RiverWare Link to External RPL Documentation

Notes

Author: Patrick Lynn

This document contains various sorts of notes concerning the development of a new capability in RiverWare to provide convenient access to external documentation for a RPL set. See the main document for the requirements and high-level design.

1.0 Questions

Which viewer formats should we support? Word supports export to: Word, HTML, XML, rich text, plain text. RiverWare documentation is PDF. All browsers can display HTML, some can display text, PDF, Word. Should the user explicitly tell us format? Do we really need to distinguish between path/URL? Path/URL is orthogonal to format (Word, PDF, HTML). If we punted on support for HTML, could we also punt on supporting URLs? Is there any need for us to know specifics about formats and format viewer/editors? Do we need to require three letter extensions, i.e., do we need to know the format of the document?

Windows has an “open with x” selector, maybe we could hook up with that?

Do we want to allow the user to provide us with command line switches for a particular program? Maybe we could provide information on popular programs in online help.

Do we allow the user to specify an anchor for an object or do we autogenerate the anchor name based on the object name? If the location is a URL then they can insert the “go to anchor” command themselves, and in this case we probably want to recognize that they have done so and not insert another similar command.

2.0 Notes

MS Word: is there a command line switch to open a document in a read-only mode? On startup, can execute a macro, specified by name, but how about moving to a bookmark?

Acrobat/AcroRead and PDF: Acroread 8 supports “open parameters” that can cause it to initially display a named destination or a given page number. Supported when executing Acroread programmatically as well as when opened by some browsers (by including the parameters in the URL).

Framemaker: can “insert hypertext command” to create a named destination that will appear in the PDF as well.

To tell a browser to go to a particular page in a PDF: http://www.mydomain.com/myPDF.pdf#page=7
To a particular “named destination”: http://www.mydomain.com/myPDF.pdf#nameddest=TOC"
In HTML, the link would look like:
    <a href="http://www.mydomain.com/myPDF.pdf#nameddest=TOC">Link text</a>

We should make sure that the copy buffer is available to other programs so that they can conveniently do searches in the viewer/editors based on RPL object name.

Note, there is of course nothing we can do to prevent a “viewer” from editing the documentation by accessing it from outside of RiverWare.

Often the function your interested in is the one being called, not the one being edited. For now we assume that it is sufficient that the user can open the functions editor by double-clicking on the function call expression and from that dialog access the help.

To facilitate the creation of a “skeleton” document for an existing ruleset, RiverWare could automatically generate a document with appropriate anchors for each object in the ruleset.

An alternative to requiring that the user insert anchors into their documentation would be to require them to include object names as part of the regular content of the documentation. RiverWare would then ask the access program to perform a string search for the name of the current object and go to that location within the document. Not all programs (if any) support this behavior and it is not guaranteed to go to the best location within the document (if the object name appears in more than one place).

RiverWare will create the full location specification by concatenation of the two pieces, and this specification to the access program, so RiverWare does not need to know if the two pieces are in fact a true directory and file name.

3.0 Design Notes

RiverWare’s current mechanism for launching another executable is encapsulated in the cwProcess class. This class supports creating the process with an arbitrary number of textual command line arguments and process termination. No other communication with the executing process is explicitly supported. Though the process is brought to the foreground when “re-executed”, I’m not sure how this is accomplished.

Is there any standard protocol for communicating with a browser? Do they all accept the URL as a command line argument? What if it is a file, would we need to prepend “file://”?

The RPL Flex/Bison grammar will need to be extended to support accomodate the object configuration data.

Probably we need to manage the viewer/editor executable(s) just as we do the online help reader. That is noticing if it’s already going, raising it if it is, noticing when it goes away. The Workspace does this for the online help reader with static members and we could do the same.

Note the legal characters for named destinations might not coincide with our RPL object naming conventions. E.g., one can not have spaces in an anchor name. We might need to replace spaces with underscores and otherwise convert names as a first step.

We probably don’t want to close an edit process automatically when we exit RiverWare, changes might be lost. Should we just leave such a process running? Probably we should ask the user what to do. Note that if RiverWare terminates unexpectedly, this might cause a problem. Maybe editors should always be independent processes.

What is the “.docx” extension? Word 2007 format? We will need to support this as well.

What is the syntax for referencing environment variables within location specifiers?

MS Word does not support “read only” mode; control for this sort of thing is through file permissions.

The File Format Manager could also know about the other formats used by RiverWare: model file, ruleset, RCL.

This would perhaps allow the user to choose the file name extension that they would like to use. Any other benefit?

How to shoe-horn program arguments into the mix?

Does Windows distinguish between different ways to access a file in a more general and useful way? We hardwire in Edit/View distinction, is there anything comparable there which might be useful?

Is it okay to assume that the access program will take care of managing number of executables and not lead to a proliferation of windows open? Should we provide a hook for the user to control this?

We assume that each program supports a command line of the form: exec location
MS Word 2007 says that the file name needs a leading “/t” switch, but it seems to work fine without it.

We will intially hardwire the knowledge of certain formats and their extensions (HTML, PDF, Word). On Windows use registry for the other fields, reasonable defaults on solaris.

We can use a “chooser” icon for the directory and file chooser buttons. The icon for this purpose in the diagnostic manager is a folder with an arrow leaving it.

Do we need to support anchors for editing?

From where should we provide access to the File Format Manager?

We might consider omitting the labels Default/Custom for the radio buttons in the config dialog, and perhaps adding tool tips. Their purpose might be clear.

How to use the anchor? If the format is HTML, assume URL and tack on as a fragment? If PDF, assume Reader 8.0 and use command line argument? What if they want to view PDF in their browser? What about only supporting the anchors for URLs, which we take to be any location beginning with “http:” or “https:” or “<alphanum>:”?

How do we determine if something is a URL? For all access programs and file formats, should the fragment be the anchor, or maybe name=anchor for PDF files?