LambdaJS quick reference

Marek Materzok

November 27, 2015

1 Syntax

$$
\begin{array}{rcl}
e & ::= & x \mid l \mid e\,(e, \ldots, e) \mid \textbf{func}\,(x, \ldots, x)\,e \mid un\,e \mid e\,bin\,e \mid e;\ e \mid e\,;;\,e \mid \\
 & & \textbf{let}\,(x = e)\,e \mid \textbf{rec}\,(x = e)\,e \mid \textbf{if}\,(e)\,e\,\textbf{else}\,e \mid \\
 & & \textbf{label}\,i : e \mid \textbf{break}\,i\,e \mid \textbf{throw}\,e \mid \textbf{try}\,e\,\textbf{catch}\,e \mid \textbf{try}\,e\,\textbf{finally}\,e \mid \\
 & & e\,[e\langle pa \rangle] \mid e\,[e\langle pa \rangle = e] \mid e\,[\textbf{delete}\,e] \mid e\,[\langle oa \rangle] \mid e\,[\langle oa \rangle = e] \mid \\
 & & \{[oa : e, \ldots]\,s : \{pa : e, \ldots\}, \ldots\} \\
l & ::= & b \mid n \mid s \mid \textbf{undef} \mid \textbf{null} \mid \textbf{empty} \\
b & ::= & \textbf{true} \mid \textbf{false} \\
n & ::= & \textit{IEEE floating-point numbers} \\
s & ::= & \textit{UTF-16 encoded strings} \\
un & ::= & \textbf{typeof} \mid \textbf{strlen} \mid \textbf{is-primitive} \mid \textbf{is-closure} \mid \ldots \\
bin & ::= & + \mid - \mid * \mid / \mid \% \mid < \mid == \mid === \mid \textbf{has-own-property} \mid \\
 & & +_s \mid <_s \mid \ldots \\
pa & ::= & \textbf{value} \mid \textbf{writable} \mid \textbf{getter} \mid \textbf{setter} \mid \textbf{enumerable} \mid \textbf{configurable} \\
oa & ::= & \textbf{proto} \mid \textbf{class} \mid \textbf{extensible} \mid \textbf{code} \mid \ldots
\end{array}
$$