Neuerungen in TUSTEP Version 2014
gegenüber Version 2013 (Übersicht)
Stand: 17.6.2014

Installation von TUSTEP

Für Installation, Konfiguration und Aufruf von TUSTEP unter Windows, Linux und Mac OS gibt es eine neue Anleitung, die auf der TUSTEP-homepage unter "Installationsanleitungen" zugänglich ist oder (wenn TUSTEP schon installiert ist) mit #*zebe,config angezeigt werden kann.

Identifizieren einer Datei *{21}*

Wenn der Inhalt einer System-Variablen, die als Träger in TUSTEP verwendet wird, mit einer Tilde beginnt, wird diese Tilde durch den Inhalt der System-Variablen HOME ersetzt.

Definieren einer TUSTEP-Sitzung *{59}*

Beim Definieren von TUSTEP-Sitzungen mit #*desi muss eine von 0 verschiedene Sitzungsnummer angegeben werden.

Bei den beiden Systemvariablen TUSTEP_DSK und TUSTEP_SCR werden noch nicht existierende Verzeichnisse nach Rückfrage eingerichtet.

Beginnen einer TUSTEP-Sitzung *{63}*

Beim Initialisieren einer TUSTEP-Sitzung ist ab Version 2014 Parameter-Modus {} voreingestellt. Wenn noch mit Parameter-Modus <> gearbeitet werden soll, muss deshalb nach dem Start von TUSTEP das Kommando #parameter,<> gegeben werden. Soll eine TUSTEP-Sitzung immer mit Parameter-Modus <> gestartet werden, so kann dies durch Einfügen des Kommandos #parameter,<> in die Datei TUSTEP.INI erreicht werden.

INI-Datei *{64}*

Existiert zu Beginn einer TUSTEP-Sitzung keine INI-Datei, so wird ab Version 2014 nicht mehr nachgefragt, ob eine solche eingerichtet werden soll. Von den bisher in der Regel in der INI-Datei über Kommandos vorgenommenen Einstellungen sind jetzt folgende voreingestellt:

```
#PARAMETER, {}
#PARAMETER, AUS
#PROTOKOLL, PORTIONIERT
#WISCHEN, AUS
#DEFINIERE, CODE=-STD-, FARBEN=-STD-
```

Kommandos / Eingabe *{86}*

Bei der Eingabe ergeben sich einige Erleichterungen:

- Da auf manchen Notebooks die Plus-Taste des Ziffernblocks nur umständlich zu erreichen oder nicht vorhanden ist, kann statt der Plus-Taste grundsätzlich auch die Tastenkombination Strg+B bzw. Ctrl+B verwendet werden.
- Auch auf der Kommandoebene kann mit Ctrl+V bzw. Strg+V der Inhalt der Zwischenablage vor der aktuelle Cursor-Position eingefügt werden.
- TAB erhält eine zusätzliche Funktion: extend file name *{87}*

  Wenn für ein parametergesteuertes Programm (z.B. #KOPIERE) Parameter eingegeben werden, springt der Cursor auf die nächstfolgende Tabulatorposition; Tabulatorpositionen sind 11, 21, ..., 71.
  In allen anderen Fällen wird ein Dateiname, dessen Anfang unmittelbar links vom Cursor steht, vervollständigt. Dabei werden nur Namen von angemeldeten Dateien (einschließlich Scratch-Dateien) berücksichtigt. Ist der Anfang des Dateinamens nicht eindeutig, so wird beim ersten Mal der Name bis zu der Stelle ergänzt, an der er mehrdeutig wird; danach wird jeweils der nächste in Frage kommende Dateiname angezeigt. Steht links vom Cursor ein Leerzeichen, Komma, Gleichheitszeichen oder Apostroph, so wird "-STD-" ergänzt.

#MBKOPIERE *{160}*

POSITIV = – * Keine Positiv-Auswahl von Dateien, deren Name einem vorgegebenen Muster entspricht.

