SVG Cuneiform Tool (v4.4)

This paper is about an application which can draw cuneiform signs and browse lists of such signs, in a specified or random order, with or without indication of their values, with or without displaying the sign. This application was designed first as a practical way to learn or teach the glyphs of a specific syllabary in a way slightly more modern than with flashcards, though not necessitating complicated devices or installing specific software: a simple Web browser on a computer or on an Android smartphone or tablet will do.

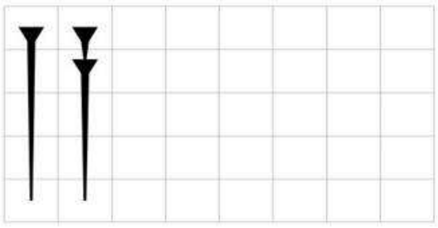
The script which is used in the application provides also an interface to display any sequence of glyphs inside an HTML page either as an *svg* element, an *img* element or a *canvas* element. Depending on the browser some of those images may be saved as graphic files and embedded into a text document in order, for example, to publish the normalized cuneiform rendering of some transliterated text.

A detailed explanation of how to use all those features is given below, but first it seems advisable to introduce the method used for displaying the glyphs. This information is not needed by those who need only use the tool, but is important for those who want to know how the tool could be adapted to a specific need, for example a cuneiform syllabary from another time period or for another language.

A formal language for describing cuneiform signs

This description language is based on decomposing each cuneiform sign into a sequence of primitive glyph elements, like horizontal, vertical, oblique wedges, hooks etc. Each glyph element composing the sign is then positioned on a grid of 8 by 5. Why those dimensions? Because it seemed to fit the usual geometry of the Middle-Elamite and Neo-Assyrian signs I am currently interested in, but other proportions could have been chosen. The columns of the grid are numbered from 0 to 7, left to right (abscissæ), and the rows from 0 to 4 (ordinates), top to bottom.

Let me provide an example right away. Syllabogram *a* consists of two primitive glyph elements: a vertical wedge beginning at coordinates (0,0) and 4 grid units high. This can be said in the formal language as "**v 0,0,4**", i.e. a code point **v** for the concerned glyph element, a space, and a coma separated list of numerical arguments. The second part of the glyph consists of two aligned vertical wedges: that primitive glyph element starts at coordinates (1,0) and is 4 units high too. In other words it is described by: "**vv 1,0,4**", where **vv** is the code point of that type of glyph element.

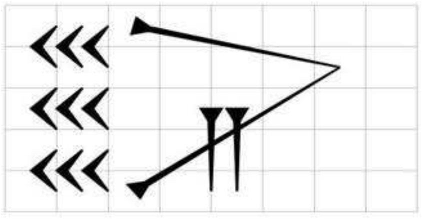


In that way the sign above can be formally described by "**v 0,0,4 vv 1,0,4**". Note that it would have been possible to describe the second part of the glyph as a combination of two vertical wedges ("**v 1,0,1 v 1,1,3**"), but such combinations appear often enough to be considered as primitive.

A more complex example is provided by syllabogram *in*. It is formally described as "**c9 0 o 2,0,6,1 o 2,4,6,1 v2p 3.5,2,2**", which means that it consists of the following primitives:

- a bunch of nine small hooks beginning at abscissa 0
- an oblique wedge from coordinates (2,0) to (6,1)
- an oblique wedge from coordinates (2,4) to (6,1)
- two close vertical wedges beginning at (3.5,2) and 2 units high

It is an occasion to note two things. First, that each primitive glyph elements requires a specific number of arguments in order to position it on the grid. Primitive **c9** requires only an abscissa argument, while primitive **o** requires two pairs of coordinate. Second, the coordinates and the length arguments can be decimal numbers, if necessary, in order to achieve a finer rendering.



Interpreting the formal language to draw the signs on screen

Now that we have a way to formalize the description of a sign it is possible to design a computer tool which understands that formal language and uses it to draw the signs based on their formal description. I achieved that by using the possibilities offered by Scalable Vector Graphics (SVG) in combination with a Javascript program. The tool can be launched simply by opening an SVG file with a recent Internet browser (Firefox as of version 10.0.1, Chrome, usual Android browsers, and also Internet Explorer 9 and higher, but with some limitations).

The signs are displayed in a stylized way, like shown on the pictures above. There are five different pre-defined styles in the tool (not all of them are available with all syllabaries), and additional ones can be added in the program.

