

DMG-Transliteration in Unicode

Eine praxisorientierte Einführung
für die Anwendung in MS Windows.

ā 1E01									
Ḃ 1E02	Ḍ 1E12	Ḏ 1E22							
ḃ 1E03	ḏ 1E13	ḥ 1E23	ḵ 1E33	ṁ 1E43	ṁ 1E53				
Ḅ 1E04	Ḙ 1E14	Ḩ 1E24	Ḵ 1E34	Ṁ 1E44	Ṁ 1E54	Ṃ 1E64			
ḅ 1E05	ḙ 1E15	ḩ 1E25	ḽ 1E35	ṁ 1E45	ṁ 1E55	ṃ 1E74			
Ḇ 1E06	Ḛ 1E16	Ḫ 1E26	Ḷ 1E36	Ṁ 1E46	Ṁ 1E56				
ḇ 1E07	ḛ 1E17	ḫ 1E27							
Ḉ 1E08									

www.michaelkreutz.net

Im Auftrag des Seminars für Orientalistik & Islamwissenschaften
der Ruhr-Universität Bochum, GB 2/ 34

6. Auflage, Webedition 1, Jan. 2006

Inhaltsverzeichnis:

0	Allgemeines.	3
1	Was ist Unicode?	3
2	Welche Sonderzeichen werden benötigt?	5
3	Unicode-Fonts.	6
4.	Unicode und Windows.	7
4.1	<i>Multikey.</i>	8
4.2	<i>Der Uniqoder.</i>	16

Unicode für Orientalisten – Wiedergabe der DMG-Transliteration in Windows und OS X.

von Michael K r e u t z , Bochum

0. Allgemeines.

Die eigentliche Schwierigkeit liegt darin, die Tastatur so zu erweitern, dass Transliterationszeichen durch simplen Tastendruck in das Dokument zu bringen sind, ohne das Menü heranziehen zu müssen. Programme, die die vom System zur Verfügung gestellte Tastaturbelegung manipulieren, helfen nicht weiter, da neue Belegungen auf der Alt- und AltGr-Ebene der Tastatur bei einem Neustart vom System meistens zurückgewiesen werden. Es stellt sich ausserdem die Frage, welcher Unicode-Zeichensatz über die entsprechenden Transliterationszeichen oder Diakritika verfügt (s.u., Punkt 3, S. 5).

1. Was ist Unicode?

Unicode ist der Versuch, einen einheitlichen Codierungsstandard für Schriftzeichen aller Art und sonstige gebräuchlichen Zeichen und Symbolen systemunabhängig, programmunabhängig und sprachenunabhängig zu schaffen. Bisläng waren einzelne Kodierungssysteme eng begrenzt als auch untereinander verschieden. In der Praxis heisst dies, dass ein Text, der andere Zeichen als die üblichen lateinischen Buchstaben (ASCII-Zeichensatz) aufweist, z.B. beim Verschicken über das Internet verstümmelt erscheint. Unicode dagegen gibt jedem Zeichen seine eigene Nummer und ermöglicht

es, unabhängig von Betriebssystem, Anwendung oder Sprache die unterschiedlichsten Zeichen für den Datenverkehr verwendbar zu machen.

Das alte ASCII (American Standard Code for Information Interchange) ist ein 7-Bit-Code, umfasst also mit 128 Zeichenpositionen noch nicht einmal die Umlaute; mit 8 Bit (= 256 Zeichen) bot ISO 8896 eine deutlich erhöhte Zahl an Codezuweisungen, die aber noch nicht einmal für den in Europa gebräuchlichen Zeichenvorrat ausreichten. Zeichen mit Diakritika mussten ausserdem per Tottaste kombiniert werden, sodass das gewünschte Ergebnis immer nur in Abhängigkeit des jeweiligen Zeichensatzes hergestellt werden konnte. Unicode räumt mit diesem Wirrwarr auf, indem es jedem irgendwie gebräuchlichen Zeichen eine Position zuordnet und dabei auch Gross-/ Kleinschreibung und kombinierte Zeichen (Grundzeichen + Diakritikum) berücksichtigt.

