による DDBJ のアクセス方法

- 2007-06-18 -

Ei-ji Nakama

nakama at com-one.com

COM-ONE Ltd.

R と XML

- R[6] で XML を扱うには XML[1] パッケージを利用します.
- R の XML[1] パッケージは libxml(`http://xmlsoft.org/`) を使います.
- R の XML の利用例は少ないですが, libxml の利用例が参考になります.
- XML-1.9.0 以降では encoding もサポートしました.
- Omegahat[a]にありましたが, 現在は CRAN にも登録されています.

[a] `http://www.omegahat.org/`

R と XML 例-1

```
> library(XML)
> f <- system.file("exampleData", "mathmlMatrix.xml",
+                  package = "XML")
> doc <- xmlTreeParse(f,useInternalNodes=T)
> # 内部形式は XPATH が使えます
> unlist(xpathApply(doc,"//*",xmlName))

[1] "mrow"  "reln"  "eq"    "apply" "times"
 [ reached getOption("max.print") -- omitted 15 entries ]]

> sapply(getNodeSet(doc, "/mrow/reln/apply/vector"),
+        saveXML)

[1] "<vector> <cn> 1 </cn> <cn> 2 </cn>\n          </vector>"
[2] "<vector> <cn> 2 </cn> <cn> 1 </cn>\n          </vector>"

> free(doc)
```

R と XML 例-2

```
> f <- system.file("exampleData", "mathmlMatrix.xml",
+                  package = "XML")
> doc <- xmlTreeParse(f,useInternalNodes=T)
> # XPATH での値の取りだし
> path <- "/mrow/reln/apply/matrix/matrixrow/cn"
> unlist(xpathApply(doc, path, xmlValue))

[1] " 0 " " 1 " " 1 " " 0 "

> free(doc)

<pointer: (nil)>
attr(,"class")
[1] "XMLInternalDocument"
```

R と XML 例-3

```
> doc<-xmlTreeParse(f,useInternalNodes=T)
> # 内部形式を XML に戻します
> sapply(getNodeSet(doc, "/"), saveXML)

[1] "<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n<!DOCTYPE mrow

> free(doc)

<pointer: (nil)>
attr(,"class")
[1] "XMLInternalDocument"
```