As of v2.2, sign variants are being taken care of in the tool: it is possible to define a glyph as a variant for a given sign. E.g. in the Achaemenid syllabary, based primarily on the glyphs from Persepolis, Behistun variants have been introduced. Variants are characterized by a name ending with a sequence like '*1', '*2' etc.: so BAR*1 is the Behistun variant of BAR. Regarding the Middle Elamite syllabary, based primarily on the Great stele of Šilhak-Inšušinak, glyph variants forms have been introduced from another stele (EKI 48).

Using the cuneiform tool for learning purposes

Why would it be useful to display signs on your computer or smartphone screen using a specific tool if you can use any word processor and a cuneiform Unicode font instead? There are indeed good cuneiform fonts, freely available on the Internet, and it is now possible to write a text in cuneiforms. But if you study a cuneiform inscription there is little chance that you can find a font matching precisely the forms of the glyphs found in your inscription.

The cuneiform tool provides you a method for analyzing the composition of the glyphs, for displaying them in a stylized way and for helping you memorize them through a functionality of display lists. A display list is a list of signs (or glyph elements), which can be displayed in a predefined order, or at random, with the indication of the sign value, or without it, with the graphic representation of the sign, or without it. It is easy to understand how one can use that functionality for exercising to learn the syllabary of an inscription.

The syllabaries currently implemented in the tool are large subsets of:

Middle Elamite: the syllabary used on the Great stele of Šilhak-Inšušinak (Scheil MDP XI,XCII; König EKI 54) as it appears from the charts in *Mémoires de la Délégation en Perse*, Vol. XI,

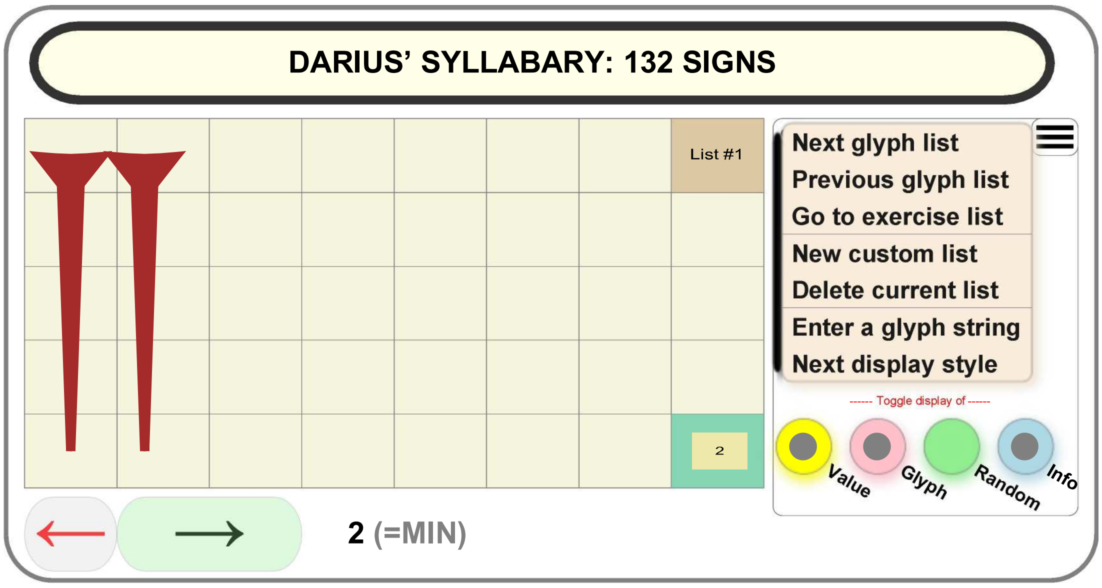
Neo-Elamite: the syllabary used on Hanni's inscription at Kul-e Farah (Scheil MDP III,LXIII; König EKI 75) as it appears on a photograph kindly provided by W. Henkelman. When a glyph is displayed in a non-Middle-Elamite syllabary, a small picture of the corresponding sign in Middle Elamite will appear (if available) in the lower right cell of the grid.