Unicode kennt verschiedene Standards. UCS-2 (Universal Coded Character Set auf 4-Byte-Grundlage) ist ein 16-Bit-Code und UCS-4 ein 32-Bit-Code mit einer Grösse von über einer Million Zeichen. In der aktuellen Version 4.0. sind 96.382 davon für die verschiedensten Zeichen reserviert. Jedes einzelne Zeichen wird einer Position zugewiesen, die sich aus den Werten der x- und der y-Achse im Hexadezimalsystem zusammensetzt, wobei die y-Achse fortlaufend durchnumeriert, die x-Achse auf 16 Positionen begrenzt ist (immer mit dem ersten Bit = 0).

Die Positionen sind in Gruppen von je 256 (0-FF) unterteilt. UTF-16 (UCS-4 Transformation Format 16 bit form) ist mit UCS-2 identisch, besitzt auf den Positionen D800 bis DFFF allerdings andere Zuweisungen. UTF-8 schliesslich bezeichnet eine Codierung, die einer UCS-4-Zeichenposition bis zu sechs Bytes zuweist, und somit eine komprimierte Form von UTF-16 darstellt. Die mit ASCII identischen Positionen 0-7F (= 0-128) werden in einem einzigen Byte untergebracht, die Positionen 128 bis 1023 in zwei, die oberhalb von 1023 in drei Bytes. Die Konvertierung von UTF-16 nach UTF-8 und umgekehrt ist ohne Datenverlust oder -Veränderung möglich, da beide dieselbe Codierung benutzen. Der aktuelle Unicode-Standard, Version 4.0 umfasst Codes (Positionen) für 95.221 Zeichen aus den Alphabeten dieser Welt, sowie die verschiedensten Sonderzeichen und Symbole.

Die Anordnung der Bytes kann auf zweierlei Art und Weise geschehen: little endian bezeichnet die Anordnung der Bytes durch den Prozessor in Reihenfolge des abnehmenden Wertes, big endian die umgekehrte Anordnung. Manche Programme wie

z.B. UniPad erlauben eine Einstellung des Bytes-Modus. Word wiederum arbeitet mit little endian, HTML-Dokumente sollen mit big endian besser zurechtkommen.

2. Welche Sonderzeichen werden benötigt?

Die Transliteration der DMG verteilt sich nach dem Unicode-Standard [1] auf vier Bereiche:

Latin Extended-A

Ā	0100	<i>Latin capital letter A with macron</i>	ا
ā	0101	<i>Latin small letter A with macron</i>	ا
Ġ	0120	<i>Latin capital letter G with dot above</i>	غ
ġ	0121	<i>Latin small letter G with dot above</i>	غ
Ī	012a	<i>Latin capital letter I with macron</i>	ي
ī	012b	<i>Latin small letter I with macron</i>	ي
Š	0160	<i>Latin capital letter S with caron</i>	ش
š	0161	<i>Latin small letter S with caron</i>	ش
Ū	016a	<i>Latin capital letter U with caron</i>	و
ū	016b	<i>Latin small letter U with caron</i>	و

Latin Extended-B

Ķ	01e6	<i>Latin capital letter G with caron</i>	ج
ķ	01e7	<i>Latin small letter G with caron</i>	ج

Spacing Modifier Letters

’	02be	<i>modifier letter right half ring</i>	ء
‘	02bf	<i>modifier letter left half ring</i>	ع

Latin Extended Additional

Đ	1e0c	<i>Latin capital letter D with dot below</i>	ض
đ	1e0d	<i>Latin small letter D with dot below</i>	ض
Ḑ	1e0e	<i>Latin capital letter D with line below</i>	ذ
ḑ	1e0f	<i>Latin small letter D with line below</i>	ذ

