tables

Tabellensatz mit Tustep

Dokumentation

Version 1.0 (21.1.2014)

Christian Moser

Inhalt

1. Allgemeine Hinweise

1.1 Zweck

Das Makro »tables« ermöglicht den Satz von Tabellen mit Tustep in einer Form ähnlich wie die Wiedergabe von HTML/CSS-Tabellen im Internet.

1.2 Voraussetzungen

1.2.1 – Das Makro erwartet die zu setzenden Tabellen als wohlgeformtes XML mit Bezeichnung des Tabellenelements und der Zeilen- und Zellenelemente. Eine Tabelle enthält genau 1 Tabellenelement, das 1 oder mehrere Zeilenelemente enthält. Die Zeilenelemente enthalten wiederum 1 oder mehrere Zellenelemente:

```
<tabellenelement>
    <zeilenelement>
        <zellenelement>...</zellenelement>
        <zellenelement>...</zellenelement>
        ...
    </zeilenelement>
    <zeilenelement>
        <zellenelement>...</zellenelement>
        <zellenelement>...</zellenelement>
        ...
    </zeilenelement>
    ...
</tabellenelement>
```

1.2.2 – Die Formatieranweisungen werden dem Skript in Form von Attributen zum Tabellenelement und zu den Zeilen- und Zellenelementen übermittelt. Diese Attribute müssen der syntaktischen Form von XML-Attributen entsprechen:

```
<element attributname="attributwert" attributname="attributwert" ...>
```

Die Reihenfolge der Attribute ist beliebig.

1.2.3 – Leere Zellenelemente können sowohl als milestones als auch als Elemente ohne Inhalt erscheinen:

```
<zellenelement/>
<zellenelement></zellenelement>
```

1.2.4 – Die Tags <tabellenelement>, <zeilenelement>, <zellenelement>, </zeilenelement> und </tabellenelement> (Beginn und Ende der Tabelle; Beginn und Ende der Zeilen, Beginn der Zellen) müssen in der Quelldatei einen neuen Tustep-Satz beginnen; vor ihnen dürfen höchstens Leerzeichen stehen.

Diese Aufteilung kann mit `#kopiere` (Parameter `ZA`) oder Tuscript (Funktion `SPLIT`) erreicht werden.

1.2.5 – Die Tabellen-, Zeilen- und Zellen-Tags bleiben im Resultat (ohne Attribute) stehen und müssen in den Satzparametern (gegebenenfalls durch nichts) aufgelöst werden. Der Inhalt der einzelnen Tabellenzellen wird unverändert in die Zieldatei übertragen, ebenfalls alle Tustep-Sätze der Quelldatei, die nicht Bestandteil einer Tabelle sind.

1.2.6 – Die Element- und Attributnamen sind frei wählbar. Sie müssen – zusammen mit weiteren Angaben – dem Makro im Vorspann als Variablenwerte mitgeteilt werden.

1.2.7 – Die Quell- und Zieldatei müssen im Vorspann des Skripts den Variablen `quelle` bzw. `ziel` zugewiesen werden. Ebenso die Breite des Satzspiegels (in Punkten) der Variable `bre`.

1.3 Anwendung

Zunächst müssen im Vorspann des Makros die Variablen mit den gewünschten Werten versehen werden. Danach kann die Datei mit `#tue,tables` ausgeführt werden.

Alternativ kann die Datei als Segment in eine Makrodatei gerettet werden (im Editor: `r,makrodatei,tables`) und danach mit `$tables` aufgerufen oder mit `INCLUDE "makrodatei" tables` in ein Makro inkludiert werden. Falls dieses Vorgehen gewählt wird, muss vor dem Abspeichern in die Makrodatei die erste (#makro) und letzte Zeile (*eof) entfernt werden.

1.4 Fehlermeldungen

Die wichtigsten Fehlerquellen (etwa fehlende obligate Angaben etc.) werden abgefangen und dem Benutzer als Fehlermeldungen im Ablaufprotokoll angezeigt. Es wurde jedoch nicht der Versuch unternommen, systematisch alle denkbaren Fehler zu antizipieren. Grundsätzlich gilt daher: Verwendung des Makros auf eigene Verantwortung.

1.5 PostScript-Ausgabe

Beim Tabellensatz werden jeweils Positionierungen mit `&!P(Vn)` vorgenommen. Falls eine solche vertikale Position auf der entsprechenden Seite/Spalte nicht (mehr) definiert ist (weil das Satzprogramm einen Seitenumbruch vorgenommen hat), können die betroffenen Tabellenzellen nicht mehr korrekt gesetzt werden. Das Satzprogramm gibt in einem solchen Fall eine Fehlermeldung in die Protokolldatei aus und das Resultat von `#*PSAUS` besteht dann nur aus einer einzelnen Seite mit einem entsprechenden Hinweis. Soll `#*PSAUS` dennoch die gesetzte Datei ausgeben, kann dies mit `seiten=0(999990)` verlangt werden (für einzelne Seiten z.B. `seiten=1–23'999990`). Je nach Konstellation und Joboptions kann dann aber der Fall eintreten, dass die PostScript-Datei nicht in eine PDF-Datei umgewandelt werden kann.

1.6 Zeilenzähler

Das Positionieren mit `&!P(Vn)` beim Tabellensatz hat leider die unerfreuliche Konsequenz, dass die Zeilenzählung nicht mehr mit den sichtbaren Zeilen im Resultat übereinstimmt. Wird also ein Zeilenzähler verwendet, so müssen die betroffenen Zeilen (und etwaige Apparateinträge mit Zeilenreferenzen) nach einem Satzdurchlauf mit der »richtigen« Zeilenzählung (als Marginalien) versehen werden.

1.7 Begleitendes Material zur Dokumentation

Die Datei »examples.pdf« enthält den Satz von verschiedenen Tabellenbeispielen. Die Quelldatei dazu findet sich in der Tustep-Datei »examples«, ein Satzprogramm dazu in der Datei »satzexamples«. Die Daten in »examples« können so zum Testen nach Belieben verändert und dann mit »satzexamples« ausgegeben werden.

1.8 Hinweise zur folgenden Beschreibung der Elemente und Attribute

Im Folgenden werden die verschiedenen Bestandteile und Optionen des Makros beschrieben. In `Courier` gesetzte Begriffe beziehen sich dabei auf die Variablennamen, die im Vorspann des Makros mit den gewünschten Werten versehen werden müssen.

Zu jedem Attribut findet sich zunächst eine ad-hoc Syntaxangabe, die keinem strengen Schema folgt und nur zum Ziel hat, dem Leser die weiter unten stehenden Ausführungen etwas besser verständlich zu machen. Es folgt – wo dies Sinn macht – die Angabe der entsprechenden XML Schema Restriction und zum Teil auch einer Schematron Assertion, die die geforderte Syntax und Regeln genau abbildet. Danach folgt eine Liste der im entsprechenden Kontext verwendeten Variablen und Erläuterungen zum jeweiligen Element/Attribut. Schließlich finden sich am Ende jeder Beschreibung einige Beispiele.

