ܣܪܛܘ *Serṭo* – a font for Syriac (Aramaic)

Johannes Heinecke

Version 1.0, 30th March 2013

Contents

1 Introduction **2**

2 The alphabets **2**
- 2.1 Consonants . . . 2
- 2.2 Vowels . . . 4
- 2.3 Transliteration and long vowels . . . 6
- 2.4 Punctuation and paragraph marks . . . 7
- 2.5 Unicode . . . 7
- 2.6 The encoding . . . 7

3 The preprocessor `serto.py` **8**
- 3.1 Using the preprocessor . . . 8

4 The format of the `*.font` files **11**

5 Two examples **11**

6 Things still missing **13**

7 Installation **14**
- 7.1 Using Metafont sources . . . 14
- 7.2 Using vector fonts . . . 14
- 7.3 Other files . . . 15

8 License **15**

9 Changelog **15**

1 Introduction

Serṭo is a form of the alphabet used for Aramaic (a western semitic language) which has been spoken in the Near East since at least 1100 BC. More precisely, Serṭois used for Syriac which is the variant of Aramaic spoken since the second century AD.[1]

Syriac used other alphabets as well, notably *Eṣtrangelo*, which is not (yet) contained in this package. Since Serṭo is as the Arabic alphabet a syllabic script, vowels are marked by diacritic marks above (or under) the consonantic letters. Modern forms of Aramaic still use either Serṭo, the Chaldean alphabet or Eṣtrangelo.

Since Syriac split up in two main dialects in the fifth century AD, two differing systems of vowel-marking were established: Whereas the western dialect (Edessean) used Greek letters as vowel symbols (these are the only ones provided by this package for the time being), the eastern dialects uses dots to indicate the vowels (Chaldean vowels).

This package enables you to typeset words or paragraphs in Serṭo using a preprocessor which chooses the correct letter form depending on context. In order to typeset paragraphs the use of a recent version of `pdflatex` is needed, which can handle the right-to-left typesetting. For older version of LaTeX, the preprocessor must be used with the option `-o` (see section 3).

This package also included an adapted version of a Chaldean font (thanks to Tony Khoshaba, who put this font to the Web).

2 The alphabets

Every letter in Serṭo (and some letters in the Chaldean alphabet) has several forms, depending on its position in the word: An initial, medial or final form. Since some letters do not connect to the following letter, there are isolated forms as well (i.e. a letter which is not connected to the right nor the left). The coding column in the following table refers to the preprocessor described below (section 3). If you do not want to use the preprocessor, please refer to the encoding table in section 2.6.

2.1 Consonants

[1] See Ungnad 1932, Brockelmann 1960, Costaz 1986 or Nöldeke 1986f (English translation Nöldeke 2001) for further information on Syriac.

serṭo forms				Chaldean	name	translit.	coding
isolated	final	medial	initial				
ܐ	ܐ			ܐ	ʾālapā	ʾ	'
ܒ	ܒ	ܒ	ܒ	ܒ	bēṯ	*b*[2]	b
				ܒ̰[3]		*v*	v
ܓ	ܓ	ܓ	ܓ	ܓ	gāmal	*g*	g
				ܓ̣		*ġ*	g
				ܓ̰		*j*	j
ܕ	ܕ			ܕ	dālaṯ	*d*	d
ܗ	ܗ			ܗ	hē	*h*	h
ܘ	ܘ			ܘ	waw	*w*	w
ܙ	ܙ			ܙ	zayn	*z*	z
ܚ	ܚ	ܚ	ܚ	ܚ	ḥēṯ	*ḥ*	.h
ܛ	ܛ	ܛ	ܛ	ܛ	ṭēṯ	*ṭ*	.t
ܝ	ܝ	ܝ	ܝ	ܝ	yāḏ	*y*	y
ܟ	ܟ	ܟ	ܟ	ܟ	kāp	*k*	k
				ܟ̣		*ḫ*	_k
				ܟ̰		*č*	^k
ܠ	ܠ	ܠ	ܠ	ܠ	lāmaḏ	*l*	l
ܡ	ܡ	ܡ	ܡ	ܡ	mīm	*m*	m
ܢ	ܢ	ܢ	ܢ	ܢ	nūn	*n*	n
ܣ	ܣ	ܣ	ܣ	ܣ	semkaṯ	*s*	s

