

# asm.js

Dave Herman, Luke Wagner, and Alon Zakai

November 4, 2012

## 1 Abstract syntax

$$\begin{aligned} b, e, f, g, x, y, z &\in \text{Identifier} \\ \text{arguments} &\notin \text{Identifier} \\ \text{eval} &\notin \text{Identifier} \end{aligned}$$
$$\begin{aligned} P &::= \text{function}(b, e) \{ \overline{\text{imp}_x} \overline{\text{fn}_f} \text{exp} \} \\ \text{imp}_x &::= \text{var } x = e.y; \\ &\quad | \text{var } x = \text{new } e.y(b); \\ \text{exp} &::= \text{return } f; \\ &\quad | \text{return } \{ \overline{x:f} \}; \\ \text{fn}_f &::= \text{function } f(\overline{x}) \{ \overline{x = \kappa_x}; \text{var } \overline{y = v}; ss \} \end{aligned}$$
$$\begin{aligned} s &::= \{ ss \} \\ &\quad | e; \\ &\quad | \text{if } (e) \ s \\ &\quad | \text{if } (e) \ s \ \text{else } s \\ &\quad | \text{return } v; \\ &\quad | \text{while } (e) \ s \\ &\quad | \text{do } s \ \text{while } (e); \\ &\quad | \text{for } (e; e; e) \ s \\ &\quad | \text{switch } (e) \{ \overline{c} \} \\ &\quad | \text{switch } (e) \{ \overline{c} \ d \} \\ &\quad | \text{break}; \\ &\quad | \text{break } lab; \\ &\quad | \text{continue}; \\ &\quad | \text{continue } lab; \\ &\quad | lab: s \end{aligned}$$
$$ss ::= \overline{s}$$
$$\begin{aligned} c &::= \text{case } e: ss \\ d &::= \text{default}: ss \\ cd &::= c \mid d \end{aligned}$$

$$\kappa_X ::= \sim\sim X \mid +X$$

$$v ::= r \mid n$$

$$e ::= \begin{array}{l} v \\ lval \\ lval = e \\ f(\bar{e}) \\ unop\ e \\ e\ binop\ e \\ e\ ?\ e : e \\ (\bar{e}) \end{array}$$

$$unop ::= + \mid \sim \mid !$$

$$binop ::= \begin{array}{l} + \mid - \mid * \mid / \mid \% \\ \mid \mid \& \mid \wedge \mid << \mid >> \mid >>> \\ \mid < \mid <= \mid > \mid >= \mid != \mid == \end{array}$$

$$lval ::= x \mid x[(e\ \&\ m)\ >>\ n]$$

## 2 Type rules

$$\sigma, \tau ::= \begin{array}{l} \text{bit} \mid \text{int} \mid \text{boolish} \\ \text{signed} \mid \text{unsigned} \\ \text{double} \\ \text{array}_\tau^n \mid \text{function} \mid \text{unknown} \mid \text{jsval} \\ \text{intish} \\ \text{imul} \\ ((\bar{\sigma}) \rightarrow \tau) \wedge \dots \wedge ((\bar{\sigma}') \rightarrow \tau') \\ \text{void} \end{array}$$

$$\begin{array}{l} \ell ::= lab \mid \epsilon \\ L ::= \{\bar{\ell}\} \\ \varepsilon ::= L \mid \text{return} \end{array}$$

$$\begin{array}{l} L ; L' = L \cup L' \\ \emptyset ; \text{return} = \text{return} \\ \{\ell, \bar{\ell}'\} ; \text{return} = \{\ell, \bar{\ell}'\} \\ \text{return} ; L = \text{return} \end{array}$$

$$\begin{array}{l} L \cup \text{return} = L \\ \text{return} \cup L = L \\ \text{return} \cup \text{return} = \text{return} \end{array}$$

$$\begin{aligned}
\text{type}(\sim\sim X) &= \text{int} \\
\text{type}(+X) &= \text{double} \\
\text{type}(n) &= \text{int} \\
\text{type}(r) &= \text{double}
\end{aligned}$$

