1 Featherweight Java progress

Proof. We use induction on typing derivations.

- T-VAR t is not a closed term, so the conclusion is trivially satisfied.
- T-FIELD We have: $t = t_0.f_i$, $\Gamma \vdash t_0.f_i :\ C_i$, $\Gamma \vdash t_0 :\ C_0$ and $fields(C_0) = \bar{C}\ \bar{f}$.
  t_0 is closed and well typed, by the induction hypothesis we have:
  - t_0 is a value, therefore $t = new\ C(\bar{v}).f_i$ and t can take a step by rule E-PROJNEW
  - t_0 can take a step to t_0', therefore $t \to t_0'.f_i$.
- T-INVK We have: $t = t_0.m(\bar{t})$, $\Gamma \vdash t_0.m(\bar{t}) :\ C$, $\Gamma \vdash \bar{t} :\ \bar{C}$.
  t_0 and $\bar{t}$ are closed and well formed terms, by the induction hypothesis they are either values or can take a step.
  - t_0 or one of $\bar{t}$ can take a step. Then $t \to t'$ where $t' = t_0'.m(\bar{t})$ or $t' = v_0.m(\bar{v}, t', \bar{v})$.
  - t_0 and $\bar{t}$ are values. Then t can take a step by rule E-INVKNEW and $t' = [\bar{x} \mapsto \bar{v}, this \mapsto v_0]b_0$, where $\bar{x}$ and b_0 are the parameters and body of method m.
- T-NEW We have: $t = new\ C(\bar{t})$ and $\Gamma \vdash \bar{t} : \bar{C}$.
  $\bar{t}$ are closed, well typed terms. By the induction hypothesis we have:
  - one of them takes a step $t_i \to t_i'$, therefore $t \to new\ C(\bar{v}, t_i', \bar{v})$.
  - all are values, therefore $t = new\ C(\bar{v})$ and is itself a value.

□

2 Featherweight Java with Field Assignment

1.

$$
\begin{array}{llllr}
t & ::= & & & \texttt{terms:} \\
 & | & & & ... \\
 & | & t.f & = \ t & \textbf{assignment} \\
 & | & l & & \textbf{store location} \\
v & ::= & & & \texttt{values:} \\
 & | & l & & \textbf{store location}
\end{array}
$$

2. Store μ is a mapping from location l to a tuple containing class type C and partial function F representing mapping from fields of the class to values (or more precisely to locations).

$$\text{(E-New)}\ \frac{l \notin dom(\mu) \quad fields(C) = \bar{C}\bar{f}}{new\ C(\bar{v}) \mid \mu \longrightarrow l \mid (\mu, l \mapsto (C, \bar{f} \mapsto \bar{v}))}$$

$$\text{(E-Proj)}\ \frac{\mu(l) = (C, F) \quad F(f_i) = v_i}{l.f_i \mid \mu \longrightarrow v_i \mid \mu}$$

$$\text{(E-Assign)}\ \frac{\mu(l) = (C, F) \quad fields(C) = \bar{C}\bar{f}}{l.f_i = v \mid \mu \longrightarrow l \mid [l \mapsto (C, [f_i \mapsto v]F)]\mu}$$

$$\text{(E-CastNew)}\ \frac{\mu(v) = (C, F) \quad C <: D}{(D)v \mid \mu \longrightarrow v \mid \mu}$$

$$\text{(E-InvkNew)}\ \frac{\mu(l) = (C, F) \quad mbody(m, C) = (\bar{x}, t_0)}{l.m(\bar{v}) \mid \mu \longrightarrow [\bar{x} \mapsto \bar{v}, this \mapsto l]t_0 \mid \mu}$$

$$\text{(E-Field)}\ \frac{t_0 \mid \mu \longrightarrow t_0' \mid \mu'}{t_0.f \mid \mu \longrightarrow t_0'.f \mid \mu'}$$

$$\text{(E-AssignC1)}\ \frac{t_0 \mid \mu \longrightarrow t_0' \mid \mu'}{t_0.f_i = t \mid \mu \longrightarrow t_0'.f_i = t \mid \mu'}$$

$$\text{(E-AssignC2)}\ \frac{t \mid \mu \longrightarrow t' \mid \mu'}{v.f_i = t \mid \mu \longrightarrow v.f = t' \mid \mu'}$$

3.

$$\text{(T-STORE)}\ \frac{\Sigma(l) = T}{\Gamma \mid \Sigma \vdash l : T}$$

$$\text{(T-Assign)}\ \frac{\begin{array}{c}\Gamma \mid \Sigma \vdash t_0 : T \\ fields(C) = \bar{D}\bar{f} \\ \Gamma \mid \Sigma \vdash t : T' \quad T' <: D_i\end{array}}{\Gamma \mid \Sigma \vdash t_0.f_i = t : T}$$

$$\text{(T-New)}\ \frac{\begin{array}{c}fields(C) = \bar{D}\bar{f} \\ \Gamma \mid \Sigma \vdash \bar{t} : \bar{C} \quad \bar{C} <: \bar{D}\end{array}}{\Gamma \mid \Sigma \vdash new\ C(\bar{t}) : T}$$

$$\text{(T-INVK)}\ \frac{\begin{array}{c}\Gamma \mid \Sigma \vdash t_0 : C_0 \\ mtype(m, C_0) = \bar{D} \to C \\ \Gamma \mid \Sigma \vdash \bar{t} : \bar{C} \quad \bar{C} <: \bar{D}\end{array}}{\Gamma \mid \Sigma \vdash t_0.m(\bar{t}) : T}$$

3 Featherweight Java big-step evaluation semantics

1. Small-step semantics describes how individual steps of every evaluation are performed, whereas the reduction rules for big-step semantics define the overall result of the computation i.e., we always reduce to a value in the rule.

2. For languages that define concurrent behaviour it seems to be more appropriate to formalize using big-step semantics since then one doesn't have to consider all the possible combinations of the interleaving computations (Eugene: not sure whether they would know this).

3. Evaluation rules:

$$\frac{t \Downarrow \textbf{new}\ \mathtt{C}(v) \quad \textbf{fields}(\mathtt{C}) = \mathtt{C}\ f}{t.f_i \Downarrow v_i} \tag{E-Select}$$

$$\frac{t_0 \Downarrow \textbf{new}\ \mathtt{C}(v) \quad \textbf{mbody}(m, \mathtt{C}) = (x, s) \quad t \Downarrow u \quad [\mathtt{this} \mapsto \textbf{new}\ \mathtt{C}(v), x \mapsto u]s \Downarrow v}{t_0.m(t) \Downarrow v} \tag{E-Invoke}$$

$$\frac{t \Downarrow \textbf{new}\ \mathtt{C}(v) \quad \mathtt{C<:D}}{(\mathtt{D})\ t \Downarrow \textbf{new}\ \mathtt{C}(v)} \tag{E-Cast}$$

$$\frac{t \Downarrow v}{\textbf{new}\ \mathtt{C}(t) \Downarrow \textbf{new}\ \mathtt{C}(v)} \tag{E-New}$$

4. If $\Gamma t : \mathtt{C}$ and $t \Downarrow v$, then $\Gamma v : \mathtt{D}$ for some $\mathtt{D} <: \mathtt{C}$.