

# Arabic and Bidirectional Challenges for Translation and Software Development





## Table of Contents

|  |           |
|--|-----------|
| <b>Introduction</b>  | <b>4</b>  |
| <b>Standard Arabic vs. Arabic Dialects</b>                   | <b>4</b>  |
| <b>Linguistic Differences between English and Arabic</b>     | <b>5</b>  |
| <b>Software Localization Issues</b>                          | <b>7</b>  |
| <b>Text Layout Issues</b>                                    | <b>8</b>  |
| <b>Software Localization Engineering Concerns</b>            | <b>9</b>  |
| <b>Selecting a tool that supports the language/features:</b> | <b>9</b>  |
| <b>Testing software</b>                                      | <b>9</b>  |
| <b>Hints and Tips</b>  | <b>9</b>  |
| <b>Arabic Language Features</b>                              | <b>10</b> |
| <b>Layout</b>  | <b>10</b> |
| <b>Contextual Analysis (Shaping)</b>                         | <b>11</b> |
| <b>Ligatures</b>   | <b>12</b> |
| <b>Text justification</b>                                    | <b>13</b> |
| <b>Other Design Issues</b>                                   | <b>14</b> |
| <b>Right-to-left orientation for GUI</b>                     | <b>14</b> |
| <b>Right-to-left editing</b>                                 | <b>15</b> |
| <b>Dates</b>   | <b>15</b> |
| <b>Summary</b>   | <b>17</b> |
| <b>Glossary</b>  | <b>17</b> |
| <b>Further Reading</b>                                       | <b>20</b> |

### Introduction

When software products are developed for distribution in Arabic speaking markets, technical issues affecting software development and the subsequent localization process must be considered. It is the goal of this White Paper to identify some of the more common issues associated with Arabic localization so as to better prepare ourselves to work around these issues and produce a better product.

One of the biggest concerns with Arabic localization is text direction. Arabic and Hebrew are considered "bi-directional" languages. Bi-directional languages, also referred to as "BiDi", refer to text input and output that uses a mix of Latin (usually English or French) text along with the target language. Arabic and Hebrew are the primary BiDi languages, but they also include Farsi, Urdu, and other languages using right-to-left (RTL) script.

As big of a challenge as bi-directional text poses to the Western developer, other issues not previously encountered in localization loom large when internationalizing and localizing for Arabic. In fact, when we as localization professionals with over 30 years of experience began examining a few of these issues: contextual analysis, rendering and shaping, alternate numeric display, Hijri dates, character extenders for justified text, "logical vs. output" order, neutral characters, symmetric swapping; we realized that we needed to create and study an entirely new glossary with definitions for some of these terms before we could even get around to resolving said issues. Companies that do not deal with the numerous challenges that language poses in a development environment on a regular basis face a daunting task in reaching Arabic-speaking markets.

Without any uncertainty, Arabic localization challenges require careful thought and resource planning when translating content or applications to this language for the first time. Fortunately, the features included in BiDi Windows make development of a localized version of a standard Windows application relatively simple. In addition, simple recognition of Arabic conventions at the development stage can prevent many of these issues to become overwhelming challenges in subsequent localization phases.

### Standard Arabic vs. Arabic Dialects

There are many dialects of Arabic, varying widely from country to country and even on a regional basis. Nearly all business documents, news, and software documentation are translated into Modern Standard Arabic. The following definition is taken from [http://en.wikipedia.org/wiki/Modern\\_Standard\\_Arabic](http://en.wikipedia.org/wiki/Modern_Standard_Arabic):

*Modern Standard Arabic is derived from Classical Arabic or Koranic Arabic, the language of the Qur'an. It has been said that*

*even university-educated Arabs have difficulty conversing in the language for very long. Upon moving from one Arabic-speaking country to another, an Arab will prefer to learn the local dialect than to use Modern Standard Arabic as he is afraid to sound pedantic if he does.*

*Modern Standard Arabic is used for nearly all writing with the exception of some modern poetry, stage plays, comic books and children's books. Local spoken dialects are usually used in these contexts. Also, Egyptian Arabic, the spoken language of Egypt, has gained in stature in the last 100 years, and texts of many sorts are written in it. Even so, the contemporary Arabic writing is predominantly in Modern Standard Arabic.*