$$\begin{aligned}
\text{signed, unsigned} &<: \text{int, jsval} \\
\text{bit, int} &<: \text{boolish} \\
\text{void, double, array}_\tau^n, \text{function, unknown} &<: \text{jsval} \\
\text{unknown, int} &<: \text{intish} \\
((\bar{\sigma}) \rightarrow \tau) \wedge \dots \wedge ((\bar{\sigma}') \rightarrow \tau') &<: \text{function} \\
((\bar{\sigma}_1) \rightarrow \tau_1) \wedge \dots \wedge ((\bar{\sigma}_n) \rightarrow \tau_n) &<: ((\bar{\sigma}_1) \rightarrow \tau_1) \wedge \dots \wedge ((\bar{\sigma}_{n-1}) \rightarrow \tau_{n-1}) \\
\text{imul} &<: (\text{intish, intish}) \rightarrow \text{signed}
\end{aligned}$$

$$\Gamma ::= \{\bar{x} : \bar{\tau}\} \mid \Gamma, \{\bar{x} : \bar{\tau}\}$$

$$\begin{aligned}
M(\text{imul}) &: \text{imul} \\
M(\text{ceil}), M(\text{sin}), M(\text{cos}) &: (\text{double}) \rightarrow \text{double}
\end{aligned}$$

$$\begin{aligned}
A(\text{Uint8Array}), A(\text{Int8Array}) &= \text{array}_{\text{int}}^8 \\
A(\text{Uint16Array}), A(\text{Int16Array}) &= \text{array}_{\text{int}}^{16} \\
A(\text{Uint32Array}), A(\text{Int32Array}) &= \text{array}_{\text{int}}^{32} \\
A(\text{Float32Array}) &= \text{array}_{\text{double}}^{32} \\
A(\text{Float64Array}) &= \text{array}_{\text{double}}^{64}
\end{aligned}$$

$$\begin{aligned}
+, - &: (\text{double, double}) \rightarrow \text{double} \\
&\wedge (\text{int, int}) \rightarrow \text{intish} \\
* &: (\text{double, double}) \rightarrow \text{double} \\
/, \% &: (\text{double, double}) \rightarrow \text{double} \\
&\wedge (\text{signed, signed}) \rightarrow \text{intish} \\
&\wedge (\text{unsigned, unsigned}) \rightarrow \text{intish} \\
|, \&, \wedge, <<, >> &: (\text{intish, intish}) \rightarrow \text{signed} \\
>>> &: (\text{intish, intish}) \rightarrow \text{unsigned} \\
<, <=, >, >=, ==, != &: (\text{signed, signed}) \rightarrow \text{bit} \\
&\wedge (\text{unsigned, unsigned}) \rightarrow \text{bit} \\
&\wedge (\text{double, double}) \rightarrow \text{bit} \\
+ &: (\text{intish}) \rightarrow \text{double} \\
\sim &: (\text{intish}) \rightarrow \text{signed} \\
! &: (\text{boolish}) \rightarrow \text{bit}
\end{aligned}$$

Program checking

$\boxed{\vdash P \text{ ok}}$

$$\frac{\begin{array}{c} \text{[T-PROGRAM]} \\ \{\bar{x}\} \cap \{\bar{f}\} = \emptyset \quad \{\bar{x}\} \cap \{b, e\} = \emptyset \quad \{\bar{f}\} \cap \{b, e\} = \emptyset \\ \forall i. b; e; \Gamma_0 \vdash \text{imp}_x \text{ ok} \quad \forall i. \Gamma_0, \Gamma_1 \vdash \text{fn}_f \text{ ok} \quad \forall i. \Gamma_0, \Gamma_1 \vdash \text{exp} \text{ ok} \end{array}}{\vdash \text{function}(b, e) \{ \text{imp}_x \text{ fn}_f \text{ exp} \} \text{ ok}}$$

Import checking

$\boxed{b; e; \Gamma \vdash \text{imp} \text{ ok}}$

$$\begin{array}{c} \text{[T-IMPORTSTD]} \\ \frac{\Gamma(x) = M(y)}{b; e; \Gamma \vdash \text{var } x = e.y; \text{ ok}} \end{array} \quad \begin{array}{c} \text{[T-IMPORTFFI]} \\ \frac{y \notin \text{dom}(M) \quad \Gamma(x) = \text{function}}{b; e; \Gamma \vdash \text{var } x = e.y; \text{ ok}} \end{array}$$

$$\begin{array}{c} \text{[T-VIEW]} \\ \frac{\Gamma(x) = \text{array}_{A(y)}^n}{b; e; \Gamma \vdash \text{var } x = e.y(b); \text{ ok}} \end{array} \quad \begin{array}{c} \text{[T-NEVIEW]} \\ \frac{\Gamma(x) = \text{array}_{A(y)}^n}{b; e; \Gamma \vdash \text{var } x = \text{new } e.y(b); \text{ ok}} \end{array}$$

