»Frosch oder Prinz. TUSTEP als Werkzeug der digitalen Geisteswissenschaften – ein Workshop der International TUSTEP User Group«

2. DHd-Jahrestagung
23. Februar 2015
13.30 Uhr–17.00 Uhr,
Karl-Franzens-Universität Graz

Skript /Cheat Sheet

Thomas Kollatz ⟨kol@steinheim-institut.org⟩
Ute Recker-Hamm ⟨recker@uni-trier.de⟩
Matthias Schneider ⟨schneiderm@uni-trier.de⟩

www.itug.de
www.steinheim-institut.org
www.admainz.de
www.kompetenzzentrum.uni-trier.de

itug international tustep user group

SALOMON LUDWIG STEINHEIM INSTITUT FÜR DEUTSCH-JÜDISCHE GESCHICHTE

ACADEMIA SCIENTIARVM ET LITERARVM MOGVNTINA GENIO LEIBNITII

Kompetenzzentrum für elektronische Erschließungs- und Publikationsverfahren in den Geisteswissenschaften

Gliederung des Workshops

1. Einleitung
2. Dateien vorbereiten: *import, #umwandle, ACCESS
3. Vergleichen und Kollationieren
4. TUSCRIPT – Die Skriptsprache des TUSTEP-Programmpakets
5. Texte setzen mit TUSTEP
6. Texte exportieren: *export

1. Einleitung

TUSTEP installieren oder vom USB-Stick nutzen

- Download: http://www.tustep.uni-tuebingen.de/download.php
- zur Erläuterung der Installation und Konfiguration (insb. unter Linux und OS_X) steht eine ausführliche Anleitung zur Verfügung: http://www.tustep.uni-tuebingen.de/inst.html
- mit dem Standardmakro `#*TUSTEP2STICK` kann TUSTEP auf einem USB-Stick installiert werden
- Für erste Schritte im Programm s. TUSTEP-Wiki[1] sowie die Einführungsdokumente Hein/Schneider (2014), Schneider/Tobner (2014) sowie Tobias Ott (2013).[2]

Grundlagen der Benutzung

TUSTEP arbeitet syntax-basiert und verfügt über eine reduzierte graphische Oberfläche (graphical user interface, GUI). Die Funktionen des Programmpakets werden über die Eingabe von Befehlen (Kommandos und Anweisungen) gesteuert.[3]

Bei der Benutzung von TUSTEP können ähnlich wie bei den Workspaces von Eclipse verschiedene ›Projekte‹ inklusive eines eigenen Verzeichnisses angelegt und unterschieden werden.

Sinnvollerweise wird jedes Arbeitsvorhaben aus Gründen der Übersichtlichkeit und einfacheren Verwaltung in einem eigenen Projekt verwaltet.

Zur Verwaltung der verschiedenen Projekte (anlegen, ändern, löschen…) steht das Standardmakro `#*DESI` (=DEfiniere SItzung) zur Verfügung.[4]

Für grundlegende Kommandos und Anweisungen s. »TUSTEP-Grundlagen« von Wolfram Schneider-Lastin (Universität Zürich) im TUSTEP-Wiki.

Die Anbindung von anderen Programmen/Programmiersprachen ist möglich (z. B. Java- oder Tcl/Tk-GUI zur Abfrage von Benutzereingaben, anschließung TUSTEP-Verarbeitung und Aufruf des Ergebnisses mit dem Browser, PDF-Viewer, Word….

Dateimanager[5]

- Beispieldatei: `dhd_wo.tf`

TUSTEP stelllt einen eigenen Dateimanager zur Verfügung, mit dem grundlegende Aufgaben zur Dateiverwaltung erledigt werden können. Hierzu zählen das Auswählen und Erstellen von Projekten und Dateien, das Anmelden, Anlegen, die Auswahl von Daten aufgrund verschiedener Parameter, Löschen oder Edieren von Dateien, der Aufruf von Makros (=Kommandozusammenstellungen) sowie das Kopieren, Verschieben, Umbenennen, Vergleichen und Umwandeln von Dateien.

---

[1] http://tustep.wikispaces.com/TUSTEP+–+Erste+Schritte.
[2] http://tustep.wikispaces.com/Einf%C3%BChrungen.
[3] TUSTEP unterscheidet zwischen Kommandos auf Kommandoebene und Anweisungen im Editor.
[4] http://tustep.wikispaces.com/Grundlagen+DESI.
[5] Vgl. Videotutorial http://www.tustep.wikispaces.com/Datei-Manager

Unter »Umwandeln« wird in TUSTEP die Konvertierung von »Fremddateien« nach TUSTEP oder von TUSTEP in eine Fremddatei verstanden (z. B. TXT → `#UMWANDLE` → TUSTEP → `#UMWANDLE` → XML).[6]
Zusätzlich können unter Windows sowie ab Version 2015 auch unter Linux verschiedene Nicht-TUSTEP-Formate mit den dafür eingestellten Standardprogrammen geöffnet werden (z. B. DOC, PDF, HTML).

- Aufruf des Dateimanagers via Kommando `#*D, #*E` oder Funktionstaste `F6`
- TUSTEP verfügt über ein eigenes Dateiformat, daher sind die o. g. Konvertierungen in der Regel notwendig. Das TUSTEP-eigene Format enthält bespielsweise Datensatznummern, die für verschiedene Zwecke wie z. B. die Registererstellung oder das teilweise Verarbeiten von Dateien oder Programmen genutzt werden können. Außerdem verfügt TUSTEP über eigene Konventionen zur Codierung von Diakritika und »Sonderzeichen«, die es ermöglicht, auch solche Sonderzeichen (und Kombinationen) abzubilden, die im verwendeten Font nicht enthalten sind.
- Mit TUSCRIPT und TXSTEP können auch externe Dateiformate (z. B. TXT, XML oder HTML) direkt, d. h. ohne vorherige Umwandlung ins TUSTEP-interne Format, verarbeitet werden.
- TUSTEP setzt auf ein restriktives Rechtemanagement, demzufolge nur solche Dateien aufgerufen, verarbeitet, gelesen etc. werden dürfen, für die zuvor die entsprechenden Rechte festgelegt wurden. Dateien können zum Lesen oder zum Schreiben angemeldet werden. Zum Schreiben angemeldete Dateien sind für die Änderung durch andere Prozesse und Benutzer gesperrt, um die Korruption von Dateien zu verhindern.

```
Dateimanager:
Lesen = Erlangen von Leserechten
Schreiben = Erlangen von Schreibrechten

Kommandoebene:
#AN,LE=datei.tf// #AN,datei.tf (=ANMELDEN mit Leserechten)
#AN,SCH=datei.tf// #AN,,datei.tf (=ANMELDEN mit Schreibrechten)
```