serṭo forms				Chaldean	name	translit.	coding
isolated	final	medial	initial				
ܐ	ܐ	ܐ	ܐ	ܐ	ʿ*ē*	ʿ	`
ܦ	ܦ	ܦ	ܦ	ܦ �p	*pē*	*p* *f*	p f
ܨ	ܨ			ܨ	*ṣādē*	*ṣ*	.s
ܩ	ܩ	ܩ	ܩ	ܩ	*qāp*	*q*	q
ܪ	ܪ	ܪ	ܪ	ܪ ܕ	*rīš*	*r*	r R
ܫ	ܫ	ܫ	ܫ	ܫ	*šīn*	*š*	^s
ܠ	ܠ			ܬ ܬ	*taw*	*t* *t*	t _t

2.2 Vowels

The package allows to typeset the greek vowels or Chaldean vowels symbols. To have the vowel symbol written in inversed form under the consonant, user upper case input.

> **ATTENTION:** I had to change the coding of *zqāpā* (ܼ) from o to =a in order to accomodate the new letter ܾ (coding o). In texts encoded up to version 0.4 o has to be replaced by =a Sorry for any inconveniances.

[2]The *beḡaḏkepaṯ* are not yet always processed. In general, the doubling of the consonant creates a ܒ݂ܓ݂ܼ *quššāyā* in the syriac text and does not change the transliteration. On the other hand, a consonant followed by + will receive a ܕ݂ܗ݂ܼ *rūkkāḥā* and in the transliteration *bgdkft* will appear as *b̲ḡḏ̲ḥp̲ṯ̲*.

[3]Modern Aramaic dialects using the Chaldean alphabet have diacritic symbols (dots and tildes) which can be typeset directly.

Greek	Chaldean	name	transliteration	coding
◌	◌	ܦܬܳܚܳܐ *pṯāḥā*	*a*	a
◌	◌	ܪܒܺܝܨܳܐ *rḇāṣā*	*e*	e
◌	◌	ܚܒܺܝܨܳܐ *ḥḇāṣā*	*i*	i
◌	◌	ܙܩܳܦܳܐ *zqāpā*	*ā*	=a
◌	◌	ܥܨܺܝܨܳܐ *ʿṣāṣā*	*u*	u
◌	◌		*ā*	o
◌		ܣܝܰܡܹܐ *syāmē*		P

Greek	coding
◌	A
◌	E
◌	I
◌	=A
◌	U

Note: The *ḥḇāṣā* and *ʿṣāṣā* of the Eastern or Chaldean vowels do in general occur together with a *mater lectionis*: ◌ܒ, ◌ܦ (or ◌ܒ, ◌ܦ)

The ܣܝܰܡܹܐ *syāmē* is processed as a vowel sign, even if it is not so from a linguistic point of view. Its coding `<S>P</S>` is chose because of its plural signification. If you do not want it over a letter, put it over a word stretch: The Aramaic at the beginning of this paragraph has been typset in the following: `<S>sy=a--Pme'</S>`.

To avoid that the simple preprocessor does not mess about with vowels in ligatures (notably *Lāmaḏ-ālaf* and *Ālaf-lāmaḏ* the vowel must be set after both consonants of the ligature:

`<S>l=a'</S>` yields (incorrect) ◌ܠ *lā* but `<S>l'=a</S>` yields the vocalized ligature ◌ܠ *lā*

`<S>'Al=Ah=a'</S>` yields (incorrect) ◌ܐ◌ܠ◌ܗ *ʾalahā* but `<S>'la=Ah=a'</S>` yields the vocalized ligature ◌ܐ◌ܠ◌ܗ *ʾalaahā*

The Chaldean letters do not have this ligature. Instead, a *tawālā* ligature is provided: ◌ܣ◌ܪ◌ܝ◌ܬ◌ܐ *siprāyūta*

The default vowels are the greek-based vowels. In order to get Chaldean vowels, it suffices to add : in front of the vowel in coding. Thus you can set the most famous Aramaic phrase in all Syriac alphabets in either vowel system:

```
<S>eliy eliy lm=an=a' s=ab=akt=aniy</S>
```
ܐܟ ܐܟ ܠܡܢ ܫܒܩܬܢܝ

```
<S>:el:iy :el:iy lm:=an:=a' s:=ab:=akt:=an:iy</S>
```
ܐܠܝ ܐܠܝ ܠܡܢ ܫܒܩܬܢܝ

```
<C>eliy eliy lm=an=a' s=ab=akt=aniy</C>
```
ܐܟ ܐܟ ܠܡܢ ܫܒܩܬܢܝ

```
<C>:el:iy :el:iy lm:=an:=a' s:=ab:=akt:=an:iy</C>
```
ܐܠܝ ܐܠܝ ܠܡܢ ܫܒܩܬܢܝ