= ... Nur solche Dateien kopieren, deren Name entweder explizit zur Spezifikation QLABEL angegeben ist, oder deren Name mindestens einem der angegebenen Muster entspricht.

NEGATIV = – * Keine Negativ-Auswahl von Dateien, deren Name einem vorgegebenen Muster entspricht.

= ... Nur solche Dateien kopieren, deren Name entweder explizit zur Spezifikation QLABEL angegeben ist, deren Name keinem der angegebenen Muster entspricht.

#EDIERE

Lestung / Hinweis *{119}*

Falls die zu edierende Datei nicht angemeldet ist, wird nachgefragt, ob sie angemeldet bzw. (falls sie auf dem Träger TUSTEP_DSK nicht existiert) eingerichtet werden soll.

Der Editor kann auch mit dem Standard-Makro *E aufgerufen werden (jetzt auch für Nicht-TUSTEP-Dateien, siehe unten).

Ignorieren von Akzent-Codierungen ("%" für "begr") *{283}*

Zu den beim Suchen übergangenen Codierungen gehören auch die Zeichenfolgen #, und #’

Ignorieren von Auszeichnungen/Schriftumschaltungen ("%%" für "begr") *{283}*

Zu den beim Suchen übergangenen Codierungen gehören auch die Zeichenfolgen #, #’ #h: #t: #o: #u: #g: sowie Tags, deren Name aus nur einem Buchstaben besteht, z.B. bzw. .

Tastenkombinationen für Steuerbefehle *{300}*

	...	Plus-...	Plus-Plus-...
Shift+Backspace	DEL	DEL_REC	UND_REC
Shift+Return	ENTER	TGL_INS	REFRESH

Verzweigen in andere Makros *{325}*

NEXT_BTAB:name	Wenn der Steuerbefehl BACKTAB zum nächsten Mal ausgeführt werden soll (z.B. beim Drücken der Shift- und Tabulatortaste), wird stattdessen das Makro mit dem angegebenen Namen aufgerufen und ausgeführt.
NEXT_BTAB:*	Die Abarbeitung des Makros wird unterbrochen; wenn der Steuerbefehl BACKTAB zum nächsten Mal ausgeführt werden soll (z.B. beim Drücken der Shift- und Tabulatortaste), wird stattdessen die Abarbeitung des Makros fortgesetzt.

Anzeigen von Meldungen *{332}*

MESSAGE:"text"	Zeigt den angegebenen Text in der Meldungszeile an. Der Text muss mit einem frei wählbaren Sonderzeichen, das im Text selbst nicht vorkommt, eingeleitet und abgeschlossen werden. Der Text wird in der für die Meldungszeile eingestellten Farbe angezeigt. Soll er in einer anderen Farbe angezeigt werden, so kann nach einem Doppelpunkt der entsprechende Hexadezimal-Code angegeben werden. Wird als Text eine leere Zeichenfolge angegeben, so wird der in der Anweisungszeile stehende Text angezeigt. Damit ist es möglich, einen dynamisch erstellten Text anzuzeigen.

STATUS:"text" Zeigt den angegebenen Text in der Statuszeile an.
Der Text muss mit einem frei wählbaren Sonderzeichen, das im Text selbst nicht vorkommt, eingeleitet und abgeschlossen werden. Der Text wird in der für die Statuszeile eingestellten Farbe angezeigt. Soll er in einer anderen Farbe angezeigt werden, so kann nach einem Doppelpunkt der entsprechende Hexadezimal-Code angegeben werden. Wird als Text eine leere Zeichenfolge angegeben, so wird der in der Anweisungszeile stehende Text angezeigt. Damit ist es möglich, einen dynamisch erstellten Text anzuzeigen.

MRK_DSP Zeigt den Inhalt des Zwischenspeichers in einem Popup-Fenster an.
Zeilenwechsel mit #:(nl). Damit können längere dynamisch erzeugte Texte angezeigt werden.

