



How can you tell this slide was made after 1991?

Добрый день

مساء الخير

Grüß Gott

你好

안녕하십니까

こんにちは



The pre-1991 view... *the world without Unicode*

In Shift-JIS

Добрый день

→→→→→→→→→→

→→→→→→→→→→

→好

Grus Gott

→→→→→

こんにちは

In Big5

サホネポヨブ ババペラ

→→→→→→→→→→

→→→→→→→→→→

你好

Gr→→ Gott

→→→→→

iii け ㄥ ン 无

In Johab

Добрый день

→→→→→→→→→→

→→→→→→→→→→

→好

Gr→ß Gott

안녕하십니까

こんにちは

In ISO-8859-6

→→→→→ →→→→

مساء الخير

→→ Gr→→ Gott

→→→→→ →→→→→

In KOI8

Добрый день

→→→→→→→→→→

→→→→→→→→→→

Gru→ Gott →→→→→

→→→→→

In ISO-8859-1

→→→→→ →→→→

→→→→→→→→→→

→→→→→→→→→→

Grüß Gott

→→→→→ →→→→→



Outline - The Essentials of Unicode

- What Unicode is and isn't
 - Definitions: script, character, encoding
 - Design principles
- How Unicode changed text processing
- East Asian languages in Unicode
- Arabic in Unicode



Universal Character Set Encoding

- Unicode is the first encoding to cover characters in all scripts in widespread use today
- Created in 1991 by the Unicode Consortium
- An international standard — Unicode = ISO10646 standard



A script is...

- A writing system for a language
- Examples of scripts

Latin: A, a, B, b...Z, z

Cyrillic: А, а, Б, б, В, в,...Я я

Hiragana: あいうえお...ろ

Chinese ideographs: 天氣

- Scripts ≠ Language



A character...

- Is..
 - An abstraction, a letter from a script
- Is not a glyph
 - Glyph = A positional representation of a character
- Is not a ligature
 - Ligature = A form created from combining two characters

Arabic heh هـ

سهل

معه

عنده

ا+ل → لا



An encoding is...

- A mapping of each character in a character set to a numeric value (code point)

- In hexadecimal

漢 字



Japanese (EUC-JP): C1B4 FABB

Japanese (Shift-JIS): BF8A 9A8E

Chinese (ISO-2022-CN): 6947 4773

South Korean (ISO-2022-KR): 7953 6D2E

But that
mapping
changes with
the encoding!...



An encoding is (2)

- Unless you use Unicode
 - Unicode maps a single code point to each character **regardless** of the language.
 - For example, these characters are the same code point in **Chinese, Japanese or Korean text**

漢	字
↓	↓
U+6F22	U+5B57



Unicode design principles - Unicode does...

1. Assign only one codepoint* per character
 - Han Unification
2. Encode characters not glyphs
3. Cover all widely used scripts
4. **Guaranteed*** round tripping (Any encoding to Unicode and back)
5. Provide efficient encoding
6. Deal only with plain text
7. Store characters in logical order (in memory)
8. Well-defined character properties
9. Allow dynamic composition
 - $A + ` = \grave{A}$
10. Have equivalence sequences (composed and uncomposed characters)



Unicode doesn't...

- Deal with display issues (character rendering)
 - Fonts determine style of character

À

漢字

A =U+0041

漢字 =U+6F22 U+5B57

A

漢字

- All glyphs for Arabic HEH are 062A
- Provide a sort (collation) order to please everyone
- These are left to higher level applications to figure out*

هلال

سهل

=U+062A

معه

عنده



What's in Unicode 5.0?

- 99,024 characters
- From the world's languages
 - Modern
 - Historic and Archaic
 - Characters from national and industry encoding standards
 - Math and technical symbols, geometric shapes, International Phonetic Alphabet
- Added scripts include: Coptic, Old Persian, Sumero-Akkadian Cuneiform, Balinese...