- vorteilhaft insbesondere bei der kollaborativen Arbeit mit Daten, z. B. auf Servern oder Gruppenlaufwerken

- Aufruf einer Datei im Editor aus dem Dateimanager: »Edieren«

```
dhd_wo.tf auswählen
Edieren auswählen
```

[6] Für die Konvertierung nach TUSTEP und zurück von RTF-Dateien stehen gesonderte Standardmakros zur Verfügung (`#*IMPORT, #*EXPORT`, s. u.).

Editor

- Gliederung des Editors in Datensatznummerierung, Textfeld, Anweisungszeile, Statuszeile, 1–2 optionale Zeilen mit Makroleisten zur Benutzerinteraktion; hier können z. B. häufig benutzte Anweisungen auf Buttons abgelegt werden
- Einstellung des Editors (Schriftart und -größe, Fenstergröße…) s. Installationsanleitung[7]

Unterstützung bei der Bearbeitung von Texten oder Programmcode

- Einfügen von Textbausteinen via Buttons[8]
- Einfügen von Markup via Mausaktionen (s. temporäre Mausleisten)
- Einzelzeichen verschieben `ALT+→, ALT+←`
- Datensätze verschieben `ALT+↑, ALT+↓`
- Zeile verdoppeln und einfügen `CTRL+B, CTRL+B, Page_Dn`
- Eingabe von {…} `CTRL+K+erstes Zeichen`, z. B. `CTRL+K+&` → {&}
- Lesezeichen /Bookmarks setzen `ALT+B`
- Lesezeichen aufrufen `ALT+I`
- Colorierungsanweisungen (≙ Syntax Highlighting)[9]
- Speichern: TUSTEP speichert automatisch Veränderungen an einer geöffneten Datei bei folgenden Aktionen: Verschieben des Bildschirms mit einer Zeigeanweisung (z. B. mit `ZU,*` oder `F5`) oder dem Mausrad /Cursor & `SHIFT+RETURN` in der Standardeinstellung des Editors.

Pattern Matching (≙ Regular Expressions)

- TUSTEP-eigene Syntax ähnlich der Regulären Ausdrücke
- Ziel: sprachliche Muster mittels formaler Beschreibung für die automatische Verarbeitung (Suchen, Analyse, Transformation, Extraktion) erschließen
- Nutzen: ermöglicht komplizierte Suchanfragen und regelbasierte Abfragen statt einfache Stringsuche, Syntax kann im gesamten Programmpaket genutzt werden (Suchen im Editor, Austauscheanweisungen in einem Makro, Manipulation von Kollationierungsergebnissen…)
- ermöglicht die Erstellung eigener Character Groups und String Groups, um Such- bzw. Austauscheanweisungen zu steuern

```
Character Group für Vokale im Deutschen
C:vo=aeiouyäöü
```

Suche nach allen Wörtern mit einem Suffix am Wortende

```
1. Definition der Suffixe
S:su=|heit|heiten|keit|keiten|nis|nisse|nissen|ung|ungen
```

---

[7] http://www.tustep.uni-tuebingen.de/inst.html.
[8] http://tustep.wikispaces.com/Editormakros.
[9] http://tustep.wikispaces.com/Editor+–+Colorierung, permanente Einstellung mithilfe einer Konfigurationsdatei: http://tustep.wikispaces.com/Editor+–+Konfiguration.

```
2. Definition für das Wortende, z. B. Nichtbuchstabe, = beliebiges Zeichen
 außer Buchstabe
C:kb=?{-}{&a}{+}
Suche nach Wörtern (=beliebig viele Buchstaben), die ein Suffix enthalten
und anschließend keinen weiteren Buchstaben
ZN,,,|{00}{&a}{S:su}{C:kb}|
```

– Aufruf der Kurzhilfe zum Pattern Matching: `STRG+K+Leertaste/CTRL+K+BLANK`
– zur Beschreibung im Handbuch s. »Parameter-Modi seit TUSTEP Version 2012«

z. B. Anweisung »Zeige nur diejenigen Datensätzen, in denen der String ›Frosch‹ steht.«

```
ZN,,,|Frosch|
```

»Zeige alle Datensätze in ihrer Umgebung an, die den String ›Frosch‹ enthalten.«

```
ZU,,,|Frosch|
```

»Zeige nur diejenigen Datensätze an, in denen der String ›Frosch‹ als eigenständiges Wort vorkommt.«

```
ZN,,__,|Frosch|
```

»Zeige nur diejenigen Datensätze an, in denen der String ›Frosch‹ – unabhängig von möglichen Diakritika – vorkommt.«

```
ZN,,%%,|Frosch|
```

– Bei Suchzeichenfolgen unterscheidet TUSTEP grundsätzlich nicht(!) zwischen Groß- und Kleinschreibung. Soll die Schreibweise unterschieden werden, so kann ein Backslash vor das betreffende Zeichen geschrieben werden, z. B.:

```
ZN,,,|\A|
```

Zur Suche jedes Versal-»A« im Text. Bei einer Ersetzungszeichenfolge werden die Zeichen in ihrer jeweiligen Schreibweise ersetzt.

Austauscheanweisung

```
Ausgangspunkt: Hier steht ein Text.
Austauscheanweisung: A,,,|text|TEXT| (oder A,,,|text|{=0+}|)
Ergebnis: Hier steht ein TEXT.
```

2. #*IMPORT von RTF-Dateien und MSWord 2003 XML-Dateien

Mit dem Standardmakro `#*import` ist es möglich, RTF- und MS-Word 2003 XML-Dateien in eine TUSTEP-Datei zu konvertieren. Die importierte Datei liegt im XML-Format vor. Der Import lässt sich über den Dateimanager bewerkstelligen.

Alternativ lassen sich Fremddateien mit dem parametergesteuerten Programm `#umwandle` in eine TUSTEP-Datei konvertieren.

