Font version 1.5x

Prepared by John Hudson, Tiro Typeworks, for the Society of Biblical Literature

SBL Hebrew is a trademark of the Society of Biblical Literature
Version 1.5x of the SBL Hebrew font is an interim release. It adds support for Dead Sea Scrolls transcription (see page 17), and for the new Unicode 5.0 holam haser for vav character (see page 16), and fixes some bugs in previous versions. For the most part, input and editing is exactly as in version 1.12 and later.

Version 2.00 is already in development, and will be released later this year; this update will include more significant changes, especially refinement of the positioning of marks relative to letters and to each other, and of the spacing between letters, especially when one or more of the letters takes multiple marks.

When we released the first version of the SBL Hebrew font, adequate application support was almost entirely limited to Office 2003 on the Windows platform. Since then, as subsequent versions have been made available, we have been able to note improvements in support, both in other Windows applications and on Macintosh and Linux platforms. The current version of the manual no longer records such information, since the pace of new support is now such that this dynamic information is better presented via the SBL website. For this information, downloads for current versions, keyboard drivers and other resources, see http://www.sbl-site.org/resources (in progress).
If you have not already downloaded the latest version of the SBL Hebrew font, you should do so now. See page 23 for download information and links to other useful websites.

Please take some time to review this manual. SBL Hebrew is a complex font that makes use of new technologies that may be unfamiliar to you if you have not worked with Unicode encoded text or font files before. Many of the lessons you may have learned from using other Biblical Hebrew fonts and software may not apply, and it may be necessary to develop some new work habits.

The SBL Hebrew font is designed for use with systems and applications that natively support appropriate Unicode text processing and OpenType Layout. Most of this manual is specific to Windows installations, since this has, to date, been the most common platform for users of the font and has also provided the best support. To install the font on a Windows system, open the Fonts folder and drag and drop the $SBL_Hbrw.ttf$ file onto the folder. The Fonts folder can be opened either by navigating in Windows Explorer to $c:\Windows\Fonts$ (where $c:\$ is the root folder containing your Windows system files), or by opening the Fonts folder from the Control Panel, accessible from the Start button Settings menu. Note that if a previous version of the font is installed, you will need to delete that from the Fonts folder before installing the new version.

The SBL Hebrew font can also be installed on older versions of Windows, but because the font uses a pure Unicode encoding, not an 8-bit codepage, it may not work consistently in all applications.

It is also possible to install the font on Linux and other open source systems using the FreeType library, and on Mac OS X. Please see the SBL website for more information.

Although SBL Hebrew is an OpenType format font, it has a .ttf file name extension and will display the regular TrueType icon, rather than the distinctive OpenType icon. The .ttf extension allows the font to be recognised on older versions of Windows (although these do not support all aspects of Biblical Hebrew text). The regular TrueType icon is displayed only because the font does not contain a digital signature. Note that the different icons available for TrueType OT fonts do not indicate anything about the presence or absence of OpenType Layout tables for glyph substitution or positioning, both of which are present present in the SBL Hebrew font. See pages 4–5 for more information.
Most fonts do not come with user manuals, and most do not need them. SBL Hebrew is a complex font that uses new encoding and layout technologies, and this manual explains these technologies to help you get the most out of your new font. This manual also discusses known issues relating to these technologies, especially the complicated but important issue of text normalisation. Although you can install the SBL Hebrew font on most current operating systems, and can immediately begin working with it, please take time to review this document. Understanding issues like optimal character ordering will help ensure that the SBL Hebrew correctly displays your text.

SBL Hebrew is a TrueType-flavour OpenType font.

The original TrueType font format was developed by Apple Computers and released in the early 1990s. It was quickly licensed by Microsoft Corporation and is the system font format for Apple and Microsoft operating systems (the packaging of the format differed on the two systems prior to the introduction of Apple’s Mac OS X, which can install Windows TT fonts). A TrueType font is a collection of tables, each containing information specific to a particular aspect of the font. For example, the glyf table contains outline information, the cmap table maps glyphs in the font to character codes for text entry and storage, and so forth.

The OpenType font format is an extension of the TrueType format, jointly developed by Microsoft and Adobe Systems. An OpenType font has the same table structure as a generic TrueType font, but with the option to include a number of additional tables. There are two key components to the OpenType extensions: PostScript outline data and OT Layout data. The first of these determines the ‘flavour’ of a font—that is, the kind of outline and rendering technology used to ‘paint’ text. As noted above, SBL Hebrew is TrueType-flavour, meaning that it uses the original TrueType outline format and rendering technology rather than PostScript. The second key component, OpenType Layout data, is essential to the correct rendering of complex scripts like Hebrew, and much of this manual is concerned with this technology and how it renders Biblical Hebrew. The SBL Hebrew font contains two kinds of OpenType Layout data: glyph substitution and glyph positioning. These are stored in the OpenType optional tables GSUB and GPOS.

For more information about the OpenType font format, see the specification and other material on the Microsoft or Adobe websites. The Microsoft introductory essay *Windows Glyph Processing*, which also discusses other aspects of complex script shaping referred to in this manual, might be a good place to start. None of this material is
essential to being able to use the SBL Hebrew font, but it will help you better understand that technologies on which the font relies.

The SBL Hebrew font uses standard Unicode character codes to encode Hebrew letters and marks. Unicode is an international character encoding standard that provides a single unique code for every semantic element necessary to encode plain text in supported scripts and languages. Today, most of the world’s major scripts and languages are supported by Unicode, and recent efforts have led to the encoding of numerous historical scripts. Because every character in Unicode has a unique code, rather than sharing codes across multiple codepages or being assigned to different codes e.g. on Windows and Mac operating systems, text encoded in Unicode can be safely exchanged between different operating systems and applications that employ Unicode. The benefit of such a standard to scholarship, where authors and publishers are often using different software, should be obvious.

In the past, Hebrew text was supported on different platforms, and even between different applications, using a variety of standard and not-so-standard 8-bit encodings. Because the fonts that supported these encodings tended to be ‘dumb’ fonts, i.e. without built-in layout intelligence, often multiple fonts would be needed to correctly display complex texts such as found in Biblical scholarship. This, of course, increased the likelihood that text produced with these encodings could not be reliably and accurately exchanged between systems and applications, because correct display relied on not only knowing which encoding standard was used but also which font was used where. Unicode and OpenType solve these problems by using a unique code for each Hebrew consonant and mark, and employing layout intelligence to map from the encoded characters to the appropriate arrangement of glyphs to display a given text. [See the examples on the following pages.]

