UNICODE™ 6.0
—Finally There?

Dr. Bruce K. Haddon

PART 1: THE MOTIVATION

Character Encodings
- Morse Code
- Baudot Code
- Hollerith
- EBCDIC
- ASCII
- etc.

ASCII (ANSI-X3.4)
- ANSI-X3.4-1986 (R1997); ISO-14962-1997
- 7-bit code
- Purpose: information interchange
- Popular choice for programming languages (e.g., C/C++, Pascal, Ada, Java/C#, etc.)
- Became the de facto code set and encoding for (too?) many applications
ASCII—The Code

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>NUL</td>
<td>SOH</td>
<td>STX</td>
<td>ETX</td>
<td>EOT</td>
<td>ENQ</td>
<td>ACK</td>
<td>BEL</td>
<td>BS</td>
<td>HT</td>
<td>LF</td>
<td>VT</td>
<td>FF</td>
<td>CR</td>
<td>SO</td>
</tr>
<tr>
<td>1</td>
<td>DLE</td>
<td>DC1</td>
<td>DC2</td>
<td>DC3</td>
<td>DC4</td>
<td>NAK</td>
<td>SYN</td>
<td>ETB</td>
<td>CAN</td>
<td>EM</td>
<td>SUB</td>
<td>ESC</td>
<td>FS</td>
<td>GS</td>
<td>RS</td>
</tr>
<tr>
<td>2</td>
<td>SP</td>
<td>!</td>
<td>&quot;</td>
<td>#</td>
<td>$</td>
<td>%</td>
<td>&amp;</td>
<td>'</td>
<td>(</td>
<td>)</td>
<td>*</td>
<td>+</td>
<td>,</td>
<td>-</td>
<td>.</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>:</td>
<td>;</td>
<td>=</td>
<td>&gt;</td>
<td>?</td>
</tr>
<tr>
<td>4</td>
<td>@</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>E</td>
<td>F</td>
<td>G</td>
<td>H</td>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
<td>M</td>
<td>N</td>
</tr>
<tr>
<td>5</td>
<td>P</td>
<td>Q</td>
<td>R</td>
<td>S</td>
<td>T</td>
<td>U</td>
<td>V</td>
<td>W</td>
<td>X</td>
<td>Y</td>
<td>Z</td>
<td>{</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>`</td>
<td>a</td>
<td>b</td>
<td>c</td>
<td>d</td>
<td>e</td>
<td>f</td>
<td>g</td>
<td>h</td>
<td>i</td>
<td>j</td>
<td>k</td>
<td>l</td>
<td>m</td>
<td>n</td>
</tr>
<tr>
<td>7</td>
<td>p</td>
<td>q</td>
<td>r</td>
<td>s</td>
<td>t</td>
<td>u</td>
<td>v</td>
<td>w</td>
<td>x</td>
<td>y</td>
<td>z</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ISO 646

- First version of “ASCII” by the International Standards Organization (with “National” variants)
- 7-bit codes
- Currently 25 National variants
- Changes certain characters, e.g., `5B16` in ASCII is `Æ` in 646-DK
- Largely obsolete

