Default Values to Parameters

16 July 2014 10:28

```
function generateRandomValue(initValue) {
  var minimum = initValue || 100;
  var maximum = 1000;

  var randomValue = Math.floor(
    Math.random() * (maximum - minimum) + minimum);

  return randomValue;
}

function process(
  x = generateRandomValue(),
  y = generateRandomValue(x)) {

    console.log('Value of x and y are : ' + x + ', ' + y);
}

process();
process(10);
process(10,20);
```

Problems with var declaration

16 July 2014 10:31

```
function problemsWithVar() {
  var x = 10;

  console.log('value of x is : ' + x);

  {
    var x = 100;
    console.log('value of x (inside) is : ' + x);
  }

  console.log('value of x (outside) now is : ' + x);

  for (var y = 0;y<10;y++) {
    var m = 100;
  }

  console.log('value of y (forloop) : ' + y);
  console.log('m also available, value is : ' + m);
}

problemsWithVar();
```

Constants

16 July 2014 10:35

```
'use strict';

const m = 10;

console.log('value of m is : ' + m);

m++;

console.log('value of m now is : ' + m);

const obj = {
  id: 100
};

obj = { name: 'Ramkumar' };
obj.id = 200;

console.log(obj.id + ', ' + obj.name);
```

Applying 'use strict' option in the function | script declaration enables Javascript VM to apply strict rules to the code block. In the above snippet, m++ causes an exception, because m is constant.

Let

16 July 2014 10:43

```
function process() {
  let m = 10;
  var a = 10;

  console.log('value of m is : ' + m);

  {
    let m = m;
    var a = a;

    m++;

    console.log('value of m (inside) : ' + m);
    console.log('value of a is : ' + a);
  }

  console.log('value of m (outside) : ' + m);

  for(let x=10;x<=100;x++);

  try {
    console.log('value of x is : ' + x);
  } catch(error) {
    console.log(error.message);
  }

  let(x=10, y=20) {
    console.log(x + ', ' + y);
  }
}

process();
```

Generator

16 July 2014 10:55

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

  console.log('Processing Started ...');
  yield parameters[0];

  console.log('Processing Continue ... 1');
  yield parameters[1];

  console.log('Processing Continue ... 2');
  yield parameters[2];
}

var generator = getValues(10,20,30);

try {
  for each(var value in generator)
    console.log(value);
} catch(error if error instanceof StopIteration) {
  console.log('Iteration Completed!');
}finally {
  if(generator)
    generator.close();
}
```

```
function getValues() { var parameters = argumen...nally
{ if(generator) generator.close(); }
Processing Started ...
10
Processing Continue ... 1
20
Processing Continue ... 2
30
undefined
```

Iterators

16 July 2014 11:01

```
function Range(minimum,maximum) {
  this.minimum = minimum;
  this.maximum = maximum;
};

var range = new Range(10,20);

var iterator = new Iterator(range);

for each(let [key,value] in iterator)
  console.log(key + ', ' + value);

Range.prototype.__iterator__ = function() {
  for(let currentValue=this.minimum;
      currentValue<=this.maximum;currentValue++) {
    yield currentValue;
  }
};

for each(var property in range) {
  console.log(property);
}
```

Output

```
minimum, 10
maximum, 20

10
11
12
13
14
15
16
17
18
19
20
```

Assignment Improvements - 1

16 July 2014 11:23

```
var x = 10;
var y = 20;

[x,y] = [y,x];

console.log(x + ', ' + y);

function process() {
  var parameters = arguments;

  var result =
    [
      parameters[0],
      parameters[1],
      parameters[2]
    ];

  return result;
}

var values = process(10,20,30);

console.log(values[0] + ', ' +
            values[1] + ', ' + values[2]);

var [a, b, c] = process(10,20,30);

console.log(a + ', ' + b + ', ' + c);

var [a,,c] = process(10,20,30);
console.log(a + ', ' + c);
```

Assignment Improvements - 2

16 July 2014 11:31

```
var customer = {
  id: 1,
  name: 'Ankush',
  address: 'Bangalore',
  credit: 23000,
  status: true,
  location: {
    country: 'India'
  }
};

function getDetails(customerObj) {
  console.log(customerObj.id + ', ' + customerObj.status);

  customerObj.status = false;
};

getDetails(customer);

console.log(customer.status);

function getDetailsEx(
  {name: cname, status: cstatus}) {

    console.log('Getting Details ... ' + cname + ', ' +
            cstatus);
}

getDetailsEx(customer);

var customers =
    [
      {
        id: 1, name: 'Raj', location: 'Mumbai'
      },
      {
        id: 2, name: 'HD Samy', location: 'Bangalore'
      },
      {
        id: 3, name: 'LP Yadav', location: 'Patna'
      }
    ];

for each(var customer in customers) {
  console.log(customer.id + ', ' + customer.name);
};

for each(let { name: cname, location: address } in customers) {
  console.log(cname + ', ' + address);
}
```

Arrow Functions

16 July 2014 11:38