While the benefit of Unicode is easy to see, it does require that the SBL Hebrew font be used in a Unicode text encoding environment, i.e. in systems and applications that use Unicode character codes when storing, manipulating and displaying text. No guarantees can be made about its performance in non-Unicode environments. [Note, however, that some applications will internally map from Unicode characters in a font to 8-bit codes used internally by the application. In this case some text may display correctly with the SBL Hebrew font, although the text may be subject to the typical exchange problems of 8-bit text.] It may be that your current document publishing workflow involves non-Unicode applications, and this will limit, for the time being, the usefulness of the SBL Hebrew
font in your work. During the past few years, Unicode has become the dominant encoding standard for most major operating systems, and is, for example, specified by the World Wide Web consortium as the default encoding of XML documents. It is very likely that the makers of your present software will be updating their applications to handle Unicode, and they may be able to give you estimates on when suitable upgrades will be available. If you are tied to software that is not being updated to handle Unicode soon, or if you simply cannot wait and like the design of the SBL Hebrew typeface, it may be possible to arrange to have custom fonts made for specific 8-bit encodings, or to make your own. For information on the kind of modifications that are permitted, or for contact information, please see the font license in Appendix A, page 20.

Uniscribe
This section explains the role of the Microsoft Unicode Script Processor (Uniscribe) in shaping complex scripts. The most important thing to note in this regard is that different versions of Uniscribe will produce different results for some Hebrew mark sequences.

The layout intelligence in OpenType fonts relies on system or application support for basic linguistic shaping. In simple terms, systems and applications deal with characters, and fonts deal with glyphs, i.e. with the visual representation of characters. Complex scripts like Hebrew require systems and applications to be aware of right-to-left and bidirectional layout and of character properties that distinguish e.g. consonants from combining marks. Some applications will have built in support for such things, while others will rely on standard system components. Microsoft's Unicode Script Processor—commonly referred to as Uniscribe—is a standard system component that includes shaping engines for various scripts, including Hebrew. Uniscribe works directly with OpenType fonts, such as SBL Hebrew, built according to Microsoft specifications. An application that uses Uniscribe will make calls to it as text is entered and edited; Uniscribe processes the input characters, and applies the layout intelligence in the font accordingly.

If you are using an application that utilises Uniscribe—these include the Microsoft Office Suite on Windows, Internet Explorer, and any other app that makes standard system calls for text input and output—you will find working with Biblical Hebrew text very easy and the SBL Hebrew font will automatically correctly display most text. I say most text, because there is always the possibility to create sequences of consonants plus marks that are linguistically non-standard and which cannot be correctly resolved by the layout features in the font. The SBL Hebrew font has been tested with every combination of consonant plus mark(s) that occurs in the Michigan-Claremont Old Testament text. The results of this testing are available as Acrobat documents from the Society of Biblical Literature Hebrew font website, and may be helpful in troubleshooting layout discrepancies with different shaping engines.
Different versions of Uniscribe ship with and are used by different applications and system versions, which means that rendering results may vary. We have been fortunate, during the development of SBL Hebrew, to test the fonts with unreleased versions of Uniscribe that will ship with upcoming software. The new versions of Uniscribe have been used to confirm that all consonant plus mark(s) sequences in the Michigan-Claremont text will be correctly rendered when these versions of Uniscribe are employed. Because SBL Hebrew has, in this respect, been designed for ‘cutting edge’ technology, some sequences will display incorrectly in systems and applications using older versions of Uniscribe. It is important to note that these sequences may be correctly encoded, and that documents with display problems are not erroneous. Typically, when a sequence of marks cannot be correctly rendered by Uniscribe, the shaping engine will insert a dotted-circle, and the mark or marks that cannot be applied to the preceding consonant will be applied to the circle. The example below shows three possible levels of display for a unique sequence of consonant plus marks from Job 7:11 (alef + hataf patah + ZWJ + meteg + dehi).

A. This example shows the correct rendering applied by Uniscribe version 1.471.4063.0, which ships with MS Office 2003. This and later versions can be expected to correctly render this and other consonant plus mark sequences.

B. This example shows incorrect rendering applied by Uniscribe version 1.325.2180.1; this is the version of Uniscribe that ships with Windows 2000. As you can see, this version of Uniscribe is unable to correctly apply dehi to a consonant that already carries a combination of hataf vowel and meteg, so inserts a dotted circle.

C. This very incorrect example shows the same sequence crudely displayed by an application that does not use Uniscribe and cannot implement the glyph positioning intelligence in the SBL Hebrew font. The marks are blindly centered below the consonant, but they collide and do not interact correctly. In some applications, the marks may not even be centered below the consonant, but will cluster between it and the next letter.

Software developers producing software for use in Biblical scholarship can arrange for a redistribution license for Uniscribe. This enables them to bundle more recent versions of Uniscribe for utilisation by their software. Please contact Microsoft Typography for more information.
Note that in the two examples rendered by Uniscribe the meteg combines with the hataf patah, forming a mark ligature in which the meteg appears between the two parts of the vowel. For more information about this and other aspects of meteg handling, see the section of this manual on page 13.

Although implementation of the Unicode Standard is generally a boon to scholars working with texts in complex scripts, there is an unfortunate and quite serious problem in the current encoding of Hebrew. This involves the canonical combining class assignments that are used when text is normalised. Normalisation is a process by which sequences of characters in text that can be variously encoded but are semantically identical are treated as identically encoded. This can frequently involve the reordering of a sequence of characters. Consider, as an example, this combination of consonant plus marks that occurs in 1 Ch 13:13. This combination could be encoded in six different ways, and each would result in exactly the same visual representation and the same semantic meaning for the reader:

1. tet + dagesh + tsere + zaqef gadol
2. tet + dagesh + zaqef gadol + tsere
3. tet + tsere + dagesh + zaqef gadol
4. tet + tsere + zaqef gadol + dagesh
5. tet + zaqef gadol + dagesh + tsere
6. tet + zaqef gadol + tsere + dagesh

The SBL Hebrew font is able to correctly render any of these sequences (although most users familiar with Hebrew would agree that the dagesh should, logically and linguistically, precede the vowel and the cantillation mark, and most would also agree that the vowel should precede the cantillation mark). If you consider that any combination of consonant plus three marks can be encoded in six different ways, it is easy to realise how even a fairly short word of five or six consonants with all their marks could be encoded in many dozens of different ways. Normalisation is important because it provides a mechanism for all these possible permutations of mark ordering to be resolved to a single canonical order. This is most important when a text not only needs to be displayed but also needs to searched, sorted or spellchecked. If a search algorithm had to look for fifty or more possible and equivalent spellings of a single word, it would be extremely inefficient and slow. So normalisation is applied to reorder every equivalent sequence of characters into a single and consistent order.

