SVG 資料第 7 回目 (その 3)

配列のメソッドの利用

メディア専門ユニット I(SVG)

2016/6/6

配列のメソッド (1)

- Array() は引数の大きさを指定して新しい配列を作成する。各要素は undefined で初期化される。
- fill() は引数の値で配列の要素をすべてその値に設定する。

```
>A=Array(5);
(5) [undefined × 5]
>A.length;
5
>A.fill(1);
(5) [1, 1, 1, 1, 1]
```

配列のメソッド (2)

- 通常、配列の各要素について操作をするためには `for` 文などのループで処理を行う。
- JavaScript では `for` 文の代わりになるメソッドがいくつか定義されている。それぞれのメソッドは引数に関数を必要とする。
  - 引数に渡される関数には最大で 3 つの引数を渡される。
  - 一つ目の引数は配列の要素
  - 2 つ目の引数は配列のインデックス
  - 3 つ目の引数はメソッドが適用される配列自身である。これを利用すると元来の配列の要素を変更できる。
- 要素の値が `undefined` のところは実行されない。

配列のメソッド (3)–forEach(func)

- 与えられた関数を配列の各要素に対して実行
- 与えられた関数の戻り値は無視される。
- 途中でループの処理を中断できない。

配列のメソッド (3)–forEach(func) 実行例

```
 >A = [1,2,3,4];
(4) [1, 2, 3, 4]
>A[5]=5;
5
>A;
(6) [1, 2, 3, 4, undefined × 1, 5]
>A.forEach(function(V,i,Vs){
    Vs[i] = V*V;
    console.log(V*V);
});
1
4
9
16
25
undefined
>A;
[1, 4, 9, 16, undefined × 1, 25]
>undefined*undefined
NaN
```

- 配列の各要素を 2 乗している。
- 関数の 3 番目の引数 (元の配列) に代入しているので、実行後、要素の値が変化する。
- 要素の値が undefined のところは実行されていない。

配列のメソッド (4)–map(func)

配列の各要素に対して引数の関数を実行し、その戻り値で新しい配列を作成する。

```
>A=[1,2,3]
(3) [1, 2, 3]
>A[5]=5
5
>A;
(6) [1, 2, 3, undefined × 2, 5]
>>A.map(function(V){return V*V;});
(6) [1, 4, 9, undefined × 2, 25]
>A;
(6) [1, 2, 3, undefined × 2, 5]
```

- 要素を 2 乗した配列を作成している。
- undefined の要素はそのまま undefined
- 元の配列は変化していない。

配列のメソッド利用の例 (1)

ランダムに円を描く

```
1<?xml version="1.0" encoding="UTF-8" ?>
2<svg xmlns="http://www.w3.org/2000/svg"
3     xmlns:xlink="http://www.w3.org/1999/xlink"
4     width="100%" height="100%">
5  <title>ランダムな色と大きさを一定の時間で表示し続ける</title>
6  <script type="text/ecmascript" xlink:href="make-svg-elm.js"></script>
7  <script type="text/ecmascript">
```

配列のメソッド利用の例 (2)

ランダムに円を描く

```
 8//<![CDATA[
 9window.onload = function() {
10  let Unit = 15, xMin = 40, xMax = 400, yMin = 40, yMax = 300;
11  let Colors = ["red", "yellow", "blue", "pink"];
12  let Canvas = document.getElementsByTagName("g")[0];
13  let Objs = Array(20).fill(0).map(function() {
14      return MKSVGElm(Canvas,"circle",
15        {"opacity":0.6, "stroke-width":"2", "stroke":"black"}, {});
16  });
17  (function renew() {
18    Objs.forEach(function(Obj){
19      SetAttributes(Obj,{
20        "cx":getRandVal(xMin, xMax), "cy":getRandVal(yMin, yMax),
21        "r":Math.sqrt(getRandVal(1, 4))*Unit,
22        "fill":Colors[getRandVal(0, Colors.length)]},{});
23      });
24    window.setTimeout(renew,getRandVal(10,20)*100);
25  })();
26}
27function getRandVal(Min, Max) {
28  return Math.floor(Min+(Max-Min)*Math.random());
29}
30//]]></script>
31<g />
32</svg>
```

配列のメソッド利用の例–解説

- 以前のプログラムではグローバル変数であったものがすべてローカルな変数に変更されている (10 から１１行目)。
- 12 行目で円を登録する g 要素を求めている。
- 円を 20 個登録するために次のようなメソッドチェインを用いている。
  1. 大きさが 20 の配列を作成 (`Array(20)`)
  2. 要素の値はすべて `undefined` なので `fill` メソッドを利用して 0 に設定
  3. その配列に `map` を利用して`<circle>`要素を作成
  4. 作成する円の属性は共通のものだけ
- 登録した円は配列 `Objs` に保存
- 一定時間をに呼ばれる関数では配列 `Objs` に対して `forEach` メソッドを用いて属性値を変更

配列のメソッド (5)

- every(func)
  配列の各要素に引数の関数を実行し、その戻り値がすべて true のとき、true を返す。途中で false になると実行が打ち切られ、false を返す。
- some(func)
  配列の各要素に引数の関数を実行し、その戻り値がすべて false のとき、false を返す。途中で true になると実行は打ち切られ、true を返す。
- filter(func)
  配列の各要素に引数の関数を実行し、その戻り値が true になるものだけを集めた新しい配列を返す。

配列のメソッド (6)–reduce と reduceRight

reduce

- 引数は処理をする関数とオプションの引数
- オプションの引数は処理を開始するときの初期値
- 処理をする関数は 4 つの引数を取る。
  - 第 1 引数はそれまでの計算結果
    オプションの引数があるときは開始時の値がこれと同じになる
  - 第 2 引数は現在の要素の値
  - 第 3 引数はインデックス
  - 第 4 引数は繰り返し」を行う配列

reduceright は配列の要素の最後から先頭に向かって処理される

実行例


```
>A=[3,5,4,7]
(4) [3, 5, 4, 7]
>A.reduce(function(x,v){return x+v;});//初期値がないときは適当に
19    //要素の総和
>A.reduce(function(x,v){return x+v;},"");//初期値をから文字列に
"3547"     //文字列として繋がれる
>A.reduceRight(function(x,v){return x+v;},"");
"7453"    //reduceRight では並び順が逆になる
```

やってみよう

- 要素の値が整数である配列に対して、次のことを行うプログラムを作成せよ。
  - 各要素を 5 で割った余りを値に持つ新しい配列
  - 奇数である要素だけ選び出す
- 大きさが 10 の配列で 1 から 10 の要素を順に持つ配列を作成せよ
- 今までに作成したものを配列のメソッドを利用して書き直せ