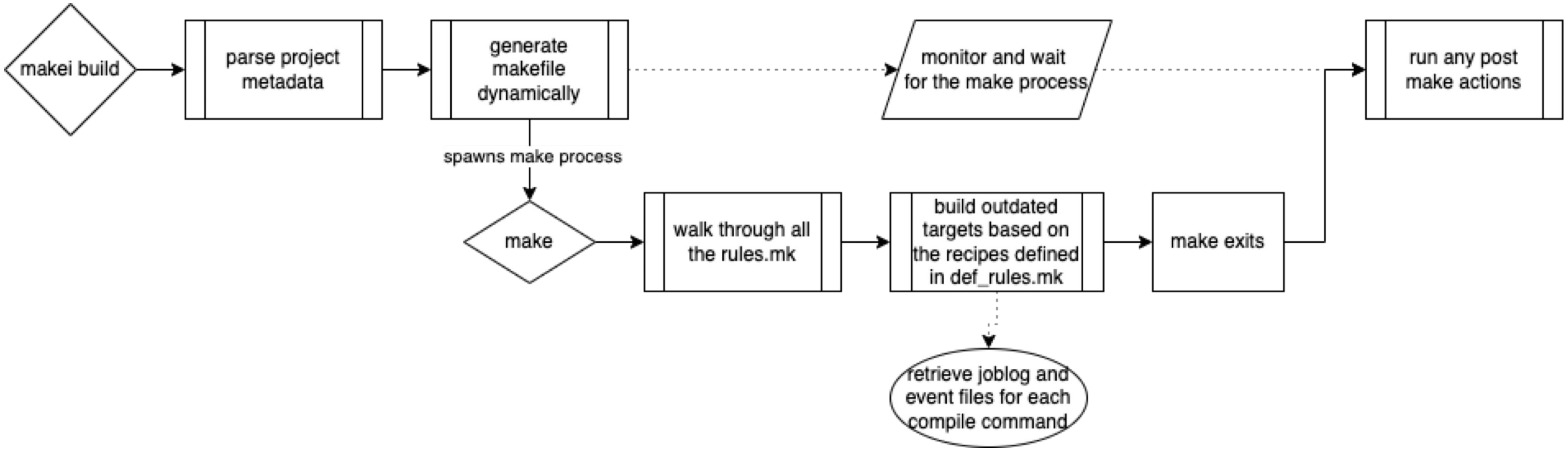
How BOB works and how to develop it

v2.4.10

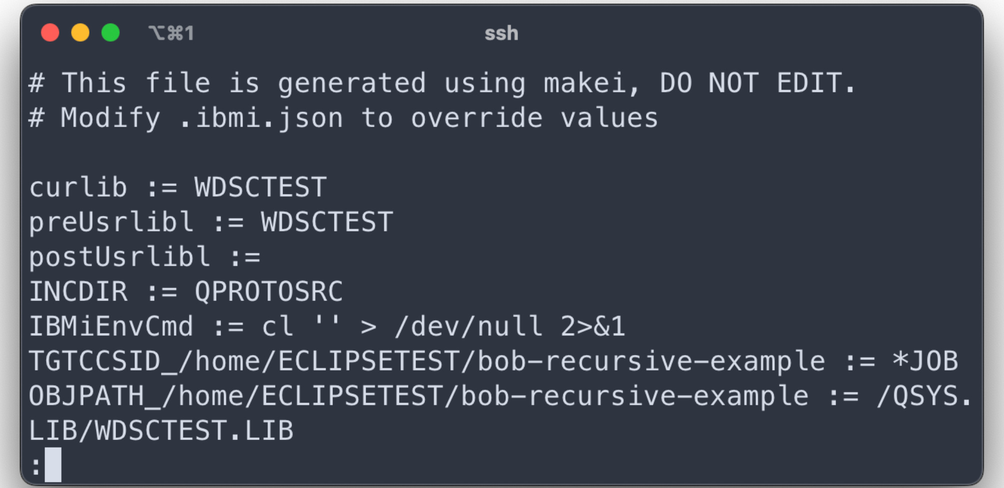
BOB Technical Details

Build Process


makei build
parse project metadata
generate makefile dynamically
monitor and wait for the make process
run any post make actions
spawns make process
make
walk through all the rules.mk
build outdated targets based on the recipes defined in def_rules.mk
make exits
retrieve joblog and event files for each compile command

