Unix Annotations

Annotations

Table of Contents

💬 Notes

- Que sí, que ya sé que el amor es ciego. Pero yo no, joder
- Social Engineering: Because there is no patch for human stupidity.
- Nobody wants to say how this works. Maybe nobody knows ... >> "VIDEOADAPTOR SECTION" on xorg.conf manual ファミコン
- "This is Unix-Land. In quiet nights, you can hear the Windows machines reboot"
- "Es prácticamente imposible enseñar programación correctamente a estudiantes que han estado expuestos al lenguaje BASIC con anterioridad.
  Como potenciales programadores, tienen la mente mutilada sin esperanza alguna de regeneración"
  Edsger Dijkstra.
- No extraña que los explotadores de profesión, hayan decidido incluir en sus actividades a la media humanidad que les faltaba, pero sorprende que algunas mujeres consideren deseable esta situación: además del castigo bíblico propio, quieren pasar por el que les cayó a los hombres. Y, encima, en modo alguno comprenden que sea un castigo, una situación nada deseable.
  Como si fueran calvinistas se portan, convencidas de que el trabajo enriquece.
  Los hombres, más entrenados, sabemos de antiguo que sólo es posible enriquecerse haciendo trabajar a los demás.
- SOFTWARE: Es el espíritu insuflado en la dura mollera de la máquina. A veces, un clásico espíritu burlón o poltergeist.
  Arturo Rosby
- "«Manos a la obra, señor Freeman, manos a la obra. No quiero decir que se haya dormido en los laureles en el trabajo. Nadie se merece más un alto en el camino y todos los esfuerzos realizados habrían sido en vano hasta que... Hmmm... Bueno... Digamos que otra vez ha llegado su hora. El hombre adecuado en el sitio equivocado puede cambiar el rumbo del mundo. Despierte, señor Freeman. Despierte y huela las cenizas»"
  G-Man al comienzo de Half Life 2.
- Los Grigori (del griego egr□goroi, que significa Observadores o vigilantes), también conocidos como hijos de Elohim (en hebreo בני האלהים, bnei ha'elohim);
  son un grupo de ángeles caídos de la mitología judeocristiana mencionados en algunos textos apócrifos Bíblicos y en el Libro del Génesis.

En estos textos se menciona que los Grigori fueron seres que se aparearon
con las "hijas del hombre" (en hebreo banot ha'adam);

naciendo de esta unión una raza de gigantes conocida como los Nephilim.

- "Si los tontos volaran, el cielo se oscurecería"

> "Recuperar alkalinidad en el cuerpo"
> 2 partes de agua
> 1 parte de jugo de limon
> 1 pizca de carbonato

- "Vade retro satana"

> The Latin text says:

> Crux Sancti Patris Benedicti / Cruz del Santo Padre Benito
> Crux Sacra Sit Mihi Lux / Mi luz sea la cruz santa
> Non Draco Sit Mihi Dux / No sea el demonio mi guía
> Vade Retro Satana / ¡Apártate, Satanás!
> Numquam Suade Mibi Vana / No sugieras cosas vanas
> Sunt Mala Quae libas / Pues maldad es lo que brindas
> Ipse Venena Bibas / Bebe tú mismo el veneno

> Crux sancta sit mihi lux / Non draco sit mihi dux
> Vade retro satana / Numquam suade mihi(mibi) vana
> Sunt mala quae libas / Ipse venena bibas
> In approximate translation:

> Let the Holy Cross be my light / Let not the dragon be my guide
> Step back Satan / Never tempt me with vain things
> What you offer me is evil / You drink the poison yourself.

> La Santa Cruz sea mi luz, /no sea el dragón mi señor.
> ¡Apártate, Satanás! /nunca me atraigas con engaños,
> maldad es tu carnada, /bebe tu propio veneno.

> INRI >> Iesus Nazarenus Rex Iudaeorum

どうもありがとうミスターロボット (dōmo arigatō misutā robotto)

My firts name
Hiragana katakana
へすす ヘスス
My second name
Hiragana katakana
あるべると アルベルト

Her name firts A then Y
Hiragana katakana
あれはんどら アレハンドラ

やみれと ヤミレト

Remenber this tools on your journey

```
chntpw               NT SAM password recovery utility
ms-sys               Create DOS/MS-compatible boot records
foremost             Recover files by "carving" them from a raw
hdparm               Utility to get/set (E)IDE/SATA/SAS device parameters
ide-smart            IDE S.M.A.R.T. test and query tool
john                 Password cracker
memtester            Userspace utility to test for faulty memory subsystem
testdisk             Tool to check and undelete partition
targa                MS Windows95/98/NT testing kit
scrounge-ntfs        Data recovery program for NTFS file systems
dmidecode            dumping bios information
galleta              An Internet Explorer cookie forensic analysis tool
pasco                An Internet Explorer cache forensic analysis tool
lm-sensors                    check temp of your box
smartmontools               smart-monitor
cpuburn                         stress your cpu
                      run desired program in background [ _repeat_ for SMP]:
      `burnP6 || echo $? &`
surfraw                       example surfraw -browser=w3m google "query"
gotmail                         download e-mail from hotmail
wapua                             web browser for wap
fcrackzip                       pass-cracker
fondu                             convert between mac fonts format
encfs                             encrypt fs ?
electricsheep               just awesone !
fatresize
archmage                        extract and convert chm files to html , txt or pdf
flasm                             assembler and disassembler for flash-(swf) bytecode
badblocks           for cheking bad blocks
xsw                                  presentations sowftware very easy to use
```

Editors Section

Improvements of Vim

- add Vundle plugin manager

```
git clone https://github.com/VundleVim/Vundle.vim.git ~/.vim/bundle/Vundle.vim
```