2. Tabellenelement und -attribute

`ta`	Name des Tabellenelements
`taprot`	Datei für Tabellenübersicht(en)

Für jede Tabelle in der Quelldatei wird eine detaillierte Spaltenübersicht erstellt. Ist die Variable `taprot` leer, werden diese Übersichten ins Ablaufprotokoll ausgegeben, ansonsten in eine Datei mit dem Namen, der `taprot` zugewiesen wird. Falls diese Datei noch nicht existiert oder nicht zum Schreiben angemeldet ist, wird sie kreiert bzw. angemeldet.

Das Tabellenelement hat insgesamt 1 obligatorisches und 6 optionale Attribute, die im Folgenden näher erläutert werden.

2.1 Attribut `tacol` (Spaltenangaben, Cell-Padding, Tabellenausrichtung)

Das Attribut `tacol` ist obligatorisch, alle anderen Attribute sind optional.

Syntax

`tacol`="1. Wert | `tacolpart` | 2. Wert | `tacolpart` | 3. Wert | `tacolpart` | ..."

`tacol`="`tacolpro/fix`|`tacolpart2`|`tacolcenter/right`|`tacolpart`|Zahl|`tacolpart2`|Zahl|`tacolpart`|Zahl|`tacolpart`|Zahl|..."

XML Schema Restriction

<xs:pattern value="(`tacolpro`|`tacolfix`)(`tacolpart2`(`tacolcenter`|`tacolright`))\{0,1\}(`tacolpart`\d+(`tacolpart2`\d+)\{0,1\})+"/>

`tacol`	Name des Attributs
`tacolpro`	Bezeichnung für prozentuale Spaltenangaben
`tacolfix`	Bezeichnung für fixe Spaltenangaben in Punkten
`tacolcenter`	Bezeichnung für Zentrierung der Tabelle
`tacolright`	Wert für Rechtsbündigkeit der Tabelle
`tacolpart`	Trenner zwischen den einzelnen Werten
`tacolpart2`	Trenner innerhalb der einzelnen Werte

1. Wert

Angabe, ob die nachfolgenden Spaltenangaben in Prozenten (`tacolpro`) oder in fixen Punkten (`tacolfix`) vorgenommen werden, optional gefolgt von einem Trenner (`tacolpart2`) und der Angabe, ob die Spalten innerhalb von `bre` zentriert (`tacolcenter`) oder rechtsbündig (`tacolright`) gesetzt werden sollen.

Hinweis Gewisse Kombinationen machen keinen Sinn:
- prozentuale Spaltenbreiten mit Tabelle rechtsbündig: Angabe zur Rechtsbündigkeit hat keinen Einfluss auf das Resultat
- prozentuale Spaltenbreiten mit Tabelle zentriert: Zentrierangabe hat keinen Einfluss auf das Resultat (außer Attribut `taind` [siehe unten] wird verwendet)

2. Wert

Der 2. Wert wird vom 1. Wert durch ein Trennzeichen (`tacolpart`) abgesetzt und enthält zwei Zahlen, getrennt durch `tacolpart2`. Die zwei Zahlen geben das Cell-Padding an, d.h. den Abstand des Zellentexts vom linken bzw. rechten Spaltenrand in Punkten. Ist nur eine Zahl (und kein Trenner) angegeben, wird der Wert sowohl für den linken, als auch für den rechten Rand eingesetzt. Dieser 2. Wert ist obligatorisch, d.h. falls kein Cell-Padding gewünscht wird, muss »0« angegeben werden.

3.–x. Wert
Nach dem 2. Wert folgt für jede Spalte einzeln (getrennt durch `tacolpart`) die Angabe der Spaltenbreite, entweder in Punkten (falls im 1. Wert `tacolpro` angegeben ist) oder aber in Prozenten (falls `tacolfix` angegeben). »`tacolpart2`Zahl« hinter einer solchen Spaltenbreitenangabe bewirkt, dass diese Breite für insgesamt x-Spalten hintereinander gilt. Es werden insgesamt so viele Angaben erwartet, wie die Tabelle Spalten enthält. Enthält eine Tabellenzeile nicht exakt soviele Spalten, wie mit dem Attribut `tacol` definiert, so bricht das Makro mit einer Fehlermeldung ab.

Hinweis: falls prozentuale Spaltenangaben gewählt werden, sollte die Summe der Prozentangaben 100 betragen. Ist dies nicht der Fall, so wird die Differenz der letzten Spalte zugeteilt. Ebenso werden etwaige Rundungsreste der letzten Spalte zugerechnet.

Hinweis `tacolpart` und `tacolpart2`: Die Trennzeichen müssen voneinander unterschiedlich sein und dürfen im gesamten Attributwert nur genau in dieser Funktion als Trennzeichen auftreten. Ferner darf keines dieser Trennzeichen eine Tilde sein.

Beispiele

Variablenwerte

`ta`	table
`tacol`	columns
`tacolpro`	p
`tacolfix`	f
`tacolcenter`	c
`tacolright`	r
`tacolpart`	–
`tacolpart2`	:

```
<table columns="f–5:8–30–50–50–30">
```
4 Spalten mit der fixen Breite von 30, 50, 50 und 30 Punkten; Abstand des Textes innerhalb der Zelle vom linken Spaltenrand = 5, vom rechten Spaltenrand = 8; Tabelle linksbündig

```
<table columns="f:r–8–30–50–50–30">
```
4 Spalten mit der fixen Breite von 30, 50, 50 und 30 Punkten; Abstand des Textes innerhalb der Zelle vom linken Spaltenrand = 8, vom rechten Spaltenrand = 8; Tabelle rechtsbündig

```
<table columns="p–0–25–25–20–30">
```
4 Spalten mit der Breite von 25, 25, 20 und 30 Prozenten der Satzspiegelbreite; kein Cell-Padding

```
<table columns="p–0–25:4">
```
4 Spalten mit der Breite von je 25 Prozent der Satzspiegelbreite; kein Cell-Padding

```
<table columns="f:r–8–30–50:2–30">
```
4 Spalten mit der fixen Breite von 30, 50, 50 und 30 Punkten; Abstand des Textes innerhalb der Zelle vom linken Spaltenrand = 8, vom rechten Spaltenrand = 8; Tabelle rechtsbündig

2.2 Attribut `taind` (Einzug der Tabelle links)

Syntax
`taind="Zahl"`

XML Schema Attributtyp
type="xs:nonNegativeInteger"

`taind`	Name des Attributs
`taindmult`	Multiplikator für Einzugsangaben

Mit dem Attribut `taind` kann ein linker Einzug der Tabelle festgelegt werden. Erwartet wird als Attributwert eine Zahl, die die Anzahl der Punkte umfasst. Diese Angabe wird durch `taindmult` modifiziert, um den linken Einzug der Tabelle in Punkten festzusetzen. Falls `taindmult` aus einer Zahl (ohne Vorzeichen) besteht, wird der Attributwert mit `taindmult` multipliziert; falls `taindmult` aus einer Zahl mit einem positiven oder negativen Vorzeichen besteht, wird `taindmult` zum Attributwert addiert bzw. vom Attributwert subtrahiert; falls `taindmult` die Form »Zahl+Zahl« oder »Zahl-Zahl« hat, wird der Attributwert mit der ersten Zahl multipliziert und die zweite Zahl zum Resultat addiert bzw. vom Resultat subtrahiert.

