```
# Makefile front-end for rust dev works.
VERSION         = 1.0
RELEASE         = 2
DATE            = $(shell date)
NEWRELEASE      = $(shell echo $$(($(RELEASE) + 1)))
PROJECT_NAME    = zabbix-api-rs
TOPDIR = $(shell pwd)
MANPAGES =
A2PS2S1C  = /usr/bin/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
#  mkdir -p tmp/src/client/v6 docs/src/client/v6 tmp/src/host
SRC1= README.md Makefile Cargo.toml zabbix-api-rs-dir-layout.txt docker-compose.yml DEV.md .env_db_mysql .env
_web
SRC2= src/error.rs src/lib.rs src/client/jsonrpc.rs src/client/mod.rs src/client/post.rs src/client/v6/mod.rs
 src/host/create.rs src/host/host/get.rs src/host/mod.rs
SRC3=
#SRC2= manage.py profiles_projects-dir-layout.txt

tmpdir:
        mkdir -p ${A2PSTMP}/src/bin ${A2PSTMP}/src/controllers ${A2PSTMP}/src/views ${A2PSTMP}/tests/requests
/snapshots
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

allpdf: .pdf
        touch .pdf
        # rm -f ${A2PSTMP}/*.pdf
        find ${A2PSTMP}  -type f -name *.pdf -exec cp {}  ${DOCS}/  \;
        rm -f /mnt/hgfs/vmware/*.pdf
        cp -f ${DOCS}/*.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);)
	rm -f ${A2PSTMP}/*.ps
	rsync -azpv ${A2PSTMP}/* ${DOCS}/
	find ${DOCS}/ -type f -name "*.ps" -exec rm -f {} \;
	touch .pdf
tree: clean
	tree -L 5 > ${PROJECT_NAME}-dir-layout.txt

.PHONY: tests test2
testslist:
	cargo test -- --list
tests:
	cargo test  -- client::v6::tests::create_host_group_and_host
	cargo test  -- client::v6::tests::create_item
	cargo test  -- client::v6::tests::create_trigger
	cargo test  -- client::v6::tests::get_api_info
	cargo test  -- client::v6::tests::create_web_scenario
	cargo test  -- client::v6::tests::get_hosts_test
	cargo test  -- client::v6::tests::get_items_test
	cargo test  -- client::v6::tests::get_host_groups_test
	cargo test  -- client::v6::tests::get_triggers_test
	cargo test  -- client::v6::tests::raw_api_call_test
	cargo test  -- client::v6::tests::get_webscenarios_test
	cargo test  -- client::v6::tests::session_should_be_returned


# 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 && git branch
commit:
	git commit -am "$(call args, Automated commit message without details, Please read the git diff)"  &&
 git 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:
	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"
	ls -lrt ${HOME}/.cargo/bin | tail -3
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
hello:
        rm -f ${HOME}/.cargo/bin/hello
        (cd docs/examples/hello && cargo install --path . -f )
        ls -lrt ${HOME}/.cargo/bin | tail -3
dockerup:
        docker-compose up -d
        docker ps --format 'table {{.Names}}\t{{.Image}}\t{{.Status}}' | egrep 'zabbix|NAMES'
dockerdown:
         docker-compose down
docker:
        docker ps --format 'table {{.Names}}\t{{.Image}}\t{{.Status}}' | egrep 'zabbix|NAMES'

itest:
        ncat -zv localhost 3080
        ZABBIX_API_URL=http://localhost:3080/api_jsonrpc.php ZABBIX_API_USER=Admin ZABBIX_API_PASSWORD=zabbix
 cargo test
help:
        @echo "Usage: make <target> <argument>"
        @echo
        @echo "Available targets are:"
        @echo "  hello                  helloworld example"
        @echo "  pdf                    Generate selected files in pdf and copy into ./docs"
        @echo "  allpdf                 Generate selected files in pdf and copy into vmware hgfs"
        @echo "  tests                  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 ""
```