Man Pages for JS/UIX 0.45

Contents:

alias	sets an alias for a (complex) command
apropos	displays a short description of a command
browse	opens a web page in a new browser window
cal	displays a monthly calendar
cat	concatenates and outputs files
cd	changes the current directory
chmod	changes a files's permissions.
clear	clears and resets the terminal display
cp	copies files from source to target
date	displays the date and time with format options
echo	writes all arguments back to the terminal
exit	exits the current shell or leaves the session
features	displays the features of this application.
fexport	exports home-directory for copy and later re-use
fimport	imports preveously exported file-data to the file-system
hallo	displays a short system information
halt	halts / shuts down the system
hello	displays a short system information
help	displays a help screen with complete command list
info	displays information about the site
invaders	starts the well known arcade game.
js	javascript evaluation (no user command, experts only; read man!)
logname	displays the current user name
ls	lists a directory
mail	launches a mail client
man	displays manual pages for a command
mkdir	creates a directory
more	displays long files in a pager
mv	moves (renames) files from source to target
news	displays system-news and information on recent changes.
pager	pager (synonym for "more")
pg	pager (synonym for "more")
pr	prints a file to a browser window
ps	displays current processes
pwd	prints the current working directory
reboot	halts and reboots the system
rm	removes files
rmdir	removes /empty) directories
set	sets shell-variables, without arguments a full list is displayed
sh	starts a new system shell (command interpreter)
splitmode	switches terminal splitmode (statusline) on/off
stty	sets terminal options
su	switches user
time	displays the current time
touch	sets the timestamp of a file or creates empty file
type	writes all arguments back employing type styles
unalias	discards an alias defined by "alias"
uname	displays the system identification
unset	discards a shell variable
vi	visual editor (simpler version of standard UN*X vi)
view	vi (visual editor) in view mode (read only)
wc	word count (words, lines, characters)
web	opens a web page in a new browser window
which	evaluates which command will be executed
write	writes to the terminal employing type styles as mark up.

alias

```
* Synopsis:
alias <name> {<value>}

sets an alias that will be used as a command.
aliases can be discarded using "unset".
if called without arguments, all set aliases are listed.

* Arguments:
    <name>   the name of the alias. names must begin with a letter
             and use only the characters "A"-"Z", "0"-"9" or "_".
    <value>  the value of the alias.


apropos

* Synopsis:
apropos <command>

displays a short description of a command.

* Arguments:
    <command>  command name.


browse

* Synopsis:
browse [-n] [<url>]

opens a given url in a new browser window.
if no url was specified, a standard site is called (<http://www.masswerk.at>).

* Arguments:
    <url>  url of a website. protocol defaults to http.

* Options:
    -n  open in a new browser window. (with JS/UIX 0.3x default value!)


cal

* Synopsis:
cal [-w] [[<month_nr>] [<year>]]

displays a monthly calendar.
defaults to current month and year if no arguments specified.

* Arguments:
    <month_nr>  number of month (1..12), default current month.
    <year>      year (1900..9999), default current year.

* Options:
    -w  show week numbers.


cat

* Synopsis:
cat <filelist>

concatenate files
joins any specified files to a new stream.
any lines in STDIN will preceed the content of theese files.
```

```
* Arguments:
    <filelist>  any number of file-paths separated by spaces.


cd

* Synopsis:
cd [<dirname>]

change directory to given path.
if called without argument, the current working directory will be set to the
value of $HOME.

path/name-conventions:
    "/" = file-separator
    "." = current directory
    ".." = parent directory.


chmod

* Synopsis:
chmod [-R] <mode> <filelist>
where <mode> is octal number or {u|g|o|a}(+|-){w|r|x|s} or {u|g|o|a}=(o|u|g)

change a files's permissions for read, write or execute.

* Arguments:
    <filelist>  file(s) to be set (you must be the file's owner).
    <mode>      either an octal number representing a bit-vector,
                where position "x" stands for:
                  00x00 ... user (owner of the file)
                  000x0 ... group
                  0000x ... others
                  0x000 ... sticky-bit

                and "x" is a 3-bit value (0-7),
                where a set or unset bit represents permissions for:
                  4 ... read
                  2 ... write
                  1 ... execute

                or in the form of {u|g|o|a}(+|-){w|r|x|s},
                where the first part represents the "who"-part as:
                  u ... user
                  g ... group
                  o ... other
                  a ... all

                to be either set (+) or unset (-) to the third part as:
                  w ... write
                  r ... read
                  x ... execute/search
                  s ... sticky-bit

                or in the form of {u|g|o|a}=(o|u|g),
                where the first part represents the "who"-part as above
                to be set to the value of the third part.