Function checking

$\boxed{\Gamma \vdash \text{fn} \text{ ok}}$

$$\begin{array}{c} \text{[T-FUNCTION]} \\ \frac{\begin{array}{c} \{\bar{x}\} \cap \{\bar{y}\} = \emptyset \quad \Gamma(f) = (\bar{\sigma}) \rightarrow \tau \quad \bar{\sigma} = \overline{\text{type}(\kappa_x)} \quad \tau \neq \text{void} \\ \Gamma, \{\bar{x} : \bar{\sigma}, \bar{y} : \text{type}(v)\}; \emptyset \vdash ss : \tau / \text{return} \end{array}}{\Gamma \vdash \text{function } f(\bar{x}) \{ \bar{x} = \kappa_x; \text{var } \bar{y} = v; ss \} \text{ ok}} \end{array}$$

$$\begin{array}{c} \text{[T-VOIDFUNCTION]} \\ \frac{\begin{array}{c} \{\bar{x}\} \cap \{\bar{y}\} = \emptyset \quad \Gamma(f) = (\bar{\sigma}) \rightarrow \text{void} \quad \bar{\sigma} = \overline{\text{type}(\kappa_x)} \\ \Gamma, \{\bar{x} : \bar{\sigma}, \bar{y} : \text{type}(v)\}; \emptyset \vdash ss : \text{void} / \varepsilon \end{array}}{\Gamma \vdash \text{function } f(\bar{x}) \{ \bar{x} = \kappa_x; \text{var } \bar{y} = v; ss \} \text{ ok}} \end{array}$$

Export checking

$\boxed{\Gamma \vdash \text{exp} \text{ ok}}$

$$\begin{array}{c} \text{[T-SINGLETON]} \\ \frac{\Gamma(f) = (\bar{\sigma}) \rightarrow \tau \quad \tau <: \text{jsval}}{\Gamma \vdash \text{return } f; \text{ ok}} \end{array} \quad \begin{array}{c} \text{[T-MODULE]} \\ \frac{\forall f. \Gamma(f) <: (\bar{\sigma}) \rightarrow \tau \wedge \tau <: \text{jsval}}{\Gamma \vdash \text{return } \{ \bar{x} : \bar{f} \}; \text{ ok}} \end{array}$$

Statement list checking

$$\boxed{\Gamma; L \vdash ss : \tau/\varepsilon}$$

$$\frac{[T\text{-NoStatements}] \quad \Gamma; L \vdash \epsilon : \tau/\emptyset}{\Gamma; L \vdash \epsilon : \tau/\emptyset} \quad \frac{[T\text{-Statements}] \quad \begin{array}{c} \forall i. \Gamma; L \vdash s_i : \tau/\varepsilon_i \\ n > 0 \quad \varepsilon = \varepsilon_1 ; \dots ; \varepsilon_n \end{array}}{\Gamma; L \vdash \bar{s} : \tau/\varepsilon}$$

