Web Data Collection with R

Character Encoding

Peter Meißner / 2016-02-29 – 2016-03-04 / ECPR WSMT

Character Encodings

Character Encodings

Character Encodings

Character Encodings are . . .

- are like family . . .
- . . . some of them you do not like but cannot avoid . . .
- . . . something we will struggle with but have cope anyways

The best thing is . . .

- R has them all

The worst thing is . . .

- R has them all

Character Encodings

- computers store everything as 0s and 1s (bits)
- in cs there are differing layers of abstraction
- one bit of information is called bit
- bits are quite uninformative as they only ave two states
- so they are are grouped into bytes (8 bits)
- one byte can have 256 different values (2^8)
- so it can store numbers 0 to 255 or 1 to 256 or . . . -127 to 128
- or it can map to characters e.g. ASCII (abcABC.:-_,;#'+*~|<>¡‘§$%&/()=?}][{}^°, . . . ")
- ASCII is a character set - the set of characters you want to be able to store - even 7 Bits would suffice to store it

Character Encodings

- for larger character sets than ASCII (ä ö ü é è . . .) on needs to get clever since one byte does not suffice to map all characters to 0s and 1s
- unfortunate people got clever in differing ways
  1. using more than one byte to map more characters ('wide' characters, UTF-16, USC-2, Windows OSs)
  2. using one or more bytes and using the first byte to encode how manies are used ('multi-byte characters', UTF-8, Unix based OSs)
- otherwise we would not have to talk about character sets and character encodings

Character Encodings

```
rawToBits(as.raw(62:66)) # as bits
```

```
##  [1] 00 01 01 01 01 01 00 00 01 01 01 01 01 01 00 00 00
## [24] 00 01 00 00 00 00 00 01 00 00 01 00 00 00 00 01 00
```

```
as.raw(62:66) # bytes as hexa-decimal
```

```
## [1] 3e 3f 40 41 42
```

```
as.numeric(as.raw(62:66)) # as numbers
```

```
## [1] 62 63 64 65 66
```

```
rawToChar(as.raw(62:66)) # bytes as chararcters
```

```
## [1] ">?@AB"
```

A character set problem

```
text             <- rawToChar(as.raw(228))
Encoding(text) <- "UTF-8"
text
```

```
## [1] "\xe4"
```

```
Encoding(text) <- "latin1"
text
```

```
## [1] "ä"
```

Results differ because for latin1 character 228 is know but not for UTF-8

An encoding problem

Of cause UTF-8 knows how to encode “ä” . . .

```
text <- "ä"
charToRaw(text)
```

```
## [1] c3 a4
```

```
Encoding(text) <- "latin1"
text
```

```
## [1] "Ã¤"
```

. . . but here the results differ because “UTF-8” has another system translating characters to bytes. In latin1 the two bytes are interpreted as two characters.

Which default encoding does your R use

```
Sys.getlocale()
```

```
## [1] "LC_CTYPE=de_DE.UTF-8;LC_NUMERIC=C;LC_TIME=de_DE.UTF
```

```
# if yor locale is something other than UTF-8,
# switch 'latin1' and 'UTF-8' and you shall be good to go
```

Changing interpretation of bytes

```
text <- "Små grodorna, små grodorna är lustiga att se."
Encoding(text) <- "UTF-8"
text
```

```
## [1] "Små grodorna, små grodorna är lustiga att se."
```

Changing interpretation of bytes

```
text <- "Små grodorna, små grodorna är lustiga att se."
Encoding(text) <- "latin1"
text
```

```
## [1] "SmÃ¥ grodorna, smÃ¥ grodorna Ã¤r lustiga att se."
```

Changing bytes and interpretation

```
text <- "Små grodorna, små grodorna är lustiga att se."
text <- iconv(text, "UTF-8", "latin1")
Encoding(text)
```

```
## [1] "latin1"
```

```
text
```

```
## [1] "Små grodorna, små grodorna är lustiga att se."
```

Noe that all sources might have another encoding than your R default locale!

```
text <- "Små grodorna, små grodorna är lustiga att se."
text <- iconv(text, "UTF-8", "latin1")
writeLines(text, "text_latin1.txt", useBytes = TRUE)
text <- readLines("text_latin1.txt")
Encoding(text)
```

```
## [1] "unknown"
```

```
text
```

```
## [1] "Sm\xe5 grodorna, sm\xe5 grodorna \xe4r lustiga att
```

```
Encoding(text) <- "latin1"
text
```

```
## [1] "Små grodorna, små grodorna är lustiga att se."
```