Hinweis Gewisse Kombinationen mit `tacol` machen keinen Sinn:
- fixe Spaltenbreiten und Tabelle rechtsbündig: `taind` wird ignoriert
- fixe Spaltenbreiten und Tabelle zentriert: `taind` wird ignoriert

Beispiele

Variablenwerte

`ta`	table
`tacol`	columns
`tacolpro`	p
`tacolfix`	f
`tacolpart`	–
`tacolpart2`	:
`taind`	einzug
`taindmult`	1

```
<table columns="f–5–30–30–30" einzug="20">
Ganze Tabelle wird links 20 Punkte eingezogen
```

```
<table columns="f:r–5–30–30–30" einzug="20">
Tabelle rechtsbündig; Einzug links wird ignoriert
```

```
<table columns="p–0–25–25–20–30" einzug="20">
Ganze Tabelle wird links 20 Punkte eingezogen
```

```
<table columns="p:c–0–25–25–20–30" einzug="20">
Tabelle zentriert, Einzug links und rechts je 20 Punkte
```

falls `taindmult` = 5

```
<table columns="f–5–30–30–30" einzug="2">
Ganze Tabelle wird links 10 Punkte eingezogen
```

falls `taindmult` = +3

```
<table columns="f–5–30–30–30" einzug="10">
Ganze Tabelle wird links 13 Punkte eingezogen
```

falls `taindmult` = –3

```
<table columns="f–5–30–30–30" einzug="20">
Ganze Tabelle wird links 17 Punkte eingezogen
```

falls `taindmult` = 1+3

```
<table columns="f–5–30–30–30" einzug="10">
Ganze Tabelle wird links 13 Punkte eingezogen
```

falls `taindmult` = 5–3

```
<table columns="f–5–30–30–30" einzug="4">
Ganze Tabelle wird links 17 Punkte eingezogen
```

2.3 Attribut `taren` (Tabelle mit Tag umschließen)

Syntax
`taren="Tustep Makro-Abkürzung"`

`taren`	Name des Attributs

Mit diesem Attribut kann die ganze Tabelle mit einem Tag umschlossen werden, das sodann in den Tustep-Satzparametern aufgelöst wird.

Hinweis Je nach Auflösung von `taren` in den Satzparametern kann es vorkommen, dass die Verwendung des Attributs `tavsp` die Wirkung von `taren` aufhebt. Dieselbe Funktion wie `taren` erfüllen aber auch auf Zeilen- bzw. Zellenebene die Attribute `trren` bzw. `tdren`.

Hinweis Je nach Auflösung in den Satzparametern muss bei der Verwendung von `taren` die Auflösung von `trren` und `tdren` mitbedacht werden.

Beispiele

Variablenwerte

`ta`	table
`tacol`	columns
`taren`	format

```
<table columns="..." format="spezial">
die Tabelle wird mit dem Tag <spezial> ... </spezial> umschlossen; Auflösung in Satzparametern z.B.:
MAC      <spezial>#F+
MAC      </spezial>#F-
→ ganze Tabelle wird fett gesetzt
```

2.4 Attribut `tajus` (**Blocksatz**)

Syntax
`tajus="tajusflush"`

XML Schema Restriction
<xs:enumeration value="`tajusflush`"/>

`tajus`	Name des Attributs
`tajusflush`	Bezeichnung für Blocksatz

Standardmäßig treibt das Makro den Tabellentext nicht aus. Falls Blocksatz gewünscht wird, kann dies mit dem Attribut `tajus` bewerkstelligt werden.

Beispiele

Variablenwerte

`ta`	table
`tacol`	columns
`tajus`	flush
`tajusflush`	yes

<table columns="...">
Tabellentext wird nicht ausgetrieben (Standardeinstellung)

<table columns="..." flush="yes">
Tabellentext wird ausgetrieben

2.5 Attribut `tavsp` (Vorschub davor)

Syntax

`tavsp`="Zeilennummer|`tavsppart2`|Zahl|`tavsppart`|Zeilennummer|`tavsppart2`|Zahl|`tavsppart`|..."

XML Schema Restriction

<xs:pattern value="(\d{1,3}`tavsppart2`\d{1,2}|\d{1,3}`tavsppart2`\d{1,2},5|\d{1,3}`tavsppart2`\d{1,2}`tavsppt`)(`tavsppart`(\d{1,3}`tavsppart2`\d{1,2}|\d{1,3}`tavsppart2`\d{1,2},5|\d{1,3}`tavsppart2`\d{1,2}`tavsppt`))*"/>

`tavsp`	Name des Attributs
`tavsppart`	Trennzeichen zwischen den einzelnen Werten
`tavsppart2`	Trennzeichen innerhalb der einzelnen Werte
`tavsppt`	Bezeichnung für Angabe in Punkten

Mit dem Attribut `tavsp` kann ein Vorschub vor einer oder vor mehreren Zeilen bestimmt werden. Erwartet wird die Angabe einer Zahl für die Zeilennummer und – getrennt davon durch `tavsppart2` – einer Zahl, die für den Umfang des Vorschubs steht. Steht unmittelbar hinter einer solchen Vorschubsangabe `tavsppt`, so wird die Zahl als Punktangabe interpretiert, sonst als Anzahl Leerzeilen. Halbe Leerzeilen können können durch ein anschließendes »,5« angegeben werden.

Hinweis `tavsppart` und `tavsppart2`: Die Trennzeichen müssen voneinander unterschiedlich sein und dürfen im gesamten Attributwert nur genau in dieser Funktion als Trennzeichen auftreten. Ferner darf keines dieser Trennzeichen eine Tilde sein.

Hinweis Je nach Konstellation kann die Verwendung von `tavsp` die Wirkung von `taren` aufheben.

Beispiele

Variablenwerte

`ta`	table
`tacol`	columns
`tavsp`	vorschub
`tavsppart`	–
`tavsppart2`	:
`tavsppt`	pt

```
<table columns="..." vorschub="1:1">
vor der ersten Zeile wird eine Leerzeile eingesetzt
```

```
<table columns="..." vorschub="2:2–3:0,5–6:8pt">
vor der 2. Zeile werden zwei Leerzeilen eingesetzt, vor der 3. Zeile eine halbe Leerzeile und vor der 6. Zeile ein Vorschub von 8 Punkten
```

2.6 Attribut `tabor` (Rahmen)

Syntax

`tabor`="1. Wert | `taborpart` | 2.-x. Wert"

`tabor`="Schriftgrad|`taborpart2`|Durchschuss|`taborpart2`|vertikale Positionierung|`taborpart`|`taborgr/grmin/grh`...|`taborpart2`|Linienstärke|`taborpart`|... "

XML Schema Restriction

<xs:pattern value="\d{1,3}`taborpart2`\d{1,3}`taborpart2`\d{1,3}(`taborpart`(`taborgr`|`taborgrmin`|`taborgrh`|`taborgrhmin`|`taborgrv`|`taborgrvmin`|`taborvl`|`taborvr`|`taborvlr`|`taborvb`|\d{1,2})`taborpart2`\d{1,2})*"/>