Temporary files generated during 'makei build'

build vars under tmp directory

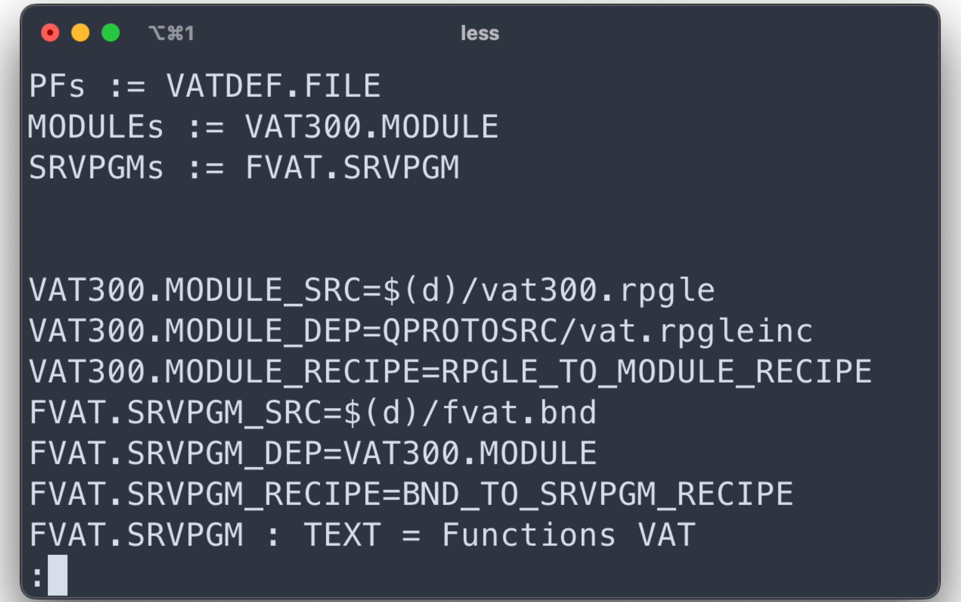


```
# This file is generated using makei, DO NOT EDIT.
# Modify .ibmi.json to override values

curlib := WDSCTEST
preUsrlibl := WDSCTEST
postUsrlibl :=
INCDIR := QPROTOSRC
IBMiEnvCmd := cl '' > /dev/null 2>&1
TGTCCSID_/home/ECLIPSETEST/bob-recursive-example := *JOB
OBJPATH_/home/ECLIPSETEST/bob-recursive-example := /QSYS.
LIB/WDSCTEST.LIB
:
```

- Resolves all per directory settings

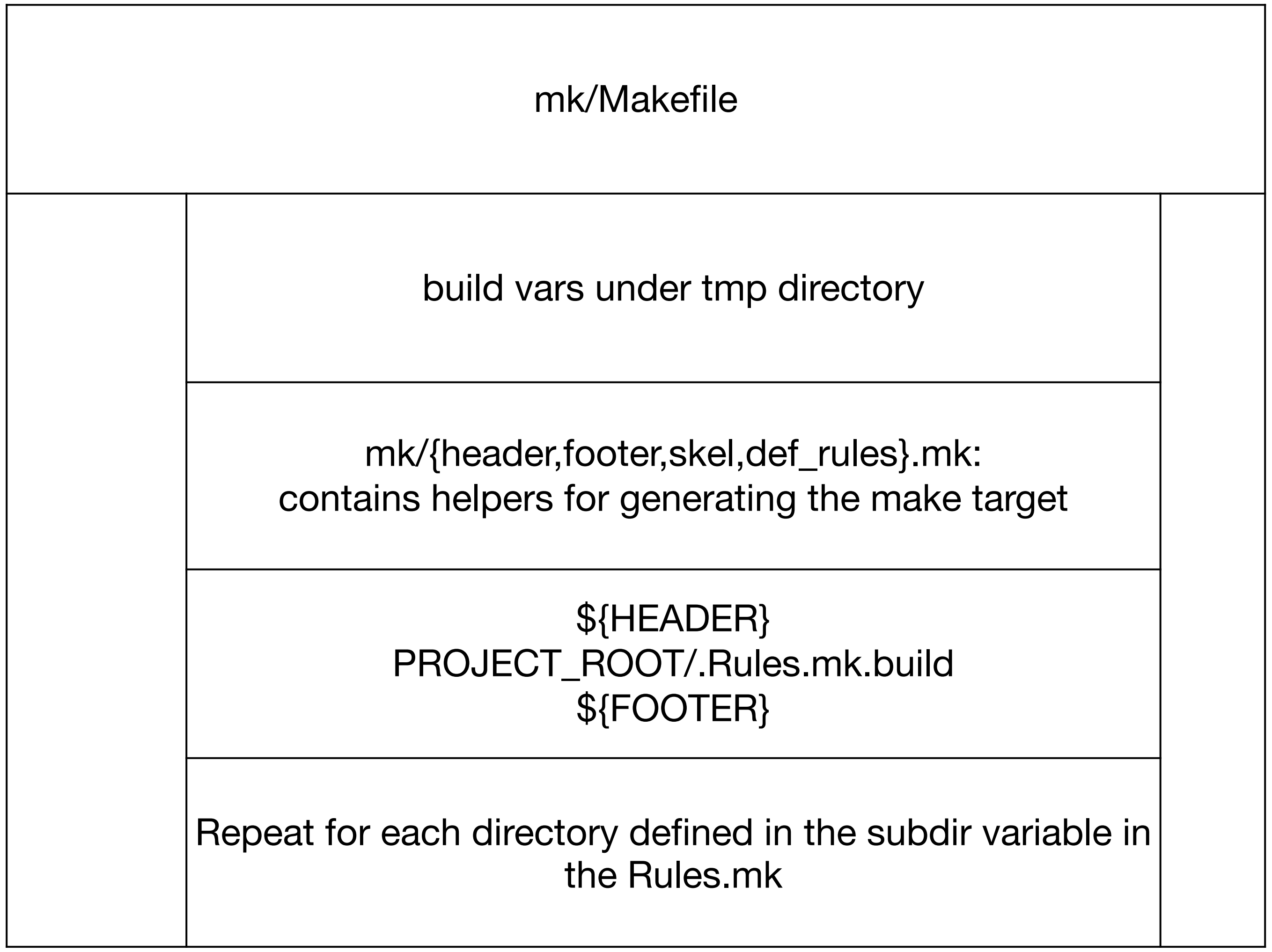
.Rules.mk.build files along with each Rules.mk

