Chapter 37: Enumerations

Section 37.1: Enum definition using Object.freeze()

Version ≥ 5.1

JavaScript does not directly support enumerators but the functionality of an enum can be mimicked.

```
// Prevent the enum from being changed
const TestEnum = Object.freeze({
    One:1,
    Two:2,
    Three:3
});
// Define a variable with a value from the enum
var x = TestEnum.Two;
// Prints a value according to the variable's enum value
switch(x) {
    case TestEnum.One:
        console.log("111");
        break;

    case TestEnum.Two:
        console.log("222");
}
```

The above enumeration definition, can also be written as follows:

```
var TestEnum = { One: 1, Two: 2, Three: 3 }
Object.freeze(TestEnum);
```

After that you can define a variable and print like before.

Section 37.2: Alternate definition

The `Object.freeze()` method is available since version 5.1. For older versions, you can use the following code (note that it also works in versions 5.1 and later):

```
var ColorsEnum = {
    WHITE: 0,
    GRAY: 1,
    BLACK: 2
}
// Define a variable with a value from the enum
var currentColor = ColorsEnum.GRAY;
```

Section 37.3: Printing an enum variable

After defining an enum using any of the above ways and setting a variable, you can print both the variable's value as well as the corresponding name from the enum for the value. Here's an example:

```
// Define the enum
var ColorsEnum = { WHITE: 0, GRAY: 1, BLACK: 2 }
Object.freeze(ColorsEnum);
// Define the variable and assign a value
var color = ColorsEnum.BLACK;
```

```
if(color == ColorsEnum.BLACK) {
   console.log(color);    // This will print "2"
   var ce = ColorsEnum;
   for (var name in ce) {
     if (ce[name] == ce.BLACK)
       console.log(name);    // This will print "BLACK"
   }
}
```

Section 37.4: Implementing Enums Using Symbols

As ES6 introduced **Symbols**, which are both **unique and immutable primitive values** that may be used as the key of an `Object` property, instead of using strings as possible values for an enum, it's possible to use symbols.

```
// Simple symbol
const newSymbol = Symbol();
typeof newSymbol === 'symbol' // true

// A symbol with a label
const anotherSymbol = Symbol("label");

// Each symbol is unique
const yetAnotherSymbol = Symbol("label");
yetAnotherSymbol === anotherSymbol; // false


const Regnum_Animale    = Symbol();
const Regnum_Vegetabile = Symbol();
const Regnum_Lapideum   = Symbol();

function describe(kingdom) {

  switch(kingdom) {

    case Regnum_Animale:
        return "Animal kingdom";
    case Regnum_Vegetabile:
        return "Vegetable kingdom";
    case Regnum_Lapideum:
        return "Mineral kingdom";
  }

}

describe(Regnum_Vegetabile);
// Vegetable kingdom
```

The Symbols in ECMAScript 6 article covers this new primitive type more in detail.

Section 37.5: Automatic Enumeration Value

`Version ≥ 5.1`

This Example demonstrates how to automatically assign a value to each entry in an enum list. This will prevent two enums from having the same value by mistake. NOTE: Object.freeze browser support

```
var testEnum = function() {
    // Initializes the enumerations
    var enumList = [
        "One",
        "Two",
        "Three"
    ];
    enumObj = {};
    enumList.forEach((item, index)=>enumObj[item] = index + 1);

    // Do not allow the object to be changed
    Object.freeze(enumObj);
    return enumObj;
}();

console.log(testEnum.One); // 1 will be logged

var x = testEnum.Two;

switch(x) {
    case testEnum.One:
        console.log("111");
        break;

    case testEnum.Two:
        console.log("222"); // 222 will be logged
        break;
}
```