`tabor`	Name des Attributs
`taborpart`	Trennzeichen zwischen den einzelnen Werten
`taborpart2`	Trennzeichen innerhalb der einzelnen Werte
`taborgr`	Bezeichnung für Gitternetzlinien: alle vertikalen und horizontalen
`taborgrmin`	Bezeichnung für Gitternetzlinien: alle außer äußerer Rahmen
`taborgrh`	Bezeichnung für horizontale Gitternetzlinien: alle
`taborgrhmin`	Bezeichnung für horizontale Gitternetzlinien: alle außer oberste und unterste
`taborgrv`	Bezeichnung für vertikale Gitternetzlinien: alle
`taborgrvmin`	Bezeichnung für vertikale Gitternetzlinien: alle außer äußerste links und rechts
`taborlmst`	Wert der Standard-Strichstärke
`taborvl`	Bezeichnung für vertikale Rahmenlinie links
`taborvr`	Bezeichnung für vertikale Rahmenlinie rechts
`taborvlr`	Bezeichnung für vertikale Rahmenlinie links und rechts
`taborvb`	Bezeichnung für Spaltenrahmen

Das Attribut `tabor` dient dazu, die Tabelle mit Rahmenlinien zu versehen.

1. Wert
Erwartet wird – abgetrennt durch `taborpart2` – die Angabe von Schriftgrad, Durchschuss und vertikale Positionierung. Letztere Angabe ist eine Zahl, die für einen Prozentwert von 1–100 steht. Mit ihrer Hilfe wird der obere und untere Abstand der horizontalen Rahmenlinien von der Schriftgrundlinie bestimmt. Der Abstand nach oben setzt sich zusammen aus dem halben Durchschuss plus X Prozent der Gevierthöhe, der Abstand nach unten entsprechend aus dem halben Durchschuss plus (100 – X Prozent) der Gevierthöhe. Der optimale Wert variiert nach optischem Geschmack und nach Fonttyp, liegt aber normalerweise um 75–80%.

2.–x. Wert

Die folgenden Werte geben die Rahmenarten an. Zur Auswahl stehen zunächst die folgenden global definierten Optionen: `taborgr`, `taborgrmin`, `taborgrh`, `taborgrhmin`, `taborgrv`, `taborgrvmin`.

Der Angabe einer solchen Option kann nach `taborpart2` eine Zahl angegeben werden, die für die Strichstärke steht (Strichstärke = Zahl/8 Punkt). Fehlt diese Angabe, so wird der Wert von `taborlmst` eingesetzt.

Weiter ist es auch möglich, vertikale Linien für einzelne Spalten zu definieren. Dies geschieht durch die Angabe der Spaltennummer, unmittelbar gefolgt von der Art der gewünschten Rahmen/Linienart: `taborvl`, `taborvr`, `taborvlr`, `taborvb`.

Auch diesen Angaben kann optional die gewünschte Strichstärke angefügt werden.

Hinweis `taborpart` und `taborpart2`: Die Trennzeichen müssen voneinander unterschiedlich sein und dürfen im gesamten Attributwert nur genau in dieser Funktion als Trennzeichen auftreten. Ferner darf keines dieser Trennzeichen eine Tilde sein.

Beispiele

Variablenwerte	
`ta`	table
`tacol`	columns
`tabor`	rahmen
`taborpart`	–
`taborpart2`	:
`taborgr`	grid
`taborgrmin`	gridmin
`taborgrh`	horiz
`taborgrhmin`	horizmin
`taborgrv`	verti
`taborgrvmin`	vertimin
`taborlmst`	1
`taborvl`	li
`taborvr`	re
`taborvlr`	lire
`taborvb`	box

```
<table columns="..." rahmen="10:2:80-grid:2">
```
alle Gitternetzlinien, Strichstärke 2/8 Punkt

```
<table columns="..." rahmen="10:2:80-horiz:2-verti:2">
```
alle horizontalen und vertikalen Gitternetzlinien, Strichstärke 2/8 Punkt

```
<table columns="..." rahmen="10:2:80-horizmin–1lire:4–5re">
```
alle horizontalen Gitternetzlinien, außer oberste und unterste, Strichstärke 1/8 Punkt (Wert von `taborlmst`); dazu vertikale Linien links und rechts in der Spalte 1, Strichstärke 4/8 Punkt, und rechte Rahmenlinie in Spalte 5, Strichstärke 1/8 Punkt (Wert von `taborlmst`)

```
<table columns="..." rahmen="10:2:80-verti–1box–5box">
```
alle vertikalen Gitternetzlinien; voller Rahmen um Sp. 1 und 5, alles Strichstärke 1/8 Punkt

2.7 Attribut `tasha` (Hinterlegung mit Grau-/Farbwert)

Syntax

`tasha`="1. Wert | `tashapart` | 2. Wert"

`tasha`="Schriftgrad|`tashapart2`|Durchschuss|`tashapart2`|vertikale Positionierung|`tashapart`|`tashatab/hev/hod`|`taborpart2`|Farbbezeichnung"

XML Schema Restriction

<xs:pattern value="\d{1,3}`tashapart2`\d{1,3}`tashapart2`\d{1,3}(`tashapart`(`tashatab`|`tashahev`|`tashahod`)(`tashapart2`(`tashaval1`|`tashaval2`|...)))?"/>

`tasha`	Name des Attributs
`tashapart`	Trennzeichen zwischen den einzelnen Werten
`tashapart2`	Trennzeichen innerhalb der einzelnen Werte
`tashatab`	Bezeichnung für Hinterlegung der ganzen Tabelle
`tashahev`	Bezeichnung für Hinterlegung gerade Zeilen
`tashahod`	Bezeichnung für Hinterlegung ungerade Zeilen
`tashavals`	Definition der Grau-/Farbwerte
`tashasta`	Standard Grau-/Farbwert

Das Attribut `tasha` dient dazu, die ganze Tabelle oder die ungeraden bzw. geraden Zeilen farbig oder grau zu hinterlegen.

1. Wert
Erwartet wird eine dreigliedrige Angabe analog zum 1. Wert von `tabor`. Die Angabe kann fehlen, falls das Attribut `tabor` verwendet wird. Der entsprechende Wert von `tabor` wird in diesem Fall für `tasha` übernommen.

2. Wert
Hier kann entweder `tashatab`, `tashahev` oder `tashahod` angegeben werden, gefolgt von `tashapart2` und einem String, der einen Grau-/Farbwert bezeichnet.

Die Grau-/Farbbezeichnungen und ihre jeweiligen Werte werden in der Variable `tashavals` definiert und zwar in der Form:

> Bezeichnung:Farbwert/Bezeichnung:Farbwert/...
>
> *Beispiel:* hellgrau:80/dunkelgrau:20/hellblau:75,40,0,0/hellgrün:60,0,100,0

Bei Grauwerten genügt die Angabe einer einzelnen Zahl für den Grauwert (0 = schwarz; 100 = weiß). Bei Farbwerten sind insgesamt 4, durch Komma abgetrennte Zahlen für die Druckfarben Cyan, Magenta, Yellow und Black (CMYK) anzugeben.