R と XML 例-4

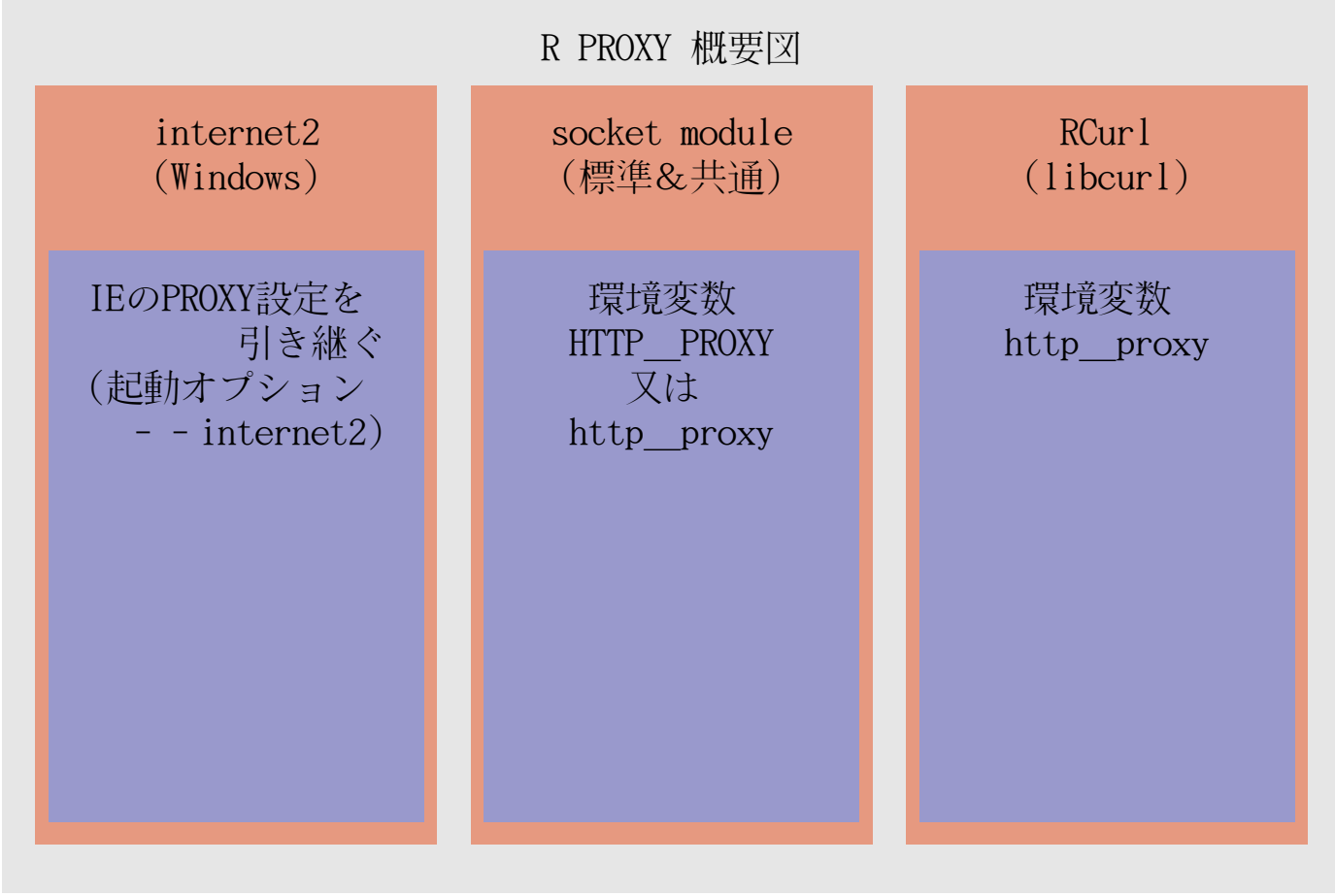
```
> doc <- xmlTreeParse(system.file("exampleData", "mtcars.xml",
+                                 package="XML"))
> # 内部形式では無い場合
> names(xmlRoot(doc))

[1] "variables" "record"    "record"    "record"
[5] "record"
 [ reached getOption("max.print") -- omitted 28 entries ]]

> r <- xmlRoot(doc)
> r[names(r) == "variables"]

$variables
<variables count="11">
 <variable unit="Miles/gallon">mpg</variable>
 <variable>cyl</variable>
 <variable>disp</variable>
```

R と HTTP

- R の RCurl[4] パッケージは libcurl(`http://curl.haxx.se/`) を使います.
- RCurl[4] は標準の socket モジュールが対応していない SSL 等も扱えます.
- XML[1] や SSOAP[5] と連携して高度な HTTP 通信も行えます.
- RCurl[4] は Omegahat から,BioConductor[3] の Extra に供給されています.

R と HTTP PROXY 設定

R で PROXY を設定する場合はモジュール毎に対応が必要です.



例えば、http_proxy=http://prs.ism.ac.jp:3128/

R と SOAP

- R[6] で SOAP を扱うには SSOAP[5] パッケージを利用します.
- SSOAP[5] は XML[1] と RCurl[4] を必要とします.
- SSOAP[5] も Omegahat から, BioConductor[3] の Extra に供給されています.

R と DDBJ

R[6] で DDBJ にアクセスするには, 今までにあげた,XML[1] , RCurl[4],SSOAP[5] が必要です.
DDBJ のチュートリアル[a] [2] に倣って, R 版を作成しました.

