JavaScript-Livecode Cheat Sheet

Comments

Comments allow you to add explanations and annotations to your code.

```
// These
/* are
 commented
 out */
```

```
-- these
# are
// all
/* commented
 out */
```

Variables

Variables are used to to store information, the stored value can be changed or accessed when you need it.

```
var myVar;
myVar = "str";
myVar = 1;

var arr = new Array();
arr["key"] = "val";
```

```
local tVar
put "str" into tVar
put 1 into tVar

put "val" into tVar["key"]
```

Constants

Constants store a value that is defined at the point of declaration and never changes.

```
const FOO = 15;
```

```
constant kFoo = 15
```

String Processing

These examples show how string values can be manipulated.

```
# General
str = 'a' + str;
str = str.slice(1);
str = str.replace("_", "-")

# Regex
var found = /[0-9]/.exec("1");
var num = found[1];

str.split("\n").filter(function(elem) {
return pattern.exec(elem) != NULL;
});
```

```
// General
put "a" before tVar
delete char 1 of tVar
replace "_" with "-" in tVar

// Regex
matchText("1", "([0-9])", tN) is true
tN is 1

filter lines of tVar with regex pattern
tPattern
```

Control Structures

Control structures are used to control what code is executed and how many times.

```
for (var i=0; i < text.length; i++) {
   char = text.charAt(i);
}

for (var i=0; i < 10; i++) {
}

while (x < 10) {
   x--;
}

if (value) {
} else if (other) {
} else {
}

switch (value) {
   case "a":
      break;
   default:
      break;
}
```

```
repeat for each char tChar in tVar
end repeat

repeat 10
end repeat

repeat with x = 1 to 10
end repeat

repeat while x < 10
   subtract 1 from x
end repeat

if true then ... else ...

if tVar then
else if tOther then
else
end if

switch tVar
   case "a"
      break
   default
      break
end switch
```

Sorting

These examples show how to sort items and lists.

```
var list = [5, 2, 3, 1, 4]
list.sort();
-> list == [1, 2, 3, 4, 5]
list.reverse();
-> list == [5, 4, 3, 2, 1]

var data = [[6, 1], [8, 3], [2, 2]];
data.sort(function(a,b) {
return a[2] - b[2]
});
-> data == [[6, 1], [2, 2], [8, 3]]
```

```
local tList
put "5,2,3,1,4" into tList
sort items of tList ascending numeric
 -> tList is "1,2,3,4,5"
sort items of tList descending numeric
 -> tList is "5,4,3,2,1"

local tData
put "6,1:8,3:2,2" into tData
set the lineDelimiter to ":"
sort lines of tData ascending numeric by
item 2 of each
 -> tData is "6,1:2,2:8,3"
```

Operators

Operators are ways of combining values such as boolean values, numbers or strings, to produce other values.

```
// Logical
true && false == false
true || false == true
!false == true

// String
"foo" + "bar" == "foobar"
var strs = ['foo','bar'];
strs.join(" ") == "foo bar"

"str".startsWith("st");
"str".endsWith("g");

// Chunks
"str".charAt(4) == "n"

var items = "a,b,c".split(",");
items[2] == "c"

var words = "hi there".split(" ");
words[0] == "hi"

var lines = "a\nb".split("\n");
lines[2] == "b"

var lines = "a,b,c".split("\n")
var items = lines[1].split(",")
items[1].charAt(0) == "a"
```

```
// Logical
true and false is false
true or false is true
not false is true

// String
"foo" & "bar" is "foobar"
"foo" && "bar" is "foo bar"
"str" begins with "st"
"str" ends with "g"

// Chunks
char 5 of "str" is "n"
item 3 of "a,b,c" is "c"
word 1 of "hi there" is "hi"
line 2 of "a" & return  & "b" is "b"

// Compound chunks
char 1 of item 1 of line 1 of "a,b,c" is "a"
```

User Input / Notification

These examples show how to pop up information dialogs, or prompts for user input.

```
var name = prompt("What is your name?");

alert("Something");
```

```
ask "What is your name?"
put it into tName

answer  "Something"
```

Array Processing

These examples show how array values can be manipulated.

```
# Split / combine
var list = "a,b,c".split(",")
list[1] is "b"
list = list.join(",");
list == "a,b,c"

for (var key in array) {
# Do something with array[key];
}

# Length
array.length();
```

```
// Split / combine
put "a,b,c" into tVar
split tVar by ","
tVar[2] is "b"
combine tVar with ","
tVar is "a,b,c"

// Iteration
repeat for each key tKey in tArray
 -- Do something with tArray[tKey]
end repeat

repeat for each element tElement in tArray
end repeat

// Length
the number of elements in tArray
```

Custom Handlers

A custom handler is a function or command that you define yourself.

```
function foo(param) {
}
// foo(value)
```

```
function foo pParam
end foo
// get foo(tVar)

command bar pParam
end bar
// bar 5
```

Event Handlers

An event handler is a hander that is triggered when an event occurs, such as the use of the mouse or keyboard.

```
# Mouse
function handleMouseUp {
}
<button onmouseup="handleMouseUp" />

function handleMouseDown {
}
<button
onmousedown="handleMouseDown" />

function handleMouseMove {
}
<div onmousemove="handleMouseMove"
/>

# Keyboard
function handleKeyUp {
}
<input onkeyup="handleKeyUp" />

function handleKeyDown {
}
<input onkeydown="handleKeyDown" />
```

```
// Mouse
on mouseUp pButton
end mouseUp

on mouseDown pButton
end mouseDown

on mouseMove
end mouseMove

// Keyboard
on keyDown pKey
end keyDown

on keyUp pKey
end keyUp
```