

Drei

syntaxe abstraite et règles de validité du typage

Laboratoire des méthodes de programmation

révision 1.6

Notations

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

Grammaire abstraite

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

Symboles

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

Portées

classes	Γ_c	$::=$	$\bar{a} \mapsto \bar{\sigma}_c$
champs	Γ_f	$::=$	$\bar{a} \mapsto \bar{\sigma}_f$
méthodes	Γ_m	$::=$	$\bar{a} \mapsto \bar{\sigma}_m$
variables	Γ_v	$::=$	$\bar{a} \mapsto \bar{\sigma}_v$

Règles de typage

Programmes de la forme $P \diamond$

$$\frac{\text{PROGRAM} \quad \text{none} \mapsto \mathbf{Class}(\epsilon|\epsilon|\epsilon) \vdash \bar{D} \Rightarrow \Gamma_c \quad \Gamma_c \vdash \bar{D} \diamond \quad \Gamma_c; \epsilon \vdash S \Rightarrow \epsilon}{\bar{D} S \diamond}$$

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

$$\frac{\text{CLASS1} \quad s \mapsto \mathbf{Class}(\bar{a}|\Gamma_f|\Gamma_m) \in \Gamma_c \quad \Gamma'_f = \Gamma_f \uplus \text{fields}(\bar{d}) \quad \Gamma'_m = \Gamma_m + \text{methods}(\bar{d}) \quad \Gamma'_c = \Gamma_c \uplus (a \mapsto \mathbf{Class}(a, \bar{a}|\Gamma'_f|\Gamma'_m))}{\Gamma_c \vdash \mathbf{class} a \text{ extends } s \{ \bar{d} \} \Rightarrow \Gamma'_c}$$

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

$$\frac{\text{CLASS2} \quad \Gamma_c; a \vdash \bar{d} \diamond}{\Gamma_c \vdash \mathbf{class} a \text{ extends } s \{ \bar{d} \} \diamond}$$

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

$$\frac{\text{FIELD} \quad \Gamma_c \vdash T \diamond}{\Gamma_c; b \vdash \text{val } a : T \diamond}$$

METHOD

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

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

$$\frac{\text{CLASSTYPE} \quad a \mapsto \text{Class}(\bar{a}|\Gamma_f|\Gamma_m) \in \Gamma_c}{\Gamma_c \vdash a \diamond} \quad \text{INTTYPE} \quad \Gamma_c \vdash \text{Int} \diamond \quad \text{NOTYPE} \quad \Gamma_c \vdash \text{None} \diamond$$

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

$$\frac{\text{SUBCLASS} \quad a \mapsto \text{Class}(\bar{a}|\Gamma_f|\Gamma_m) \in \Gamma_c}{\Gamma_c \vdash a <: a_i} \quad \text{INTREFL} \quad \Gamma_c \vdash \text{Int} <: \text{Int} \quad \text{NONEREFL} \quad \Gamma_c \vdash \text{None} <: \text{None}$$

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

$$\frac{\text{IDENT} \quad a \mapsto \text{Var}(T) \in \Gamma_v}{\Gamma_c; \Gamma_v \vdash a : T}$$

$$\frac{\text{SELECT} \quad \Gamma_c; \Gamma_v \vdash t : b \quad b \mapsto \text{Class}(\bar{b}|\Gamma_f|\Gamma_m) \in \Gamma_c \quad a \mapsto \text{Field}(T) \in \Gamma_f}{\Gamma_c; \Gamma_v \vdash t.a : T}$$

$$\frac{\text{CALL} \quad \Gamma_c; \Gamma_v \vdash t : b \quad b \mapsto \text{Class}(\bar{b}|\Gamma_f|\Gamma_m) \in \Gamma_c \quad a \mapsto \text{Meth}(\bar{T}|T) \in \Gamma_m \quad \Gamma_c; \Gamma_v \vdash \bar{t} : \bar{U} \quad \Gamma_c \vdash \bar{U} <: \bar{T}}{\Gamma_c; \Gamma_v \vdash t.a(\bar{t}) : T}$$