[a] http://www.xml.nig.ac.jp/tutorial/index_jp.html

R と DDBJ - エントリーの取得1

```
> # 1. SSOAP のロード
> library(SSOAP)
> # 2. WSDL の指定
> url <- "http://xml.nig.ac.jp/wsdl/GetEntry.wsdl"
> GetEntry <- processWSDL (url)
> iface <- genSOAPClientInterface(def = GetEntry)
> # 3. WEB サービスの呼び出し
> result<-iface@functions$getXML_DDBJEntry("AB000003")
> # 4. 視覚的な確認
> xmlRoot(xmlTreeParse(result))
```

```
<DDBJXML>
 <LOCUS>AB000003</LOCUS>
 <LENGTH>641</LENGTH>
 <MOLECULAR_FORM>linear</MOLECULAR_FORM>
 <DIVISION>PLN</DIVISION>
 <LAST_UPDATE>12-SEP-2002</LAST_UPDATE>
```

RとDDBJ - エントリーの取得2

```
> # ※前の頁の 1-2 は実行済み
> # 3. アクセッション番号 AB000002-AB000005 を一度の実行で取得
> acsession <- c("AB000002","AB000003","AB000004","AB000005")
> # 4. WEB サービスの呼び出し
> result <- sapply(acsession,
+                  iface@functions$getFASTA_DDBJEntry)
> # 5. 視覚的な確認
> print(result)

                                                    AB000002
">AB000002|Rhizoctonia solani genes for 18S rRNA, 5.8S rRNA, "
                                                    AB000003
">AB000003|Rhizoctonia solani genes for 18S rRNA, 5.8S rRNA, "
                                                    AB000004
">AB000004|Rhizoctonia solani genes for 18S rRNA, 5.8S rRNA, "
                                                    AB000005
">AB000005|Rhizoctonia solani genes for 18S rRNA, 5.8S rRNA, "
```

SRS と GetEntry、Blast 連携 1

キーワードとして ‘prion’, Human Division, Molecule‘mRNA’ を指定します.

```
> # SSOAP のロード
> library(SSOAP)
> # WSDL の指定
> SRS <- processWSDL("http://xml.nig.ac.jp/wsdl/SRS.wsdl")
> SRSiface<-genSOAPClientInterface(def=SRS)
> # WEB サービスの呼び出し（条件 & は &amp; にすること）
> result<-SRSiface@functions$searchSimple(
+          "[ddbj-AllText:prion*] &amp;
+           [ddbj-Division:hum]   &amp; [ddbj-Molecule:mrna]")
> print(result)
```

```
[1] "DDBJ:AF187843\nDDBJ:AF187844\nDDBJ:AK090575\nDDBJ:AY008282
```

SRS と GetEntry、Blast 連携 2

その結果からコーディング領域のアミノ酸配列を抜き出します (GetEntry).

```
> # 検索結果を改行で分割し，配列に格納
> id <- unlist(strsplit(result,"\n"))
> url <- "http://xml.nig.ac.jp/wsdl/GetEntry.wsdl"
> getentry<-genSOAPClientInterface(def = processWSDL(url))
> # アクセッション No(substring で DDBJ:をサプレス)
> # sapply でアクセッション No を引数にして
> # DAD エントリーを FASTA 形式で取得
> result<-sapply(substring(id,6),
+                    getentry@functions$getFASTA_DADEntry)
> print(result)

[1] ">AF187843-1|AAG43448.1|148|Homo sapiens doppel protein\nMH
[2] ">AF187844-1|AAG43449.1|176|Homo sapiens prion-like protei
[3] "character(0)"
[4] ">AY008282-1|AAG21693.1|253|Homo sapiens prion protein\nMAN
```

SRS と GetEntry、Blast 連携 3

相同性検索の一つである Blastp を 比較対象 Swiss-plot のデータベースに指定して実行

```
> blastiface<-genSOAPClientInterface(def=processWSDL(
+                       "http://xml.nig.ac.jp/wsdl/Blast.wsdl"))
> # blast サービスを呼び出す
> result<-blastiface@functions$searchParam("blastp",
+                                          "SWISS",
+                                          unlist(result),
+                                          "-m 8")
> print(result)
```

```
[1] "AF187843-1|AAG43448.1|148|Homo\tsp|Q9UKY0|PRND_HUMAN\t99.3
```