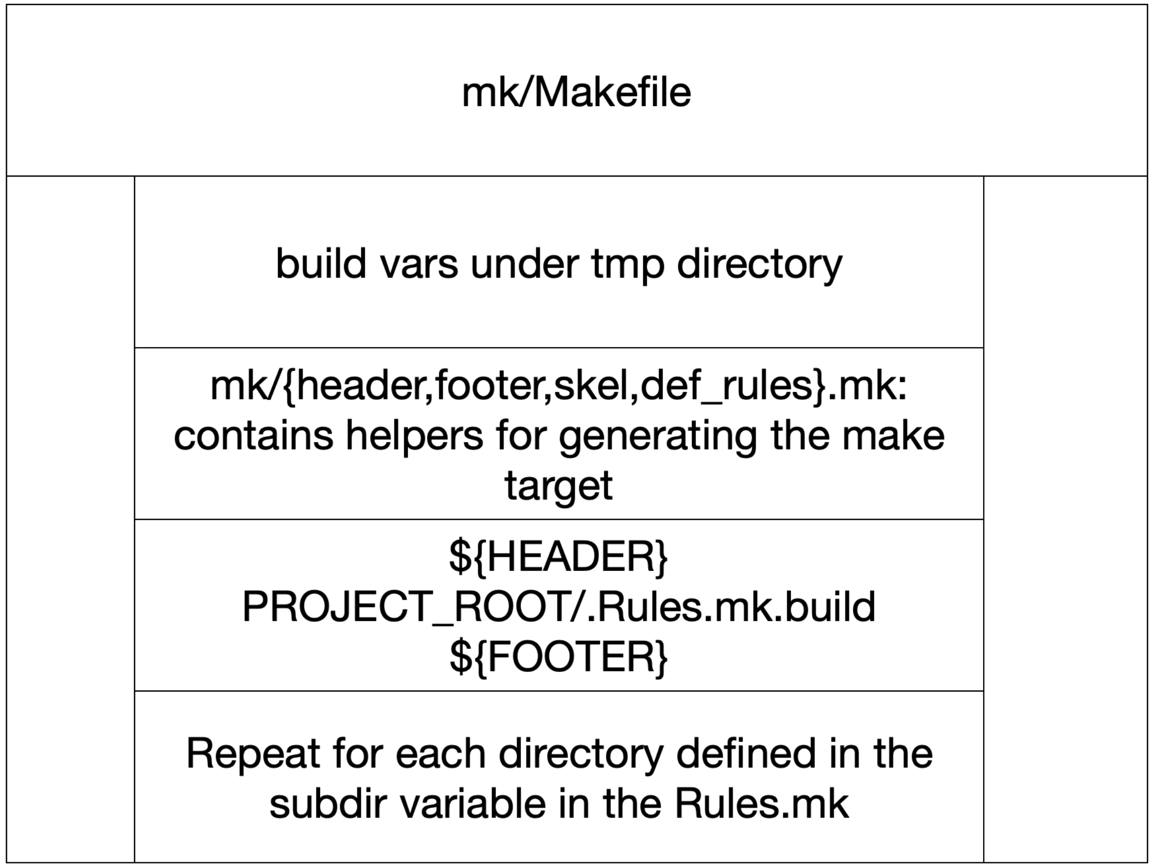


```
PFs := VATDEF.FILE
MODULEs := VAT300.MODULE
SRVPGMs := FVAT.SRVPGM


VAT300.MODULE_SRC=$(d)/vat300.rpgle
VAT300.MODULE_DEP=QPROTOSRC/vat.rpgleinc
VAT300.MODULE_RECIPE=RPGLE_TO_MODULE_RECIPE
FVAT.SRVPGM_SRC=$(d)/fvat.bnd
FVAT.SRVPGM_DEP=VAT300.MODULE
FVAT.SRVPGM_RECIPE=BND_TO_SRVPGM_RECIPE
FVAT.SRVPGM : TEXT = Functions VAT
:
```

