Storage

By Michael Wiltbank

Storage

- The two forms of storage are sessionStorage and localStorage
- Both are client facing forms of storage

localStorage.setItem(…)

Methods

- clear( )
- getItem(*keyname*)
- key(*key*) *user-agent defined*
- removeItem(*keyname*)
- setItem(*keyname, keyvalue*)

Properties

- .length

Loops

- **for -** loops a specified number of times
  - for (var i = 0; i < 10; i++) { … }
- **for/in -** loops through the properties of an object
  - for (var x in object) { … }
- **while** - loops while a specified condition is true
  - while(bool) { … }
- **do/while** – after one do, loops while a specified condition is true
  - do { … } while (bool)

Conditional Statements

- **if -** if a condition is true then a block of code will be executed
- **else** - if the condition is false to a block of code will be executed
- **else if** - sets up a new condition to test, if the first condition is false
- **switch** - to specify many alternative blocks of code to be executed

```
if ( bool ) { … }
else { … }
else if ( bool ) { … }
switch (condition) {
```

```
switch (condition) {
case n : …
case n : …
default: …    }
```

Functions

function *functionName*(*parameters*) {
  code to be executed separated by semicolons
}

- Functions are defined with keyword ‘function’
- Functions can be defined as an expression
- Functions contain properties and methods
- Functions are objects

Variables

var *identifier* = data

- Declare variables with keyword ‘var’
- JavaScript variables are containers for holding data
- Each variable needs a unique identifier
- Many data types but most common are strings and numbers

var *identifier* = *data* + *variable* + *data*

Parameters

```
function functionName(parameter1, parameter 2, ...) {
  code to be executed
}
```

- Parameters are the names listed in the function definition
- Arguments are the actual values of the passed to the function
- Arguments create an object
- Calling a function with missing parameters sets them to undefined
- If too many parameters then you can reach them with the arguments object

Arrays

var arrayName = [“data1”, “data2”, “data3”];

- An index number is created starting at ‘0’
  - Array[1] == “data2”
- Arrays are objects
- Array Variables can can be objects
- .push (array.push(“data4”);)
- .length (*or Math.max.apply(arrayName) or Math.min.apply(arrayName)*)
- sort() and reverse() and random()

Associative Arrays

- Arrays with named indexes are Associative Arrays

```
var arrayName = [ ];
arrayName[“indexName”] = arrayVariable;
```

- javaScript does not support named indexes

Objects

var family = {father:"Michael", mother:"Ashley", child: "Grant", established: 2006 };

objectName.propertyName or *objectName*["*propertyName*"]

objectName.methodName( )

- An object has properties and methods
- Object properties can also be objects