- Add the vimrc

```
""""""""""""""""""""""""""""""""""""""""""
" Allan MacGregor Vimrc configuration
""""""""""""""""""""""""""""""""""""""""""
set encoding=utf8
""""" START Vundle Configuration
" Disable file type for vundle
filetype off                  " required

" set the runtime path to include Vundle and initialize
set rtp+=~/.vim/bundle/Vundle.vim
call vundle#begin()

" let Vundle manage Vundle, required
Plugin 'gmarik/Vundle.vim'
```

```
" Utility
Plugin 'scrooloose/nerdtree'
Plugin 'majutsushi/tagbar'
Plugin 'ervandew/supertab'
Plugin 'BufOnly.vim'
Plugin 'wesQ3/vim-windowswap'
Plugin 'SirVer/ultisnips'
Plugin 'junegunn/fzf.vim'
Plugin 'junegunn/fzf'
Plugin 'godlygeek/tabular'
Plugin 'ctrlpvim/ctrlp.vim'
Plugin 'benmills/vimux'
Plugin 'jeetsukumaran/vim-buffergator'
Plugin 'gilsondev/searchtasks.vim'
Plugin 'Shougo/neocomplete.vim'
Plugin 'tpope/vim-dispatch'

" Generic Programming Support
Plugin 'jakedouglas/exuberant-ctags'
Plugin 'honza/vim-snippets'
Plugin 'Townk/vim-autoclose'
Plugin 'tomtom/tcomment_vim'
Plugin 'tobyS/vmustache'
Plugin 'janko-m/vim-test'
Plugin 'maksimr/vim-jsbeautify'
Plugin 'vim-syntastic/syntastic'
Plugin 'neomake/neomake'

" Markdown / Writting
Plugin 'reedes/vim-pencil'
Plugin 'tpope/vim-markdown'
Plugin 'jtratner/vim-flavored-markdown'
Plugin 'LanguageTool'

" Git Support
Plugin 'kablamo/vim-git-log'
Plugin 'gregsexton/gitv'
Plugin 'tpope/vim-fugitive'
"Plugin 'jaxbot/github-issues.vim'

" PHP Support
Plugin 'phpvim/phpcd.vim'
Plugin 'tobyS/pdv'

" Erlang Support
Plugin 'vim-erlang/vim-erlang-tags'
Plugin 'vim-erlang/vim-erlang-runtime'
Plugin 'vim-erlang/vim-erlang-omnicomplete'
Plugin 'vim-erlang/vim-erlang-compiler'

" Elixir Support
Plugin 'elixir-lang/vim-elixir'
Plugin 'avdgaag/vim-phoenix'
Plugin 'mmorearty/elixir-ctags'
Plugin 'mattreduce/vim-mix'
Plugin 'BjRo/vim-extest'
Plugin 'frost/vim-eh-docs'
Plugin 'slashmili/alchemist.vim'
Plugin 'tpope/vim-endwise'
Plugin 'jadercorrea/elixir_generator.vim'
```

```
" Elm Support
Plugin 'lambdatoast/elm.vim'

" Theme / Interface
Plugin 'AnsiEsc.vim'
Plugin 'ryanoasis/vim-devicons'
Plugin 'vim-airline/vim-airline'
Plugin 'vim-airline/vim-airline-themes'
Plugin 'sjl/badwolf'
Plugin 'tomasr/molokai'
Plugin 'morhetz/gruvbox'
Plugin 'zenorocha/dracula-theme', {'rtp': 'vim/'}
Plugin 'junegunn/limelight.vim'
Plugin 'mkarmona/colorsbox'
Plugin 'romainl/Apprentice'
Plugin 'Lokaltog/vim-distinguished'
Plugin 'chriskempson/base16-vim'
Plugin 'w0ng/vim-hybrid'
Plugin 'AlessandroYorba/Sierra'
Plugin 'daylerees/colour-schemes'
Plugin 'effkay/argonaut.vim'
Plugin 'ajh17/Spacegray.vim'
Plugin 'atelierbram/Base2Tone-vim'
Plugin 'colepeters/spacemacs-theme.vim'

call vundle#end()            " required
filetype plugin indent on    " required
""""" END Vundle Configuration

""""""""""""""""""""""""""""""""""""""""""
" Configuration Section
""""""""""""""""""""""""""""""""""""""""""
set nowrap

" OSX stupid backspace fix
set backspace=indent,eol,start

" Show linenumbers
set number

" Set Proper Tabs
set tabstop=4
set shiftwidth=4
set smarttab
set expandtab

" Always display the status line
set laststatus=2

" Enable Elite mode, No ARRRROWWS!!!!
let g:elite_mode=1

" Enable highlighting of the current line
set cursorline

" Theme and Styling
syntax on
set t_Co=256
```

```
" if (has("termguicolors"))
"   set termguicolors
" endif

let base16colorspace=256  " Access colors present in 256 colorspace
colorscheme spacegray

let g:spacegray_underline_search = 1
let g:spacegray_italicize_comments = 1

" Vim-Airline Configuration
let g:airline#extensions#tabline#enabled = 1
let g:airline_powerline_fonts = 1
let g:airline_theme='hybrid'
let g:hybrid_custom_term_colors = 1
let g:hybrid_reduced_contrast = 1

" Syntastic Configuration
set statusline+=%#warningmsg#
set statusline+=%{SyntasticStatuslineFlag()}
set statusline+=%*

let g:syntastic_always_populate_loc_list = 1
let g:syntastic_auto_loc_list = 1
let g:syntastic_check_on_open = 1
" let g:syntastic_check_on_wq = 0
" let g:syntastic_enable_elixir_checker = 1
" let g:syntastic_elixir_checkers = ["elixir"]

" Neomake settings
autocmd! BufWritePost * Neomake
let g:neomake_elixir_enabled_makers = ['mix', 'credo', 'dogma']

" Vim-PDV Configuration
let g:pdv_template_dir = $HOME ."/.vim/bundle/pdv/templates_snip"

" Markdown Syntax Support
augroup markdown
    au!
    au BufNewFile,BufRead *.md,*.markdown setlocal filetype=ghmarkdown
augroup END

" Vim-Alchemist Configuration
let g:alchemist#elixir_erlang_src = "/Users/amacgregor/Projects/Github/alchemist-source"
let g:alchemist_tag_disable = 1

" Vim-Supertab Configuration
let g:SuperTabDefaultCompletionType = "<C-X><C-O>"

" Settings for Writting
let g:pencil#wrapModeDefault = 'soft'   " default is 'hard'
let g:languagetool_jar  = '/opt/languagetool/languagetool-commandline.jar'

" Vim-pencil Configuration
augroup pencil
  autocmd!
  autocmd FileType markdown,mkd call pencil#init()
  autocmd FileType text         call pencil#init()
augroup END
```

```
" Vim-UtilSnips Configuration
" Trigger configuration. Do not use <tab> if you use https://github.com/Valloric/YouCompleteMe.
let g:UltiSnipsUsePythonVersion = 3
let g:UltiSnipsExpandTrigger="<tab>"
let g:UltiSnipsJumpForwardTrigger="<c-b>"
let g:UltiSnipsJumpBackwardTrigger="<c-z>"
let g:UltiSnipsEditSplit="vertical" " If you want :UltiSnipsEdit to split your window.

" Vim-Test Configuration
let test#strategy = "vimux"

" Neocomplete Settings
let g:acp_enableAtStartup = 0
let g:neocomplete#enable_at_startup = 1
let g:neocomplete#enable_smart_case = 1
let g:neocomplete#sources#syntax#min_keyword_length = 3

" Define dictionary.
let g:neocomplete#sources#dictionary#dictionaries = {
    \ 'default' : '',
    \ 'vimshell' : $HOME.'/.vimshell_hist',
    \ 'scheme' : $HOME.'/.gosh_completions'
        \ }

" Define keyword.
if !exists('g:neocomplete#keyword_patterns')
    let g:neocomplete#keyword_patterns = {}
endif
let g:neocomplete#keyword_patterns['default'] = '\h\w*'

function! s:my_cr_function()
  return (pumvisible() ? "\<C-y>" : "" ) . "\<CR>"
  " For no inserting <CR> key.
  "return pumvisible() ? "\<C-y>" : "\<CR>"
endfunction

" Close popup by <Space>.
"inoremap <expr><Space> pumvisible() ? "\<C-y>" : "\<Space>"

" AutoComplPop like behavior.
"let g:neocomplete#enable_auto_select = 1


" Enable omni completion.
autocmd FileType css setlocal omnifunc=csscomplete#CompleteCSS
autocmd FileType html,markdown setlocal omnifunc=htmlcomplete#CompleteTags
autocmd FileType javascript setlocal omnifunc=javascriptcomplete#CompleteJS
autocmd FileType python setlocal omnifunc=pythoncomplete#Complete
autocmd FileType xml setlocal omnifunc=xmlcomplete#CompleteTags

" Enable heavy omni completion.
if !exists('g:neocomplete#sources#omni#input_patterns')
  let g:neocomplete#sources#omni#input_patterns = {}
endif
"let g:neocomplete#sources#omni#input_patterns.php = '[^. \t]->\h\w*\|\h\w*::'
"let g:neocomplete#sources#omni#input_patterns.c = '[^.[:digit:] *\t]\%(\.\|->\)'
"let g:neocomplete#sources#omni#input_patterns.cpp = '[^.[:digit:] *\t]\%(\.\|->\)\|\h\w*::'

" For perlomni.vim setting.
" https://github.com/c9s/perlomni.vim
let g:neocomplete#sources#omni#input_patterns.perl = '\h\w*->\h\w*\|\h\w*::'
```

```
" Elixir Tagbar Configuration
let g:tagbar_type_elixir = {
    \ 'ctagstype' : 'elixir',
    \ 'kinds' : [
        \ 'f:functions',
        \ 'functions:functions',
        \ 'c:callbacks',
        \ 'd:delegates',
        \ 'e:exceptions',
        \ 'i:implementations',
        \ 'a:macros',
        \ 'o:operators',
        \ 'm:modules',
        \ 'p:protocols',
        \ 'r:records',
        \ 't:tests'
    \ ]
    \ }

" Fzf Configuration
" This is the default extra key bindings
let g:fzf_action = {
  \ 'ctrl-t': 'tab split',
  \ 'ctrl-x': 'split',
  \ 'ctrl-v': 'vsplit' }

" Default fzf layout
" - down / up / left / right
let g:fzf_layout = { 'down': '~40%' }

" In Neovim, you can set up fzf window using a Vim command
let g:fzf_layout = { 'window': 'enew' }
let g:fzf_layout = { 'window': '-tabnew' }

" Customize fzf colors to match your color scheme
let g:fzf_colors =
\ { 'fg':      ['fg', 'Normal'],
  \ 'bg':      ['bg', 'Normal'],
  \ 'hl':      ['fg', 'Comment'],
  \ 'fg+':     ['fg', 'CursorLine', 'CursorColumn', 'Normal'],
  \ 'bg+':     ['bg', 'CursorLine', 'CursorColumn'],
  \ 'hl+':     ['fg', 'Statement'],
  \ 'info':    ['fg', 'PreProc'],
  \ 'prompt':  ['fg', 'Conditional'],
  \ 'pointer': ['fg', 'Exception'],
  \ 'marker':  ['fg', 'Keyword'],
  \ 'spinner': ['fg', 'Label'],
  \ 'header':  ['fg', 'Comment'] }

" Enable per-command history.
" CTRL-N and CTRL-P will be automatically bound to next-history and
" previous-history instead of down and up. If you don't like the change,
" explicitly bind the keys to down and up in your $FZF_DEFAULT_OPTS.
let g:fzf_history_dir = '~/.local/share/fzf-history'

""""""""""""""""""""""""""""""""""""""""""""
" Mappings configurationn
""""""""""""""""""""""""""""""""""""""""""""
map <C-n> :NERDTreeToggle<CR>
map <C-m> :TagbarToggle<CR>
```

```
" Omnicomplete Better Nav
inoremap <expr> <c-j> ("\<C-n>")
inoremap <expr> <c-k> ("\<C-p>")

" Neocomplete Plugin mappins
inoremap <expr><C-g>     neocomplete#undo_completion()
inoremap <expr><C-l>     neocomplete#complete_common_string()

" Recommended key-mappings.
" <CR>: close popup and save indent.
inoremap <silent> <CR> <C-r>=<SID>my_cr_function()<CR>

" <TAB>: completion.
inoremap <expr><TAB>  pumvisible() ? "\<C-n>" : "\<TAB>"

" <C-h>, <BS>: close popup and delete backword char.
inoremap <expr><C-h> neocomplete#smart_close_popup()."\<C-h>"
inoremap <expr><BS> neocomplete#smart_close_popup()."\<C-h>"

" Mapping selecting Mappings
nmap <leader><tab> <plug>(fzf-maps-n)
xmap <leader><tab> <plug>(fzf-maps-x)
omap <leader><tab> <plug>(fzf-maps-o)

" Shortcuts
nnoremap <Leader>o :Files<CR>
nnoremap <Leader>O :CtrlP<CR>
nnoremap <Leader>w :w<CR>

" Insert mode completion
imap <c-x><c-k> <plug>(fzf-complete-word)
imap <c-x><c-f> <plug>(fzf-complete-path)
imap <c-x><c-j> <plug>(fzf-complete-file-ag)
imap <c-x><c-l> <plug>(fzf-complete-line)

" Vim-Test Mappings
nmap <silent> <leader>t :TestNearest<CR>
nmap <silent> <leader>T :TestFile<CR>
nmap <silent> <leader>a :TestSuite<CR>
nmap <silent> <leader>l :TestLast<CR>
nmap <silent> <leader>g :TestVisit<CR>

" Vim-PDV Mappings
autocmd FileType php inoremap <C-p> <ESC>:call pdv#DocumentWithSnip()<CR>i
autocmd FileType php nnoremap <C-p> :call pdv#DocumentWithSnip()<CR>
autocmd FileType php setlocal omnifunc=phpcd#CompletePHP

" Disable arrow movement, resize splits instead.
"if get(g:, 'elite_mode')
"       nnoremap <Up>    :resize +2<CR>
"       nnoremap <Down>  :resize -2<CR>
"       nnoremap <Left>  :vertical resize +2<CR>
"       nnoremap <Right> :vertical resize -2<CR>
"endif

map <silent> <LocalLeader>ws :highlight clear ExtraWhitespace<CR>
```

```
" Advanced customization using autoload functions
inoremap <expr> <c-x><c-k> fzf#vim#complete#word({'left': '15%'})

" Vim-Alchemist Mappings
autocmd FileType elixir nnoremap <buffer> <leader>h :call alchemist#exdoc()<CR>
autocmd FileType elixir nnoremap <buffer> <leader>d :call alchemist#exdef()<CR>
```

