Literary Data: Some Approaches

Andrew Goldstone
http://www.rci.rutgers.edu/~ag978/litdata

April 2, 2015. XML.

sapply

```
sapply(xs, f, ...)
```

- `xs` can be a list or a vector
- provided `f` yields a single value, returns a vector (not a list)
- whatever's in `...` is passed on to `f` each time

```
lst <- list(c("Charles", "Simic"), c("Edmund", "Spenser"),
            c("Wallace", "Stevens"))
lapply(lst, str_c, collapse=" ")
```

```
[[1]]
[1] "Charles Simic"

[[2]]
[1] "Edmund Spenser"

[[3]]
[1] "Wallace Stevens"
```

```r
sapply(lst, str_c, collapse=" ")
```

```
[1] "Charles Simic"   "Edmund Spenser"
[3] "Wallace Stevens"
```

XML

- plain-text format
- all markup in between `<...>`
- markup structures text in strict hierarchy

grammar

```
XML:  node
node: <tag>node*</tag>
node: <tag/>
node: text
```

```
<teiHeader>
  <fileDesc>
    <titleStmt>
      <title>Lady Audley's Secret, Volume 1</title>
      <author>Braddon, M.E. (Mary Elizabeth) (1837-1915)
      </author>
      ...
    </titleStmt>
    ...
  </fileDesc>
</teiHeader>
```

attributes

```
<tag>:  <tagname attrs*>
<tag/>: <tagname attrs* />
attr:   attrname="attrvalue"
```

```
<head>CHAPTER I.</head>
<head type="sub">LUCY.</head>
```

```
<pb n="6" xml:id="VAB7086-010"/>
```

the rule

What is wrong with

```
<l><sentence>
The apparition of these faces in the crowd;</l>
<l>Petals on a wet, black bough.</sentence></l>
```

?

extras

- comments `<!-- comment -->`
- processing directives: `<? ... ?>`
  - `<?xml version="1.0" encoding="utf-8"?>`
- unparsed: `<![CDATA[...]]>`
- entities: `Toronto: Bell & Cockburn`

The Text Encoding Initiative (TEI)

- defines a set of XML tags and attributes
- text as “ordered hierarchy of content objects”
- *Guidelines* (www.tei-c.org/Guidelines/P5/): only 1664 pages!
- *TEI Lite* (www.tei-c.org/Guidelines/Customization/Lite/): fewer tears

getting to grips in R

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE ETS SYSTEM "http://www.lib.umich.edu/tcp/docs/code/eeb
<ETS>
<TEMPHEAD>
<REVDESCR>
...
```

```
library("XML")
congreve <- xmlParse("tei-sample/ecco/K001985.000.xml")
congreve_root <- xmlRoot(congreve) # top of the hierarchy
xmlName(congreve_root)
```

```
[1] "ETS"
```

more design principles: encapsulation

```r
class(congreve)
```

```
[1] "XMLInternalDocument"
[2] "XMLAbstractDocument"
```

```r
class(congreve_root) # hmm
```

```
[1] "XMLInternalElementNode"
[2] "XMLInternalNode"
[3] "XMLAbstractNode"
```

more design principles: polymorphism

```
congreve_root[[1]]
```

```
<TEMPHEAD>
  <REVDESCR>
    <CHANGE>
      <DATE>2008-09-19</DATE>
      <RESPSTMT>
        <NAME>Simon Charles</NAME>
        <RESP>MURP</RESP>
      </RESPSTMT>
      <ITEM>Proofed and reviewed</ITEM>
    </CHANGE>
  </REVDESCR>
</TEMPHEAD>
```

traversing the tree

```r
kids <- xmlChildren(congreve_root)  # next level down
class(congreve_root)                # oookay
```

```
[1] "XMLInternalElementNode"
[2] "XMLInternalNode"
[3] "XMLAbstractNode"
```

```r
sapply(kids, xmlName)
```

```
  TEMPHEAD       EEBO
"TEMPHEAD"     "EEBO"
```

```r
congreve_root[["TEMPHEAD"]][[
    "REVDESCR"]][[
        "CHANGE"]][[
            "RESPSTMT"]][[
                "NAME"]]
```

```
<NAME>Simon Charles</NAME>
```

extracting node sets

- XPath: like file paths!

```
getNodeSet(congreve_root,
           "/ETS/TEMPHEAD/REVDESCR/CHANGE/RESPSTMT/NAME")
```

```
[[1]]
<NAME>Simon Charles</NAME>

attr(,"class")
[1] "XMLNodeSet"
```

- but shorter!