*Modern Standard Arabic shares most of its vocabulary, syntax and morphology with its parent language, Classical Arabic, but there are noticeable differences; these differences number fewer than one might expect, given the 1,500-year separation between the two. Modern Standard Arabic has always been an artificially regulated language, and has not evolved as a naturally-spoken language might.*

Because Modern Standard Arabic is not acquired as a native language, (it is taught as a second language), the number of speakers of the language is difficult to determine. The geographical center of this language covers the northernmost part of Africa from Mauritania to Egypt, the Levant, the Arabian Peninsula, and Iraq. It is estimated that some 165,000,000 people throughout the Islamic world have some knowledge of Standard Arabic.

### **Linguistic Differences between English and Arabic**

Because demand for widespread translation from English to Arabic is relatively recent, the common glossary or vocabulary of Arabic may be perceived as somewhat deficient in native scientific or technical terms. Many technical terms are created in Arabic translation through transliteration or "coining a phrase." This phenomenon makes it challenging to find standards for technical terms commonly used in English. Over time, as more and more technical content is translated into (and eventually authored in) Arabic, this challenge will decrease.

In some cases, target terms in Arabic are ambiguous. For instance, Arabic makes no distinction between "administration" and "management." Translation involving computer terminology is a challenge. For instance, "calculate" and "calculator" are ambiguous as well as "compute" and "computer." While it is not easy to express computing terms in Arabic, the linguist can create custom Arabic terms that can accurately express the exact

meanings of the source language terms. If necessary, a glossary for the reader can be created that explains the real meanings of the source terms.

Basic linguistic differences between English and Arabic can make translation challenging in technical and scientific work. Basically, Arabic is a language that has developed primarily through literature, religious texts, and poetry. Very little modern scientific or technical writing has originated in Arabic, creating a shortage of equivalent terminology. The following table highlights some of the inherent differences in language structure that the linguist must overcome when translating scientific or technical content from English into Arabic.

| English  | Arabic  |
|--|---|
| Only a few grammatical items are compound structures   | Majority of grammatical items are compound structures   |
| Rigid word order   | Flexible word order   |
| Very few inflections   | Highly inflectional   |
| Uses abbreviations, acronyms, formulae, and idioms   | Rarely uses abbreviations, acronyms, formulae and idioms  |
| Narrow range of gender distinction   | Wide range of gender distinction  |
| Tense-aspect is very clear-cut and distinct  | There is no clear-cut tense aspect distinction  |
| No dative nor dual   | Contains <i>dative</i> (indicates indirect object of a verb) and <i>dual</i> (of, relating to, or being a number category that indicates two persons or things)   |
| Scientific and technical terminology covers all relevant fields  | Shortage of scientific and technical terminology that may cover all fields  |
| Archaic expressions are nearly obsolete  | Archaic expressions are still in use  |
| Many compound lexical structures are used  | Compound lexical structures are rare  |
| Metaphor and other forms of figurative language are reserved for poetic use, literature and related fields | Metaphor and other forms of figurative language are frequently used, even in Modern Standard Arabic   |
| Adverbs are usually formed by adding (ly) to adjectives  | Adverbs are formed by prepositional pre-modification of nouns and adjectives; English prepositions such as <i>before</i> , <i>after</i> , <i>above</i> , <i>over</i> , <i>below</i> , <i>under</i> , <i>behind</i> and <i>between</i> are adverbs in Arabic |
| Capitalization is occasionally used for semantic implication (e.g. Mosaic, Nativity)                       | Dos not use any form of capitalization  |

| English  | Arabic  |
|--|---|
| Does not use vocalization<br>(Vocalization = to change a consonant into a vowel during articulation.)    | Vocalization has a semantic function                              |
| Punctuation affects the interpretation of text   | Punctuation has little, if any, bearing on interpretation of text |
| Apart from such suffixes as <i>-ling</i> and <i>-ette</i> there is no paradigmatic diminutive in English | Paradigmatic diminutive exists                                    |
| There are about 20 configurations of vowel sounds  | Few vowel sounds; used mainly in vocalization                     |

Due to the linguistic constraints listed above, it is common for qualified Arabic linguists to charge substantially higher rates for translating technical content vs. general documents and literature.