- then inside vim type:

```
run :PluginInstall "OR " :BundleInstall
```

> OR To install from command line:

```
  vim +PluginInstall +qall
```

more details in https://github.com/VundleVim/Vundle.vim

Some package issues *UltiSnips* adding this option to your .vimrc: for force the version

```
let g:UltiSnipsUsePythonVersion = 2
```

maybe you need install python support in vim

```
sudo apt-get install vim-python-jedi
```

YouCompleteMe code appended randomly when moving around with arrow keys in insert mode

```
pumvisible() ? "\pumvisible" : "\
```

This occurs often but randomly and it will persist for the entirety of the session. Loading or creating a different file will not fix the issue. It only goes away if neovim is relaunched.
Just as a reference to anyone who comes across this with the same issue, I found that the plugin "Townk/vim-autoclose" was causing the conflict. This plugin is old and hasn't been updated in a while and causes quite a few other conflicts.
I removed it and used "Raimondi/delimitMate" instead.

```
# check if vim has python
vim --version | grep --color python
```

- then are you done

🎵 Audio Section

Audio Recording and Encoding in Linux

> src : http://mocha.freeshell.org/audio.html

Linux is a professional class operating system, so if you plan to do audio recording and encoding then you chose the proper system.

I use Pulseaudio on some boxes and just Alsa on other boxes. As of this writing (Summer 2011) Pulseaudio has all the major bugs worked out, so all the old web discussions about it causing issues are no longer relevant as far as I'm concerned.

When using Pulseaudio you are strongly advised to install the GUI management tools in order to control the audio streams and volumes and so forth. You can do it at the command line which is great, but it is a bit cumbersome unless you are using a recording script (which I will demonstrate later).

If only using Alsa, then there is usually an added step of specifying the audio device in the recording command. For example, arecord expects a "-D plughw:0,0" or "-D hw:0,0". You also need to use the alsamixer utility to control the selection of external sources and gain levels. Actually, gurus will also use alsamixer on Pulseaudio systems as well when recording from sources such as line-in or microphones.

arecord examples
ffmpeg examples
speex format
voice activated recording
useful scripts

Examples of recording and piping the output to a compressed format:

```
arecord -v -f cd -t raw | lame -r - output.mp3

arecord -v -f cd -t raw | lame -r -b 192 - output.mp3

arecord -v -f cd -t raw | lame -r -m s -a -b 64 - output.mp3

arecord -v -f cd -t raw | lame -r -h -V 0 -b 128 -B 224 - output.mp3

arecord -v -f cd -t raw | lame -r --preset extreme - output.mp3

arecord -vv -f cd output.wav

arecord -f dat output.wav

arecord -f S16_LE -c1 -r22050 -t raw | lame -r -s 22.05 -m m -b 64 -
Output.mp3

arecord -f S16_LE -c1 -r22050 -t raw | oggenc - -r -C 1 -R 22050 -q 2 -o
Output.ogg

arecord -f S16_LE -c1 -r22050 -t raw -d xxxx | oggenc-aotuvb5d - -r -C 1 -R
22050 -o Output.ogg
```

(-d xxxx limits the recording time in seconds, useful for scripts)
(oggenc -q option is -1 to 10 where higher numbers mean higher quality/bitrate, the default is 3)

It's probably a good idea to read the man pages of arecord, lame, and oggenc (the aotuv version is just an enhanced version). Basically the "-f" in arecord allows you to select a pre-packaged format like CD (44.1 kHz, 16 bits little endian, 2 channels) or DAT (48 kHz, 16 bits little endian, 2 channels), or you can get specific and choose S16_LE (16 bit little endian) -c1 (mono) -r22050 (22.050 kHz sampling rate), and so forth.

What you pipe arecord out to needs to logically match how you are recording with arecord. Notice that when I record in "CD" or "DAT" formats I choose

high quality options in lame. When I use lower quality options in arecord, I also use lower encoding demand parameters in lame and oggenc.

Digital audio recording is all about specifying a quantizing method, sample rate, number of channels, bits per channel, and compression schemes and bitrates. Read the man pages.

Using ffmpeg to record and encode:

```
#!/bin/bash

ffmpeg -f alsa -ac 2 -ar 44100 -ab 160k -i pulse -acodec libvorbis
OUTPUT.ogg

ffmpeg -f alsa -ac 2 -ar 44100 -ab 160k -i pulse -acodec libmp3lame
OUTPUT.mp3

ffmpeg -f alsa -ac 2 -ar 44100 -i pulse OUTPUT.flac
```

The placement of "-i pulse" and "-acodec xxx" is critical in order for ffmpeg to encode in 2 channels, if -i pulse was at the beginning of the command line it would only see mono audio. Use the Pulseaudio tools to select the audio source to record, or use the option to specify an audio device (-i hw:0,0) if using Alsa only.

Some record/encode examples with the Speex format:

Example of transcoding an MP3 to 16 kHz speex:

```
sox INPUT.MP3 -t wav -r 16000 - | speexenc - --denoise OUTPUT.SPX
```

Example encoding of wav file. Speex prefers input file sampling rates of 8 or 16 kHz:

```
speexenc --vbr --denoise INPUT.WAV OUTPUT.SPX
```

Piping arecord to speexenc (realtime speex encoding method):

```
$ arecord -f S16_LE -c1 -r8000 -t raw | speexenc - --vad OUTPUT.SPX
```

Sound Activated Recording, VOX recording, Sound Operated Recording:

This stuff goes by many names. Basically, it describes a method to only record incoming audio signals that meet or exceed a user-specified threshold level. It's useful when recording from police scanner radios or microphones that are setup to record intermittent audio.

On Winblows there are numerous programs to accomplish this. On Linux there

are only two methods that I've discovered that work properly.

Audacity. Audacity now has an option under the Transport menu to do "Sound Activated Recording" and allows you to set an input threshold dB level. It appears to be stable, I've set it up to record from my motherboard's line-in for up to 20 hour periods with no crashes or memory leaks.

VACR. This is a python script that works well from the command line. It has limted options but the default threshold detection method works very well. There are a few places in the script to specify the sample rate and output file name. There is also a line to tweak the threshold level gate.

