File Input/Output

This tutorial and example stack use the **open file**, **read from file**, **close file**, and **write to file** commands along with the **specialFolderPath** function to read information into a LiveCode stack from an external file (in this case, a text file, but it could be a binary file if so specified), as well as write data from a LiveCode stack into an external file.

Upon completion of this tutorial, you should be able to:

•Use LiveCode to write a file to disk

•Use LiveCode to read a file from disk

•Understand the **specialFolderPath** function in LiveCode

Resources of Interest:

•http://www.sonsothunder.com/devres/ (http://www.sonsothunder.com/devres/revolution/tips/file010.htm)Revolution (http://www.sonsothunder.com/devres/revolution/revolution.htm)/tips/file010.htm (http://www.sonsothunder.com/devres/revolution/tips/file010.htm)

•http://www.sonsothunder.com/devres/Revolution/Revolution.htm (http://www.sonsothunder.com/devres/revolution/revolution.htm)

You can download the sample stack from this url: https://tinyurl.com/yaouhap3 (https://tinyurl.com/yaouhap3)

FileInputOutput.zip

Create Stack & File to Read

1. Launch LiveCode. Create a new stack by performing the menu command **File -> New Stack -> Default Size**. Save your file.

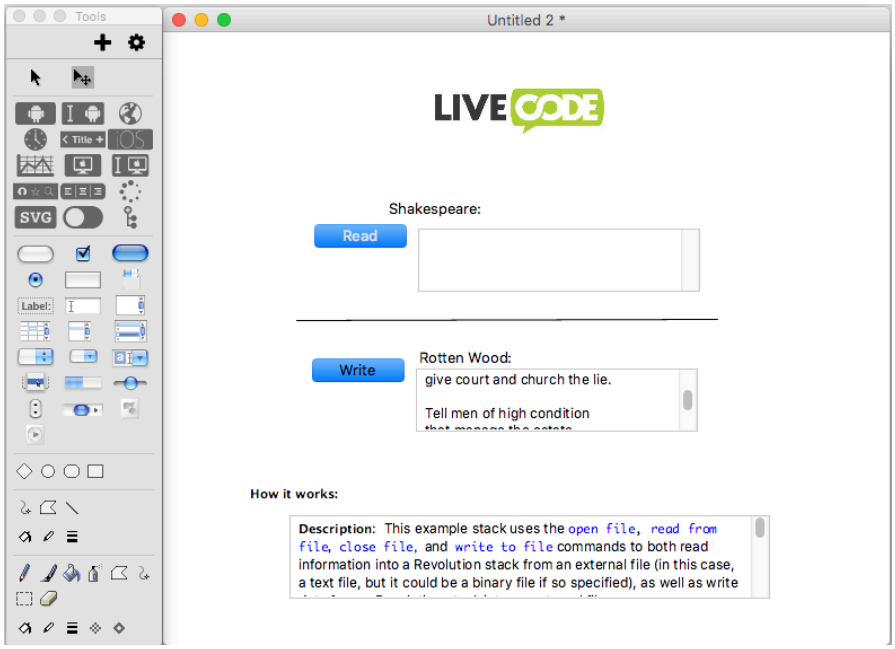
2. Now we are going to create a script at the stack level that will create the text file that the stack will open to read and then write its contents in a text field. To create this handler, perform the menu command **Object -> Stack Script** and write the following in the resulting script editor window:

```
on preOpenStack
        put empty into fld "Shakespeare" -- initializes empty field
end preOpenStack
```

```
on openStack -- This creates the text file on the desktop that we will later read into a stack's text field.
        open file specialFolderPath("desktop") & "/SetUp.txt" for write
        put "Let me not to the marriage of true minds admit impediments." into myText
        write myText to file specialFolderPath("desktop") & "/SetUp.txt"
        close file specialFolderPath("desktop") & "/SetUp.txt"
end openStack
```

The openStack handler uses the open file command along with the specialFolderPath function to create a text file on your Macintosh or Windows desktop. The contents of the text file are on the second line of the script of the openStack handler and are the first two lines of a famous poem by William Shakespeare (if you are feeling particularly creative, substitute any text for that of the Bard's).

Create Stack Objects



3. Create the following objects for your stack:

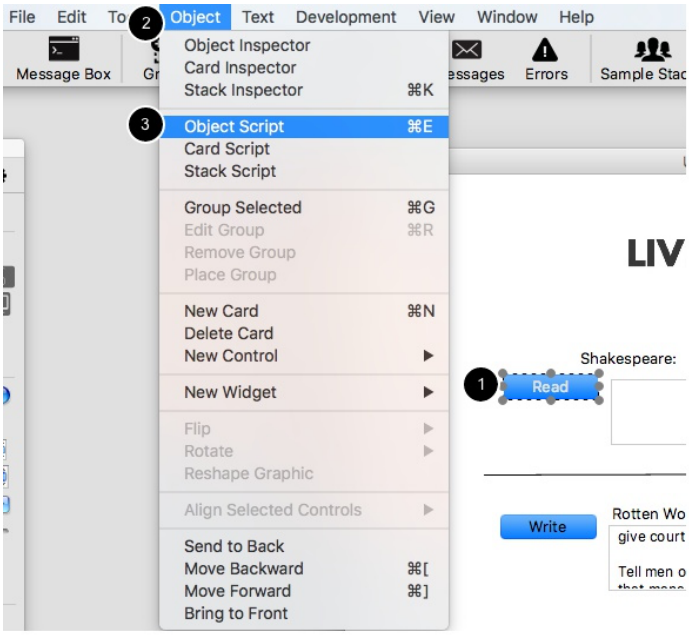
   a push button named "Read"

   a push button named "Write"

   a text field named "Shakespeare"

   a text field named "Rotten Wood"

Read from File



4. We will now script the "Read" button. To do so, click on on it then access the script editor by performing the menu command **Object -> Object Script** as shown above (alternatively, you can use the appropriate keystroke combination to directly open the script editor, which on a Macintosh is apple-e or command-e).

In the script editor window, enter the following script:

```
on mouseUp
	open file specialFolderPath("desktop") & "/SetUp.txt" for read  -- opens the file created by the stack script earlier referenced
	read from file specialFolderPath("desktop") & "/SetUp.txt" until EOF -- reads the contents of the file created earlier by the stack script
	put it into fld "Shakespeare"  -- puts the contents of that file into the text field we just created in our stack
	close file specialFolderPath("desktop") & "/SetUp.txt"  -- closes the file created by the stack script
end mouseUp
```

Write to File

5. Similar to instruction #4 above, access the script editor window for the button "Write" and give that button the following script:

```
on mouseUp
	open file specialFolderPath("desktop") & "/myData.txt" for write  -- creates a file for LiveCode to write to
	put field "Rotten Wood" into myText  -- places the text from the field "Rotten Wood" into a local variable
	write myText to file specialFolderPath("desktop") & "/myData.txt"  -- writes the text placed in the local variable into the open text file we created
	close file specialFolderPath("desktop") & "/myData.txt"  -- closes/saves the text file containing the text form the field/local variable into a text file on the desktop
end mouseUp
```

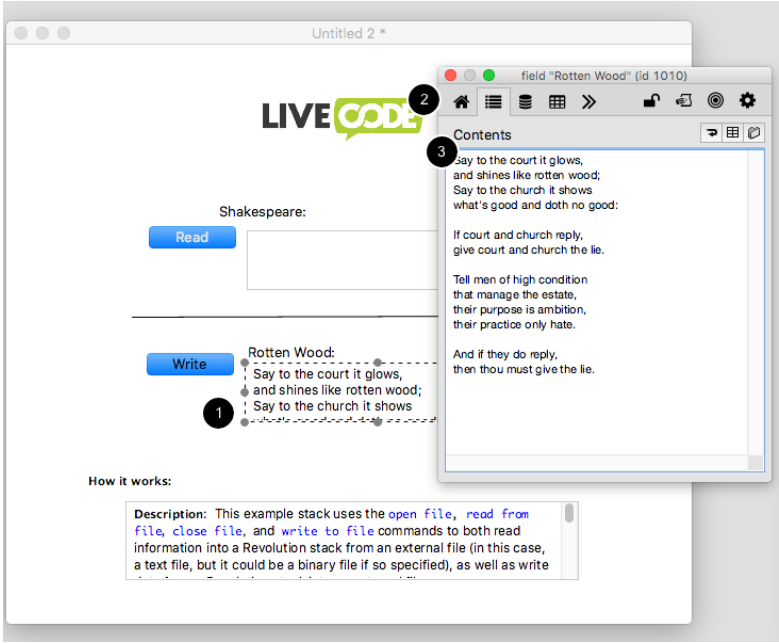
NOTE: We have used the "&" concatenator to create these text files for read/write purposes. "string" & "string" runs the two strings together without spaces, as in

"Foo" & "bar" evaluates to "Foobar" (no spaces)

However,

"Foo" && "bar" evaluates to "Foo bar" (space between)

Create the Text to Write to File



6. Now we will input the text into the field "Rotten Wood" that the "Write" button will write to a text file on the desktop. To do so, *either* switch to browse/run mode by clicking on the plain arrow tool on the Tools Palette and click inside the "Rotten Wood" field and type the following text *or* access the "Contents" component of the field's **Object Inspector** by selecting the field by clicking on it and then performing the menu command **Object -> Object Inspector** and selecting "Contents" from the tab at the top, as indicated in the image above (alternatively, double-clicking on the field will also bring up the **Object Inspector)**. Enter the text below (or whatever text which you fancy).

Say to the court it glows,

and shines like rotten wood;

Say to the church it shows

what's good and doth no good:

If court and church reply,

give court and church the lie.

Tell men of high condition

that manage the estate,

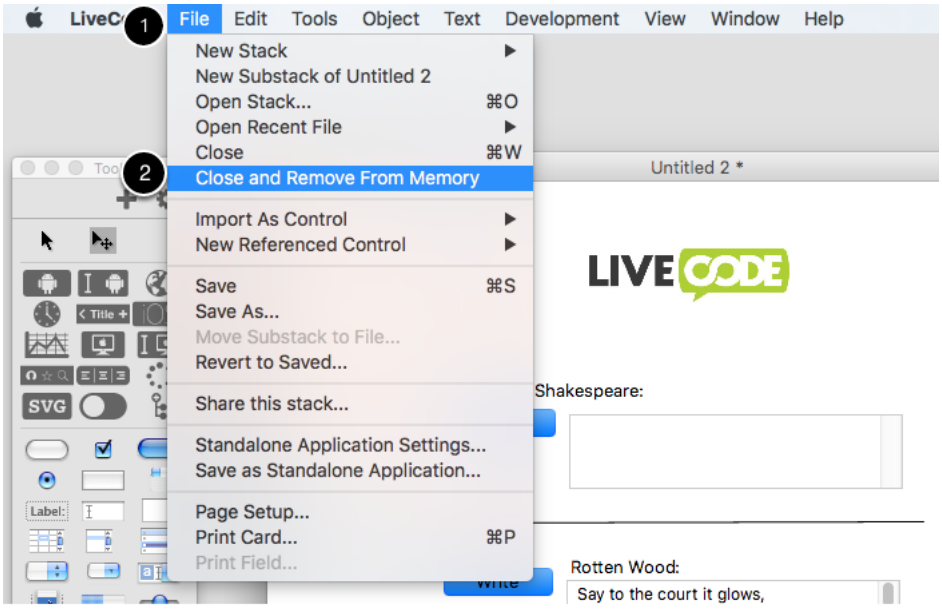
their purpose is ambition,

their practice only hate.

And if they do reply,

then thou must give the lie.

Save & Test!



7. Save your stack by performing the menu command **File -> Save As...** ; close it and remove its contents from memory by performing the menu command **File -> Close and Remove from Memory**.

8. Reopen your stack and test your buttons in browse/run mode by clicking on the simple arrow tool in the Tools Palette as indicated above; don't forget to look for the text files created on your desktop!

---

3 Comments

Simon Knight Thursday Nov 12 2009 at 04:52 PM

Hi Judy,
Thank you for a very clear tutorial. I am new to Rev Talk and this type of tutorial is just what is needed. One question I have is how do I process a binary file? I have an application written in RealBasic that reads and writes two byte data words. The app reads the binary data into a class object that allows me to read and write the binary based on its location within the file as written in the file specification document e.g. a method call of Get_data_Word (Block, Word) where Block and Word are integers. I have just completed my first RevTalk application and am still having problems adjusting to un-typed variables.
Thanks
Simon

Shane Carroll Tuesday Nov 24 2009 at 02:28 AM

Hello,

I am very interested in seeing a tutorial that might show how to read and parse delimited text into and array. Do you know if there is an available tutorial that might go the extra step and show that?

regards,
Shane

Hanson Schmidt-Cornelius Tuesday Aug 02 2011 at 06:42 PM

Hi Shane,

the following lesson may help you:

http://lessons.runrev.com/spaces/lessons/buckets/784/lessons/9678-How-do-I-convert-tab-delimited-data-into-an-array (http://lessons.runrev.com/spaces/lessons/buckets/784/lessons/9678-How-do-I-convert-tab-delimited-data-into-an-array)-

Thanks,

Hanson