jodliterate Group

John D. Baker

December 29, 2013

Contents

jodliterate Overview

`jodliterate` is a J utility script that generates *literate* documents directly from JOD dictionary groups.

jodliterate Interface

```
grplit         make latex for group (y)
setjodliterate prepare for processing
ifacesection   interface section summary string
```

jodliterate and JOD

`jodliterate` makes some assumptions about the J code and document text it processes. It assumes:

1. All source code is stored in JOD dictionaries.
2. JOD group documents `2 9 disp 'groupname'` are pandoc compatible markdown text fragments. *This is the most serious limitation of* ***`jodliterate`****.* Most of my group documents are not markdown. I am slowly converting them. If you compose group documents use pandoc markdown and `jodliterate` will be a lot more useful.
3. Any group class[1] interface is a simple J verbatim block that is marked with a LaTeX section command. Examine this group's document `2 9 disp 'jodliterate'` for an example.

[1] A *group class* is a JOD group with a declared list of interface and root words: `IFACEWORDS`*groupname* and `ROOTWORDS`*groupname*. Root words allow `allrefs` to find all the words that belong to the group. Class groups typically load into locales. `jodliterate.ijs` is an example of a class group.

```
% latex section
\subsection{\texttt{groupname} Interface}

~~~~ { .j}
grplit          NB. make latex for group (y)
setjodliterate  NB. prepare for processing
~~~~~
```

4. Non-J code is confined to the JOD group header[2] and is marked with pseudo-pandoc delimiters. Everything in curly {} brackets follows pandoc source code highlighting conventions. `jodliterate`'s header contains LaTeX code that is marked like this.

```
NB.<<~~~~ { .latex numberLines startFrom="1" }
NB.>>~~~~
```

Running `jodliterate`

`jodliterate` runs on Windows, Linux and Mac versions of J.[3]

To use `jodliterate` you must:

1. Install JOD. JOD is a J addon. It is usually installed with JAL.
2. Install JODSOURCE. JODSOURCE is also a J addon. Install it with JAL.[4] It contains the JOD dictionaries required to build JOD. JODSOURCE also contains `jodliterate`.

[2] This limitation may be eased if it becomes a hindrance.
[3] `jodliterate` will run on Windows versions of J 6.02 and beyond. The Linux and Mac versions of `jodliterate` require J 7.01 and beyond.
[4] On 64 bit versions of J the standard JAL install of JODSOURCE may fail. See The Jod Page for an alternate install method.

3. Setup a version of pandoc that supports J syntax highlighting. See the blog post Pandoc based J Syntax Highlighting for details.

4. Build `jodliterate`. `jodliterate` is a J group. JOD makes J scripts from groups. To make `jodliterate` do:

```
require 'general/jod'
od ;:'joddev jod utils' [ 3 od ''
mls 'jodliterate'
```

5. Open the JOD dictionaries containing the group you want to document.

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

6. Load `jodliterate`. After making `jodliterate` it can be loaded like any J script.

```
load 'jodliterate'
```

7. Set a working directory. `jodliterate` generates markdown and LaTeX files. All such files are written to the directory specified by `setjodliterate`.

```
setjodliterate 'c:\temp'          NB. windows
setjodliterate '/home/john/temp'  NB. linux
```

8. Run `grplit` on the group you want to document. `grplit` generates LaTeX files in the working directory. The root LaTeX file is given the group name.

```
grplit 'jodliterate' NB. makes jodliterate.tex
```

9. Use LaTeX to compile the files generated by `grplit`. Files may be processed with `pdflatex`, `xelatex` or `lualatex`. Your choice will be dictated by the presence of Unicode characters. For more details see the preamble file `JODLiteratePreamble.tex` in the working directory.

There are some LaTeX settings you may want to adjust.

1. Edit `JODLiteratePreamble.tex` to change the number of index columns. The default is four but this may result in long names running together. To decrease or increase index columns change:

   ```
   \begin{multicols}{4}[\section*{\indexname}]
   ```

2. `jodliterate` wraps long source code lines. Wrapping is controlled by `WRAPLIMIT`. The default is 110 characters. Change this value if you change font or page size. Wrapped lines are preceeded with the string `WRAPLEAD` with default `>...>`.