Die Variable `tashasta` schließlich enthält den Standard Grau-/Farbwert, der eingesetzt wird, falls keine Angaben zur Farbe gemacht werden.

Beispiel: `tashasta` = "75,40,0,0"
→ falls im Attribut `tasha` keine Angabe zur Farbe gemacht wird, wird die Hinterlegung "hellblau" (= CMYK 75,40,0,0) vorgenommen

Hinweis `tashapart` und `tashapart2`: Die Trennzeichen müssen voneinander unterschiedlich sein und dürfen im gesamten Attributwert nur genau in dieser Funktion als Trennzeichen auftreten. Ferner darf keines dieser Trennzeichen eine Tilde sein.

Beispiele

Variablenwerte	
`ta`	table
`tacol`	columns
`tasha`	shadow
`tashapart`	–
`tashapart2`	:
`tashatab`	tab
`tashahev`	hev
`tashahod`	hod
`tashavals`	hellgrau:90/dunkelgrau:20/hellblau:75,40,0,0/hellgrün:60,0,100,0
`tashasta`	90

<table columns="..." shadow="10:2:80-tab:dunkelgrau">
ganze Tabelle »dunkelgrau«; (Grauwert 20) hinterlegt

<table columns="..." shadow="10:2:80-hod">
Zebratabelle: ungerade Zeilen mit »hellgrauer« Hinterlegung (Grauwert 90, übernommen aus `tashasta`)

<table columns="..." border="10:2:80–..." shadow="hev:hellgrün">
Zebratabelle: gerade Zeilen mit »hellgrüner« Hinterlegung (Wert 60,0,100,0); Angaben zu Schriftgrad, Durchschuss und vertikaler Positionierung werden `tabor` entnommen

3. Zeilenelement und -attribute

`tr` Name des Zeilenelements

Das Zeilenelement hat insgesamt 5 optionale Attribute.

3.1 Attribut `trren` (Zeile mit Tag umschließen)

Syntax

`trren`="Tustep Makro-Abkürzung"

`trren`	Name des Attributs

Mit diesem Attribut kann die Zeile mit einem Tag umschlossen werden, das sodann in den Tustep-Satzparametern aufgelöst wird.

Hinweis Je nach Auflösung in den Satzparametern muss bei der Verwendung von `trren` die Auflösung von `taren` und `tdren` mitbedacht werden.

Beispiel: `taren` wird in den Satzparametern mit #F+/#F- (Fettdruck) aufgelöst, `trren`, das in Zeile 3 steht, mit #/+/#/– (Kursivdruck). Als Resultat ergibt sich, dass alle Zeilen fett gesetzt werden, außer Zeile 3 kursiv (und nicht etwa fett & kursiv).

Beispiele

Variablenwerte

`tr`	zeile
`trren`	format

```
<zeile format="wichtig">
die Zeile wird mit dem Tag <wichtig> ... </wichtig> umschlossen; Auflösung in Satzparametern z.B.:
MAC     <wichtig>#F+
MAC     </wichtig>#F-
→ die Zeile wird fett gesetzt
```

3.2 Attribut `trjus` (Blocksatz)

Syntax
`trjus="trjusflush"`

XML Schema Restriction
<xs:enumeration value="trjusflush"/>

`trjus`	Name des Attributs
`trjusflush`	Bezeichnung für Blocksatz

Standardmäßig treibt das Makro den Tabellentext nicht aus. Falls Blocksatz gewünscht wird, kann dies für eine Zeile mit dem Attribut `trjus` bewerkstelligt werden.

Beispiele

Variablenwerte

`tr`	zeile
`trjus`	flush
`trjusflush`	yes

```
<zeile>
Text in der Zeile wird nicht ausgetrieben (Standardeinstellung)
```

```
<zeile flush="yes">
Text in der Zeile wird ausgetrieben
```

3.3 Attribut `trvsp` (Vorschub davor)

Syntax
`trvsp="Zahl"`

XML Schema Restriction
<xs:pattern value="\d{1,3}|\d{1,3},5|\d{1,3}trvsppt"/>

`trvsp`	Name des Attributs
`trvsppt`	Bezeichnung für Angabe in Punkten

Mit dem Attribut `trvsp` kann ein Vorschub vor der entsprechenden Zeile verlangt werden. Erwartet wird als Attributwert eine Zahl, die für den Umfang des Vorschubs steht. Steht unmittelbar hinter einer solchen Vorschubsangabe `trvsppt`, so wird die Zahl als Punktangabe interpretiert, sonst als Anzahl Leerzeilen. Halbe Leerzeilen können können durch ein anschließendes »,5« angegeben werden.

Hinweis Je nach Konstellation kann die Verwendung von `trvsp` die Wirkung von `taren` aufheben.

Beispiele

Variablenwerte

`tr`	zeile
`trvsp`	abstand
`trvsppt`	pt

```
<zeile abstand="2">
vor der Zeile werden 2 Leerzeilen eingesetzt
```

```
<zeile abstand="3,5">
vor der Zeile werden 3,5 Leerzeilen eingesetzt
```

```
<zeile abstand="9pt">
vor der Zeile werden 9 Punkt Vorschub eingesetzt
```

3.4 Attribut `trbor` (Rahmen)

Syntax

`trbor="trboru/o/b | trborpart2 | Zahl"`

XML Schema Restriction

<xs:pattern value="trborb(trborpart2\d{1,2}){0,1}|(trboro|trboru)(trborpart2\d{1,2}){0,1}(trborpart(trboro|trboru)(trborpart2\d{1,2}){0,1}){0,1}"/>

Schematron Assertion (context="tr/@trbor")

<sch:assert test="matches(substring-before(ancestor::ta[1]/@tabor, 'taborpart'), '\d{1,2}taborpart2\d{1,2}taborpart2{1,2}')">Das Attribut trbor verlangt die Angabe von Schriftgrad, Durchschuss und vertikale Positionierung im Attribut tabor des Elements ta.</sch:assert>

`trbor`	Name des Attributs
`trborpart`	Trennzeichen zwischen den einzelnen Werten
`trborpart2`	Trennzeichen innerhalb der einzelnen Werte
`trboru`	Rahmenlinie unten
`trboro`	Rahmenlinie oben
`trborb`	Rahmen um die Zeile
`trborlmst`	Standard-Strichstärke

Das Attribut `trbor` dient dazu, eine Zeile mit Rahmenlinien zu versehen.

Als Werte zur Auswahl stehen `trboru`, `trboro` und `trborb`, gefolgt von `trborpart2` und der Angabe der gewünschten Strichstärke als Zahl (Strichstärke = Zahl/8 Punkte). Wird keine Angabe zur Strichstärke gemacht, wird der Wert von `trborlmst` eingesetzt. `trboru` und `trboro` können gemeinsam verwendet werden, getrennt voneinander durch `trborpart`.

