Feldsteuerzeichen enthält,
' Feldcode-Ansicht abschalten.
Feldsteuerzeichen = False
Ergebnis1 = ""      ' Text, der später ggf. den Feldcode enthalten soll.
If InStr(Selection, ChrW(19)) > 0 And InStr(Selection, ChrW(21)) > 0 _
And Selection.Fields.Count > 0 Then
```

```
    "   ' Markierten Bereich einengen auf Feldfunktion ohne Steuerzeichen:
    "   Ergebnis1 = "Die Markierung enthält Feldfunktionen (dezimale Codes 19 und 21, " _
    "      & "auf dem Bildschirm als geschweifte Klammern angezeigt). " _
    "      & "Im Folgenden wird nur die (erste) Feldcode-Anzeige analysiert:" & vbCrLf & vbCrLf
    "   Pos1 = InStr(Selection, ChrW(19))
    "   Pos2 = InStr(Selection, ChrW(21))
    "   Rng0.SetRange Start:=Rng0.Start + Pos1, End:=Rng0.Start + Pos2 - 1
    "   Rng0.Select
      ActiveWindow.View.ShowFieldCodes = False
       N = Len(Selection)
  End If
' ************************************************************************
  ' Bei der Analyse von Einzel-Ranges machen folgende Elemente Probleme:
  '   a) Shapes (Textfelder oder Zeichenobjekte),
  '   b) Tabellensteuerzeichen und
  '   c) Feldfunktionen (Feld-Steuerzeichen).
  ' Eine Beseitigung dieser Elemente in der Originaldatei und Verwendung von "Undo"  ist zu
  ' nach der Analyse riskant. Deshalb erfolgt in einer Zwischendatei ("doc2") die Umwandlung
  ' von Shapes in Frames bzw. deren Löschung, der Umwandlung des markierten Tabellenbereichs
  ' in Text und schließlich die Beseitigung von Feldfunktionen im zu analysierenden Text.
' ************************************************
' Fehlerbehandlung: Beim Befehl "Selection.Copy"
  ' Ergänzung März 2012, wegen Absturz des "Copy-Befehl" bei Tabellenreihen-Endmarke:
  " If Len(Selection) <= 2 Then   ' macht Probleme bei Seitenzahl in Positionsrahmen.
  " If Selection = Chr(13) & Chr(7) Then  ' macht Problem bei Feldfunktion in Tabelle.
  If rng0.Text = Chr(13) & Chr(7) Then
    MsgBox "Es ist nur das Tabellen-Steuerzeichen (Code = 7) markiert."
    Exit Sub
  End If
  '*****************
  On Error Resume Next
  Selection.Copy
  If Err <> 0 Then
    MsgBox "Die Markierung enthält keinen analysierbaren Inhalt!"
    Exit Sub
  End If
  On Error GoTo Zeile_Abbruch_weil_kein_Schriftzeichen
' ************************************************
' In Word 2010 wird die Absatzschriftart unzulänglich übertragen, und
' einzelne Leerzeichen werden nicht mehr vor Absatzmarken eingefügt (2. Juni 2014):
  If Selection.Text = " " Then B_Leerzeichen = True
  If Selection.Font.Name <> "" Then B_Font = True
  Selection.CopyFormat
' ************************************************************************
' Zwischendatei "doc2" erzeugen und dort die zu untersuchende
' Zeichenkette einfügen:
' ************************************************************************
  Set doc1 = ActiveDocument  ' (vorhandene Datei)
  " B_Zwischendatei = True
  Application.ScreenUpdating = False    ' Zwischendatei soll unsichtbar bleiben.
  Set doc2 = Documents.Add  '  DocumentType:=wdNewBlankDocument
  Selection.WholeStory
  Selection.Delete         ' ggf. Briefkopf entfernen
' ************************************************
' In Word 2010 Absatzschriftart und Einzel-Leerzeichen ggf. separat übertragen:
  Selection.WholeStory
  If B_Leerzeichen = True Then
    Selection.TypeText Text:=" "
  Else
    Selection.Paste
  End If
  Selection.WholeStory
  If B_Font = True Then
```

```
    Selection.PasteFormat
  End If
' ********************************************
  Selection.WholeStory
  Selection.Range.HighlightColorIndex = wdNoHighlight
  Selection.Font.Color = wdColorAutomatic
' ********************************************
' Beseitigung von Shapes in der Zwischendatei:
  If doc2.Shapes.Count > 0 Then
     j = ActiveDocument.Shapes.Count
     If j = 0 Then
      '  MsgBox "Die Markierung enthält keine Textfelder oder Zeichenobjekte (Shapes)."
     ElseIf j = 1 Then
       Shape_Meldung = "Die Markierung enthält" _
           & " ein Textfeld oder Zeichenobjekt (Shape)." & vbCrLf & vbCrLf
     Else
       Shape_Meldung = "Die Markierung enthält " & j _
           & " Textfelder oder Zeichenobjekte (Shapes)." & vbCrLf & vbCrLf
     End If
     ' ***
     ' Rückwärts umwandeln bzw. löschen, weil sonst alle anderen Shapes umnummeriert würden:
     ' (was dann zum Absturz des Makros führen würde):
     For i = j To 1 Step -1
       If ActiveDocument.Shapes(i).Type = msoTextBox Then     ' Typ 17
          ' Textfelder (Textbox) umwandeln in Textrahmen(Frame).
          ActiveDocument.Shapes(i).ConvertToFrame
       Else
         ' AutoForm, Zeichnungsobjekt oder sonstige Grafik (Shape).
         ActiveDocument.Shapes(i).Delete
       End If
     Next i
     ' ***
  End If
  ' alle Shapes in der Markierung sind jetzt beseitigt.
  ' ****************************************************************
  ' Tabellenbeseitigung in der Zwischendatei (doc2):
  ' Vor der Umwandlung der Tabellen in Text alle Tabellen gelb markieren:
  For Each aTable In Selection.Tables
     aTable.Range.HighlightColorIndex = wdYellow
  Next aTable
  '
  ' Echte Absatzmarken in Tabellen vorübergehend beseitigen:
  Selection.Find.ClearFormatting
  Selection.Find.Highlight = True
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
     .Text = "^p"
     .Replacement.Text = "<!p!>"
     .Replacement.Font.Color = wdColorRed
     .Wrap = wdFindContinue   '  (ergänzt 29.7.2013)
     .Forward = True
     .Format = True
     .MatchCase = False
     .MatchWholeWord = False
     .MatchWildcards = False
     .MatchSoundsLike = False
     .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceAll
  '
  ' Umwandlung der Tabellen in Text (ggf. mehrere kleine Tabellen):
  For Each aTable In Selection.Tables
      aTable.Rows.ConvertToText Separator:=ChrW(7)  ' Tabellenfeld-Steuerzeichen.
```

```
Next aTable
Selection.WholeStory
'
' Absatzmarken in Tabellenzeilenschluss in doppeltes Tabellenfeldsteuerzeichen umwandeln:
' (dies entspricht dem Quellcode einer Tabelle in Word 2000 bis Word 2003):
Selection.Find.ClearFormatting
Selection.Find.Highlight = True
Selection.Find.Replacement.ClearFormatting
With Selection.Find
  .Text = "^p"
  .Replacement.Text = ChrW(7) & ChrW(7)
  .Forward = True
  .Wrap = wdFindContinue
  .Format = True
  .MatchCase = False
  .MatchWholeWord = False
  .MatchWildcards = False
  .MatchSoundsLike = False
  .MatchAllWordForms = False
End With
Selection.Find.Execute Replace:=wdReplaceAll
'
' Absatzmarken innerhalb Tabellenfeldern wiederherstellen:
Selection.Find.ClearFormatting
Selection.Find.Highlight = True
With Selection.Find
  .Text = "<!p!>"
  .Font.Color = wdColorRed
  .Replacement.Text = "^p"
  .Forward = True
  .Format = True
  .MatchCase = False
  .MatchWholeWord = False
  .MatchWildcards = False
  .MatchSoundsLike = False
  .MatchAllWordForms = False
End With
Selection.Find.Execute Replace:=wdReplaceAll
'
'
' Durch das Umwandeln einer Tabelle in Text änderte sich die Zahl der markierten Zeichen:
N = Len(Selection)
' ********************************************************************************************
' Feldfunktionen bei markierten Feldergebnissen in Zwischendatei beseitigen:
' ********************************************************************************************
' Solange nur Feldergebnis sichtbar, werden Steuerzeichen von Feldern nicht gefunden:
Application.ScreenUpdating = False
Selection.Fields.ToggleShowCodes
If InStr(Selection, ChrW(19)) > 0 And InStr(Selection, ChrW(21)) > 0 _
And Selection.Fields.Count > 0 Then
'  Feldfunktion vorübergehend beseitigen, aber Feldfunktion speichern:
  Feldsteuerzeichen = True
  Ergebnis1 = Selection   ' String mit Feldcode-Ansicht
  """""" 26.11.2012
  Ergebnis1 = Replace(Ergebnis1, ChrW(19), "{")
  Ergebnis1 = Replace(Ergebnis1, ChrW(21), "}")
  """""""""""""""
' ****************************
' Durch Felder geschützte Formelzeichen für Analyse vorübergehend beseitigen (31. März 2011)
' wie zum Beispiel  {SYMBOL 87 \f "Symbol" \s 10} oder {SYMBOL 937 \f "Arial" \s 10 \u}:
  If InStr(Selection, ChrW(19) & "SYMBOL") > 0 _
  Or InStr(Selection, ChrW(19) & " SYMBOL") > 0 Then
    Ergebnis1 = "Die Markierung enthält Feldfunktionen " _
```

```
    & "(dezimale Codes 19 und 21, " _
    & "hier als geschweifte Klammern angezeigt) " _
    & "mit einem oder mehreren geschützten Sonderzeichen;" _
    & vbCrLf & vbCrLf & "Feldcode-Ansicht: " _
    & vbCrLf & Ergebnis1 & vbCrLf & vbCrLf _
    & "Die Zeichenanalyse ergibt:" & vbCrLf
  With Selection.Find
    .ClearFormatting
    .Text = "^19Symbol"
    .Forward = True
    .Wrap = wdFindContinue      ' es wird sonst nur ein Zeichen ersetzt!
    .Format = False
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
    Do While .Execute           ' solange etwas gefunden wird
      If InStr(Selection, "\f ") > 0 Then
        ''''''''''''
        k1 = InStr(Selection, "\f ") + 3     ' Anfang des Schriftarts-Namen
        k2 = InStr(Selection, ChrW(21))      ' Ende des Feldes
        strFontname = Mid(Selection, k1, k2 - k1)
        If InStr(strFontname, "\") > 0 Then _
          strFontname = Mid(strFontname, 1, InStr(strFontname, "\") - 1)
        strFontname = Trim(strFontname)
        '
        If InStr(Selection, "0x") = 0 Then
          ' Dezimalcode:
          Selection.Font.Name = strFontname
          Selection.Text = ChrW(Val(Mid(Selection, InStr(3, Selection, " "))))
        Else
          ' Hexadezimaler Code (muss ohne "0x" eingefügt werden!):
          k1 = InStr(Selection, "0x") + 2 ' Anfang des reinen Hexwertes
          k2 = InStr(Selection, "\")  ' Ende des Zeichencodes
          '  Selection.Text = ChrW(Val("&H" & '((reiner_Hexwert))' & "&"))
          Selection.Text = ChrW(Val("&H" & Mid(Selection, k1, k2 - k1) & "&"))
          Selection.Font.Name = strFontname
        End If
      End If
    Loop
  End With
' Falls zwischen Feldanfang und dem Wort "Symbol" ein Leerzeichen ist:
  With Selection.Find
    .ClearFormatting
    .Text = "^19 Symbol"
    .Forward = True
    .Wrap = wdFindContinue      ' es wird sonst nur ein Zeichen ersetzt!
    .Format = False
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
    Do While .Execute           ' solange etwas gefunden wird
      If InStr(Selection, "\f ") > 0 Then
        k1 = InStr(Selection, "\f ") + 3     ' Anfang des Schriftarts-Namen
        k2 = InStr(Selection, ChrW(21))      ' Ende des Feldes
        strFontname = Mid(Selection, k1, k2 - k1)
        If InStr(strFontname, "\") > 0 Then _
          strFontname = Mid(strFontname, 1, InStr(strFontname, "\") - 1)
        strFontname = Trim(strFontname)
        '
```

```
            If InStr(Selection, "0x") = 0 Then
              ' Dezimalcode:
              Selection.Font.Name = strFontname
              Selection.Text = ChrW(Val(Mid(Selection, InStr(3, Selection, " "))))
            Else
              ' Hexadezimaler Code (muss ohne "0x" eingefügt werden!):
              k1 = InStr(Selection, "0x") + 2 ' Anfang des reinen Hexwertes
              k2 = InStr(Selection, "\")  ' Ende des Zeichencodes
              '  Selection.Text = ChrW(Val("&H" & '((reiner_Hexwert))' & "&"))
              Selection.Text = ChrW(Val("&H" & Mid(Selection, k1, k2 - k1) & "&"))
              Selection.Font.Name = strFontname
            End If
          End If
        Loop
      End With
    Else
       Ergebnis1 = "Die Markierung enthält Feldfunktionen " _
        & "(dezimale Codes 19 und 21, " _
        & "hier als geschweifte Klammern angezeigt); " _
        & vbCrLf & vbCrLf & "Feldcode-Ansicht: " & vbCrLf & Ergebnis1 _
        & vbCrLf & vbCrLf & "Die Zeichenanalyse ergibt:" & vbCrLf
    End If
  ' ***********************************
  '  Markierung in Zwischendatei "doc2" um letzte Absatzmarke verkürzen:
    Selection.WholeStory
    Set rng0 = Selection.Range
    rng0.SetRange Start:=rng0.Start, End:=rng0.End - 1
    rng0.Select
    ' *****************
  Else  '  keine Feld oder keine Feldsteuerzeichen in der Markierung:
    "  rng0.Select  ' macht Ärger bei Tabellenzellen untereinander! 18. 12.2014.
  End If
  Selection.Fields.ToggleShowCodes
  Application.ScreenUpdating = True
  If Feldsteuerzeichen = True Then
     Selection.Fields.Unlink
  End If
  ' ********************************************************************
  ' Bei dem Befehl "Overstrike" funktioniert der obige Unlink-Befehl nicht.
  ' Beseitigung durch Kopieren und als Inhalt wieder einfügen (ergänzt 2. November 2011):
  If Selection.Fields.Count > 0 Then
    Dim F0 As Field
    Dim RngX As Range
    Set RngX = Selection.Range
    For Each F0 In RngX.Fields
      F0.Select
      Selection.Copy
      Selection.PasteSpecial Link:=False, DataType:=wdPasteText, _
        Placement:=wdInLine, DisplayAsIcon:=False
    Next F0
    Selection.WholeStory
    Selection.MoveEnd Unit:=wdCharacter, Count:=-1
  End If
' ********************************************************************
'  Analyse des markierten Bereichs (die eigentliche Zeichenanalyse)
' ********************************************************************
  N = Len(Selection)
  ReDim Rg(N)
  ReDim x(N)
  ReDim Y(N)
  ReDim z(N)
  ReDim f(N)
  ReDim Windowssonderschrift(N)
```

```
'  Bestimmung der Schriftart des markierten Bereichs "f(0)":
f(0) = Selection.Font.Name
' Wenn die Schriftart-Anzeige leer ist, ist sie unterschiedlich:
If f(0) = "" Then
  f(0) = "unterschiedlich"
End If
'
Dim RgN As Range ' Markierung in der Zwischendatei
Set RgN = Selection.Range
'
For i = 1 To N
  Windowssonderschrift(i) = False
  Set Rg(i) = Selection.Range
  "  Rg(i).SetRange Start:=Rg(i).Start + i - 1, End:=Rg(i).Start + i
  Rg(i).SetRange Start:=RgN.Start + i - 1, End:=RgN.Start + i
  "Rg(i).Select
  f(i) = Rg(i).Font.Name
  x(i) = Rg(i).Text
          '  y(i) = AscW(x(i))  ' Ergibt negative Zahlen bei x(i) > 32'767!
          '  z(i) = Hex(y(i))
  ' Da bei VBA das Integerformat nur bis + 32767 reicht, muss durch einen Umweg
  ' für den Dezimalcode (y) der Typ "Long" erzwungen werden.
  z(i) = Hex(AscW(x(i)))
  Y(i) = Val("&H" & z(i) & "&")
  ' Der Hexadezimale Unicode wird in der Regel vierstellig angegeben:
  Select Case Len(z(i))
    Case Is = 3
      z(i) = "0" & z(i)
    Case Is = 2
      z(i) = "00" & z(i)
    Case Is = 1
      z(i) = "000" & z(i)
  End Select
  ' *************
  ' Fallunterscheidung:
  ' Steuerzeichen sollen im Ergebnis erklärt, aber nicht dargestellt werden:
  If Y(i) <= 32 Then
     Select Case Y(i)
     Case 32
        x(i) = """Leerzeichen"""
     Case 31
       x(i) = """Bedingter Trennstrich"""
     Case 30
       x(i) = """Geschützter Bindestrich"""
     Case 13
       x(i) = """Absatzzeichen"""
     Case 11
       x(i) = """manueller Zeilenumbruch"""
     Case 12
       x(i) = """Abschnittswechsel/Seitenumbruch"""
     Case 14
       x(i) = """Spaltenumbruch"""
     Case 9
       x(i) = """Tabulator"""
     Case 1
       x(i) = """Steuerzeichen für Bild oder Formel-Editor-Objekt"""
     Case 7
       x(i) = """Tabellenfeld-Steuerzeichen"""
     Case Else
       x(i) = """Steuerzeichen"""
     End Select
   ' Bei Windows-Sonderschriften gelten völlig andere Regeln:
   ElseIf x(i) = "(" Or x(i) Like "[" & ChrW(61472) & "-" & ChrW(61695) & "]" Then
```

```
        Rg(i).Select
        With Dialogs(wdDialogInsertSymbol)
           fs = .Font
           Zeichennummer1 = .CharNum
        End With
        "   If fs <> "(normaler Text)" And fs <> "" Then
        "   If Zeichennummer1 < 0 Then
        If fs <> "(normaler Text)" And fs <> "" _
        Or Zeichennummer1 <> 40 And x(i) = "(" Then
           ' Or Zeichennummer1 < 0 And x(i) = "(" Then
           Windowssonderschrift(i) = True
           If fs = "(normaler Text)" Or fs = "" Then fs = "???"
           f(i) = fs
           If Zeichennummer1 < 0 Then
              Y(i) = (Zeichennummer1 + 4096 + 61440)
              x(i) = ChrW(Zeichennummer1 + 4096 + 61440)
            Else
              Y(i) = Zeichennummer1 + 61440
              x(i) = ChrW(Zeichennummer1 + 61440)
           End If
           z(i) = Hex(Y(i))
        End If
      "  rng0.Select  ' führte zur Markierung der Originaldatei, 19.12.2014.
     ' Fehlerhafte Zuweisung einer Windows-Sonderschrift:
     ElseIf f(i) = "Symbol" Or f(i) Like ("Wingding*") Or f(i) = "Webdings" Then
        f(i) = f(i) & "???"
        x(i) = "???"
     ' einige weitere nicht-Unicode-kompatible Sonderschriftarten:
     ElseIf f(i) = "BSsymb9" Or f(i) = "BeuthPi2" Then
        x(i) = "Sonderschriftzeichen"
     End If
     ' *************
     If f(i) <> f(0) Then f(0) = "unterschiedlich"
  Next i
' ****************************************
' Zwischendatei "doc2" wieder schließen:
' ****************************************
' Bei Tabellenanalyse in separater Datei muss diese geschlossen werden.
"    If B_Zwischendatei = True Then
       doc2.Close (wdDoNotSaveChanges)
       doc1.Activate
       Application.ScreenUpdating = True
"   End If
' ************************************************
' Ergebniszusammenstellung für Messagebox:
' ************************************************
  Ergebnis2 = Shape_Meldung & Ergebnis1 & "Schriftart: " & f(0)
  For j = 1 To N
    If Y(j) > 55295 And Y(j) < 57344 Then
    '  Zeichen höherer Unicode-Ebene:
      If N > j And Y(j) > 55295 And Y(j) < 56320 Then
        If Y(j + 1) > 56319 And Y(j + 1) < 57344 Then
          ' Wenn ja, liegt Zeichen höherer Unicode-Ebene vor:
          ' Berechnung des Unicodes höherer Ebenen nach ISO 10646:2003, Anhang C:
          Y(0) = ((Y(j) - 55296) * 1024) + (Y(j + 1) - 56320) + 65536
          z(0) = Hex(Y(0))
          '  z(0) ist der Unicode des gesuchten Zeichen höherer Unicode-Ebene.
          Ergebnis2 = Ergebnis2 & "," & vbCrLf _
            & "der Unicode des Zeichens höherer Ebene ist hexadezimal: " & z(0) _
            & " und dezimal: " & Y(0)
          j = j + 1
        Else
          Ergebnis2 = Ergebnis2 & ", " & vbCrLf & "der Unicode von  " & x(j) _
```

```
            & "  ist hexadezimal: " & z(j) & "  und dezimal: " & Y(j) _
            & vbCrLf & "     (unvollständiges Surrogat)"
        End If
      Else
        Ergebnis2 = Ergebnis2 & ", " & vbCrLf & "der Unicode von  " & x(j) _
          & "  ist hexadezimal: " & z(j) & "  und dezimal: " & Y(j) _
          & vbCrLf & "     (unvollständiges Surrogat)"
      End If
    ElseIf Windowssonderschrift(j) = True Then
    ' Windows-Sonderzeichen:
      Ergebnis2 = Ergebnis2 & ", " & vbCrLf & "der Code von  " & x(j) _
        & "  ist hexadezimal: " & Hex(Y(j) - 61440) & "/" & Hex(Y(j)) & "  und dezimal: " _
        & ((Y(j) - 61440)) & "/" & Y(j) & ", Schriftart: " & f(j)
      If f(j) = "Symbol" Then
        Ergebnis2 = Ergebnis2 & "," & vbCrLf _
          & "     Bedeutung: " & Symbolname(Hex(Y(j) - 61440))
      ElseIf f(j) = "Wingdings" Then
         Ergebnis2 = Ergebnis2 & "," & vbCrLf _
          & "     Bedeutung: " & Wingdingsname(Hex(Y(j) - 61440))
      End If
    Else
    ' normales Unicode-Zeichen:
      ' Bei Dezimalcodes bis 255 wird bei Windows eine "0" davorgesetzt:
      If Y(j) < 256 Then
        Y(j) = "0" & Y(j)
      End If
      If Y(j) = 160 Then x(j) = """Festes Leerzeichen"""
      Ergebnis2 = Ergebnis2 & ", " & vbCrLf & "der Unicode von  " & x(j) _
        & "  ist hexadezimal: " & z(j) & "  und dezimal: " & Y(j)
    End If
  Next j
  Ergebnis2 = Ergebnis2 & "."
  '
' ********************************************************************
' Nur wenn nicht mehr als 40 Zeichen, Ergebnis als Messagebox anzeigen,
' ansonsten direkt in die Ausgabe als Datei ( geändert 6. Oktober 2012)
' ********************************************************************
' Beim Analyse-Ergebnis kann eine MessageBoxW maximal 44 Zeilen anzeigen,
' eine normale MsgBox sogar nur 16 Zeilen:
  If N < 40 Then
    ' Ergebnisanzeige in Messagebox mit Abfrage, ob Ergebnis in separater Datei gewünscht:
    MsgBoxAnzeige = Ergebnis2 & vbCrLf & vbCrLf _
          & "Wollen Sie das Ergebnis ausführlicher in einer separaten Datei?"
    Abfrage = MessageBoxW(0, StrPtr(MsgBoxAnzeige), StrPtr("Zeichenanalyse"), _
          vbYesNoCancel + vbDefaultButton2)
    '   Alternative einfache Funktion, wenn keine Windows-Umgebung, leider ohne Unicode:
    '   Abfrage = MsgBox(MsgBoxAnzeige, vbYesNoCancel + vbDefaultButton2, "Zeichenanalyse")
    ' Bei "Ja" kommt "6" heraus, bei "Nein" kommt "7" heraus, bei Abbrechen eine "2".
    '
    If Abfrage <> 6 Then
      ' Speicher löschen und Programm beenden.
      GoTo Zeile_Suchverzeichnis_löschen_und_beenden
    End If
"  Else
"    MsgBox "Da insgesamt " & N & " Zeichen markiert sind, erfolgt das Ergebnis " _
"       & "in einer separaten Datei. Es passt nicht mehr in eine Messagbox."
  End If
' ********************************************************************
' Wenn mehr als 40 Zeichen oder Abfrage = "ja" (Abfrage = 6),
' ausführlichere Ergebnisanzeige in Extra-Datei:
' ********************************************************************
  Selection.Copy
  ' **********
```

```
  ' Zeitmessen für die Makro-Dauer
  Dim Zeit1, Zeit2, Zeitdauer
  Zeit1 = Timer
  ' ***********
  ' Zur Bestimmung des Unicode-Namens Datei mit der Unicode-Liste öffnen:
  '   Das Makro braucht zum Nachschlagen des Unicode-Zeichennamens
  '   im Internet unter "http://unicode.org/Public/UNIDATA/UnicodeData.txt"
  '   (zumindest beim ersten Mal) mindestens 6 Sekunden.  Von daher ist es
  '   besser, diese Text-Datei (von Hand) vorher auf der Festplatte zu speichern.
  '   Dann beträgt die Dauer weniger als 1,2 Sekunden.
  '   Auf ein automatisches Speichern der Datei auf der Festplatte per Makro
  '   habe ich bewusst verzichtet.
  '   Bevorzugt wird die Kompakt-Version, in der Makro und Text-Liste in einer
  '   einzigen Dokumentvorlage kombiniert sind, die ich zum Download in meiner
  '   Homepage anbiete:
  ' ************
  Application.StatusBar = "Bitte warten - das Makro läuft."
  Application.ScreenUpdating = False
' ************************************
' Die ggf. gespeicherte Datei mit der Unicode-Liste suchen:
  '  kompakte Datei mit Unicode-Liste suchen:
  If Dir(Options.DefaultFilePath(wdUserTemplatesPath) & "\Zeichen-Analyse-kompakt.dot") <> "" Then
    Unicodeliste1 = Options.DefaultFilePath(wdUserTemplatesPath) & "\Zeichen-Analyse-
        kompakt.dot"
  ElseIf Dir(Options.DefaultFilePath(wdStartupPath) & "\Zeichen-Analyse-kompakt.dot") <> "" Then
    Unicodeliste1 = Options.DefaultFilePath(wdStartupPath) & "\Zeichen-Analyse-kompakt.dot"
  '  alternativ die Original-Textdatei suchen:
  ElseIf Dir(Options.DefaultFilePath(wdUserTemplatesPath) & "\UnicodeData.txt") <> "" Then
    Unicodeliste1 = Options.DefaultFilePath(wdUserTemplatesPath) & "\UnicodeData.txt"
  ElseIf Dir(Options.DefaultFilePath(wdStartupPath) & "\UnicodeData.txt") <> "" Then
    Unicodeliste1 = Options.DefaultFilePath(wdStartupPath) & "\UnicodeData.txt"
  ' Nicht auf der Festplatte, deshalb im Internet nachsehen:
  Else
    Unicodeliste1 = "http://unicode.org/Public/UNIDATA/UnicodeData.txt"
  End If
' ****************************************
' Bei fehlendem Internetzugang gibt es eine Fehlermeldung:
On Error GoTo Zeile_Internetzugang_fehlt
'''''''''''''''''''''''''
  ' Unicodeliste1 öffnen:
  Documents.Open FileName:=Unicodeliste1, _
      ConfirmConversions:=False, ReadOnly:=True, AddToRecentFiles:=False, _
      PasswordDocument:="", PasswordTemplate:="", Revert:=False, _
      WritePasswordDocument:="", WritePasswordTemplate:="", _
      Format:=wdOpenFormatAuto, Encoding:=1252
On Error GoTo 0
  '  Zum Beschleunigung beim Suchen in der Datei "Unicodeliste1":
  ActiveWindow.View.Type = wdNormalView
' ************************************************************
' Ausführlichere Ergebniszusammenstellung für Ausgabe in Datei:
' ************************************************************
  Ergebnis2 = Shape_Meldung & Ergebnis1
  For j = 1 To N
    If Y(j) > 55295 And Y(j) < 57344 Then
    '  Zeichen höherer Unicode-Ebene
      If N > j And Y(j) > 55295 And Y(j) < 56320 Then
        If Y(j + 1) > 56319 And Y(j + 1) < 57344 Then
          ' Wenn ja, liegt Zeichen höherer Unicode-Ebene vor:
          ' Berechnung des Unicodes höherer Ebenen nach ISO 10646:2003, Anhang C:
          Y(0) = ((Y(j) - 55296) * 1024) + (Y(j + 1) - 56320) + 65536
          z(0) = Hex(Y(0))
          '  z(0) ist der Unicode des gesuchten Zeichen höherer Unicode-Ebene.
          x(j) = x(j) & x(j + 1)
```

```
            Ergebnis2 = Ergebnis2 _
              & "Der Unicode des Zeichens höherer Ebene  x(" & j _
              & ")  ist hexadezimal: " & z(0) & " und dezimal: " & Y(0) & "," & Chr(11) _
              & "(der Unicode des Surrogat 1 ist hexadezimal: " & z(j) _
              & " und dezimal: " & Y(j) & ", " & Chr(11) _
              & "der Unicode des Surrogat 2 ist hexadezimal: " & z(j + 1) _
              & " und dezimal: " & Y(j + 1) & "), " & Chr(11) _
              & EnglischName(z(0)) & ", " & Chr(11) _
              & "die Schriftart ist " & f(j) & ". " & vbCrLf
            j = j + 1
          Else
            Ergebnis2 = Ergebnis2 & "der Unicode von  x(" & j _
              & ")  ist hexadezimal: " & z(j) & "  und dezimal: " & Y(j) _
              & Chr(11) & "(unvollständiges Surrogat)" & ". " & vbCrLf
          End If
        Else
          Ergebnis2 = Ergebnis2 & "Der Unicode von  x(" & j _
             & ") ist hexadezimal: " & z(j) & "  und dezimal: " & Y(j) _
             & Chr(11) & "(unvollständiges Surrogat)" & ". " & vbCrLf
        End If
      ElseIf Windowssonderschrift(j) = True Then
      '  Windowssonderschriftzeichen:
        Ergebnis2 = Ergebnis2 & "Der Code von  x(" & j _
          & ")  ist hexadezimal: " & Hex(Y(j) - 61440) & "/" & Hex(Y(j)) & "  und dezimal: " _
          & ((Y(j) - 61440)) & "/" & Y(j) & ", Schriftart: " & f(j)
      '    If f(j) = "Symbol" Then
          If f(j) = "Symbol" Or f(j) = "SymbolPS" Then
            Ergebnis2 = Ergebnis2 & "," & vbCrLf _
                & "     Bedeutung: " & Symbolname(Hex(Y(j) - 61440)) & ". " & vbCrLf
          ElseIf f(j) = "Wingdings" Then                    '  (im März 2011 neu ergänzt)
            Ergebnis2 = Ergebnis2 & "," & vbCrLf _
                & "     Bedeutung: " & Wingdingsname(Hex(Y(j) - 61440)) & ". " & vbCrLf
          Else
            Ergebnis2 = Ergebnis2 & ". " & vbCrLf
          End If
      ElseIf f(j) = "BSsymb9" Or f(j) = "BeuthPi2" Then
      '  Sonderschrift-Zeichen (hier eventuell weitere Sonderschriftarten aufnehmen):
        Ergebnis2 = Ergebnis2 & "Der Code des Sonderzeichens " _
          & "ist hexadezimal: " & z(j) & "  und dezimal: " & Y(j) & ", " _
          & "Schriftart:" & Chr(160) & f(j) & ". " & vbCrLf
      ElseIf Y(j) < 32 Then
      '  Steuerzeichen (ohne englischen Zeichennamen und ohne Schriftart):
      Ergebnis2 = Ergebnis2 & "Der Unicode von  x(" & j _
          & ")  ist hexadezimal: " & z(j) & "  und dezimal: " & Y(j) & ". " & vbCrLf
      Else
      '  normales Unicode-Zeichen:
        Ergebnis2 = Ergebnis2 & "Der Unicode von  x(" & j _
          & ")  ist hexadezimal: " & z(j) & "  und dezimal: " & Y(j) & ", " _
          & "Schriftart:" & Chr(160) & f(j) & ",  " & EnglischName(z(j)) & ". " & vbCrLf
      End If
    Next j
    ' ****************************************
    ' Unicode-Liste schließen, wenn sie nicht das zu untersuchende Dokument ist:
    If ActiveDocument <> doc1 Then
      ActiveDocument.Close SaveChanges:=False
    Else
      ActiveWindow.ActivePane.View.Type = wdPrintView
    End If
    ' ****************************************
    ' Anzeige in separater Datei:
    ' ****************************************
    Documents.Add DocumentType:=wdNewBlankDocument
    ActiveWindow.View.Type = wdPrintView
```

```
' Falls in der Normal.dot schon ein Briefkopf ist mit Name und Datum,
' soll dieser ersetzt werden:
Selection.WholeStory
' Absatzabstand auf Null stellen:
With Selection.ParagraphFormat
  .Alignment = wdAlignParagraphLeft
  .LeftIndent = CentimetersToPoints(0.5)
  .FirstLineIndent = CentimetersToPoints(-0.5)
  .SpaceBefore = 0
  .SpaceAfter = 0
  .SpaceAfterAuto = False
  .Alignment = wdAlignParagraphLeft
End With
Selection.Font.Size = 11
'
If Val(Y(1)) < 30 And N < 3 Then
  Selection.TypeText Text:="Das markierte Zeichen bzw. die Zeichenfolge setzt sich " _
                    & "wie folgt zusammen:" & vbCrLf
  Selection.TypeText Text:=Ergebnis2
Else
   Selection.TypeText Text:="Das markierte Zeichen bzw. die Zeichenfolge "
   ' Das Zeichen muss mitsamt seiner Schriftart eingefügt werden.
   ' Und danach muss wieder die Ausgangsschrift usw. wiederhergestellt werden.
   ' Dies gelang mir über den Umweg des nachträglichen Einfügens:
   Set Rng1 = Selection.Range
   Selection.TypeText Text:=vbCrLf & "setzt sich wie folgt zusammen:" & vbCrLf & vbCrLf
   Selection.TypeText Text:=Ergebnis2
   Set Rng2 = Selection.Range
   Rng1.Select
   ' """""""""""""""""
   ' Ergänzung 23.11. 2012 wegen Probleme in Word 2010:
   "   Selection.Collapse Direction:=wdCollapseEnd
"    Dim Start1 As Integer
"    Start1 = Rng1.Start
   """""""""""""""""
   '  Zu untersuchenden Text in einer neuen Zeile angeben:
   Selection.TypeParagraph
   Selection.Paste  ' Hier den zu analysierenden Text einfügen.
   """""""""""""""""
    Rng1.SetRange Start:=Rng1.Start, End:=Selection.End
"    Rng1.SetRange Start:=Start1, End:=Selection.End
   """""""""""""""""
   Rng1.Select
   Selection.Range.HighlightColorIndex = wdYellow
   ' *********
   ' Alle Textmarken in der Markierung auflisten (25.11.2012).
   Dim TM As Bookmark
   Dim Textmarkenliste As String
   Textmarkenliste = "Die Markierung enthält folgende Textmarken (""Bookmarks"")" & vbCrLf _
             & "(der Text hinter dem Komma ist ggf. die zugehörige Stelle):" & vbCrLf
   If Selection.Bookmarks.Count > 0 Then
     For Each TM In Selection.Bookmarks
       Textmarkenliste = Textmarkenliste & "     " & TM.Name & ",  " & TM.Range.Text & vbCrLf
     Next
   Else
     Textmarkenliste = "Die Markierung enthält keine Textmarken." & vbCrLf
   End If
   ' *********
   ' Bei Seitenumbrüchen oder Spaltenumbrüchen auf "Normalansicht" umschalten (5.10.2012):
   If InStr(Selection, ChrW(12)) > 0 Or InStr(Selection, ChrW(14)) > 0 Then
      ActiveWindow.View.Type = wdNormalView
   End If
   ' *********
```

```
    ' Wenn in "Ergebnis" Hyperlinks, diese aktivieren:
    Selection.HomeKey Unit:=wdStory
    Selection.Find.ClearFormatting
    With Selection.Find
      .Text = "http://www.unicode.org/charts/PDF/U*.pdf"
      .Replacement.Text = ""
      .Forward = True
      '  .Wrap = wdFindContinue   (führt zu Endlosschleife!)
      .Format = True
      .MatchCase = False
      .MatchWholeWord = False
      .MatchAllWordForms = False
      .MatchSoundsLike = False
      .MatchWildcards = True
      Do While .Execute
        ActiveDocument.Hyperlinks.Add Anchor:=Selection.Range, _
          Address:=Selection.Range.Text, _
          SubAddress:="", ScreenTip:="", TextToDisplay:=Rng2.Text
        Selection.Collapse Direction:=wdCollapseEnd
      Loop
    End With
    Selection.EndKey Unit:=wdStory
  End If
' ***************
' Die Original-Zeichen sollen in der Auswertung mitsamt ihrer Schriftart dargestellt und gelb
' markiert werden, außer wenn sie bereits durch eine Beschreibung ersetzt worden sind:
  For j = 1 To N
    Selection.HomeKey Unit:=wdStory
    Selection.Find.ClearFormatting
    Selection.Find.Replacement.ClearFormatting
    With Selection.Find
      .Text = "x(" & j & ")"
      .Forward = True
      .Format = True
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
     Do While .Execute
       If Len(x(j)) < 3 Then
         Selection.Font.Name = f(j)
         Selection.Range.HighlightColorIndex = wdYellow
         Selection.Font.Bold = True
         Selection.TypeText Text:=x(j)
       Else
         Selection.Font.Name = "Times New Roman"
         Selection.Range.HighlightColorIndex = wdYellow
         Selection.Font.Bold = True
         Selection.Text = x(j)
       End If
     Loop
    End With
  Next j
  ' ************************
  Selection.EndKey Unit:=wdStory
  Selection.TypeParagraph
  '
  Selection.TypeText Text:= _
      "Die englischen Namen aller Unicode-Schriftzeichen sind in ISO/IEC 10646 genormt " _
      & " und weitgehend in "
  ActiveDocument.Hyperlinks.Add Anchor:=Selection.Range, _
      Address:="http://unicode.org/Public/UNIDATA/UnicodeData.txt", SubAddress:="", _
```