2.3 Transliteration and long vowels

As mention in section 3 below, the preprocessor can produce a transliteration as well. The transliteration can be defined in the table used by the preprocessor `serto.font` and `assyr.font`. The current definition uses the transcription as shown in the tables in sections 2.1 and 2.2, with the exception of long vowels. In words using a vowel symbol together with a *mater lectionis*, the transliteration shows the transcription instead, for instance ܚܝܬ transliterates as *ḥēṯ* and not *ḥeyṯ* and ܩܵܝܹܡ *qāyēm*:

Greek	Chaldean	transliteration	coding
◌ܵ	◌ܰܐ	*ā*	`=a'`
◌ܹ	◌ܶܝ	*ē*	`ey`
◌ܶ	◌ܶ	*ē*	`e'`
◌ܺ	◌ܺ	*ī*	`iy`
◌ܽ	◌ܽ	*ū*	`uw`

2.4 Punctuation and paragraph marks

form	coding
܁	.
܃	:
※	.X.
⁘	.:.

form	coding
⁝	: :
÷	:–
≈	.~.
⁒	./.

2.5 Unicode

This package has a limited Unicode support in that texts encoded in UTF8 can be directly typeset. In order to activate the UTF8 interpretation, either use UTF8 for your whole document by declaring `\usepackage[utf8]{inputenc}` in the preamble of you document, or just put `%\usepackage[utf8]{inputenc}` somewhere at the beginning of your document.

2.6 The encoding

The following table shows the internal encoding of the defined letters of Serṭo and the Chaldean variant.

	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*A*	*B*	*C*	*D*	*E*	*F*
"0n		◌	◌	◌	◌	◌	◌	◌	◌				◌	◌		
"1n		◌	◌	◌	◌	◌		◌			◌	◌				
"2n		※	⁘	⁝	÷	≈	⁒							◌	◌	
"3n											:					
"4n		◌	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌
"5n	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌					–
"6n										◌	◌	◌	◌	◌	◌	◌
"7n	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌					
"8n		◌	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌
"9n	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌
"An	◌											◌	◌	◌		

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
"0n		◌	◌	◌	◌	◌	◌						◌	◌		
"1n		◌	◌	◌	◌	◌					◌	◌				
"2n					◌		◌				◌			◌		
"3n													◌	◌	◌	
"4n	◌	◌	◌		◌	◌		◌	◌	◌	◌	◌	◌	◌	◌	◌
"5n	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌	◌					
"6n		◌	◌		◌	◌		◌	◌	◌	◌	◌		◌	◌	◌
"7n	◌	◌	◌	◌	◌	◌	◌		◌	◌	◌		◌			
"8n																
"9n																
"An																

3 The preprocessor `serto.py`

Typesetting of texts is still not yet possible with standard LaTeX, since the right-to-left typesetting (as for instance as in ArabTeX) has not yet been implemented. For the time being I propose a preprocessor (written in PYTHON[4]) and `pdflatex`.

I'm well aware that `serto.py` is not part of the most beautiful pieces of software code, on the contrary, it's rather spaghetti code. Many things could have been in a more intelligent way, but it works, which is the most important thing. If you find the time to improve it please share your changes with me!

3.1 Using the preprocessor

The preprocessor is called with the LaTeX-file as argument:

```
serto.py [-o] ppfilename.tex > filename.tex
```

The resulting LaTeX-file can be LaTeXed as usually. Please make sure to have the `\usepackage{serto}` included in your preamble.

The option `-o` is necessary if you use an older version of LaTeX which is not capable to typeset texts from the right to the left (`TeX--XeT`extension). The `-o` -option tells the preprocessor to inverse the letters on its own. In order to

[4]Every version from 2.4 onwards up to version 2.7 should do; Python 3.* however does not work PYTHON is an interpreted object oriented programming language available for many operating systems (http://www.python.org).

typeset whole paragraphs `pdflatex` is the better solution. Usually it comes with every modern TEX-distribution. At least on Ubuntu 12.04 (texlive package), the standardly installed `pdflatex` behaves correctly.

