The Full OAT Language Type System

March 24, 2011

n	Constant int		
b	Constant bool		
$cstr$	Constant string		
id	Identifiers		
cid	Class identifiers		
$j,\ k,\ m$	Index		
typ	::=		Types
	|	`bot`	bottom
	|	`bool`	bool
	|	`int`	int
	|	*ref*	reference
	|	*ref* `?`	nullable
ref	::=		References
	|	`string`	string
	|	*cid*	class
	|	*typ* `[]`	array
$unop$	::=		Unary operators
	|	`-`	unary signed negation
	|	`!`	unary logical negation
	|	$\sim$	unary bitwise negation
$binop$	::=		Binary operators
	|	`+`	binary signed addition
	|	`*`	binary signed multiplication
	|	`-`	binary signed subtraction
	|	`==`	binary equality
	|	`!=`	binary inequality
	|	`<`	binary signed less-than
	|	`<=`	binary signed less-than or equals
	|	`>`	binary signed greater-than
	|	`>=`	binary signed greater-than or equals
	|	`&`	binary bool bitwise and
	|	|	binary bool bitwise or
	|	`[&]`	binary int bitwise and

| `[|]` binary int bitwise or
| `<<` binary shift left
| `>>` binary logical shift right
| `>>>` binary arithmetic shift right

$const$ ::= Constants
| `null` null
| b bool
| n int
| $cstr$ string

$path$ ::= Paths
| `this` . id identifiers in this class
| lhs_or_call . id path identifiers, e.g, a.b.f().c

$call$ ::= Calls
| id ($\overline{exp_j}^{\,j}$) global functions
| `super` . id ($\overline{exp_j}^{\,j}$) super methods
| $path$ ($\overline{exp_j}^{\,j}$) path methods, e.g. a.f().b.g()

lhs_or_call ::= Left-hand sides or calls
| lhs left-hand sides
| $call$ calls

lhs ::= Left-hand sides
| id variables
| $path$ paths
| lhs_or_call [exp] array index

exp ::= Expressions
| $const$ constant
| `this` this
| `new` [exp_1] (`fun` id `->` exp_2) new
| `new` cid ($\overline{exp_j}^{\,j}$) constructor
| lhs_or_call left-hand sides or calls
| $binop\ exp_1\ exp_2$ binarith
| $unop\ exp$ unarith

exp_{opt} ::= Optional expressions
| `None` none
| `Some` exp some

$init$ ::= Initializer
| exp exp
| $\{\ \overline{init_j}^{\,j\in 1..m}\ \}$ array

$vdecl$	$::=$		Variable declarations
	$\|$	$typ\ id$=$init$`;`	
$vdecls$	$::=$		A list of variable declarations
	$\|$	ϵ	nil
	$\|$	$vdecl\ vdecls$	cons
$stmt$	$::=$		Statements
	$\|$	lhs=exp`;`	assignments
	$\|$	$call$`;`	call
	$\|$	`fail` `(`exp`)` `;`	fail
	$\|$	`if` `(`exp`)` $stmt\ stmt_{opt}$	if
	$\|$	`if?` `(`$ref\ id$=exp`)` $stmt\ stmt_{opt}$	if null
	$\|$	`cast` `(`$cid\ id$=exp`)` $stmt\ stmt_{opt}$	cast
	$\|$	`while` `(`exp`)` $stmt$	while
	$\|$	`for` `(`$vdecls$`;` exp_{opt}`;` $stmt_{opt}$`)` $stmt$	for
	$\|$	$\{block\}$	block
$stmt_{opt}$	$::=$		Optional statements
	$\|$	`None`	none
	$\|$	`Some` $stmt$	some
$block$	$::=$		Blocks
	$\|$	$vdecls\ \overline{stmt_j}^{\,j}$	
$args$	$::=$		A list of arguments
	$\|$	ϵ	
	$\|$	$typ\ id$`,` $args$	
$rtyp$	$::=$		Return types
	$\|$	`unit`	unit
	$\|$	typ	types
$efdecl$	$::=$		External function declarations
	$\|$	$rtyp\ id$`(`$args$`)` `extern`	
$fdecl$	$::=$		Function declarations
	$\|$	$typ\ id$`(`$args$`)`$\{block$ `return` exp`;` $\}$	
	$\|$	`unit` id`(`$args$`)`$\{block$ `return` `;` $\}$	
$cinits$	$::=$		A list of field initialization
	$\|$	ϵ	nil
	$\|$	`this` `.`id=$init$`;` $cinits$	cons
$ctor$	$::=$		Constructors

	|	`new` `(` $args$ `)` `(` $\overline{exp_j}^{\,j}$ `)` $cinits\{block\}$	
cid_{ext}	::=		Optional extensions
	|	`None`	base
	|	`<:` cid	extension
$fields$	::=		A list of field declarations
	|	ϵ	nil
	|	$typ\ id$ `;` $fields$	cons
$fdecls$	::=		A list of function declarations
	|	ϵ	nil
	|	$fdecl\ fdecls$	cons
$cdecl$	::=		Classes
	|	`class` $cid\ cid_{ext}\{fields\ ctor\ fdecls\}$`;`	
$gdecl$	::=		Global declarations
	|	$vdecl$	constants
	|	$fdecl$	function declarations
	|	$efdecl$	external function declarations
	|	$cdecl$	class declarations
$prog$	::=		Programs
	|	ϵ	
	|	$gdecl\ prog$	
γ	::=		Variable contexts
	|	$\cdot$	empty
	|	$\gamma, id : typ$	cons
Γ	::=		A stack of variable contexts
	|	$\cdot$	empty
	|	Γ `;` γ	cons
$ftyp$	::=		Function types
	|	`(` $\overline{typ_j}^{\,j}$ `)` `->` $rtyp$	
$ptyp$	::=		Path types
	|	typ	
	|	$ftyp$	
Δ	::=		Function contexts
	|	$\cdot$	empty
	|	$\Delta, id : ftyp$	cons

Θ	$::=$		Method contexts
	$\vert$	$\cdot$	empty
	$\vert$	$\Theta, id : \mathit{ftyp}$	cons
Φ	$::=$		Field contexts
	$\vert$	$\cdot$	empty
	$\vert$	$\Phi, id : \mathit{typ}$	cons
Σ	$::=$		Class signatures
	$\vert$	$\cdot$	empty
	$\vert$	$\Sigma\texttt{,}\ \mathit{cid}\ \mathit{cid}_{ext}\{\Phi\texttt{;}\ \overline{\mathit{typ}_j}^{\,j}\ \texttt{;}\ \Theta\}$	cons
cid_{opt}	$::=$		Optional class
	$\vert$	$\texttt{Some}\ \mathit{cid}$	some in the scope of class cid
	$\vert$	$\texttt{None}$	none, otherwise
typ_{opt}	$::=$		Optional typ, the return of hasField
	$\vert$	$\texttt{None}$	none
	$\vert$	$\texttt{Some}\ \mathit{typ}$	some
ftyp_{opt}	$::=$		Optional ftyp, the return of hasMethod
	$\vert$	$\texttt{None}$	none
	$\vert$	$\texttt{Some}\ \mathit{ftyp}$	some

