# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>License and disclaimer</td>
<td>3</td>
</tr>
<tr>
<td><strong>1 Introduction</strong></td>
<td>3</td>
</tr>
<tr>
<td>1.1 arabluatex is for LuaLaTeX</td>
<td>5</td>
</tr>
<tr>
<td><strong>2 The basics of arabluatex</strong></td>
<td>5</td>
</tr>
<tr>
<td>2.1 Activating arabluatex</td>
<td>5</td>
</tr>
<tr>
<td>Font setup</td>
<td>5</td>
</tr>
<tr>
<td>2.2 Options</td>
<td>6</td>
</tr>
<tr>
<td>2.2.1 Classic contrasted with modern typesetting of Arabic</td>
<td>6</td>
</tr>
<tr>
<td>2.3 Typing Arabic</td>
<td>8</td>
</tr>
<tr>
<td>2.3.1 Local options</td>
<td>9</td>
</tr>
<tr>
<td><strong>3 Standard ArabTEX input</strong></td>
<td>9</td>
</tr>
<tr>
<td>3.1 Consonants</td>
<td>9</td>
</tr>
<tr>
<td>3.2 Additional characters</td>
<td>11</td>
</tr>
<tr>
<td>3.3 Vowels</td>
<td>11</td>
</tr>
<tr>
<td>3.3.1 Long vowels</td>
<td>11</td>
</tr>
<tr>
<td>3.3.2 Short vowels</td>
<td>12</td>
</tr>
<tr>
<td><strong>4 arabluatex in action</strong></td>
<td>13</td>
</tr>
<tr>
<td>4.1 The vowels and diphthongs</td>
<td>13</td>
</tr>
<tr>
<td>Short vowels</td>
<td>13</td>
</tr>
<tr>
<td>‘alif maqṣūrah</td>
<td>13</td>
</tr>
<tr>
<td>‘alif tawvisum</td>
<td>14</td>
</tr>
<tr>
<td>‘alif maḏḏah</td>
<td>14</td>
</tr>
<tr>
<td>Silent ‘ṣ</td>
<td>14</td>
</tr>
<tr>
<td>‘Amr</td>
<td>14</td>
</tr>
<tr>
<td>Silent ‘ḍ</td>
<td>14</td>
</tr>
<tr>
<td>tanwīn</td>
<td>14</td>
</tr>
<tr>
<td>4.2 Other orthographic signs</td>
<td>15</td>
</tr>
<tr>
<td>‘alif marbuṭah</td>
<td>15</td>
</tr>
<tr>
<td>hamzah</td>
<td>15</td>
</tr>
<tr>
<td>maddah</td>
<td>17</td>
</tr>
<tr>
<td>saddah</td>
<td>17</td>
</tr>
<tr>
<td>The definite article and the ‘alif</td>
<td>18</td>
</tr>
<tr>
<td>’l-waṣil‘</td>
<td>18</td>
</tr>
<tr>
<td>4.3 Special orthographies</td>
<td>21</td>
</tr>
<tr>
<td>The name of God</td>
<td>21</td>
</tr>
<tr>
<td>The conjunctive</td>
<td>21</td>
</tr>
<tr>
<td>4.4 Quoting</td>
<td>22</td>
</tr>
<tr>
<td>novoc</td>
<td>22</td>
</tr>
<tr>
<td>voc</td>
<td>23</td>
</tr>
<tr>
<td>fullvoc</td>
<td>23</td>
</tr>
<tr>
<td>4.4.1 Quoting the hamzah</td>
<td>23</td>
</tr>
<tr>
<td>4.5 The ‘pipe’ character (</td>
<td>)</td>
</tr>
<tr>
<td>4.6 Putting back on broken contextual analysis rules</td>
<td>24</td>
</tr>
<tr>
<td>4.7 Stretching characters: the lātīf</td>
<td>26</td>
</tr>
<tr>
<td>4.8 Digits</td>
<td>26</td>
</tr>
<tr>
<td>4.8.1 Numerical figures</td>
<td>26</td>
</tr>
<tr>
<td>4.8.2 The abjad</td>
<td>26</td>
</tr>
<tr>
<td>4.9 Additional characters</td>
<td>27</td>
</tr>
<tr>
<td>4.10 Arabic emphasis</td>
<td>28</td>
</tr>
<tr>
<td>4.10.1 Underlining words or numbers</td>
<td>28</td>
</tr>
<tr>
<td><strong>5 Arabic poetry</strong></td>
<td>28</td>
</tr>
<tr>
<td>Scaling and distortion of characters</td>
<td>31</td>
</tr>
<tr>
<td>Footnotes</td>
<td>31</td>
</tr>
<tr>
<td>Line numbering</td>
<td>31</td>
</tr>
<tr>
<td>5.1 Example</td>
<td>31</td>
</tr>
<tr>
<td><strong>6 Special applications</strong></td>
<td>32</td>
</tr>
<tr>
<td>Linguistics</td>
<td>32</td>
</tr>
<tr>
<td>Brackets</td>
<td>33</td>
</tr>
<tr>
<td>Additional Arabic marks</td>
<td>33</td>
</tr>
<tr>
<td>The ‘zero width joiner’ character (U+200D)</td>
<td>35</td>
</tr>
<tr>
<td>6.1 The Qur’an</td>
<td>35</td>
</tr>
<tr>
<td>Caveat</td>
<td>36</td>
</tr>
<tr>
<td><strong>7 Color</strong></td>
<td>36</td>
</tr>
</tbody>
</table>
Abstract

This package provides for LuaLATEX an ArabTEX-like interface to generate Arabic writing from an ASCII transliteration. It is particularly well-suited for complex documents such as technical documents or critical editions where a lot of left-to-right commands intertwine with Arabic writing. arabluatex is able to process any ArabTEX input notation. Its output can be set in the same modes of vocalization as ArabTEX, or in different roman transliterations. It further allows many typographical refinements. It will eventually interact with some other packages yet to come to produce from .tex source files, in addition to printed books, TEI xml compliant critical editions and/or lexicons that can be searched, analyzed and correlated in various ways.
1 Introduction

In comparison to Prof. Lagally’s outstanding ArabTeX,\(^1\) ArabLuaTeX is at present nothing more than a modest piece of software. Hopefully—if I may say so—it will eventually provide all of its valuable qualities to the LuaLaTeX users.

\(^1\)See http://ctan.org/pkg/arabtex


arabtex dates back to 1992. As far as I know, it was then the first and only way to typeset Arabic texts with TeX and \LaTeX. To achieve that, arabtex provided—and still does—an Arabic font in \textit{Nas\={i}h} style and a macro package that defined its own input notation which was, as the author stated, “both machine, and human, readable, and suited for electronic transmission and e-mail communication”.\footnote{Lagally (2004, p. 2).} Even if the same can be said about Unicode, Arab\TeX ASCII input notation still surpasses Unicode input, in my opinion, when it comes to typesetting complex documents, such as scientific documents or critical editions where footnotes and other kind of annotations can be particularly abundant. It must also be said that most text editors have trouble in displaying Arabic script connected with preceding or following \LaTeX commands: it often happens that commands seem misplaced, not to mention punctuation marks, or opening or closing braces, brackets or parentheses that are unexpectedly displayed in the wrong direction. Of course, some text editors provide ways to get around such difficulties by inserting invisible Unicode characters, such as LEFT-TO-RIGHT or RIGHT-TO-LEFT MARKS (U+200E, U+200F), RTL/LTR “embed” characters (U+202B, U+202A) and RLO/LRO “bidi-override” characters (U+202E, U+202D).\footnote{Gáspár Sinai’s Yudit probably has the best Unicode support. See \url{http://www.yudit.org}.} Nonetheless, it remains that inserting all the time these invisible characters in complex documents rapidly becomes confusing and cumbersome.

The great advantage of Arab\TeX notation is that it is immune from all these difficulties, let alone its being clear and straightforward. One also must remember that computers are designed to process code. Arab\TeX notation is a way of encoding Arabic language, just as TeX “mathematics mode” is a way of processing code to display mathematics. As such, not only does it allow greater control over typographical features, but it also can be processed in several different ways: so without going into details, depending on one’s wishes, Arab\TeX input can be full vocalized Arabic (\textit{scriptio plena}), vocalized Arabic or non-vocalized Arabic (\textit{scriptio defectiva}); it further can be transliterated into whichever romanization standard the user may choose.

But there may be more to be said on that point, as encoding Arabic also naturally encourages the coder to vocalize the texts—without compelling him to do so, of course. Accurate coding may even have other virtuous effects. For instance, hyphens may be used for tying particles or prefixes to words, or to mark inflectional endings, and so forth. In other words, accurate coding produces accurate texts that can stand to close grammatical scrutiny and to complex textual searches as well.

Having that in mind, I started arabluatex. With the help of Lua, it will eventually interact with some other packages yet to come to produce from .tex source files, in addition to printed books, TEI xml compliant critical editions and/or lexicons that can be searched, analyzed and correlated in various ways.
1.1 **arabluatex is for LuaL\TeX**

It goes without saying that arabluatex requires LuaL\TeX. \LaTeX\ and \LaTeXe\ have arabtex, and \XeLaTeX\ has arabxetex. Both of them are much more advanced than arabluatex, as they can process a number of different languages, whereas arabluatex can process only Arabic for the time being. More languages will be included in future releases of arabluatex.

In comparison to arabxetex, arabluatex works in a very different way. The former relies on the TECkit engine which converts Arab\TeX input on the fly into Unicode Arabic script, whereas the latter passes Arab\TeX input on to a set of Lua functions. At first, \LaTeX \ commands are taken care of in different ways: some, as \texttt{\emph}, \texttt{\textbf} and the like are expected to have Arabic text as arguments, while others, as \texttt{\LR}, for “left-to-right text”, are not. Then, once what is Arabic is carefully separated from what is not, it is processed by other Lua functions which rely on different sets of correspondence tables to do the actual conversion in accordance with one’s wishes. Finally, Lua returns to \TeX the converted strings—which may in turn contain some other Arab\TeX input yet to be processed—for further processing.

2 **The basics of arabluatex**

2.1 **Activating arabluatex**

arabluatex is loaded the usual way:

```latex
\usepackage{arabluatex}
```

The only requirement of arabluatex is LuaL\TeX; it will complain if the document is compiled with another engine. That aside, arabluatex does not load packages such as polyglossia. Although it can work with polyglossia, it does not require it.

**Font setup** Any Arabic font can be defined to be used with arabluatex. For example, assuming that fontspec is loaded, this line may be inserted in the preamble, just above the line that loads arabluatex:

```latex
\newfontfamily{arabicfont}{⟨fontname⟩}[Script=Arabic]
```

where ⟨fontname⟩ is the standard name of the Arabic font to be used.

By default, if no Arabic font is selected, arabluatex will issue a warning message and attempt to load the Amiri font\footnote{To date, both packages support Arabic, Maghribi, Urdu, Pashto, Sindhi, Kashmiri, Uighuric and Old Malay; in addition to these, arabtex also has a Hebrew mode, including Judeo-Arabic and Yiddish.} like so:—

```latex
\newfontfamily{arabicfont}{Amiri}[Script=Arabic]
```

\footnote{Hosny (2017).}
Rem. By default Amiri places the kasrah in combination with the taśdīd below the consonant, like so: ِّـ. That is correct, as at least in the oldest manuscripts ّـِ may stand for َّـَ as well as ِّـ. See Wright (1896, i. 14 C–D). The placement of the kasrah above the consonant may be obtained by selecting the ss05 feature of the Amiri font, like so:

\newfontfamily\arabicfont{Amiri}[Script=Arabic,RawFeature={+ss05}]

Other Arabic fonts may behave differently.

2.2 Options

arabluatex may be loaded with five global options, the first four of which are mutually exclusive and may be overridden at any point of the document (see below section 2.3.1 on page 9):

voc

In this mode, which is the one selected by default, every short vowel written generates its corresponding diacritical mark: dammah (ُّـ), fathah (َـ) and kasrah (ِّـ). If a vowel is followed by נ, viz. ⟨uN, aN, iN⟩, then the corresponding tanwīn (ٌـ, اًـ, ًة, ىًَـ or ٍـ) is generated. Finally, ⟨a, a, i⟩ at the commencement of a word indicate a “connective ‘alif” (‘alifu َل-وَسْلِي), but voc mode does not show the waṣlah above the ‘alif; instead, the accompanying vowel may be expressed at the beginning of a sentence (ِاَاُا).

fullvoc

In addition to what the voc mode does, fullvoc expresses the sukūn and the waṣlah.

novoc

None of the diacritics is showed in novoc mode, unless otherwise specified (see “quoting” technique below section 4.4 on page 22).

trans

This mode transliterates the ArabTgX input into one of the accepted standards. At present, three standards are supported (see below section 8 on page 41 for more details):

dmg Deutsche Morgenländische Gesellschaft, which is selected by default;
loc Library of Congress;
arabica Arabica.

More standards will be included in future releases of arabluatex.

export = true|false

This option acts as a named argument and does not need a value as it defaults to true if it is used. It enables arabluatex to produce a duplicate of the original .tex source file in which all ascn strings are replaced with Unicode equivalents. See below section 12 on page 57 for more information.

2.2.1 Classic contrasted with modern typesetting of Arabic

By default, arabluatex typesets Arabic in a classic, traditional style the most prominent features of which are the following:

6See the documentation of amiri, Hosny (2017, p. 6).
– ‘Classic’ maddah: when ‘alif and hamzah accompanied by a simple vowel or tanwīn is preceded by an ‘alif of prolongation (ا), then a mere hamzah is written on the line, and a maddah is placed over the ‘alif, like so:—

\[\text{samA'uN} \quad \text{samā'} \text{un}, \ J\text{a'a} \quad \text{gā'a}, \ yatasA'al\text{Una} \quad \text{yatasā'alūna}\]

(see on page 17 for further details).

– The euphonic tašdīd is generated (see on page 17).

– In fullvoc mode, the sukūn is expressed.

– In such words as اًءيَش, اًءمِظ and the like, the hamzah alone is not written over the letter yāʾ with no diacritical points below as in اًئيَش, اًئمِظ, but over a horizontal stroke placed in the continuation of the preceding letter.

Please note that only few Arabic fonts provide such contrivances. In case this feature is not supported by some Arabic font, it is advisable to use \SetArbEasy.

Such refinements as ‘classic’ maddah may be discarded by the \SetArbEasy command, either globally in the preamble or locally at any point of the document. The difference between \SetArbEasy and its ‘starred’ version \SetArbEasy* is that the former keeps the sukūn that is generated by the fullvoc mode, while the latter further takes it away. Default ‘classic’ rules may be set back at any point of the document with the \SetArbDflt command. Assimilation rules laid on item (b) on page 18 may also be applied by the ‘starred’ version of this command \SetArbDflt* either in the preamble or at any point of the document. 8 Examples follow:—

(a) \SetArbDflt:

i. voc وَمَّاتَ أَسَسَقَّا قَبْلَ أَن يُمَّ كَانَهُ فِي نَجْوُمِ السَّمَاء

ii. fullvoc وَمَّاتَ أَسَسَقَّا قَبْلَ أَن يُمَّ كَانَهُ فِي نَجْوُمِ السَّمَاء

iii. trans wa-māta 'stisqā' an yutimma kitāba-hu fī nuğūm 's-samā'

(b) \SetArbDflt*:

i. voc وَمَّاتَ أَسَسَقَّا قَبْلَ أَن يُمَّ كَانَهُ فِي نَجْوُمِ السَّمَاء

ii. fullvoc وَمَّاتَ أَسَسَقَّا قَبْلَ أَن يُمَّ كَانَهُ فِي نَجْوُمِ السَّمَاء

iii. trans wa-māta 'stisqā' an yutimma kitāba-hu fī nuğūm 's-samā'

(c) \SetArbEasy:

i. voc وَمَّاتَ أَسَسَقَّا قَبْلَ أَن يُمَّ كَانَهُ فِي نَجْوُمِ السَّمَاء

ii. fullvoc وَمَّاتَ أَسَسَقَّا قَبْلَ أَن يُمَّ كَانَهُ فِي نَجْوُمِ السَّمَاء

iii. trans wa-māta 'stisqā' an yutimma kitāba-hu fī nuğūm 's-samā'

Note that in old mss. such forms as اًءمَّسلاِموُجُنيِفُهَباَتِكَّمِتُينَأَلبَقًءآَقْسِتْسٱَتاَمَو are also found; see Wright (1896, i. 24 D).

For an example, see section 5.1 on page 31.
2.3 Typing Arabic

Once arabluatex is loaded, a \arb{⟨Arabic text⟩} command is available for inserting Arabic text in paragraphs, like so:—

From Wright (1896, i. 1 A):— Arabic, like Hebrew and Syriac, is written and read from right to left. The letters of the alphabet (\arb{.hurUf-u 'l-hijA'-i}, \arb{.hurUf-u 'l-tahajjI}, \arb{.hurUf-u 'l-hijA'iyyaT-u}, or \arb{.hurUf-u 'l-mu`jam-i}) are twenty-eight in number and are all consonants, though three of them are also used as vowels (see § 3).

The following example comes from Wright (1896, i. 213 C):—

Running paragraphs of Arabic text should rather be placed inside an Arabic environment.
2.3.1 Local options

As seen above in section 2.2 on page 6, arabluatex may be loaded with four mutually exclusive global options: voc (which is the default option), fullvoc, novoc and trans. Whatever choice has been made globally, it may be overridden at any point of the document, as the \arb command may take any of the voc, fullvoc, novoc or trans modes as optional argument, like so:

```latex
\arb[voc]{{Arabic text}};  
\arb[fullvoc]{{Arabic text}};  
\arb[novoc]{{Arabic text}};  
\arb[trans]{{Arabic text}}.
```

The same optional arguments may be passed to the environment arab: one may have \begin{arab}[(mode)] \ldots \end{arab}, where (mode) may be any of voc, fullvoc, novoc or trans.