jodliterate Source Code

```
NB.*jodliterate s-- generates literate source code documents directly from JOD groups.
NB.
NB. verbatim:
NB.
NB. http://bakerjd99.wordpress.com/2012/10/01/semi-literate-jod/
NB.
NB. interface word(s):
NB. ------------------------------------------------------------------------
NB. grplit          make latex for group (y)
NB. ifacesection    interface section summary string
NB. setjodliterate  prepare for processing
NB.
NB. author:  John D. Baker
NB. created: 2012oct01
NB. ------------------------------------------------------------------------
NB. 12oct03 (x) grplit argument added to suppress root tex overwrites
NB. 12oct04 group IFACEWORDSgroupname hyperlinked
NB. 12oct05 replaced ;: parsing with (wfl) - handles bad j code
NB. 12oct08 added error handling - replaced (write) with (writeas)
NB. 12oct11 adjusted LaTeX preamble - changing monofonts
NB. 12oct12 added (sbtokens) - useful for analyzing code text
NB. 12oct17 added (wrapvrblong) - long source lines now wrapped
NB. 13dec29 added to (jacks) GitHub repository

coclass  'ajodliterate'
```

```
coinsert 'ijod'

NB.*dependents
NB. declared global here to avoid confusing LaTeX names with J names
NB. (*)=: JLTITLETEX JLOVIEWTEX JLGRPLITTEX JODLiteratePreamble

NB. Roger Hui's word formation state machine - similiar to ;: but
NB. parses text with LFs, retains whitespace and handles open quotes.

NB. hide script locals !(*)=. mfl sfl
mfl=. 256$0                          NB. X other
mfl=. 1   (9,a.i.' ')         }mfl   NB. S whitespace (space and horizontal tab)
mfl=. 2   ((a.i.'Aa')+/i.26)  }mfl   NB. A A-Z a-z excluding N B
mfl=. 3   (a.i.'N')           }mfl   NB. N the letter N
mfl=. 4   (a.i.'B')           }mfl   NB. B the letter B
mfl=. 5   (a.i.'0123456789_') }mfl   NB. 9 digits and _
mfl=. 6   (a.i.'.')           }mfl   NB. D .
mfl=. 7   (a.i.':')           }mfl   NB. C :
mfl=. 8   (a.i.'''')          }mfl   NB. Q quote
mfl=. 9   (13)                }mfl   NB. CR
mfl=. 10  (10)                }mfl   NB. LF

sfl=. _2]\"1 }."."."."."._2 (0 : 0)
' X     S    A    N    B    9    D    C    Q    CR     LF   ']0
 1 1   12 1   2 1   3 1   2 1   6 1   1 1   1 1   7 1   10 1    1 1    NB. 0  initial
 1 2   12 2   2 2   3 2   2 2   6 2   1 0   1 0   7 2   10 2    1 2    NB. 1  other
 1 2   12 2   2 0   2 0   2 0   2 0   1 0   1 0   7 2   10 2    1 2    NB. 2  alp/num
```

```
 1 2  12 2  2 0  2 0  4 0  2 0  1 0  1 0  7 2  10 2   1 2   NB. 3  N
 1 2  12 2  2 0  2 0  2 0  2 0  5 0  1 0  7 2  10 2   1 2   NB. 4  NB
 9 0   9 0  9 0  9 0  9 0  9 0  1 0  1 0  9 0  10 2   1 2   NB. 5  NB.
 1 4  13 0  6 0  6 0  6 0  6 0  6 0  1 0  7 4  10 2   1 2   NB. 6  num
 7 0   7 0  7 0  7 0  7 0  7 0  7 0  7 0  8 0  10 2   1 2   NB. 7  '
 1 2  11 2  2 2  3 2  2 2  6 2  1 2  1 2  7 0  10 2   1 2   NB. 8  ''
 9 0   9 0  9 0  9 0  9 0  9 0  9 0  9 0  9 0  10 2   1 2   NB. 9  comment
 1 2  11 2  2 2  4 2  2 2  6 2  1 2  1 2  7 2  10 2  11 0   NB. 10 CR
 1 2  11 2  2 2  4 2  2 2  6 2  1 2  1 2  7 2  10 2   1 2   NB. 11 CRLF
 1 2  12 0  2 2  3 2  2 2  6 0  1 2  1 2  7 2  10 2   1 2   NB. 12 space
 1 2  13 0  2 2  3 2  2 2  6 0  1 2  1 2  7 2  10 2   1 2   NB. 13 space after num
)

NB. word formation for lines
wfl=: (0;sfl;mfl) & ;:

JLDIRECTORY=: ''
NB.*enddependents


NB.<<~~~~ { .latex }

NB. group title and author - standard \maketitle
JLTITLETEX=: 0 : 0

% latex author and title
\author{~#~author~#~}
```

```
\title{\texttt{~#~group~#~} Group}
)

NB. group overview header
JLOVIEWTEX=: 0 : 0

% this jodliterate overview
\section{\texttt{~#~group~#~} Overview}
)

NB. group root tex - columns may need adjusting
JLGRPLITTEX=: 0 : 0

% Main jodliterate (grplit) latex file. (grplit) generates "group"
% named versions of this file for each JOD group it processes.

\input{JODLiteratePreamble.tex}

\input{~#~group~#~title.tex}
\maketitle
\tableofcontents

\newpage
% commands for adjusting distance
% between columns and inserting a rule
%\setlength{\columnsep}{3em}
%\setlength{\columnseprule}{0.5pt}
```

```
%\twocolumn
\input{~#~group~#~oview.tex}

\newpage
%\onecolumn
\input{~#~group~#~code.tex}

\newpage
\phantomsection
\addcontentsline{toc}{section}{\texttt{=:} Index}
\printindex

\end{document}
)

NB. main jodliterate LaTeX preamble
JODLiteratePreamble=: 0 : 0

% jodliterate latex preamble.
%
% This file is a highly customized version of the preamble
% material generated by pandoc's -s option when producing
% .tex output. pandoc highlighting is overridden and