Die Verwendung von `trbor` setzt voraus, dass in `tabor` der 1. Wert (Schriftgrad, Durchschuss und vertikale Positionierung) angegeben ist.

Hinweis `trborpart` und `trborpart2`: Die Trennzeichen müssen voneinander unterschiedlich sein und dürfen im gesamten Attributwert nur genau in dieser Funktion als Trennzeichen auftreten. Ferner darf keines dieser Trennzeichen eine Tilde sein.

Hinweis `trbor` überdruckt (nicht: ersetzt) etwaige Angaben zu `tabor`, d.h. ein Unterschied wird nur dann sichtbar, falls die Strichstärke zu `trbor` größer ist als diejenige zu `tabor`.

Beispiele

Variablenwerte

`tr`	zeile
`trbor`	rahmen
`trborpart`	–
`trborpart2`	:
`trboru`	unten
`trboro`	oben
`trborb`	box
`trborlmst`	1

```
<zeile rahmen="oben:2">
```
Rahmenlinie oben, Strichstärke 2/8 Punkte

```
<zeile rahmen="box">
```
Rahmen um die Zeile, Strichstärke 1/8 Punkte (von `trborlmst` übernommen)

```
<zeile rahmen="unten:2-oben:8">
```
Rahmenlinie unten, Strichstärke 2/8 Punkte; Rahmenlinie oben, Strichstärke 8/8 Punkte

3.5 Attribut `trsha` (Hinterlegung mit Grau-/Farbwert)

Syntax

`trsha="Farbbezeichnung"`

XML Schema Restriction

<xs:enumeration value="tashavals1"/>
<xs:enumeration value="tashavals..."/>

Schematron Assertion (context="tr/@trsha")

<sch:assert test="matches(substring-before(ancestor::ta[1]/@tasha, 'tashapart'), '\d{1,2}tashapart2\d{1,2}tashapart2{1,2}') or matches(substring-before(ancestor::ta[1]/@tabor, 'taborpart'), '\d{1,2}taborpart2\d{1,2}taborpart2{1,2}')">Das Attribut trsha verlangt die Angabe von Schriftgrad, Durchschuss und vertikale Positionierung im Attribut tasha oder alternativ im Attribut tabor des Elements ta.</sch:assert>

`trsha` Name des Attributs

Das Attribut `trsha` dient dazu, eine Zeile farbig oder grau zu hinterlegen.

Als Attributwert wird eine Farbbezeichnung erwartet, die in der Variable `tashavals` definiert ist (siehe oben zu `tasha`). Die Verwendung des Attributs setzt voraus, dass im Tabellenattribut `tasha` der 1. Wert (Schriftgrad, Durchschuss und vertikale Positionierung) angegeben ist. Falls dieser fehlt, wird der entsprechende Wert aus dem Tabellenelement `tabor` übernommen. Fehlen beide, bricht das Makro mit einer Fehlermeldung ab.

Hinweis `trsha` ersetzt für die betreffenden Zeile etwaige Grau-/Farbwerte aus `tasha`.

Beispiele

Variablenwerte

`ta`	tabelle
`tacol`	spa
`tabor`	rahmen
`tasha`	shadow
`tr`	zeile
`trsha`	farbe
`tashavals`	hellgrau:90/dunkelgrau:20/hellblau:75,40,0,0/hellgrün:60,0,100,0

<tabelle spa="..." shadow="10:2:80">
<zeile farbe="hellblau">
Zeile wird »hellblau« hinterlegt (Wert hellblau = 75,40,0,0 aus `tashavals`); Dimensionen der Hinterlegung entnommen aus `tasha`

<tabelle spa="...">
<zeile farbe="hellblau">
Programmabbruch, weil weder 1. Wert von `tasha` noch von `tabor` vorhanden

<tabelle spa="..." rahmen="10:2:80">
<zeile farbe="hellblau">
Zeile wird »hellblau« hinterlegt (Wert hellblau = 75,40,0,0 aus `tashavals`); Dimensionen der Hinterlegung entnommen aus `tabor`

<tabelle spa="..." rahmen="10:2:80" shadow="10:2:80">
<zeile farbe="hellblau">
Zeile wird »hellblau« hinterlegt (Wert hellblau = 75,40,0,0 aus `tashavals`); Dimensionen der Hinterlegung entnommen aus `tasha`

4. Zellenelement und -attribute

`td` Name des Zellenelements

Schematron Assertion (context = "`td`")

<sch:let name="subs" value="subsequence((tokenize(ancestor::`ta`[1]/@`tacol`, '`tacol-part`')), 3)"/>
<sch:let name="erg" value="for $n in $subs return if (contains($n,'`tacolpart2`')) then $n else replace($n,'(\d+)','$1`tacolpart21`')"/>
<sch:let name="anz" value="sum(for $n in $erg return number(subsequence((tokenize($n, '`ta-colpart2`')), 2)))«/>
<sch:assert test="count(parent::`tr`/`td`) = $anz">Anzahl der Tabellenspalten entspricht nicht der Angabe im Attribut `tacol` des Elements `ta`.</sch:assert>

Das Zellenelement hat insgesamt 8 optionale Attribute.

4.1 Attribut `tdali` (Textausrichtung)

Syntax

```
tdali="tdalicenter/right"
```

XML Schema Restriction

```
<xs:enumeration value="tdalicenter"/>
<xs:enumeration value="tdaliright"/>
```

`tdali`	Name des Attributs
`tdalicenter`	Bezeichnung für Zentrierung
`tdaliright`	Bezeichnung für Rechtsbündigkeit

Das Attribut `tdali` dient dazu, den Text in einer Zelle auszurichten. Standardmäßig setzt das Makro den Zellentext linksbündig. Mit `tdalicenter` und `tdaliright` kann verlangt werden, dass der Zellentext zentriert innerhalb der Zelle bzw. rechtsbündig gesetzt wird.

Beispiele

Variablenwerte

`td`	td
`tdali`	align
`tdalicenter`	center
`tdaliright`	right

```
<td>
Zellentext linksbündig (Standardeinstellung)
```

```
<td align="center">
Zellentext zentriert
```

```
<td align="right">
Zellentext rechtsbündig
```

4.2 Attribut `tdval` (vertikale Textausrichtung)

Syntax

```
tdval="tdvalcenter/down"
```

XML Schema Restriction

```
<xs:enumeration value="tdvalcenter"/>
<xs:enumeration value="tdvaldown"/>
```

`tdval`	Name des Attributs
`tdvalcenter`	Bezeichnung für vertikale Zentrierung
`tdvaldown`	Bezeichnung für Verschiebung nach unten

Das Attribut `tdval` dient dazu, den Text in einer Zelle vertikal auszurichten.

Mit `tdvalcenter` wird die erste Zeile des betroffenen Zellentexts vertikal – relativ zur höchsten Zelle innerhalb der Zeile – zentriert, mit `tdvaldown` wird die erste Zeile des betroffenen Zellentexts auf die Höhe der unterste Zeile der höchsten Zelle positioniert.

Zu beachten ist, dass die Positionierung für die erste Zeile innerhalb der betroffenen Zelle vorgenommen wird, d.h. dass im Normalfall sinnvolle Ergebnisse nur dann erzielt werden, falls der Zelltext einzeilig ist.