The normalisation issue

This section explains an important problem in the encoding of Hebrew in the Unicode Standard. This problem will not affect all text, but can be a serious problem that users need to be aware of.
Normalisation is achieved by giving every mark a canonical combining class. This is a number that indicates how close to the base character (the consonant, in the case of Hebrew) each mark should be ordered and which marks may be reordered relative to each other. Some canonical combining classes contain only individual characters, indicating that these can always be reordered relative to characters with different classes; some classes contain multiple characters, indicating that these cannot be reordered relative to each other. If two or more characters are included in the same class, this means that their relative ordering is semantically meaningful and not equivalent; therefore, they must not be reordered or the meaning will be lost.

So far, so good, but when the Hebrew script was encoded in the Unicode Standard, combining classes were assigned in a way that failed to take into account the many peculiarities of Biblical texts. This has resulted in every Hebrew vowel being assigned its own canonical combining class, meaning that combinations of two vowels applied to a single consonant may be reordered during normalisation. This is not a problem for modern Hebrew spelling, but can be disastrous for Biblical texts due to textual conventions such as the tendency to record changing pronunciation by applying new vocalisation but preserving the original consonant structure of words. A good—and important—example is the change in pronunciation of יְרוּשָׁלֵם to יְרוּשָׁלַיִם, which required the Masoretes to add hiriq between the lamed and the final mem in order to approximate the new pronunciation as יְרוּשָׁלַיִם, while preserving the consonant structure of the ancient manuscripts. The final four characters of this word are correctly encoded in the order lamed + patah + hiriq + final mem, and the word is correctly displayed יְרוּשָׁלַיִם; however, the faulty canonical combining classes for patah and hiriq in Unicode cause hiriq to always precede patah when reordering due to normalisation is applied. The resulting sequence lamed + hiriq + patah + final mem is both textually incorrect and cannot be correctly displayed by the SBL Hebrew font: יְרוּשָׁלַיִם (note collision of vowels under lamed). Unicode normalisation can easily break Biblical Hebrew text.

The good news is that most software does not automatically apply normalisation, and software developers familiar with Biblical Hebrew are likely to be aware of the problem. There remains a risk, however, especially when documents are being exchanged between different platforms or published on the Internet, that a piece of software beyond the original author or editor’s control—a web browser on the receiving end of an electronic document, for example—may apply normalisation. This is, of course, not a font issue per se; it is a text encoding issue that can affect any document and result in tex-

For more thorough discussion of this example, see Tov, Emanuel. Textual criticism of the Hebrew Bible. 2nd edition, Minneapolis, 1992. p43.

Here are the Hebrew words from this example enlarged.
Correct, before normalisation: יְרוּשָׁלֵם
Incorrect, after normalisation: יְרוּשָׁלַיִם

Note that the Unicode Standard includes multiple normalisation forms, i.e. different ways of normalising text, and not all of these involve reordering of marks. The form that is of concern for Biblical Hebrew encoding is Normalisation Form C, which does reorder marks.
tual error, incorrect display, or both. To avoid these problems, the latest generation of Hebrew shaping engines, such as Uniscribe for Windows Vista and Office 2007, perform corrective mark ordering during rendering, to ensure that normalised sequences can be correctly rendered by fonts built to a common specification. This will remove many of the possible display problems associated with Unicode normalisation. However, there will remain a number of cases in which the results of normalisation may display ‘correctly’ but in which the meaning of the text is changed. For instance, if meteg is positioned at the right side of a vowel, it may be moved to the left side during normalisation: both positions will render correctly, and either is textually legitimate. So mechanisms are necessary to guard against some mark reordering during normalisation.

Software developers who need to apply normalisation to text to facilitate efficient searching can, of course, avoid the problems of the Unicode mark reordering by using a custom normalisation routine that does not reorder Hebrew vowels. Combining classes for such a custom routine, based on the mark ordering recommendations of the next section, are suggested in Appendix B, page 20. However, this does not address the issue of Unicode normalisation being applied ‘downstream’ of document creation, e.g. in web browsers.

The Unicode Technical Committee recommends the use of the Combining Grapheme Joiner (CGJ, U+034F) character as a control character to override mark reordering during normalisation, and the SBL Hebrew font has been updated to handle this mechanism. Using this mechanism, the problem case of mark reordering discussed on page 9 would be resolved by inserting the CGJ character between the patah and hiriq marks: \(\text{lamed} + \text{patah} + \text{CGJ} + \text{hiriq} + \text{final mem}\). The CGJ character can be inserted between any two adjacent mark characters that may be subject to reordering during normalisation, and will prevent this reordering from happening. Software developers are currently discussing ways in which this can be implemented automatically, so that the burden is not on document authors to identify situations in which mark reordering might occur and to manually insert CGJ at these places. It may be some time before such solutions become widely implemented.

If you are not concerned about mark reordering, e.g. because you are working in a completely closed text environment that does not employ normalisation or which uses a custom normalisation rather than the Unicode one, you do not need to worry about the CGJ character in this regard. If you are producing documents that may be published to the Internet or in other circumstances in which you do not have control of normalisation, CGJ provides a means for you to prevent unwanted reordering. If you are working with specific Bi-
ble passages, and are concerned about normalisation reordering of marks, you may find it helpful to check the chapter and verse to see what words may be affected in a way that changes the recommended mark order discussed in the following sections of this manual. To assist in this, we will be producing a document for each Bible book, identifying such words. These will be available from the SBL website and from http:www.tiro.com/resources/hebrew (in progress).

Note that not all mark reordering results in broken rendering or textual ambiguity. As noted above, some rendering engines may correct reordering during display, further reducing the number of instances in which CGJ needs to be inserted; the most important use of CGJ is in instances where the reordered text may render ‘cleanly’ but where textually important distinctions are lost, e.g. where a meteg moves from one side of a vowel mark to another.