$\boxed{\Sigma \vdash \mathit{typ}}$ Σ shows that typ is well-formed.

$$\frac{}{\Sigma \vdash \texttt{bool}} \quad \text{TYP\_BOOL}$$

$$\frac{}{\Sigma \vdash \texttt{int}} \quad \text{TYP\_INT}$$

$$\frac{}{\Sigma \vdash \mathit{ref}} \quad \text{TYP\_REF}$$

$$\frac{\Sigma \vdash_r \mathit{ref}}{\Sigma \vdash \mathit{ref}\,?} \quad \text{TYP\_NULLABLE}$$

$\boxed{\Sigma \vdash_r \mathit{ref}}$ Σ shows that ref is well-formed.

$$\frac{}{\Sigma \vdash_r \texttt{string}} \quad \text{REF\_STRING}$$

$$\frac{\mathit{cid}\ \mathit{cid}_{ext}\{\Phi\texttt{;}\ \overline{\mathit{typ}_j}^{\,j}\ \texttt{;}\ \Theta\} \in \Sigma}{\Sigma \vdash_r \mathit{cid}} \quad \text{REF\_CLASS}$$

$$\frac{\Sigma \vdash \mathit{typ}}{\Sigma \vdash_r \mathit{typ}\,\texttt{[]}} \quad \text{REF\_ARRAY}$$

$\boxed{\Sigma \vdash \mathit{typ}_1 \texttt{<:} \mathit{typ}_2}$ Σ shows that typ_1 is a subtype of typ_2.

$$\frac{}{\Sigma \vdash \texttt{bool<:bool}} \quad \text{ST\_BOOL}$$

$$\frac{}{\Sigma \vdash \texttt{int<:int}} \quad \text{ST\_INT}$$

$$\frac{\Sigma \vdash_r \mathit{ref}_1 \texttt{<:} \mathit{ref}_2}{\Sigma \vdash \mathit{ref}_1 \texttt{<:} \mathit{ref}_2} \quad \text{ST\_REF}$$

$$\frac{\Sigma \vdash_r \mathit{ref}_1 \texttt{<:} \mathit{ref}_2}{\Sigma \vdash \mathit{ref}_1\texttt{?} \texttt{<:} \mathit{ref}_2\texttt{?}} \quad \text{ST\_NULLABLE}$$

$$\frac{\Sigma \vdash_r \mathit{ref}_1 \texttt{<:} \mathit{ref}_2}{\Sigma \vdash \mathit{ref}_1 \texttt{<:} \mathit{ref}_2\texttt{?}} \quad \text{ST\_REF\_NULLABLE}$$

$$\frac{}{\Sigma \vdash \texttt{bot} \texttt{<:} \mathit{ref}\,\texttt{?}} \quad \text{ST\_NULL\_NULLABLE}$$

$\boxed{\Sigma \vdash_r \mathit{ref}_1 \texttt{<:} \mathit{ref}_2}$ Σ shows that ref_1 is a sub-reference of ref_2.

$$\frac{}{\Sigma \vdash_r \texttt{string<:string}} \quad \text{SR\_STRING}$$

$$\frac{\Sigma \vdash_c \mathit{cid}_1 \texttt{<:} \mathit{cid}_2}{\Sigma \vdash_r \mathit{cid}_1 \texttt{<:} \mathit{cid}_2} \quad \text{SR\_CLASS}$$

$$\frac{}{\Sigma \vdash_r \mathit{typ}\,\texttt{[]} \texttt{<:} \mathit{typ}\,\texttt{[]}} \quad \text{SR\_ARRAY}$$

$\boxed{\Sigma \vdash_c \mathit{cid}_1 \texttt{<:} \mathit{cid}_2}$ Σ shows that cid_1 is a sub-class of cid_2.

$$\frac{\mathit{cid}\ \mathit{cid}_{\mathit{ext}}\{\Phi\texttt{;}\ \overline{\mathit{typ}_j}^{\,j}\ \texttt{;}\ \Theta\} \in \Sigma}{\Sigma \vdash_c \mathit{cid} \texttt{<:} \mathit{cid}} \quad \text{SC\_REF}$$

$$\frac{}{\Sigma_1\texttt{,}\ \mathit{cid}_1 \texttt{<:} \mathit{cid}_2\{\Phi\texttt{;}\ \overline{\mathit{typ}_j}^{\,j}\ \texttt{;}\ \Theta\}\texttt{,}\Sigma_2 \vdash_c \mathit{cid}_1 \texttt{<:} \mathit{cid}_2} \quad \text{SC\_INHERITANCE}$$

$$\frac{\Sigma \vdash_c \mathit{cid}_1 \texttt{<:} \mathit{cid}_2 \quad \Sigma \vdash_c \mathit{cid}_2 \texttt{<:} \mathit{cid}_3}{\Sigma \vdash_c \mathit{cid}_1 \texttt{<:} \mathit{cid}_3} \quad \text{SC\_TRANS}$$

$\boxed{\texttt{hasField}\,\Sigma\,cid\,\texttt{.}\,id\texttt{=}typ_{opt}}$ Check if cid has a field id.

$$\frac{cid\ cid_{ext}\{\Phi\texttt{;}\ \overline{typ_j}^{\,j}\ \texttt{;}\ \Theta\} \in \Sigma \quad id\,\texttt{:}\,typ \in \Phi}{\texttt{hasField}\,\Sigma\,cid\,\texttt{.}\,id\texttt{=Some}\ typ} \quad \text{HASFIELD\_BASE\_SOME}$$

$$\frac{cid\ \texttt{None}\{\Phi\texttt{;}\ \overline{typ_j}^{\,j}\ \texttt{;}\ \Theta\} \in \Sigma \quad id \notin \Phi}{\texttt{hasField}\,\Sigma\,cid\,\texttt{.}\,id\texttt{=None}} \quad \text{HASFIELD\_BASE\_NONE}$$

$$\frac{cid_1\ \texttt{<:}\ cid_2\{\Phi\texttt{;}\ \overline{typ_j}^{\,j}\ \texttt{;}\ \Theta\} \in \Sigma \quad id \notin \Phi \quad \texttt{hasField}\,\Sigma\,cid_2\,\texttt{.}\,id\texttt{=}typ_{opt}}{\texttt{hasField}\,\Sigma\,cid_1\,\texttt{.}\,id\texttt{=}typ_{opt}} \quad \text{HASFIELD\_INHERITANCE}$$