Die Skriptsprache `TUSCRIPT` (Handbuch: Makros) erlaubt den Direktzugriff auf Fremddateien.

3. Vergleichen und Kollationieren

Einfacher Aufruf ohne Parameter:

```
#vergleiche,<VersionA>,<VersionB>,modus=w,protokoll=+
```

Vergleichen und Unterschiede in Datei ausgeben:

```
#=Datei für das Protokoll der Differenzen anlegen:
#datei,<diff-prot>,fragen=–

#=Vergleichen
#vergleiche,<VersionA>,<VersionB>,modus=w,
protokoll=<diff-prot>,parameter=*
drt       PS-12
kt        |Vergleichsprotokoll___&D1 @/Seite &#3|
*eof

#=Druckausgabe der Differenzen in Postskript und PDF:
#datei,<drk.ps>'<drk.pdf>,fdf-ap
#drucke,<diff-prot>,typ=PS-12,datei=drk.ps,loeschen=+
#*ps2pdf,<drk.ps>,<drk.pdf>
```

Vergleichen (dabei Groß-, Kleinschreibung beachten) und Korrekturdatei anlegen:

```
#vergleiche,<VersionA>,<VersionB>,modus=w,protokoll=+,
korrektur=<korr>,parameter=*
gku       1
umg       3 1 3
*eof
```

Korrekturen ausführen:

```
#kausfuehre,<VersionA>,<Ziel>,loeschen=+,korrektur=<korr>
```

Drei Textfassungen kollationieren (Zeilensynopse):

```
#datei,<korr>,fragen=-
#= 1. und 2. Auflage miteinander vergleichen,
#= dabei Satzzeichenunterschiede ignorieren
#vergleiche,<VersionA>,<VersionB>,modus=w,protokoll=-,
korrektur=<korr>,loeschen=+,parameter=*
xv        |.||,||;||:||!||?||
umg       3 1 3
vkz       |2._Aufl.|
sw        1
*eof

#= 1. und 3. Auflage dito miteinander vergleichen
#= aber Korrekturdatei dabei NICHT überschreiben:
#vergleiche,<VersionA>,<VersionC>,modus=w,protokoll=-,
korrektur=<korr>,loeschen=-,parameter=*
xv        |.||,||;||:||!||?||
umg       3 1 3
vkz       |3._Aufl.|
sw        2
*eof

#= Korrekturdatei sortieren:
#sortiere,<korr>,<korr>,sortierfeld=1+44,loeschen=+
#= Kollation für die Zeilensynopse aufbereiten:
#vaufbereite,<VersionA>,modus=kumuliert,loeschen=+,parameter=*,
korrektur=<korr>
sw        1 2
vkz       |2._Aufl.|3._Aufl.|
dr        1 10 66
drt       PS-12
*eof

#= Druckausgabe:
#drucke,,typ=PS-12,datei=drk.ps,loeschen=+
#*ps2pdf,drk.ps,drk.pdf
```

4. TUSCRIPT – Die Skriptsprache von TUSTEP

Zugriff auf TAGS

Modifikation von TAGS

Erstellen eines Wortindex mit der Dictionary-Funktion

Sämtliche Funktionen werden im Handbuch mit Beispielen aufgeführt.

Lösungsorientierte Beispiele für den Einsatz von `TUSCRIPT` finden sich auf den Seiten von `RosettaCode`. Hier ist der direkte Vergleich unterschiedlichster Skriptsprachen möglich: http://rosettacode.org/wiki/Category:TUSCRIPT

`TUSCRIPT` ist insbesondere für server- und webbasiertes Arbeiten geeignet. Beispiele dafür finden sich hier: http://steinheim-institut.de:50580/cgi-bin/blau

5. Texte setzen mit TUSTEP

Satz = Substantiv zu *setzen*:

> bei buchdruckern die gegossenen buchstaben aus den fächern ihres kastens herausnehmen und in silben, wörter, zeilen und seiten zusammen setzen.
>
> Deutsches Wörterbuch, Bd. 16, Sp. 658 f.

Setzen = Herstellung von Druckvorstufen mit typographischen Gestaltungsmöglichkeiten (Zeichen, Abbildungen, Weißraum), ≙ ›Formatierung‹, Layout-Herstellung

- Werksatzsystem: »Ein Werksatzsystem verfolgt nicht den Ansatz des interaktiven Arbeitens im Umbruch (wysiwyg [=what you see is what you get, MS]), sondern der regelbasierten Verarbeitung vorstrukturierter Texte. Das regelbasierte Arbeiten führt zu hohen Verarbeitungsgeschwindigkeiten. Umbruchleistungen von mehreren 100.000 Seiten pro Minute sind möglich.« (Ott 2014, S. 116)
- Verarbeitung von nicht-lateinischen Alphabeten: Griechisch, Koptisch, Hebräisch, Arabisch, Russisch, Phonetisch
- Darstellung von diversen Sonderzeichen inklusive Kombinationen, auch wenn diese nicht als eigene Glyphe im Font enthalten sind, z. B. ḋ̲
- XML first und XML last sind möglich[10]

[10] XML first = »Bezeichnung für diejenigen Publikationsworkflows, bei denen die Autorendaten zuerst nach XML überführt werden und anschließend in den Satz und die übrigen Ausgabekanäle gegeben werden.« (Ott 2014, S. 146)
XML last = »Beschreibt einen Publikationsworkflow, bei dem die XML-Daten erst nach Abschluss der Satzarbeiten aus den Satz- oder Druckdaten konvertiert werden.« (Ott 2014, S. 147)