Achaemenid Elamite: the syllabary used on Darius' *Charte de fondation* at Persepolis (DPf) as it appears on a photograph of the Oriental Institute of Chicago. Since that inscription does not provide enough different glyphs, additional ones have been taken from good photographs of other Persepolitan inscriptions (DPa, XPa), and even from DSz (according to the plates in Vallat, *Table élamite de Darius Ier*, RA, 1970), another foundation charter for Darius' palace, but from Susa. The latter has a quite different writing style (e.g. the small horizontal wedges appear as a small triangle with no shaft at all) so it was necessary to somehow normalize the glyphs according to the usual style of DPf.

Though I did not try to, there is probably no major difficulty to adapt the program to the cuneiform signs used in other languages and time periods, except that some basic glyph elements and rendering styles could need to be added.

Directions for use

The tool is embedded in the current Web page just below. It can be launched directly also by using the URL: http://kursoj.pagesperso-orange.fr/cunei/svg_cunei.svg.



The tool is provided with a menu and a set of command buttons acting as follows:

Bottom green button with a right arrow sign on it: clicking on it enables the next sign in the list to be displayed. When the last sign of the list is reached, the next click enables the first sign again.

The smaller button with a red left arrow sign enables to backtrack to the previously displayed sign (provided we are not at the beginning of the list).

The menu is displayed at startup but can be hidden or displayed again by clicking on the menu button. It contains the following commands:

"Next glyph list": by a click on this menu item the user switches to the next "display list". A display list is a list of signs to be displayed. Some display lists are already built in the tool, new ones can be user-defined. The first built-in display list to appear in row encompasses all glyphs currently declared in the 'syllabary' variable, i.e. the syllabary of the Great Stele. The last list is a reminder of the catalogue of available glyph elements. There is also a special list, named "exercise list", which is populated by the users from the signs they are presented with when browsing other lists. All the display lists are arranged in round robin and the "next glyph list" command is a way to cross the round robin indefinitely in the forward direction. In order to keep track of the currently displayed list a special cell of the grid bears the number of the current grid. (Caution! If a list is deleted or added the list numbers are reassigned.)

Example: a predefined list encompasses all the Middle Elamite glyphs used in inscriptions IRS 22 through 25. Other predefined lists contain all the glyphs with some shape similarity (rectangle, romb, BE motif etc.)

"Previous glyph list": by a click on this menu item the user switches to the previous display list in the round robin of display lists.

"Go to exercise list": by a click on this menu item the user switches directly to the exercise list. This command is a kind of shortcut to avoid multiple successive "next" or "previous glyph list" commands to reach the exercise list. When the exercise list is being displayed the user can be presented with "hints", i.e. words, phrases or sentences containing the current glyph. They are taken from real inscriptions: the hint serves both as an illustration of the glyph in a real context and as a device to better memorize the glyph, or to help recognize it.

"New custom list": by a click on this menu item the user can specify a custom sign list. It may be specified: (1) as a list of glyph values (e.g. *ta sha li qa pi*, case independent, and the separators may be spaces, hyphens or commas), (2) or as a simple search pattern (e.g. @*n* in order to enlist all glyphs whose value contains an *n*). A mixture of the two ways to specify the elements of the custom list is possible (e.g. entering @*tu ud* will retrieve *tu tu2 tu4 tuk tum4 ud*). Due to Javascript's security limitations the tool has no access to the user's file system. So it is not possible to save the custom lists with the tool, but it is rather easy to save their defining strings in a specific text file and re-enter them each time you need.

"Delete current list": by a click on this menu item the user can delete the current display list. When a list is deleted it is normally withdrawn from the round robin. Deleting the exercise list, however, gives a different result: the list is emptied, but remains in the round robin, available for further use.

"Enter a glyph string": by a click on this menu item the user is prompted to enter a string of syllabograms he wants to be displayed. The glyphs are displayed with a reduced size on a panel at the top of the picture. It is the same panel which is used to display hints (see above). Example input string: *DIŠ-ŠIL-HA-AG AN-IN-SU-UŠ-NA-AG* (or *dish shil ha ag an in su ush na ag*). A limited treatment of determinatives (*f.*, *v.*, *d.*, *h.*) is provided. Polyphony of VC signs ending in a stop is also implemented (e.g. you may enter *ak* or *aq* instead of *ag*), so you may enter also *v.šil-ha-ak d.in-su-uš-na-ak* instead of the above mentioned. As of v4.2, the conversion of the diacritics acute and grave into indexes 2 and 3 is provided. A limited treatment is provided also for some frequent writings in elamology, like *ka*$_4$ (instead of *qa*), *pu* (instead of *bu*), *tin* (instead of *din*), *tá* (instead of *da*) and some others.

