

GNU screen terminal commands

cheat sheet

Author: Yuri Slobodyanyuk, admin@yurisk.info

Command	Description
<code>~/.screenrc & /etc/screenrc</code>	Commands the screen runs on start up.
<code>screen -ls</code>	List active screen sessions
<code>screen -Q windows</code>	List windows' names inside screen session
<code>screen -S <session name></code>	Create a new screen session with the name <session name>
<code>screen -x</code> <code>screen -r <session name></code>	Attach to the running session, also by its name
<code>screen -dRR</code>	Attach to the screen session, detach on other display if attached. If no session exists, will create a new one.
<code>C-a d</code>	Detach from the session, session keeps running. Here, and further C means Ctrl.
<code>C-a c</code>	Create new window in the session.
<code>C-a C-a</code>	Switch to the previous window.
<code>C-a "</code>	List all windows with option to highlight and enter any of them.
<code>C-w</code>	Show a list of active windows with their numbers.
<code>C-a <number></code>	Switch to the window number number .
<code>C-a '</code>	Switch to the window by its name.
<code>C-a n</code>	Switch to the next window.
<code>C-a p</code>	Switch to the previous window.
<code>exit</code>	Exit and close current window. If it was the last window in a session, exits screen terminating the session.
<code>C-a k</code>	Kill the current window forcefully (not recommended).
<code>C-a : quit</code>	Quit screen session completely terminating it. Alternatively - exit all screen windows.
<code>C-a A</code>	Rename current window.

Command	Description
C-a S	Split windows display horizontally. Use C-a c to create a new window inside the new split or C-X to close this part of split.
C-a 	Split windows display vertically. Available starting screen 4.01, i.e. not available on Mac 2020 which still uses screen 4.00.
C-a tab	Jump to the next region in a split window display.
C-a X	Remove the region in focus.
C-a [or C-a <esc>	Enter buffer navigation mode to scroll output buffer, copy, edit and paste later. Navigation commands as per vim if Vim is set as editor. <esc> to leave the buffer mode.
<space>	Start/stop selection while in the buffer mode to select the text. Press <space> or <Enter> to copy the selected text. E.g. to select/copy the whole buffer: C-a [gg <space> G <space> <esc>
C-a]	Paste the selected text at the cursor of the terminal, or create a new window and say start Vim there and paste into it while in Insert mode.
C-a h	Make screenshot (as a text) of the currently visible terminal window and save the output to hardcopy.<n> , where <n> is auto-incrementing number of your window.
C-a H	Start/end logging all output of the current window into a file screenlog.N where N is the window number. The data is appended, not overwritten if the file exists. Contents displayed until enabling the logging is not logged.
C-a a	Send Ctrl-a sequence to the running command, useful to jump to the line start in bash.
C-a M	Monitor window for activity. When enabled, will notify you of any activity while you work in other window.
C-a _	Monitor window for 30 seconds of silence, will notify you in any other window as `Window 0: silence for 30 seconds`
C-a ?	Show all key bindings help.

Command	Description
Save session state	This is not possible. If you use the same layout each session, you can put start up commands to re-create it in <code>.screenrc</code> file in your home directory, but still - you cannot save the current session state, i.e. contents of the windows and their layout.
Sharing session (e.g. for pair programming)	
<p>Original session (say <i>user1</i>):</p> <ol style="list-style-type: none"> 1. Set suid root bit on <code>screen</code> binary: <code>sudo chmod +s /usr/bin/screen</code> 2. Inside session you want to share: <code>C-a :</code> then <code>multiuser on</code> to enable sharing session. 3. Add usernames to share the session with: <code>C-a : acladd <username></code> <p>Connecting user (say <i>user2</i>):</p> <ol style="list-style-type: none"> 1. Run in shell: <code>screen -x <sharing username>/</code>, in our example <code>screen -x user1/</code> 	<p>Sets up sharing the session. Another user connecting to the session views real-time its output, can enter and run commands himself. Also see <code>aclchg</code>, <code>acldel</code>, <code>aclgrp</code> for controlling what the connecting user can and cannot do. E.g. to remove <i>write</i> permissions from all users on all windows: <code>:aclchg * -w #</code></p>
C-a *	See who is connected to your shared screen session.