The preprocessor recognizes two types of commands. Within a single line you can put Syriac words between `<S>` and `</S>`: For example `<S>ser.t=a'</S>` becomes ܣܪܛܐ.

`<ST>` and `</ST>` generate the enclosed part in Serṭo and generates a transliteration as well (`<ST>mdiyt=a'</ST>` becomes ܡܕܝܬܐ *mdīta*ʾ "city"), whereas `<T>` and `</T>` can be used for parts only need in transliterated form (`<T>ser.t=a'</T>` becomes *serṭā*). Since in transliteration a "neutral vowel" is needed, which does not appear in Serṭo, the code `@` can be used: `<ST>^s@m=a'</ST>` produces ܫܡܥ *šᵉmāʿ*

For multiple lines, start a block using `<SERTO>`[5] in a line on its own. This block can be closed by `</SERTO>`. If you need transliterated Syriac, use `</TRANS>` and `</TRANS>`. The commands `<SERTO>`/`</SERTO>` and `</TRANS>`/`</TRANS>` do not work properly with the `-o` option of the preprocessor and and older LATEX. If you add TEX-commands in these blocks, a right-to-left typesetting version of LATEX is obligatory.

For the time being the preprocessor tries to set the hard sign ܦܘܼܓ *quššāyā* on top of a consonant if the consonant is doubled in the input:

`<S>q.tl</S>` yields ܩܠ but `<S>q.t.tl</S>` yields ܩܬܠ

In cases where you need a ܦܘܼܓ *quššāyā* without wanting to double the consonant, a `*` can be used after the letter to typeset a dot above a letter: `<S>h*=anon</S>` produces ܗܢܢ *hānon* and `<S>^sl=amk+on</S>` results in ܫܠܡܟܘܢ *šlāmḵon*

To avoid a *quššāyā* (when you need to adjacent identical consontants, either use a vowel on the first, use the stretching symbol:

`<S>maml'e</S>` yields ܡܲܡܠܶܐ *maml*ʾ*e*
`<S>m^ml'</S>` yields ܡܡܠ *mml*ʾ
`<C>maml'e</C>` yields ܡ݂ܲܡ݂ܠ݂ܶܐ *maml*ʾ*e*
`<C>m^ml'</C>` yields ܡ݂ܡ݂ܠ݂ *mml*ʾ

[5] Using `<SERTO>` or `<TRANS>` implies the using of the `-e` option of the preprocessor `serto.py`. If you do not use the `-e` option the resulting LATEX-file is likely to be syntactically incorrect.

To get the soft sign ܗ *rūkkāḫā* a + must *follow* the letter: `<S>'ab+d=a'</S>` yields ܐܲܒ̰ܕ

An *ālaf* is autmatically prefixed before an initial vowel:

`<S>etqa.tel</S>` and `<S>'etqa.tel</S>` both yield ܐܵܬ̰ܩ̰ܠ

Sometimes the letter *rīš* is written with two points. To achieve this, use `R` instead of `r` in the input:

`<S>^sapiyRe'</S>` yields ܣ̰ܦ̰ܝ̰ܪ̰ܥ̈

Silent consonants have a bar ̱ *mḇaṭlānā* (*linea occultans*) under the line which is produced by = just before the consonant (attention `=a`, however, yields ܐ̰):

`<S>'an=tt</S>` yields ܐ̰ܢ̰

In order to have the *linea occultans* on top of the letter, use ==:

`<S>h==wiyt</S>` yields ܗ̰ܘ̰

There is no automatic stretching yet, but the `--` can be used to insert a "manual stretch":

`<S>napiyqt=a'</S>` becomes ܢ̰ܦ̰ܝ̰ܩ̰ܬ̰ but `<S>na--piyq--t=a'</S>` is printed as ܢ̰ܦ̰ܝ̰ܩ̰ܬ̰

This works also for the Chaldean letters `<C>n:ap:iyqt:=a'</C>` becomes ܢ̰ܦ̰ܝ̰ܩ̰ܬ̰ but `<C>n:a--p:iyq--t:=a'</C>` is printed as ܢ̰ܦ̰ܝ̰ܩ̰ܬ̰

If you do not use the preprocessor, you can activate Serṭo by the command `\serto`. In this case you have to choose the correct letters yourself, and use the commands `\upperserto{`*vowelnumber*`}{`*letter*`}` or `\lowerserto{`*vowelnumber*`}{`*letter*`}` to set vowels. Please see the encoding table in section 2.6 for the correct vowel numbers.

