$$
\begin{aligned}
mod &::= \textbf{MODULE}(mname, dep\ ..., defn\ ...)\\
M &::= mod\ ...\\
mname &::= name\\
dep &::= mname@ph\\
deps &::= dep\ ...\\
defn &::= \langle dname@ph, kind, ast\rangle\\
defns &::= defn\ ...\\
dname &::= name\\
kind &::= \textbf{VALUE} \mid \textbf{MACRO}\\
ph &::= integer\\
binding &::= \langle ph, mname, dname\rangle\\
bindings &::= binding\ ...\\
\Sigma &::= \{cell\ ...\}\\
cell &::= \langle binding, kind, val\rangle\\
&\mid \langle mname@ph\rangle
\end{aligned}
$$

$$
\begin{aligned}
s\text{-}exp &::= name \mid \texttt{(}\ s\text{-}exp\ ...\texttt{)}\\
name &::= \text{a token such as } \texttt{x}, \texttt{clock}, \text{or } \texttt{lambda}\\
ast &::= val\\
&\mid \textbf{REF}(binding)\\
&\mid var\\
&\mid \textbf{APP}(ast, ast, ...)\\
val &::= func\\
&\mid \textbf{QUOTE}(name)\\
&\mid \textbf{SYNTAX}(stx)\\
&\mid \textbf{UNBOUND}\\
func &::= \textbf{LAMBDA}(var\ ..., ast)\\
var &::= \textbf{VAR}(name)\\
stx &::= id \mid \texttt{(}\ stx\ ...\texttt{)}\\
id &::= \textbf{STX}(name, bindings)\\
body &::= stx@ph
\end{aligned}
$$

$\mathsf{cmodule} : s\text{-}exp\ M \to M$

$$
\mathsf{cmodule}[\![\ \texttt{(module}\ mname\ mname_{init}\ s\text{-}exp\ ...\texttt{)}\ , M]\!]
$$
$$
= \mathsf{cbody}[\![body\ ..., mname,\ mname_{init}@0, \varepsilon, M, \Sigma_{init}]\!]
$$

subject to $\mathsf{visit}[\![mname_{init}, 0, M, \varnothing]\!] = \Sigma_{init}$,
$\mathsf{import}[\![mname_{init}, 0, M]\!] = bindings$,
$\mathsf{wrap}[\![s\text{-}exp, bindings]\!]@0\ ... = body\ ...$

$\mathsf{cbody} : body\ ...\ mname\ \ dep\ ...\ defn\ ...\ M\ \Sigma \to M$

$$
\mathsf{cbody}[\![\varepsilon, mname, deps, defns, M, \Sigma]\!] = M\big[mname \longleftarrow \textbf{MODULE}(mname, deps, defns)\big]
$$

$$
\mathsf{cbody}[\![\ \texttt{(}\ id_{beg}\ stx\ ...\texttt{)}\ @ph\ body_{rest}\ ..., mname, deps, defns, M, \Sigma]\!]
$$
$$
= \mathsf{cbody}[\![stx@ph\ ...\ body_{rest}\ ..., mname, deps, defns, M, \Sigma]\!]
$$

subject to $\mathsf{resolve}[\![id_{beg}, ph]\!] = \langle ph, \texttt{base}, \texttt{begin}\rangle$

$$
\mathsf{cbody}[\![\ \texttt{(}\ id_{bfs}\ stx\ ...\texttt{)}\ @ph\ body_{rest}\ ..., mname, deps, defns, M, \Sigma]\!]
$$
$$
= \mathsf{cbody}[\![stx@ph{+}1\ ...\ body_{rest}\ ..., mname, deps, defns, M, \Sigma]\!]
$$

subject to $\mathsf{resolve}[\![id_{bfs}, ph]\!] = \langle ph, \texttt{base}, \texttt{begin-for-syntax}\rangle$

$$
\mathsf{cbody}[\![\ \texttt{(}\ id_{def}\ id_{new}\ stx\ \texttt{)}\ @ph\ body_{rest}\ ..., mname, deps, defn\ ..., M, \Sigma]\!]
$$
$$
= \mathsf{cbody}[\![body_{new}\ ..., mname, deps, defn\ ...\ defn_{new}, M, \Sigma_{new}]\!]
$$