Ḥ	1e24	<i>Latin capital letter H with dot below</i>	ح
ḥ	1e25	<i>Latin small letter H with dot below</i>	ح
H̆	1e2a	<i>Latin capital letter H with breve below</i>	خ
h̆	1e2b	<i>Latin small letter H with breve below</i>	خ
Ș	1e62	<i>Latin capital letter S with breve below</i>	ص
ș	1e63	<i>Latin small letter S with breve below</i>	ص
Ṭ	1e6c	<i>Latin capital letter T with dot below</i>	ط
ṭ	1e6d	<i>Latin small letter T with dot below</i>	ط
T̅	1e6e	<i>Latin capital letter T with line below</i>	ث
t̅	1e6f	<i>Latin small letter T with line below</i>	ث
Ẓ	1e92	<i>Latin capital letter Z with dot below</i>	ظ
ẓ	1e93	<i>Latin small letter Z with dot below</i>	ظ

3. Unicode-Fonts.

Bislang gibt es für sämtliche Systeme nur vier Fonts, die den vollständigen Transliterationssatz enthalten: Code2000, Arial Unicode MS, TITUS Cyberbit Basic [2] und Gentium [3]. Ersterer kommt für unsere Zwecke deshalb nicht in Frage, da er auf einer Zierschrift basiert; der Arial-Font dagegen ist nur vollständig als Teil von Windows 2000/XP. Einzig TITUS (basiert auf Times New Roman) und Gentium sind allgemein zugänglich und optisch zufriedenstellend. TITUS ist im Ausdruck übrigens besser als auf dem Bildschirm.

[1] www.unicode.org/charts

[2] <http://titus.fkidg1.uni-frankfurt.de/unicode/tituut.asp>

[3] http://scripts.sil.org/cms/scripts/page.php?site_id=nrsi&item_id=Gentium

Beide Zeichensätze enthalten auch griechische Zeichen, TITUS ausserdem hebräische, arabische und kyrillische. Ich empfehle dennoch Gentium, weil sein Erscheinungsbild klarer ist und auch über kursive Zeichen verfügt. Auch dieser Text ist hauptsächlich in Gentium geschrieben.

4. Unicode und Windows.



Angesichts der herrschenden Monokultur auf Windows-Rechnern soll auch hier die Verfügbarkeit von MS Word vorausgesetzt werden. Die Anforderungen an ein Programm zur Eingabe von Transliterationszeichen lassen sich wie folgt umreißen:

Es sollte eine Erweiterung für Word sein und kein separates Tool.

Die Erweiterung der über die Tastatur generierbaren Zeichen kann nur mit Hilfe von Makrobefehlen geschehen, und nicht über eine Änderung der Tastaturbelegung. Der einzelne Makrobefehl muss zusammengesetzt sein aus Taste + Grundzeichen und Taste + Diakritikum. Hotkey-Befehle (shortcuts) sollten nicht benötigt werden. Die Makros müssen möglichst für die eigenen Zwecke konfigurierbar sein.

Die Möglichkeit zur benutzerangepassten Makrokonfiguration sollte es erlauben, Schriftsystem und Zeichensatz miteinander zu verknüpfen.

Systemunterstützung erfährt Unicode seit den Systemen NT und 2000. Dennoch müssen die einzelnen Plattformen unterschiedlich behandelt werden, einschliesslich älterer Systeme. Grundsätzlich kommen dabei nur zwei Anwendungen in Frage; wer mit Windows 95/98/XP arbeitet, liest bitte hier weiter, wer mit Windows 2000 oder Millennium arbeitet, unten bei 4.1.2 (Seite 16):

4.1 *MultiKey*.

