1 Bytecode Abstract Syntax

b	$::=$	$v\ P\ \bar{i}\ \overline{M}$	*bytecodefile*
v	$::=$	*magic major.minor*	*magic+version number*
P	$::=$	*fully qualified path*	*path to module*
i	$::=$	*import* μ *URI* : τ *as* x	*module import*
μ	$::=$	$[\texttt{metadata}]\ \ [\texttt{type}]$	
M	$::=$	$\texttt{module}\ P : \tau = e$	*top level modules*
	$\mid$	$\texttt{type}\ P = T\ [\delta]$	
e	$::=$	x	*expressions*
	$\mid$	$\texttt{new}\ \tau\ \{x \Rightarrow \overline{d}\}$	
	$\mid$	$e.m(\overline{e})$	
	$\mid$	$e.f$	
	$\mid$	$e.f = e$	
	$\mid$	$\mathscr{L}$	
	$\mid$	$e.\texttt{match}\ \overline{x : p.L \Rightarrow e}\ [\texttt{else}\ e]$	
	$\mid$	$\overline{e \mid d}$	
$\mathscr{L}$	$::=$	*string*	*literals*
	$\mid$	*integer*	

d	$::=$	$\texttt{val}\ f : \tau = e$	*declarations*
	$\mid$	$\texttt{var}\ f : \tau = e$	
	$\mid$	$\texttt{def}\ m(\overline{x : \tau}) : \tau = e$	
	$\mid$	$\texttt{type}\ L = T\ [\delta]$	
T	$::=$	c	*type desc.*
	$\mid$	$\texttt{extag}\ c$	
	$\mid$	$\texttt{datatag}\ \overline{p.L}\ c$	
c	$::=$	τ	*case desc.*
	$\mid$	$\texttt{extends}\ p.L\ \tau$	
τ	$::=$	$\tau\ \{\texttt{x} \Rightarrow \overline{\sigma}\}_s$	*type*
	$\mid$	$p.L$	
	$\mid$	$\top$	
	$\mid$	$\bot$	
	$\mid$	$?$	
p	$::=$	x	*paths*
	$\mid$	$p.f$	
s	$::=$	$\texttt{stateful} \mid \texttt{pure}$	
σ	$::=$	$\texttt{val}\ f : \tau$	*decl type*
	$\mid$	$\texttt{var}\ f : \tau$	
	$\mid$	$\texttt{def}\ m : \Pi\overline{x{:}\tau}.\tau$	
	$\mid$	$\texttt{type}\ L = T\ [\delta]$	
	$\mid$	$\texttt{type}_s\ L\ [\delta]$	
δ	$::=$	$\texttt{metadata}\ \ e$	*metadata*

Notation: overbar means a list of elements, as in Java