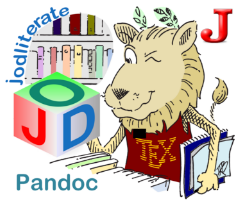
Using jodliterate



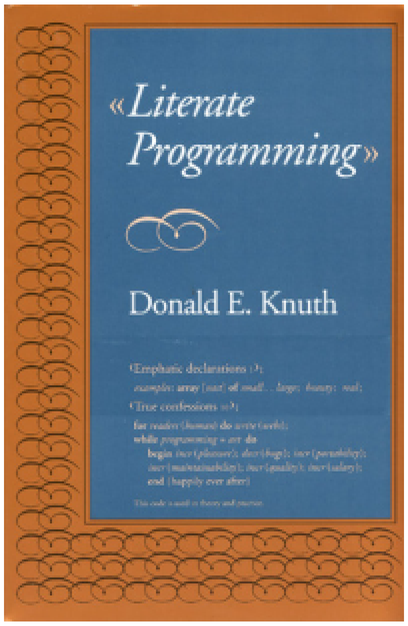
John D. Baker
https://analyzethedatanotthedrivel.org/

November 5, 2020

0.1 Using jodliterate

The JODSOURCE addon, (a part of the JOD system), contains a handy *literate programming* tool that enables the generation of *beautiful* J source code documents.

The *Bible*, *Koran*, and *Bhagavad Gita* of Literate Programming is Donald Knuth's masterful tome of the same name.



Knuth applied Literate Programming to his TEX systems and produced what many consider enduring masterpieces of program documentation.

`jodliterate` is certainly not worthy of TEX level accolades but with a little work it's possible to produce fine documents. This J kernel notebook outlines how you can install and use `jodliterate`. Jupyter notebooks are typically executed but to accommodate J users that do hot have Jupyter this notebook is also available on GitHub as a static PDF document.

Notebook Preliminaries

[1]:
```
NB. show J kernel version
9!:14 ''
```

```
j902/j64avx2/windows/beta-k/commercial/www.jsoftware.com/2020-11-03T10:24:54/cla
ng-9-0-0/SLEEF=1
```

[2]:
```
NB. load JOD in a clear base locale
load 'general/jod' [ clear ''

NB. The distributed JOD profile automatically RESETME's.
NB. To safely use dictionaries with many J tasks they must
NB. be READONLY. To prevent opening the same put dictionary
NB. READWRITE comment out (dpset) and restart this notebook.
dpset 'RESETME'

NB. Converting Jupyter notebooks to LaTeX is
NB. simplified by ASCII box characters.
portchars ''

NB. Verb to convert character tables to newline delimited
NB. lists. Useful for displaying J tables in Jupyter
ctl_ijod_=:}.@(,@(1&(,"1)@(-.@(*./\."1@(=&' '@])))) # ,@((10{a.)&(,"1)@]))

NB. Appends line feed character if necessary.
tlf_ijod_=:] , ((10{a.)"_ = {:) }. (10{a.)"_

NB. Verb to show large boxed displays in
NB. the notebook without ugly wrapping.
sbx_ijod_=: ' ... ' ,"1~ 75&{."1@":
```

Installing jodliterate To use `jodliterate` you need to:

1. Install a current version of J.
2. Install the J addons JOD, JODSOURCE, and JODDOCUMENT.
3. Build the JOD development dictionaries from JODSOURCE.
4. Install a current version of pandoc.
5. Install a current version of TEX and LATEX.
6. Make the `jodliterate` J script.
7. Run `jodliterate` on a JOD *group* with pandoc compatible document fragments.
8. Compile the files of the previous step to produce a PDF

When presented with long lists of program prerequisites my impulse is to *run!* Life is too short for configuration wars. Everything should be easy. Installing `jodliterate` requires more work than phone apps but compared to enterprise installations setting up `jodliterate` is trivial. We'll go through it step by step.

Step 1: Install a current version of J J is freely available at jsoftware.com. J installation instructions can be found on the J Wiki on this page.

Follow the appropriate instructions for your OS.

Note: JOD runs on Windows, Linux, and MacOS versions of J, hence these are the only platforms that currently support `jodliterate`.