```
        ScreenTip:="", TextToDisplay:="http://unicode.org/Public/UNIDATA/UnicodeData.txt"
    Selection.TypeText Text:=Chr(11) & "oder über "
    ActiveDocument.Hyperlinks.Add Anchor:=Selection.Range, _
        Address:="http://www.unicode.org/charts", SubAddress:="", _
        ScreenTip:="", TextToDisplay:="http://www.unicode.org/charts"
    Selection.TypeText Text:=" zu finden."
    Selection.TypeParagraph
    ' Ergänzung 03.04.2011
    Selection.Find.ClearFormatting
    Selection.Find.Replacement.ClearFormatting
    With Selection.Find
      .Text = "Times New Roman"
      .Replacement.Text = "Times^sNew^sRoman"
      .Forward = True
      .Wrap = wdFindContinue
      .Format = False
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
    End With
    Selection.Find.Execute Replace:=wdReplaceAll
    ' Ergänzung 4.10.2012
     Options.DefaultHighlightColorIndex = wdRed
    '
    Selection.Find.ClearFormatting
    Selection.Find.Replacement.ClearFormatting
    Selection.Find.Replacement.Highlight = True
    With Selection.Find
      .Text = "???"
      .Replacement.Text = ""
      .Forward = True
      .Wrap = wdFindContinue
      .Format = True
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
    End With
    Selection.Find.Execute Replace:=wdReplaceAll
    Options.DefaultHighlightColorIndex = wdYellow
' ************************
' Am Schluss die Textmarken auflisten:
    Selection.EndKey Unit:=wdStory
    Selection.TypeParagraph
    Selection.TypeText Text:=Textmarkenliste
' ************************
    Call Ueberschriften_fett
' ************************
' Bildschirm wieder anstellen:
    Application.ScreenUpdating = True
' ****************************************************
    ' Zeitmessen für die Makro-Dauer auf drei Stellen hinter dem Komma
      Zeit2 = Timer
      Zeitdauer = Format((Zeit2 - Zeit1), "##,##0.000")
      Selection.TypeParagraph
      Selection.TypeText Text:="Die Rechenzeit war " & Zeitdauer & " Sekunden."
      Selection.HomeKey Unit:=wdStory  ' An den Anfang des Ergebnisses gehen.
' ****************************************************
' Zwischenspeicher aufräumen und alles zurückstellen:
Zeile_Suchverzeichnis_löschen_und_beenden:
```

```
' ***************************************************
' Im Suchen/Ersetzen-Menü alles zurückstellen
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  With Selection.Find
     .Text = ""
     .Replacement.Text = ""
     .Forward = True
     .Wrap = wdFindContinue
     .Format = False
     .MatchCase = False
     .MatchWholeWord = False
     .MatchWildcards = False
     .MatchSoundsLike = False
     .MatchAllWordForms = False
  End With
  ' Ansicht zurückstellen:
  If Unsichtbares_zeigen = True Then
    ActiveWindow.View.ShowAll = False
  End If
Exit Sub
' ***************************************************
' ***************************************************
Zeile_Abbruch_weil_kein_Schriftzeichen:
  MsgBox "Die Markierung enthält Elemente, die keine Schriftzeichen sind." _
     & vbCrLf & "Bitte ändern Sie zur Schriftzeichen-Analyse den Umfang der Markierung!"
Exit Sub
' ***************************************************
Zeile_Internetzugang_fehlt:
  MsgBox "Bitte aus dem Internet die Datei " _
    & """http://unicode.org/Public/UNIDATA/UnicodeData.txt""" _
    & " auf der Festplatte " & vbCrLf & "unter """ & Pfad1 & """ speichern" _
    & " oder wenigstens einen Internet-Zugang herstellen."
End Sub
' ---------------------------------------------------------------------------------------------------------


Private Sub Ueberschriften_fett()
' 28. März 2014.
' Ergebnis-Zwischenüberschriften fett machen:
  Selection.Find.ClearFormatting
  Selection.Find.Replacement.ClearFormatting
  Selection.Find.Replacement.Font.Bold = True
  With Selection.Find
    .Text = "Feldcode-Ansicht:"
    .Replacement.Text = ""
    .Replacement.Font.Size = 12
    .Forward = True
    .Wrap = wdFindContinue
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceOne
  '
  With Selection.Find
    .Text = "Die Zeichenanalyse ergibt:"
    .Replacement.Text = ""
    .Replacement.Font.Size = 12
    .Forward = True
```

```
    .Wrap = wdFindContinue
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceOne
  '
  With Selection.Find
    .Text = "Die Markierung enthält folgende Textmarken"
    .Replacement.Text = "Die Markierung enthält folgende Textmarken"
    .Replacement.Font.Size = 12
    .Forward = True
    .Wrap = wdFindContinue
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceOne
  Selection.EndKey Unit:=wdStory
End Sub
' ------------------------------------------------------------------------------------------------------


Private Function EnglischName(Z1 As String) As String
'  Makro erstellt von Dipl.-Ing. Hanna-Chris Gast am 20. Juli 2010, Stand 3. November 2010.
'  Diese Funktion sucht  aus der bereits downgeloadeten und geöffneten Unicode-Liste
'     http://unicode.org/Public/UNIDATA/UnicodeData.txt  den englischen
'     Zeichennamen heraus.
  Dim Rng1, Rng2, y1
  '
  ActiveDocument.Range(0, 0).Select
  '
  Selection.Find.ClearFormatting
  With Selection.Find
      .Text = "^p" & Z1 & ";"
      .Replacement.Text = ""
      .Forward = True
      .Wrap = wdFindContinue
      .Format = False
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
  End With
  Selection.Find.Execute
  '
  If Selection.Find.Found = True Then
     ' Zeile ist gefunden, jetzt Resultat einlesen:
     Selection.Collapse Direction:=wdCollapseEnd
     Set Rng1 = Selection.Range
     '  bis zum nächsten Semikolon suchen
     Selection.Find.ClearFormatting
     With Selection.Find
         .Text = ";"
         .Replacement.Text = ""
         .Forward = True
```

```
        .Wrap = wdFindContinue
        .Format = False
        .MatchCase = False
        .MatchWholeWord = False
        .MatchWildcards = False
        .MatchSoundsLike = False
        .MatchAllWordForms = False
      End With
      Selection.Find.Execute
      Set Rng2 = Selection.Range
      Rng2.SetRange Start:=Rng1.Start, End:=Rng2.End - 1
      Rng2.Select
    ' Gefundene Erklärung einlesen:
      EnglischName = "der" & Chr(160) & "genormte" & Chr(160) & "englische" _
            & Chr(160) & "Zeichenname" & Chr(160) & "ist:" & Chr(160) & Selection
  Else
    y1 = Val("&H" & Z1 & "&")
    Select Case y1
      Case Val(&H3400&) To Val(&H4DBF&)
        EnglischName = "das" & Chr(160) & "Zeichen gehört zu ""CJK Ideograph Extension A""," _
              & Chr(11) & "siehe http://www.unicode.org/charts/PDF/U3400.pdf"
      Case Val(&H4E00&) To Val(&H9FCB&)
        EnglischName = "das" & Chr(160) & "Zeichen ist ein ""CJK-Ideograph""," & Chr(11) _
              & "siehe http://www.unicode.org/charts/PDF/U4E00.pdf"
      Case Val(&HAC00&) To Val(&HD7AF&)
        EnglischName = "das" & Chr(160) & "Zeichen ist eine ""Hangul Syllable"", " & Chr(11) _
              & "siehe http://www.unicode.org/charts/PDF/UAC00.pdf"
      Case 57344 To 63743
        EnglischName = "das" & Chr(160) & "Zeichen ist nicht genormt und gehört zur " _
              & """Privat-Use-Area"""
      Case Val(&H20000) To Val(&H2A6DF & "&")
        EnglischName = "das" & Chr(160) & "Zeichen gehört zu ""CJK Ideograph Extension B""," _
              & Chr(11) & "siehe  http://www.unicode.org/charts/PDF/U20000.pdf"
      Case Val(&H2A700) To Val(&H2B734 & "&")
        EnglischName = "das" & Chr(160) & "Zeichen gehört zu ""CJK Ideograph Extension C""," _
              & Chr(11) & "siehe http://www.unicode.org/charts/PDF/U2A700.pdf"
      Case Else
        EnglischName = "das" & Chr(160) & "Zeichen ist nicht in der " _
              & "Unicode-Liste enthalten"
    End Select
  End If
End Function
' ---------------------------------------------------------------------------------------------------

Private Function Symbolname(Z1 As String) As String
'  Makro erstellt  am 30. August  2010 von Dipl.-Ing. Hanna-Chris Gast, Stand 12. März 2011.
'  Diese Funktion gibt die (englischen) Unicode-Namen und entsprechende Unicodes an,
'  wenn der (niedrige) hexadezimale Symbol-Code eingegeben wird
'  Quellen "http://unicode.org/Public/MAPPINGS/VENDORS/APPLE/SYMBOL.TXT" und andere
'  Ein paar Fehler habe ich hierbei beseitigt, zum Beispiel: Kleines Phi und Phi-Symbol
'  waren vertauscht, und in einigen Fällen habe ich Ergänzungen vorgenommen.
'
  Select Case Z1
    Case Is = "20"
          Symbolname = "NO-BREAK SPACE (Unicode = hex. 00A0)"
    Case Is = "21"
          Symbolname = "EXCLAMATION MARK (Unicode = hex. 0021)"
    Case Is = "22"
          Symbolname = "FOR ALL (Unicode = hex. 2200)"
    Case Is = "23"
          Symbolname = "NUMBER SIGN (Unicode = hex. 0023)"
    Case Is = "24"
```

```
        Symbolname = "THERE EXISTS (Unicode = hex. 2203)"
Case Is = "25"
        Symbolname = "PERCENT SIGN (Unicode = hex. 0025)"
Case Is = "26"
        Symbolname = "AMPERSAND (Unicode = hex. 0026)"
Case Is = "27"
        Symbolname = "CONTAINS AS MEMBER (Unicode = hex. 220B)"
Case Is = "28"
        Symbolname = "LEFT PARENTHESIS (Unicode = hex. 0028)"
Case Is = "29"
        Symbolname = "RIGHT PARENTHESIS (Unicode = hex. 0029)"
Case Is = "2A"
        Symbolname = "ASTERISK OPERATOR (Unicode = hex. 2217)"
Case Is = "2B"
        Symbolname = "PLUS SIGN (Unicode = hex. 002B)"
Case Is = "2C"
        Symbolname = "COMMA (Unicode = hex. 002C)"
Case Is = "2D"
        Symbolname = "MINUS SIGN (Unicode = hex. 2212)"
Case Is = "2E"
        Symbolname = "FULL STOP (Unicode = hex. 002E)"
Case Is = "2F"
        Symbolname = "SOLIDUS (Unicode = hex. 002F)"
Case Is = "30"
        Symbolname = "DIGIT ZERO (Unicode = hex. 0030)"
Case Is = "31"
        Symbolname = "DIGIT ONE (Unicode = hex. 0031)"
Case Is = "32"
        Symbolname = "DIGIT TWO (Unicode = hex. 0032)"
Case Is = "33"
        Symbolname = "DIGIT THREE (Unicode = hex. 0033)"
Case Is = "34"
        Symbolname = "DIGIT FOUR (Unicode = hex. 0034)"
Case Is = "35"
        Symbolname = "DIGIT FIVE (Unicode = hex. 0035)"
Case Is = "36"
        Symbolname = "DIGIT SIX (Unicode = hex. 0036)"
Case Is = "37"
        Symbolname = "DIGIT SEVEN (Unicode = hex. 0037)"
Case Is = "38"
        Symbolname = "DIGIT EIGHT (Unicode = hex. 0038)"
Case Is = "39"
        Symbolname = "DIGIT NINE (Unicode = hex. 0039)"
Case Is = "3A"
        Symbolname = "COLON (Unicode = hex. 003A)"
Case Is = "3B"
        Symbolname = "SEMICOLON (Unicode = hex. 003B)"
Case Is = "3C"
        Symbolname = "LESS-THAN SIGN (Unicode = hex. 003C)"
Case Is = "3D"
        Symbolname = "EQUALS SIGN (Unicode = hex. 003D)"
Case Is = "3E"
        Symbolname = "GREATER-THAN SIGN (Unicode = hex. 003E)"
Case Is = "3F"
        Symbolname = "QUESTION MARK (Unicode = hex. 003F)"
Case Is = "40"
        Symbolname = "APPROXIMATELY EQUAL TO (Unicode = hex. 2245)"
Case Is = "41"
        Symbolname = "GREEK CAPITAL LETTER ALPHA (Unicode = hex. 0391)"
Case Is = "42"
        Symbolname = "GREEK CAPITAL LETTER BETA (Unicode = hex. 0392)"
Case Is = "43"
        Symbolname = "GREEK CAPITAL LETTER CHI (Unicode = hex. 03A7)"
```

```
Case Is = "44"
        Symbolname = "INCREMENT (Unicode = hex. 2206)" _
        & vbCrLf & "    oder GREEK CAPITAL LETTER DELTA (Unicode = hex. 0394)"
Case Is = "45"
        Symbolname = "GREEK CAPITAL LETTER EPSILON (Unicode = hex. 0395)"
Case Is = "46"
        Symbolname = "GREEK CAPITAL LETTER PHI (Unicode = hex. 03A6)"
Case Is = "47"
        Symbolname = "GREEK CAPITAL LETTER GAMMA (Unicode = hex. 0393)"
Case Is = "48"
        Symbolname = "GREEK CAPITAL LETTER ETA (Unicode = hex. 0397)"
Case Is = "49"
        Symbolname = "GREEK CAPITAL LETTER IOTA (Unicode = hex. 0399)"
Case Is = "4A"
        Symbolname = "GREEK THETA SYMBOL (Unicode = hex. 03D1)"
Case Is = "4B"
        Symbolname = "GREEK CAPITAL LETTER KAPPA (Unicode = hex. 039A)"
Case Is = "4C"
        Symbolname = "GREEK CAPITAL LETTER LAMDA (Unicode = hex. 039B)"
Case Is = "4D"
        Symbolname = "GREEK CAPITAL LETTER MU (Unicode = hex. 039C)"
Case Is = "4E"
        Symbolname = "GREEK CAPITAL LETTER NU (Unicode = hex. 039D)"
Case Is = "4F"
        Symbolname = "GREEK CAPITAL LETTER OMICRON (Unicode = hex. 039F)"
Case Is = "50"
        Symbolname = "GREEK CAPITAL LETTER PI (Unicode = hex. 03A0)"
Case Is = "51"
        Symbolname = "GREEK CAPITAL LETTER THETA (Unicode = hex. 0398)"
Case Is = "52"
        Symbolname = "GREEK CAPITAL LETTER RHO (Unicode = hex. 03A1)"
Case Is = "53"
        Symbolname = "GREEK CAPITAL LETTER SIGMA (Unicode = hex. 03A3)"
Case Is = "54"
        Symbolname = "GREEK CAPITAL LETTER TAU (Unicode = hex. 03A4)"
Case Is = "55"
        Symbolname = "GREEK CAPITAL LETTER UPSILON (Unicode = hex. 03A5)"
Case Is = "56"
        Symbolname = "GREEK SMALL LETTER FINAL SIGMA (Unicode = hex. 03C2)"
Case Is = "57"
       Symbolname = "OHM SIGN (Unicode = hex. 2126)" _
        & vbCrLf & "    oder GREEK CAPITAL LETTER OMEGA (Unicode = hex. 03A9)"
Case Is = "58"
        Symbolname = "GREEK CAPITAL LETTER XI (Unicode = hex. 039E)"
Case Is = "59"
        Symbolname = "GREEK CAPITAL LETTER PSI (Unicode = hex. 03A8)"
Case Is = "5A"
        Symbolname = "GREEK CAPITAL LETTER ZETA (Unicode = hex. 0396)"
Case Is = "5B"
        Symbolname = "LEFT SQUARE BRACKET (Unicode = hex. 005B)"
Case Is = "5C"
        Symbolname = "THEREFORE (Unicode = hex. 2234)"
Case Is = "5D"
        Symbolname = "RIGHT SQUARE BRACKET (Unicode = hex. 005D)"
Case Is = "5E"
        Symbolname = "UP TACK (Unicode = hex. 22A5)"
Case Is = "5F"
        Symbolname = "LOW LINE (Unicode = hex. 005F)"
Case Is = "60"
        Symbolname = vbCrLf _
        & "    (kein Unicode-Zeichen für RADICAL EXTENDER vorhanden)," _
        & vbCrLf _
        & "    Ersatz-Vorschlag: MACRON (Unicode = hex. 00AF) oder COMBINING OVERLINE " _
```

```
            & "(Unicode = hex. 0305)"
    Case Is = "61"
            Symbolname = "GREEK SMALL LETTER ALPHA (Unicode = hex. 03B1)"
    Case Is = "62"
            Symbolname = "GREEK SMALL LETTER BETA (Unicode = hex. 03B2)"
    Case Is = "63"
            Symbolname = "GREEK SMALL LETTER CHI (Unicode = hex. 03C7)"
    Case Is = "64"
            Symbolname = "GREEK SMALL LETTER DELTA (Unicode = hex. 03B4)"
    Case Is = "65"
            Symbolname = "GREEK SMALL LETTER EPSILON (Unicode = hex. 03B5)"
    Case Is = "66"
            Symbolname = "GREEK PHI SYMBOL (Unicode = hex. 03D5)"
'                           (gegenüber Originallisten korrigiert)
    Case Is = "67"
            Symbolname = "GREEK SMALL LETTER GAMMA (Unicode = hex. 03B3)"
    Case Is = "68"
            Symbolname = "GREEK SMALL LETTER ETA (Unicode = hex. 03B7)"
    Case Is = "69"
            Symbolname = "GREEK SMALL LETTER IOTA (Unicode = hex. 03B9)"
    Case Is = "6A"
            Symbolname = "GREEK SMALL LETTER PHI (Unicode = hex. 03C6)"
'                           (gegenüber Originallisten korrigiert)
    Case Is = "6B"
            Symbolname = "GREEK SMALL LETTER KAPPA (Unicode = hex. 03BA)"
    Case Is = "6C"
            Symbolname = "GREEK SMALL LETTER LAMDA (Unicode = hex. 03BB)"
    Case Is = "6D"
            Symbolname = "MICRO SIGN (Unicode = hex. 00B5)" _
             & vbCrLf & "     oder GREEK SMALL LETTER MU (Unicode = hex. 03BC)"
    Case Is = "6E"
            Symbolname = "GREEK SMALL LETTER NU (Unicode = hex. 03BD)"
    Case Is = "6F"
            Symbolname = "GREEK SMALL LETTER OMICRON (Unicode = hex. 03BF)"
    Case Is = "70"
            Symbolname = "GREEK SMALL LETTER PI (Unicode = hex. 03C0)"
    Case Is = "71"
            Symbolname = "GREEK SMALL LETTER THETA (Unicode = hex. 03B8)"
    Case Is = "72"
            Symbolname = "GREEK SMALL LETTER RHO (Unicode = hex. 03C1)"
    Case Is = "73"
            Symbolname = "GREEK SMALL LETTER SIGMA (Unicode = hex. 03C3)"
    Case Is = "74"
            Symbolname = "GREEK SMALL LETTER TAU (Unicode = hex. 03C4)"
    Case Is = "75"
            Symbolname = "GREEK SMALL LETTER UPSILON (Unicode = hex. 03C5)"
    Case Is = "76"
            Symbolname = "GREEK PI SYMBOL (Unicode = hex. 03D6)"
    Case Is = "77"
            Symbolname = "GREEK SMALL LETTER OMEGA (Unicode = hex. 03C9)"
    Case Is = "78"
            Symbolname = "GREEK SMALL LETTER XI (Unicode = hex. 03BE)"
    Case Is = "79"
            Symbolname = "GREEK SMALL LETTER PSI (Unicode = hex. 03C8)"
    Case Is = "7A"
            Symbolname = "GREEK SMALL LETTER ZETA (Unicode = hex. 03B6)"
    Case Is = "7B"
            Symbolname = "LEFT CURLY BRACKET (Unicode = hex. 007B)"
    Case Is = "7C"
            Symbolname = "VERTICAL LINE (Unicode = hex. 007C)"
    Case Is = "7D"
            Symbolname = "RIGHT CURLY BRACKET (Unicode = hex. 007D)"
    Case Is = "7E"
```

```
        Symbolname = "TILDE OPERATOR (Unicode = hex. 223C)"
Case Is = "A0"
        Symbolname = "EURO SIGN (bzw. nicht definiert)(Unicode = hex. 20AC)"
Case Is = "A1"
        Symbolname = "GREEK UPSILON WITH HOOK SYMBOL (Unicode = hex. 03D2)"
Case Is = "A2"
        Symbolname = "PRIME (Unicode = hex. 2032)"
Case Is = "A3"
        Symbolname = "LESS-THAN OR EQUAL TO (Unicode = hex. 2264)"
Case Is = "A4"
        Symbolname = "FRACTION SLASH (Unicode = hex. 2044)"
Case Is = "A5"
        Symbolname = "INFINITY (Unicode = hex. 221E)"
Case Is = "A6"
        Symbolname = "LATIN SMALL LETTER F WITH HOOK (Unicode = hex. 0192)"
Case Is = "A7"
        Symbolname = "BLACK CLUB SUIT (Unicode = hex. 2663)"
Case Is = "A8"
        Symbolname = "BLACK DIAMOND SUIT (Unicode = hex. 2666)"
Case Is = "A9"
        Symbolname = "BLACK HEART SUIT (Unicode = hex. 2665)"
Case Is = "AA"
        Symbolname = "BLACK SPADE SUIT (Unicode = hex. 2660)"
Case Is = "AB"
        Symbolname = "LEFT RIGHT ARROW (Unicode = hex. 2194)"
Case Is = "AC"
        Symbolname = "LEFTWARDS ARROW (Unicode = hex. 2190)"
Case Is = "AD"
        Symbolname = "UPWARDS ARROW (Unicode = hex. 2191)"
Case Is = "AE"
        Symbolname = "RIGHTWARDS ARROW (Unicode = hex. 2192)"
Case Is = "AF"
        Symbolname = "DOWNWARDS ARROW (Unicode = hex. 2193)"
Case Is = "B0"
        Symbolname = "DEGREE SIGN (Unicode = hex. 00B0)"
Case Is = "B1"
        Symbolname = "PLUS-MINUS SIGN (Unicode = hex. 00B1)"
Case Is = "B2"
        Symbolname = "DOUBLE PRIME (Unicode = hex. 2033)"
Case Is = "B3"
        Symbolname = "GREATER-THAN OR EQUAL TO (Unicode = hex. 2265)"
Case Is = "B4"
        Symbolname = "MULTIPLICATION SIGN (Unicode = hex. 00D7)"
Case Is = "B5"
        Symbolname = "PROPORTIONAL TO (Unicode = hex. 221D)"
Case Is = "B6"
        Symbolname = "PARTIAL DIFFERENTIAL (Unicode = hex. 2202)"
Case Is = "B7"
        Symbolname = "BULLET (Unicode = hex. 2022)"
Case Is = "B8"
        Symbolname = "DIVISION SIGN (Unicode = hex. 00F7)"
Case Is = "B9"
        Symbolname = "NOT EQUAL TO (Unicode = hex. 2260)"
Case Is = "BA"
        Symbolname = "IDENTICAL TO (Unicode = hex. 2261)"
Case Is = "BB"
        Symbolname = "ALMOST EQUAL TO (Unicode = hex. 2248)"
Case Is = "BC"
        Symbolname = "HORIZONTAL ELLIPSIS (Unicode = hex. 2026)"
Case Is = "BD"
        Symbolname = "VERTICAL LINE EXTENSION (for arrows) (Unicode = hex. 23D0)" _
         & vbCrLf & "    möglicher Ersatz: Senkrechter Strich (Unicode = hex. 007C)"
Case Is = "BE"
```

```
        Symbolname = "HORIZONTAL ARROW EXTENDER (Unicode = hex. 23AF)," _
          & vbCrLf & "    möglicher Ersatz: Geviertstrich (Unicode = hex. 2014, EM DASH)"
Case Is = "BF"
        Symbolname = "DOWNWARDS ARROW WITH CORNER LEFTWARDS " _
        & "(Unicode = hex. 21B5)"
Case Is = "C0"
        Symbolname = "ALEF SYMBOL (Unicode = hex. 2135)"
Case Is = "C1"
        Symbolname = "BLACK-LETTER CAPITAL I (Unicode = hex. 2111)"
Case Is = "C2"
        Symbolname = "BLACK-LETTER CAPITAL R (Unicode = hex. 211C)"
Case Is = "C3"
        Symbolname = "SCRIPT CAPITAL P (Unicode = hex. 2118)"
Case Is = "C4"
        Symbolname = "CIRCLED TIMES (Unicode = hex. 2297)"
Case Is = "C5"
        Symbolname = "CIRCLED PLUS (Unicode = hex. 2295)"
Case Is = "C6"
        Symbolname = "EMPTY SET (Unicode = hex. 2205)," _
        & vbCrLf & "     meistens aber ""Durchmesser"" (Unicode = hex. 2300)"
Case Is = "C7"
        Symbolname = "INTERSECTION (Unicode = hex. 2229)"
Case Is = "C8"
        Symbolname = "UNION (Unicode = hex. 222A)"
Case Is = "C9"
        Symbolname = "SUPERSET OF (Unicode = hex. 2283)"
Case Is = "CA"
        Symbolname = "SUPERSET OF OR EQUAL TO (Unicode = hex. 2287)"
Case Is = "CB"
        Symbolname = "NOT A SUBSET OF (Unicode = hex. 2284)"
Case Is = "CC"
        Symbolname = "SUBSET OF (Unicode = hex. 2282)"
Case Is = "CD"
        Symbolname = "SUBSET OF OR EQUAL TO (Unicode = hex. 2286)"
Case Is = "CE"
        Symbolname = "ELEMENT OF (Unicode = hex. 2208)"
Case Is = "CF"
        Symbolname = "NOT AN ELEMENT OF (Unicode = hex. 2209)"
Case Is = "D0"
        Symbolname = "ANGLE (Unicode = hex. 2220)"
Case Is = "D1"
        Symbolname = "NABLA (Unicode = hex. 2207)"
Case Is = "D2"
        Symbolname = "REGISTERED SIGN (Unicode = hex. 00AE) mit Serifen"
Case Is = "D3"
        Symbolname = "COPYRIGHT SIGN (Unicode = hex. 00A9) mit Serifen"
Case Is = "D4"
        Symbolname = "TRADE MARK SIGN (Unicode = hex. 2122) mit Serifen"
Case Is = "D5"
        Symbolname = "N-ARY PRODUCT (Unicode = hex. 220F)"
Case Is = "D6"
        Symbolname = "SQUARE ROOT (Unicode = hex. 221A)"
Case Is = "D7"
        Symbolname = "DOT OPERATOR (Unicode = hex. 22C5)"
Case Is = "D8"
        Symbolname = "NOT SIGN (Unicode = hex. 00AC)"
Case Is = "D9"
        Symbolname = "LOGICAL AND (Unicode = hex. 2227)"
Case Is = "DA"
        Symbolname = "LOGICAL OR (Unicode = hex. 2228)"
Case Is = "DB"
        Symbolname = "LEFT RIGHT DOUBLE ARROW (Unicode = hex. 21D4)"
Case Is = "DC"
```

```
        Symbolname = "LEFTWARDS DOUBLE ARROW (Unicode = hex. 21D0)"
Case Is = "DD"
        Symbolname = "UPWARDS DOUBLE ARROW (Unicode = hex. 21D1)"
Case Is = "DE"
        Symbolname = "RIGHTWARDS DOUBLE ARROW (Unicode = hex. 21D2)"
Case Is = "DF"
        Symbolname = "DOWNWARDS DOUBLE ARROW (Unicode = hex. 21D3)"
Case Is = "E0"
        Symbolname = "LOZENGE (Unicode = hex. 25CA)"
Case Is = "E1"
        Symbolname = "LEFT-POINTING ANGLE BRACKET (Unicode = hex. 2329)"
Case Is = "E2"
        Symbolname = "REGISTERED SIGN, (Unicode = hex. 00AE) Sans Serif"
Case Is = "E3"
        Symbolname = "COPYRIGHT SIGN, (Unicode = hex. 00A9) Sans Serif"
Case Is = "E4"
        Symbolname = "TRADE MARK SIGN, (Unicode = hex. 2122) Sans Serif"
Case Is = "E5"
        Symbolname = "N-ARY SUMMATION (Unicode = hex. 2211)"
Case Is = "E6"
        Symbolname = "LEFT PARENTHESIS UPPER HOOK (Unicode = hex. 239B)"
Case Is = "E7"
        Symbolname = "LEFT PARENTHESIS EXTENSION (Unicode = hex. 239C)"
Case Is = "E8"
        Symbolname = "LEFT PARENTHESIS LOWER HOOK (Unicode = hex. 239D)"
Case Is = "E9"
        Symbolname = "LEFT SQUARE BRACKET UPPER CORNER (Unicode = hex. 23A1)"
Case Is = "EA"
        Symbolname = "LEFT SQUARE BRACKET EXTENSION (Unicode = hex. 23A2)"
Case Is = "EB"
        Symbolname = "LEFT SQUARE BRACKET LOWER CORNER (Unicode = hex. 23A3)"
Case Is = "EC"
        Symbolname = "LEFT CURLY BRACKET UPPER HOOK (Unicode = hex. 23A7)"
Case Is = "ED"
        Symbolname = "LEFT CURLY BRACKET MIDDLE PIECE (Unicode = hex. 23A8)"
Case Is = "EE"
        Symbolname = "LEFT CURLY BRACKET LOWER HOOK (Unicode = hex. 23A9)"
Case Is = "EF"
        Symbolname = "CURLY BRACKET EXTENSION (Unicode = hex. 23AA)"
Case Is = "F0"
        Symbolname = "(nicht definiert, bzw. Apple-Logo)"
Case Is = "F1"
        Symbolname = "RIGHT-POINTING ANGLE BRACKET (Unicode = hex. 232A)"
Case Is = "F2"
        Symbolname = "INTEGRAL (Unicode = hex. 222B)"
Case Is = "F3"
        Symbolname = "TOP HALF INTEGRAL (Unicode = hex. 2320)"
Case Is = "F4"
        Symbolname = "INTEGRAL EXTENSION (Unicode = hex. 23AE)"
Case Is = "F5"
        Symbolname = "BOTTOM HALF INTEGRAL (Unicode = hex. 2321)"
Case Is = "F6"
        Symbolname = "RIGHT PARENTHESIS UPPER HOOK (Unicode = hex. 239E)"
Case Is = "F7"
        Symbolname = "RIGHT PARENTHESIS EXTENSION (Unicode = hex. 239F)"
Case Is = "F8"
        Symbolname = "RIGHT PARENTHESIS LOWER HOOK (Unicode = hex. 23A0)"
Case Is = "F9"
        Symbolname = "RIGHT SQUARE BRACKET UPPER CORNER (Unicode = hex. 23A4)"
Case Is = "FA"
        Symbolname = "RIGHT SQUARE BRACKET EXTENSION (Unicode = hex. 23A5)"
Case Is = "FB"
        Symbolname = "RIGHT SQUARE BRACKET LOWER CORNER (Unicode = hex. 23A6)"
```

```
  Case Is = "FC"
        Symbolname = "RIGHT CURLY BRACKET UPPER HOOK (Unicode = hex. 23AB)"
  Case Is = "FD"
        Symbolname = "RIGHT CURLY BRACKET MIDDLE PIECE (Unicode = hex. 23AC)"
  Case Is = "FE"
        Symbolname = "RIGHT CURLY BRACKET LOWER HOOK (Unicode = hex. 23AD)"
  Case Else
        Symbolname = "(kein Symbol-Zeichen festgelegt!)"
 End Select
End Function
' ----------------------------------------------------------------------------------------------------


Private Function Wingdingsname(Z1 As String) As String
' Erstellt von Dipl.-Ing. Hanna-Chris Gast, 5. März 2011, Stand 12. Oktober 2011.
' Diese Funktion gibt englische Zeichenerklärung und ggf. entsprechende Unicodes an,
' wenn der (niedrige) hexadezimale Symbol-Code eingegeben wird
' Quelle:  http://www.alanwood.net/demos/wingdings.html.
'
 Select Case Z1
    Case Is = "20"
        Wingdingsname = "SPACE (Unicode = hex. 0020)"
  Case Is = "20"
        Wingdingsname = "SPACE (Unicode = hex. 0020)"
  Case Is = "21"
        Wingdingsname = "PENCIL (Unicode = hex. 270F)"
  Case Is = "22"
        Wingdingsname = "BLACK SCISSORS (Unicode = hex. 2702)"
  Case Is = "23"
        Wingdingsname = "UPPER BLADE SCISSORS (Unicode = hex. 2701)"
  Case Is = "24"
        Wingdingsname = "EYEGLASSES (Unicode = hex. 1F453)"
  Case Is = "25"
        Wingdingsname = "BELL (Unicode = hex. 1F514)"
  Case Is = "26"
        Wingdingsname = "OPEN BOOK (Unicode = hex. 1F4D6)"
  Case Is = "27"
        Wingdingsname = "  (CANDLE) (kein Unicode)"
  Case Is = "28"
        Wingdingsname = "BLACK TELEPHONE (Unicode = hex. 260E)"
  Case Is = "29"
        Wingdingsname = "TELEPHONE LOCATION SIGN (Unicode = hex. 2706)"
  Case Is = "2A"
        Wingdingsname = "ENVELOPE (Unicode = hex. 2709)"
  Case Is = "2B"
        Wingdingsname = "ENVELOPE WITH ADDRESS AND STAMP (kein Unicode)"
  Case Is = "2C"
        Wingdingsname = "CLOSED MAILBOX WITH LOWERED FLAG (Unicode = hex. 1F4EA)"
  Case Is = "2D"
        Wingdingsname = "CLOSED MAILBOX WITH RAISED FLAG (Unicode = hex. 1F4EB)"
  Case Is = "2E"
        Wingdingsname = "OPEN MAILBOX WITH RAISED FLAG (Unicode = hex. 1F4EC)"
  Case Is = "2F"
        Wingdingsname = "OPEN MAILBOX WITH LOWERED FLAG (Unicode = hex. 1F4ED)"
  Case Is = "30"
        Wingdingsname = "FILE FOLDER (Unicode = hex. 1F4C1)"
  Case Is = "31"
        Wingdingsname = "OPEN FILE FOLDER (Unicode = hex. 1F4C2)"
  Case Is = "32"
        Wingdingsname = "PAGE FACING UP (Unicode = hex. 1F4C4)"
  Case Is = "33"
        Wingdingsname = "PRINTED PAGE (kein Unicode)"
  Case Is = "34"
```

```
        Wingdingsname = "STACK OF PRINTED PAGES (kein Unicode)"
Case Is = "35"
        Wingdingsname = "FILING CABINET (kein Unicode)"
Case Is = "36"
        Wingdingsname = "HOURGLASS  (Unicode = hex. 231B)"
Case Is = "37"
        Wingdingsname = "KEYBOARD (Unicode = hex. 2328)"
Case Is = "38"
        Wingdingsname = "MOUSE (kein Unicode)"
Case Is = "39"
        Wingdingsname = "TRACKBALL (kein Unicode)"
Case Is = "3A"
        Wingdingsname = "PERSONAL COMPUTER (Unicode = hex. 1F4BB)"
Case Is = "3B"
        Wingdingsname = "HARD DISK (kein Unicode)"
Case Is = "3C"
        Wingdingsname = "FLOPPY DISK (Unicode = hex. 1F4BE)"
Case Is = "3D"
        Wingdingsname = "FLOPPY DISK (kein Unicode)"
Case Is = "3E"
        Wingdingsname = "TAPE DRIVE (Unicode = hex. 2707)"
Case Is = "3F"
        Wingdingsname = "WRITING HAND (Unicode = hex. 270D)"
Case Is = "40"
        Wingdingsname = "WRITING LEFT HAND (kein Unicode)"
Case Is = "41"
        Wingdingsname = "VICTORY HAND (Unicode = hex. 270C)"
Case Is = "42"
        Wingdingsname = "OK HAND SIGN (Unicode = hex. 1F44C)"
Case Is = "43"
        Wingdingsname = "THUMBS UP SIGN (Unicode = hex. 1F44D)"
Case Is = "44"
        Wingdingsname = "THUMBS DOWN SIGN (Unicode = hex. 1F44E)"
Case Is = "45"
        Wingdingsname = "WHITE LEFT POINTING INDEX (Unicode = hex. 261C)"
Case Is = "46"
        Wingdingsname = "WHITE RIGHT POINTING INDEX (Unicode = hex. 261E)"
Case Is = "47"
        Wingdingsname = "WHITE UP POINTING INDEX (Unicode = hex. 261D)"
Case Is = "48"
        Wingdingsname = "WHITE DOWN POINTING INDEX (Unicode = hex. 261F)"
Case Is = "49"
        Wingdingsname = "OPEN HAND (kein Unicode)"
Case Is = "4A"
        Wingdingsname = "WHITE SMILING FACE (Unicode = hex. 263A)"
Case Is = "4B"
        Wingdingsname = "NEUTRAL FACE (Unicode = hex. 1F610)"
Case Is = "4C"
        Wingdingsname = "WHITE FROWNING FACE (Unicode = hex. 2639)"
Case Is = "4D"
        Wingdingsname = "BOMB (Unicode = hex. 1F4A3)"
Case Is = "4E"
        Wingdingsname = "SKULL AND CROSSBONES (Unicode = hex. 2620)"
Case Is = "4F"
        Wingdingsname = "WHITE FLAG (Unicode = hex. 2690)"
Case Is = "50"
        Wingdingsname = "PENNANT ON POLE (kein Unicode)"
Case Is = "51"
        Wingdingsname = "AIRPLANE (Unicode = hex. 2708)"
Case Is = "52"
        Wingdingsname = "WHITE SUN WITH RAYS (Unicode = hex. 263C)"
Case Is = "53"
        Wingdingsname = "DROPLET (Unicode = hex. 1F4A7)"
```