Die optimale Lösung heisst MultiKey [1], und wie der TITUS-Font auch, ist das Programm kostenlos. Die Tastaturumschaltung zwischen den Zeichensystemen erfolgt mit den F-Tasten, wobei F10 = lateinische, F11 = griechische und alt-F12 = arabische Tastatur bedeutet. Wer mit Windows XP arbeitet, wird das Programm nur für das

Schreiben mit lateinischen Buchstaben samt Transliteration benötigen; Windows 2000 oder Millennium hingegen verweigern generell die Zusammenarbeit wegen ihrer Unicode-Präkonfiguration. Für Windows95/98-Benutzer wiederum eröffnet MultiKey die Möglichkeit, auch arabische und griechische (polytonisch) Zeichen zu Papier zu bringen! Von welcher Version das Officepaket (Word) ist, spielt übrigens keine Rolle.

[1] www.oeaw.ac.at/kal/multikey

Die Benutzung von Makros bedeutet nicht nur, den über die Tastatur erreichbaren Zeichenumfang nach Belieben zu gestalten; sie bedeutet auch, dass für die Darstellung der DMG-Diakritika der Benutzer mit insgesamt drei Tasten auskommt!

Wie das geht, obwohl die Anzahl der möglichen Diakritika mehr als drei beträgt? Ganz einfach: Man beachte, dass die für Diakritika reservierten Tastaturpositionen keine Tottastenfunktion (dead key) haben. So kann die Kombination g und < als Makro auslösende Befehlskette mit dem Ergebnis ě definiert werden, und s und + als eine solche mit dem Ergebnis š. Das Pluszeichen ist also keine Tottaste, sondern nur das zweite Element einer Befehlskette, die ein Makro aktiviert. Die Gesamtheit aller zweiten Elemente lässt sich anhand empirischer Grundlage auf drei Tastaturpositionen unterbringen: Die Raute (#) steht so – nach unserer Definition – für jeden diakritischen Balken, das Plus (+) für jeden diakritischen Punkt, und die nach rechts geöffnete spitze Klammer (<) für das Háček. Einfacher geht es nicht.

Nach Download und Installation muss die Definitionsdatei (.ini-Datei) geöffnet werden, die sich ausser bei Windows XP im temporären Ordner befindet, also im Pfad `c:\temp\`, bzw. bei XP unter `C:\Program Files\Microsoft Office\Office10\STARTUP\MltKey97.dot`.

Als nächstes wird die Datei im Notepad/Wordpad angepasst. Dies erfolgt in mehreren Schritten:

a) Am Kopf muss unter [Unicode Fonts] der bevorzugte Zeichensatz eingegeben werden, also z.B. Gentium oder TITUS. Somit steht da also:


```
[Unicode Fonts]
...
Latin=Gentium (oder TITUS Cyberbit Basic)
...
```

```
MultiKey - Notepad
File Edit Format View Help

[Unicode Fonts]
Greek=Gentium
;Greek=TITUS Cyberbit Basic
Latin=Gentium
Cyrillic=TITUS Cyberbit Basic
Arabic=Times New Roman
Hebrew=Gentium

[Alias]
Latin Unicode=Gentium
Greek Unicode=Greek Unicode
Cyrillic Unicode=Cyrillic Unicode
Hebrew Unicode=Hebrew Unicode
Hebrew Unicode RTL=Hebrew Unicode RTL
Arabic Unicode=Arabic Unicode
Arabic Unicode RTL=Arabic Unicode RTL
Times New Roman=Latin Unicode
Garamond=Latin Unicode
Arial=Latin Unicode
Lucida Sans Unicode=Latin Unicode
GoEast Code=EETimes New Roman
EETimes New Roman=EETimes New Roman
WL CyrillicTimes=WL CyrillicTimes
Cyril Times=Cyril Times
WP Japanese=WP Japanese
Payne=Payne
Aisa Unicode=Greek Unicode
Aisa=Aisa
Greek=Greek
```