Beispiele

Variablenwerte

`td`	td
`tdval`	valign
`tdvalcenter`	center
`tdvaldown`	down

```
<td>
Zellentextbeginn oben (Standard)
```

```
<td valign="center">
Zellentextbeginn vertikale Mitte der höchsten Zelle innerhalb der Tabellenzeile
```

```
<td valign="down">
Zellentextbeginn Höhe unterste Zeile der höchsten Zelle innerhalb der Tabellenzeile
```

4.3 Attribut `tdind` (Texteinzug)

Syntax
`tdind="Zahl|tdindpart|Zahl/tdindfzr"`

XML Schema Restriction
`<xs:pattern value="\d{1,3}|\d{1,3}tdindpart\d{1,3}|\d{1,3}tdindparttdindfzr"/>`

`tdind`	Name des Attributs
`tdindpart`	Trennzeichen
`tdindmult`	Multiplikator für Einzugsangaben
`tdindfzr`	Bezeichnung für Folgezeilen rechtsbündig

Das Attribut `tdind` dient dazu, den Zellentext mit einem Einzug zu versehen. Der standardmäßige Textbeginn wird mit der Angabe des Cell-Paddings in `tacol` festgelegt. `tdind` positioniert relativ zu diesem Textbeginn.

Erwartet wird die Angabe von zwei Zahlen – voneinander getrennt durch `tdindpart`. Die erste Zahl definiert den Erstzeileneinzug in Punkten, die zweite Zahl den Folgezeileneinzug. Ist nur eine Zahl angegeben, so wird der Wert für den Folgezeileneinzug auf »0« gesetzt. Anstelle der zweiten Zahl kann `tdindfzr` angegeben werden, was zur Folge hat, dass die Folgezeilen rechtsbündig gesetzt werden. Die angegebenen Zahlen werden zur Berechnung des Einzugs durch `tdindmult` modifiziert. Falls `tdindmult` aus einer Zahl (ohne Vorzeichen) besteht, wird der Attributwert mit `tdindmult` multipliziert; falls `tdindmult` aus einer Zahl mit einem positiven oder negativen Vorzeichen besteht, wird `tdindmult` zum Attributwert addiert bzw. vom Attributwert subtrahiert; falls `tdindmult` die Form »Zahl+Zahl« oder »Zahl-Zahl« hat, wird der Attributwert mit der ersten Zahl multipliziert und die zweite Zahl zum Resultat addiert bzw. vom Resultat subtrahiert.

Hinweis `tindpart`: Das Trennzeichen darf keine Tilde sein und darf im gesamten Attributwert nur genau in dieser Funktion als Trennzeichen auftreten.

Beispiele

Variablenwerte

`td`	td
`tdind`	einzug
`tdindpart`	:
`tdindmult`	1
`tdindfzr`	r

```
<td einzug="10:15">
Erstzeileneinzug 10 Punkte; Folgezeileneinzug 15 Punkte
```

```
<td einzug="10:0">
Erstzeileneinzug 10 Punkte; Folgezeileneinzug 0 Punkte

<td einzug="10">
Erstzeileneinzug 10 Punkte; Folgezeileneinzug 0 Punkte

<td einzug="10:10">
Erstzeileneinzug 10 Punkte; Folgezeileneinzug 10 Punkte

<td einzug="10:5">
Erstzeileneinzug 10 Punkte; Folgezeileneinzug 5 Punkte

<td einzug="10:r">
Erstzeileneinzug 10 Punkte; Folgezeileneinzug rechtsbündig

<td einzug="0:r">
Erstzeileneinzug 0 Punkte; Folgezeileneinzug rechtsbündig
```

4.4 Attribut `tdwid` (Zellentextbreite vergrößern)

Syntax
`tdwid="Zahl"`

XML Schema Restriction
`<xs:pattern value="\d{1,3}"/>`

`tdwid`	Name des Attributs

Mit diesem Attribut kann die Zellentextbreite vergrößert werden.

Der Zellentext beginnt standardmäßig an der Position Spaltenbeginn plus Cell-Padding links und endet bei Spaltenende minus Cell-Padding rechts. Falls `tdwid` angegeben wird, wird zur letztgenannten Position die angegebene Punktzahl dazugerechnet.

Zu beachten ist, dass `tdwid` keinen Einfluss auf die Spaltenbreite hat, d.h. dass die mit `tdwid` verbreiterte Zelle gegebenenfalls in die Folgezelle(n) hineinragt und diese für ein sinnvolles Ergebnis normalerweise leer sein müssen. Ebenso wird es im Normalfall nicht sinnvoll sein, vertikale Rahmenlinien zu zeichnen, falls `tdwid` Verwendung findet.

Beispiele

Variablenwerte	
`td`	td
`tdwid`	breite

```
<td breite="40">
der Zellentext wird rechts um 40 Punkt verbreitert
```

4.5 Attribut `tdren` (Zelle mit Tag umschließen)

Syntax
`tdren="Tustep Makro-Abkürzung"`

`tdren`	Name des Attributs

Mit diesem Attribut kann die Zelle mit einem Tag umschlossen werden, das sodann in den Tustep-Satzparametern aufgelöst wird.

Hinweis Je nach Auflösung in den Satzparametern muss bei der Verwendung von `tdren` die Auflösung von `taren` und `trren` mitbedacht werden.

Beispiele

Variablenwerte

`td`	zelle
`tdren`	format

```
<zelle format="wichtig">
die Zelle wird mit dem Tag <wichtig> ... </wichtig> umschlossen; Auflösung in Satzparametern z.B.:
MAC      <wichtig>#F+
MAC      </wichtig>#F-
→ die Zelle wird fett gesetzt
```

4.6 Attribut `tdjus` (Blocksatz)

Syntax
`tdjus="tdjusflush"`

XML Schema Restriction
<xs:enumeration value="tdjusflush"/>

`tdjus`	Name des Attributs
`tdjusflush`	Bezeichnung für Blocksatz

Standardmäßig treibt das Makro den Tabellentext nicht aus. Falls Blocksatz gewünscht wird, kann dies für eine Zelle mit dem Attribut `tdjus` bewerkstelligt werden.

Beispiele

Variablenwerte

`td`	td
`tdjus`	flush
`tdjusflush`	yes

```
<td>
Text in der Zelle wird nicht ausgetrieben (Standardeinstellung)
```

```
<td flush="yes">
Text in der Zelle wird ausgetrieben
```

4.7 Attribut tdbor (Rahmen)

Syntax

tdbor="tdborl/o/r/b|tdborpart2|Zahl"

XML Schema Restriction

<xs:pattern value="(tdborl|tdboro|tdborr|tdboru|tdborb)(tdborpart2\d{1,2}){0,1}(tdborpart(tdborl|tdboro|tdborr|tdboru|tdborb)(tdborpart2\d{1,2}){0,1}){0,3}"/>

Schematron Assertion (context ="td/@tdbor")

<sch:assert test="matches(substring-before(ancestor::ta[1]/@tabor, 'taborpart'), '\d{1,2}taborpart2\d{1,2}taborpart2{1,2}')">Das Attribut tdbor verlangt die Angabe von Schriftgrad, Durchschuss und vertikale Positionierung im Attribut tabor des Elements ta.</sch:assert>

tdbor	Name des Attributs
tdborpart	Trennzeichen zwischen den einzelnen Werten
tdborpart2	Trennzeichen innerhalb der einzelnen Werte
tdborl	Bezeichnung für Rahmenlinie links
tdboro	Bezeichnung für Rahmenlinie oben
tdborr	Bezeichnung für Rahmenlinie rechts
tdboru	Bezeichnung für Rahmenlinie unten
tdborb	Bezeichnung für Rahmen um Zelle
tdborlmst	Standard-Strichstärke

Das Attribut tdbor dient dazu, eine Zelle mit Rahmenlinien zu versehen.