```
Case Is = "54"
        Wingdingsname = "SNOWFLAKE (Unicode = hex. 2744)"
Case Is = "55"
        Wingdingsname = "WHITE LATIN CROSS (kein Unicode)"
Case Is = "56"
        Wingdingsname = "SHADOWED WHITE LATIN CROSS (Unicode = hex. 271E)"
Case Is = "57"
        Wingdingsname = "CELTIC CROSS (kein Unicode)"
Case Is = "58"
        Wingdingsname = "MALTESE CROSS (Unicode = hex. 2720)"
Case Is = "59"
        Wingdingsname = "STAR OF DAVID (Unicode = hex. 2721)"
Case Is = "5A"
        Wingdingsname = "STAR AND CRESCENT (Unicode = hex. 262A)"
Case Is = "5B"
        Wingdingsname = "YIN YANG (Unicode = hex. 262F)"
Case Is = "5C"
        Wingdingsname = "DEVANAGARI OM (Unicode = hex. 0950)"
Case Is = "5D"
        Wingdingsname = "WHEEL OF DHARMA (Unicode = hex. 2638)"
Case Is = "5E"
        Wingdingsname = "ARIES (Unicode = hex. 2648)"
Case Is = "5F"
        Wingdingsname = "TAURUS (Unicode = hex. 2649)"
Case Is = "60"
        Wingdingsname = "GEMINI (Unicode = hex. 264A)"
Case Is = "61"
        Wingdingsname = "CANCER (Unicode = hex. 264B)"
Case Is = "62"
        Wingdingsname = "LEO (Unicode = hex. 264C)"
Case Is = "63"
        Wingdingsname = "VIRGO (Unicode = hex. 264D)"
Case Is = "64"
        Wingdingsname = "LIBRA (Unicode = hex. 264E)"
Case Is = "65"
        Wingdingsname = "SCORPIO (Unicode = hex. 264F)"
Case Is = "66"
        Wingdingsname = "SAGITTARIUS (Unicode = hex. 2650)"
Case Is = "67"
        Wingdingsname = "CAPRICORN (Unicode = hex. 2651)"
Case Is = "68"
        Wingdingsname = "AQUARIUS (Unicode = hex. 2652)"
Case Is = "69"
        Wingdingsname = "PISCES (Unicode = hex. 2653)"
Case Is = "6A"
        Wingdingsname = "AMPERSAND (lower Case)  (Unicode = hex. 0026)"
Case Is = "6B"
        Wingdingsname = "AMPERSAND (ITALIC) (Unicode = hex. 0026)"
Case Is = "6C"
        Wingdingsname = "BLACK CIRCLE (Unicode = hex. 25CF)"
Case Is = "6D"
        Wingdingsname = "SHADOWED WHITE CIRCLE (Unicode = hex. 274D)"
Case Is = "6E"
        Wingdingsname = "BLACK SQUARE (Unicode = hex. 25A0)"
Case Is = "6F"
        Wingdingsname = "WHITE SQUARE (Unicode = hex. 25A1)"
Case Is = "70"
        Wingdingsname = "BOLD WHITE SQUARE (kein Unicode)"
Case Is = "71"
        Wingdingsname = "LOWER RIGHT SHADOWED WHITE SQUARE (Unicode = hex. 2751)"
Case Is = "72"
        Wingdingsname = "UPPER RIGHT SHADOWED WHITE SQUARE (Unicode = hex. 2752)"
Case Is = "73"
```

```
        Wingdingsname = "BLACK MEDIUM LOZENGE (Unicode = hex. 2B27)"
Case Is = "74"
        Wingdingsname = "BLACK LOZENGE (Unicode = hex. 29EB)"
Case Is = "75"
        Wingdingsname = "BLACK DIAMOND (Unicode = hex. 25C6)"
Case Is = "76"
        Wingdingsname = "BLACK DIAMOND MINUS WHITE X (Unicode = hex. 2756)"
Case Is = "77"
        Wingdingsname = "BLACK MEDIUM DIAMOND (Unicode = hex. 2B25)"
Case Is = "78"
        Wingdingsname = "X IN A RECTANGLE BOX (Unicode = hex. 2327)"
Case Is = "79"
        Wingdingsname = "APL FUNCTIONAL SYMBOL QUAD UP CARET (Unicode = hex. 2353)"
Case Is = "7A"
        Wingdingsname = "PLACE OF INTEREST SIGN (Unicode = hex. 2318)"
Case Is = "7B"
        Wingdingsname = "WHITE FLORETTE (Unicode = hex. 2740)"
Case Is = "7C"
        Wingdingsname = "BLACK FLORETTE (Unicode = hex. 273F)"
Case Is = "7D"
        Wingdingsname = "HEAVY DOUBLE TURNED COMMA QUOTATION MARK " _
        & "ORNAMENT (Unicode = hex. 275D)"
Case Is = "7E"
        Wingdingsname = "HEAVY DOUBLE COMMA QUOTATION MARK ORNAMENT " _
        & "(Unicode = hex. 275E)"
Case Is = "7F"
        Wingdingsname = "(WHITE VERTICAL RECTANGLE)   (Unicode = hex. 25AF)"
Case Is = "80"
        Wingdingsname = "CIRCLED DIGIT ZERO (Unicode = hex. 24EA)"
Case Is = "81"
        Wingdingsname = "CIRCLED DIGIT ONE (Unicode = hex. 2460)"
Case Is = "82"
        Wingdingsname = "CIRCLED DIGIT TWO (Unicode = hex. 2461)"
Case Is = "83"
        Wingdingsname = "CIRCLED DIGIT THREE (Unicode = hex. 2462)"
Case Is = "84"
        Wingdingsname = "CIRCLED DIGIT FOUR (Unicode = hex. 2463)"
Case Is = "85"
        Wingdingsname = "CIRCLED DIGIT FIVE (Unicode = hex. 2464)"
Case Is = "86"
        Wingdingsname = "CIRCLED DIGIT SIX (Unicode = hex. 2465)"
Case Is = "87"
        Wingdingsname = "CIRCLED DIGIT SEVEN (Unicode = hex. 2466)"
Case Is = "88"
        Wingdingsname = "CIRCLED DIGIT EIGHT (Unicode = hex. 2467)"
Case Is = "89"
        Wingdingsname = "CIRCLED DIGIT NINE (Unicode = hex. 2468)"
Case Is = "8A"
        Wingdingsname = "CIRCLED NUMBER TEN (Unicode = hex. 2469)"
Case Is = "8B"
        Wingdingsname = "NEGATIVE CIRCLED DIGIT ZERO (Unicode = hex. 24FF)"
Case Is = "8C"
        Wingdingsname = "DINGBAT NEGATIVE CIRCLED DIGIT ONE (Unicode = hex. 2776)"
Case Is = "8D"
        Wingdingsname = "DINGBAT NEGATIVE CIRCLED DIGIT TWO (Unicode = hex. 2777)"
Case Is = "8E"
        Wingdingsname = "DINGBAT NEGATIVE CIRCLED DIGIT THREE (Unicode = hex. 2778)"
Case Is = "8F"
        Wingdingsname = "DINGBAT NEGATIVE CIRCLED DIGIT FOUR (Unicode = hex. 2779)"
Case Is = "90"
        Wingdingsname = "DINGBAT NEGATIVE CIRCLED DIGIT FIVE (Unicode = hex. 277A)"
Case Is = "91"
        Wingdingsname = "DINGBAT NEGATIVE CIRCLED DIGIT SIX (Unicode = hex. 277B)"
```

```
Case Is = "92"
        Wingdingsname = "DINGBAT NEGATIVE CIRCLED DIGIT SEVEN (Unicode = hex. 277C)"
Case Is = "93"
        Wingdingsname = "DINGBAT NEGATIVE CIRCLED DIGIT EIGHT (Unicode = hex. 277D)"
Case Is = "94"
        Wingdingsname = "DINGBAT NEGATIVE CIRCLED DIGIT NINE (Unicode = hex. 277E)"
Case Is = "95"
        Wingdingsname = "DINGBAT NEGATIVE CIRCLED NUMBER TEN (Unicode = hex. 277F)"
Case Is = "96"
        Wingdingsname = "BUD AND LEAF NORTH EAST (kein Unicode)"
Case Is = "97"
        Wingdingsname = "BUD AND LEAF NORTH WEST (kein Unicode)"
Case Is = "98"
        Wingdingsname = "BUD AND LEAF SOUTH WEST (kein Unicode)"
Case Is = "99"
        Wingdingsname = "BUD AND LEAF SOUTH EAST (kein Unicode)"
Case Is = "9A"
        Wingdingsname = "BOLD VINE LEAF NORTH EAST (kein Unicode)"
Case Is = "9B"
        Wingdingsname = "BOLD VINE LEAF NORTH WEST (kein Unicode)"
Case Is = "9C"
        Wingdingsname = "BOLD VINE LEAF SOUTH WEST (kein Unicode)"
Case Is = "9D"
        Wingdingsname = "BOLD VINE LEAF SOUTH EAST (kein Unicode)"
Case Is = "9E"
        Wingdingsname = "MIDDLE DOT (Unicode = hex. 00B7)"
Case Is = "9F"
        Wingdingsname = "BULLET (Unicode = hex. 2022)"
Case Is = "A0"
        Wingdingsname = "BLACK SMALL SQUARE   (Unicode = hex. 25AA)"
Case Is = "A1"
        Wingdingsname = "WHITE CIRCLE (Unicode = hex. 25CB)"
Case Is = "A2"
        Wingdingsname = "BOLD WHITE CIRCLE (kein Unicode)"
Case Is = "A3"
        Wingdingsname = "EXTRA BOLD WHITE CIRCLE (kein Unicode)"
Case Is = "A4"
        Wingdingsname = "FISHEYE (Unicode = hex. 25C9)"
Case Is = "A5"
        Wingdingsname = "BULLSEYE (Unicode = hex. 25CE)"
Case Is = "A6"
        Wingdingsname = "ähnlich SHADOWED WHITE CIRCLE (Unicode = hex. 274D)"
Case Is = "A7"
        Wingdingsname = "BLACK SMALL SQUARE 7 (Unicode = hex. 25AA)"
Case Is = "A8"
        Wingdingsname = "WHITE MEDIUM SQUARE (Unicode = hex. 25FB)"
Case Is = "A9"
        Wingdingsname = "BLACK THREE POINTED STAR (kein Unicode)"
Case Is = "AA"
        Wingdingsname = "BLACK FOUR POINTED STAR (Unicode = hex. 2726)"
Case Is = "AB"
        Wingdingsname = "BLACK STAR (Unicode = hex. 2605)"
Case Is = "AC"
        Wingdingsname = "SIX POINTED BLACK STAR (Unicode = hex. 2736)"
Case Is = "AD"
        Wingdingsname = "EIGHT POINTED BLACK STAR (Unicode = hex. 2734)"
Case Is = "AE"
        Wingdingsname = "TWELVE POINTED BLACK STAR (Unicode = hex. 2739)"
Case Is = "AF"
        Wingdingsname = "EIGHT POINTED PINWHEEL STAR (Unicode = hex. 2735)"
Case Is = "B0"
        Wingdingsname = "SQUARE REGISTER MARK (kein Unicode)"
Case Is = "B1"
```

```
        Wingdingsname = "POSITION INDICATOR  (Unicode = hex. 2316)"
Case Is = "B2"
        Wingdingsname = "WHITE CONCAVE-SIDED DIAMOND (Unicode = hex. 27E1)"
Case Is = "B3"
        Wingdingsname = "SQUARE LOZENGE (Unicode = hex. 2311)"
Case Is = "B4"
        Wingdingsname = "QUESTION MARK IN WHITE DIAMOND (kein Unicode)"
Case Is = "B5"
        Wingdingsname = "CIRCLED WHITE STAR (Unicode = hex. 272A)"
Case Is = "B6"
        Wingdingsname = "SHADOWED WHITE STAR (Unicode = hex. 2730)"
Case Is = "B7"
        Wingdingsname = "CLOCK FACE ONE O CLOCK (Unicode = hex. 1F550)"
Case Is = "B8"
        Wingdingsname = "CLOCK FACE TWO O CLOCK (Unicode = hex. 1F551)"
Case Is = "B9"
        Wingdingsname = "CLOCK FACE THREE O CLOCK (Unicode = hex. 1F552)"
Case Is = "BA"
        Wingdingsname = "CLOCK FACE FOUR O CLOCK (Unicode = hex. 1F553)"
Case Is = "BB"
        Wingdingsname = "CLOCK FACE FIVE O CLOCK (Unicode = hex. 1F554)"
Case Is = "BC"
        Wingdingsname = "CLOCK FACE SIX O CLOCK (Unicode = hex. 1F555)"
Case Is = "BD"
        Wingdingsname = "CLOCK FACE SEVEN O CLOCK (Unicode = hex. 1F556)"
Case Is = "BE"
        Wingdingsname = "CLOCK FACE EIGHT O CLOCK (Unicode = hex. 1F557)"
Case Is = "BF"
        Wingdingsname = "CLOCK FACE NINE O CLOCK (Unicode = hex. 1F558)"
Case Is = "C0"
        Wingdingsname = "CLOCK FACE TEN O CLOCK (Unicode = hex. 1F559)"
Case Is = "C1"
        Wingdingsname = "CLOCK FACE ELEVEN O CLOCK (Unicode = hex. 1F55A)"
Case Is = "C2"
        Wingdingsname = "CLOCK FACE TWELVE O CLOCK (Unicode = hex. 1F55B)"
Case Is = "C3"
        Wingdingsname = "(WHITE ARROW POINTING DOWNWARDS THEN CURVING " _
        & "LEFTWARDS (kein Unicode)"
Case Is = "C4"
        Wingdingsname = "WHITE ARROW POINTING DOWNWARDS THEN CURVING " _
        & "RIGHTWARDS (kein Unicode)"
Case Is = "C5"
        Wingdingsname = "WHITE ARROW POINTING UPWARDS THEN CURVING " _
        & "LEFTWARDS (kein Unicode)"
Case Is = "C6"
        Wingdingsname = "WHITE ARROW POINTING UPWARDS THEN CURVING " _
        & "RIGHTWARDS (kein Unicode)"
Case Is = "C7"
        Wingdingsname = "WHITE ARROW POINTING LEFTWARDS THEN CURVING " _
        & "UPWARDS (kein Unicode)"
Case Is = "C8"
        Wingdingsname = "WHITE ARROW POINTING RIGHTWARDS THEN CURVING " _
        & "UPWARDS (kein Unicode)"
Case Is = "C9"
        Wingdingsname = "WHITE ARROW POINTING LEFTWARDS THEN CURVING " _
        & "DOWNWARDS (kein Unicode)"
Case Is = "CA"
        Wingdingsname = "WHITE ARROW POINTING RIGHTWARDS THEN CURVING " _
        & "DOWNWARDS (kein Unicode)"
Case Is = "CB"
        Wingdingsname = "QUILT SQUARE 2 (kein Unicode)"
Case Is = "CC"
        Wingdingsname = "BLACK QUILT SQUARE 2 (kein Unicode)"
```

```
Case Is = "CD"
        Wingdingsname = "LEAF COUNTERCLOCKWISE SOUTH WEST (kein Unicode)"
Case Is = "CE"
        Wingdingsname = "LEAF COUNTERCLOCKWISE NORTH WEST (kein Unicode)"
Case Is = "CF"
        Wingdingsname = "LEAF COUNTERCLOCKWISE SOUTH EAST (kein Unicode)"
Case Is = "D0"
        Wingdingsname = "LEAF COUNTERCLOCKWISE NORTH EAST (kein Unicode)"
Case Is = "D1"
        Wingdingsname = "LEAF NORTH WEST (kein Unicode)"
Case Is = "D2"
        Wingdingsname = "LEAF SOUTH WEST (kein Unicode)"
Case Is = "D3"
        Wingdingsname = "LEAF NORTH EAST (kein Unicode)"
Case Is = "D4"
        Wingdingsname = "LEAF SOUTH EAST (kein Unicode)"
Case Is = "D5"
        Wingdingsname = "ERASE TO THE LEFT (Unicode = hex. 232B)"
Case Is = "D6"
        Wingdingsname = "ERASE TO THE RIGHT (Unicode = hex. 2326)"
Case Is = "D7"
        Wingdingsname = "THREE-D TOP-LIGHTED LEFTWARDS ARROWHEAD (kein Unicode)"
Case Is = "D8"
        Wingdingsname = "THREE-D TOP-LIGHTED RIGHTWARDS ARROWHEAD " _
        & "(Unicode = hex. 27A2)"
Case Is = "D9"
        Wingdingsname = "THREE-D RIGHT-LIGHTED UPWARDS ARROWHEAD " _
        & "(kein Unicode)"
Case Is = "DA"
        Wingdingsname = "THREE-D LEFT-LIGHTED DOWNWARDS ARROWHEAD " _
        & "(kein Unicode)"
Case Is = "DB"
        Wingdingsname = "CIRCLED HEAVY WHITE LEFTWARDS ARROW (kein Unicode)"
Case Is = "DC"
        Wingdingsname = "CIRCLED HEAVY WHITE RIGHTWARDS ARROW " _
        & "(Unicode = hex. 27B2)"
Case Is = "DD"
        Wingdingsname = "CIRCLED HEAVY WHITE UPWARDS ARROW (kein Unicode)"
Case Is = "DE"
        Wingdingsname = "CIRCLED HEAVY WHITE DOWNWARDS ARROW (kein Unicode)"
Case Is = "DF"
        Wingdingsname = "WIDE-HEADED LEFTWARDS ARROW (kein Unicode)"
Case Is = "E0"
        Wingdingsname = "WIDE-HEADED RIGHTWARDS ARROW (kein Unicode)"
Case Is = "E1"
        Wingdingsname = "WIDE-HEADED UPWARDS ARROW (kein Unicode)"
Case Is = "E2"
        Wingdingsname = "WIDE-HEADED DOWNWARDS ARROW (kein Unicode)"
Case Is = "E3"
        Wingdingsname = "WIDE-HEADED NORTH WEST ARROW (kein Unicode)"
Case Is = "E4"
        Wingdingsname = "WIDE-HEADED NORTH EAST ARROW (kein Unicode)"
Case Is = "E5"
        Wingdingsname = "WIDE-HEADED SOUTH WEST ARROW (kein Unicode)"
Case Is = "E6"
        Wingdingsname = "WIDE-HEADED SOUTH EAST ARROW (kein Unicode)"
Case Is = "E7"
        Wingdingsname = "HEAVY WIDE-HEADED LEFTWARDS ARROW (kein Unicode)"
Case Is = "E8"
        Wingdingsname = "HEAVY WIDE-HEADED RIGHTWARDS ARROW (Unicode = hex. 2794)"
Case Is = "E9"
        Wingdingsname = "HEAVY WIDE-HEADED UPWARDS ARROW (kein Unicode)"
Case Is = "EA"
```

```
            Wingdingsname = "HEAVY WIDE-HEADED DOWNWARDS ARROW (kein Unicode)"
    Case Is = "EB"
            Wingdingsname = "HEAVY WIDE-HEADED NORTH WEST ARROW (kein Unicode)"
    Case Is = "EC"
            Wingdingsname = "HEAVY WIDE-HEADED NORTH EAST ARROW (kein Unicode)"
    Case Is = "ED"
            Wingdingsname = "HEAVY WIDE-HEADED SOUTH WEST ARROW (kein Unicode)"
    Case Is = "EE"
            Wingdingsname = "HEAVY WIDE-HEADED SOUTH EAST ARROW (kein Unicode)"
    Case Is = "EF"
            Wingdingsname = "LEFTWARDS WHITE ARROW (Unicode = hex. 21E6)"
    Case Is = "F0"
            Wingdingsname = "RIGHTWARDS WHITE ARROW (Unicode = hex. 21E8)"
    Case Is = "F1"
            Wingdingsname = "UPWARDS WHITE ARROW (Unicode = hex. 21E7)"
    Case Is = "F2"
            Wingdingsname = "DOWNWARDS WHITE ARROW (Unicode = hex. 21E9)"
    Case Is = "F3"
            Wingdingsname = "LEFT RIGHT WHITE ARROW (Unicode = hex. 2B04)"
    Case Is = "F4"
            Wingdingsname = "UP DOWN WHITE ARROW (Unicode = hex. 21F3)"
    Case Is = "F5"
            Wingdingsname = "NORTH WEST WHITE ARROW (Unicode = hex. 2B01)"
    Case Is = "F6"
            Wingdingsname = "NORTH EAST WHITE ARROW (Unicode = hex. 2B00)"
    Case Is = "F7"
            Wingdingsname = "SOUTH WEST WHITE ARROW (Unicode = hex. 2B03)"
    Case Is = "F8"
            Wingdingsname = "SOUTH EAST WHITE ARROW (Unicode = hex. 2B02)"
    Case Is = "F9"
            Wingdingsname = "WHITE RECTANGLE (Unicode = hex. 25AD)"
    Case Is = "FA"
            Wingdingsname = "WHITE SMALL SQUARE (Unicode = hex. 25AB)"
    Case Is = "FB"
            Wingdingsname = "BALLOT X (Unicode = hex. 2717)"
    Case Is = "FC"
            Wingdingsname = "CHECK MARK (Unicode = hex. 2713)"
    Case Is = "FD"
            Wingdingsname = "BALLOT BOX WITH X (FONT-WEIGHT: BOLD) (Unicode = hex. 2612)"
    Case Is = "FE"
            Wingdingsname = "BALLOT BOX WITH CHECK (FONT-WEIGHT: BOLD) " _
            & "(Unicode = hex. 2611)"
    Case Is = "FF"
            Wingdingsname = "WINDOWS LOGO (kein Unicode)"
    Case Else
            Wingdingsname = "(KEIN WINGDINGS-ZEICHEN FESTGELEGT!)"
  End Select
End Function
' ---------------------------------------------------------------------------------------------
```

```
Private Sub Ueberschriften_fett()
'  28. März 2014.
'  Ergebnis-Zwischenüberschriften fett machen:
   Selection.Find.ClearFormatting
```

```
  Selection.Find.Replacement.ClearFormatting
  Selection.Find.Replacement.Font.Bold = True
  With Selection.Find
    .Text = "Feldcode-Ansicht:"
    .Replacement.Text = ""
    .Replacement.Font.Size = 12
    .Forward = True
    .Wrap = wdFindContinue
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceOne
  '
  With Selection.Find
    .Text = "Die Zeichenanalyse ergibt:"
    .Replacement.Text = ""
    .Replacement.Font.Size = 12
    .Forward = True
    .Wrap = wdFindContinue
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceOne
  '
  With Selection.Find
    .Text = "Die Markierung enthält folgende Textmarken"
    .Replacement.Text = "Die Markierung enthält folgende Textmarken"
    .Replacement.Font.Size = 12
    .Forward = True
    .Wrap = wdFindContinue
    .Format = True
    .MatchCase = False
    .MatchWholeWord = False
    .MatchWildcards = False
    .MatchSoundsLike = False
    .MatchAllWordForms = False
  End With
  Selection.Find.Execute Replace:=wdReplaceOne
  Selection.EndKey Unit:=wdStory
End Sub
' ------------------------------------------------------------------------------------------
```

### 6.2.4 Untergeordnete Funktion zur Ermittlung des Unicode-Zeichennamen

Die folgenden Funktionen sind im downloadbaren Makro-Modul vor den eigentlichen Prozeduren angeordnet. Die erste fragt aus einer bereits geöffneten Textdatei die (englischen) Unicode-Namen ab.

```
Private Function EnglischName(Z1 As String) As String
'  Makro erstellt von Dipl.-Ing. Hanna-Chris Gast am 20. Juli 2010, Stand 3. November 2010.
'  Diese Funktion sucht  aus der bereits downgeloadeten und geöffneten Unicode-Liste
'     http://unicode.org/Public/UNIDATA/UnicodeData.txt  den englischen
```

```
'    Zeichennamen heraus.
  Dim Rng1, Rng2, y1
  '
  ActiveDocument.Range(0, 0).Select
  '
  Selection.Find.ClearFormatting
  With Selection.Find
      .Text = "^p" & Z1 & ";"
      .Replacement.Text = ""
      .Forward = True
      .Wrap = wdFindContinue
      .Format = False
      .MatchCase = False
      .MatchWholeWord = False
      .MatchWildcards = False
      .MatchSoundsLike = False
      .MatchAllWordForms = False
  End With
  Selection.Find.Execute
  '
  If Selection.Find.Found = True Then
     ' Zeile ist gefunden, jetzt Resultat einlesen:
     Selection.Collapse Direction:=wdCollapseEnd
     Set Rng1 = Selection.Range
     '  bis zum nächsten Semikolon suchen
     Selection.Find.ClearFormatting
     With Selection.Find
         .Text = ";"
         .Replacement.Text = ""
         .Forward = True
         .Wrap = wdFindContinue
         .Format = False
         .MatchCase = False
         .MatchWholeWord = False
         .MatchWildcards = False
         .MatchSoundsLike = False
         .MatchAllWordForms = False
      End With
      Selection.Find.Execute
      Set Rng2 = Selection.Range
      Rng2.SetRange Start:=Rng1.Start, End:=Rng2.End - 1
      Rng2.Select
    ' Gefundene Erklärung einlesen:
      EnglischName = "der" & Chr(160) & "genormte" & Chr(160) & "englische" _
            & Chr(160) & "Zeichenname" & Chr(160) & "ist:" & Chr(160) & Selection
  Else
    y1 = Val("&H" & Z1 & "&")
    Select Case y1
      Case Val(&H3400&) To Val(&H4DBF&)
        EnglischName = "das" & Chr(160) & "Zeichen gehört zu ""CJK Ideograph Extension A""," _
                & Chr(11) & "siehe http://www.unicode.org/charts/PDF/U3400.pdf"
      Case Val(&H4E00&) To Val(&H9FCB&)
        EnglischName = "das" & Chr(160) & "Zeichen ist ein ""CJK-Ideograph""," & Chr(11) _
                & "siehe http://www.unicode.org/charts/PDF/U4E00.pdf"
      Case Val(&HAC00&) To Val(&HD7AF&)
        EnglischName = "das" & Chr(160) & "Zeichen ist eine ""Hangul Syllable"", " & Chr(11) _
                & "siehe http://www.unicode.org/charts/PDF/UAC00.pdf"
      Case 57344 To 63743
        EnglischName = "das" & Chr(160) & "Zeichen ist nicht genormt und gehört zur " _
                 & """Privat-Use-Area"""
      Case Val(&H20000) To Val(&H2A6DF & "&")
        EnglischName = "das" & Chr(160) & "Zeichen gehört zu ""CJK Ideograph Extension B""," _
                & Chr(11) & "siehe  http://www.unicode.org/charts/PDF/U20000.pdf"
```

```
    Case Val(&H2A700) To Val(&H2B734 & "&")
      EnglischName = "das" & Chr(160) & "Zeichen gehört zu ""CJK Ideograph Extension C""," _
            & Chr(11) & "siehe http://www.unicode.org/charts/PDF/U2A700.pdf"
    Case Else
      EnglischName = "das" & Chr(160) & "Zeichen ist nicht in der " _
            & "Unicode-Liste enthalten"
  End Select
 End If
End Function
' ----------------------------------------------------------------------------------------------
```

### 6.2.5 Unterfunktion zur Angabe der Bedeutung und der entsprechenden Unicode-Zeichen bei Symbol-Zeichen

Die folgende Funktion habe ich unter Verwendung zwei Quellen aus dem Internet erstellt, die ich auf Unterseiten des Unicode-Consortiums fand [2] [31]. In einigen Fällen wich ich von den Quellen ab, so bei dem kleinen Phi und dem Phi-Symbol, bei denen das Unicode-Consortium die Codes nachträglich vertauscht hatte [21], so dass sich hier auch bei einigen Unicode-Schriftarten Abweichungen bestehen. Nur für ein Symbol-Zeichen gibt es immer noch keine echte Unicode-Entsprechung. In diesem und in einigen anderen Fällen suchte ich optisch ähnliche und besser handhabbare Unicode-Ersatzzeichen, weil bei Strichen wie Pfeil- und Wurzelzeichen-Verlängerungen keine inhaltliche Bedeutung vorliegt.

```
Private Function Symbolname(Z1 As String) As String
' Dipl.-Ing. Hanna-Chris Gast, 30. August 2010, 12. März 2011, Stand 25. September 2014.
'  Diese Funktion gibt die (englischen) Unicode-Namen und entsprechende Unicodes an,
'  wenn der (niedrige) hexadezimale Symbol-Code eingegeben wird
'  Quellen "http://unicode.org/Public/MAPPINGS/VENDORS/APPLE/SYMBOL.TXT" und andere
'  Ein paar Fehler habe ich hierbei beseitigt, zum Beispiel: Kleines Phi und Phi-Symbol
'  waren vertauscht, und in einigen Fällen habe ich Ergänzungen vorgenommen.
'
 Select Case Z1
  Case Is = "20"
        Symbolname = "NO-BREAK SPACE (Unicode = hex. 00A0)"
  Case Is = "21"
        Symbolname = "EXCLAMATION MARK (Unicode = hex. 0021)"
  Case Is = "22"
        Symbolname = "FOR ALL (Unicode = hex. 2200)"
  Case Is = "23"
        Symbolname = "NUMBER SIGN (Unicode = hex. 0023)"
  Case Is = "24"
        Symbolname = "THERE EXISTS (Unicode = hex. 2203)"
  Case Is = "25"
        Symbolname = "PERCENT SIGN (Unicode = hex. 0025)"
  Case Is = "26"
        Symbolname = "AMPERSAND (Unicode = hex. 0026)"
  Case Is = "27"
        Symbolname = "CONTAINS AS MEMBER (Unicode = hex. 220B)"
  Case Is = "28"
        Symbolname = "LEFT PARENTHESIS (Unicode = hex. 0028)"
  Case Is = "29"
        Symbolname = "RIGHT PARENTHESIS (Unicode = hex. 0029)"
```

[31] Eine Liste von Apple [19] und eine ältere Liste von Adobe [20], welche als Unicode-Entsprechung für einige Zeichen nur Adobe-eigene Codes in der Private-Use-Area anbietet.

```
Case Is = "2A"
        Symbolname = "ASTERISK OPERATOR (Unicode = hex. 2217)"
Case Is = "2B"
        Symbolname = "PLUS SIGN (Unicode = hex. 002B)"
Case Is = "2C"
        Symbolname = "COMMA (Unicode = hex. 002C)"
Case Is = "2D"
        Symbolname = "MINUS SIGN (Unicode = hex. 2212)"
Case Is = "2E"
        Symbolname = "FULL STOP (Unicode = hex. 002E)"
Case Is = "2F"
        Symbolname = "SOLIDUS (Unicode = hex. 002F)"
Case Is = "30"
        Symbolname = "DIGIT ZERO (Unicode = hex. 0030)"
Case Is = "31"
        Symbolname = "DIGIT ONE (Unicode = hex. 0031)"
Case Is = "32"
        Symbolname = "DIGIT TWO (Unicode = hex. 0032)"
Case Is = "33"
        Symbolname = "DIGIT THREE (Unicode = hex. 0033)"
Case Is = "34"
        Symbolname = "DIGIT FOUR (Unicode = hex. 0034)"
Case Is = "35"
        Symbolname = "DIGIT FIVE (Unicode = hex. 0035)"
Case Is = "36"
        Symbolname = "DIGIT SIX (Unicode = hex. 0036)"
Case Is = "37"
        Symbolname = "DIGIT SEVEN (Unicode = hex. 0037)"
Case Is = "38"
        Symbolname = "DIGIT EIGHT (Unicode = hex. 0038)"
Case Is = "39"
        Symbolname = "DIGIT NINE (Unicode = hex. 0039)"
Case Is = "3A"
        Symbolname = "COLON (Unicode = hex. 003A)"
Case Is = "3B"
        Symbolname = "SEMICOLON (Unicode = hex. 003B)"
Case Is = "3C"
        Symbolname = "LESS-THAN SIGN (Unicode = hex. 003C)"
Case Is = "3D"
        Symbolname = "EQUALS SIGN (Unicode = hex. 003D)"
Case Is = "3E"
        Symbolname = "GREATER-THAN SIGN (Unicode = hex. 003E)"
Case Is = "3F"
        Symbolname = "QUESTION MARK (Unicode = hex. 003F)"
Case Is = "40"
        Symbolname = "APPROXIMATELY EQUAL TO (Unicode = hex. 2245)"
Case Is = "41"
        Symbolname = "GREEK CAPITAL LETTER ALPHA (Unicode = hex. 0391)"
Case Is = "42"
        Symbolname = "GREEK CAPITAL LETTER BETA (Unicode = hex. 0392)"
Case Is = "43"
        Symbolname = "GREEK CAPITAL LETTER CHI (Unicode = hex. 03A7)"
Case Is = "44"
        Symbolname = "INCREMENT (Unicode = hex. 2206)" _
        & vbCrLf & "     oder GREEK CAPITAL LETTER DELTA (Unicode = hex. 0394)"
Case Is = "45"
        Symbolname = "GREEK CAPITAL LETTER EPSILON (Unicode = hex. 0395)"
Case Is = "46"
        Symbolname = "GREEK CAPITAL LETTER PHI (Unicode = hex. 03A6)"
Case Is = "47"
        Symbolname = "GREEK CAPITAL LETTER GAMMA (Unicode = hex. 0393)"
Case Is = "48"
        Symbolname = "GREEK CAPITAL LETTER ETA (Unicode = hex. 0397)"
```

```
Case Is = "49"
        Symbolname = "GREEK CAPITAL LETTER IOTA (Unicode = hex. 0399)"
Case Is = "4A"
        Symbolname = "GREEK THETA SYMBOL (Unicode = hex. 03D1)"
Case Is = "4B"
        Symbolname = "GREEK CAPITAL LETTER KAPPA (Unicode = hex. 039A)"
Case Is = "4C"
        Symbolname = "GREEK CAPITAL LETTER LAMDA (Unicode = hex. 039B)"
Case Is = "4D"
        Symbolname = "GREEK CAPITAL LETTER MU (Unicode = hex. 039C)"
Case Is = "4E"
        Symbolname = "GREEK CAPITAL LETTER NU (Unicode = hex. 039D)"
Case Is = "4F"
        Symbolname = "GREEK CAPITAL LETTER OMICRON (Unicode = hex. 039F)"
Case Is = "50"
        Symbolname = "GREEK CAPITAL LETTER PI (Unicode = hex. 03A0)"
Case Is = "51"
        Symbolname = "GREEK CAPITAL LETTER THETA (Unicode = hex. 0398)"
Case Is = "52"
        Symbolname = "GREEK CAPITAL LETTER RHO (Unicode = hex. 03A1)"
Case Is = "53"
        Symbolname = "GREEK CAPITAL LETTER SIGMA (Unicode = hex. 03A3)"
Case Is = "54"
        Symbolname = "GREEK CAPITAL LETTER TAU (Unicode = hex. 03A4)"
Case Is = "55"
        Symbolname = "GREEK CAPITAL LETTER UPSILON (Unicode = hex. 03A5)"
Case Is = "56"
        Symbolname = "GREEK SMALL LETTER FINAL SIGMA (Unicode = hex. 03C2)"
Case Is = "57"
       Symbolname = "OHM SIGN (Unicode = hex. 2126)" _
        & vbCrLf & "     oder GREEK CAPITAL LETTER OMEGA (Unicode = hex. 03A9)"
Case Is = "58"
        Symbolname = "GREEK CAPITAL LETTER XI (Unicode = hex. 039E)"
Case Is = "59"
        Symbolname = "GREEK CAPITAL LETTER PSI (Unicode = hex. 03A8)"
Case Is = "5A"
        Symbolname = "GREEK CAPITAL LETTER ZETA (Unicode = hex. 0396)"
Case Is = "5B"
        Symbolname = "LEFT SQUARE BRACKET (Unicode = hex. 005B)"
Case Is = "5C"
        Symbolname = "THEREFORE (Unicode = hex. 2234)"
Case Is = "5D"
        Symbolname = "RIGHT SQUARE BRACKET (Unicode = hex. 005D)"
Case Is = "5E"
        Symbolname = "UP TACK (Unicode = hex. 22A5)"
Case Is = "5F"
        Symbolname = "LOW LINE (Unicode = hex. 005F)"
Case Is = "60"
        Symbolname = "RADICAL EXTENDER" & vbCrLf _
        & "     (kein Unicode-Zeichen dafür vorhanden)," & vbCrLf _
        & "     Ersatz-Vorschlag: MACRON (Unicode = hex. 00AF)" & vbCrLf _
        & "     oder COMBINING OVERLINE (Unicode = hex. 0305)"
Case Is = "61"
        Symbolname = "GREEK SMALL LETTER ALPHA (Unicode = hex. 03B1)"
Case Is = "62"
        Symbolname = "GREEK SMALL LETTER BETA (Unicode = hex. 03B2)"
Case Is = "63"
        Symbolname = "GREEK SMALL LETTER CHI (Unicode = hex. 03C7)"
Case Is = "64"
        Symbolname = "GREEK SMALL LETTER DELTA (Unicode = hex. 03B4)"
Case Is = "65"
        Symbolname = "GREEK SMALL LETTER EPSILON (Unicode = hex. 03B5)"
Case Is = "66"
```