- Expand Rules.mk with BOB-specific syntax


mk/Makefile
build vars under tmp directory
mk/{header,footer,skel,def_rules}.mk:
contains helpers for generating the make target
${HEADER}
PROJECT_ROOT/.Rules.mk.build
${FOOTER}
Repeat for each directory defined in the subdir variable in the Rules.mk



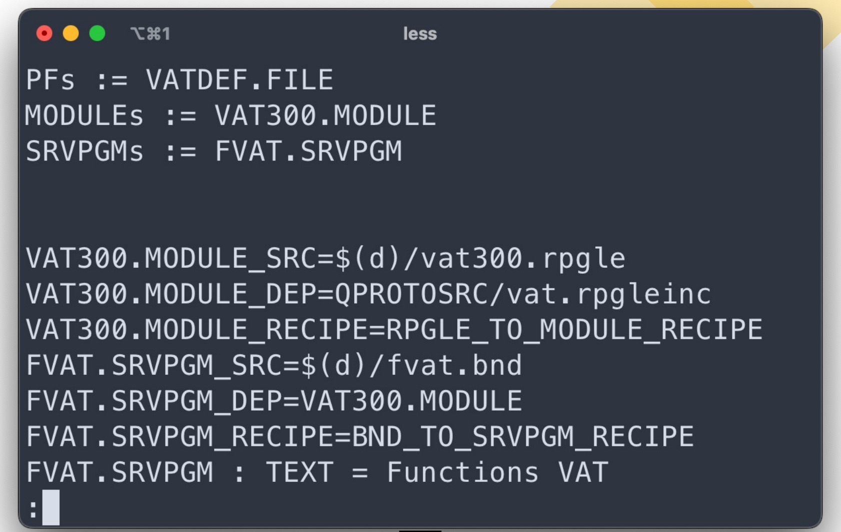
mk/Makefile

build vars under tmp directory

mk/{header,footer,skel,def_rules}.mk: contains helpers for generating the make target

${HEADER}
PROJECT_ROOT/.Rules.mk.build
${FOOTER}

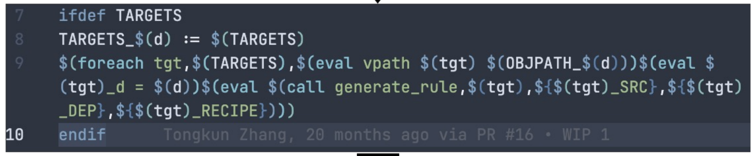
Repeat for each directory defined in the subdir variable in the Rules.mk



```
PFs := VATDEF.FILE
MODULEs := VAT300.MODULE
SRVPGMs := FVAT.SRVPGM


VAT300.MODULE_SRC=$(d)/vat300.rpgle
VAT300.MODULE_DEP=QPROTOSRC/vat.rpgleinc
VAT300.MODULE_RECIPE=RPGLE_TO_MODULE_RECIPE
FVAT.SRVPGM_SRC=$(d)/fvat.bnd
FVAT.SRVPGM_DEP=VAT300.MODULE
FVAT.SRVPGM_RECIPE=BND_TO_SRVPGM_RECIPE
FVAT.SRVPGM : TEXT = Functions VAT
:
```

mk/footer.mk:7



```
7   ifdef TARGETS
8   TARGETS_$(d) := $(TARGETS)
9   $(foreach tgt,$(TARGETS),$(eval vpath $(tgt) $(OBJPATH_$(d)))$(eval $
    (tgt)_d = $(d))$(eval $(call generate_rule,$(tgt),${$(tgt)_SRC},${$(tgt)
    _DEP},${$(tgt)_RECIPE})))
10  endif       Tongkun Zhang, 20 months ago via PR #16 • WIP 1
```