Update 11/23/11. I found the script below on a German language forum. This method uses sox (major points for that) and works very well. The big advantages of this script is that sox lets you encode how you want (fine grain control), and you can also do post-processing, in this case the creator had a normalizing section and also generated a nice spectra energy graph. I've been using this script in some form or another for about a month now.

```
#!/bin/bash
#source http://www.sis-germany.de/index.php?page=Thread&amp;threadID=1444

NAME=`date +%m-%d-%Y_%H-%M-%S`

#choose a method below, either the decibel or % of level method
#also choose number of channels, sample rate, encoding, etc..
#you need to find a level that works for your hardware

#rec -c 1 -r 22050 $NAME.wav silence 1 0 -22d -1 00:00:05 -22d
#rec -c 1 -r 22050 $NAME.wav silence 1 0 8% -1 00:00:05 8%
rec -c 1 -r 22050 $NAME.mp3 silence 1 0 8% -1 00:00:05 8%
#rec -c 1 -r 22050 $NAME.mp3 silence 1 0 25% -1 00:00:05 25%

#uncomment appropriate line below for normalizing when finished

#echo "Normalize..."
#sox $NAME.wav $NAME-norm.wav gain -n -1
#sox $NAME.mp3 $NAME-norm.mp3 gain -n -1

#uncomment appropriate line below for a spectral graph if desired

#echo "Calculating Spectrogram..."
#sox $NAME.wav -n spectrogram -x 1024 -y 768 -z 100 -t "$NAME.wav" -c '' -o
$NAME.png
#sox $NAME.mp3 -n spectrogram -x 1024 -y 768 -z 100 -t "$NAME.mp3" -c '' -o
$NAME.png
#echo "Done."
```

Script methods:

Here is a script to record from the Pulseaudio monitor source. The monitor source is basically the source that you hear the audio coming from out of your speakers.

```
#!/bin/bash
# Records the PulseAudio monitor channel.
# Uncomment the wav or mp3 output extension for OUTFILE then specify if a
# particular bitrate is needed for mp3

if [ -n "$1" ]; then
`OUTFILE="$1"
else
  `IME=$(date +%d-%b-%y_%H%M-%Z)
  #Choose wav or mp3 output
  #OUTFILE="recording_$TIME.wav"
  OUTFILE="recording_$TIME.mp3"
fi

# Get sink monitor:
MONITOR=$(pactl list | grep -A2 '^Source #' | grep "Name: .*\\.monitor$" | awk'{print $NF}' | tail -n1);
# Record it raw, and convert to a wav, uncomment the method you need below.
echo "Recording. Ctrl-C or close window to stop";
#44.1 kHz, 16 bit little endian, 2 channels, line below is the deprecated method
#parec -d "$MONITOR" | sox -t raw -r 44100 -sLb 16 -c 2 - "$OUTFILE"
#Defaults to 128 kbps when encoding directly to mp3
#parec -d "$MONITOR" | sox -t raw -r 44100 -b 16 -L -e signed -c 2 - "$OUTFILE"
#-C192 uses 192 kbps when encoding directly to mp3
parec -d "$MONITOR" | sox -t raw -r 44100 -b 16 -L -e signed -c 2 - -C192 "$OUTFILE"
```

Here is an example script that grabs an Internet stream and converts it to ogg. This is a useful script for a cron job.

```
#!/bin/bash
# set up the variables below as needed, they're self explanitory
DATE1=$(date +"%d%m%Y")
DATE2=$(date +"%d-%m-%Y")
STREAM=http://www.123.com/xxxxx.ram
DURATION=2.2h
MUSIC_DIR=$HOME/temp

cd $MUSIC_DIR
# Download the stream and convert it to wave format:
mplayer -cache 2048 -playlist $STREAM \
        -vc null -vo null -ao pcm:fast:waveheader:file=output.wav &amp;

sleep $DURATION # Length of the program being recorded as background.
kill $!         # End the most recently backgrounded job = mplayer

# Convert to ogg format and place the appropriate tags:

oggenc -q 6 output.wav \
         -t "title string: $DATE2" \
         -l "album string" \
         -a "artist string" \
         -o output_$DATE1.ogg
rm output.wav
```

Server Section

Virtual Servers in Apache

Introduction

The Apache web server is the most popular way of serving web content on the internet. It accounts for more than half of all active websites on the internet and is extremely powerful and flexible.

Setting the rules

First, useradd creates a new user. As you (iain) already exist, you want to call usermod instead. So that would be:

```
sudo usermod -aG www-data ambagasdowa
addgroup www-data
```

(note the -a on Debian-based servers (Ubuntu included) that will add you to that group, and keep your membership to other groups.
Forget it and you will belong to the www-data group only - could be a bad experience if one of them was wheel.
On SUSE-type servers the option is -A instead of -aG so read man usermod carefully to get it right.)

Second, you don't want apache to have full rw access to /var/www: this is potentially a major security breach.
As a general rule, allow only what you need, and nothing more (principle of least privilege).
In this case, you need apache (www-data) and you (www-data group) to write (and read) in /var/www/example.com/public_html, so

```
sudo chown -R www-data:www-data /var/www/example.com/public_html
sudo chmod -R 770 /var/www/example.com/public_html
```

Edit: to answer your original question, yes, any member of www-data can now read and execute /var/www
(because the last bit of your permissions is 5 = read + exec).
But because you haven't used the -R switch, that applies only to /var/www,
and not to the files and sub-directories it contains. Now, whether they can write is another matter,
and depends on the group of /var/www, which you haven't set. I guess it is typically root:root, so no,
they (probably) can't write.

Edit on 2014-06-22: added a note that -aG option is valid on Debian-based servers.
It apparently varies with the distribution, so read man carefully before executing.

Step One — Create the Directory Structure

The first step that we are going to take is to make a directory structure that will hold the site data that we will be serving to visitors.

Our document root (the top-level directory that Apache looks at to find content to serve) will be set to individual directories under the /var/www directory. We will create a directory here for both of the virtual hosts we plan on making.

Within each of these directories, we will create a public_html folder that will hold our actual files. This gives us some flexibility in our hosting.

For instance, for our sites, we're going to make our directories like this:

```
sudo mkdir -p /var/www/example.com/public_html
sudo mkdir -p /var/www/test.com/public_html
```

The portions in red represent the domain names that we are wanting to serve from our VPS.

Step Two — Grant Permissions

Now we have the directory structure for our files, but they are owned by our root user. If we want our regular user to be able to modify files in our web directories, we can change the ownership by doing this:

```
sudo chown -R $USER:$USER /var/www/example.com/public_html
sudo chown -R $USER:$USER /var/www/test.com/public_html
```

The `$USER` variable will take the value of the user you are currently logged in as when you press "ENTER". By doing this, our regular user now owns the public_html subdirectories where we will be storing our content.

We should also modify our permissions a little bit to ensure that read access is permitted to the general web directory and all of the files and folders it contains so that pages can be served correctly:

```
sudo chmod -R 755 /var/www
```

Your web server should now have the permissions it needs to serve content, and your user should be able to create content within the necessary folders.

Step Three — Create Demo Pages for Each Virtual Host

We have our directory structure in place. Let's create some content to serve.

We're just going for a demonstration, so our pages will be very simple. We're just going to make an index.html page for each site.

Let's start with example.com. We can open up an index.html file in our editor by typing:

```
nano /var/www/example.com/public_html/index.html
```

In this file, create a simple HTML document that indicates the site it is connected to. My file looks like this:

```
<html>
  <head>
    <title>Welcome to Example.com!</title>
  </head>
  <body>
    <h1>Success!  The example.com virtual host is working!</h1>
  </body>
</html>
```

Save and close the file when you are finished.

We can copy this file to use as the basis for our second site by typing:

```
cp /var/www/example.com/public_html/index.html /var/www/test.com/public_html/index.html
```

We can then open the file and modify the relevant pieces of information:

```
nano /var/www/test.com/public_html/index.html
```

```
<html>
  <head>
    <title>Welcome to Test.com!</title>
  </head>
  <body>
    <h1>Success!  The test.com virtual host is working!</h1>
  </body>
</html>
```

Save and close this file as well. You now have the pages necessary to test the virtual host configuration.

Step Four — Create New Virtual Host Files

Virtual host files are the files that specify the actual configuration of our virtual hosts and dictate how the Apache web server will respond to various domain requests.

Apache comes with a default virtual host file called 000-default.conf that we can use as a jumping off point. We are going to copy it over to create a virtual host file for each of our domains.

We will start with one domain, configure it, copy it for our second domain, and then make the few further adjustments needed. The default Ubuntu configuration requires that each virtual host file end in .conf.

Create the First Virtual Host File

Start by copying the file for the first domain:

```
sudo cp /etc/apache2/sites-available/000-default.conf /etc/apache2/sites-available/example.com.conf
```

Open the new file in your editor with root privileges:

```
sudo nano /etc/apache2/sites-available/example.com.conf
```

The file will look something like this (I've removed the comments here to make the file more approachable):

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

As you can see, there's not much here. We will customize the items here for our first domain and add some additional directives. This virtual host section matches any requests that are made on port 80, the default HTTP port.

First, we need to change the ServerAdmin directive to an email that the site administrator can receive emails through.