3 Standard Arab\TeX input

3.1 Consonants

Table 1 gives the Ar\TeX equivalents for all of the Arabic consonants.
<table>
<thead>
<tr>
<th>Letter</th>
<th>Transliteration(^9)</th>
<th>ArabTEX notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>'u, 'a, 'i</td>
<td>'u or 'a or 'i</td>
</tr>
<tr>
<td>t</td>
<td>'u, 'a, 'i</td>
<td></td>
</tr>
<tr>
<td>d</td>
<td>'u, 'a, 'i</td>
<td></td>
</tr>
<tr>
<td>r</td>
<td>'u, 'a, 'i</td>
<td>'u or 'a or 'i</td>
</tr>
<tr>
<td>z</td>
<td>'u, 'a, 'i</td>
<td></td>
</tr>
<tr>
<td>s</td>
<td>'u, 'a, 'i</td>
<td></td>
</tr>
<tr>
<td>ā</td>
<td>'u, 'a, 'i</td>
<td></td>
</tr>
<tr>
<td>f</td>
<td>'u, 'a, 'i</td>
<td></td>
</tr>
<tr>
<td>q</td>
<td>'u, 'a, 'i</td>
<td></td>
</tr>
<tr>
<td>k</td>
<td>'u, 'a, 'i</td>
<td></td>
</tr>
<tr>
<td>l</td>
<td>'u, 'a, 'i</td>
<td></td>
</tr>
<tr>
<td>m</td>
<td>'u, 'a, 'i</td>
<td></td>
</tr>
<tr>
<td>n</td>
<td>'u, 'a, 'i</td>
<td></td>
</tr>
<tr>
<td>h</td>
<td>'u, 'a, 'i</td>
<td></td>
</tr>
<tr>
<td>w</td>
<td>'u, 'a, 'i</td>
<td></td>
</tr>
<tr>
<td>y</td>
<td>'u, 'a, 'i</td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>'u, 'a, 'i</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Standard ArabTEX (consonants)

**Rem. a.** Please note that in all cases of elision, the 'alif 'l-waslī is expressed only by the vowel that accompanies the omitted hamzah: ⟨u, a, i⟩ as in wa-ʾnhazama. For more details on the definite article and the 'alif 'l-waslī see section 4.2 on page 18.

\(^9\)See below section 8 on page 41.

\(^{10}\)See below, Rem. a. For 'alif as a consonant, see Wright (1896, i. 16 D). The hamzah itself is encoded ⟨⟩ and may be followed by either ⟨u, a⟩ or ⟨i⟩. See below section 4.2 on page 15.

\(^{11}\)For the letter with no diacritical points below, see Rem. b. below.
That said, as a consonant is actually the spiritus lenis of the Greeks and is distinguished by the hamzah (ṣ) as it is shown in the above table. However, the bare ‘alif may also be encoded as . Whether it be followed by a vowel or not, like so: wa-. An ُ, wa-.n (where the dot symbolizes the absence of vowel), wa-.Aan ُ, wa-.An ُ, wa-.An ُ, and wa-. An ُ.

Rem. b. The letter ی with two points below, المَّ, may also be written without diacritical points as ی. When it is used as a consonant, it is encoded aY, where a recalls the fatḥah placed above the preceding letter in vocalized Arabic, like so: qaY’uN, šayʾ un، saY’aN، šayʾ un، šaY’aN، šayʾ un، saY’aN، šayʾ un، šaY’aN، šayʾ un، šaY’aN، šayʾ un، šaY’aN، šayʾ un، šaY’aN، šayʾ un، šaY’aN، šayʾ un

The same result may be achieved by encoding this letter as ـ.y, like so: qa.y’uN، šayʾ un، sa.y’aN، šayʾ un، sa.y’aN، šayʾ un، sa.y’aN، šayʾ un، sa.y’aN، šayʾ un، sa.y’aN، šayʾ un، sa.y’aN، šayʾ un، sa.y’aN، šayʾ un، sa.y’aN، šayʾ un، sa.y’aN، šayʾ un، sa.y’aN، šayʾ un

3.2 Additional characters

Table 2 gives the ArabTeX equivalents for some additional Persian characters.

<table>
<thead>
<tr>
<th>Letter</th>
<th>Transliteration</th>
<th>ArabTeX notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ت</td>
<td>p</td>
<td>p</td>
</tr>
<tr>
<td>د</td>
<td>ch</td>
<td>ċ</td>
</tr>
<tr>
<td>ژ</td>
<td>zh</td>
<td>Ċ</td>
</tr>
<tr>
<td>ف</td>
<td>v</td>
<td>v</td>
</tr>
<tr>
<td>گ</td>
<td>g</td>
<td>g</td>
</tr>
<tr>
<td>چ</td>
<td>n</td>
<td>n</td>
</tr>
</tbody>
</table>

Table 2: Standard ArabTeX (additional characters)

Rem. The alveolar consonants ژ and چ are processed as solar letters by arabluatex.

3.3 Vowels

3.3.1 Long vowels

Table 3 gives the ArabTeX equivalents for the Arabic long vowels.

Table 3: Standard ArabTeX (long vowels)

---

12 See below section 8 on page 41.
13 The characters that are listed in this table are not included in this standard. However, as arabica is based on dmg, the dmg equivalents have been used here.
14 This character is not found in Brockelmann et al. (1935, p. 2). It is taken from the DIN 31 635 (2011) standard.
15 See note 14.
16 See below section 8 on page 41.
### Table 3: Standard ArabTEX (long vowels)

<table>
<thead>
<tr>
<th>Letter</th>
<th>Transliteration</th>
<th>ArabTEX notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>و</td>
<td>ū ū ū</td>
<td>U</td>
</tr>
<tr>
<td>ي</td>
<td>ī ī ī</td>
<td>A or Y</td>
</tr>
<tr>
<td>ي ٰ</td>
<td>ā ā ā</td>
<td>_a</td>
</tr>
<tr>
<td>ٗـ</td>
<td>ū ū ū</td>
<td>_u</td>
</tr>
<tr>
<td>ٖـ</td>
<td>ī ī ī</td>
<td>_i</td>
</tr>
</tbody>
</table>

**Rem. a.** The long vowels ُā, ū, ī, otherwise called `burāfā` 'l-madd, the letters of prolongation, involve the placing of the short vowels َā, َū, َī respectively. arabtex does that automatically in case any from voc, fullvoc or trans modes is selected e.g. قَال qāla, قَلَّ يَقُول qayla.

**Rem. b.** Defective writings, such as َـ al-ʾalif `l-maqṣūrat, or defective writings of ُā and ُī are encoded _a_ and _i_ respectively, e.g. _d_alikaَ كِلٰذ al-mal_a'ikaَ كِلٰذ, hu_dayfaT-u bn-u `l-yamAn_iَ كِئٰلَملَا nu_dayfaT-u bn-u `l-yamAn_iَ كِئٰلَملَا for Ḥuḏayfatَ كِلٰذ bnَ كِئٰلَملَا etc.

**Rem. c.** The letter ي with two points below, اَهِتحَتنِمةاَّنَثُملاُءآَيلَا may also be written without diacritical points as َيَلَي. When it is used as a long vowel, it is encoded _iY_ where _i_ recalls the kasrah placed below the preceding letter in vocalized Arabic, like so: _iY_ َلَيَلَيَلُي, _iY_ َلَيَلَيَلُي, _iY_ َلَيَلَيَلُي yamšī.

### 3.3.2 Short vowels

Table 4 gives the ArabTEX equivalents for the Arabic short vowels.

<table>
<thead>
<tr>
<th>Letter</th>
<th>Transliteration</th>
<th>ArabTEX notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ـ</td>
<td>a a a</td>
<td>a</td>
</tr>
<tr>
<td>ـ</td>
<td>u u u</td>
<td>u</td>
</tr>
<tr>
<td>ـ</td>
<td>i i i</td>
<td>i</td>
</tr>
<tr>
<td>ـ َـ</td>
<td>an an an</td>
<td>aN</td>
</tr>
<tr>
<td>ـ ِـ</td>
<td>un un un</td>
<td>uN</td>
</tr>
<tr>
<td>ـ ًـ</td>
<td>in in in</td>
<td>iN</td>
</tr>
</tbody>
</table>

**Table 4: Standard ArabTEX (short vowels)**

---

17 For the letter ٰ with no diacritical points, see Rem. c. below.
18 = al-ʾalif `l-maqṣūrat. 
19 See below section 8 on page 41.
Whether Arabic texts be vocalized or not is essentially a matter of personal choice. So one may use voc mode and decide not to write vowels except at some particular places for disambiguation purposes, or use novoc mode, not write vowels—as novoc normally does not show them—except, again, where disambiguation is needed. However, it may be wise to always write the vowels, leaving to the various modes provided by arabluatex to take care of showing or not showing the vowels.

That said, there is no need to write the short vowels fatḥah, dammah or kasrāh except in the following cases:
- at the commencement of a word, to indicate that a connective ʾalif is needed, with the exception of the article (see below section 4.4 on page 22);
- when arabluatex needs to perform a contextual analysis to determine the carrier of the hamzah;
- in the various transliteration modes, as vowels are always expressed in romanized Arabic.

4 arabluatex in action

4.1 The vowels and diphthongs

Short vowels  As said above, they are written (a, u, i):

_halaqa_ (or xalaqa) ḥalaqa, ʿamsuN ʾamsun, karImuN Karīmun.

.bi-hi bi-hi, ʾaqiTuN ʾaqīṭun.

.la-hu la-hu, ʾhuǧgiatan.

Long vowels  They are written (U, A, I):

.qAla qala, bIʿa bīʿa, tUruN ṭūrūn, tImuN ṭīnūn, murUʿaTuN murūʿatūn.

ʾalif maqṣūrah  It is written ⟨_A⟩ or ⟨Y⟩:

.al-fat_A al-fatā, al-maqh_A al-maqhā, ʾil_A ʾilā.
ʾalif otiosum Said ʾalif “ʾ-l-ʾiqāyat”, “the guarding ʾalif”, after و at the end of a word, both when preceded by ẓammah and by fatha is written ⟨UA⟩ or ⟨AW, aWA⟩:

na.sarUA اوُرَصَن, katabUA اوُبَتَك, ya.gzUA اوُزَغَي, ramaW اوْمَر, banaWA اوْنَب, banaw.

ʾalif maḥḏūfah and defective ʾū, ī They are written ⟨_a, _i _u⟩:

al-l_ah-uُهّٰللَا, ʾil_ahuNٌهٰلِإ, al-ra.hm_an-uُنٰمحَّرلَا, l_akinنِكٰـل, h_ahunAاَنُهٰه, Hunayn-u bn-u 'is.h_aq-aَقٰحسِإُنبُنيَنُح, rabb_iّٖبَر, al-ʿA.s_iٖصاَعلَا.

Silent Some words ending with ُ are usually written ُوَـ or ُوٰ instead of ُاَـ: see Wright (1896, i. 12 A). arabluatex preserves that particular writing; the same applies to words ending in ُيَـ for ُاَـ. Long vowels ⟨U, I⟩ shall receive no sukūn after a ʿalif maḥḏūfah and are discarded in trans mode:

hay_aUTuNٌةوٰيَح, sal_aUTuNٌةوٰلَص, mi^sk_aUTuNٌةوٰكشِم, tawr_aITuNٌةيٰروَت, ar-rib_aIT-uُةيٰبِّرلَا.

And so also: al-rib_aUA اوٰبِّرلَا, ribaNU اوًبِر, 'Amrٌةوْرَمَع, 'AmriNU اوٍرمَع, 'Amri_iU bn-i _hAlidiNِدِلاَخِنْبوِرْمَعُنْبُدَّمَحُم.

And so also: al-rib_aUA اوٰبِّرلَا, ribaNU اوًبِر, 'Amrٌةوْرَمَع, 'AmriNU اوٍرمَع, 'Amri_iU bn-i _hAlidiNِدِلاَخِنْبوِرْمَعُنْبُدَّمَحُم.

Tanwīn The marks of doubled short vowels, ُـ, ًـ, ٍـ are written ⟨uN, aN, iN⟩ respectively. arabluatex deals with special cases, such as ًـ taking an ا after all consonants except ُ, and tanwīn preceding ى as in ُهُدٰى, which is written ⟨aN_A⟩ or ⟨aNY⟩:
mAluN َالمْ māl‘un, bAbaN ُبابَيْ bāb‘un, madInaN ُمديَّة madīnat‘un, bintiN ُبنتْ bint‘un maqhaN_A ُمَكَّهْ maqha‘un, fataN_Y ُفَتْاَنْ fata‘un.

arabluatex is aware of special orthographies: ـsay’uN ُءيَّشْ šayʾun, ـsay’aN ُءيَّشْ šayʾan, ـsay’iN ُءيَّشْ šay’in.

In some cases, it may be useful to mark the root form of defective words so as to produce a more accurate transliteration of ending tanwīn. As seen above, tanwīn preceding َيْ is written ⟨aN_A⟩ or (aNY). Such forms as قاَضَيْ قاذِيَّ may likewise be written ⟨iNI⟩:—

al-qA.dI َيِضاَقْ al-qāḍī, qA.diaN ِإِضاَقْ qāḍiy, qA.dIN قاَضَيْ قاذِيَّ.

4.2 Other orthographic signs
tāʾ marbūṭah  It is written ⟨T⟩:

madInaTuN ُمَدِينَة madīnat‘un, madInaN ُمَدِينَة madīnat‘un, madInaN ُمَدِينَة madīnat‘un.

hamzah  It is written ⟨’⟩, its carrier being determined by contextual analysis. In case one wishes to bypass this mechanism, he can use the “quoting” feature that is described below in section 4.4 on page 22.


hamzah followed by the long vowel ِيْ is encoded: ‘_U: ‘_U_A ُأْوَلِيْ أُولِيَّ ‘ūlā, ‘_U1U أُوْلُأْ أُولُأَكْ ‘ūlū, ‘_U1A’ika ُأَكِئْلَ أُولِأْ ’ūlāʾika.

hamzah followed by the long vowel ِيْ is encoded: ‘_I: ‘_ImAnuN ُيَنِمْ إِنْ ʾīmān.

**Middle hamzah:** xa.ti’-Ina ُتَخَيْأَتَ xaṭīʾat‘un, xa.tI’aTuN ُتَحَيَّتْ xaṭīyat‘un, su’ila ُسُلْيَلْ suʾīla, ‘as’ilaTuN ُعَلَأَتْ ‘as’ilat‘un, mas’alaTuN ُمَعْلَأَتْ masʿalat‘un, ‘as’alu ُعَلَأَ أُسْأَلْ ‘as’al‘un, yatasA’alUna ُتَأْلَعْ قَأْتَ تَأْلَعْ yatasalʿāna, murU’aTuN ُعَمْوَأَتْ murūʿat‘un, ta’amuN ُتَأْمُعَتْ laʾam’at‘un, ta’xIruN ُتَأْخَرَتْ taʾaḥharat‘un, ta’a’axxara ُتَأْخَرَةَ تَأْخَرَةَ taʾaḥharat‘un, qa’iLun ُقَأْلِلْ qā’il‘un, .hIna’i_diN ُحَيَّنَةَ .hīna’idīn, hay’aTuN ُحَيَّتْ قَأْتَ hay’ātat‘un, hay’AtuN ُحَيَّاتُ hay’ātān.

*For another way of encoding the initial hamzah followed by a long vowel, see the *taḫfīf ‘l-hamzat* on the following page.*
From Wright (1896, i. 14 B):— All consonants, whatsoever, not even ʾalif hemzatum excepted, admit of being doubled and take tašdīd. Hence we speak and write raʿAsuN رَأَسْ رَاُسُ، saʿAluN سَأَلْ سَأَلُ، naʿAjuN نَأَجُ نَأَجَ، naʿāqN نَأَقُ نَأَقَ.

Final hamzah: xa.taʿuN خَتَا نَ، xa.taʿaN خَتَا نَ، xa.taʿiN خَتَا نِ. taqraʿuN طَقْ رَأْنُ، taqraʿaN طَقْ رَأْنَ، taqraʿiN طَقْ رَأْنِ.

Hamzat i: if the hamzah has ḍazmah and is preceded by ʾalif hamzatum, it must be changed into the letter of prolongation that is homogeneous with the preceding vowel; hence: ʾāmana َنَمآ ʾāmana، ʾūminu ُنِمْوُأ ʾūminu، ʾīmāN ِنِمِإ ʾīmān، nisāʾu-hu ُهُؤآَسِن nisāʾu-hu، nisāʾi-hi ِهِئآَسِن nisāʾi-hi، nisāʾī ِئآَسِن nisāʾī.

Tahfīf ʾl-hamzat: if the hamzah has ḍazmah and is preceded by ʾalif hamzatum, it must be changed into the letter of prolongation that is homogeneous with the preceding vowel; hence: ʾāmana ِنِمَآ ʾāmana، ʾūminu ُنِمْوُأ ʾūminu، ʾīmāN ِنِمِإ ʾīmān، nisāʾu-hu ُهُؤآَسِن nisāʾu-hu، nisāʾi-hi ِهِئآَسِن nisāʾi-hi، nisāʾī ِئآَسِن nisāʾī.

Imperatives of verbs that have the hamzah as the first radical are other cases of tahfīf ʾl-hamzat: ʾIsir ِالْسِّرُ ʾIssir، ʾIdan ِالْدَّانُ ʾIdan، ʾUml َعُمْلُ ʾUml. arabluatex also provides ways of encoding those words when the initial ʾalif comes into waṣl, so as to make the ʾalif waṣl fall away when preceded by َو or َف: waʿSir َوُلْسِرُ wa-ʾSir، faʿDan َوُلْسِرُ fa-ʾDan، faʿTi َوُلْسِرُ fa-ʾTi، waʿTamirUA َوُلْسِرُ wa-ʾTamirUA، or be retained outside the imperative, as in faʿIʿTazarat َوُلْسِرُ fa-ʾTazarat، ba da taʾtilAfīN َوُلْسِرُ ba da taʾtilAfīN.
The strange spelling of \textit{miʾat}\textsuperscript{un}: \textit{miʾaTuN} \\ \textit{miʾatAni} \\ \textit{miʾatāni} \\ \textit{miʾatayni} \\ \textit{miʾūna} \\ \textit{miʾātun} \\ \textit{miʾạn} \\ Of course, the ‘pipe’ character can be used to prevent this rule from being applied (see section 4.5 on page 24): \textit{miʾa|TuN} \\

\textit{maddah} \ At the beginning of a syllable, \textit{ʾalif} with hamzah and fatḥah (ٖ) followed by \textit{ʾalif} ʾ-\textit{maddi} (alif of prolongation) or \textit{ʾalif} with hamzah and ǧazmah (ٖ) are both represented in writing \textit{ʾalif} with \textit{maddah}: ٖ (see Wright 1896, i. 25 A–B).

Hence one should keep to this distinction and encode \textit{ʾaʾkulu} ٞ ʾākulu and \textit{ʾAkiluN} ٞ ʾākil\textsuperscript{un} respectively.

\textit{šaddah} \ \textit{tašdīd} is either necessary or euphonic.

\textbf{The necessary \textit{tašdīd} \ always follows a vowel, whether short or long (see Wright 1896, i. 15 A–B).} It is encoded in writing the consonant that carries it twice:

\textit{`allaqa} ُلْئَا \textit{allaga}, \textit{mAdduN} ُدْلْلَمَّ \textit{mādd\textsuperscript{un}}, \textit{ʾammara} ُلْتَرْمَأ \textit{ammara}, \textit{murrūN} ُلْرَمَأ \textit{murr\textsuperscript{un}}.

