```
# Makefile front-end for rust dev works.
VERSION         = 1.0
RELEASE         = 2
DATE            = $(shell date)
NEWRELEASE      = $(shell echo $$(($(RELEASE) + 1)))
PROJECT_NAME    = hellocrud
TOPDIR = $(shell pwd)
MANPAGES =
MACPATH=/usr/local/bin
A2PS2S1C  = ${MACPATH}/enscript -H1 --highlight-bar-gray=08 -fCourier8 -Ebash  --non-printable-format=space
A2PSTMP   = ./tmp
DOCS      = ./docs

SHELL := /bin/bash

.PHONY: all commit tests examples pdf doc docs sqld cmkrest cmd-demo cmk-agent-ctl cmkc

all: help
#https://stackoverflow.com/questions/6273608/how-to-pass-argument-to-makefile-from-command-line
args = `arg="$(filter-out $@,$(MAKECMDGOALS))" && echo $${arg:-${1}}`
%:
	@:

versionfile:
	echo "version:" $(VERSION) > etc/version
	echo "release:" $(RELEASE) >> etc/version
	echo "source build date:" $(DATE) >> etc/version

manpage:
	for manpage in $(MANPAGES); do (pod2man --center=$$manpage --release="" ./docs/$$manpage.pod > ./docs/$
$manpage.1); done

clean: cleantmp
	-rm -rf *~
	-rm -rf docs/*.1
	-find . -type f -name *~   -exec rm -f {} \;
	-find . -type f -name *.ps  -exec rm -f {} \;
	-find . -type d -name target  -exec rm -rf {} \;

clean_hard:
	-rm -rf $(shell $(PYTHON) -c "from distutils.sysconfig import get_python_lib; print get_python_lib()")/
adagios

clean_hardest: clean_rpms


#Ref: https://stackoverflow.com/questions/1490949/how-to-write-loop-in-a-makefile
# Adding *.rs need to be conveted into pdf
SRC1= README.md Makefile Cargo.toml hellocrud-dir-layout.txt
SRC2= src/classes.rs src/functions.rs src/hello.rs src/main.rs src/test.rs
SRC3=

pdfdir:
	mkdir -p ${A2PSTMP}/src $(DOCS)/src
cleantmp:
	rm -f ${A2PSTMP}/*.ps ${A2PSTMP}/*.pdf
.ps: cleantmp
	$(foreach var, $(SRC1), ${A2PS2S1C}  --output=${A2PSTMP}/$(var).ps $(var) ;)
	$(foreach var, $(SRC2), ${A2PS2S1C}  --output=${A2PSTMP}/$(var).ps $(var) ;)
	$(foreach var, $(SRC3), ${A2PS2S1C}  --output=${A2PSTMP}/$(var).ps $(var) ;)
	touch .ps
# No VM needed since MacOS is Unix based.
macpdf: .pdf
	touch .pdf
	rm -f ${DOCS)}/*.pdf
	find  ${A2PSTMP}/ -type f -name *.pdf -exec cp {}  ${DOCS}/  \;
	ls -lrt  ${DOCS}/*.pdf

allpdf: .pdf
	touch .pdf
	rm -f ${A2PSTMP}/*.pdf
	find  docs  -type f -name *.pdf -exec cp {}  ${A2PSTMP}/  \;
	rm -f /mnt/hgfs/vmware/*.pdf
	cp -f ${A2PSTMP}/*.pdf  /mnt/hgfs/vmware/
	ls -lrt  /mnt/hgfs/vmware/*.pdf


pdf: .pdf
	touch .pdf
.pdf: .ps
	$(foreach var, $(SRC1), (cd ${A2PSTMP};ps2pdf $(var).ps $(var).pdf);)
	$(foreach var, $(SRC2), (cd ${A2PSTMP};ps2pdf $(var).ps $(var).pdf);)
	$(foreach var, $(SRC3), (cd ${A2PSTMP};ps2pdf $(var).ps $(var).pdf);)
	rsync -azpv ${A2PSTMP}/*    ${DOCS}/
	find ${DOCS}/src/ -type f -name "*.ps" -exec rm -f {} \;
	touch .pdf
```

```
tree: clean
	tree -L 4 > ${PROJECT_NAME}-dir-layout.txt

.PHONY: tests test2
tests:
	(cd hello && cargo test)

# enable makefile to accept argument after command
#https://stackoverflow.com/questions/6273608/how-to-pass-argument-to-makefile-from-command-line

args = `arg="$(filter-out $@,$(MAKECMDGOALS))" && echo $${arg:-${1}}`
%:
	@:

status:
	git status
commit:
	git commit -am "$(call args, Automated commit message without details, Please read the git diff)"  && g
it push
pull:
	git pull
install:
	cargo install  --force --path .
	ls -lrt ${HOME}/.cargo/bin

cmk-agent-ctl:
	(cd cmk-agent-ctl && cargo install  --force --path . )
cmd-demo:
	(cd cmd-demo && ./cmd-install.sh)

cmkc:
	(cd cmkc && cargo install  --force --path . )

cmkrest:
	(cd cmkrest &&  cargo install  --force --path .)

install2:
#	cd sqld/sqlc    && cargo install  --force --path .
#	cd sqld/sqlc    && cargo install  --force --path .
	ls -lrt ${HOME}/.cargo/bin

installcmk:
	cd cmk    && cargo install  --force --path .
	ls -lrt ${HOME}/.cargo/bin | tail -10

installsqld:
	cd sqld   && make install
	ls -lrt ${HOME}/.cargo/bin | tail -10


clip:
	cargo clippy
build:
	cargo build &&  find target/debug  -maxdepth 1 -type f -perm /755 | egrep -v "\.d"
sqld:
	cd sqld && cargo build && find target/debug  -maxdepth 1 -type f -perm /755 | egrep -v "\.d"
	cd sqld && cp -p target/debug/sqld target/debug/bottomless-cli ${HOME}/.cargo/bin
	ls -lrt ${HOME}/.cargo/bin
format:
	cargo fmt -v

examples:
	cargo build --examples && ls -lrt target/debug/examples/ |egrep -v "\.d"
doc:
	rustdoc README.md --crate-name docs
	cargo doc
	ls target/doc doc
#	rustdoc src/bin/cmdbc.rs  --crate-name docs

saas_pg:
	LOCO_APP_NAME=saas_pg LOCO_TEMPLATE=saas loco new -p ./

lightweight_service:
	LOCO_APP_NAME=lightweight_service LOCO_TEMPLATE=lightweight-service loco new


help:
	@echo "Usage: make <target> <argument>"
	@echo
	@echo "Available targets are:"
	@echo "  pdf                 Generate selected files in pdf and copy into ./docs"
	@echo "  allpdf              Generate selected files in pdf and copy into vmware hgfs"
	@echo "  test                   build and test run"
	@echo "  format                 run cargo fmt to format the rust code"
	@echo "  build                  call up cargo build"
	@echo "  examples               build all test programs in examples dir"
	@echo "  install                install the binary in --path ."
	@echo "  help                   Showing this help "
	@echo "  clean                  clean all artifact files"
	@echo "  commit {"my message"}  ie, git commit and push with defalut message"
	@echo "  status                 ie, git status"
	@echo "  pull                   ie, git pull"
	@echo ""
```