Step 2: Install the J addons JOD, JODSOURCE and JODDOCUMENT After installing J install the J addons. J addons are installed with the J package manager pacman. Pacman has three IDE flavors: a command-line flavor and two GUI flavors. The GUI flavors depend on JQT or JHS. The GUI flavors of pacman are only available on some versions of J whereas the command line version is part of the base J install and is available on all platforms.

I install all the addons. I recommend that you do the same.

JOD depends on some J modules like `jfiles`, `regex`, and `task` that are sometimes distributed as addons. If you install all addons JOD's modules and dependents are both installed.

Installing addons with command line pacman Start J and do:

[3]:
```
NB. install J addons with command-line pacman

load 'pacman'    NB. load pacman jpkg services
```

[4]:
```
'help' jpkg ''   NB. what can you do for me?
```

```
Valid options are:
 history, install, manifest, remove, reinstall, search,
 show, showinstalled, shownotinstalled, showupgrade,
 status, update, upgrade

https://code.jsoftware.com/wiki/JAL/Package_Manager/jpkg
```

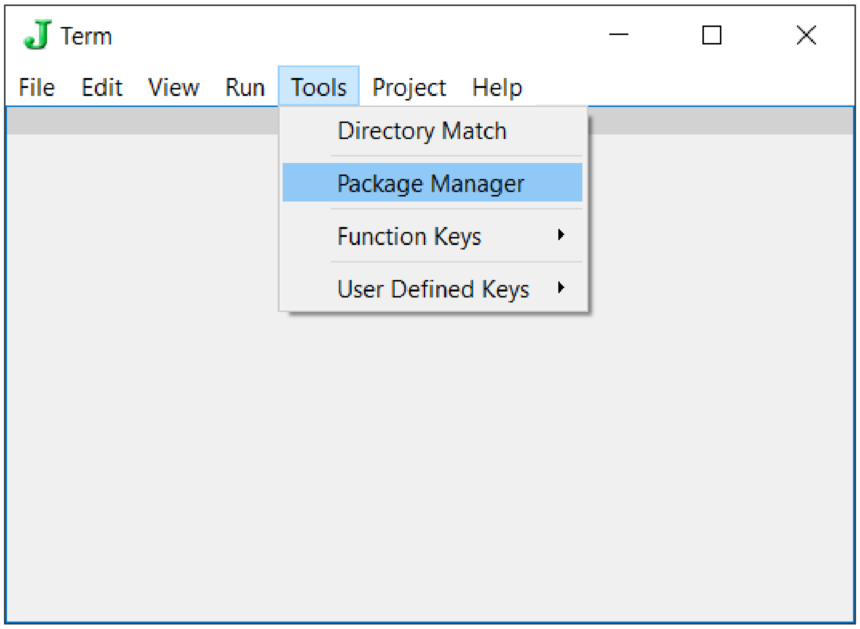
[5]:
```
NB. install all addons
NB. see https://code.jsoftware.com/wiki/Pacman

NB. uncomment next line if addons not installed
NB. 'install' jpkg '*'  NB.
```