subject to $\mathsf{resolve}[\![id_{def}, ph]\!] = \langle ph, \texttt{base}, \texttt{define}\rangle$,
$\mathsf{strip}[\![id_{new}]\!] = dname$, $\mathsf{cexpr}[\![stx, ph, \varepsilon, \Sigma]\!] = ast$,
$\langle dname@ph, \textbf{VALUE}, ast\rangle = defn_{new}$,
$\langle ph, mname, dname\rangle = binding_{new}$,
$\mathsf{add}[\![body_{rest}, binding_{new}]\!]\ ... = body_{new}\ ...$,
$\mathsf{maybe\text{-}eval}[\![ph, ast, \Sigma]\!] = val$,
$\Sigma\big[binding_{new} \longleftarrow \langle \textbf{VALUE}, val\rangle\big] = \Sigma_{new}$

$$\mathsf{cbody}[\![\,(\,id_{defstx}\ id_{new}\ stx\,)\,@ph\ body_{rest}\ ...,\ mname,\ deps,\ defn\ ...,\ M,\ \Sigma\,]\!]$$
$$=\ \mathsf{cbody}[\![body_{new}\ ...,\ mname,\ deps,\ defn\ ...\ defn_{new},\ M,\ \Sigma_{new}]\!]$$

subject to $\mathsf{resolve}[\![id_{defstx}, ph]\!] = \langle ph, \texttt{base}, \texttt{define-syntax}\rangle$,
$\mathsf{strip}[\![id_{new}]\!] = dname$, $\mathsf{cexpr}[\![stx, ph{+}1, \varepsilon, \Sigma]\!] = ast$,
$\langle dname@ph, \textbf{M}\textsc{acro}, ast\rangle = defn_{new}$,
$\langle ph, mname, dname\rangle = binding_{new}$,
$\mathsf{add}[\![body_{rest}, binding_{new}]\!]\ ... = body_{new}\ ...$,
$\mathsf{eval}[\![ast, \Sigma]\!] = val$,
$\Sigma\big[binding_{new} \longleftarrow \langle \textbf{M}\textsc{acro}, val\rangle\big] = \Sigma_{new}$

$$\mathsf{cbody}[\![\,(\,id_{macro}\ stx\ ...\,)\,@ph\ body_{rest}\ ...,\ mname,\ deps,\ defns,\ M,\ \Sigma\,]\!]$$
$$=\ \mathsf{cbody}[\![stx_{new}@ph\ body_{rest}\ ...,\ mname,\ deps,\ defns,\ M,\ \Sigma]\!]$$

subject to $\mathsf{resolve}[\![id_{macro}, ph]\!] = binding$,
$\Sigma(binding) = \langle \textbf{M}\textsc{acro}, func\rangle$,
$\mathsf{eval}[\![\textbf{A}\textsc{pp}(func, \textbf{S}\textsc{yntax}(\,(\,id_{macro}\ stx\ ...\,)\,)), \Sigma]\!] = \textbf{S}\textsc{yntax}(stx_{new})$

$$\mathsf{cbody}[\![\,(\,id_{req}\ stx_{in}\,)\,@ph\ body_{rest}\ ...,\ mname,\ dep\ ...,\ defns,\ M,\ \Sigma\,]\!]$$
$$=\ \mathsf{cbody}[\![body_{new}\ ...,\ mname,\ dep\ ...\ dep_{new},\ defns,\ M,\ \Sigma_{new}]\!]$$

subject to $\mathsf{resolve}[\![id_{req}, ph]\!] = \langle ph, \texttt{base}, \texttt{require}\rangle$,
$\mathsf{parse\text{-}req}[\![stx_{in}, ph]\!] = \langle mname_{in}, ph_{\Delta}\rangle$,
$\mathsf{visit}[\![mname_{in}, ph_{\Delta}, M, \Sigma]\!] = \Sigma_{new}$,
$\mathsf{import}[\![mname_{in}, ph_{\Delta}, M]\!] = bindings_{new}$,
$\mathsf{add}[\![body_{rest}, bindings_{new}]\!]\ ... = body_{new}\ ...$,
$mname_{in}@ph_{\Delta} = dep_{new}$

$$\mathsf{cbody}[\![\,(\,id_{mod}\ id_{sub}\ id_{init}\ stx\ ...\,)\,@ph\ body_{rest}\ ...,\ mname,\ deps,\ defns,\ M,\ \Sigma\,]\!]$$
$$=\ \mathsf{cbody}[\![body_{rest}\ ...,\ mname,\ deps,\ defns,\ M_{new},\ \Sigma]\!]$$

