## Data-centric XML

Character Sets

## Character Sets: Rationale

-Computer stores data in sequences of bytes

- each byte represents a value in range $0 . .255$
-Text data are intended to denote characters, not numbers
- Encoding defines a mechanism to associate bytes and characters
-Encoding can only cover finite number of character $\rightarrow$ character set
- Many terminology issues (character set, repertoire, encoding, coded character set, ...)


## Character Sets: History

- ASCII: American Standard Code for Information Interchange
- 7-bit character set, 1963 proposed, 1968 finalized
- ANSI X3.4-1986
- 32(34) control characters, 96(94) graphical characters
- Also known as CCITT International Alphabet \#5 (IA5), ISO 646
- national variants, international reference version
- DIN 66003: @ vs. §, [ vs. Ä, I vs. Ö, ] vs. Ü, ...


## Character Sets: History (2)

- 8-bit character sets: 190.. 224 graphic characters
- ISO 8859: European/Middle-East alphabets
- ISO-8859-1: Western Europe (Latin-1)
- ISO-8859-2: Central/Eastern Europe (Latin-2)
- ISO-8859-3: Southern Europe (Latin-3)
- ISO-8859-4: Northern Europe (Latin-4)
- ISO-8859-5: Cyrillic
- ISO-8859-6: Arabic
- ISO-8859-7: Greek
- ISO-8859-8: Hebrew
- ISO-8859-9: Turkish (Latin-5; replace Icelandic chars with Turkish)
- ISO-8859-10: Nordic (Latin-6; Latin 4 + Inuit, non-Skolt Sami)
- ISO-8859-11 (1999): Thai
- ISO-8859-13: Baltic Rim (Latin-7)
- ISO-8859-14: Celtic (Latin-8)
- ISO-8859-15: Western Europe (Latin-9, Latin-1 w/o fraction characters, plus Euro sign, Š, ž, œ, Y)
- ISO-8859-16: European (Latin-10, omit many symbols in favor of letters)


## Character Sets: History (3)

- Many proprietary 8-bit characters sets:
- IBM code pages (e.g. cp437)
- Windows code pages (e.g. windows-1252)
- Macintosh character sets (e.g. Mac-Roman)
- Multibyte Character Sets: one- or two-byte sequences
- Chinese: Big5 (traditional Chinese), GB-2312 (simplified Chinese)
- Japanese: JIS 0208, JIS 0212
- Korean, Vietnamese
- Multi-encoding standards: ISO 2022 escape sequences
- ISO-2022-JP (RFC 1554):
- ASCII: ESC (B
- JIS X 0208-1978: ESC \$ @
- JIS X 0208-1983: ESC \$ B
- JIS X 0201-Roman: ESC (J
- GB2312-1980: ESC \$ A
- KSC5601-1987: ESC \$ (C
- JIS X 0212-1990: ESC \$ (D


## Moji－Bake

－文字化け
－＂character changing＂，＂ghost characters＂
－artifacts of unknown／incorrect／inconsistent character set usage

## Character Sets: Terminology

- Character Model for the Web (http://www.w3.org/TR/ charmod/)
- Character: "The smallest component of written language that has semantic meaning; refers to the abstract meaning and/ or shape" (Unicode)
- Glyph: Unit of visual rendering
- different glyphs for the same character depending on font; also consider ligatures, Arabic character shapes
- Repertoire: Set of characters to be encoded
- Coded character set: assigning each character a number/ code position
- Character encoding form: representation of character codes in code units (not necessarily bytes)


## Character Sets: Terminology (2)

- Character encoding scheme: serialization of code units into byte sequences
- IANA charset


## Unicode

- Simultaneously published by Unicode Consortium and ISO
- Current version Unicode 5.1 == ISO/IEC 10646-2003 (including amendments 1 through 4)
- ISO 10646 has only character assignments; Unicode defines also algorithms, character properties, ...
- 100489 graphic characters
- 159 format characters
- 65 control characters
- 873882 reserved characters
- Coded Character Set is called UCS-4
- UCS-2 is a subset with < 65536 characters


## Unicode Principles

- Universality: A single repertoire for all languages
- Efficiency: Simple to parse and process
- Characters, not glyphs
- Semantics: characters shall have well-defined meaning
- Plain text: characters represent plain text
- Logical order: In memory, characters come in logical order
- Unification: characters duplicate across scripts are unified
- Dynamic composition: Accented characters can be composed dynamically
- Equivalent sequences: Precomposed characters have decomposed equivalence
- Convertibility: Unicode can be converted accurately into other CCS


## Unicode Characters

- Have stable code point
- e.g. U+00DF
- Have stable character name
- e.g. LATIN SMALL LETTER SHARP S
- Unicode standard gives "demo" glyph
- e.g. ß
- Unicode character database gives properties
- e.g. "Letter, lower case" (LI)