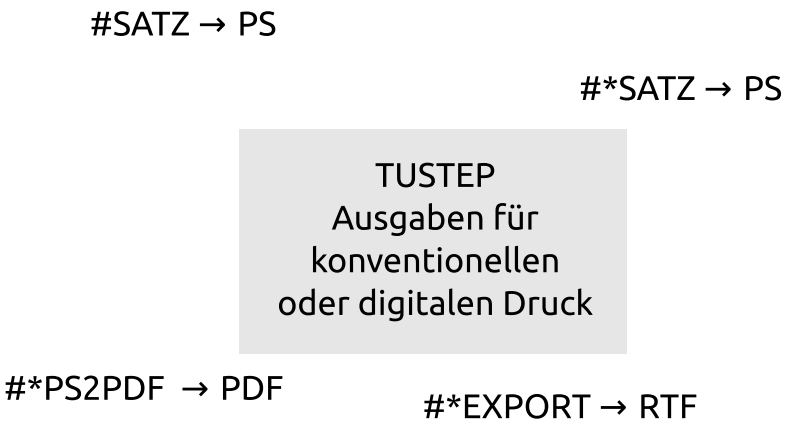

Abb. 1: Ausgabemöglichkeiten mit TUSTEP

Eigenschaft	#*SATZ	#SATZ
Codierungen	XML-Tags[11]	native Codes, XML-Tags
Anwendungen	Monographien, Sammelbände, Hausarbeiten, Rohsatz, Webservices	zstl. Editionen mit bis zu neun Apparaten, Marginalia, zeilensynoptische Ausgaben
Bedienung	Kommandozeile /Skript oder GUI	Kommandozeile /Skript
Lernkurve	sehr flach[12]	steilere Lernkurve[13]
Textelemente	Fließtext, Inhaltsverzeichnis, Fußnoten, Register, mehrspaltiger Satz, Einschaltungen, Brüche, Akkoladen	→ & weitere Diversa, z. B. kritische Apparate
Tabellen	Empfehlung: Nutzung des Tabellensatzmakros von Christian Moser	→ s. TUSTEP-Wiki
Farben	vordefinierte Tags für Schriftfarben	native Steueranweisungen

[11] Viele der nativen TUSTEP-Satz-Steuerzeichen können auch bei #*SATZ verwendet werden. Soll das Dokument allerdings mit Hilfe des Modus Export als RTF exportiert werden, sind im Sinne der besseren Kompatibilität lediglich die in der #*SATZ-Dokumentation erlaubten Tags vorgesehen. Für Details s. Satz-Beschreibung ›Modus Export‹.

[12] Die Beschreibung des Standardmakros #*SATZ umfasst lediglich 27 Seiten (Stand 11.11.2014).

[13] Für die Benutzung des Programms #SATZ sind mehrere Verarbeitungsschritte vorzunehmen, welche dem Benutzer beim Standardmakro #*SATZ abgenommen werden.

Abbildungen	möglich bei Verwendung nativer #SATZ-Codes	→
Schriften	7 vorgegebene Fonts[14]	Einbindung eigener OTF-, TTF, Type1-Fonts möglich
Fazit	schnell zu erlernen und einfach zu bedienen, für viele Anwendungsbereiche ausreichend	für komplizierte und umfangreiche Projekte

→ Für viele Anwendungsbereiche ist die Funktionalität von #*SATZ ausreichend. In jedem Fall bietet sich das Standardmakro für den Einstieg ins Setzen mit TUSTEP an. Sobald die Grenzen des Makros erreicht sind, fällt der Umstieg auf die Benutzung des Programms #SATZ leichter.

Aufruf von #*SATZ mit einer Eingabemaske/GUI

Das Standardmakro #*SATZ bringt eine eigene GUI zum Aufruf des Programms mit. Hier können die zu setzende Textdatei, die gewünschte Schriftart sowie die übrigen obligatorischen und optionalen Parameter zur Steuerung des Satzlaufs angegeben werden.

Aufruf:

```
#*M,SATZ
```

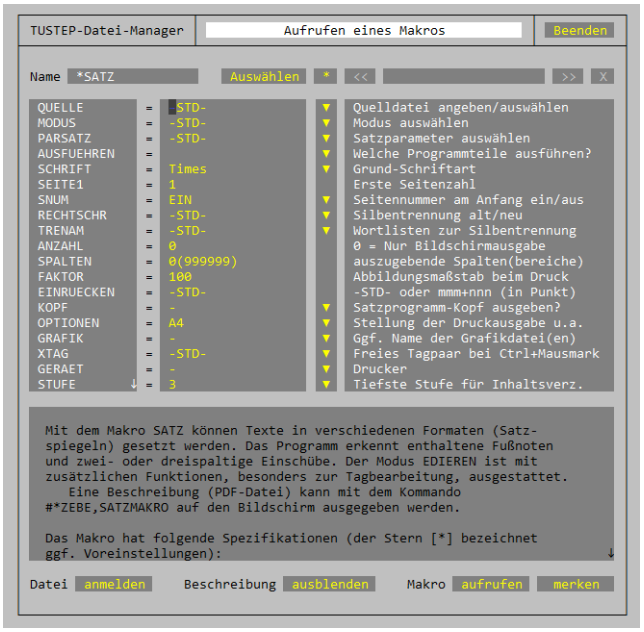

Abbildung 2: graphische Eingabemaske für #*SATZ

[14] Times, Newton, Palatino, Helvetica, Helvetica-Narrow, New Century Schoolbook, Bookman-Light.

Eingabe und Bearbeitung von Satzdaten im Editor (Modus Ediere)

Das Standardmakro `#*SATZ` bietet darüber hinaus eine eigene Editordefinition (=spezielle GUI) an, mit der die Texte für den Satz erstellt, kontrolliert und aufbereitet werden können. Hierzu stehen vorgefertigte Bearbeitungsmöglichkeiten zur Verfügung. Zum Beispiel:

- Tagprüfungen
- Eingabehilfen (Anzeige aller vorgesehen Tags, inkl. Erklärung und dazugehörigem Shortcut)
- Suchmöglichkeiten nach typischen Fehlern
- Bereinigung von Mehrfachblanks und Leerzeilen
- Einfügen von Abständen in Abkürzungen
- Ausgabe als Preview (PS-Datei) oder als Export (RTF-Datei)[15]