```
            Symbolname = "GREEK PHI SYMBOL (Unicode = hex. 03D5)"
'                           (gegenüber Originallisten korrigiert)
    Case Is = "67"
            Symbolname = "GREEK SMALL LETTER GAMMA (Unicode = hex. 03B3)"
    Case Is = "68"
            Symbolname = "GREEK SMALL LETTER ETA (Unicode = hex. 03B7)"
    Case Is = "69"
            Symbolname = "GREEK SMALL LETTER IOTA (Unicode = hex. 03B9)"
    Case Is = "6A"
            Symbolname = "GREEK SMALL LETTER PHI (Unicode = hex. 03C6)"
'                           (gegenüber Originallisten korrigiert)
    Case Is = "6B"
            Symbolname = "GREEK SMALL LETTER KAPPA (Unicode = hex. 03BA)"
    Case Is = "6C"
            Symbolname = "GREEK SMALL LETTER LAMDA (Unicode = hex. 03BB)"
    Case Is = "6D"
            Symbolname = "MICRO SIGN (Unicode = hex. 00B5)" _
             & vbCrLf & "     oder GREEK SMALL LETTER MU (Unicode = hex. 03BC)"
    Case Is = "6E"
            Symbolname = "GREEK SMALL LETTER NU (Unicode = hex. 03BD)"
    Case Is = "6F"
            Symbolname = "GREEK SMALL LETTER OMICRON (Unicode = hex. 03BF)"
    Case Is = "70"
            Symbolname = "GREEK SMALL LETTER PI (Unicode = hex. 03C0)"
    Case Is = "71"
            Symbolname = "GREEK SMALL LETTER THETA (Unicode = hex. 03B8)"
    Case Is = "72"
            Symbolname = "GREEK SMALL LETTER RHO (Unicode = hex. 03C1)"
    Case Is = "73"
            Symbolname = "GREEK SMALL LETTER SIGMA (Unicode = hex. 03C3)"
    Case Is = "74"
            Symbolname = "GREEK SMALL LETTER TAU (Unicode = hex. 03C4)"
    Case Is = "75"
            Symbolname = "GREEK SMALL LETTER UPSILON (Unicode = hex. 03C5)"
    Case Is = "76"
            Symbolname = "GREEK PI SYMBOL (Unicode = hex. 03D6)"
    Case Is = "77"
            Symbolname = "GREEK SMALL LETTER OMEGA (Unicode = hex. 03C9)"
    Case Is = "78"
            Symbolname = "GREEK SMALL LETTER XI (Unicode = hex. 03BE)"
    Case Is = "79"
            Symbolname = "GREEK SMALL LETTER PSI (Unicode = hex. 03C8)"
    Case Is = "7A"
            Symbolname = "GREEK SMALL LETTER ZETA (Unicode = hex. 03B6)"
    Case Is = "7B"
            Symbolname = "LEFT CURLY BRACKET (Unicode = hex. 007B)"
    Case Is = "7C"
            Symbolname = "VERTICAL LINE (Unicode = hex. 007C)"
    Case Is = "7D"
            Symbolname = "RIGHT CURLY BRACKET (Unicode = hex. 007D)"
    Case Is = "7E"
            Symbolname = "TILDE OPERATOR (Unicode = hex. 223C)"
    Case Is = "A0"
            Symbolname = "EURO SIGN (bzw. nicht definiert)(Unicode = hex. 20AC)"
    Case Is = "A1"
            Symbolname = "GREEK UPSILON WITH HOOK SYMBOL (Unicode = hex. 03D2)"
    Case Is = "A2"
            Symbolname = "PRIME (Unicode = hex. 2032)"
    Case Is = "A3"
            Symbolname = "LESS-THAN OR EQUAL TO (Unicode = hex. 2264)"
    Case Is = "A4"
            Symbolname = "FRACTION SLASH (Unicode = hex. 2044)"
    Case Is = "A5"
```

```
        Symbolname = "INFINITY (Unicode = hex. 221E)"
Case Is = "A6"
        Symbolname = "LATIN SMALL LETTER F WITH HOOK (Unicode = hex. 0192)"
Case Is = "A7"
        Symbolname = "BLACK CLUB SUIT (Unicode = hex. 2663)"
Case Is = "A8"
        Symbolname = "BLACK DIAMOND SUIT (Unicode = hex. 2666)"
Case Is = "A9"
        Symbolname = "BLACK HEART SUIT (Unicode = hex. 2665)"
Case Is = "AA"
        Symbolname = "BLACK SPADE SUIT (Unicode = hex. 2660)"
Case Is = "AB"
        Symbolname = "LEFT RIGHT ARROW (Unicode = hex. 2194)"
Case Is = "AC"
        Symbolname = "LEFTWARDS ARROW (Unicode = hex. 2190)"
Case Is = "AD"
        Symbolname = "UPWARDS ARROW (Unicode = hex. 2191)"
Case Is = "AE"
        Symbolname = "RIGHTWARDS ARROW (Unicode = hex. 2192)"
Case Is = "AF"
        Symbolname = "DOWNWARDS ARROW (Unicode = hex. 2193)"
Case Is = "B0"
        Symbolname = "DEGREE SIGN (Unicode = hex. 00B0)"
Case Is = "B1"
        Symbolname = "PLUS-MINUS SIGN (Unicode = hex. 00B1)"
Case Is = "B2"
        Symbolname = "DOUBLE PRIME (Unicode = hex. 2033)"
Case Is = "B3"
        Symbolname = "GREATER-THAN OR EQUAL TO (Unicode = hex. 2265)"
Case Is = "B4"
        Symbolname = "MULTIPLICATION SIGN (Unicode = hex. 00D7)"
Case Is = "B5"
        Symbolname = "PROPORTIONAL TO (Unicode = hex. 221D)"
Case Is = "B6"
        Symbolname = "PARTIAL DIFFERENTIAL (Unicode = hex. 2202)"
Case Is = "B7"
        Symbolname = "BULLET (Unicode = hex. 2022)"
Case Is = "B8"
        Symbolname = "DIVISION SIGN (Unicode = hex. 00F7)"
Case Is = "B9"
        Symbolname = "NOT EQUAL TO (Unicode = hex. 2260)"
Case Is = "BA"
        Symbolname = "IDENTICAL TO (Unicode = hex. 2261)"
Case Is = "BB"
        Symbolname = "ALMOST EQUAL TO (Unicode = hex. 2248)"
Case Is = "BC"
        Symbolname = "HORIZONTAL ELLIPSIS (Unicode = hex. 2026)"
Case Is = "BD"
        Symbolname = "VERTICAL LINE EXTENSION (for arrows) (Unicode = hex. 23D0)" _
         & vbCrLf & "    möglicher Ersatz: Senkrechter Strich (Unicode = hex. 007C)"
Case Is = "BE"
        Symbolname = "HORIZONTAL ARROW EXTENDER (Unicode = hex. 23AF)," _
          & vbCrLf & "    möglicher Ersatz: Geviertstrich (Unicode = hex. 2014, EM DASH)"
Case Is = "BF"
        Symbolname = "DOWNWARDS ARROW WITH CORNER LEFTWARDS " _
        & "(Unicode = hex. 21B5)"
Case Is = "C0"
        Symbolname = "ALEF SYMBOL (Unicode = hex. 2135)"
Case Is = "C1"
        Symbolname = "BLACK-LETTER CAPITAL I (Unicode = hex. 2111)"
Case Is = "C2"
        Symbolname = "BLACK-LETTER CAPITAL R (Unicode = hex. 211C)"
Case Is = "C3"
```

```
        Symbolname = "SCRIPT CAPITAL P (Unicode = hex. 2118)"
Case Is = "C4"
        Symbolname = "CIRCLED TIMES (Unicode = hex. 2297)"
Case Is = "C5"
        Symbolname = "CIRCLED PLUS (Unicode = hex. 2295)"
Case Is = "C6"
        Symbolname = "EMPTY SET (Unicode = hex. 2205)," _
        & vbCrLf & "     meistens aber ""Durchmesser"" (Unicode = hex. 2300)"
Case Is = "C7"
        Symbolname = "INTERSECTION (Unicode = hex. 2229)"
Case Is = "C8"
        Symbolname = "UNION (Unicode = hex. 222A)"
Case Is = "C9"
        Symbolname = "SUPERSET OF (Unicode = hex. 2283)"
Case Is = "CA"
        Symbolname = "SUPERSET OF OR EQUAL TO (Unicode = hex. 2287)"
Case Is = "CB"
        Symbolname = "NOT A SUBSET OF (Unicode = hex. 2284)"
Case Is = "CC"
        Symbolname = "SUBSET OF (Unicode = hex. 2282)"
Case Is = "CD"
        Symbolname = "SUBSET OF OR EQUAL TO (Unicode = hex. 2286)"
Case Is = "CE"
        Symbolname = "ELEMENT OF (Unicode = hex. 2208)"
Case Is = "CF"
        Symbolname = "NOT AN ELEMENT OF (Unicode = hex. 2209)"
Case Is = "D0"
        Symbolname = "ANGLE (Unicode = hex. 2220)"
Case Is = "D1"
        Symbolname = "NABLA (Unicode = hex. 2207)"
Case Is = "D2"
        Symbolname = "REGISTERED SIGN (Unicode = hex. 00AE) mit Serifen"
Case Is = "D3"
        Symbolname = "COPYRIGHT SIGN (Unicode = hex. 00A9) mit Serifen"
Case Is = "D4"
        Symbolname = "TRADE MARK SIGN (Unicode = hex. 2122) mit Serifen"
Case Is = "D5"
        Symbolname = "N-ARY PRODUCT (Unicode = hex. 220F)"
Case Is = "D6"
        Symbolname = "SQUARE ROOT (Unicode = hex. 221A)"
Case Is = "D7"
        Symbolname = "DOT OPERATOR (Unicode = hex. 22C5)"
Case Is = "D8"
        Symbolname = "NOT SIGN (Unicode = hex. 00AC)"
Case Is = "D9"
        Symbolname = "LOGICAL AND (Unicode = hex. 2227)"
Case Is = "DA"
        Symbolname = "LOGICAL OR (Unicode = hex. 2228)"
Case Is = "DB"
        Symbolname = "LEFT RIGHT DOUBLE ARROW (Unicode = hex. 21D4)"
Case Is = "DC"
        Symbolname = "LEFTWARDS DOUBLE ARROW (Unicode = hex. 21D0)"
Case Is = "DD"
        Symbolname = "UPWARDS DOUBLE ARROW (Unicode = hex. 21D1)"
Case Is = "DE"
        Symbolname = "RIGHTWARDS DOUBLE ARROW (Unicode = hex. 21D2)"
Case Is = "DF"
        Symbolname = "DOWNWARDS DOUBLE ARROW (Unicode = hex. 21D3)"
Case Is = "E0"
        Symbolname = "LOZENGE (Unicode = hex. 25CA)"
Case Is = "E1"
        Symbolname = "LEFT-POINTING ANGLE BRACKET (Unicode = hex. 2329)"
Case Is = "E2"
```

```
            Symbolname = "REGISTERED SIGN, (Unicode = hex. 00AE) Sans Serif"
    Case Is = "E3"
            Symbolname = "COPYRIGHT SIGN, (Unicode = hex. 00A9) Sans Serif"
    Case Is = "E4"
            Symbolname = "TRADE MARK SIGN, (Unicode = hex. 2122) Sans Serif"
    Case Is = "E5"
            Symbolname = "N-ARY SUMMATION (Unicode = hex. 2211)"
    Case Is = "E6"
            Symbolname = "LEFT PARENTHESIS UPPER HOOK (Unicode = hex. 239B)"
    Case Is = "E7"
            Symbolname = "LEFT PARENTHESIS EXTENSION (Unicode = hex. 239C)"
    Case Is = "E8"
            Symbolname = "LEFT PARENTHESIS LOWER HOOK (Unicode = hex. 239D)"
    Case Is = "E9"
            Symbolname = "LEFT SQUARE BRACKET UPPER CORNER (Unicode = hex. 23A1)"
    Case Is = "EA"
            Symbolname = "LEFT SQUARE BRACKET EXTENSION (Unicode = hex. 23A2)"
    Case Is = "EB"
            Symbolname = "LEFT SQUARE BRACKET LOWER CORNER (Unicode = hex. 23A3)"
    Case Is = "EC"
            Symbolname = "LEFT CURLY BRACKET UPPER HOOK (Unicode = hex. 23A7)"
    Case Is = "ED"
            Symbolname = "LEFT CURLY BRACKET MIDDLE PIECE (Unicode = hex. 23A8)"
    Case Is = "EE"
            Symbolname = "LEFT CURLY BRACKET LOWER HOOK (Unicode = hex. 23A9)"
    Case Is = "EF"
            Symbolname = "CURLY BRACKET EXTENSION (Unicode = hex. 23AA)"
    Case Is = "F0"
            Symbolname = "(nicht definiert, bzw. Apple-Logo)"
    Case Is = "F1"
            Symbolname = "RIGHT-POINTING ANGLE BRACKET (Unicode = hex. 232A)"
    Case Is = "F2"
            Symbolname = "INTEGRAL (Unicode = hex. 222B)"
    Case Is = "F3"
            Symbolname = "TOP HALF INTEGRAL (Unicode = hex. 2320)"
    Case Is = "F4"
            Symbolname = "INTEGRAL EXTENSION (Unicode = hex. 23AE)"
    Case Is = "F5"
            Symbolname = "BOTTOM HALF INTEGRAL (Unicode = hex. 2321)"
    Case Is = "F6"
            Symbolname = "RIGHT PARENTHESIS UPPER HOOK (Unicode = hex. 239E)"
    Case Is = "F7"
            Symbolname = "RIGHT PARENTHESIS EXTENSION (Unicode = hex. 239F)"
    Case Is = "F8"
            Symbolname = "RIGHT PARENTHESIS LOWER HOOK (Unicode = hex. 23A0)"
    Case Is = "F9"
            Symbolname = "RIGHT SQUARE BRACKET UPPER CORNER (Unicode = hex. 23A4)"
    Case Is = "FA"
            Symbolname = "RIGHT SQUARE BRACKET EXTENSION (Unicode = hex. 23A5)"
    Case Is = "FB"
            Symbolname = "RIGHT SQUARE BRACKET LOWER CORNER (Unicode = hex. 23A6)"
    Case Is = "FC"
            Symbolname = "RIGHT CURLY BRACKET UPPER HOOK (Unicode = hex. 23AB)"
    Case Is = "FD"
            Symbolname = "RIGHT CURLY BRACKET MIDDLE PIECE (Unicode = hex. 23AC)"
    Case Is = "FE"
            Symbolname = "RIGHT CURLY BRACKET LOWER HOOK (Unicode = hex. 23AD)"
    Case Else
            Symbolname = "(kein Symbol-Zeichen festgelegt!)"
  End Select
End Function
' ---------------------------------------------------------------------------------------------
```

### 6.2.6 Unterfunktion zur Angabe der Bedeutung und der entsprechenden Unicode-Zeichen für Wingdingszeichen

Für die seltener vorkommenden Zeichen der Schriftart "Wingdings" verwendete ich eine Liste von Alan Wood [18], die ich aber inzwischen erweiterte.

```
Private Function Wingdingsname(Z1 As String) As String
' Erstellt von Dipl.-Ing. Hanna-Chris Gast, 5. März 2011, 12. Oktober 2011, Stand 19. Dezember
2014.
' Diese Funktion gibt englische Zeichenerklärung und teilweise auch entsprechende Unicodes an,
' wenn der (niedrige) hexadezimale Code eingegeben wird.
' Quelle (modifiziert):  http://www.alanwood.net/demos/wingdings.html.
'
' Inzwischen (19.12.2014) gibt es zu weiteren Wingdingszeichen entsprechende Unicode-Zeichen,
' die hier noch nicht erwähnt sind, sie sind aber in dem Makro für die Ersetzung von
' Symbol- und Wingdingszeichen im Makromodul "Symbol_Wingdings_in_Unicode" erfasst.
'
  Select Case Z1
     Case Is = "20"
          Wingdingsname = "SPACE (Unicode = hex. 0020)"
   Case Is = "20"
          Wingdingsname = "SPACE (Unicode = hex. 0020)"
   Case Is = "21"
          Wingdingsname = "PENCIL (Unicode = hex. 270F)"
   Case Is = "22"
          Wingdingsname = "BLACK SCISSORS (Unicode = hex. 2702)"
   Case Is = "23"
          Wingdingsname = "UPPER BLADE SCISSORS (Unicode = hex. 2701)"
   Case Is = "24"
          Wingdingsname = "EYEGLASSES (Unicode = hex. 1F453)"
   Case Is = "25"
          Wingdingsname = "BELL (Unicode = hex. 1F514)"
   Case Is = "26"
          Wingdingsname = "OPEN BOOK (Unicode = hex. 1F4D6)"
   Case Is = "27"
          Wingdingsname = "  (CANDLE)"
   Case Is = "28"
          Wingdingsname = "BLACK TELEPHONE (Unicode = hex. 260E)"
   Case Is = "29"
          Wingdingsname = "TELEPHONE LOCATION SIGN (Unicode = hex. 2706)"
   Case Is = "2A"
          Wingdingsname = "ENVELOPE (Unicode = hex. 2709)"
   Case Is = "2B"
          Wingdingsname = "ENVELOPE WITH ADDRESS AND STAMP"
   Case Is = "2C"
          Wingdingsname = "CLOSED MAILBOX WITH LOWERED FLAG (Unicode = hex. 1F4EA)"
   Case Is = "2D"
          Wingdingsname = "CLOSED MAILBOX WITH RAISED FLAG (Unicode = hex. 1F4EB)"
   Case Is = "2E"
          Wingdingsname = "OPEN MAILBOX WITH RAISED FLAG (Unicode = hex. 1F4EC)"
   Case Is = "2F"
          Wingdingsname = "OPEN MAILBOX WITH LOWERED FLAG (Unicode = hex. 1F4ED)"
   Case Is = "30"
          Wingdingsname = "FILE FOLDER (Unicode = hex. 1F4C1)"
   Case Is = "31"
          Wingdingsname = "OPEN FILE FOLDER (Unicode = hex. 1F4C2)"
   Case Is = "32"
          Wingdingsname = "PAGE FACING UP (Unicode = hex. 1F4C4)"
   Case Is = "33"
          Wingdingsname = "PRINTED PAGE"
   Case Is = "34"
          Wingdingsname = "STACK OF PRINTED PAGES"
   Case Is = "35"
```

```
        Wingdingsname = "FILING CABINET"
Case Is = "36"
        Wingdingsname = "HOURGLASS  (Unicode = hex. 231B)"
Case Is = "37"
        Wingdingsname = "KEYBOARD (Unicode = hex. 2328)"
Case Is = "38"
        Wingdingsname = "MOUSE"
Case Is = "39"
        Wingdingsname = "TRACKBALL"
Case Is = "3A"
        Wingdingsname = "PERSONAL COMPUTER (Unicode = hex. 1F4BB)"
Case Is = "3B"
        Wingdingsname = "HARD DISK"
Case Is = "3C"
        Wingdingsname = "FLOPPY DISK (Unicode = hex. 1F4BE)"
Case Is = "3D"
        Wingdingsname = "FLOPPY DISK"
Case Is = "3E"
        Wingdingsname = "TAPE DRIVE (Unicode = hex. 2707)"
Case Is = "3F"
        Wingdingsname = "WRITING HAND (Unicode = hex. 270D)"
Case Is = "40"
        Wingdingsname = "WRITING LEFT HAND"
Case Is = "41"
        Wingdingsname = "VICTORY HAND (Unicode = hex. 270C)"
Case Is = "42"
        Wingdingsname = "OK HAND SIGN (Unicode = hex. 1F44C)"
Case Is = "43"
        Wingdingsname = "THUMBS UP SIGN (Unicode = hex. 1F44D)"
Case Is = "44"
        Wingdingsname = "THUMBS DOWN SIGN (Unicode = hex. 1F44E)"
Case Is = "45"
        Wingdingsname = "WHITE LEFT POINTING INDEX (Unicode = hex. 261C)"
Case Is = "46"
        Wingdingsname = "WHITE RIGHT POINTING INDEX (Unicode = hex. 261E)"
Case Is = "47"
        Wingdingsname = "WHITE UP POINTING INDEX (Unicode = hex. 261D)"
Case Is = "48"
        Wingdingsname = "WHITE DOWN POINTING INDEX (Unicode = hex. 261F)"
Case Is = "49"
        Wingdingsname = "OPEN HAND"
Case Is = "4A"
        Wingdingsname = "WHITE SMILING FACE (Unicode = hex. 263A)"
Case Is = "4B"
        Wingdingsname = "NEUTRAL FACE (Unicode = hex. 1F610)"
Case Is = "4C"
        Wingdingsname = "WHITE FROWNING FACE (Unicode = hex. 2639)"
Case Is = "4D"
        Wingdingsname = "BOMB (Unicode = hex. 1F4A3)"
Case Is = "4E"
        Wingdingsname = "SKULL AND CROSSBONES (Unicode = hex. 2620)"
Case Is = "4F"
        Wingdingsname = "WHITE FLAG (Unicode = hex. 2690)"
Case Is = "50"
        Wingdingsname = "PENNANT ON POLE"
Case Is = "51"
        Wingdingsname = "AIRPLANE (Unicode = hex. 2708)"
Case Is = "52"
        Wingdingsname = "WHITE SUN WITH RAYS (Unicode = hex. 263C)"
Case Is = "53"
        Wingdingsname = "DROPLET (Unicode = hex. 1F4A7)"
Case Is = "54"
        Wingdingsname = "SNOWFLAKE (Unicode = hex. 2744)"
```

```
Case Is = "55"
        Wingdingsname = "WHITE LATIN CROSS"
Case Is = "56"
        Wingdingsname = "SHADOWED WHITE LATIN CROSS (Unicode = hex. 271E)"
Case Is = "57"
        Wingdingsname = "CELTIC CROSS"
Case Is = "58"
        Wingdingsname = "MALTESE CROSS (Unicode = hex. 2720)"
Case Is = "59"
        Wingdingsname = "STAR OF DAVID (Unicode = hex. 2721)"
Case Is = "5A"
        Wingdingsname = "STAR AND CRESCENT (Unicode = hex. 262A)"
Case Is = "5B"
        Wingdingsname = "YIN YANG (Unicode = hex. 262F)"
Case Is = "5C"
        Wingdingsname = "DEVANAGARI OM (Unicode = hex. 0950)"
Case Is = "5D"
        Wingdingsname = "WHEEL OF DHARMA (Unicode = hex. 2638)"
Case Is = "5E"
        Wingdingsname = "ARIES (Unicode = hex. 2648)"
Case Is = "5F"
        Wingdingsname = "TAURUS (Unicode = hex. 2649)"
Case Is = "60"
        Wingdingsname = "GEMINI (Unicode = hex. 264A)"
Case Is = "61"
        Wingdingsname = "CANCER (Unicode = hex. 264B)"
Case Is = "62"
        Wingdingsname = "LEO (Unicode = hex. 264C)"
Case Is = "63"
        Wingdingsname = "VIRGO (Unicode = hex. 264D)"
Case Is = "64"
        Wingdingsname = "LIBRA (Unicode = hex. 264E)"
Case Is = "65"
        Wingdingsname = "SCORPIO (Unicode = hex. 264F)"
Case Is = "66"
        Wingdingsname = "SAGITTARIUS (Unicode = hex. 2650)"
Case Is = "67"
        Wingdingsname = "CAPRICORN (Unicode = hex. 2651)"
Case Is = "68"
        Wingdingsname = "AQUARIUS (Unicode = hex. 2652)"
Case Is = "69"
        Wingdingsname = "PISCES (Unicode = hex. 2653)"
Case Is = "6A"
        Wingdingsname = "AMPERSAND (lower Case)  (Unicode = hex. 0026)"
Case Is = "6B"
        Wingdingsname = "AMPERSAND (ITALIC) (Unicode = hex. 0026)"
Case Is = "6C"
        Wingdingsname = "BLACK CIRCLE (Unicode = hex. 25CF)"
Case Is = "6D"
        Wingdingsname = "SHADOWED WHITE CIRCLE (Unicode = hex. 274D)"
Case Is = "6E"
        Wingdingsname = "BLACK SQUARE (Unicode = hex. 25A0)"
Case Is = "6F"
        Wingdingsname = "WHITE SQUARE (Unicode = hex. 25A1)"
Case Is = "70"
        Wingdingsname = "BOLD WHITE SQUARE"
Case Is = "71"
        Wingdingsname = "LOWER RIGHT SHADOWED WHITE SQUARE (Unicode = hex. 2751)"
Case Is = "72"
        Wingdingsname = "UPPER RIGHT SHADOWED WHITE SQUARE (Unicode = hex. 2752)"
Case Is = "73"
        Wingdingsname = "BLACK MEDIUM LOZENGE (Unicode = hex. 2B27)"
Case Is = "74"
```

```
        Wingdingsname = "BLACK LOZENGE (Unicode = hex. 29EB)"
Case Is = "75"
        Wingdingsname = "BLACK DIAMOND (Unicode = hex. 25C6)"
Case Is = "76"
        Wingdingsname = "BLACK DIAMOND MINUS WHITE X (Unicode = hex. 2756)"
Case Is = "77"
        Wingdingsname = "BLACK MEDIUM DIAMOND (Unicode = hex. 2B25)"
Case Is = "78"
        Wingdingsname = "X IN A RECTANGLE BOX (Unicode = hex. 2327)"
Case Is = "79"
        Wingdingsname = "APL FUNCTIONAL SYMBOL QUAD UP CARET (Unicode = hex. 2353)"
Case Is = "7A"
        Wingdingsname = "PLACE OF INTEREST SIGN (Unicode = hex. 2318)"
Case Is = "7B"
        Wingdingsname = "WHITE FLORETTE (Unicode = hex. 2740)"
Case Is = "7C"
        Wingdingsname = "BLACK FLORETTE (Unicode = hex. 273F)"
Case Is = "7D"
        Wingdingsname = "HEAVY DOUBLE TURNED COMMA QUOTATION MARK " _
        & "ORNAMENT (Unicode = hex. 275D)"
Case Is = "7E"
        Wingdingsname = "HEAVY DOUBLE COMMA QUOTATION MARK ORNAMENT " _
        & "(Unicode = hex. 275E)"
Case Is = "7F"
        Wingdingsname = "(WHITE VERTICAL RECTANGLE)   (Unicode = hex. 25AF)"
Case Is = "80"
        Wingdingsname = "CIRCLED DIGIT ZERO (Unicode = hex. 24EA)"
Case Is = "81"
        Wingdingsname = "CIRCLED DIGIT ONE (Unicode = hex. 2460)"
Case Is = "82"
        Wingdingsname = "CIRCLED DIGIT TWO (Unicode = hex. 2461)"
Case Is = "83"
        Wingdingsname = "CIRCLED DIGIT THREE (Unicode = hex. 2462)"
Case Is = "84"
        Wingdingsname = "CIRCLED DIGIT FOUR (Unicode = hex. 2463)"
Case Is = "85"
        Wingdingsname = "CIRCLED DIGIT FIVE (Unicode = hex. 2464)"
Case Is = "86"
        Wingdingsname = "CIRCLED DIGIT SIX (Unicode = hex. 2465)"
Case Is = "87"
        Wingdingsname = "CIRCLED DIGIT SEVEN (Unicode = hex. 2466)"
Case Is = "88"
        Wingdingsname = "CIRCLED DIGIT EIGHT (Unicode = hex. 2467)"
Case Is = "89"
        Wingdingsname = "CIRCLED DIGIT NINE (Unicode = hex. 2468)"
Case Is = "8A"
        Wingdingsname = "CIRCLED NUMBER TEN (Unicode = hex. 2469)"
Case Is = "8B"
        Wingdingsname = "NEGATIVE CIRCLED DIGIT ZERO (Unicode = hex. 24FF)"
Case Is = "8C"
        Wingdingsname = "DINGBAT NEGATIVE CIRCLED DIGIT ONE (Unicode = hex. 2776)"
Case Is = "8D"
        Wingdingsname = "DINGBAT NEGATIVE CIRCLED DIGIT TWO (Unicode = hex. 2777)"
Case Is = "8E"
        Wingdingsname = "DINGBAT NEGATIVE CIRCLED DIGIT THREE (Unicode = hex. 2778)"
Case Is = "8F"
        Wingdingsname = "DINGBAT NEGATIVE CIRCLED DIGIT FOUR (Unicode = hex. 2779)"
Case Is = "90"
        Wingdingsname = "DINGBAT NEGATIVE CIRCLED DIGIT FIVE (Unicode = hex. 277A)"
Case Is = "91"
        Wingdingsname = "DINGBAT NEGATIVE CIRCLED DIGIT SIX (Unicode = hex. 277B)"
Case Is = "92"
        Wingdingsname = "DINGBAT NEGATIVE CIRCLED DIGIT SEVEN (Unicode = hex. 277C)"
```

```
Case Is = "93"
        Wingdingsname = "DINGBAT NEGATIVE CIRCLED DIGIT EIGHT (Unicode = hex. 277D)"
Case Is = "94"
        Wingdingsname = "DINGBAT NEGATIVE CIRCLED DIGIT NINE (Unicode = hex. 277E)"
Case Is = "95"
        Wingdingsname = "DINGBAT NEGATIVE CIRCLED NUMBER TEN (Unicode = hex. 277F)"
Case Is = "96"
        Wingdingsname = "BUD AND LEAF NORTH EAST"
Case Is = "97"
        Wingdingsname = "BUD AND LEAF NORTH WEST"
Case Is = "98"
        Wingdingsname = "BUD AND LEAF SOUTH WEST"
Case Is = "99"
        Wingdingsname = "BUD AND LEAF SOUTH EAST"
Case Is = "9A"
        Wingdingsname = "BOLD VINE LEAF NORTH EAST"
Case Is = "9B"
        Wingdingsname = "BOLD VINE LEAF NORTH WEST"
Case Is = "9C"
        Wingdingsname = "BOLD VINE LEAF SOUTH WEST"
Case Is = "9D"
        Wingdingsname = "BOLD VINE LEAF SOUTH EAST"
Case Is = "9E"
        Wingdingsname = "MIDDLE DOT (Unicode = hex. 00B7)"
Case Is = "9F"
        Wingdingsname = "BULLET (Unicode = hex. 2022)"
Case Is = "A0"
        Wingdingsname = "BLACK SMALL SQUARE   (Unicode = hex. 25AA)"
Case Is = "A1"
        Wingdingsname = "WHITE CIRCLE (Unicode = hex. 25CB)"
Case Is = "A2"
        Wingdingsname = "BOLD WHITE CIRCLE"
Case Is = "A3"
        Wingdingsname = "EXTRA BOLD WHITE CIRCLE"
Case Is = "A4"
        Wingdingsname = "FISHEYE (Unicode = hex. 25C9)"
Case Is = "A5"
        Wingdingsname = "BULLSEYE (Unicode = hex. 25CE)"
Case Is = "A6"
        Wingdingsname = "ähnlich SHADOWED WHITE CIRCLE (Unicode = hex. 274D)"
Case Is = "A7"
        Wingdingsname = "BLACK SMALL SQUARE 7 (Unicode = hex. 25AA)"
Case Is = "A8"
        Wingdingsname = "WHITE MEDIUM SQUARE (Unicode = hex. 25FB)"
Case Is = "A9"
        Wingdingsname = "BLACK THREE POINTED STAR"
Case Is = "AA"
        Wingdingsname = "BLACK FOUR POINTED STAR (Unicode = hex. 2726)"
Case Is = "AB"
        Wingdingsname = "BLACK STAR (Unicode = hex. 2605)"
Case Is = "AC"
        Wingdingsname = "SIX POINTED BLACK STAR (Unicode = hex. 2736)"
Case Is = "AD"
        Wingdingsname = "EIGHT POINTED BLACK STAR (Unicode = hex. 2734)"
Case Is = "AE"
        Wingdingsname = "TWELVE POINTED BLACK STAR (Unicode = hex. 2739)"
Case Is = "AF"
        Wingdingsname = "EIGHT POINTED PINWHEEL STAR (Unicode = hex. 2735)"
Case Is = "B0"
        Wingdingsname = "SQUARE REGISTER MARK"
Case Is = "B1"
        Wingdingsname = "POSITION INDICATOR  (Unicode = hex. 2316)"
Case Is = "B2"
```

```
        Wingdingsname = "WHITE CONCAVE-SIDED DIAMOND (Unicode = hex. 27E1)"
Case Is = "B3"
        Wingdingsname = "SQUARE LOZENGE (Unicode = hex. 2311)"
Case Is = "B4"
        Wingdingsname = "QUESTION MARK IN WHITE DIAMOND"
Case Is = "B5"
        Wingdingsname = "CIRCLED WHITE STAR (Unicode = hex. 272A)"
Case Is = "B6"
        Wingdingsname = "SHADOWED WHITE STAR (Unicode = hex. 2730)"
Case Is = "B7"
        Wingdingsname = "CLOCK FACE ONE O CLOCK (Unicode = hex. 1F550)"
Case Is = "B8"
        Wingdingsname = "CLOCK FACE TWO O CLOCK (Unicode = hex. 1F551)"
Case Is = "B9"
        Wingdingsname = "CLOCK FACE THREE O CLOCK (Unicode = hex. 1F552)"
Case Is = "BA"
        Wingdingsname = "CLOCK FACE FOUR O CLOCK (Unicode = hex. 1F553)"
Case Is = "BB"
        Wingdingsname = "CLOCK FACE FIVE O CLOCK (Unicode = hex. 1F554)"
Case Is = "BC"
        Wingdingsname = "CLOCK FACE SIX O CLOCK (Unicode = hex. 1F555)"
Case Is = "BD"
        Wingdingsname = "CLOCK FACE SEVEN O CLOCK (Unicode = hex. 1F556)"
Case Is = "BE"
        Wingdingsname = "CLOCK FACE EIGHT O CLOCK (Unicode = hex. 1F557)"
Case Is = "BF"
        Wingdingsname = "CLOCK FACE NINE O CLOCK (Unicode = hex. 1F558)"
Case Is = "C0"
        Wingdingsname = "CLOCK FACE TEN O CLOCK (Unicode = hex. 1F559)"
Case Is = "C1"
        Wingdingsname = "CLOCK FACE ELEVEN O CLOCK (Unicode = hex. 1F55A)"
Case Is = "C2"
        Wingdingsname = "CLOCK FACE TWELVE O CLOCK (Unicode = hex. 1F55B)"
Case Is = "C3"
        Wingdingsname = "(WHITE ARROW POINTING DOWNWARDS THEN CURVING " _
        & "LEFTWARDS"
Case Is = "C4"
        Wingdingsname = "WHITE ARROW POINTING DOWNWARDS THEN CURVING " _
        & "RIGHTWARDS"
Case Is = "C5"
        Wingdingsname = "WHITE ARROW POINTING UPWARDS THEN CURVING " _
        & "LEFTWARDS"
Case Is = "C6"
        Wingdingsname = "WHITE ARROW POINTING UPWARDS THEN CURVING " _
        & "RIGHTWARDS"
Case Is = "C7"
        Wingdingsname = "WHITE ARROW POINTING LEFTWARDS THEN CURVING " _
        & "UPWARDS"
Case Is = "C8"
        Wingdingsname = "WHITE ARROW POINTING RIGHTWARDS THEN CURVING " _
        & "UPWARDS"
Case Is = "C9"
        Wingdingsname = "WHITE ARROW POINTING LEFTWARDS THEN CURVING " _
        & "DOWNWARDS"
Case Is = "CA"
        Wingdingsname = "WHITE ARROW POINTING RIGHTWARDS THEN CURVING " _
        & "DOWNWARDS"
Case Is = "CB"
        Wingdingsname = "QUILT SQUARE 2"
Case Is = "CC"
        Wingdingsname = "BLACK QUILT SQUARE 2"
Case Is = "CD"
        Wingdingsname = "LEAF COUNTERCLOCKWISE SOUTH WEST"
```