% the standard index is redefined.

\documentclass[12pt]{article}
```

```
\usepackage[landscape]{geometry}
\usepackage[headings]{fullpage}
\usepackage{lmodern}
\usepackage{amssymb,amsmath}
\usepackage{ifxetex,ifluatex}

% provides \textsubscript
\usepackage{fixltx2e}

% use microtype if available
\IfFileExists{microtype.sty}{\usepackage{microtype}}{}
\ifnum 0\ifxetex 1\fi\ifluatex 1\fi=0 % if pdftex
  \usepackage[utf8]{inputenc}
\else % if luatex or xelatex
  \usepackage{fontspec}
  \ifxetex
    \usepackage{xltxtra,xunicode}
  \fi
  \defaultfontfeatures{Mapping=tex-text,Scale=MatchLowercase}
  % replace EUROUC with unicode euro character
  % if you need this character - the presence of
  % this single character in the preamble forces use of xelatex, lualated
  %\newcommand{\euro}{EUROUC}
  % can set other monospace fonts if they're available
  % I rather like Source Code Pro see:
  % http://blogs.adobe.com/typblography/2012/09/source-code-pro.html
  %\setmonofont{FreeMono}
```

```
  %\setmonofont{Source Code Pro}
\fi

% Redefine labelwidth for lists; otherwise, the enumerate package will cause
% markers to extend beyond the left margin.
\makeatletter\AtBeginDocument{%
  \renewcommand{\@listi}
    {\setlength{\labelwidth}{4em}}
}\makeatother
\usepackage{enumerate}

% build document index
\usepackage{makeidx}

% colors
\usepackage{color}
\definecolor{shadecolor}{RGB}{248,248,248}
% j control structures
\definecolor{keywcolor}{rgb}{0.13,0.29,0.53}
% j explicit arguments x y m n u v
\definecolor{datacolor}{rgb}{0.13,0.29,0.53}
% j numbers - all types see j.xml
\definecolor{decvcolor}{rgb}{0.00,0.00,0.81}
\definecolor{basencolor}{rgb}{0.00,0.00,0.81}
\definecolor{floatcolor}{rgb}{0.00,0.00,0.81}
% j local assignments
\definecolor{charcolor}{rgb}{0.31,0.60,0.02}
```

```
\definecolor{stringcolor}{rgb}{0.31,0.60,0.02}
\definecolor{commentcolor}{rgb}{0.56,0.35,0.01}
% primitive adverbs and conjunctions
%\definecolor{othercolor}{rgb}{0.56,0.35,0.01}
\definecolor{othercolor}{RGB}{0,0,255}
% global assignments
\definecolor{alertcolor}{rgb}{0.94,0.16,0.16}
% primitive J verbs and noun names
\definecolor{funccolor}{rgb}{0.00,0.00,0.00}

\usepackage{fancyvrb}
\DefineShortVerb[commandchars=\\\{\}]{\|}
\DefineVerbatimEnvironment{Highlighting}{Verbatim}{commandchars=\\\{\}}
% Add ',fontsize=\small' for more characters per line

% pandoc generated syntax coloring commands - names
% are fixed in generated code but definitions may
% be set to any valid text formatting command
\usepackage{framed}
\newenvironment{Shaded}{}{}
\newcommand{\KeywordTok}[1]{\textcolor{keywcolor}{\textbf{{#1}}}}
% works better with Source Code Pro
%\newcommand{\KeywordTok}[1]{\textcolor{keywcolor}{{#1}}}
\newcommand{\DataTypeTok}[1]{\textcolor{datacolor}{{#1}}}
%\newcommand{\DecValTok}[1]{\textcolor{decvcolor}{{#1}}}
\newcommand{\DecValTok}[1]{{#1}}
\newcommand{\BaseNTok}[1]{\textcolor{basencolor}{{#1}}}
```

```
\newcommand{\FloatTok}[1]{\textcolor{floatcolor}{{#1}}}
\newcommand{\CharTok}[1]{\textcolor{charcolor}{\textbf{{#1}}}}
\newcommand{\StringTok}[1]{\textcolor{stringcolor}{{#1}}}
\newcommand{\CommentTok}[1]{\textcolor{commentcolor}{\textit{{#1}}}}
\newcommand{\OtherTok}[1]{\textcolor{othercolor}{{#1}}}
\newcommand{\AlertTok}[1]{\textcolor{alertcolor}{\textbf{{#1}}}}
%\newcommand{\FunctionTok}[1]{\textcolor{funccolor}{{#1}}}
\newcommand{\FunctionTok}[1]{{#1}}
\newcommand{\RegionMarkerTok}[1]{{#1}}
\newcommand{\ErrorTok}[1]{\textbf{{#1}}}
\newcommand{\NormalTok}[1]{{#1}}

% JOD oriented auxiliary commands for post processing pandoc generated latex
\newenvironment{JODGroupHeader}{}{}
\newenvironment{JODPostProcessor}{}{}

\usepackage{fancyhdr}
\pagestyle{fancy}

\ifxetex
  \usepackage[setpagesize=false, % page size defined by xetex
              unicode=false,     % unicode breaks when used with xetex
              xetex]{hyperref}
\else
  \usepackage[unicode=true]{hyperref}
\fi
```

```
\hypersetup{breaklinks=true,
            bookmarks=true,
            pdfauthor={},
            pdftitle={},
            colorlinks=true,
            urlcolor=blue,
            linkcolor=magenta,
            pdfborder={0 0 0}}
\setlength{\parindent}{0pt}
\setlength{\parskip}{6pt plus 2pt minus 1pt}
\setlength{\emergencystretch}{3em}  % prevent overfull lines
\setcounter{secnumdepth}{0}

% reset latex index to use four columns - default is two
% which results in lots of wasted page space in landscape
% NOTE: adjust if index names run together
