Egyptian as a Complex Script in Word Processing

Bob Richmond. DRAFT Version 2016-07-29

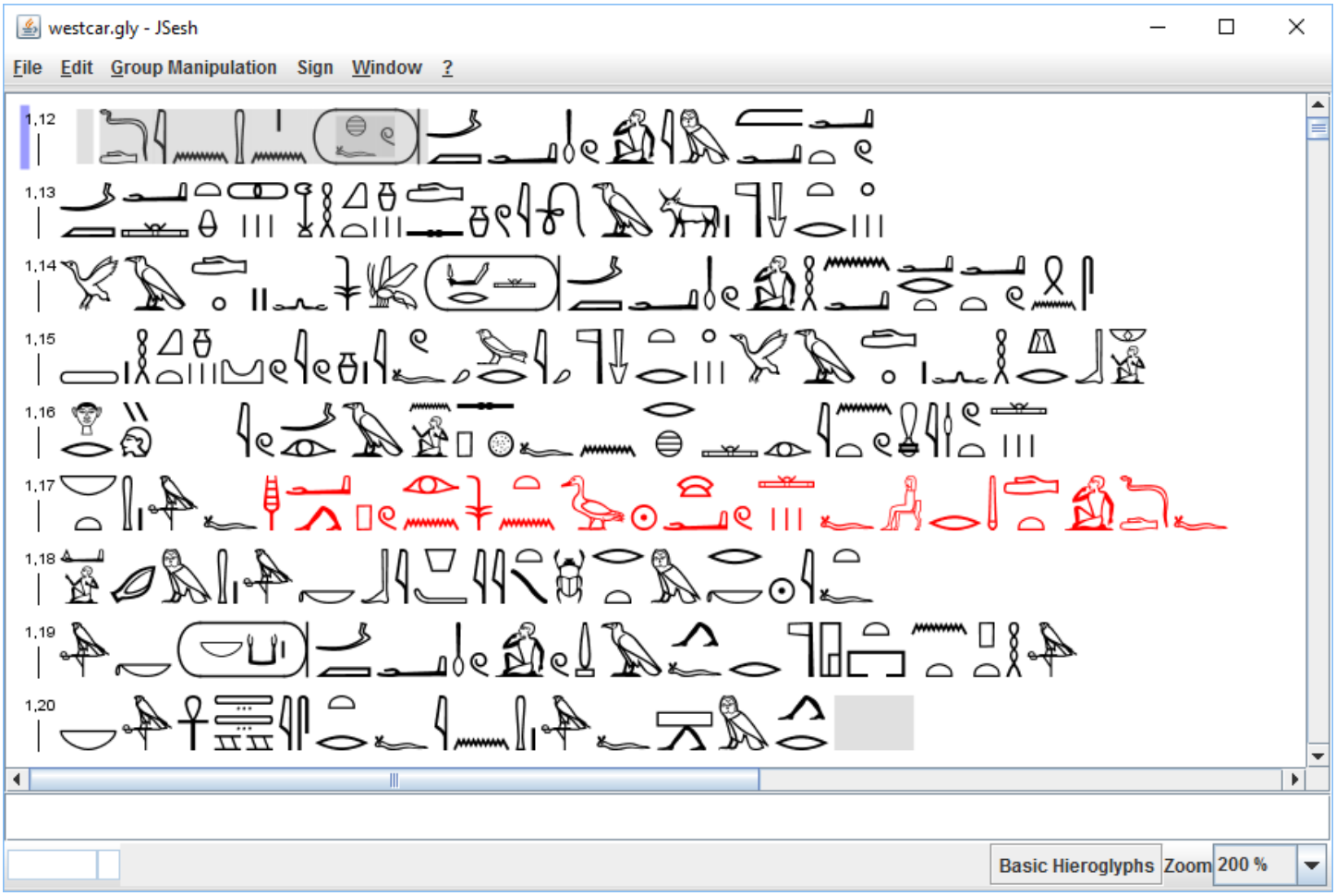
This document uses the control characters described in the *Proposal to encode three control characters for Egyptian Hieroglyphs* L2/16-018 with the controls implemented using fake code points. There will be some changes to hieroglyphic text from L2/16-018 so the exact detail here is still subject to change.