```
Case Is = "CE"
        Wingdingsname = "LEAF COUNTERCLOCKWISE NORTH WEST"
Case Is = "CF"
        Wingdingsname = "LEAF COUNTERCLOCKWISE SOUTH EAST"
Case Is = "D0"
        Wingdingsname = "LEAF COUNTERCLOCKWISE NORTH EAST"
Case Is = "D1"
        Wingdingsname = "LEAF NORTH WEST"
Case Is = "D2"
        Wingdingsname = "LEAF SOUTH WEST"
Case Is = "D3"
        Wingdingsname = "LEAF NORTH EAST"
Case Is = "D4"
        Wingdingsname = "LEAF SOUTH EAST"
Case Is = "D5"
        Wingdingsname = "ERASE TO THE LEFT (Unicode = hex. 232B)"
Case Is = "D6"
        Wingdingsname = "ERASE TO THE RIGHT (Unicode = hex. 2326)"
Case Is = "D7"
        Wingdingsname = "THREE-D TOP-LIGHTED LEFTWARDS ARROWHEAD"
Case Is = "D8"
        Wingdingsname = "THREE-D TOP-LIGHTED RIGHTWARDS ARROWHEAD " _
        & "(Unicode = hex. 27A2)"
Case Is = "D9"
        Wingdingsname = "THREE-D RIGHT-LIGHTED UPWARDS ARROWHEAD " _
        & " "
Case Is = "DA"
        Wingdingsname = "THREE-D LEFT-LIGHTED DOWNWARDS ARROWHEAD " _
        & " "
Case Is = "DB"
        Wingdingsname = "CIRCLED HEAVY WHITE LEFTWARDS ARROW"
Case Is = "DC"
        Wingdingsname = "CIRCLED HEAVY WHITE RIGHTWARDS ARROW " _
        & "(Unicode = hex. 27B2)"
Case Is = "DD"
        Wingdingsname = "CIRCLED HEAVY WHITE UPWARDS ARROW"
Case Is = "DE"
        Wingdingsname = "CIRCLED HEAVY WHITE DOWNWARDS ARROW"
Case Is = "DF"
        Wingdingsname = "WIDE-HEADED LEFTWARDS ARROW"
Case Is = "E0"
        Wingdingsname = "WIDE-HEADED RIGHTWARDS ARROW"
Case Is = "E1"
        Wingdingsname = "WIDE-HEADED UPWARDS ARROW"
Case Is = "E2"
        Wingdingsname = "WIDE-HEADED DOWNWARDS ARROW"
Case Is = "E3"
        Wingdingsname = "WIDE-HEADED NORTH WEST ARROW"
Case Is = "E4"
        Wingdingsname = "WIDE-HEADED NORTH EAST ARROW"
Case Is = "E5"
        Wingdingsname = "WIDE-HEADED SOUTH WEST ARROW"
Case Is = "E6"
        Wingdingsname = "WIDE-HEADED SOUTH EAST ARROW"
Case Is = "E7"
        Wingdingsname = "HEAVY WIDE-HEADED LEFTWARDS ARROW"
Case Is = "E8"
        Wingdingsname = "HEAVY WIDE-HEADED RIGHTWARDS ARROW (Unicode = hex. 2794)"
Case Is = "E9"
        Wingdingsname = "HEAVY WIDE-HEADED UPWARDS ARROW"
Case Is = "EA"
        Wingdingsname = "HEAVY WIDE-HEADED DOWNWARDS ARROW"
Case Is = "EB"
```