\textbf{The euphonic \textit{tašdīd} \ always follows a vowelless consonant which is passed over in pronunciation and assimilated to a following consonant.} It may be found (Wright 1896, i. 15 B–16 C):—

(a) With the \textit{solar letters} \\

\textbf{šaddah} \ \textit{tašdīd} is either necessary or euphonic.
Unlike arabetx and arabxetex, arabluatex never requires the solar letter to be written twice, as it automatically generates the euphonic taṣdīd above the letter that carries it, whether the article be written in the assimilated form or not, e.g. al-ʾsams-u ʾaš-šams*, or a<solar letter>-s-ʾsams-u ʾaš-šams*.


(b) With the letters ر, ل, م, و, ي after ن with ǧazmah, and also after the tanwīn:—

Note the absence of sukūn above the passed over ن in the following examples, each of which is accompanied by a consistent transliteration: min rabbi-hi ʾan yaqtula, ʾay yaqtula.

With tanwīn: kitAbuN mubInuN ʾkīlābṣ mubīnṣ.

Rem. This particular feature must be put into operation by the \SetArbDflt* command explicitly. See above section 2.2.1 on page 6 for further details. Other kinds of assimilations, including the various cases of ʾidāgām, will be included in arabluatex gradually.

(c) With the letter ت after the dentals ث, د, ذ, ض, ط, Ẓ in certain parts of the verb: this kind of assimilation, e.g. ُتثِبَل for ُتثِبَل labiṯtu, will be discarded here, as it is largely condemned by the grammarians (see Wright 1896, i. 16 B–C).

The definite article and the ʾalif“ ʾl-waṣl At the beginning of a sentence, ʾ is never written, as ʾ ʾl-ḥamād ῆl-lāh; instead, to indicate that the ʾalif is a connective ʾalif (ʾalif“ ʾl-waṣl‘), the hamzah is omitted and only its accompanying vowel is expressed:


As said above on page 6, fullvoc is the mode in which arabluatex expresses the sukūn and the waslah. arabluatex will take care of doing that automatically provided that the vowel which is to be absorbed by the final vowel of the preceding word be properly encoded, like so:—

(a) Definite article at the beginning of a sentence is encoded ʾal-, or ʾa<solar letter>- if one wishes to mark the assimilation—which is in no way required, as arabetx will detect all cases of assimilation.

(b) Definite article inside sentences is encoded ʾl-, or ʾ<l<solar letter>-.
In all remaining cases of elision, the 'alif 'l-waṣl is expressed by the vowel that accompanies the omitted hamzah: (u, a, i).

**Article:** bAb-u 'l-madrasaT-i, al-maqA laT-u 'l-'U1_A al-maqa`alat-u 'l-ūlā, al-lu.gaT-u 'l-'ara biyyaT-u 'l-arabīyyat-u, fī .sinA`aT-i 'l-.tibb-i fi sināatility-likībī, 'i1_A 'l-intiqA.d-i ila 'l-.intiqād-i, fī 'l-ibtidA`-i 'l-ʿarabiyyat-u 'l-najm-a, fa-lammA raʾawu 'n-naǧm.

**Particles:**

(a) li-: 'alif 'l-waṣl is omitted in the article li-l-.rajul-i li-r-raǧul-i. If the first letter of the noun be ل, then the ل of the article also falls away, but arthritis is aware of that: li-l-laylat-i li-l-laylat-i.

(b) la-: the same applies to the affirmative particle la-l-.haqq-u la-l-.haqq-i.

(c) With the other particles, 'alif 'l-waṣl is expressed: fī 'l-madīnaT-i fī 'l-madīnat-i, wa-'l-rajul-u wa-'r-raǧul-u, bi-'l-qalam-i bi-'l-qalam-i, bi-'l-ruʿab-i bi-'r-ruʿab-i.

**Perfect active, imperative, nomen actionis:** qAla isma` qāla 'sma`, qāla uqtul huwa inhzama, wa-ustumila wa-stumila, qadi in sarafa qadi n sarafa, al-iqtidAr-u al-iqtidār-u, 'i1_A 'l-intiqA.d-i 'i1_A 'l-intiqād-i, lawistaqbala lawi 'staqbala.

**Other cases:** 'awi ismu-hu 'awi ismu-hu, zayduN ibn-u 'amriNU-Zayd`um bu `Amr`, umar-u `Amr`, `Umar-u ibn-u 'l-ha.t.tAb-i `Umar`um bu `l-Haṭṭāb`, imru`-u `l-qays-i Imru`-u `l-Qays`i, la-aymun-u la-aymun-u `l-lāh`i.

---

22 "Zayd is the son of 'Amr": the second noun is not in apposition to the first, but forms part of the predicate. Hence زید بن عمرو "Zayd, son of 'Amr".

23 "Umar is the son of al-Ḥaṭṭāb" (see note 22).
ʾalif ʾ-waṣl preceded by a long vowel  The long vowel preceding the connective ʾalif is shortened in pronunciation (Wright 1896, i. 21 B–D). This does not appear in the Arabic script, but arabluatex takes it into account in some transliteration standards:—


ʾalif ʾ-waṣl preceded by a diphthong  The diphthong is resolved into two simple vowels (Wright 1896, i. 21 D–22 A) viz. ay → āî and aw → āû. arabluatex detects the cases in which this rule applies:—

fi ʿaynay ʾl-malik-i, ix.say ʾl-qaw m-a ʾl-ah-i, mu.s.tafaw ʾl-ʾamIr-i. muṣṭafa ʾl-ʾilāl, raʾaytumu ʾr-raǧul, maṇi ʾl-kaḏḏāb-i, qatalati ʾl-ʾUm-i.u. qatalati ʾr-Rūm-i.

ʾalif ʾ-waṣl preceded by a consonant with sukūn  The vowel which the consonant takes is either its original vowel, or that which belongs to the connective ʾalif or the kasrah; in most of the cases (Wright 1896, i. 22 A–C), it is encoded explicitly, like so:—

ʾantumu ʾl-kāḏib-Una, raʾyatumu ʾl-raǧul-i, mani ʾl-ka_d_dAb-u. qatalati ʾl-ʾUm-u. qatalati ʾr-Rūm-i.

However, the Arabic script does not show the kasrah or the dammah which may be taken by the nouns having tanwīn although it is explicit in pronunciation and must appear in some transliteration standards. arabluatex takes care of that automatically:—


20
4.3 Special orthographies

The name of God The name of God, ﷲ، is compounded of the article ﷲ، and ﷲ (noted ﷲ with the defective ʾalif) so that it becomes ﷲ; then the hamzah is suppressed, its vowel being transferred to the ل before it, so that there remains ﷲ (I refer to Lane, Lexicon, I. 83 col. 1). Finally, the first ل is made quiescent and incorporated into the other, hence the laṣdid above it. As arabluatex never requires a solar letter to be written twice (see above, on page 17), the name of God is therefore encoded al-l_ah-u or 'l-l_ah-u:—

al-l_ah-u ﷲ al-lāh u, yA|24 al-l_ah-u ﷲ yā al-lāh", ’a-fa|25-al-
_ah-l la-ta.g alanna a-fa-al-lāh' la-taq alanna, bi-'l-
_ah-š bi-'l-lāh', wa-'l-l_ah-i wa-l-lāh', bi-sm-i 'l-l_ah-i
š bi-sm' l-lāh', al-.hamd-u li-l-l_ah-i li-l-lāh',
li-l-l_ah-i 'l-qA'il-u u li-l-lāh | li-l-lāh |'l-gā il".

The conjunctive Although it is compounded of the article ﷲ، the demonstrative letter ل and the demonstrative pronoun ذا، both masculine and feminine forms that are written defectively are encoded alla_dI and allatI respectively. Forms starting with the connective ʾalif are encoded 'lla_dI and 'llatI:—

'a_hAfu mina 'l-malik-i 'lla_dI ya.zlimu 'l-nAs-a, 'udtu 'l-
say-h-a 'lla_dI huwa marI duN u 'udtu 'š-sayh" llaḏi huwa marid", ma 'anA bi-'lla_dI qa'iluN la-ka 'say aN mā 'ānā bi-'llaḏī qā'lūN la-ka 'šay aN.

'ari-nA 'illa_dayni 'a.dallA-nA mina 'l-jinn-i wa-'l-'ins-i 'ārī 'illa_dayni 'adallā-nā mina 'l-jinn wa-š-'insi.

The other forms are encoded regularly as al-l or 'l-l:—

fa-ʿinnA na_dkuru 'l-.sawt-ayni 'l-la_dayni rawaynA-humA ʿan ja.h.zaT-a fa-ʿinnā naḏkuru ʿṣ-sawt ayni ʿl-lādayni rawaynā-humā ān Ġahzat".

24 Note the “pipe” character '|' here after ya and below after fa before footnote mark 25: it is needed by the dmg transliteration mode as in this mode any vowel at the commencement of a word preceded by a word that ends with a vowel, either short or long, is absorbed by this vowel viz. ʿalạ 'ṭ-ṭarīq i. See section 4.5 on page 24 on the “pipe” and section 8 on page 41 on dmg mode.

25 See note 24.
And also: al-la_dAniُِّللَا al-ladāni, al-la_dayniُِّللَا al-lātāni, al-latAniُِّللَا al-lātāni, al-lA'Iُِّللَا al-lāʾāti, and so forth.

4.4 Quoting

It is here referred to “quoting” after the arabtex package. The “quoting” mechanism of arabiutex is designed to be very similar in effect to the one of arabtex.

To start with an example, suppose one types the following in novoc mode:

ملعمّلع ةءيهلا is it َمِّلُع، he was taught the science of astronomy, or َمَّلَع، he taught the science of astronomy? In order to disambiguate this clause, it may be sensible to put a dāmmah above the first ع, which is achieved by “quoting” the vowel u, like so: "ullima, or, with no other vowel than the required u: "ullm.

This is how the “quoting” mechanism works: metaphorically speaking, it acts as a toggle switch. If something, in a given mode, is supposed to be visible, “quoting” hides it; conversely, if it is supposed not to, it makes it visible.

As shown above, “quoting” means inserting one straight double quote (" before the letter that is to be acted upon. Its effects depend on the mode which is currently selected, either novoc, voc or fullvoc:

novoc In this mode, “quoting” essentially means make visible something that ought not to be so.

(a) Quoting a vowel, either short or long, makes the dāmmah, fathah or kasrah appear above the appropriate consonant:

"ullima 'ilm-a 'l-hay'aT-i َعَلَّمْ عَلَّم هُهَهْيَة ُيْلَمْ عَلَّم هُهَهْيَة, ya.gz"UA اوُزغي yaġzū.

(b) The same applies when “quoting” the tanwīn:

wa-'innAsawfatudriku-nA'l-manAyAmuqadd"araT"AN َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف َوَأَنَا سوَف wa-ʾinnā sawfa tudriku-na 'l-manāyā muqaddarat

(c) If no vowel follows the straight double quote, then a sukūn is put above the preceding consonant:

qAla isma" ْعمسالاق qāla 'isma', jA'at" hinduN جَمَاتْ هَنَدَا " Hind"", sabIhuN bi-man q"u.ti'at" qadamA-hu شْبِه بِمِن قَطْعُتْ QadamA-hu café qadamA-hu shābiN bi-man quṭiʿat qadamA-hu.

Note here the “pipe” character ‘|’: as already stated on page 17, the sequence ‘A usually encodes ʾalif with hamzah followed by ʾalif of prolongation, which is represented in writing ʾalif with maddah: ۸. The “pipe” character prevents this rule from being applied. See section 4.5 on page 24.

See Lagally (2004, p. 22)
(d) At the commencement of a word, the straight double quote is interpreted as ʾalif "t-wašl":—

wa-"ust"u'mila وَأَسْتَعْمِل qāla qā'il", ibn-u ḫbI 'u.saybi'aT-

"a Ibn ḫAbī Usaybi'at".

(b) tanwīn:—

madīnaT"an madinat", bAb"an bāb", hud"A huda",

"iN sây" "iN sây".

One may more usefully "quote" the initial vowels to write the waṣlah above the ʾalif or insert a straight double quote after a consonant not followed by a vowel to make the sukūn appear:—

(a) ʾalif" t-wašl":—

fI "sti.q.sA'-iN fi 'sti.q.sA'-uN wa-"sti.q.sA'-uN qāla "uhrub fa-lan tuqtala qāla "uhrub fa-lan tuqtala.

(b) sukūn:—

qāla "uqtul" fa-lan tuqtala qāla "uhrub fa-lan tuqtala, mA JĀat" mini imra'at; mā gā'at mini imra'at; kam qad maḍat min laylat; kam qad maḍat min laylat.

fullvoc In this mode, “quoting” can be used to take away any short vowel (or tanwīn, as seen above) or any sukūn:—

al-jamr-u 'l-.sayfiyy-u 'lla_dī kāna bi-q"rAn" nUn-a\n\n\nالجمرُ الصيفيُّ الّذي كان بِzyćرَانُو

4.4.1 Quoting the hamzah

As said above in section 4.2 on page 15, the hamzah is always written (‘), its carrier being determined by contextual analysis. "Quoting" that straight single quote character like so: (‘) allows to determine the carrier of the hamzah freely, without any consideration for the context. Table 5 gives the equivalents for all the possible carriers the hamzah may take.
Table 5: "Quoted" hamzah

As one can see from table 5, the carrier of the hamzah is inferred from the letter that precedes the straight double quote (""). Of course, any "quoted" hamzah may take a short vowel, which is to be written after the ArabTEX equivalent for the hamzah itself, namely ⟨‘⟩. For example, َؤ is encoded ⟨w’⟩, while ْؤ is encoded ⟨w”⟩. In the latter example, the second straight double quote encodes the sukūn in voc mode in accordance with the rule laid above on pages 22–23.

4.5 The ‘pipe’ character (|)

In the terminology of ArabTEX, the “pipe” character '|' is referred to as the “invisible consonant”. Hence, as already seen above in section 4.4.1 on the preceding page, its usage to encode the hamzah alone, with no carrier: ْؤ.

Aside from that usage, the “pipe” character is used to prevent almost any of the contextual analysis rules that are described above from being applied. Two examples have already been given to demonstrate how that particular mechanism works in note 24 on page 21 and in note 26 on page 22. One more example follows:—

\[
\text{bi-Qrānnūn, "in Crannon" (Thessaly, Greece).}^{29}
\]

As one can see, the “pipe” character between the two ⟨n⟩ prevents the necessary tašdīd rule (page 17) from being applied.

4.6 Putting back on broken contextual analysis rules

In complex documents such as critical editions where footnotes and other kind of annotations can be particularly abundant, the contextual analysis rules that are described above may be broken by \LaTeX{} commands. To take an example, consider the following:—

\[
\text{bi-Qrānnūn, "in Crannon" (Thessaly, Greece).}^{29}
\]

28See below section 8 on page 41.
29See more context on the previous page.
This is wrong:
\begin{arab}[fullvoc]
fa-lammA ra‘aW\LRfootnote{A footnote which interferes with the contextual analysis.} 'l-na`gma...
\end{arab}

This is wrong:

\begin{arab}[trans]
\uc{z}ayduN\LRfootnote{A footnote which interferes with the contextual analysis.}
\end{arab}

\LRFootnote{A footnote which interferes with the contextual analysis.}

According to the rule stated on page 20, the diphthong in ra‘aw must be resolved into two simple vowels before the ‘alif” l-‘awṣī, as ُرَاوْ أُنْتَجَمْ.

The \arbnul command is provided so as to put back on contextual analysis rules in such situations. It takes as argument the word that must be brought back for any given rule to be applied as it ought to. Depending on the contexts that have to be restored, \arbnul may be found just after or before Arabic words.

In any case, no space must be left after or before the Arabic word that \arbnul is applied to.

The following shows how the Arabic should have been written in the preceding example and gives further illustrations of the same technique:—

\begin{arab}[fullvoc]
fa-lammA ra‘aW\arbnul{'l-na`gma}\LRfootnote{A footnote which interferes with the contextual analysis.}
\end{arab}

\begin{arab}[trans]
\uc{z}ayduN\arbnul{ibn}\LRfootnote{See \vref{fn:zayd-is-son}.}
\end{arab}
4.7 Stretching characters: the taʾwīl

A double hyphen ⟨--⟩ stretches the ligature in which one letter is bound to another. Although it is always better to rely on automatic stretching, this technique can be used to a modest extent, especially to increase legibility of letters and diacritics which stand one above the other:

\[ \text{Zayd } \text{unie } '\text{bn } '\text{Amr}^{in} \]

4.8 Digits

4.8.1 Numerical figures

The Indian numbers, ar-raqam\(\text{ u }'l\text{-hindiyy}\), are ten in number, and they are compounded in exactly the same way as our numerals:

\[ 1874 \quad 123-456,789 \quad fI \text{sanaT-i 1024} \]

4.8.2 The ʾabǧad

The numbers may also be expressed with letters from right to left arranged in accordance with the order of the Hebrew and Aramaic alphabets (see Wright 1896, i. 28 B–C). The ʾabǧad numbers are usually distinguished from the surrounding words by a stroke placed over them.

\[ \text{ʾabǧad numbers are inserted with the } \texttt{\textbackslash abjad}\{\text{number}\} \text { command in any of the voc, fullvoc and novoc modes, where } \{\text{number}\} \text { may be any number between 1 and 1999, like so:---} \]

\[ \text{\texttt{\abjad{45} kitAbu-hu fI 'l-'AdAt-i 45 kitābu-hu fī 'l-ādāt.}} \]

Rem. a. As can be seen in the above given example, arabiatex expresses the ʾabǧad numbers in Roman numerals if it finds the \texttt{\abjad} command in any of the transliteration modes.

Rem. b. \texttt{\abjad} may also be found outside Arabic environments. In that case, arabiatex does not print the stroke as a distinctive mark over the number for it is not surrounded by other Arabic words. In case one nonetheless wishes to print the stroke, he can either use the \texttt{\aoline} command that is described below in section 4.10.1 on page 28 or insert the ʾabǧad number in \texttt{\arb[novoc]{}}:---

26
The \arb\{trans\}{'abjad} number for 1874 is \abjad{1874}.
The \arb\{trans\}{'abjad} number for 1874 is \abjad{1874}.
The \arb\{trans\}{'abjad} number for 1874 is \abjad{1874}.
The \arb\{trans\}{'abjad} number for 1874 is \abjad{1874}.
The \arb\{trans\}{'abjad} number for 1874 is \abjad{1874}.

\abjad may also be used to convert values of counters into 'abjad numbers, like so:

1. The \arb\{trans\}{'abjad} number for the current page (\thepage) is \abjad{\thepage}.

The 'abjad number for the current page (27) is \abjad{27}.

This technique can be used to produce abjad-numbered lists as will be demonstrated on page 53.

