

[Page One](#)
[Campus
Computing
News](#)
[New Statistics
Server is Online
and Ready for
Use](#)
[LaTeX Short
Course Added
for August](#)
[Summer Hours](#)
[Today's Cartoon](#)
[RSS Matters](#)
[The Network
Connection](#)
[Link of the
Month](#)
[WWW@UNT.EDU](#)
[Short Courses](#)
[IRC News](#)
[Staff Activities](#)
[Subscribe to
Benchmarks
Online](#)

Research and Statistical Support

University of North Texas

RSS Matters

Link to the last RSS article here: [Editing HTML with the Kupu Editor on the Web2Survey Zope Server](#) - Ed.

Open Source Technology in the Classroom

Part I: Creating Web Based Documents that Give Access to the R Statistical Environment: Examples Using the "Through the Web" HTML Editor - Kupu Editor

By [Dr Rich Herrington](#), ACS Research and Statistical Support Services Consultant

Special Announcements: RSS will be maintaining a blog devoted to research and statistics related news - [RSS-Blogs](#); Additionally, RSS will be maintaining a Zope/Plone website devoted organizing communities and resources involved in survey research - [RSS-Surveys](#).

In last month's (June 2006) [RSS](#) column, we discussed using the [open-source - client-side - WYSIWYG - HTML based Kupu Editor](#) to create/edit HTML web pages (Kupu is installed on a [Zope](#) web server). Other [RSS](#) articles have discussed using the [GNU](#) licensed [R](#) and the proprietary licensed [S-Plus](#) (both implementations of the [S language](#)) in the classroom setting as a [low cost alternative to SPSS and SAS](#). More relevant to our purposes here, past [RSS](#) articles have demonstrated how [R](#) and [web server](#) technology can be used in conjunction to create web based tutorials that support [statistical software](#) instruction in classrooms that are [Internet](#) enabled (e.g. [Kernel Density Estimation](#), [Robust Statistics](#), [Two Interfaces to R](#)).

A number of [web server interfaces exist for R](#); currently [RSS](#) supports a few of these interfaces which are based on [CGI](#) web programming practices (see [Rinterface.htm](#) on the [RSS homepage](#) - e.g. [Rcgi](#)). I have used these R web interfaces in many of my [R based RSS articles](#) and in courses that I teach, but I have never discussed the more specific details of the [HTML forms](#) that were used. I think a detailed discussion, concerning how this was achieved, would be useful to interested instructors who wish to use R in their internet equipped classrooms or [distance education](#) based courses that utilize [virtual](#) classrooms. My goal in this article is to demonstrate how instructors can create web based documents for their internet based classroom instruction, so that the R statistical environment is available through a web browser. Moreover, newer web based [HTML editors](#), based on [Ajax](#) programming constructs (i.e. [Kupu](#)), can facilitate the utilization of [server side based](#) entry-

points (e.g. [CGI](#)) or [client side based](#) entry-points (e.g. [Ajax](#) using the [XMLHttpRequest API](#)) to a server running R - the goal in this series is to give concrete examples of how this is done.

From the perspective of the instructor, all that is needed are appropriate lines of [HTML](#) (and/or [JavaScript](#)) to define the forms, since RSS maintains the R web servers. RSS currently maintains several web servers that support both teaching and research functions, one of which is to give instructors and researchers remote access to R for purposes of creating web based tutorials, demonstrations, and classroom instruction (for examples of these web based R interfaces, see: [rss.acs.unt.edu](#) (1, 2, 3) and [kryton.cc.unt.edu](#) (1, 2, 3). Minimally, all the instructor would need to do is create the necessary HTML forms that can access R on the web servers that RSS currently maintains (i.e. [rss.acs.unt.edu](#) & [kryton.cc.unt.edu](#)). These HTML forms can then be embedded or blended into course content that is web based (e.g. these HTML forms could be hosted on [UNT Vista](#)). It is also worth noting that RSS also maintains [web2survey.unt.edu](#), which provides the capability for [online survey/evaluation collection](#) and reporting - the important point here being that [web2survey.unt.edu](#) can also be used for instructional support for new and [emerging web technologies](#) (e.g. [training courses for on-line survey creation](#), survey methodology, web-programming, etc.). Integrating these new technologies together in a [e-learning environment](#) looks promising. *Note that web2survey does not provide a web interface to R itself, but HTML forms maintained on this server (or any other server) can make HTTP calls to the rss and kryton web servers to access R.*