Standard-Makros

#*CB Mit dem Makro *CB kann unter Linux das Clipboard in die TUSTEP-Zwischenablage und umgekehrt kopiert werden.

MODUS = EXPORT TUSTEP-Zwischenablage –-> Clipboard
= IMPORT Clipboard –-> TUSTEP-Zwischenablage

#*DERI Mit dem Makro *DERI können Icons für Remote-Sitzungen erstellt werden. Mit einem solchen Icon kann eine Verbindung zu einem Linux-Rechner hergestellt und eine TUSTEP- Sitzung gestartet werden.

Das Makro hat keine Spezifikationen.

#*E Mit dem Makro *E kann eine Datei zum Lesen oder Schreiben angemeldet und mit dem Editor bearbeitet werden. Die Spezifikationen LESEN, SCHREIBEN und TRAEGER entsprechen denen beim Kommando #ANMELDE; die Spezifikation MAKRO entspricht der beim Kommando #EDIERE.

Wenn die betreffende Datei keine TUSTEP-Datei ist, werden die Daten umgewandelt, in eine interne TUSTEP-Datei geschrieben und der Editor mit dieser internen Datei gestartet. Falls die Daten geändert werden, wird nach dem Beenden des Editors gefragt, ob die Daten in die ursprüngliche Datei zurück-

geschrieben werden sollen.
Die Spezifikation CODE entspricht der beim Kommando #UMWANDLE.

Wird beim Aufruf zu den Spezifikationen LESEN und SCHREIBEN nichts angegeben, so wird ein Fenster angezeigt, in dem sowohl der Projekt- und Dateiname als auch der Träger ausgewählt werden kann.

#*EDIDEF — Festlegen, mit welchen Definitionen der Editor arbeiten soll, wenn er über den Dateimanager des Betriebssystems gestartet wird.

#*PB — Mit dem Makro *PB kann unter MAC OS das Pasteboard in die TUSTEP-Zwischenablage und umgekehrt kopiert werden.

MODUS = EXPORT — TUSTEP-Zwischenablage –-> Pasteboard
= IMPORT — Pasteboard –-> TUSTEP-Zwischenablage

#*PS2PDF — Mit dem Makro *PS2PDF kann eine PostScript-Datei in eine PDF-Datei umgewandelt werden.

#*ZEPS — Mit dem Makro *ZEPS kann eine Postscript-Datei auf dem Bildschirm angezeigt werden. Wenn die Datei angezeigt wird, kann sie über die Druckfunktion des Anzeigeprogramms auf einen Drucker ausgegeben werden.

DATEI = ... — Name der anzuzeigenden Postcript-Datei.

Parametergesteuerte Programme

#KOPIERE *{851}*

AAZ — 2 = wie 0, jedoch zusätzlich auch die Zeichenfolgen #[09] und #[0009] (= Tabulator) am Anfang von Eingabesätzen vor der Auswertung des Parameters AA entfernen.

Dies gilt für Parameter AAZ auch in #FA, #RV und #SV.

#RAUFBEREITE

Z — Der Parameter Z ist jetzt auch bei MODUS=KWIC erlaubt *{925}*

ZZZ — Der Parameter ZZZ ist jetzt auch bei MODUS=KWIC erlaubt *{944}*

Ergänzungen um Referenz/Kontext (nur bei MODUS=KWIC) *{941}*

Die beiden folgenden Parameter wirken sich nur in der ZIEL-Datei aus.
Mit ihnen kann zwischen dem Satz, der das Schlüsselwort enthält, und dem ersten Satz, der dazu Referenz und Kontext enthält, bzw. nach dem letztem Satz, der zum selben Schlüsselwort Referenz und Kontext enthält, jeweils ein eigener Satz eingefügt werden.

VRK	Zeichenfolgen, die (typenabhängig) vor dem ersten Satz mit Referenz und Kontext ergänzt werden sollen.
NRK	Zeichenfolgen, die (typenabhängig) nach dem letzten Satz mit Referenz und Kontext ergänzt werden sollen.