subject to $\mathsf{resolve}[\![id_{mod}, ph]\!] = \langle ph, \texttt{base}, \texttt{module}\rangle$,
$\mathsf{strip}[\![id_{sub}]\!] = mname_{sub}$, $\mathsf{strip}[\![id_{init}]\!] = mname_{init}$,
$\mathsf{cmodule}[\![\,(\,\texttt{module}\ mname_{sub}\ mname_{init}\ \mathsf{strip}[\![stx]\!]\ ...\,)\,, M]\!] = M_{new}$

$$\mathsf{cbody}[\![\,(\,id_{mod}\ id_{sub}\ id_{init}\ stx\ ...\,)\,@ph\ body_{rest}\ ...,\ mname,\ deps,\ defns,\ M,\ \Sigma\,]\!]$$
$$=\ \mathsf{cmodule}[\![\,(\,\texttt{module}\ mname_{sub}\ mname_{init}\ \mathsf{strip}[\![stx]\!]\ ...\,)\,, M_{new}]\!]$$

subject to $\mathsf{resolve}[\![id_{mod}, ph]\!] = \langle ph, \texttt{base}, \texttt{module*}\rangle$,
$\mathsf{strip}[\![id_{sub}]\!] = mname_{sub}$, $\mathsf{strip}[\![id_{init}]\!] = mname_{init}$,
$\mathsf{cbody}[\![body_{rest}\ ...,\ mname, deps, defns, M, \Sigma]\!] = M_{new}$

$$\mathsf{cbody}[\![\,(\,id_{mod}\ id_{sub}\ (\,)\ stx\ ...\,)\,@ph\ body_{rest}\ ...,\ mname,\ deps,\ defns,\ M,\ \Sigma\,]\!]$$
$$=\ \mathsf{cbody}[\![\mathsf{shift}[\![stx, 0{-}ph]\!]@0\ ...,\ mname_{sub},\ mname@0{-}ph,\ \varepsilon,\ M_{new},\ \Sigma_{init}]\!]$$

subject to $\mathsf{resolve}[\![id_{mod}, ph]\!] = \langle ph, \texttt{base}, \texttt{module*}\rangle$,
$\mathsf{strip}[\![id_{sub}]\!] = mname_{sub}$,
$\mathsf{cbody}[\![body_{rest}\ ...,\ mname, deps, defns, M, \Sigma]\!] = M_{new}$,
$\mathsf{visit}[\![mname, 0{-}ph, M_{new}, \varnothing]\!] = \Sigma_{init}$

$\mathsf{cexpr}\ :\ stx\ ph\ env\ \Sigma\ \rightarrow\ ast$

$$\mathsf{cexpr}[\![id, ph, env, \Sigma]\!]\ =\ var$$
subject to $\mathsf{env\text{-}lookup}[\![env, \textbf{V}\textsc{ar}(\mathsf{strip}[\![id]\!])]\!] = var$

$\mathsf{cexpr}[\![id, ph, env, \Sigma]\!] \;=\; \textbf{Ref}(binding)$
subject to $\mathsf{resolve}[\![id, ph]\!] = binding$

$\mathsf{cexpr}[\![\,(\,id_{lambda}\;(\,id\;...\,)\;stx\,)\,, ph, name_{env}\;..., \Sigma]\!]$
$= \textbf{Lambda}(\textbf{Var}(\mathsf{strip}[\![id]\!])\;..., \mathsf{cexpr}[\![stx, ph, \textbf{Var}(\mathsf{strip}[\![id]\!])\;...\;name_{env}\;..., \Sigma]\!])$
subject to $\mathsf{resolve}[\![id_{lambda}, ph]\!] = \langle ph, \texttt{base}, \texttt{lambda}\rangle$

$\mathsf{cexpr}[\![\,(\,id_{quote}\;stx\,)\,, ph, env, \Sigma]\!] \;=\; \textbf{Quote}(\mathsf{strip}[\![stx]\!])$
subject to $\mathsf{resolve}[\![id_{quote}, ph]\!] = \langle ph, \texttt{base}, \texttt{quote}\rangle$

$\mathsf{cexpr}[\![\,(\,id_{qstx}\;stx\,)\,, ph, env, \Sigma]\!] \;=\; \textbf{Syntax}(stx)$
subject to $\mathsf{resolve}[\![id_{qstx}, ph]\!] = \langle ph, \texttt{base}, \texttt{syntax}\rangle$

