

Lab System Contents

Overview

Lab Header

Lab Sections

Running Labs

Lab Author

Rich-Text

Sounds

Program Access

Lab Overview

A J Lab file combines a text description and executable J code in a single file. The Lab system lets you step through the file, displaying the text and executing the code. The user is in a normal J session, and can try out other J expressions while running the lab.

A Lab is structured into one or more *chapters*, each containing one or more *sections*. The user can go through chapters in any order, but the sections within each chapter must be run sequentially.

Lab files are either ordinary text files with extension `.ijt`; or rich-text files with extension `.rtf`. Rich-text files permit greater control over text formatting, but work only in Windows 95/NT, and take more effort to set up.

You can create the `ijt` files using Lab Author or any text editor such as Notepad; and create the `rtf` files with word processors such as Word or WordPad, see Rich-Text.

Lab files provided with J are stored under the subdirectory `system\extras\labs`. The directory `system\extras\labs\personal` is available for the user, though you can create labs in any directory – if so, you should update file `system\extras\labs\labdir.ijs`.

Lab Header

A Lab file has a *header* defining various nouns, followed by one or more chapters that start with lines beginning `Lab Chapter` followed by the chapter name. If there is only one chapter, the line `Lab Chapter` need not be given.

Each chapter consists of sections that start with lines beginning `Lab Section` optionally followed by a section title. Where the section title is not given, it is assumed to be a continuation of the previous section.

Header

The header is executed as a script in the `jlab` locale, and may be used to define the following nouns:

LABTITLE	Title (must be given first)
LABAUTHOR	Author (optional, may include address, email etc.)
LABCOMMENTS	Comments (optional)
LABERRORS	1=continue after errors in code (default 0)
LABNOSESSION	1=no output to J session (default 0)
LABWIDTH	Text width (default 61)
LABWINDOWED	1=run Lab with text in a window (default 1)
LABWRAP	1=wrap text to LABWIDTH (default 1)

The title and author (if given) are shown when the lab is run.

Comments are not shown when the lab is started.

Continue after errors should be on if you intend to demonstrate errors in your J code, otherwise errors are signalled to the user. By default this is 0 (off).

Set no session output if you do not want to write to the J execution session. This may be appropriate for labs where there are no code examples; the labs may still contain code within keywords.

Run in window shows the lab text in a window. By default this is 1 (on); you may want to set this off for labs that create their own windows.

Text wrap and text width: by default text wrap is on, and means that any text output to the J execution window is wrapped to the specified text width (default 61).

The header may also contain any other required definitions, such as initialization code that is to be run at the beginning of each chapter. You can invoke any such definitions within `PREPARE` keywords in the first section of the chapter (see below). Since the header is defined in the `jlab` locale, subsequent references must use the full locale name.

Chapters

Chapters are delimited by a line beginning `Lab Chapter`, for example:

Lab Chapter Function Rank

Lab Sections

Sections

Sections are in two parts, either of which may be empty. The first part is the text to be displayed, ended by a line starting with `)`. Lines after the `)` are J sentences that are executed. If no `)` line is given, the section is assumed to be text only.

For example, the following is a lab section named “Numbers”:

```
Lab Section Numbers
The integer function (i.) generates numbers:
)
i.10          NB. first 10 numbers

i.4 3        NB. first 12 numbers in a 4 by 3 table
```

The lab system responds to Ctrl+A (advance) by reading the next section, displaying the text, and executing the J sentences. All display is normal output to the active jx window, and the user has complete access to the J session.

Section Keywords

Lines in the second part of a Lab Section (the J sentences to be executed) may begin with one of the keywords `SCRIPT`, `PREPARE` or `SOUND`.

- `SCRIPT` allows you to enter a character string that will be stored in the global variable `SCRIPT`, in the `jlab` locale. For example, this can be used to build up an example script.
- `PREPARE` allows you to enter sentences that will be run silently before the rest of the section is run.
- `SOUND` is used to play sounds, where there is an accompanying sound file, see Lab Sounds.

The `SCRIPT` and `PREPARE` keywords are used to delimit text or sentences to be run before the rest of the section, and must occur at the beginning of the section. Any text on the same line as the keyword is ignored. For example, you can use these facilities to define the global `SCRIPT`, or load required code, and check whether it is OK to continue the lab.

A typical use of `PREPARE` is when your Lab creates Windows forms. To ensure that `wd` commands are sent to the selected form, start the code with a `pselect` command, for example:

```
PREPARE
wd 'pselect myform'
PREPARE
```

To prevent further execution of the section, signal an error. The utility `assert_lab_` may be used for this purpose; the left argument is the message to display when a 0 occurs in the right argument.

For example:

Lab Section Printing

The following prints the result:

```
)  
PREPARE read in print fns -----  
load 'print'  
load 'myutils'  
ERRORMSG=. 'Unable to load myutils',LF,LF,'Check they are installed'  
ERRORMSG assert_lab_3=nameclass <'myprintfn'  
PREPARE -----  
myprint RES
```

Text Width

When using ijt (plain text) files, the recommended text width is 61 characters, which should display on all screens with typical screen fonts. You can create wider lines, but some users may have to scroll the screen in order to read them.

If LABWRAP is set on, text beginning at the left margin is automatically wrapped to width LABWIDTH when it is written to the J execution window. To avoid this, for example when including J code, indent text by one or more spaces.

Ignored Lines

The system ignores any lines beginning NB. ==.

Files created with Lab Author have sections delimited with the line below, which has the text width used by the editor:

```
NB. =====
```

Running Labs

A lab is invoked from menu item Studios|Labs, which runs the verb `lab` in the `j` locale. See Lab Program Access.