ServerAdmin admin@example.com

After this, we need to add two directives. The first, called ServerName, establishes the base domain that should match for this virtual host definition. This will most likely be your domain. The second, called ServerAlias, defines further names that should match as if they were the base name. This is useful for matching hosts you defined, like www:

```
  ServerName example.com
  ServerAlias www.example.com
```

The only other thing we need to change for a basic virtual host file is the location of the document root for this domain. We already created the directory we need, so we just need to alter the DocumentRoot directive to reflect the directory we created:

```
  DocumentRoot /var/www/example.com/public_html
```

In total, our virtualhost file should look like this:

```
<VirtualHost *:80>
    ServerAdmin admin@example.com
    ServerName example.com
    ServerAlias www.example.com
    DocumentRoot /var/www/example.com/public_html
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Save and close the file.

Copy First Virtual Host and Customize for Second Domain

Now that we have our first virtual host file established, we can create our second one by copying that file and adjusting it as needed.

Start by copying it:

```
sudo cp /etc/apache2/sites-available/example.com.conf /etc/apache2/sites-available/test.com.conf
```

Open the new file with root privileges in your editor:

sudo nano /etc/apache2/sites-available/test.com.conf
You now need to modify all of the pieces of information to reference your second domain. When you are finished, it may look something like this:

```
<VirtualHost *:80>
    ServerAdmin admin@test.com
    ServerName test.com
    ServerAlias www.test.com
    DocumentRoot /var/www/test.com/public_html
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Save and close the file when you are finished.

Step Five — Enable the New Virtual Host Files
Now that we have created our virtual host files, we must enable them. Apache includes some tools that allow us to do this.

We can use the a2ensite tool to enable each of our sites like this:

sudo a2ensite example.com.conf
sudo a2ensite test.com.conf
When you are finished, you need to restart Apache to make these changes take effect:

sudo service apache2 restart
You will most likely receive a message saying something similar to:

- Restarting web server apache2
  AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.0.1. Set the 'ServerName' directive globally to suppress this message
  This is a harmless message that does not affect our site.

Step Six — Set Up Local Hosts File (Optional)
If you haven't been using actual domain names that you own to test this procedure and have been using some example domains instead, you can at least test the functionality of this process by temporarily modifying the hosts file on your local computer.

This will intercept any requests for the domains that you configured and point them to your VPS server, just as the DNS system would do if you were using registered domains. This will only work from your computer though, and is simply useful for testing purposes.

Make sure you are operating on your local computer for these steps and not your VPS server. You will need to know the computer's administrative password or otherwise be a member of the administrative group.

If you are on a Mac or Linux computer, edit your local file with administrative privileges by typing:

sudo nano /etc/hosts

If you are on a Windows machine, you can find instructions on altering your hosts file here.

The details that you need to add are the public IP address of your VPS server followed by the domain you want to use to reach that VPS.

For the domains that I used in this guide, assuming that my VPS IP address is 111.111.111.111, I could add the following lines to the bottom of my hosts file:

127.0.0.1 localhost
127.0.1.1 guest-desktop
111.111.111.111 example.com
111.111.111.111 test.com
This will direct any requests for example.com and test.com on our computer and send them to our server at 111.111.111.111. This is what we want if we are not actually the owners of these domains in order to test our virtual hosts.

Save and close the file.

Step Seven — Test your Results
Now that you have your virtual hosts configured, you can test your setup easily by going to the domains that you configured in your web browser:

http://example.com
You should see a page that looks like this:

Apache virt host example

Likewise, if you can visit your second page:

http://test.com
You will see the file you created for your second site:

Apache virt host test

If both of these sites work well, you've successfully configured two virtual hosts on the same server.

If you adjusted your home computer's hosts file, you may want to delete the lines you added now that you verified that your configuration works. This will prevent your hosts file from being filled with entries that are not actually necessary.

If you need to access this long term, consider purchasing a domain name for each site you need and setting it up to point to your VPS server.

Conclusion
If you followed along, you should now have a single server handling two separate domain names. You can expand this process by following the steps we outlined above to make additional virtual hosts.

There is no software limit on the number of domain names Apache can handle, so feel free to make as many as your server is capable of handling.

User www-data

if we need run some command as www-data user then

```
sudo -u www-data bash-command
```

</> Improvements FMP

source https://pehapkari.cz/blog/2017/03/27/multiple-php-versions-the-easy-way/
In Debian systems , can install both php version 5.6 and 7

Installing PHP 7.0

Install PHP 7.0 from Debian archive. This will be (sadly) the default version in Stretch, 7.1 came out too late to squeeze into Stretch's timeline. Do this using the following command:

```
$ apt-get install php7.0-cli php7.0-fpm
```

Notice the different pattern of the package name. Older Debian installations used simply php5 whereas newer infrastructure uses phpX.Y. This is the obvious part that efficiently allows us to use multiple PHPs in parallel. With this structure, you can install each of the minor versions next to each other.

Installing PHP 5.6

Here's the catch. Debian only offers a single PHP version in the official repository. Fortunately there are packages directly from a maintainer of Debian's PHP packages, Ondřej Surý. Visit his page about packaging to learn more. (There is also a PPA repository in case you'd rather use Ubuntu instead of Debian.)

We'll now add his repository (as well as enable HTTPS for APT and register the APT key):

```
$ apt-get install apt-transport-https
$ curl https://packages.sury.org/php/apt.gpg | apt-key add -
$ echo 'deb https://packages.sury.org/php/ stretch main' > /etc/apt/sources.list.d/deb.sury.org.list
$ apt-get update
```

Now that we have the repository added, we can install the packages from there:

```
$ apt-get install php5.6-cli php5.6-fpm
```

This will install PHP 5.6 in parallel to PHP 7.0 installed earlier. We can check this is true by simply running:

```
$ php7.0 -v
PHP 7.0.15-1 (cli)
$ php5.6 -v
PHP 5.6.30-5+0~20170223133422.27+stretch~1.gbp1ee0cb (cli)
```

Note that for conviniency there is also a php command provided by alternatives (which defaults to the newest version):

```
$ php -v
PHP 7.0.15-1 (cli)
```

You can switch the default version using update-alternatives, just run the following command and pick the version you prefer:

```
$ update-alternatives --config php

There are 2 choices for the alternative php (providing /usr/bin/php).

  Selection    Path             Priority   Status
------------------------------------------------------------
  0            /usr/bin/php7.2   72        auto mode
* 1            /usr/bin/php5.6   56        manual mode
  2            /usr/bin/php7.2   72        manual mode

Press <enter> to keep the current choice[*], or type selection number:
```

install the needed module

> libapache2-mod-php5.6

or

> libapache2-mod-php7.2

as default

Enabling error display in php via htaccess only

```
php_flag display_startup_errors on
php_flag display_errors on
php_flag html_errors on
php_flag  log_errors on
php_value error_log  /home/path/public_html/domain/PHP_errors.log
```

Setting up multiple apache2 instances on Ubuntu 16.04

src:https://gist.github.com/aaronbloomfield/92c707631a0191152bc7faf0124fd651

PHP handling on apache is done via modules of one sort or another, and running multiple version is problematic on a single instance. The solution is to run two instances of apache on the same machine. This allows one instance to run PHP 7 (the default on 16.04), and another can run PHP 5. Since normal http/https is on ports 80 and 443, the second instance will run on ports 81 and 444. Since it is running on the same machine, all file system and database access is the exact same.

All the commands herein have to be run as root, or with `sudo` prefixed to the command.

1. Read /usr/share/doc/apache2/README.multiple-instances
2. Run `sh ./setup-instance php5` from `/usr/share/doc/apache2/examples`, where `php5` is the suffix for the second site; all commands for the second site will have that suffix. This will keep all of the same configuration for all sites on the new instance, including SSL certificates, other virtual hosts, etc. After running this command, you will see output like this:

```
Setting up /etc/apache2-php5 ...
systemd is in use, no init script installed
use the 'apache2@php5.service' service to control your new instance
sample commands:
systemctl start apache2@php5.service
systemctl enable apache2@php5.service
Setting up symlinks: a2enmod-php5 a2dismod-php5 a2ensite-php5 a2dissite-php5 a2enconf-php5 a2disconf-php5 apa
Setting up /etc/logrotate.d/apache2-php5 and /var/log/apache2-php5 ...
Setting up /etc/default/apache-htcacheclean-php5
```