#VERGLEICHE

Angaben zum Zuordnen *{1040}*

NGZ	n	Bei wortweisem Vergleichen versucht das Programm, in den erzeugten Korrekturanweisungen die gefundenen Unterschiede möglichst wortgenau dem Text der Version A zuzuordnen. Dabei müssen die Wörter nicht zeichengenau übereinstimmen. Mit dem (neuen) Parameter NGZ kann angegeben werden, dass bei diesem Zuordnen bis zu n aufeinanderfolgende Wörter keine Unterschiede aufweisen dürfen. So bewirkt die Angabe des Zahlenwerts 1 bei NGZ, dass bei Version A: alles sehr genau zu sehen Version B: alles auf ein har zu sehen die Korrekturanweisung 1.1,2–3[sehr genau]=auf ein har statt 1.1,1[alles ::]+auf ein 1.1,2[sehr]=har 1.1,3[genau]– erzeugt wird.

#*SATZ

*SATZ wurde auf den Parameter-Modus {} umgestellt.
Er gilt für Makroanweisungen, Parameter und für den Editor.
Nur im Editor ist er für den Benutzer von Bedeutung, wenn er Zeichen- und Stringgruppen, Häufigkeits- oder Zahlenwertbedingungen u.ä. benutzt.

Das Programm "merkt sich" die Angaben zu QUELLE, MODUS, PARSATZ, AUSFUEHREN und OPTIONEN. Wenn die Angaben

beibehalten werden sollen, genügt dann bei gleicher Quelldatei der Aufruf #*satz. Bei Angabe einer anderen Quelldatei werden jetzt die Angaben auf die Voreinstellungen zurückgesetzt.

Neue und wegfallende TAGS, erweiterte Funktion

<titel> schaltet die Ausgabe der Seitennummer aus, </titel> schaltet sie ein. Damit wird die Titelei auch ohne Angabe von SNUM=AUS beim Programmaufruf unpaginiert gesetzt.
<lias/> Leere linke Ausgangsseite erzwingen. Wirkt nur am Dateiende, wenn der Satz auf rechter Seite endet.
<ez4/> Einzug ab aktueller Position und Folgezeilen um 4 Quadrate.
<ea4/> Zeilenwechsel, Einzug um 4 Quadrate nur in aktueller Zeile.
<red> Satz mit roter Schrift. Ebenso 21 andere vorgegebene Farben.

Neue OPTION

Bei mehrspaltigem Satz ohne Fußnoten und Binnenspalten wird Spaltenausgleich durchgeführt, damit die Spalten an Kapitelenden gleich lang werden. Der Spaltenausgleich kann mit dem Kommando #*SATZ,quelle, ..., OPTION=NOSPAUS unterdrückt werden.

Modus EDIEREN

Prüfung der Fußnoten

Fußnoten können zur Prüfung und Bearbeitung einzeln angezeigt werden. Mit linkem Mausklick auf das neue Feld "Fußnoten" wird auf die Fußnoten-Bearbeitungsleiste umgeschaltet. Klick auf "Hilfe" oder rechter Mausklick auf "Fußnoten" liefert einen Hilfetext. Mit Alt+5 (Buchstabenblock) erhält man den Anfang einer Austauschanweisung über den Bereich der angezeigten Fußnote. Mit Alt+6 (Buchstabenblock) wird die letzte Austauschanweisung für die gerade angezeigte Fußnote gestartet.

Mit der Tastenkombination Umschalt+Alt+ B (wie Block) kann die Belegung des Ziffernblocks von Ziffern (Voreinstellung) auf Funktionen umgestellt werden.