I'm using the current release of Microsoft Word on Windows 10 to write this document. Currently, LibreOffice and OpenOffice (on Windows at least) have bugs in their complex script support. They are used the same way when they work (I suspect both are still using old versions of open source components so presumably we'll have working versions by next year sometime if not sooner).

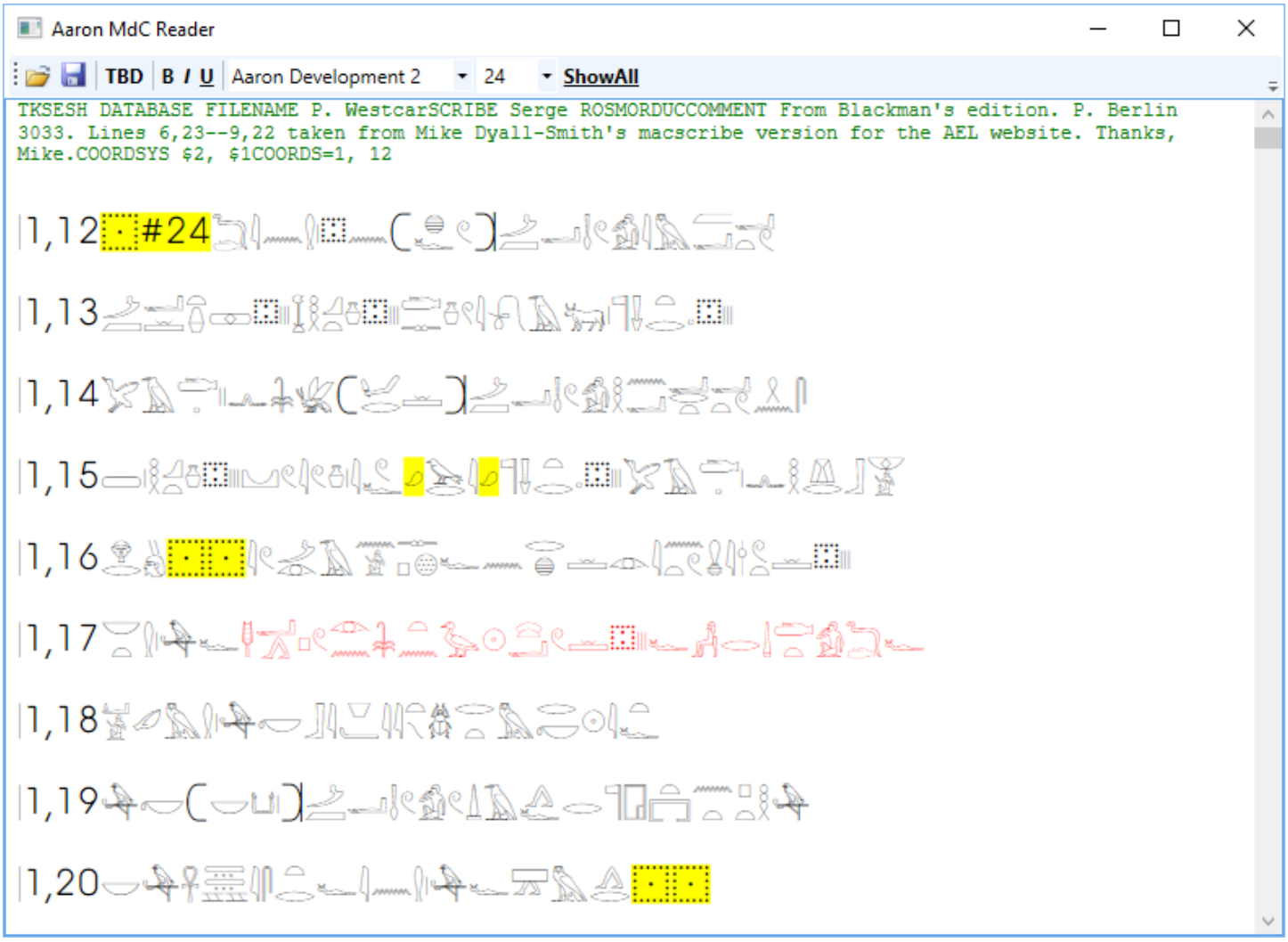
Software developers call it dogfooding – using your own software (and fonts) to look out for bugs. That's what I'm doing right now as I write.