```
            Wingdingsname = "HEAVY WIDE-HEADED NORTH WEST ARROW"
    Case Is = "EC"
            Wingdingsname = "HEAVY WIDE-HEADED NORTH EAST ARROW"
    Case Is = "ED"
            Wingdingsname = "HEAVY WIDE-HEADED SOUTH WEST ARROW"
    Case Is = "EE"
            Wingdingsname = "HEAVY WIDE-HEADED SOUTH EAST ARROW"
    Case Is = "EF"
            Wingdingsname = "LEFTWARDS WHITE ARROW (Unicode = hex. 21E6)"
    Case Is = "F0"
            Wingdingsname = "RIGHTWARDS WHITE ARROW (Unicode = hex. 21E8)"
    Case Is = "F1"
            Wingdingsname = "UPWARDS WHITE ARROW (Unicode = hex. 21E7)"
    Case Is = "F2"
            Wingdingsname = "DOWNWARDS WHITE ARROW (Unicode = hex. 21E9)"
    Case Is = "F3"
            Wingdingsname = "LEFT RIGHT WHITE ARROW (Unicode = hex. 2B04)"
    Case Is = "F4"
            Wingdingsname = "UP DOWN WHITE ARROW (Unicode = hex. 21F3)"
    Case Is = "F5"
            Wingdingsname = "NORTH WEST WHITE ARROW (Unicode = hex. 2B01)"
    Case Is = "F6"
            Wingdingsname = "NORTH EAST WHITE ARROW (Unicode = hex. 2B00)"
    Case Is = "F7"
            Wingdingsname = "SOUTH WEST WHITE ARROW (Unicode = hex. 2B03)"
    Case Is = "F8"
            Wingdingsname = "SOUTH EAST WHITE ARROW (Unicode = hex. 2B02)"
    Case Is = "F9"
            Wingdingsname = "WHITE RECTANGLE (Unicode = hex. 25AD)"
    Case Is = "FA"
            Wingdingsname = "WHITE SMALL SQUARE (Unicode = hex. 25AB)"
    Case Is = "FB"
            Wingdingsname = "BALLOT X (Unicode = hex. 2717)"
    Case Is = "FC"
            Wingdingsname = "CHECK MARK (Unicode = hex. 2713)"
    Case Is = "FD"
            Wingdingsname = "BALLOT BOX WITH X (FONT-WEIGHT: BOLD) (Unicode = hex. 2612)"
    Case Is = "FE"
            Wingdingsname = "BALLOT BOX WITH CHECK (FONT-WEIGHT: BOLD) " _
            & "(Unicode = hex. 2611)"
    Case Is = "FF"
            Wingdingsname = "WINDOWS LOGO"
    Case Else
            Wingdingsname = "(KEIN WINGDINGS-ZEICHEN FESTGELEGT!)"
  End Select
End Function
' ---------------------------------------------------------------------------------------------
```

## 6.3 Auswertung und Tests

### 6.3.1 Testtabelle mit griechischer Buchstaben

Bei der Erzeugung eines griechischen Buchstaben mit Akzenten gibt es mehrere Möglichkeiten, die das von mir entwickelte Analyse-Makro aus Abschnitt 6.2 alle erkennt. Die folgende TabelleTabelle 6. gibt einen Vergleich zwischen der Funktion mit Alt + C von Word und dem hier neu entwickelten Analyse-Makro. Der eingebaute Befehl in Word 2003 mit Alt + C reagiert auf Zeichen der Windows-Sonderschriftarten wie "Symbol" und "Wingdings" unterschiedlich, je nach dem, wie das Zeichen ursprünglich erzeugt wurde. Teilweise erhält man mit der eingebauten Word-Funktion "Alt + C" überhaupt kein brauchbares Ergebnis, während das neue Makro hier alles analysieren kann. Das neue Makro zeigt dagegen unabhängig davon, wie das Zeichen der Windows-Sonderschriftart erzeugt wurde, in beiden Fällen einheitlich denselben Code an. Lediglich bei der Schriftart-Anzeige merkt man einen kleinen Unterschied. Damit ist das neue Analyse-Makro auch für Word 2007 noch von Vorteil.

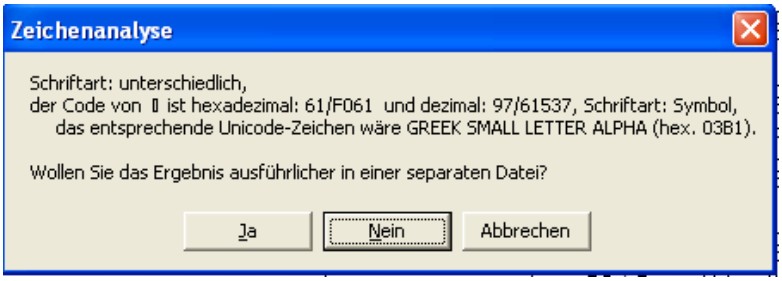
**Tabelle 6.3: Unterschiedliche Erzeugung von "Alpha" und "Alpha mit Akut"**

| Zeichen | Beschreibung | Quelltext | Word-Funktion Alt + C | Bestimmung mit dem neuen Makro |
|---|---|---|---|---|
| α | Alpha in der Sonderschrift "Symbol"<br>("Einfügen → Symbol → Schriftart Symbol") | "( "<br>= hex.<br>28<br>20 | (keine Reaktion) | hex. 61/F061<br>(dez. 97/61537) |
| α | Alpha in der Sonderschrift "Symbol"<br>(Buchstabe "a",<br>danach Schriftart = "Symbol") | "a ð"<br>= hex.<br>61<br>F0 | Φ061 | hex. 61/F061<br>(dez. 97/61537) |
| ά | Alpha in Schriftart "Symbol"<br>+ kombinierendem Akut in "Times New Roman" | hex<br>28<br>20<br>01<br>03 | α0301 | hex. 61/F061 + 0301<br>(dez. 97/61537 + 769) |
| α | Griechischer Kleinbuchstabe Alpha<br>(Times New Roman) | hex.<br>B1<br>03 | 03B1 | hex. 03B1<br>(dez. 945) |
| ά | Griechischer Kleinbuchstabe Alpha<br>mit kombinierenden Akut<br>(DejaVu Serif) | hex.<br>B1<br>03<br>01<br>03 | α0301 | hex. 03B1 + 0301<br>(dez.945 + 769) |
| ά | Griechischer Kleinbuchstabe Alpha mit Tonos | hex.<br>AC<br>03 | 03AC | hex. 03AC<br>(dez. 940) |
| ά | Griechischer Kleinbuchstabe Alpha mit Akut | hex.<br>71<br>1F | 1F71 | hex. 1F71<br>(dez. 8049) |

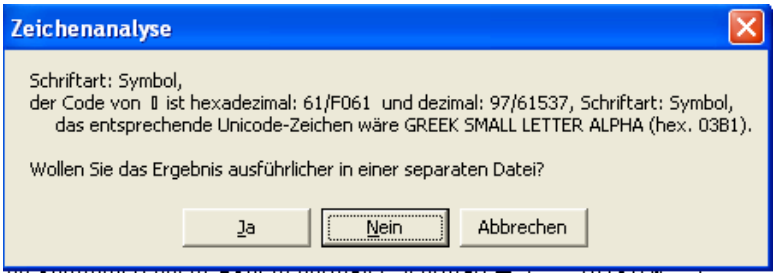
Zur Prüfung der Makros dieses Aufsatzes verwendete ich Listen mit Zeichen von den Steuerzeichen wie Tabulator und Absatzmarke, lateinische, griechische, hebräische und chinesische Schriftzeichen bis hin zu Hieroglyphen (siehe auch Abschnitt 4.2), die ich aber hier nicht alle angeben will.

### 6.3.2 Beispiele der Ergebnisanzeige des Analyse-Makros mit griechischen, hebräischen und arabischen Zeichen

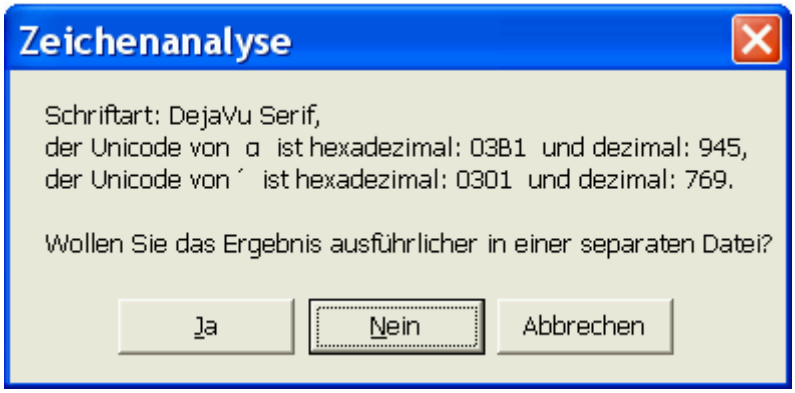
Bild 6.1 zeigt Screenshots mit einigen Ergebnissen zu dem Alpha aus Tabelle 6.3 in einer "MessageBoxW":



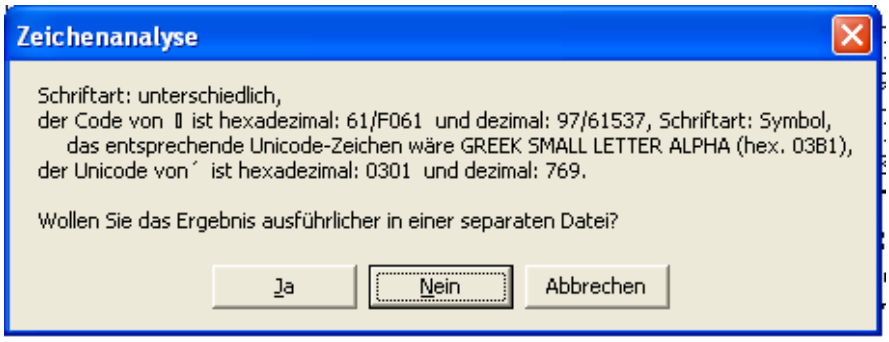

a) Kleines Alpha *("Einfügen → Symbol → Schriftart Symbol")*




b) Kleinen Alpha *(Buchstabe "a", danach Schriftart = "Symbol")*




c) kleines Alpha in einer Unicode-Schriftart mit kombinierenden Akut




d) kleines Alpha in der Windows-Sonderschriftart "Symbol" mit Akut

**Bild 6.1: Screenshots des Analyse-Makros in einer Message-Box**

Die Anzeige der Unicode-Namen und (bei höheren Unicode-Ebenen) der Surrogate erfolgt erst in der ausführlichen "separaten Datei", siehe folgende Bilder.

Das folgende Bild 6.2 zeigt zwei Beispiele für die ausführliche Analyse in einer separaten Datei. Hier werden auch die (englischen) Namen der Unicode-Zeichen angegeben:

Das·markierte·Zeichen·bzw.·die·Zeichenfolge·ά·setzt·sich·wie·folgt·zusammen:¶
¶
Der·Unicode·von·α·ist·hexadezimal:·03B1·und·dezimal:·945,·Schriftart:°DejaVu·Serif·
der°genormte°englische°Zeichenname°ist:°GREEK·SMALL·LETTER·ALPHA.·¶
Der·Unicode·von·́·ist·hexadezimal:·0301·und·dezimal:·769,·Schriftart:°DejaVu·Serif·
der°genormte°englische°Zeichenname°ist:°COMBINING·ACUTE·ACCENT.·¶
¶
Die·englischen·Namen·aller·Unicode-Schriftzeichen·sind·in·ISO/IEC·10646·genormt·und·weitgehend·
in·http://unicode.org/Public/UNIDATA/UnicodeData.txt↵
oder·über·http://www.unicode.org/charts·zu·finden.¶
¶
Die·Rechenzeit·war·0,469·Sekunden.¶

**a) Alpha in Unicode-Schriftart mit kombinierendem Akzent, wie in Bild 6.1c**

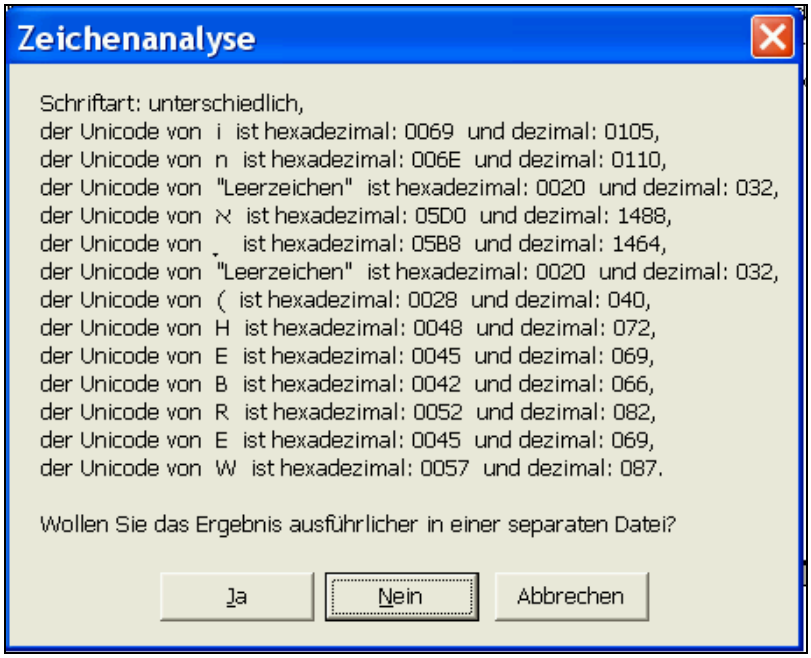
Das·markierte·Zeichen·bzw.·die·Zeichenfolge· ·setzt·sich·wie·folgt·zusammen:¶
¶
Der·Unicode·des·Zeichens·höherer·Ebene· ·ist·hexadezimal:·13050·und·dezimal:·77904,↵
(der·Unicode·des·Surrogat·1·ist·hexadezimal:·D80C·und·dezimal:·55308,·↵
der·Unicode·des·Surrogat·2·ist·hexadezimal:·DC50·und·dezimal:·56400),·↵
der°genormte°englische°Zeichenname°ist:°EGYPTIAN·HIEROGLYPH·B001,·↵
die·Schriftart·ist·Gardiner.·¶
¶
Die·englischen·Namen·aller·Unicode-Schriftzeichen·sind·in·ISO/IEC·10646·genormt·und·weitgehend·
in·http://unicode.org/Public/UNIDATA/UnicodeData.txt↵
oder·über·http://www.unicode.org/charts·zu·finden.¶
¶
Die·Rechenzeit·war·0,641·Sekunden.¶

**b) Hieroglyphe (Unicode-Ebene 1)**

**Bild 6.2: Screenshot der Ergebnisse des Analyse-Makros in einer "separaten Datei"**

Das folgende Bild 6.3 zeigt die Analyse von mehreren Buchstaben gleichzeitig anhand einiger **hebräischer** und **lateinischer** Buchstaben gleichzeitig. Man beachte die unterschiedlichen Schreibrichtungen der Schriften. Bei hebräischen und arabischen Schriftzeichen verläuft die Schreibrichtung von rechts nach links, während bei lateinischen Buchstaben die Schreibrichtung von links nach rechts geht. Dies ist in diesem Makro berücksichtigt. Bei hebräischen Buchstaben mit kombinierenden Zeichen wie in אָ *(HEBREW LETTER ALEF mit HEBREW POINT QAMATS)* versagen die Bordmittel von Word 2000 und Word 2003 völlig (Beispiel aus [17]). Hier erkennt mein Makro, dass es sich bei dem hebräischen Buchstaben um zwei einzelne Zeichen handelt, siehe Bild 6.3.

Zu erkennen ist hier auch, dass Leerzeichen zum besseren Verständnis durch das Wort "Leerzeichen" ersetzt werden.

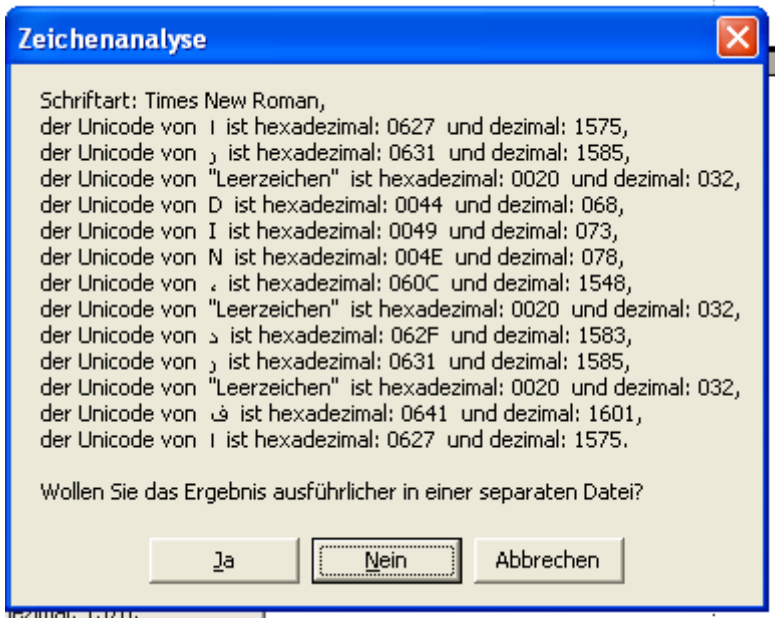



Das·markierte·Zeichen·bzw.·die·Zeichenfolge·in·אָ·*(HEBREW*·setzt·sich·wie·folgt·zusammen:¶
¶
Der·Unicode·von··**i**··ist·hexadezimal:·0069··und·dezimal:·00105,·Schriftart:°Times·New·Roman,··der°genormte°englische°Zeichenname°ist:°LATIN·SMALL·LETTER·I.·¶
Der·Unicode·von··**n**··ist·hexadezimal:·006E··und·dezimal:·00110,·Schriftart:°Times·New·Roman,··der°genormte°englische°Zeichenname°ist:°LATIN·SMALL·LETTER·N.·¶
Der·Unicode·von··**"Leerzeichen"**··ist·hexadezimal:·0020··und·dezimal:·0032,·Schriftart:°Times·New·Roman,··der°genormte°englische°Zeichenname°ist:°SPACE.·¶
Der·Unicode·von··**א**··ist·hexadezimal:·05D0··und·dezimal:·1488,·Schriftart:°Microsoft·Sans·Serif,··der°genormte°englische°Zeichenname°ist:°HEBREW·LETTER·ALEF.·¶
Der·Unicode·von··**ָ**··ist·hexadezimal:·05B8··und·dezimal:·1464,·Schriftart:°Microsoft·Sans·Serif,··der°genormte°englische°Zeichenname°ist:°HEBREW·POINT·QAMATS.·¶
Der·Unicode·von··**"Leerzeichen"**··ist·hexadezimal:·0020··und·dezimal:·0032,·Schriftart:°Times·New·Roman,··der°genormte°englische°Zeichenname°ist:°SPACE.·¶
Der·Unicode·von··**(**··ist·hexadezimal:·0028··und·dezimal:·0040,·Schriftart:°Times·New·Roman,··der°genormte°englische°Zeichenname°ist:°LEFT·PARENTHESIS.·¶
Der·Unicode·von··**H**··ist·hexadezimal:·0048··und·dezimal:·0072,·Schriftart:°Times·New·Roman,··der°genormte°englische°Zeichenname°ist:°LATIN·CAPITAL·LETTER·H.·¶
Der·Unicode·von··**E**··ist·hexadezimal:·0045··und·dezimal:·0069,·Schriftart:°Times·New·Roman,··der°genormte°englische°Zeichenname°ist:°LATIN·CAPITAL·LETTER·E.·¶
Der·Unicode·von··**B**··ist·hexadezimal:·0042··und·dezimal:·0066,·Schriftart:°Times·New·Roman,··der°genormte°englische°Zeichenname°ist:°LATIN·CAPITAL·LETTER·B.·¶
Der·Unicode·von··**R**··ist·hexadezimal:·0052··und·dezimal:·0082,·Schriftart:°Times·New·Roman,··der°genormte°englische°Zeichenname°ist:°LATIN·CAPITAL·LETTER·R.·¶
Der·Unicode·von··**E**··ist·hexadezimal:·0045··und·dezimal:·0069,·Schriftart:°Times·New·Roman,··der°genormte°englische°Zeichenname°ist:°LATIN·CAPITAL·LETTER·E.·¶
Der·Unicode·von··**W**··ist·hexadezimal:·0057··und·dezimal:·0087,·Schriftart:°Times·New·Roman,··der°genormte°englische°Zeichenname°ist:°LATIN·CAPITAL·LETTER·W.·¶
¶
Die·englischen·Namen·aller·Unicode-Schriftzeichen·sind·in·ISO/IEC·10646·genormt··und·weitgehend·in·http://unicode.org/Public/UNIDATA/UnicodeData.txt↵
oder·über·http://www.unicode.org/charts·zu·finden.¶
¶
Die·Rechenzeit·war·0,703·Sekunden.¶

**Bild 6.3: Screenshot des Ergebnisses für die Analyse einer Zeichenkette mit hebräischen und lateinischen Buchstaben**

Das folgende Beispiel enthält arabische und lateinische Buchstaben. Es handelt sich um einen willkürlichen Ausschnitt aus einer arabischen Seite über das DIN:

| Normung، اختصـــار بـــه DIN، فـــارس د ی در ی‌محاوره ن) |
|---|

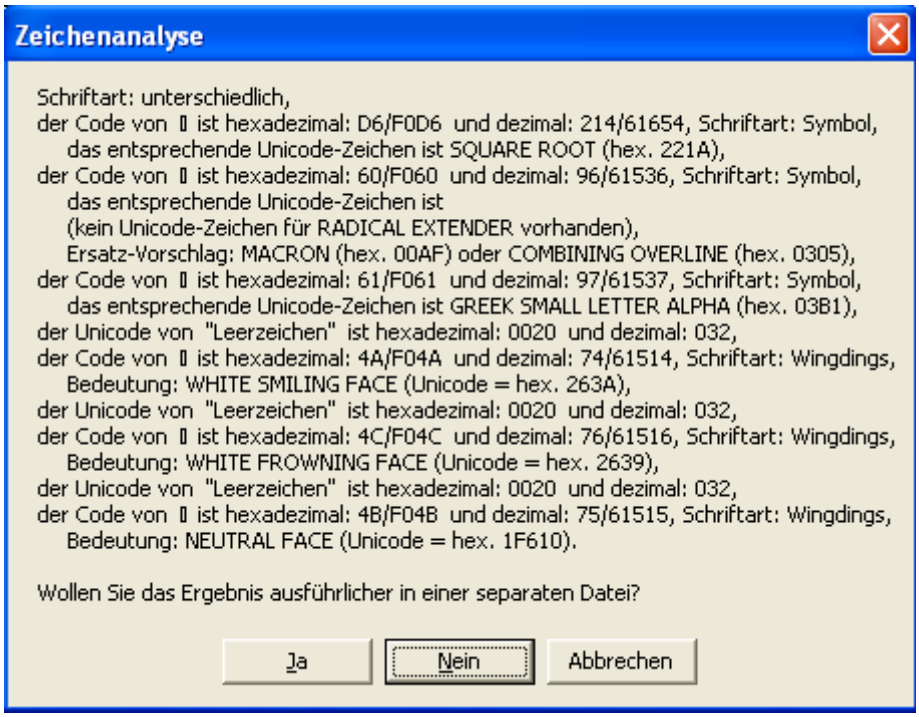



Das markierte Zeichen bzw. die Zeichenfolge ان DIN فـــار ر¶
setzt sich wie folgt zusammen:¶
¶
Der Unicode von ا ist hexadezimal: 0627 und dezimal: 1575, Schriftart: Times New Roman,
der genormte englische Zeichenname ist: ARABIC LETTER ALEF .¶
Der Unicode von ر ist hexadezimal: 0631 und dezimal: 1585, Schriftart: Times New Roman,
der genormte englische Zeichenname ist: ARABIC LETTER REH .¶
Der Unicode von **"Leerzeichen"** ist hexadezimal: 0020 und dezimal: 032, Schriftart: Times New Roman, der genormte englische Zeichenname ist: SPACE .¶
Der Unicode von **D** ist hexadezimal: 0044 und dezimal: 068, Schriftart: Times New Roman,
der genormte englische Zeichenname ist: LATIN CAPITAL LETTER D .¶
Der Unicode von **I** ist hexadezimal: 0049 und dezimal: 073, Schriftart: Times New Roman,
der genormte englische Zeichenname ist: LATIN CAPITAL LETTER I .¶
Der Unicode von **N** ist hexadezimal: 004E und dezimal: 078, Schriftart: Times New Roman,
der genormte englische Zeichenname ist: LATIN CAPITAL LETTER N .¶
Der Unicode von ، ist hexadezimal: 060C und dezimal: 1548, Schriftart: Times New Roman,
der genormte englische Zeichenname ist: ARABIC COMMA .¶
Der Unicode von **"Leerzeichen"** ist hexadezimal: 0020 und dezimal: 032, Schriftart: Times New Roman, der genormte englische Zeichenname ist: SPACE .¶
Der Unicode von د ist hexadezimal: 062F und dezimal: 1583, Schriftart: Times New Roman,
der genormte englische Zeichenname ist: ARABIC LETTER DAL .¶
Der Unicode von ر ist hexadezimal: 0631 und dezimal: 1585, Schriftart: Times New Roman,
der genormte englische Zeichenname ist: ARABIC LETTER REH .¶
Der Unicode von **"Leerzeichen"** ist hexadezimal: 0020 und dezimal: 032, Schriftart: Times New Roman, der genormte englische Zeichenname ist: SPACE .¶
Der Unicode von ف ist hexadezimal: 0641 und dezimal: 1601, Schriftart: Times New Roman,
der genormte englische Zeichenname ist: ARABIC LETTER FEH .¶
Der Unicode von ا ist hexadezimal: 0627 und dezimal: 1575, Schriftart: Times New Roman,
der genormte englische Zeichenname ist: ARABIC LETTER ALEF .¶
¶
¶
Die englischen Namen aller Unicode-Schriftzeichen sind in ISO/IEC 10646 genormt und weitgehend in http://unicode.org/Public/UNIDATA/UnicodeData.txt
oder über http://www.unicode.org/charts zu finden.¶
¶
Die Rechenzeit war 1,172 Sekunden.¶

**Bild 6.4: Screenshots des Ergebnisses für die Analyse einer Zeichenkette mit arabischen und lateinischen Buchstaben**

### 6.3.3 Anzeige bei Symbol- und Wingdingszeichen

Bild 6.5 zeigt das Analyse-Ergebnis für Zeichen der Schrifarten Symbol und Wingdings. Der Test erfolgte mit einer Wurzel, einem kleinen Alpha und der Wurzelzeichenverlängerung (Schriftart Symbol) sowie drei Zeichen der Schriftart Wingdings. √ α ☺ ☹ 😐:




Das·markierte·Zeichen·bzw.·die·Zeichenfolge·√ α·☺·☹·😐·setzt·sich·wie·folgt·zusammen:¶
¶
Der·Code·von··√··ist·hexadezimal:·D6/F0D6··und·dezimal:·214/61654,·Schriftart:·Symbol,¶
·····das·entsprechende·Unicode-Zeichen·ist·SQUARE·ROOT·(hex.·221A).·¶
Der·Code·von··‾··ist·hexadezimal:·60/F060··und·dezimal:·96/61536,·Schriftart:·Symbol,¶
·····das·entsprechende·Unicode-Zeichen·ist·¶
·····(kein·Unicode-Zeichen·für·RADICAL·EXTENDER·vorhanden),¶
·····Ersatz-Vorschlag:·MACRON·(hex.·00AF)·oder·COMBINING·OVERLINE·(hex.·0305).·¶
Der·Code·von··α··ist·hexadezimal:·61/F061··und·dezimal:·97/61537,·Schriftart:·Symbol,¶
·····das·entsprechende·Unicode-Zeichen·ist·GREEK·SMALL·LETTER·ALPHA·(hex.·03B1).·¶
Der·Unicode·von··"Leerzeichen"··ist·hexadezimal:·0020··und·dezimal:·032,·Schriftart:°Times·New·
Roman,··der°genormte°englische°Zeichenname°ist:°SPACE.·¶
Der·Code·von··☺··ist·hexadezimal:·4A/F04A··und·dezimal:·74/61514,·Schriftart:·Wingdings,¶
·····Bedeutung:·WHITE·SMILING·FACE·(Unicode·=·hex.·263A).·¶
Der·Unicode·von··"Leerzeichen"··ist·hexadezimal:·0020··und·dezimal:·032,·Schriftart:°Times·New·
Roman,··der°genormte°englische°Zeichenname°ist:°SPACE.·¶
Der·Code·von··☹··ist·hexadezimal:·4C/F04C··und·dezimal:·76/61516,·Schriftart:·Wingdings,¶
·····Bedeutung:·WHITE·FROWNING·FACE·(Unicode·=·hex.·2639).·¶
Der·Unicode·von··"Leerzeichen"··ist·hexadezimal:·0020··und·dezimal:·032,·Schriftart:°Times·New·
Roman,··der°genormte°englische°Zeichenname°ist:°SPACE.·¶
Der·Code·von··😐··ist·hexadezimal:·4B/F04B··und·dezimal:·75/61515,·Schriftart:·Wingdings,¶
·····Bedeutung:·NEUTRAL·FACE·(Unicode·=·hex.·1F610).·¶
¶
¶
Die·englischen·Namen·aller·Unicode-Schriftzeichen·sind·in·ISO/IEC·10646·genormt··und·weitgehend·
in·http://unicode.org/Public/UNIDATA/UnicodeData.txt↵
oder·über·http://www.unicode.org/charts·zu·finden.¶
¶
Die·Rechenzeit·war·0,234·Sekunden.¶

**Bild 6.5: Screenshot vom Analyse-Ergebnis für √ α ☺ ☹ 😐**

### 6.3.4 Fernöstliche Schriftzeichen und weitere Schriftzeichen ohne Angabe von Namen in der Unicode-Liste

Die Unicode-Liste [2] mit den englischen Namen nach ISO/IEC 10646, aus denen das Analyse-Makro in Abschnitt 6.2 die Namen der Schriftzeichen heraussucht, ist nicht vollständig. Ein großer Teil der chinesischen und japanischen Schriftzeichen sowie die koreanischen Silbenzeichen fehlen darin, so dass das Analyse-Makro deshalb nur den Unicode, aber keine Zeichennamen angeben kann. Stattdessen gebe ich im Makro die folgenden unter a) und b) aufgeführten Internet-Quellen an, aus denen der Anwender gegebenenfalls sich weitere Informationen beschafften kann.

## Bereiche, für die es in der verwendeten Unicode-Liste keine Namen gibt:

**a) Chinesisch und Japanisch (CJK, Chinesisch/Japanisch/Koreanisch)**

**U+4E00** bis **U+9FAF CJK Ideograph** (Kanji)
http://www.unicode.org/charts/PDF/U4E00.pdf (= 26,8 MByte)

**U+3400** bis **U+4DBF** CJK **Ideograph Extension A**
http://www.unicode.org/charts/PDF/U3400.pdf

**U+20000** bis **U+2A6DF** CJK **Ideograph Extension B**
http://www.unicode.org/charts/PDF/U20000.pdf

**U+2A700** bis **U+2B734** CJK **Ideograph Extension C**
http://www.unicode.org/charts/PDF/U2A700.pdf

**b) Silbenzeichen im Koreanischen**

**U+AC00 bis U+D7AF Hangul Syllable** http://www.unicode.org/charts/PDF/UAC00.pdf
oder: http://www.unicode.org/Public/UNIDATA/HangulSyllableType.txt
Dagegen sind die heute in Korea fast nur noch gebräuchlichen alphabetischen **Hangul**-Schriftzeichen U+1100 bis U+11FF in der Unicode-Liste einzeln mit Schriftzeichen-Namen aufgeführt.

**c) Surrogate**

Für Surrogate können keine Buchstabennamen zugeordnet werden.
**U+D800 bis U+DFFF**

**d) Private-Use-Areas**

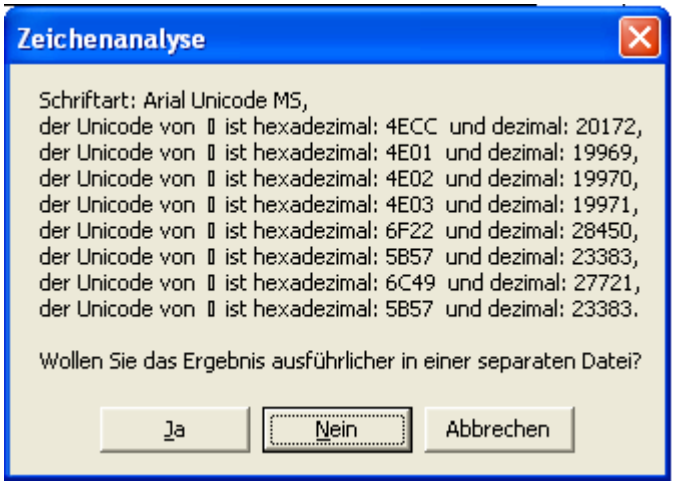
Für die "Private Use Areas" (Benutzerdefinierte Zeichen, Private Use Zone) kann sich jeder Anwender mit eigenen Schriftarten (Fonts) eigene Zeichen festlegen. Am bekanntesten sind die Windows-Schriftarten "Symbol" und "Wingdings", von denen ich für "Symbol" die Unicode-Entsprechungen in das Analyse-Makro integriert habe.

– U+ E0 00 bis U+ F8 FF (Ebene 0)
– U+ F 00 00 bis U+ F FF FD (Ebene 15)
– U+ 10 00 00 bis U+ 10 FF FD (Ebene 16)

**e) Viele Codepoints, die vielleicht noch in der Zukunft belegt werden**

Hier meldet das Analyse-Makro:
"Das Zeichen ist nicht in der Unicode-Liste enthalten".

Ein Beispiel für chinesische Zeichen zeigt Bild 6.6. Hier habe ich eine mir unverständliche Zeichenfolge willkürlich aus dem Internet kopiert (**𠂇丁丐七漢字汉字**).




**Beim Anklicken von "Ja" ergibt sich in einer neuen Datei:**

Das markierte Zeichen bzw. die Zeichenfolge 𠀋丁丂七漢字汉字
setzt sich wie folgt zusammen:

Der Unicode von 𠀋 ist hexadezimal: 4ECC und dezimal: 20172, Schriftart: Arial Unicode MS,
das Zeichen ist ein "CJK-Ideograph",
siehe http://www.unicode.org/charts/PDF/U4E00.pdf.

Der Unicode von 丁 ist hexadezimal: 4E01 und dezimal: 19969, Schriftart: Arial Unicode MS,
das Zeichen ist ein "CJK-Ideograph",
siehe http://www.unicode.org/charts/PDF/U4E00.pdf.

Der Unicode von 丂 ist hexadezimal: 4E02 und dezimal: 19970, Schriftart: Arial Unicode MS,
das Zeichen ist ein "CJK-Ideograph",
siehe http://www.unicode.org/charts/PDF/U4E00.pdf.

Der Unicode von 七 ist hexadezimal: 4E03 und dezimal: 19971, Schriftart: Arial Unicode MS,
das Zeichen ist ein "CJK-Ideograph",
siehe http://www.unicode.org/charts/PDF/U4E00.pdf.

Der Unicode von 漢 ist hexadezimal: 6F22 und dezimal: 28450, Schriftart: Arial Unicode MS,
das Zeichen ist ein "CJK-Ideograph",
siehe http://www.unicode.org/charts/PDF/U4E00.pdf.

Der Unicode von 字 ist hexadezimal: 5B57 und dezimal: 23383, Schriftart: Arial Unicode MS,
das Zeichen ist ein "CJK-Ideograph",
siehe http://www.unicode.org/charts/PDF/U4E00.pdf.

Der Unicode von 汉 ist hexadezimal: 6C49 und dezimal: 27721, Schriftart: Arial Unicode MS,
das Zeichen ist ein "CJK-Ideograph",
siehe http://www.unicode.org/charts/PDF/U4E00.pdf.

Der Unicode von 字 ist hexadezimal: 5B57 und dezimal: 23383, Schriftart: Arial Unicode MS,
das Zeichen ist ein "CJK-Ideograph",
siehe http://www.unicode.org/charts/PDF/U4E00.pdf.

Die englischen Namen aller Unicode-Schriftzeichen sind in ISO/IEC 10646 genormt und weitgehend
in http://unicode.org/Public/UNIDATA/UnicodeData.txt
oder über http://www.unicode.org/charts zu finden.

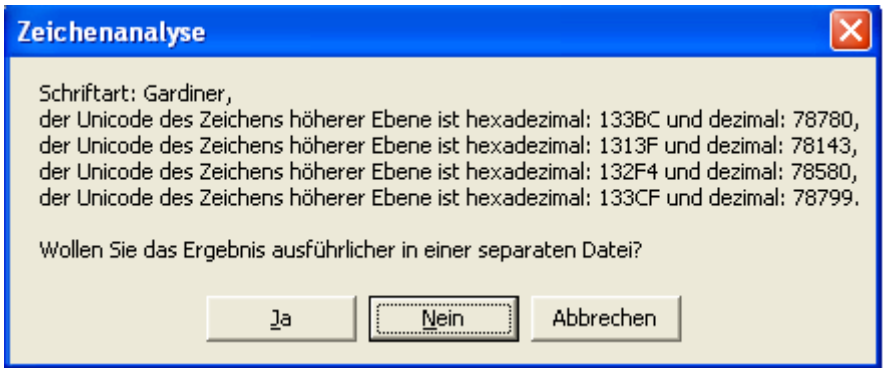
Die Rechenzeit war 2,953 Sekunden.

**Bild 6.6: Screenshot des Ergebnisses für die Analyse chinesischer Schriftzeichen**

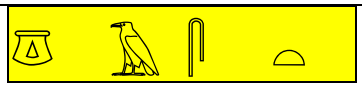
Weiteres siehe auch im Anhang Abschnitt A.5.

### 6.3.5 Hieroglyphen als Beispiel für Schriftzeichen höhere Unicode-Ebenen

Beispiel:




**Beim Anklicken von "Ja" ergibt sich in einer neuen Datei:**

Das markierte Zeichen bzw. die Zeichenfolge  setzt sich wie folgt zusammen:

Der Unicode des Zeichens höherer Ebene  ist hexadezimal: 133BC und dezimal: 78780,
(der Unicode des Surrogat 1 ist hexadezimal: D80C und dezimal: 55308,
der Unicode des Surrogat 2 ist hexadezimal: DFBC und dezimal: 57276),
der genormte englische Zeichenname ist: EGYPTIAN HIEROGLYPH W011,
die Schriftart ist Gardiner.

Der Unicode des Zeichens höherer Ebene  ist hexadezimal: 1313F und dezimal: 78143,
(der Unicode des Surrogat 1 ist hexadezimal: D80C und dezimal: 55308,
der Unicode des Surrogat 2 ist hexadezimal: DD3F und dezimal: 56639),
der genormte englische Zeichenname ist: EGYPTIAN HIEROGLYPH G001,
die Schriftart ist Gardiner.

Der Unicode des Zeichens höherer Ebene  ist hexadezimal: 132F4 und dezimal: 78580,
(der Unicode des Surrogat 1 ist hexadezimal: D80C und dezimal: 55308,
der Unicode des Surrogat 2 ist hexadezimal: DEF4 und dezimal: 57076),
der genormte englische Zeichenname ist: EGYPTIAN HIEROGLYPH S029,
die Schriftart ist Gardiner.

Der Unicode des Zeichens höherer Ebene  ist hexadezimal: 133CF und dezimal: 78799,
(der Unicode des Surrogat 1 ist hexadezimal: D80C und dezimal: 55308,
der Unicode des Surrogat 2 ist hexadezimal: DFCF und dezimal: 57295),
der genormte englische Zeichenname ist: EGYPTIAN HIEROGLYPH X001,
die Schriftart ist Gardiner.

Die englischen Namen aller Unicode-Schriftzeichen sind in ISO/IEC 10646 genormt und weitgehend in http://unicode.org/Public/UNIDATA/UnicodeData.txt
oder über http://www.unicode.org/charts zu finden.

Die Rechenzeit war 2,547 Sekunden.

**Bild 6.7: Screenshot für Hieroglyphen**

Die Hieroglyphen-Bezeichnungen wie "G001" für das Aleph richten sich nach den Listen von Gardiner, siehe im Internet am einfachsten bei Wikipedia [3].

### 6.3.6 Anzeige bei Word-Feldfunktionen

Testzeile mit einem stets aktuellen Tagesdatum (Feldfunktion), einem Overstrike-Feld (d), einem Geschützten Sonderzeichen (Ω), und dazwischen normale Buchstaben sowie Symbol-Zeichen (Alpha und Beta):

Datum: 14.01.2015 aαdβ c Ωfg

Auch diese Zeile wird jetzt richtig analysiert, siehe Bild 6.8. Es wird zusätzlich auch der Feld-Code angezeigt:

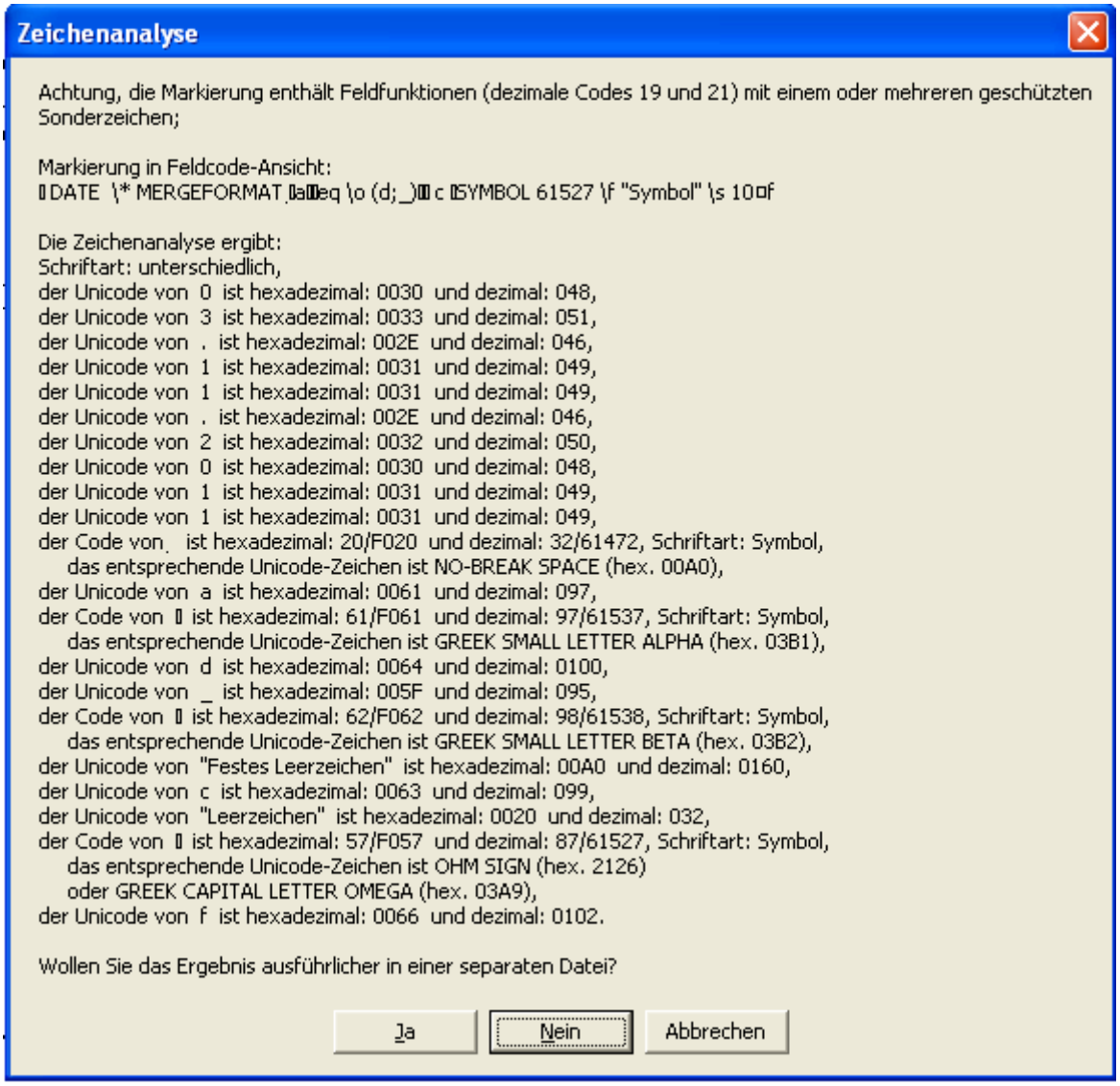



**Bild 6.8: Screenshot vom "4. 11. 2011" der Analyse eines Feldes mit dem Datum**

Will man den Feldcode selbst analysieren, muss man "Ja" anklicken und sich das Ergebnis in einer separaten Datei anzeigen lassen und für die Zeile:

DATE * MERGEFORMAT a eq \o (d;_) c SYMBOL 61527 \f "Symbol" \s 10 f

das Analyse-Makro einfach nochmals einsetzen. Das ergibt für Feld-Anfang und Feld-Ende:

. . .
Der Unicode von **"Steuerzeichen"** ist hexadezimal: 0013 und dezimal: 019.
. . .
Der Unicode von **"Steuerzeichen"** ist hexadezimal: 0015 und dezimal: 021.

### 6.3.7 Beispiel für die Analyse in einer Tabelle

Eine Mustertabelle mit einer Feldfunktion in Feld (Zelle) Z 22:

| | | |
|---|---|---|
| Z 11 | Z 12 | Z 13 |
| Z 21 | Z 22 = Datum 14.01.2015 | Z 23 |

ergibt bei der Analyse einer Tabellenspalte folgendes Ergebnis:

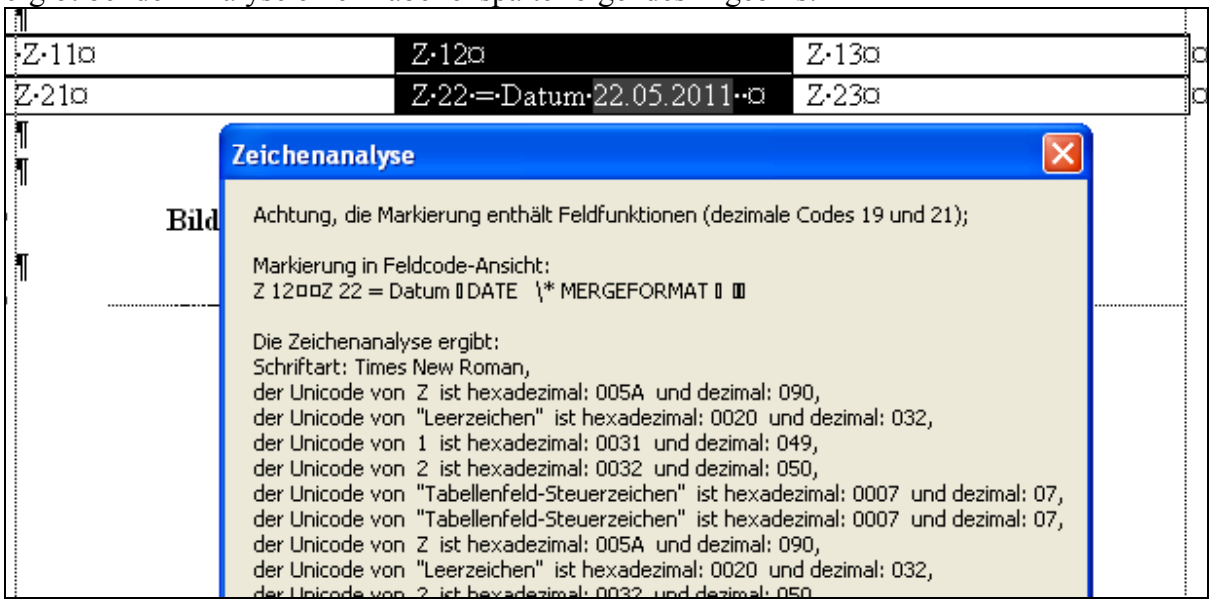



**Bild 6.9: Screenshot des Ergebnisses für die Analyse einer Tabellenspalte**

**Nachtrag 22. März 2012:**
Probleme gab es, wenn das Tabellensteuerzeichen am Ende einer Tabellen-Reihe markiert wurde (was in der Praxis so gut wie nie vorkommt). Hierbei stürzt der VBA-Befehl "Copy" ab.

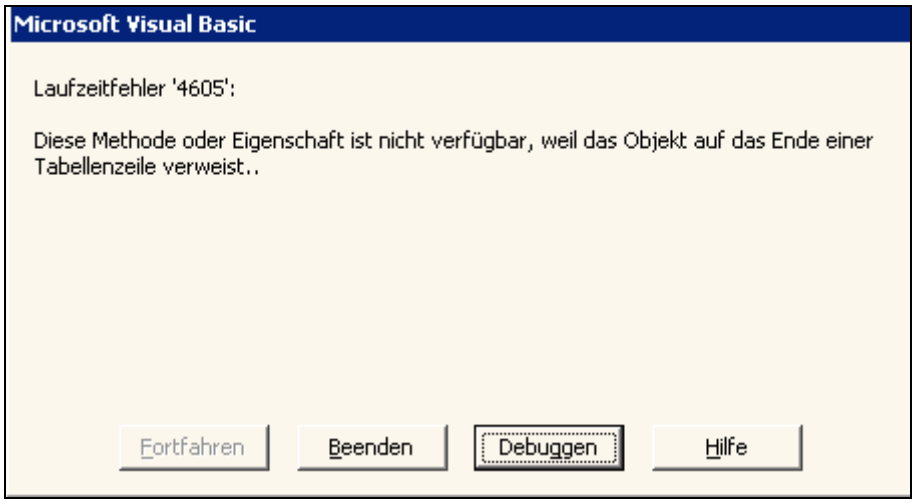



**Bild 6.10: Fehlermeldung beim Markieren des Steuerzeichens hinter einer Tabelle**

Lösung: Den Code der Tabellen-Steuerzeichen (hexadezimal und dezimal "7") ermittelte ich aus dem Quellcode einer kleinen Word-Testdatei, die ich mit dem Editor öffnete: "a b c ". Zwischen den Tabellen-Zellen ist nur einmal das Steuerzeichen "7", am Ende einer Tabellen-Reihe gibt es dieses Steuerzeichen zweimal. Ein anderes Ergebnis erhält man, wenn man die Markierung als String speichert und dann erst analysiert. Dann würde vor jedem Tabellenfeld-Steuerzeichen eine Absatzmarke hinzugefügt.

In dem fertigen Makro wird jetzt per Messagebox auf das Tabellensteuerzeichen hingewiesen, wie ich es bei Analyse des Quelltextes einer Word-Datei ermittelte.

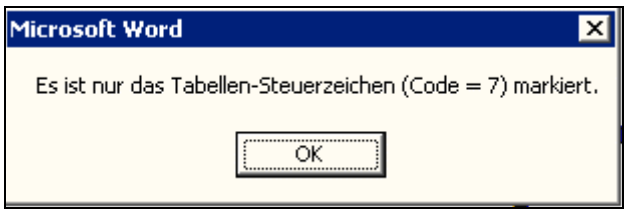



**Bild 6.11: Meldung eines Tabellen-Steuerzeichens**

### 6.3.8 Windows-Sonderzeichen mit fehlerhafter Schriftart-Zuordnung

Windows-Sonderschriftarten wie Symbol, Wingdings und Webdings haben bei korrekter Erzeugung als Codes entweder denselben Code wie "(", "Runde Klammer links" (im folgenden Bild grün markiert), oder einen Code im PUA-Bereich zwischen dezimal 61472 bis 61695 (blau markiert). Vom Analyse-Makro wird in beiden Fällen dasselbe Ergebnis angezeigt.

Weichen die Codes davon ab (gelb markiert), kann man von einer Störung in der Datei ausgehen. Hier sollte man dann den Zeichen wieder eine Unicode-Schriftart zuweisen.

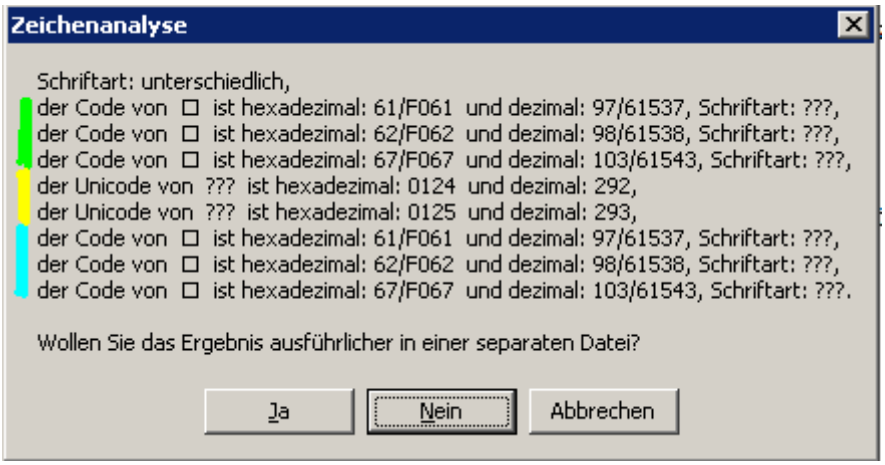



**Bild 6.12: Analyse der Symbol-Zeichenfolge αβγ∃%αβγ**

Bei einer Windows-Sonderschrift, die auf dem eigenen Computer nicht vorhanden ist, werden die Codes der Zeichen richtig analysiert, die Schriftart aber nur zum Teil ermittelt (siehe folgendes Bild mit der Analyse einiger Zeichen von **SymbolPS)**.

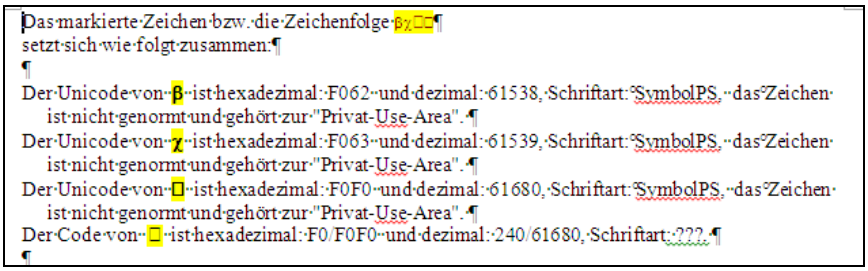



**Bild 6.13: Analyse einer nichtinstallierten Windows-Sonderschriftart βχ□□**

## 6.4 Beurteilung des Analyse-Makros in Abschnitt 6.2

Dieses Makro liefert die bei Word 2000 noch fehlende Möglichkeit, ein Zeichen zu analysieren (ab Word 2003 mit mit der Tastenkombination Alt + C). Durch die Analyse von kombinierten Zeichen und auch der richtigen Decodierung von Zeichen der Sonderschriftarten "Symbol" und Wingdings hat das Zeichenanalyse-Makro sogar noch Vorteile gegenüber den Funktionen in Word 2010. Den Hauptvorteil des Analyse-Makro sehe ich in der weitgehenden Angabe des Buchstabennamens, sogar für Symbol- und Wingdingszeichen, auch wenn dies nur auf Englisch möglich ist.

Durch die Analyse eines Textabschnitts mit über 100 Einzelzeichen können auch kompliziertere Zeichenfolgen untersucht werden.

Auch in Tabellen und bei Feldfunktionen ist das Analyse-Makro einsetzbar.

Bei Texten, in denen Links-nach-rechts-Schriftzeichen (lateinische Buchstaben) und rechts-nach-links-Schriftzeichen (arabische oder hebräische Buchstaben) innerhalb desselben Absatzes vorkommen, etwa in einem arabischen Computer-Handbuch, entspricht die Analyse meines Makros der **logischen Reihenfolge**, nicht der optischen Reihenfolge auf dem Bildschirm. Das Analyse-Makro erkennt leider nicht, wie es auf dem Bildschirm aussieht, sondern nur die logische Reihenfolge (in der man den Text auch vorlesen würde). Die Anzeige auf dem Bildschirm weicht bei solchen Schrift-Gemischen in Word 2010 von der Anzeige in Word 2000 ab, da Word 2000 die **Bidirektionale Steuerzeichen**[32] noch nicht umsetzt (Stand April 2013).

### Verbesserungen seit 2011

Das Analyse-Makro in Abschnitt 6.2 kann jetzt (Stand 2011) auch Sonderzeichen analysieren, die durch eine Feldfunktion geschützt sind, oder Zeichen über mehrere Tabellenfelder hinweg analysieren. Es können jetzt auch Textteile mit über 44 Zeichen gleichzeitig analysiert werden. Ferner können jetzt Texte trotz Textfelder und graphischer Objekte (Shapes) analysiert werden.

Bei der Untersuchung von Texten mit Feldfunktionen in der Feldcode-Ansicht habe ich inzwischen (Stand: März 2014) auf die Analyse des Feldcodes verzichtet und lasse gleich das Feld-Ergebnis analysieren. In der Feldcode-Ansicht erscheinen (anders als im Quelltext) nach meiner inzwischen gemachten Erfahrung **niemals** unbekannte Sonderzeichen, die einer Analyse bedürften.

### Noch zu verbessern

Bei fernöstlichen Silben-Zeichen (Chinesisch/Japanisch/Koreanisch) wird im Makro kein Buchstabenname angegeben, sondern nur auf die Quellen von Unicode [2] im Internet verwiesen, die ich nur als leider zu umfangreiche pdf-Dateien vorfand. Vielleicht wäre hier für den Anwender stattdessen ein Hinweis auf Übersetzungstools im Internet wie das von Google [13] oder anderen sinnvoller.

Bei Tabellenausschnitten wird bei dem letzten ausgewählten Feld einer Tabellen-Reihe ein doppeltes Tabellenfeldzeichen angezeigt wie am Ende der Tabellen-Reihe. Es wird nämlich der ausgewählte Tabellenbereich als separate Tabelle in einer Zwischendatei analysiert. Vielleicht sollte man hier dann besser angeben: "Es wurden Reihen a bis b und Spalten x bis y markiert"?

---

[32] Bidirektionale Steuerzeichen: Unicode dez. 8206 bis dez. 8238 bzw. hex. U+200E bis U+202E.

# 7 Installieren der Makros in der "Normal.dot" und Zuordnung von Tastenkürzeln

Eine automatische Übertragung der Makros dieses Aufsatzes per spezielles Installationsprogramm löste bei mir den Antivirenschutz aus, deshalb empfehle ich hier die Übertragung von Hand (siehe auch [24]). Inzwischen biete ich eine kompakte Dokumentvorlage an (als Zip-Datei in meiner Homepage), in der Makro und Tastenkürzel bereits fertig installiert sind, siehe Abschnitt 8.

## 7.1 Installieren von Makromodulen

In Word ist mit Alt + F11 oder über den Menüpfad das Makro-Fenster zu öffnen. Dabei ist es egal, welche Worddatei gerade geöffnet ist. Achtung: Im linken Teilfenster im Bild 7.1, hier etwas durch das Pull-Down-Menü verdeckt) muss die "Normal.dot" gewählt sein.

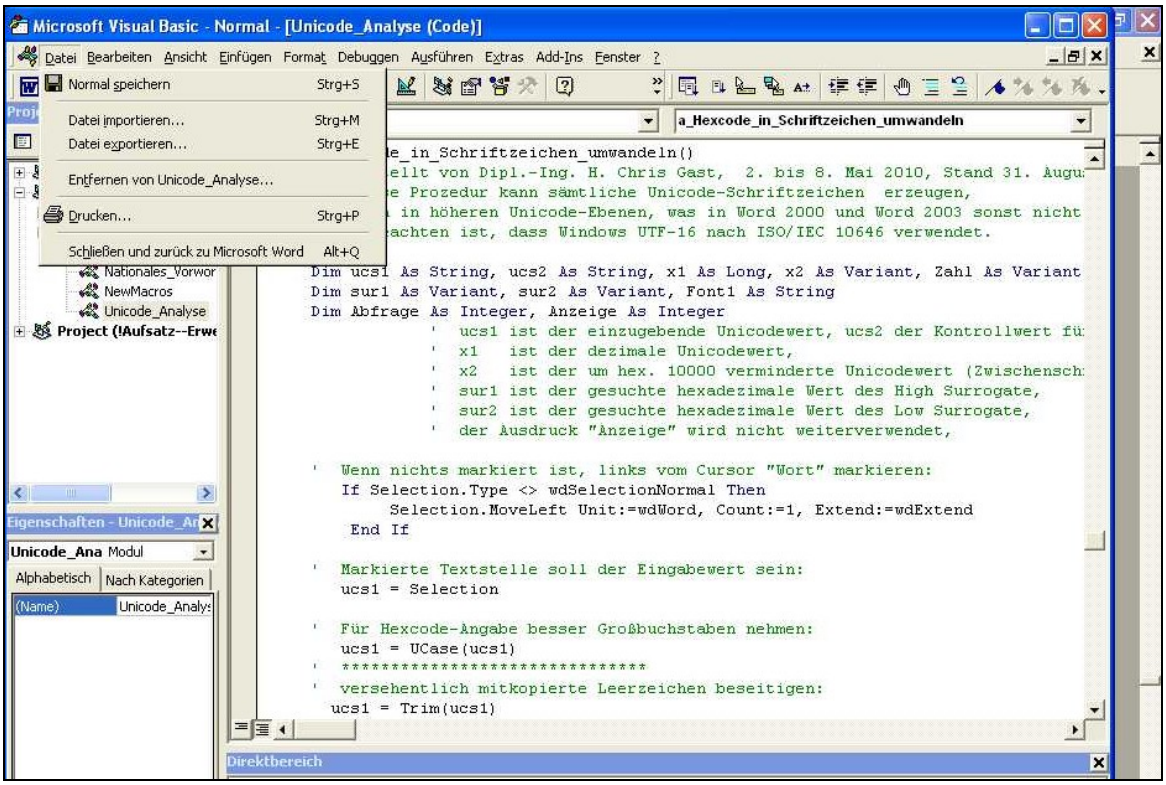



**Bild 7.1: VBA-Fenster**

Oben links ist das Pulldown-Menü "Datei" anklicken, dann "Datei importieren." Damit wird dann das gewünschte Makromodul installiert.

Für das Analyse-Makro ist auch eine Textdatei mit den (englischen) Unicode-Namen zu installieren. Diese Liste ist im Internet zu finden unter:

"http://unicode.org/Public/UNIDATA/UnicodeData.txt" (siehe auch [2]).

Diese Liste ist dann auf der Festplatte entweder bei den Word-Vorlagen oder im Ordner "…Word\STARTUP" zu speichern.

## 7.2 Zuordnen von Tastenkürzeln zu Makros und Befehlen

Die geöffnete Word-Datei darf bei der Zuordnung von Tastenkürzel nicht ein Makro mit demselben Namen wie die Normal.dot enthalten, sonst klappt die eindeutige Zuordnung von Tastenkürzeln nicht. In diesem Fall sollte man die Tastenkürzel-Zuordnung von einer anderen

Word-Datei aus ausführen, die selbst keine Makros enthält, um Verwirrung im Computer zu vermeiden. Die Word-Datei mit dem ursprünglichen Makro sollte also geschlossen sein.

~~Für das Analyse-Makro aus Abschnitt 6.2 benutzte ich anfangs das Tastenkürzel Alt + Shift + C.~~

Ich stellte inzwischen fest, dass Alt + Shift + C in Kopf- und Fußzeilen den Befehl "**Bereich schließen**" auslöst. Das ist nicht veränderbar. Von daher wechselte ich zu dem Tastenkürzel AltGr + C.

### 7.2.1 Zuordnung per Hand in Word 2000 bis Word 2003

Oben in der Menüleiste von Word 2000 bis Word 2003 ist auf **"Extras" → "Anpassen"** zu gehen, dann erscheint das folgende linke Menü auf dem Bildschirm (siehe Bild 7.2). Darin ist unten in der Mitte "*Tastatur"* anzuklicken, und im neu erscheinenden Menü (siehe rechtes Teilbild) unter "Kategorien" ist dann "Makros" oder die zutreffende Befehlskategorie auszuwählen (die Ordnung entspricht teilweise den Menüleisten).

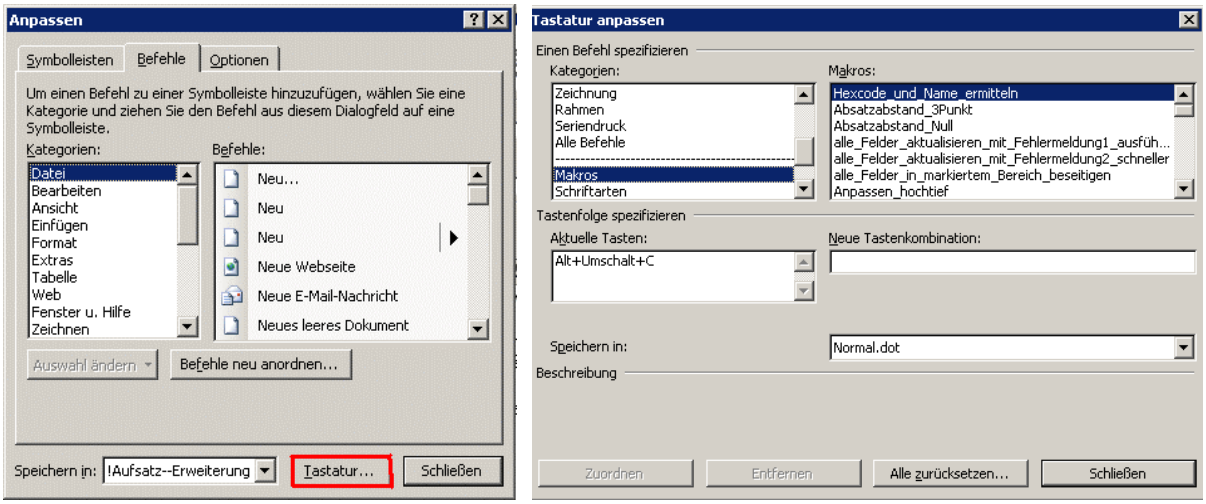



**Bild 7.2: "Extras" → "Anpassen" → "Tastatur" → "Kategorie" = "Makros"**

Hier im Bild 7.2 rechts sehe ich dann zum Beispiel im Feld "Aktuelle Tasten:", dass ich meinem Makro "Sub *Hexcode_und_Name_ermitteln()*" bereits das Tastenkürzel Alt + Shift + C zugeordnet hatte[33]. Dies kann ich jetzt ändern durch Eingabe einer neuen Tastenkombination im Feld "Neue Tastenkombination". Achtung: In den Feld-Layouts unterscheiden sich hier Word 2000 und Word 2003 etwas.

Die Zuordnung von Tastenkürzeln zu Schriftzeichen (Sonderzeichen), die über "Einfügen → Symbol" erzeugt wurden, erscheint hier nicht. Diese findet man nur, in dem man die Schrift- bzw. Sonderzeichen unter **"Einfügen → Symbol → Symbol"** sucht und dort "Tastenkombination" anklickt, oder indem man beim Drucken statt "Dokument" im Druckmenü "Tastenbelegung" wählt. Dann erst sieht man, ob man dem Zeichen bereits ein Tastenkürzel zugeordnet hat oder nicht.

---

[33] "Umschalt" = "Shift".

**Zuordnung von Tastenkürzeln per Hand in Word 2010**

Mit der Maus Rechtsklick in das Menüband am oberen Bildschirmrand (siehe Bild 7.3):

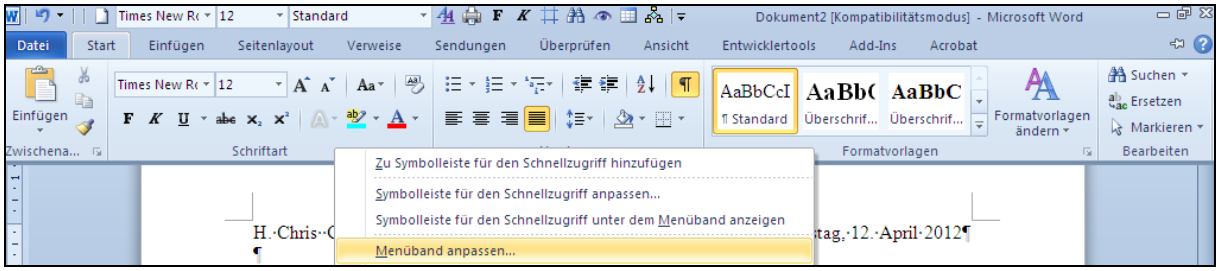



**Bild 7.3: Rechtsklick auf das Menü-Band "Menüband anpassen"**

Dann unten links im folgenden Bild 7.4 unten "Tastenkombinationen: Anpassen" wählen:

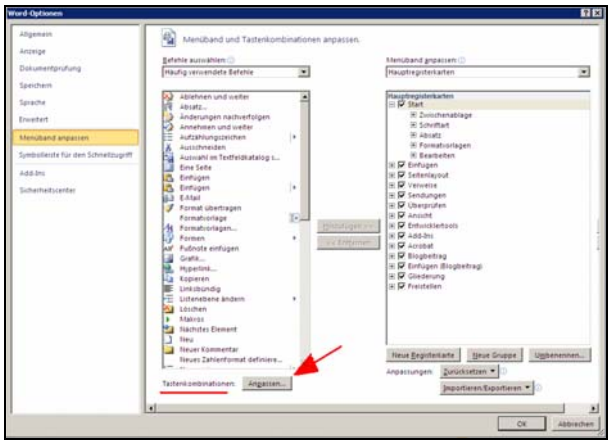


**Bild 7.4: "Menüband anpassen" → "Tastenkombinationen: Anpassen" anklicken**

Dann links unter "Kategorien" "Makros" wählen und rechts unter "Makros" das betreffende Makro auswählen (siehe Bild 7.5).

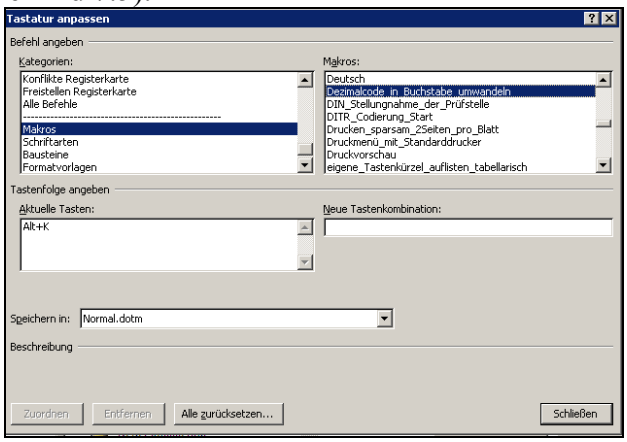



**Bild 7.5: "Menüband anpassen" → "Tastenkombinationen: Anpassen" → links "Kategorie = "Makros" wählen → rechts "Makro" auswählen.**

Es werden, wenn vorhanden, links die "Aktuelle Tasten" angezeigt. Rechts muss man die "Neue Tastenkombination" eingeben.

### 7.2.2 Zuordnen von Tastenkürzel per Makro

Die Tastenzuordnung kann auch per Makro erfolgen. Im folgenden Beispiel wird dem Makro "*Sub Hexcode_und_Name_ermitteln*()" in der Vorlage "*Normal.dot*" das Tastenkürzel *Alt+Shift+C* zugeordnet.

```
Sub Tastenkürzel_zuordnen_für_das_Analysemakro()
' Erstellt von H. Chris Gast, 23. August 2010, Stand 18. Oktober 2010.
' Ordnet dem Analysemakro in der Normal-dot-Datei das Tastenkürzel "Alt+Shift + C" zu.
  With Application
      .CustomizationContext = NormalTemplate
      .KeyBindings.Add KeyCode:=BuildKeyCode(wdKeyAlt, wdKeyShift, wdKeyC), _
          KeyCategory:=wdKeyCategoryCommand, _
          Command:="Hexcode_und_Name_ermitteln"
  End With
  '
   MsgBox "Tastaturkürzel ist installiert. " & vbCrLf _
        & "Drücken Sie ""Alt + Shift + c"" zur Zeichenanalyse."
End Sub
' ---------------------------------------------------------------------------------------------------
```

In dem folgenden "AutoOpen"-Makro wird ein Tastenkürzel nur für die Datei zugeordnet, die gerade geöffnet ist.

Mit dem Öffnen der Word-Datei, in dem dieses Makro installiert ist, wird hier dem Makro "Sub *a011_Hexcode_und_Name_ermitteln()*" aus Abschnitt 6.2 das Tastenkürzel AltGr + C zugewiesen, solange die Datei geöffnet ist. Liegt diese Datei allerdings nicht als Word-Datei, sondern nur als pdf-Datei vor, funktionieren diese Word-Makros natürlich nicht.