Outline - The Essentials of Unicode

- What Unicode is and isn't
- How Unicode changed text processing
 - Software globalization
 - Internationalization example
 - Unicode representations
 - Multilingual natural language processing
- East Asian text processing in Unicode
- Arabic

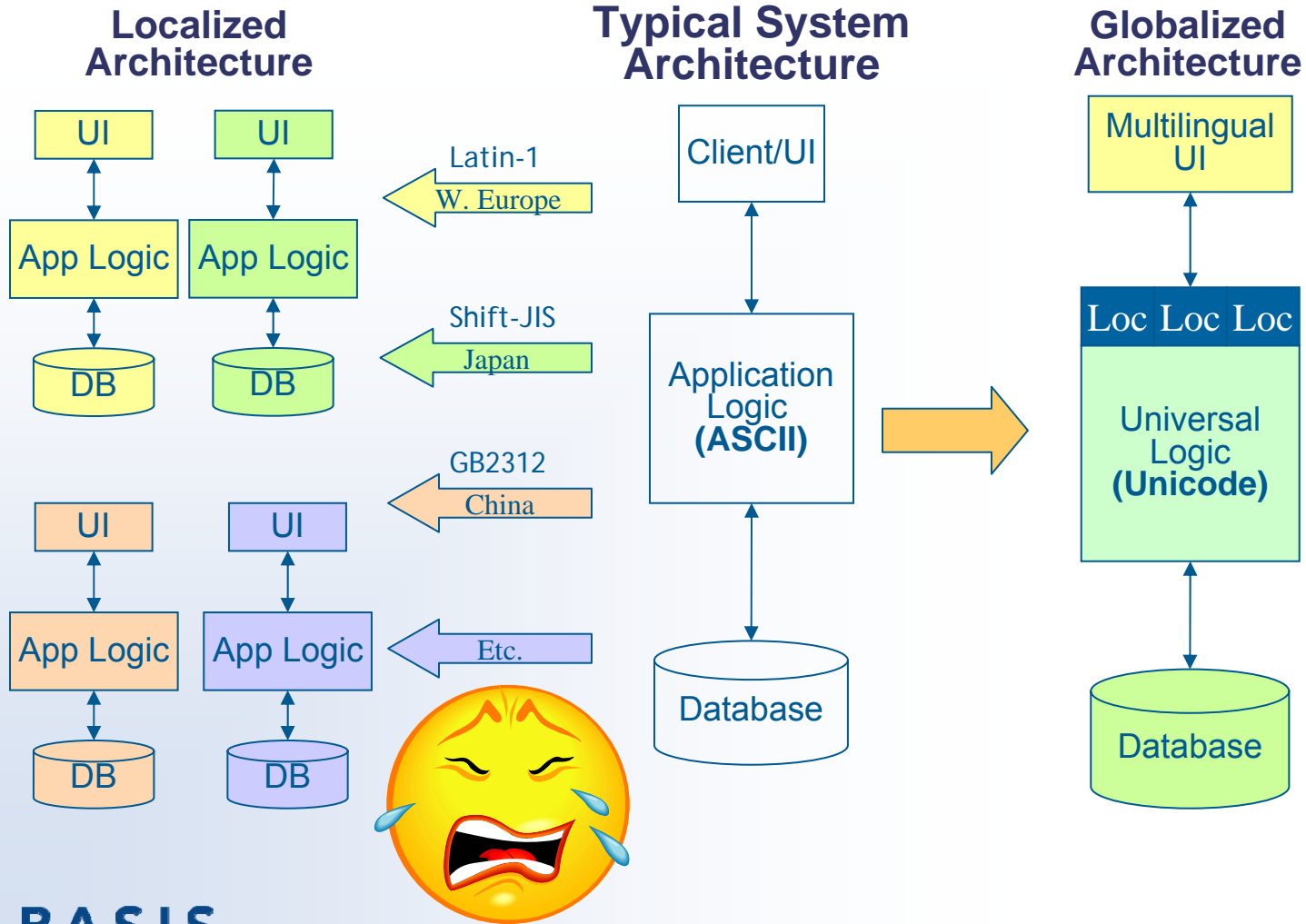


Before Unicode

- Software globalization was painful
- Multilingual text processing was nearly impossible