$\mathsf{cexpr}[\![\,(\,id_{macro}\;stx\;...\,)\,, ph, env, \Sigma]\!] \;=\; \mathsf{cexpr}[\![stx_{new}, ph, env, \Sigma]\!]$
subject to $\mathsf{resolve}[\![id_{macro}, ph]\!] = binding$,
$\Sigma(binding) = \langle \textbf{Macro}, func\rangle$,
$\mathsf{eval}[\![\textbf{App}(func, \textbf{Syntax}(\,(\,id_{macro}\;stx\;...\,)\,)), \Sigma]\!] = \textbf{Syntax}(stx_{new})$

$\mathsf{cexpr}[\![\,(\,stx\;...\,)\,, ph, env, \Sigma]\!] \;=\; \textbf{App}(\mathsf{cexpr}[\![stx, ph, env, \Sigma]\!], ...)$

$\mathsf{cexpr}[\![id, ph, env, \Sigma]\!] \;=\; \textbf{Unbound}$

$\mathsf{eval} : ast\;\Sigma \rightarrow val$

$\mathsf{eval}[\![val, \Sigma]\!] \;=\; val$

$\mathsf{eval}[\![\textbf{Ref}(binding), \Sigma]\!] \;=\; val$
subject to $\Sigma(binding) = \langle \textbf{Value}, val\rangle$

$\mathsf{eval}[\![\textbf{App}(ast, ast_{arg}, ...), \Sigma]\!] \;=\; \mathsf{eval}[\![ast_{body}\big[var\;... \longleftarrow val\;...\big], \Sigma]\!]$
subject to $\mathsf{eval}[\![ast, \Sigma]\!] = \textbf{Lambda}(var\;..., ast_{body})$,
$\mathsf{eval}[\![ast_{arg}, \Sigma]\!]\;... = val\;...$

$\mathsf{maybe\text{-}eval} : ph\;ast\;\Sigma \rightarrow val$

$\mathsf{maybe\text{-}eval}[\![0, ast, \Sigma]\!] \;=\; \textbf{Unbound}$

$\mathsf{maybe\text{-}eval}[\![ph, ast, \Sigma]\!] \;=\; \mathsf{eval}[\![ast, \Sigma]\!]$

$\mathsf{ast\text{-}shift} : ast\;ph \rightarrow ast$

$\mathsf{ast\text{-}shift}[\![var, ph]\!] \;=\; var$

$\mathsf{ast\text{-}shift}[\![\textbf{Lambda}(var\;..., ast), ph]\!] \;=\; \textbf{Lambda}(var\;..., \mathsf{ast\text{-}shift}[\![ast, ph]\!])$

$\mathsf{ast\text{-}shift}[\![\textbf{Quote}(name), ph]\!] \;=\; \textbf{Quote}(name)$

$\text{ast-shift}[\![\mathbf{APP}(ast, ...), ph]\!] \;=\; \mathbf{APP}(\text{ast-shift}[\![ast, ph]\!], ...)$

$\text{ast-shift}[\![\mathbf{REF}(\langle ph_{ref}, mname, dname\rangle), ph]\!] \;=\; \mathbf{REF}(\langle ph_{ref}{+}ph, mname, dname\rangle)$

$\text{ast-shift}[\![\mathbf{SYNTAX}(stx), ph]\!] \;=\; \mathbf{SYNTAX}(\text{shift}[\![stx, ph]\!])$

$\text{ast-shift}[\![\mathbf{UNBOUND}, ph]\!] \;=\; \mathbf{UNBOUND}$

$\text{visit} \;:\; mname\;\; ph\;\; M\;\; \Sigma \;\to\; \Sigma$

$\text{visit}[\![mname, ph_\Delta, M, \Sigma]\!] \;=\; \Sigma$
subject to $\Sigma(mname@ph_\Delta) = \mathbf{READY}$

$\text{visit}[\![mname, ph_\Delta, M, \Sigma]\!] \;=\; \text{visit-body}[\![defn\ ..., mname, ph_\Delta, \Sigma_{deps}]\!]$
subject to $M(mname) = \mathbf{MODULE}(mname, deps, defn\ ...)$,
$\Sigma\big[mname@ph_\Delta \longleftarrow \mathbf{READY}\big] = \Sigma_{init}$,
$\text{visit*}[\![deps, ph_\Delta, M, \Sigma_{init}]\!] = \Sigma_{deps}$

