Mozilla Extensions to ECMAScript, 3rd Edition

Prepared by Allen Wirfs-Brock based upon Mozilla online doucmentation

August 2007

Feature	1st JavaScript Version	Other Implementations	Explanation	Notes
Function.prototype.name property	?	?	ReadOnly property that provides the name of a function (or empty string if the function is anonymous)	Doug Crockford wants to add this to "ES3.1" along with a property that provides the names of the formal parameters
String HTML wrapper methods: anchor, big, blink, bold,fixed,fontcolor, fontsize,italics,link,small, strike,sub,sub	1.0	?	Wrappers strings with various HTML tags	Predates ECMAScript but not included in the standard. Essentially legacy functions that nobody thinks should be standardized.
Object.prototype toSource method	1.3	?	Generates a crude string serialization of an object	Predates ECMAScript 3 but not included in the standard
Conditional function definition	1.5		Named function expressions are dynamically bound to the corresponding variable in the surrounding scope when execute	JScript treats them as Function Declarations instead of function expressions. Neither conforms to ECMA-262. JavaScript contends that Jsript's behavior is a bug

multiple catch clauses	1.5	?	try { }
			catch (e if e =="InvalidNameException) {
			/* handler for invalid name exception */ }
			catch (e) {
			/* default changle */ }	

Feature	1st JavaScript Version	Other Implementations	Explanation	Notes
getter/setter properties	1.5	Safari 3 Opera 9.5	property access may be implemented via user defined methods	works for both . and [ ] access. Property enumeration, existence, deletion. Treat getter/setter pairs as a single property
getters/setters definition syntax in object literals	1.5	Safari 3 Opera 9.5	syntax for defining properties in object initializers {_x: 0, get x() {return this._x;} , set x(arg) {this._x=arg}}	

__defineGetter__
__defineSetter__
	1.5	Safari 3
		Opera 9.5
			methods for dynamically adding getter/setter methods for an object:
			var d = Date.prototype;
			d.__defineGetter__("year",
			function() { return this.getFullYear(); });
			d.__defineSetter__("year",
			function(y) { this.setFullYear(y); });	

const definitions	1.5	?	define a constant binding. Like var but not assignable or redeclarable: const g = 5; g = 10; /* does not change the value of g*/	
Array indexOf	1.6	?	find first occurance of a value	"Array extras" see http://www.webreference.com/programming/javascript/ncz/column4/index.html Many third party libraries add these functions to Array.prototype

Feature	1st JavaScript Version	Other Implementations	Explanation	Notes
Array lastIndexOf	1.6	?	find last occurance of a value	"Array extras"
Array every()	1.6	?	evaluate a function on every element in an array, but stop when the function does not return true	"Array extras"
Array filter()	1.6	?	collect (into a new array) all the elements of an array that satisfy a predicate function	"Array extras"
Array forEach()	1.6	?	evaluate a function on every element of an array	"Array extras"
Array map()	1.6	?	collect (In a new array) the results of evaluating a function on every element of an array	"Array extras"
Array some()	1.6	?	evaluate a function on every element on an array, but stop when the function returns true	"Array extras"

Feature	1st JavaScript Version	Other Implementations	Explanation	Notes
Array/String generic methods	1.6	?	Many array and string functions can be be applied to any "array like" object by passing the object as the first argument	Mozilla's documentation isn't explicit about the exact set of functions but http://www.snailshell.de/blog/archives/2005/10/entry_9.html says: Array: concat, every, filter, forEach, indexOf, join, lastIndexOf, map, pop, push, reverse, shift, slice, some, sort, splice, unshift String: charAt, charCodeAt, concat, indexOf, lastIndexOf, localeCompare, match, quote, replace, search, slice, split, substr, substring, toLocaleLowerCase, toLocaleUpperCase, toLowerCase, toUpperCase
for each in statement	1.6	?	iterate over the values of an object's properties for each (x in obj) { …}	derived from E4X
partial E4X support	1.6	?		Can't find any Mozilla documentation about what is actually there

Feature	1st JavaScript Version	Other Implementations	Explanation	Notes
iterators	1.7	?	Iterator() global function, __iteratator__ property convention, StopIteration exception, iterators are generators, for in/for each in statements use iterators	

generators/yield statement	1.7	?	Co-routine like functions with embeded yield statements:
			Function intGen(begin, end) {
			for (var I = begin; i<=end; ++i) yield I; }	

array comprehensions	1.7	?	array initializers using iterators and conditionals: var evens = [I for (I in range(0.20) if (even(i))]	Similar to Python comprehensions Actual syntax not documented
let statement	1.7	?	define a code block with local variables let (x=1, y=2) { … }	
let expressions	1.7	?	define a single expression code block with local variables	
let definitions	1.7	?	define individual local variables within a code block {… let x =1, y=2;…;let z;…}	
let definition in for statement	1.7	?	use let to define control variables scoped to a single for loop: for (let i=0; i<10; i++) {…}	
destructuring assignment	1.7	?	assignment can be used to destructure an array or object into multiple local variables: var first, second, third; [first,second,third] = [1,2,3]; It's also useful for functions that want to return multple values: function f() {return [a ,b]}; [x,y] = f();	very little actual documentation for destructuring assignment features

Feature	1st JavaScript Version	Other Implementations	Explanation	Notes
destructuring var	1.7	?	destructuring assignment can be used as an initializer in var/let definitions: var [first,second,third]= ["a","b","c"]	
destructuring for	1.7	?	destructuring assignment can be used to define the iteration variable(s) of a for statement: for (let [key,value] in obj) {…}	
expression closure shorthand	1.8	none	function expressions whose body is a single return statement can be abbreviated such as: function (x) x + 1 instead of function (x) {return x+1;}	JavaScript 1.8 not yet final, see http://ejohn.org/blog/javascript-18-progress/
generator expressions	1.8	none	define single generators using array comprehension-like syntax	
Array reduce()	1.8	none	evaluate a function on every element of an array and accumulate the result values	"More Array Extras"
Array reducedRight()	1.8	none	like reduce but in reverse order	"More Array Extras"