About MLTE Demo

MLTE Demo is a quick & dirty demo application I've been using to test the new Unicode-savvy text engine built in Mac OS 9.0, known as **MLTE** (Multi-Lingual Text Engine) or **Textension**. The first version of this demo was written in a few hours starting from the WASTE 2.0a12 demo, ripping out all WASTE calls, and replacing them with equivalent MLTE calls. The full source code is included and can be reused freely.
NOTE: This application is completely untested and may crash your machine and cause data loss. Run it at your own risk.

See Apple's website:

<http://developer.apple.com/techpubs/macos8/TextIntlSvcs/MultilingualTextEditor/multilingualtexteditor.html>

for more information about MLTE.
See:

<http://www.merzwaren.com/waste/>

for more information about WASTE 2.0, another text engine for the Macintosh written by yours truly (Marco Piovanelli). You can contact me at the following address:

<mailto:marco.piovanelli@pobox.com>

About the Format of this File

The file you're reading now is not a traditional Macintosh 'TEXT' file. It's a Unicode text file saved in the canonical **UTF-16** transformation format, an industry standard supported by several modern applications and operating systems. There are several interesting things about **UTF-16** files:

• Most characters in a UTF-16 file take up two bytes. On the other hand, in plain ASCII files, each byte corresponds to exactly one character.
• UTF-16 files come in two slightly different flavors known as **big-endian** (UTF-16BE) and **little-endian** (UTF-16LE). The difference between these two flavors is the order of the two bytes that make up each character. Big-endian means that the most significant byte comes first in the file, little-endian means that the least significant byte comes first. Little-endian UTF-16 files are common on Windows platforms. Most other platforms, including Mac OS and Mac OS X, prefer big-endian files. Unicode-savvy programs should always recognize both flavors.
• UTF-16 files usually begin with two bytes known as the **byte-order mark** (or "BOM" for short). This mark has two important functions. First, it works like a "signature", or a strong hint for text editors that the file in question is actually a Unicode file rather than some other format. Secondly, it allows editors to tell the

byte order “flavor” of the file. The BOM is 0xFEFF for big-endian files and 0xFFFE for little-endian files. Unicode-savvy programs should always make sure the Unicode files they save begin with a BOM.
• There are several different conventions for line and paragraph breaks in UTF-16 files. The Unicode standard includes two characters meant explicitly and unambiguously to separate lines and paragraphs: U+2028 LINE SEPARATOR and U+2029 PARAGRAPH SEPARATOR. However, a lot of UTF-16 files use the old conventions carried over from the ASCII days, which differ from platform to platform. For example, line breaks in UTF-16 files coming from Windows platforms are often marked by the pair U+000D U+000A (CR+LF).
• On the Macintosh platform, the recommended file type for UTF-16 files is ‘utxt’. However, ‘TEXT’ files beginning with a BOM should be treated as UTF-16 files as well.

This particular file contains WASTE 2.0 formatting information in its resource fork, as four resources of types ‘WEcf’, ‘WEpf’, ‘WEst’ and ‘WEru’. The format of these resources, or formatting “scraps”, is designed to be efficient, extensible and usable both with traditional TEXT files and with Unicode text files. They can store a rich complement of formatting information, including styles, rulers and embedded objects such as pictures. WASTE 2.0 (starting from version 2.0a15) provides convenient APIs to create and parse such scraps and the code in this demo shows how MLTE-based applications can parse them without WASTE.

Version History

1.0a1 (2000-01-27)
• First public release.

1.0a2 (2000-01-31)
• Keyboard equivalents now work.
• Added Carbon target to the project.
• Added Features menu to selectively enable ATSUI font features.
• Enabled embedded objects (graphics, sound, movies).

1.0a3 (2000-05-30)
• Added support for reading styled ‘utxt’ files containing WASTE 2.0 formatting information.
• Added support for little-endian ‘utxt’ files.

1.0a4 (2001-12-30)
• Updated project file to CodeWarrior Pro 7.
• Updated WASTE streams stuff to the latest version (2.1a4).
• Minor changes for compatibility with the latest Universal Headers (3.4).
• Fixed a problem that would prevent the carbonated demo from launching with CarbonLib 1.5.