VAT300.MODULE: $(d)/vat300.rpgle QPROTOSRC/vat.rpgleinc ; [env var setup...] ${RPGLE_TO_MODULE_RECIPE}

To setup for development

- Follow instruction at https://ibm.github.io/ibmi-bob/#/contributing/getting-started
  - All driven by `noxfile.py`
  - `nox –s dev`
    - Sets up the python virtual environment and necessary prerequisites
  - `nox –s lint`
    - Run lint on the project
  - `nox –s test`
    - Run junit tests
    - Minimal set –needs to be expanded
  - Release process
    - Switch to the master branch and pull the latest code.
    - Update the CHANGELOG file under the changelogs folder. Make sure you add the new version.
    - Use `nox -s release -- {major, minor, patch}` to release a new version. For example, if you want to release a new patch version, you can run nox -s release -- patch. This will bump the version number, create a new tag, and push the tag to the remote repository.
    - Once the new tag is pushed, the CI will automatically
      - build the RPM and upload it to the release
      - create the spec file and create a new pull request to the spec file repository

BOB Components

- https://github.com/IBM/ibmi-bob
    - bin (python)
        - crtfrmstmf – compile source types with no stream file support
        - makei – all the other BOB functionality
    - docs (markdown) – documentation available at https://ibm.github.io/ibmi-bob/#/
    - src
        - mk – (makefile) makefile templates
        - makei – (python) makei functionality (generate makefile and launch gmake with right env)
        - scripts – (unix scripts) function during make recipes
    - tests (python) – unit tests
    - tools (nox) – dev env setup, rpm build, rpm deploy to IBM catalog

makei/

- The Python package for the makei command
- build.py – Implements `makei build/compile`
- const.py – Defines default values
- crtfrmstmf.py
  - Some compilers does not support build from stream file or build from utf-8
  - Copy a stream file to the QTEMP/QSOURCE and compile from there
  - Retrieve the joblog, event files and update the file path

makei

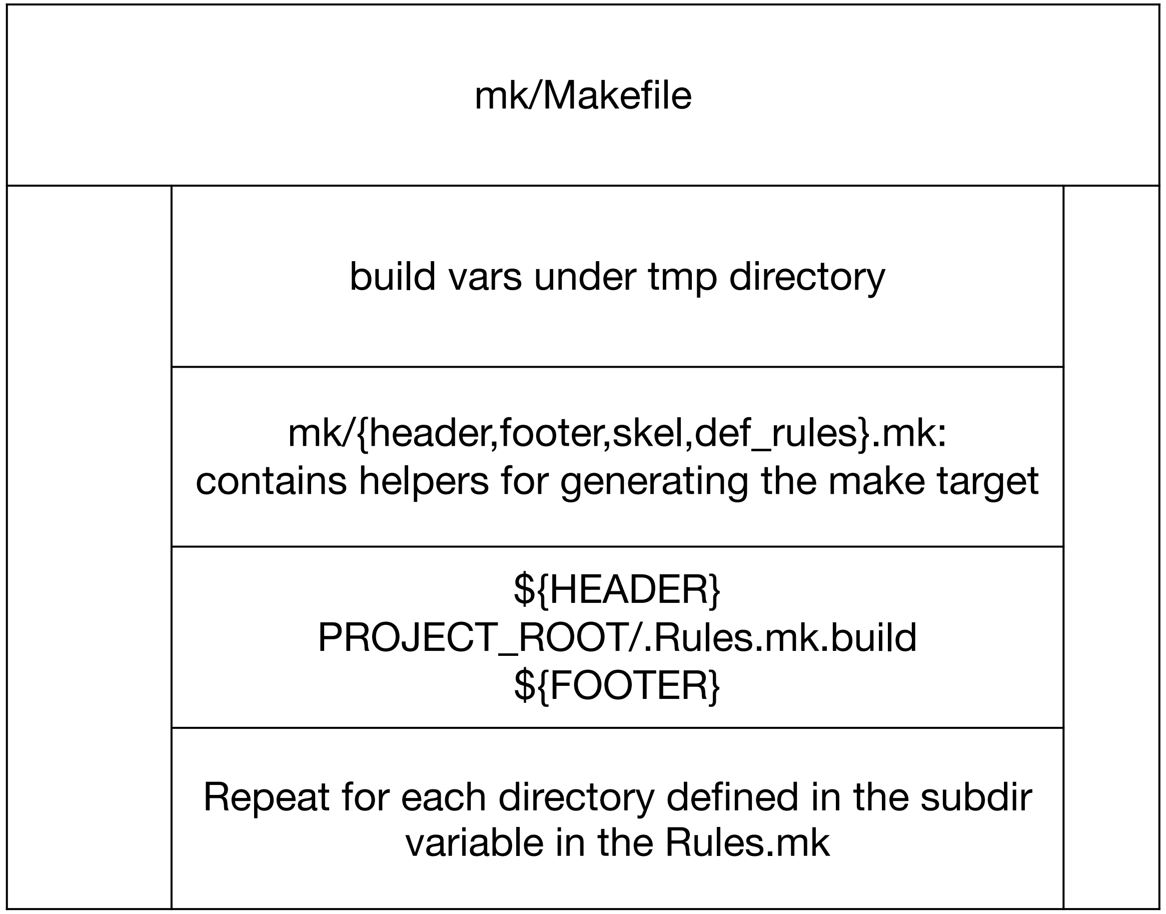
- init_project.py
  - Implements `makei init`
  - An interactive program to initialize a new Bob project
- utils.py - helper functions

