ななちゃんのIT教室

何のためのプログラム？：プログラム品質の巻

by nara.yasuhiro@gmail.com

ななちゃんが
プログラムの品質について考えるという お話

第 0.2 版　2017 年 7 月 4 日

フリー素材
http://freeillustration.net

いらすとやフリー素材
http://www.irasutoya.com/

もくじ

第1回 プログラムの品質？

なな: プログラムの品質？

先生: 抽象的な話をすると分かりにくいので、具体例で考えましょう。input に数を入力して、ボタンをクリックすると、その数の平方根をアラート表示するプログラムを考えてみましょう。下記のプログラムは、どちら（sample1.html と sample2.html + sample2.js）も同じような動作をします。

sample1.html

```
<input type=text id=in1>
<input type=button onclick=alert(Math.sqrt(in1.value)) value="平方根">
```

sample2.html

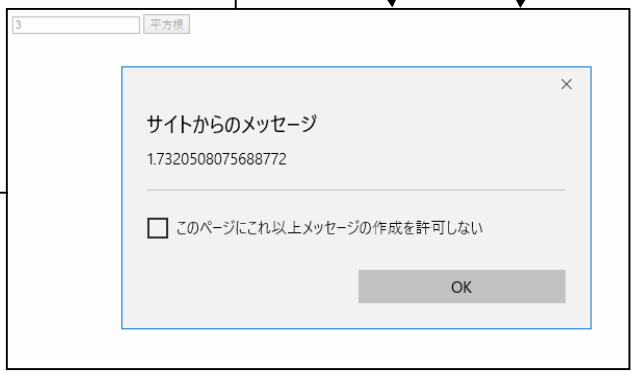
```
<!DOCTYPE html>
<html lang="ja">
  <head>
    <meta charset="utf-8">
    <title>sample2: squareroot</title>
    <script type="text/javascript" src="./sample2.js">
    </script>
  </head>
  <body>
     <input type=text id=in1>
     <input type=button id=btn value="平方根">
  </body>
</html>
```

＋

sample2.js

```
var inp1, btn;

window.onload = function() {
    in1 = document.getElementById('in1');
    btn = document.getElementById('btn');
    btn.addEventListener('click', function(event) {
        var x, squareroot;
        x = in1.value;
        squareroot = Math.sqrt(Number(x));
        alert(squareroot);
    }, false);
}
```



なな: ええ？ sample1 のほうは、ファイルが 1 つで、2 行、sample2 のほうは、ファイルが 1 つで、13 行と 12 行。同じ動作をするの？ 何が違うの？？？

先生: 品質が違うということなの。

なな: sample2 のほうが品質が高くて、良い、sample1 は品質が低いからよくないということ？

先生: そこが難しいところね。

第2回　品質？

なな: そもそも、品質って？　高ければ高いほど良いのでは？

先生: 最初から JavaScript について説明すると難しいので、たとえ話で説明させてね。英語の勉強について考えてみましょう。「中学から何年も英語を勉強しているのに英語が話せない」とか、「英語が母国語ではない外国人が英語を使いこなしているのがうらやましい、日本の英語教育に問題があるのではないか」などということが言われることがあるわね。

なな: 聞いたことがあるわ。「外国人は、文法的に間違いがあっても気にしないで、どんどんしゃべるからコミュニケーションができる」、「日本人は文法的なことを気にしすぎてコミュニケーションが苦手」とか。

先生: そうね。いわば、品質が低くても実用的なのが良いか、実用的でなくても品質を重視すべきかという問題ね。それから、こういう問題もあるの。英語が話せるつもりだったけど、ビジネスで使い物にならないとか、ビジネスの場面ではずかしい思いをしたとか。

なな: 仕事の場面で、「その点についてより詳しい説明をお願いしたいのですが」といいたいのに、「もっとちゃんと言ってよね」というような意味になって、通じることは通じるけど、苦笑されたりとか。

先生: そうそう。それから、勉強意欲の問題も。日本人には同じに聞こえる音でも区別が必要な音、r/l とか、n/ng があるわね。大人の先生は、「自分は間違った音を身に着けてしまって、大人になってから直すのに苦労した」、「だから、最初の 2 か月は音の区別、音の基礎を徹底的に教える必要がある」と考える。でも、例えば、大学の韓国語の授業でそういうことをやると、学生は興味が持てなくて脱落してしまう。最初は会話文なんかで興味を持たせて、細かい発音は、徐々に直すのがむしろ自然で、効率的だったりする。子供が自然に言葉を覚える過程は、親に間違いを直されながらコミュニケーションをすることから始まって、発音がだんだん明瞭になって、文法は学校に入ってから勉強する。