ISO 646—Basis Code

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>NUL</td>
<td>SOH</td>
<td>STX</td>
<td>ETX</td>
<td>EOT</td>
<td>ENQ</td>
<td>ACK</td>
<td>BEL</td>
<td>BS</td>
<td>HT</td>
<td>LF</td>
<td>VT</td>
<td>FF</td>
<td>CR</td>
<td>SO</td>
</tr>
<tr>
<td>1</td>
<td>DLE</td>
<td>DC1</td>
<td>DC2</td>
<td>DC3</td>
<td>DC4</td>
<td>NAK</td>
<td>SYN</td>
<td>ETB</td>
<td>CAN</td>
<td>EM</td>
<td>SUB</td>
<td>ESC</td>
<td>FS</td>
<td>GS</td>
<td>RS</td>
</tr>
<tr>
<td>2</td>
<td>SP</td>
<td>!</td>
<td>&quot;</td>
<td>#</td>
<td>$</td>
<td>%</td>
<td>&amp;</td>
<td>'</td>
<td>(</td>
<td>)</td>
<td>*</td>
<td>+</td>
<td>,</td>
<td>-</td>
<td>.</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>:</td>
<td>;</td>
<td>=</td>
<td>&gt;</td>
<td>?</td>
</tr>
<tr>
<td>4</td>
<td>@</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>E</td>
<td>F</td>
<td>G</td>
<td>H</td>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
<td>M</td>
<td>N</td>
</tr>
<tr>
<td>5</td>
<td>P</td>
<td>Q</td>
<td>R</td>
<td>S</td>
<td>T</td>
<td>U</td>
<td>V</td>
<td>W</td>
<td>X</td>
<td>Y</td>
<td>Z</td>
<td>[</td>
<td>]</td>
<td>^</td>
<td>_</td>
</tr>
<tr>
<td>6</td>
<td>`</td>
<td>a</td>
<td>b</td>
<td>c</td>
<td>d</td>
<td>e</td>
<td>f</td>
<td>g</td>
<td>h</td>
<td>i</td>
<td>j</td>
<td>k</td>
<td>l</td>
<td>m</td>
<td>n</td>
</tr>
<tr>
<td>7</td>
<td>p</td>
<td>q</td>
<td>r</td>
<td>s</td>
<td>t</td>
<td>u</td>
<td>v</td>
<td>w</td>
<td>x</td>
<td>y</td>
<td>z</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ISO 646—Swedish/Finnish Variant

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>NUL</td>
<td>SOH</td>
<td>STX</td>
<td>ETX</td>
<td>EOT</td>
<td>ENQ</td>
<td>ACK</td>
<td>BEL</td>
<td>BS</td>
<td>HT</td>
<td>LF</td>
<td>VT</td>
<td>FF</td>
<td>CR</td>
<td>SO</td>
</tr>
<tr>
<td>1</td>
<td>DLE</td>
<td>DC1</td>
<td>DC2</td>
<td>DC3</td>
<td>DC4</td>
<td>NAK</td>
<td>SYN</td>
<td>ETB</td>
<td>CAN</td>
<td>EM</td>
<td>SUB</td>
<td>ESC</td>
<td>FS</td>
<td>GS</td>
<td>RS</td>
</tr>
<tr>
<td>2</td>
<td>SP</td>
<td>!</td>
<td>&quot;</td>
<td>#</td>
<td>$</td>
<td>%</td>
<td>&amp;</td>
<td>'</td>
<td>(</td>
<td>)</td>
<td>*</td>
<td>+</td>
<td>,</td>
<td>-</td>
<td>.</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>:</td>
<td>;</td>
<td>=</td>
<td>&gt;</td>
<td>?</td>
</tr>
<tr>
<td>4</td>
<td>@</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>E</td>
<td>F</td>
<td>G</td>
<td>H</td>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
<td>M</td>
<td>N</td>
</tr>
<tr>
<td>5</td>
<td>P</td>
<td>Q</td>
<td>R</td>
<td>S</td>
<td>T</td>
<td>U</td>
<td>V</td>
<td>W</td>
<td>X</td>
<td>Y</td>
<td>Z</td>
<td>Ä</td>
<td>Ö</td>
<td>Å</td>
<td>^</td>
</tr>
<tr>
<td>6</td>
<td>`</td>
<td>a</td>
<td>b</td>
<td>c</td>
<td>d</td>
<td>e</td>
<td>f</td>
<td>g</td>
<td>h</td>
<td>i</td>
<td>j</td>
<td>k</td>
<td>l</td>
<td>m</td>
<td>n</td>
</tr>
<tr>
<td>7</td>
<td>p</td>
<td>q</td>
<td>r</td>
<td>s</td>
<td>t</td>
<td>u</td>
<td>v</td>
<td>w</td>
<td>x</td>
<td>y</td>
<td>z</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
ISO/IEC 8859