makei/ibm_job.py

- Expose the IBMJob class to run a CL command or SQL statements (within the same job)
- job = IBMJob()
- job.job_id
- job.run_cl() / job.run_sql()
- job.dump_joblog()

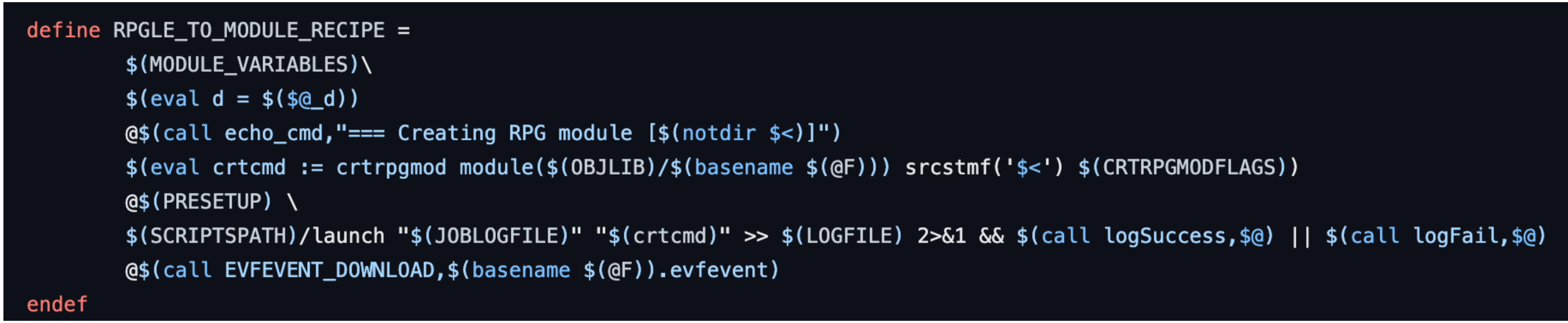
mk/

- Makefile
  - The entry makefile for the make process
  - Include all of the other makefiles
- header.mk
- footer.mk
- skel.mk
- def_rules.mk


mk/Makefile
build vars under tmp directory
mk/{header,footer,skel,def_rules}.mk:
contains helpers for generating the make target
${HEADER}
PROJECT_ROOT/.Rules.mk.build
${FOOTER}
Repeat for each directory defined in the subdir variable in the Rules.mk

def_rules.mk – the beating heart of BOB

- https://github.com/IBM/ibmi-bob/blob/master/src/mk/def_rules.mk
- There are specific recipes for each source-target combination



```
define RPGLE_TO_MODULE_RECIPE =
        $(MODULE_VARIABLES)\
        $(eval d = $($@_d))
        @$(call echo_cmd,"=== Creating RPG module [$(notdir $<)]")
        $(eval crtcmd := crtrpgmod module($(OBJLIB)/$(basename $(@F))) srcstmf('$<') $(CRTRPGMODFLAGS))
        @$(PRESETUP) \
        $(SCRIPTSPATH)/launch "$(JOBLOGFILE)" "$(crtcmd)" >> $(LOGFILE) 2>&1 && $(call logSuccess,$@) || $(call logFail,$@)
        @$(call EVFEVENT_DOWNLOAD,$(basename $(@F)).evfevent)
endef
```