3. Install PHP 5: see http://askubuntu.com/questions/756181/installing-php-5-6-on-xenial-16-04 for the source of the directions for this step
   - `add-apt-repository ppa:ondrej/php`
   - `apt-get update`
   - `apt-get install php5.6 php5.6-mbstring php5.6-mcrypt php5.6-mysql php5.6-xml`
4. Change line 1 of `/etc/apache2-php5/mods-available/php5.load` to use the correct path (likely `/usr/lib/apache2/modules/libphp5.6.so`). Alternatively, one can use the `php5.6.load` and `php5.6.conf` files installed in `/etc/apache2/mods-available` (these will have to be moved to `/etc/apache2-php5/mods-available`).
5. You may have to configure `/etc/apache2-php5/mods-available/php5.conf` (or `php5.6.conf`), for example, setting `php_admin_flag engine On` for the users' home directories.
6. On the second instance, disable anything that will conflict with the default PHP 5 module; for me this was disabling FastCGI and FPM, but yours may have php7.0; both are listed below. Also enable PHP 5. Note that PHP 5 was likely enabled on the default web server via the installation of the php5.6 package, so that needs to be disabled. Note that depending on what you did in step 4, you may have to replace "php5.6" below with "php5". Also note that some of these modules and configurations that are being disabled may not exist on your system -- that's fine.

```
a2dismod-php5 php7.0
a2dismod-php5 fastcgi
a2enmod-php5 php5.6
a2disconf-php5 php7.0-fpm
a2dismod-php5 proxy_fcgi
a2dismod php5.6
```

7. ~~Remove /etc/apache2-php5/userperms.conf, if it exists, and replace with empty file (that is a FastCGI/FPM file, which is not needed in PHP5, which is not being configured here to use FastCGI/FPM).~~ (this was only for my local configuration)
8. Make sure the porst you want to use (81/444 in this example) are available: run `nmap localhost` (you may have to install the `nmap` package first) to see which ports are currently in use. To find the processes running on a given port, run `lsof -i :81` (replace 81 with the port you are investigating).
9. In /etc/apache-php5/, edit ports.conf, and change ports 80 and 443 to the ports that you want to use (81 and 444, in this example).
10. In /etc/apache-php5/sites-available/, edit 000-default.conf, default-ssl.conf, and any other web site configuration files that you want running on the second instance. The port numbers have to be changed there as well (from 80/443 to 81/444).
11. Although the new service (`apahce2-php5`) was installed (in `/etc/init.d/`), the system isn't aware of it yet. To fix this, see the commands listed in `/usr/share/doc/apache2/README.multiple-instances` (which one you use will vary depending on your system).
12. Restart apache2: `service apache2-php5 restart`

At this point, it should work: using the regular ports for http/https (i.e., no port number) will use PHP 7. Using ports 81/444 for http/https, respectively, will use PHP 5. For example: http://example.com:81/phpinfo.php and https://example.com:444/phpinfo.php.

Setting Database Server [mariadb]

After install mariadb server and client

Secure database server

run the mysql_secure_installation script to secure the database where you can:

```
set root password (if not set in the configuration step above).
disable remote root login
remove test database
remove anonymous users and
reload privileges
```

```
$ sudo mysql_secure_installation
```

How to Create a New User in Maria

a new user within the MySQL shell:

```
CREATE USER 'newuser'@'localhost' IDENTIFIED BY 'password';
```

Sadly, at this point newuser has no permissions to do anything with the databases. In fact, if newuser even tries to login (with the password, password), they will not be able to reach the MySQL shell.

Therefore, the first thing to do is to provide the user with access to the information they will need.

```
GRANT ALL PRIVILEGES ON *.* TO 'newuser'@'localhost';
-- or for grants
grant all privileges on *.* to 'newuser'@'localhost' with grant option;
```

The asterisks in this command refer to the database and table (respectively) that they can access—this specific command

allows to the user to read, edit, execute and perform all tasks across all the databases and tables.

Once you have finalized the permissions that you want to set up for your new users, always be sure to reload all the privileges.

```
FLUSH PRIVILEGES;
```

Your changes will now be in effect.

Configure connection from LAMP to MSSQL

> firts install lamp

```
apt-get install apache2 mysql-server mysql-client php5-cli php5 php5-mysql
```

then add modrewrite module to apache

```
a2enmod rewrite
```

change in /etc/apache2/apache.conf

```
<Directory>
/var/www/
AllowOverride none
to
AllowOverride All
</Directory>
```

Restart apache Service

Allowed memory size of 147281674 bytes exhausted in .php on line 57
Then you start to search on the Internet with the above error message and you find advices like:
memory_limit = 32M to your server's main php.ini file (recommended, if you have access)
memory_limit = 32M to a php.ini file in your application's php file
ini_set('memory_limit', '32M'); ini_set('memory_limit', '-1'); in your sites/default/settings.php file
php_value memory_limit 32M in your .htaccess file
Add more memory

The easiest way to find is to access the phpinfo() function on your system by launching a web-browser and typing the name (or IP address) of your Debian server like this:

and look for the following entry:
Loaded Configuration File etc/php5/apache2/php.ini
that is basically where your global php.ini file resides on your system.
Any application that uses Apache will read the value of the parameter:
memory_limit = 128M
in cakephp overerite the .htaccess of the individual application
are in the root directory

```
    myApp/
      app/
      config/
      cake/
      .../
      .htaccess
```

Backup your databases

```
#!/bin/bash
mysqldump -u root -p --all-databases > alldb_backup.sql
# or
mysqldump -h server -u root -p --all-databases > all_dbs.sql
# and
mysql -u root -p < all_dbs.sql
# other options
mysqldump [options] --databases db_name1 [db_name2 db_name3...]

mysqldump -u SomeUser -p --databases mydb1 mydb2 mydb3 > myDbs.sql
```

```
# backup databases example
mysqldump -u root -p --databases policies portal_company portal_secure portal_user_info portal_users > integr

mysql -u ambagasdowa -p < integradev.sql
```

> TITLE => connect cakephp/LinuxBox To MSSQL

Summary

install the packages

```
freetds-bin
tdsodbc
unixodbc
php5-sybase
```

=== Add the server Definition Entry in

> /etc/freetds/freetds.conf

hostidentifier => the name of the server this is a wrapper for the connection

```
[hostidentifier]
host = 192.168.0.30
port = 1433
tds version = 8.0
```

- Add the driver configuration /etc/odbcinst.ini
- the routes in debian are /usr/lib/i386-linux-gnu/odbc/libtdsodbc.so
- and /usr/lib/i386-linux-gnu/odbc/libtdsS.so
- for x64 systems the routes are :
- Driver = /usr/lib/x86_64-linux-gnu/odbc/libtdsodbc.so
- Setup = /usr/lib/x86_64-linux-gnu/odbc/libtdsS.so

```
[ms-sql]
Description = TDS connection
Driver = /usr/lib/libtdsodbc.so.0
Setup = /usr/lib/libtdsS.so
UsageCount = 1
FileUsage = 1
```

- Add the connection configuration /etc/odbc.ini

```
[odbc-dbname]
Description = sqlserver1
Driver = ms-sql
ServerName = hostsql1
UID = TuUsuario
Port = 1433
Database = dbname
```

Driver debe de ser el nombre del driver que hemos definido en odbcinst.ini
ServerName debe de ser el identificador de host que hemos definido en freetds.conf

- Test the connection

```
# tsql -S 192.168.0.30 -p 1433 -U TuUsuario -P TuPassword
locale is “en_US.UTF-8"
locale charset is “UTF-8"
1>
```

```
# isql -v odbc-dbname TuUsuario TuPassword

| Connected!                            |
|                                       |
| sql-statement                         |
| help [tablename]                      |
| quit                                  |
|                                       |
+---------------+

SQL>
```

```
apt-get install libsybdb5 freetds-common php5-sybase
/etc/init.d/apache2 restart
```

At the end of the process, if all goes fine, you will find in the mssql section of phpinfo();

Additionally in cakephp

I corrected the problem by adding the 'port' => '' to my database.php
default config as in:

```
  var $default = array(
          'driver' => 'mssql',
          'persistent' => false,
          'host' => 'hostidentifier',
          'login' => 'sa',
          'password' => 'pass',
          'database' => 'MyDb',
          'encoding' => 'utf8',
          'port' => '1433'
  );
```

Happy coding!

Screencast

- capture image example

```
      ffmpeg -f x11grab -r 25 -s 1440x900 -i :0.0 -vcodec huffyuv screencast.avi
      # then Recommended
      ffmpeg -i screencast.avi -vf scale=320:-1 -r 10 -f image2pipe -vcodec ppm - | convert -fuzz 4% -layers
      # OR
# output as gif
      ffmpeg -i screencast.avi -loop_output 0 -r 5 -s 320x200 -pix_fmt rgb24 output.gif
      # OR
      ffmpeg -i screencast.avi -pix_fmt rgb24 output.gif
# finally
      convert output.gif -fuzz 8% -layers Optimize finalgif.gif
```

Git docs

update a repo to github

cd to path

```
git status
git add .
git commit -m "some description"
git push
```

Update a repo to local

```
git pull
```

How to ignore error on git pull about my local changes would be overwritten by merge?

try one :
Alright with the help of the other two answers I've come up with a direct solution:

```
git checkout HEAD^ file/to/overwrite
git pull
```

try two (working for nextcloud):
This works for me to override all local changes and does not require an identity:

```
git reset --hard
git pull
```

You can't merge with local modifications. Git protects you from losing potentially important changes.

You have three options.

1. Commit the change using
   git commit -m "My message"
2. Stash it.

Stashing acts as a stack, where you can push changes, and you pop them in reverse order.