$\text{visit*} \;:\; deps\;\; ph\;\; M\;\; \Sigma \;\to\; \Sigma$

$\text{visit*}[\![\varepsilon, ph_\Delta, M, \Sigma]\!] \;=\; \Sigma$

$\text{visit*}[\![mname@ph\ dep\ ..., ph_\Delta, M, \Sigma]\!] \;=\; \text{visit*}[\![dep\ ..., ph_\Delta, M, \Sigma_{new}]\!]$
subject to $\text{visit}[\![mname, ph_\Delta{+}ph, M, \Sigma]\!] = \Sigma_{new}$

$\text{visit-body} \;:\; defns\;\; name\;\; ph\;\; \Sigma \;\to\; \Sigma$

$\text{visit-body}[\![\varepsilon, mname, ph_\Delta, \Sigma]\!] \;=\; \Sigma$

$\text{visit-body}[\![\langle dname@ph_{def}, \mathbf{VALUE}, ast\rangle\ defn\ ..., mname, ph_\Delta, \Sigma]\!]$
$=\; \text{visit-body}[\![defn\ ..., mname, ph_\Delta, \Sigma]\!]$
subject to $ph_\Delta{+}ph_{def} = 0$

$\text{visit-body}[\![\langle dname@ph_{def}, kind, ast\rangle\ defn\ ..., mname, ph_\Delta, \Sigma]\!]$
$=\; \text{visit-body}[\![defn\ ..., mname, ph_\Delta, \Sigma]\!]$
subject to $\text{pmax}[\![-1, ph_\Delta{+}ph_{def}]\!] = -1$

$\text{visit-body}[\![\langle dname@ph_{def}, kind, ast\rangle\ defn\ ..., mname, ph_\Delta, \Sigma]\!]$
$=\; \text{visit-body}[\![defn\ ..., mname, ph_\Delta, \Sigma_{new}]\!]$
subject to $ph_\Delta{+}ph_{def} = ph$, $\text{eval}[\![\text{shift}[\![ast, ph_\Delta]\!], \Sigma]\!] = val$,
$\Sigma\big[\langle ph, mname, dname\rangle \longleftarrow \langle kind, val\rangle\big] = \Sigma_{new}$

$\text{run} \;:\; mname\;\; ph\;\; M\;\; \Sigma \;\to\; \Sigma$

$\text{run}[\![mname, ph_\Delta, M, \Sigma]\!] \;=\; \Sigma$
subject to $\Sigma(mname@ph_\Delta) = \mathbf{READY}$

$\text{run}[\![mname, ph_\Delta, M, \Sigma]\!] \ = \ \text{run-body}[\![defn\ ..., mname, ph_\Delta, \Sigma_{deps}]\!]$
subject to $M(mname) = \textbf{Module}(mname, deps, defn\ ...)$,
$\Sigma\big[mname@ph_\Delta \longleftarrow \textbf{Ready}\big] = \Sigma_{init}$,
$\text{run*}[\![deps, ph_\Delta, M, \Sigma_{init}]\!] = \Sigma_{deps}$

$\text{run*} \ : \ deps\ \ ph\ \ M\ \ \Sigma \ \rightarrow \ \Sigma$

$\text{run*}[\![\varepsilon, ph_\Delta, M, \Sigma]\!] \ = \ \Sigma$

$\text{run*}[\![mname@ph\ dep\ ..., ph_\Delta, M, \Sigma]\!] \ = \ \text{run*}[\![dep\ ..., ph_\Delta, M, \Sigma_{new}]\!]$
subject to $\text{run}[\![mname, ph_\Delta{+}ph, M, \Sigma]\!] = \Sigma_{new}$

$\text{run-body} \ : \ defns\ \ name\ \ ph\ \ \Sigma \ \rightarrow \ \Sigma$

$\text{run-body}[\![\varepsilon, mname, ph_\Delta, \Sigma]\!] \ = \ \Sigma$

$\text{run-body}[\![\langle dname@ph_{def}, \textbf{Value}, ast\rangle\ defn\ ..., mname, ph_\Delta, \Sigma]\!]$
$= \ \text{run-body}[\![defn\ ..., mname, ph_\Delta, \Sigma_{new}]\!]$
subject to $ph_\Delta{+}ph_{def} = 0$, $\text{eval}[\![\text{shift}[\![ast, ph_\Delta]\!], \Sigma]\!] = val$,
$\Sigma\big[\langle 0, mname, dname\rangle \longleftarrow \langle\textbf{Value}, val\rangle\big] = \Sigma_{new}$

