Dynamic Documents with Word & R

This is an example of a reproducible document that contains code that is dynamically evaluated to create the content such as an R plot or a numerical summary. The idea is that we create our Word/OpenOffice document by writing text and then putting the code in as we would type it in R. The only difference is that we must use a particular style for representing the code in the document. This allows us to find the code of that form and to evaluate it and replace it.

Let’s look at an example. Suppose we wanted to put a plot into the document. We might have something like:

plot(density(rnorm(10000)))

We have used the Rplot style for this paragraph. Now let’s suppose we wanted to have a simple frequency table appear as the result of an R command. We would use the Rcode style:

data(mtcars)  
table(mtcars$cyl)

Now, when we save this file, we can read it back into R and identify the nodes of interest and then process the code within those and generate and insert it back into the document. Then we can render that.

We can also have inlined expressions such as mean(dog) which we mark-up with the style Rexpr.

We have also introduced markup for different concepts in documents such as Rfunc, Rclass, etc.

To support literate programming of functions and packages, we have a style named Rfunction. This is used like Rcode but to explicitly indicate that the code is a function definition. For example,

f =  
function(x, y)  
 sum(x) ^ y

## Processing in R

Having created the document, we can process it in R. When using .docx files, R can read the contents of this archive[[1]](#footnote-0). Using the ROOXML package, we can find all the Rcode, Rplot and Rexpr nodes and then extract their code. We do this with getCodeNodes which finds the text associated with these node styles.

We can then evaluate these code blocks and convert the results into XML that is appropriate for display in Word. We can put the results back into the original document, but that is not appropriate. Instead, we can create a new one. We can save the result of processWordDoc to a file. (This is slightly tricky as we have to ensure that we both copy and update the auxiliary files and then combine them into a .docx file/archive.)

How should we convert R objects to XML for Word? Well, numbers should be displayed as numbers with an appropriate style. Strings can be emitted as strings, with or without quotes. TRUE and FALSE values can be represented by these strings. Vectors of these values might be displayed in their R form, e.g.

[1] 2 3 5

In fact, all R objects can be displayed “verbatim” by inserting them into the Word document using a “Verbatim” style.

Plots of course produce images. We want to determine the appropriate format for Word, or leave it to the user to specify what the appropriate graphics device should be.

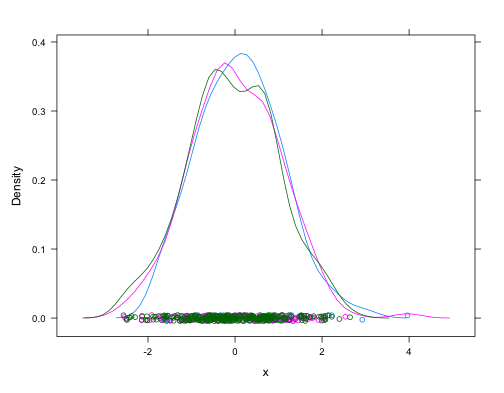
Let’s see what a plot would look like in the “.docx” archive:

Figure The distribution of the number of words in sentences from the 2nd Presidential Debate 2008. The four groups are Brokaw (moderator), Obama, McCain and the Question (from town-hall audience). The colors should be in a legend that did not get copied from R (via Command-C)

## Converting WordML with R code to …

If there was an XSL transformation for WordML to, e.g. HTML or FO, we could override those by including them in our own XSL files and providing templates for these style nodes. We could use the Sxslt package. There is also the R2HTML package. Even-though that does not produce XML nodes, we can use it to generate content as strings and then parse it to create XML nodes.

## Help Files

We can use the same markup or styles within R help files as edited by Word. We can generate template Word files or “stubs” from R and allow the author of the code to document the function. (Of course, anyone else can also document the R object!) When the document is saved, we can read it back into R by processing the different components.

## XML or DCOM

In this document, we appear to be focusing on reading XML documents generated by Word. In fact, we can deal with non-XML documents, assuming we are on Windows. Using either of the rcom or RDCOMClient packages, we can a) construct Word documents from within R and render them in Word, and b) read the content of the documents back into R.

## Bibliography

As came up recently on the R mailing list, we can generate “extra” entries in the bibliography of a document by identifying references to functions, packages, etc. We can “trawl” the document for references to object we can recognize.

## Actions, buttons and controls

It would be “nice” to be able to have a button in the Word toolbar which, when clicked, would send the code in the “current”/“active” code segment (of style Rcode, Rplot, etc.) to R. We would also like to be able to set the style of a paragraph/some code by clicking on a button. Of course, we can do this with the Formatting Palette.

## Formatting Results

Is it possible for us to specify formatting of the result “around” the code that specifies the computations. For instance, if were to say this is a table with this many rows and that many columns, and then put the code “within” that, then we would have a context in which to transform the R objects resulting from the computations.

|  |  |  |
| --- | --- | --- |
| table(mtcars) |  |  |
|  | values go here |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

We also have an Routput style which displays R objects “as R would print them”. For example,

$x

[1] 1.5088728 -0.9190237 0.7577564 -1.2154530 0.7098854 -1.5542338

[7] -0.9658266 0.8329972 0.4465283 -2.8216304

$y

mpg cyl disp hp drat wt qsec vs am gear carb

Mazda RX4 21.0 6 160.0 110 3.90 2.620 16.46 0 1 4 4

Mazda RX4 Wag 21.0 6 160.0 110 3.90 2.875 17.02 0 1 4 4

Datsun 710 22.8 4 108.0 93 3.85 2.320 18.61 1 1 4 1

Hornet 4 Drive 21.4 6 258.0 110 3.08 3.215 19.44 1 0 3 1

Without the hierarchical nature of the document, we have to rely on convention to determine which output is associated with which R code. For the most part, they will be contiguous, but not necessarily. We would like to be able to put an id and a ref to associate the two.

When processing a document, we can remove the Routput elements. For Rcode, Rplot, etc., we can turn these into comments and anchor them on the otput we insert into the text of the document by evaluate the code.

## Validating Documents

We can find all references to R functions, packages, etc. and then verify that they are spelled correctly, etc. If we change the name of a function, we can modify the document to update the references. We can also examine the code to determine dependencies within a document.

## Queries

It would be nice to know if we could and how to

Turn off grammar checking within certain styles, e.g. Rcode, Routput, …

## Tests

A test of a Rexpr: sum(x, na.rm = TRUE)

## The Styles

We have introduced the following styles:

**Rcode, Rplot, Rexpr, Rfunction, Routput**

**Rfunc, Rvariable, Rpackage, OmegahatPackage**

**Rlogical**

Rplot “is based on” or extends Rcode. We want to use this concept to ensure that one need only change the top-most styles to affect all that are based on it.

## References/Bibliography

1. Office XML 2003 Integrating Office with the Rest of the World. Evan Lenz, Mary McRae and Simon St.Laurent
2. <http://msdn.microsoft.com/en-us/library/aa338205.aspx>
3. R (http://www.r-project.org)
4. Rcompression package
5. XMLpackage
6. ROOXML package
7. Open XML Explained e-book (http://72.15.199.198/attachment/1970.ashx)

tudsasd

1. A .docx file is a zipped archive containing various XML files and some others. [↑](#footnote-ref-0)