Univerza v Ljubljani, FMF
finančna matematika

Programiranje v R-ju

Označevanje

Vladimir Batagelj
FMF, matematika

različica: 14. januar 2014 / 09 : 17

Kazalo

Označevanje – SGML

En od pomembnejših pojmov pri delu s podatki postaja *označevanje*: dele besedila oklenemo z značkami, ki določajo, kaj je dani del besedila in kako naj bo oblikovan.

Čeprav vsi opisi oblikovanega besedila temeljijo na označevanju, predstavlja prelomnico SGML (Standard Generalized Markup Language) sprejet leta 1986. SGML je sestav za pripravo definicij označevalnih jezikov: HTML – HyperText Markup Language, ISO 12083 – Electronic Manuscript Standard, TEI – Text Encoding Initiative, CALS/ JTA – Computer-aided Acquisition and Logistic Support, NITF – News Industry Text Format, DDI – Data Documentation Initiative, CML – Chemical Markup Language,

SGML je namenjen opisu *zgradbe* podatkov, ki je ločen (pravokoten na) od opisa *oblike* (prikaza). Oblika je določena s *slogi*. V HTMLju so to prekrivni slogi CSS W3C/CSS, W3 schools/CSS, MIRK'04.

Za podporo razvoja rešitev je bilo razvitih več orodij Clark.

XML

Z razvojem spleta se je pojavila potreba, da lahko uporabnik razširi HTML s svojimi oznakami. Prirejena izpeljanka SGMLja je XML – Extensible Markup Language. XML omogoča hranjenje, izmenjavo in lažjo obdelavo podatkov.

XML je ohranil pomembnejše zmogljivosti in glavne značilnosti SGMLja. Predvsem je z odpravo 'potuh' sestav precej poenostavil.

XML se vse bolj uveljavlja tudi pri uporabah informacijske tehnologije v analizi podatkov. V okviru projekta Omega je bil izdelan paket XML za podporo dela z XML v R-ju.

Značke

Kakor v HTMLju se značke pojavljajo v parih **<***ime***>** in **</***ime***>** – *oklepajne* značke. Tudi *samostojne* značke so v bistvu oklepajne – imajo obliko **<***ime***></***ime***>** ali okrajšano **<***ime* **/>**.

Del opisa, ki ga nek par značk oklepa, je *vsebina* te značke. Ta je lahko *prazna* (za samostojne znacke), *enostavna* (niz znakov, ki ne vsebuje drugih značk) ali pa je *sestavljena*.

V imenih značk se velikost črk upošteva. XML podpira *samoopisnost* imen. *Ime*na značk so nizi znakov, ki ne vsebujejo presledkov ali dvopičij in se ne začnejo s številko, ločilom ali podnizom **xml** (**XML**, **Xml**, ...).

Značka ima lahko lastnosti **<***ime* l_1**="**v_1**"** l_2**="**v_2**"** ... l_k**="**v_k**" >** . Vrednosti lastnosti so v navednicah **"** ali **'** .

Večkratni presledki se ne skrčijo v enega; nova vrsta je predstavljena z znakom **LF**. *Pojasnila* so kot v SGML obdana z **<!--** ... **-->** .

Dobro oblikovani opisi

Opis v nekem označevalnem jeziku *jezik* nad XML je *dobro oblikovan* (well formed), če

- vsak opis oklepa *glavna* (korenska) značka **<***jezik***>** ;
- v opisu podatkov so ta vsebini poljubnih dveh značk ali ločeni ali pa je ena vsebovana v drugi (*gnezdenje*) – zaporedje
  <*a***>** . . . **<***b***>** . . . **</***a***>** . . . **</***b***>** ni dovoljeno;

Dobro oblikovani opis lahko predstavimo kot drevo – DOM (Document Object Model).

Dobro oblikovani opisi

Podatke lahko v opisu predstavimo kot vsebino ali vrednost neke lastnosti. Priporočilo: prave podatke predstavimo kot vsebino; lastnosti povedo podatke o podatkih.