Aufruf:

```
#*SATZ,fr_sa1.tf,MO=ED
```

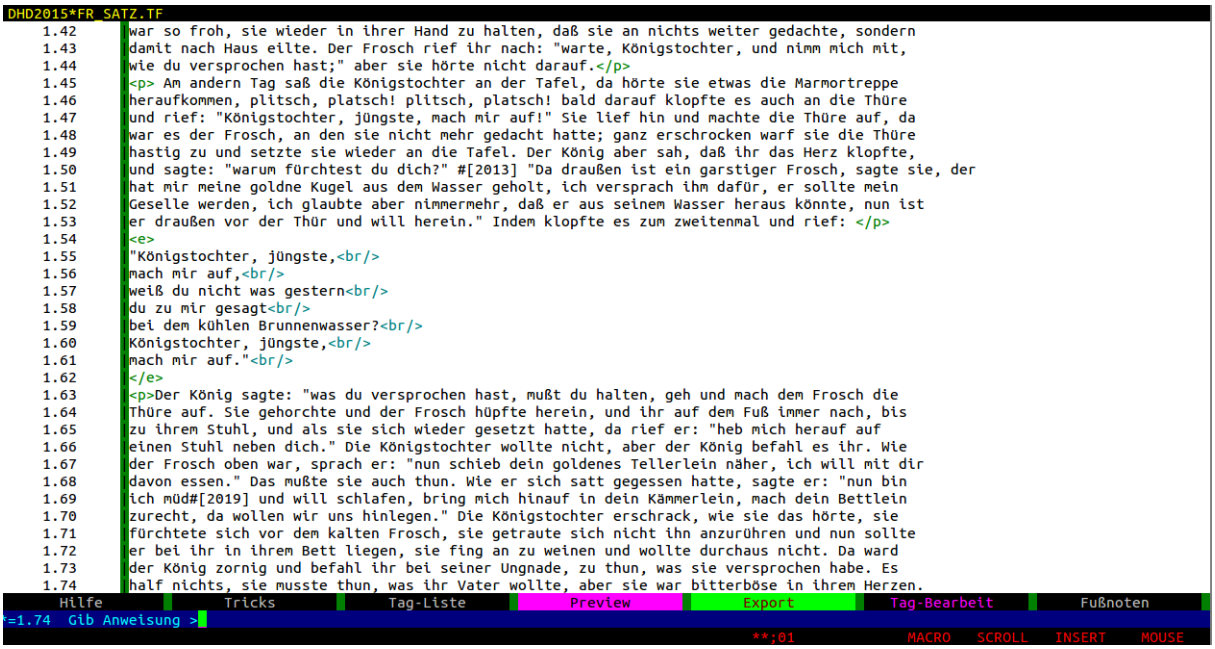

Abbildung 3: Satz-Editor im Modus Ediere

[15] Im Gegensatz zum Standardmakro `#*EXPORT` sind hier keine Eingriffe zur Steuerung der Ausgabe möglich.

Setzen des Froschkönigs (FR_SA1.TF) mit unterschiedlichen Einstellungen (=Parametern)

– Aufruf über die Kommandoebene

exemplarische Aufrufe mit unterschiedlichen Parametern

```
#*SATZ,fr_sa1.tf, MO=satz, PAR=a4, AUSF=s'd
#*SATZ,fr_sa1.tf, MO=satz, PAR=a4weit, AUSF=s'd
#*SATZ,fr_sa1.tf, MO=satz, PAR=a42spalt, AUSF=s'd, SCH=palatino
#*SATZ,fr_sa1.tf, MO=satz, PAR=a43spalt, AUSF=s'd, SCH=bookman
```

– Alternative: Aufruf über #*SATZ-GUI: #*M,SATZ, hier Auswählen der Optionen über die Menüpunkte MODUS,PARSATZ, AUSFUEHREN und SCHRIFT.
– per Klick auf die Dreicke werden die zur Verfügung stehenden Auswahlmöglichkeiten angezeigt
– Start des Makros mit den ausgewählten Parametern → aufrufen

Benutzerspezifische Einstellungen
In der Datei SATZPATR.SEG kann der Benutzer eigene Dokument- und Formatvorlagen definieren bzw. bestehende Vorlagen an individuelle Präferenzen anpassen. Zum detaillierten Vorgehen hierbei s. ›Anhang 2: Eigene Satzparameter. Änderungen in #*SATZ‹ der ›Beschreibung des Standard-Makros #*SATZ‹.[16]
Beispiel: Aufruf mit Parametersatz »a4ms« aus der Datei SATZPATR.SEG mit individuellen Definitionen zu Satzspiegel, Abständen, Zeilendurchschuss…

```
#*SATZ,fr_sa1.tf, MO=satz, PAR=a4ms, AUSF=s'd, SCH=schoolbook
```

Aufruf mit Parametersatz »a4ms2h« für zweispaltigen Satz mit den sonst gleichen Definitionen wie bei »a4ms«

```
#*SATZ,fr_sa1.tf, MO=satz, PAR=a4ms2h, AUSF=s'd, SCH=schoolbook
```

Experimente mit dem Text

Fügen Sie im Text Tags ein, welche Sie in der Beschreibung von #*SATZ finden, etwa:

```
<t1>, <cou>, <hv>, <b>, <c>, <center/>,
<right/>, <page/>, <br/>,  <kl>, <kr>
```

und lassen den Text anschließend wieder setzen, um sich das Ergebnis anzusehen. Achten Sie dabei auf die Wohlgeformtheit der Daten (jedes öffnende Tag braucht ein gleich lautendes schließendes Tag) →

```
<b>...</b>
```

Bei der Eingabe können Sie die Tastenkombinationen ALT+A bzw. ALT+E nutzen, um ein korrespondierendes Anfangs- oder Endetag einzufügen.
Die Wohlgeformtheit können Sie überprüfen mithilfe der Anweisung TP (=Tagprüfung) sowie über die Schaltfläche ›Paare prüfen‹ im Modus ›Ediere‹ unterhalb des Menüs ›Tags bearbeiten‹.

[16] Zum Umgang mit Segmentdateien s. http://tustep.wikispaces.com/Grundlagen+Segmentdatei.

Ausblick

Skripte in Makrodatei `export_segm`
- Ausgabe als HTML-Datei mit Abfrage für benutzerspezifisches Syntax Highlighting
- Ausgabe als HTML-Datei für die Weiterverarbeitung als ePub
- Ausgabe als PS und PDF-Datei direkt von der XML-Datei ausgehend
- Ausgabe als RTF (z. B. für *round-tripping*)
- Ausgabe als plain text (z.B: für statistische Berechnungen)

Anmelden der Makrodatei `export_segm`
Dateimanager öffnen: `#*D`
`export_segm` auswählen und `Lesen` auswählen
Dateimanager verlassen via `ESC` oder `BEENDEN`
Datei `export_segm` als Makrodatei definieren:[17]

```
#DE,MAKRO=1:export_segm
```

definiert die Datei `export_segm` als erste Makrodatei.