% from: http://www.latex-community.org/viewtopic.php?f=4&t=1735
\usepackage{multicol}
\makeatletter
\renewenvironment{theindex}
  {\if@twocolumn
      \@restonecolfalse
   \else
      \@restonecoltrue
   \fi
   \setlength{\columnseprule}{0pt}
   \setlength{\columnsep}{35pt}
```

```
    \begin{multicols}{4}[\section*{\indexname}]
    \markboth{\MakeUppercase\indexname}%
             {\MakeUppercase\indexname}%
    \thispagestyle{plain}
    \setlength{\parindent}{0pt}
    \setlength{\parskip}{0pt plus 0.3pt}
    \relax
    \let\item\@idxitem}%
  {\end{multicols}\if@restonecol\onecolumn\else\clearpage\fi}
\makeatother

\makeindex

\begin{document}

)
NB.>>~~~~

NB.*end-header

NB. pandoc highlight string from '=:=:' mark
BEGININDEX=:'\AlertTok{=:=:}'

NB. marks start of JOD group header in pandoc latex
BEGINJODHEADER=:'\begin{JODGroupHeader}'

NB. marks start of JOD group postprocessor in pandoc latex
BEGINJODPOSTP=:'\begin{JODPostProcessor}'
```

```
NB. marks the start of J script text that is not J
BEGINNOTJ=:'NB.<<~~~~'

NB. carriage return character
CR=:13{a.

NB. pandoc highlight string from '=.=.' mark
ENDINDEX=:'\CharTok{=.=.}'

NB. marks end of JOD group header in pandoc latex
ENDJODHEADER=:'\end{JODGroupHeader}'

NB. marks end of JOD group postprocessor in pandoc latex
ENDJODPOSTP=:'\end{JODPostProcessor}'

NB. marks the end of J script text that is not J
ENDNOTJ=:'NB.>>~~~~'

NB. interface word list name prefix
IFACEWORDSPFX=:'IFACEWORDS'

NB. interface words (IFACEWORDSjodliterate) group
IFACEWORDSjodliterate=:<;._1 ' grplit setjodliterate ifacesection'

NB. jodliterate author - inserted in latex \author{}
JLAUTHOR=:'John D. Baker'
```

```
NB. suffix of jodliterate code file
JLCODEFILE=:'code.tex'

NB. suffix of jodliterate overview file
JLOVIEWFILE=:'oview.tex'

NB. suffix of jodliterate title file
JLTITLEFILE=:'title.tex'

NB. temporary latex file
LATEXTMP=:'jltemp.tex'

NB. line feed character
LF=:10{a.

NB. marks start of J code for pandoc -- requires pandoc with j syntax coloring
MARKDOWNHEAD=:'~~~~ { .j }'

NB. marks end J code for pandoc
MARKDOWNTAIL=:'~~~~'

NB. temporary markdown file
MARKDOWNTMP=:'jltemp.markdown'

NB. root words (ROOTWORDSjodliterate) group
ROOTWORDSjodliterate=:<;._1 ' IFACEWORDSjodliterate ROOTWORDSjodliterate grplit sbtokens setjodliterate'
```

```
NB. white space characters
WHITESPACE=:10 13 9 32{a.

NB. leading >... line wrap mark
WRAPLEAD=:'>..>'

NB. maximum number of code listing characters - adjust for given LaTeX pagesize
WRAPLIMIT=:110

NB. invalid j string marking start of wrapped line - cannot contain '=:'
WRAPPREFIX=:')=.)=.'

NB. pandoc LaTeX fragment from (WRAPPREFIX) - these strings must correspond
WRAPPREFIXTEX=:'\RegionMarkerTok{)}\CharTok{=.}\RegionMarkerTok{)}\CharTok{=.}'

NB. retains string after first occurrence of (x)
afterstr=:] }.~ #@[ + 1&(i.~)@([ E. ])

NB. trims all leading and trailing blanks
alltrim=:] #~ [: -. [: (*./\. +. *./\) ' '&=

NB. retains string before first occurrence of (x)
beforestr=:] {.~ 1&(i.~)@([ E. ])
```

```
betweenstrs=:4 : 0

NB.*betweenstrs v-- select sublists between  nonnested delimiters
NB. discarding delimiters.
NB.
NB. dyad:  blcl =. (clStart;clEnd) betweenstrs cl
NB.        blnl =. (nlStart;nlEnd) betweenstrs nl
NB.
NB.   ('start';'end') betweenstrs 'start yada yada end boo hoo start ahh end'
NB.
NB.   NB. also applies to numeric delimiters
NB.   (1 1;2 2) betweenstrs 1 1 66 666 2 2 7 87 1 1 0 2 2

's e'=. x
llst=. ((-#s) (|.!.0) s E. y) +. e E. y
mask=. ~:/\ llst
(mask#llst) <;.1 mask#y
)

NB. boxes open nouns
boxopen=:<^:(L. = 0:)

changestr=:4 : 0

NB.*changestr v-- replaces substrings - see long documentation.
NB.
NB. dyad:  clReps changestr cl
```

```
NB.
NB.   NB. first character delimits replacements
NB.   '/change/becomes/me/ehh' changestr 'blah blah ...'

pairs=. 2 {."(1) _2 [\ <;._1 x      NB. change table
cnt=._1 [ lim=. # pairs
while. lim > cnt=.>:cnt do.         NB. process each change pair
  't c'=. cnt { pairs               NB. /target/change
  if. +./b=. t E. y do.             NB. next if no target
    r=. I. b                        NB. target starts
    'l q'=. #&> cnt { pairs         NB. lengths
    p=. r + 0,+/\(<:# r)$ d=. q - l NB. change starts
    s=. * d                         NB. reduce < and > to =
    if. s = _1 do.
      b=. 1 #~ # b
      b=. ((l * # r)$ 1 0 #~ q,l-q) (,r +/ i. l)} b
      y=. b # y
      if. q = 0 do. continue. end.  NB. next for deletions