Novejše izdaje spletnih pregledovalnikov Internet Explorer (vsaj 5.5) in Mozilla/ Netscape (vsaj 7) omogočajo preverjanje dobre oblikovanosti opisov in ličen (določen s prekrivnimi slogi) prikaz dobro oblikovanih opisov.

S slogi lahko določimo prikaz dobro oblikovanih opisov. CD.xml, CD.css, CD.css+xml.

Primer: zbirka knjig - opis podatkov v XML

```
<?xml version="1.0" encoding="Windows-1250" ?>
<?xml-stylesheet type="text/css" href="zbirka.css" ?>
<!-- 3. december 2002 -->
<zbirka>
 <knjiga jezik="slo" vezava="trd">
  <avtor>Janez Novak</avtor>
  <naslov>Gojenje glist</naslov>
  <zalozba>DZS</zalozba>
  <kraj>Ljubljana</kraj>
  <leto>1995</leto>
  <opomba>Zelo zanimivo</opomba>
 </knjiga>
 <knjiga jezik="slo" vezava="meh">
   <avtor>Peter Škafar</avtor>
   <naslov>Razmišljanja</naslov>
   <zalozba>MK</zalozba>
   <kraj>Ljubljana</kraj>
   <leto>2001</leto>
   <opomba>Za vztrajne.</opomba>
 </knjiga>
</zbirka>
```

Primer: zbirka knjig - prekrivni slogi

```
zbirka { font-size: 20pt;
   font-family: Courier New;
   background-color: Wheat;
   width: 400;}
knjiga { display: block;
   margin-bottom: 30pt;
   margin-left: 0;
   background-color: Cornsilk;}
avtor { font-family: Arial;}
naslov { font-family: Impact;
   display: block;
   color: Maroon;
   margin-left: 20pt;}
zalozba { display: block;
   margin-left: 20pt;}
kraj { margin-left: 20pt;}
leto { }
opomba { display: block;
   margin-left: 20pt;
   color: DarkGreen;
   font-size: 12pt;}
```

Janez Novak
Gojenje glist
DZS
Ljubljana 1995
Zelo zanimivo

Peter Škafar
Razmišljanja
MK
Ljubljana 2001
Za vztrajne.

Pravilni opisi

Zaenkrat še nismo povedali, kateri opisi so 'slovnično' pravilni – v kakšnih medsebojnih odnosih so lahko posamezne značke, katere lastnosti imajo, ...? Slovnico označevalnega jezika določimo z DTD (Document Type Definition), s katerim povemo, kako so posamezne značke med seboj povezane. Opisi, ki zadoščajo dani slovnici so *pravilni* (valid).

Slovnico lahko opišemo tudi s shemami XML Schema, ki same temeljijo na XMLju.

Za naše namene zadostujejo dobro oblikovani opisi.

XML in R

Obstaja več pristopov k obdelavi opisov XML:

- DOM (Document Object Model)
- SAX (Simple API for XML)
- StAX (Streaming API for XML)
- VTD (Virtual Token Descriptor)

V R-ju sta na voljo DOM in SAX. Namestiti moramo paket XML – paket vsebuje knjižnico libxml2, ki podpira DOM, SAX in XPath.

Najpogosteje se uporablja DOM. Ta zgradi v pomnilniku drevo opisa. S pregledovanjem in spreminjanjem tega drevesa lahko podatke v opisu obdelamo.

Pristop DOM odpove, ko je podatkov veliko – ne moremo jih shraniti v pomnilnik. Tedaj uporabimo SAX. Ta temelji na dogodkih - pojavitvah posameznih značk pri prehodu skozi opis. Vsakemu dogodku je potrebno prirediti ustrezne dejavnosti.

