CS144 Notes: Content Encoding

- Q: Only “bits” are transmitted over the Internet. How does a browser/application interpret the bits and display them correctly? File extension? Content?

  - MIME (Multi-purpose Internet Mail Extensions)
    - Standard ways to “transmit” multimedia content over the Internet
    - Originally developed for email exchange, but currently used for all Internet data transmission
  - Character encoding/Character set
    - Mapping between numeric numbers and alphabetic characters
    - Many different character encodings

MIME

- MIME types
  - Specified as “type/subtype”. RFC2046 standard.
    - IANA manages the official registry of all media types
  - In HTTP, it is specified in “Content-Type” header
    - E.g., Content-Type: application/pdf
  - Popular types:
    - Text: text/plain, text/html, text/xml, …
    - Image: image/jpeg, image/png, image/gif, …
    - Audio: audio/mpeg, audio/vnd.wave, audio/mp4, …
    - Video: video/mp4, video/avi, video/x-flv, …
    - Application: application/pdf, application:/octet-stream, …
    - Multipart: multipart/mixed, multipart/form-data
    - Q: What multimedia types/format should a browser support?

      - HTML5 is content-type/codec agnostic, no particular format support is required
      - Browsers are “expected” to support "popular" codecs like JPG, PNG for images
      - Big controversy on patent licensing for H.264 in the past due to GIF/MP3 experience

  - Multipart/form-data
    - Make it possible to upload files in a single request
    - Object boundaries are specified in Content-Type header

        - E.g., Content-Type: multipart/form-data; boundary=--EndOfFile
        - Content-Disposition: specifies how to "present" each part. Inline or attachment?
    - Example at http://oak.cs.ucla.edu/cs144/examples/multipart.html

- Q: How do we transmit a binary multimedia data over Internet?
  - Direct binary stream vs as Encoded as a sequence of printable characters
    - Base32, Base64, Quoted-Printable, …
  - Q: Why?

- Q: How does a Web server decide the Content-Type of a file?

UNICODE

- Q: How does a browser translate a sequence bits as characters if it is text?

- Early character encodings before UNICODE

  - ASCII (American Standard Code for Information Interchange)
    - created in 1963. First published as standard in 1967.
    - 7bits. defines codes for 128 characters
    - the basis of most of current encoding of roman characters

  - EBCDIC (Extended Binary Coded Decimal Interchange Code)
    - created in 1963 by IBM for IBM mainframes
    - 8bits. designed to be easy to represent in punch cards
    - still used by some IBM mainframes.

  - ISO-8859-1 (= Latin-1)
    - 8bits. consisting of 191 characters from the Latin script
    - ASCII non-control characters have the same encoding
    - used throughout Western Europe and America.
    - ISO-8859-15, Windows-1252: more characters for French, Estonian,...

- Local/regional encoding
  - * local character codes developed by each country
  - * GB2312 (Simplified Chinese): two encodings - EUC-CN and HZ
    GBK (Superset of GB2312. has both traditional and simplified characters)
    - de facto standard in China
  - * EUC-KR (Korean), ISO-2022-KR (Korean)
  - * DBCS (Double Byte Code Character Set)
    - one or two bytes are used to represent a character
    - used mostly in Asia

- Q: What are the problems of multiple encoding standards?

- code page (= character encoding)
  - a unique number given to a particular character encoding by a system
    - * On Windows
      code page 862: Hebrew, 727: Greek, 949: Korean
  - depending on the region, the os sets global code page for the computer
  - Q: What are the problems of a system-wide code-page setting?

- UNICODE
  - Motivation: a single unique number for every character
    - * independent of language, platform, program
  - originally defined to be 16bit standard
    - * not true any longer. In principle, unlimited number of bits
  - Every character maps to a CODE POINT
    - * A -> U+0041
    - * Hello -> U+0048 U+0065 U+006C U+006C U+006F.
  - a CODE POINT may be encoded as bytes through an encoding scheme
    - * UTF-16
      - the first encoding scheme used for Unicode

- U+0041 -> 00 41 (little endian/big endian)
- Unicode byte order mark: U+FEFF
  - stored at the beginning of a Unicode string
  - gives hints on the endian mode

- used by Windows 2000/XP/Vista, Mac OS X Cocoa, Java, .NET, ...

- Q: Any problem with UTF-16 scheme?

  - space waste

  - legacy applications cannot handle UNICODE string correctly even if the string has only alphabets

    Q: What will C do when it encounters string 00 41?

  - UNICODE applications cannot handle legacy input

  - Q: What will a UNICODE program do for the input 41 42 43 44?

- Q: If I lose one character from 00 41 00 42 00 43 in the middle, what will I get?

  - we need to get the complete string without any error

- UTF-16 did not take off much for internet applications

* UTF-8

- All ASCII characters are mapped to a single byte
  - A: U+0041 -> 41
  - no need to rewrite existing applications to handle English

- Allow easy recovery of the string from error

        - even if a byte is missing, recover from the next character

    - Q: How can we achieve this?

```
0000 -- 007F:  00000000 0zzzzzzz -> 0zzzzzzz
0080 -- 07FF:  00000yyy yyzzzzzz -> 110yyyyy 10zzzzzz
0800 -- FFFF:  xxxxyyyy yyzzzzzz -> 1110xxxx 10yyyyyy 10zzzzzz
```

    - Q: What will be UTF-8 encoding of character A (U+0041)?

    - Q: How can we tell the beginning of a new character from UTF-8 encoding?

        11010111 10111000 11101010 10111101 10110110 01111000

        - Q: How many characters in the above UTF-8 encoding?

        - Q: How to recover if the second byte is lost during transmission?

    - Q: If two strings are of the same length, are their encodings of the same length?

        - variable length encoding vs. fixed-length encoding

* Commonly used for many Web sites and international applications

- UTF-7
  * 7bit encoding. Guarantee the highest bit is always 0.

- Q: How can we use/specify UNICODE?

  - HTTP: Text type character encoding is specified as the “charset” parameter
    - E.g., Content-Type: text/html; charset=UTF-8
    - This can be overridden inside a Web page by including in `<head>`:
      `<meta http-equiv="Content-Type" content="text/html; charset=utf-8">`
  - C/C++:
    - Declare strings as “wchar_t” (wide char) and use wcs functions instead of str functions. (e.g., wcslen instead of strlen). Prefix a string with L like L”Hello”
    - Internally, all unicode string is represented as UCS-2 (~UTF-16) encoding. Particular input/output encoding can be specified using locale() function
  - Java:
    - All char values represented as Unicode characters
    - Input/output encoding can be set for `InputStreamReader/OutputStreamWriter` objects.