```
getNodeSet(congreve_root, "/ETS//NAME")
```

```
[[1]]
<NAME>Simon Charles</NAME>

attr(,"class")
[1] "XMLNodeSet"
```

and…vectorized

```
speakers <- getNodeSet(congreve_root, "//SPEAKER")
length(speakers)
```

```
[1] 1162
```

```
class(speakers)
```

```
[1] "XMLNodeSet"
```

Could do:

```
spkr_names <- character()
for (i in seq_along(speakers)) {
  spkr_names[i] <- speakers[[i]]    # sloooow
}
```

```r
spkr_names <- xmlSApply(speakers, xmlValue)
head(spkr_names)
```

```
[1] "Val."  "Jere." "Val."  "Jere." "Val."
[6] "Jere."
```

```r
sort(table(spkr_names), decreasing=T)[1:5]
```

```
spkr_names
    Scan.      Val.     Tatt. Sir Samp.
      171       165       133       113
     Ang.
       97
```

attributes

```r
divs <- getNodeSet(congreve_root, "//DIV1")
xmlGetAttr(divs[[1]], "TYPE")
```

```
[1] "title page"
```

```r
xmlSApply(divs, xmlGetAttr, "TYPE")
```

```
 [1] "title page"        "dedication"
 [3] "prologue"          "prologue"
 [5] "epilogue"          "dramatis personae"
 [7] "act"               "act"
 [9] "act"               "act"
[11] "act"
```

```r
# An XPath can match attributes:
acts <- getNodeSet(congreve_root, '//DIV1[@TYPE="act"]')
length(acts)
```

```
[1] 5
```

namespaces: a pain in your neck

```r
crisis <- xmlParse("tei-sample/mjp/Crisis130_22.2.tei.xml")
all_divs <- getNodeSet(crisis, "//div")
length(all_divs) # what.
```

```
[1] 0
```

```r
xmlNamespaceDefinitions(crisis)[[1]][c("id", "uri")]
```

```
$id
[1] ""

$uri
[1] "http://www.tei-c.org/ns/1.0"
```

```r
# "def" is arbitrary here
all_divs <- getNodeSet(crisis, "//def:div",
    namespaces=c(def="http://www.tei-c.org/ns/1.0"))
xmlSApply(all_divs, xmlGetAttr, "type") %>% table()
```

```
.
advertisements       articles
             4              6
         front         images
             1              2
         issue         poetry
             1              1
```

```r
ns <- c(def="http://www.tei-c.org/ns/1.0")
poem <- getNodeSet(crisis, "//def:div[@type='poetry']",
                   namespaces=ns)[[1]]
poem
```

```
<div type="poetry">
  <ab>THE NEGRO SPEAKS OF RIVERS </ab>
  <ab>LANGSTON HUGHES </ab>
  <ab>I'VE known rivers: I've known rivers ancient as the world and olc
  <ab>My soul has grown deep like the rivers. </ab>
  <ab>I bathed in the Euphrates when dawns were young. </ab>
  <ab>I built my hut near the Congo and it lulled me to sleep. </ab>
  <ab>I looked upon the Nile and raised the pyramids above it. </ab>
  <ab>I heard the singing of the Mississippi when Abe Lincoln went down
  <ab>I've known rivers; Ancient, dusky rivers. </ab>
  <ab>My soul has grown deep like the rivers. </ab>
</div>
```

more with attributes

```r
# h/t Nicole
fe <- xmlParse("fair-em/A21328-sheriko.xml")
speeches <- getNodeSet(fe, "//def:sp", namespaces=ns)
speeches[[1]]
```

```
<sp who="Lubeck">
  <speaker>Marques.</speaker>
  <l met="100">WHat meanes faire Britaines mighty Conqueror</l>
  <l met="100">So suddenly to cast away his staffe?</l>
  <l met="100">And all in passion, to forsake the tylt.</l>
</sp>
```

- How can we tally proportions of metrical deviations by speaker?

```r
# not fast
meters <- xmlApply(speeches, getNodeSet, "def:l",
                   namespaces=ns) %>%
    lapply(xmlSApply, xmlGetAttr, "met",
           default="<missing>")

ll <- vector("list", length(speeches))
for (j in seq_along(speeches)) {
    s <- speeches[[j]]
    if (length(meters[[j]] > 0)) {
        ll[[j]] <- data_frame(sp=xmlGetAttr(s, "who",
                                default="<missing>"),
                              meter=meters[[j]])
    }
}
spkrs_meter <- do.call(rbind, ll)
```

```
metrical_devs <- spkrs_meter %>% group_by(sp) %>%
    summarize(total_lines=n(),
              deviations=sum(meter != "100")) %>%
    mutate(dev_pct=deviations / total_lines * 100) %>%
    arrange(desc(dev_pct))
```

```
metrical_devs %>%
    print_tabular()
```

sp	total_lines	deviations	dev_pct
Manuile	1	1	100
Elner	16	15	94
Citizen	26	21	81
Messenger	10	8	80
Trotter	45	36	80
<missing>	4	3	75
Rosilio	4	3	75
Ambassador	10	7	70
Mariana	85	52	61
Valingford	125	71	57
Em	189	104	55
Goddard	118	57	48
Demarch	34	15	44
Manvile	93	38	41
Blanch	33	13	39
Lubeck	124	46	37
Soldier	11	4	36
William	246	80	33
Zweno	118	34	29
Mountney	109	28	26
VWilliam	6	1	17
Dirot	5	0	0
Miller	2	0	0
William	2	0	0

html

- really just like XML
- except when it isn't
- (homework)