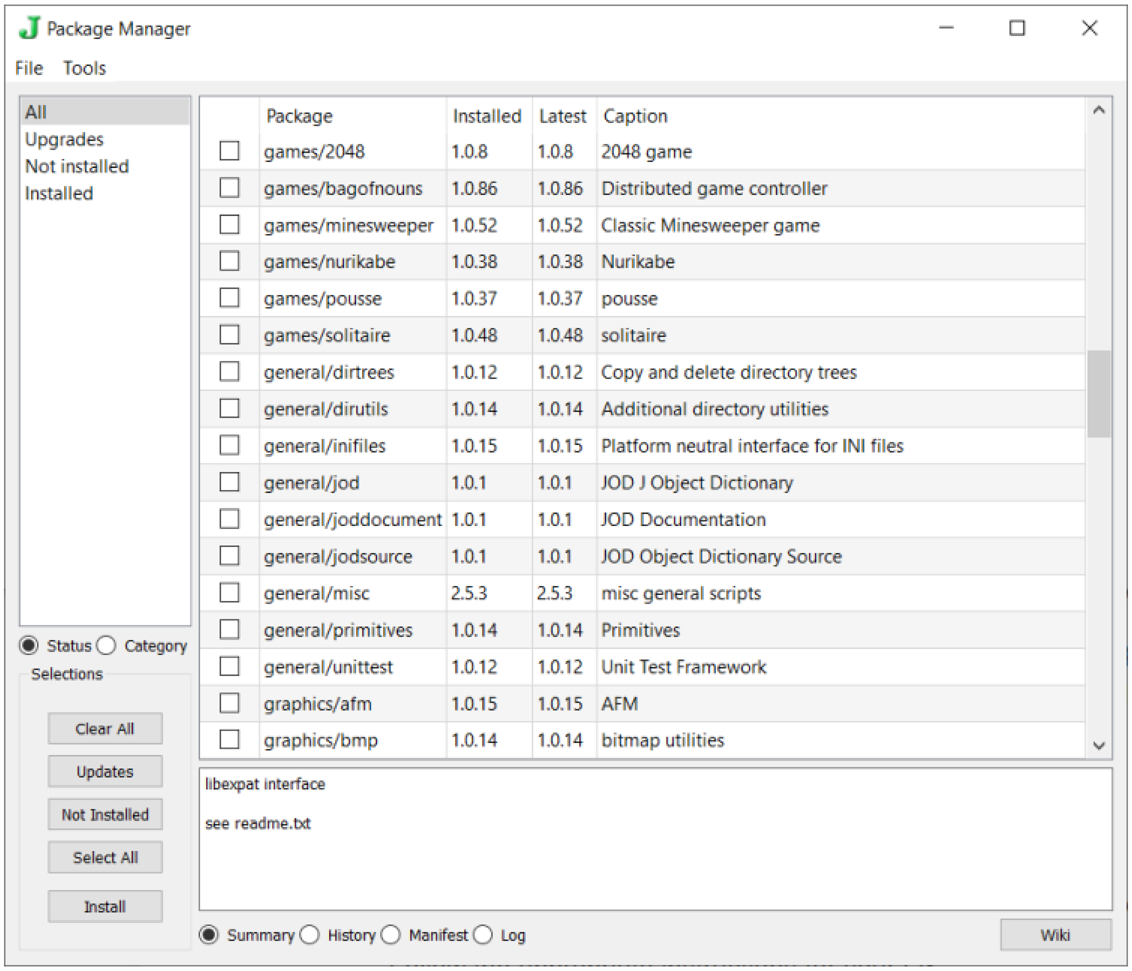
[6]:
```
3 {. 'showinstalled' jpkg '' NB. first few installed addons
```

```
+---------+------+------+----------------------------+
|api/expat|1.0.11|1.0.11|libexpat                    |
+---------+------+------+----------------------------+
|api/gles |1.0.31|1.0.31|Modern OpenGL API           |
+---------+------+------+----------------------------+
|api/java |1.0.2 |1.0.2 |api: Java to J shared library|
+---------+------+------+----------------------------+
[7]: 'showupgrade' jpkg ''   NB. list addon updates
```

Installing addons with JQT GUI pacman I mostly use the Windows JQT version of pacman to install and maintain J addons. You can find pacman on the tools menu.



pacman shows all available addons and provides tools for installing, updating, and removing them.



	Package	Installed	Latest	Caption
☐	games/2048	1.0.8	1.0.8	2048 game
☐	games/bagofnouns	1.0.86	1.0.86	Distributed game controller
☐	games/minesweeper	1.0.52	1.0.52	Classic Minesweeper game
☐	games/nurikabe	1.0.38	1.0.38	Nurikabe
☐	games/pousse	1.0.37	1.0.37	pousse
☐	games/solitaire	1.0.48	1.0.48	solitaire
☐	general/dirtrees	1.0.12	1.0.12	Copy and delete directory trees
☐	general/dirutils	1.0.14	1.0.14	Additional directory utilities
☐	general/inifiles	1.0.15	1.0.15	Platform neutral interface for INI files
☐	general/jod	1.0.1	1.0.1	JOD J Object Dictionary
☐	general/joddocument	1.0.1	1.0.1	JOD Documentation
☐	general/jodsource	1.0.1	1.0.1	JOD Object Dictionary Source
☐	general/misc	2.5.3	2.5.3	misc general scripts
☐	general/primitives	1.0.14	1.0.14	Primitives
☐	general/unittest	1.0.12	1.0.12	Unit Test Framework
☐	graphics/afm	1.0.15	1.0.15	AFM
☐	graphics/bmp	1.0.14	1.0.14	bitmap utilities

libexpat interface

see readme.txt

The GUI version is easy to use. Press the `Select All` button and then press the `Install` button to install all the addons. To update addons select the `Upgrades` menu and select the addons you want to update.