$\boxed{\texttt{hasMethod}\,\Sigma\,cid\,\texttt{.}\,id\texttt{=}ftyp_{opt}}$ Check if cid has a method id.

$$\frac{cid\ cid_{ext}\{\Phi\texttt{;}\ \overline{typ_j}^{\,j}\ \texttt{;}\ \Theta\} \in \Sigma \quad id\,\texttt{:}\,ftyp \in \Theta}{\texttt{hasMethod}\,\Sigma\,cid\,\texttt{.}\,id\texttt{=Some}\ ftyp} \quad \text{HASMETHOD\_BASE\_SOME}$$

$$\frac{cid\ \texttt{None}\{\Phi\texttt{;}\ \overline{typ_j}^{\,j}\ \texttt{;}\ \Theta\} \in \Sigma \quad id \notin \Theta}{\texttt{hasMethod}\,\Sigma\,cid\,\texttt{.}\,id\texttt{=None}} \quad \text{HASMETHOD\_BASE\_NONE}$$

$$\frac{cid_1\ \texttt{<:}\ cid_2\{\Phi\texttt{;}\ \overline{typ_j}^{\,j}\ \texttt{;}\ \Theta\} \in \Sigma \quad id \notin \Theta \quad \texttt{hasMethod}\,\Sigma\,cid_2\,\texttt{.}\,id\texttt{=}ftyp_{opt}}{\texttt{hasMethod}\,\Sigma\,cid_1\,\texttt{.}\,id\texttt{=}ftyp_{opt}} \quad \text{HASMETHOD\_INHERITANCE}$$

$\boxed{\vdash const\,\texttt{:}\,typ}$ $const$ has type typ.

$$\frac{}{\vdash \texttt{null:bot}} \quad \text{CONST\_BOT}$$

$$\frac{}{\vdash b\,\texttt{:bool}} \quad \text{CONST\_BOOL}$$

$$\frac{}{\vdash n\,\texttt{:int}} \quad \text{CONST\_INT}$$

$$\frac{}{\vdash cstr\,\texttt{:string}} \quad \text{CONST\_STRING}$$

$\boxed{binop\,\texttt{:}\,ftyp}$ $binop$ is of type $ftyp$.

$$\frac{}{\texttt{+:(int int)->int}} \quad \text{BINTYP\_PLUS}$$

$$\frac{}{\texttt{*:(int int)->int}} \quad \text{BINTYP\_TIMES}$$

$$\frac{}{\texttt{-:(int int)->int}} \quad \text{BINTYP\_MINUS}$$

$$\frac{}{\texttt{==:(}\mathit{typ}\ \mathit{typ}\texttt{)->bool}} \quad \text{BINTYP\_EQ}$$

$$\frac{}{\texttt{!=:(}\mathit{typ}\ \mathit{typ}\texttt{)->bool}} \quad \text{BINTYP\_NEQ}$$

$$\frac{}{\texttt{<:(int int)->bool}} \quad \text{BINTYP\_LT}$$

$$\frac{}{\texttt{<=:(int int)->bool}} \quad \text{BINTYP\_LTE}$$

$$\frac{}{\texttt{>:(int int)->bool}} \quad \text{BINTYP\_GE}$$

$$\frac{}{\texttt{>=:(int int)->bool}} \quad \text{BINTYP\_GTE}$$

$$\frac{}{\texttt{[\&]:(int int)->int}} \quad \text{BINTYP\_IAND}$$

$$\frac{}{\texttt{\&:(bool bool)->bool}} \quad \text{BINTYP\_AND}$$

$$\frac{}{\texttt{[|]:(int int)->int}} \quad \text{BINTYP\_IOR}$$

$$\frac{}{\texttt{|:(bool bool)->bool}} \quad \text{BINTYP\_OR}$$

$$\frac{}{\texttt{<<:(int int)->int}} \quad \text{BINTYP\_SHL}$$

$$\frac{}{\texttt{>>:(int int)->int}} \quad \text{BINTYP\_SHR}$$

$$\frac{}{\texttt{>>>:(int int)->int}} \quad \text{BINTYP\_SAR}$$

$\boxed{\mathit{unop} : \mathit{ftyp}}$ unop is of type ftyp.

$$\frac{}{\texttt{-:(int)->int}}\ \textsc{utyp\_neg}$$

$$\frac{}{\texttt{!:(bool)->bool}}\ \textsc{utyp\_lognot}$$

$$\frac{}{\sim\texttt{:(int)->int}}\ \textsc{utyp\_not}$$

$\boxed{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash_p path : ptyp}$ Σ, Δ, Γ and cid_{opt} show that $path$ has type $ptyp$.

$$\frac{\texttt{hasField}\,\Sigma\,cid\,\texttt{.}\,id\texttt{=Some}\,typ \quad \texttt{hasMethod}\,\Sigma\,cid\,\texttt{.}\,id\texttt{=None}}{\Sigma;\,\Delta;\,\Gamma;\,\texttt{Some}\,cid \vdash_p \texttt{this}\,\texttt{.}\,id : typ}\ \textsc{p\_this\_field}$$

$$\frac{\texttt{hasMethod}\,\Sigma\,cid\,\texttt{.}\,id\texttt{=Some}\,ftyp \quad \texttt{hasField}\,\Sigma\,cid\,\texttt{.}\,id\texttt{=None}}{\Sigma;\,\Delta;\,\Gamma;\,\texttt{Some}\,cid \vdash_p \texttt{this}\,\texttt{.}\,id : ftyp}\ \textsc{p\_this\_method}$$

$$\frac{\begin{array}{l}\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash_l lhs\_or\_call : cid\\ \texttt{hasField}\,\Sigma\,cid\,\texttt{.}\,id\texttt{=Some}\,typ \quad \texttt{hasMethod}\,\Sigma\,cid\,\texttt{.}\,id\texttt{=None}\end{array}}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash_p lhs\_or\_call\,\texttt{.}\,id : typ}\ \textsc{p\_path\_field}$$

$$\frac{\begin{array}{l}\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash_l lhs\_or\_call : cid\\ \texttt{hasMethod}\,\Sigma\,cid\,\texttt{.}\,id\texttt{=Some}\,ftyp \quad \texttt{hasField}\,\Sigma\,cid\,\texttt{.}\,id\texttt{=None}\end{array}}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash_p lhs\_or\_call\,\texttt{.}\,id : ftyp}\ \textsc{p\_path\_method}$$

$\boxed{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash call : rtyp}$ Σ, Δ, Γ and cid_{opt} show that $call$ has type $rtyp$.

