JavaScript - LiveCode Cheat Sheet

Comments

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

JavaScript	LiveCode
`// 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.

JavaScript LiveCode

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

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

```
var arr =
{};
arr["key"]
= "val";
```

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

Constants

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

JavaScript LiveCode

```
const
FOO =
15;
```

```
constant
kFoo = 15
```

Control Structures

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

JavaScript LiveCode

```
for (var i=0; i
< text.length;
i++) {
    char =
text.charAt(i);
}
for (var i=0; i
< 10; i++) {
}


while (x > 1) {
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 >
1
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
```

Operators

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

JavaScript LiveCode

```
//
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"
"string"
begins
with
"st"
"string"
ends
with "g"


//
Chunks
char 5
of
"string"
is "n"
item 3
of
"a,b,c"
is "c"
word 1
of "hi
there"
```

```
// Logical
true && false == false
true || false == true
!false == true
// String
"foo" + "bar" == "foobar"
var strs = ['foo','bar'];
strs.join(" ") == "foo
bar"


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


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


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


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


var lines =
"anb".split("n");
lines[2] == "b"
```

```
is "hi"
line 2
of "a" &
return
& "b" is
"b"
```

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

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

String Processing

These examples show how string values can be manipulated.

JavaScript

LiveCod

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

Regex

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

```
// (
put
befo
del
of t
rep
with
tVar
//
matc
"([0
is t
tN i
```

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

```
filte
tVar w
patter
```

Array Processing

These examples show how array values can be manipulated.

JavaScript

LiveCode

```
# 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
```

```
// S
com
put
into
spl
by "
tVar
"b"
com
with
tVar
"a,b
// 
rep
each
in t
-- 
som
wit
tArr
end

rep
each
tEle
tArr
end
```

```
array.length();
```

```
// Leng
the nur
element
```

Sorting

These examples show how to sort items and lists.

JavaScript

LiveCode

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

```
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"
```

```
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 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"
```

User Input / Notification

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

JavaScript LiveCode

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

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

```
alert("Something");
```

```
answer
"Something"
```

Custom Handlers

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

JavaScript LiveCode

```
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.

JavaScript

LiveCode

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

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

Keyboard

```
function handleKeyUp {
}
<input onkeyup="handleKeyUp"
/>
```

```
// Mouse
on
mouseUp
pButton
end
mouseUp
on
mouseDow
pButton
end
mouseDow

on
mouseMov
end
mouseMov

//
Keyboar
on
keyDown
pKey
end
keyDown
```

```
on keyUp p
end keyUp
```

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

Offline (Leave a message)