Drei

syntaxe abstraite et règles de validité du typage

LAMP

version pour l'ÉNS Lyon

Notations

Notation	Interprétation
$\overline{a}$	séquence $a_1, \ \ldots \ , a_n$ pour $n \in \mathbb{N}$
ϵ	séquence vide
$\|\overline{a}\|$	longueur de la séquence $\overline{a}$
$\overline{a}, \overline{b}$	concaténation des séquences $\overline{a}$ et $\overline{b}$
$\overline{a} \mapsto \overline{\sigma}$	$a_1 \mapsto \sigma_1, \ \ldots \ , a_n \mapsto \sigma_n$
$dom(\overline{a} \mapsto \overline{\sigma})$	$\overline{a}$
$\Gamma_c; \Gamma_v \vdash \overline{t} : \overline{T}$	$\Gamma_c; \Gamma_v \vdash t_1 : T_1, \ \ldots \ , \Gamma_c; \Gamma_v \vdash t_n : T_n$
$\Gamma \vdash \overline{X} \ \Rightarrow \ \Gamma'$	Γ_n pour $\begin{cases} \Gamma \vdash X_1 \ \Rightarrow \ \Gamma_1 \\ \vdots \\ \Gamma_{n-1} \vdash X_n \ \Rightarrow \ \Gamma_n \end{cases}$
$\Gamma + (a \mapsto \sigma)$	$\begin{cases} \Gamma, a \mapsto \sigma & \text{si } a \notin dom(\Gamma) \\ \Gamma', a \mapsto \sigma, \Gamma'' & \text{si } \Gamma = \Gamma', a \mapsto \sigma', \Gamma'' \end{cases}$
$\Gamma \uplus \Gamma'$	Γ, Γ' si $dom(\Gamma) \cap dom(\Gamma') = \epsilon$
$fields(\overline{d})$	$\biguplus_{\texttt{val}\ a:T \,\in\, \overline{d}} (a \mapsto \texttt{Field}(T))$
$methods(\overline{d})$	$\biguplus_{\texttt{def}\ a(\overline{a}:\overline{T}):T=t \,\in\, \overline{d}} (a \mapsto \texttt{Meth}(\overline{T} \| T))$
$params(\overline{a}, \overline{T})$	$\biguplus_{a,T \,\in\, (\overline{a},\overline{T})} (a \mapsto \texttt{Var}(T))$

Grammaire abstraite

nom	a, b			
programmes	P	$::=$	$\overline{D}\ S$	
classes	D	$::=$	$\texttt{class}\ a\ \texttt{extends}\ s\ \{\overline{d}\}$	déclaration de classe
	s	$::=$	$a \mid \texttt{none}$	super classe
membres	d	$::=$	$\texttt{val}\ a : T$	déclaration de champ
		$\mid$	$\texttt{def}\ a(\overline{a} : \overline{T}) : T = t$	déclaration de méthode
types	T, U	$::=$	a	type de classe
		$\mid$	$\texttt{Int}$	type entier
		$\mid$	$\texttt{None}$	type indéterminé
expressions	t, u	$::=$	a	variable
		$\mid$	$\texttt{new}\ a(\overline{t})$	création d'instance
		$\mid$	$t.a$	sélection de champ
		$\mid$	$t.a(\overline{t})$	appel de méthode
		$\mid$	n	nombre entier
		$\mid$	$\mathit{unop}\ t$	opération unaire
		$\mid$	$t\ \mathit{binop}\ t'$	opération binaire
		$\mid$	$\texttt{readInt}$	lecture d'entier
		$\mid$	$\texttt{readChar}$	lecture de caractère
		$\mid$	$\{\ \overline{S}\ t\ \}$	block
		$\mid$	$\texttt{empty}$	
énoncés	S	$::=$	$\texttt{while}\ t\ S$	exécution en boucle
		$\mid$	$\texttt{if}\ t\ \texttt{then}\ S\ \texttt{else}\ S'$	exécution conditionelle
		$\mid$	$\texttt{var}\ a : T = t$	déclaration de variable
		$\mid$	$\texttt{set}\ a = t$	définition de variable
		$\mid$	$\texttt{do}\ t$	instruction
		$\mid$	$\texttt{printInt}(t)$	impression d'entier
		$\mid$	$\texttt{printChar}(t)$	impression de caractère
		$\mid$	$\{\ \overline{S}\ \}$	énoncé composite
op. unaires	unop	$::=$	$- \mid\ !$	
op. binaires	binop	$::=$	$+ \mid - \mid * \mid / \mid \%$	
		$\mid$	$= \mid \neq \mid < \mid \leq \mid \geq \mid >$	
		$\mid$	$\wedge$	