## Software Localization Issues

Software applications localized in Arabic and related languages have special requirements:

- **Local Data:** support all locale sensitive data, like date/time/number/currency format and calendar information. Users may require a choice of *Hijri* vs. *Gregorian* date and calendar display. For more information on Hijri, see “Dates” on page 15.
- **Break Iterator:** support for language/script specific cursor placement, word, line and sentence breaking. In Middle Eastern languages, more enhanced support is required compared to Latin languages.
- **Search and Replace:** there are special search options that are specific to BiDi languages (for example, Hamza, a sign in Arabic orthography used to represent the sound of a glottal stop, transliterated in English as an apostrophe.)
- **Collation:** enable support for sorting and indexing according to local conventions. Sorting is different in some Arabic script languages, (Arabic, Farsi and Urdu.)
- **Right to Left (RTL) User Interface:** the complete UI must be capable of RTL layout. In most cases this involves mirrored menus, messages and dialogues. Icons are often moved to the right margin while the scroll bars are always moved to the left margin of a menu.
- **Tables reversed:** because text in Arabic is read RTL, tables must be “flip flopped” or reversed. The first table column in English must move to the right edge of the table in Arabic, and so forth. See the example below:

## Arabic and Bidirectional Challenges

*Example of TABLE direction using DIR="LTR"*

|                        |          |          |          |          |          |          |          |          |          |          |
|------------------------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|
| <b>European Digits</b> | <b>0</b> | <b>1</b> | <b>2</b> | <b>3</b> | <b>4</b> | <b>5</b> | <b>6</b> | <b>7</b> | <b>8</b> | <b>9</b> |
| <b>Arabic Digits</b>   | ٠        | ١        | ٢        | ٣        | ٤        | ٥        | ٦        | ٧        | ٨        | ٩        |

*Example of TABLE direction using DIR="RTL" for Arabic*

|          |          |          |          |          |          |          |          |          |          |                        |
|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|------------------------|
| <b>9</b> | <b>8</b> | <b>7</b> | <b>6</b> | <b>5</b> | <b>4</b> | <b>3</b> | <b>2</b> | <b>1</b> | <b>0</b> | <b>European Digits</b> |
| ٩        | ٨        | ٧        | ٦        | ٥        | ٤        | ٣        | ٢        | ١        | ٠        | <b>Arabic Digits</b>   |

## Text Layout Issues

When text is entered in a BiDi software application (or software localized for Arabic), the user is able to invoke different language input for Latin vs. Arabic text. To avoid typing Latin text in reverse, Arabic-enabled software uses a concept called *push mode* wherein the cursor remains stationary and the Latin string is "pushed" to the left to make room for the next character. This behavior resembles how sequentially entered numbers will display on a calculator. When the users finished inputting Latin text (signaled by changing keyboard language), the cursor moves to the extreme left of the Latin string, waiting for an Arabic character.

All strings, paragraphs and documents have a code indicating language direction (e.g., RTL or LTR). The implied language direction is not always enough to produce understandable text, when Latin characters are mixed with Arabic.

Neutral characters (punctuation and character abbreviations, such as "&" and "%") can be problematic. A set of directional formatting control characters are required to control the ordering of neutral characters in displayed output.

The examples below show Incorrect Order (without use of markers) and Correct Order (with use of markers.)

| <b>Incorrect Order</b> | <b>Correct Order</b>  | <b>Comment</b>  |
|------------------------|-----------------------|---|
| <div>محمد(محسن)</div>  | <div>محمد(محسن)</div> | The closing bracket at the end of the sentence appears as RTL.              |
| <div>منال.</div>       | <div>منال.</div>      | The full-stop (period) at the end of the sentence appears at the beginning. |



Example of hidden Control Codes or Markers used to achieve output results shown above:

|     |                    |      |                              |
|-----|--------------------|------|------------------------------|
| RLM | Right-to-Left Mark | 200F | Acts as an Arabic character. |
| LRM | Left-to-Right Mark | 200E | Acts as a Latin character.   |

## **Software Localization Engineering Concerns**

- Have you ensured that all dialog box content and other strings are isolated from the functional software code (such as in “rc” files and string tables)?
- How are you going to handle updates to content (such as modified strings)?