Step 3: Build the JOD development dictionaries from JODSOURCE JOD source code is distributed in the form of JOD dictionary dumps. Dictionary dumps are large J scripts that serialize JOD dictionaries. Dumps contain everything stored in dictionaries. You will find source code, binary data, test scripts, documentation, build macros, and more in typical JOD dictionaries.

`jodliterate` is stored as a JOD dictionary group. A dictionary group is simply a collection of J words with optional *header* and *post-processor* scripts. JOD generates J scripts from groups. Before we can *make* `jodliterate` we must load the JOD development dictionaries. The JODSOURCE addon includes a J script that loads development dictionaries.

Again, start J and do:

[8]:
```
require 'general/jod'
```

[9]:
```
NB. set a JODroot user folder
NB. if not set /jod/ is the default
NB. use paths for your OS
UserFolders_j_=: UserFolders_j_ , 'JODroot';'c:/temp'

NB. show added folder
UserFolders_j_ {~ (0 {"1 UserFolders_j_) i. <'JODroot'
```

```
+-------+-------+
|JODroot|c:/temp|
+-------+-------+
```

[10]:
```
NB. load JOD developement dictionaries
load_dev_tmp=: 3 : 0
if. +./ (;:'joddev jod utils') e. od '' do.
  'dev dictionaries exist'
else.
  0!:0<jpath'~addons/general/jodsource/jodsourcesetup.ijs'
end.
)

load_dev_tmp 0
```

```
dev dictionaries exist
```

[11]:
```
NB. joddev, jod, utils should exist

erase 'load_dev_tmp'
(;:'joddev jod utils') e. od ''
```

```
1 1 1
```

Step 4: Install a current version of pandoc pandoc is easily one of the most useful markup utilities on the intertubes. If you routinely deal with markup formats like markdown, XML, LaTeX, json and you aren't using pandoc you are working too hard.

Be lazy! Install pandoc.

`jodliterate` uses the `task` addon to *shell out* to pandoc. Versions of pandoc after `2.9.1.1` support J syntax high-lighting.

[12]:
```
NB. show pandoc version from J - make sure you are running
NB. a recent version of pandoc. There may be different
NB. versions in many locations on various systems.

NB. some common paths to pandoc
NB. PREFERREDPANDOC_ijod_=: '"C:\Program Files\Pandoc\pandoc"'
PREFERREDPANDOC_ijod_=: '"C:\Users\john.baker\AppData\Local\Pandoc\pandoc"'

NB. use when correct version is on the shell path
NB. PREFERREDPANDOC_ijod_ 'pandoc'

NB. the pandoc jodliterate uses
THISPANDOC_ajodliterate_=: PREFERREDPANDOC_ijod_

chkpandoc=: 3 : 0
if. +./@('pandoc'&E.) panver=. ;0{ <;._2 tlf (shell THISPANDOC_ajodliterate_,'␣
↪--version') -. CR do.
  'NOTE: adjust pandoc path if version (',panver,') is not >= 2.9.1.1'
else.
  'ERROR: pandoc not set - adjust THISPANDOC_ajodliterate_'
  'THISPANDOC_ajodliterate_=: ''pandoc'' NB. when pandoc on path'
end.
)

smoutput shell THISPANDOC_ajodliterate_,' --version'
chkpandoc 0
```

```
pandoc 2.9.1.1
Compiled with pandoc-types 1.20, texmath 0.12, skylighting 0.8.3
Default user data directory: C:\Users\john.baker\AppData\Roaming\pandoc
Copyright (C) 2006-2019 John MacFarlane
Web:  https://pandoc.org
This is free software; see the source for copying conditions.
There is no warranty, not even for merchantability or fitness
for a particular purpose.

NOTE: adjust pandoc path if version (pandoc 2.9.1.1) is not >= 2.9.1.1
```