                (the current version does not support setUID or setGID.
                these bits will be ignored.)

* Options:
    -R  recursive (include nested files and directories).
```

```
clear

* Synopsis:
clear

clears and resets the terminal display.


cp

* Synopsis:
cp [-ipr] <sourcefile> {<sourcefile>} <target>

copy files from source- to target-file.

* Arguments:
    <sourcefile>  file(s) or directories to be copied
                  if called with multiple source-files the target must be
                  a directory
    <target>      the file name of the new file or the name of a directory.

* Options:
    -i  ignore error warnings
    -p  copy file permissions
    -r  recursive - include nested files


date

* Synopsis:
date [-l|u] [+format]

diplays the date and time as local (default) or UTC
as: weekday, day month year hours:minutes:seconds [UTC]
the output can be formated by an optional format-string.

* Arguments:
    <format>  a string consisting of any of the following characters:
    %%a        week-day abrv., Sun-Sat
    %%d        day, 1-31
    %%D        date as mm/dd/yy
    %%h        month abrv., Jan-Dec
    %%H        hours, 00-23
    %%j        year-day, 001-366
    %%m        month, 01-12
    %%M        minutes, 00-59
    %%n        new line
    %%r        time in AM/PM
    %%S        seconds, 00-59
    %%t        tab (insert space)
    %%T        time as hh:mm:ss
    %%w        week-day, 0-6, Sun=0
    %%y        last two digits of the year, 00-99

    example:  date +%D%t%T
              gives "11/05/03 16:50:01"

* Options:
    -l  local time (default)
    -u  UTC time


echo
```

```
* Synopsis:
echo [<args>]

writes the given arguments back to the terminal

* Arguments:
    <args>  any text separated by any amount of space.


exit

* Synopsis:
exit

exits the current shell.
if the current shell is the login-shell, the session is closed.


features

* Synopsis:
features

displays the features of this application.


fexport

* Synopsis:
fexport

file-export and backup.
exports the files and directories residing in the home-directory (as set in
$HOME) to a browser form for later re-use. copy this data and keep it on your
local machine for later import. (hidden files won't be exported.)
you can mount exported files and directories with "fimport".


fimport

* Synopsis:
fimport

imports/mounts exported files and directories to the current home-directory.
if files or directories with the same name exist, these will have precedence
over any files on the import-list. timestamps will be set according to import-
data. this may back-date directories with newer content.
see "fexport" for exporting data.


hallo

* Synopsis:
hallo

displays a short information about this system.


halt

* Synopsis:
halt

halt / shut down the system
```

```
hello

* Synopsis:
hello

displays a short information about this system.


help

* Synopsis:
help

displays a help screen with a short list of available commands.


info

* Synopsis:
info

displays information about this site.
aliases: "masswerk", "mass:werk".


invaders

* Synopsis:
invaders

starts the well kown arcade game: space invaders for JS/UIX.
please note that there is only one life and only one shot at a time.

usage: use cursor <LEFT> and cursor <RIGHT> to move, press <SPACE> to fire.
(alternatively you may use the vi-movements "h"=left and "l"=right.)
press "p" for pause, "q" or <ESC> to quit.


js

* Synopsis:
js -l[t]|t <varname>

js -s[n] <varname> <value>
js -e <expression>

javascript evaluation (no user command, experts only!).
lists or sets javascript objects and object properties, evaluates expressions.
CAUTION: an error in an eval-string will cause an javascript-error bringing
down the JS/UIX-system! setting a variable may override and harm the system.

* Arguments:
    <varname>     name of a variable, object or property
                  may be in form of "varname", "varname[index]",
                  "varname.prop[index]", "varname[index][index]" and so on.
    <value>       a numeric or string value for set (option -s)
    <expression>  expression to be evaled (option -e)

* Options:
    -l[t]  list an object or property
    -s[n]  set an object's value or object's property's value
           "-sn" for numeric (plain) value (default: string)
    -t     report object's type or object's property's type
    -e     eval expression (use single quotes to hide specials from shell)
```

```
logname

* Synopsis:
logname

displays the current user name


ls

* Synopsis:
ls <dirname>

lists a directory.