MODULE_VARIABLES allows overriding of compile parameters

Launch – bash script that executes compile command

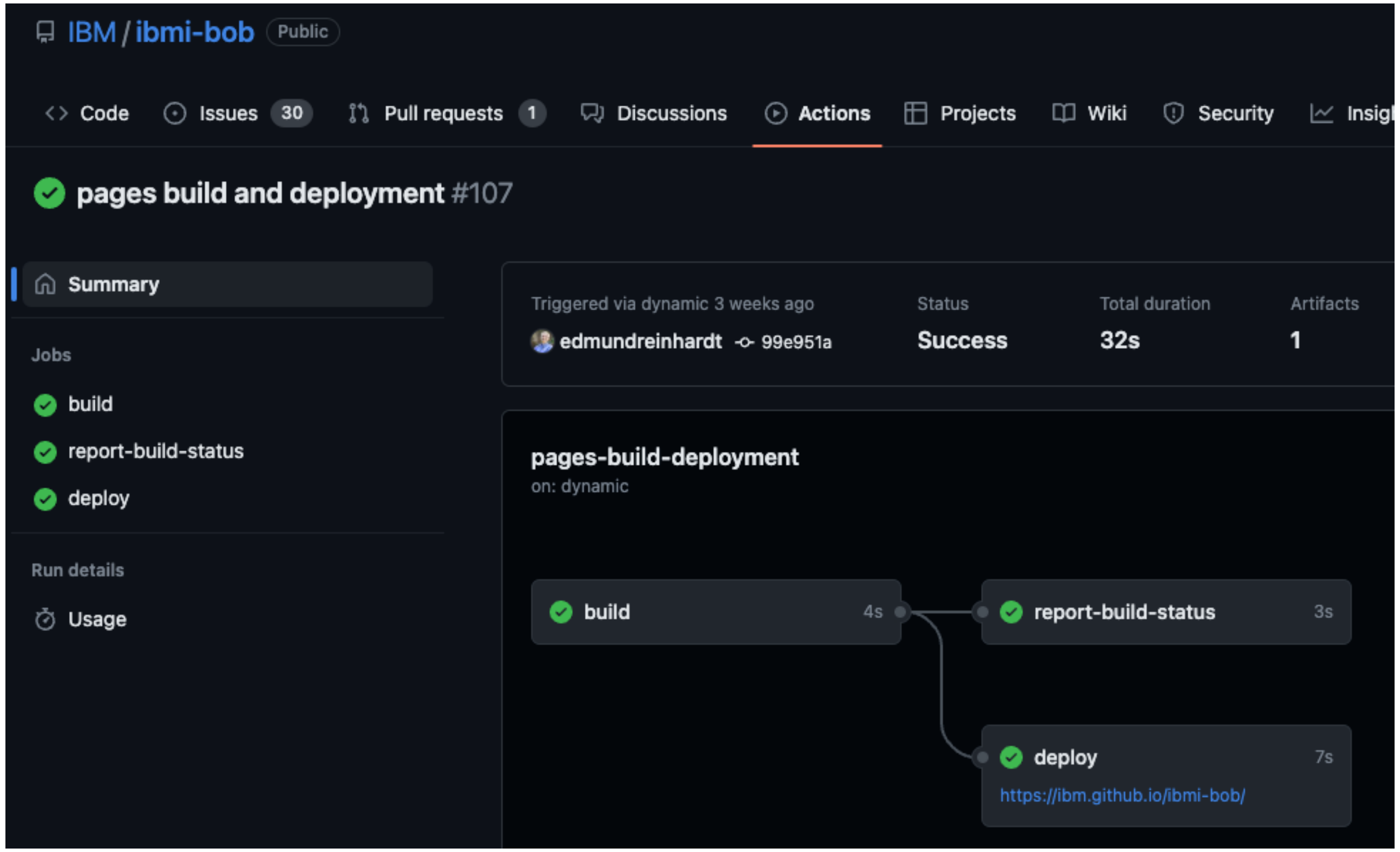
EVFEVENT_DOWNLOAD - copies back event file from QSYS to .evfevent

src/scripts

- launch
  - Setup environment and run the given CL command, extract joblog.json
- extractPseudoSrc / extractAndLaunch
  - Extract the CL command from a pseudo source (and run it)
- getJobLog
  - Append the joblog for a given job to the joblog.json

docs/

- Simply edit markdown in /docs directory
- Update _sidebar.md when adding new pages
- Generated via docsify into https://ibm.github.io/ibmi-bob/
- This is done automatically via github actions