### **Selecting a tool that supports the language/features:**

- Allow translators to see everything in a WYSIWYG view
- Allow automatic testing for various errors (duplicate hot keys, truncations, spelling checks, translation quality, etc.)
- Eliminate the need for various compilation procedures
- Have a tool that supports Flipping/Mirroring

## **Testing software**

Pay careful attention to the following areas:

- Hot keys (they change when the corresponding string is translated)
- Reading order
- Truncations – alignment bugs in dialog boxes
- Alignment for RTL text
- Image/table orientation (RTL)
- Do directional icons (Prev, Next) make sense in RTL context (e.g. arrow icons may need to be reversed)?

## **Hints and Tips**

- Watch for correct reading order of neutral characters (e.g. “.”) in pop up menus
- Avoid using Art/Images with “text art” that needs to be localized
- If text is included in art of images to be translated, be sure that the text is stored in a separate graphic layer

- Using SW development tools that support your BiDi language (e.g. Arabic) drastically reduces the number of bugs.

## Arabic Language Features

There are a number of obvious differences between English and Arabic, but the differences fall into five main categories:

1. Arabic **characters are laid out in RTL** (Right to Left) order, *with the exception of numbers*, which are laid out in LTR (Left to Right), like English. Text is right aligned on the page, and written from top to bottom.
2. Letters **change shape** depending on context. Each letter has up to four forms:
  - a. **Initial form**, which is the first letter in a word
  - b. **Final form**, when the letter is the last in a word
  - c. **Medial form**, when a letter is surrounded by other characters in a word
  - d. **Isolated form**, when a character stands by itself (one letter word)
3. Arabic can include **diacritics**. Such marks, placed above or below letters, typically represent vowel sounds or other modifiers.
  - a. Diacritics are primarily used in children's and religious text, but are supported by BiDi Windows. (For this reason, diacritics are not discussed further in this paper.)
  - b. Vertical kerning of diacritical marks is required for acceptable output.
4. Arabic makes extensive use of **ligatures**, (the process whereby two letters printed together are replaced by a single new character.) English employs such ligatures as "ff" and "fi".
5. **With Arabic, numbers may be represented by either** Hindi digits or Arabic digits, depending upon the target region or country for the Arabic content. Numbers are displayed LTR in all BiDi countries.

## Layout

"Layout" involves differences in how text is input (Arabic and Latin) and stored vs. how text is physically displayed. Display is mixed because Latin characters and numerals display LTR, while Arabic words display RTL. Arabic-enabled applications and/or systems always include a logical-to-physical transformation algorithm. All text input is in *logical* order—all output (e.g. screen display or hard copy) is in *physical* order.

The physical appearance of a bilingual string can differ enormously from how it is stored internally in an application (e.g. user input in software.) Non-contiguous cursor movement, text selection, and semantic re-ordering of

sections of text must be taken into consideration for BiDi-enabled applications.

The intention of Unicode is to be able to display mixed-language text without carrying data about the text. Unicode can solve many layout/text display challenges, but there are situations in BiDi in which text display can be less than desirable. A good example is the hyphen.

In Arabic text, how should the logical string **494-9571** be displayed?

- If this is a telephone number, the entire number should be displayed as shown.
- If it is a subtraction, it should be displayed **9571-494**. (e.g. RTL display for "494 minus 9571")

The computer cannot tell how to properly display this text without additional information from the user. To resolve such ambiguities, Unicode provides support for right-to-left marks and left-to-right marks. These non-printing characters tell the system how to interpret the direction of characters that follow.

### Contextual Analysis (Shaping)

The Arabic alphabet has thirty-six alphabetics (no upper case), ten numeric symbols, and a few special alphanumeric characters. An Arabic code page usually consists of all these symbols combined with the English alphabet.

All Arabic alphabetic characters can have **up to four display representations** depending on their relative position in a word: *initial*, *final*, *medial*, or *isolated*. Its left and right neighbors determine an Arabic character's shape. The algorithm determining which of the four shapes to use is called *contextual analysis*.