$$\frac{id\,\texttt{:}\,\texttt{(}\,\overline{typ_j}^{\,j}\,\texttt{)->}rtyp \in \Delta \quad \overline{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash exp_j \texttt{<:} typ_j}^{\,j}}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash id\,\texttt{(}\,\overline{exp_j}^{\,j}\,\texttt{)}\,\texttt{:}\,rtyp}\ \textsc{call\_func}$$

$$\frac{id \notin \Delta \quad id\,\texttt{:}\,\texttt{(}\,\overline{typ_j}^{\,j}\,\texttt{)->}rtyp \quad \overline{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash exp_j \texttt{<:} typ_j}^{\,j}}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash id\,\texttt{(}\,\overline{exp_j}^{\,j}\,\texttt{)}\,\texttt{:}\,rtyp}\ \textsc{call\_builtin}$$

$$\frac{\begin{array}{l}cid_1 \texttt{<:} cid_2\{\Phi;\ \overline{typ'_k}^{\,k}\ \texttt{;}\ \Theta\} \in \Sigma \quad \texttt{hasMethod}\,\Sigma\,cid_2\,\texttt{.}\,id\texttt{=Some}\ \texttt{(}\,\overline{typ_j}^{\,j}\,\texttt{)->}rtyp\\ \overline{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash exp_j \texttt{<:} typ_j}^{\,j}\end{array}}{\Sigma;\,\Delta;\,\Gamma;\,\texttt{Some}\,cid_1 \vdash \texttt{super}\,\texttt{.}\,id\,\texttt{(}\,\overline{exp_j}^{\,j}\,\texttt{)}\,\texttt{:}\,rtyp}\ \textsc{call\_super\_method}$$

$$\frac{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash_p path\,\texttt{:}\,\texttt{(}\,\overline{typ_j}^{\,j}\,\texttt{)->}rtyp \quad \overline{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash exp_j \texttt{<:} typ_j}^{\,j}}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash path\,\texttt{(}\,\overline{exp_j}^{\,j}\,\texttt{)}\,\texttt{:}\,rtyp}\ \textsc{call\_path\_method}$$

$\boxed{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash_l lhs\_or\_call : typ}$ Σ, Δ, Γ and cid_{opt} show that lhs_or_call has type typ.

$$\frac{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash lhs : typ}{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash_l lhs : typ} \quad \text{LC\_LHS}$$

$$\frac{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash call : typ}{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash_l call : typ} \quad \text{LC\_CALL}$$

$\boxed{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash lhs : typ}$ Σ, Δ, Γ and cid_{opt} show that lhs has type typ.

$$\frac{id : typ \in \Gamma}{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash id : typ} \quad \text{LHS\_VAR}$$

$$\frac{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash_p path : typ}{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash path : typ} \quad \text{LHS\_PATH}$$

$$\frac{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash_l lhs\_or\_call : typ\,\texttt{[]} \quad \Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash exp : \texttt{int}}{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash lhs\_or\_call\,\texttt{[}\,exp\,\texttt{]} : typ} \quad \text{LHS\_INDEX}$$

$\boxed{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash exp : typ}$ Σ, Δ, Γ and cid_{opt} show that exp has type typ.

$$\frac{\vdash const : typ}{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash const : typ} \quad \text{EXP\_CONST}$$

$$\frac{}{\Sigma;\ \Delta;\ \Gamma;\ \texttt{Some}\ cid \vdash \texttt{this} : cid} \quad \text{EXP\_THIS}$$

$$\frac{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash exp_1 : \texttt{int} \quad \Sigma;\ \Delta;\ \texttt{(}\Gamma;\ \texttt{(}id : \texttt{int}\texttt{))};\ cid_{opt} \vdash exp_2 : typ}{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash \texttt{new}\,\texttt{[}exp_1\texttt{]}\,\texttt{(}\,\texttt{fun}\ id\texttt{->}exp_2\texttt{)} : typ\,\texttt{[]}} \quad \text{EXP\_NEW}$$

$$\frac{cid\ cid_{ext}\{\Phi;\ \overline{typ_j}^{\,j}\ ;\ \Theta\} \in \Sigma \quad \overline{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash exp_j \texttt{<:}\ typ_j}^{\,j}}{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash \texttt{new}\ cid\ \texttt{(}\ \overline{exp_j}^{\,j}\ \texttt{)} : cid} \quad \text{EXP\_CTOR}$$

$$\frac{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash_l lhs\_or\_call : typ}{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash lhs\_or\_call : typ} \quad \text{EXP\_LHS\_OR\_CALL}$$

$$\frac{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash exp_1 \texttt{<:}\ typ_1 \quad \Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash exp_2 \texttt{<:}\ typ_2 \quad binop : \texttt{(}typ_1\ typ_2\texttt{)}\ \texttt{->}\ typ}{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash binop\ exp_1\ exp_2 : typ} \quad \text{EXP\_BINARITH}$$

$$\frac{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash exp \texttt{<:}\ typ_1 \quad unop : \texttt{(}typ_1\texttt{)}\ \texttt{->}\ typ}{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash unop\ exp : typ} \quad \text{EXP\_UNARITH}$$

$\boxed{\Sigma;\ \Delta;\ \Gamma;\ cid_{opt} \vdash exp_{opt} : typ}$ Σ, Δ, Γ and cid_{opt} show that exp_{opt} is well-formed.

$$\frac{}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash \texttt{None} : \texttt{bool}} \quad \text{OPT\_EXP\_NONE}$$

$$\frac{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash exp : \texttt{bool}}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash \texttt{Some}\ exp : \texttt{bool}} \quad \text{OPT\_EXP\_SOME}$$

$\boxed{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash exp <: typ}$ Σ, Δ, Γ and cid_{opt} show that exp has a subtype of typ.

$$\frac{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash exp : typ' \quad \Sigma \vdash typ' <: typ}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash exp <: typ} \quad \text{EXPSUB\_INTRO}$$

$\boxed{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash_i init : typ}$ Σ, Δ, Γ and cid_{opt} show that $init$ has type typ.

$$\frac{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash exp : typ}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash_i exp : typ} \quad \text{INIT\_EXP}$$

$$\frac{\overline{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash_i init_j : typ_j}^{\,j\in 1..m} \qquad \vee\ \overline{typ_j}^{\,j\in 1..m} = typ}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash_i \{\,\overline{init_j}^{\,j\in 1..m}\,\} : typ\,\texttt{[]}} \quad \text{INIT\_ARRAY}$$

$\boxed{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash init <: typ}$ Σ, Δ, Γ and cid_{opt} show that $init$ has a subtype of typ.

$$\frac{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash_i init : typ' \quad \Sigma \vdash typ' <: typ}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash init <: typ} \quad \text{SINIT\_INTRO}$$

$\boxed{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash vdecls : \Gamma'}$ $vdecls$ are well-formed under Σ, Δ, Γ and cid_{opt}, and extend the context to be Γ'.