As discussed in the previous section, the SBL Hebrew font can correctly display equivalent sequences of marks applied to consonants in different orders. The font cannot correctly display every possible permutation, especially when more than two marks are involved, but it seldom matters, for instance, whether a below vowel or above cantillation mark is applied to a consonant first. That said, there are many benefits to establishing consistent habits in ordering marks when creating documents. By following the recommended mark ordering outlined in the table on the next page, you will avoid entering sequences that cannot be correctly displayed—unless, of course, they are beyond the general ability of the SBL Hebrew font to render the Old Testament text—, and you will also make searching for Hebrew words and phrases easier for yourself and colleagues with whom you exchange electronic documents.

The basic principles of the mark ordering recommendation is that marks affecting the pronunciation of consonants are applied first; then the holam mark; then below marks, vowels and consonants, as they occur from right-to-left except the prepositive marks yetiv and dehi, which are applied after other low marks because they are positioned relative to them; then above marks, including metatextual marks such as the masora circle, as they occur from right-to-left except the postpositive marks pashta, telisha qetana and zinor. The table of marks on the next page shows the order in which they should be entered (note that this corresponds, also, to the recommendations for custom normalisation in Appendix B, page 21).
### Table 1

**Recommended mark ordering**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Base consonant</td>
</tr>
<tr>
<td>2</td>
<td>shin dot</td>
</tr>
<tr>
<td></td>
<td>sin dot</td>
</tr>
<tr>
<td>3</td>
<td>dagesh/mapiq</td>
</tr>
<tr>
<td>4</td>
<td>rafe</td>
</tr>
<tr>
<td>5</td>
<td>holam</td>
</tr>
<tr>
<td>6</td>
<td>Below marks from the following group as they occur from right-to-left:</td>
</tr>
<tr>
<td></td>
<td>sheva</td>
</tr>
<tr>
<td></td>
<td>hataf segol</td>
</tr>
<tr>
<td></td>
<td>hataf patah</td>
</tr>
<tr>
<td></td>
<td>hataf qamats</td>
</tr>
<tr>
<td></td>
<td>hiriq</td>
</tr>
<tr>
<td></td>
<td>tsere</td>
</tr>
<tr>
<td></td>
<td>segol</td>
</tr>
<tr>
<td></td>
<td>patah</td>
</tr>
<tr>
<td></td>
<td>qamats (gadol)</td>
</tr>
<tr>
<td></td>
<td>qamats qatan</td>
</tr>
<tr>
<td></td>
<td>qubuts</td>
</tr>
<tr>
<td></td>
<td>meteg⁴</td>
</tr>
<tr>
<td></td>
<td>atnah</td>
</tr>
<tr>
<td></td>
<td>atnah hafukh</td>
</tr>
<tr>
<td></td>
<td>tipeha</td>
</tr>
<tr>
<td></td>
<td>tevir</td>
</tr>
<tr>
<td></td>
<td>munah</td>
</tr>
<tr>
<td></td>
<td>mahapakh</td>
</tr>
<tr>
<td></td>
<td>merka</td>
</tr>
<tr>
<td></td>
<td>merka kefula</td>
</tr>
<tr>
<td></td>
<td>darga</td>
</tr>
<tr>
<td></td>
<td>yerah ben yomo</td>
</tr>
<tr>
<td></td>
<td>low punctum extra.²</td>
</tr>
<tr>
<td>7</td>
<td>Prepositive below marks from the following group:</td>
</tr>
<tr>
<td></td>
<td>yetiv</td>
</tr>
<tr>
<td></td>
<td>dehi</td>
</tr>
<tr>
<td>8</td>
<td>Above marks from the following group as they occur from right-to-left:</td>
</tr>
<tr>
<td></td>
<td>shalshellet</td>
</tr>
<tr>
<td></td>
<td>zaqef qatan</td>
</tr>
<tr>
<td></td>
<td>zaqef gadol</td>
</tr>
<tr>
<td></td>
<td>revia</td>
</tr>
<tr>
<td></td>
<td>zarqa</td>
</tr>
<tr>
<td></td>
<td>qarney para</td>
</tr>
<tr>
<td></td>
<td>gershayim</td>
</tr>
<tr>
<td></td>
<td>geresh muqdam</td>
</tr>
<tr>
<td></td>
<td>geresh</td>
</tr>
<tr>
<td></td>
<td>telisha gedola</td>
</tr>
<tr>
<td></td>
<td>iluy</td>
</tr>
<tr>
<td></td>
<td>qadma (azla)</td>
</tr>
<tr>
<td></td>
<td>ole</td>
</tr>
<tr>
<td></td>
<td>pazer</td>
</tr>
<tr>
<td></td>
<td>masora/number dot³</td>
</tr>
<tr>
<td></td>
<td>high punctum extra.²</td>
</tr>
<tr>
<td></td>
<td>masora circle⁴</td>
</tr>
<tr>
<td>9</td>
<td>Postpositive above marks from the following group:</td>
</tr>
<tr>
<td></td>
<td>segolta</td>
</tr>
<tr>
<td></td>
<td>pashta</td>
</tr>
<tr>
<td></td>
<td>telisha qetana</td>
</tr>
<tr>
<td></td>
<td>zinor</td>
</tr>
</tbody>
</table>

**Notes:**

1. See page 13 for specific information about meteg ordering.
2. See page 14 for specific information about puncta extraordinaria. Note that the high punctum is centered above the letter and above any other high marks; it should usually be ordered last of the marks in its group. Similarly the low punctum should usually be ordered after other below marks.
3. See page 15 for specific information about the masora/number dot.
4. Although it may visually be positioned slightly to the right of other above marks, the masora circle is usually ordered after any other marks in this group. See page 15 for specific information about masora circle handling.
The Hebrew mark *meteg* can appear in a variety of positions relative to other below marks, and all these are supported in the SBL Hebrew positioning lookups. However, not all are equally well supported in applications at the time of writing. This section explains how to encode sequences of consonant plus meteg with other marks to achieve different visual results, and shows how these will be ideally rendered.

The normal position of *meteg* relative to vowel marks is to the left, e.g. וּוַֽיִּמְצָא (1 Kings 1:3). Such a sequence is encoded consonant + vowel + meteg, that is, as the marks are ordered from right to left.

Much less commonly, *meteg* can appear to the left of a cantillation mark, e.g. עֲבָדִֽ֑ים (Ex 20:2). Again, this sequence is encoded as the marks are ordered from right to left: consonant + vowel + cantillation + meteg.