Wie oben für das Standardmakro `#*SATZ` gezeigt, können auch selbst geschriebene und definierte Makros über den Dateimanager aufgerufen werden. Hierzu muss die Makrodatei zunächst wie beschrieben definiert werden. Anschließend kann beim Aufruf

```
#*M
```

über den Dialog `Auswählen` aus den Standardmakros und eigenen Makrodateien ausgewählt werden. Wählt man die Datei `export_segm` aus, so stehen die hierin enthaltenen Makros zur Verfügung und können mit Angabe der jeweils obligatorischen Variablen aufgerufen werden. Alternativ können die Prozeduren über die nachstehenden Kommandos aufgerufen werden.

Aufruf der Ausgabe als HTML mit Abfrage von Patterns für Syntax-Highlighting

```
$html,frosch-1.xml,frosch-1.html
```

im anschließend erscheinenden Fenster Angabe von Patterns, z. B.

```
Frosch{00}{&a}'digital'
```

Als Ergebnis wird im Standardbrowser der fertige HTML-Export geöffnet.

Aufruf der Ausgabe als modifiziertes HTML als Grundlage zur Konvertierung in ein ePub mit Calibre[18]

```
$html4epub,frosch-1.xml,frosch-1a.html
```

---

[17] Es können in einem Projekt bis zu drei Makrodateien definiert werden. Die Makrodateien enthalten unterschiedliche Segmente, in denen Programme abgelegt werden können. In einer normalen Segmentdatei können bis zu 9 999 Segmente enthalten sein. Eine Erweiterung stellen »Mega-Segment-Dateien« mit bis zu 999 999 Segmentplätzten dar.

[18] http://calibre-ebook.com/.

Bei der Herstellung ist ggf. darauf zu achten, dass die Einstellungen angepasst werden. In der CSS-Vorschrift ist etwa ein statischer Verweis auf den Windows-Schriftenordner (C:\Windows\Fonts) enthalten, um den Font Ubuntu einbinden zu können. Ein fertig zusammengestelltes ePub wird bei den Workshop-Unterlagen mitgeliefert: ›Froschkonig (1. Auflage), TUSTEP – Grimm, Jacob; Grimm, Wilhelm.epub‹.

Aufruf der Ausgabe als PS-/PDF-Datei direkt von der XML-Datei ausgehend

```
$satz_fr,frosch-1.xml
```

Anschließend liegen im Projektverzeichnis die Zieldateien `fr_satz.ps` und `fr_satz.pdf` vor. Unter Mac und Windows wird zudem die Postscript-Datei geöffnet, unter Linux die PDF-Datei.

Aufruf der Ausgabe als Plain Text (TXT)

```
$plaintxt,frosch-1.xml,fr_1.txt,fr_2.txt
```

In die Datei `fr_1.txt` wird der Inhalt der XML-Datei ohne Markup, aber unter Beibehaltung von formatierenden Leerzeichen und Leerzeilen exportiert. In der Datei `fr_2.txt` sind diese Formatierungselemente entfernt.

Aufruf der Ausgabe als RTF mithilfe des Modus ›Export‹

```
$satz2rtf,frosch-1.xml
```

Aus der TEI–XML-Datei wird eine TUSTEP-Textdatei mit den für das Standardmakro `#*SATZ` vorgesehenen Tags erstellt, die anschließend als RTF-Datei ausgegeben wird.

6. Texte exportieren: #*EXPORT

Mit dem Standardmakro

`#*export` ist es möglich eine Datei aus dem TUSTEP internen Dateiformat in eine RTF-Datei zu konvertieren. Der Export lässt sich über den Dateimanager bewerkstelligen.

7. Anhang

TUSTEP-Kursangebote

TUSTEP-Satzprogramm, Dr. Michael Trauth (ZIMK Trier), 16.–20. März 2015
Einführung in TUSTEP, Dr. Michael Trauth (ZIMK Trier), Oktober 2015
TUSTEP-Vertiefungskurs, Dr. Michael Trauth (ZIMK Trier), August 2015
TUSTEP-Forum, Dr. Michael Trauth (ZIMK Trier), 14-tägig mittwochs 18–20.00 Uhr

TUSTEP-Workshop Blaubeuren, jährlich Anfang Januar, 7.–10. Januar 2016[19]
ITUG-Tagung, jährlich, 7.–9. Oktober 2015, Weimar[20]

Informationen zu Kursen in Trier s. ›Porta‹[21] (Uni Trier)

weitere regelmäßige Kursangebote in Fribourg (CH), und Zürich (CH)

Online-Informationsquellen

TUSTEP-Wiki: http://tustep.wikispaces.com/.

Informationssammlung mit Kurzeinführungen, Programmbeispielen sowie Tipps & Tricks rund um TUSTEP.

Homepage von TUSTEP bei der Universität Tübingen: http://www.tustep.uni-tuebingen.de/.

Hier kann TUSTEP kostenlos als Open Source für Linux, Mac OS und Windows bezogen werden.

Homepage der *International TUSTEP User Group*: http://www.itug.de/.

Hier kann u.a. die TUSTEP-Mailingliste subskribiert werden, über welche Anfragen zu konkreten Problemstellungen sowie TUSTEP- und allgemein DH-Nachrichten sowie Veranstaltungshinweise kommuniziert werden. Des Weiteren sind hier Informationen zu TUSTEP-bezogenen Veranstaltungen und Kursen zu finden.

TUSTEP-Handbücher

Anzeige der Onlinehilfe: `#Hilfe` auf Kommandoebene oder Anweisung `»hilfe«` bzw. `CTRL+O` auf der Editor-/Anweisungsebene

TUSTEP-Beschreibung durchsuchen: `#SUCHE`

TUSTEP-Handbuch. Quellen: `#*ZEBE` (Kommandoebene), »handbuch.pdf« im TUSTEP-Installationsverzeichnis.

TUSTEP Import & Export. Quellen: `#*ZEBE,IMPORT` (Kommandoebene), »importexport.pdf« im TUSTEP-Installationsverzeichnis.

Beschreibung des Standard-Makros #*Satz. Quellen: `#*ZEBE,SATZMAKRO` (Kommandoebene), »satzmakro.pdf« im TUSTEP-Installationsverzeichnis.