To stash type:

git stash

Do the merge, and then pull the stash:

git stash pop

3. Discard the local changes

using git reset --hard. or git checkout -t -f remote/branch
3. a) Discard local changes for a specific file

using git checkout filename

Pdf section

- Merge PDF files with PHP

I've done this before. I had a pdf that I generated with fpdf, and I needed to add on a variable amount of PDFs to it.

So I already had an fpdf object and page set up (http://www.fpdf.org/) And I used fpdi to import the files (http://www.setasign.de/products/pdf-php-solutions/fpdi/) FDPI is added by extending the PDF class:

```
class PDF extends FPDI {
   $pdffile = "Filename.pdf";
   $pagecount = $pdf->setSourceFile($pdffile);
   for($i=0; $i<$pagecount; $i++){
       $pdf->AddPage();
       $tplidx = $pdf->importPage($i+1, '/MediaBox');
       $pdf->useTemplate($tplidx, 10, 10, 200);
}
```

This basically makes each pdf into an image to put into your other pdf. It worked amazingly well for what I needed it for.
Not quite sure why both the accepted answer and even the FDPI homepage seem to give botched or incomplete examples.
Here's mine which works and is easy to implement. As expected it requires fpdf and fpdi libraries:

FPDF: http://www.fpdf.org/en/download.php
FPDI: https://www.setasign.com/products/fpdi/downloads

```
require('fpdf.php');
require('fpdi.php');

$files = ['doc1.pdf', 'doc2.pdf', 'doc3.pdf'];

$pdf = new FPDI();

// iterate over array of files and merge
foreach ($files as $file) {
   $pageCount = $pdf->setSourceFile($file);
   for ($i = 0; $i < $pageCount; $i++) {
       $tpl = $pdf->importPage($i + 1, '/MediaBox');
       $pdf->addPage();
       $pdf->useTemplate($tpl);
   }
}

// output the pdf as a file (http://www.fpdf.org/en/doc/output.htm)
$pdf->Output('F','merged.pdf');
```

☁ Cloud Section

> nextcloud issues

> version 12

issues about bin row something -> edit my.cnf , add under

```
[mysql]
...
binlog_format='MIXED'
...
```

</> Code Section

Installing RichFilemanager app

clone the repo

```
git clone http://github.com/servocoder/RichFilemanager.git
```

then cp the config files from manual
You can see 2 files in the config folder. Both should be copied and renamed as the following:

```
>filemanager.config.default.json -> filemanager.config.json
```

Main configuration file. All scalar configuration options should be defined here. Reference.

```
>filemanager.init.js.example -> filemanager.init.js
```

after that run in RichFilemanager dir

```
  composer update
```

Add Google repo

- add the key

```
wget -q -O - https://dl.google.com/linux/linux_signing_key.pub | sudo apt-key add -
```

- then add the repo

```
echo "deb http://dl.google.com/linux/chrome/deb/ stable main" >> /etc/apt/sources.list.d/google.list
```

Composer installation

Introduction

Composer is a popular dependency management tool
for PHP, created mainly to facilitate installation and updates for project
dependencies. It will check which other packages a specific project depends on
and install them for you, using the appropriate versions according to the
project requirements.

This tutorial will show how to install and get started with Composer on a
Debian 8 server.

Prerequisites

For this tutorial, you will need:

- One Debian 8 server with a non-root sudo user, as shown in Initial Server Setup with Debian 8

Step 1 — Installing the Dependencies

Before we download and install Composer, we need to make sure our server has all necessary dependencies installed.

First, update the package manager cache.

```
sudo apt-get update
```

Now, let's install the dependencies. We'll need `curl` in order to download Composer and `php5-cli`, a PHP package, for installing and running it. Composer uses `git`, a version control system, to download project dependencies. You can install all three of these packages at once with this command:

```
sudo apt-get install curl php5-cli git
```

Now that the essential dependencies are installed, let's proceed and install Composer itself.

Step 2 — Downloading and Installing Composer

We will follow the instructions as written in Composer's official documentation with a small modification to install Composer globally under `/usr/local/bin`. This will allow every user on the server to use Composer.

Download the installer to the `/tmp` directory.

```
php -r "copy('https://getcomposer.org/installer', '/tmp/composer-setup.php');"
```

Visit Composer's pubkeys and signatures page and copy the SHA-384 string at the top . Then, run the following command by replacing `sha_384_string` with the string you copied.

```
php -r "if (hash_file('SHA384', '/tmp/composer-setup.php') === 'sha_384_string') { echo 'Installer verified
```

This command checks the hash of the file you downloaded with the correct hash from Composer's website. If it matches, it'll print **Installer verified**. If it doesn't match, it'll print **Installer corrupt**, in which case you should double check that you copied the SHA-384 string correctly.

Next, we will install Composer. To install it globally under `/usr/local/bin`, we'll use the `--install-dir` flag; `--filename` tells the installer the name of Composer's executable file. Here's how to do this in one command:

```
sudo php /tmp/composer-setup.php --install-dir=/usr/local/bin --filename=composer
```

You'll see a message like the following:
Output:

```
All settings correct for using Composer
Downloading...

Composer (version 1.3.2) successfully installed to: /usr/local/bin/composer
Use it: php /usr/local/bin/composer
```

You can verify that Composer is correctly installed by checking its version.

```
composer --version
```

You should see the version that was installed. At that time of this writing,
the version is:

Composer version
Composer version 1.3.2 2017-01-27 18:23:41
Finally, you can safely remove the installer script as you no longer need it.

```
rm /tmp/composer-setup.php
```

Composer is now set up and running, waiting to be used by your project. In the next section, you'll generate the `composer.json` file, which includes the PHP libraries that your project depends on.

Pear installation

- make a dir under named pear in home or {/usr/share/}pear
- download the file go-pear.php

```
$ wget http://pear.php.net/go-pear.phar
# and run
$ php go-pear.phar
```

as root for a global installation

add webdav client support

install as root this downloads the needed libs in /usr/share/php

```
pear install HTTP_WebDAV_Client-1.0.2
```

Android

udev

Set up a device for development

Before you can start debugging on your device, there are a few things you must do:

```
On the device, open the Settings app, select Developer options, and then enable USB debugging.
Note: If you do not see Developer options, follow the instructions to enable developer options.
Set up your system to detect your device.
    Windows: Install a USB driver for Android Debug Bridge (adb). For an installation guide and links to OEM
    Mac OS X: It just works. Skip this step.
    Ubuntu Linux: Use apt-get install to install the android-tools-adb package. This gives you a community-ma

    Make sure that you are in the plugdev group. If you see the following error message, adb did not find you

    error: insufficient permissions for device: udev requires plugdev group membership

    Use id to see what groups you are in. Use sudo usermod -aG plugdev $LOGNAME to add yourself to the plugde

    The following example shows how to install the Android adb tools package.

    apt-get install android-tools-adb
```

Try this:

```
Create udev rules

sudo gedit /etc/udev/rules.d/51-android.rules

Insert this lines, save and exit

SUBSYSTEM=="usb", ATTR{idVendor}=="1bbb", MODE="0666", GROUP="plugdev"

Restart udev

sudo service udev restart

Enable USB Debug in your phone (In Android 4.2.x and up Developer Options is hidden, to make it visible, do t
    Tap seven times in Build Number: Settings > About Phone > Build Number
    You will get a message saying you have enabled Developer Options or something like that, go back to Setti

Connect phone with USB cable

Open Terminal and type

adb devices
```

I have Alcatel OneTouch Pop C7 (7041D)

List of devices attached
6H9PY5ZPJV9HO7R4 device

OS-Admin section

disable web autoplay in firefox
set to true this options in about:config

```
  media.autoplay.enabled
  media.block-autoplay-until-in-foreground
```

>_ Notes Section

monitor off

El objetivo en cuestión para configurar ese tema es logind.conf y está en

```
/etc/systemd
```

No se si es igual en todas las distros o cambia entre unas y otras.

```
sudo nano /etc/systemd/logind.conf
```

En este archivo aparecen unas opciones mas que interesantes, no estoy seguro al 100% pero creo que se puede aplicar la configuración adecuada para cualquier caso editando este archivo, a mi me aparece lo siguiente:

```
[Login]
#NAutoVTs=6
#ReserveVT=6
#KillUserProcesses=no
#KillOnlyUsers=
#KillExcludeUsers=root
#InhibitDelayMaxSec=5
#HandlePowerKey=poweroff
#HandleSuspendKey=suspend
#HandleHibernateKey=hibernate
#HandleLidSwitch=suspend
#HandleLidSwitchDocked=ignore
#PowerKeyIgnoreInhibited=no
#SuspendKeyIgnoreInhibited=no
#HibernateKeyIgnoreInhibited=no
#LidSwitchIgnoreInhibited=yes
#HoldoffTimeoutSec=30s
#IdleAction=ignore
#IdleActionSec=30min
#RuntimeDirectorySize=10%
#RemoveIPC=yes
```

Descomentando las linea que queramos usar y cambiándole el atributo adecuado que nos interese se consigue la configuración deseada.