4.9 Additional characters

In the manuscripts, the unpointed letters, \textit{ahl-	extasciitilde{h}ur	extasciitilde{u}f \textasciitilde{l}-muhmalat\textasciitilde{u}}, are sometimes further distinguished from the pointed by various contrivances, as explained in Wright (1896, i. 4 B–C). One may find these letters written in a smaller size below the line, or with a dot or another mark below. As representing all the possible contrivances leads to much complexity and also needs to be agreed among scholars, new ways of encoding them will be proposed and gradually included as Arabluatex will mature.

For the time being, the following is included:

<table>
<thead>
<tr>
<th>Letter</th>
<th>Transliteration</th>
<th>ArabTEX notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ب</td>
<td>b</td>
<td>b</td>
</tr>
<tr>
<td>د</td>
<td>d</td>
<td>d</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>f</td>
</tr>
<tr>
<td>q</td>
<td>q</td>
<td>q</td>
</tr>
<tr>
<td>k</td>
<td>k</td>
<td>k</td>
</tr>
<tr>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>ٓ</td>
<td>(</td>
<td>(</td>
</tr>
<tr>
<td>ٓ</td>
<td>)</td>
<td>)</td>
</tr>
</tbody>
</table>

Table 6: Additional Arabic codings

\textit{'af	extasciitilde{a}m\textasciitilde{a}ntu\textasciitilde{s} Gal.(M) , f\textasciitilde{a}n\textasciitilde{a}nts (sic) Gal.(E1), f\textasciitilde{a}n\textasciitilde{a}nts (sic) Gal.(E1), f\textasciitilde{a}m\textasciitilde{a}ntus Gal.(M) f\textasciitilde{a}nts (sic) Gal.(E1).}

\textsuperscript{30} See below section 8 on page 41.
4.10 Arabic emphasis

As already seen in section 4.8.2 on page 26, the ʾabǧad numbers are distinguished from the surrounding words by a stroke placed over them. This technique is used to distinguish further words that are proper names or book titles.

One may use the \aemph{⟨Arabic text⟩} command to use the same technique to emphasize words, like so:

\abjad{45}: kitAbu-hu \aemph{fI 'l-ʿĀdāt-i}.

Rem. a. As the above example shows, arabluatex places the horizontal stroke under the emphasized words in any of the transliteration modes.

Rem. b. \aemph* is also provided should one wish to always have the horizontal stroke printed over the emphasized words, like so: \abjad{45}: kitAbu-hu \aemph*[fI 'l-ʿĀdāt-i].

4.10.1 Underlining words or numbers

Three additional, non context-sensitive commands are provided to distinguish words or numbers:

(a) \aoline, which is equivalent to \aemph* described above.
(b) \aoline*, which is the same as \aoline, but better suited for ʾabǧad numbers.\footnote{See the example provided above section 4.8.2 on page 26.}
(c) \auline, which can be used to underline Arabic words.

5 Arabic poetry

arabluatex provides a special environment for typesetting Arabic poetry. Every line in this environment must end with \.

The \arabverse environment may take up to six optional ‘named arguments’ each of which is set using the syntax ⟨key⟩=⟨value⟩, like so:

\begin{arabverse}[key1=value1, key2=value2, ...]
<verses>
\end{arabverse}

The description of the optional arguments follows:

**mode** \texttt{mode=⟨mode⟩}, either voc, fullvoc, novoc or trans. The default mode is the one that is set at load time as already seen section 2.2 on page 6.

**width** \texttt{width=⟨length⟩} \textbf{Default: 0.3\linewidth}

The default width of each hemistich that the verse consists of. It may be expressed in any accepted unit of measurement, such as 4cm or 2in. However, one must keep in mind that the total length of the two hemistichs added to the one of the gutter that separates them must not exceed the length of the base line, unless one wishes to have the hemistichs distributed on subsequent lines.
The gutter consists of the blank space that is between the two hemistichs. By default, it is commensurate with the width of the hemistich, but it may be expressed in any accepted unit of measurement as well.

The name of the metre is printed after the lines and set flush left in voc, fullvoc and novoc modes or flush right in trans mode.

This named argument does not need a value as it defaults to true if it is used. If so, a delimiter is printed between each of the hemistichs. By default, it is set to the ‘star’ character ‘*’. The \SetHemistichDelim{delimiter} command may be used at any point of the document to change this default setting.

As the preceding one, this named argument does not need a value as it defaults to true if it is used. If so, Unicode Arabic input is expected in the arabverse environment instead of ascii ArabTeX or Buckwalter input schemes. See section 10 on page 48 for more details.

The color in which lines of poetry are to be rendered.

This named argument does not need a value as it defaults to true if it is used. If export is set as a global option as well (see above on page 6), all the lines will be converted to Unicode and exported to the external selected file. See below section 12 on page 57 for more details.

Inside the arabverse environment, each line is typeset by the \bayt command which takes two mandatory arguments and may accept one optional argument. Additionally, every \bayt command must be followed with \ like so:

\bayt{ṣadr}{⟨tadwīr⟩} {⟨ʿaǧuz⟩}\

That two subsequent hemistichs should be connected with one another is technically named tadwīr. Should that happen, either the ṣadr or the ʿaǧuz or both of them, may be connected to one another by letters that are naturally bound to the following or the preceding ones over the tadwīr. The optional argument of the \bayt command is designed to deal with the various situations that may arise:

(a) If the two hemistichs be connected with one another by a prominent horizontal flexible stroke, the tatwīl should be used, like so: [--] (see section 4.7 on page 26). Of course, the ending word of the ṣadr and the word at the commencement of the ʿaǧuz must have the tatwīl too so that the proper shapes of the letters be selected. Consider for example the following:

\bayt*[ṣadr]{⟨tadwīr⟩} {⟨ʿaǧuz⟩}\

\footnote{A ‘starred’ version \bayt* is also defined. arablatex uses it internally when export is set to true to instruct some Lua functions that lines of poetry have already been processed. That aside, \bayt and \bayt* do the same, and only \bayt should be used.}
As one can see, triple hyphens have been used. In the ṣadr, the first hyphen triggers the rules that are related to the definite article and the ʾalif ʾil-wasl, while the following two select the figure of the letter lām connected with a following letter. In the ʾaǧuz, the last two hyphens select the letter yāʾ connected with a preceding letter, while the first one is simply discarded in this mode, but still may appear as it should, if the trans mode be selected:—

(b) In some other cases, it may seem difficult, if not fairly impossible, to split a given word into two parts. This happens mostly because of the šaddah. Consider for example the following:—

In the first line, the word مـَهْلٱىَلَعُنيِعَتْسَأْدَقيِّنَأَرْيَغ should be split into مـْمَهلَا as the first part of it belongs to the ṣadr and the second to the ʾaǧuz. One solution to avoid splitting this word in such a way is to write inside the tadwīr the part of it that belongs to either hemistich, without omitting to add a space after it. In the second line,
the word ُمُأ should be split into ُمـْمُأ, so that the only way to avoid splitting it into two parts is to write it all inside the *tadweēr*. In that case, as the word is to be placed in the middle, it has been surrounded by spaces.

**Scaling and distortion of characters** The *arabverse* environment and the \texttt{bayt} command are designed to typeset the verses in a two-column, fixed width layout. This may result in a somewhat distorted text. Should that happen, one may adapt the layout by modifying the values of the above described \texttt{width} and \texttt{gutter} named arguments until the visual aspect of the layout be satisfactory. It has to be noted that distortion and warping may be even more perceptible in Roman than in Arabic characters.

**Footnotes** Footnotes are not set by default inside the \texttt{bayt} command, but there are two easy ways to have them printed.

If they are little in number, each footnote may be split into pairs of \texttt{footnotemark}{} (please mind the braces or “declare” \texttt{footnotemark} using \texttt{\MkArbBreak} to take it out of the Arabic environment\textsuperscript{34}) in the argument of the \texttt{bayt} command and \texttt{footnotetext} outside the \texttt{bayt} command.

If the footnotes are abundant in number, it is advised to load the \texttt{footnotehyper} package which \texttt{arabluatex} will then use to typeset any kind of footnote that is called from the arguments of the \texttt{bayt} command.\textsuperscript{35}

**Line numbering** Inside the *arabverse* environment, the \texttt{linenumbers} environment of the \texttt{lineno} package can be used to have the lines of succeeding verses numbered. Please refer to the documentation of this package for more information or to the example below for a basic implementation of this technique.

### 5.1 Example

Here follow the first lines of Imruʾu ’l-Qaysi’s *Muʿallaqah*. In this example, \texttt{\SetArbDflt*} has been selected so as to mark the *īdġām* that is fit to this declamatory poetry:\textsuperscript{36}

```
\begin{arab}[fullvoc]
qAla imru'u 'l-qaysi fI mu'allaqati-hi:
\end{arab}
\begin{arabverse}[mode=fullvoc, metre={(al-.darbu 'l-.tAnI mina
'l-.arU. di 'l-.UL_A mina 'l-.tawIlii)}]
\SetArbDflt*
\begin{linenumbers*}
```

\textsuperscript{34}See section 11.1 on page 50.

\textsuperscript{35}The footnote package can also be used for the same effect. However, it must be loaded \texttt{after} \texttt{arabluatex}.

\textsuperscript{36}Please note that for the time being only the assimilation rules that are laid on item (b) on page 18 are applied. See section 2.2.1 on page 6 for more information. None of the editions of the *Muʿallaqāt* that I know of feature the *īdġām* in the Arabic text, although it is often strongly marked in declamation.
قول أمراً القيسي في معنی:

1. قفا بيك من ذري جيب ومزنة
2. يمعج就可以 من جندب وغمجل
3. قرى على الأرمع في عرضاتها
4. كأي عدد أن ينام فحول
5. وقفا بها حجي علي مطمأة
6. وإن شفا مبرة عبرة مروفة

(الضرب الثاني من العروض الأول من الأطول)

قال المرو القيسي في معنی:

6 Special applications

Linguistics The same horizontal stroke as the tatwîl (see section 4.7 on page 26) may be encoded ⟨B⟩; ⟨BB⟩ will receive the tašdīd. This is useful to make linguistic annotations and comments on vowels:—
Brackets  The various bracket symbols are useful in technical documents such as critical editions for indicating that some words or some letters must be added or removed. `arabluatex` will automatically fit those symbols to the direction of the text. For the time being, the following symbols are supported:

- parentheses: `()`
- square brackets: `[ ]`
- angle brackets: `<>`
- braces: `{ }`

Parentheses, square and angle brackets may be input directly at the keyboard; however, words or letters that are to be read between braces must be passed as arguments to the `\braces` command:

```
\begin{arab}
\braces{wa-qAla} 'inna 'abI kAna mina '{l-muqAtilaTi
\braces{umI min u.zamA'i buyUti '{l-zamAzimaTi.
\end{arab}
```

```
وَقَالَ أَيَّامَ كَانَ مِنَ الْمَغْجُودِ فَكَذَّبَتْ أَمَيْ عِنْدَهُمَا بَوْحٌ الْمَعَارِمَةَ
```

Additional Arabic marks  In addition to common letters, many symbols and ligatures are encoded in Arabic Unicode standard, such as honorifics consisting of complex ligatures, and annotation signs used in the Qur’an or in classical poetry.

```
\arbmark[(rl]|lr){(shorthand)}\] can be used to insert such characters either in Unicode or in romanized Arabic environments. It takes as argument a shorthand defined beforehand in a default list which consists of the following at the time of writing:

<table>
<thead>
<tr>
<th>Codepoint</th>
<th>Shorthand</th>
<th>Glyph</th>
<th>Transliteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>FDFD</td>
<td>bismillah</td>
<td>ﷴبَيْسِمُ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ</td>
<td>bi’-smi ‘Llāhi ‘r-raḥmāni ‘r-raḥīmi</td>
</tr>
<tr>
<td>FDF5</td>
<td>salam</td>
<td>ﷴسَلَّمَ</td>
<td>šalla ‘Llāhu ‘alay-hi wa-sallama</td>
</tr>
<tr>
<td>FDFA</td>
<td>slm</td>
<td>ﷴسَلَّمَ</td>
<td>šalla ‘Llāhu ‘alay-hi wa-sallama</td>
</tr>
<tr>
<td>FDFB</td>
<td>jalla</td>
<td>ﷴجَالَّا</td>
<td>ġalla ġalāla-hu</td>
</tr>
</tbody>
</table>
```

Table 7: Additional Arabic marks

The mark to be inserted is determined by contextual analysis, or by an optional argument, either rl to have the Arabic glyph printed, or lr to print the transliterated equivalent.

`\newarbmark` is also provided should one wish to define new marks in addition to the default list.
to the marks defined above. This command takes three arguments, like so:—
\texttt{\newarbmark{⟨shorthand⟩}{⟨RTL codepoint⟩}{⟨LTR rendition⟩}}

As regards the right-to-left codepoint, it may be either typed in Unicode or selected as Unicode codepoint. To that end, the \LaTeX command \texttt{\symbol{"XYZT}} or its plain \TeX variant \texttt{\char"XYZT\relax} may be used, where \texttt{XYZT} are uppercase hex digits (0 to 9 or A to F).

It is also possible to use the so-called ‘\texttt{^^^^} notation’ like so: \texttt{^^^^xyzt}, where \texttt{xyzt} are lowercase hex digits (0 to 9 or a to f).

As regards the third argument (left-to-right rendition), it may be either left empty or typed by means of \texttt{\arb[trans]{⟨arabtex code⟩}} so as to have it printed in romanized Arabic.

It must be noted that \texttt{\newarbmark} expects Arab\TeX input scheme inside \texttt{\arb[trans]{}}, to the exclusion of buckwalter input scheme.

The example below provides an implementation of this technique. It may be observed that \texttt{\arbcolor} is used so as to have the marks printed in red:—

\begin{verbatim}
1 \SetArbDflt*
2 \newarbmark{sly}{\arbcolor[red]{^^^^06d6}}{}
3 \newarbmark{jim}{\arbcolor[red]{^^^^06da}}{}
4 \begin{arab}
5 sUraTu 'l-nisA'i, 19:
6 \end{arab}
7 \begin{center}
8 \begin{arab}
9 \arbmark{bismillah}
10 \end{arab}
11 \end{center}
12 \begin{arab}[fullvoc]
13 y_a'yuyuhA 'lla_dina 'a'man\ NA 1A ya.hillu la-kum 'an tari_tUA
14 'l-nisA'a\ karha\N\arbmark{sly} wa-l\A ta\ dulu\-hun\ na li-ta_dhab\NA
15 bi-ba\ di\ A_'a'tayt\-mU-hun\ na 'illa 'an ya't\ina bi-fA.hi^saTiN
16 mudayyinaTin\arbmark{jim} wa-A\-sirU-hun\ na
17 bi-'l-ma\-ru\fA\arbmark{jim} fa-'in kar\-iht\-U-hun\ na fa-'as\_A\_{a}
18 'an takrah\A "say'a\N wa-ya\'g\ala 'l-l_ahU fi-hi _hayra\N
19 ka_tira\N ((19))
20 \end{arab}
21 \end{arab}[fullvoc]
\end{verbatim}
The ‘Zero width joiner’ character (U+200D)  The ‘Zero width joiner’ character (U+200D) belongs to the ‘General Punctuation’ block (range 2000–206F) of the Unicode standard. It is a non-printing character which, when it is placed between two characters that would for some reason not be connected, causes them to be printed in their connected forms.

It is encoded & in Arab Tex scheme.

In elegantly printed books where many of the letters are interwoven with one another so as to form ligatures, it may be convenient to bring the letters into line in some instances. In the following example, the ‘zero width joiner’ is used to prevent two adjacent letters, viz. سَ and ح from standing one above the other in the name of ʾIsḥāq (قٰحْسِإ): 37 —

\begin{arab}[fullvoc]
\ayah{123} {is\underline{.}\mathsf{h}_{a}qa}
\rende

An example follows:—

\begin{arab}[fullvoc]
\ayah{123} ۝١٢٣
\rende

6.1 The Qurʾān

This sub-part is destined to become a part of its own, as fine typesetting of Qurʾānic text is planned in the versions of arabtex to come in the medium-term. New functions and new Arabic modes will be available as arabtex will mature.

For the time being, \ayah{} is provided so as to typeset the number of the ʾāyah that it is referred to inside the dedicated mark—Unicode U+06DD: ۝—in Arabic script or inside parentheses in romanized Arabic:—

\ayah(123) ۝(123).

An example follows:—

\begin{arab}
\underline{الَّذِي} بَلَغَ عَمرَهُ بِالْأَرْضِ وَبَلَغَ عَمرَهُ بِالْأَرْضِ
\rende

37underline and highLight are taken from the lua-ul package which is loaded by arabtex. See Krüger (2020).
\maketitle

\section*{New feature}

\subsection*{7 Color}

\texttt{arabluatex} is able to render in color either words, parts of words or diacritics. As the techniques implemented in this section may lead to some complexity, the reader should first become well acquainted with the following points:\footnote{Regarding the colors themselves and the way new colors can be defined in addition to those that are already available, please refer to the \texttt{xcolor} package.}

- (a) The “pipe” character (\texttt{|}, section 4.5 on page 24);
- (b) ‘Quoting’ technique (section 4.4 on page 22), and more specifically ‘quoting the hamzah’ (on page 23);
- (c) Putting back on broken contextual analysis rules (section 4.6 on page 24);
- (d) Arabic marks (section 6 on page 33).