* Arguments:
    <dirname>  ralative or absolute file path.
               if called with option "i" or "l" also the name of a plain file.

* Options:
    -C  force output to colums
    -F  show file type (appended to filename)
        "/" ... directory
        "*" ... executable
        "@" ... link
        <nothing> ... plain file
    -L  force output to one file by line
    -a  show hidden '.'-files.
    -i  show inode-id (file serial number)
    -l  long output, format:
        "mode  inodes  user  group  bytes  mdate [YYYY/MM/DD hh.mm:ss]  name"


mail

* Synopsis:
mail [<user@host>]

opens a mail window to given address or the webmaster if none specified.

* Arguments:
    <user@host>  mail address.


man

* Synopsis:
man <command>

displays a manual page for system commands.
if an entry for the command is found, it will be displayed using the standard
pager.

* Arguments:
    <command>  command name.
               for an alias its value is displayed.

* Options:
    -p  opens a new browser window with the full list.


mkdir
```

```
* Synopsis:
mkdir <dirname> {<dirname>}

make one or more new directory/ies

* Arguments:
    <dirname>  directory/ies to be inited


more

* Synopsis:
more <filename>

displays the specified file in a pager. if used in a pipe, any lines in STDIN
will preceed the content of any specified file. Any outgoing lines in STDOUT
will be stripped off of any type-styles.

for navigation use
    <SPACE>  for the next page, or
    "q"      for quit


mv

* Synopsis:
mv [-i] <filename> {<filename>} <target>

move (rename) files from source to target.

* Arguments:
    <filename>  file(s) or directories to be moved
                if called with multiple files the target must be a directory
    <target>    the file name of the new file or the name of a directory.

* Options:
    -i  ignore error warnings


news

* Synopsis:
news

displays system-news and information on recent changes. (displays /etc/news)


pager

* Synopsis:
pager <filename>

synonym for "more".

=> see "more".


pg

* Synopsis:
pager <filename>

synonym for "more".
```

```
=> see "more".


pr

* Synopsis:
pr <filelist>

print files (to a new browser window) - ready for copy&paste.

* Arguments:
    <filelist>  list of files to be printed.
                any content of a lefthand pipe will preceed the content of
                these files.


ps

* Synopsis:
ps

displays a list of active processes with PID (Process-ID) and name.


pwd

* Synopsis:
pwd

print working directory.
outputs the path of the current working directory.


reboot

* Synopsis:
reboot

halt and reboot the system


rm

* Synopsis:
rm [-ir] <filename> {<filename>}

remove (discard) files.
use "rmdir" or "rm -r" for directories.

* Arguments:
    <filename>  file(s) to be removed

* Options:
    -i  ignore error warnings
    -r  recursive - discard directories and included files


rmdir

* Synopsis:
rmdir [-i] <dirname> {<dirname>}

remove (discard) directories.
directories must be empty! use "rm -r" for populated directories.
```

```
 * Arguments:
     <dirname>  directory/ies to be removed

 * Options:
     -i  ignore error warnings


 set

 * Synopsis:
 set [<varname> {<varname>} [= {<value>}]]

 sets a variable in the command shell.
 variables can be retrieved by "$<varname>" in any term not in single-quotes.
 see "man sh" for more. to discard a variable use "unset".
 if called without arguments all set variables and values are listed.

 The system supports currently the following special variables:
   GID      group-id
   HOME     home directory
   HOST     login-host
   PATH     command path
   PID      process id of current process environment
   PS       shell prompt
   UID      user-id
   USER     user-name
   VERSION  os/term-version

 * Arguments:
     <varname>  the name of the variable. names must begin with a letter
                and use only the characters "A"-"Z", "0"-"9" or "_".
     <value>    the value of the variable. use quotes and escapes ("\") for
                complex expressions.
                if no value is assigned, the variable holds an empty value.


 sh

 * Synopsis:
 shell, commands, aliases, and variables.

 A simple implementation of sh. As command opens a subshell.
 Currently the following features are supported:
 quotings, escapes, variables, aliases, pipes, subshells, simple scripts.

 Quoting levels:
    double-quotes  string with variable interpolation
    single-quotes  literal string without interpolation
    backticks (`)  will be expanded to the output processed by a subshell called
                   with this string as its arguments.

