Wittier Webapps with RInside: Painless deploying R / C++ Apps on the Net

Dirk Eddelbuettel
dirk.eddelbuettel@R-Project.org

R/Finance 2012
Lightning Talk
12 May 2012
Let’s talk about history

About two decades ago, two important things happened:

1. Tim Berners-Lee introduces HTTP, browsers, and the World Wide Web as a means for sharing data, results, and much more.
2. Robert Gentleman and Ross Ihaka hack away on a Scheme-inspired language implementation tailored for Programming with Data.

And ever since, people have tried to fuse these two.
About two decades ago, two important things happened:
Let’s talk about history

About two decades ago, two important things happened:

- Tim Berners-Lee introduces http, browsers, ... as a means for sharing data, results, and much more.
Let’s talk about history

About two decades ago, two important things happened:

- Tim Berners-Lee introduces http, browsers, ... as a means for sharing data, results, and much more.
- Robert Gentleman and Ross Ihaka hack away on a Scheme-inspired language implementation tailored for *Programming with Data*.
About two decades ago, two important things happened:

- Tim Berners-Lee introduces http, browsers, ... as a means for sharing data, results, and much more.
- Robert Gentleman and Ross Ihaka hack away on a Scheme-inspired language implementation tailored for Programming with Data.

And ever since, people have tried to fuse these two.
And was has this history given us?

Start with R and add ...
And was has this history given us?

Start with R and add ... a veritable alphabet soup:

- html, xml, css, ...
- cgi, ajax, javascript, websockets, ...
- xml, json, yaml, ...
- wdsl, soap, dom, ...
- java, perl, python, php, ruby, ...

And was has this history given us?

Start with R and add ... a veritable alphabet soup:

- html, xml, css, ...
- cgi, ajax, javascript, websockets, ...
- xml, json, yaml, ...
- wdsi, soap, dom, ...
- java, perl, python, php, ruby, ...

Did anybody actually enjoy writing webapps this way?
And was has this history given us?

Start with R and add ... a veritable alphabet soup:

- html, xml, css, ...
- cgi, ajax, javascript, websockets, ...
- xml, json, yaml, ...
- wdsl, soap, dom, ...
- java, perl, python, php, ruby, ...

Did anybody actually enjoy writing webapps this way?

In the beginning, the Universe was created. This has made a lot of people very angry and has been widely regarded as a bad move.

-Douglas Adams
There has got to be a better way

My focus here in on combining C++ and R into applications. But how do I connect these to the Web?

- When I was researching this issue I still looked at the wrong place: Wikipedia on JavaScript frameworks
- A better answer lurked here: Wikipedia on Web Application Frameworks
- Years ago a sharp colleague implemented remote control of co-located 'bots' (essentially lean and mean headless C++ applications) via an *embedded web interface*

This lead to Wt aka “witty”.
Key features of Wt

A fuller discussion / list is on the Wt homepage, but in short:

- Automatic graceful degradation and progressive enhancement
- Supports server-initiated events (comet programming)
- A unified rendering API (SVG/HTML5 canvas/VML)
- Both client-side and server-side validation
- Various security features to avoid Cross-site scripting and CSRF vulnerabilities.
- Includes a compact, modern C++ ORM layer (Wt::Dbo)
- Uses WebSockets if available for communication between client and server, with fallbacks to Ajax or plain web pages

Source: http://en.wikipedia.org/wiki/Wt_(web_toolkit)
A more personal list:

- Zero effort installation on Debian / Ubuntu as Wt is packaged, and packaged well.
- A number of rather nice examples are included, and even the Wt website itself is written as a Wt / C++ app.
- Plenty of Doxygen-generated documentation on the API.
- Good mailing list support

It is really easy to get going.
So going from this Qt app ...
... to this Wt app is really painless

Density Estimation

- Density estimation scale factor (div. by 100)
  - 100
- R Command for data generation
  ```r
  c(rnorm(100,0,1), rnorm(50,5,1))
  ```

- Gaussian
- Epanechnikov
- Rectangular
- Triangular
- Cosine

Resulting chart

Kernel: gaussian

Status
Finished request from 192.168.1.249 using Mozilla/5.0 (Ubuntu; X11; Linux i686; rv:8.0) Gecko/20100101 Firefox/8.0
Overview

This example demonstrates some of the capabilities of the Wt library, in combination with the RInside classes for embedding the R statistical language and environment.

It reimplements a standard GUI / application setting: drawing from a random distribution, and estimation a non-parametric density for which the user selects the kernel and bandwidth. RInside already contains an example of this using gc to provide a standard application.

Here we show how to do the same in a web application which, thanks to the abstractions provided by the Wt, is rather straightforward.

User Input for Density Estimation

<table>
<thead>
<tr>
<th>Density estimation scale factor (div. by 100)</th>
<th>Gaussian</th>
</tr>
</thead>
<tbody>
<tr>
<td>100 ±</td>
<td>Epanechnikov</td>
</tr>
<tr>
<td>R Command for data generation</td>
<td>Rectangular</td>
</tr>
<tr>
<td></td>
<td>Triangular</td>
</tr>
<tr>
<td></td>
<td>Cosine</td>
</tr>
</tbody>
</table>

Resulting R Chart

Kernel: gaussian
These points are mostly **RInside** issues:

- Single instance of R via RInside—so in the example, all ‘session-specific data’ goes back and forth to clients.
- Possibly cache it based on a per-client hash map cookie
- Or rather look into making RInside `fork()` on new connections (as e.g. Rserve does).
- Currently no error recovery, so each `eval()` should probably be wrapped in a `try/catch` block.
The Wt toolkit is available and documented at http://www.webtoolkit.eu/wt.

RInside is available via CRAN and http://dirk.eddelbuettel.com/code/rinside.html; the sources contain the Qt and Wt examples shown here.

We mention Rcpp too as RInside relies heavily on it: available via CRAN and at http://dirk.eddelbuettel.com/code/rcpp.html.