## Combining Characters

- Characters of class "combining" can be composed to new forms
- Used for accented characters and Hangul syllables
- e.g. U+0055,U+0308 -> U+00DB
(LATIN CAPITAL LETTER U, COMBINING DIAERESIS -> LATIN CAPITAL LETTER U WITH DIAERESIS)
- Normal Form D (NFD): canonical decomposition
- considers canonical order of multiple combining characters
- Normal Form C (NFC): canonical decomposition, followed by canonical composition


## Compatibility Characters

- encoded in Unicode solely for compatibility with existing standards
- non-compatibility encodings already exist
- compatibility decomposition
- e.g. U+212B (ANGSTROM SIGN) -> U+00C5 (LATIN CAPITAL LETTER A WITH RING ABOVE)
- e.g. U+0133 (LATIN SMALL LIGATURE IJ) -> U+0069, U+006A
- Normal Form KD (NFKD): compatibility-decompose, then apply NFD
- Normal Form KC (NFKC): compatibility-decompose, then apply NFC


## Types of Code Points

- Graphic
-Format (e.g. paragraph separator)
-Control: usage defined outside Unicode
-Private-use: usage defined outside Unicode
-Surrogate: reserved for use with UTF-16
-Non-character: reserved for internal use, restricted interchange
-Reserved: reserved for future assignment


## Allocation of Code Points

-Structured in planes $\left(2^{16}\right)$, rows $\left(2^{8}\right)$, cells (1)

- Plane 0: Basic Multilingual Plane (BMP)
- Plane 1: Supplementary Multilingual Plane
- Plane 2: CJK Unified Ideographs Extension B
- Plane 14: Tags
- Plane 15, 16: Private Use Areas
-BMP is further subdivided into blocks:
- Alphabets, extension symbols, CJK Ideographs, Hangul, Surrogates, Private Use Area, Compatibility characters


## Encoding Forms

- Unicode supports code units of 8,16 , and 32 bits
- UTF-32: made code point 1:1 to code unit
- encoding schemes need to specify byte order (e.g. UTF-32BE) or Byte Order Mark (BOM, U+FEFF)
-UTF-16: 16-bit code units
- characters < 65536 map 1:1
- other characters use surrogate pair (two code units)
- CES needs to specify byte order or use BOM
- UTF-8: 8-bit code units
- variable length (1..4 bytes), ASCII subset uses 1 byte
- maps 1:1 to CES, optional usage of BOM as "UTF-8 signature"
- null-byte free (except for U+0000)


## Usage of Unicode in XML

-All characters in a document come from Unicode

- usage of unassigned (reserved) characters is wellformed
[84] Letter ::= BaseChar |Ideographic
[85] BaseChar ::= [\#x0041-\#x005A]|[\#x0061-\#x007A]|[\#x00C0-\#x00D6]| [\#x00D8-\#x00F6] | [\#x00F8-\#x00FF] | [\#x0100-\#x0131] ... | [\#xAC00-\#xD7A3]
[86] Ideographic ::= [\#x4E00-\#x9FA5]|\#x3007|[\#x3021-\#x3029]
[87] CombiningChar ::= [\#x0300-\#x0345]|[\#x0360-\#x0361]|[\#x0483-\#x0486]
... | \#x309A
[88] Digit ::= ...
[89] Extender ::= ...
- XML 1.1 replaces explicit lists with ranges that also span yet-unassigned characters


## Encodings of XML Documents

- All XML processors must support UTF-8 and UTF-16
- UTF-16 documents must begin with byte order mark
- Other documents must include XML declaration, and must provide encoding= parameter
- Standard values are "UTF-8", "UTF-16", "ISO-10646-UCS-2", "ISO-10646-UCS-4", "ISO-8859-n", "ISO-2022-JP", "Shift_JIS", "EUC-JP"
- Other CES should use registered IANA names, or start with "x-"
- Higher layers may provide encoding (e.g. HTTP, MIME)
- If no encoding is provided by a higher layer, it is an error if
- the declared encoding differs from the actual one,
- or no encoding is declared, and the document does not start with a BOM, and is not encoded in UTF-8
- It is a fatal error if a document is passed to the processor in an unsupported encoding


## Auto-Detection of Encodings

- non-normative: the parser may or may not implement this algorithm
- Reading four bytes is sufficient
- With BOM:
- 0000 FE FF: UTF-32, big endian (1234)
- FF FE 00 00: UTF-32, little endian (4321)
- 0000 FF FE: UTF-32, unusual byte order
- FE FF 00 00: UTF-32, unusual byte order
- FE FF \#\# \#\#: UTF-16, big endian
- FF FE \#\# \#\#: UTF-16, little endian
- EF BB BF: UTF-8


## Auto-Detection of Encodings (2)

- Without BOM
- 000000 3C: UTF-32BE
- 3C 0000 00: UTF-32LE
- 0000 3C 00, 00 3C 0000 : UTF32, unusual byte order
- 00 3C 00 3F: UTF-16BE
- 3C 00 3F 00: UTF-16LE
- 3C 3F 78 6D: UTF-8, ASCII, ISO-8859, ... (<?xm)
- 4C 6F A7 94: EBCDIC with some code page

