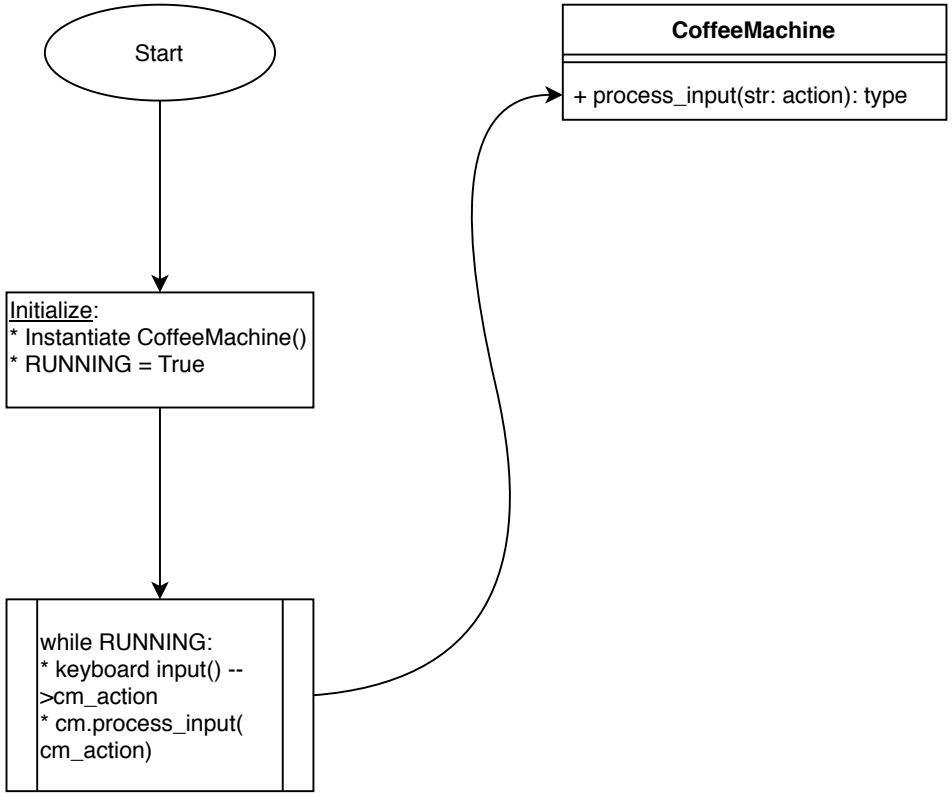
Start
CoffeeMachine
+ process_input(str: action): type
Initialize:
* Instantiate CoffeeMachine()
* RUNNING = True
while RUNNING:
* keyboard input() --
>cm_action
* cm.process_input(
cm_action)

HEADER:
* initialize CONSTANTS

__init__():
* initialize variables
* set initial Class State

process_input(action):
* check Class State &
choose appropriate response
* display message for next
input
==========
RESPONSES:
==========
if state = MAIN MENU
then:
if action is BUY, display
COFFEE CHOICE MENU,
change state to BUY
if action is FILL, change state
to FILL
if action is TAKE, change state
to TAKE
if action is REMAINING,
change state to REMAINING
if action is EXIT, change state
to EXIT

if state = BUY
then send coffee selection (1,
2, 3) to BUY_COFFEE method
if response is anything else,
then return to MAIN MENU

BUY COFFEE method:
call CHECK SUPPLIES
method to see if supplies are
available for selection
if supplies are available,
report available, collect money
& deduct supplies from
inventory
if supplies not available, then
report 'not enough' message
set status to MAIN MENU

if state = FILL
then send any response to
REPLENISH SUPPLIES
method

FILL SUPPLIES method:
on first sequential call to
REPLENISH SUPPLIES,
initialize FILL STEP to -1,
for each successive call to
method,
increment FILL STEP,
response will be quantity
amount to increase supplies
state will remain FILL until final
ingredient, then changed to
MAIN MENU

if state = TAKE
then call TAKE MONEY
method
& set state to MAIN MENU

TAKE MONEY method:
report money amount taken
zero out COLLECTED MONEY
& set status to MAIN MENU

if state = REMAINING
then call REPORT STATUS
method
& set state to MAIN MENU

REPORT STATUS method:
print out inventory levels of all
supplies & collected money
set status to MAIN MENU

if state = EXIT
then set Global variable
RUNNING to FALSE