[13]:
```
NB. make sure your version of pandoc
NB. supports J syntax-highlighting

NB. check that J is on the supported languages list
pcmd=: THISPANDOC_ajodliterate_,' --list-highlight-languages'
ctl 80 list shell pcmd
```

```
abc           asn1          asp           ats           awk
actionscript  ada           agda          alertindent   apache
bash          bibtex        boo           c             cs
cpp           cmake         css           changelog     clojure
coffee        coldfusion    commonlisp    curry         d
dtd           default       diff          djangotemplate dockerfile
doxygen       doxygenlua    eiffel        elixir        elm
email         erlang        fsharp        fortran       gcc
glsl          gnuassembler  m4            go            html
hamlet        haskell       haxe          ini           isocpp
idris         fasm          nasm          j             json
```

jsp	java	javascript	javascriptreact	javadoc
julia	kotlin	llvm	latex	lex
lilypond	literatecurry	literatehaskell	lua	mips
makefile	markdown	mathematica	matlab	maxima
mediawiki	metafont	modelines	modula2	modula3
monobasic	mustache	ocaml	objectivec	objectivecpp
octave	opencl	php	povray	pascal
perl	pike	postscript	powershell	prolog
protobuf	pure	purebasic	python	qml
r	relaxng	relaxngcompact	roff	ruby
rhtml	rust	sgml	sml	sql
sqlmysql	sqlpostgresql	scala	scheme	stata
tcl	tcsh	texinfo	mandoc	typescript
vhdl	verilog	xml	xul	yaml
yacc	zsh	dot	noweb	rest
sci	sed	xorg	xslt	

Step 5: Install a current version of LaTeX `jodliterate` uses LaTeX to compile PDF documents. When `setjodliterate` runs it sets an output directory and writes a LaTeX preamble file `JODLiteratePreamble.tex` to it. It's a good idea to review this file to get an idea of the LaTeX packages `jodliterate` uses. It's possible that some of these packages are not in your LaTeX distribution and will have to be installed.

To ease the burden of LaTeX package maintenance I use freely available TeX versions that automatically install missing packages.

1. On Windows I use MiKTeX
2. On other platforms I use TeXLive

If your system automatically installs packages the first time you compile `jodliterate` output it may fetch missing packages from The Comprehensive TeX Archive Network (CTAN). If new packages are installed reprocess your files a few times to insure all the required packages are downloaded and installed.

Step 5.5: Use an online version of LaTeX If you don't want to bother with installing and maintaining a LaTeX system you can use online systems like OverLeaf.com. If you opt for OverLeaf.com you will have to copy the files `jodliterate` generates to and from OverLeaf.com. OverLeaf.com integrates with GitHub so ferrying copies is not an onerous chore.

Here's some `jodliterate` files on Overleaf.com

Step: 6 Make the jodliterate J script Once the JOD development dictionaries are built (Step 3) making `jodliterate` is easy. Start J and do:

[14]:
```
require 'general/jod'

NB. open dictionaries
od ;:'joddev jod utils' [ 3 od ''
```

```
+-+--------------------+------+---+-----+
|1|opened (rw/ro/ro) ->|joddev|jod|utils|
+-+--------------------+------+---+-----+
```

[15]:
```
NB. generate jodliterate
sbx mls 'jodliterate'
```

```
+-+--------------------+--------------------------------------------- ...
|1|load script saved ->|c:/users/john.baker/onedrive - jackson companies/jo ...
+-+--------------------+--------------------------------------------- ...
```

mls creates a standard J load script. Once generated this script can be loaded with the standard J load utility. You can test this by restarting J without JOD and loading jodliterate.

[16]:
```
NB. load generated script
load 'jodliterate'
```

```
NB. (jodliterate) interface word(s):
NB. --------------------------------
NB. THISPANDOC      NB. full pandoc path - use (pandoc) if on shell path
NB. formifacetex    NB. formats hyperlinked and highlighted interface words
NB. grplit          NB. make latex for group (y)
NB. ifacesection    NB. interface section summary string
NB. ifc             NB. format interface comment text
NB. setjodliterate  NB. prepare LaTeX processing - sets out directory writes
preamble
NB. wordlit         NB. make latex from word list (y)

NOTE: adjust pandoc path if current version (pandoc 2.9.1.1) is not >= 2.9.1.1
```

Step 7: Run jodliterate on a JOD group with pandoc compatible document fragments This sounds a lot worse than it is. There is a group in utils called sunmoon that has an interesting *pandoc compatible document fragment*.