1 Acquisition of some hieroglyphic text

I'll start with the Westcar Papyrus sample (westcar.gly) unedited just as supplied with JSesh 5.5 where it looks like:



JSesh obviously cannot safely use Unicode text until the standard is defined so I'll copy into a helper application (the Aaron MdC reader app here is one of my prototypes).



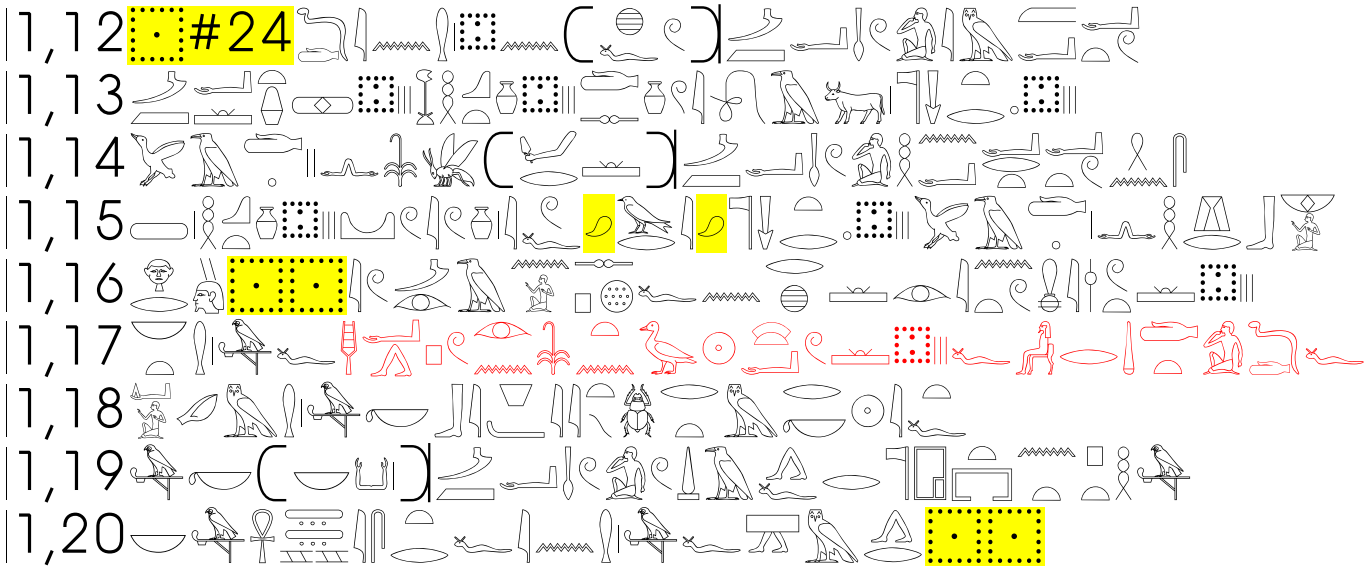
This MdC Reader app happens not to process shading yet. The part shade qualifier #24 is left as is. MdC items like the dot (shown as) and the variant sign (F51\R90) (which is not in Unicode yet) are highlighted in yellow.

Notice there are 6 vertical joiner control codes visible. A close look of the first of these shows the original MdC transcription used 1 instead of Z1 as a stroke. The other cases use 3 instead of Z2 or Z2A as a plural. This Reader/Font deliberately doesn't fix these errors at the moment.

So, time to copy/paste the text into this document.