$$\frac{}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash \epsilon : \Gamma} \quad \text{VDECLS\_NIL}$$

$$\frac{\begin{array}{l}\Sigma;\,\Delta;\,(\Gamma;\,\gamma);\,cid_{opt} \vdash init <: typ \quad \Sigma \vdash typ \\ id \notin \Delta\ \texttt{and}\ \gamma \quad \Sigma;\,\Delta;\,(\Gamma;\,(\gamma, id : typ));\,cid_{opt} \vdash vdecls : \Gamma'\end{array}}{\Sigma;\,\Delta;\,(\Gamma;\,\gamma);\,cid_{opt} \vdash typ\ id{=}init;\ vdecls : \Gamma'} \quad \text{VDECLS\_CONS}$$

$\boxed{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash stmt : \texttt{ok}}$ Σ, Δ, Γ and cid_{opt} show that $stmt$ is well-formed.

$$\frac{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash lhs : typ \quad \Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash exp : typ}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash lhs{=}exp; : \texttt{ok}} \quad \text{STMT\_ASSIGN}$$

$$\frac{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash call : \texttt{unit}}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash call; : \texttt{ok}} \quad \text{STMT\_CALL}$$

$$\frac{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash exp : \texttt{string}}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt} \vdash \texttt{fail}\,(exp); : \texttt{ok}} \quad \text{STMT\_FAIL}$$

$$\frac{\begin{array}{c}\Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash exp:\texttt{bool}\\ \Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash stmt:\texttt{ok}\quad \Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash stmt_{opt}:\texttt{ok}\end{array}}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash \texttt{if (}exp\texttt{)}\, stmt\; stmt_{opt}:\texttt{ok}}\quad\text{STMT\_IF}$$

$$\frac{\begin{array}{c}\Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash exp:ref\texttt{?}\\ \Sigma;\,\Delta;\,\texttt{(}\Gamma;\,\texttt{(}id:ref\texttt{))};\,cid_{opt}\vdash stmt:\texttt{ok}\quad \Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash stmt_{opt}:\texttt{ok}\end{array}}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash \texttt{if?(}ref\; id{=}exp\texttt{)}\, stmt\; stmt_{opt}:\texttt{ok}}\quad\text{STMT\_IFNULL}$$

$$\frac{\begin{array}{c}\Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash exp:cid'\quad \Sigma\vdash cid\texttt{<:}cid'\\ \Sigma;\,\Delta;\,\texttt{(}\Gamma;\,\texttt{(}id:cid\texttt{))};\,cid_{opt}\vdash stmt:\texttt{ok}\quad \Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash stmt_{opt}:\texttt{ok}\end{array}}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash \texttt{cast (}cid\; id{=}exp\texttt{)}\, stmt\; stmt_{opt}:\texttt{ok}}\quad\text{STMT\_CAST}$$

$$\frac{\begin{array}{c}\Sigma;\,\Delta;\,\texttt{(}\Gamma;\,\cdot\texttt{)};\,cid_{opt}\vdash vdecls:\Gamma'\quad \Sigma;\,\Delta;\,\Gamma';\,cid_{opt}\vdash exp_{opt}:\texttt{bool}\\ \Sigma;\,\Delta;\,\Gamma';\,cid_{opt}\vdash stmt_{opt}:\texttt{ok}\quad \Sigma;\,\Delta;\,\Gamma';\,cid_{opt}\vdash stmt:\texttt{ok}\end{array}}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash \texttt{for (}vdecls\texttt{;}\; exp_{opt}\texttt{;}\; stmt_{opt}\texttt{)}\, stmt:\texttt{ok}}\quad\text{STMT\_FOR}$$

$$\frac{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash exp:\texttt{bool}\quad \Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash stmt:\texttt{ok}}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash \texttt{while (}exp\texttt{)}\, stmt:\texttt{ok}}\quad\text{STMT\_WHILE}$$

$$\frac{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash block:\texttt{ok}}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash \{block\}:\texttt{ok}}\quad\text{STMT\_BLOCK}$$

$\boxed{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash block:\texttt{ok}}$ Σ, Δ, Γ and cid_{opt} show that *block* is well-formed.

$$\frac{\Sigma;\,\Delta;\,\texttt{(}\Gamma;\,\cdot\texttt{)};\,cid_{opt}\vdash vdecls:\Gamma'\quad \overline{\Sigma;\,\Delta;\,\Gamma';\,cid_{opt}\vdash stmt_j:\texttt{ok}}^{\,j}}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash vdecls\;\overline{stmt_j}^{\,j}:\texttt{ok}}\quad\text{BLOCK\_INTRO}$$

$\boxed{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash stmt_{opt}:\texttt{ok}}$ Σ, Δ, Γ and cid_{opt} show that `op_stmt` is well-formed.

$$\frac{}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash \texttt{None}:\texttt{ok}}\quad\text{OPT\_STMT\_NONE}$$

$$\frac{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash stmt:\texttt{ok}}{\Sigma;\,\Delta;\,\Gamma;\,cid_{opt}\vdash \texttt{Some}\; stmt:\texttt{ok}}\quad\text{OPT\_STMT\_SOME}$$

$\boxed{\Sigma;\,\Delta;\,\Gamma\vdash args:\Gamma'}$ *args* are well-formed under Σ, Δ and Γ, and extend the context to be Γ'.

$$\frac{}{\Sigma;\,\Delta;\,\Gamma\vdash \epsilon:\Gamma}\quad\text{ARGS\_NIL}$$

$$\frac{id\notin\Delta\,\texttt{and}\,\gamma\quad \Sigma\vdash typ\quad \Sigma;\,\Delta;\,\texttt{(}\Gamma;\,\gamma,id:typ\texttt{)}\vdash args:\Gamma'}{\Sigma;\,\Delta;\,\texttt{(}\Gamma;\,\gamma\texttt{)}\vdash typ\; id\texttt{,}\; args:\Gamma'}\quad\text{ARGS\_CONS}$$

$\boxed{\Sigma\texttt{;}\ \Delta\texttt{;}\ \Gamma\texttt{;}\ cid_{opt} \vdash fdecl\texttt{:}\ \texttt{ok}}$ Σ, Δ, Γ and cid_{opt} show that $fdecl$ is well-formed.

