Type System

15 July 2014 14:19

```
var a = 10;
console.log(typeof(a))

a = "GS, Bangalore";
console.log(typeof(a));

a = false;
console.log(typeof(a));

console.log(typeof(b));

var c=null;
console.log(typeof(c));

var x = 10;
console.log(typeof(x));

var y = "100";
console.log(x+parseInt(y));

var z = "100isIBMsAge";
console.log(parseInt(z));

var z1 = 10 * z;
console.log(z1);
console.log(typeof(z1));
```

Comparison Operators

15 July 2014 14:22

```
var x = 10;
var y = "10";

console.log(x == y);
console.log(x === y);

var a = 20;
var b = "20";

console.log(a != b);
console.log(a !== b);
```

```
>>> var x = 10; var y = "10";  console.log(x == y); ...0";  console.log(a != b); console.log(a !== b);
true
false
false
true
```

`

Arrays in Javascript

15 July 2014 14:35

```
var names = [];
var names = new Array();

names = ["A", "B", "C", "D", "E"];

//for(var item in names) {
//    console.log(names[item]);
//}

console.log('Using Foreach ...');

for each(var name in names)
   console.log(name);

names[10] = "GS";

console.log(names.length);
console.log(names);

delete names[10];

console.log(names.length);
console.log(names);

var values = [];

for (var counter=0;counter<=10;counter++)
   values.push(counter.toString());

console.log(values);

var totalNoOfItems = values.length;

for(var counter=0;counter<totalNoOfItems;counter++)
   console.log(values.pop());

console.log(values);

var complexArray =
   [
      [ 10, 20, 30 ],
      [ "A", "B", "C", "D", "E" ],
      { id: 10, name: 'Ram' }
   ];

console.log(complexArray);
```

Error and Exception Handling

15 July 2014 14:42

```
var v = v || 10;

try {
   if(v <= 10)
      throw new Error("Invalid Value Initialized!");
} catch(error) {
   console.log("Error Occurred, Details : " + error.message);
} finally {
   console.log('Finally Program Completed!');
}

// GOOD PRACTICE
console.log('Using custom objects in TRY .. CATCH ..');

try {
   if(v <= 10)
      throw {
         id: 10,
         message: 'Invalid Value',
         system: 'browser'
      };
} catch(error) {
   console.log(error.id + ', ' +
      error.message + ', ' + error.system);
}
```

Functions - Simple

15 July 2014 14:53

```
function add(x, y) {
   var result = 0;

   if(x && y)
      result = x + y;

   return result;
}

var output = add(10,20);
console.log(output);

var output1 = add(10);
console.log(output1);

var output2 = add(10,20,30,40,50);
console.log(output2);

var addEx = function() {
   var parameters = arguments;
   var result = 0;

   for each(var parameter in parameters)
      result += parameter;

   return result;
};

// This overwrites addEx function
//var addEx = function() {
//    console.log('Gone!');
//};

console.log(addEx(10,20,30,40,50));
console.log(addEx(10));
console.log('End of App!');
```

Self-Invoking Functions

15 July 2014 14:59

```
(function () {
   console.log('Initialization Completed ...');
})();


var obj1 = (
   function() {
      var obj = {id:10};
      console.log('Initialization Completed ...');
      return obj;
   })();

obj1.id = 20;

console.log(obj1.id);
```

Identifier function caller details

15 July 2014 15:05

```
function A() {
   console.log('A is working ...');

   var caller = A.caller;

   if(caller) {
      console.log(caller.toString());
      console.log('Who called me ? ... ' + caller.name);
   }
}

function B() {
   console.log('B is working ...');

   A();
}

B();

var sourceCode='function X() { return 10; } X();';
eval(sourceCode);
```

Functions that accept functions as callbacks

15 July 2014 15:16

```
var processOrder = function(
   orderId, orderDate, ccNumber, orderValue, ccValidationCallback) {

   var MIN_ORDER_VALUE = 1;

   var validation = orderId && orderDate && ccNumber && orderValue &&
      orderValue >= MIN_ORDER_VALUE;

   if(!validation)
      throw new Error("Invalid Arguments Specified!");

   var processingStatus = false;

   if(ccValidationCallback && typeof(ccValidationCallback) == 'function') {
      processingStatus = ccValidationCallback(ccNumber);

      console.log("Processing Completed ...");
   }

   return processingStatus;
};

var processingStatus = processOrder(
   'ORD10001', new Date(), 'VISA-1111-1212-1212', 1200,
   function(ccNumber) {
      var TOTAL_CC_LENGTH = 19;
      var validationStatus = ccNumber &&
         ccNumber.length == TOTAL_CC_LENGTH &&
         ccNumber.startsWith("VISA");

      return validationStatus;
   });

console.log('Processing Status : ' + processingStatus);
```

Functions that return functions

15 July 2014 15:23

```
function getData(downloadUrl) {
   console.log('Connecting the Server ... URL .. ' + downloadUrl);

   var buffer = 0;

   return function() {
     console.log('Returning Latest Data ...');

     return buffer++;
   };
};

var getDataRef = getData('http://services.gs.com/customers');

console.log('Data : ' + getDataRef());
console.log('Data : ' + getDataRef());
console.log('Data : ' + getDataRef());
console.log('Data : ' + getDataRef());
console.log('Data : ' + getDataRef());
console.log('Data : ' + getDataRef());
```

Understanding Closure

15 July 2014 15:32

```
function Iterate() {
   var parameters = arguments;

   if(!parameters)
     throw new Error('Invalid Arguments Specified!');

   var currentIndex = 0;
   var totalItems = parameters.length;

   return {
     get: function() {
       return parameters[currentIndex++];
     },
     moveNext: function() {
       var isValueAvailable = currentIndex >= 0 &&
           currentIndex < totalItems;

       return isValueAvailable;
     }
   };
};

var iterator = Iterate(10,20,30,40,50);

try {
  while(iterator.moveNext()) {
    console.log(iterator.get());
  }
} catch(error) {
  console.log('Error Occurred .. ' + error.message);
} finally {
  console.log('Iterator completed!');
}
```