Globalization before and after Unicode





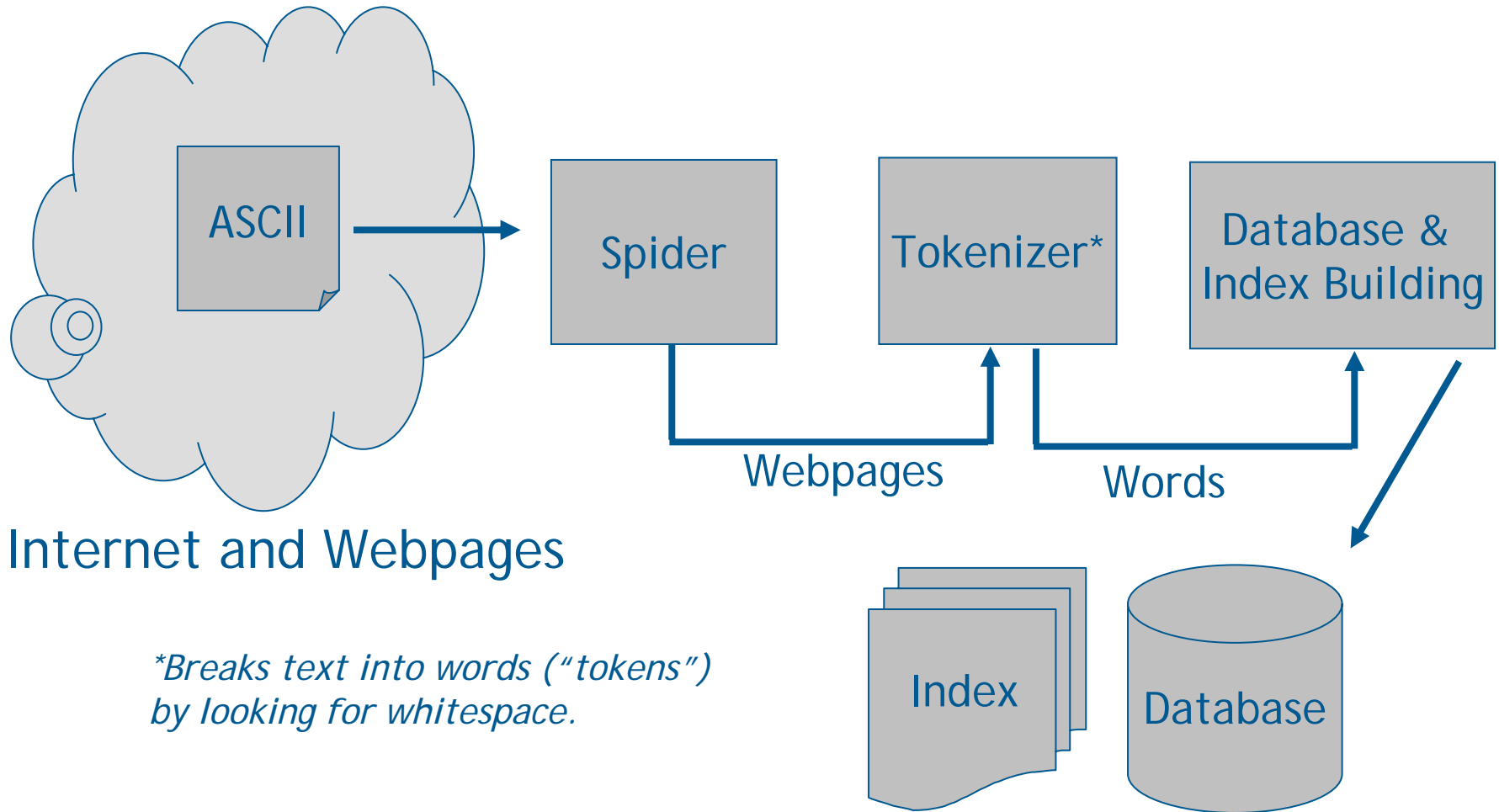
Internationalizing with Unicode

- How do you add multilingual support to your English-only search engine?
- Let's look at a simpler case of just adding Japanese...