    elseif. s = 1 do.
      y=. y #~ >: d r} b            NB. first target char replicated
    end.
    y=.(c $~ q *# r) (,p +/i. q)} y  NB. insert replacements
  end.
end. y                              NB. altered string
)

NB. character table to newline delimited list
ctl=:}.@(,@(1&(,"1)@(-.@(*./\."1@(=&' '@])))) # ,@((10{a.)&(,"1)@])
```

```
cutnestidx=:4 : 0

NB.*cutnestidx v-- cut list into nested runs and other.
NB.
NB. Nested runs are delimited by begin and end tags. This verb is
NB. oriented toward XML parsing where typical begin  end tags are
NB. <ul>  </ul>  and tags  with  attributes  like: <hoo  boy="2">
NB. </hoo>
NB.
NB. This verb can process numeric lists but care must be taken to
NB. insure the pad item (1{.0$y) does  not  match begin  and  end
NB. values.
NB.
NB. dyad:  (ilIdx ;< blcl) =. (clStart;clEnd) cutnestidx cl
NB.        (ilIdx ;< blnl) =. (nlStart;nlEnd) cutnestidx nl
NB.
NB.   xml=. 'yada <ol><li>one</li><ol><li>sub one</li></ol></ol> boo'
NB.   ('<ol';'</ol>') cutnestidx xml
NB.
NB.   88 99 cutnestidx (i.5),88,(10?10),99 88 5 5 5 5 5 99

if. #y do.
  's e'=. ,&.> x                NB. start end lists
  ut=. 1{.0$y                   NB. padding
  assert. -.s -: e              NB. they must differ
  assert. -.(s -:ut) +. e-:ut
  sp=. s E. ut=.y,ut            NB. start mask
```

```
  NB. quit if no delimiters
  if. -.1 e. sp do. (i.0);<<y return. end.

  ep=. e E. ut                    NB. end mask
  assert. (+/sp) = +/ep           NB. basic balance
  dp=. sp + - ep                  NB. start end marks
  assert. 0 *./ . <: +/\ dp       NB. nested balance
  ep=. I. _1=dp [ sp=. I. 1=dp    NB. start end indexes
  ut=. +/\dp -. 0                 NB. scanned marks
  dp=. /:~ sp,ep                  NB. all indexes
  sp=. (firstones 1<:ut)#dp       NB. starts of nested
  ep=. (#e)+(0=ut)#dp             NB. starts of other
  dp=. /:~ ~.0,sp,ep              NB. cut starts
  ut=. }: 1 dp} (>:#y)#0          NB. cut mask
  (dp i. sp);<ut <;.1 y           NB. nest indexes cut list
else.
  (i.0);<<y                       NB. empty arg result
end.
)

decomm=:3 : 0

NB.*decomm v--  removes comments  from j words. The (x) argument
NB. specifies whether all blank lines are removed or retained.
NB.
NB. monad:  decomm ctWord
NB.
```

```
NB.   decomm jcr 'decomm'  NB. decomment self
NB.
NB. dyad:  pa decomm ctWord
NB.
NB.   1 decomm jcr 'decomm'  NB. remove blanks (default)
NB.   0 decomm jcr 'decomm'  NB. retain all blank lines

1 decomm y
:
NB. mask of unquoted comment starts
c=. ($y)$'NB.' E. ,y
c=. +./\"1 c > ~:/\"1 y e. ''''
y=. ,y

NB. blank out comments
y=. ' ' (I. ,c)} y
y=. y $~ $c

NB. remove blank lines - default
if. x do. y #~ y +./ . ~: ' ' end.
)

NB. boxes UTF8 names
fboxname=:([: < 8 u: >) ::]

NB. erase files - cl | blcl of path file names
ferase=:1!:55 ::(_1:)@(fboxname&>)@boxopen
```

```
NB. 1 if file exists 0 otherwise
fexist=:1:@(1!:4) ::0:@(fboxname&>)@boxopen

NB. 0's all but first 1 in runs of 1's - like (firstone) but differs for nulls
firstones=:> (0: , }:)

formifacetex=:3 : 0

NB.*formifacetex v-- formats hyperlinked and highlighted interface words.
NB.
NB. monad:  cl =. formifacetex blclIwords

NB. jod refs !(*)=. get
head=. '\begin{Shaded}',LF,'\begin{Highlighting}[]',LF
tail=. '\end{Highlighting}',LF,'\end{Shaded}',LF
ctok=. '\CommentTok{'
ntok=. '\NormalTok{'
href=. '\hyperlink{'

NB. fetch current short descriptions
'rc tab'=. 0 8 get y
words=. 0 {"1 tab
desc=.  1 {"1 tab

NB. set hyperlinks on words - colors on comments
words=. (<href) ,&.> words ,&.> (<'}{',ntok) ,&.> (<"1 (>words),"1 ' ') ,&.> <'}}'
desc=. (<ctok) ,&.> (alltrim&.> desc) ,&.> '}'
```

```
tex=. ;words ,&.> desc ,&.> LF
head,tex,tail
)

formtexindexes=:3 : 0

NB.*formtexindexes v-- format latex index commands from global marks.
NB.
NB. monad:  blcl =. formtexindexes blclMarked

NB. extract =:=: =.=. marked text
inames=. ;@('{}'&betweenstrs)&.> (-#ENDINDEX) }.&.> (#BEGININDEX) }.&.> y

NB. find any indirect ()=: and multiple ''=: assignments
'pma pia'=. I.&.> <"1 ''')' =/ {:@(-.&' ')&> inames

NB. form latex index commands
indexes=. (<'\AlertTok{=:}\index{') ,&.> inames ,&.> (<'@\texttt{') ,&.> inames ,&.> <'}}'

NB. replace indirect and multiple assignments with fixed proxies
indexes=. ((#pma) # <'\AlertTok{=:}\index{00multiple@\texttt{''...''=:}}') pma} indexes
indexes=. ((#pia) # <'\AlertTok{=:}\index{01indirect@\texttt{(...)=:}}') pia} indexes