$$\frac{\begin{array}{c}\Sigma\texttt{;}\ \Delta\texttt{;}\ \texttt{(}\Gamma\texttt{;}\ \cdot\texttt{)} \vdash args\texttt{:}\Gamma' \quad \Sigma\texttt{;}\ \Delta\texttt{;}\ \texttt{(}\Gamma'\texttt{;}\ \cdot\texttt{)}\texttt{;}\ cid_{opt} \vdash vdecls\texttt{:}\Gamma'' \\ \overline{\Sigma\texttt{;}\ \Delta\texttt{;}\ \Gamma''\texttt{;}\ cid_{opt} \vdash stmt_j\texttt{:}\ \texttt{ok}}^{j} \quad \Sigma\texttt{;}\ \Delta\texttt{;}\ \Gamma''\texttt{;}\ cid_{opt} \vdash exp\texttt{<:}\,typ \quad \Sigma \vdash typ\end{array}}{\Sigma\texttt{;}\ \Delta\texttt{;}\ \Gamma\texttt{;}\ cid_{opt} \vdash typ\ id\,\texttt{(}args\texttt{)}\,\{vdecls\ \overline{stmt_j}^{j}\ \texttt{return}\ exp\texttt{;}\ \}\texttt{:}\ \texttt{ok}}\ \textsc{fdecl\_func}$$

$$\frac{\begin{array}{c}\Sigma\texttt{;}\ \Delta\texttt{;}\ \texttt{(}\Gamma\texttt{;}\ \cdot\texttt{)} \vdash args\texttt{:}\Gamma' \quad \Sigma\texttt{;}\ \Delta\texttt{;}\ \texttt{(}\Gamma'\texttt{;}\ \cdot\texttt{)}\texttt{;}\ cid_{opt} \vdash vdecls\texttt{:}\Gamma'' \\ \overline{\Sigma\texttt{;}\ \Delta\texttt{;}\ \Gamma''\texttt{;}\ cid_{opt} \vdash stmt_j\texttt{:}\ \texttt{ok}}^{j}\end{array}}{\Sigma\texttt{;}\ \Delta\texttt{;}\ \Gamma\texttt{;}\ cid_{opt} \vdash \texttt{unit}\ id\,\texttt{(}args\texttt{)}\,\{vdecls\ \overline{stmt_j}^{j}\ \texttt{return}\ \texttt{;}\ \}\texttt{:}\ \texttt{ok}}\ \textsc{fdecl\_proc}$$

$\boxed{\Sigma \vdash id\texttt{:}ftyp\ \texttt{can override}\ cid_{ext}}$ Σ shows that id with type $ftyp$ can override parent class cid_{ext}.

$$\frac{}{\Sigma \vdash id\texttt{:}ftyp\ \texttt{can override None}}\ \textsc{or\_object}$$

$$\frac{\texttt{hasMethod}\ \Sigma\ cid\texttt{.}id\texttt{=None}}{\Sigma \vdash id\texttt{:}ftyp\ \texttt{can override <:}cid}\ \textsc{or\_nomethod}$$

$$\frac{\texttt{hasMethod}\ \Sigma\ cid\texttt{.}id\texttt{=Some (}\ \overline{typ'_j}^{j}\ \texttt{)->}typ' \quad \overline{\Sigma \vdash typ'_j\texttt{<:}typ_j}^{j} \quad \Sigma \vdash typ\texttt{<:}typ'}{\Sigma \vdash id\texttt{: (}\ \overline{typ_j}^{j}\ \texttt{)->}typ\ \texttt{can override <:}cid}\ \textsc{or\_func}$$

$$\frac{\texttt{hasMethod}\ \Sigma\ cid\texttt{.}id\texttt{=Some (}\ \overline{typ'_j}^{j}\ \texttt{)->unit} \quad \overline{\Sigma \vdash typ'_j\texttt{<:}typ_j}^{j}}{\Sigma \vdash id\texttt{: (}\ \overline{typ_j}^{j}\ \texttt{)->unit can override <:}cid}\ \textsc{or\_proc}$$

$\boxed{cid_{ext}\texttt{;}\ \Phi \vdash fields\texttt{:}\Phi'}$ Extending Φ to be Φ' by adding field declarations with parent class cid_{ext}.

$$\frac{}{cid_{ext}\texttt{;}\ \Phi \vdash \epsilon\texttt{:}\Phi}\ \textsc{genF\_nil}$$

$$\frac{id \notin \Phi \quad \texttt{None;}\ \Phi, id\texttt{:}\ typ \vdash fields\texttt{:}\Phi'}{\texttt{None;}\ \Phi \vdash typ\ id\texttt{;}\ fields\texttt{:}\Phi'}\ \textsc{genF\_base}$$

$$\frac{id \notin \Phi \quad \texttt{hasField}\ \Sigma\ cid\texttt{.}id\texttt{=None} \quad \texttt{<:}cid\texttt{;}\ \Phi, id\texttt{:}\ typ \vdash fields\texttt{:}\Phi'}{\texttt{<:}cid\texttt{;}\ \Phi \vdash typ\ id\texttt{;}\ fields\texttt{:}\Phi'}\ \textsc{genF\_inheritance}$$

$\boxed{\Sigma\texttt{;}\ cid_{ext}\texttt{;}\ \Phi\texttt{;}\ \Theta \vdash fdecls\texttt{:}\Theta'}$ Extending Θ to be Θ' by adding method declaratons with parent class cid_{ext}.

$$\frac{}{\Sigma\texttt{;}\ cid_{ext}\texttt{;}\ \Phi\texttt{;}\ \Theta \vdash \epsilon\texttt{:}\Theta}\ \textsc{genM\_nil}$$

$$\frac{id \notin \Phi \,\texttt{and}\, \Theta \quad \Sigma\texttt{;}\ cid_{ext}\texttt{;}\ \Phi\texttt{;}\ \Theta, id\texttt{:}\ \texttt{(}\ \overline{typ_j}^{\,j}\ \texttt{)->}typ \vdash fdecls\texttt{:}\Theta'}{\Sigma\texttt{;}\ cid_{ext}\texttt{;}\ \Phi\texttt{;}\ \Theta \vdash typ\ id\ \texttt{(}\overline{typ_j\ id_j}^{\,j}\texttt{)}\ \texttt{\{}vdecls\ \overline{stmt_k}^{\,k}\ \texttt{return}\ exp\texttt{;}\ \texttt{\}}\ fdecls\texttt{:}\Theta'} \quad \textsc{genM\_typ}$$

$$\frac{id \notin \Phi \,\texttt{and}\, \Theta \quad \Sigma\texttt{;}\ cid_{ext}\texttt{;}\ \Phi\texttt{;}\ \Theta, id\texttt{:}\ \texttt{(}\ \overline{typ_j}^{\,j}\ \texttt{)->unit} \vdash fdecls\texttt{:}\Theta'}{\Sigma\texttt{;}\ cid_{ext}\texttt{;}\ \Phi\texttt{;}\ \Theta \vdash \texttt{unit}\ id\ \texttt{(}\overline{typ_j\ id_j}^{\,j}\texttt{)}\ \texttt{\{}vdecls\ \overline{stmt_k}^{\,k}\ \texttt{return}\texttt{;}\ \texttt{\}}\ fdecls\texttt{:}\Theta'} \quad \textsc{genM\_unit}$$