■ =ASCII

English Search Engine

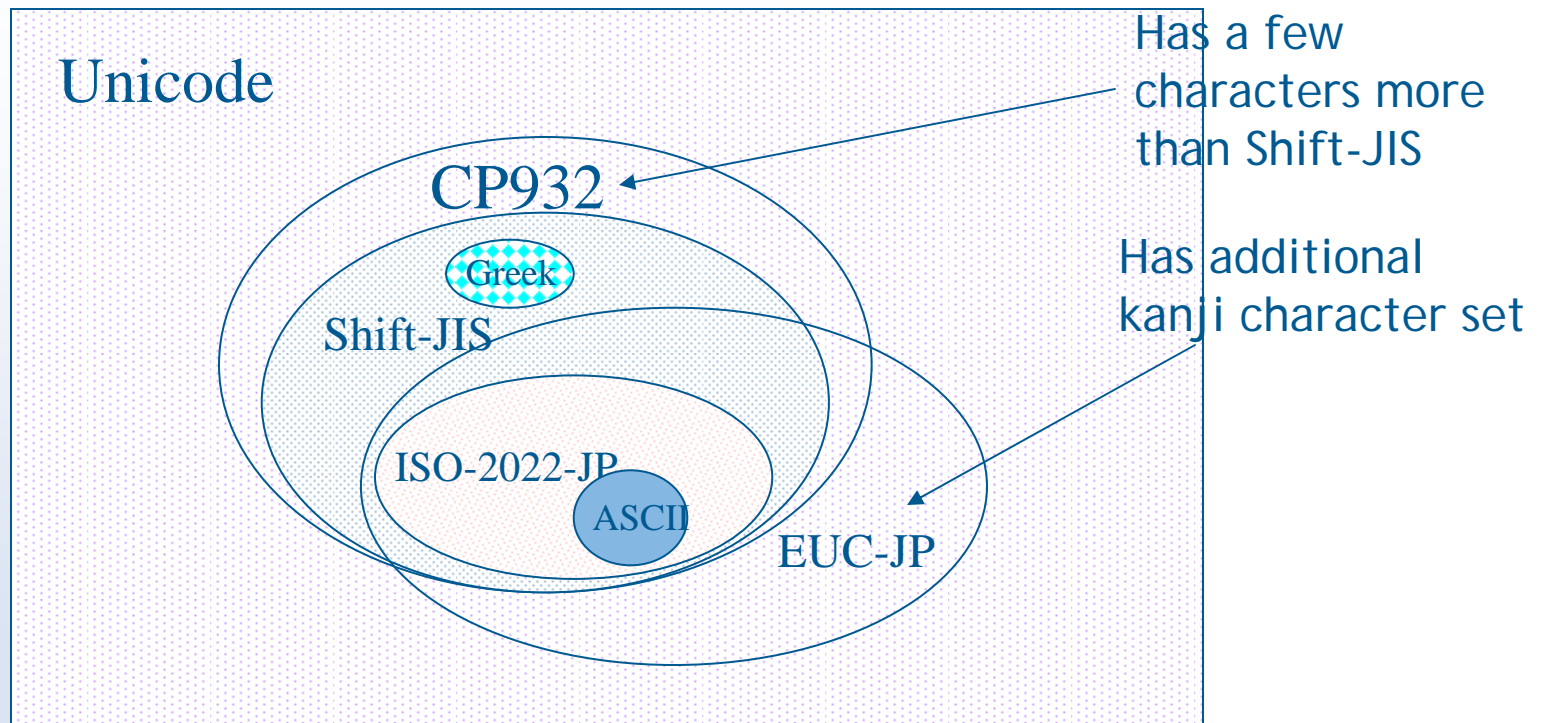
—Database and Index Building Process








Japanese encodings

Japanese character coverage by Japanese encodings:
no single encoding covers all characters