- 8 bit codes
- Currently, 16 variants (called “Parts”)
- 7-bit subset of each = ASCII (exactly)
- Each 8859 variant (Part) redefines the code points from \$0x\$FF16
- e.g., ISO/IEC 8859-1 is “Latin-1”, ISO/IEC 8859-5 is “Latin/Cyrillic”,
  ISO/IEC 8859-9 is “Latin-5”,
  ISO/IEC 8859-11 is “Latin/Thai5”,
  ISO/IEC 8859-15 is “Latin-9” (or “Latin-0”)

Other “National” and ISO Standards

- Japanese Industrial Standards
- Series of encodings (>15), all including ASCII and “wide character” ASCII
- Big 5, GB, GB/T, CNS (Chinese)
- KS (Korean)
- TCVN (Vietnamese)
- UNIX Extended Code (UEC)
- Escaping convention to allow intermixing of ASCII and any of the above (Open Consortium, OSF, UI, USLP: 1991)
**Shift-JIS**

- ASCII
  - \( 21-7E_{16} \)
  - half-width katakana
  - \( A1-DF_{16} \)
- JIS X 0208:1977
  - 1st byte \( 81-9F_{16}, \ E0-EF_{16} \)
  - 2nd byte \( 40-7E_{16}, \ 80-FC_{16} \)

**EUC-JP**

- ASCII or JIS-Roman
  - \( 21-7E_{16} \)
  - half-width katakana
  - \( 8E_{16} \) followed by \( A1-DF_{16} \)
- JIS X 0208:1977
  - 1st byte \( 81-9F_{16}, \ E0-EF_{16} \)
  - 2nd byte \( 40-7E_{16}, \ 80-FC_{16} \)
- JIS X 0212:1990
  - 1st byte \( 8F_{16} \)
  - 2nd byte \( A1-FE_{16} \)
  - 3rd byte \( A1-FE_{16} \)

**Terminology (1)**

- “Character Set”
- “Glyph”
- “(Natural) Encoding”
- “Code page/set”
- “Code point”
- “Transcoding”
- “Transformation”

**Terminology (2)**

- “Single byte, simple”
- “Double byte (simple)”
  - “Multi-byte (simple)”
- “Single byte, complex”
- “Bi-Directional” (‘bi-di’)
- “Universal”

**PART 2 THE SOLUTION**
ISO/IEC-10646

- “Universal” character set—each code point is 32 bits, “0” + 31 bits (UCS-4)
- Initial approach, use 16-bit “planes,” each containing defined national subsets
- 15 bits for “plane” number, 16 bits define character encoding within plane

The Unicode Standard

- Consortium of, now,
  - 8 “full” members,
  - 4 “institutional” members,
  - 1 “supporting” member,
  - 27 “associate” members, and
  - a long list of individual, student, and liaison members
- Interoperability with ISO 8859-1 Latin-1 (including ASCII)

- Encompassing all scripts in use—now, all scripts ever used (or shall be used!)

The Unicode Consortium®

Full Members
- Adobe Systems, Inc.
- Apple Computer, Inc.
- Google
- IBM Corporation
- Microsoft Corporation
- Oracle Corporation
- SAP AG
- Yahoo!

Institutional Members
- Adobe Systems, Inc.
- Apple Computer, Inc.
- Google
- IBM Corporation
- Microsoft Corporation
- Oracle Corporation
- SAP AG
- Yahoo!

Supporting Members
- Monotype Imaging

Associate Members
- Adobe Systems, Inc.
- Adobe Systems, Inc.
- Apple Computer, Inc.
- Google
- IBM Corporation
- Microsoft Corporation
- Oracle Corporation
- SAP AG
- Yahoo!