The Lab Select form has an Intro to Labs button, which runs the lab `system\extras\labs\labintro.txt`, not listed with the other labs.

The Category selection box allows you to select labs in specific directories. These are defined in script `system\extras\labs\labdir.ijs`, which you can modify as needed to include your own lab directories. Subdirectories of `system\extras\lab` are automatically included in the list of categories.

You can step through a lab by pressing `Ctrl+A`, when the `J` session has focus. A lab that creates a form may allow stepping through when the form has focus, by defining a handler for `Ctrl+A`, such as:

```
myform_actrl_fkey_z=: 3 : 'lab_j_ 0 [ wd''smselout;smfocus'''
```

You can also jump to other chapters of the lab from menu item Studios|Jump.

Lab Author

The Lab Authoring system can be used to create lab files. Load it from menu item Studio/Author.

A lab is created as a header, plus lab chapters and sections. Each section is displayed in two panes - the top pane is the text, and the lower pane is the code. Either may be empty.

Press the Run button to run the code in any section. The Lab is run in the session window, i.e. LABWINDOWED is ignored. Any errors are displayed in a message box. The code is not run automatically as you navigate through the lab.

You can step through the lab by pressing Ctrl+A when the Author form has focus. This simulates the normal running of the lab. Note that the Author form has to have the focus to advance.

To load and run the lab exactly as the user would run it, use the menu item File/Run Lab.

Note that if your Lab creates Windows forms, you need to ensure that wd commands are sent to the correct form. Start your code sections with appropriate PREPARE statements, for example:

```
PREPARE
wd 'pset myform'
PREPARE
```

The Wrap button wraps the text pane to the current width. Use Edit/UnWrap to undo; this is only available immediately after a wrap has been done. Otherwise, use Edit/Restore Section to restore the pane to its original state.

The width indicator shows the current default width. Only part of the width indicator will show if the choice of Font and Width makes it too wide to display in the window – in which case, resize the window, or reduce the Font or Width settings.

The Sounds items are enabled only when the lab has sounds. Check Enable to play the sounds when running each section. Use the Insert button to insert a sound in the code.

The menu items are mostly self-explanatory, for example:

Section/Restore Section	Restores the panes to the initial values when you moved to the section
Edit/Font...	Sets the font used in the two panes
Edit/Header...	Sets the lab header

Rich-Text

The essential difference between the ijt and rtf file is that the text part of each section in the rtf file can contain formatting such as bold, superscript and color. When the lab is run, the formatted text is displayed in a Windows form with a rich-text control.

If you close the form, the lab can be run in the session in the usual way.

The facilities available in the Windows rich-text control are those of WordPad, and are a subset of those in Word and other word processors. Formatting instructions not supported by the rich-text control are ignored.

The layout of the rtf file is exactly the same as for the ijt file. Formatting for the non-text sections is ignored, and it may be helpful to apply distinctive formatting to each section to make it easy to read.

LABWIDTH and LABWRAP are ignored by the rich-text control, except when the text window is closed and the lab run in a J session.

Lab Sounds

A lab can include sounds, for example to provide additional commentary on the text. The lab selection screen indicates which labs have sounds with [sound] following the name, and has an on/off switch. Since sound is optional, labs should work correctly without sounds.

Each section may have 0 or more sounds, interspersed with code for the section. If there is no sound, or sounds are off, all the code is run in a single step. Otherwise, Ctrl+A advances through the section by running any code up to the next sound, then playing the sound.

Sounds are read from a keyed file with extension .jf with the same name as the lab, in the same directory; thus the sound file system\extras\lab\cantor.jf matches the lab file system\extras\lab\cantor.ijt. The key names are included in the code with lines beginning SOUND , for example:

```
Lab Section Numbers
The integer function (i.) generates numbers:
)
i.10          NB. first 10 numbers
SOUND integer1
i.4 3        NB. first 12 numbers in a 4 by 3 table
SOUND integer2
SOUND integer3
```

You should create the sounds using any sound recorder, and then import the sound files into the keyed file. First create a directory containing all the wav files to be imported, then use the Lab Author menu item Sounds/Organizer to import the files. Each file is stored with a keyword of the short filename; thus c:\sounds\integer1.wav would be stored under key integer1.

The sound files are normal keyed files and can be accessed directly using the J keyed file utilities.

Lab Program Access

The verb `lab` in locale `j` runs the lab system. It is invoked by selecting the lab menu items in the Studio menu; and by pressing `Ctrl+A` when a lab has been loaded.

`lab` can also be called under program control as described below, and this can be helpful when creating and testing a lab. If you are using `lab` in this way, you may want to define `lab_z_ = .lab_j_` to remove the need for the locale reference.

The form is:

<code>lab ''</code>	Lab dialog for files in <code>system\extras\labs</code> directory. Invoked by menu item <code>Studio Labs...</code>
<code>lab 0 [,num]</code>	Show next Lab Section (or section <code>num</code>). Invoked by menu item <code>Studio Advance</code> , or by <code>Ctrl+A</code>
<code>lab 1</code>	Show jump dialog Invoked by menu item <code>Studio Jump...</code>
<code>lab 2</code>	Run Lab Author Invoked by menu item <code>Studio Author...</code>
<code>lab 'directory'</code>	Show lab dialog for files with extension <code>.ijt</code> in given directory.
<code>lab 'filename' [,num]</code>	Invoke lab on given file (at chapter <code>num</code>)

Note that you can run any file as a lab file, e.g.

```
lab 'system\extras\labs\labintro.txt'
```