```
// LEGACY
function Execute(callback) {
  var result = callback(10, 20);

  return function(x, y) {
    return result + x + y;
  };
};

var output = Execute(
  function(a, b) {
    return a * b
  }) (100,100);

console.log(output);


// MODERNIZED
function ExecuteEx(callback) {
  return (x, y) => callback(10,20) + x + y;
};

var outputReference = ExecuteEx((a,b) => {
    return a*b;
  });

var output = outputReference(100,100);

console.log(output);
```

Object Properties Enhancements

16 July 2014 11:52

```
var obj = {
  id: 10,
  name: 'Rajesh',
  location: 'Bangalore',
  licenseKey: 'L8349834',
  latitude: 100,
  longitude: 200
};

obj.id = 20;

delete obj.name;

for each(var property in obj)
  console.log(property);

var customer = {};

Object.defineProperty(customer, 'id', {
  writable: false,
  value: 10,
  configurable: false,
  enumerable: true
});

Object.defineProperty(customer, 'licenseKey', {
  writable: false,
  value: 'L834983498',
  configurable: false,
  enumerable: false
});

customer.id = 200;
delete customer.id;

console.log(customer.id);

for each(var property in customer)
  console.log(property);

function Customer(id, name, address, licenseKey) {
  var customerId = id;
  var customerName = name;
  var customerAddress = address;
  var customerLicenseKey = licenseKey;

  Object.defineProperty(this, "Id", {
    writable: false,
    value: customerId,
    configurable: false,
    enumerable: true
  });

  Object.defineProperty(this, "Name", {
    get: function() { return customerName; },
    set: function(newValue) { customerName = newValue; },
    configurable: false,
    enumerable: true
  });

  Object.defineProperty(this, "Address", {
    get: function() { return customerAddress; },
    set: function(newValue) { customerAddress = newValue; },
    configurable: false,
    enumerable: true
  });

  Object.defineProperty(this, "LicenseKey", {
    get: function() { return customerLicenseKey; },
    set: function(newValue) { customerLicenseKey = newValue; },
    configurable: false,
    enumerable: false
  });
};

var newCustomer = new Customer(
  1, 'Northwind', 'Bangalore', 'L849834');


console.log(newCustomer.Id + ', ' +
            newCustomer.Name + ', ' +
            newCustomer.Address + ', ' +
            newCustomer.LicenseKey);

newCustomer.Id = "C849834";

var iterator = new Iterator(newCustomer);
for each(let [pk, pv] in iterator)
  console.log(pk + ', ' + pv);

newCustomer.watch('Address',
                  function(obj, newValue, oldValue) {
                    console.log('Address has been Changed!');
                    console.log(obj);
                    console.log(newValue);
                    console.log(oldValue);
                  });

newCustomer.Address = 'Chennai';
```

Expression (Generator Expression | Array Expression)

16 July 2014 12:06

```
function getValues() {
  var valuesToReturn = [10,20,30,40,50,60];

  return valuesToReturn;
};

var values = getValues();
var processedValues = [];

for each(var value in values) {
  if(value % 3 == 0)
    processedValues.push(value * value);
}

console.log(processedValues);

var newProcessedValues =
    (v * v for each(v in getValues()) if (v % 3 == 0));

for each(var processedValue in newProcessedValues)
  console.log(processedValue);

function getData() {
  var customers = [
    {id: 11, name: 'Raj', status: true },
    {id: 12, name: 'Ramesh', status: true },
    {id: 13, name: 'Rakesh', status: false },
    {id: 14, name: 'Mukhesh', status: true },
    {id: 15, name: 'Rajesh', status: false }
   ];

  return customers;
};

function isValidCustomer(customer) {
  return customer.status;
};

var processedCustomers =
    [customer
       for each(customer in getData())
       if(isValidCustomer(customer))];

console.log(processedCustomers);

function printValues() {
  return (
    val for each(val in arguments) if (val % 4 == 0));
};

var output = printValues(10,20,30,40,50,60);
```

```
for each(var processedValue in output)
  console.log(processedValue);
```

Array related Enhancements

16 July 2014 12:12

```
function Customer(id, name, status) {
  this.id = id;
  this.name = name;
  this.status = status;
};

var customers =
    [
      new Customer(1, 'Peter', true),
      new Customer(2, 'Jason', false),
      new Customer(3, 'Johnson', true),
      new Customer(4, 'Mira', false),
      new Customer(5, 'Neeraj', true),
      new Customer(6, 'Gopal', false)
    ];

customers.forEach(function(item, index) {
  if(item.status)
    item.creditLimit = 2300;
  else item.creditLimit = 0;
});

console.log(customers);

var mappedCustomers= customers.map(
  function(item) {
    return item.name.toUpperCase();
  });

console.log(mappedCustomers);

var filteredCustomers = customers.filter(
  function(item) {
    return item.status &&
      item.creditLimit >= 2000;
  });

console.log(filteredCustomers);
```