Supporting Members
- Monotype Imaging
History of Unicode Standard

- Unicode 6.0 (October, 2010)
- Unicode 5.2 (October, 2009)
- Unicode 5.1 (April, 2008)
- Unicode 5.0 (November, 2006)
- Unicode 4.10 (March, 2005)
- Unicode 4.0.1 (March, 2004)
- Unicode 4.0 (March, 2003)
- Unicode 3.2.0 (March, 2002)
- Unicode 3.1.1 (August, 2001)
- Unicode 3.1.0 (March, 2001)
- Unicode 3.0.1 (August, 2000)
- Unicode 3.0 (September, 1999)
- Unicode 2.0 (July, 1996)
- Unicode 1.0 (October, 1991)

The Unicode Standards.

Version 6.0, 2011
ISBN 978-1-932133-01-6

Version 5.0, 2006


Addison-Wesley Developers Press, Reading, MA.

Unicode Design Principles

- Universality
  - A single, universal repertoire for all human (and some non-human) writing – see next slide

- Efficiency
  - Easy to parse and process
  - A compact representation that fits into an average of no more than sixteen bits.

- Characters, not glyphs
  - Encode each abstract character once (but ...)

- Semantics
  - Well-defined character semantics

- Plain text
  - Characters represent plain text

Logical Order
- Storage default is logical order, not printed order

Unification
- Han, and other, unification, e.g., CJKV conceptually same ideograms unified

Dynamic Composition
- Accented forms may be composed

Stability
- Characters once assigned cannot be reassigned

Convertibility
- Round trip preservation, hence many a’s, alpha, aleph, etc.

Compatibility with “wide” characters, Arabic contextual forms, ligatures, etc.

Unicode Character Set (Examples)

- Arabic
- Gregian
- Karashnith
- Shavian
- IPA Numbers (Decimal, Counting Rods, Cuneiform)
- Dingbats, Emoji
- Arrows, Blocks, Box Drawing Forms, and Geometric Shapes
- Miscellaneous Symbols Presentation Forms
- Braille Patterns Musical Symbols
- (Western, Byzantine, & Ancient Greek)

Natural Encoding: UTF-32
(subset of UCS-4 : since Unicode Version 2)

0 – 10FFFF16
(17 planes)

1114,112 code points
The Planes Currently in Use

- Basic Multilingual Plane (BMP)
  - Values 0000-FFFF\textsubscript{16}, less the values D800-DFFF\textsubscript{16}, represent themselves
- All other values, 10000-10FFFF\textsubscript{16}, are packed into pairs of their “surrogates”, the values D800-DFFF\textsubscript{16}

Efficient Encoding: UTF-16

- “Surrogates” D800–DFFF \( \times 2 \) 2048 values
- 1,048,576 code points

Count of Unicode Characters
PART 3 USING UNICODE

Interoperable Encoding: UTF-8

<table>
<thead>
<tr>
<th>UTF-32</th>
<th>UTF-8</th>
<th>bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000FF</td>
<td>0-......</td>
<td>7</td>
</tr>
<tr>
<td>000000FF</td>
<td>110-....</td>
<td>11</td>
</tr>
<tr>
<td>000FFFF</td>
<td>1110-...</td>
<td>16</td>
</tr>
<tr>
<td>000FFFFF</td>
<td>11110-..</td>
<td>21</td>
</tr>
<tr>
<td>0FFFFF</td>
<td>111110-</td>
<td>26</td>
</tr>
<tr>
<td>FFFFFFF</td>
<td>1111110-</td>
<td>31</td>
</tr>
</tbody>
</table>

Compatibility—“round tripping”

- ASCII is in twice (21-7E16, FF01-FF5E16)
- 42 sets of decimal digits, 0-9
- 18 space characters (not counting tabs, etc.)
- 22 hyphen or dash characters
- composed and decomposed characters