TUSTEP Installation · Konfiguration · Aufruf. Quellen: `#*ZEBE,CONFIG` (Kommandoebene), »config.pdf« im TUSTEP-Installationsverzeichnis.

---

[19] http://tustep.uni-tuebingen.de/ver.html.
[20] http://itug.de.
[21] http://www.porta.uni-trier.de.

Literatur

Deutsches Wörterbuch von Jacob und Wilhelm Grimm, 16 Bde. in 32 Teilbänden, Leipzig 1854–1961. Quellenverzeichnis Leipzig 1971. Online: http://www.woerterbuchnetz.de.

Ott, Tobias: Erste Schritte in TUSTEP: http://www.tustep.uni-tuebingen.de/tustein.htm, Tübingen 2013.

Ott, Tobias: Crossmediales Publizieren im Verlag, Berlin 2014.

Ott, Wilhelm: Strategies and Tools for Textual Scholarship: The Tübingen System of Text Processing Programs (TUSTEP), in: Literacy and Linguistics Computing, Jg. 15, Nr 1 /2000, S. 93–108.

Schneider, Matthias; Tobner, René: Einführende Erläuterungen zur Arbeit mit TUSTEP. Importieren und Exportieren von externen Dateiformaten, Tipps und Tricks zum Editor, http://tustep.wikispaces.com/Einführungen, Trier 2014.

TEI (Text Encoding Initiative), http://www.tei-c.org

Cheat-Sheet für den TUSTEP Workshop

Kommandoebene

```
Gib Komando>  [F4], [ESC]   –   Beenden
Gib Komando>  [F6] oder *D  –   Aufruf des Dateimanagers
Gib Komando>  [F9]   –   Blättern in der History (rückwärts)
Gib Komando>  [F10]  –   Blättern in der History (vorwärts)
```

Editor

Funktionstasten

```
[F1]   –  Sprung an Dateianfang
[F2]   –  Sprung an Dateiende
[F4], [ESC]  –  Beenden
[F5]   –  Zeige Umgebung der zuletzt aktiven Zeile
[F6]   –  Liste der zuletzt geöffneten Dateien
[F7]   –  Zeige bis zur zuletzt aktiven Zeile
[F8]   –  Zeige ab der zuletzt aktiven Zeile
[F9]   –  Anzeigen der letzten Editoranweisungen (rückwärts)
[F10]  –  Anzeigen der letzten Editoranweisungen (vorwärts)
[F11]  –  Bildschirm teilen, aktivieren des linken bzw. oberen Fensters
[F12]  –  Bildschirm teilen, aktivieren des  rechten bzw. unteren Fensters
```

Zeigeanweisungen

```
Gib Anweisung> zn,,,  –  zeigeNur
Gib Anweisung> za,,,  –  zeigeAb
Gib Anweisung> zb,,,  –  zeigeBis
Gib Anweisung> zu,,,  –  zeigeUm
Gib Anweisung> ggz    –  Zeige Liste der letzten Zeigeanweisungen
```

Austauschanweisungen

```
Gib Anweisung> a,,,|quelle|ziel|
Gib Anweisung> gga  –  zeige Liste der letzten Austauschanweisungen
```

Sonstiges

```
Gib Anweisung> [ctrl]-[k]-[blank]  –  Anzeige einer Übersicht zum
                                      Pattern Matching
Gib Anweisung> [alt]-[q]  –  Auswahlmenu für Eingabe von >Sonderzeichen<
```

Editoranweisungen für Tags

```
Gib Anweisung> [alt]-[a]  –  Ergänze Anfangstag
Gib Anweisung> [alt]-[e]  –  Ergänze Endetag
Gib Anweisung> [alt]-[v]  –  Nächstes offene Tag Richtung Dateiende
Gib Anweisung> [alt]-[r]  –  Nächstes offene Tag Richtung Dateianfang
Gib Anweisung> tl  –  Liste der verwendeten Tags
Gib Anweisung> tlh –  Liste der verwendeten Tags hierarchisch
Gib Anweisung> tp  –  Tagprüfung (Wohlgeformtheit)
Gib Anweisung> tpv –  Tagprüfung (Wohlgeformtheit) Rtg. Dateiende
Gib Anweisung> tpr –  Tagprüfung (Wohlgeformtheit) Rtg. Dateianfang
```

TUSCRIPT

Skripte werden in der Regel in Segmentdateien abgelegt. Segmente werden zur Bearbeitung in eine andere Datei geholt und anschließend zurückgeschrieben (»gerettet«).

```
Gib Anweisung> h,segmentdatei,? – Inhaltsverzeichnis der Segmentdatei
Gib Anweisung> h,segmentdatei,segment – Segment in Editor (zur
                                          Bearbeitung) holen
Gib Anweisung> r,segmentdatei,segment – Segment in Segmentdatei
                                         schreiben (retten)
```

Interpretationsmodus

```
$$ MODE TUSCRIPT,{}
```

»Es folgt ein Skript…«

{}: Konventionen für Such-, Austausch- und Recherchiertabellen

Wertzuweisung

Variablenname=Wert (»Zeichenfolge« | Rechenanweisung | Funktion)

Schleifen

```
LOOP
...
ENDLOOP
```

Bedingungen

```
IF (Bedingung) THEN
...
ELSEIF (Bedingung) THEN
...
```

```
ELSE
...
ENDIF
```

Auswahl

```
SELECT variablenname
CASE "Zeichenfolge"
...
CASE "Zeichenfolge_a", "Zeichenfolge_b"
...
DEFAULT
...
ENDSELECT
```

Suchtabellen, Austauschtabellen, Recherchiertabellen

S_TABLE **Suchtabellen** enthalten die Zeichenfolgen, nach denen gesucht wird;

X_TABLE **Austauschtabellen** enthalten Zeichenfolgen, nach denen gesucht wird, sowie die Zeichenfolgen durch die diese ersetzt werden sollen;

R_TABLE **Recherchiertabellen** enthalten Zeichenfolgen, die gesucht werden sollen. Mehrere Zeichenfolgen können durch Optionen verbunden werden: /AND /OR /EXACT…;