```
Sub AutoOpen()
' Erstellt von H. Chris Gast, 23. August 2010
' Ordnet dem Analysemakro in dieser Datei das Tastenkürzel "AltGr + C" zu.
  With Application
     .CustomizationContext = ActiveDocument
     .KeyBindings.Add KeyCode:=BuildKeyCode(wdKeyAlt, wdKeyControl, wdKeyC), _
        KeyCategory:=wdKeyCategoryCommand, _
         Command:="a011_Hexcode_und_Name_ermitteln"
   End With
  '
   MsgBox "Markieren Sie ein Schriftzeichen und drücken Sie ""AltGR + c"" zur Analyse."
End Sub
' ---------------------------------------------------------------------------------------------------
```

Die beiden Makros dieses Unterabschnittes befinden sich nicht in dem Makro-Modul zur Übertragung in die Normal-dot.

Die VBA-Namen der Tasten entsprechen nicht der deutschen Tastatur.

# 8 Kompakt-Variante für den Download

Für den Download biete ich eine kompate Dokumentvorlage (".dot"-Datei) an:

Dipl.-Ing. Hanna-Chris Gast

## Word-Makros für Zeichen-Analyse und andere

Bei der Analyse von Schrift- und Sonderzeichen in einem Word-Text kann das hier enthalten **Analyse-Makro** sämtliche normgerechten Schriftzeichen (Unicode) analysieren und (außer bei chinesischen, japanischen und koreanischen Silbenzeichen) deren englischen Namen nach ISO/IEC 10646 ausgeben. Daneben kann das Analyse-Makro aber auch die Codes von nicht-normgerechten Sonderschriftarten ermitteln sowie bei den Schriftarten "Symbol" und "Wingdings" die Bedeutungen angeben. Das Analyse-Makro bietet die Möglichkeit, mehrere Zeichen einer Zeichenkette gleichzeitig zu analysieren. Damit können auch Schriftzeichen mit (diakritischen) Akzentzeichen gut analysiert werden, die sonst nur umständlich zu analysieren sind, oder etwa einen Text mit einer Kombination von Schriftzeichen aus Schriftsystemen, die von rechts nach links geschrieben werden, mit Schriftzeichen, die von links nach rechts geschrieben werden. Dies würde sich mit dem Befehl "Alt + C" nicht analysieren lassen. Das Analyse-Makro funktioniert auch dann, wenn im Text nur Platzhalter ("leere Kästchen") zu sehen sind. Dem Analyse-Makro habe ich das **Tastenkürzel** "AltGr + C" zugeordnet.

**Beispiel eines Analyse-Protokolls mit diesem Makro**

Der Code von ☎ ist hexadezimal: 28/F028 und dezimal: 40/61480, Schriftart: Wingdings,
Bedeutung: BLACK TELEPHONE (Unicode = hex. 260E).

Dem in dieser Dokumentvorlage enthaltenen Makro für die Erzeugung von Schriftzeichen aus dem Dezimalen Code das Tastenkürzel "Alt + K".

Ferner enthält diese Dokumentvorlage jetzt auch mein Makro zur Umwandlung von Zeichen der Schriftarten Symbol und Wingdings in Unicode-Zeichen und dem damit verknüpften Befehl für "verbessertes Inhalte einfügen", Tastenkürzel "Strg + Shift + i".

Diese Dokumentvorlage mit den Makros ist im "Word\STARTUP-Order" für Word unterzubringen, damit die darin enthaltenen Makros genutzt werden können (dabei Dateiname nicht ändern):

Pfad zum StartUp-Order unter Windows XP:

.....C:\Dokumente und
Einstellungen\Benutzernamen\Anwendungsdaten\Microsoft\Word\StartUp

Pfad zum StartUp-Order unter Windows 7:

c:\users\Benutzernamenkürzel\appdata\roaming\microsoft\word\STARTUP

Klicken Sie auf den grünen Button und öffnen Sie dadurch den Word-STARTUP-Ordner. Dorthin kopieren Sie bitte die vorliegende Dokumentvorlage:

Im Explorer den Pfad zum StartUP-Ordner

**********

## Impressum


hcgast "ät" siebener-kurier.de *(dabei "ät" durch "@" ersetzen).*
Bergstr.1
14109 Berlin
Im Internet: http://www.siebener-kurier.de/chris-aufsaetze
dort: http://www.siebener-kurier.de/chris-aufsaetze/Word-Erweiterung-Unicode-Makros.pdf
bzw.: http://www.siebener-kurier.de/chris-aufsaetze/Symbol+Wingdings-in-Unicode.pdf
Fertiges Makro: http://www.siebener-kurier.de/chris-aufsaetze/Zeichen-Analyse-kompakt.zip.
Diese Makros wurden zuerst veröffentlicht als Beilage zum Siebener-Kurier Nr. 60 (August 2010),
ISSN 0948-6089.


Dipl.-Ing. Hanna-Chris Gast, Stand 21. Dezember 2014.

Das Makro zum grünen Button, um im Windows-Explorer den Word-StartUp-Ordner zu finden:

```
Private Sub CommandButton1_Click()
' H. Chris Gast, Stand: 20. August 2014.
' "Word-StartUp"-Ordner im Explorer_öffnen und diese Datei schließen.
'
' Zuerst prüfen, ob weitere Word-Dateien geöffnet sind (wenn ja, dann lassen sich
' nämlich ".dot"-Dateien im StartUp-Ordner nicht ersetzen):
 If Application.Windows.Count > 1 Then
    MsgBox _
      "Bitte erst alle anderen Word-Dateien schließen und dann dieses Makro erneut starten.", _
      vbExclamation
    Exit Sub
  Else
    Shell "Explorer.exe /e, " & Options.DefaultFilePath(wdStartupPath), vbNormalFocus
    Application.Quit SaveChanges:=False
  End If
End Sub
' ---------------------------------------------------------------------------------------------
```

Außer den Tastenkürzeln habe ich für Word 2000 bis Word 2003 ein Symbolleiste ergänzt, um die Makros für Zeichen-Analyse und Zeichen-Synthese per Button (Icon) aufzurufen, siehe im folgenden Bild die drei Button links. Die weiteren Buttons gehören zu einem Makro für die Umwandlung der Schriftarten Symbol und Wingdings in Unicode, das unter [24] beschrieben ist.

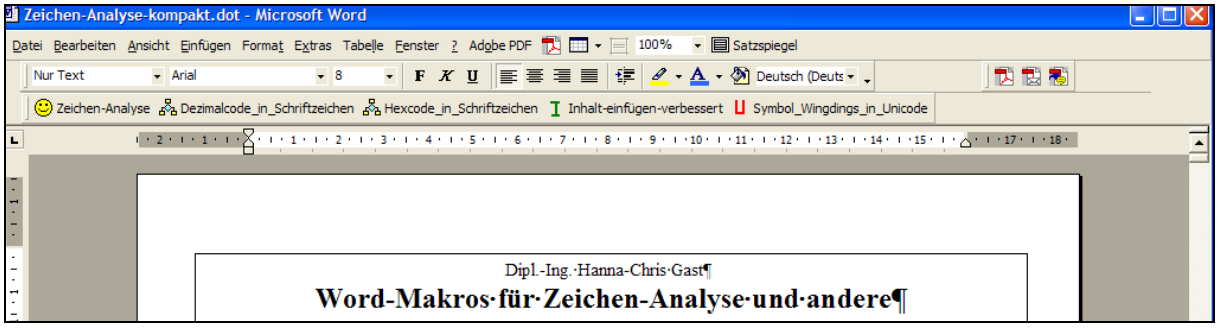



**Bild 8.1: Screenshot der Dokumentvorlage "Zeichen-Analyse-kompakt.dot"**

In Word 2007 und höher sind diese Buttons über das Ribbon-Tab "Add-Ins" zugänglich.

# Anhang A
# Einige Zwischenschritte und Ergänzungen zum Analyse-Makro

Bis die Makros in den Abschnitten 5 und 6 auch für Unicode-Zeichen höherer Ebenen (Hexcode > FFFF) funktionierten, habe ich erst ziemlich viele andere Wege durchprobiert, die nicht funktionierten. In den Abschnitten A.1 bis A.4 gebe ich einige Zwischenschritte an.
Abschnitt A.5 enthält zusätzliche Information zum Umgang mit chinesisch/japansich/koreanischen Zeichen.

## A.1 Hexadezimale Zahlen in Dezimale Zahlen umrechnen und umgekehrt

Am Anfang stand ich vor dem Problem, mit Hexzahlen zu rechnen, was mir zuerst nicht gelang. Schließlich schaffte ich es dadurch, die Hexzahlen in Dezimalzahlen umzuwandeln, dann mit diesen die Rechnungen vornehmen, und das Ergebnis dann wieder in hexadezimale Zahlen zurück-umzuwandeln.

Für die Umrechnung von Hexzahlen in Dezimalzahlen eignet sich die **Val-Funktion**. Die Hilfe-Funktion (von Word 2000) schreibt dazu:
*"Die Val-Funktion liest die Zeichen einer Zeichenfolge bis zum ersten Zeichen, das nicht als Teil einer Zahl interpretiert werden kann. ... Die Funktion erkennt allerdings die Präfixe &O (für oktal) und **&H (für hexadezimal) zur Angabe der Zahlenbasis**..."*[34]

```
Sub Hexzahlen_in_Dezimalzahlen_Umrechnen()
   ' Die Val-Funktion gibt normalerweise nur "Integer"-Werte aus, die aber bei VBA
   ' anders als in der Mathematik nur bis (dezimal) "+32'767" erlaubt sind.
   ' Hier musste deshalb der Variablentyp (Long) erzwungen werden.
   Dim x1 As String, x2 As Long
   '
   x1 = InputBox("Bitte  Hexadezimale Zahl eingeben", "Umrechnung", "2F")
   x2 = Val("&H" & x1 & "&")
   '  Der Zusatz [& "&"] erzwingt den Datentyp Long.
   MsgBox ("Dezimalwert von " & x1 & " ist " & x2)
   '
   ' Erklärung:  Variablen vom Datentyp Long  (= lange Ganzzahl) werden als 32-Bit-Zahlen (4 Bytes)
   ' mit Vorzeichen im Bereich von -2'147'483'648 bis +2'147'483'647 gespeichert.
   ' Das Typkennzeichen für Long ist das Zeichen (&).
   ' Variablen vom Datentyp Integer werden als 16-Bit-Zahlen (2 Bytes)
   ' in einem Bereich von -32'768 bis +32'767 gespeichert.
   ' Das Typkennzeichen für Integer ist das Zeichen (%).
   ' Jeder Versuch, "x" eine Zahl außerhalb dieses Bereichs zuzuweisen, führt zu einem Fehler.
End Sub
' ---------------------------------------------------------------------------------------------------
```

[34] Siehe auch http://support.microsoft.com/kb/161304, Stand: 13. Juli 2004.
Hiermit möchte ich außerdem Mihał Krajewski für seinen Hinweis darauf danken!

```
Sub Dezimalzahl_in_Hexadezimalzahl_Umrechnen()
' Dieses Umrechnen ist in VBA sehr einfach, als Beispiel ist "47" gewählt:
   Dim x1 As Long, x2 As String
   x1 = InputBox("Bitte dezimale Zahl eingeben", "Umrechnung", 10)
   x2 = Hex(x1)
   MsgBox ("Hexwert von " & x1 & " ist " & x2)
End Sub
' ---------------------------------------------------------------------------------------------
```

Die Umrechnung von Konstanten kann mit der Taschenrechner-Funktion "Calc.exe" von Windows erfolgen:

"Start → Ausführen → "calc.exe" eingeben.
Dann unter "Ansicht → Wissenschaftlich" anklicken.
Danach sind Knöpfe wie "HEX", "DEZ" usw. zu sehen.

## A.2 Analyse einer Zeichenkette (ohne Sonderschriftart-Erkennung)

Recht nützlich war für mich als Zwischenschritt zum endgültigen Analyse-Makro das folgende Makro mit der Befehlszeile: **"x(i) = Mid(Selection, i, 1)"**. Es ergibt jedoch Fehler, wenn diese Zeichen einer Windows-Sonderschrift enthalten, die per "Einfügen → Symbol" erzeugt worden sind, wie etwa bei der Formel "$\omega = 2\pi f$". Auch die Warnung, wenn eine fehlerhafte Anzeige möglich ist, funktioniert nicht immer. Deshalb habe ich diesen Weg nicht weiter verfolgt und einen neuen Ansatz mit der Zerlegung des markierten Bereiches in "Ranges" verfolgt, siehe Abschnitt 6.2.

Dieses Makro kann die Schriftart ermitteln, wenn sie einheitlich ist.

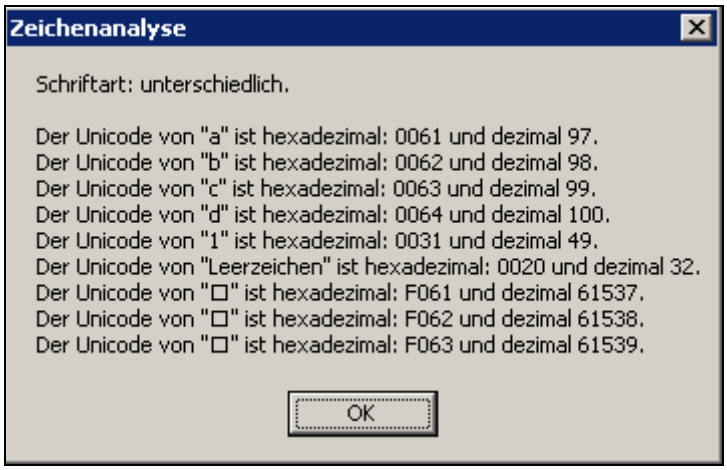
```
' Die folgende Funktions-Deklaration muss vor dem ersten Makro des Moduls eingefügt werden:
Private Declare Function MessageBoxW Lib "user32.dll" (ByVal hWnd As Long, _
    ByVal lpText As Long, ByVal lpCaption As Long, ByVal uType As Long) As Long

Sub a012_Analyse_einer_Zeichenkette_außer_bei_Sonderschriftarten()
' Erstellt von Dipl.-Ing. Hanna-Chris Gast, 31. Mai 2010, Stand 1. November 2010.
' Funktioniert in Word 2000 bis Word 2000.
' Problem: Schrifterkennung funktioniert bei Windows-Sonderschriftarten nicht immer.
'
  Dim x(46) As String, z(46) As String
  Dim i As Integer, j As Integer, N As Integer
  Dim f2 As String
  Dim Ergebnis As String
  Dim Rng0 As Range
  Dim Abfrage As Integer  ' Wird hier nur der Form halber gebraucht.
  '
  ' Maximale Länge der Zeichenkette wird begrenzt wegen der maximalen Zeilenzahl
  ' der MessageBoxW (in eine normale MsgBox passen sogar nur 16 bis 20 Zeilen):
  N = Len(Selection)
  If N > 46 Then
      MsgBox "Maximal 46 Zeichen markieren! Es sind hier " & N & " Zeichen markiert."
      Exit Sub
  End If
  '
```

```
' Analyse des markierten Bereichs:
For i = 1 To N
  x(i) = Mid(Selection, i, 1)
  z(i) = Hex(AscW(x(i)))
  If Val("&H" & z(i) & "&") = 32 Then
    x(i) = "Leerzeichen"
  ElseIf Val("&H" & z(i) & "&") < 30 Then
     x(i) = "Steuerzeichen"
  End If
 '  Der Hexadezimale Unicode wird in der Regel vierstellig angegeben:
    Select Case Len(z(i))
        Case Is = 3
        z(i) = "0" & z(i)
        Case Is = 2
        z(i) = "00" & z(i)
        Case Is = 1
        z(i) = "000" & z(i)
    End Select
Next i
'
'  Bestimmung der Schriftart (teilweise fehlerhaft bei Windows-Sonderschriften):
f2 = Selection.Font.Name
'  Wenn die Schriftanzeige leer ist, ist sie unterschiedlich:
If f2 = "" Then f2 = "unterschiedlich"
'
' ********************
' Ergebnisanzeige:
'  Abfrage für Windows-Sonderschriftarten wie Symbol oder Wingdings,
'  dann stimmt die Analyse unter Umständen nicht:
Set Rng0 = Selection.Range
Selection.Find.ClearFormatting
With Selection.Find
    .Text = "[" & ChrW(61472) & "-" & ChrW(61695) & "]"
    .Replacement.Text = ""
    .Wrap = wdFindStop
    .Forward = True
    .MatchWildcards = True
'   Der MatchWildcards-Eintrag ist hier unbedingt nötig!!!
End With
Selection.Find.Execute
If Selection.Find.Found = True And InStr(Rng0, "(") > 0 Then
  Ergebnis = "Die Markierung enthält Windows-Sonderzeichen, " _
     & "die von diesem Makro möglicherweise fälschlich als ""("" interpretiert werden," _
     & vbCrLf & "hierfür eignet sich das Makro ""Hexcode_und_Name_ermitteln()""." _
     & vbCrLf & vbCrLf
Else
  Ergebnis = "Schriftart: " & f2 & "." & vbCrLf & vbCrLf
End If
Rng0.Select
'
For j = 1 To N
   Ergebnis = Ergebnis & "Der Unicode von """ _
      & x(j) & """ ist hexadezimal: " & z(j) & " und dezimal " _
```

```
        & Val("&H" & z(j) & "&") & "." & vbCrLf
  Next j
' MsgBox (Ergebnis)              ' Die MsgBox kann nur ca. 20 Zeilen.
   Abfrage = MessageBoxW(0, StrPtr(Ergebnis), StrPtr("Zeichenanalyse"), vbOKOnly)
End Sub
'-------------------------------------------------------------------------------------------------------
```

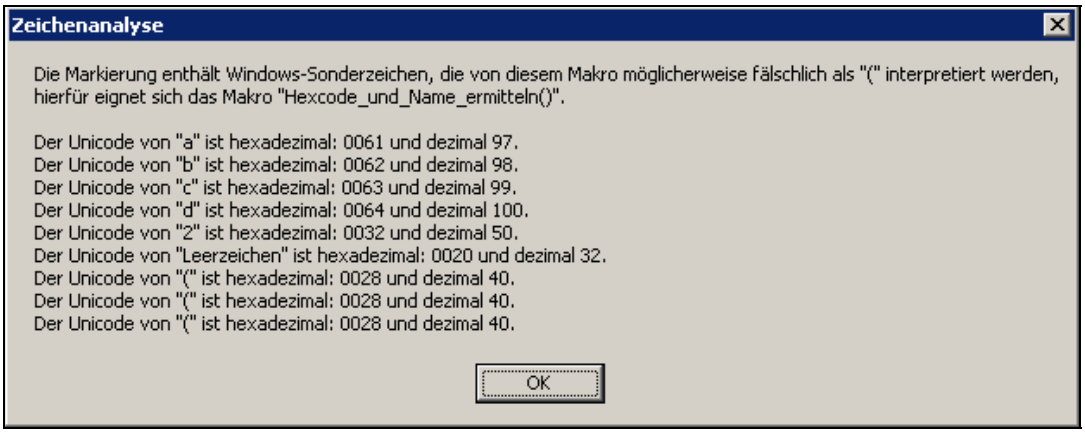
**Test:**

1) normale Buchstaben (abc) in Symbol-Schrift umgewandelt: abcd1 αβχ




**Screenshot: Analyse funktioniert!**

2) Ggriechische Buchstaben über "Einfügen → Symbol → Schriftart Symbol": abcd2 αβχ




**Screenshot: Analyse ist fehlerhaft (Klammer statt Sonderzeichen)**

Dieses Makro ist für Symbol-Zeichen und andere nicht-normgerechte Sonderschriftarten ungeeignet, es zeigt eine "Klammer" statt dem Sonderzeichen.

Dieses Makro zeigt bei einer Tabelle die Steuerzeichen "13" (= Absatzmarke) und "7" an, auch am Schluss der Tabelle. Dies entspricht der Analyse eines Strings (Zeichenkette), während das fertige Makro in Abschnitt 6 nur die Steuerzeichen "7" anzeigt, wie es dem Quelltext von Word 2000 bzw. Word 2003 entspricht.

## A.3 Makro zur Suche aller Zeichen aus höheren Unicode-Ebenen

Die Schrift- und Sonderzeichen Zeichen höherer Unicode-Ebenen findet man in Word nur durch die Suche nach den zugehörigen Surrogat-Paaren.

Dieses Makro zum Suchen aller Zeichen höherer Ebenen ist von Word 2000 bis Word 2010 einsetzbar. In einer Message-Box ("MsgBox") werden dann die Surrogate ausgegeben. Ab Word 2007 funktioniert es allerdings nicht mehr, wenn auf ein Zeichen höherer Ebenen ein kombinierendes Zeichen folgt.

```
Sub alle_Unicodes_höherer_Ebenen_suchen()
'   Erstellt am 1. Juni 2010 von Dipl.-Ing. Hanna-Chris Gast
    Dim Dateiende As Boolean
'
'   Gehe an den Anfang des Dokuments
    Selection.HomeKey Unit:=wdStory
    '
    ' Schleifenbefehl:
    Do Until Dateiende
    '
    '  Suche nach dem Codebereich der Surrogate
       Selection.Find.ClearFormatting
       Selection.Find.Highlight = False
       With Selection.Find
           .Text = "[" & ChrW(&HD800) & "-" & ChrW(&HDBFF) & "][" _
                   & ChrW(&HDC00) & "-" & ChrW(&HDFFF) & "]"
           .Replacement.Text = ""
           .Forward = True
           .MatchWildcards = True
     '     Der MatchWildcards-Eintrag ist hier unbedingt nötig!
           .ClearFormatting
           .MatchCase = False
     '     .Wrap = wdFindContinue      '(führt zu Endlosschleife!)
        End With
        Selection.Find.Execute
       '
        If Selection.Find.Found = True Then
            MsgBox ("Die Surrogate sind: " & Hex(AscW(Left(Selection, 1))) & " und " _
              & Hex(AscW(Mid(Selection, 2, 1))))
           Selection.Collapse Direction:=wdCollapseEnd
        Else
            Exit Do
        End If
    Loop
End Sub
'-----------------------------------------------------------------------------------------------
```

## A.4 Makro zur Suche aller kombinierenden diakritischen Zeichen

Das folgende Makro findet in Word 2000 alle diakritischen Zeichen im Bereich hex. 0300 bis hex. 036F (außer am Dateianfang und wenn es allein in einem Tabellenfeld steht, was aber extrem unwahrscheinlich ist). Lässt man das Fragezeichen in der Suchzeile weg, werden auch diakritische Zeichen gefunden, die allein in einem Tabellenfeld stehen, aber dafür ist der Cursor dann nicht mehr erkennbar.

```
Sub Das_nächste_Kombinierende_diakritische_Zeichen_suchen()
' Erstellt von Dipl.-Ing. H.-Chris Gast am 3. Juni 2010, jedoch  hier vereinfacht.
' Dieses Makro findet in Word 2000 alle kombinierenden diakritischen Zeichen,
' in Word 2003 und Word 2007 aber nur, wenn sie einzeln stehen.
'
' Suche nach dem Codebereich der kombinierenden diakritischen Zeichen
  Selection.Find.ClearFormatting
  Selection.Find.Highlight = False
  With Selection.Find
     '   Der Bereich hex. 0300 bis hex. 036F ist hier in Dezimalwerte umgerechnet.
     .Text = "?[" & ChrW(768) & "-" & ChrW(879) & "]"
     .Replacement.Text = ""
     .Forward = True
     .MatchWildcards = True
'     Der MatchWildcards-Eintrag ist hier unbedingt nötig!
     .ClearFormatting
     .MatchCase = False
  End With
  Selection.Find.Execute
End Sub
'-----------------------------------------------------------------------------------------------
```

### Auswertung:

Nicht immer sind neuere Wordversionen besser. In Word 2003 und Word 2007 sind mit diesem Makro kombinierende diakritische Zeichen nur zu finden, wenn nur ein einziges diakritisches Zeichen bei einem Buchstaben steht. Für Buchstaben mit mehreren diakritischen Zeichen ist die Suche nach dem diakritischen Zeichen wesentlich aufwendiger. In Word 2007 funktioniert dieses Makro nicht bei kombinierenden diakritischen Zeichen hinter Unicode-Zeichen höherer Ebenen.

## A.5 Details zu Chinesisch und Japanisch (sowie koreanische Silbenzeichen)

**CJK** (Abkürzung für die Chinesische, Japanische und Koreanische Schrift, englisch Chinese-Japanese-Korean) ist ein Begriff für Zeichen in der elektronischen Datenverarbeitung, welche im Chinesischen (Hànzì), im Japanischen (Kanji) und in Korea (Hanja) verwendet werden [3]. Die Bedeutung ist sprachabhängig. Es sind keine Buchstaben für Einzellaute.

Bei den in 6.3.4 aufgeführten CJK-Zeichen erhalte ich nach dem **Unicode-Consortium** [2] (siehe Bild A.1) bzw. nach **ISO/IEC 10646** (siehe Bild A.2) statt endgültiger Schriftzeichennamen nur die entsprechenden Codes für Chinesisch ("C"), Japanisch ("J"), Koreanisch ("K") und Vietnamesisch ("V"). Die Angaben wie "G0-5827" oder "T1-4561" sind die Codes in den jeweiligen Landesnormen. Inzwischen wird auch in China Unicode verwendet; die Codes der dortigen nationalen Norm entsprechen dann quasi einer "UTF-Codierung" [3].

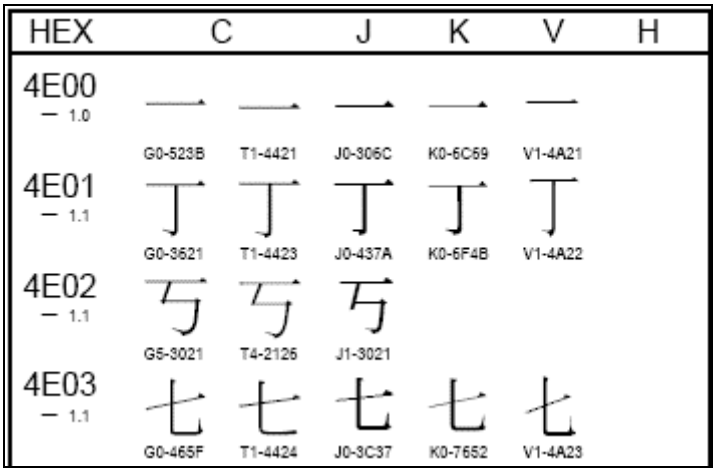



**Bild A.1: Auszug aus http://www.unicode.org/charts/PDF/U4E00.pdf, Seite 451, Unicode 6.0, 2010** [2]

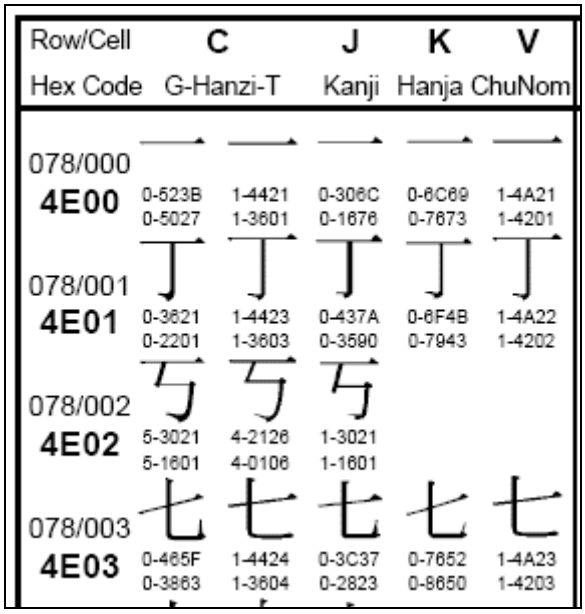



**Bild A.2:** Auszug aus: **ISO/IEC 10 646:2003**, *Information technology — Universal Multiple-Octet Coded Character Set (UCS) aus Datei: ISO_2003_charts_4E00_680F_CJKUnifiedMain1.pdf, ISO/IEC 10646:2003 (E), Seite* **382**

Ich denke, mit diesen Angaben ist dem Anwender des Analyse-Makros nicht viel geholfen, und habe darauf verzichtet, hierfür das Makro zu erweitern.

Im Folgenden wird das Beispiel für die chinesischen Zeichen aus Bild 6.6 **(𠂇丁乛七漢字汉字)** weiter untersucht. Im Internet fand ich mehrere Übersetzungsportale für chinesische Zeichen, von denen ich Google am besten fand:

Mit "**Google-Übersetzer**" erhalte ich http://translate.google.com [13] (siehe Bild A.3):


**Bild A.3: Screenshot des Google-Übersetzers für Chinesisch (22.Mai 2011)**

**Übersetzung** aus dem Chinesischen: **"Bingdingkaoqi chinesischen Schriftzeichen"**
**Umschrift** aus dem Chinesischen: **"Bīng dīng kǎo qī hànzì hànzì"**

**Übersetzung** aus dem Japanischen: ***Kanji Kanji sieben*** 𠂇 丁 乛
**Umschrift** aus dem Japanischen: 𠂇丁 乛 Nana kanji 汉字

## Auswertung für das obige Zitat:

— Die Umschrift aus dem Chinesischen ist vollständig, aber es wird nur ein Teil der Zeichen übersetzt.
— Die Umschrift aus dem Japanischen ergibt nur für einige Zeichen der Textauswahl ein Ergebnis.
— Umschrift und Übersetzung aus dem Koreanischen ergaben nichts.

Umgekehrt erhalte ich bei einem japanischen Text wie "日本語" nur für "Japanisch-Deutsch" im Google-Übersetzer eine eindeutige Übersetzung (nämlich "Japanisch").

Hier ist für Anwender ohne jede Kenntnis dieser fernöstlichen Sprachen vielleicht die Anwendung der Übersetzungsfunktion von Google eine sinnvollere Lösung als mein Analyse-Makro. Zwar ist das auch noch nicht optimal, aber bisher das beste, was ich fand.

Eine gebürtige chinesische Studentin, die ich im Januar 2011 in Berlin befragte, sagte mir, sie kenne von **𠂇丁乛七漢字汉字** einen Teil der Zeichen nicht. Somit ist Google nicht schlechter als der chinesische Durchschnittsakademiker.

## A.6 Problem mit Copy-Paste in Word 2010

Das Analyse-Makro macht in Word 2010 einen Fehler bei der Analyse eines einzelnen Leerzeichens, weil dort der "Paste"-Befehl in die Zwischendatei nicht funktioniert. Und zwar wird in Word 2010, anders als in Word 2000, kein Leerzeichen per Copy und Paste zwischen zwei Absatzmarken eingefügt.

Ferner wurde bei Absätzen, die in Word 2010 erzeugt wurden, vom Analyse-Makro die Absatz-Schriftart nicht erkannt, im Gegensatz zu Absätzen, die mit Word 2000 erzeugt wurden. Und zwar wird in Word 2010, anders als in Word 2000, per Copy und Paste die Absatzformatierung nicht immer mit-übertragen.

**Abhilfe:**
Bei einheitlicher Schriftart wird die Formatierung zusätzlich übertragen, und bei einzelnem Leerzeichen wird das Leerzeichen gesondert in die Zwischendatei eingefügt. Dies funktioniert auch im alten Word.

```
  ' ************************************************
' In Word 2010 wird die Absatzschriftart unzulänglich übertragen, und
' einzelne Leerzeichen werden nicht mehr vor Absatzmarken eingefügt (2. Juni 2014):
Dim B_Leerzeichen As Boolean: B_Leerzeichen = False
Dim B_Font As Boolean: B_Font = False
  If Selection.Text = " " Then B_Leerzeichen = True
  If Selection.Font.Name <> "" Then B_Font = True
  Selection.CopyFormat
'  ************************************************
'  Zwischendatei "doc2" erzeugen und dort die zu untersuchende Zeichenkette einfügen:
  Set doc1 = ActiveDocument
  '' B_Zwischendatei = True
  Application.ScreenUpdating = False    ' Zwischendatei soll unsichtbar bleiben.
  Set doc2 = Documents.Add  '  DocumentType:=wdNewBlankDocument
  Selection.WholeStory
  Selection.Delete  ' Briefkopf entfernen
'  ************************************************
'  In Word 2010 Absatzschriftart und Einzel-Leerzeichen ggf. separat übertragen:
  Selection.WholeStory
  If B_Leerzeichen = True Then
    Selection.TypeText Text:=" "
  Else
    Selection.Paste
  End If
  Selection.WholeStory
  If B_Font = True Then
   Selection.PasteFormat
  End If
'  ************************************************
```

# Anhang B
# Weitere Verbesserungen für Word 2000

In diesem Anhang gebe ich einige für Word 2000 nützliche Makros an (die natürlich auch in neueren Word-Versionen funktionieren). Damit kann Word 2000 eigentlich alles, was ich brauche. Ich sehe keine Notwendigkeit, mich privat auf eine neue Word-Version umzustellen. Weitere Makros für Word siehe auch [24].

## B.1 Beseitigen der bei Word 2000 lästigen Symbol-Leiste "Web"

In Word 2000 tauchte beim Anklicken im Inhaltsverzeichnis oder beim Anklicken eines Querverweises jedes Mal die Symbolleiste "Web" wieder auf. In Google fand ich das Problem häufig gelistet, jedoch überwiegend nur mit der Fragestellung und fast gar nicht mit einer brauchbaren Lösung. Deshalb hier nun eine Lösung, die sich bei mir bewährte (dreimal im Internet gefunden[35], unter anderem in [16]).

In der Vorlage "Normal.dot" ist die folgende Datei

```
Sub AutoExec()
  CommandBars("Web").Enabled = False
End Sub
```

im Ordner "ThisDocument" unterzubringen (siehe Bild B.1):

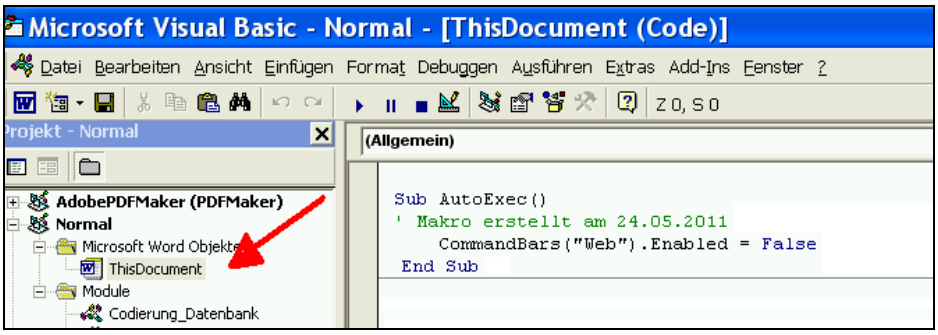



**Bild B.1: Screenshot des Makro-Explorer**

Dieses Makro funktioniert nicht für die Symbol-Leiste des "PDFMaker" von Adobe. Diese kann erst unmittelbar beim Öffnen einer Datei deaktiviert werden.

## B.2 Inhalt unformatiert einfügen

Eines der von mir am häufigsten gebrauchten Makros dient zum Einfügen von Inhalt ohne Formatierung.

```
Sub inhalte_einfügen()
  Selection.PasteSpecial Link:=False, DataType:=wdPasteText, _
        Placement:=wdInLine, DisplayAsIcon:=False
End Sub
```

Als Tastenkürzel habe ich mir hierfür "Strg + i" gewählt.

---

[35] Hiermit sei A. Gumtow gedankt, der mir aus den hunderten von Google-Fundstellen die richtige herausgesucht hat.

## B.3 Tastenkürzel anpassen in Word 2000

Ich habe mir in Word 2000 den Befehl für "Seitenansicht" an Word 2003 (Alt + D, A) und den Befehle für "Speichern unter" an Adobe (Strg + Shift + S) angepasst (siehe auch 7.2).

Insgesamt finde ich den Umgang mit dem alten Word 2000 und dem zugehörigen Photo-Editor angenehmer als mit dem Nachfolge-Systeme Word 2003 bis Word 2010.

## B.4 Tabellen auch in Word 2000 in "Raster 5" formatieren

Bei dem Makro-Befehl "Gitternetz 5" in Word 2000[36] gehen die Textfarben sowie die Eigenschaften "kursiv" und "fett" verloren, egal wie man die Optionen ("Schalter") setzt.

Soll in Word 2000 eine Tabelle nachträglich mit "Raster 5" formatiert werden, wie es sonst erst ab Word 2003 möglich ist, mit Erhalt von farbigem Text und Erhalt des Schriftschnitts "*kursiv*", kann das mit folgendem Makro erreicht werden [27]:

```
Sub Tabelle_formatieren_wie_Raster_5()
'  von "http://www.schreibbuero-richter.de", 28. März 2012.
   Dim tbl As Word.Table
   Dim cel As Word.Cell
'
   Set tbl = Selection.Tables(1)
'
   ' automatische Größenänderung ausschalten
   If tbl.AllowAutoFit Then
      tbl.AllowAutoFit = False
   End If
'
   ' "Abstand zwischen Zellen" deaktivieren/auf 0 setzen
   If tbl.Spacing > 0 Then
      tbl.Spacing = 0
      ' nur den Abstand auf 0 stellen, reicht leider nicht
      ' auch die Checkbox muss deaktiviert werden,
      ' dafür gibt es in VBA keinen Schalter, deswegen so:
      With Dialogs(1080)
         If .AllowSpacing <> 0 Then
            .AllowSpacing = 0
            .Execute
         End If
      End With
   End If
'
   ' erst Innenlinien:
   With tbl.Borders(wdBorderHorizontal)
      .LineStyle = wdLineStyleSingle
      .LineWidth = wdLineWidth075pt
      .Color = wdColorBlack
   End With
   With tbl.Borders(wdBorderVertical)
```

[36] als Makrobefehl: "Selection.Tables(1).AutoFormat Format:=wdTableFormatGrid5,. . ."