Han Unification

<table>
<thead>
<tr>
<th>Unicode</th>
<th>China</th>
<th>Taiwan</th>
<th>Japan</th>
<th>Korea</th>
</tr>
</thead>
<tbody>
<tr>
<td>4580</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>4581</td>
<td>与</td>
<td>与</td>
<td>与</td>
<td>—</td>
</tr>
<tr>
<td>5224</td>
<td>判</td>
<td>判</td>
<td>判</td>
<td>判</td>
</tr>
<tr>
<td>5668</td>
<td>器</td>
<td>器</td>
<td>器</td>
<td>器</td>
</tr>
<tr>
<td>5857</td>
<td>字</td>
<td>字</td>
<td>字</td>
<td>字</td>
</tr>
<tr>
<td>6077</td>
<td>海</td>
<td>海</td>
<td>海</td>
<td>海</td>
</tr>
<tr>
<td>9038</td>
<td>逸</td>
<td>逸</td>
<td>逸</td>
<td>逸</td>
</tr>
<tr>
<td>9AA8</td>
<td>骨</td>
<td>骨</td>
<td>骨</td>
<td>骨</td>
</tr>
</tbody>
</table>
### Unicode Characteristics

- Character name
- General Category
- Canonical Combining Classes
- Bi-directional Category
- Character Decomposition Mapping
- Decimal digit value
- Digit value
- Numeric value

### Example (1) of UnicodeData.txt

<table>
<thead>
<tr>
<th>Character</th>
<th>General Category</th>
<th>Canonical Combining Classes</th>
<th>Bi-directional Category</th>
<th>Character Decomposition Mapping</th>
<th>Decimal digit value</th>
<th>Digit value</th>
<th>Numeric value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0C66</td>
<td>TELUGU DIGIT ZERO</td>
<td>Nd</td>
<td>L</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0C67</td>
<td>TELUGU DIGIT ONE</td>
<td>Nd</td>
<td>L</td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0C68</td>
<td>TELUGU DIGIT TWO</td>
<td>Nd</td>
<td>L</td>
<td></td>
<td></td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>0C69</td>
<td>TELUGU DIGIT THREE</td>
<td>Nd</td>
<td>L</td>
<td></td>
<td></td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>0C6A</td>
<td>TELUGU DIGIT FOUR</td>
<td>Nd</td>
<td>L</td>
<td></td>
<td></td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>0C6B</td>
<td>TELUGU DIGIT FIVE</td>
<td>Nd</td>
<td>L</td>
<td></td>
<td></td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>0C6C</td>
<td>TELUGU DIGIT SIX</td>
<td>Nd</td>
<td>L</td>
<td></td>
<td></td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>0C6D</td>
<td>TELUGU DIGIT SEVEN</td>
<td>Nd</td>
<td>L</td>
<td></td>
<td></td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>0C6E</td>
<td>TELUGU DIGIT EIGHT</td>
<td>Nd</td>
<td>L</td>
<td></td>
<td></td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>0C6F</td>
<td>TELUGU DIGIT NINE</td>
<td>Nd</td>
<td>L</td>
<td></td>
<td></td>
<td>9</td>
<td>9</td>
</tr>
</tbody>
</table>

### Example (2) of UnicodeData.txt

<table>
<thead>
<tr>
<th>Unicode Code</th>
<th>Character</th>
<th>General Category</th>
<th>Canonical Combining Classes</th>
<th>Bi-directional Category</th>
<th>Character Decomposition Mapping</th>
<th>Decimal digit value</th>
<th>Digit value</th>
<th>Numeric value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0024</td>
<td>DOLLAR SIGN</td>
<td>Sc</td>
<td>ET</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00A2</td>
<td>CENT SIGN</td>
<td>Sc</td>
<td>ET</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00A5</td>
<td>YEN SIGN</td>
<td>Sc</td>
<td>ET</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20A0</td>
<td>EURO-CURRENCY</td>
<td>Sc</td>
<td>ET</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20A2</td>
<td>CRUZEIRO SIGN</td>
<td>Sc</td>
<td>ET</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20A3</td>
<td>FRENCH FRANC</td>
<td>Sc</td>
<td>ET</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20A4</td>
<td>LIRA SIGN</td>
<td>Sc</td>
<td>ET</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20A5</td>
<td>NAIRA SIGN</td>
<td>Sc</td>
<td>ET</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20A7</td>
<td>PESETA SIGN</td>
<td>Sc</td>
<td>ET</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20AC</td>
<td>EURO SIGN</td>
<td>Sc</td>
<td>ET</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Composed and Decomposed Characters