When *meteg* is applied to a consonant bearing a *hataf* vowel (*hataf segol, hataf patah or hataf qamats*), the default font rendering is also for the meteg to be positioned to the left of the vowel, e.g. הנהוֹתא (Ps 50:21). However, it is common for this mark combination to be displayed with a medial *meteg* positioned between the two parts of the *hataf* vowel, e.g. אלהים (2 Chr 32:15). This is handled by the SBL Hebrew font, using an OpenType ligature substitution, when the Zero Width Joiner (ZWJ, U+200D) control character is inserted between the *hataf* vowel and the *meteg*: consonant + *hataf* vowel + ZWJ + *meteg*.

Finally, there are those instances in which *meteg* needs to be placed to the right of a vowel. These are relatively common on the first consonant of a word in the *Biblia Hebraica Stuttgartensia* text, e.g. אַלְּהֵיכֶ֔ם (Ex 20:4), but may also occur in mid-word, e.g. הָלֹא (Deut 22:26). In such cases, *meteg* should be encoded before the vowel, i.e. as the marks are ordered from right to left: consonant + *meteg* + vowel. Note that this ordering should also be used in those rare cases where *meteg* occurs to the right of a *hataf* vowel, e.g. הָמְלָא (Psa 85:7).

Note that because of the canonical combining class assigned by Unicode to the *meteg* character, it may be subject to reordering during normalisation (see pages 8–11). For instance, the example from Exodus 20—֥֣ךָתַֽעֲשֶׂ֨ה־לְ—would be reordered during normalisation such that the *meteg* shifts to the left of the *patah*: ֥֣ךָתַֽעֲשֶׂ֨ה. If the relative position of meteg to a vowel is textually important, care must be taken to prevent reordering through automatic or manual insertion of the CGJ character, e.g. *bet + meteg + CGJ + patah.*
**Puncta extraordinaria**

This section explains the current recommendation for encoding the puncta extraordinaria, illustrating how they should appear if rendered correctly.


---

**Puncta extraordinaria** (extraordinary points) occur fifty-six times in the Old Testament text: fifty-three times above letters and three times below letters. The electronic edition of the *Biblia Hebraica Stuttgartensia* text notes that the function of these marks is not entirely clear, but it has variously been proposed that: a) the marks are merely emphasis and draw special attention to the theological implications of the word; b) the marks are early critical marks which indicate an omission or change that the scribes desired to make, but dared not; c) the marks represent drops of ink or even bits of dirt that were slavishly copied from one manuscript to the next; d) the marks indicate a special or unusual pronunciation of the word, or that the word should not be read at all; or e) some mixture of the above, on a case-by-case basis.

The most complex example of *puncta extraordinaria* is in Ps 27:13, in which *puncta* appear both above and below, and with other marks.

The lower punctum was previously encoded in SBL Hebrew using the generic combining dot below (U+0323). Documents produced with this earlier encoding will need to be updated.

**Atnah hafukh and qamats qatan**

This section explains two new characters in Unicode 4.1 and their implementation in this version of SBL Hebrew.

The *atnah hafukh* (U+05A2) character has been recently included in Unicode 4.1 for the benefit of those users who wish to encode an explicit distinction between this accent and *yerah ben yomo* (U+05AA). Many editions do not make this distinction, using the *yerah ben yomo* character and its glyph to represent both accents. Since this is a new character, with some disunification legacy issues (the form of the *yerah ben yomo* character used in many typefaces is actually that of the *atnah hafukh*), support in this version of SBL Hebrew is preliminary and subject to review: the character code is supported, but the glyph is identical to that used for *yerah ben yomo*.

The new *qamats qatan* (U+05C7) character makes it possible, if desired, to encode an explicit distinction between this vowel (short *qamats*) and the long *qamats* (U+05B8, *qamats gadol*). To visualise this distinction, following a convention adopted in some editions, the glyph for this new character is noticeably taller than that for *qamats gadol*. 
Masora/Number dot
This section explains the use of the upper dot mark in Masoretic notes and for Hebrew numerals.

Not to be confused with the high *puncta extraordinaria*, this high dot mark is used primarily in the context of Masoretic notes, as in this example from Num 32:24:

<table>
<thead>
<tr>
<th>בַּנְרֶ֣כֶם גֵּרָה</th>
<th>לֵשָׁ֣נִים הָֽלָּ֗הּ לֵזֶֽנִים֙</th>
</tr>
</thead>
<tbody>
<tr>
<td>אֱ</td>
<td>̇</td>
</tr>
</tbody>
</table>

Like the low *puncta extraordinaria* the masora/number dot is not encoded in the Unicode Standard as a specifically Hebrew character, so a generic combining dot above (U+0307) is used.

The same character is sometimes used to indicate a consonant used as a numeral in the traditional Hebrew numbering system: א=1, ב=2, ג=3 ... י=10, כ=20, ל=30, and so on, with nine consecutive letters of the alphabet used for units, tens and hundreds up to 400 (some sources identify final letter forms for 500–900).

This numbering system is extended in some reference books to counting in thousands by doubling the dots: א=1000, ב=2000, ג=3000, etc.. The double dot is encoded using the generic combining diaeresis character (U+0308).

Masora circle handling
This section explains how to encode and control the positioning of the masora circle.

The circle indicating a Masoretic note can occur over a consonant, between consonants, over a *maqaf*, or over a word space. Ideally, it should be centred over a word, but there is no way to specify this in Unicode combining mark encoding or OpenType Layout. Manual care must be taken with the positioning. The SBL Hebrew font, by default, places the masora circle above the preceding consonant, *maqaf* or word space, as on these examples from the first chapter of Genesis:

<table>
<thead>
<tr>
<th>מְרַחֶ֖פֶת</th>
<th>אֵל־מָק֣וֹם</th>
</tr>
</thead>
<tbody>
<tr>
<td>פְּרִ֞י</td>
<td>עֵ֣ץ</td>
</tr>
</tbody>
</table>

The masora glyph is centered on a zero-width, which means that if the default positioning is inhibited it will automatically be placed between the character to which it is applied and the next. The positioning can be inhibited by inserting the ZWNJ control character between the masora circle and the consonant, *maqaf* or word space to which it is applied. In the examples below, also from Genesis 1, the character sequence is *consonant + [vowel/cantillation mark(s)] + ZWNJ + masora circle*.