$\text{run-body}[\![defn_{skip}\ defn\ ..., mname, ph_\Delta, \Sigma]\!]$
$= \ \text{run-body}[\![defn\ ..., mname, ph_\Delta, \Sigma]\!]$

$\text{wrap} \ : \ s\text{-}exp\ \ bindings \ \rightarrow \ stx$

$\text{wrap}[\![name, bindings]\!] \ = \ \textbf{Stx}(name, bindings)$

$\text{wrap}[\![\ (\ s\text{-}exp\ ...\ )\ , bindings]\!] \ = \ (\ \text{wrap}[\![s\text{-}exp, bindings]\!]\ ...\ )$

$\text{strip} \ : \ stx \ \rightarrow \ s\text{-}exp$

$\text{strip}[\![\textbf{Stx}(name, bindings)]\!] \ = \ name$

$\text{strip}[\![\ (\ stx\ ...\ )\ ]\!] \ = \ (\ \text{strip}[\![stx]\!]\ ...\ )$

$\text{add-to-stx} \ : \ stx\ \ bindings \ \rightarrow \ stx$

$\text{add-to-stx}[\![\textbf{Stx}(name, binding\ ...), binding_{new}\ ...]\!]$
$= \ \textbf{Stx}(name, binding\ ...\ binding_{new}\ ...)$

$\text{add-to-stx}[\![\ (\ stx\ ...\ )\ , bindings]\!] \ = \ (\ \text{add-to-stx}[\![stx, bindings]\!]\ ...\ )$

$\text{add} \ : \ body\ \ bindings \ \rightarrow \ body$

$\text{add}[\![stx@ph, bindings]\!] \ = \ \text{add-to-stx}[\![stx, bindings]\!]@ph$

$\mathsf{resolve} : id\ ph \to maybe\text{-}binding$

$\mathsf{resolve}[\![\mathbf{Stx}(dname, binding_{pre}\ ...\ binding\ binding_{post}\ ...), ph]\!] = binding$
subject to $binding = \langle ph, mname, dname\rangle$

$\mathsf{resolve}[\![id, ph]\!] = \#\mathrm{f}$

$\mathsf{shift} : stx\text{-}or\text{-}ast\ ph \to stx\text{-}or\text{-}ast$

$\mathsf{shift}[\![\mathbf{Stx}(name, \langle ph, mname, dname\rangle\ ...), ph_{shift}]\!]$
$= \mathbf{Stx}(name, \langle ph{+}ph_{shift}, mname, dname\rangle\ ...)$

$\mathsf{shift}[\![\ (\ stx\ ...\ )\ , ph_{shift}]\!] = (\ \mathsf{shift}[\![stx, ph_{shift}]\!]\ ...\ )$

$\mathsf{shift}[\![ast, ph_{shift}]\!] = \mathsf{ast\text{-}shift}[\![ast, ph_{shift}]\!]$

$\mathsf{parse\text{-}req} : stx\ ph \to \langle name, ph\rangle$

$\mathsf{parse\text{-}req}[\![id_{in}, ph]\!] = \langle \mathsf{strip}[\![id_{in}]\!], ph\rangle$

$\mathsf{parse\text{-}req}[\![\ (\ id_{fs}\ stx_{in}\ )\ , ph]\!] = \mathsf{parse\text{-}req}[\![stx_{in}, ph{+}1]\!]$
subject to $\mathsf{strip}[\![id_{fs}]\!] =$ `for-syntax`

$\mathsf{parse\text{-}req}[\![\ (\ id_{ft}\ stx_{in}\ )\ , ph]\!] = \mathsf{parse\text{-}req}[\![stx_{in}, ph{-}1]\!]$
subject to $\mathsf{strip}[\![id_{ft}]\!] =$ `for-template`

$\mathsf{import} : mname\ ph_{\Delta}\ M \to bindings$

$\mathsf{import}[\![mname, ph_{\Delta}, M]\!] = \langle ph_{\Delta}{+}ph_{def}, mname, dname\rangle\ ...$
subject to $M(mname) = \mathbf{Module}(mname, dep\ ..., defn\ ...)$,
$defn\ ... = \langle dname@ph_{def}, kind, ast\rangle\ ...$