In order to get bold letters, you can use `\sertob` with or without preprocessor (see section 3 for more information on the preprocessor.

4 The format of the `*.font` files

These files are necessary to tell the preprocessor where (in the font) a certain letter is found, and whether it has different forms. The format is straight forward, with, however, a few idiosynchrasies. In general there are two sections, the first (starting with a line `#FONT`) indicates which letter has which form in which position and a second (`#TRANS`) to define the transcription.

The first part consists of lines like the following
`b beth 66+124 66 66 66+124 1`
which reads, coding `b` is for the letter *beth*, its isolated form is character 66 followed by character 124, its initial and medial form is character 66, it's final form is character 66 followed by character 124 and the next letter (if any) must take it's medial form.

However, if one of the four last columns has a value of -1, no form is provided in the font. Values from 0 to 15 are reserved for accents/vowels above the line, values from 16 to 31 are reserved for vowels under the line.

The lines

```
 ~  blank   32 32 32 32 0
 Q  shadda  6 6 6 6 2
 -- stretch 45 45 45 45 1
```

must not be deleted.

The coding for digits (starting with 0 up to 9 in the first column cannot have the + in the position definition

The transcription definition defines for each coding symbol (defined in the font-section) a valid TEX-string to be used if transcription is needed, e.g.

```
.t \d{t}
```

i.e. the coding `.t` will be representend by $\underset{\cdot}{t}$ in transcribed portions.

5 Two examples

Some of the following can be found in `example.ptex` which comes with this package.

The following input was used to generate the output below:

```
\documentclass[12pt,a4paper]{article}
usepackage{serto}

\begin{document}

<S>men qadiy^se' ho' tetqada^s</S>

<ST>kmo' dat+basb@suwn pagdo'</ST>

\end{document}
```

kmā daṯbasb^{e}sūn pagdā

The preceding text was set typeset with the following input:

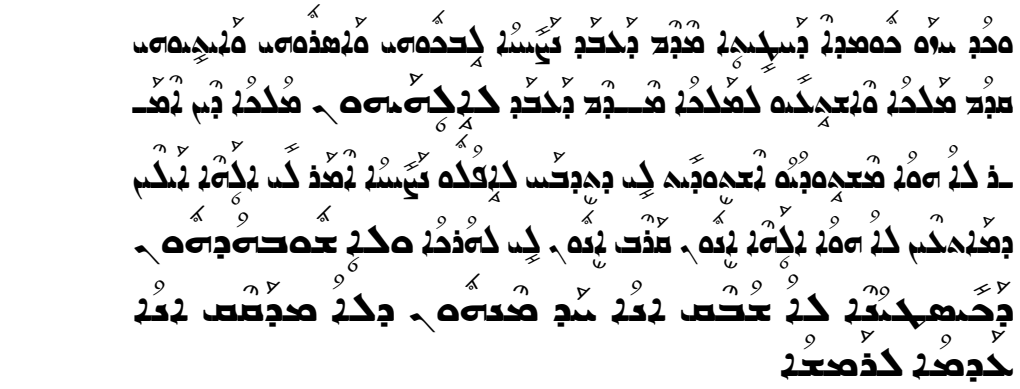
```
\documentclass[12pt,a4paper]{article}
\usepackage{serto}
\begin{document}

<SERTO>

wk=ad .hzaw kuwmd'e da.h.tIyt=A' : medem
da'bad na.siy.h=a': lAbkuwh=y=
wa'sruwh=y= wa'ytIywh=y= qd=am malk=a'.
we'^stA'iyw lmalk=a' me----dem da'bad
{\sertob l'Al=Ahayhwn}. m=alk=a' deyn 'emar. l'=a hw=a'
me^stAwd=ay=aw 'e^stEwdiyt lIy dtEdba.h
l'Ap=aluw: na.siy.h=a' 'emar. liy 'la=Ahe'
'ayleyn dma't'eyn. l'=a hw=a' 'la=Ahe'
'Enuwn. qareb 'Enuwn lIy lh=ark=a'. \sertob wl'=A
^suwbh=adhuwn dakiys.ty=ane' l'=a ^s=abeq
'=n=a' .had menhuwn dl'=a mdaqeq '=n=a'
```

```
‘adm=a’ lram^s=a’.
</SERTO>
\end{document}
```

Replacing <SERTO> by <CHALDEAN> results in this



The file was preprocessed using `serto.py`.