The Figure below shows the four forms of the Arabic characters 'ain,' 'ba,' 'qaf' and 'ha' respectively, indicated in **red**.

| Initial | Isolated | Medial | Final |
|---------|----------|--------|-------|
| عرب     | ع        | يعرب   | مع    |
| يرد     | ب        | يبني   | حلب   |
| قلم     | ق        | يقدم   | دمشق  |
| هالة    | ه        | يهيم   | نبه   |

If your localized software application requires a significant amount of user input, contextual analysis is an important design concern. Examples of such software would be word processors, text editors, or software that requires extensive comments, notes and annotations to be input by the user. In regular translation, the four character forms are also a significant challenge for proofing content.

As indicated, shaping is the process of selecting the appropriate glyphs to represent a set of codes. The Unicode term for this process is *rendering*, and it is also sometimes referred to as *glyphing*.

Another shaping issue is *symmetric swapping*. This is the process of **changing the direction of a neutral character** depending on language. The characters concerned are "bracket" characters: < and >, [ and ], ( and ), and { and }. In Latin, we write the logical string 3 < 4 as 3 < 4, while in RTL, it is displayed as 4 > 3 (RTL display for "3 is less than 4".) Because it is read RTL, the shape of the less than symbol ( < ) has changed.

### Ligatures

As in English, certain combinations of two (even three) characters form one shape. (In English "ff" and "fl" are common examples.) Ligature selection is dependent not only on the characters themselves but also on the selected Arabic font. Some fonts do not use ligatures at all and others may have as many as 200 different ligatures defined. Only 63 ligatures found in the Traditional Arabic font are supported by Arabic Windows.



### Other Design Issues

#### Right-to-left orientation for GUI

The user interface for BiDi applications is modified to have a “right-to-left feel.” Examples include text that is right aligned, scroll bars on the left, and so forth.

Dialog boxes should be laid out with design elements going RTL, and RTL controls should be used.

Edit controls support both RTL and LTR text entry and display, user selectable by using the `CTRL+left SHIFT` and `CTRL+right SHIFT` key combinations. Pressing these keys does three things:

- Makes the text align to the appropriate side of the edit control.
- Changes the reading order to match the text direction.
- Changes the keyboard language to match the selected direction.

Studying the GUI of Applets provided with Arabic or BiDi Windows gives a good idea of proper modifications needed to provide the “right-to-left feel.” In Windows Paintbrush, tools are switched to the right and their column order reversed. The vertical scroll bar is placed on the left and the horizontal scroll bar defaults to right aligned. Text is displayed to the left of the selected point, and mixed text is supported, along with use of the dual font dialog. The icon for text entry is changed to display Arabic letters instead of English ones. Internal code changes are also required in order to display text properly.

The figure below shows a BiDi Arabic user interface for Microsoft Workspace:



### Right-to-left editing

The start of text entry with Arabic text is at the top right-hand side of the page. Other modifications required for Arabic text editors include:

- Rulers should originate on the right-hand side and read from right to left.
- The first column of tables or columnar text is on the right, last on the left. Spreadsheet columns originate at the right-hand side of the page.
- Arabic paragraphs can be right justified or left justified, but still read from right to left.
- The first item of a menu bar should be on the right-hand-side, and the vertical scrollbar should be on the left-hand-side.

Not everything is switched. The Arabic model is for a right-to-left document, *not* a right-to-left desktop. Thus, locations of the system menu or minimize/maximize icons are not swapped.

### Dates

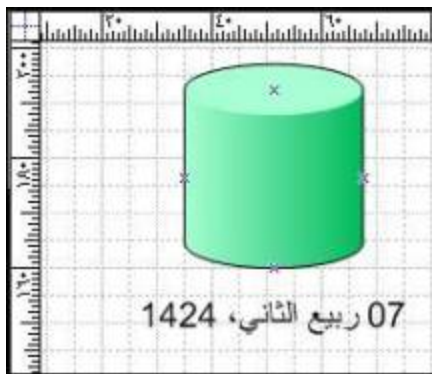
Two separate calendar systems, and the choice of two alphabets to display them, complicate choices for date display. Arabic countries use both Gregorian (Western) and *Hijri* (Islamic) dates. Hijri date is the official date in