. . . XML in R

V knjižnici XML je veliko različnih funkcij za delo z opisi XML. Osnovne stvari lahko opravimo z naslednjimi:

```
xmlParse(file)    - ustvari drevo opisa z datoteke
htmlParse(file)   - podobno, za spletne strani

xmlName(t)        - ime točke
xmlSize(t)        - velikost točke
t[[i]]            - i-ti otrok točke
xmlChildren(t)    - otroci točke; t[[i]] ali  t[['ime']]
xmlAttrs(t)       - lastnosti točke
xmlGetAttr(t,a)   - vrednost lastnosti a
xmlValue(t)       - vrednost točke (besedilo)
xmlParent(t)      - prednik točke

newXMLNode(z,v)   - nova točka z značko z in vrednostjo v

saveXML(t,file)   - predelaj drevo v niz in shrani na datoteko

xmlSApply(t,f)    =  sapply(xmlChildren(t),f)
xmlApply(t,f)     =  apply(xmlChildren(t),f)
```

. . . XML in R

`xmlParse` in `htmlParse` imata vrsto inkarnacij, ki ustrezajo različnim izbiram (pribitjem) njunih parametrov. Najpomembnejši med njimi je `useInternalNodes`.

Če je `useInternalNodes` enak `FALSE` (npr. v `xmlTreeParse`), razčlenjevalnik ustvari drevesno predstavitev opisa v R-ju kot gnezdene sezname potomcev. Natančneje: razčlenitev je sestavljena iz dveh polj `doc` in `dtd`. Polje `doc` naprej vsebuje podpolja `file`, `version` in `children`. Polje `children` vsebuje seznam členov vrste `XMLNode`. Ti so sestavljeni iz štirih polj: `name`, `attributes`, `children` in `value`.

Če je `useInternalNodes` enak `TRUE` (npr. v `xmlParse`), razčlenjevalnik ustvari notranjo kazalčno predstavitev drevesa opisa. Ta predstavitev je učinkovitejša in omogoča tudi doseganje prednikov. Uporabljamo jo z `XPATH`.

. . . XML in R

```
> d <- newXMLNode("A")
> sapply(c("X","Y","Z","X","Y"),newXMLNode,parent=d)
> d
<A>
  <X/>
  <Y/>
  <Z/>
  <X/>
  <Y/>
</A>
> d[[3]]
<Z/>
> xmlAttrs(d[[3]])["P"] <- "bla"
> newXMLNode("B",parent=d[[3]])
> d[[3]][[1]][[1]] <- "besedilo"
> d
<A>
  <X/>
  <Y/>
  <Z P="bla">
    <B>besedilo</B>
  </Z>
  <X/>
  <Y/>
</A>
```

. . . XML in R / pristop xmlOutputDOM (ali xmlTree)

addTag, closeTag, addNode, value, addPI, addComment, addCData, . . .

```
> d <- xmlOutputDOM("naslovi")
> d$addTag("naslov",close=FALSE)
>  d$addTag("oseba",close=FALSE)
>   d$addTag("priimek","Batagelj")
>   d$addTag("ime","Vladimir")
>  d$closeTag() # oseba
>  d$addTag("posta",close=FALSE,attrs=c(kje="sluzba"))
>   d$addTag("ustanova","FMF, matematika")
>   d$addTag("ulica","Jadranska 19")
>   d$addTag("pst","1000")
>   d$addTag("kraj","Ljubljana")
>   d$addTag("drzava","Slovenija")
>  d$closeTag() # posta
> d$closeTag() # naslov
> print(d$value())
<naslovi>
 <naslov>
  <oseba>
   <priimek>Batagelj</priimek>
   <ime>Vladimir</ime>
  </oseba>
  <posta kje="sluzba">
   <ustanova>FMF, matematika</ustanova>
   <ulica>Jadranska 19</ulica>
   <pst>1000</pst>
   <kraj>Ljubljana</kraj>
   <drzava>Slovenija</drzava>
  </posta>
 </naslov>
</naslovi>
```

. . . XML in R / gnezdeni seznami potomcev

```
> setwd("C:/Users/Batagelj/work/R/XML/test")
> library(XML)
> doc <- xmlTreeParse('zbirka1.xml')
> doc
> names(doc)
[1] "doc" "dtd"
> names(doc$doc)
[1] "file"     "version"  "children"
> names(doc$doc$children)
[1] "zbirka"
> names(doc$doc$children$zbirka)
  knjiga   knjiga
"knjiga" "knjiga"
> doc$doc$children$zbirka[[1]]
```


