DOM + JavaScript
によるドキュメント管理

プログラミング数学テキスト
（第2学年 理コース）

千葉商科大学付属高等学校
数学科　樽　正人

2017年9月

目 次

1　DOMによるドキュメント管理

1.1　DOMとは

DOMはW3Cにおいて次のように勧告されています。（原文掲載）

What is the Document Object Model?
The Document Object Model is a platform- and language-neutral interface that will allow programs and scripts to dynamically access and update the content, structure and style of documents. The document can be further processed and the results of that processing can be incorporated back into the presented page. This is an overview of DOM-related materials here at W3C and around the web.

これを（私）個人の解釈を入れて和文訳すると，次のようになります。

DOMって何?
DOMとは，パソコンなどの端末環境や言語に依存することのないインターフェースであり，プログラムやスクリプトが，文章の内容や構造，スタイルに動的にアクセスしたり更新したりすることを可能にします。DOMにより，文章はさらに進んだ処理が可能となり，その処理結果を，表示しているページに組み込むことが可能になります。以上は，W3Cやその関連のwebサイトに参照されているDOM関連資料の概要です。

DOMによって管理された文章は，適当なプログラム言語やスクリプト言語によって，動的にページへのアクセスや更新が可能になるとありますが，これは本来webページは，あらかじめ用意されたページやデータをインターネットを通じて取得するだけであったものが，DOMと適当なプログラム言語の組合せによってユーザー側で更新させることが可能になるという意味です。

1.2　DOM+JavaScript

JavaScriptとはHTMLに組込んでおき，イベント（ボタンをクリックなど）によって従来HTMLが苦手とした動きのあるページ処理を可能にするスクリプト言語です。当然ですが，この言語だけで分厚い本が一冊できるくらいの関数やsyntax(文法)があります。本授業で全てを学ぶことはできません。ここでは，DOMとの連携に欠かせない関数と多数のデータ解析の際に必須となる繰り返し構文，および条件によって処理を振り分ける条件分岐についてサンプルを見ながら勉強していくことにします。

1.2.1　IDによるDOMの管理

DOMによってオブジェクト化されたもの（インスタンスなどと呼ぶ）は，IDによって識別管理することができます。これを利用するための関数としてJavaScriptには，getElementById(”ID名”)があります。これによって，webの閲覧者がページを動的に更新させることが可能になります。次にその例を示します。

サンプル 1.2.1

サンプル 3.2.1 の
ブラウザ画面


```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<script language="javascript">
function chText()
{
   // 更新内容を出力するエリアを指定
   var result_area = document.getElementById("uname");

   // 更新する内容を取得
   var inputData = document.getElementById("input_name").value;

   // 更新内容を更新エリアに出力する処理
   result_area.innerHTML = inputData;
}
</script>
<title>ID による DOM の管理</title>
</head>
<body>
<article>
<section>
<h3>
こんにちは。<span id="uname">●●●</span>さん。
</h3>
<hr/>
<form>
 <input type="text" value="あなたの名前" id="input_name" name="input_name"/>
 <input type="button" value="名前の変更" onClick="chText();"/>
</form>
</section>
</article>
</body>
</html>
```

このページでは,「名前の変更」というボタンをクリックすると ”chText() ” という関数が呼び出され，ID で指定されたオブジェクトを拾い出し，画面が変遷するというものです。
このコードを理解して行くには，はじめから敷居が高いですので，次に示す例題から少しずつ進めましょう。

1.3 実践 JavaScript

1.3.1 条件分岐 if 文

条件を与え，その条件を満たすとき（**真**）と満たさないとき（**偽**）で処理を変えることを**条件分岐**といいます。JavaScript には if 文が用意されています。
syntax（記法）は

```
if(条件文){
真のときの処理
}
else{
偽のときの処理
}
```

サンプル 1.3.1　問題に正解した場合と不正解した場合で処理を変える例です。

新しい知識になりますが，関数 answerCheck() では引数として KOTAE を参照しています。これは，送信ボタンをクリックしたときに ID が answer のフィールドの内容を KOTAE という引数に代入していることになります。
また，sup という新しいタグが出てきますが，これは SmallUPper で上付き文字を表します。

比較演算子
A==B
(A と B が等しいければ真)

A!=B
(A と B が等しくなければ真)

A>B
(A の方が B より大きければ真)

A<B
(A の方が B より小さければ真)

A>=B
(A が B 以上であれば真)

A<=B
(A が B 以下であれば真)

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>if 文のサンプル</title>
<script language="JavaScript">
function ansCheck()
{
  // 左辺値を変数 Num1 に，右辺値を変数 Num2 に代入する。
  var Num1 = document.getElementById("left-var").value;
  var Num2 = document.getElementById("right-var").value;