 Commands may be separated by ";".
 The pipe-character "|" will stream the output of the left side to the STDIN-
 stream of the command on its right side.
 The output redirector ">" writes the output of the command to a file specified
 on its right side. ">>" appends the output to an existing file if any.

 Order of Interpolation:
 First all control-characters ("`", "|", ";", ">", ">>") will be traced, then
 any terms in backticks will be evaluated in a new subshell and the return
 values will be inserted and parsed as arguments.
 Afterwards all variables of the current arguments will be expanded. If the
 first argument is an alias, the alias will be expanded, its value parsed and
 copied in front the first remaining argument.
 In case a backslash ("\") is found at the end of a line, the line is
```

```
  concatenated with the following one to a single line.

  Order of Execution:
  If the now first argument is a shell-command (set, unset, alias, unalias, cd)
  it will be executed in the same shell.
  Else, if an executable file with the name of the command is found in any
  directory specified in the PATH-variable, this command will be executed in a
  new sub-process spawned as child of the current shell. If the first argument
  contains a slash it will be interpretated as relative path-name of a binary
  or an executable shell-script to be processed in a new sub-shell.
  Finally, if the first-argument is not a valid file-name, an error message will
  be put to STDERR.

  Permissions, Modes:
  In order to be executable a script or command must either be set to execute
  privileges for the effective user or group or - in the case of a script called
  in the form "sh <filename>" - with sufficient read permissions.
  Permissions can be set using "chmod".
  (Since the shell is the only script-language present, the *magic cookie*
  "#!/bin/sh" may be absent. Permissions take precedence.)

  Variable Interpolation:
  Variables will be expanded in any double-quoted or unquoted term.
  Use $<varname> or ${<varname>} to retrieve the value of any defined variable.
  variables can be hidden from the shell using single-quotes or escapes with
  backslash ("\").

  Positional Parameters:
  In shell-scripts the term $<number> - where <number> is in the range 0-9 -
  expands to positional paramters. $0 will expand to the command or script name
  while the variable $1-$9 will give the value of the first argument and so on.

  Currently the system employs a number of special variables:
    GID      group-id
    HOME     home directory
    HOST     login-host
    PATH     command path
    PID      process id of current process environment
    PS       shell prompt
    UID      user-id
    USER     user-name (log-name)
    VERSION  os/term-version

  Special Files, Command History:
  There are two special files to the shell:
  The first is "etc/profile" which is executed by the login-shell on start up
  for initialization.
  The second one is "~/.history" where the command history is stored. (You can
  access the command history using cursor up/down in the command line.)

  * Arguments:
      <filename>  a script to be opened in a subshell
      <args>      currently, if the first argument is not a valid filename,
                  the arguments will be interpreted as arguments to be executed
                  by a new subshell.


  shell

  * Synopsis:
  JS/UIX-shell

  see "sh" for more.

  => see "sh".
```

```
splitmode

* Synopsis:
splitmode <mode>

displays a statusline to demonstrate screen splitting.
(splitting will be terminated by the next "clear" command.)

* Arguments:
    <mode>  "on"  switch statusline on
            "off" switch statusline off


stty

* Synopsis:
stty <option>

set terminal options.

* Options:
    -a         list all options
    -g         list all options in formated output
    [-]blink   [no] cursor blinking
    [-]block   [no] block cursor
    [-]smart   [no] smart console (minimal scrolling)
    [-]rows n  [re]set max. terminal line to n
    sane       reset to sane values


su

* Synopsis:
su <username>

switch the user.

* Arguments:
    <username>  user, name must consist of the characters [A-Za-z0-9_]
                only the first 8 characters are recognized (rest ignored).


time

* Synopsis:
time [-l|u]

diplays the time as local (default) or UTC
as: hours:minutes:seconds [UTC]

* Options:
    -l  local time (default)
    -u  UTC time


touch

* Synopsis:
touch <filenamename> {<filenamename>}

set the file last modified date (mdate) to current time.
if the file does'nt exist an empty file be created.
```

```
* Arguments:
    <filenamename>  name of the file to be modified or created.


type

* Synopsis:
type [-ipru|-n <num>] [<args>]

writes the given arguments back to the terminal in specified type style.

* Arguments:
    <args>  any text separated by any amount of space.