SRS と GetEntry、Blast 連携 4

取得したデータを加工する

```
> # 結果を改行コードで分割する
> blastline<-unlist(strsplit(result,"\n"))
> print(blastline[1:3]) # 多いので三つだけ表示

[1] "AF187843-1|AAG43448.1|148|Homo\tsp|Q9UKY0|PRND_HUMAN\t99.3
[2] "AF187843-1|AAG43448.1|148|Homo\tsp|Q9GJY2|PRND_SHEEP\t77.3
[3] "AF187843-1|AAG43448.1|148|Homo\tsp|Q9GK16|PRND_BOVIN\t76.6

> # ID を抜き出す
> id <- as.vector(sapply(blastline,function(x){
+   unlist(strsplit(x,"\\|"))[5]
+ }))
> print(id)

[1] "Q9UKY0" "Q9GJY2" "Q9GK16" "Q9QUG3" "O46501"
 [ reached getOption("max.print") -- omitted 71 entries ]]
```

SRS と GetEntry、Blast 連携 5

アノテーションを GetEntry で取得し, アクセッション番号, プロテイン ID, Swiss-Prot ID, プロテインシンボル, プロテインの定義 を表示する.

```
> # ID から GetUNIPROTEntry 取得
> entry<-sapply(id, getentry@functions$getUNIPROTEntry)
> # 空白を圧縮
> entry<-gsub("\\s{2,}"," ",entry)
> # 改行で分割
> entry<-strsplit(entry,"\n")
> print(entry)

[1] "c(\"ID PRND_HUMAN Reviewed; 176 AA.\", \"AC Q9UKY0; Q9H311
[2] "c(\"ID PRND_SHEEP Reviewed; 178 AA.\", \"AC Q9GJY2; Q9MYU2
[3] "c(\"ID PRND_BOVIN Reviewed; 178 AA.\", \"AC Q9GK16; A2VDR5
[4] "c(\"ID PRND_MOUSE Reviewed; 179 AA.\", \"AC Q9QUG3; Q9QZT5
[5] "c(\"ID PRIO_CANFA Reviewed; 255 AA.\", \"AC Q16501;\", \"H
```

SRS と GetEntry、Blast 連携 6

取得データの編集

```
> # ID の編集
> ID<-sapply(entry, function(x)
+          paste(substring(x[grep("^ID",x)],4),collapse=" "))
> # DE の編集
> DE<-sapply(entry, function(x)
+           paste(substring(x[grep("^DE",x)],4),collapse=" "))
> ENTRY<-cbind(ID,DE)
> print(ENTRY)
```

```
         ID
Q9UKY0 "PRND_HUMAN Reviewed; 176 AA."
Q9GJY2 "PRND_SHEEP Reviewed; 178 AA."
         DE
Q9UKY0 "Prion-like protein doppel precursor (PrPLP) (Prion prot
Q9GJY2 "Prion-like protein doppel precursor (PrPLP)."
 [getOption("max.print") を越えました -- 未 74 行を省略します ]
```

参考文献

[1] Duncan Temple Lang (duncan@wald.ucdavis.edu). *XML: Tools for parsing and generating XML within R and S-Plus.*, 2007. R package version 1.9-0.

[2] Center for Information Biology and DNA Data Bank of Japan. Web サービス チュートリアル. `http://www.xml.nig.ac.jp/tutorial/index_jp.html`.

[3] Robert C Gentleman, Vincent J. Carey, Douglas M. Bates, et al. Bioconductor: Open software development for computational biology and bioinformatics. *Genome Biology*, 5:R80, 2004.

[4] Duncan Temple Lang. *RCurl: HTTP request interface*. R package version 0.8-0.

[5] Duncan Temple Lang. *SSOAP: Client-side SOAP access for S*, 2007. R package version 0.4-3.

[6] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2007. ISBN 3-900051-07-0.