  // 下の Num1 == Num2 は,『左辺値と右辺値は等しい？』という比較演算。
  if(Num1 == Num2){
      document.getElementById("result").innerHTML = "等しい";
  }
  else{
      document.getElementById("result").innerHTML = "等しくない";
  }
}
</script>
</head>
<body>
<article>
<section>
<h3>半角数字で答えを入力したら，送信ボタンをクリックしてください。</h3>
<p>
<form>
左辺値：<input type="text" id="left-var" size="3"/> 
右辺値：<input type="text" id="right-var" size="3"/>
<input type="button" value="送信" onClick="ansCheck();"/>
</form>
```

出力：---
</p>
</section>
</article>
</body>
</html>

1.3.2　繰り返し構文 for

　指定した回数（無限も可能）の間，処理を繰り返されます。JavaScript には for 文が用意されています。
syntax（記法）は

```
for(初期値; 終了値; ステップ値){
  処理
}
```

サンプル 1.3.2　初項 3，公比 2 の等比数列の初項から第 20 項までを示す例です。

ここでは累乗の値を求めるために，Math オブジェクトの pow 関数を利用しています。特に理数系を学ぶ者にとって，数学関数を利用したい場面は多いです。この Math というオブジェクトは数学に関わる関数や円周率 π，自然対数の底 e といった特殊な定数も収めています。

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>for 文のサンプル</title>
<script language="JavaScript">
<!--
function mkGeometric()
{
   var SYOKOU = 3;//初項 3
   var KOUHI = 2;//公比 2
   var KOUSUU = 1;//項数 1 からスタート
   var RESULT = "";//結果

   for(KOUSUU; KOUSUU < 21; KOUSUU++){
     RESULT += "第" + KOUSUU + "項=" + SYOKOU*Math.pow(KOUHI,(KOUSUU - 1)) + "<br/>";
   }
   document.getElementById("result").innerHTML = RESULT;
}
//-->
</script>
</head>
<body>
<article>
<section>
<h3>初項 3，公比 2 の等比数列の初項から第 20 項まで。</h3>
<form>
<input type="button" value="表示" onClick="mkGeometric();"/>
</form>
<hr/>
```

```
<span id="result"></span>
</section>
</article>
</body>
</html>
```

Math オブジェクトは，http://developer.mozilla.org/ja/Core_JavaScript_1.5_Reference/Global_Objects/Math を参照。

2　理数系の JavaScript プログラミング

いよいよここからは，数学的考察や著名な定理を用いてプログラミングを実践していきましょう。

2.1　約数，素数，完全数，双子素数

2.1.1　約数の抽出

例えば 18 の約数は，{1,2,3,6,9,18} です。これをコンピュータを使って出力してみましょう。

プログラムの流れ

18 を 1 から順に 18 までの正の整数で割って余りを調べ，余りが 0 のものを約数として出力します。

必要なもの

ｆｏｒ文　1〜18 まで順に割り算の処理を繰り返すために必要。
ｉｆ文　余りが 0 のときは約数として処理を分けるために必要。
剰余演算子％　割り算の余りを計算するために必要。

演算子%の syntax は，
(割られる数) % (割る数)
戻り値は余りです。

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>18 の約数を抽出</title>
<script language="JavaScript">
<!--
function extractCommon()
{
   var i;
   var RESULT = ""; // 変数 RESULT の初期化
   for(i=1;i<=18;i++){
      if((18 % i) == 0){
         RESULT += i + ",";
      }
   }
   document.getElementById("result").innerHTML = RESULT;
}
//-->
```