<table>
<thead>
<tr>
<th>מְרַחֶ֖פֶת</th>
<th>אֵל־מָק֣וֹם</th>
</tr>
</thead>
<tbody>
<tr>
<td>וַיְבָ֣רֶךְ</td>
<td>מִ֔ים</td>
</tr>
</tbody>
</table>

Note that, as shown in the last example from Genesis 1:28, the ZWNJ character can also be used to position a masora circle after a word break at the end of a line.
Holam male / vav haluma
This section explains the encoding of the vowel holam male, distinct from the consonant plus vowel sequence vav with holam haser.

Documents using the deprecated conventions of holam + vav = holam male / vav + holam = vav haluma will need to be updated.

Some texts—both manuscript and typographic—make a visual distinction in the position of the dot between a vav haluma, i.e. vav followed by a holam haser (1), and the independent vowel holam male (1). Some texts do not make this distinction, displaying both as i, and some electronic documents may even encode both graphemes the same way, relying on the reader’s expertise to distinguish them.

Earlier versions of the SBL Hebrew font used a mark+base ordering convention to distinguish holam male and vav haluma, but this has been deprecated because the holam male encoding was counter to basic Unicode rules about base+mark ordering. The correct encoding of this distinction has been the subject of much discussion in Unicode circles. The consensus, now formally enshrined in the standard, is that the character sequence vav + holam (U+05B9) is the correct encoding for holam male, since this is far more common than the relatively rare vav haluma. This was first implemented in version 1.12 of the SBL Hebrew font:

\[ \text{י} = \text{vav + holam (holam male)} \]

Version 1.5x of the font supports the new Unicode 5.0 combining mark character \text{HEBREW POINT HOLAM HAYER FOR VAV} (U+05BA). This character is explicitly intended to encode vav haluma and behaves relative to vav as the common holam dot to other consonants. This new character provides a robust encoding distinction for those users who wish to differentiate holam male and vav haluma as illustrated in the fourth and fifth words of this example from Genesis 4:13:

\[ נַעַמְר בֶּן אָֽלֶ֑ישָׁו יִהְוָֽה הַדָּ֣וֹלַת וְעָנֵ֖י מֹשֶֽׁא׃ \]

Nun hafukha—‘inverted nun’—is not a letter, but a form of special punctuation that occurs only a handful times in the Hebrew Bible (e.g. Numbers 10:35–36). Hebrew manuscripts show a number of different forms for this character, depending on the script style, only one of which actual resembles an inverted nun letter. This character is encoded in Unicode as the character U+05C6.

\[ \text{ך} = \text{nun hafukha} \]

In pointed texts, the inverted nun carries a dot above it. For this, the masora/number dot character (U+0307) should be used:

\[ \dot{ך} = \text{nun hafukha + masora/number dot} \]
Support for Dead Sea Scrolls transcription is new to the SBL Hebrew 1.5x general release. The font supports transcription conventions as employed in numerous works related to both the Dead Sea Scrolls and to other ancient manuscripts from the Judean Desert. The encoding for these conventions is as follows:

- `א` ring above U+030A
- `א` dot above U+0307
- `◦` white circle U+25E6
- `•` black circle U+2022
- `א` strikeout U+0336
- `{א}` braces U+007B / U+007D
- `[א]` brackets U+005B / U+005D

Example:

```
[שִׁנָּ֖ן] [שִׁנָּ֖ן
[בָּשְׁנָ֖ן
[בָּשְׁנָ֖ן
[לָטְרָ֖ף] [לָטְרָ֖ף
[לָטְרָ֖ף
[לָטְרָ֖ף
```

It should be noted that there is some discrepancy between how different applications handle display of parenthetical characters such as braces and brackets in bidirectional text layout. Care should be taken to check the results when text created in one application is imported into another application: it is possible for parenthetical punctuation to become reversed. In general, because most of the transcription characters are not explicitly right-to-left, care must be taken in formatting documents, correctly setting paragraph directionality, etc..

Most word processing and text layout applications have a ‘strikeout’ function that provides an easy means to transcribe crossed out letters without the need to use an additional character code after each letter. However, use of the combining mark U+0336 enables a plain text distinction to be made, which will reliably survive text exchange between different applications and platforms.

Input support for the Dead Sea Scrolls transcription characters is available from version 1.5 of the Tiro and SIL Biblical Hebrew keyboard drivers, available from the SBL website.
Support in non-Windows systems & applications

This section explains the current state of support for SBL Hebrew on Macintosh & Linux operating systems.

The term ‘font support’ can refer to a number of different things, from installability to text display to complex glyph rendering, so when wondering whether a particular operating system or application (or particular combination of OS and application) supports use of a font it is necessary to understand the limitations that can apply.

**Macintosh.** The Mac OS X operating system can natively install and render the SBL Hebrew ‘data-fork’ format Windows TrueType font. Older versions of the Mac operating system cannot natively install this format, and need a Mac-specific ‘resource-fork’ format font file. At present, no resource-fork version of the SBL Hebrew font is available, since there are so few Unicode enabled applications for older Mac operating systems. Note that some applications, particularly new versions of much Adobe software (on both Mac and Windows), can install fonts, including SBL Hebrew, directly in their own font folders and use their own rendering technology, bypassing system font handling; some of these applications may provide support on older versions of the Mac operating system.

The latest version of OS X, Tiger, implements some system level support for basic OpenType Layout features, converting them on the fly to Apple’s own AAT format features. However, this does not include support for complex script layout, so the all-important mark attachment is not supported at the system level. This means that applications relying on standard system calls for text processing will not be able to handle Biblical Hebrew text using SBL Hebrew, even though they may support Unicode text encoding and basic right-to-left layout.

Applications that use their own text layout engines, designed to work with OpenType fonts, offer the most hope for Macintosh users. At the time of writing, none of these applications offer the same high level of Biblical Hebrew rendering as Microsoft Office 2003 on Windows, but some are gradually getting close.

Mellel is a word processing application for OS X that employs OpenType Layout natively using its own text engine. Biblical Hebrew test results from Mellel are encouraging, with almost all tested letter+mark sequences rendering correctly. One problem in the current release is that there is no way to turn off visual display of control characters, so if you insert e.g. the Zero Width Joiner character in a sequence Mellel will display a small mark above the letters. [In Microsoft Word on Windows, display of control characters is off by default, but can be turned on to facilitate editing; this behaviour is preferred.]