**Composed:**

- \( \ddot{a} \) (\( \dddot{a} \equiv U+00E5 \))

**Decomposed:**

- \( \tilde{a} \) (= \( U+0061 \ U+030A \))

**Multiple accents:**

- \( \ddot{\dddot{a}} \) (= \( U+0065 \ U+0334 \))

- \( \tilde{\dddot{a}} \) (= \( U+0061 \ U+030A \ U+0334 \))
Example (3) of UnicodeData.txt

<table>
<thead>
<tr>
<th>Code</th>
<th>Character Description</th>
<th>Value</th>
<th>Flag</th>
<th>Loc</th>
<th>Enc</th>
<th>Code</th>
<th>Character Description</th>
<th>Value</th>
<th>Flag</th>
<th>Loc</th>
<th>Enc</th>
</tr>
</thead>
<tbody>
<tr>
<td>00E4</td>
<td>LATIN SMALL LETTER A WITH DIAERESIS</td>
<td>0061</td>
<td>Ll</td>
<td>0</td>
<td>L</td>
<td>0064</td>
<td>LATIN SMALL LETTER A DIAERESIS</td>
<td>00C4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00E5</td>
<td>LATIN SMALL LETTER A WITH RING ABOVE</td>
<td>0061</td>
<td>Ll</td>
<td>0</td>
<td>L</td>
<td>0065</td>
<td>LATIN SMALL LETTER A RING</td>
<td>00C5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00E6</td>
<td>LATIN SMALL LETTER AE</td>
<td>0061</td>
<td>Ll</td>
<td>0</td>
<td>L</td>
<td>0065</td>
<td>LATIN SMALL LETTER A E</td>
<td>00C6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00E7</td>
<td>LATIN SMALL LETTER C WITH CEDILLA</td>
<td>0063</td>
<td>Ll</td>
<td>0</td>
<td>L</td>
<td>0067</td>
<td>LATIN SMALL LETTER C CEDILLA</td>
<td>00C7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00E8</td>
<td>LATIN SMALL LETTER E WITH GRAVE</td>
<td>0065</td>
<td>Ll</td>
<td>0</td>
<td>L</td>
<td>0068</td>
<td>LATIN SMALL LETTER E WITH GRAVE</td>
<td>00C8</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Equivalent (UAX #15)

- Easier to consider “equivalence” rather than “equality”
- Two kinds of equivalence:
  - Canonical equivalence—sequences of bytes represent the same (abstract) character, i.e., they “look” the same.
  - Compatibility equivalence—sequences of bytes represent the same fundamental character, i.e., they (in some sense) act the same.

Comparison and Normalization (UAX #15—newly revised in 6.0)

What does it mean to ask: “When are two (Unicode) strings equal?”

- The bytes are identical: YES! BUT INADEQUATE
- The characters look the same, i.e., are the same ignoring differences in ways accents are combined
- The characters are the “fundamentally” the same, i.e., ignoring compatibility differences
- What if there are multiple accents on a character?
- Is “decomposed characters” the best way to represent accented characters?

Unicode Publications

- **US** Unicode Standard
  - The standard, i.e., the book, with the character set, code points, and conformance requirements
- **UAX** Unicode Standard Annexes
  - Subsections of the standard, included in the Standard, containing explanatory details.
- **UTS** Unicode Technical Standard (electronic only)
  - Associated standards, such as compression, collation, XML usage, etc.
- **UTR** Unicode Technical Report (electronic only)
  - Other informative material, e.g., the encoding model, property model, mathematical support, security, etc.
Canonical Equivalence (UAX #15)

- Combining diacritics (accents) come after the character in the Unicode stream
- Hangul (Korean) decomposition represents syllabic to phonemic conversion.
- Singletons are “decompositions” that produce just one character (and never reverse).