Die Tasten sind dann wie folgt belegt:

0 Umschaltung insert/replace
1 Cursor springt auf Position für Fortsetzung der Eingabe (nur im Eingabemodus)
2 zeigt ab Zeile nach Sternposition
3 zeigt Anfang der folgenden Seite (nur Textmodus)
4 zeigt Seite, die in Anw.zeile angegeben ist (nur Textmodus)
5 zeigt aktuelle Seite (nur Textmodus)
6 zeigt Anfang der aktuellen Seite (nur Textmodus)
7 springt auf nächstes Blank

```
     8  zeigt ab Zeile vor Sternposition
     9  zeigt Anfang vorangehender Seite
  Mit derselben Tastenkombination kann auf die Belegung mit
  Ziffern zurückgeschaltet werden.

  Näheres siehe #*SATZ,,NEUERUNGEN; ausführliche Beschreibung
  mit #*ZEBE,SATZMAKRO


#SATZ

Parameter

  Satzspiegel

    HOE                Wenn durch ;8 hinter dem zweiten      {1077}
                       Wert von HOE Durchschussveränderung
                       in Achtelpunkt-Schritten verlangt ist,
                       kann durch ;n unmittelbar dahinter
                       angegeben werden, um wieviele
                       Achtelpunkt der Durchschuss maximal
                       verringert werden darf.
                       Voreinstellung: 8 Achtelpunkt.

Steueranweisungen

  Freiraum; Einbinden von Grafiken

    $$!!$${          Seitenhoher Freiraum.                 {1130}
                       Wird diese Anweisung unmittelbar
                       hinter einem Seitenwechsel-Kommando
                       gegeben, so wird der Freiraum dort
                       unmittelbar ausgegeben, nicht erst
                       nach dem nächsten Zeilenwechsel.

    $$!!#n$${        Seitenhoher Freiraum für Abbildung n.
                       Wird diese Anweisung unmittelbar
                       hinter einem Seitenwechsel-Kommando
                       gegeben, so wird der Freiraum mit
                       der Abbildung dort unmittelbar
                       ausgegeben, nicht erst nach dem
                       nächsten Zeilenwechsel.

  Zeilenumbruch, Spatien

    _ @+               Bei @+ fällt auch ein vor dem         {1143}
                       Spatium frei stehender fester
                       Ausschluss einschl. des davor
                       stehenden Spatiums weg

  Einzüge, Einrückungen

    &=(Pm+nnn;z)       Die Zusatzangaben ";z", die die       {1150}
                       Einrückung auf z Zeilen beschränkt,
                       gilt auch für Einrückungen, die sich
                       auf eine zuvor mit &!Mn gemerkte
                       Position beziehen.
```

Zeilenzähler; Marginalien *{1158}*

&!R(:3)	Ab hier bei zweistelligen Zeilennummern ein zusätzliches (ziffernbreites) Spatium vor der Zeilennummer einfügen, um die Einerstellen für auf der selben Seite vorkommende dreistellige Zeilennummern rechtsbündig übereinander zu setzen. Dies sollte erst eingesetzt werden, wenn der Umbruch feststeht.
&!R(:2)	Zurücksetzen auf Voreinstellung: ohne zusätzliches Spatium vor zweistelligen Zeilennummern.

Satzzeichen, Sonderzeichen

&!B(–)	Die Spatien zwischen &B(–) und &B(+) werden nicht zum Austreiben bzw. Zusammenschieben der Zeilen auf die aktuelle Zeilenbreite herangezogen. *{1182}*
&!B(+)	siehe &!B(–). Am Abschnittende wird ein nach &!B(–) noch fehlendes &!B(+) automatisch ergänzt. *{1182}*

Makros für die Satzumgebung

#*PSAUS *{1268}*

In #*PSAUS gibt es eine weitere Spezifikaton:

TITLE = text	Der angegeben Text (der keine Akzente oder nicht-lateinischen Buchstaben enthält) wird als "%%TITLE: text" in den Prolog der PS-Datei eingesetzt.

#*SILLIST *{1300}*

TRENNUNGEN = name	In der zu TRENNUNGEN angegebenen Datei können auch Einträge mit mehr als einer Trennstelle stehen (das Makro *SILARCH führt identische Wörter mit mehr als einer Trennstelle als separate Einträge mit den jeweilgen Trennstellen auf).