## Arabic and Bidirectional Challenges

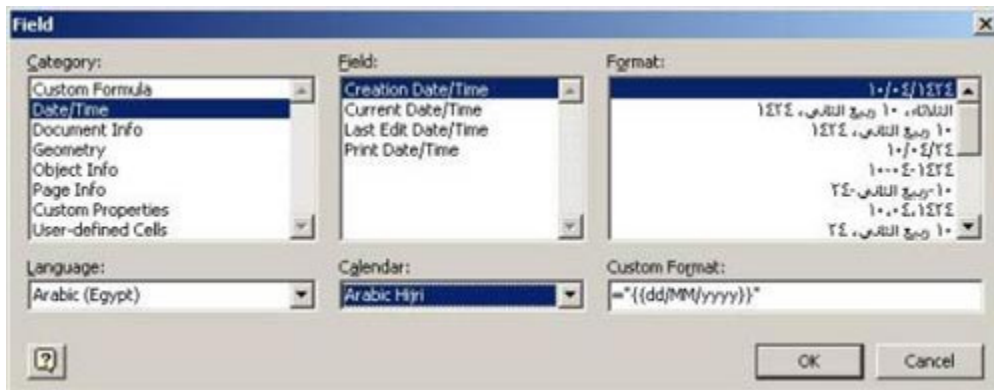
Saudi Arabia, the largest Arabic market. Most applications require a conversion and selection utility with an option to display the date in either Gregorian or Hijri formats. Your BiDi software application should respect the Date preferences given by the user in Control Panel.

Four different sets of month names are used in Arabic and applications need to provide for a user-selectable set of month names. Month names can be either Gregorian or Hijri months, and either the Latin or Arabic alphabet might be used to write them. Thus, there are many options for date display.

The figure below illustrates Hijri calendar date display:



The figure below shows a control panel in Arabic enabled VISIO, for setting the date to conform to the Hijri calendar:






## Summary

In summary, Arabic is a language that requires more preparation and up front design decisions than many other languages. Because widespread demand for Arabic translation is relatively recent, some standards are yet to be established. The Arabic speaking world encompasses more than 280 million people (some 165 million people of which have some knowledge of Modern Standard Arabic.) This is a growing market for western vendors. Increased technical information exchange between western countries and the Middle East will lead to more extensive target language glossary choices.

## Glossary

| Term                       | Definition   |
|----------------------------|--|
| <b>BiDi</b>                | Bi-directional. Displaying text left-to-right (LTR) and right-to-left (RTL) simultaneously (for example, with Arabic characters as RTL text and with Latin characters as LTR). Also used loosely to mean right-to-left, as in "BiDi text." Arabic or Hebrew applications are called "BiDi applications." |
| <b>Bi-directional</b>      | Displaying text left-to-right (LTR) and right-to-left (RTL) simultaneously (for example, with Arabic characters as RTL text and with Latin characters as LTR).   |
| <b>Bilingual</b>           | We use the term bilingual app to refer to an enabled app with a bilingual user interface (such as the language of menus, dialogs, messages and Help is user-selectable).   |
| <b>Contextual Analysis</b> | Mapping from code points to glyphs. Usually called <i>shaping</i> . Unicode term is <i>rendering</i> .   |
| <b>Diacritics</b>          | Vowel markings printed above or below letters.   |
| <b>Enabled</b>             | An application that is enabled if it provides BiDi support but still has a Latin user interface.   |
| <b>Glyph</b>               | A printed character representing a single font point.  |
| <b>Glyphing</b>            | Synonym for <i>Rendering</i> or <i>Shaping</i> .   |
| <b>Hijri Date</b>          | Lunar calendar used in Islamic countries.  |
| <b>Hindi Digits</b>        | Numerals used in traditional Arabic text.<br>  |
| <b>Kashida</b>             | Arabic character (looking rather like an elongated hyphen) used to extend the joiner between two Arabic letters. Used for justification flush to both margins.   |