\texttt{\arbcolor} \hfill \texttt{\arbcolor} takes the text to be colored into \texttt{(color)} as an argument:—
As this example shows, \arbcolor has been used to render headings in red with the same encoding both in vocalized and in romanized Arabic. The same technique also applies to syllables inside words. arabluatex takes care of selecting the appropriate shape of the letters while coloring them:—

```
\begin{arab}
\arbcolor[red]{al-bābu 'l-ḫāmisu} fī ṭabaqāti 'l-ʾaṭibbāʾi 'llaḏīna kānū munḏu zamāni Ġālīnūsa wa-qarīb
\end{arab}
```

\begin{arab}[trans]
\begin{arab}[red]{al-bābu 'l-ḫāmisu} fī ṭabaqāti 'l-ʾaṭibbāʾi 'llaḏīna kānū munḏu zamāni Ġālīnūsa wa-qarīb
\end{arab}
```

7.1 Tricks of the trade

Diacritics Depending on the mode selected, either voc, novoc or fullvoc, coloring the diacritics requires more attention for the insertion of \arbcolor may prevent contextual analysis from being applied.
Furthermore, depending on the surrounding letters, the standard encoding of short vowels \( \langle u, a, i \rangle \) may result either in diacritics or in a connective 'alif with the waslah or its accompanying vowel. As for the sukūn, it is generated by contextual analysis. Thus applying colors to bare diacritics requires them to have specific encodings.

Table 8 gives the ArabTeX equivalents for the diacritics to be printed inside or just after \texttt{\textbackslash arabcolor}.

<table>
<thead>
<tr>
<th>Diacritic</th>
<th>Transliteration</th>
<th>ArabTeX notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>َـ</td>
<td>a a a a</td>
<td>.a</td>
</tr>
<tr>
<td>ُـ</td>
<td>u u u u</td>
<td>.u</td>
</tr>
<tr>
<td>ِـ</td>
<td>i i i i</td>
<td>.i</td>
</tr>
<tr>
<td>ْـ</td>
<td>o</td>
<td>.o</td>
</tr>
</tbody>
</table>

Table 8: ArabTeX diacritics for \texttt{\textbackslash arabcolor}

The following examples show how the letters, or the diacritics above or under them or both the letters and the diacritics can be rendered in different colors:

‘voc’ mode:

\texttt{i^staraytu-hubi-_taman\texttt{\textbackslash arabcolor[red]{iN}}'a~g\texttt{\textbackslash arabcolor[red]{}.a}ba-ka}  
\texttt{ištaraytu-hu bi-\texttt{\textbackslash arabcolor[red]{i}}'ağa\texttt{\textbackslash arabcolor[red]{'a}}}\texttt{ba-ka}.

‘fullvoc’ mode:

\texttt{i^stara\texttt{\textbackslash arabcolor[red]{y}}'otu-hubi-_taman\texttt{\textbackslash arabcolor[red]{iN}}'a~g\texttt{\textbackslash arabcolor[red]{}.a}ba-ka}  
\texttt{ištaraytu-hu bi-\texttt{\textbackslash arabcolor[red]{i}}'ağa\texttt{\textbackslash arabcolor[red]{'a}}}\texttt{ba-ka}.

39See below section 8 on page 41.
As can be seen, `fullvoc` required the letters `y`, `n` and `^g` before `\arbcolor` to be ‘quoted’. Otherwise, unwanted `sukūns` would have been generated because of the absence of a vowel after those consonants.

**tanwīn** \`\arbnull` must be used with `fathātān (ّ)` so as to put back on contextual analysis rules:

```
mu`allim\arbcolor[red]{\arbnull{m}aN}\{mu`allim\} "mũ`allim"nn,

istisqA\arbcolor[red]{\arbnull{A'}aN}\{A'stisqā\} \"istisqā\"nn,

^say\arbcolor[red]{\arbnull{ay'}aN}\{say\} "šay\"nn,

^gAmi`aT\arbcolor[red]{\arbnull{T}aN}\{gāmi`at\} "gāmi`at\"nn.
```

Ret. Note that in the last example (`gāmi`at\"nn`), the ‘pipe’ character has been inserted before \`\arbcolor`. Otherwise, the \texttt{dmg} mode of the transliteration rules would have interpreted the \texttt{tā’ marbūṭah} as final (e.g. `k` instead of the expected `t`).

The **tanwīn** preceding a ى conveys even more intricate business to the rendering with the utmost accuracy in both romanized and non-romanized modes. First, a new Arabic mark needs to be defined. It should print ى in Arabic script and not a thing in transliteration. It is to be appended after `\arbcolor`, like so:

```
1 \newarbmark(Y){---0649}\}
2 \arb{hud}\arbcolor[red]{\arbnull{A}}\arbmark(Y)}
3 \arb{trans}{hud}\arbcolor[red]{\arbnull{A}}\arbmark(Y)}
```

\textit{"hudi"}

**waṣlah and maddah** Both can be generated with the help of `\arbnull`:

```
wa\arbcolor[red]{\arbnull{wa}}i\{wa\} \textit{wa-}\textit{stisqā`un}41.

FI "al`.i\arbcolor[red]{\arbnull{1-i}}btid`i\{i\} \textit{fi}\ \textit{li-btid`a}\i\textit{.}

\arbcolor[red]{\arbnull{`a}}\arbnull{\{}kulu\{k\}\{\textit{.akulu},
\arbcolor[red]{\arbnull{`A}}\arbnull{\{}kulu\{k\}\{\textit{.ākil}nn.
```

The Unicode codepoint of the maddah is 0653, while bare ʾalif is 0627. So:

```
1 \newarbmark{alifmaddahred}(`---062\arbcolor[red](`---0653))\%
2 \{\arb{trans}{\arbnull{\}}{\arbnull{a}}\arbcolor[red]{\arbnull{\}}\}
3 \arb{\arbmark{alifmaddahred}kulu}
4 \arb{\arbmark{alifmaddahred}kulu}}.
```

\textit{See also on page 44 “Discarding the ʾi`rāb” for more information.}
\textit{41 To the knowledge of the writer, the waṣlah alone is not part of the Arabic Unicode block.}
Rem. In the preceding example, any consonant could have been passed as argument to the \arbnull command.

\textbf{šaddah}  In the following example, it is assumed that the šaddah above the letter ل, \textit{al-mu'allimūna}, is to be rendered in red. Thus the Arabic mark must generate the šaddah alone—of which the Unicode codepoint is 0651—in Arabic script and the letter ‘l’ in transliteration:

\begin{verbatim}
1 \newarbmark{lamshaddah}{0651}\{l\}
2 \arb\{fullvoc\}{al-mu' al}\arbcolor{red}{\arbmark{lamshaddah}}.imUna}
3 \arb\{trans\}{al-mu' al}\arbcolor{red}{\arbmark{lamshaddah}}.imUna}.
\end{verbatim}

The definite article and the euphonic tašdīd  The intricate business of rendering in color the initial 'alif al-waṣl of the definite article followed by a solar consonant must be unraveled.

From the examples provided above, in \textit{fi 'l-nāsi}, the initial 'alif’ 'l-waṣl' can be rendered in red like so: \arbcolor{red}{\arbnull{al-}a}.

Then, the following two letters, namely 1-n, must print the string \texttt{lām + nūn + šaddah} in Arabic, and exactly \texttt{n-n} in transliteration. Thus an Arabic mark is needed:

\begin{verbatim}
1 \newarbmark{lnn}{064406460651}{n-n}
2 \arb\{fullvoc\}{fi\arbnull{al-}}
3 \arbcolor{red}{\arbnull{al-}a}\arbmark{lnn}Asi}
4 \arb\{trans\}{fi\arbnull{al-}}
5 \arbcolor{red}{\arbnull{al-}a}\arbmark{lnn}Asi}.
\end{verbatim}

\textbf{hamzah}  The ‘quoting’ technique provides an easy way to determine the carrier of the hamzah, as shown in table 5 on page 24—:

\begin{verbatim}
yatas\arbnull{a}\arbcolor{red}{\"}.alUna  yatas‘a-
lina, "say\arbcolor{red}{\"}\\arbnnull{\"}N  šay\textsuperscript{un}, "say\ar bcolor{red}{\"}\\arbnnull{\"}N  šay\textsuperscript{un}, \arbcolor{red}{\"}\}N  šay\textsuperscript{un},\arbcolor{red}{\"}\}N  šay\textsuperscript{un},\arbcolor{red}{\"}\}N  šay\textsuperscript{un}.
\end{verbatim}
8 Transliteration

It may be more appropriate to speak of “romanization” than “transliteration” of Arabic. As seen above in section 2.2 on pages 6–9, the “transliteration mode” may be selected globally or locally.

This mode transliterates the Arab\TeX{} input into one of the accepted standards. As said above on page 6, three standards are supported at present; \texttt{dmg} \textit{Deutsche Morgenländische Gesellschaft}, which was adopted by the International Convention of Orientalist Scholars in Rome in 1935.\footnote{See Brockelmann et al. (1935).} \texttt{dmg} transliteration convention is selected by default;
\texttt{loc} \textit{Library of Congress}: this standard is part of a large set of standards for romanization of non-roman scripts adopted by the American Library Association and the Library of Congress;\footnote{See \url{http://www.loc.gov/catdir/cpso/roman.html} for the source document concerning Arabic language.} \texttt{arabica} \textit{Journal of Arabic and Islamic Studies/Revue d’études arabes et islamiques}: this standard is most widely used by scholars in the field of Arabic studies.\footnote{See \url{http://www.brill.nl/files/brill.nl/specific/authors_instructions/ARAB.pdf}.} More standards will be included in future releases of \texttt{arabluatex}.

\begin{verbatim}
\SetTranslitConvention
\end{verbatim}

\texttt{Convention} The transliteration mode, which is set to \texttt{dmg} by default, may be changed at any point of the document by the \texttt{\SetTranslitConvention{⟨mode⟩}} command, where \texttt{⟨mode⟩} may be either \texttt{dmg}, \texttt{loc} or \texttt{arabica}. This command is also accepted in the preamble should one wish to set the transliteration mode globally, e.g.:

\begin{verbatim}
1 \usepackage{arabluatex}
2 \SetTranslitConvention{loc}
\end{verbatim}

\begin{verbatim}
\SetTranslitStyle
\end{verbatim}

\texttt{Style} Any transliterated Arabic text is printed in italics by default. This also can be changed either globally in the preamble or locally at any point of the document by the \texttt{\SetTranslitStyle{⟨style⟩}} command, where \texttt{⟨style⟩} may be any font shape selection command, e.g. \texttt{\upshape}, \texttt{\itshape}, \texttt{\slshape}, and so forth.

\begin{verbatim}
\SetTranslitFont
\end{verbatim}

\texttt{Font} \texttt{\SetTranslitFont{⟨font selection command⟩}} allows any specific font to be selected for rendering transliterated text with the font-selecting commands of the \texttt{fontspec} or \texttt{luatexjs} package. Of course, this font must have been defined properly. To take one example, here is how the \textit{Gentium Plus} font can be used for rendering transliterated text:

\begin{verbatim}
1 \newfontfamily\translitfont{Gentium Plus}[Ligatures=TeX]
2 \SetTranslitFont{\translitfont}
\end{verbatim}
Proper names  Proper names or book titles that must have their first letters uppercased may be passed as arguments to the \uc{\textit{word}} command. \uc is a clever command, for it will give the definite article \textit{al-} in lower case in all positions. Moreover, if the initial letter, apart from the article, cannot be uppercased, viz. ‘or’, the letter next to it will be uppercased:—

\begin{verbatim}
\uc{\text{	extit{Hunayn}}} bn\text{\textit{n}}
\uc{\text{	extit{Ulmān}}} bn\text{\textit{a}}
\uc{\text{	extit{Zayd}}} bn\text{\textit{u}}
\uc{\text{	extit{Abd}}} bn\text{\textit{i}}
\uc{\text{	extit{Llāh}}} in \text{\textit{Saʿd}}
\end{verbatim}

However, \uc must be used cautiously in some very particular cases, for the closing brace of its argument may prevent a rule from being applied. To take an example, as seen above on page 20, the transliteration of \textit{Muḥammad\text{\textit{un}} ‘n-nabī}, as nouns having the \textit{tanwīn} take a \textit{kasrah} in pronunciation before \textit{ʾalifu ‘l-waṣli}. In that case, encoding \textit{Muḥammad} like so: \begin{verbatim}\uc\text{\textit{m}}u\text{\textit{hammaduN}}\end{verbatim} is wrong, because the closing brace would prevent arabluatex from detecting the sequence \textit{⟨\textit{-uN}⟩} immediately followed by \textit{⟨\textit{‘l-}⟩}. Fortunately, this can be circumvented in a straightforward way by inserting only part of the noun in the argument of \uc vz. up to the first letter that is to be uppercased, like so: \begin{verbatim}\uc\text{\textit{m}}u\text{\textit{h}}a\text{\textit{maduN}}\end{verbatim}.

Hyphenation  In case transliterated Arabic words break the TeX hyphenation algorithm, one may use the \texttt{-} command to insert discretionary hyphens. This command will be discarded in all of the Arabic modes of arabluatex, but will be processed by any of the transliteration modes:—

\begin{verbatim}
\uc\text{\textit{Abū Bakr}} in \text{\textit{Muḥammad}} bn \text{\textit{Zakariyyāʾ}}
\end{verbatim}

\begin{comment}
\texttt{-} command
\begin{verbatim}
\uc\text{\textit{Abū Bakr}} in \text{\textit{Muḥammad}} bn \text{\textit{Zakariyyāʾ}}
\end{verbatim}
\end{comment}

\textbf{‘Long’ proper names}  \uc is also able to process proper names consisting of several subsequent words:—

\begin{verbatim}
\uc\text{\textit{Abū Zayd}} in \text{\textit{Ḫunaynu bnu ʾIsḥāqa ‘l-‘Ibādiyyu}}
\end{verbatim}

\textbf{Proper names outside Arabic environments}  Transliterated proper names inserted in paragraphs of English text should be printed in the same typeface as the surrounding text. \prname{\textit{(Arabic proper name)}} is provided to that effect: \textsuperscript{\ref{footnote:prname}} —

\textsuperscript{\ref{footnote:prname}}  Just as \uc, \prname is also able to process proper names consisting of several subsequent words.
From Wright (1896, i. 23 C):— If the name following ʿin be that of the mother or the grandfather, the ʾ is retained; as ʿIrāb ibnu Maryama, “Jesus the son of Mary”; ʿAmmār ibnu Manṣūr, “Ammār the (grand)son of Manṣūr”.

The following example shows how \prname can be used in conjunction with the nameauth package to have Arabic proper names printed first in full then in partial forms:\[1\]—

\begin{quote}
\begin{verbatim}
\begin{nameauth}
\begin{verbatim}
\text{Hunayn} & \prname{ʾAbū Zayd} & \prname{Ḥunayn}, \prname{i}bn 'Isḥāq al-ʿIbādiyy} & > \\
\text{Razi} & \prname{ʾAbū Bakr Muḥammad ibn Zakariyyāʾ} & \prname{al-Rāzī} & > \\
\end{verbatim}
\end{nameauth}
\end{verbatim}
\end{quote}

\prname* \text{ Remark: arabluatex also provides \prname* which only renders in upright roman style already transliterated proper names without applying any further processing. It is mostly used internally and applied to proper names exported in Unicode to an external selected file.} \[2\]

\section{Additional note on dmg convention}

According to Brockelmann et al. (1935, p. 6), Arabic ʿrāb may be rendered into dmg in three different ways:
(a) In full: ʿAmrun;
(b) As superscript text: ʿAmr”;
(c) Discarded: Amr.

By default, arabluatex applies rule (b). Once delimited by a set of Lua functions, ʿrāb is passed as an argument on to a \arbup command which is set to \textsuperscript.

\NoArbUp \text{ \NoArbUp may be used either in the preamble or at any point of the document}

\ArbUpDflt

\text{46See the documentation of nameauth for more details: https://ctan.org/pkg/nameauth}

\text{47See below section 12 on page 57 for more details.}
in case one wishes to apply rule (a). The default rule (b) can be set back with \ArbUpDflt at any point of the document.

Finally, \SetArbUp{formatting directives} can be used to customize the way ʾiʿrāb is displayed. To take one example, here is how Arabic ʾiʿrāb may be rendered as subscript text:—

\textsubscript{#1}

Arabic dmg transliteration for raʿaytu ǧāmiʿan muḥaddamat₂₃, miʿdanatu-hu.

As shown in the above example, #1 is the token that is replaced with the actual tanwīn in the formatting directives of the \SetArbUp command.

### ʾiʿrāb boundaries
Every declinable noun (muʿrāb) may be declined either with or without tanwīn, viz. munṣarif₃₄ or ḡayr munṣarif₅₆. The former is automatically parsed by arabluatex, whereas the latter has to be delimited with an hyphen, like so:

muʿrāb: mu`allimuN, kAʾinuN, qA. diNī

ḡayr munṣarif: al-mu`allim-u, kitAb-Ani, raʾsaʾ-Ani, sAriq-Una, qA.d-Una, al-.zulm-Atu

Rem. a. As the tanwīn is passed over in pronunciation when it is followed by the letters ر, ل, م, و, ي (see item (b) on page 18), it may be desirable to further distinguish it by putting it above the line, but not to do the same for ḡayr munṣarif terminations. This can be achieved by simply omitting the hyphen before any ḡayr munṣarif termination:

kāna ʾganiyyān l_akinna-hu labīsā ḡubbatan mubāya ʾaydu-hā

Rem. b. Although the hyphen before the tanwīn is optional as arabluatex always parses nouns with such termination, it may also be used to mark better the inflectional endings:

manaʿa ʾl-nās-a ʿaffaT-nā min mu_hā.tabati-hi ʾa.had-uN bi-sayyidi-nā

Discarding the ʾiʿrāb
As said above (item (c) on the preceding page), the ʾiʿrāb may be discarded in some cases, as in transliterated proper names or book titles. arabluatex is able to render words ending with tāʾ marbūṭah in different ways, depending on their function:—
(a) Nouns followed by an adjective in apposition: 
\[
\text{madīnāT kabīraT madīnah kabīrah, al-madīnāT al-kabīraT al-madīnah al-kabīrah.}
\]
(b) Nouns followed by another noun in the genitive (contract state): 
\[
\text{.hikmaT al-l_ah ḥikmat Allāh, fi.d.daT al-darāhim fīḍdat ad-darāhim.}
\]

Rem. It may so happen, as in the absence of the article before the annexed word, that arabluatex be unable to determine which of the above two cases the word ending with tāʾ marbūṭah falls into. The ‘pipe’ character (see section 4.5 on page 24) may be appended to that word to indicate that what follows is in the construct state: \[\text{uc(r)isālaT fī tartīb qirāʾaT} \] kutub \[\text{uc(`}g\text{)AllūnūS Risālah fī tartīb qirāʾ ut kutub ġālinūs.}\]

Uncertain short vowels

In some printed books, it may happen that more than one short vowel be placed on a consonant in cases where the vocalization is uncertain or ambiguous, like so: ُعَف. In transliteration, the uncertain vowels go between slashes and are separated by commas: fa`uaila ُعَف fa`/u,a,i/la.

8.2 Examples

Here follows in transliteration the story of Īḫā and his donkey (ُهُراَمِحَواَحُج). See the code on page 9:—


9 Buckwalter input scheme

Even though arabluatex is primarily designed to process the ArabTEX notation, it
can also process the Buckwalter input scheme to a large extent.\textsuperscript{48} The Buckwalter scheme is actually processed in two steps, as it is first converted into \texttt{ArabTEX}. Then, once this is accomplished, the \texttt{ArabTEX} scheme is processed through the above described functions. In this way, the Buckwalter input scheme can make the most of the \texttt{arabluatex} special features that are presented in section 2.2 on page 6.

The input scheme, which is set to \texttt{arabtex} by default, may be changed at any point of the document by the \texttt{\SetInputScheme{⟨scheme⟩}} command, where \texttt{⟨scheme⟩} may be either \texttt{arabtex} or \texttt{buckwalter}. This command is also accepted in the preamble should one wish to set the input scheme globally, like so:—

\begin{verbatim}
1 \usepackage{arabluatex}
2 \SetInputScheme{buckwalter}
\end{verbatim}

\textbf{‘base’, ‘xml’ and ‘safe’ schemes} \texttt{arabluatex} can use any of the so-called Buckwalter ‘base’, ‘xml’ or ‘safe’ schemes as they are described in Habash (2010, pp. 25–26).\textsuperscript{49} However, the following limitation apply to the ‘base’ and ‘xml’ schemes: the braces \{ and \}, which are used to encode ٞ and ٛ, must be replaced with square brackets viz. [ and ] respectively.

It is therefore recommended to use the Buckwalter ‘safe’ scheme.

Table 9 gives the Buckwalter equivalents that are currently used by \texttt{arabluatex}. The additional characters that are defined in table 6 on page 27 are also available.

\begin{table}[h]
\begin{center}
\begin{tabular}{|c|c|c|c|c|c|}
\hline
\textbf{Letter} & \textbf{Transliteration}\textsuperscript{50} & \textbf{Buckwalter notation} \\
& \texttt{dmg loc arabica base/xml safe} & \\
\hline
\textbackslash & a & a & a & A & A \\
\textbackslash & b & b & b & b & b \\
\textbackslash & t & t & t & t & t \\
\textbackslash & th & th & th & v & v \\
\textbackslash & ţ & ţ & ţ & j & j \\
\textbackslash & ĥ & ĥ & ĥ & H & H \\
\textbackslash & h & h & h & x & x \\
\textbackslash & d & d & d & d & d \\
\textbackslash & d & dh & dh & * & V \\
\textbackslash & r & r & r & r & r \\
\textbackslash & z & z & z & z & z \\
\textbackslash & s & s & s & s & s \\
\textbackslash & ť & ť & ť & $ & $ \\
\textbackslash & ș & ș & ș & S & S \\
\textbackslash & ț & ț & ț & T & T \\
\hline
\end{tabular}
\end{center}
\caption{Buckwalter scheme}
\end{table}

\textsuperscript{48}See http://www.qamus.org/transliteration.htm
\textsuperscript{49}I am grateful to Graeme Andrews who suggested that the ‘safe’ scheme be included in \texttt{arabluatex}.
\textsuperscript{50}See section 8 on page 41.
<table>
<thead>
<tr>
<th>Letter</th>
<th>Transliteration</th>
<th>Buckwalter notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ظ</td>
<td>ظ</td>
<td>Z  Z</td>
</tr>
<tr>
<td>ء</td>
<td>ء</td>
<td>E  E</td>
</tr>
<tr>
<td>َ</td>
<td>َ</td>
<td>G  G</td>
</tr>
<tr>
<td>َ</td>
<td>َ</td>
<td>F  F</td>
</tr>
<tr>
<td>َ</td>
<td>َ</td>
<td>k  k</td>
</tr>
<tr>
<td>َ</td>
<td>َ</td>
<td>l  l</td>
</tr>
<tr>
<td>َ</td>
<td>َ</td>
<td>m  m</td>
</tr>
<tr>
<td>َ</td>
<td>َ</td>
<td>n  n</td>
</tr>
<tr>
<td>َ</td>
<td>َ</td>
<td>h  h</td>
</tr>
<tr>
<td>َ</td>
<td>َ</td>
<td>w  w</td>
</tr>
<tr>
<td>َ</td>
<td>َ</td>
<td>y  y</td>
</tr>
<tr>
<td>َ</td>
<td>َ</td>
<td>ah ah a</td>
</tr>
<tr>
<td>َ</td>
<td>َ</td>
<td>C  C</td>
</tr>
<tr>
<td>َ</td>
<td>َ</td>
<td>a a a</td>
</tr>
<tr>
<td>َ</td>
<td>َ</td>
<td>u u u</td>
</tr>
<tr>
<td>َ</td>
<td>َ</td>
<td>i i i</td>
</tr>
<tr>
<td>َ</td>
<td>َ</td>
<td>an an an</td>
</tr>
<tr>
<td>َ</td>
<td>َ</td>
<td>um um um</td>
</tr>
<tr>
<td>َ</td>
<td>َ</td>
<td>in in in</td>
</tr>
<tr>
<td>َ</td>
<td>َ</td>
<td>o o o</td>
</tr>
<tr>
<td>َ</td>
<td>َ</td>
<td>ā ā ā</td>
</tr>
<tr>
<td>َ</td>
<td>َ</td>
<td>(tašīl) - - - -</td>
</tr>
</tbody>
</table>

Table 9: Buckwalter scheme

**Transliteration**  The Buckwalter notation can also be transliterated into any accepted romanization standard of Arabic. See above section 8 on page 41 for more information. However, it should be pointed out again that only accurate coding
produces accurate transliteration. It is therefore at the very least highly advisable to use the hyphen for tying the definite article and the inseparable particles (viz. prepositions, adverbs and conjunctions) to words, like so:—

\begin{itemize}
\item Al-EaAlamu \(\text{ُمَلاَعلَا}\) al-\(\text{ُمَلاَعالَا}\), Al-camsu \(\text{ُسمَّشلَا}\) aš-\(\text{ُسمَّشلَا}\), bi-SinaAEapi \(\text{ٌّبِّطلاِةَعاَنِصِب}\) \(\text{bi-\(\text{ٌّبِّطلاِةَعاَنِصِب}\) i} \'ṭ-\text{ṭibb}\).
\item wa-Al-l\(\text{ِهّٰللاَو}\) wa-\(\text{ِهّٰلِلُدمَحلَا}\), Al-Hamdu li-l\(\text{ِهّٰلِلُدمَحلَا}\) li-\(\text{ِهّٰلِلُدمَحلَا}\).
\end{itemize}

Similarly, it is not advisable to use | and [ ('base' and 'xml' schemes) or M and L ('safe' scheme) to encode the 'alif' 'l-mamdūdat' and the 'alif' 'l-waṣl' for such signs are supposed to be generated by arabluatex internal functions. Besides, as they do not per se convey any morphological information on what they are derived from, they cannot be transliterated accurately. To take one example, <ilY Al-LntiqaADi gives \(\text{ِضاَقِتنٱلاىَلِأ}\) as expected, but only <ilY Al-intiqADi can be transliterated as \(\text{ʾilạ} \ 'l-intiqāḍi\) with the correct vowel \(\langle i \rangle\) in place of the 'alif' 'l-waṣl'.

10 Unicode Arabic input

As said above in section 9 on page 45 about the Buckwalter input scheme, even though arabluatex is primarily designed to process the ArabTEX notation, it also accepts Unicode Arabic input. It should be noted that arabluatex does in no way interfere with Unicode Arabic input: none of the voc, fullvoc, novoc or trans options will have any effect on plain Unicode Arabic for the time being.

That said, there are two ways of inserting Unicode Arabic:

\begin{itemize}
\item \texttt{txarb} (a) The \texttt{txarb} command for inserting Unicode Arabic text in paragraphs;
\item \texttt{txarab} (b) The \texttt{txarab} environment for inserting running paragraphs of Arabic text, like so:—
\end{itemize}

\begin{verbatim}
\begin{txarab}
<Unicode Arabic text>
\end{txarab}
\end{verbatim}

11 \LaTeX{} Commands in Arabic environments

**General principle** \LaTeX{} commands are accepted in Arabic environments. The general principle which applies is that any single-argument command with up to two optional arguments—that is: \texttt{\backslash command[(opt1)][(opt2)]{arg}}—such as \texttt{\textbf{\texttt{(text)}}}, \texttt{\textbf{\texttt{(text)}}} and the like, is assumed to have Arabic text in its mandatory argument:
The same applies to footnotes:—

\begin{arab}
\[\text{fullvoc}\]
\begin{lr}{\text{\textbf{zayduN}\textsc{ibnu}\text{\textbf{a}mr-iNU}}}
\end{arab}

Some commands, however, do not expect running text in their arguments, or one may wish to insert English text e.g. in footnotes or in marginal notes. arablatex provides a set of commands to handle such cases.

\LR\{\textit{arg}\}\} is designed to typeset its argument from left to right. It may be used in an Arabic environment, either \arb\{\textit{Arabic text}\} or \begin{arab}\{\textit{Arabic text}\}\end{arab}, for short insertions of left-to-right text, or to insert any \LaTeX\ command that would otherwise be rejected by arablatex, such as commands the argument of which is expected to be a dimension or a unit of measurement.

\RL\{\textit{arg}\}\{\textit{text}\}\} does the same as \LR\{\textit{arg}\}\}, but typesets its argument from right to left. Even in an Arabic environment, this command may be useful.

\LR\footnote{\textit{text}} and \RL\footnote{\textit{text}} typeset left-to-right and right-to-left footnotes respectively in Arabic environments. Unlike \footnote{\textit{text}}, the arguments of both \LR\footnote{\textit{text}} and \RL\footnote{\textit{text}} are not expected to be Arabic text. For example, \LR\footnote{\textit{text}} can be used to insert English footnotes in running Arabic text:—

\begin{arab}[fullvoc]
\begin{lr}{\text{\textbf{zayduN}\textsc{ibnu}\text{\textbf{a}mr-iNU}}}
\end{arab}

\textit{zayduN}\textsc{ibnu}\text{\textbf{a}mr-iNU}

This is odd in Arabic script, but using such features as \textbf{emph} or \textbf{emph} or \textbf{textbf} is a matter of personal taste.

\RLframebox has been adapted from \framebox for insertions of right-to-left text.
When footnotes are typeset from right to left, it may happen that the numbers of
the footnotes that are at the bottom of the page be typeset in the wrong direction.
For example, instead of an expected number 18, one may get 81. arabluatex is not
responsible for that, but should it happen, it may be necessary to redefine in the
preamble the \textsuperscript{\texttt{thefootnote}} macro like so:—
\begin{verbatim}
\renewcommand*{\thefootnote}{\textsuperscript{\LR{\arabic{footnote}}}}
\end{verbatim}
Another solution is to put in the preamble, below the line that loads arabluatex, the
\texttt{\FixArbFtnmk} command. However, for more control over the layout of footnotes
marks, it is advisable to use the \texttt{scrextend} package.53
The \texttt{\LRmarginpar} command does for marginal notes the same
as \texttt{\LRfootnote} does for footnotes. Of course, it is supposed to be used in Arabic
environments. Note that \texttt{\marginpar} also works in Arabic environments, but it
acts as any other single-argument command inserted in Arabic environments. The
general principle laid on page 48 applies.
\texttt{\setRL} and \texttt{\setLR} can be used to change the direction of paragraphs, either
form left to right or from right to left. As an example, an easy way to typeset a
right-to-left sectional title follows:—
\begin{verbatim}
1 \setRL
2 \section*{\arb{barzawayhi li-buzurjumihra bn-i 'l-buxtikAni}}
3 \setLR
4 \begin{arab}
5 qAla barzawayhi bn-u 'azhar-a, ra's-u 'a.tibbA'-i fAris-a...
6 \end{arab}
\end{verbatim}

11.1 New commands

In some particular cases, it may be useful to define new commands to be inserted
in Arabic environments. From the general principle laid on page 48, it follows that
any command that is found inside an Arabic environment is assumed to have Arabic

53See \url{http://ctan.org/pkg/koma-script}; read the documentation of KOMA-script for details about
the \texttt{\deffootnotemark} and \texttt{\deffootnote} commands.
text in its argument which arabluatex will process as such before passing it on to the command itself for any further processing. As a result of this feature, such a command as:

```
\newcommand{\fvarabic}{\arb[fullvoc]{\#1}}
```

will work as expected, but will always output non-vocalized Arabic if it is inserted in a novoc Arabic environment because its argument will have been processed by the novoc rules before the command \fvarabic itself can see it.

The \MkArbBreak\{\csvlist\} command can be used in the preamble to give any command—either new or already existing—the precedence over arabluatex inside Arabic environments. It takes as argument a comma-separated list of commands each of which must be stripped of its leading character \, like so:—

```
\MkArbBreak{onecmd, anothercmd, yetanothercmd, ...}
```

For example, here follows a way to define a new command \fvred to distinguish words with a different color and always print them in fully vocalized Arabic:—

```
\begin{arab}{voc}
tumma "intalaqa _dU 'l-qarn-ayni 'il_A 'ummaT-iN 'u_hr_A fI
_fvred{((ma.tli`-i 'l-^sams-i))} wa-lA binA'-a la-hum
yu'amminu-hum mina 'l-^sams-i.
\end{arab}
```

It must be noted that the arguments, either optional or mandatory, of commands declared with \MkArbBreak are not to be processed by arabluatex. Therefore, as in the previous example, any of their argument to be rendered in Arabic must be inserted again in \arb. These commands themselves may have up to two optional and/or mandatory arguments followed by one optional argument, like so:—

(a) \command{\opt1} (no argument, lowermost combination)
(b) \command[\opt1]{\opt2} (one optional argument)
(c) \command[\opt1]{\opt2}{\opt3} (one mandatory argument)
(d) \command[\opt2]{\opt1}{\opt3} (one optional and one mandatory argument)
(e) [...]
(f) \command[\opt1][\opt2]{\opt3} (uppermost combination)

As said above, \MkArbBreak prevents arabluatex from processing the arguments of ‘declared’ commands as Arabic text. This technique proves sufficient in most cases. However, a ‘starred’ version of this command—\MkArbBreak*\{\csvlist\}—is also provided. It goes a step further, as it directs arabluatex to close the current Arabic environment before any of the ‘declared’ commands, then resume it just after.
It must be noted that \texttt{\textbackslash MkArbBreak*} must be used with the utmost care and should never be used if \texttt{\textbackslash MkArbBreak} gives satisfaction. At any rate, the latter must always be tested before the former.

### 11.2 Environments

Environments such as \texttt{\begin{quote} ... \end{quote}} may be nested inside the \texttt{arab} environment. Up to one optional argument may be passed to each nested environment, like so:

```
\begin{arab}
\begin{quote}[fullvoc]
\begin{arab}[fullvoc]

In the following example, the quoting package is used:

```
\setquotestyle{arabic}
\begin{arab}[fullvoc]

كان أبو الهلال أهدي إلى موسي داجة. كانت داججة أنها أهداوه دون ما كان يصف لوجه. ولكنه يحكم ويحسن خلقه أظهر العجب من جمله وطيب شرحه. وكان أبو الهلال يعرف بالامساك النشيد.

فقال: "وكيف رأيت يا أبا عمان تلك الداججة؟" قال: "كانت عجبا من العجب" فقال: "وقدري ما حسن؟ وقيدري ما سبب؟ فإن الداججة فإن تطيب النشيد والطيب، وقيدري يأتي غور؟ كأ يشمس؟ وفي أي مكان كأ علمنا؟"

 فلا يزال في هذا والآخر يضغطك ضحكة تعرفه تعرفه. ولا يعرفه أبو الهلال.

```
11.2.1 Lists

Lists environments are also accepted inside the `arab` environment. One may either use any of the three standard list environments, viz. `itemize`, `enumerate` and `description` or use packages that provide additional refinements such as `paralist` or `enumitem`.

To take a first example, should one wish to typeset a list of manuscripts, the `description` environment can be used like so:—

```latex
\setRL\paragraph{\arbnovoc{(rumUzi 'l-kitAbi)}}\setLR
\begin{arab}
\begin{description}
\item[b] max.tU.tu 'l-maktabaTi 'l-'ahliyyaTi bi-'uc{bArIs} 2860
\item[s] max.tU.tu 'l-maktabaTi 'l-'ahliyyaTi bi-'uc{bArIs} 2859
\item[m] max.tU.tu majlisi \arbnovoc{'sUrAY malY} .tahrAna 521.
\end{description}
\end{arab}
```

As a second example, the contents of a treatise may be typeset with the standard list environments, like so:—

```latex
\setRL\centerline{\arbnovoc{al-qAnUnu fI 'l-.tibbi}}\setLR
\begin{arab}
\begin{itemize}
\item \textbf{al-fannu 'l-'awwalu} fI 'l-.tibbi wa-maw.dU`Ati-hi mina 'l-'umUri 'l-.tabI`iyyaTi wa-yastamilu 'al_A sittaTi ta`AlImiN
\begin{itemize}
\item \textbf{al-ta`lImu 'l-'awwalu} [wa-huwa fa.slAni]
\begin{itemize}
\item \textbf{al-fa.slu 'l-'awwalu}
\end{itemize}
\end{itemize}
\end{itemize}
\end{arab}
```
As a third example, abjad-numbered lists can be typeset in conjunction with the `enumitem` package, like so:

```
\begin{verbatim}
\% preamble:---
usepackage(enumitem)
newlist(enumabjad)[enumerate]{10}
setlist[enumabjad]{nosep, label={\abjad{\arabic*}}}
usepackage{multicol}

From \textcite{i. 29 B--C}{Wright}:--- The derived forms of the triliteral verb are usually reckoned fifteen in number, but the learner may pass over the last four, because (with the exception of the twelfth) they are of very rare occurrence.
\begin{multicols}{3}
\begin{arab}[fullvoc]
\begin{enumerate}
\item fa`ala
\item fa``ala
\item fA`ala
\item 'af`ala
\item tafa``ala
\item tafA`ala
\item infa`ala
\item ifta`ala
\item if`alla
\item istaf`ala
\item if`Alla
\item if`aw`ala
\item if`awala
\item if`anlala
\item if`anl_A
\end{enumerate}
\end{arab}
\end{multicols}
\end{verbatim}
```

From Wright (1896, i. 29 B--C):— The derived forms of the triliteral verb are usually reckoned fifteen in number, but the learner may pass over the last four, because (with the exception of the twelfth) they are of very rare occurrence.

54 See the documentation of `enumitem` for more details: [https://ctan.org/pkg/enumitem](https://ctan.org/pkg/enumitem)
Caveat  The various French definition files of the babel package viz. acadian, canadien, francais, french or french must redefine the list environments, which breaks the standard definition file that is used by arabluatex. Therefore, babel-french must be loaded with the StandardLists=true option, like so:—

1 \usepackage{french}{babel}
2 \frenchsetup{StandardLists=true}

This option will prevent babel-french from interfering with the layout of the document. Then the paralist or enumitem packages can be used to make the lists ‘compact’ as babel-french do.

11.3 csquotes

The recommended way of inserting quotation marks in running Arabic text is to use csquotes. With the help of the \DeclareQuoteStyle command, one can define an Arabic style, like so:—

1 \usepackage{csquotes}
2 \DeclareQuoteStyle{arabic}
3 {\textquotedblright}{\textquotedblleft}
4 {\textquoteright}{\textquoteleft}

Then, use this newly defined style with \setquotestyle, like so:—

1 \setquotestyle{arabic}
2 \begin{arab}
3 fa-qi\textcolor{red}{\em}la\textcolor{red}{\em} hu ju.\textcolor{red}{\em} h\textcolor{red}{\em}a: \enquote{.garIb-u\textcolor{red}{\em}b 'amru-ka yA .sadIqI
4 'a-tu.saddiqu 'l-.himAr-a wa-tuka_d_diba-nI?}
5 \end{arab}
6 \setquotestyle{english}

Rem. Do not forget to set back the quoting style to its initial state once the Arabic environment is closed. See the last line in the code above.