* * * * * * *

Parameter-Modi seit TUSTEP Version 2012

Seit der TUSTEP-Version 2012 steht zur Kennzeichnung von Zeichen- und Stringgruppen, Häufigkeitsangaben, Verweisen usw. in Anweisungen bzw. Parametern zur Mustererkennung eine weitere (dritte) Konvention zur Verfügung, bei der geschweifte Klammern statt wie bisher spitze Klammern verwendet werden.

Sie war notwendig geworden, um Parameter oder Kommandomakros, die solche Angaben enthalten, auch mit einem XML-Editor schreiben oder ändern zu können.
Sie empfiehlt sich unabhängig davon vor allem für neue Projekte, weil die entsprechenden Codierungen leichter zu merken und leichter zu lesen sind.

Nach welcher Konvention die Parameter und die Editor-Anweisungen interpretiert werden, kann mit dem Kommando #PARAMETER eingestellt oder innerhalb von Parametern mit dem Parameter par festgelegt werden. In Kommandomakros wird der Interpretationsmodus mit der Anweisung $$MODE eingestellt.

Mit dieser Übersicht soll eine kompakte Umstellungshilfe von #parameter,modus=<> auf #parameter,modus={} bzw. von $$ MODE <> auf $$MODE {} geboten werden. Sie soll die ausführliche Beschreibung im Handbuch ergänzen.

Modus	Modus	
<>	{}	

Vordefinierte Zeichengruppen

<%	?	ein beliebiges Zeichen
>%	{!}	ASCII-Zeichen
	{;}	TUSTEP-Zeichen außer ASCII-Zeichen
<@	{@}	Zeichen außer Buchstaben und Ziffern
>@	{%}	Zeichen hinter % zur Akzentcodierung
>*	{\a}	Kleinbuchstaben
<*	{\A}	Großbuchstaben
</	{&a}	Kleinbuchstaben & Großbuchstaben
	{\0}	normale Ziffern
>/	{&0}	normale Ziffern & hochgestellte Ziffern
><<><%	*	null bis beliebig viele beliebige Zeichen

Zeichen- und Stringgruppen

– Definition im Editor:

>[vo]=aeiou	C:vo=aeiou
	Z:vo=aeiou
<[di]=|ei|	S:di=|ei|

– Definition in Parametern (Sp. 1–3):

>xy	>xy	Definition einer Zeichengruppe
>1z		Definition einer Zeichengruppe
<1z		Definition einer Zeichengruppe

```
<xy       <xy       Definition einer Stringgruppe
>1s                 Definition einer Stringgruppe
<1s                 Definition einer Stringgruppe
```

– in der Definiton von Zeichengruppen (ab Sp. 11):

```
><        {–}       nachfolgende Zeichen aus der Gruppe entfernen
<>        {+}       nachfolgende Zeichen in die Gruppe aufnehmen
```

– Bezug auf eine Zeichen- oder Stringgruppe:

```
          [...]     lokale Zeichengruppe, z.B. m[ae][iy]er

>[xy]     {Z:xy}    selbstdefinierte Zeichengruppe xy
          {C:xy}    alternative Schreibweise für {Z:xy}
<[xy]     {S:xy}    selbstdefinierte Stringgruppe xy
>1                  selbstdefinierte Zeichen- oder Stringgruppe
<1                  selbstdefinierte Zeichen- oder Stringgruppe
```

Häufigkeitsbedingungen in Suchzeichenfolgen

```
><n       {n}       genau n Elemente
><n<>m    {n-m}     n bis m Elemente, möglichst wenige
<>m><n    {n--m}    n bis m Elemente, möglichst viele

><0       {0}       0 oder 1 Element = {0–1}
<>0       {00}      1 bis unendlich viele Elemente = {1–0}
```