Combining sequence

\[ \text{Combining sequence:} \quad \text{C} \rightarrow \text{C} \]

Ordering of combining marks

\[ \text{Ordering of combining marks:} \quad \text{q} \rightarrow \text{q} \]

Hangul

\[ \text{Hangul:} \quad \text{김} \rightarrow \text{김} \]

Singleton

\[ \text{Singleton:} \quad \text{Ω} \rightarrow \text{Ω} \]

Compatibility Equivalence (UAX #15)

- Compatible forms arise for a variety of reasons
  - Difference is simply visual
  - Difference is semantic, but is actually the same character
  - Difference is in some behavior, but is actually the same character
  - “Round-trip” requires the distinction

Normalization Processes (UAX #15)

- Three processes:
  - canonical decomposition (D)—replacing all composite characters by their decomposed sequences, and applying canonical diacritic ordering
  - compatibility decomposition (K)—canonical decomposition, then replacing compatibility forms of a character with the fundamental form
  - canonical composition (C) after decomposition, replacing some decomposed characters by their “usual” composed forms, if possible

Example: Composition / Compatibility (UAX #15)

- In the first example, decomposition separates the accent, but composition puts it back together again
- In the second, the ångstrom character and the ohm character are replaced by the letter “a with ring” and the letter “omega”
Unicode does have all these characters
If a compatibility normalization were applied:

$$H = \int d\tau (E^2 + \mu H^2)$$

Some careful distinctions need to be made

mid-eastern texts inherently bi-directional
The Unicode standard (Unicode Standard Annex #9) specifies an embedding algorithm
- Direction characteristics (strong, weak, neutral)
- Directionality overrides
- Language overrides
The order of characters in a file follows the “natural” order (no directionality).

ما هي الشفرة الموحدة "يونيكود"؟
Example: Chinese busses

Sorting

(UTS #10 Collation—newly revised in 6.0)
- Sorting by code value does not do the job!
- Unicode specifies five “levels” of collation, applied to NFKD normalization
  - i.e., base, accent, case, punctuation, identity
- Orderings
  - dictionary, language specific
  - telephone directory
  - radical and stroke order, or phonetic, for the Han characters
  - etc.
- There are other considerations (UTS#10 is 74 pages long!)

Sorting Example (UTS #10)

L1 Base characters  role < roles < rule
L2 Accents  role < rôle < roles
L3 Case  role < Role < rôle
L4 Punctuation  role < “role” < Role
Ln Tie-Breaker  role < role < “role”

Regular Expressions (UTS #18)

- Ranges specified by:
  - hex codes,
  - Type (digit, letter, separator, etc.)
  - Language block (Latin, Greek, Thai, etc.)
  - Function (SOL, EOL, white space, etc.)
- Other features
  - Level (see Sorting/Collation)
  - Normalized/Un-normalized
Byte Order Marker

- U+FEFF ZERO WIDTH NO-BREAK SPACE
- U+FFFE not a character code

Bytes at the beginning of a file:
- FF FE UTF-16 high byte first
- FE FF UTF-16 low byte first
- EF BB BF UTF-8
- 00 00 FE FF UTF-32 high byte first
- 00 00 FF FE UTF-32 low byte first

Shavian Script

New Scripts in Version 4.0

- BMP, Plane 0
  - Limbu
  - Tai Le
- Plane 1
  - Shavian
  - Linear B
  - Ugaritic Cuneiform
  - Cypriot syllabary
  - Osmanya
  - High Voltage Sign (26A116)
  - Rejected for 4.0
  - Klingon
  - Total 1226 new

4.0 Statistics

- Graphic 94,245
- Format 137
- Control 65
- Private Use 137,468
- Noncharacter 66

New Scripts in Version 5.0

- BMP, Plane 0
  - N’ko
  - Balinese
  - Phags-Pa
- Plane 1
  - Cuneiform
  - Counting Rods
  - Phoenician
  - Small additions to Latin, Greek, Cyrillic, Hebrew, Devanagari, Kannada
  - Some symbols
  - Total 1369 new