"Next display style": by a click on this menu item the user changes the style the sign is displayed in. When the last style is reached, the next click puts the first style into action again.

The round-shaped buttons located below the menu behave like radio buttons and perform the following functions:

Yellow button: by clicking on it the user toggles the display of a text describing the syllabic value of the displayed syllabogram or the kind of the displayed glyph element. Browsing a list without displaying the value of the signs is a way to test if one can recognize them. Signs which are not easily recognized can be enlisted in the exercise list for further practice.

Pink button: by clicking on it the user toggles the display of the glyph itself. The tool can thus be used to prompt the user for a particular glyph and let them control their answer with the pink button.

Green button: by clicking on it the user toggles between a fixed displaying order of the glyphs in the current list or a random order. Using random order is a way to avoid recognizing the signs by their position in a sequence and to focus on their mere aspect.

Light blue button: by clicking on it the user toggles between a fixed information panel and an animated one which disappears after some seconds. This button is useful because the panel used for displaying hints and user-supplied glyph strings is located behind the information panel and some browsers do not implement the animation features. If your browser does, you should probably click this button right from the beginning of your session in order to avoid being bothered with the information panel when panel below it is active.

Other regions of the tool's display are clickable in order to perform the following functions:

When the glyph grid is clicked a pop-up window opens. If the click happens in the rightmost upper cell of the grid the user is enabled to confirm that the entire current display list be added to the exercise list. If it happens everywhere in the grid outside this special cell the tool proposes enlisting on the exercise list the currently displayed glyph only.

When the panel for hints is clicked, the next available hint is displayed. Some glyphs are rather frequent and have many examples associated with them. Others have no example at all, because they occur rarely in the targeted inscriptions and it is uneasy to capture their precise design in them. If the panel is currently used to display a user-supplied glyph string, a click simply erases the panel.

The upper right rounded square menu button controls the displaying of the main menu. When it is clicked, the menu disappears or reappears. When the menu is hidden the user has access to an additional menu item labelled next syllabary. By a click on this item the user switches to the next syllabary. All syllabaries are arranged in round robin and the "next syllabary" command is a way to cross the round robin indefinitely in the forward direction. Caution: the user-defined display lists and the exercise list are syllabary dependent. They will not be reset when one changes the current syllabary, but they are not available in the new syllabary. Returning to the previous syllabary will reactivate them.

Web interface for text publishing

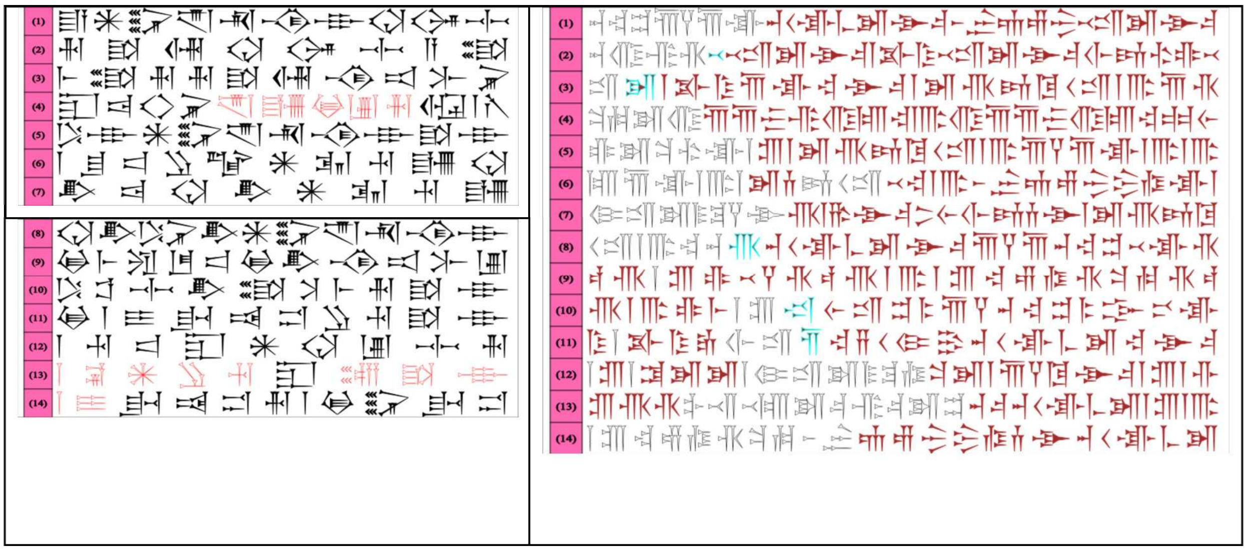
The script on which the SVG cuneiform tool is based provides also an interface to display a cuneiform text in an HTML page, in a way which enables saving it as a graphic file for further use, like text publishing. Use examples are provided through the links below:

- http://kursoj.pagesperso-orange.fr/cunei/webinterface_ex1.htm (Examples in Middle Elamite)
- http://kursoj.pagesperso-orange.fr/cunei/webinterface_ex2.htm (Examples in Achaemenid Elamite)

Specification of the Web interface function:

```
function draw_glyph_string_in_html(
  title, img_type, syll_name, style, line_length, scale_factor, y_compression, string) {
  // Draw a string of glyphs into an HTML page.
  //   - title:         Title to display in an h1 element; if relevant, is also the id
  //                    of a calling DOM object: if no DOM object has this value for id
  //                    the result will be displayed directly in document.body
  //   - img_type:      Type of output expected. Possible values:
  //          - 'SVG':    The string is displayed in an SVG image embedded in the HTML page
  //          - 'CANVAS': The string is displayed in an SVG image embedded in the HTML page
  //          - 'IMG':    The string is displayed in an img image embedded in the HTML page
  //          - 'DATA':   The SVG image is presented as a URI with base64 encoding (to be copied
  //                      and used as an src attribute of an img tag)
  //          Depending on the browser and on the image type the image can be saved in SVG format
  //          or in other graphic formats.
  //   - syll_name:     Syllabary name
  //   - style:         Rendering style. If this argument is left undefined, the default style
  //                    for that syllabary applies.
  //   - line_length:   Length of each line of text, expressed in a specific unit. Line
  //                    wrapping occurs when a line surpasses the line length.
  //   - scale_factor:  Ratio between the specific unit used for line_length (and also for the
  //                    line_height which is assigned to 5) and the pixel size. A value of 10
  //                    is usually suited to produce legible glyphs. SVG images can be zoomed as
  //                    needed without loss, so that argument is less useful than for CANVAS image
  //                    type.
  //   - y_compression: Compression factor of the vertical axis. Typical values are 0.8 for ME
  //                    syllabary and 1 for the other syllabaries
  //   - string:        Text to be displayed. The string may have newline characters (to skip
  //                    to a newline), square brackets (to delimit restitutions), line
  //                    numbering (through a parenthesized number at the beginning of the line),
  //                    tab characters (to segment the text into blocks which are to be rendered
  //                    each in a proper image)
```

Samples of the results achievable with the Web interface are inserted here:



NB: The glyphs show different styles depending on their status: plain, restituted or half-restituted. Restituted glyphs have no filling and a different stroke colour, half-restituted have a different colour for filling and stroke.

Current shortcomings

Using the tool needs no particular skills except understanding what the buttons and menu items are for. But modifying the program to adapt it to one's specific needs (syllabary, style, display lists, examples…) may require some skills in Javascript programming. However many modifications can be performed simply by changing the values of the main declared variables at the beginning of the program, which does not require real knowledge of Javascript, just the capability to put one's specific data at the right place following the given model.

The tool is not a commercial program. It is provided "as is" and is far from being bullet-proof. In particular, if you intend to modify the glyph design, be aware that no check is performed on the arguments you provide for the description of each glyph element: neither on the argument number, nor on their value. An error window may pop up if the formal language parser encounters an unknown glyph element code (which can be the result of a previous error on the argument number), but most often an error will be revealed by a wrong drawing.

The tool is intended to be launched through a Web browser, most of which now implement an SVG rendering module. Some recent browsers may however operate the tool in degraded mode. This is the case of Internet Explorer because it lacks the animation features of SVG. That is why the information panel is not animatable and must be operated manually with the “Info” button.

For any question or comment about the SVG Cuneiform Tool, feel free to contact me. Feedback is highly appreciated.

Marc Bavant
http://uva.academia.edu/MarcBavant
mjb0(at)wanadoo(dot)fr