```
</script>
</head>
<body>
<article>
<section>
<h3>18 の約数を抽出します。</h3>
<form><input type="button" value="抽出" onClick="extractCommon();" /></form>
<hr/>
<span id="result"></span>
</section>
</article>
</body>
</html>
```

実践 2.1

(1) 前ページのソースに手を加えて，18 の約数の総和を出すプログラムを作成しよう。
【ファイル名：prac_commonDivisor_1.html　　関数名：commonSum();】

(2) (1) を改良して，入力フィールドから任意な数を入力し，その数の約数の総和を出すプログラムを作成しよう。
【ファイル名：prac_commonDivisor_2.html　　関数名：commonSum();】
ただし，関数 commonSum() に，入力フィールドの値を引数として渡すように改良すること。

2.1.2　素数の抽出

素数とは，1 と自分自身以外に約数を持たない数のことを言います。素数は 2 からはじまり，{2,3,5,7,9,11,13,$\cdots$} と無数にあります。コンピュータを使ってこの素数を出力するプログラムを作りましょう。
ここでは，1〜100 までの中から素数を抽出するプログラムを作成します。そのために，次の 3 ステップの段階を経ながら完成させるようにしてみます。

ステップ 1

『出力』というボタンを作成します。このボタンをクリックすると出力欄として用意した`<span id="result"></span>`の innerHTML に，1〜100 までの数をカンマ区切りで出力するだけのプログラムを作成します。
for 文を利用　ファイル名 prac_prime_1.html で保存します。

ステップ 2

ステップ 1 を基に，1〜100 までの数の，それぞれの約数の個数をカンマ区切りで出力するように改良します。
剰余演算子%を利用　ファイル名 prac_prime_2.html で保存します。

ステップ 3

ステップ 2 を基に，1〜100 までの数のうち，約数の個数が 2 個（1 と自分自身のみ）のもののみを出力するように改良して完成です。
if 文の利用　ファイル名 prac_prime_3.html で保存します。

if 文の復習例題

例題 1

次は，入力値を代入した変数 Num が 2 の倍数なら，ダイアログボックスで”2 の倍数”と表示する HTML ソースです。下線を答え，ファイル名 if_review1.html で保存しなさい。

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>2 の倍数をチェック</title>
<script language="JavaScript">
<!--
function check()
{
     Num = ______________________________

     if((Num % 2)==0){
          alert("2 の倍数です。");
     }
}
//-->
</script>
</head>
<body>
<article>
<section>
<h3>2 の倍数のチェック</h3>
<form>
<input type="text" size="3" id="number"/>
<input type="button" value="チェック" onClick="check();" />
</form>
</section>
</article>
</body>
</html>
```

例題 2

次は，入力値を代入した変数 Num が 2 の倍数なら，ダイアログボックスで”2 の倍数”，そうでなければ”2 の倍数でない”と表示する HTML のソースです。下線を答え，ファイル名 if_review2.html で保存しなさい。

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>2 の倍数をチェック 2</title>
<script language="JavaScript">
```

```
<!--
function check()
{
     Num = _______________________________

     if((Num % 2)==0){
          alert("2 の倍数です。");
     }
     ______{
          ---------------------------------
     }
}
//-->
</script>
</head>
<body>
<article>
<section>
<h3>2 の倍数のチェック 2</h3>
<form>
<input type="text" size="3" id="number"/>
<input type="button" value="チェック" onClick="check();" />
</form>
</section>
</article>
</body>
</html>
```

実践 2.1.2

1. 前ページ if の復習例題を参考にして，入力値が 3 の倍数で無いならば[*1]，その入力値をダイアログボックスに表示する HTML ソースを作成し，ファイル名 prac_non_times_3.html で保存しなさい。ヒント：else 文は必要ない。

2. 『出力』というボタンを作成します。このボタンをクリックすると出力欄に，1〜100 までの数を改行しながら出力するプログラムを作成し，ファイル名 prac_list.html で保存しなさい。

3. 1〜100 までの数のうち，2 の倍数だけを改行しながら出力するように改良し，ファイル名 prac_common.html で保存しなさい。

4. 1〜100 までの数のうち，3 の倍数以外を改行しながら出力するように改良し，ファイル名 prac_noncommon.html で保存しなさい。

5. 1〜100 までの数のうち，2 の倍数で 3 の倍数でない数を改行しながら出力するように改良し，ファイル名 prac_stage2_middle.html で保存しなさい。

*1
A==B は A と B が等しいならば。
A!=B は A と B が等しくないならば
になります。

寄り道

for 文を入れ子（2 重化）にし，下のような九九算の出力結果を得られるように改良し，ファイル名 prac_99calc.html で保存しなさい。if 文は使用しない。

出力結果

1 2 3 4 5 6 7 8 9
2 4 6 8 10 12 14 16 18
3 6 9 12 15 18 21 24 27
4 8 12 16 20 24 28 32 36
5 10 15 20 25 30 35 40 45
6 12 18 24 30 36 42 48 54
7 14 21 28 35 42 49 56 63
8 16 24 32 40 48 56 64 72
9 18 27 36 45 54 63 72 81

2.1.3 完全数の抽出

完全数とは，自分自身を除く約数（真の約数という）の総和と自分自身が等しい数のことを言います。例えば 6 は自分自身を除く約数が {1,2,3} ですから，その総和は 6 となり一致するので完全数です。8 は {1,2,4} ですから，その総和は 7 となり一致しないので完全数ではありません。
本節では，完全数かどうかコンピュータを使って調べるプログラムを作りましょう。

ステップ 1

『出力』というボタンを作成します。このボタンをクリックすると出力欄として用意した`<span id="result"></span>`の innerHTML に，1〜10000 までの数を改行しながら画面に出力するプログラムを作成します。

ファイル名 prac_perfect_1.html で保存します

ステップ 2

ステップ 1 を基に，改行された 1〜10000 までのそれぞれの数のすぐ後にコロン（ : ）に続き，それぞれの約数を半角スペース区切りで出力するように改良します。

ファイル名 prac_perfect_2.html で保存します

ステップ 3

ステップ 2 を基に，1〜10000 までの数のうち，真の約数の和が自分自身と等しい数のみ出力するように改良して完成です。

ファイル名 prac_perfect.html で保存します

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>完全数の抽出【ステップ？】</title>
<script type="text/javascript">
<!--
function perfectNum(){

}
-->
</script>
</head>
<body>
<article>
<section>
<h3>完全数の抽出【ステップ？】</h3>
<form>
<button onClick="perfectNum();">出力</button>
</form>
<hr>
<span id="result">この欄に出力</span>
</section>
</article>
</body>
</html>
```