なな: 最近は、小学校から、子供英語を勉強して、成長とともに大人の表現を覚えようということなのかも。

先生: そうなのよ。子供が「質問をしてもよろしいでしょうか」というと不自然だし、大人が仕事のお客様に「聞いていいかな」というと失礼でしょう。 JavaScript でも、何のためにプログラムを作っているかによって、書き方が変わるということなの。

なな: なんとなく、雰囲気は分かったけど、まだ、ピンと来ないわ。

先生: まず、「何のためにプログラムを書くか」ということのバラエティを整理してみましょう。

- JavaScript を知りたい（正確性より、何ができるかを知りたい）
- JavaScript を使ってみたい（興味を維持したい）
- 自分専用の便利ツールを作りたい（自分のパソコン上で動けば良い）
- 自分専用の便利ツールを人にも使ってもらいたい（少数の他人がユーザ）
- 部署のツールとして部員に使ってもらいたい（類似環境で動作すれば良い）
- 社内ツールとして配布する（社内で統制されている範囲の環境で動作すれば良い）
- チームでプログラムを開発したい（複数のプログラマで共同開発）
- お客様への販売促進に活用する（古いブラウザへの対応、使いやすさへの配慮）

なな: たしかに、趣味のプログラムと、製品のプログラムとは違いそうね。

先生: それぞれの場面について、どういうプログラムが適切なのかを考えてみましょう。

なな: それぞれの場面で、JavaScript の品質を使い分けるということね。

第3回　JavaScript を知りたい、使ってみたい（正確性より興味）

先生：先ほどの平方根の例の場合、大人向けの、JavaScript 入門テキストでは、下記のようなプログラムが多いのね。

sample3.html

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>sample2: squareroot</title>
    <script src="sample3.js">
    </script>
  </head>
  <body>
      <input type=text id=in1>
      <input type=button onclick=go()>
  </body>
</html>
```

＋

sample3.js

```
function go() {
    var in1 = document.getElementById('in1');
    var x = in1.value;
    var squareroot = Math.sqrt(x);
    alert(squareroot);
}
```

なな：平方根の計算をするだけなのに、覚えることがたくさんあるのね。ちょっとうんざり。

先生：13 行 + 6 行あるわね。しかも、1 文字タイピングミスしただけで、たいていは動かない。ウェブページからコピペすると動くけど、キーボードからゼロから入力すると、まず動かない。自力では、やる気を失う。「プログラミングは大変」、「私はプログラミングに向いてない」と思ってしまう。

なな：品質以前の問題ね。

先生：下記の 2 行だけを、「sample1.html」という名前のファイルに書き込んで、ファイルのアイコンをダブルクリックすると、プログラムが動く。枠に「3」と入力すると、「1.41421356….」と表示される。sqrt は、square root だから平方根。ここを sin に変えたら三角関数になりそうと考えられる。ファイルに保存する時、文字コードが「UTF-8」になっていることを確認してね。

sample1.html

```
<input type=text id=in1>
<input type=button onclick=alert(Math.sqrt(in1.value)) value="平方根">
```

なな：たった 2 行だったら、タイピングミスも起こりにくいし、チェックもなんとかできそう。プログラムが動いたという感激があって、もっとやろうと思う。

先生：英語で言えば、「What’s this?」と聞くと、「A book.」などと答えてくれてうれしくなるレベルね。「What」とか、「’s」とかの意味は分からないけど、コミュニケーションはできる。大人の会話とは、到底、言えないけど、英語を話した気分になれる。

なな：このプログラムは、ふつうのパソコン環境で、特別なプログラムを追加導入しないで使えるのね。

第4回　自分専用の便利ツールを作りたい（自分のパソコン上で動けば良い）

先生：前回のプログラムだけど、もっと複雑なことをやろうとすると問題が起こるの。そもそも、意味が分かりづらい。実は、「<input type=button onclick=」と「value="平方根">」の部分は html という言語、「**alert(Math.sqrt(in1.value))**」の部分は JavaScript という言語。見た目は簡単でも、意味は複雑。

sample1.html