XeTeX is a typesetting system that marries Donald Knuth’s TeX system with Unicode text encoding and OpenType Layout. Only a
small number of users appear to be using this for Biblical Hebrew, but the examples they have submitted are impressive and indicate that XeTeX provides some of the best support for Hebrew outside of the Windows version of Word and other Office applications. XeTeX makes use of the ICU layout library discussed below.

Linux. As with many other aspects of open source computing, the level of support for Biblical Hebrew on Linux depends very much on specific distribution, library and application configurations. This section of the manual is necessarily short simply because we have limited experience in this area, and have only recently begun to explore it.

Users of the SBL Hebrew font who are interested in Hebrew rendering on Linux systems are encouraged to investigate the ICU and Pango, two code libraries that provide support for Unicode text processing including OpenType Layout support. Tests of current version of Pango, compiled into the Firefox web browser, show correct positioning of single accent to single vowel combinations, but errors in second accent and accent-to-accent relative positioning. Results from applications using ICU seem better.

Linux-based users of the SBL Hebrew font are particularly encouraged to join the MSN user community, and to share tips and tricks for working with Biblical Hebrew text on this platform.

Conclusion

As should be clear from this manual, the SBL Hebrew font is on the ‘cutting edge’ of text processing and layout technology, and much software still needs to catch up to the font in order to perfectly render all aspects of Biblical Hebrew texts. That said, in the seventeen months since the last version of this manual was written, there have been encouraging advances, especially on the Mac and Linux systems.

During development of the font, key software developers such as the Microsoft development lead responsible for the Uniscribe Hebrew engine have consulted on the best way to encode and shape Biblical Hebrew and have provided beta versions of upcoming software releases for testing. This consultation has enabled us to make a font that outperforms all previous Hebrew text rendering, and which will enable Biblical scholars to create, edit and exchange documents between the increasing number of systems and applications that support Unicode text encoding and OpenType Layout.
Appendix A
End User License Agreement

This license agreement explains the rights and responsibilities that you accept when you install the SBL Hebrew font on your computer. Please take a few minutes to review this.

1. The digitally encoded machine readable font software for producing the typefaces licensed to you is the property of Tiro Typeworks. It is licensed to you for use under the terms of this end user license agreement. If you have any questions about this license agreement, or have a need to use the font software in a way not covered by this agreement, please write to license@tiro.com.

2. You may use this font software free of charge for all non-commercial purposes. If you wish to obtain a license for commercial use of this font software, please contact the Society of Biblical Literature at sblexec@sbl-site.org, or write to license@tiro.com. Fees for commercial licenses are at the individual discretion of the Society of Biblical Literature and Tiro Typeworks.

3. You may redistribute this font software free of charge as long as the software is unmodified, all copyright and trademark notices are intact, and the font contains or is accompanied by a copy of this license agreement. You may not charge any fee for the distribution of this font software or alter the terms of this license agreement.

4. You may decompile and modify this font software for non-commercial and personal use by you as an individual or internal use within your organisation. Tiro Typeworks maintains copyright to all derivative fonts in any format. You may not delete, edit or add to copyright, trademark or license information in the font. You may not change the embedding bit. You may not redistribute any modified version of the font software, either free of charge or for a fee. Copies of modified fonts should be submitted to Tiro Typeworks (license@tiro.com) and to the Society of Biblical Literature (sblexec@sbl-site.org), along with any relevant documentation. Tiro Typeworks reserves the right to incorporate any such changes into its own fonts.

5. You may embed the font software in non-commercial electronic documents, including but not limited to web pages and e-books. Font embedding must respect the embedding bit in the font, which must not be changed. The embedding bit for this font software is set to ‘Editable Embedding’, meaning that documents containing this font software may be viewed, printed and edited, but the embedded font may not be installed on the recipient user’s system.

6. All other rights are reserved by Tiro Typeworks, except as otherwise designated in contract between Tiro Typeworks and the Society of Biblical Literature.

7. Neither Tiro Typeworks nor the Society of Biblical Literature warrant the performance or results you may obtain by using this font software. Excepting any warranty, condition, representation or term that cannot or may not be excluded or limited by law applicable to you in your jurisdiction, Tiro Typeworks and the Society of Biblical Literature make no warranties, conditions, representations, or terms (express or implied whether by statute, common law, custom, usage or otherwise) as to any matter including, without limitation, noninfringement of third party rights, merchantability, integration, satisfactory quality, or fitness for any particular purpose.

8. Neither Tiro Typeworks nor the Society of Biblical Literature accept any liability for injury, death, financial loss or damage to person or property (including computer hardware, software or data) resulting from the use of this font software.

9. The act of installing this font software on any computer system constitutes acceptance of the terms of this license agreement, without exception.
Appendix B
Custom combining classes.

As explained on pages 8–11, Unicode normalisation may break Biblical Hebrew text by reordering marks that should not be reordered. This appendix provides alternate combining classes to use in custom normalisation routines. Nota bene: these alternate combining classes are outside of any recognised standard, and text produced using custom normalisations may still be subject to other normalisations in software beyond the author’s control (e.g. web browsers). This list is provided purely as a suggestion, and no guarantee is made that it will be supported as is in software developed by SBL, Tiro Typeworks, or any of their project partners.