```
      .LineStyle = wdLineStyleSingle
      .LineWidth = wdLineWidth075pt
      .Color = wdColorBlack
   End With
'
   ' dann Außenlinien:
   With tbl.Borders(wdBorderBottom)
      .LineStyle = wdLineStyleSingle
      .LineWidth = wdLineWidth150pt
      .Color = wdColorBlack
   End With
   With tbl.Borders(wdBorderTop)
      .LineStyle = wdLineStyleSingle
      .LineWidth = wdLineWidth150pt
      .Color = wdColorBlack
   End With
   With tbl.Borders(wdBorderLeft)
      .LineStyle = wdLineStyleSingle
      .LineWidth = wdLineWidth150pt
      .Color = wdColorBlack
   End With
   With tbl.Borders(wdBorderRight)
      .LineStyle = wdLineStyleSingle
      .LineWidth = wdLineWidth150pt
      .Color = wdColorBlack
   End With
'
   ' 1. Zeile/n zur Kopfzeile machen
   tbl.Range.Cells(1).Select
   Selection.Collapse wdCollapseStart
   Selection.SelectRow
   Selection.Rows.HeadingFormat = True
End Sub
```

## B.5 Befehl "Edit Field" auch für Word 2000

In Word 2000 fehlte noch die Funktion, dass man mit einem Mausklick rechts ein vorhandenes Feld bearbeiten kann.

Das folgende Makro kann hier ein bisschen Abhilfe schaffen. Glücklicherweise funktioniert neuerdings (Stand: Sommer 2013) der SendKeys-Befehl in Word 2000 unter Windows XP, was bisher fast unmöglich war. Ein bisschen ist das immer noch eine etwas heikle Sache. Aber nur mit SendKeys kann man in den Word-Dialog für Felder den Feldtyp eingeben, soviel ich weiß. Der entscheidende Hinweis: SendKeys funktioniert bei Aufruf des Makros durch Tastenkürzel nur dann, wenn das Makro vor dem Sendkeys-Befehl eine Pause von 1 Sekunde macht!

In Word 2010 funktioniert das folgende Makro gut bei Inhaltsverzeichnissen, ist aber nicht brauchbar bei anderen Feldern wie Datumsfeldern.

Für das folgende Makro habe ich mir ein Tastenkürzel festgelegt.

```
Sub Feld_Dialog_halbgefüllt_öffnen_Word_2000()
' H. Chris Gast, 10. August 2013.
' Dieses Makro ersetzt teilweise den in Word 2000 noch fehlenden Befehl "Edit Field",
' der ab Word 2003 mit rechtem Mausklick wählbar ist.
'
' Inhaltsverzeichnis und Datumsangaben können so recht gut bearbeitet werden.  Bei Querverweisen
' gehen die vorhandenen Einstellungen verloren und müssen neu gewählt werden.
' Die alten Schalter werden nicht ausgelesen. Das Makro ist aber in Word 2000 trotzdem hilfreich.
' ****
Dim a1 As String ' der gesamte ausgelesene Feldcode
Dim a2 As String ' der einfache Feldcode ohne Zusätze
Dim b1 As String ' der Zusatz im Feldcode hinter dem Leerzeichen
'  ************************************
'  Das Feld in oder an der markierten Stelle markieren:
  Selection.Expand Unit:=wdSentence
  If Selection.Range.Fields.Count = 0 Then
    MsgBox "Es ist kein Feld markiert!"
    Exit Sub
  End If
  ' Beim Inhaltsverzeichnis ist unter Umständen nur eine Zeile mit Feldcode "Hyperlink" markiert;
  ' mit den folgenden 2 Befehlen wird die Markierung an den Beginn des Inhaltsverzeichnisses
  ' verschoben.
  ActiveDocument.Fields.ToggleShowCodes
  ActiveDocument.Fields.ToggleShowCodes
  ' Wenn nicht genug markiert ist, Markierung erweitern:
  If Selection.Range.Fields.Count = 0 Then
    Selection.Range.Words(1).Fields(1).Select
  Else
    Selection.Range.Fields(1).Select
  End If
'  ****************************************
'  Feldcode-Stücke aus dem markierten Feld auslesen:
  a1 = Selection.Range.Fields(1).Code       ' zum Beispiel a1 = "CREATEDATE  \@ "d. MMMM yyyy"
  a2 = Trim(Left(a1, InStr(3, a1, " ")))    ' zum Beispiel: a2 = "CreateDate" (ohne Leerzeichen)
  b1 = Trim(Mid(a1, InStr(3, a1, " ") + 4)) ' zum Beispiel: b1 = "d. MMMM yyyy"
  b1 = Replace(b1, ChrW(34), "")            ' Gänsefüßchen entfernen!
'  ****************************************
'  Je nach Feldtyp unterschiedliche Dialoge aufrufen:
'  ****************************************
  If UCase(a2) = "TOC" Then
    ' Inhaltsverzeichnis:
    With Dialogs(wdDialogInsertTableOfContents)
```

```
      If .Display = -1 Then
        ' User selected OK, remove old and insert new
        Selection.Delete
        .Execute
      End If
    End With
    Exit Sub
  ElseIf InStr(LCase(a1), "ref _ref") > 0 Then
    ' Querverweis auf andere Abschnitte:
    With Dialogs(wdDialogInsertCrossReference)
      If .Display = -1 Then
        ' User selected OK, remove old and insert new
        Selection.Delete
        .Execute
        Selection.Fields.Update
      End If
    End With
    Exit Sub
  End If
' ******************************************
' Per Sendkey im Dialog "Dialogs(wdDialogInsertField)" den ermittelten Feldcode einfügen:
' ******************************************
' Bei Start des Makros per Tastenkürzel vor SendKeys-Befehl 1 Sekunden warten
' (siehe auch http://support.microsoft.com/kb/138624):
  Dim tt As Double
  tt = Timer
  Do While Timer < tt + 1#      ' 1,0 Sekunde
    DoEvents
  Loop
' ********************
' Die SendKeys-Befehle sind je nach Word-Version unterschiedlich:
"   If Val(Application.Version) = 9 Then
    ' Word-Version ist  Word 2000.
    ' (die SendKey-Befehle werden am besten vor dem Dialog eingegeben):
    SendKeys "(%b)"           ' Häkchen entfernen bei "Formatierung beibehalten"
    SendKeys "(%n)" & a2      ' Codenamen einfügen
    If UCase(a2) = "REF" Then
      ' Verweisung auf Textmarke:
      " b1 = Left(Mid(a1, InStr(2, a1, " ") + 1), InStr(6, a1, " ") - 6)
      SendKeys "(%o)(%m)(%n)" ' & b1 ", True   ' & "(%z)"
     " SendKeys "(%z)"
    Else
      ' Datumsangaben und anderes.
      SendKeys "(%o)" " & b1 & "(%z)"   ' es wurde hier die falsche Option eingefügt!
    End If
    With Dialogs(wdDialogInsertField)
      .Show
    End With
  ' **************
"   End If
  ActiveDocument.Fields(1).ShowCodes = False
End Sub
' ---------------------------------------------------------------------------------------------------------
```

**Zu beachten ist:**
Um SendKeys in einem Makro zu verwenden, das durch ein Tastenkürzel gestartet wird, muss man eine Warteschleife von **einer** Sekunde einbauen.

Die SendKeys werden am besten vor die Zeile mit dem "WdDialog" gestellt. Ein "True" für "Warten" hinter die SendKeys-Befehle führte zu Fehlfunktionen. Zwischen "Sendkeys" und "WdDialog" darf keine Verzögerung eingebaut werden.

## B.6 Lösung zu dem Problem mit "Gehe zu Seite" in Word 2000

Bei "Gehe zu Seite" ist in Word 2000 das Eingabefeld nicht leer. Es steht darin die letzte Seiteneingabe. Ich muss dann diese erst löschen, bevor ich die neue Zielseitennummer eingeben kann.

Endlich fand ich dafür eine Abhilfe. Ich programmiere ein Ersatzmakro für diesen Befehl:

```
Sub GoTo_Eingabefeld()
'  Erstellt von H. Chris Gast, 30. Juni 2011.
'  Ruft Dialogfeld auf, ohne dass in Word 2000 fehlerhaft das Eingabefeld schon einen
'  Zahlenwerte vom letzten Mal enthält.
  Dialogs(wdDialogEditGoTo).Show
End Sub
' ---------------------------------------------------------------------------------------------------
```

Es zeigte sich, dass bei diesem Makro das Eingabefeld stets leer ist.

Ich wies diesem Makro das Tastenkürzel "Strg + G" zu. Statt des fehlerhaften eingebauten Befehls wird nun stets das neue Makro ausgeführt.

```
Sub Tastenkürzel_für_GoTo_Eingabefeld_zuordnen()
' Erstellt von H. Chris Gast, 3. Juli 2011
  With Application
     .CustomizationContext = NormalTemplate
'     .CustomizationContext = ActiveDocument
     .KeyBindings.Add KeyCode:=BuildKeyCode(wdKeyControl, wdKeyG), _
          KeyCategory:=wdKeyCategoryCommand, _
          Command:="GoTo_Eingabefeld"
  End With
  MsgBox "Mit ""Strg + G"" rufen Sie das Menü auf, um zu einer Seite zu gehen."
End Sub
' ************************
```

*(ich weise die gewünschten Tastenkürzel lieber von Hand zu*
*oder bei einem neuem Computer*
*per Formblatt zur Übertragung von allen Tastenkürzeln, siehe* [26]:

## B.7 Suche nach Text mit Farbschattierung in Word 2000

Mit dem folgenden zwei Makros kann man auch in Word 2000 nach einer oder mehreren Hintergrundsfarben suchen, was sonst mit dem normalen Suchbefehl nicht funktioniert.

```
Sub Farbe_in_Text_suchen()
' H. Chris Gast, 11. Januar 2015.
' Suche die nächste Textstelle mit (irgendeiner) Farbe.
' Es wird nicht nach Tabellenschattierungen gesucht.
' Funktioniert auch bei Word 2000.
' Man kann in Word 2000 zwar nach "Highlight" suchen, aber die Suche nach "Shading" mit
' "Selection.Find.Font.Shading.BackgroundPatternColor = …" funktioniert
' nicht in Word 2000.
' Deshalb die folgende Lösung "For Each...Characters".
' Dabei wird auch gleichzeitig "Highlight" und Schriftfarbe gesucht.
' **********
Dim aChr As Range
Dim Rng0 As Range
Set Rng0 = Selection.Range
Rng0.SetRange Start:=Rng0.End, End:=ActiveDocument.Range.End
  '
  For Each aChr In Rng0.Characters
    If aChr.HighlightColorIndex <> wdAuto _
    Or aChr.Font.Shading.BackgroundPatternColor <> wdColorAutomatic _
    Or aChr.Font.Shading.ForegroundPatternColor <> wdColorAutomatic _
    Or aChr.Font.Color <> wdColorAutomatic Then
      aChr.Select
      ' ' Hier gegebenfalls Abfragen nach einer bestimmten Farbe einfügen.
      MsgBox "Hier wurde eine farbige Text-Stelle gefunden!"
      Exit Sub
   End If
  Next aChr
MsgBox "Keine (weiteren) farbigen Textstellen im Text."
End Sub


Sub hellrosa_Schattierung_suchen()
' H. Chris Gast, 11. Januar 2015.
' Suche die nächste Textstelle mit einer von zwei bestimmten hellrosa Farbschattierungen.
' Funktioniert auch bei Word 2000.
' Will man alle hellrosa-Töne finden, muss man den "ColorIndex = 5" abfragen, dann
' findet das Makro aber auch Farbtöne, die meines Erachtens eher lila aussehen.
' **********
Dim aChr As Range
Dim Rng0 As Range
  Set Rng0 = Selection.Range
  Rng0.SetRange Start:=Rng0.End, End:=ActiveDocument.Range.End
  '
  For Each aChr In Rng0.Characters
"   If Selection.Font.Shading.BackgroundPatternColorIndex = 5 Then
    If aChr.Font.Shading.BackgroundPatternColor = 16764159 _
    Or aChr.Font.Shading.BackgroundPatternColor = wdColorRose Then
      aChr.Select
      MsgBox "Hier wurde eine hell-rosa Schattierung gefunden!"
      Exit Sub
    End If
  Next aChr
  MsgBox "Keine (weiteren) rosa Textstellen im Text."
End Sub
```

# Anhang C
# Abstract in English

"Extension of Word 2000 for the implementation and analysis of **Unicode characters in Higher Planes** like Hieroglyphs and other archaic characters; **Macro** for Word 2000 to Word 2010 **for Analysing** all kind of characters in a Word text with showing off the English character names according to ISO/IEC 10646".

This book offers macros for the extension of Word 2000 to make it able to handle Unicodes of higher Unicode-planes like Hieroglyphs (e. g.  ) and other archaic characters.

For this Windows XP had to be adapted to cope with Surrogates since Windows XP works with UTF 16. Then I could search for fonts. There is no such thing as one single font for all characters but only for a rather small range (see chapter 4).

Now I had to enlarge Word 2000 by macros to write those characters of higher Unicode planes. I added the function of inserting Unicode characters by writing the hexcode and pressing Alt + C, which is normally not yet part of Word 2000, and the function of inserting characters by using the decimal code (see chapter 5).

Chapter 6 contains a **Macro for Analysing** characters in all versions of Word (Word 2000 to Word 2010) which analyses up to 120 characters in a Word-document with one mouse click. It gives out the Unicode numbers (code point) of the marked characters and can also display their English names according ISO/IEC 10646.

For example: When marking with the cursor for example "Unicode.org" and starting the macro (I have assigned to the macro the key-combination "Control + Alt + C"), I will get the codes (see Fig. C.1, left hand side). In most cases this result is enough. When pressing "Ja" ('yes'), the macro looks up the English names of the characters in the Internet on the webpage of Unicode.org (see Fig. C.1, right hand side):

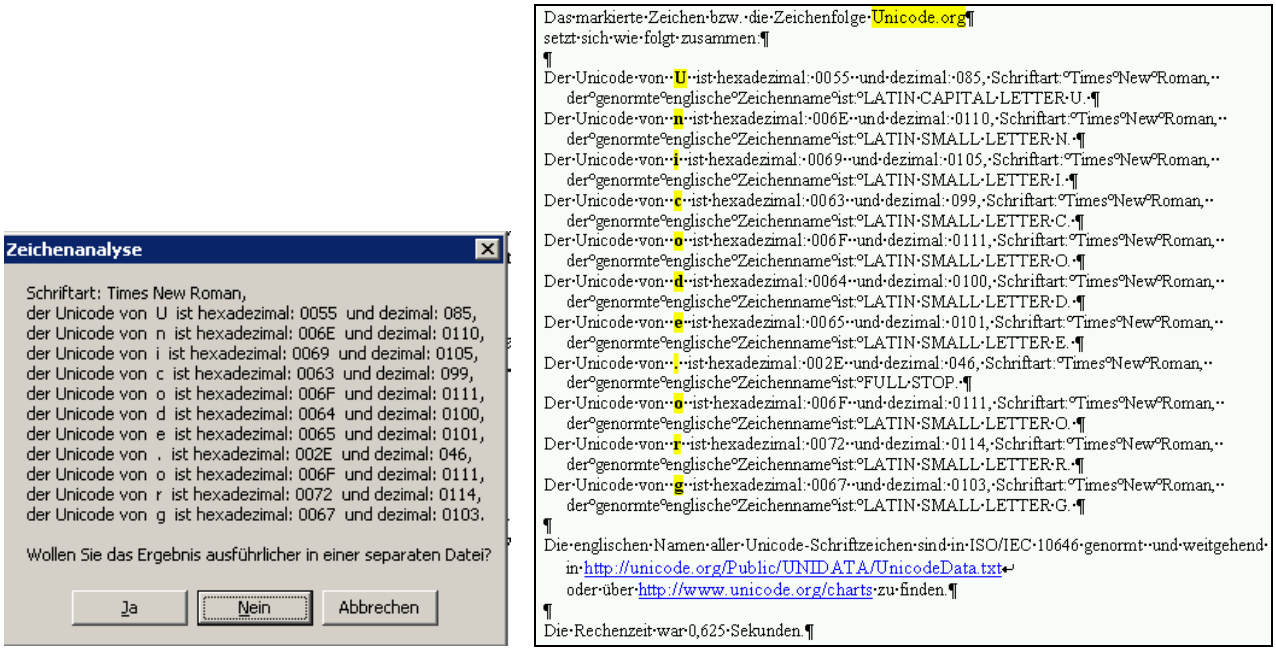



**Fig. C.1 Screenshot of the analysis**

This also works for very unknown characters like: ✕ ☎ ☎ (see Fig. C.2).The macro gives the also meaning of Symbol and Wingding characters. Perhaps here I should press "Ja" to get the name also of the first character of the marked sample [37] which belongs to a higher Unicode plane.

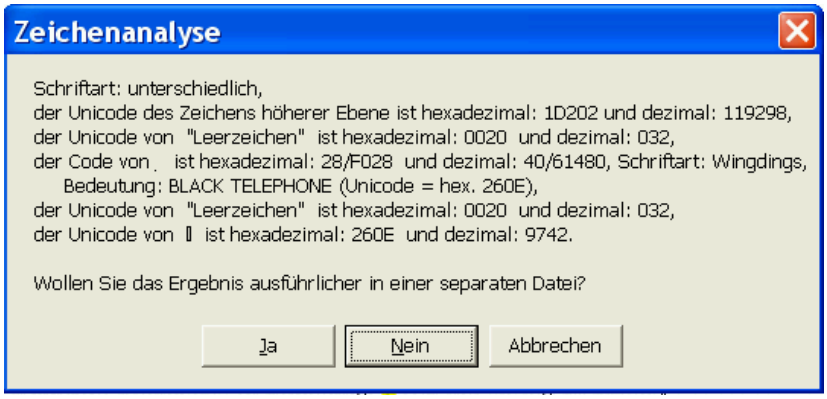



**Fig. C.2: Screenshot for the analysis of unknown characters**

The analysing macro recognises characters with diacritical marks (see Fig. C.3). The advantage in comparison to Word (even Word 2010) is that the macro always analyses the character and the combining accent together. There is no more danger of analysing only one part of the combined letter.

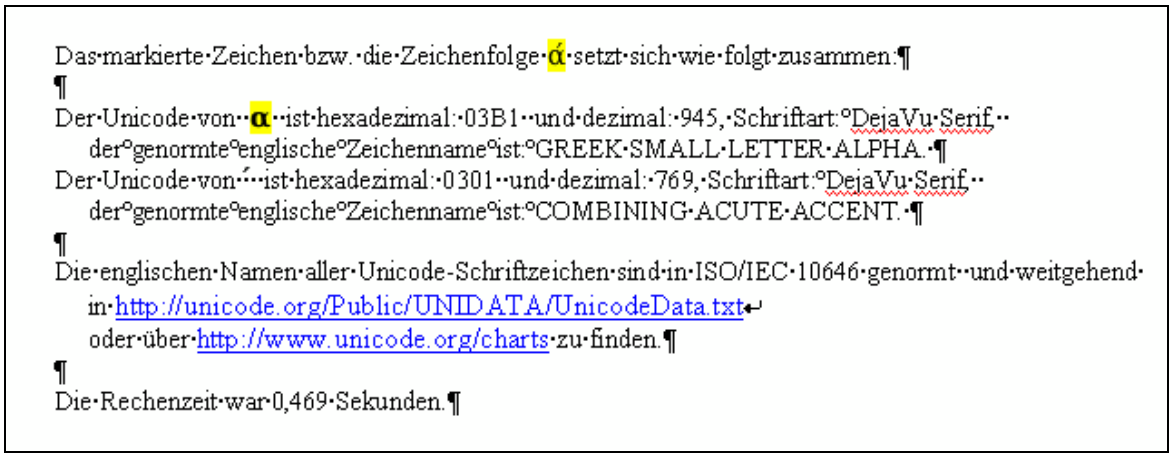



**Fig. C.3: Screenshot of the analysis of a small letter Alpha with acute**

[37] GREEK VOCAL NOTATION SYMBOL-3

Fig. C.4 shows the analysis of two Hieroglyphs as an example of a higher Unicode-plane:

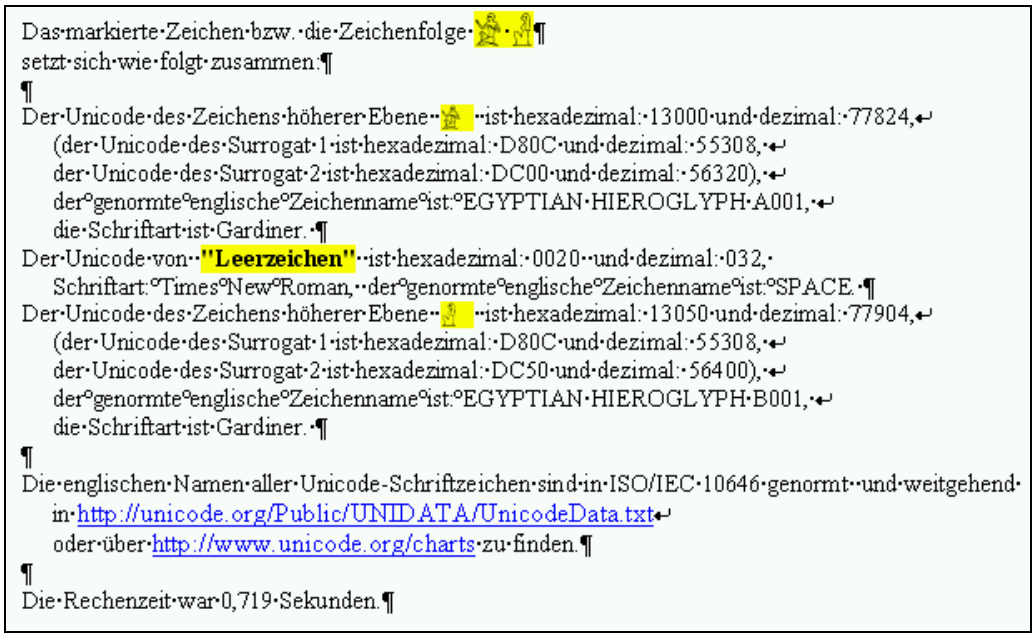



**Fig. C.4: Two Hieroglyphs as a sample for a higher Unicode plane**

The macro can also handle combination from left to right like Latin characters and from right to left like Arabic and Hebrew characters (see Fig. C.5):


**Fig. C.5: A Hebrew letter with Latin letters in the same line**

Last not lease the analysing macro also recognizes special marks (signs) like NoBreakSpace, NoBreakHyphen, Soft Hyphen, Tabulator, Paragraph, etc. even when they are only Control characters in Word, see Fig. C.6 ( - - ).

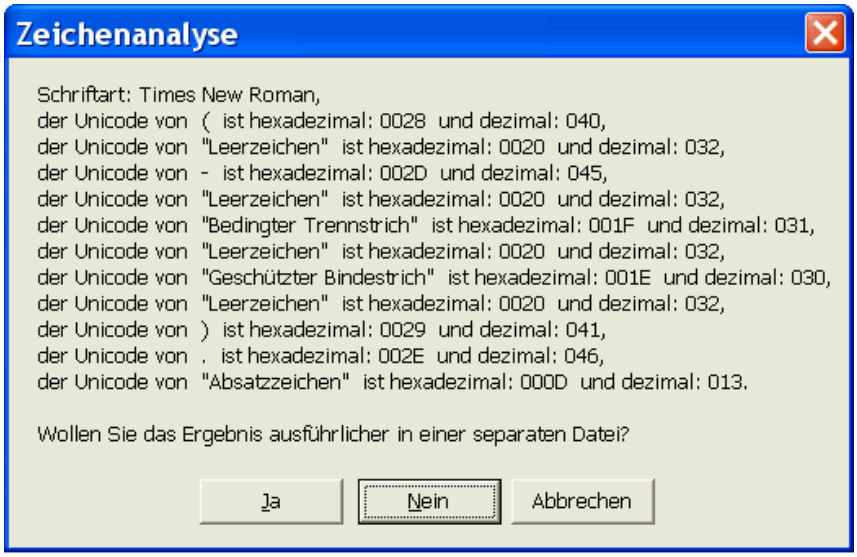



**Fig. C.6: Special marks**

The limitation of my analyzing macro for Word is, that it cannot give you the names of Chinese, Japanese or Korean syllable-characters.

This booklet can be downloaded:

*[1]* Gast, Hanna-Chris: "Erweiterung von Word 2000 zur Darstellung und Analyse von Schriftzeichen höherer Unicode-Ebenen sowie Erstellung eines Makros für Word 2000 bis Word 2010 zur Analyse von Schrift- und Sonderzeichen mit Ausgabe des Schriftzeichen-Namens nach ISO/IEC 10646";
Beilage zum Siebener-Kurier Nr. 60 (August 2010), ISSN 0948-6089;
*In the Internet as download (Version 2014):*
http://www.siebener-kurier.de/chris-aufsaetze/Word-Erweiterung-Unicode-Makros.pdf.

*The macro for **analysing characters** of chapter 6 for Word 2000 up to Word 2010 can be separately downloaded in ready-to-use-word-file (within a Zip-File):*
http://www.siebener-kurier.de/chris-aufsaetze/Zeichen-Analyse-kompakt.zip.
*The macro for Analysing therein has the shortcut "AltGr + C".*

**Unfortunately this book is only available in German language.**

Hanna-Chris Gast

P.S. If you cannot recognize a character in a website or in a pdf-document, just copy it into a word-file and analyse it with this macro. The macro also works well, if you only see "blank boxes" or other "blank place-holders" like "□ " or "□" on the screen.

# Literaturquellen

**ISO/IEC 8859-1**, *Information technology – 8-bit single-byte coded graphic character se*ts – Part 1: Latin alphabet No. 1

**ISO/IEC 10 646,** *Information technology — Universal Coded Character Set (UCS)*
http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html.
**ISO/CEI 10 646:2003**, *Technologies de l'information — Jeu universel de caractères codés sur plusieurs octets (JUC)*; ***(veraltet**, aber für französische Namen bis Änderung 2 noch von Interesse).*

[2] **Unicode-Consortium**, (die Schriftzeichen nach ISO/IEC 10646).
The Unicode Consortium, P.O. Box 391476, Mountain View, CA 94039-1476, U.S.A., http://www.unicode.org/charts, und http://www.unicode.org/Public/UNIDATA
(Stand: Version Unicode 7 (2014);
**Textdatei mit Unicode-Liste für Download**:
http://www.unicode.org/Public/UNIDATA/UnicodeData.txt (Stand: 2014);
Französische Bezeichnungen für Unicode-Zeichen, soweit vorhanden, siehe
http://www.unicode.org/fr oder http://www.unicode.org/fr/charts und
http://www.unicode.org/fr/charts/PDF (alle französischen Tabellen nur bis Stand 2007).

[3] **Wikipedia**, Wikimedia Foundation Inc., P.O. Box 78350, San Francisco, CA 94107-8350, USA; — Wikimedia Deutschland, Gesellschaft zur Förderung Freien Wissens e. V., Postfach 30 32 43, 10729 Berlin; deutsche, englische und griechische Wikipedia-Seiten, u. a.http://de.wikipedia.org/wiki/Liste_der_Unicode-Blöcke; (8. Mai 2010).

[4] **FileFormat.Info**, Andrew Marcuse, 317 Buck Lane, Suite 853, Haverford, Pennsylvania 19041, USA, http://www.fileformat.info (10. April 2010),

[5] **Cardo.ttf,** David J. Perry, Fonts for Scholars, Rye High School, Rye, New York USA, Version April 2011.
*Anmerkung: Brauchbare kostenlose Schrift für Griechische Schriftzeichen.*

[6] **Code2000.ttf**, James Kass; 2807 Erskine Creek Road Space 93; Lake Isabella, California 93240, U.S.A.; (13. März 2010).
*Anmerkung: kostet ab 5 Dollar; nur kurz getestet, dann aber nicht verwendet und gelöscht*.

[7] **Code2001.ttf,** James Kass; 2807 Erskine Creek Road Space 93; Lake Isabella, California 93240, U.S.A., *Anmerkung: kostenlose Schrift, kann Ägäische Zahlzeichen*.

[8] **Unicode Fonts for Ancient Scripts**", http://users.teilar.gr/~g1951d, "fonts designed by George Douros" 2009, TEI, Technological Educational Institution of Larissa (Λάρισα), (31. März 2010).

[9] **Musica.otf**, in: "Unicode Fonts for Ancient Scripts", http://users.teilar.gr/~g1951d, "fonts designed by George Douros" 2009, TEI, Technological Educational Institution of Larissa (Λάρισα), (31. März 2010).

[10] **Aegean.otf**, in: "Unicode Fonts for Ancient Scripts", http://users.teilar.gr/~g1951d, "fonts designed by George Douros" 2009, TEI, Technological Educational Institution of Larissa (Λάρισα), (2. Mai 2010).

[11] **Gardiner.otf**, in: "Unicode Fonts for Ancient Scripts", http://users.teilar.gr/~g1951d, "fonts designed by George Douros" 2009, TEI, Technological Educational Institution of Larissa (Λάρισα), (8. Mai 2010).

[12] **UTF-8-Codetabelle mit Unicode-Zeichen**, Tomas Schild, Lange Furche 31, 72072 Tübingen, http://www.utf8-zeichentabelle.de (25. April 2010).

[13] **Google**, Google-Übersetzer: Google Inc., 1600 Amphitheatre Parkway, Mountain View, CA 94043, USA http://www.google.de/intl/de/impressum.html, bzw.
http://translate.google.com; Aufruf zuletzt 22. Mai 2011.

[14] **Microsoft Corporation**, One Microsoft Way, Redmond, WA 98052-7329, USA; bzw. Microsoft Deutschland GmbH, Konrad-Zuse-Straße 1, 85716 Unterschleißheim; http://www.microsoft.com , http://support.microsoft.com/kb/161304, http://msdn.microsoft.com/en-us/library/dd374069(v=VS.85).aspx (28. April 2010).

[15] Philipp Stephani, "Unicode", in AktiveVB, Kraichgaustraße 2, 76698 Ubstadt-Weiher; http://www.activevb.de/rubriken/kolumne/kol_20/unicode.html (26. Juni 2010).

[16] Dave Rado: "**Finding and replacing symbols**", Beth Melton and Dave Rado: "How to stop the web toolbar from jumping up at you whenever you click on a page number in the table of contents", in "The Word MVP Site", Unit 1407, 83-85 Spring Street, Bondi Junction, Sydney, NSW 2022, Australien; http://word.mvps.org/ bzw. http://word.mvps.org/faqs/MacrosVBA/ (30. April 2009 und 24. Mai 2011).

[17] **Informationsverbund Deutschschweiz (IDS)** Katalogisierungsregeln IDS (KIDS) Hauptbibliothek Universität Zürich, Strickhofstr. 35, CH-8057 Zürich, SCHWEIZ; http://www.informationsverbund.ch bzw. *für Hebräisch*: (Download September 2009). **Achtung:** Inzwischen durch Querverweis auf die Norm DIN 31636 ersetzt (21. September 2010).

[18] **Alan Wood:** "Wingdings character set and equivalent Unicode characters", Stand 26. Januar 2011, http://www.alanwood.net/demos/wingdings.html.

[19] **Apple-Liste** für Zeichen der Schriftart "Symbol", Stand 5. April 2005. http://unicode.org/Public/MAPPINGS/VENDORS/APPLE/SYMBOL.TXT;

[20] **Adobe-Liste** für Zeichen der Schriftart "Symbol", Stand: 30. März 1999; http://unicode.org/Public/MAPPINGS/VENDORS/ADOBE/symbol.txt.

[21] **Unicode-Erklärung zum Phi-Symbol**, Unicode Technical Report #25, UNICODE SUPPORT FOR MATHEMATICS, Revision 11:2001-2008, http://unicode.org/reports/tr25/#_Toc231, Abruf 2. Jan. 2011.

[22] Hanna-Chris Gast: "**Griechische Schrift für den Computer** — Vergleich der Transkriptionen in Bibliotheken und Standesämtern, Auflistung aller griechischen Zeichen mit Unicode sowie Tastenbelegung für Windows", 2. erweiterte Auflage, Selbstverlag, Berlin 2011, Im Internet: http://www.siebener-kurier.de/chris-aufsaetze/Griechische_Schrift_Computer.pdf.

[23] Hanna-Chris Gast: "**Schrift- und Sonderzeichen in Word**", Selbstverlag, Berlin, Stand: 2014, http://www.siebener-kurier.de/chris-aufsaetze/Word-Schriftzeichen.pdf.

[24] Hanna-Chris Gast: "Word-Makro zur Umcodierung von **Symbol- und Wingdingszeichen in Unicode Zeichen**"; Gedruckt als Beilage zum Siebener-Kurier Nr. 61 (August 2011); Stand 2014: http://www.siebener-kurier.de/chris-aufsaetze/Symbol+Wingdings-in-Unicode.pdf.

[25] Hanna-Chris Gast: "**Einige Makros für Word** (und deren Installation)", Stand 2014, Als Word-Datei zum Download: http://www.siebener-kurier.de/chris-aufsaetze/Word-Makros.doc.

[26] Hanna-Chris Gast: "**Formblatt mit Makros zur Übertragung von Word Tasten-kombinationen** (Tastenkürzel) und AutoKorrekturen"; Berlin 2011/2013; http://www.siebener-kurier.de/chris-aufsaetze/Word-Tastenkuerzel-uebertragen.doc.

[27] Schreibbüro **Nora Richter,** Georg-Schumann-Str. 8, 04105 Leipzig; *http://www.schreibbuero-richter.de*, und E-Mails, März 2012.

# Einige weitere lieferbare Bücher


Hanna-Chris Gast:
**"Kyrillische Schrift für den Computer –**
**Benennung der Buchstaben,**
**Vergleich der Transkriptionen in Bibliotheken und Standesämtern,**
**Auflistung der Unicodes sowie Tastaturbelegung für Windows XP",**
Shaker-Verlag Aachen, 2010;
ISBN: 978-3-8322-9656-8.
http://www.Siebener-Kurier/chris-aufsaetze/Kyrillische_Schrift_Computer.pdf

Mit dieser Arbeit möchte ich dem Computer-Benutzer, der mit kyrillischen Buchstaben zu tun hat, ohne eine passende russische Tastatur zu besitzen, eine Hilfe beim Schreiben kyrillischer Buchstaben geben. Ferner möchte ich auch allen denen, die im deutschsprachigen Raum kyrillische Buchstaben ins lateinische Alphabet umsetzen müssen, eine Übersicht zu geben, welche Umschrift die für sie maßgebliche ist.

Dieser Aufsatz enthält eine Übersicht über die Transkriptionen bzw. Transliterationen kyrillischer Buchstaben in deutschen, österreichischen und Schweizer Bibliotheken einerseits, und in deutschen, österreichischen und Schweizer Ämtern andererseits. Diese Transkriptionen werden tabellarisch der Internationalen Norm ISO 9 und den Regeln der amerikanischen Bibliotheken (ALA-LC) gegenübergestellt. Ich gehe in diesem Aufsatz hauptsächlich auf die Transkription aus der russischen Sprache ein.

**********


Hanna-Chris Gast:
**"Griechische Schrift für den Computer –**
**Auflistung sämtlicher griechischen Zeichen mit Unicode, Tastenbelegung für Windows,**
**Vergleich der Transkriptionen in Bibliotheken und Standesämtern"**,
2. erweiterte Auflage (mit geändertem Titel),
Selbstverlag, Berlin 2011
ISBN 978-3-00-036684-0
Im Internet unter:
http://www.siebener-kurier.de/chris-aufsaetze

Mit dieser Arbeit möchte ich dem Anwender des Textsystems Microsoft-Word, der mit griechischen Buchstaben zu tun hat, Hilfestellung beim Schreiben griechischer Buchstaben geben. Auch möchte ich all denen, die im deutschsprachigen Raum griechische Buchstaben ins lateinische Alphabet transkribieren, eine Übersicht geben, welche Umschrift die für sie maßgebliche ist.

## Impressum

© Dipl.-Ing. H. Chris Gast
Bergstr.1
14109 Berlin
www.siebener-kurier.de/chris-aufsaetze
E-Mail: hcgast "ät" siebener-kurier.de *(dabei "ät" durch "@" ersetzen).*



Gast, Hanna-Chris: "Erweiterung von Word 2000 zur Erzeugung von Schriftzeichen höherer Unicode-Ebenen
sowie Erstellung eines Makros für Word 2000 bis Word 2010 zur **Analyse von Schrift- und Sonderzeichen** mit Ausgabe des Schriftzeichen-Namens nach ISO/IEC 10646";
Gedruckt als Beilage zum Siebener-Kurier Nr. 60 (August 2010), ISSN 0948-6089;
Erweiterte Auflage als Sonderdruck des Siebener-Kuriers, Juni 2011.

Im Internet als Download unter:
http://www.siebener-kurier.de/chris-aufsaetze/Word-Erweiterung-Unicode-Makros.pdf.

Das **Analysemakro** aus diesem Aufsatz ist in einer fertigen **Dokumentvorlage** downloadbar unter: http://www.siebener-kurier.de/chris-aufsaetze/Zeichen-Analyse-kompakt.zip.
Darin habe ich diesem Makro das Tastenkürzel "AltGr + C" zugeordnet.

Stand: Mittwoch, 14. Januar 2015.