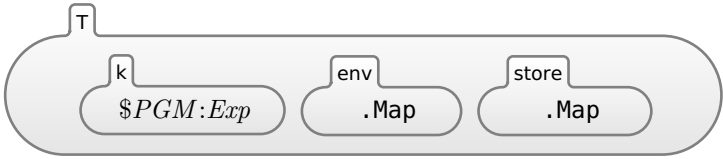


LAMBDA

MODULE LAMBDA

SYNTAX $Exp ::= Id$
 | $\lambda Id. Exp$
 | $Exp \ Exp \text{ [strict(strict())]}$
 | $(Exp) \text{ [bracket(bracket())]}$

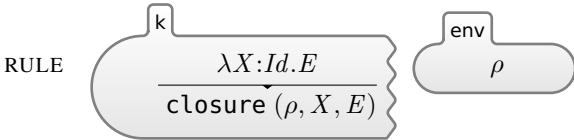
CONFIGURATION:



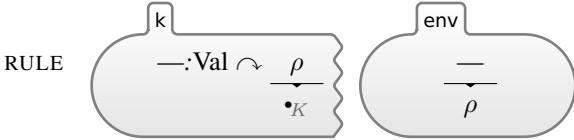
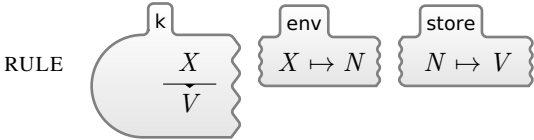
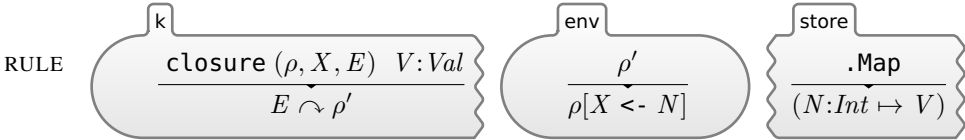
SYNTAX $Val ::= \text{closure} (Map, Id, Exp) \text{ [klabel(klabel('closure'))]}$

SYNTAX $Exp ::= Val$

SYNTAX $KResult ::= Val$



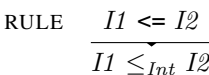
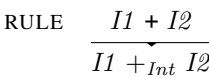
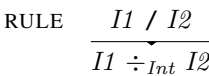
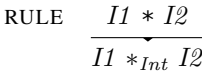
[structural(structural())]



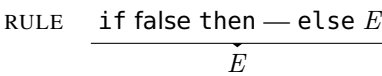
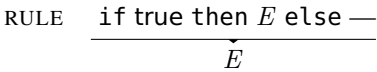
[structural(structural())]

SYNTAX $Val ::= Int$
 | $Bool$

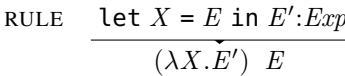
SYNTAX $Exp ::= Exp * Exp \text{ [strict(strict())]}$
 | $Exp / Exp \text{ [strict(strict())]}$
 | $Exp + Exp \text{ [strict(strict())]}$
 | $Exp \leq Exp \text{ [strict(strict())]}$



SYNTAX $Exp ::= \text{if } Exp \text{ then } Exp \text{ else } Exp \text{ [strict(strict(1))]}$

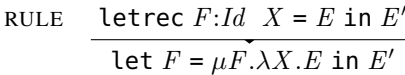


SYNTAX $Exp ::= \text{let } Id = Exp \text{ in } Exp$



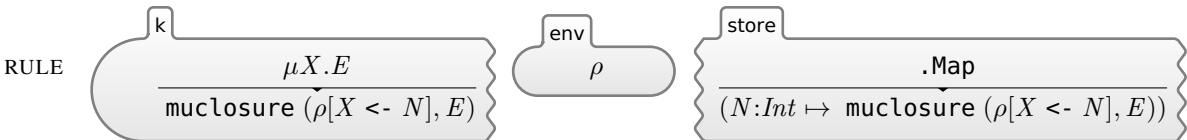
[macro(macro())]

SYNTAX $Exp ::= \text{letrec } Id \ Id = Exp \text{ in } Exp$
 | $\mu Id. Exp$

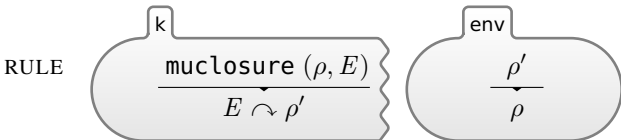


[macro(macro())]

SYNTAX $Exp ::= \text{muclosure} (Map, Exp) \text{ [klabel(klabel('muclosure'))]}$

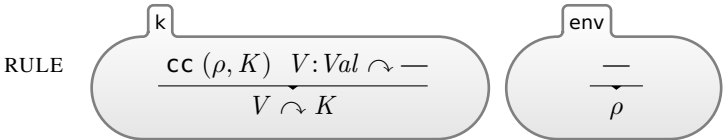
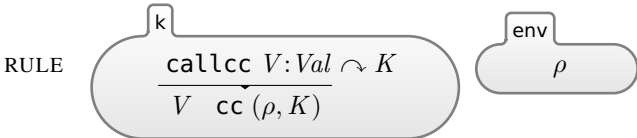


[structural(structural())]



SYNTAX $Exp ::= \text{callcc } Exp \text{ [strict(strict())]}$

SYNTAX $Val ::= \text{cc} (Map, K) \text{ [klabel(klabel('cc'))]}$



END MODULE