Zahlenwertbedingungen in Suchzeichenfolgen

```
          {#}       Zahl mit beliebigem Wert
>={n-n}   {#n}      Zahl mit dem Wert n
<{n-n}    {!n}      Zahl mit einem Wert ungleich n
>={n-m}   {#n-m}    Zahl mit einem Wert von n bis m
<{n-m}    {!n-m}    Zahl mit einem Wert kleiner n oder größer m
```

Verweise in Suchzeichenfolgen

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

Elementbereiche in Suchzeichenfolgen

```
          {|}       Begrenzungszeichen zwischen Elementbereichen
```

Verweise in Ersatzzeichenfolgen

```
>=nn      {+n=}     n-tes Element von links gezählt
<=nn      {-n=}     n-tes Element von rechts gezählt
>=00      {+0=}     alle Elemente der Kernzeichenfolge
<=00      {–0=}     alle Elemente der Kernzeichenfolge
>=(n-m)   {+n-m=}   n-tes bis m-tes Element von links gezählt
>=(n–0)   {+n–0=}   n-tes bis letztes Element von links gezählt
<=(n-m)   {-n-m=}   n-tes bis m-tes Element von rechts gezählt
<=(0-m)   {–0-m=}   erstes bis m-tes Element von rechts gezählt
          {=n=}     alle Elemente des n-ten Elementbereichs
          {=0=}     alle Elemente der Kernzeichenfolge
```

	{=n-m=}	alle Elemente des n-ten bis m-ten Elementbereichs
>+nn etc	{...+}	... Kleinbuchstaben –-> Großbuchstaben
>-nn etc	{...–}	... Großbuchstaben –-> Kleinbuchstaben
>:nn etc	{...;}	... a, b, ..., $, ... –-> ä, ^b, ..., ^$, ...
>;nn etc	{...!}	... ä, ^b, ..., ^$, ... –-> a, b, ..., $, ...

Einzelzeichen

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

Sonstiges

<|	{[}	linker Rand
>|	{]}	rechter Rand
><	{|}	in Sortieralphabet-Tabellen: Umschalten auf höchste Wertigkeiten

Im Editor:

CTRL+K + erstes Sonderzeichen, das in den Klammern vorgesehen ist, erleichtert die Eingabe von {...} .

CTRL+K + Blank zeigt eine Übersicht über die nach #pa,{} geltenden Konventionen zum Schreiben der Parameter und der entsprechenden Editor-Anweisungen.

Parameter-modes valid from TUSTEP version 2012

Starting with TUSTEP version 2012, for marking character groups and string groups, frequencies, pointers etc. in pattern matching expressions, a new (third) convention is available, using curly brackets instead of (as before) angle brackets.

This has become necessary in order to be able to write down expressions containing these elements by means of an XML editor. We recommend to use the new conventions above all for new projects: the new syntax is by far easier to remember and to interpret by a human reader.

Which conventions shall be used for interpreting the parameters and the editor instructions, can be defined using the command #PARAMETER or, within parameters, by the parameter par. In command macros, the mode of interpretation can be defined using the $$MODE instruction.

In command sequences / in the editor:

#parameter, modus=<>

(equivalent to #parameter,modus=neu):
use angle brackets for marking character- and string groups, frequencies, pointers etc.

#parameter, modus={}

use curly brackets for marking character- and string groups, frequencies, pointers etc.

Within parameters:

par <> the following parameters are to be interpreted according to the <> parameter mode.

par {} the following parameters are to be interpreted according to the {} parameter mode.

The specification given with the help of parameter PAR has priority over the interpretation mode given with the command #PARAMETER; it is valid only for the subsequent parameters up to a further parameter PAR or the end of the parameter set.

In command macros:

$$ MODE <> use angle brackets for marking character- and string groups, frequencies, pointers etc.

$$ MODE {} use curly brackets for marking character- and string groups, frequencies, pointers etc.

For the reason of compatibility, <> is used when no indication is given in the $$ MODE instruction.

This overview tries to give a compact help for the changeover from #parameter,modus=<> to #parameter,modus={} and from $$ MODE <> to $$MODE {} and is thought as a supplement to the full description in the users manual.