Start J and do:

[17]:
```
require 'general/jod'

od 'utils' [ 3 od ''
```

```
+-+--------------+-----+
|1|opened (ro) ->|utils|
+-+--------------+-----+
```

[18]:
```
NB. display short explanations for (sunmoon) words
sbx hlpnl }. grp 'sunmoon'
```

```
+----------------+--------------------------------------------------- ...
|IFACEWORDSsunmoon|interface words (IFACEWORDSsunmoon) group          ...
```

```
|NORISESET        |indicates sun never rises or sets in (sunriseset0) and ( ...
|ROOTWORDSsunmoon |root words (ROOTWORDSsunmoon) group                      ...
|arctan           |arc tangent                                              ...
|calmoons         |calendar dates of new and full moons                     ...
|cos              |cosine radians                                           ...
|fromjulian       |converts Julian day numbers to dates, converse (tojulian ...
|moons            |times of new and full moons for n calendar years         ...
|round            |round (y) to nearest (x) (e.g. 1000 round 12345)         ...
|sin              |sine radians                                             ...
|sunriseset0      |computes sun rise and set times - see group documentatio ...
|sunriseset1      |computes sun rise and set times - see group documentatio ...
|tabit            |promotes only atoms and lists to tables                  ...
|tan              |tan radians                                              ...
|today            |returns todays date                                      ...
|yeardates        |returns all valid dates for n calendar years             ...
+-----------------+-------------------------------------------------------- ...
```

[19]:

```
NB. display part of the (sunmoon) group document header
NB. this is pandoc compatible markdown - note the LaTeX
NB. commands - pandoc allows markdown/LaTeX mixtures
900 {. 2 9 disp 'sunmoon'
```

```
`sunmoon` is a collection of basic astronomical algorithms
The key verbs are `moons`, `sunriseset0` and `sunriseset1.`
All of these verbs were derived from BASIC programs published
in *Sky & Telescope* magazine in the 1990's. The rest of
the verbs in `sunmoon` are mostly date and trigonometric
utilities.

\subsection{\texttt{sunmoon} Interface}

~~~~ { .j }
  calmoons      NB. calendar dates of new and full moons
  moons         NB. times of new and full moons for n calendar years
  sunriseset0   NB. computes sun rise and set times - see group documentation
  sunriseset1   NB. computes sun rise and set times - see group documentation
~~~~

\subsection{\textbf\texttt{sunriseset0} \textsl{v--} sunrise and sunset times}

This  verb has been adapted from a BASIC program submitted by
Robin  G.  Stuart  *Sky & Telescope's*  shortest  sunrise/set
program  cont
```

[20]:

```
NB. run jodliterate on (sunmoon)
require 'jodliterate'
```

```
NB. set the output directory - when running in Jupyter
NB. use a subdirectory of your notebook directory other
NB. directories generate access errors - you really don't
NB. want web browsers roaming your drives unsupervised

NB. ltxpath=: 'C:\Users\john\AnacondaProjects\testfolder\grplit\'
ltxpath=: 'C:\Users\john.baker\bixml\blog\grplit\'

NB. (x) argument sets LaTeX \author{} text
'Batman (\texttt{dn@jl.com})' setjodliterate ltxpath
```

```
+-+--------------------------------------+
|1|C:\Users\john.baker\bixml\blog\grplit\|
+-+--------------------------------------+
```

[21]:
```
NB. (grplit) returns a list of generated
NB. LaTeX and command files. The *.bat
NB. file compiles the generated LaTeX

,. grplit 'sunmoon'
```

```
+-----------------------------------------------------+
|1                                                    |
+-----------------------------------------------------+
|C:\Users\john.baker\bixml\blog\grplit\sunmoon.tex    |
+-----------------------------------------------------+
|C:\Users\john.baker\bixml\blog\grplit\sunmoontitle.tex|
+-----------------------------------------------------+
|C:\Users\john.baker\bixml\blog\grplit\sunmoonoview.tex|
+-----------------------------------------------------+
|C:\Users\john.baker\bixml\blog\grplit\sunmooncode.tex |
+-----------------------------------------------------+
|C:\Users\john.baker\bixml\blog\grplit\sunmoon.bat    |
+-----------------------------------------------------+
```