| Term                      | Definition   |
|---------------------------|--|
| <b>Layout</b>             | The mapping from logical order to visual order of a text string.   |
| <b>Ligatures</b>          | Characters joined together to form a single glyph. Arabic supports both two-character ligatures and three-character ligatures.   |
| <b>Localized</b>          | An application that has been modified to have the look and feel of having been created in the target market, including translation of the content into the target language.  |
| <b>Logical Order</b>      | Sequence of code points in which the initial characters of a word precede later characters, whether the word is displayed RTL or LTR.  |
| <b>Left-to-right Mark</b> | A non-printing code point informing the system how to lay out the following character(s).  |
| <b>LTR</b>                | Left to right.   |
| <b>Multilingual</b>       | Supporting multiple user interfaces in different languages that can be selected by the user on the fly.  |
| <b>Neutral</b>            | In Unicode, a character that does not have a direction of its own, but takes its direction from the characters around it. Most punctuation marks are neutrals.   |
| <b>Physical Order</b>     | Sequence of code points in the same order as displayed. Also called Visual order.  |
| <b>Push Mode</b>          | The entry of text in the opposite direction from the reading order--so called because the caret remains stationary and the characters are "pushed" away from it.   |
| <b>Reading Order</b>      | The primary language of a line or paragraph. If the reading order is RTL, the runs will be laid out in sequence from RTL, or vice versa for LTR. Example: the logical string "abc~~" is displayed as shown in LTR reading order, but as "~~abc" in RTL. (~~ represents an Arabic word) |
| <b>Right-to-left Mark</b> | A non-printing code point informing the system how to lay out the following character(s).  |
| <b>RTL</b>                | Right to left.   |
| <b>Rendering</b>          | Unicode term for Shaping.  |
| <b>Symmetric Swapping</b> | The display of a directional character with its symmetric opposite in RTL. Examples are <{[ ]}>.   |
| <b>Shaping</b>            | Mapping from code points to glyphs. Unicode term is <i>rendering</i> . Also called Contextual analysis.  |

| Term                | Definition   |
|---------------------|--|
| <b>Unicode</b>      | A character coding system designed to support the worldwide interchange, processing, and display of the written texts of the diverse languages and technical disciplines of the modern world. In addition, it supports classical and historical texts of many written languages. |
| <b>Visual Order</b> | Sequence of code points in the same order as displayed. Also called Physical order.  |

### Further Reading

The following Web sites contain some useful information related to developing applications with an Arabic user interface, as well as information about the Arabic language itself.

#### Resources for Development in Microsoft Windows

[http://www.microsoft.com/middleeast/arabicdev/office/office2003/wpapers\\_sharepoint.asp](http://www.microsoft.com/middleeast/arabicdev/office/office2003/wpapers_sharepoint.asp)

<http://msdn.microsoft.com/library/?url=/library/en-us/wceinternational5/html/wce50oriArabicOSDesignDevelopment.asp>

<http://www.microsoft.com/middleeast/arabicdev/Windows/winXP/kBase/wFontA.asp>

#### Overview of Arabic language issues

[http://en.wikipedia.org/wiki/Arabic\\_language](http://en.wikipedia.org/wiki/Arabic_language)

#### Arabicizing Windows Applications to Read and Write Arabic:

[http://www.uga.edu/islam/arabic\\_windows.html](http://www.uga.edu/islam/arabic_windows.html)

#### Authoring HTML for Middle Eastern Content:

<http://www.microsoft.com/globaldev/handson/dev/Mideast.msp>

#### Historic development of Arabic language

[http://arabworld.nitle.org/texts.php?module\\_id=1&reading\\_id=17](http://arabworld.nitle.org/texts.php?module_id=1&reading_id=17)

#### Challenges with Scientific Translation

<http://www.translationdirectory.com/article10.htm>

#### Overview of Modern Standard Arabic

<http://fizzylogic.com/users/bulbul/lmp/profiles/modern-standard-arabic.html>

### ENLASO's Localization Solutions

For more information on how ENLASO can assist you with all of your localization needs, please contact us at [marketing@translate.com](mailto:marketing@translate.com) or call us at 303 516 0857.

**ENLASO Corporation – [www.translate.com](http://www.translate.com)**