See the discussion on pages 8–11 for more information about normalisation issues, and the Unicode Standard for details of standard normalisation and combining classes. Below are suggested canonical combining classes to use in custom normalisation routines for Biblical Hebrew text. As in standard normalisation, the expectation is that only marks in different combining classes will be reordered during normalisation; marks with the same combining class value should not be reordered. The lower the combining class value, the closer to the base character (e.g. Hebrew consonant) the mark should be ordered. In this table, the recommended combining class is provided first, then the existing Unicode combining class in grey, and then the Unicode codepoint and a descriptive name.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>24</td>
<td>U+05C1</td>
<td>Point Shin Dot</td>
</tr>
<tr>
<td>11</td>
<td>25</td>
<td>U+05C2</td>
<td>Point Sin Dot</td>
</tr>
<tr>
<td>21</td>
<td>21</td>
<td>U+05BC</td>
<td>Point Dagesh or Mapiq</td>
</tr>
<tr>
<td>23</td>
<td>23</td>
<td>U+05BF</td>
<td>Point Rafe</td>
</tr>
<tr>
<td>27</td>
<td>19</td>
<td>U+05B9</td>
<td>Point Holam</td>
</tr>
<tr>
<td>27</td>
<td>19</td>
<td>U+05BA</td>
<td>Point Holam Haser for Vav</td>
</tr>
<tr>
<td>220</td>
<td>220</td>
<td>U+05C5</td>
<td>Lower Punctum</td>
</tr>
<tr>
<td>220</td>
<td>220</td>
<td>U+0591</td>
<td>Accent Atnah</td>
</tr>
<tr>
<td>220</td>
<td>220</td>
<td>U+05A2</td>
<td>Accent Atnah Hafukh</td>
</tr>
<tr>
<td>220</td>
<td>220</td>
<td>U+0596</td>
<td>Accent Tipeha</td>
</tr>
<tr>
<td>220</td>
<td>220</td>
<td>U+05B9</td>
<td>Accent Tevir</td>
</tr>
<tr>
<td>220</td>
<td>220</td>
<td>U+05A3</td>
<td>Accent Munah</td>
</tr>
<tr>
<td>220</td>
<td>220</td>
<td>U+05A4</td>
<td>Accent Mahapakh</td>
</tr>
<tr>
<td>220</td>
<td>220</td>
<td>U+05A5</td>
<td>Accent Merkha</td>
</tr>
<tr>
<td>220</td>
<td>220</td>
<td>U+05A6</td>
<td>Accent Merkha Kefula</td>
</tr>
<tr>
<td>220</td>
<td>220</td>
<td>U+05A7</td>
<td>Accent Darga</td>
</tr>
<tr>
<td>220</td>
<td>220</td>
<td>U+05AA</td>
<td>Accent Yerah Ben Yomo</td>
</tr>
<tr>
<td>220</td>
<td>10</td>
<td>U+05B0</td>
<td>Point Sheva</td>
</tr>
<tr>
<td>220</td>
<td>11</td>
<td>U+05B1</td>
<td>Point Hataf Segol</td>
</tr>
<tr>
<td>220</td>
<td>12</td>
<td>U+05B2</td>
<td>Point Hataf Patah</td>
</tr>
<tr>
<td>220</td>
<td>13</td>
<td>U+05B3</td>
<td>Point Hataf Qamats</td>
</tr>
<tr>
<td>220</td>
<td>14</td>
<td>U+05B4</td>
<td>Point Hiriq</td>
</tr>
<tr>
<td>220</td>
<td>15</td>
<td>U+05B5</td>
<td>Point Tsere</td>
</tr>
<tr>
<td>220</td>
<td>16</td>
<td>U+05B6</td>
<td>Point Segol</td>
</tr>
<tr>
<td>220</td>
<td>17</td>
<td>U+05B7</td>
<td>Point Patah</td>
</tr>
<tr>
<td>220</td>
<td>18</td>
<td>U+05B8</td>
<td>Point Qamats</td>
</tr>
<tr>
<td>220</td>
<td>18</td>
<td>U+05C7</td>
<td>Point Qamats Qatan</td>
</tr>
<tr>
<td>220</td>
<td>20</td>
<td>U+05BB</td>
<td>Point Qubuts</td>
</tr>
<tr>
<td>220</td>
<td>22</td>
<td>U+05BD</td>
<td>Point Meteg</td>
</tr>
<tr>
<td>222</td>
<td>222</td>
<td>U+059A</td>
<td>Accent Yetiv</td>
</tr>
<tr>
<td>U+05AD</td>
<td>Accent Dehi</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+05C4</td>
<td>Upper Punctum</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+0593</td>
<td>Accent Shalshelet</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+0594</td>
<td>Accent Qa'ef Qat'an</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+0595</td>
<td>Accent Qa'ef Gadol</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+0597</td>
<td>Accent Reviya</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+0598</td>
<td>Accent Zarqa</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+059F</td>
<td>Accent Qarnei Parva</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+059E</td>
<td>Accent Gershayim</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+059D</td>
<td>Accent Geresh Muqdam</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+059C</td>
<td>Accent Geresh</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+0592</td>
<td>Accent Segolta</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+05A0</td>
<td>Accent Telisha Gedola</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+05AC</td>
<td>Accent Iluy</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+05A8</td>
<td>Accent Qadma</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+05AB</td>
<td>Accent Ole</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+05AF</td>
<td>Mark Masora Circle</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+05A1</td>
<td>Accent Pazer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+0307</td>
<td>Mark Number/Masora Dot</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+05AE</td>
<td>Accent Zinor</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+05A9</td>
<td>Accent Telisha Qetana</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U+0599</td>
<td>Accent Pashta</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The makers of the SBL Hebrew font would like to thank the individuals and organisations who participated in ad hoc discussions to determine an appropriate mark ordering for normalised Biblical Hebrew, especially Peter Constable (SIL International) and Eli Evans (Libronix/Logos). Other contributors included Joan Wardell (SIL International), Patrick Durusau (SBL), Ralph Hancock, Paul Nelson (Microsoft), Bob Pritchett (Libronix/Logos), & Kent Richards (SBL).
Appendix C
Resource links

This appendix provides URLs for font and document downloads and online resources from the Society of Biblical Literature and other parties.

http://www.sbl-site.org/Resources/
The latest version of the SBL Hebrew font, this manual, and other information relating specifically to SBL Hebrew; downloadable Windows keyboard drivers for input of Biblical Hebrew text.

http://www.sbl-site.org/
Society of Biblical Literature website.

http://www.tiro.com/resources/hebrew
Tiro Typeworks resources for Biblical Hebrew, including normalisation example documents. [In development.]

http://www.unicode.org/
The Unicode Consortium website. Includes latest version of Unicode Standard and resources.

http://www.microsoft.com/typography/specs/default.htm
OpenType specification and Microsoft Hebrew font specification.

http://www.microsoft.com/typography/developers/opentype/default.htm
‘Windows Glyph Processing’ article by John Hudson; an introduction to OpenType font and Unicode script processing on Windows.

https://www.mellel.com/
Mellel is a powerful multilingual word processor for Mac OS X. It’s handling for Biblical Hebrew is not yet perfect, but it is very good.

Download site for the open source XeTeX page layout application for Mac OS X.

http://www.winsoft.fr/
Winsoft make the Middle East versions of Adobe software, including InDesign ME, which was used to typeset this manual.

International Components for Unicode: an open source set of libraries for Unicode support, software internationalization and globalization. Provides excellent OpenType Layout support.

http://www.pango.org/
An open-source framework for Unicode text layout and rendering.