Step 8: Compile the files of the previous step to produce a PDF

[22]:
```
_250 {. shell ltxpath,'sunmoon.bat'
```

```
/?/c:/users/john.baker/appdata/local/progr
ams/miktex 2.9/fonts/opentype/public/lm/lmmono12-regular.otf>
Output written on sunmoon.pdf (22 pages, 114284 bytes).
Transcript written on sunmoon.log.

C:\Users\john.baker\bixml\blog\grplit>endlocal
```

Instead of compiling LaTeX with shell commands issued from a J kernel running under Jupyter it's easier to navigate to the output directory and manually run the generated scripts. This is particularily the case if you have to edit the files. From the good old fashioned DOS prompt do something like:

```
C:\>cd C:\Users\john.baker\bixml\blog\grplit
C:\Users\john.baker\bixml\blog\grplit>sunmoon.bat
```

[23]:
```
NB. uncomment to display generated PDF
 NB. shell ltxpath,'sunmoon.pdf'
```

Storing jodliterate pandoc compatible document fragments in JOD Effective use of `jodliterate` requires a melange of Markdown, LaTeX, JOD, and J skills combined with a healthy attitude about *experimentation*. You have to try things and see if they work!

However, before you can *try* `jodliterate` you have to `put` document fragments in JOD dictionaries.

`jodliterate` uses two types of document fragments:

1. Markdown overview group documents: `2 9 put 'groupname'`
2. LaTeX overview macros: `4 put 'groupname','_oview_tex'`

Markdown group documents are transformed by pandoc into LaTeX but the overview macros are not altered in any way. This enables the use of arbitrarily complex LaTeX. The following examples show how to insert document fragments.

Create a jodliterate Demo Dictionary

[24]:
```
NB. create a demo dictionary - (didnum) insures new name
require 'general/jod'

NB. new dictionary in default JOD directory
sbx newd itslit_ijod_=: 'aaa',":didnum_ajod_ ''
```

```
+-+--------------------+------------------------------------------+------- ...
|1|dictionary created ->|aaa13803283056015167423528139558147326172 2|c:/user ...
+-+--------------------+------------------------------------------+------- ...
```

[25]:
```
NB. 1 if new dictionary created
(<itslit) e. od ''
```

```
1
```

[26]:
```
od itslit [ 3 od '' NB. open only new dictionary
```

```
+-+-------------+------------------------------------------+
|1|opened (rw) ->|aaa138032830560151674235281395581473261722|
+-+-------------+------------------------------------------+
```

[27]:
```
NB. define some words
freq=:~. ; #/.~
```

```
movmean=:-@[ (+/ % #)\ ]
geomean=:# %: */
bmi=: 704.5"_ * ] % [: *: [
polyprod=:+//.@(*/)

wlst=: ;:'freq movmean geomean bmi polyprod'

NB. put in dictionary
put wlst

NB. short word explanations
t=: ,:  'freq';'frequency distribution'
t=: t , 'movmean';'moving mean'
t=: t , 'geomean';'geometric mean of a list'
t=: t , 'bmi';'body mass index - (x) inches (y) lbs'
t=: t , 'polyprod';'polynomial product'

0 8 put t
```

```
+-+-----------------------------+-----------------------------------------+
|1|5 word explanation(s) put in ->|aaa1380328305601516742352813955814732617 22|
+-+-----------------------------+-----------------------------------------+
```

[28]:
```
NB. make header and macro groups
grp 'litheader' ; wlst
grp 'litmacro'  ; wlst
```

```
+-+-------------------------+-----------------------------------------+
|1|group <litmacro> put in ->|aaa138032830560151674235281395581473261722|
+-+-------------------------+-----------------------------------------+
```

[29]:
```
IFACEWORDSlitheader=: wlst
put 'IFACEWORDSlitheader'
```

```
+-+------------------+-----------------------------------------+
|1|1 word(s) put in ->|aaa138032830560151674235281395581473261722|
+-+------------------+-----------------------------------------+
```

Use Group Document Overview Markdown

[30]:
```
NB. add group header markdown
litheader=: (0 : 0)
`litheader` is a markdown demo group.

This markdown text will be
[transmogrified](https://calvinandhobbes.fandom.com)
by `pandoc` to \LaTeX. A group interface will be
generated from the `IFACEWORDSlitheader`
```

```
list. Interface lists are usually, but
not always, associated with a *class group*.

\subsection{\texttt{litheader} Interface}

`{~{insert_interface_md_}~}`
)

NB. store markdown as a JOD group document
2 9 put 'litheader';litheader
```

```
+-+----------------------------+-----------------------------------------+
|1|1 group document(s) put in ->|aaa138032830560151674235281395581473261722|
+-+----------------------------+-----------------------------------------+
```