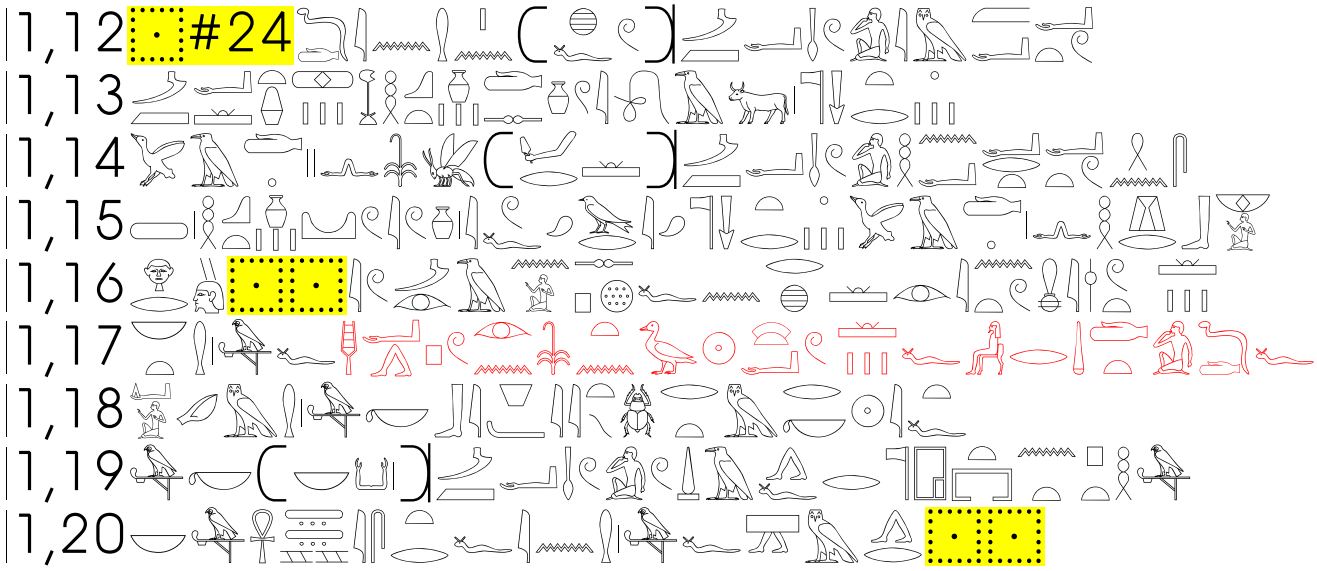
2. Hieroglyphic inside a word processer

So here it is: the first few lines of P. Westcar as text, copied from my app:



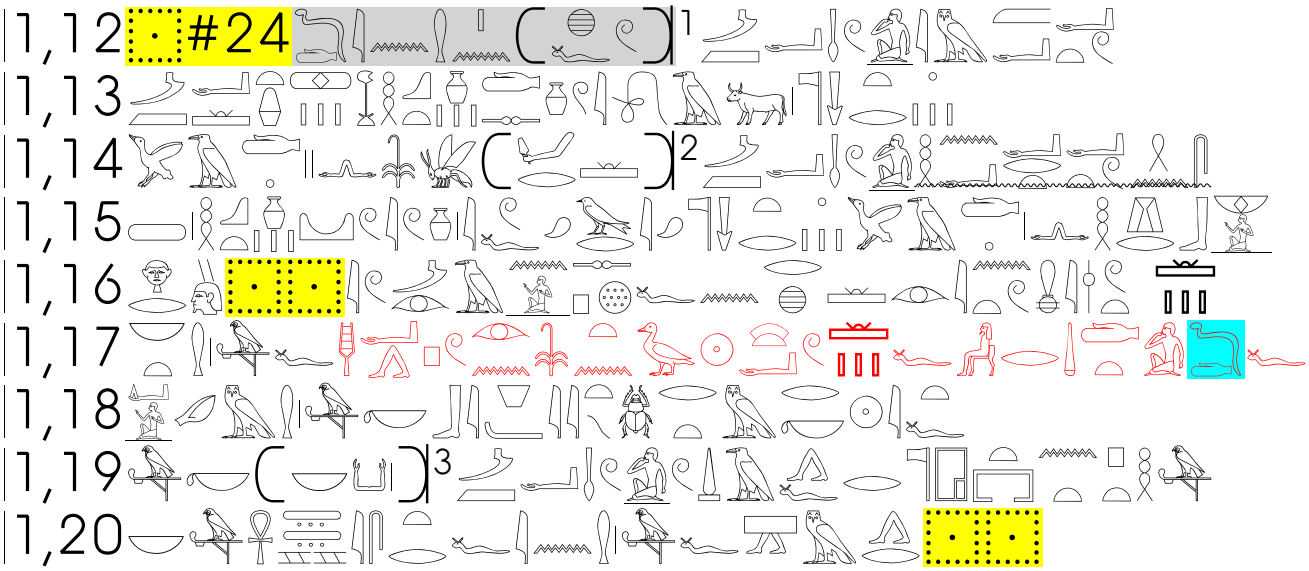
First I'll fix the numbers. I'll need the hieroglyphs 𓎆 and 𓏺𓏺𓏺 to copy and paste. Consider 𓎆𓏺𓏺𓏺, all I need to do is copy the 𓏺𓏺𓏺, select the 𓏺 and paste. Because 𓎆 joins with 𓏺𓏺𓏺 the 𓎆𓏺𓏺𓏺 automatically switches to 𓏺𓏺𓏺.