$\boxed{\Sigma \vdash fields\texttt{:}\ \texttt{ok}}$ Σ shows that *fields* is well-formed.

$$\frac{}{\Sigma \vdash \epsilon\texttt{:}\ \texttt{ok}} \quad \textsc{wfF\_nil}$$

$$\frac{\Sigma \vdash typ \quad \Sigma \vdash fields\texttt{:}\ \texttt{ok}}{\Sigma \vdash typ\ id\texttt{;}\ fields\texttt{:}\ \texttt{ok}} \quad \textsc{wfF\_cons}$$

$\boxed{\Sigma\texttt{;}\ \Delta\texttt{;}\ \Gamma\texttt{;}\ cid \vdash fdecl\texttt{:}\ \texttt{ok}}$ A method *fdecl* of class *cid* is well-formed.

$$\frac{\begin{array}{l}\Sigma\texttt{;}\ \Delta\texttt{;}\ \Gamma\texttt{;}\ \texttt{Some}\ cid \vdash typ\ id\ \texttt{(}\overline{typ_j\ id_j}^{\,j}\texttt{)}\ \texttt{\{}vdecls\ \overline{stmt_k}^{\,k}\ \texttt{return}\ exp\texttt{;}\ \texttt{\}}\texttt{:}\ \texttt{ok}\\ cid\ cid_{ext}\ \in \Sigma \quad \Sigma \vdash id\texttt{:}\ \texttt{(}\ \overline{typ_j}^{\,j}\ \texttt{)->}typ\ \texttt{can override}\ cid_{ext}\end{array}}{\Sigma\texttt{;}\ \Delta\texttt{;}\ \Gamma\texttt{;}\ cid \vdash typ\ id\ \texttt{(}\overline{typ_j\ id_j}^{\,j}\texttt{)}\ \texttt{\{}vdecls\ \overline{stmt_k}^{\,k}\ \texttt{return}\ exp\texttt{;}\ \texttt{\}}\texttt{:}\ \texttt{ok}} \quad \textsc{wfM\_typ}$$

$$\frac{\begin{array}{l}\Sigma\texttt{;}\ \Delta\texttt{;}\ \Gamma\texttt{;}\ \texttt{Some}\ cid \vdash \texttt{unit}\ id\ \texttt{(}\overline{typ_j\ id_j}^{\,j}\texttt{)}\ \texttt{\{}vdecls\ \overline{stmt_k}^{\,k}\ \texttt{return}\texttt{;}\ \texttt{\}}\texttt{:}\ \texttt{ok}\\ cid\ cid_{ext}\ \in \Sigma \quad \Sigma \vdash id\texttt{:}\ \texttt{(}\ \overline{typ_j}^{\,j}\ \texttt{)->unit}\ \texttt{can override}\ cid_{ext}\end{array}}{\Sigma\texttt{;}\ \Delta\texttt{;}\ \Gamma\texttt{;}\ cid \vdash \texttt{unit}\ id\ \texttt{(}\overline{typ_j\ id_j}^{\,j}\texttt{)}\ \texttt{\{}vdecls\ \overline{stmt_k}^{\,k}\ \texttt{return}\texttt{;}\ \texttt{\}}\texttt{:}\ \texttt{ok}} \quad \textsc{wfM\_unit}$$

$\boxed{\Sigma\texttt{;}\ \Delta\texttt{;}\ \Gamma\texttt{;}\ cid \vdash cinits\texttt{:}\ \texttt{ok}}$ Σ, Δ and Γ show that *cinits* is well-formed.

$$\frac{}{\Sigma\texttt{;}\ \Delta\texttt{;}\ \Gamma\texttt{;}\ cid \vdash \epsilon\texttt{:}\ \texttt{ok}} \quad \textsc{cinits\_nil}$$

$$\frac{\begin{array}{l}cid\ cid_{ext}\texttt{\{}\Phi\texttt{;}\ \overline{typ_j}^{\,j}\ \texttt{;}\ \Theta\texttt{\}} \in \Sigma \quad id\texttt{:}typ \in \Phi\\ \Sigma\texttt{;}\ \Delta\texttt{;}\ \Gamma\texttt{;}\ \texttt{None} \vdash init\texttt{<:}typ \quad \Sigma\texttt{;}\ \Delta\texttt{;}\ \Gamma\texttt{;}\ cid \vdash cinits\texttt{:}\ \texttt{ok}\end{array}}{\Sigma\texttt{;}\ \Delta\texttt{;}\ \Gamma\texttt{;}\ cid \vdash \texttt{this}\ \texttt{.}id\texttt{=}init\texttt{;}\ cinits\texttt{:}\ \texttt{ok}} \quad \textsc{cinits\_cons}$$

$\boxed{\Sigma\texttt{;}\ \Delta\texttt{;}\ \Gamma\texttt{;}\ cid \vdash ctor\texttt{:}\ \texttt{ok}}$ *ctor* is well-formed.

$$\frac{\begin{array}{l}\Sigma\texttt{;}\ \Delta\texttt{;}\ \texttt{(}\Gamma\texttt{;}\ \cdot\ \texttt{)} \vdash \overline{typ_j\ id_j}^{\,j}\texttt{:}\Gamma'\\ \Sigma\texttt{;}\ \Delta\texttt{;}\ \Gamma'\texttt{;}\ cid \vdash \texttt{this.\_name=}cid\texttt{;}\ cinits\texttt{:}\ \texttt{ok} \quad \Sigma\texttt{;}\ \Delta\texttt{;}\ \Gamma'\texttt{;}\ \texttt{Some}\ cid \vdash block\texttt{:}\ \texttt{ok}\end{array}}{\Sigma\texttt{;}\ \Delta\texttt{;}\ \Gamma\texttt{;}\ cid \vdash \texttt{new}\ \texttt{(}\overline{typ_j\ id_j}^{\,j}\texttt{)}\ \texttt{()}\ cinits\texttt{\{}block\texttt{\}}\texttt{:}\ \texttt{ok}} \quad \textsc{ctor\_base}$$

$$\frac{\begin{array}{c}\Sigma;\,\Delta;\,(\Gamma;\,\cdot\,)\vdash\overline{typ_j\;id_j}^{\,j}:\Gamma' \quad cid_1<:cid_2\{\Phi;\;\overline{typ_j}^{\,j}\;;\Theta\}\in\Sigma\\ cid_2\;cid_{ext2}\{\Phi_2;\;\overline{typ'_k}^{\,k}\;;\Theta_2\}\in\Sigma \quad \overline{\Sigma;\,\Delta;\,\Gamma';\,\texttt{None}\vdash exp_k<:typ'_k}^{\,k}\\ \Sigma;\,\Delta;\,\Gamma';\,cid_1\vdash\texttt{this.\_name=}cid_1;\;cinits:\texttt{ok} \quad \Sigma;\,\Delta;\,\Gamma';\,\texttt{Some}\;cid_1\vdash block:\texttt{ok}\end{array}}{\Sigma;\,\Delta;\,\Gamma;\,cid_1\vdash\texttt{new (}\overline{typ_j\;id_j}^{\,j}\texttt{) (}\overline{exp_k}^{\,k}\texttt{)}\;cinits\{block\}:\texttt{ok}}\quad\text{CTOR\_INHERITANCE}$$

