magnus Documentation

Release 0.2.0

Kevin Leptons

Jul 07, 2018

Contents

Software development tools.

Kevin Leptons <kevin.leptons@gmail.com> April, 2017

CHAPTER 1

INTRODUCTION

MAGNUS is an software development environment. It's target is Debian platform, Command Line Inteface and programming languages below:

- C/C++
- Bash
- Python

Why only Debian is supported?. Because resource is limited, focus one thing give higher quality. Debian is good environment for server but not client. In future, if GNU community make a good Desktop Environment and Debian become most common operating system then problem will solve. We write and run code in Debian, not other operating system. That's hurt, but sorry for other brothers in GNU family, you aren't supported.

Why only Command Line Interface is supported?. Because simplicity. With Command Line Interface, if you want to do something, just type <command>. With Graphic Interface, if you want to do something, you must open <window>, open <child-window>, open <granchind-window>, select <tab>, select <checkbox>, fill <textbox>, press <button> and etc. Command Line Interface save your life, give you time to do other work.

Why few of programing languages are supported?. Because Software Development Kit correspond with each language is large, include too many languages make package become huge. Other way, above languages is enough to do almost anythings. C/C++ for low-letency programming level. Bash for shell's operating system programming level. Python for fast-implement programming level. If you use other languages, MAGNUS isn't for you.

However, sometime you need to work with other programming languages, ATOM is suggestion. It is MIT License, graphic interface and support many many programming languages. ATOM here https://atom.io/.

With MAGNUS, you must spend long time to learn, but in long term, you will get benefit. If you haven't few of weeks or months to learn, dont't use MAGNUS, it will make you become crazy.

This document is mixed, include two parts. First is usage document for who wants to use MAGNUS. Second is development document for who wants to develop MAGNUS.

CHAPTER 2

USAGE

INSTALL

SYSTEM REQUIREMENTS

Table 2.1: System requirements for magnus

Property	Value
Debian GNU/Linux	>= v8.0, <= v9.0
Free storage	>= 100MB

INSTALL FROM PACKAGE

Download magnus package. Goto github and find download link in USAGE section: https://github.com/kevin-leptons/magnus

Listing 2.1: Install magnus

```
# example for version 0.1.0
# if you use other version, package name is different
dpkg -i magnus_0.1.0-0_all.deb
apt-get install -f
```

UNINSTALL

Listing 2.2: Uninstall magnus

```
apt-get purge magnus
apt-get autoclean
apt-get autoremove
```

VIM

SAVING [HYBIRD]

If you close buffer or repace current buffer by new file, MAGNUS save all of current buffer for you.

If you want to save buffers actively, you can use shortcut keys below:

Table 2.2: Shorcut keys

Keys	Mode	Notes
<F9>	NORMAL	Save and close current buffer
<F10>	NORMAL	Save all of buffers
<F12>	NORMAL	Save all of buffers then exit

AUTO PAIRS [PASSIVE]

Close brackets if it is opened.

Table 2.3: Auto-pairs examples

Input	Output
{	{ }
[	[]
“”	“”

COMMAND-T [ACTIVE]

Quick open files.

Table 2.4: Shorcut keys

Keys	Mode	Notes
t	NORMAL	Open search window
<C>-c	INMODE	Exit open file mode
<UP>, <DOWN>	INMODE	Move cursor between files
<ENTER>	INMODE	Open file in current cursor

It open sub window, then user should type file name into input. If file name matched, it list file names. Enter to open file to current window.

NERDCOMMENTER - [ACTIVE]

Comment/uncomment selected block.

Table 2.5: Shortcut keys

Key	Mode	Notes
cc	NORMAL	Comment selected block
cu	NORMAL	Uncomment selected block

YOUCOMPLETEME - [HYBIRD]

Auto commplete.

Table 2.6: Active conditions

Condition	Operation
At least 2 characters is typed	Open sub window and show suggestions
<DOT>, <COMMA> is typed	Open sub window and show suggestions

Table 2.7: Shorcut keys

Keys	Mode	Notes
<TAB>	INMODE	Select next suggestion
<S-TAB>	INMODE	Select prev suggestion
<UP>	INMODE	Select above suggestion
<DOWN>	INMODE	Select below suggestion
<ENTER>	INMODE	Select suggestion in cursor

SEARCH - [ACTIVE]

Table 2.8: Shorcut keys

Keys	Mode	Notes
<BS>	NORMAL	Start searching
<TAB>	INMODE	Select next result
<ENTER>	INMODE	Start selection mode: SELMODE
<TAB>	SELMODE	Move to position by characters

HIGHLIGHTING - [PASSIVE]

Some programming languages such as C, C++, Python... and data format such as RestructureText, HTML... are enabled to highlight now, but it is very very simple and keep for everything are distraction. MAGNUS will highlight keywords, macros, identities and strings, other things aren't highlight.