NB. adjust j locative chars _ they give latex indexing grief
if. #pos=. I. '_'&e.&> indexes do.
  indexes=. ('#_#\_'&changestr&.> pos{indexes) pos} indexes
end.
```

```
indexes
)

NB. size of file in bytes
fsize=:1!:4 ::(_1:)@(fboxname&>)@boxopen

gbodylatex=:3 : 0

NB.*gbodylatex v-- group body latex.
NB.
NB. monad:  clTex =. gbodylatex clGroupname

if. #mtxt=. markdfrgroup y do. latexfrmarkd mtxt else. '' end.
)

gheadlatex=:3 : 0

NB.*gheadlatex v-- group header latex.
NB.
NB. monad:  clTex =. gheadlatex clGroupname

if. #mtxt=. markdfrghead y do.
  BEGINJODHEADER,LF,(tlf latexfrmarkd mtxt),ENDJODHEADER,2#LF
else.
  ''
end.
)
```

```
gpostlatex=:3 : 0

NB.*gpostlatex v-- group post processor latex.
NB.
NB. monad:  clTex =. gpostlatex clGroupname

if. #mtxt=. markdfrgpost y do.
  BEGINJODPOSTP,LF,(tlf latexfrmarkd mtxt),ENDJODPOSTP
else.
  ''
end.
)

grouplatex=:3 : 0

NB.*grouplatex v-- group latex with pandoc syntax highlighting.
NB.
NB. monad:  clTex =. grouplatex clGroupname
NB.
NB.   NB. requires open JOD dictionary with a 'jod' group
NB.   gtex=. grouplatex 'jod'
NB.
NB. dyad:  clTex =. paIndex grouplatex clGroupname
NB.
NB.   0 grouplatex 'jod' NB. do not replace marks with index

1 grouplatex y
:
```

```
NB. jod refs !(*)=. badrc_ajod_ grp jderr_ajod_
if. badrc_ajod_ gnames=. grp y do. gnames return. end.

ltx=. (gheadlatex ; gbodylatex ; gpostlatex) y
ltx=. ]`indexgrouptex@.(1 -: x) ; tlf&.> ltx -. a:
ltx=. ('#',WRAPPREFIXTEX,'#\AlertTok{',WRAPLEAD,'}') changestr ltx
'\section{\texttt{',(alltrim y),'} Source Code}',LF,LF,ltx
)

grplit=:3 : 0

NB.*grplit v-- make latex for group (y).
NB.
NB. monad:  (paRc ; blclTeXfiles) =. grplit clGroupname
NB.
NB.   grplit 'jodliterate'  NB. document self
NB.
NB. dyad:  (paRc ; blclTeXfiles) =. paOw grplit glGroupname
NB.
NB.   NB. do not overwrite root tex - allows for latex tweaking
NB.   0 grplit 'jodliterate'

1 grplit y
:
NB. jod refs !(*)=. badrc_ajod_ get grp jderr_ajod_ ok_ajod_
try.

if. 3~:(4!:0) <'badrc_ajod_' do. 0;'!error: jod is not loaded' return. end.
```

```
if. 0=#JLDIRECTORY  do. 0;'!error: working directory is not set' return. end.
group=. y -. ' '
if. badrc_ajod_ gdoc=. 2 9 get group do. gdoc return. end.

NB. latex from any group document markdown
hype=. 0
if. #gdoc=. ;{:,>1{gdoc do.
  gdoc=. latexfrmarkd gdoc
  if. badrc_ajod_ glist=. grp group do. glist return. end.
  ifstr=. ifacesection group
  if. (+./ifstr E. gdoc) *. (<IFACEWORDSPFX,group) e. glist do.
    iwords=. ifacewords group
    gdoc=. iwords setifacelinks ifstr;gdoc
    hype=. 1  NB. hyperlinks set
  end.
end.

NB. root .tex file - gets group name
wdir=. JLDIRECTORY
jlroot=. wdir,group,'.tex'
if. chroot=. x -: 1 do.
  root=. ('/~#~group~#~/',group) changestr JLGRPLITTEX
  (toJ root) writeas jlroot
end.

NB. author title .tex file
agstrs=. '/~#~author~#~/',(alltrim JLAUTHOR),'/~#~group~#~/',alltrim y
```

```
(toJ agstrs changestr JLTITLETEX) writeas jltitle=. wdir,group,JLTITLEFILE

NB. group overview .tex file
ohd=. ('/~#~group~#~/',alltrim y) changestr JLOVIEWTEX
ohd=. ohd,LF,gdoc
(toJ ohd) writeas jloview=. wdir,group,JLOVIEWFILE

NB. group source code - return file names
gltx=. grouplatex group
if. hype do. gltx=. iwords setifacetargs gltx end.
(toJ gltx) writeas jlcode=. wdir,group,JLCODEFILE
ok_ajod_ (-.chroot) }. jlroot;jltitle;jloview;jlcode

catchd.
  0;'!error: (grplit) failure - last J error ->';13!:12 ''
end.
)

ifacesection=:3 : 0

NB.*ifacesection v-- interface section summary string.
NB.
NB. This verb produces the interface section summary  string. For
NB. (jodliterate)  to  include an  updated  hyperlinked interface
NB. summary it  must find  this string in  generated  latex. Edit
NB. this verb if you change the section layout.
NB.
NB. monad:  cl  =. ifacesection clGroupname
```

```
'\subsection{\texttt{',y,'} Interface}'
)

ifacewords=:3 : 0

NB.*ifacewords v-- return interface word list.
NB.
NB. Assume the interface is out of date fetch current  definition
NB. from  dictionary.  We  need  the  value   not   the   storage
NB. representation so define it in the JOD scratch object.
NB.
NB. monad:  blcl =. ifacewords clGroupname

NB. jod refs !(*)=. get
iname=. (IFACEWORDSPFX,y) -. ' '
(;SO__JODobj) get iname
iname=. iname,'__SO__JODobj'
words=. ". iname
words [ (4!:55) <iname
)

indexgrouptex=:3 : 0

NB.*indexgrouptex v-- insert index commands in pandoc highlight group latex.
NB.
NB. dyad:  cl =. clGroupName indexgrouptex clTex
```

```
'pos ltx'=. (BEGININDEX;ENDINDEX) cutnestidx y