[31]:
```
NB. run jodliterate on group
setjodliterate ltxpath
{: grplit 'litheader'
```

```
+----------------------------------------------------+
|C:\Users\john.baker\bixml\blog\grplit\litheader.bat|
+----------------------------------------------------+
```

[32]:
```
NB. compile latex
_250 {. shell ltxpath,'litheader.bat'
```

```
/c:/users/john.baker/appdata/local/programs/miktex 2.9/fonts/opentype
/public/lm/lmmono12-regular.otf>
Output written on litheader.pdf (4 pages, 50907 bytes).
Transcript written on litheader.log.

C:\Users\john.baker\bixml\blog\grplit>endlocal
```

[33]:
```
NB. uncomment to show PDF
NB. shell ltxpath,'litheader.pdf'
```

Use Macro Overview LaTeX

[34]:
```
NB. add a LaTeX overview - this code will not
NB. be altered by jodliterate the suffix
NB. '_oview_tex' is required to associate
NB. the overview with the group 'litmacro'

litmacro_oview_tex=: (0 : 0)

This \LaTeX\ code will not be
touched by \texttt{jodliterate}.
```

```
\subsection{Business Babel}

``Truth management is enabled.''

\emph{Excerpt from an actual business document!}
Obviously composed in an irony free zone.

\subsection{Some Complicated \LaTeX}

\medskip

\[
\frac{1}{\Bigl(\sqrt{\phi \sqrt{5}}-\phi\Bigr) e^{\frac25 \pi}} =
1+\frac{e^{-2\pi}} {1+\frac{e^{-4\pi}} {1+\frac{e^{-6\pi}}
{1+\frac{e^{-8\pi}} {1+\ldots} } } }
\]

)

NB. store LaTeX as JOD text macro
4 put 'litmacro_oview_tex';LATEX_ajod_;litmacro_oview_tex
```

```
+-+--------------------+------------------------------------------+
|1|1 macro(s) put in ->|aaa13803283056015167423528139558147326172 2|
+-+--------------------+------------------------------------------+
```

[35]:
```
NB. run jodliterate on group
{: grplit 'litmacro'
```

```
+--------------------------------------------------+
|C:\Users\john.baker\bixml\blog\grplit\litmacro.bat|
+--------------------------------------------------+
```

[36]:
```
NB. compile latex
_250 {. shell ltxpath,'litmacro.bat'
```

```
lm/lmsy6.p
fb><C:/Users/john.baker/AppData/Local/Programs/MiKTeX 2.9/fonts/type1/public/lm
/lmsy8.pfb>
Output written on litmacro.pdf (4 pages, 141689 bytes).
Transcript written on litmacro.log.

C:\Users\john.baker\bixml\blog\grplit>endlocal
```

[37]:
```
NB. display PDF
NB. shell ltxpath,'litmacro.pdf'
```

Using jodliterate with larger J systems The main `jodliterate` verb `grplit` works with single JOD groups. Larger systems are typically made from many groups. JOD macro and test scripts are one way to work around this limitation. The JOD development dictionaries contain several macros that illustrate this approach.

[38]:
```
od ;:'joddev jod utils' [ 3 od ''

NB. list macros with substring 'latex'
4 2 dnl 'latex'
```

```
+-+-------------+---------------------+
|1|buildjodlatex|buildjodliteratelatex|
+-+-------------+---------------------+
```

[39]:
```
NB. display start of macro that
NB. applies jodliterate to JOD code
250 {. 4 disp 'buildjodlatex'
```

```
NB.*buildjodlatex s--  generates syntax highlighted JOD source LaTeX.
NB.
NB. Files are written to the put dictionary's document directory.
NB.
NB. assumes: current versions of pandoc (pandoc 2.9.1.1 or later)
NB.          check noun (THISPANDOC
```

Final Remarks `jodliterate` is an idiosyncratic anal-retentive software utility; it's mainly for people that consider source code an art form. *Nobody likes ugly undocumented art!*

If you have any questions, suggestions, or complaints please leave a comment on this post. To include others join one of J discussion forums and post your queries there.

May the source be with you!