Symboles

classe	σ_c	::=	$\texttt{Class}(\overline{a}\vert\Gamma_f\vert\Gamma_m)$ $\overline{a}$: parents, Γ_f : champs, Γ_m : méthodes
champ	σ_f	::=	$\texttt{Field}(T)$ T : type du champ
méthode	σ_m	::=	$\texttt{Meth}(\overline{T}\vert T)$ $\overline{T}$: types des paramètres, T : type de retour
variable	σ_v	::=	$\texttt{Var}(T)$ T : type de la variable

Portées

classes	Γ_c	::=	$\overline{a} \mapsto \overline{\sigma_c}$
champs	Γ_f	::=	$\overline{a} \mapsto \overline{\sigma_f}$
méthodes	Γ_m	::=	$\overline{a} \mapsto \overline{\sigma_m}$
variables	Γ_v	::=	$\overline{a} \mapsto \overline{\sigma_v}$

Règles de typage

Programmes de la forme $P \diamond$

$$\textsc{Program}\ \frac{\texttt{none} \mapsto \texttt{Class}(\epsilon|\epsilon|\epsilon) \vdash \overline{D} \Rightarrow \Gamma_c \qquad \Gamma_c \vdash \overline{D} \diamond \qquad \Gamma_c; \epsilon \vdash S \Rightarrow \epsilon}{\overline{D}\ S \diamond}$$

Classes (insertion dans les portées) de la forme $\Gamma_c \vdash D \Rightarrow \Gamma'_c$

$$\textsc{Class1}\ \frac{\begin{array}{c} s \mapsto \texttt{Class}(\overline{a}|\Gamma_f|\Gamma_m) \in \Gamma_c \qquad \Gamma'_f = \Gamma_f \uplus \mathit{fields}(\overline{d}) \\ \Gamma'_m = \Gamma_m + \mathit{methods}(\overline{d}) \qquad \Gamma'_c = \Gamma_c \uplus (a \mapsto \texttt{Class}(a, \overline{a}|\Gamma'_f|\Gamma'_m)) \end{array}}{\Gamma_c \vdash \texttt{class}\ a\ \texttt{extends}\ s\ \{\overline{d}\} \Rightarrow \Gamma'_c}$$

Classes (vérification des membres) de la forme $\Gamma_c \vdash D \diamond$

$$\textsc{Class2}\ \frac{\Gamma_c; a \vdash \overline{d} \diamond}{\Gamma_c \vdash \texttt{class}\ a\ \texttt{extends}\ s\ \{\overline{d}\} \diamond}$$

Membres de la forme $\Gamma_c; b \vdash d \diamond$

$$\textsc{Field}\ \frac{\Gamma_c \vdash T \diamond}{\Gamma_c; b \vdash \texttt{val}\ a : T \diamond}$$

$$\textsc{Method}\ \frac{\begin{array}{c}\Gamma_c \vdash T \diamond \qquad \Gamma_c \vdash \overline{T} \diamond \qquad b \mapsto \texttt{Class}(\bar{b}|\Gamma_f|\Gamma_m) \in \Gamma_c \\ \forall c \in \bar{b}\ .\ c \mapsto \texttt{Class}(\overline{c}|\Gamma'_f|\Gamma'_m) \in \Gamma_c \wedge a \mapsto \texttt{Meth}(\overline{U}|U) \in \Gamma'_m \implies \begin{cases} \Gamma_c \vdash \overline{U} <: \overline{T} \\ \Gamma_c \vdash T <: U \end{cases} \\ \Gamma_v = params((this, \overline{a}), (b, \overline{T})) \qquad \Gamma_c; \Gamma_v \vdash t : T' \qquad \Gamma_c \vdash T' <: T\end{array}}{\Gamma_c; b \vdash \texttt{def}\ a(\overline{a} : \overline{T}) : T = t \diamond}$$