2.2　ヘロンの公式

三角形の三辺の長さから，面積を求める公式として有名です。

△ABC の各辺を a, b, c とし，その面積を S で表します。そのとき，

$S = \sqrt{t(t-a)(t-b)(t-c)}$　ただし，$t = \dfrac{a+b+c}{2}$ とする。

この公式を利用して，任意の三辺を持つ三角形の面積を求めるプログラムを作成しましょう。まずは，三角形の三辺を入力するページを作成します。

入力の際には注意が必要になります。例えば，最大辺を c とした場合，
$c > a$ かつ
$c > b$ かつ
$c < a + b$
でなければ三角形にはなりません。もし，三角形が成立しない数値が入力された場合には，それを知らせるダイアログを表示するようにして完成とします。

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>ヘロンの公式の利用</title>
<script language=”JavaScript”>
<!--


//-->
</script>
</head>
<body>
<article>
<section>
<h3>ヘロンの公式を利用して，三角形の面積を求めます。<br/>
半角数字で任意の三辺の長さ a，b，c を入力したら，計算のボタンをクリック。
</h3>
<form>
a = <input type=”text” id=”edge_a” size=”3”><br/>
b = <input type=”text” id=”edge_b” size=”3”><br/>
c = <input type=”text” id=”edge_c” size=”3”><br/>
<sup>注意：仮に最大辺が c ならば，その値は a+b より小さくなければ三角形にならないことに注意してください。</sup><br/>
<input type=”button” value=”計算”>
</form>
<hr/>
面積：<span id=”result”></span>
</section>
</article>
</body>
</html>
```

画面で出来栄えを確認しましたら，次にヘロンの公式を利用して面積を求める関数（ここではcalcTriangleAreaという名前にします。）を作成します。次のような仕様にしましょう。

- 計算ボタンをクリックするとcalcTriangleArea関数を呼び出す。
- calcTriangleArea関数では，三辺の値が入った入力フィールドをgetElementByIdで参照して，あらかじめ用意した変数に**文字列でなく数値として代入**する。そのためにはeval関数で数値に変換（キャストという）しなければならない。次のような感じ，
  var a = eval(document.getElementById(”edge_a”).value);
  var b = eval(document.getElementById(”edge_b”).value);
  var c = eval(document.getElementById(”edge_c”).value);

  evalとはevaluation(計算とか評価)の意味
- 次にこれらの変数をヘロンの公式に当てはめて計算し，例えばTriangleAreaという変数に計算結果を代入。次のような感じ
  var t = (a + b + c) / 2;
  var TriangleArea = Math.sqrt(t*(t - a)*(t - b)*(t - c));
- 最後に計算結果を代入したTriangleAreaをresultエリアに表示する。次のような感じ，
  document.getElementById(”result”).innerHTML = TriangleArea;

実践 2.2

以上の解説を元に，calcTriangleArea関数を作成しましょう。
次に，前ページの右枠内にも記載がありました，3辺a，b，cの値が適切でないとき，すなわち三角形にならないときにはダイアログを表示して，数値を正しい範囲に変更するメッセージを出す関数を作成します。関数名をWarningとしましょう。
このWarning関数は，calcTriangleArea関数内で呼び出すようにします。処理の流れはつぎのようになります。

1. ユーザーがa，b，cの値を代入し，実行ボタンを押下するとcalcTriangleArea関数が呼び出される。
2. calcTriangleArea関数は，入力フィールドからa，b，cの値を取得し，Warning関数にその値を渡して，戻り値を得る。
3. Warning関数はa，b，cが適当な数値であるかを確認し，適当であればtrueを返し，そうでなければfalseを返す。
4. Warning関数からの戻り値を基に，if文で条件分岐させる。trueであれば，ヘロンの公式で面積をそのまま求める。falseであれば，ダイアログで『入力された値では，三角形が作れません。値を変えて，もう一度やり直してください』というメッセージを出して終了する。