**Einfügen zweier neuer Zeilen.
Die Eintragungen für "Greek" und "Arabic" sind optional.**

 Worauf es ankommt, ist natürlich der Eintrag unter Latin, da wir die Transliterationszeichen auf Grundlage des lateinischen Alphabetes mit deutscher Tastaturbelegung ins Dokument setzen.
Wer Arabisch schreiben will und dies mit MultiKey tun möchte, muss berücksichtigen, dass Gentium keine arabischen Lettern enthält.

b) Danach muss der Eintrag unter [Alias] geändert werden:

Latin Unicode=Gentium

c) Anschliessend werden unter [Latin] die Strings eingetragen, die für den lateinischen Zeichensatz wirksam werden sollen. Dazu wird am Ende des Zeichenblocks eine neue Zeile eröffnet, beginnend mit:

Gentium=

Es folgen (ohne Leertastenanschlag am Anfang) die Makro-Strings für die Arbeit mit Diakritika. Die Syntax dazu lautet wie folgt:

<Zeichensatz>=<String 1> <String 2>...

Die Syntax für die einzelnen Strings ist in der Gebrauchsanweisung von MultiKey falsch angegeben. Korrekterweise muss sie lauten:


<Zeichen 1><Zeichen 2>'<Unicode-Position des resultierenden Zeichens>

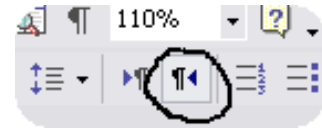
Wenn also a, gefolgt von einem - zu dem Ergebnis ā führen soll, dann lautet der entsprechende String: a-´0101 . Wie man sieht, sind Input und Output durch Apostroph getrennt. Und nur der Output wird mit der Unicode-Position angegeben.

d) Unter [Latin] muss also eingetragen werden (mit Leerstelle zwischen den einzelnen Strings):

```
Gentium=a#´0101 i#´012b u#´016b cC´02bf xX´02be t#´1e6f
g<´01e7 h+´1e25 d#´1e0f s<´0161 s+´1e63 d+´1e0d t+´1e6d
z+´1e93 g+´0121 h#´1e2b A#´0100 I#´012a U#´016a T#´1e6e
G<´01e6 H+´1e24 D#´1e0e S<´0160 S+´1e62 D+´1e0c T+´1e6c
Z+´1e92 G+´0120 H#´1e2a ->´2192 **´2022
```


Alt-F12 Arabic (dto.)

 Wer mit WindowsXP arbeitet, für den ist grundsätzlich nur F10 Latin interessant, da das System von sich aus die Möglichkeit bietet, mit nichtlateinischen Schriften zu arbeiten. Die Schreibrichtung muss dazu gegebenenfalls auf Rechts-nach-links umgestellt werden. Die Unterstützung für Griechisch unter Windows XP ist jedoch recht armselig, trotz Unicode-Kompatibilität. Auch hier empfiehlt sich also MultiKey.



Jetzt also Word starten (Makros aktivieren!) und dann F10 drücken!

Und nun frisch ans Werk. Die von uns definierten Strings schlüsseln sich wie folgt auf:

F10 → Latin Unicode → TITUS Cyberbit Basic :

1.	2.	Ergebnis	Unicodepos.
a	#	ā	0101
A	#	Ā	0100
i	#	ī	012b
I	#	Ī	012a
u	#	ū	016b
U	#	Ū	016a
t	#	ṭ	1e6f
T	#	Ṭ	1e6e
d	#	ḏ	1e0f
D	#	Ḑ	1e0e
h	#	ḥ	1e2b
H	#	Ḥ	1e2a


g	<	ġ	01e7
G	<	Ġ	01e6
s	<	š	0161
S	<	Š	0160
h	+	ħ	1e25
H	+	Ĥ	1e24
d	+	ḏ	1e0d
D	+	Ḑ	1e0c
s	+	ṣ̌	1eb3
S	+	Ṣ̌	1e62
z	+	ẓ	1e93
Z	+	Ẓ	1e92
t	+	ṭ	1e6d
T	+	Ṭ	1e6c
g	+	ġ̣	0121
G	+	Ġ̣	0120
c	C	ˆ	02d3
x	X	˘	02d2
-	>	→	2192
*	*	•	2022