En mi caso y a modo de ejemplo, para deshabilitar el interruptor usado en el cierre de la pantalla de un portátil que provoca la suspensión del sistema y o la hibernación del servidor, incluso provocaba el cierre de la sesión ssh de acceso remoto.

```
HandleSuspendKey=ignore
HandleHibernateKey=ignore
HandleLidSwitch=ignore
```

Guardamos los cambios, reiniciamos el sistema operativo y comprobamos el resultado.

Using Expect and Bash

Example 1

```
/usr/bin/expect<<EOF
spawn sudo xcodebuild -license
expect {
    "*License.rtf" {
        send "\r";
    }
    timeout {
        send_user "\nExpect failed first expect\n";
        exit 1;
    }
}
expect {
    "*By typing 'agree' you are agreeing" {
        send "agree\r";
        send_error "\nUser agreed to EULA\n";
     }
     "*Press 'space' for more, or 'q' to quit*" {
         send "q";
         exp_continue;
     }
     timeout {
         send_error "\nExpect failed second expect\n";
         exit 1;
     }
}
EOF
```

Example 2

If you want to automate only a part of your script you can call expect inside test.sh like so :

```
#!/bin/sh

echo -n Enter User Id:
read userid
echo -n "Enter Password for remote user:"
read -s password
hostname=`hostname`

expect -c '
enter code here
spawn ssh ${userid}@'"$hostname"'
expect "Password: "
send '"$password\r"'
expect "$ "
send "pbrun su - tibco\r"
expect "$ "
send "exit\r"
expect "$ "
send "pbrun bash\r"
expect "$ "
send "ps -ef |grep apache\r"
expect "$ "
send "exit\r"
 '
```

To access variables set from outside the expect -c you have to use single and double quotes like so: '"$var"'

set output modes

xrandr -> view avaliable modes

```
Screen 0: minimum 320 x 200, current 1366 x 1024, maximum 8192 x 8192
eDP-1 connected 1366x768+0+0 (normal left inverted right x axis y axis) 309mm x 173mm
   1366x768      60.06*+
   1360x768      59.80    59.96
   1024x768      60.04    60.00
VGA-1 connected primary 1280x1024+0+0 (normal left inverted right x axis y axis) 419mm x 262mm
  1440x900      59.89 +  74.98
  1280x1024     75.02*   70.00    60.02
  1280x800      59.81
  1152x864      75.00
  1024x768      75.03    70.07    60.00
```

then:
xrandr --output VGA-1 --left-of eDP-1
xrandr --output VGA-1 --mode 1440x900

extract Extension and basename

extract the Extension

```
${basename##*.}
```

and for extract the basename

```
${basename%.*}
```

How to decode AAC (.m4a) audio files into WAV?

codecs conversion formats

```
for f in *.m4a; do avconv -i "$f" "${f/%m4a/wav}"; done
```

search code with find

Find a pattern inside of several files and extract that file to another dir with his dir structure

```
find ./ -type f -print0 | xargs -0 grep --color RegistroPatronal=\"E6480571109\" | grep -v src | awk -F: '{pr
```

firts find files with pattern
second clean output and print firts column that have the path of the file
third select only needed files in this case filter by year
last copy the file to another path
done

Find inside in a pdf

```
find /path -name '*.pdf' -exec sh -c 'pdftotext "{}" - | grep --with-filename --label="{}" --color "your patt
```

and copy to path

```
find ./ -name '*.pdf' -exec sh -c 'pdftotext "{}" - | grep --with-filename --label="{}" --color "yout pattern
```

The "-" is necessary to have pdftotext output to stdout, not to files. The --with-filename and --label= options will put the file name in the output
of grep. The optional --color flag is nice and tells grep to output using colors on the terminal.
(In Ubuntu, pdftotext is provided by the package xpdf-utils or poppler-utils.)
This method, using pdftotext and grep, has an advantage over pdfgrep if you want to use features of GNU grep that pdfgrep doesn't support.
Note: pdfgrep-1.3.x supports -C option for printing line of context.

Get the major size files on top

```
du -h --summarize --total * | sort -rh
```

ls hdd devices by uuid and mount it

make a dir source ex: */media/externalx*

```
ambagasdowa@uruk:~/Github$ sudo lsblk --output NAME,FSTYPE,LABEL,UUID,MODE
NAME   FSTYPE LABEL       UUID                                 MODE
sda                                                         brw-rw----
|-sda1 swap               2d2f3618-a26b-4719-8e70-8d4ef3a88d0f brw-rw----
|-sda2 ext4               127a8f38-e25e-4dae-ba89-b195af71df70 brw-rw----
|-sda3                                                      brw-rw----
`-sda5 ext4               d5abb0c9-3205-4bf4-bf6e-629e95c3af1f brw-rw----
sdb                                                         brw-rw----
|-sdb1 swap               fc532612-ead2-4b82-8ac3-72bc1b9ca39f brw-rw----
|-sdb2                                                      brw-rw----
|-sdb5 ext4               837dd734-eeba-48a9-a001-e6711c2032a1 brw-rw----
`-sdb6 ext4               e3ca1ebb-56e5-486c-970d-38be0ef5034d brw-rw----
sdc                                                         brw-rw----
`-sdc1 vfat   MYLINUXLIVE D85A-828F                         brw-rw----
sr0                                                         brw-rw----
```

in fstab add

```
#mount external hdd
UUID=837dd734-eeba-48a9-a001-e6711c2032a1 /media/externalx auto defaults,noauto 0 1
UUID=e3ca1ebb-56e5-486c-970d-38be0ef5034d /media/externaly auto defaults,noauto 0 1
```

locale

perl: warning: Setting locale failed.

rewrite with

> sudo locale-gen en_US en_US.UTF-8

or reconfigure the package

> sudo dpkg-reconfigure locales

some kind of fix in enviorement

```
sudo -i
locale
export LANGUAGE=en_US.UTF-8; export LANG=en_US.UTF-8; export LC_ALL=en_US.UTF-8; locale-gen en_US.UTF-8
dpkg-reconfigure locales
reboot
```

SQL

- Sql notes

MariaDB Connect Engine

Some Connect Engine Notes

Example: Connect MariaDB on Linux to Microsoft SQL Server

This example uses the tabname option to work around a difference between MariaDB and SQL Server.
We want to retrieve some AdventureWorks data stored in the Person.Address table. However,
MariaDB does not have the idea of a table schema, and so we will change the name of the table to "PersonAddress" in MariaDB.
We specify the actual table name with the tabname, so the SQL Server ODBC driver can pass on the table name that SQL Server recognises.

```
$ /opt/mariadb/bin/mysql --socket=/opt/mariadb-data/mariadb.sock
MariaDB [(none)]> CREATE DATABASE MSSQL;
MariaDB [(none)]> USE MSSQL;
MariaDB [MSSQL]> INSTALL SONAME 'ha_connect';
MariaDB [MSSQL]> CREATE TABLE PersonAddress engine=connect
                                            table_type=ODBC
                                            tabname='Person.Address'
                                            Connection='DSN=SQLSERVER_ADVENTUREWORKS;';
ERROR 1105 (HY000): Unsupported SQL type -11
MariaDB [MSSQL]> \! grep -- -11 /usr/local/easysoft/unixODBC/include/sqlext.h
#define SQL_GUID                        (-11)
MariaDB [MSSQL]> CREATE TABLE PersonAddress (  AddressID int,
                                               AddressLine1 varchar(60),
                                               AddressLine2 varchar(60),
                                               City varchar(30),
                                               StateProvinceID int,
                                               PostalCode varchar(15),
                                               rowguid varchar(64),
                                               ModifiedDate datetime )
                                  engine=connect
                                  table_type=ODBC
                                  tabname='Person.Address'
                                  Connection='DSN=SQLSERVER_SAMPLE;';
MariaDB [MSSQL]> SELECT City FROM PersonAddress WHERE AddressID = 32521;
+-----------+
| City      |
+-----------+
| Sammamish |
+-----------+
```

Because there is no direct equivalent for the SQL Server data type uniqueidentifier.
We have to map this type in the rowguid column to a MariaDB VARCHAR type.
Even though this is the only problematic column, we need to include the others in the CREATE TABLE statement.
Otherwise, the table would only contain the rowguid column.

The Seven Sins against TSQL Performance

There are seven common antipatterns in TSQL coding that make code perform badly, and three good habits which will generally ensure that your code runs fast. If you learn nothing else from this list of great advice from Grant, just keep in mind that you should 'write for the optimizer'.

Using the wrong data types
Using Functions in Comparisons within the ON or WHERE Clause
Employing Multi-Statement User Defined Functions (UDFs)
The “Run Faster” Switch: Allowing “Dirty Reads’
Applying Query Hints indiscriminately
Allowing “Row By Agonizing Row” processing (cursos etc ...)
Indulging in Nested Views
It’s not enough that your code is readable: it must perform well too.

Links

https://josephmaryam.files.wordpress.com

http://iteadjmj.com/inicio.html