Statement checking

$$\boxed{\Gamma; L \vdash s : \tau/\varepsilon}$$

$$\begin{array}{c} [T\text{-Block}] \quad \frac{\Gamma; \emptyset \vdash ss : \tau/\varepsilon}{\Gamma; L \vdash \{ ss \} : \tau/\varepsilon} \quad [T\text{-ExprStmt}] \quad \frac{\Gamma \vdash e : \sigma}{\Gamma; L \vdash e ; : \tau/\emptyset} \\ \\ [T\text{-If}] \quad \frac{\begin{array}{c} \Gamma \vdash e : \text{boolish} \\ \Gamma; \emptyset \vdash s : \tau/\varepsilon \\ \varepsilon' = \varepsilon \cup \emptyset \end{array}}{\Gamma; L \vdash \text{if } (e) \text{ } s : \tau/\varepsilon'} \quad [T\text{-IfElse}] \quad \frac{\begin{array}{c} \Gamma \vdash e : \text{boolish} \\ \Gamma; \emptyset \vdash s_1 : \tau/\varepsilon_1 \quad \Gamma; \emptyset \vdash s_2 : \tau/\varepsilon_2 \\ \varepsilon = \varepsilon_1 \cup \varepsilon_2 \end{array}}{\Gamma; L \vdash \text{if } (e) \text{ } s_1 \text{ else } s_2 : \tau/\varepsilon} \\ \\ [T\text{-ReturnExpr}] \quad \frac{\Gamma \vdash e : \tau}{\Gamma; L \vdash \text{return } e ; : \tau/\text{return}} \quad [T\text{-ReturnVoid}] \quad \frac{}{\Gamma; L \vdash \text{return}; : \text{void}/\text{return}} \\ \\ [T\text{-While}] \quad \frac{\begin{array}{c} \Gamma \vdash e : \text{boolish} \\ \Gamma; L \cup \{\epsilon\} \vdash s : \tau/\varepsilon \\ \varepsilon' = \emptyset \cup \varepsilon - (L \cup \{\epsilon\}) \end{array}}{\Gamma; L \vdash \text{while } (e) \text{ } s : \tau/\varepsilon'} \quad [T\text{-DoWhile}] \quad \frac{\begin{array}{c} \Gamma; L \cup \{\epsilon\} \vdash s : \tau/\varepsilon \\ \Gamma \vdash e : \text{boolish} \\ \varepsilon' = \varepsilon - (L \cup \{\epsilon\}) \end{array}}{\Gamma; L \vdash \text{do } s \text{ while } (e); : \tau/\varepsilon'} \\ \\ [T\text{-For}] \quad \frac{\begin{array}{c} \forall i. \Gamma \vdash e_i : \sigma_i \\ \Gamma; L \cup \{\epsilon\} \vdash s : \tau/\varepsilon \\ \varepsilon' = \emptyset \cup \varepsilon - (L \cup \{\epsilon\}) \end{array}}{\Gamma; L \vdash \text{for } (e_1; e_2; e_3) \text{ } s : \tau/\varepsilon'} \end{array}$$

Statement checking (cont'd)

$$\boxed{\Gamma; L \vdash s : \tau/\varepsilon}$$

$$\begin{array}{c} \text{[T-BREAK]} \\ \hline \Gamma; L \vdash \mathbf{break}; : \tau/\{\epsilon\} \end{array} \quad \begin{array}{c} \text{[T-BREAKLABEL]} \\ \hline \Gamma; L \vdash \mathbf{break} \text{ } lab; : \tau/\{lab\} \end{array}$$

$$\begin{array}{c} \text{[T-CONTINUE]} \\ \hline \Gamma; L \vdash \mathbf{continue}; : \tau/\emptyset \end{array} \quad \begin{array}{c} \text{[T-CONTINUELABEL]} \\ \hline \Gamma; L \vdash \mathbf{continue} \text{ } lab; : \tau/\emptyset \end{array}$$

$$\begin{array}{c} \text{[T-LABEL]} \\ \Gamma; L \cup \{lab\} \vdash s : \tau/\varepsilon \\ \varepsilon' = \varepsilon - (L \cup \{lab\}) \\ \hline \Gamma; L \vdash lab : s : \tau/\varepsilon' \end{array}$$

[T-SWITCH]

$$\begin{array}{c} \Gamma \vdash e : \sigma \\ \forall i. \Gamma; L \cup \{\epsilon\} \vdash c_i : \sigma, \tau/\varepsilon_i \\ \Gamma; L \cup \{\epsilon\} \vdash cd : \sigma, \tau/\varepsilon \\ \varepsilon \neq \mathbf{return} \vee \exists i. \varepsilon_i \cup \emptyset \neq \emptyset \\ \varepsilon' = (\varepsilon \cup \bigcup_i \varepsilon_i) - (L \cup \{\epsilon\}) \\ \hline \Gamma; L \vdash \mathbf{switch} \text{ } (e) \text{ } \{ \bar{c} \text{ } cd \} : \tau/\varepsilon' \end{array}$$