* Options:
    -n <num>     number representing the type style as a bit vector;
                 for details see the other options identifying styles
                 by the following characters:
    -p           plain     (0)
    -r           reverse   (1)
    -u           underline (2)
    -i           italics   (4)
    -s           stroke    (8)

    -> example:  "type -n 5 <args>" is same as "type -ir <args>".


unalias

* Synopsis:
unalias <name>

discards an alias defined by "alias".

* Arguments:
    <name>  the name of the alias. names must begin with a letter
            and use only the characters "A"-"Z", "0"-"9" or "_".


uname

* Synopsis:
uname

displays the system identification


unset

* Synopsis:
unset <varname>

discards a variable defined by "set".
reserved variables must not be discarded. (see "man set").

* Arguments:
    <varname>  the name of the variable. names must begin with a letter
               and use only the characters "A"-"Z", "0"-"9" or "_".


vi

* Synopsis:
vi [<filename>]
```

```
opens a (simple) implementation of the visual editor (vi).
current beta restrictions: no numeral modifiers, no search expressions.
as the standard vi this implementation is a modal application.
use <esc> to enter movements, ":" to enter the command-line, or one of the
insert-, append-, change-, replacement-keys to enter edit mode.
<esc> brings you always back to movements; leave with ":q!" without changes.

Basic Commands: (+<return>)

   :q[uit]            quit (if no changes made)
   :q[uit]!           forced quit, ignore changes
   :w [filename]      write [filename]
   :w! [filename]     forced write, overwrite existing files
   :wq[!] [filename]  forced write and quit
   :x[!] [filename]   like "wq" - write only when changes have been made
   :ZZ                like "x"
   :1                 display first line
   :$                 display last line
   :N                 display line N

Cursor Movements:

   h  left  (or cursor)     k  line up   (or cursor)
   l  right (or cursor)     j  line down (or cursor)

   0  go to the first character of the current line
   ^  go to the first non-blank character of the current line
   $  go to the end of the current line
   -  go up one line and to the first non-blan character
   +  go down one line and to the first non-blan character
   w  one word forward
   b  one word backward
   e  forward to end of word
   z  display current-line on top

Editing Comands:

  a   append after cursor
  A   append after end of line
  i   insert before cursor
  I   insert before first non-blank character of the line
  o   open a new line below the current line
  O   open a new line above the current line
  c[motion]  change text (insert between old and new cursor position)
             (this command is currently restricted to the same line)
  cc  change the current line
  C   change to the end of the current line
  R   replace text

Deleting, Copy and Paste, Undo:

  x   delete character under (and after) the cursor
  X   delete character before the cursor
  dd  delete current line and put it in the copy buffer
  D   delete to end of line
  J   join lines (delete new line at end of the current line)

  Copy & Paste (currently restricted to lines only):

  yy  yank current line (put to copy buffer)
  p   put (insert) copy buffer to end line after current line
  P   put (insert) copy buffer above current line

  u  undo last change
  U  redo last undo
```

```
 This implementation accepts pipes as valid input. If called as "view"
 vi is opened in read only mode.

 * Arguments:
     <filename>  a file to be opened.


 view

 * Synopsis:
 view [<filename>]

 synonym for "vi" in view-mode (read only mode).
 files must be saved with new name or changes will be lost.

 => see "vi".


 wc

 * Synopsis:
 wc [-clw]

 word count.
 counts the characters, words, and lines of a specified file or from STDIN.

 * Options:
     -c  count characters
     -l  count lines
     -w  count words


 web

 * Synopsis:
 web [-n] [<url>]

 synonym for "browse".

 => see "browse".


 which

 * Synopsis:
 which <command>

 evaluates the command path for the given command.
 if the command is found it is displayed with full path-name.

 * Arguments:
     <command>  name of the command to be found.


 write

 * Synopsis:
 write <args>

 writes the arguments back to the terminal using type styles.

 * Arguments:
     <args>       any arguments (treated as strings separated by spaces).
                  type styles can be specified as follows:
                   %+<typestyle>  switch type style on
                   %-<typestyle>  switch type style off
                   %n             new line
                   %%             escaped "%"
                  where <typestyle> is marked by one the following characters:
                   p  plain (+p discards all active styles, -p is ineffective)
                   r  reverse
                   u  underline
                   i  italics
                   s  strike.
                  type styles may overlap.

     -> example:  write "Do not use %+rREVERSE%-r for 100%% of the text."


 (c) mass:werk 2003; <http://www.masswerk.at>
```