CS 5500

Spring 2013

The Power (and Complexity) of Plain Text

Standard formats

JFIF (JPEG File Interchange Format)
ISO/IEC 10918, ISO/IEC FCD 10918-5

PDF (Portable Document Format)
ISO/IEC 32000-1:2008

Standard formats

JFIF (JPEG File Interchange Format)
    ISO/IEC 10918, ISO/IEC FCD 10918-5

PDF (Portable Document Format)
    ISO/IEC 32000-1:2008

Proprietary formats

.doc

.docx

Plain text

US-ASCII

Plain text

US-ASCII

Latin-1
- ISO/IEC 8859-1

Plain text

US-ASCII

Latin-1
    ISO/IEC 8859-1

Latin-9
    ISO/IEC 8859-15

Plain text

US-ASCII

Latin-1
    ISO/IEC 8859-1

Latin-9
    ISO/IEC 8859-15

Unicode

Plain text

US-ASCII

Latin-1
- ISO/IEC 8859-1

Latin-9
- ISO/IEC 8859-15

Unicode
- UTF-8
- UTF-16
- UTF-16BE
- UTF-16LE
- UTF-32
- UTF-32BE
- UTF-32LE

Plain text

US-ASCII

Latin-1
- ISO/IEC 8859-1

Latin-9
- ISO/IEC 8859-15

Unicode
- UTF-8
- UTF-16
- UTF-32

Unicode transformation formats

```
(define c0 #\x00fc)    ; umlaut u
(define c1 #\u)        ; lower case u
(define c2 #\x0308)    ; combining diaresis
(define c3 #\xfb03)    ; ligature ffi
(define chaos "Χαος")  ; Greek

(define s1
  (string-append
   (string c0 c1 c2 #\space c3 #\space)
   chaos))
```

Unicode transformation formats

```
> s1
"üü ﬃ Χαος"

> (string->utf8 s1)
#vu8(195 188 117 204 136 32                    ; üü
     239 172 131 32                            ; ﬃ
     206 167 206 177 206 191 207 130)          ; Χαος

> (string->utf16 s1)
#vu8(0 252 0 117 3 8 0 32                      ; üü
     251 3 0 32                                ; ﬃ
     3 167 3 177 3 191 3 194)                  ; Χαος

> (string->utf32 s1)
#vu8(0 0 0 252 0 0 0 117 0 0 3 8 0 0 0 32
     0 0 251 3 0 0 0 32
     0 0 3 167 0 0 3 177 0 0 3 191 0 0 3 194)
```

Specifying the character set for XML documents

```
<!DOCTYPE html PUBLIC
    "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<title>
Readings for CS 5500: Managing Software Development
</title>
<link type="text/css"
      rel="stylesheet"
      href="cs5500.css" />
```

Unicode equivalence

canonical equivalence

    ü ↔ u followed by a combining diaresis

compatibility equivalence

    the ligature ﬃ → ffi
    the Roman numeral Ⅵ → VI
    ¼ → 1/4

Unicode normalization forms

NFD
: Normalization Form Canonical Decomposition
(decomposed by canonical equivalence)

NFC
: Normalization Form Canonical Composition
(decomposed and then recomposed by canonical equivalence)

NFKD
: Normalization Form Compatibility Decomposition
(decomposed by compatibility equivalence)

NFKC
: Normalization Form Compatibility Composition
(decomposed by compatibility equivalence, then recomposed by canonical equivalence)

Umlaut u

Umlaut u (U+00fc) is canonically equivalent to
u (U+0075) followed by a combining diaresis (U+0308)

```
> (list s1 s2)
("ü" "ü")

> (string-ref s1 0)
#\ü

> (string-ref s2 0)
#\u

> (string-ref s2 1)
#\x308                        ; combining diaresis
```

Umlaut u

Umlaut u (U+00fc) is canonically equivalent to
u (U+0075) followed by a combining diaresis (U+0308)

```
> (define s3
    (string-normalize-nfd s1))

> (string->list s1)
(#\ü)

> (string->list s3)
(#\u #\x308)
```

Umlaut u

Umlaut u (U+00fc) is canonically equivalent to
u (U+0075) followed by a combining diaresis (U+0308)