IBM / ibmi-bob Public

pages build and deployment #107

Triggered via dynamic 3 weeks ago	Status	Total duration	Artifacts
edmundreinhardt -o- 99e951a	Success	32s	1

pages-build-deployment
on: dynamic

docsify 4.12.2

A magical documentation site generator.

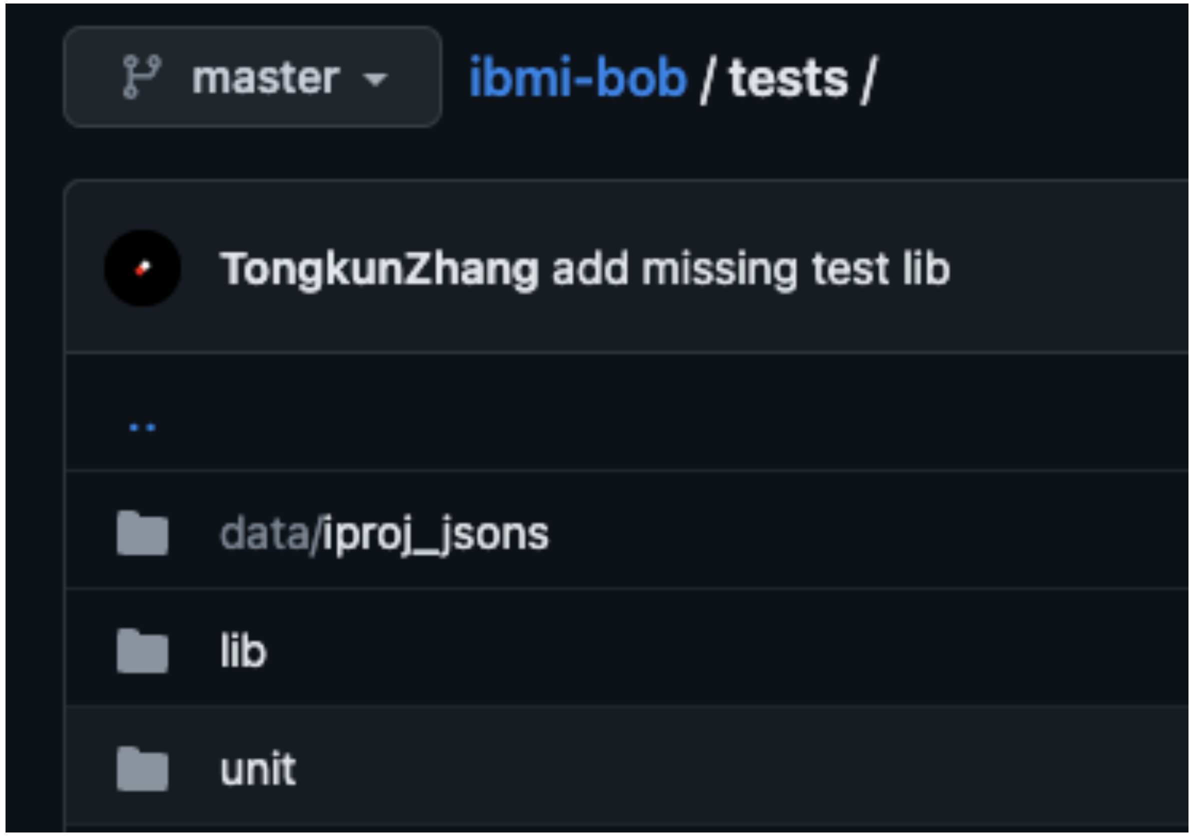
Simple and lightweight
No statically built html files
Multiple themes

tools/

- To support the nox command lines
- Specifically supporting the release functionality and the generation of bob.spec for the rpm
- See https://ibm.github.io/ibmi-bob/#/contributing/getting-started for more information on the release process

tests/

- Python JUnit tests in unit/
- Supporting test framework in lib/
- Test data in data/


master
ibmi-bob / tests /
TongkunZhang add missing test lib
..
data/iproj_jsons
lib
unit