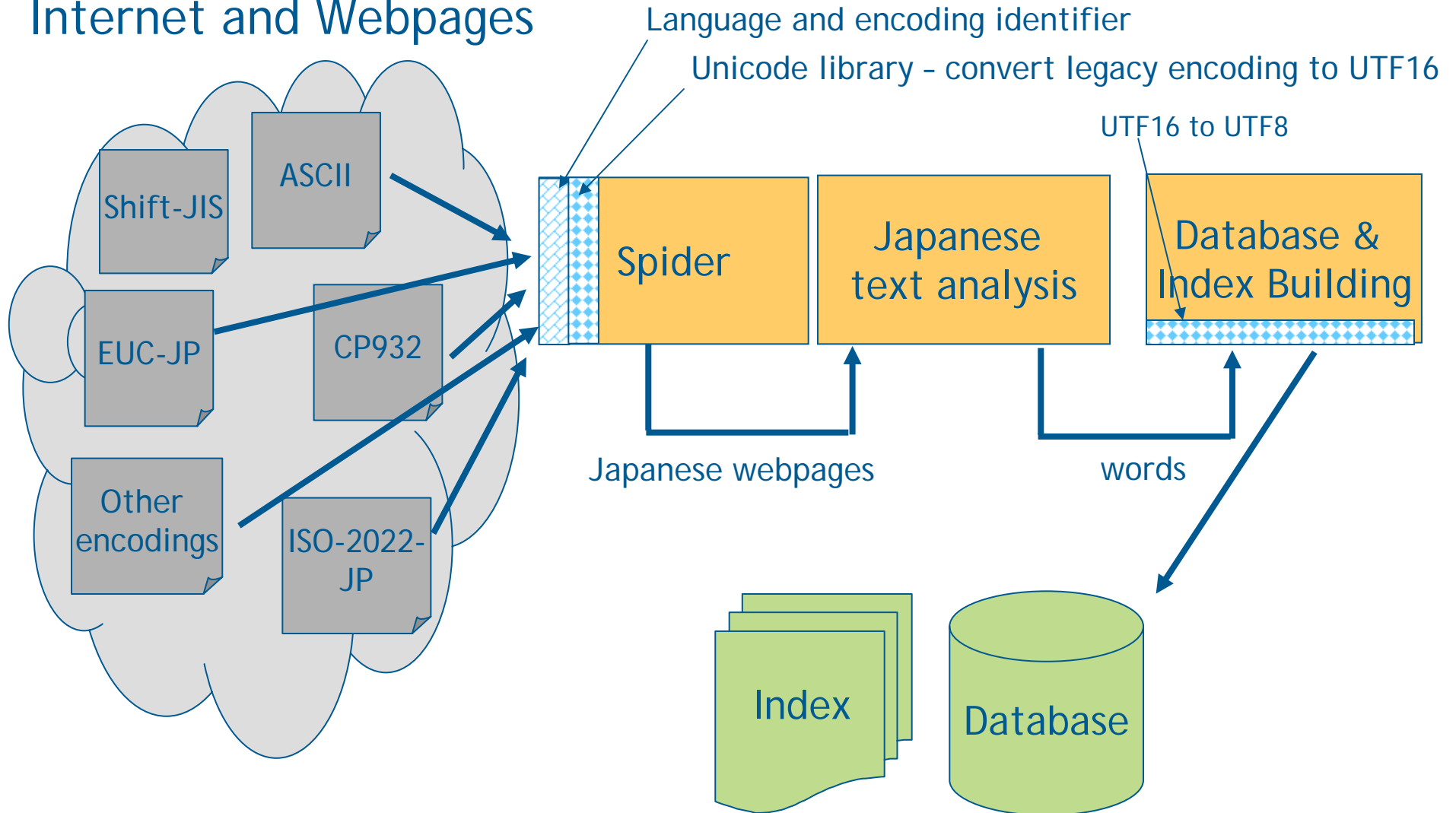


Japanese Search Engine

—Database and Index Building Process

-  =legacy encoding
-  =UTF-8 (a Unicode rep.)
-  =UTF-16 (a Unicode rep.)

Internet and Webpages





Multilingual text processing with Unicode

- Identify language and encoding of input text
- Convert all text to Unicode
- Process all text as Unicode
- Convert text back to legacy on output if needed



Choosing a Unicode representation

Character	Bytes in UTF-16	Bytes in UTF8	Bytes in UTF-32	Code point
Latin letter “A”	00 41	41	00 00 00 41	U+0041
Latin letter “é” with acute	00 E9	c3 a9	00 00 00 E9	U+00E9
あ Hiragana “a”	42 30	e4 88 b0	00 00 42 30	U+4230
ア Katakana “a”	a2 30	ea 88 b0	00 00 a2 30	U+A230
ア Half-width katakana “a”	71 ff	e7 87 bf	00 00 71 ff	U+71FF
亜 Chinese ideograph	9c 4e	e9 b1 8e	00 00 9c 4e	U+9C4E

- Multiple Unicode representations, but same code point per character regardless of byte representation



When to use UTF-8

- ☺ Unicode-enabling ASCII-only code base
 - No embedded null-bytes
 - ASCII text appears as itself
- ☺ ASCII storage
 - ASCII characters have same byte representation as ASCII
 - 1 byte per ASCII character
- ☹ Text processing
 - 1-4 bytes per character
 - This buffer has 36 bytes, how many characters?