SESSION - [PASSIVE]

On opening, if file <./tmp/session.vim> is exist, MAGNUS loads it as session restoring.

On exiting, if file <./tmp/session.vim> is exist, MAGNUS create session file and save to <./tmp/session.vim>. So you MUST create session file by command to allow MAGNUS auto save/load session. You can done it by shortcut key:

Ctrl + F12

In somecases, session restoring occurs some errors because functions, options, etc... no longer exist. For example: upgrade to new MAGNUS version. Then you MUST remove file <./tmp/session.vim> by toggle shorcut key:

Ctrl + F12

If <./tmp/session.vim> is early exist then MAGNUS will removes it else MAGNUS will create it. Then exit and re-open vim to allow everything work correctly.

Table 2.9: Shorcut keys

Keys	Mode	Notes
<C-F12>	NORMAL	Toggle create/remove session file

MAKE [ACTIVE]

Build and goto error wiht C/C++

Table 2.10: Shortcut keys

Keys	Mode	Notes
<F5>	Normal	Run "./ctl build", open error window
<F7>	Normal	Goto prev error
<F8>	Normal	Goto next error

OTHER SHORCUT KEYS - [ACTIVE]

Table 2.11: Shorcut keys

Keys	Mode	Notes
<TAB>	NORMAL	Open new tab
<TAB>	NORMAL	Select next tab
<C-TAB>	NORMAL	Select prev tab
<F2>	NORMAL	Start invoke bash commands
<F4>	NORMAL	Show ouput of prev bash commands

REFERENCES

auto-pairs https://github.com/jiangmiao/auto-pairs

command-t https://github.com/wincent/command-t

incsearch https://github.com/haya14busa/incsearch.vim

incsearch-easymotion https://github.com/haya14busa/incsearch-easymotion.vim

incsearch-fuzzy https://github.com/haya14busa/incsearch-fuzzy.vim

nerdcommenter https://github.com/scrooloose/nerdcommenter

vim-easymotion https://github.com/easymotion/vim-easymotion

you-complete-me https://github.com/Valloric/YouCompleteMe

PROGRAMMING

C/C++

See references below:

C language - Wikipedia https://en.wikipedia.org/wiki/C_(programming_language)

C++ - Wikipedia https://en.wikipedia.org/wiki/C%2B%2B

GCC - Wikipedia https://en.wikipedia.org/wiki/GNU_Compiler_Collection

GDB - Wikipedia https://en.wikipedia.org/wiki/GNU_Debugger

CMake - Wikipedia https://en.wikipedia.org/wiki/CMake

Bash

See references below:

Bash - Wikipedia https://github.com/kevin-leptons/magnus

Python

See references below:

Python - Wikipedia https://github.com/kevin-leptons/magnus

CHAPTER 3

DEVELOPMENT

INTRODUCTION

This is development document for who wants to contribute MAGNUS. If you are first time meet this document, read follow each section, it help you step by step.

INSTALL

SYSTEM REQUIREMENTS

Make sure that your system meet requirements:

Table 3.1: System requirements

Property	Value
Debian GNU/Linux	>= 8.0, <= 9.0
Bash	>= 4.3, <= 5.0
Git	>= 2.0, <= 3.0
Free storage	>= 1.5GB

Listing 3.1: Verify system environment

```
# install required packages
apt-get install git

# verify pakcage versions
bash --version
git --version

# verify free storage
df --block-size=GB
```

CLONE SOURCE CODE

Listing 3.2: Get source code

```
git clone https://github.com/kevin-leptons/magnus
cd magnus
```

INSTALL VIRTUAL ENVIRONMENT

Listing 3.3: Install Python virtual environment

```
# create virtual environment
./env init

# enter virtual environment
. venv/bin/active

# exit virtual environment
deactive
```

Before development tasks, you must ensure that you entered Python virtual environment by command:

Listing 3.4: Enter Python virtual environment

```
. venv/bin/active
```

If MAGNUS development change Python version using, virtual environment must clean and install again.