XPATH

Z XML je povezan 'jezik' XPATH, ki omogoča izbrati podmnožice podatkov (poddrevesa), ki nas zanimajo. Je precej podoben potem, ki jih poznamo pri datotekah:

```
/p            - od vrha po poti p
//p           - točka z delno potjo p kjerkoli
p[@a]         - točka s potjo p in lastnostjo a
p[@a='v']     - točka s potjo in lastnostjo a z vrednostjo v
p/@a          - vrednost lastnosti a v točki s potjo p

getNodeSet(t,p) - iz drevesa t vrne točke, ki ustrezajo poti p
xpathApply(t,p,f) - uporabi f na vseh točkah s potjo p
```


. . . XML in R / uporabe XPATH

```
> library(XML)
> doc <- xmlParse('zbirka1.xml')
> doc['/zbirka']
> doc['/zbirka/knjiga/avtor']
> doc['//avtor']
> leta <- unlist(xpathApply(doc,'//leto',xmlValue))
> leta
> doc['//knjiga[@vezava="meh"]']
> unlist(doc['//knjiga/@jezik'],use.names=FALSE)
> doc['//knjiga'][[1]][3:5]
> doc['//knjiga'][[1]][c('leto','kraj')]
> names(doc['//knjiga'][[1]])
> class(doc['//knjiga'])
```

Primer: ECB exchange rates

```
> pa <- "http://www.ecb.europa.eu/stats/eurofxref/eurofxref-daily.xml"
> p <- readLines(con <- url(pa)); close(con)
> d <- xmlTreeParse(p)
> names(d)
> d$doc
> d$doc$children$Envelope
> r <- d$doc$children$Envelope['Cube'][[1]]
> r[[1]]
> xmlApply(r[[1]],xmlAttrs)
> cur <- unlist(xmlApply(r[[1]],xmlGetAttr,'currency'),use.names=FALSE)
> rat <- unlist(xmlApply(r[[1]],xmlGetAttr,'rate'),use.names=FALSE)
> cr <- cbind(cur,lapply(rat,as.numeric))
> rownames(cr) <- 1:length(rat)
> colnames(cr) <- c('currency','rate')
```

```
http://www.bsi.si/_data/tecajnice/dtecbs.xml
http://www.ecb.europa.eu/stats/eurofxref/eurofxref-hist.xml
```

Primer: aktivne točke v spletni strani

```
> pa <- "http://vlado.fmf.uni-lj.si/"
> p <- readLines(con <- url(pa)); close(con)
> doc <- htmlParse(p)
> u <- unlist(xpathApply(doc,"//a[@href]",xmlGetAttr,"href"))
```


StatDataML

V R-ju paket StatDataML omogoča zapis R-ovih podatkov(ij) v XML. Podrobno je zapis opisan v priročniku paketa.

Za delo z zapisom sta na voljo funkciji `writeSDML` in `readSDML` – za podrobnosti si oglejte `help`.

```
> library(StatDataML)
> help(writeSDML)

> data(iris)
> writeSDML(iris, file = "iris.sdml")
```


Viri XML

- AirBase – air quality data
- National Weather Service
- Wikipedia database
- Vereenigde geoctrooieerde Oostindische Compagnie (VOC)
- The Open Archives Initiative
- UW XML Data Repository
- The Comparative Toxicogenomics Database (CTD)
- DataLossDB is a research project aimed at documenting known and reported data loss incidents world-wide.
- CiteSeer.PSU
- The DBLP Computer Science Bibliography

Viri podatkov

ECB – Statistical Data Warehouse; National Statistics UK; OECD; World Bank; WTO (World Trade Organization); UNCTAD (The United Nations Conference on Trade and Development); NationMaster; FRED (Federal Reserve Economic Data); European Commission / Economic and Financial Affairs.

GML