```
<input type=text id=in1>
<input type=button onclick=alert(Math.sqrt(in1.value)) value="平方根">
```

先生：そこで、JavaScript 部分を、html 部分と分ける。

sample4.html

html / JavaScript

```
<input type=text id=in1>
<input type=button onclick=go() value="平方根">
<script>
    function go() {
        alert(Math.sqrt(in1.value));
    }
</script>
```

ここだけ混在

なな：go() の部分だけは、html と JavaScript が混在しているけど、その他の JavaScript 部分は、<script> ～ </script> にまとめられているわけね。

先生：こういうふうに分けておけば、平方根の計算を拡張して、1 ～ 10 の合計を計算するとか、体重と身長とから BMI（肥満度）を計算するとか、できそうでしょう？

なな：自分で使うだけなら、これで十分ということね。

先生：専門のプログラマでない人は、こういう目的が多いはずなの。電卓代わりとか、いろんな研究での確認計算とか、ちょっとしたデザイン画の生成とか。私は、一般の人に、このような形で、気軽に JavaScript を活用してもらいたいと思っているわ。

なな：「自分のパソコン上で動けば良い」って、どういう意味？

先生：自分が使っているパソコン、OS、特定のブラウザ上で使えればそれで用が済むということよ。

第5回　自分専用の便利ツールを人にも使ってもらいたい

先生: 次に、少数の知人にも使ってもらおうという場合。ブラウザも、各人の好みで、Internet Explorer、Microsoft Edge、Firefox、Safari、Google Chrome とか、いろいろあるでしょ。そういう場合、文字化けが起こったり、エラーになったりしないように、下記の太字の部分を書き加えたほうが良いのよ。今回のプログラムでは不要かも知れないけど、内容をもっと複雑にした時に、思わぬ問題が起こることを未然に防止できるの。

sample5.html

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
      <input type=text id=in1>
      <input type=button onclick=go() value=”平方根”>
      <script>
          function go() {
              alert(Math.sqrt(in1.value));
          }
      </script>
  </body>
</html>
```

な な: 便利なプログラムを作ったら、人に見せたいし、いばってみたいしね。そういうのが、自分の励みになるし。でも、行数が倍増するのね。

先生: プログラムは、毎回、ゼロから書くんじゃなくて、前に作ったプログラムをコピーして、太字の部分をそのまま使うようにすれば、手間は増えないわ。

注意: 「<!DOCTYPE html>」の行を入れると、「HTML 5」としての文法チェックが行われます。以前は許されていたが、HTML 5 では許されなくなった記述がエラーになります。たとえば、スタイル属性のサイズ指定では、単位が省略できません。width=30 ではエラーになり、width=30px とすることが必要になったりします。また、図形を拡大/縮小する時のスムージングの方法（レンダリング方法）が変わったりします。 ですから、「<!DOCTYPE html>」を入れると、今まで出なかったエラーが出たり、表示が少し変わったりします。

第6回　部署のツールとして部員に使ってもらいたい

先生: 次に、会社なんかの同僚に使ってもらう場合。会社では、保守のしやすさから、パソコンの OS や、バージョンは揃えてあることが多いの。でも、どのブラウザを使うかは、多少、好みがあるかも。そして、会社で大切なことは、良かれと思ってプログラムをあげたのに、動作不良なんかが起こると、仕事が中断してしまい、迷惑をかけてしまうこと。たとえば、職場全体で、Windows 7 から、Winows 10 に移行なんてことがあり得るけど、そんな時に、みんなに提供したプログラムが動かなくなったりすると大変。今回の例題プログラムでは、太字の 1 行を追加したほうが良いの。

sample6.html

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
   </head>
   <body>
        <input type=text id=in1>
        <input type=button onchange=go() value=”平方根”>
        <script>
             var in1 = document.getElementById(‘in1’);
             function go() {
                  alert(Math.sqrt(in1.value));
             }
        </script>
   </body>
</html>
```

な な: これは何？

先生: html の input タグと、JavaScript のオブジェクトを結びつけるものなの。ブラウザの仕様には明確に書かれていないけど、「**多くの**」ブラウザでは、ウェブページを最初に表示する時に、自動的に計算しておいてくれるのね。でも、完全に保証されているとは限らないので、プログラムにちゃんと書いておいたほうが「**無難**」と言えるわけ。

第7回　社内ツールとして配布する

先生：次に、プログラムをもっと複雑に、高度化した上で、社内に広く配布することを考えましょう。例題の処理内容を複雑にすると、説明しにくくなるので、複雑、高度化の部分は想像して、ということで。ここで、ポイントになるのは、自分自身が職場を移ることになった場合、他の人がプログラムの保守を受け継ぐということ。

sample7.html

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
      <input type=text id=in1>
      <input type=button onclick=go() value='平方根'>
      <script src=sample7.js>
      </script>
  </body>
</html>
```