$\boxed{\Sigma;\,\Delta;\,\Gamma\vdash cdecl:\texttt{ok}}$ *cdecl* is well-formed.

$$\frac{\Sigma\vdash fields:\texttt{ok} \quad \Sigma;\,\Delta;\,\Gamma;\,cid\vdash ctor:\texttt{ok} \quad \overline{\Sigma;\,\Delta;\,\Gamma;\,cid\vdash fdecl_k:\texttt{ok}}^{\,k}}{\Sigma;\,\Delta;\,\Gamma\vdash\texttt{class}\;cid\;cid_{ext}\{fields\;ctor\;\overline{fdecl_k}^{\,k}\};:\texttt{ok}}\quad\text{CDECL\_INTRO}$$

$\boxed{\Sigma;\,\Delta\vdash prog:\Sigma';\,\Delta'}$ Extending contexts by adding function and class declarations.

$$\frac{}{\Sigma;\,\Delta\vdash\epsilon:\Sigma;\,\Delta}\quad\text{GENSD\_NIL}$$

$$\frac{\Sigma;\,\Delta\vdash prog:\Sigma';\,\Delta'}{\Sigma;\,\Delta\vdash vdecl\;prog:\Sigma';\,\Delta'}\quad\text{GENSD\_VDECL}$$

$$\frac{\begin{array}{c}cid\notin\Sigma \quad cid_{ext}\in\Sigma \quad cid_{ext};\,\cdot\vdash fields:\Phi \quad \Sigma;\,cid_{ext};\,\Phi;\,\cdot\vdash\overline{fdecl_k}^{\,k}:\Theta\\ \Sigma,cid\;cid_{ext}\{\Phi;\;\overline{typ_j}^{\,j}\;;\Theta\};\,\Delta\vdash prog:\Sigma';\,\Delta'\end{array}}{\Sigma;\,\Delta\vdash\texttt{class}\;cid\;cid_{ext}\{fields\;\texttt{new (}\overline{typ_j\;id_j}^{\,j}\texttt{) (}\overline{exp_m}^{\,m}\texttt{)}\;cinits\{block\}\;\overline{fdecl_k}^{\,k}\};\;prog:\Sigma';\,\Delta'}\quad\text{GENSD\_CDECL}$$

$$\frac{id\notin\Delta \quad \Sigma;\,\Delta,id:\texttt{(}\overline{typ_j}^{\,j}\texttt{)->}rtyp\vdash prog:\Sigma';\,\Delta'}{\Sigma;\,\Delta\vdash rtyp\;id\,\texttt{(}\overline{typ_j\;id_j}^{\,j}\texttt{)}\;\texttt{extern}\;prog:\Sigma';\,\Delta'}\quad\text{GENSD\_EFUNC}$$

$$\frac{id\notin\Delta \quad \Sigma;\,\Delta,id:\texttt{(}\overline{typ_j}^{\,j}\texttt{)->}typ\vdash prog:\Sigma';\,\Delta'}{\Sigma;\,\Delta\vdash typ\;id\,\texttt{(}\overline{typ_j\;id_j}^{\,j}\texttt{)}\,\{vdecls\;\overline{stmt_k}^{\,k}\;\texttt{return}\;exp;\,\}\;prog:\Sigma';\,\Delta'}\quad\text{GENSD\_FUNC\_TYP}$$

$$\frac{id\notin\Delta \quad \Sigma;\,\Delta,id:\texttt{(}\overline{typ_j}^{\,j}\texttt{)->unit}\vdash prog:\Sigma';\,\Delta'}{\Sigma;\,\Delta\vdash\texttt{unit}\;id\,\texttt{(}\overline{typ_j\;id_j}^{\,j}\texttt{)}\,\{vdecls\;\overline{stmt_k}^{\,k}\;\texttt{return ;}\,\}\;prog:\Sigma';\,\Delta'}\quad\text{GENSD\_FUNC\_UNIT}$$

$\boxed{\Sigma;\,\Delta;\,\Gamma\vdash prog:\texttt{ok}}$ Σ, Δ and Γ show that *prog* is well-formed.

$$\frac{}{\Sigma;\,\Delta;\,\Gamma\vdash\epsilon:\texttt{ok}}\quad\text{PROG\_NIL}$$

$$\frac{\Sigma\vdash typ \quad \Sigma;\,\cdot;\,\cdot;\,\texttt{None}\vdash init<:typ \quad id\notin\Delta\,\text{and}\,\Gamma \quad \Sigma;\,\Delta;\,\gamma,id:typ\vdash prog:\texttt{ok}}{\Sigma;\,\Delta;\,\gamma\vdash typ\;id\texttt{=}init;\;prog:\texttt{ok}}\quad\text{PROG\_VDECL}$$

$$\frac{\Sigma;\,\Delta;\,\Gamma;\,\texttt{None}\vdash fdecl:\texttt{ok} \quad \Sigma;\,\Delta;\,\Gamma\vdash prog:\texttt{ok}}{\Sigma;\,\Delta;\,\Gamma\vdash fdecl\;prog:\texttt{ok}}\quad\text{PROG\_FDECL}$$

$$\frac{\Sigma;\,\Delta;\,\Gamma \vdash \mathit{cdecl} : \mathtt{ok} \quad \Sigma;\,\Delta;\,\Gamma \vdash \mathit{prog} : \mathtt{ok}}{\Sigma;\,\Delta;\,\Gamma \vdash \mathit{cdecl}\;\mathit{prog} : \mathtt{ok}} \quad \text{PROG\_CDECL}$$

$\boxed{\vdash \mathit{prog} : \mathtt{ok}}$ *prog* is well-formed.

$$\frac{\begin{array}{l}\mathtt{Object\;None}\,\{\mathtt{\_name : String\,;}\,\epsilon;\,\mathtt{get\_name : () \text{->} String}\};\,\cdot \vdash \mathit{prog} : \Sigma;\,\Delta \\ \Sigma;\,\Delta;\,\cdot \vdash \mathit{prog} : \mathtt{ok}\end{array}}{\vdash \mathit{prog} : \mathtt{ok}} \quad \text{WF\_INTRO}$$