Types de la forme $\Gamma_c \vdash T \diamond$

$$\textsc{ClassType}\ \frac{a \mapsto \texttt{Class}(\overline{a}|\Gamma_f|\Gamma_m) \in \Gamma_c}{\Gamma_c \vdash a \diamond} \qquad \textsc{IntType}\ \Gamma_c \vdash \texttt{Int} \diamond$$

$$\textsc{NoType}\ \Gamma_c \vdash \texttt{None} \diamond$$

Sous-typage de la forme $\Gamma_c \vdash T <: T$

$$\textsc{SubClass}\ \frac{a \mapsto \texttt{Class}(\overline{a}|\Gamma_f|\Gamma_m) \in \Gamma_c}{\Gamma_c \vdash a <: a_i} \qquad \textsc{IntRefl}\ \Gamma_c \vdash \texttt{Int} <: \texttt{Int}$$

$$\textsc{NoneRefl}\ \Gamma_c \vdash \texttt{None} <: \texttt{None}$$

Expressions de la forme $\Gamma_c; \Gamma_v \vdash t : T$

$$\textsc{Ident}\ \frac{a \mapsto \texttt{Var}(T) \in \Gamma_v}{\Gamma_c; \Gamma_v \vdash a : T}$$

$$\textsc{Select}\ \frac{\Gamma_c; \Gamma_v \vdash t : b \qquad b \mapsto \texttt{Class}(\overline{b}|\Gamma_f|\Gamma_m) \in \Gamma_c \qquad a \mapsto \texttt{Field}(T) \in \Gamma_f}{\Gamma_c; \Gamma_v \vdash t.a : T}$$

$$\textsc{Call}\ \frac{\begin{array}{c}\Gamma_c; \Gamma_v \vdash t : b \qquad b \mapsto \texttt{Class}(\overline{b}|\Gamma_f|\Gamma_m) \in \Gamma_c \\ a \mapsto \texttt{Meth}(\overline{T}|T) \in \Gamma_m \qquad \Gamma_c; \Gamma_v \vdash \overline{t} : \overline{U} \qquad \Gamma_c \vdash \overline{U} <: \overline{T}\end{array}}{\Gamma_c; \Gamma_v \vdash t.a(\overline{t}) : T}$$

$$\textsc{New}\ \frac{\begin{array}{c}a \mapsto \texttt{Class}(\overline{b}|\Gamma_f|\Gamma_m) \in \Gamma_c \\ \Gamma_f = \overline{a \mapsto \texttt{Field}(T)} \qquad \Gamma_c; \Gamma_v \vdash \overline{t} : \overline{U} \qquad \Gamma_c \vdash \overline{U} <: \overline{T}\end{array}}{\Gamma_c; \Gamma_v \vdash \texttt{new}\ a(\overline{t}) : a}$$

$$\textsc{IntLit}\ \Gamma_c; \Gamma_v \vdash n : \texttt{Int} \qquad\qquad \textsc{Unop}\ \frac{\Gamma_c; \Gamma_v \vdash t : \texttt{Int}}{\Gamma_c; \Gamma_v \vdash \mathit{unop}\ t : \texttt{Int}}$$

$$\textsc{Binop}\ \frac{\mathit{binop} \notin \{=, \neq\} \qquad \Gamma_c; \Gamma_v \vdash t : \texttt{Int} \qquad \Gamma_c; \Gamma_v \vdash u : \texttt{Int}}{\Gamma_c; \Gamma_v \vdash t\ \mathit{binop}\ u : \texttt{Int}}$$