Nicht bewährt haben sich die Tastenkombinationen mit (.) für den diakritischen Punkt, und (-) für die Vokallängung, da ansonsten statt z.B. Arist. An. post. immer Arisṭ An. posṭ und statt bi-l-kitāb jedesmal bīl-kitāb entsteht. Zwar kann man den entsprechenden String auch ausser Kraft setzen, dazu muss man sich jedoch immer bewusst sein, welche Tastenkombination einen Makrobefehl zur Folge hat und dann bei der Unterbrechung des strings einige Fingerakrobatik beweisen: Um den Punkt hinter das t zu setzen anstatt darunter, muss erst t gedrückt werden, dann AltGr-shift-[Punkt] und anschliessend noch einmal [Punkt]. Auf Dauer ist das wenig praktikabel. Ich habe mich daher für folgende Lösung entschieden: Jeder diakritische Balken, egal in welcher Form und auf welcher Höhe befindlich, wird mit (#) gebildet (unter WindowsXP ist der Backslash anstelle dessen unmöglich, dies geht nur mit Windows98), jeder diakritische Punkt mit (+), und jegliches Háček mit (<).

Natürlich ist dieses System grundsätzlich erweiterbar. Dazu muss nur die ini.-Datei wie oben erklärt erweitert werden. Die entsprechenden Unicode-Positionen können über das Internet erfragt werden.

Wer mit Windows98 arbeitet, sollte auch wissen, wie die Diakritika im arabischen Tastaturmodus (alt-F12) generiert werden. Benutzer von WindowsXP sind zwar grundsätzlich nicht darauf angewiesen, sollten sich aber darüber im klaren sein, dass die Makros von MultiKey zuweilen mit denen des Systems kollidieren. Es empfiehlt sich daher u.U. auch hier, mit MultiKey Arabisch zu schreiben. Die Tasten für die arabischen Vokalzeichen werden nach dem jeweiligen Buchstaben gedrückt:

e	→	fatḥa
u	→	ḍamma
i	→	kasra
=	→	hamza (zw. Alif und neuem Buchstaben automatisch über das Alif gesetzt)
H	→	tā' marbūṭa
~	→	madda
(nicht vorh.)		tašdīd
(nicht vorh.)		Alif waṣl

 Auch wenn Word bereits geschlossen ist, bleibt das Makro aktiv (Microsoft hat dieses Feature in seine Systemsoftware eingebaut, damit Makros im Ernstfalle grösstmöglichen Schaden anrichten). VOR dem Gebrauch anderer Anwendungen - dies gilt v.a. für den Browser – sollte daher noch einmal die F10-Taste gedrückt oder das Makro (sichtbar in der Taskleiste) über das Kontextmenü beendet werden, um ein mögliches Inkompatibilitätsproblem zu vermeiden. Es kann sonst z.B. sein, dass das Adressfeld des Browsers verrückt spielt und die Eingabe jeglicher URL unmöglich macht.

! Unter Windows98 verhalten sich die Makros in Word recht eigenwillig. Dies scheint auf eine Interferenz mit anderen Anwendungen zurückzugehen. Welche das sind, bleibt das grosse Geheimnis. Das Problem ist jedenfalls schnell behoben, indem man man mehrmals die Strg-Taste drückt.

! Zuweilen passiert es unter XP, dass beim Starten von Word die Startup-Datei von MultiKey nicht mitgeladen wird (erkennbar am Fehlen des MultiKey-Symbols in der Toolbar). Die Ursache ist unbekannt. Es empfiehlt sich für alle Fälle, auf dem Desktop einen Shortcut dieser Datei zu placieren, um für den genannten Fall MultiKey manuell starten zu können.