6 Things still missing

The todo-list is long. I try to add features etc. as soon as possible. Please tell me items you would like to have, but which are not yet on this list. Any volunteers are welcome!

- proper typsetting of texts (without preprocessor, maybe in the ArabTEX package)
- proper treatment of *matrēs lectionis* (long vowels)
- proper treatment of the silent bar ܡܒܛܠܢܐ *mḇaṭlānā*
- proper treatment of *begāḏkepāṯ* with hard sign ܩܘܫܫܝܐ *quššāyā* and soft sign ܪܘܟܟܝܐ *rūkkāḥā*
- automatic transcription mode
- interpunction
- proper dealing with ligatures
- ܐܣܛܪܢܓܠܐ *ʾesṭrangelā*

- numbers the `inputenc.sty` package.
- support for existing Syriac fonts

7 Installation

The easiest way to install the fonts and the preprocessor is by installing the debian package (this includes only the `pfb`, `tfm` and `afm` files for the fonts), the needed styles and the preprocessor, but not the METAFONT sources:

```
sudo dpkg -i serto-1.0.deb
```

If you are not on a Debian or Ubuntu plateform, you need to install manually from the `.tgz` file:

7.1 Using Metafont sources

Put the `*.mf` files into a subdirectory `serto` of your metafont branch in your `texmf`-directory. For example using the texlive distribution under Linux, you should put them into `/usr/local/share/texmf/fonts/source/serto/`. Do not forget to call `texhash` in order to make the tex software find the newly installed fonts.

7.2 Using vector fonts

Using vector fonts depends a little from the TEX-installation used, the following is tested for Ubuntu 12.04 and 10.04, it will probably work on Debian platforms as well, or other platforms using the texlive installation.

- copy `syriac.map` to `/usr/local/share/texmf/fonts/map/dvips/config/`
- copy `*.afm` to `/usr/local/share/texmf/fonts/afm/syriac/serto/`
- copy `*.pfb` to `/usr/local/share/texmf/fonts/type1/syriac/`
- add `Map syriac.map` to `/etc/texmf/updmap.d/10local.cfg`
- run `sudo update-updmap`
- run `sudo updmap-sys`

7.3 Other files

The Stylefile etc. `*.sty`, `*.fd` go into a directory for stylefiles, e.g. /usr/local/share/texmf/tex/latex/serto/.

The preprocessor `serto.py` and the encoding file `serto.font` somewhere where it can be found (e.g `/usr/local/bin`). They must reside in the same directory unless you specify in the environment variable `SERTOFONTDIR` the directory containing `serto.font` and `assyr.font`. Possibly you have to adjust the first line of the preprocessor `#!/usr/bin/python` if your python interpreter is somewhere else.

8 License

This Material is subjec to the LaTeX Project Public License 1.3 (http://ctan.org/license/lppl1.3).

9 Changelog

- Version 1.0
  - adding a character for the *linea occultans* above the letter
  - `SERTOFONTDIR` enviromnment variable to specify the directory of `*.font` files
  - some UTF8 support
- Version 0.7
  - Chaldean vowels
  - Integration of the Chaldean font provided by Tony Khoshaba
  - Major adjustements to the `serto.py` preprocessor
- Version 0.2, 0.3 and 0.4
  - can't remember, didn't keep track of changelog those days...
- Version 0.1
  - Initial version

References

Brockelmann, Carl: 1960. *Syrische Grammatik mit Paradigmen, Literatur, Chrestomathie und Glossar*. Leipzig: VEB Enzyklopädie, 8th edn. 2

Costaz, Louis: 1986. *Grammaire Syriaque*. Beyrouth: Dar el-Machreq. 2

Nöldeke, Theodor: 1986f. *Kurzgefaßte syrische Grammatik. (Neubearbeitung von Schall, Anton)*. Darmstadt: Wissenschaftliche Buchgesellschaft. 2

Nöldeke, Theodor: 2001. *Compendious Syriac Grammar. Translated by James A. Crichton*. Winona Lake, IN: Eisenbrauns. 2

Ungnad, Arthur: 1932. *Syrische Grammatik mit Übungsbuch*. München: Beck, 2nd edn. 2