5.0 Statistics

- Graphic 98,684
- Format 140
- Control 65
- Private Use 137,468
- Noncharacter 66
Unicode 6.0

- Alchemical Symbols *
- Bamum Supplement *
- Batak
- Brahmi *
-CJK Unified Ideographs Extension D *
- Emoji symbols
- Emoticons *
- Ethiopic Extended-A
- Kannada Supplement *
- Kana Horizontal
- Miscellaneous Symbols And Pictographs *
- Playing Cards *
- Transport And Map Symbols *

Adds 2088 characters
* Added in Plane 1

---

Emoji

---

Unicode Applications

- HTML, DHTML, XHTML
- XML, WML, CSS
- OpenOffice 3.1+
- Windows Office 12+
- Mac 9.2, X
- IBM AIX
- CORBA 3.0
- Java (U4.0) & C# (U5.1)
- C/C++ (wchar_t)
- JavaScript (ECMAScript)
- Browsers (Netscape 4+, IE 5+, Chrome, Safari)
- VB, VB.NET
- Ingres 2.6+
- IBM DB2
- LDAP
- Solaris 8, 9, 10 (UTF-8)
- Perl 5.6 (UTF-8), 5.8
- Oracle 8+ (UTF-8)
- TCL 8.1 (UTF-8)
- many others …

---

Unicode in the JDK

<table>
<thead>
<tr>
<th>Java JDK</th>
<th>Unicode</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>1.1.5</td>
</tr>
<tr>
<td>1.1</td>
<td>2.0</td>
</tr>
<tr>
<td>1.1.7</td>
<td>2.1</td>
</tr>
<tr>
<td>1.4</td>
<td>3.0</td>
</tr>
<tr>
<td>5.0</td>
<td>3.2</td>
</tr>
<tr>
<td>6.0</td>
<td>4.0</td>
</tr>
<tr>
<td>7.0</td>
<td>6.0</td>
</tr>
</tbody>
</table>

Hence char is 16 bits
UTF-16 (but no support for supplemental characters)

Code points (UTF-32)
java.text.Normalizer still at 3.2
If and when there is Java 7
Java, C#: Strings and Codepoints

- Codepoints to `String` (Constructor in `String`)
  ```java
  public String(int[] codePoints, int offset, int count)
  ```

- `String` to Codepoints
  ```java
  public int[] codePoints(String s)
  {
      final int[] utf32 = new int[s.length()];
      int index = 0;
      for (int offset = 0; offset != s.length();
          offset += Character.charCount(utf32[index++]))
      {
          utf32[index] = s.codePointAt(offset);
          return Arrays.copyOf(utf32, index);
      }
  }
  ```

Accepted Proposals for New Scripts

- Scripts for the Basic Multilingual Plane (BMP)
  - Mostly small changes and additions to existing blocks

- Scripts for Plane 1
  - Meroitic Hieroglyphs
  - Linear A *
  - Elbasan
  - Chakma
  - … plus ½ dozen more

Research Areas

- Sort Specification Languages
- Sort implementation techniques
- "Large" font management
- Converting to a 21 bit world
- Normalization libraries
- Han refinement
- Archeological research
- Extension of Unicode to further scripts

Further information

- Unicode:
  - [http://www.unicode.org/](http://www.unicode.org/)

- HTML, XML, the Web
  - [http://www.w3.org/TR/unicode-xml/](http://www.w3.org/TR/unicode-xml/)
  - Unicode in XML and other Markup Languages (Unicode Technical Report #20 W3C Note 13 June 2003)
  - [http://www.w3.org/TR/charmod/](http://www.w3.org/TR/charmod/)
  - Character Model for the World Wide Web 1.0 (W3C Working Draft 22 August 2003)

- History:
  - [http://www.loc.gov/marc/specifications/speccharucs.html](http://www.loc.gov/marc/specifications/speccharucs.html)

- Support: