A Platform Independent File Browser

Sidney Marshall

August 6, 2017 (updated June 10, 2024)

Contents

1 File Names and Session Names **2**

2 Starting the File Browser **2**

3 The Seven Types of Windows **3**
3.1 The Single File System Window 3
3.1.1 Buttons etc. 3
3.2 The Dual Window 4
3.2.1 Buttons etc. 5
3.3 The Terminal or Shell Window 7
3.4 The Edit Window 7
3.5 The Find-Replace Window 8
3.6 The File Difference Window 8
3.7 The Progress Window 8

4 Mouse Button Summary **8**

5 Drag and Drop **9**

6 File Transfers [This is not yet implemented.] **9**
6.1 Resuming A File Transfer 9

7 Discussion on File Name Encodings **9**
7.1 URIs, URLs, and URNs 10

8 Comments on jsch — the remote SSH Library **10**

[*The File Browser is still a work in progress so details are sure to change. It is currently useful enough to maintain the squaredancing website.*]

Abstract

There is a need for a platform-independent file browser that will work on Microsoft and Linux platforms. This Java program attempts to fill this need. It is loosely modeled after the `WinSCP` program and the windows file browser. The current program allows transfers between two file systems. The file systems can be either local or remote or one of each. It also allows opening a terminal or shell window on a remote machine.

It is implemented entirely in Java and should run on any platform supporting Java 8.

1 File Names and Session Names

Local file names are either of the form "`/c:/...`" (windows) or "`/xxx...`" (Linux). For remote files the format is "`//user@x.y.z/c:/...`" (windows assuming an sftp server) or "`//user@x.y.z/...`" (Linux). For remote file names if there is no closing '`/`' then the home directory (obtained by `pwd` is used (as in `//swm@glados.cs.rit.edu`). Using remote file names causes the file browser to open up an sftp connection. All opened sftp connections are visible at the top-level of the displayed file tree. All functions work for remote files except noticing changed files — a refresh button must be pressed to resynchronize the file browser's display to match the remote files.

A session name is of the form `user@host` and is used for remote shells.

Note the difference between `//user@host` for remote files and `user@host` for session names.

Passwords are kept (somewhat encrypted) in a file named `.filebrowserrc` in the home directory so passwords need to be entered only once. Currently there is no option to suppress this feature. This may be a security problem.

Transferring files between two remote machines viewed through different file browser invocations also works.

2 Starting the File Browser

The file browser should run on any platform that supports Java.

There are five ways to use the file browser.

Started with no arguments or clicking on the program icon the file browser will start with a single file system Window on the root of the local file system.

If the first argument is `-ssh` all subsequent arguments are interpreted as session names of the form `user@host` and a remote shell is opened for each argument.

If the first argument is `-diff` the next two arguments are interpreted as `URL`s and a diff is performed between the two files.

Otherwise, if there are exactly two arguments then a dual Window is opened using the arguments as `URL`s for the two sides of the dual Window.

Otherwise, (for one or three or more arguments) each argument is interpreted as a `URL` and a separate single file system Window is opened on each one.

Otherwise, if there are no arguments, a single file system Window is opened on the root of the file system.

Link support is not yet complete and will probably never work for Microsoft file systems.

3 The Seven Types of Windows

There are seven types of windows used by the file browser. They are:

Single File	Displays files and directories
Dual	Displays and updates between two directories
Shell	A shell on a remote system
Edit	An editor generally based on a file
Find/Replace	does a find or replace on parent window
Diff	Shows the results of differencing two files
Progress	Shows the progress of a file transfer

3.1 The Single File System Window

The single file system Window somewhat mimics the windows file browser except it works on remote directories and files too.

A list of files, directories, and links (local or remote) are displayed. The columns are the name of the file/directory, the size, and the modification date. Directories and file system roots have a gray background. Links have a yellow background and the size field is replaced by the link target.

By clicking on the `Tree View` button a tree view can be seen side-by-side with the list view. The boundary between the two views can be moved with the mouse. When viewing the tree view clicking on the `List View` button the display reverts to a single window displaying the list of files and directories. Moving the vertical divider will also display/hide the tree view.

3.1.1 Buttons etc.

The top row of buttons are

Left Arrow	Go to the last-viewed directory, a.k.a. “back”
Up Arrow	Goto parent directory

Right Arrow	Go to the next-viewed directory, a.k.a. "forward"
Show/Hide dot files	Show or hide files/directories beginning in "."
Tree View / List view	Show or hide the tree view
Clone	Start another file browser displaying the same directory as the current one

Note: Using the "up arrow" to get to the top of the file hierarchy will give, in the drop-down menu, a list of all remembered directories, both local and remote.

The next row contains an editable text field that generally displays the directory being displayed. Both drag and drop and clipboard cut and paste work on this text field. Editing this text field and clicking the `Go` button will display the directory in the text field. If the directory does not exist then the deepest existing directory along the specified path is displayed. The `Add` button adds the currently selected directory to the top level `file system root(s)` drop down menu. The `Delete` button removes the currently selected directory to the top level `file system root(s)` drop down menu.

The display area displays the tree of directories on the left (if not hidden) and the files of the selected directory on the right.

Right clicking on a (remote) directory on the right of the display will bring up a terminal or shell window on the same file system.

Right clicking on a file on the right of the display will bring up an editor on this file.

The bottom row of buttons are used for debugging and are

Print Selected Path	Prints the selected path on stdout
Print Tree	Prints the displayed tree
Delete tree	Deletes the selected file or tree (use carefully — there is no undo!)
Touch File	"Touches" or creates the file in the editable file name window
Create Directory	Creates a directory named in the editable file name window

Drag and Drop works in this type of window.

3.2 The Dual Window

If the file browser is started with exactly two directories then a combined Window is generated. This display compares files from two file systems (this could be the same file system) side by side.

A combined view allows comparison of two directories (or files?) and an update operation that, with the right options, will transfer files or create directories to make the two directories mirrors of each other.

3.2.1 Buttons etc.

The top line of the display is

(left) Up Arrow	Go to left-side parent directory
(left) Text window	Path of left-side directory
(left) Down arrow	Drop down menu of left-side sub directories
refresh (button)	Refreshes display by comparing left and right directories; right-clicking this button will check file dates with no daylight savings adjustments
(right) Up Arrow	Go to right-side parent directory
(right) Text window	Path of right-side directory
(right) Down arrow	Drop down menu of right-side sub directories

The body of the display is a table with seven columns:

x	Checked means this file or directory will be copied or deleted when the Synchronized button is pressed
path	Suffix to actual paths (the prefix is in the left or right text window)
left size	Size (in bytes) of left-side file or directory
left date	Date of left-side file or directory
icon	Icon indicating direction and type of transfer (see below)
right size	Size (in bytes) of right-side file or directory
right date	Date of right-side file or directory

A link will have a yellow background and the size field will be replaced with the target of the link.

Clicking on the left check box will reverse the checked state.

Clicking on a (direction) icon will reverse its direction.

Right-clicking on a size or date of a directory will bring up a single file system window on the corresponding directory. Right-clicking on a size or date of a file will bring up a file editor on the corresponding file.

Right-clicking on the direction icon will bring up an edit window containing the `diff` between the left and right files. This only works for files.

The bottom buttons are

Select All	Select all lines

Check All	Check all lines
Check Selected	Check all selected lines
Target Left	Make the target for selected lines the left-side file or directory
Target right	Make the target for selected lines the right-side file or directory
Unselect All	Unselect all lines
Uncheck All	Uncheck all lines
Uncheck Selected	Uncheck all selected lines
Default Target	Make the target for selected lines the default
Synchronize	Execute the specified transfer for all checked lines; right-clicking this button will only update file dates

Below the bottom buttons is a line of all 16 possible icons. Clicking on one of these cons will select all lines in the table with this icon. The direction of the "arrow" points to the side that will be modified (created, deleted, or overwritten). The colors indicate:

Green	Creation of file or directory
Red	Overwriting or deleting of a file or directory
Magenta	Overwriting a newer file with an older file
Brown	Overwriting a file with a directory or a directory with a file

The icons are:

Left Overwrite Newer Right	Overwrite left file with a newer right file
Left Overwrite Older Right	Overwrite left file with an older right file
Left Create File	Copy right file to a new left file
Left Create Directory	Copy right directory tree to a new left directory
Left Delete Directory	Delete left directory
Left Delete File	Delete left file
Right Directory Overwrite Left File	Copy right directory over left file

Right File Overwrite Left Directory	Copy right file over left directory
Right Overwrite Newer Left	Overwrite right file with a newer left file
Right Overwrite Older Left	Overwrite right file with an older left file
Right Create File	Copy left file to a new right file
Right Create Directory	Copy left directory tree to a new right directory
Right Delete Directory	Delete right directory
Right Delete File	Delete right file
Left Directory Overwrite Right File	Copy left directory over right file
Left File Overwrite Right Directory	Copy left file over right directory

Below each icon is a fraction. The numerator indicates how many lines with this icon are checked. The denominator indicates the total number of this kind of icon (checked or unchecked).

The last line indicates the file or directory currently being worked on.

To update only file dates the recommended procedure is to do a normal update first and then right click `refresh` and then (after possibly checking appropriate files and changing directions) right click `Synchronize`.

3.3 The Terminal or Shell Window

A Shell Window is a shell on a remote system. You can type commands and see results. Most terminal controls are implemented and more are being implemented. The terminal or shell Window is quite usable for most purposes.

Right clicking on the body of a shell window will bring up a `FileBrowser` on the home directory.

X11-window tunneling is not implemented as I haven't figured out how to do this (or have many other people).

3.4 The Edit Window

It is possible to bring up an edit window on a local or remote file by right-clicking on the file in the Single File System Window or by clicking on the date or size in the dual window. This is an ordinary editing window which should display the original contents of the file before you start editing. The buttons below are:

Read File	Rereads the contents of the file into the edit window

Write File	Writes the current contents of the edit window back to the file
Print	Prints the current contents of the edit window

3.5 The Find-Replace Window

Right clicking in an Edit Window or a Command Window brings up a Find-Replace Window. There are two editable fields (find and replace) and four buttons in the window:

Find	Finds and selects the next occurrence of the text in the adjacent find field
Replace	Replaces the selected text with the contents of the adjacent replace field
Replace/Find	Does a replace operation followed by a find operation
Find Lines	Finds a particular line number

If a `Find` search does not find any further matches then the selection is set to the beginning of the file and the status line displays `not found`. This means the search will start at the beginning, i.e., the search will start at the beginning the next time a search is attempted.

The status line also contains a check box on the right labelled `Ignore Case`. If checked, the search is done without requiring that the case of the search string maches. If unchecked, the search string must match exactly, including case.

The `Replace` operations modify the contents of the associated Edit Window.

The `Find Lines` will set the cursor at the line number indicated by the search window.

3.6 The File Difference Window

This window appears after right-clicking on an arrow in the dual window. It displays the difference between the left and right files in a diff format.

3.7 The Progress Window

This window appears automatically when there is a transfer to or from a remote file. It indicates the name of the file and the percentage of the file transfer completed.

4 Mouse Button Summary

Right button
Edit Window (including FileEditWindow)

bring up find/replace window
Single Window File Browser
on directory (tree or file view)
bring up a shell window
on file
bring up File Edit Window
Dual Window File Browser
length or date field
bring up File Edit Window
direction arrow
bring up diff window of files on both sides
shell window
bring up file browser

5 Drag and Drop

6 File Transfers [This is not yet implemented.]

The target file has a `.partfile` extension added to it. When the file transfer is successfully completed this `.partfile` file is moved to a file with the proper file name. If the transfer does not complete successfully then the `.partfile` file is left and the original target file is untouched (leaving the file date intact).

6.1 Resuming A File Transfer

Files are transferred to a temporary file with a `.partfile` suffix and, on successful completion of the transfer, are moved to the actual file name. If a file transfer is interrupted in the middle the `.partfile` file will remain. The file transfer can be resumed by appending to the `.partfile` file.

To determine if a transfer can be resumed or must be restarted, the file dates of the source file and the `.partfile` file are compared. If the `.partfile` file's date is more recent than the source file's date then the transfer can be resumed. Otherwise, the file transfer must be started over from the beginning as the source file's contents have changed.

It is always safe to overwrite or delete the `.partfile` file. For now, transfers are always restarted from the beginning.

7 Discussion on File Name Encodings

The original `jsch` program assumes that file names are legal UTF-8 strings. Unfortunately, windows file names are encoded in a variant of ISO-8859 resulting in file names that are illegal UTF-8 names. The conversion process between the file name bytes and UTF-8 loses information and the transformation is not reversible. I have modified the `jsch` program to encode file names (both in and

out) as URI's which can handle an arbitrary sequence of bytes. This is done by encoding problematical bytes with the sequence %xx where xx is the hex representation of the problematical byte. Per cent signs are encoded as %25 and other bytes that could result in parsing problems for a URI are also appropriately encoded.

Whereas Linux allows any character except null and / in a file name, windows is more particular and has a longer list of characters not allowed in a file name. If a Linux file name contains a byte that is an illegal in a windows file name then the transfer will fail. I am not sure what to do about this.

Note that no attempt is made to convert a legal UTF-8 name to a re-encoded ISO-8859 windows name.

7.1 URIs, URLs, and URNs

A URI is a uniform resource identifier while a URL is a uniform resource locator. Hence every URL is a URI, abstractly speaking, but not every URI is a URL. This is because there is another subcategory of URIs, uniform resource names (URNs), which name resources but do not specify how to locate them. The mailto, news, and isbn URIs are examples of URNs.

A URI represents a reference in the syntactic sense defined by RFC 2396. A URI may be either absolute or relative. A URI string is parsed according to the generic syntax without regard to the scheme, if any, that it specifies. In other words, a URI is little more than a structured string that supports the syntactic, scheme-independent operations of comparison, normalization, resolution, and relativization.

A URL, by contrast, represents the syntactic components of a URL together with some of the information required to access the resource that it describes. A URL must be absolute, that is, it must always specify a scheme. A URL string is parsed according to its scheme. In other words, a URL is a structured string that supports the syntactic operation of resolution as well as the network I/O operations of looking up the host and opening a connection to the specified resource.

8 Comments on jsch — the remote SSH Library

The library jsch is an implementation of the SSH protocol written entirely in Java. The library allows setting up a SSH session and then opening up an sftp or ssh channel. I have modified this library to take file names with non UTF-8 file names. This was accomplished by duplicating the original library code for put, get, ls, readLink, symLink, rename, rm, setMtime, rmdir, mkdir, stat, lstat, setStat, and getHome to code that takes URIs instead of Strings.

I have not yet figured out how to tunnel X11 over SSH.