There are some obvious conveniences (at least in educational settings) in having a [public-domain, open-source](#) statistical language, that can be accessed from an HTML web form (this list is not exhaustive and is not in any particular order): **a)** this allows [remote script processing](#) and user [interactivity](#) through nothing more complicated than a simple [web browser](#); **b)** the need to install [statistical software](#) on the client's local storage media can be reduced (e.g. how often do complications arise for students who are attempting to install SAS on their home PCs? RSS's experience is that this is an all too frequent occurrence); **c)** as statistical software updates and [bug fixes](#) become rapidly available (via a [decentralized](#) group of open-source developers), these updates can then be rapidly applied for all concerned, due to the [centralized](#) nature of the server hosting the statistical software (see [The Cathedral and the Bazaar](#) for a more in depth discussion of these points); **d)** due to the non-proprietary nature of the software, there are cost-savings attributed to lowered statistical software expenses for organizations to whom every dollar counts (e.g. public/state institutions: libraries, colleges, universities, non-profit organizations, small businesses, etc.); **e)** even though there are many web based demos, tutorials, and data manipulation routines for statistics/research instruction (e.g. [Java applets](#)); usually these [applets](#) are created for a particular purpose, and are not part of a larger environment for statistical analysis and/or data manipulation. In contrast, R provides functionality for classroom [demonstrations/tutorials](#) while giving students access to a [larger, extensible, integrated environment for data analysis](#) (see [Teaching-with-R.pdf](#)).

More specifically, what does this larger, extensible, integrated statistical environment look like? And, just as important, why should we be motivated to use R in our classrooms?

A Brief Introduction to the R Statistical Environment

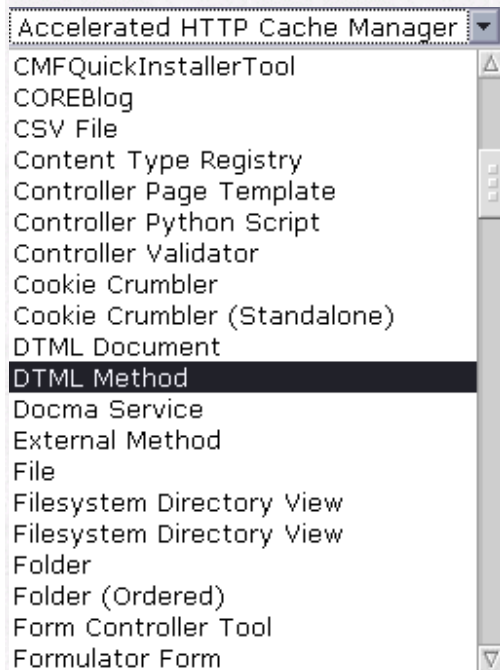
The educational computing society known as the [Association for Computing Machinery \(ACM\)](#) presented their prestigious [Software System Award](#) to [John Chambers](#), a researcher with [Bell Labs](#), the research and development arm of [Lucent Technologies](#). The presentation of the ACM Software System Award to John Chambers marks the first time in the 17-year history of this award, that it has been awarded for data-analysis software and the first time it has been given to a statistician. John Chambers is the creator of the [S System](#) for statistical computing, which the ACM said, "forever altered how people analyze, visualize and manipulate data" (see [John Chambers Gets ACM 1998 Software System Award for Creating 'S System'](#), for the complete story). The [S System](#) has been continually [evolving since 1976](#), and is currently available in commercial product [S-Plus](#), and the GNU licensed [R](#). The implementation of S that we will concern ourselves with is the GNU version of S - [R](#). RSS has devoted [numerous columns](#) to the maintenance and utilization of the R statistical

environment in an educational setting. [Considerable documentation](#) exists for R, most of which is available under some form of public-domain licensing. In addition to base R documentation, extensive documentation exists for [supporting user contributed packages](#). The [cross-platform R](#) project is increasingly evolving into a system that is closely integrated with the underlying operating system environments on which R is maintained. This places R somewhere on a continuum between a full-fledged language for operating system scripting, and a powerful environment/language for statistical and graphical data analysis that rivals legacy statistical systems such as [SAS](#) or [SPSS](#). Certainly, R presents a wonderful opportunity for educators, researchers and IT professionals, who wish to bring the cutting-edge work that is being done at the interface between [statistical science](#) and [computing technology](#), to bear on their respective activities. In the remainder of this current column, we will look at demonstrating, in some small way, web/internet programming practices that can make R available to educational and research audiences - all at little or no cost to the organizations utilizing these technologies.

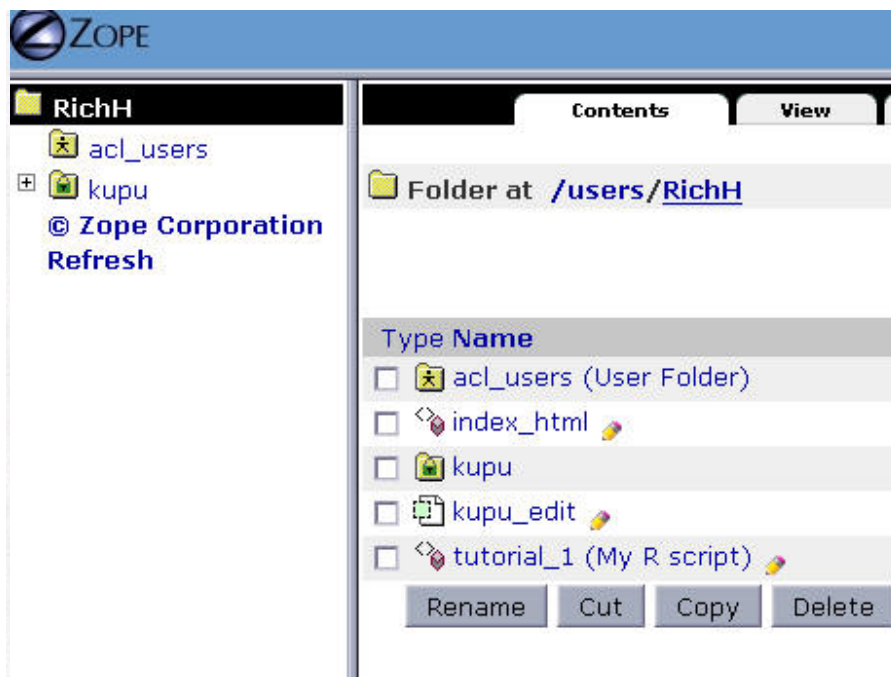
CGI Based Approaches:

Creating an HTML Form with Kupu Editor on the Web2survey Zope Server

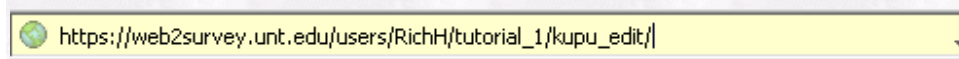
This section assumes that readers have read the RSS article [Editing HTML with the KupuEditor on the Web2Survey Zope Server](#). However, any valid HTML editor that produces HTML source code can be used (e.g. [NVU](#)). Nevertheless, here our purposes are twofold: a) a further demonstration of the [Kupu Editor](#); and b) a demonstration of the HTML/JavaScript necessary to use CGI scripts on RSS's R web servers. In this section, we will be picking up at the point where the folder **kupu** and the [Zope Page Template \(ZPT\) kupu_edit](#) have been created (see [June 2006 Benchmarks Online](#)). Now, we want to create a [DTML method](#) web page so that we can insert the necessary HTML markup tags that will contain our R script (i.e. an HTML form which contains R scripts). To do this, we start by using the object-drop-down-list on the [Zope management interface \(ZMI\)](#). We see the following when we select the list drop down menu:



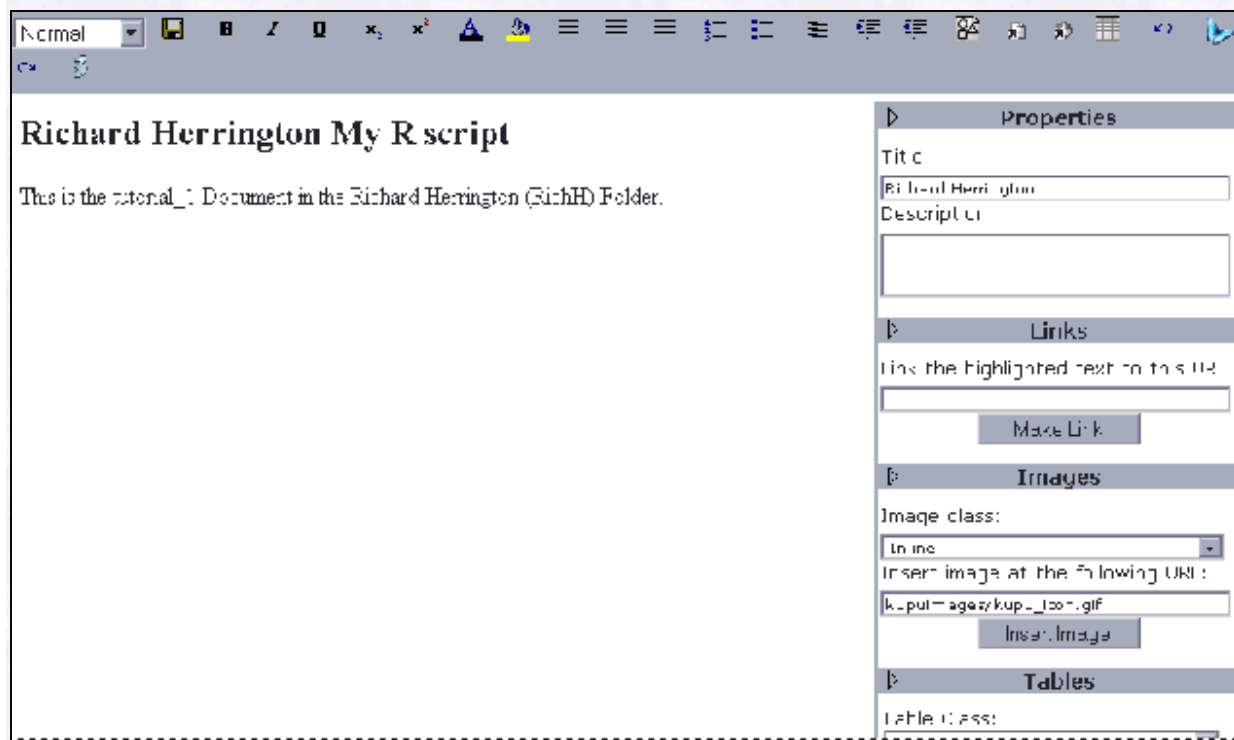
From the drop down form window, we want to create a DTML method object and fill in the HTML form information: set the form field **Id** to a value of **tutorial_1** (remember - no spaces or special characters other than underscore) and set the form field **Title** a string value of: **My R Script** (spaces are allowed here). When we are finished applying these changes, we see the following in our web browser:




Now, we can begin editing the web page with Kupu editor by accessing the URL:



We see the following in our web browser:



Click the  **Edit** button to enter into the HTML source view; this is essentially the HTML markup language (without the formatting tags reflecting the WYSIWYG view). After editing the source HTML, click the button to take the browser window back to the **Preview** or **WYSIWYG** mode. Changes can be saved to the server by clicking the **Save Document to Server** button:



The HTML source view is presented below:

```

8
<h2>Richard Herrington My R script</h2>
<p>
This is the tutorial_1 Document
in the Richard Herrington (RichH) Folder.
</p>

```

I have set up four different R web template forms on web2survey.unt.edu to reflect four different possible uses of the R CGI scripts. Any of these scripts can be modified to reflect: different button labels; different initial R script lines; and which R web server that is accessed.. An overview of these template HTML forms is presented below with a link to a text file that is useful for cutting and pasting into your HTML editor (e.g. Kupu, NVU, Microsoft FrontPage):

HTML template A:

https://web2survey.unt.edu/Utilities/Rweb_template1.txt (note: this is for cut and paste purposes)

This form uses a CGI script based on [Jeff Bainfield's Rweb code base](#). The HTML form presented below is a small part of a larger system of Bainfield's HTML and JavaScript pages that make up Rweb (<http://rss.acs.unt.edu/Rweb/>). These more extensive Rweb pages can be used to automate (with HTML forms) to give a more [guided view of the statistical analysis of data](#). The form template presented below has a default R script placed in the HTML form (which can be modified as needed). This R script produces a histogram of 10 random normal deviates. This form **does NOT allow data upload; does NOT use JavaScript; and necessitates understanding the R language**. The HTTP address can be changed to utilize other R servers (e.g. the *kryton.cc.unt.edu* server - make sure to use the same subdirectory names in the URL path).

```

<body>
<br>
<form action="http://rss.acs.unt.edu/cgi-bin/Rweb/Rweb.cgi"
  enctype="multipart/form-data" method="post">
<p><textarea name="Rcode" rows="20" cols="45">
hist(rnorm(10))
</textarea> </p>
<p><input value="Submit"
  type="submit">
  <input value="Erase"
  type="reset"></p>
</form>
</body>

```

HTML template B:

https://web2survey.unt.edu/Utilities/Rweb_template2.txt (note: this is for cut and paste purposes)

This form template is similar to **HTML template A** except that it utilizes [JavaScript](#) and allows a [tab-delimited data file](#) to be uploaded to the server. For details on how this works, see: <http://www.unt.edu/rss/Rinterface.htm#Upload>

[FrontPage Save Results Component]

```

<head>
<script language="JavaScript">
<!-- hide the following from non JavaScript Browsers
    function checkData(form){
        var URLData = form.URLData.value
        var FileData = form.FileData.value
        if (URLData && FileData){
            alert("You have requested data from both a URL and a file.
                Please only select one.")
            return false
        }
        return true
    }
function RScript(){
    Debug = window.open('', 'R Script', 'width=150,height=150,
        scrollbars,menu,resizable');}
// End of comment hiding JavaScript code -->
</script>
</head>
<body>
<form
    onSubmit = "checkData()"
    action="http://rss.acs.unt.edu/cgi-bin/Rweb/RwebJavaScript.cgi"
    enctype="multipart/form-data"
    method="post"
    target="RScript()">
<TEXTAREA NAME="Rcode" ROWS=15 COLS=60>
hist(rnorm(10))
</TEXTAREA>
<BR>
<INPUT TYPE="submit" VALUE="Submit">
<INPUT TYPE="reset" VALUE="Erase Everything ">
<BR>
Enter a dataset URL :
<BR>
<INPUT TYPE="TEXT" NAME="URLData" SIZE=50>
<BR>
Select a local file to submit:
<BR>
<INPUT type="file" name="FileData" size=40>
</form>
</body>

```

HTML template C:

https://web2survey.unt.edu/Utilities/Rweb_template3.txt (note: this is for cut and paste purposes)

The CGI script that is used in this R web interface is based on M.J. Ray's [Rcgi](#) code base. The HTML form presented allows the script contents to be edited submitted/resubmitted. Additionally, an HTML table is presented with links to [help for packages and functions](#), and [tutorials on using R for graphical and statistical analysis of data](#). R script listings and text output appear in the browser window along with the R script in the HTML form window. The script HTML form window has two buttons: one button creates a postscript view of any graphics that have been generated (a postscript viewer must be installed the client's local operating system - e.g. [ghostview](#)); the other button creates a .GIF view of graphics (no viewer is necessary) that is displayed in a separate browser window (for a full view of this web interface for R, see: <http://rss.acs.unt.edu/cgi-bin/Rprog>)

```

<form method="post" action="http://kryton.cc.unt.edu/cgi-bin/R/Rprog">
<p><textarea name="INPUT" rows="5" cols="64">
hist(rnorm(10))
</textarea><br><input value="Run Program" type="submit"></p>
</form>

```

HTML template D:

https://web2survey.unt.edu/Utilities/Rweb_template4.txt (note: this is for cut and paste purposes)

This form is similar to **HTML template C** except that the form window is hidden. After the initial HTML [POST](#), the form window is visible with script contents visible as well. Subsequently, the form contents can be edited for resubmission.


```
<form action="http://kryton.cc.unt.edu/cgi-bin/R/Rprog" method="post">
<input name="INPUT" value="hist(rnorm(10))"
type="hidden"><p><input value="R Script" type="submit"></p>
</form>
```

Now, we'll use the [Kupu Editor](#) on [Zope](#) (web2survey.unt.edu) to create a form that uses a [CGI](#) script hosted on rss.acs.unt.edu (or alternatively, kryton.cc.unt.edu). One can use [other HTML editors](#) for creating these HTML forms rather than using the Kupu Editor.

We'll work with **form template A**: Copy and paste the text from **form A** (https://web2survey.unt.edu/Utilities/Rweb_template1.txt) into HTML source view of the Kupu editor:



```
<body>
<br>
<form action="http://rss.acs.unt.edu/cgi-bin/Rweb/Rweb.cgi"
  enctype="multipart/form-data" method="post">
<p><textarea name="Rcode" rows="20" cols="45">
hist(rnorm(10))
</textarea> </p>
<p><input value="Submit"
  type="submit">
  <input value="Erase"
  type="reset"></p>
</form>
</body>
```

Click the  **Edit** button to access the **Preview/WYSIWYG** mode. We want to add the [H1](#) level heading title: **"This is my example R Script"**. To do this, we: 1) type the text above the HTML form; 2) highlight the text with the mouse; 3) modify the highlighted font by choosing a H1 level for the title - do this by accessing the drop down list on the menu bar (choose **Heading 1**):

Normal

Normal

Heading 1

Heading 2

Heading 3

Heading 4

Heading 5


Heading 6


Formatted

Example R Script

10))

Submit Erase


Finally, 4) Click the  **Save Document To Server** button to save the current document changes to the server. We see the following in the web browser:


Heading 1  **B** *I* U x_2 x^2    

This is my example R Script

```
hist(rnorm(10))
```

Now, click the  **Edit** button again to enter the HTML source view mode once again:



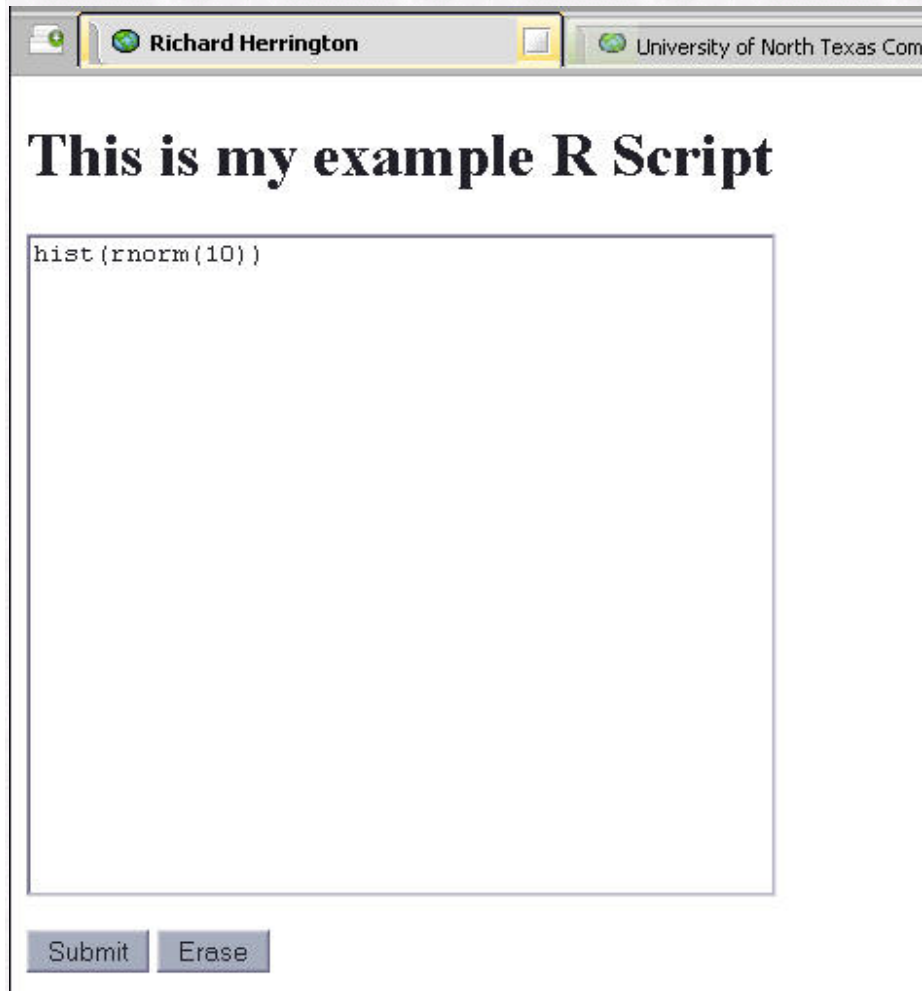
```
<h1>This is my example R Script </h1>
<form action="http://rss.acs.unt.edu/cgi-bin/Rweb/Rweb.cgi"
  enctype="multipart/form-data" method="post">
<p><textarea name="Rcode" rows="20" cols="45">hist(rnorm(10))
</textarea> </p>
<p><input value="Submit" type="submit">
  <input value="Erase" type="reset"></p>
</form>
```

Notice that HTML code has been added to the source view - this reflects the [addition of the <h1></h1> heading tags](#). Using this approach, we can insert [POST](#) references to CGI scripts on remote servers. The POST methods in the HTML form allow immediate server processing of the R script, with the subsequent output being returned to the web browser for display. Now, let's access the form in a non-editing mode. To do this, back space in the browser [URL](#) field until you have the following URL:

https://web2survey.unt.edu/users/RichH/tutorial_1

Press the "Enter" key or click the "Go" icon on the web browser to load the web page named **tutorial_1**

We see the following in the web browser:



The screenshot shows a web browser window with the following elements:

- Address bar: https://web2survey.unt.edu/users/RichH/tutorial_1
- Page title: **Richard Herrington**
- Page content: **This is my example R Script**
- Form area: A large text input field containing the R code `hist(rnorm(10))`.
- Buttons: **Submit** and **Erase** buttons at the bottom of the form.

Next, **edit the contents of the form to reflect the R script presented below**. Save the document changes to the server and click the **Submit** button at the bottom of the HTML form to submit the R script to the server for processing (clicking the **Erase** button will reset the contents of the form window without submitting the form):

This is my example R Script

```
x<-rnorm(10)
x
par(mfrow=c(2,2))
hist(x)
plot(density(x))
qqplot(x)
```

Upon submission, we see the following in our web browser:

Results from Rweb

You are using Rweb1.03 on the server at rss.acs.unt.edu

```
R : Copyright 2006, The R Foundation for Statistical Computing
Version 2.3.1 (2006-06-01)
ISBN 3-900051-07-0
```

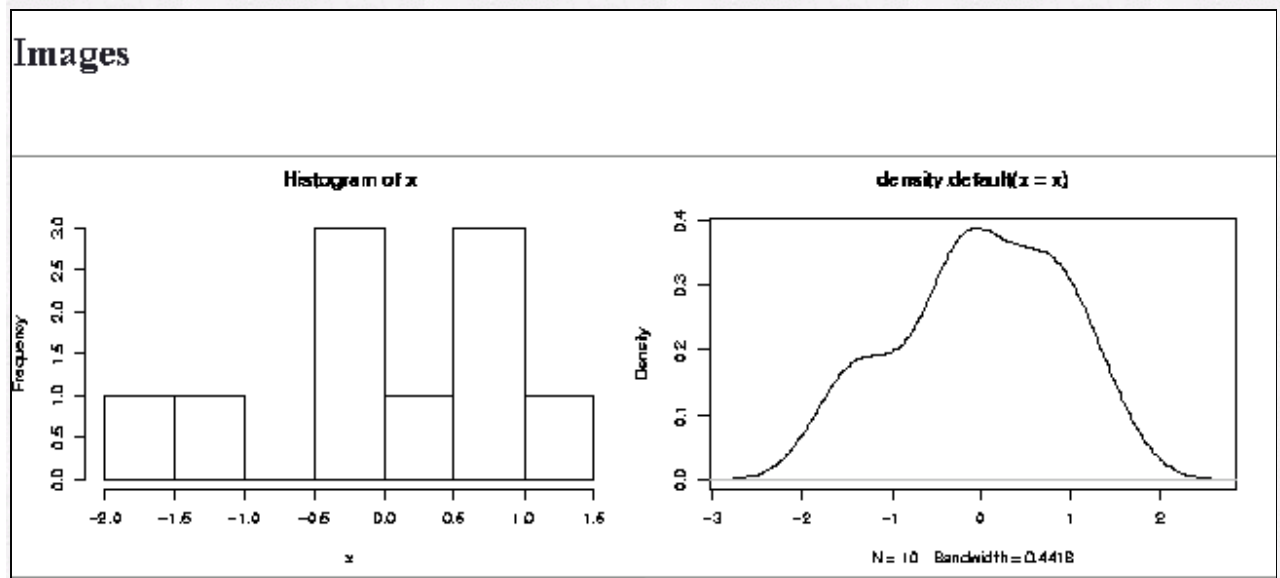
```
R is free software and comes with ABSOLUTELY NO WARRANTY
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details
```

```
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
```

```
Type 'demo()' for some demos, 'help()' for on-line help,
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
```

```
Rweb:> postscript(file= "/tmp/Rout.13731.ps")
Rweb:>
Rweb:> x<-rnorm(10)
Rweb:> x
 [1]  0.006955375 -1.255901650 -0.146972690 -0.389436260
 [6]  0.716074271  0.645183168  0.890775501 -0.206327364
Rweb:> par(mfrow=c(2,2))
Rweb:> hist(x)
Rweb:> plot(density(x))
```

Explanation of the R script: The data object "x" (in this case a vector) is assigned 10 pseudo-random numbers from a Gaussian distribution with location zero and scale one: "x<-rnorm(10)". The next line displays the contents of the vector "x". Next, the "par(mfrow=c(2,2))" line sets up a grid of four plot regions (two by two - however only 2 out of 4 are used). After the plot regions are set up, the following commands produce a plot region with a histogram and a plot region with a [nonparametric kernel density estimate plot](#). [Graphical](#) output appear further down in the Rweb output page, after the R script listing and the R script output are displayed:



Where we are going from here

We close this article, by giving some indication of the topics that we'll be exploring in the coming months as part of the multipart series: *Open Source Technologies in the Classroom*.

R packages:

R supports a number of packages that facilitate CGI and HTML scripting (both server-side and client-side). For example: [CGIwithR](#); [R2HTML](#); [xtable](#); [XML](#); [Rpad](#); [RMySQL](#); [Rdbi](#); [httpRequest](#); [RApache](#), [RSPython](#), [RSPerl](#), etc. These are only a small fraction of the libraries that can support web/internet programming with R. We hope to sample these packages and provide our readers with usable examples of how R can be integrated into internet equipped classrooms and be useful for applications in research:

Zope/Plone:

Zope 3: [Zope 3 applications](#)

Content Management Systems: [Plone](#); [Silva](#); [Kupu Editor](#)

E-learning Environment: [DLCMS](#)

Collaboration Tools: [CoreBLOG](#); [ZiddleyWiki](#) (for example, see [RSS-Wiki](#) - this site is in development)

Web/Internet Programming:

Programming: [Python](#); [Rpy](#)

Remote scripting: [xmlHttpRequest](#); [JSON on Zope](#); [SOAP on Zope](#); [Ajax for Zope](#) - also see [ZopePrototype](#); [TurboGears](#)

This is not an exhaustive list by any means, but these are the "core" technologies that I would like to explore in the coming month's - If you have any interesting applications of these in an educational or research settings and care collaborate/share, feel free to email me at richherr@cc.admin.unt.edu.

Also, if you are interested in applying the technologies to survey research, register with [RSS-Surveys](#). From all of us here at [RSS](#) (Rich, Patrick, Mike, Jon) - good luck in your technology explorations and may the power of open source be with you!

Related Websites

[Kupu Editor](#)

[Kupu Documentation](#)

[Lenya Kupu Documentation](#)

[O'Reilly Article: Rich Web Text Editing with Kupu](#)

[Kupu Zope Project](#)

[Zope Org](#)

[UCLA R Portal](#)

Please note that information published in *Benchmarks Online* is likely to degrade over time, especially links to various Websites. To make sure you have the most current information on a specific topic, it may be best to search the UNT Website - <http://www.unt.edu> . You can also search *Benchmarks Online* - <http://www.unt.edu/benchmarks/archives/back.htm> as well as consult the UNT Helpdesk - <http://www.unt.edu/helpdesk/> Questions and comments should be directed to benchmarks@unt.edu

[Return to top](#)