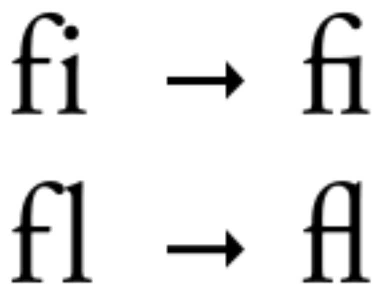
```
> (define s4
    (string-normalize-nfc s3))

> (string->list s4)
(#\ü)

> (map string->list (list s1 s2 s3 s4))

((#\ü)
 (#\u #\x308)
 (#\u #\x308)        ; NFD (for either of the above)
 (#\ü))              ; NFC (for all of the above)
```

Ligatures



from Wikipedia Commons (released into public domain)
http://en.wikipedia.org/wiki/File:Ligature_drawing.svg

Ligatures

Latin small ligature ﬃ (U+fb03) is compatibility equivalent to
`ffi`

```
> (string-length s1)
1

> (display s1)
ﬃ

> (define s2
    (string-normalize-nfd s1))

> (string-length s2)
1

> (display s2)              ; s1 is the same as s2
ﬃ
```

Ligatures

Latin small ligature ﬃ (U+fb03) is compatibility equivalent to `ffi`

```
> (display s2)
ﬃ

> (define s3
    (string-normalize-nfkd s1))

> (string-length s3)
3

> (display s3)
ffi
```

Roman numerals

Roman numeral six (U+2165) is compatibility equivalent to upper case V (U+0056) followed by upper case I (U+0049).

```
> (define c1 #\x2165)

> (define s1 (string c1))

> (define c2 #\V) ; upper case V

> (define s2 "VI")

> (char=? c1 c2)
#f

> (list s1 s2)
("Ⅵ" "VI")
```

Roman numerals

Roman numeral six (U+2165) is compatibility equivalent to upper case V (U+0056) followed by upper case I (U+0049).

```
> (define s3
    (string-normalize-nfd s1))

> (define s4
    (string-normalize-nfkd s1))

> (list s1 s2 s3 s4)
("Ⅵ" "VI" "Ⅵ" "VI")
```

Unicode normalization forms

NFD
  Normalization Form Canonical Decomposition
  (decomposed by canonical equivalence)

NFC
  Normalization Form Canonical Composition
  (decomposed and then recomposed by canonical equivalence)

NFKD
  Normalization Form Compatibility Decomposition
  (decomposed by compatibility equivalence)

NFKC
  Normalization Form Compatibility Composition
  (decomposed by compatibility equivalence, then recomposed by canonical equivalence)

Unicode case mappings

```
> (define s1 "ΧΑΟΣ")

> (define s2
    (string-append s1 s1))

> (define s3
    (string-append s1 " " s1))

> (list s1 s2 s3)
("ΧΑΟΣ"
 "ΧΑΟΣΧΑΟΣ"
 "ΧΑΟΣ ΧΑΟΣ")
```

Unicode case mappings

```
> (list s1 s2 s3)
("ΧΑΟΣ"
 "ΧΑΟΣΧΑΟΣ"
 "ΧΑΟΣ ΧΑΟΣ")

> (map string-titlecase (list s1 s2 s3))
("Χαος"
 "Χαοσχαος"
 "Χαος Χαος")
```

Unicode case mappings

```
> (list s1 s2 s3)
("ΧΑΟΣ"
 "ΧΑΟΣΧΑΟΣ"
 "ΧΑΟΣ ΧΑΟΣ")

> (map string-titlecase (list s1 s2 s3))
("Χαος"
 "Χαοσχαος"
 "Χαος Χαος")
```

Multiple end-of-line conventions

LINE FEED (LF, U+000A)
CARRIAGE RETURN (CR, U+000D)
CARRIAGE RETURN, LINE FEED (CR/LF)
NEXT LINE (NEL, U+0085)
CARRIAGE RETURN, NEXT LINE (CR/NEL)
LINE SEPARATOR (LS, U+2028)

Tab characters have no standard interpretation

Recommendations

When feasible, limit yourself to the ASCII character set.

Use the locale-specific default for files.

Use your programming language's preferred representation (if any) for computational purposes.

Let your programming language's i/o libraries deal with the conversion between external and internal representations.

Let your programming language's i/o libraries deal with the multiple end-of-line conventions.

Don't use tab characters.