Character	Bytes in UTF8
A	41
é	c3 a9
あ	e4 88 b0
ア	ea 88 b0
ア	e7 87 bf
亜	e9 b1 8e



When to use UTF-16

- ☺ Text processing
 - Fixed-width* two-bytes per character
 - 36 bytes=18 characters
- ☹ Storing ASCII
 - ASCII characters are two-bytes ("A" +0041)
- ☹ Unicode-enabling ASCII-only code base
 - Need to widen data paths
 - Embedded null bytes
- ☹ *Surrogate Pairs (4-bytes per character)
U+D800 to U+DFFF
 - Makes UTF-16 multi-byte
 - Beyond Basic Multilingual Plane...

Character	Bytes in UTF-16
A	00 41
é	00 E9
あ	42 30
ア	a2 30
ア	71 ff
亜	9c 4e



Beyond the Basic Multilingual Plane (BMP)

- Characters beyond the Basic Multilingual Plane (BMP)
 - Plane 0 (BMP) = range 0000-FFFF
 - *Encodes UTF-16 characters*
 - Plane 1 = range 1 0000 - 1 FFFF
 - *Rarer Chinese ideographs (CJK extension B)*
 - *Musical notes etc.*
 - Unicode encoding: Total of 17 planes



What of UTF-32?

- ☺ Truly fixed-width, 4-bytes per character
- Storage ☹
 - 4-bytes per character
 - A = 00 00 00 41

2⁸ (1-byte encodings) = space for 256 characters

2¹⁶ (2-byte encoding) = space for 65,536 characters

2³² (4-byte encoding) = space for 4,294,967,296 characters



Advantages of Unicode for NLP

- Able to implement multiple languages in one encoding
- Code reusability
 - Implement multiple languages in one API
 - Characters have defined language-independent properties



Outline - The Essentials of Unicode

- What Unicode is and isn't
- How Unicode changed text processing
- East Asian text processing in Unicode
- Arabic in Unicode



East Asian languages in Unicode

- Chinese
 - China requires support of GB18030 = Unicode support
- Japanese
 - CP932 to Unicode conversion issue
 - Map 0x5c to backslash or yen sign?
 - U+005C (the Reverse Solidus i.e., backslash)
 - U+00A5 (the Yen Sign)
- Korean
 - Unicode algorithmic decomposition to Jamo
 - Thus able to work with individual Jamo in NLP
 - 0x5c Won sign vs. backslash issue in CP949



Outline - The Essentials of Unicode

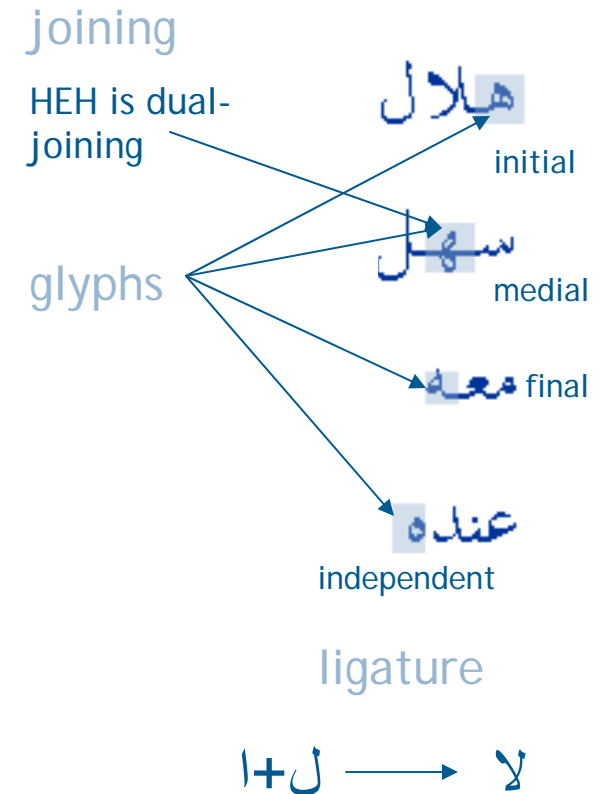
- What Unicode is and isn't
- How Unicode changed text processing
- East Asian languages in Unicode
- Arabic in Unicode
 - Unicode properties: joining, glyphs, ligatures
 - Natural Language Processing with Unicode