Listing 3.5: Reinstall virtual environment

```
# if you are in virtual environment, exit from it
deactive

# clean current virtual environment
./env clean

# install virtual environment
./env init
```

INSTALL DEPEDENCY PACKAGES

Listing 3.6: Install dependency packages

```
./env install
```

During development, other people can change dependency packages in **dev-requires.txt** :

- Add packages
- Remove packages
- Modify versions

So you must call `./env install` to reinstall dependency packages. If not, you will see errors, warning about missing or not compative packages.

If virtual environment is installed, you also must call `./env install` to reinstall dependency packages, because virtual environment and dependency packages before was removed.

REFERENCES

Debian https://www.debian.org/

Bash https://www.gnu.org/software/bash/

Advanced Package Tool https://wiki.debian.org/Apt

Git https://git-scm.com/

Python Virtual Environment http://python-guide-pt-br.readthedocs.io/en/latest/dev/virtualenvs/

FILE SYSTEM

DIRECTORY STRUCTURE

Table 3.2: File description

File	Type	Summary
readme.md	S	Quick manual
license.md	S	Content of license
changelog.md	S	Features history
env	S	Control Python virtual environment
ctl	S	Control development tasks
dev-requires.txt	S	Specify Python packages use for development
.travis.yml	S	Specify Travis operation
.gitignore	S	Specify git ignore files
_config.yml	S	Sepcify github page configurations
asset/	S	Static resources such as image, videos, audio
doc/	S	Source file of documentation
dsrc/	S	Source files of dependency packages
src/	S	Source files of program
test/	S	Test functions
tool/	S	Building scripts
.git/	S	Metadata of Git
dest/	T	Temporary build files of source
dist/	T	Temporary build files of packaging
tsrc/	T	Temporary source files of dependency packages
venv/	T	Python virtual environment files
tmp/	T	General temporary files

Table 3.3: File type description

File type	Description
S	Source files, track by Git
T	Template files

FILE NAMING

File name include regular files, links, directories must name follow regex expression:

> [a-z0-9-.]{1,16}

Table 3.4: File naming examples

Name	Valid
readme.md	Yes
README.md	No
ReadMe.md	No
read_me.md	No

CHARACTER ENCODING

Only UTF-8.

REFERENCES

Regex Expression https://en.wikipedia.org/wiki/Regular_expression

Character encoding https://en.wikipedia.org/wiki/Character_encoding

BUILDING

FILES SYSTEM

Table 3.5: Source files description

File	Description
src/debian/	Specification Debian package
src/dpkg/	Depedency package was built
src/man/	Manual for magnus command
src/vim/	Vim scripts
src/cli.py	Magnus command
dsrc/	Source files of dependency packages
tsrc/	Temporary build of dependency packages
dest/man/	Temporary build files of man pages
dest/vim/	Temporary build files of vim scripts
dest/doc/	Temporary build files of documentation

BUILD DEPENDENCY PACKAGES

Building of dependency packages is rare perform. It only occur when add new dependency packages. Build result put into **src/dpkg/** directory and use to build source files.

YouCompleteMe is special dependency package, it require more thang 2GB RAM and about 1GB storage to build and building is slow.

Listing 3.7: Build dependency packages

```
# remove temporary files if needed
# remove dest files if needed
./ctl dbuild --clean-tmp
./ctl dbuild --clean-dest

# build new files
./ctl dbuild

# or show list of dependency packages
# build specific package, for example: ycm
./ctl dbuild --list
./ctl build ycm
```

BUILD SOURCE FILES

Listing 3.8: Build source files

```
./ctl build
```

Listing 3.9: Clean all of build files

```
./ctl build --clean
```

BUILD DOCUMENTS

Listing 3.10: Build documents

```
./ctl doc
```

Listing 3.11: Clean document's build files

```
./ctl doc --clean
```

TESTING

FILE SYSTEM

Table 3.6: Testing file system

File	Description
test/coding.py	Test coding style
test/build.py	Test building, packaging

TEST ALL OF UNITS

Listing 3.12: Test all of units

```
./ctl test
```

TEST SPECIFIC FILE

Listing 3.13: Test specific file

```
pytest test/<unit-test>.py
```

DISTRIBUTION

FILES SYSTEM

Table 3.7: Distribution files description

File	Description
dist/magnus_*/	Temporary distribute file system
dist/magnus_*.deb	Package file to distribution

UPDATE INFORMATION

Provide correct information of:

- **pkg_ver** variable in **ctl** file.
- Addition, deletion, modification of features in **changelog.md** file.

PACK

Listing 3.14: Pack files

```
./ctl dist
```