Modus	Modus	
<>	{}	
Predefined character groups		
<%	?	any character
>%	{!}	ASCII character
	{;}	TUSTEP character beyond ASCII characters
<@	{@}	characters except letters and digits
>@	{%}	character used after % for encoding accents
>*	{\a}	lower case letters
<*	{\A}	upper case letters
</	{&a}	lower and upper case letters
	{\0}	normal decimal digits
>/	{&0}	normal and superscript digits
><><><%	*	zero up to any number of any characters

Character and string groups

– defining them in the editor:

>[vo]=aeiou	C:vo=aeiou	
	Z:vo=aeiou	
<[di]=|ei|	S:di=|ei|	

– defining them in parameters (col. 1–3):

>xy	>xy	defines a character group
>1z		defines a character group
<1z		defines a character group
<xy	<xy	defines a string group
>1s		defines a string group
<1s		defines a string group

– in the string defining a character group (col. 11 sqq.):

><	{–}	remove following characters from the group
<>	{+}	include following characters into the group

– referring to a character or string group

	[...]	local character group, as m[ae][iy]er
>[xy]	{Z:xy}	user defined character group xy
	{C:xy}	as {Z:xy}
<[xy]	{S:xy}	user defined string group xy
>1		user-defined character or string group
<1		user-defined character or string group

Frequency conditions in search strings

><n	{n}	exactly n elements
><n<>m	{n-m}	n to m elements, as few as possible
<>m><n	{n--m}	n to m elements, as many as possible

><0	{0}	0 or 1 element = {0–1}
<>0	{00}	1 up to infinitely many elements = {1–0}

Numerical values in search strings

	{#}	number with any value
>={n-n}	{#n}	number with the value n
<{n-n}	{!n}	number with a value not equal n
>={n-m}	{#n-m}	nuber with a value from n to m
<{n-m}	{!n-m}	number with a value smaller than n or greater than m

Pointers in search strings

>=nn	{+n=}	n-th element, counting from left a != A
<=nn	{-n=}	n-th element, counting from right a != A
>:nn	{+n:}	n-th element, counting from left a == A
<:nn	{-n:}	n-th element, counting from right a == A

Fields of elements in search strings

	{|}	Delimiter between fields of elements

Pointers in replacement strings

>=nn	{+n=}	n-th element, counting from left
<=nn	{-n=}	n-th element, counting from right
>=00	{+0=}	all elements of the core string
<=00	{–0=}	all elements of the core string
>=(n-m)	{+n-m=}	n-th to m-th element, conting from left
>=(n–0)	{+n–0=}	n-th to last element, counting from left
<=(n-m)	{-n-m=}	n-th to m-th element, counting from right
<=(0-m)	{–0-m=}	first to m-th element, counting from right
	{=n=}	all elements of the n-th element field
	{=0=}	all elements of the core string
	{=n-m=}	all elements of the n-th up to the m-th element field
>+nn etc	{...+}	... lower case –-> upper case
>-nn etc	{...–}	... upper case –-> lower case
>:nn etc	{...;}	... a, b, ..., $, ... –-> ä, ^b, ..., ^$, ...
>;nn etc	{...!}	... ä, ^b, ..., ^$, ... –-> a, b, ..., $, ...

Characters to be escaped:

?	\?	question mark
*	*	asterisk
[	\[	opening square bracket
]	\]	closing square bracket
{	\{	opening curly bracket
}	\}	closing curly bracket
>a >A	\a	lower case letter a
<a <A	\A	upper case letter A
\	\\	backslash
<<	<	opening angle bracket
>>	>	closing angle bracket

Other

<|	{[}	left margin
>|	{]}	right margin
><	{|}	in tables for sorting alphabets: switch to highest values

In the editor:

CTRL+K + first special character occuring within the brackets	facilitates the keying of {...} .
CTRL+K + blank	shows an overview over the conventions valid when #parameter,{} has been chosen.