Arabic in Unicode

- Provides guidance in character properties for
 - Joining rules
 - Glyph types (character forms)
 - Ligature rules (combined characters)
 - Directionality
 - Bidirectional algorithm
- One code point per *character (heh)*, not per *glyph*, nor per *ligature*
- Arabic Presentation Forms A and B
 - Not in general use
 - Provided for compatibility with legacy encodings





Arabic in Unicode

- Arabic Presentation Forms A
(U+FB50-U+FDFF)
- Arabic Presentation Forms B
(U+FE70-U+FEFF)
 - 1 code point per glyph or ligature
 - Created for compatibility with legacy encodings and fonts
 - Used internally by rendering engines
 - Not in general use
- Basic Arabic block
(U+0600-U+06FF)
 - 1 code point per character
 - Actively used
 - Ligatures and glyphs formed by a higher-level application or rendering engine

ء	تج	نم	نم	كل
FBFB	FC0B	FC1B	FC2B	FC3B
ى	تخ	نم	نم	كم
FBFC	FC0C	FC1C	FC2C	FC3C

ر	ض	غ	ل	و
FEAE	FEBE	FECE	FEDE	FEEE
ز	ظ	غ	ل	ى
FEAF	FEBF	FECF	FEDF	FEEF

ة	ع	ى	ن	و
0629	0639	0649	063B	06C9
ت	نم	ي	ن	ى
062A	063A	064A	063C	06CC



Normalizing Arabic in Unicode

- Normalization becomes necessary*
- Errors/variations from keyboard mapping
 - E.g., User types Arabic with Persian keyboard - yeh mixup
 - Persian and other Arabic script languages use more characters than Arabic
- Stylistic variation
 - Use of yeh vs. alef maksura

Arabic Presentation Form A

ء FBFB	تج FC0B	خم FC1B	غم FC2B	كل FC3B
ي FBFC	تخ FC0C	خم FC1C	غم FC2C	كم FC3C

Farsi yeh

Arabic Presentation Form B

ر FEAE	ض FEBE	غ FECE	ل FEDE	و FEEE
ز FEAF	ظ FEBF	غ FECF	ل FEDF	ي FEFF

Alef maksuura

Basic Arabic block (U+0600 - U+06FF)

ة 0629	ع 0639	ي 0649	ن 063B	و 06C8
ت 062A	غ 063A	ي 064A	ن 068C	ي 06CC

Arabic yeh

*cf. 4:30pm talk - Arabic, Farsi and Urdu Text Normalization by Zina Saadi



Plus and Minuses of Arabic in Unicode

- Single encoding for processing any Arabic text
- Normalization issues
 - Mixing of Non-Arabic characters into Arabic text
 - Use of Arabic letters (with Right-to-Left directionality) for numeric separators (require Left-to-Right directionality)
- Increased control over look of text
 - Using Unicode, Persian writers are no longer forced to join suffixes to stems.



Summary

- Unicode has greatly increased ease of processing multiple languages
 - More unified processing is possible
- Unicode allows applications to span languages and encodings
- Unicode's wide adoption today further helps in Natural Language Processing
 - Normalization
 - Code reusability
 - Especially for Middle Eastern and Asian languages



Unicode resources

- See <http://www.unicode.org> for
 - Unicode Standard Annexes
 - Unicode character databases
 - Unicode minor versions
 - Unicode Technical Reports
 - Unicode Technical Standards
 - Online edition of Unicode 4.0
<http://www.unicode.org/versions/Unicode4.0.0/bookmarks.html>
- *The Unicode Standard, version 5.0* by the Unicode Consortium
 - Addison-Wesley Professional; 5th edition (November 3, 2006)
 - ISBN: 0321480910



Thank you!

- Questions?
- Tina Lieu - tina@basistech.com