I'll copy the above and make those changes:

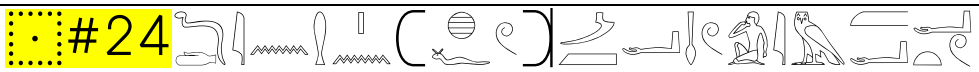
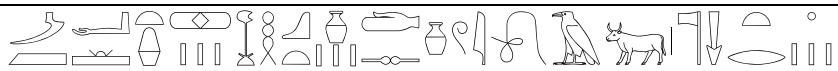
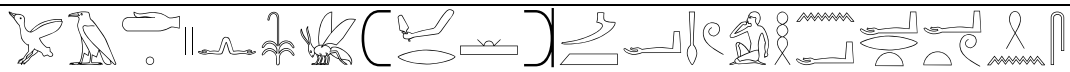
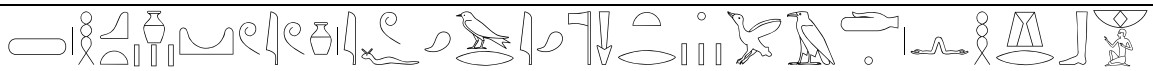
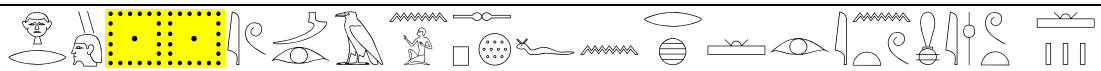
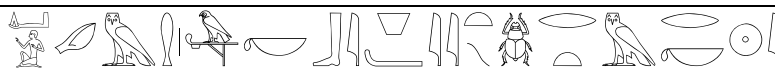
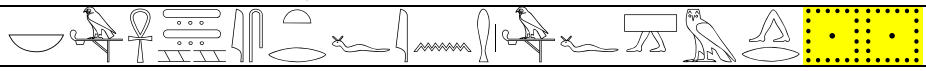


Done. No control characters to be seen.

I can now apply standard word processing attributes to the text. Here are a few examples of formatting applied to a copy.

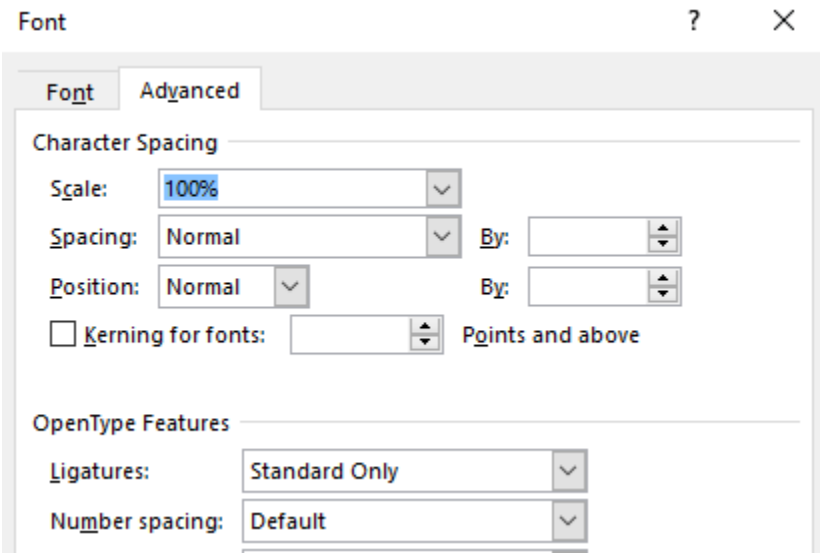


I can arrange the text in a table:

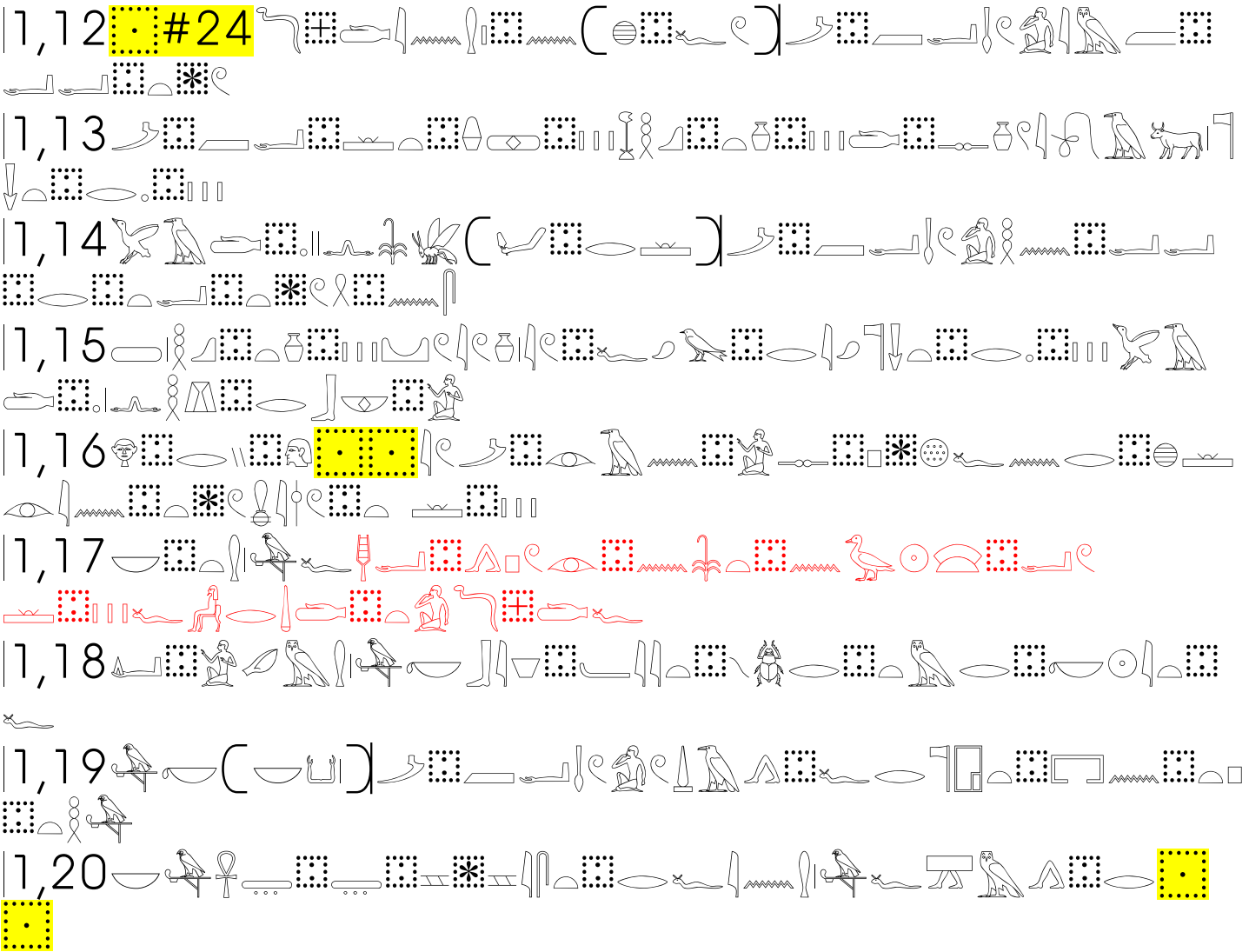
1,12	
1,13	
1,14	
1,15	
1,16	
1,17	
1,18	
1,19	
1,20	

This is a rich environment for working with hieroglyphic. It's far easy to do basic editing than using traditional MdC tools although some detailed aspects of features such as shading missing parts of a source text can't be treated in the exactly the same way. For an Egyptologist used to embedding graphics (created in specialist software) in documents, it makes for a truly liberating experience for work that fits within the scope of what can be done here.