Dictionary

```
DICT name action/option key, num, cnt, value1, value2
```

Dateizugriffe auf Daten mit Anfangs- und Endekennungen (Tags)

```
ACCESS daten: modus/option "dateiname" ...
   nummer, tagbeg/beg+text+tagend/end, typ, stack
    LOOP
        READ/EXIT daten
    ENDLOOP
ENDACCESS daten
```

Typ der gelesenen Textportion

0 = nur Text, 1 = Anfangskennung, 2 = Endekennung, 3 = Anfangs- und Endekennung, 4 = nur leeres Tag

Funktionen zur Dateiverwaltung

```
OPEN (datei,READ|WRITE,–std-)
CREATE (datei,seq-o|fdf-o,–std-)
FILE_NAMES (–,–std-) – (– = aktuelles Projekt; –std-= Standardträger)
```

Weitere Funktionen

```
EXCHANGE      (variable,X_TABLE)
EXTRACT       (variable,anfpos,endpos)
CONCAT        (variable,"zeichenfolge")
APPEND        (variable_1,variable_2)
FILTER        (variable,posauswahl,negauswahl)
SPLIT         (variable,Trenner,teil1,teil2,...)
DIGIT_INDEX   (variable) – erstellt Sortierindex mit Positionsangaben
INDEX_SORT    (variable, index) – Sortierung nach Positionsangaben in
               der Variablen index
REDUCE        (variable) – entfernt aufeinanderfolgende identische
               Zeichenfolgen
ACCUMULATE    (variable,anzahl) – wie REDUCE mit Häufigkeitsangabe
REQUEST       (url,query)
```

Definitionen · Pattern Matching

Einzelzeichen inklusive Escapezeichen/Maskierung

```
\?        Fragezeichen
\*        Stern
\[        eckige Klammer auf
\]        eckige Klammer zu
\{        geschweifte Klammer auf
\}        geschweifte Klammer zu
\a        Kleinbuchstabe a
\A        Großbuchstabe A
\\       Backslash
```

Zeichen- und Stringgruppen

```
?         ein beliebiges TUSTEP-Zeichen
*         null bis beliebig viele beliebige TUSTEP-Zeichen
[...]     lokale Zeichengruppe, z.B. m[ae][iy]er
{C:xy}    Charactergruppe xy
{S:xy}    Stringgruppe xy
{–}       nachfolgende Zeichen aus der Zeichengruppe entfernen
{+}       nachfolgende Zeichen in die Zeichengruppe aufnehmen
```

Vordefinierte Zeichengruppen

```
{\a}       Kleinbuchstaben
{\A}       Großbuchstaben
{&a}       Kleinbuchstaben & Großbuchstaben
{\0}       normale Ziffern ohne hochgestellte Ziffern
```

Häufigkeitsbedingungen in Suchzeichenfolgen

```
{n}       genau n Elemente
{n-m}     n bis m Elemente, möglichst wenig
{n--m}    n bis m Elemente, möglichst viele
{0}       0 oder 1 Element, gleichbedeutend mit {0–1}
{00}      1 bis unendlich viele Elemente, gleichbed. mit {1–0}
{0–0}     0 bis unendlich viele Elemente
```

Verweise in Suchzeichenfolgen

```
{+n=}     n-tes Element von links gezählt,  a != A
{-n=}     n-tes Element von rechts gezählt, a != A
{+n:}     n-tes Element von links gezählt,  a == A
{-n:}     n-tes Element von rechts gezählt, a == A
```

Verweise in Ersatzzeichenfolgen

```
{+n=}     n-tes Element von links gezählt
{-n=}     n-tes Element von rechts gezählt
{=n=}     alle Elemente des n-ten Elementbereichs
{=0=}     alle Elemente der Kernzeichenfolge
{+n-m=}   n-tes bis m-tes Element von links gezählt
{+n–0=}   n-tes bis letztes Element von links gezählt
{-n-m=}   n-tes bis m-tes Element von rechts gezählt
{–0-m=}   erstes bis m-tes Element von rechts gezählt
{=n-m=}   alle Elemente des n-tes bis m-ten Elementbereichs
{...+}     ... aus Kleinbuchstaben werden Großbuchstaben
{...–}     ... aus Großbuchstaben werden Kleinbuchstaben
```

Sonstiges in Suchzeichenfolgen

```
{[}       Wechsel vom linken Rand zur Kernzeichenfolge
{]}       Wechsel von der Kernzeichenfolge zum rechten Rand
{|}       Wechsel von einem Elementbereich in den nächsten
```

Beispiele

Anweisung »Zeige alle Doppelvokale an.« →

```
Definition einer Character Group + Suchanweisung
C:vo=aeiouyäöü
ZN,,,|{C:vo}{+1=}|
```

»Zeige Wörter der lateinischen A-Deklination in ihrer Umgebung an.« →

```
Definition einer String Group + Suchanweisung
S:ad=|a|ae|am|arum|is|as|
ZU,,,|{00}{&a}{S:ad}|
```

»Zeige die Vorkommen des Namens Johannes in seinen unterschiedlichen Schreibweisen an.«

```
Suchanweisung mit lokalen Zeichengruppendefinitionen
ZN,,,|[IJ]o{0}ha{1-2}n{0}[ei]{0}s|
```

→ gefunden werden hiermit u. a.: Johannes, Johannis, Johanne, Johann, Joan, Ioannis, Ioannes, Iohannes, Iohannis, Ioanne, Iohanne, Ioanni, Iohanni.
Das französische Jean kann ggf. durch eine Erweiterung der Suchzeichenfolge beim ersten Vokalvorkommen ergänzt werden.

Übungsaufgaben für `dhd_wo.tf`

– Formulieren Sie einen möglichst kurzen Suchstring, um alle Überschriftentags (<h1>, <h2> und <h3>) präzise zu finden.

```
ZN,,,|<{0}/h?>|
```

– Suchen Sie alle Wörter mit Doppelkonsonanten.

```
Character group definieren, alle Buchstaben außer Vokale.
Suche nach einem Konsonanten, der sich wiederholt.
C:ks={&a}{-}aeiouyäöü{+}
ZN,,,|{C:ks}{+1=}|
```

– Wie viele Wörter beginnen mit einem groß geschriebenen S?

```
Character group definieren, jedes Zeichen außer Blank.
Großes >S< und beliebige viele Zeichen außer Blank.
C:nb=?{-} {+}
ZN,,,|\S{00}{00}{C:nb}|
```