Als Werte zur Auswahl stehen tdborl, tdboro, tdborr, tdboru und tdborb, gefolgt von tdborpart2 und der Angabe der gewünschten Strichstärke als Zahl (Strichstärke = Zahl/8 Punkte). Wird keine Angabe zur Strichstärke gemacht, wird der Wert von tdborlmst eingesetzt. Es können mehrere Angaben gemacht werden, voneinander getrennt durch tdborpart.

Die Verwendung von tdbor setzt voraus, dass in tabor der 1. Wert (Schriftgrad, Durchschuss und vertikale Positionierung) angegeben ist.

Hinweis tdbor überdruckt (nicht: ersetzt) etwaige Angaben zu tabor und tdbor, d.h. ein Unterschied wird nur dann sichtbar, falls die Strichstärke zu tdbor größer ist als diejenige zu tabor bzw. tdbor.

Hinweis tdborpart und tdborpart2: Die Trennzeichen müssen voneinander unterschiedlich sein und dürfen im gesamten Attributwert nur genau in dieser Funktion als Trennzeichen auftreten. Ferner darf keines dieser Trennzeichen eine Tilde sein.

Beispiele

Variablenwerte	
`td`	td
`tdbor`	rahmen
`tdborpart`	–
`tdborpart2`	:
`tdborl`	li
`tdboro`	ob
`tdborr`	re
`tdboru`	un
`tdborb`	box
`tdborlmst`	1

<td rahmen="ob:4">
Rahmenlinie oben, Strichstärke 4/8 Punkte

<td rahmen="re">
Rahmenlinie rechts, Strichstärke 1/8 Punkte (von `tdborlmst` übernommen)

<td rahmen="li:8-ob:2-re">
Rahmenlinie links, Strichstärke 8/8 Punkte; Rahmenlinie oben, Strichstärke 2/8 Punkte; Rahmenlinie rechts, Strichstärke 1/8 Punkte (von `tdborlmst` übernommen)

<td rahmen="box:4">
Rahmen um Zelle, Strichstärke 4/8 Punkte

4.8 Attribut `tdsha` (Hinterlegung mit Grau-/Farbwert)

Syntax

`tdsha="Farbbezeichnung"`

XML Schema Restriction

<xs:enumeration value="tashavals1"/>
<xs:enumeration value="tashavals..."/>

Schematron Assertion (context = "td/@tdsha")

<sch:assert test="matches(substring-before(ancestor::ta[1]/@tasha, 'tashapart'), '\d{1,2}tashapart2\d{1,2}tashapart2{1,2}') or matches(substring-before(ancestor::ta[1]/@tabor, 'taborpart'), '\d{1,2}taborpart2\d{1,2}taborpart2{1,2}')">Das Attribut tdsha verlangt die Angabe von Schriftgrad, Durchschuss und vertikale Positionierung im Attribut tasha oder alternativ im Attribut tabor des Elements ta.</sch:assert>

`tdsha`	Name des Attributs

Das Attribut `tdsha` dient dazu, eine Zelle farbig oder grau zu hinterlegen.

Als Attributwert wird eine Farbbezeichnung erwartet, die in der Variable `tashavals` definiert ist (siehe oben zu `tasha`). Die Verwendung des Attributs setzt voraus, dass im Tabellenattribut `tasha` der 1. Wert (Schriftgrad, Durchschuss und vertikale Positionierung) angegeben ist. Falls dieser fehlt, wird der entsprechende Wert aus dem Tabellenelement `tabor` übernommen. Fehlen beide, bricht das Makro mit einer Fehlermeldung ab.

Hinweis Falls mit `tasha` oder `trsha` bereits ein Grau-/Farbwert für die Zeile, in der sich das Element mit `tdsha` befindet, definiert ist, wird `tdsha` ignoriert und ein entsprechender Hinweis ins Ablaufprotokoll ausgegeben.

Beispiele

Variablenwerte

`ta`	tabelle
`tacol`	spa
`tabor`	rahmen
`tasha`	shadow
`tashatab`	tab
`tashahev`	hev
`tashavals`	hellgrau:90/dunkelgrau:20/hellblau:75,40,0,0/hellgrün:60,0,100,0
`tr`	zeile
`trsha`	farbe
`td`	zelle
`tdsha`	farbe

```
<tabelle spa="..." shadow="10:2:80">
<zeile>
<zelle farbe="hellblau">
```
Zelle wird »hellblau« hinterlegt (Wert hellblau = 75,40,0,0 aus `tashavals`); Dimensionen der Hinterlegung entnommen aus `tasha`

```
<tabelle spa="...">
<zeile>
<zeile farbe="hellblau">
```
Programmabbruch, weil weder 1. Wert von `tasha` noch von `tabor` vorhanden

```
<tabelle spa="..." rahmen="10:2:80">
<zeile>
<zelle farbe="hellblau">
```
Zeile wird »hellblau« hinterlegt (Wert hellblau = 75,40,0,0 aus `tashavals`); Dimensionen der Hinterlegung entnommen aus `tabor`

```
<tabelle spa="..." rahmen="10:2:80" shadow="10:2:80">
<zeile>
<zelle farbe="hellblau">
```
Zeile wird »hellblau« hinterlegt (Wert hellblau = 75,40,0,0 aus `tashavals`); Dimensionen der Hinterlegung entnommen aus `tasha`

```
<tabelle spa="..." shadow="10:2:80-tab:hellgrün">
<zeile>
<zelle farbe="hellblau">
```
Zelle wird »hellgrün« hinterlegt (Wert von `tasha`), Farbangabe in `tdsha` wird ignoriert

```
<tabelle spa="..." shadow="10:2:80-hev:hellgrün">
<zeile>
<zelle farbe="hellblau">
```
Zelle wird »hellblau« hinterlegt, falls die Zelle sich in einer ungeraden Zeile befindet; Zelle wird »hellgrün« hinterlegt, falls die Zelle sich in einer geraden Zeile befindet (wegen `tashahev` in `tasha`)

```
<tabelle spa="..." shadow="10:2:80-tab:hellgrün">
<zeile farbe="dunkelgrau">
<zelle farbe="hellblau">
```
Zelle wird »dunkelgrau« hinterlegt (wegen `trsha`), Farbangabe in `tdsha` wird ignoriert