NEW

$$\frac{\Gamma_f = \bar{a} \mapsto \text{Field}(T) \quad a \mapsto \text{Class}(\bar{b}|\Gamma_f|\Gamma_m) \in \Gamma_c \quad \Gamma_c; \Gamma_v \vdash \bar{t} : \bar{U} \quad \Gamma_c \vdash \bar{U} <: \bar{T}}{\Gamma_c; \Gamma_v \vdash \text{new } a(\bar{t}) : a} \quad \text{INTLIT} \quad \Gamma_c; \Gamma_v \vdash n : \text{Int}$$

UNOP

$$\frac{\Gamma_c; \Gamma_v \vdash t : \text{Int}}{\Gamma_c; \Gamma_v \vdash \text{unop } t : \text{Int}}$$

BINOP

$$\frac{\text{binop} \notin \{=, \neq\} \quad \Gamma_c; \Gamma_v \vdash t : \text{Int} \quad \Gamma_c; \Gamma_v \vdash u : \text{Int}}{\Gamma_c; \Gamma_v \vdash t \text{ binop } u : \text{Int}}$$

OBJCOMP

$$\frac{\text{binop} \in \{=, \neq\} \quad \Gamma_c; \Gamma_v \vdash t : T \quad \Gamma_c; \Gamma_v \vdash u : U \quad \Gamma_c \vdash T <: U \vee \Gamma_c \vdash U <: T}{\Gamma_c; \Gamma_v \vdash t \text{ binop } u : \text{Int}}$$

READINT

$$\Gamma_c; \Gamma_v \vdash \text{readInt} : \text{Int}$$

READCHAR

$$\Gamma_c; \Gamma_v \vdash \text{readChar} : \text{Int}$$

BLOCK

$$\frac{\Gamma_c; \Gamma_v \vdash \bar{S} \Rightarrow \Gamma'_v \quad \Gamma_c; \Gamma'_v \vdash t : T}{\Gamma_c; \Gamma_v \vdash \{ \bar{S} t \} : T}$$

EMPTY

$$\Gamma_c; \Gamma_v \vdash \text{empty} : \text{None}$$

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

$$\frac{\text{IF} \quad \Gamma_c; \Gamma_v \vdash t : \text{Int} \quad \Gamma_c; \Gamma_v \vdash S \Rightarrow \Gamma_v \quad \Gamma_c; \Gamma_v \vdash S' \Rightarrow \Gamma_v}{\Gamma_c; \Gamma_v \vdash \text{if } t \text{ then } S \text{ else } S' \Rightarrow \Gamma_v}$$

$$\frac{\text{WHILE} \quad \Gamma_c; \Gamma_v \vdash t : \text{Int} \quad \Gamma_c; \Gamma_v \vdash S \Rightarrow \Gamma_v}{\Gamma_c; \Gamma_v \vdash \text{while } t \text{ } S \Rightarrow \Gamma_v}$$

$$\frac{\text{VAR} \quad \Gamma_c \vdash T \diamond \quad \Gamma_c; \Gamma_v \vdash t : U \quad \Gamma_c \vdash U <: T \quad \Gamma'_v = \Gamma_v + a \mapsto \text{Var}(T)}{\Gamma_c; \Gamma_v \vdash \text{var } a : T = t \Rightarrow \Gamma'_v}$$

$$\frac{\text{SET} \quad a \mapsto \text{Var}(T) \in \Gamma_v \quad \Gamma_c; \Gamma_v \vdash t : U \quad \Gamma_c \vdash U <: T}{\Gamma_c; \Gamma_v \vdash \text{set } a = t \Rightarrow \Gamma_v}$$

$$\frac{\text{Do} \quad \Gamma_c; \Gamma_v \vdash t : T}{\Gamma_c; \Gamma_v \vdash \text{do } t \Rightarrow \Gamma_v}$$

$$\frac{\text{PRINTINT} \quad \Gamma_c; \Gamma_v \vdash t : \text{Int}}{\Gamma_c; \Gamma_v \vdash \text{printInt}(t) \Rightarrow \Gamma_v}$$

$$\frac{\text{PRINTCHAR} \quad \Gamma_c; \Gamma_v \vdash t : \text{Int}}{\Gamma_c; \Gamma_v \vdash \text{printChar}(t) \Rightarrow \Gamma_v}$$

$$\frac{\text{COMPOUND} \quad \Gamma_c; \Gamma_v \vdash \bar{S} \Rightarrow \Gamma'_v}{\Gamma_c; \Gamma_v \vdash \{ \bar{S} \} \Rightarrow \Gamma_v}$$