11.4 Two-argument special commands

textcolor  The two-argument command \textcolor{⟨color⟩}{⟨Arabic text⟩} is supported inside \begin{arab} ... \end{arab}. One simple example follows:55—

1 \begin{arab}
2 \textcolor{red}{\uc{m}uha_d_dabu \uc{1-d}Ini \uc{a}bdu}
3 \uc{1-r}{a}.hImi bnu \uc{a}liyyI\textcolor{red}{\em} huwa ^say_hu-n\textcolor{red}{\em}a 'l-'imAmu

\footnote{arabluatex provides its own \arbcolor command which is able to render syllabes or diacritics in colors. See section 7 on page 36.}

11.5 quran

arabluatex is compatible with the quran package so that both can be used in conjunction with one another for typesetting the Qurʾān. As quran draws the text of the Qurʾān from a Unicode encoded database, its commands have to be passed as arguments to the \txarb command for short insertions in left-to-right paragraphs, or inserted inside the txarab environment for typesetting running paragraphs of Qurʾānic text (see above section 10 on page 48 for more details). Please note that arabluatex takes care of formatting the Arabic: therefore, it is recommended to load the quran package with the \nopar option, after arabluatex itself has been loaded, like so:

\begin{arab}
\textcolor{red}{\begin{arab}
\begin{textcolor{red}{\begin{arab}
\end{arab}}}
\end{arab}}\end{arab}
\end{arab}
As an example, the following code will typeset the *sūrat al-Fātiḥah*:

\begin{txarab}
\quransurah[1]
\end{txarab}

\begin{verbatim}
٤﴾نيّدلاِموَيِكِلام﴾٣﴿ِميحَّرلاِنٰمحَّرلا﴾٢﴿َنيمَلاعلاِّبَرِهَّلِلُدمَحلا﴾١﴿ِميحَّرلاِنٰمحَّرلاِهَّللاِمسِبِبوضغَملاِريَغمِهيَلَعَتمَعنَأَنيذَّلاَطارِص﴾٦﴿َميقَتسُملاَطارِّصلااَنِدها﴾٥﴿ُنيعَتسَنَكاّيِإَوُدُبعَنَكاّيِإ﴾٧﴿َنيّلاّضلااَلَومِهيَلَع
\end{verbatim}

12 Exporting Unicode Arabic to an external file

arabluatex is able to produce a duplicate of the original .tex source file in which all arabtex or buckwalter strings will have been replaced with Unicode equivalents, either in Arabic script or in any accepted standard of transliteration. Exporting ascii strings to Unicode while preserving the exact selected global or local options is a fairly complex operation which may require LuaLaTeX to be run several times as will be explained below.

12.1 Commands and environments

**export**  
**export global option**  
First, arabluatex must be loaded with the export global option enabled, like so:—

\begin{verbatim}
\% preamble
\usepackage[export]{arabluatex}
\% or:
\usepackage[export=true]{arabluatex}
\end{verbatim}

Once that is done, compiling the current file will produce a new empty external .tex file with the same preamble as the original file.

**\SetArbOutSuffix**  
By default, _out is appended as a suffix to the external file name. Any other suffix may be set with the command \SetArbOutSuffix{⟨suffix⟩}.

**arabexport**  
**Exporting running paragraphs**  
Then, the arabexport environment is provided to actually exporting running paragraphs with or without Arabic environments to the external selected file, like so:—

57 See above on page 6 for more information.
\begin{arabexport}

\textit{Running paragraphs of either Arabic or non-Arabic text}

\end{arabexport}

\texttt{arabluatex} converts to Unicode and writes to the external file what is found inside Arabic environments. As to non-Arabic text, it is appended untouched to this file, which is formatted as follows:

(a) Unicode Arabic text, either in Arabic script or in transliteration, is inserted as argument of $\texttt{txarb}$\footnote{See above section \ref{section:txarb} on page \pageref{section:txarb}.} or $\texttt{txtrans}$\footnote{$\texttt{txtrans}$ is used internally by several Lua functions to format transliterated Arabic. Therefore, it is not documented.} accordingly.

(b) Additionally, Arabic paragraphs may receive $\texttt{arbpardir}$, which \texttt{arabluatex} uses to determine the direction of Arabic paragraphs to be set by default, or either $\texttt{setRL}$ or $\texttt{setLR}$ depending on what may have been set locally.\footnote{See above on page \pageref{setRL}.}

(c) Proper names are inserted as arguments of $\texttt{prname}$\footnote{See above on page \pageref{prname}.}

Appending words or commands to the external file only $\texttt{ArbOutFile}\langle (\texttt{newline}) \rangle$

\texttt{ArbOutFile*}\{\langle \texttt{argument} \rangle\} silently exports its argument to the external file. It may take the string \texttt{newline} as an optional argument, in which case a carriage return is appended to the contents of the argument. $\texttt{ArbOutFile*}\langle (\texttt{newline})\rangle\{\langle \texttt{argument} \rangle\}$ does the same as $\texttt{ArbOutFile}$, but also inserts its argument into the current .\texttt{tex} source file.

Exporting Arabic poetry Lines of Arabic poetry are exported as described above on page \pageref{export-arabic-poetry} when the export option that is specific to the \texttt{arabverse} environment is set to true. As a result of this particular feature, \texttt{arabverse} environments must be left outside $\begin{arabexport}$ ... $\end{arabexport}$.

Please note that inside \texttt{arabverse} environments \texttt{bayt} is replaced with $\texttt{bayt*}$\footnote{See above note \ref{note:bayt} on page \pageref{note:bayt} for more information.}

12.2 Nested Arabic environments

The exporting mechanism described above converts only the outermost level of nested Arabic environments. This may be sufficient in some cases, but if nested Arabic environments be found in the original .\texttt{tex} source file, then the Unicode converted file must be opened and compiled in turn, and so on until the innermost Arabic environment be converted and exported. In such cases, \texttt{arabluatex} issues a warning, so that authors do not have to check the entire file that just has been exported:

\begin{arabluatex} Warning: There are still \texttt{'arabtex'} strings to be converted. Please open \langle jobname\rangle<suffix>.\texttt{tex} and compile it one more time.\end{arabluatex}

Where \langle \texttt{jobname} \rangle is the name of the original .\texttt{tex} source file, and \langle \texttt{suffix} \rangle the suffix appended to the file that is to be opened and compiled again.
12.3 Further processing of Unicode converted files

Unicode files can be further processed by document converters such as John McFarlane’s pandoc\(^{63}\). To take here one simple example, here is how file_out.tex can be converted from LuaLaTeX into Open Document format (.odt):

```
1 pandoc file_out.tex -s -o file_out.odt
```

However, specific commands such as \texttt{txarb}, \texttt{txtrans} or \texttt{prname*}, which are not known to pandoc, must be redefined explicitly in the preamble to prevent the converter from gobbling their arguments, like so:

```
1 \% preamble:
2 \usepackage{arabluatex} % note that ‘export’ has been removed
3 \renewcommand{\txarb}{[1]{#1}}
4 \renewcommand{\txtrans}{[1]{\textit{#1}}}
5 \renewcommand{\arbup}{[1]{\textsuperscript{#1}}}
6 \% now that \texttt{prname{}} has been replaced with \texttt{prname{*}{}} it should
7 \% be safe to say:
8 \renewcommand{\prname}{[2]{#2}}
9 % &c
```

13 Future work

A short, uncommented, list of what is planned in the versions of arabluatex to come follows:

(a) Short-term:
   i. TEI xml support: arabluatex will interoperate with TEI xml through new global and local options that will output Arabic in a TEI xml compliant file in addition to the usual PDF output: see on page 4.

(b) Medium-term:
   i. More languages: the list of supported languages will eventually be the same as arabtex: see note 4 on page 5.
   ii. Formulate propositions for extending the ArabTeX notation and the transliteration tables. Include them in arabluatex. See section 4.9 on page 27.

14 Implementation

The most important part of arabluatex relies on Lua functions and tables. Read the .lua files that accompany arabluatex for more information.

```
1 \RequirePackage{iftex}
```

arabluatex requires LuaLaTeX of course. Issue a warning if the document is processed with another engine.

\textsuperscript{63}See http://pandoc.org/
\RequireLuaTeX

Declare the global options, and define them:

\RequirePackage{xkeyval}
\DeclareOptionX{voc}{\def\al@mode{voc}}
\DeclareOptionX{fullvoc}{\def\al@mode{fullvoc}}
\DeclareOptionX{novoc}{\def\al@mode{novoc}}
\DeclareOptionX{trans}{\def\al@mode{trans}}
\define@boolkey{arabluatex.sty}@pkg@{export}[true]{%
  \if@pkg@export%
    \AtBeginDocument{\luadirect{arabluatex.openstream()}%\AtEndDocument{\luadirect{arabluatex.closestream()}}%
  \else\fi}
\ExecuteOptionsX{voc}
\ProcessOptionsX\relax
\def\al@mode@voc{voc}
\def\al@mode@fullvoc{fullvoc}
\def\al@mode@novoc{novoc}
\def\al@mode@trans{trans}

Packages that are required by \texttt{arabluatex}:

\RequirePackage{xcolor}
\RequirePackage{luacolor}
\RequirePackage{etoolbox}
\RequirePackage{arabluatex-patch}
\RequirePackage{fontspec}
\RequirePackage{luacode}
\RequirePackage{xparse}
\RequirePackage{adjustbox}
\RequirePackage{xstring}
\RequirePackage{lua-ul}

The following boolean will be set to \texttt{true} in \texttt{RL} mode:

providebool{al@rlmode}

Here begins the real work: load \texttt{arabluatex.lua}:

\luadirect{dofile(kpse.find_file("arabluatex.lua"))}

Font setup. If no Arabic font is selected, issue a warning message and attempt to load the Amiri font which is included in \TeXlive:

\AtBeginDocument{\ifdefined\arabicfont\relax{else
  \PackageInfo{arabluatex}{\string\arabicfont\ is not defined.\MessageBreak
  arabluatex will try to load Amiri}%
  \newfontfamily\arabicfont{Amiri}[Script=Arabic]\fi}

This neutralizes what may be defined by other packages:

\AtBeginDocument{\def\setRL{\booltrue{al@rlmode}}\pardir TRT%\texdir TRT}
\setLR  The same applies to \setLR:
\begin{verbatim}
39 \AtBeginDocument{\def\setLR{\boolfalse{al@rlmode} \pardir TLT\textdir TLT}}
\end{verbatim}

\LR  This command typesets its argument from left to right. As \LR may be already defined, we need to redefine for it to suit our purpose:
\begin{verbatim}
41 \AtBeginDocument{\ifdef{\LR}{\RenewDocumentCommand{\LR}{m}{\bgroup\textdir TLT\rmfamily#1\egroup}}{\NewDocumentCommand{\LR}{m}{\bgroup\textdir TLT\rmfamily#1\egroup}}}
\end{verbatim}

\RL  This one typesets its argument from right to left. Same remark as above regarding the need of redefinition.
\begin{verbatim}
44 \AtBeginDocument{\ifdef{\RL}{\RenewDocumentCommand{\RL}{m}{\bgroup\textdir TRT\rmfamily#1\egroup}}{\NewDocumentCommand{\RL}{m}{\bgroup\textdir TRT\rmfamily#1\egroup}}}
\end{verbatim}

\MkArbBreak  The \MkArbBreak{⟨csv list of commands⟩} command can be used to give any command—either new or already existing—the precedence over arabluatex inside Arabic environments. It is actually coded in Lua.
\begin{verbatim}
47 \NewDocumentCommand{\MkArbBreak}{s m}{\IfBooleanTF{#1}{\luadirect{arabluatex.mkarbbreak(\luastringN{#2}, "out")}}{\luadirect{arabluatex.mkarbbreak(\luastringN{#2}, "dflt")}}}
\end{verbatim}

\aemph  Arabic emphasis. Needs to be redefined as well. The function is actually coded in Lua.
\begin{verbatim}
\aemph  The ‘starred’ version of this command always puts the stroke over its argument. As of v1.19, arabluatex uses lua-ul to render the strokes, thus allowing line breaks and manual hyphenation for transliterated Arabic.
\end{verbatim}

\aoline  \aoline and \auline derive from \newunderlinetype provided by the lua-ul package whereas \aoline*, which uses \overline in math-mode, is better suited for so-called ʾabǧad numbers.
\begin{verbatim}
52 \newunderlinetype@\overLine{\leaders\vrule height 3ex depth -2.9ex}
53 \def\aoline{\ifstar\aoline\@@aoline}
54 \def\@aoline#1{\ensuremath{\overline{\mbox{#1}}}}
55 \def\@@aoline#1{{\@aoverLine#1}}
56 \newunderlinetype@\underLine{\leaders\vrule height -.65ex depth .75ex}
57 \def\auline#1{{\@aunderLine#1}}
58 \AtBeginDocument{\ifdef{\aemph}{\RenewDocumentCommand{\aemph}{s m}{\IfBooleanTF{#1}{\luadirect{tex.sprint(arabluatex.aemph(\luastringN{#2}, "over"))}}}{\luadirect{\text.print(arabluatex.aemph(\luastringN{#2}, "over"))}}}}}
\end{verbatim}

61
\arbcolor \arbcolor[(color)]{(Arabic text)} takes the Arabic text to be colored as argument.

\SetInputScheme \arbcolorstar takes the Arabic text to be colored as argument.

\SetArbEasy By default, \arbcolorstar applies complex rules to generate euphonic tašdīd, ʿalif mamdūdah and sukūn depending on the modes which are selected, either voc, fullvoc or trans. Such refinements can be discarded with \SetArbEasy, either globally in the preamble or at any point of the document. Note that \SetArbEasy keeps the sukūn that is generated, while the starred version \SetArbEasystar takes it away. Default complex rules can be set back at any point of the document with \SetArbDflt.

\SetArbDfltstar As of v1.6, \arbcolorstar does not applies any more the assimilation rules that are laid on item (b) on page 18; a new starred version \SetArbDfltstar is now available to the user should he wish to apply them.

\SetTranslitFont By default, the font that is used for transliterated text is the main font of the document. Any other font may also be selected with the font-selecting commands of the fontspec package.

\SetTranslitStyle By default any transliterated Arabic text is printed in italics. This can be changed either globally in the preamble or at any point of the document:
\SetTranslitConvention \SetTranslitConvention\{\textit{convention}\}\} can be used to change the transliteration convention, which is \texttt{dmg} by default:\n\begin{verbatim}
\def\al@trans@convention{dmg}
\end{verbatim}

\def\al@trans@convention{#1}

By default, \texttt{\arbup} is set to \texttt{\textsuperscript{}}. This is how the \textit{tanwīn} that takes place at the end of a word should be displayed in \texttt{dmg} mode. \texttt{\NoArbUp} may be used either in the preamble or at any point of the document in case one wishes to have the \textit{tanwīn} on the line. The default rule can be set back with \texttt{\ArbUpDflt} at any point of the document. Finally \texttt{\SetArbUp} can be used to customize the way \textit{tanwīn} is displayed: this command takes the formatting directives as argument, like so: \texttt{\SetArbUp\{\textit{code}\}}.\n
\NewDocumentCommand{\al@arbup@dflt}{m}{\textsuperscript{#1}}\%
\NewDocumentCommand{\al@arbup}{m}{\al@arbup@dflt{#1}}\%
\NewDocumentCommand{\arbup}{m}{\al@arbup{#1}}\%
\NewDocumentCommand{\ArbUpDflt}{}{\let\al@arbup=\al@arbup@dflt}\%
\NewDocumentCommand{\NoArbUp}{}{\RenewDocumentCommand{\al@arbup}{m}{##1}}\%
\NewDocumentCommand{\SetArbUp}{m}{{\RenewDocumentCommand{\al@arbup}{m}{#1}}}\%
\Prname  Proper Arabic names or book titles should be passed to the \texttt{\prname} command so that they have their first letters uppercased. \texttt{\prname} is actually coded in Lua.\n\NewDocumentCommand{\prname}{s m}{\bgroup\SetTranslitStyle{\relax}\IfBooleanTF{#1}{\txtrans{#2}}{\arb[trans]{\uc{#2}}}\egroup}\%
\NewDocumentCommand{\prname*}{s m}{\bgroup\SetTranslitStyle{\relax}\IfBooleanTF{#1}{\txtrans{#2}}{\arb[trans]{\uc{#2}}}\egroup}\%
\NewDocumentCommand{\txtarb}{s m}{\bgroup\SetTranslitStyle{\relax}\IfBooleanTF{#1}{\txtrans{#2}}{\arb[trans]{\uc{#2}}}\egroup}\%

\SetTranslitStyle{\relax}\IfBooleanTF{#1}{\txtrans{#2}}{\arb[trans]{\uc{#2}}}\egroup}\%
\NewDocumentCommand{\prname}{s m}{\bgroup\SetTranslitStyle{\relax}\IfBooleanTF{#1}{\txtrans{#2}}{\arb[trans]{\uc{#2}}}\egroup}\%
\NewDocumentCommand{\prname*}{s m}{\bgroup\SetTranslitStyle{\relax}\IfBooleanTF{#1}{\txtrans{#2}}{\arb[trans]{\uc{#2}}}\egroup}\%
\NewDocumentCommand{\txtarb}{s m}{\bgroup\SetTranslitStyle{\relax}\IfBooleanTF{#1}{\txtrans{#2}}{\arb[trans]{\uc{#2}}}\egroup}\%

\texttt{\txtarb} sets the direction to right-to-left and selects the Arabic font. It is used internally by several Lua functions, but available to the user should he wish to insert utf8 Arabic text in his document.
\texttt{txtrans} \texttt{txtrans} is used internally by several Lua functions to insert transliterated Arabic text. Therefore, it is not documented.

\texttt{txarab} The \texttt{txarab} environment does for paragraphs the same as \texttt{txarb} does for short insertions of utf8 Arabic text.

\texttt{txarabtr} \texttt{txarabtr} environment is used internally by several Lua functions to insert running paragraphs of transliterated Arabic text Therefore, it is not documented.

\texttt{\textbackslash arb} The \texttt{\textbackslash arb} command detects which Arabic mode is to be used, either globally if no option is set, or locally, then passes its argument to the appropriate Lua function.
\arbmark \arbmark[(r|l)r] \{\texttt{\langle shorthand\rangle}\} takes one argument from a list of defined elements. The mark to be inserted is determined by contextual analysis or by an optional argument, either rl or lr. This command is coded in Lua.