! MultiKey funktioniert nur als Funktionserweiterung zu Word. Wer auf die Idee kommen sollte, mit Transliteration auf Unicode-Basis z.B. in Access zu arbeiten, der muss zu anderen Mitteln greifen.

Das im folgenden unter 4.1.2 vorgestellte Programm läuft auch unter 95/98/XP, steht aber an Komfort hinter MultiKey zurück. Für den äusserst seltenen Fall einer Inkompatibilität von MultiKey und WindowsXP sollte es immer noch als Lösung in Frage kommen:

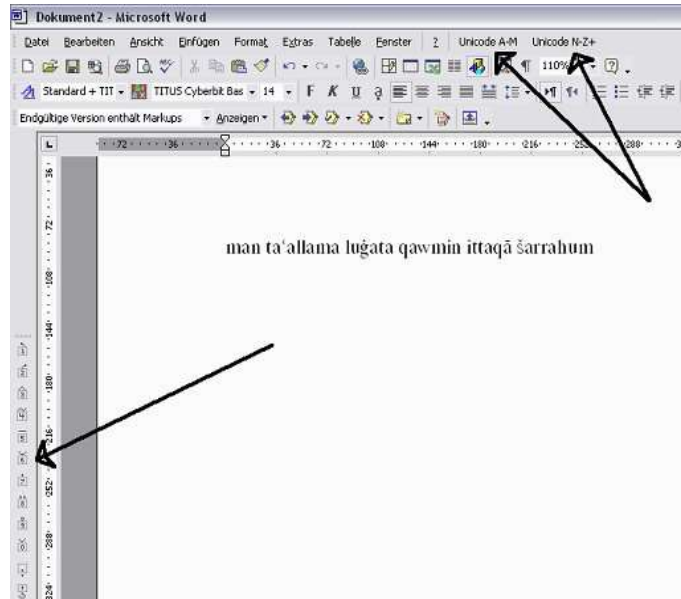
4.2 Der Uniqoder.

Für die Benutzer von Windows 2000 und Millenium kommt für unsere Zwecke einzig der Uniqoder [1] in Frage. Er funktioniert nach demselben Prinzip wie MultiKey, d.h. als Makroanwendung im Startup-Folder von Office, mit dem Unterschied, dass er nicht frei konfigurierbar ist. Die Diakritika werden aus einer separaten Toolbar, bzw. über ein

spezielles Menü angewählt. Auch hierbei ist der Gentium- oder TITUS-, bzw. Arial Unicode MS-Font unabdingbar. Die Adresse ist leider nicht mehr aktiv, was hoffentlich nur ein vorübergehender Zustand ist:

[1] <http://hem.fyristory.com/dahloe/uniqoder>

Die wichtigsten Diakritika sind links in der Toolbar anwählbar; a und anschließender Klick auf das Makron ergeben ā; die Zeichen für ε und ε [°] müssen über das Zusatzmenü Unicode A-M abgerufen werden. Leider kann die Toolbar den individuellen Bedürfnissen nicht angepasst werden. Daher betrachte ich dieses ansonsten hervorragende Programm nur als zweite Wahl gegenüber MultiKey.



Wichtig ist, dass immer der TITUS/Gentium-Font aktiviert sein muss, um alle Kombinationen mit Diakritika zu realisieren, da nur diejenigen Zeichenkombinationen erstellt werden können, die im Zeichenvorrat des Fonts vorhanden sind.

© 2006 by Michael Kreutz, 44879 Bochum

www.michaelkreutz.net

Seminar für Orientalistik u. Islamwissenschaften

Ruhr-Universität Bochum, GB 2/34

Universitätsstr. 150

D-44780 Bochum

Telefon:+ 49-(0)234-32 28126

Telefax:+ 49-(0)234-32 14671