if. #pos do. ; (formtexindexes pos{ltx) pos} ltx else. y end.
)

NB. standarizes J path delimiter to unix/linux forward slash
jpathsep=:'/'&(('\' I.@:= ])})

jtokenize=:3 : 0

NB.*jtokenize v-- tokenizes j text with (wfl).
NB.
NB. Similar  to  (;:&.>)@(<;.2) but preserves  whitespace  and is
NB. able to parse invalid j text  containing open quotes. When an
NB. open quote  is encountered it is treated like an unterminated
NB. string.
NB.
NB. monad:  blblcl =. jtokenize clJtext
NB.
NB.   jtokenize 5!:5 <'jtokenize'

ct=. wfl y,LF
(ct -:&> <,LF) <;.2 ct
)

latexfrmarkd=:3 : 0
```

```
NB.*latexfrmarkd v-- latex from markdown using pandoc.
NB.
NB. monad:  clTex =. latexfrmarkd clMarkdown

NB. require 'task' !(*)=. shell
if. #y do.
  ferase mrktmp=. JLDIRECTORY,MARKDOWNTMP
  ferase ltxtmp=. JLDIRECTORY,LATEXTMP
  (toJ y) writeas mrktmp
  NB. highlighting style is overridden in latex preamble
  shell 'pandoc --highlight-style=tango ',mrktmp,' -o ',ltxtmp
  assert. 0 < fsize ltxtmp
  tex=. read ltxtmp
  tex [ ferase ltxtmp [ ferase mrktmp
else.
  y
end.
)

markdfrghead=:3 : 0

NB.*markdfrghead v-- markdown text from group header.
NB.
NB. monad:  cl =. markdfrghead clGroupname
NB.
NB.   mtxt=. markdfrghead 'jod'
NB.   (toHOST mtxt) write 'c:/temp/jodhdr.markdown'
```

```
NB. jod refs !(*)=. badrc_ajod_ get HEADEND_ajodmake_
if. badrc_ajod_ hdr=. 2 get y do. hdr return. end.
if. 0=#hdr=. ;1{,>1{hdr       do. ''  return. end.
hdr=. hdr,LF,HEADEND_ajodmake_

NB. handle any non j code regions
'idx chd'=. (BEGINNOTJ;ENDNOTJ) cutnestidx hdr

if. #idx do.
  psj=. idx -.~ i.#chd
  chd=. (markgnonj&.> idx{chd) idx} chd
  chd=. (markgassign&.> psj{chd) psj} chd
  hdr=. ;chd
else.
  hdr=. markgassign hdr
end.

if. #hdr do. markdj hdr else. '' end.
)

markdfrgpost=:3 : 0

NB.*markdfrgpost v-- markdown from group post processor.
NB.
NB. monad:  clMarkdown =. markdfrgpost clGroupname

NB. jod refs !(*)=. get
'rc post'=.  2 {. 4 get 'POST_',y -.' '
```

```
if. rc do. markdj markgassign ; {: , post else. '' end.
)

markdfrgroup=:3 : 0

NB.*markdfrgroup v-- markdown text from group.
NB.
NB. monad:  cl =. markdfrgrp clGroupname
NB.
NB.   mtxt=. markdfrgroup 'jod'
NB.   (toHOST mtxt) write 'c:/temp/jcode.markdown'

NB. jod refs !(*)=. badrc_ajod_ get gdeps grp
if. badrc_ajod_ gnl=. grp y   do. gnl return. end.
if. badrc_ajod_ gdp=. gdeps y do. gdp return. end.
if. #gnl=. (gnl -. gdp) -. a: do. markdfrwords gnl else. '' end.
)

markdfrwords=:3 : 0

NB.*markdfrwords v-- markdown text from word list.
NB.
NB. This verb takes a  blcl of JOD word names and returns a UTF-8
NB. encoded cl of  word source  code in markdown format. Markdown
NB. is  a simple but versatile  text markup format that is almost
NB. ideal for documenting program source code, see:
NB.
```

```
NB. http://daringfireball.net/projects/markdown/
NB.
NB. monad:  clMarkdown =. markdfrwords blclWords
NB.
NB.   markdfrwords ;:'go ahead mark us up'
NB.
NB.   NB. markdown text from JOD group words
NB.   mtxt=. markdfrwords }. grp 'jod'

NB. jod refs !(*)=. badrc_ajod_ get wttext__MK__JODobj
if. badrc_ajod_ src=. 0 10 get y do. src return. end.

NB. commented source code (name,source) table.
if. badrc_ajod_ src=. 0 0 1 wttext__MK__JODobj >1{src do. src
else.
  src=. markgassign&.> {:"1 >1{src
  NB. similar to (markdj) but faster here
  utf8 ; (<LF,MARKDOWNHEAD,LF) ,&.> src ,&.> <LF,MARKDOWNTAIL,LF
end.
)

markdj=:3 : 0

NB.*markdj v-- mark j code for markdown.
NB.
NB. monad:  clM =. markdj clJ
```

```
utf8 (LF,MARKDOWNHEAD,LF),(tlf y),MARKDOWNTAIL,LF
)

markgassign=:3 : 0

NB.*markgassign v-- mark j code for latex indexing.
NB.
NB. This verb tokenizes j code and replaces all global
NB. assignments with syntactically incorrect j strings that will
NB. be transformed by pandoc into easily located latex strings
NB. that will then be converted by a post pandoc processor into
NB. valid latex index commands. This works because regex based
NB. pandoc coloring does not "understand" j's parsing rules.
NB.
NB. monad:  cl =. markgassign clJcode
NB.
NB.  jcode=. 'markgassign=:' , 5!:5 <'markgassign'
NB.  markgassign jcode

if. 0=#jcode=. y -. CR do. y return. end.
jcode=. WRAPLIMIT wrapvrblong jcode
jtokens=. jtokenize jcode

NB. only interested in global assignment lines
if. #gix=. I. ; (<'=:') e. L: 1 jtokens do.
  jgl=. gix{jtokens
  jshp=. $jat=. >jgl
  jix=. I. jat = <'=:' [ jat=. ,jat
```

```
  NB. extract global assignments
  NB. ignoring interleaving blanks