It is instructive to look at the control codes underlying what is visible.To do this I'll make a copy, select it, right click on the section and choose the font dialog



The apply the option to switch ligatures to none:

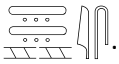


This ‘show codes’ method is handy for many editing purposes once you get used to hieroglyphs in Unicode. It is much quicker than traditional MdC modes of working for a range of purposes but employs a similar pattern of working.

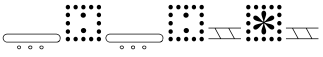
3. Hieroglyphic as a Complex Script

Egyptian hieroglyphs class as a writing system that uses a **complex script** because hieroglyphs need to be sized and positioned in relationship to each other in a layout for display. A large proportion of the worlds writings are not complex in this sense for example the Latin script and Chinese characters. But there are popular scripts used for living languages such as Arabic, Devanagari and Thai.

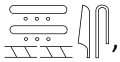
Software that supports such scripts may contain elements to simplify ways of working with the text in terms of editing, keyboard input and so forth. Since Word doesn’t have such specialist features for Egyptian right now we only have default complex script support for editing.

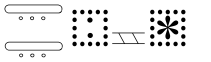
Consider fragment  . Click on the right edge of the  cluster to place the flashing input caret in front of  .

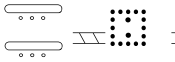
Now use the left arrow key to move the caret to the left. Instead of moving to the left of  the caret moves to

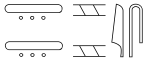
It actually takes 7 left arrow presses to get from right to left. The underlying code sequence  shows why: there is a total of 7 characters, hieroglyphs and codes.

A simple method to break up a cluster is to use this fact.

Start with the caret in front of  in  , tap left arrow once and press space bar.

You now see  . The full code sequence for  has been broken by the space.

Next delete the  and move the  in front of the space to get  .

Finally delete the space to get the rearrangement  .

True this is a little clumsy and Egyptologists will normally want to copy text to edit into a specialist helper application for hieroglyphic where editing is easier then copy back to Word. Nevertheless, once you get the hang of the method it’s very convenient for simple edits.

Inserting and deleting hieroglyphs or control characters into a text using the L2/16-018 protocol works well because results are normally as expected. This is by design. E.g. insert  between  and  in the sequence and the result as expected is  . Likewise insert  in  for  or in  for .

Maybe one-day Word will add features to help. But the point is this works now so useful tools can be deployed once hieroglyphic writing is approved and released as part of the Unicode standard.

4. Final remarks

Hieroglyphic as text opens up huge improvements in efficiency and functionality for how Egyptologists and others work with hieroglyphic writing. I hope this comes over in this document.

It will be obvious from this note that it is impossible to shield the user from control codes entirely. Indeed, it can sometime be useful to work at the control level.

Improved support for hieroglyphic writing in general purpose software would help and input methods would be very useful. This may take years to come to fruition.

Control characters cannot be entirely hidden in all situations and one reason for this is very obvious. Font developers get to pick and choose what characters and features to support in their font. A font designed to do a great job of rendering hieratic transcription to hieroglyphic is unlikely to contain unnecessary baggage such as Ptolemaic hieroglyphs from an extended repertoire. Nor contain support for irrelevant arrangements of hieroglyphs. At one extreme a font for casual users may only contain a small subset of hieroglyphs and arrangements of them. Missing character symbols and visible control codes are to be expected when a specific font is used to render hieroglyphic text composed for a more fully featured font.

The L2/16-018 system used here works well for a huge volume of material in printed works and digital encodings and covers a large part of what Egyptologists and others want to do with the writing system. TopBib, grammars, dictionaries, teaching materials and so forth work well. The proposed group joiners expand capability into vertical and ‘tall group’ orthography while maintaining the usability level seen here for classic horizontal writings styles – a topic for another day.

There are open issues on support for certain (mostly rare) writings in hieroglyphic Unicode. There are other ways L2/16-018 could expanded. All good but one reason I wrote this document is to help explain the big picture and illustrate why adding complexity to bread and butter writing with hieroglyphs would be a really bad idea for the future users of hieroglyphic writing in Unicode.