[T-SWITCHRETURN]

$$\begin{array}{c} \Gamma \vdash e : \sigma \\ \forall i. \Gamma; L \cup \{\epsilon\} \vdash c_i : \sigma, \tau/\varepsilon_i \\ \forall i. \varepsilon_i \cup \emptyset = \emptyset \\ \Gamma; L \cup \{\epsilon\} \vdash cd : \sigma, \tau/\mathbf{return} \\ \hline \Gamma; L \vdash \mathbf{switch} \text{ } (e) \text{ } \{ \bar{c} \text{ } cd \} : \tau/\mathbf{return} \end{array}$$

Case checking

$$\boxed{\Gamma; L \vdash cd : \sigma, \tau/\varepsilon}$$

$$\begin{array}{c} \text{[T-CASE]} \\ \Gamma \vdash e : \sigma \\ \Gamma; L \vdash ss : \tau/\varepsilon \\ \hline \Gamma; L \vdash \mathbf{case} \text{ } e : ss : \sigma, \tau/\varepsilon \end{array} \quad \begin{array}{c} \text{[T-DEFAULT]} \\ \Gamma; L \vdash ss : \tau/\varepsilon \\ \hline \Gamma; L \vdash \mathbf{default} : ss : \sigma, \tau/\varepsilon \end{array}$$

Expression checking

$$\boxed{\Gamma \vdash e : \tau}$$

$$\frac{[\text{T-INT}]}{\Gamma \vdash n : \mathbf{int}}$$

$$\frac{[\text{T-DOUBLE}]}{\Gamma \vdash r : \mathbf{double}}$$

$$\frac{[\text{T-VARREF}]}{\Gamma \vdash x : \tau}$$

$$\frac{[\text{T-ASSIGN}]}{\Gamma \vdash x = e : \tau}$$

$$\frac{[\text{T-LOAD}] \quad m = 2^k - 1 \quad \Gamma(x) = \mathbf{array}_{\tau}^n \quad \Gamma \vdash e : \mathbf{int}}{\Gamma \vdash x[(e \ \& \ m) \gg n/8] : \tau}$$

$$\frac{[\text{T-STORE}] \quad m = 2^k - 1 \quad \Gamma(x) = \mathbf{array}_{\tau}^n \quad \Gamma \vdash e_1 : \mathbf{int} \quad \Gamma \vdash e_2 : \tau}{\Gamma \vdash x[(e_1 \ \& \ m) \gg n/8] = e_2 : \tau}$$

$$\frac{[\text{T-FUNCALL}] \quad \Gamma(f) = (\bar{\sigma}) \rightarrow \tau \quad \forall i. \Gamma \vdash e_i : \sigma_i}{\Gamma \vdash f(\bar{e}) : \tau}$$

$$\frac{[\text{T-FFI}] \quad \Gamma(f) = \mathbf{function} \quad \forall i. \Gamma \vdash e_i : \mathbf{jsval}}{\Gamma \vdash f(\bar{e}) : \mathbf{unknown}}$$

$$\frac{[\text{T-CONDITIONAL}] \quad \Gamma \vdash e_1 : \mathbf{boolish} \quad \Gamma \vdash e_2 : \tau \quad \Gamma \vdash e_3 : \tau}{\Gamma \vdash e_1 ? e_2 : e_3 : \tau}$$

$$\frac{[\text{T-PAREN}] \quad \forall i \leq n. \Gamma \vdash e_i : \tau_i}{\Gamma \vdash (\bar{e}) : \tau_n}$$

$$\frac{[\text{T-UNOP}] \quad \mathit{unop} : \_ \wedge (\sigma) \rightarrow \tau \wedge \_ \quad \Gamma \vdash e : \sigma}{\Gamma \vdash \mathit{unop} \ e : \tau}$$

$$\frac{[\text{T-BINOP}] \quad \mathit{binop} : \_ \wedge (\sigma_1, \sigma_2) \rightarrow \tau \wedge \_ \quad \Gamma \vdash e_1 : \sigma_1 \quad \Gamma \vdash e_2 : \sigma_2}{\Gamma \vdash e_1 \ \mathit{binop} \ e_2 : \tau}$$

$$\frac{[\text{T-SUB}] \quad \Gamma \vdash e : \sigma \quad \sigma <: \tau}{\Gamma \vdash e : \tau}$$

$$\frac{[\text{T-CAST}] \quad \Gamma \vdash e : \mathbf{double}}{\Gamma \vdash \sim\sim e : \mathbf{signed}}$$