\NewDocumentCommand{\arbmark}{O{} m}{\bgroup\SetInputScheme{arabtex}\luadirect{tex.sprint(arabluatex.processarbmarks(\luastringN{#2}, \luastringN{#1}))}\egroup}

\newarbmark \newarbmark lets the user define additional Arabic marks. As \arbmark, this command is coded in Lua. It takes three arguments: the abbreviated form to be used as argument of \arbmark, the rendition in Arabic script and the rendition in romanized Arabic.

\NewDocumentCommand{\newarbmark}{m m m}{\luadirect{arabluatex.newarbmark(\luastringN{#1}, \luastringN{#2}, \luastringN{#3})}}

\arab The \arab environment does for paragraphs the same as \arb does for short insertions of Arabic text.

\NewDocumentEnvironment{arab}{!O{\al@mode} +b}{\par\edef\@tempa{#1}{\ifx\@tempa\al@mode@voc\booltrue{al@rlmode}\bgroup\pardir TRT\textdir TRT\arabicfont\luadirect{tex.sprint(arabluatex.processvoc(\luastringN{#2}, \luastringO{\al@arb@rules}, \luastringO{\al@input@scheme}))}\egroup\else\ifx\@tempa\al@mode@fullvoc\booltrue{al@rlmode}\bgroup\pardir TRT\textdir TRT\arabicfont\luadirect{tex.sprint(arabluatex.processfullvoc(\luastringN{#2}, \luastringO{\al@arb@rules}, \luastringO{\al@input@scheme}))}\egroup\else\ifx\@tempa\al@mode@novoc\booltrue{al@rlmode}\bgroup\pardir TRT\textdir TRT\arabicfont\luadirect{tex.sprint(arabluatex.processnovoc(\luastringN{#2}, \luastringO{\al@arb@rules}, \luastringO{\al@input@scheme}))}\egroup\else\ifx\@tempa\al@mode@trans\bgroup\pardir TLT\textdir TLT\al@trans@font\al@trans@style\luadirect{tex.sprint(arabluatex.processtrans(\luastringN{#2}, \luastringO{\al@trans@convention}, \luastringO{\al@arb@rules}, \luastringO{\al@input@scheme}))}\egroup\else\fi\fi\fi\fi\fi\fi\fi\fi\fi}
The \texttt{arabverse} environment may receive different options: \texttt{mode}, \texttt{width}, \texttt{gutter}, \texttt{metre}, \texttt{color}, \texttt{utf}, \texttt{delim} and \texttt{export}; all of them are defined here just before the \texttt{arabverse} environment.

\begin{verbatim}
\newlength{\al@bayt@width}
\newlength{\al@gutter@width}
\setlength{\al@bayt@width}{.3\textwidth}
\setlength{\al@gutter@width}{.15\al@bayt@width}
\define@key[al]{verse}{width}{\setlength{\al@bayt@width}{#1}}
\define@key[al]{verse}{gutter}{\setlength{\al@gutter@width}{#1}}
\define@key[al]{verse}{metre}{\arb{#1}}
\define@key[al]{verse}{color}{\color{#1}}
\define@key[al]{verse}{utf}{true}
\define@key[al]{verse}{delim}{true}
\define@key[al]{verse}{export}{true}
\define@key[al]{verse}{mode}{fullvoc, voc, novoc, trans}{\al@mode{#1}}
\presetkeys[al]{verse}{metre={}, utf=false, delim=false}
\end{verbatim}

Then follows the environment itself:

\begin{verbatim}
\NewDocumentEnvironment{arabverse}{!O{}}{
\bgroup\setkeys[al]{verse}{width, gutter, color, utf, delim, metre}{#1}\
\if@pkg@export\ifal@verse@export\
\ArbOutFile{\begin{arabverse}}\
% \ifx\al@mode\al@mode@trans\
% \luadirect{ arabluatex.tooutfile(\luastringN{[#1]})}\
% \else\
\IfSubStr[1]{#1}{utf}\
% \luadirect{arabluatex.tooutfile(\luastringN{[#1, utf]})}\
% \fi\
\else\fi\egroup\
\par\centering\noindent\bgroup\setkeys[al]{verse}{metre={}, utf=false, delim=false}\
\end{arabverse}}\
\hfill\setkeys[al]{verse}{width, gutter, color, utf, delim, mode, export}{#1}\
\end{verbatim}

66
Each verse consists of two hemistichs; therefore the \texttt{bayt} command takes two arguments, the first receives the \textit{sadr} and the second the \textit{ʿaǰuz}. That two subsequent hemistichs should be connected with one another is technically named \textit{tadwīr}. In some of these cases, the hemistichs may be connected by a prominent horizontal flexible stroke which is drawn by the \texttt{alverse@stroke} command.

\texttt{SetHemistichDelim} A hemistich delimiter also may be defined. By default, it is set to the `star' character: *#. The \texttt{SetHemistichDelim{⟨delimiter⟩}} command can be used at any point of the document to change this default setting.
\arind \arind\{\langle root\rangle\} is a command specialized in the construction of indexes. As a mandatory argument, it takes the Arabic root under which a given word is to be indexed. Additionally, it may receive three optional ‘named’ arguments: index, root and form.

\NewDocumentCommand{\SetDefaultIndex}{m}{\edef\@tempa{#1}\ifx\@tempa\empty\def\al@default@index{\jobname}\else\def\al@default@index{#1}\fi}
\def\al@index@mode{\al@mode}
\NewDocumentCommand{\SetIndexMode}{m}{\def\al@index@mode{#1}}
\define@cmdkeys[al]{index}{alind@}{index,root,form}
\NewDocumentCommand{\arind}{o m}{\IfNoValueTF{#1}{\ifdefined\al@default@index%\csname index\endcsname[\al@default@index]{#2}%\else%\csname index\endcsname{#2}%\fi}{\bgroup\setkeys[al]{index}{#1}\def\al@one{%\ifdefined\alind@root!LR{\alind@root}\else!LR{1}\fi}\def\al@two{%\ifdefined\alind@form @arb[\al@index@mode]{\alind@form}\else\fi}\ifdefined\alind@index%\csname index\endcsname[\alind@index]{#2\al@one\al@two}%\else%\ifdefined\al@default@index%\csname index\endcsname[\al@default@index]{#2\al@one\al@two}%\else%\csname index\endcsname[\al@index@mode]{#2\al@one\al@two}%\else%\fi%\fi%\egroup}}

\abjad \abjad\{\langle number\rangle\} expresses its argument in Arabic letters in accordance with the ʾabǧad arrangement of the alphabet. \langle number\rangle must be between 1 and 1999. It is now coded in Lua so that polyglossia is no longer needed. See arabluatex.lua for more information.
\RenewDocumentCommand{\abjad}{m}{%
  \ifbool{al@rlmode}%
    \aoline*{
      \luadirect{tex.sprint(arabluatex.abjadify(\luastring{#1}))}}
  {\luadirect{tex.sprint(arabluatex.abjadify(\luastring{#1}))}}}
\else%
\NewDocumentCommand{\abjad}{m}{%
  \ifbool{al@rlmode}%
    \aoline*{
      \luadirect{tex.sprint(arabluatex.abjadify(\luastring{#1}))}}
  {\luadirect{tex.sprint(arabluatex.abjadify(\luastring{#1}))}}}
\fi}
\ayah
\ayah{\textit{(number)}} prints up to 3-digit numbers inside ‘end of Ayah’ sign (U+06DD) or inside parentheses depending on the mode which is selected.
\arahnull
The \arahnull command does nothing by itself. It is processed only if it is found in Arabic context so as to put back on contextual analysis in case it has been broken by other commands.
\arbraces
\arbraces{\textit{(Arabic text)}} puts its argument between braces. This macro is written in Lua and is dependent on the current value of \textit{tex.textdir}.
\LRmarginpar
\LRmarginpar is supposed to be inserted in an Arabic environment. It typsets his argument in a marginal note from left to right.
\LRfootnote
\LRfootnote and \RLfootnote are supposed to be used in Arabic environments for insertions of non Arabic text. \LRfootnote typsets its argument left-to-right...
\RLfootnote
while \RLfootnote typsets its argument left-to-right.
\FixArbFtnmk
In the preamble, just below \usepackage{arabluatex}, \FixArbFtnmk may be of some help in case the footnote numbers at the bottom of the page are printed in the wrong direction. This quick fix uses and loads \textit{scrextend} if it is not already loaded.
\NewDocumentCommand{\FixArbFtnmk}{}{\%
Exporting Unicode Arabic to external file

\SetArbOutSuffix

By default, \_out is the suffix to be appended to the external file in which arabluatex exports Unicode in place of arabtex or buckwalter strings. Any other suffix may be set with \SetArbOutSuffix{(suffix)}.

\ArbOutFile

\ArbOutFile*[\langle newline\rangle]{\langle string\rangle} silently exports \langle string\rangle to the external selected file. It may take \langle newline\rangle as an optional argument in which case a carriage return is appended to string.

\ArbOutFile*

\ArbOutFile*[\langle newline\rangle]{\langle string\rangle} does the same as \ArbOutFile but also inserts \langle string\rangle in the current .tex source file.

arabexport

The arabexport environment processes and prints its argument unchanged to the current .pdf file. Additionally, if arabluatex is loaded with the export option, this argument is exported to the external selected .tex file with Unicode in place of the original arabtex or buckwalter strings.

arab@v@export

The arab@v@export environment does for arabverse the same as arabexport. It is used internally by arabverse.
\par
#2
\luadirect{arabluatex.doexport("arabverse")}
\luadirect{tex.sprint(arabluatex.arbtoutf(\luastringN{#2}))}
\luadirect{arabluatex.doexport("no")}
\else\par\#2\fi\else\par\#2\fi
}\par
\arbpardir  \arbpardir is automatically inserted by arabluatex at the beginning of Arabic paragraphs converted to Unicode so that they are printed in the right direction.
\NewDocumentCommand{\arbpardir}{}{\ifx\al@mode\al@mode@trans\setLR\else\setRL\fi}

Errors and Warnings
\newcommand{\al@warning}[1]{\PackageWarning{arabluatex}{#1}}
\newcommand{\al@error}[2]{\PackageError{arabluatex}{#1}{#2}}
\newcommand{\al@wrong@nesting}{\al@error{(RL/LR)\string\footnote\space is not allowed in \string\RL{} commands}{Get rid of the surrounding \string\RL{} command.}}
\newcommand{\al@wrong@mark}{\al@warning{Unknown Arabic mark in \string\arbmark{}. Replaced with \MessageBreak<??>. Please check your code}}

That is it. Say goodbye before leaving.

Patches
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{arabluatex-patch}[2016/11/14 v1.0 patches for arabluatex]

I have put in a separate .sty file external lines of code that I had to patch for a good reason. I hate doing this, and hopefully, most of these lines will disappear as soon as they are not required anymore.

The following is taken from latex.ltx. I had to make this patch for I could not find a way to process the list environments in right-to-left mode. The LuaTEX primitives \bodydir and \pagedir will eventually allow us to get rid of this:
\def\list#1#2\%
\ifnum \@listdepth >5\relax
\@toodeep
\else
\global\advance\@listdepth\@ne
\csname @list\romannumeral\the\@listdepth\endcsname
\def@itemlabel{#1}
\let\makelabel@mklab
\else
\global\advance\@listdepth\@ne
\csname @list\romannumeral\the\@listdepth\endcsname
\def@itemlabel{#1}
\let\makelabel@mklab
\fi
\rightmargin\z@\listparindent\z@\itemindent\z@\csname @list\romannumeral\the\@listdepth\endcsname\def@itemlabel{#1}\let\makelabel@mklab

71
\@mb@listfalse

#2\relax

\@trivlist

\parskip\parsep

\parindent\listparindent

\advance\linewidth -\rightmargin

\advance\linewidth -\leftmargin

patch begins:
\ifbool{al@rlmode}{\advance@t@l@leftmargin rightmargin\%}{\advance@t@l@leftmargin leftmargin\%}
\parshape \@ne \@totalleftmargin \linewidth
\ignorespaces}
\def\@item[#1]{%
\if@noparitem
\@donoparitem
\else
\if@inlabel
\indent \par
\fi
\ifhmode
\unskip\unskip \par
\fi
\if@newlist
\if@nobreak
\@nbitem
\else
\addpenalty@beginpenalty
\addvspace@topsep
\addvspace{-\parskip}\%
\fi
\else
\addpenalty\@itempenalty
\addvspace\itemsep
\fi
\global\@inlabeltrue
\fi
\everypar{%
\@minipagetrue
\global\@newlistfalse
\if@inlabel
\global\@inlabelfalse
{\setbox\z@lastbox
\ifvoid\z@
\kern-\itemindent
\kern-\itemindent
\fi
\box\labels
\penalty\z@
\fi
\fi

patch ends.
\parshape \@ne \@totalleftmargin \linewidth
\ignorespaces}
This is adapted from Vafa Khalighi’s bidi package. Thanks to him.

References


**Change History**

v1.0.

General: Initial release ............... 1

v1.0.1.

General: Minor update of the documentation ............... 1

v1.1.

\abjad: New and more flexible \abjad command. ............... 68

v1.2.

\SetArbEasy: New \SetArbEasy/\SetArbDflt for ‘modern’ or ‘classic’ Arabic styles. ............... 62

v1.3.

\arbup: ʾiʿrāb is now written as superscript text in dmg mode by default. ............... 63

v1.4.

\SetInputScheme:
  \SetInputScheme can be used to process other input schemes such as ‘Buckwalter’ ............... 62
  \SetTranslitFont: For selecting a specific font for transliterated texts ............... 62

v1.4.3.

\abraces: New \abraces command which expresses its argument between braces. ............... 69

v1.4.4.

\SetArbEasy*: this starred version discards the sukūn in addition to what is already discarded by \SetArbEasy. ............... 62

v1.5.

General: Compatibility with the quran package ............... 56

Environments may be nested inside the arab environment ... 52

txarab: New txarab environment for typesetting running paragraphs in Unicode Arabic ... 64

v1.6.

arabverse: New environment arabverse for typesetting Arabic poetry ............... 66

\bayt: New macro \bayt for typesetting each verse inside the arabverse environment ............... 67

\SetArbDflt*: This starred version applies the assimilation rules in addition to what \SetArbDflt already does. ............... 62

\SetHemistichDelim: New \SetHemistichDelim command for changing the default delimiter between hemistichs ............... 67
v1.7.
\arbnull: New \arbnull command for putting back on any contextual analysis rule broken by other commands. .......... 69

v1.8.
General: arabica transliteration standard is now supported .... 41
v1.8.5.
General: Six additional Persian characters are now available ... 11
v1.9.
\MkArbBreak: New \MkArbBreak command for inserting user-defined macros in Arabic environments .......... 61

v1.9.2.
\aemph*: Starred version which always puts the stroke over its argument .......... 61

v1.10.
General: \uc supersedes \cap .... 42
\prname: New command for typesetting Arabic proper names in transliteration ............ 63

v1.11.
\arbmark: New command for inserting additional marks in Arabic environments .... 65
\newarbmark: Allows defining additional sets of Arabic marks 65

v1.12.
General: \abjad can now process \LaTeX counters .... 27
\arbcolor: Standard color command for Arabic environments .... 62
\MkArbBreak*: ‘starred’ version which closes Arabic environments before processing declared commands. ............ 61

v1.13.
arabexport: Processes and print its argument in the current file and exports it in full Unicode in the external selected .tex file. ... 70
arabverse: New options color and export to arabverse environment. ............... 66
\arbmark: New optional argument: either rl or lr ............ 65
\ArbOutFile: Silently exports its argument in the selected external file. ............... 70
\arbpardir: Sets the direction of Arabic paragraphs once they are converted to Unicode. ............ 71
\prname*: Renders proper names already converted to Unicode in upright roman style ............ 63
\SetArbOutSuffix: Sets a suffix to be appended to the filename of the external Unicode file. .... 70

v1.15.\ayah: Prints End of Ayah sign .... 69
v1.16.\aemph: Now uses ulem .... 61
v1.18.\aemph: Now uses lua-ul .... 61
v1.19.\auline: Non context-sensitive command to underline Arabic words is provided .......... 61

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

Symbols \@M .............. 435 \@aoverLine .... 52, 55
\@@aoline .... 53, 55 \@aoline .... 53, 54 \@aunderLine .... 56, 57


arind .......... 261
AtEndDocument .. 12
auline .......... 28, 52
ayah .......... 35, 312

b babel (package) ..... 55
babel-french (package) . 55
bayt .......... 29, 229
begin .......... 200
bidi (package) ....... 73
boolfalse ...... 37, 111, 121, 127, 133, 157, 163, 169
box .......... 430, 455
Brockelmann, Carl ... 11, 41, 43

c centering ....... 209
cube penalty ...... 435, 437
color ............. 188
color (option) ..... 29
color@endgroup ... 462
color@setgroup ... 462
cquotes (package) ... 55
c
DeclareOptionX ... 4, 5, 6, 7
deffootnote . 328, 331
def@boolkey ... 8, 189, 190, 191
def@choicekey 192
def@cmdkeys ... 273
def@key ......... 185, 186, 187, 188
delim (option) ... 29
d
empty ........... 263
dendarab@v@export 220
denumitem (package) 53–55
environments:
arab .......... 8, 154
arab@v@export ... 349
arabexport .... 57, 340
arabverse ...... 28, 181
txarab ....... 48, 109
txarabtr ...... 113
everypar ....... 421, 438
ExecuteOptionsX .... 14
export (option) . 6, 29, 57

F
FixArbFtnmk .... 50, 325
fontspec (package) 5, 41, 62
footnote (package) ...... 31
footnotehyper (package) 31
fullvoc (option) .... 6, 9

G
global ... 378, 419, 423, 425, 448, 449
Grammar of the Arabic
Language, A ... 6–8, 10, 14, 16–18, 20, 26, 27, 43, 54
gutter (option) ... 29

H
Habash, Nizar Y .... 46
\box .......... 233, 448, 449, 457, 461
\Height ... 238, 254, 258
\hspace ... 242, 248

I
if@inlabel ... 401, 424
if@newlist ... 407
if@nmbrlist ... 442
if@nobreak ... 408, 433
if@noitemmargin ... 440
if@noparitem ... 398
if@pkg@export 9, 199, 226, 335, 341, 352
if@verse@delim ... 240
if@verse@export ... 199, 226, 352

J
jobname .......... 264

K
KOMA-script (package) 50
Krüger, Marcel ... 35

L
labelsep ... 453, 459
labelwidth 452, 454, 457
Lagally, Klaus .... 4, 22
Lane, Edward William . 21
lastbox ...... 426
leaders ....... 52, 56
leavevmode ... 105, 120, 126, 132, 233
leftmargin .... 392, 394