$$\textsc{ObjComp}\ \frac{\begin{array}{c}\mathit{binop} \in \{=, \neq\} \\ \Gamma_c; \Gamma_v \vdash t : T \qquad \Gamma_c; \Gamma_v \vdash u : U \qquad \Gamma_c \vdash T <: U \vee \Gamma_c \vdash U <: T\end{array}}{\Gamma_c; \Gamma_v \vdash t\ \mathit{binop}\ u : \texttt{Int}}$$

$$\textsc{ReadInt}\ \Gamma_c; \Gamma_v \vdash \texttt{readInt} : \texttt{Int} \qquad\qquad \textsc{ReadChar}\ \Gamma_c; \Gamma_v \vdash \texttt{readChar} : \texttt{Int}$$

$$\textsc{Block}\ \frac{\Gamma_c; \Gamma_v \vdash \overline{S}\ \Rightarrow\ \Gamma'_v \qquad \Gamma_c; \Gamma'_v \vdash t : T}{\Gamma_c; \Gamma_v \vdash \{\ \overline{S}\ t\ \} : T} \qquad\qquad \textsc{Empty}\ \Gamma_c; \Gamma_v \vdash \texttt{empty} : \texttt{None}$$

Enoncés de la forme $\Gamma_c; \Gamma_v \vdash S \;\Rightarrow\; \Gamma'_v$

$$\textsc{If}\ \frac{\Gamma_c; \Gamma_v \vdash t : \texttt{Int} \qquad \Gamma_c; \Gamma_v \vdash S \;\Rightarrow\; \Gamma_v \qquad \Gamma_c; \Gamma_v \vdash S' \;\Rightarrow\; \Gamma_v}{\Gamma_c; \Gamma_v \vdash \texttt{if}\ t\ \texttt{then}\ S\ \texttt{else}\ S' \;\Rightarrow\; \Gamma_v}$$

$$\textsc{While}\ \frac{\Gamma_c; \Gamma_v \vdash t : \texttt{Int} \qquad \Gamma_c; \Gamma_v \vdash S \;\Rightarrow\; \Gamma_v}{\Gamma_c; \Gamma_v \vdash \texttt{while}\ t\ S \;\Rightarrow\; \Gamma_v}$$

$$\textsc{Var}\ \frac{\Gamma_c \vdash T \diamond \qquad \Gamma_c; \Gamma_v \vdash t : U \qquad \Gamma_c \vdash U <: T \qquad \Gamma'_v = \Gamma_v \uplus a \mapsto \texttt{Var}(T)}{\Gamma_c; \Gamma_v \vdash \texttt{var}\ a : T = t \;\Rightarrow\; \Gamma'_v}$$

$$\textsc{Set}\ \frac{a \mapsto \texttt{Var}(T) \in \Gamma_v \qquad \Gamma_c; \Gamma_v \vdash t : U \qquad \Gamma_c \vdash U <: T}{\Gamma_c; \Gamma_v \vdash \texttt{set}\ a = t \;\Rightarrow\; \Gamma_v}$$

$$\textsc{Do}\ \frac{\Gamma_c; \Gamma_v \vdash t : T}{\Gamma_c; \Gamma_v \vdash \texttt{do}\ t \;\Rightarrow\; \Gamma_v} \qquad \textsc{PrintInt}\ \frac{\Gamma_c; \Gamma_v \vdash t : \texttt{Int}}{\Gamma_c; \Gamma_v \vdash \texttt{printInt}(t) \;\Rightarrow\; \Gamma_v}$$

$$\textsc{PrintChar}\ \frac{\Gamma_c; \Gamma_v \vdash t : \texttt{Int}}{\Gamma_c; \Gamma_v \vdash \texttt{printChar}(t) \;\Rightarrow\; \Gamma_v} \qquad \textsc{Compound}\ \frac{\Gamma_c; \Gamma_v \vdash \overline{S} \;\Rightarrow\; \Gamma'_v}{\Gamma_c; \Gamma_v \vdash \{\ \overline{S}\ \} \;\Rightarrow\; \Gamma_v}$$