  jat2=. (jat -.&.> ' ') -. a:
  anames=. (<:I.(<'=:') -:&> jat2){jat2
  NB. =:=: and =.=. are invalid in j
  faketoks=. (<'=:=:') ,&.> anames ,&.> <'=.=.'
  jat=. <"1 jshp $ faketoks jix} jat
  jat=. (#&> jgl) {.&.> jat
  NB. adjust last LF
  (-LF={:y) }. ;;jat gix} jtokens
else.
  y
end.
)

markgnonj=:3 : 0

NB.*markgnonj v-- mark non j code region global assignments.
NB.
NB. Non J code is often inserted in J scripts as  character nouns
NB. using explicit  multi-line  '0 :  0'  definitions. This  verb
NB. marks the assigned noun name. Only '=: 0 :  0' will be  found
NB. and marked.
NB.
NB. verbatim:
NB.
NB. IamFound =: 0 : 0
NB. .... non j code ...
```

```
NB. )
NB.
NB. monad:  cl =. markgnonj clNonj

ct=.  <;.2 tlf y
mrk=. '=:0:0'
pos=. I. mrk&-:&> (-#mrk)&{.&.> ct -.&.> <WHITESPACE
ct=. ;(LF ,&.>~ markgassign&.> pos{ct) pos} ct
(-LF={:y) }. ct
)

NB. reads a file as a list of bytes
read=:1!:1&(]`<@.(32&>@(3!:0)))

NB. trim right (trailing) blanks
rtrim=:] #~ [: -. [: *./\. ' '"_ = ]

NB. blcl of nonempty noncomment J cl tokens
sbtokens=:a: -.~ (<13 10 9{a.) -.&.>~ [: alltrim&.> [: wfl [: ctl [: decomm [: ];._1 (10{a.) , (13{a.) -.~ ]

setifacelinks=:4 : 0

NB.*setifacelinks v-- set hyperref links in any overview
NB. interface words section.
NB.
NB. dyad:  cl =. blclIwords setifacelinks (clIfstr ; clTex)
```

```
'ifstr tex'=. y
rmrk=. '\end{Shaded}'
head=. ifstr&beforestr tex
tail=. ifstr&afterstr tex

if. +./rmrk E. tail do.
  ifbk =. formifacetex x
  tail =. rmrk&afterstr tail
  head,ifstr,(2#LF),ifbk,tail
else.
  tex
end.
)

setifacetargs=:4 : 0

NB.*setifacetargs v-- set hyperlink targets in group latex.
NB.
NB. dyad:  cl =. blclIwords setifacetargs clTex

targs=. (<'\NormalTok{') ,&.> x ,&.> <'}\AlertTok{=:}\index'
rstrs=. (<'\hypertarget{') ,&.> x ,&.> (<'}{\NormalTok{') ,&.> x ,&.> <'}}\AlertTok{=:}\index'
chgs=.  ;('#' ,&.> targs) ,&.> '#' ,&.> rstrs
chgs changestr y
)

setjodliterate=:3 : 0
```

```
NB.*setjodliterate v-- prepare for processing.
NB.
NB. monad:  (paRc ; clDir) =. setjodliterate clWorkingDir
NB.
NB.   setjodliterate 'c:\temp'           NB. windows
NB.   setjodliterate '/home/john/temp'   NB. linux

try.

if. 3~:(4!:0) <'badrc_ajod_' do. 0;'!error: jod is not loaded' return. end.

NB. profile (*)=. IFWIN
JLDIRECTORY_ajodliterate_=: jpathsep`winpathsep@.(IFWIN) tslash2 y

NB. write main latex preamble once per directory
preamble=. 'JODLiteratePreamble.tex'
if. -.fexist JLDIRECTORY,preamble do.
  (toJ JODLiteratePreamble) writeas JLDIRECTORY,preamble
end.
1;JLDIRECTORY

catchd.
  0;'!error: (setjodliterate) failure - last J error ->';13!:12 ''
end.
)

NB. appends trailing line feed character if necessary
tlf=:] , ((10{a.)"_ = {:) }. (10{a.)"_
```

```
NB. converts character strings to J delimiter LF
toJ=:((10{a.) I.@(e.&(13{a.))@]} ])@:(#~ -.@((13 10{a.)&E.@,))

NB. appends trailing / iff last character is not \ or /
tslash2=:([: - '\/' e.~ {:) }. '/' ,~ ]

NB. character list to UTF-8
utf8=:8&u:

NB. standarizes J path delimiter to windows back slash
winpathsep=:'\'&(('/' I.@:= ])})

wrapvrblong=:3 : 0

NB.*wrapvrblong v-- wraps verbatim text lines with length > (x).
NB.
NB. Wraps  lines with length > (x) and prefixes each wrapped line
NB. with the syntactically invalid  j string ')=.)=.' This string
NB. is transformed  by  pandoc into  an easily found sequence  of
NB. LaTeX commands.
NB.
NB. monad:  cl =. wrapvrblong clTxt
NB. dyad:   cl =. iaLength wrapvrblong clTxt

WRAPLIMIT wrapvrblong y
:
NB. always trim trailing blanks
```

```
ct=. <@rtrim;._2 tlf y -. CR

NB. only wrap lines exceeding limit
if. #pos=. I. x < #&> ct do.
  wlen=. x-#WRAPLEAD
  wt=. (-wlen) (<\)&.> pos{ct
  slen=. 1&,@:<:@#&.> wt
  NB. lead wrapped lines with prefix
  wt=. (slen #&.> <(<'')),<LF,WRAPPREFIX) ,.&.> wt
  wt=. a: -.~ L: 1 ,&.> wt
  NB. last wrapped line LF terminated
  wt=. wt , L: 1 <LF
  nwpos=. (i.#ct) -. pos
  ct=. ((nwpos{ct) ,&.> LF) nwpos} ct
  ;;wt pos} <"0 ct
else.
  (-LF~:{:y) }. ; ct ,&.> LF
end.
)

NB. write file as list of bytes - throws unambiguous error on failure
writeas=:(1!:2 ]`<@.(32&>@(3!:0))) ::([: 'cannot write file'&(13!:8) 1:)

NB.POST_jodliterate post processor (-.)=:

smoutput 0 : 0
NB. (jodliterate) interface word(s):
```

```
NB.   grplit          NB. make latex for group (y)
NB.   ifacesection    NB. interface section summary string
NB.   setjodliterate  NB. prepare for processing
)

cocurrent 'base'
coinsert  'ajodliterate'
```

Index