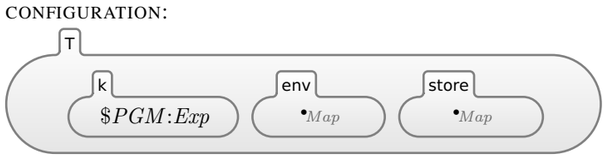
LAMBDA

MODULE LAMBDA

SYNTAX $Exp ::= Id$
$\mid \lambda Id.Exp$
$\mid Exp\ \ Exp$ [strict]
$\mid (Exp)$ [bracket]

CONFIGURATION:

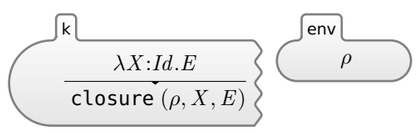


SYNTAX $Val ::= \texttt{closure}\ (Map, Id, Exp)$ [klabel('closure)]

SYNTAX $Exp ::= Val$

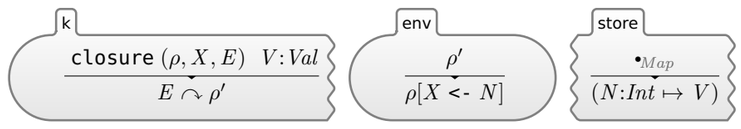
SYNTAX $KResult ::= Val$

RULE

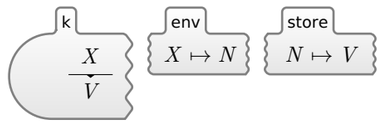


[structural]

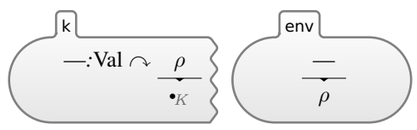
RULE



RULE



RULE



[structural]

SYNTAX $Val ::= Int$
$\mid Bool$

SYNTAX $Exp ::= Exp * Exp$ [strict]
$\mid Exp\ /\ Exp$ [strict]
$\mid Exp + Exp$ [strict]
$\mid Exp <= Exp$ [strict]

RULE $\dfrac{I1 * I2}{I1 *_{Int} I2}$

RULE $\dfrac{I1\ /\ I2}{I1 \div_{Int} I2}$

RULE $\dfrac{I1 + I2}{I1 +_{Int} I2}$

RULE $\dfrac{I1 <= I2}{I1 \leq_{Int} I2}$

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

RULE $\dfrac{\texttt{if true then}\ E\ \texttt{else}\ —}{E}$

RULE $\dfrac{\texttt{if false then}\ —\ \texttt{else}\ E}{E}$

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

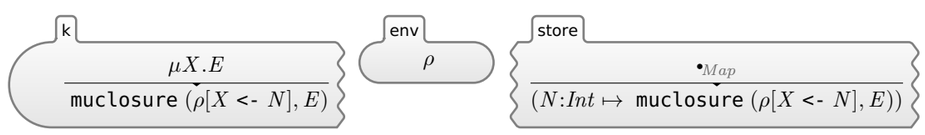
RULE $\dfrac{\texttt{let}\ X = E\ \texttt{in}\ E' : Exp}{(\lambda X.E')\ \ E}$ [macro]

SYNTAX $Exp ::= \texttt{letrec}\ Id\ \ Id = Exp\ \texttt{in}\ Exp$
$\mid \mu Id.Exp$

RULE $\dfrac{\texttt{letrec}\ F:Id\ \ X = E\ \texttt{in}\ E'}{\texttt{let}\ F = \mu F.\lambda X.E\ \texttt{in}\ E'}$ [macro]

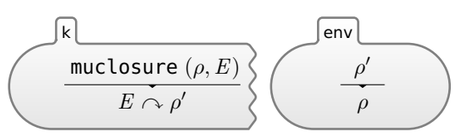
SYNTAX $Exp ::= \texttt{muclosure}\ (Map, Exp)$ [klabel('muclosure)]

RULE



[structural]

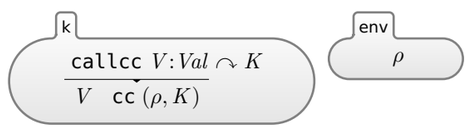
RULE



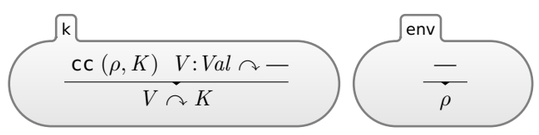
SYNTAX $Exp ::= \texttt{callcc}\ Exp$ [strict]

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

RULE



RULE



END MODULE