sample7.js

```
var in1 = document.getElementById('in1');

function go() {
    alert(Math.sqrt(in1.value));
}
```

なな：職場を移る？　左遷？

先生：前向きに考えてね。栄転とか、念願の海外転勤とか。そういう場合に、プログラムの保守を他の人に引き継いでもらうことが必要になるの。引き継いだ人が、プログラムに機能追加をするとか、タブレット端末が増えてきて、マウス操作以外に、タッチ操作でも使いたくなるとか。

なな：自分の作ったプログラムが優秀で、みんなが末永く使う場合、保守のバトンタッチを見越しておかないといけないということね。プログラムの書き方は、具体的にどうすれば良いの。

先生：プログラムを、他人に読みやすくする、理解しやすくする、修正しやすくするというのがポイント。そのためには、「html 部分と、JavaScript 部分を別ファイルにする」必要があるの。プログラムが複雑になってくると、全体の行数も増えてくるので、ファイルを分けたほうが良いの。特に、html の中に、<script>～</script> が複数存在している場合、それらをひとつにまとめた上で、別ファイルにすると良いの。

第8回　チーム開発

先生: プログラムが複雑で、大きくなってくると、1 人のプログラマでは足らなくなってくるわね。機能追加を迅速に行うために複数のプログラマで作業を分割して同時に作業するとか、1 人のプログラマが休暇中でも、別のプログラマが保守できるようにするとか。すると、JavaScript ファイルを複数に分けて、それぞれを <script> タグで読み込むというようなことが必要になるの。そして、<script> タグは、<head> 部分にまとめて置きたくなる。<head> 部分に置くと、<input> タグが画面に現れる前に JavaScript が実行されてしまって、動作しなくなる。そこで、「画面表示が完了してから JavaScript」を動作開始させる」といった制御が必要になる。それが、第1回の sample2 にあった、window.onload なんだけど、複数のプログラマが、それぞれの担当部分を別々のファイルに書き、それぞれ window.onload を使うと競合してしまう。競合というか、後ろにある <script> で読み込まれたファイルの window.onload だけが有効になって、先に読み込まれる <script> の window.onload は無効になってしまう。これを防ぐには、window.addEventListener('load',… を使う必要がある。これを使えば、複数のものが有効になる。

sample8.html

```
<!DOCTYPE html>
<html lang="ja">
  <head>
    <meta charset="utf-8">
    <title>sample2: squareroot</title>
    <script type="text/javascript" src="./sample8.js">
    </script>
  </head>
  <body>
      <input type=text id=in1>
      <input type=button id=btn value='平方根'>
  </body>
</html>
```

JavaScript が全く含まれなくなりました。

Sample8.js

```
var in1, btn;

window.addEventListener('load', function() {
    in1 = document.getElementById('in1');
    btn = document.getElementById('btn');
    btn.addEventListener('click', function(event) {
        var x, squareroot;
        x = in1.value;
        squareroot = Math.sqrt(x);
        alert(squareroot);
    }, false);
}, false);
```

なな: ここまで来ると、大人向け入門テキストより、さらに上を行くことになるのね。

第9回 お客様への販売促進に活用する（古いブラウザへの対応、使いやすさへの配慮）

先生: いよいよ最終回。これまでは、社内システムが対象でした。社内システムは、身内が助け合いながら作業を行う環境ね。今回は社外向けのウェブページで使うとか、プログラム自体を販売して商売するとかの話です。

なな: 何が変わるの。

先生: まず、古いブラウザが使われる可能性があるということ。社内だったら、保守のしやすさから、みんなで最新バージョンのOSを使おうなどということになる。でも、古いブラウザを使っている顧客に対応できなかったり、使いにくいウェブページなら、顧客が競合他社に移ってしまう可能性がある。下記は、2011年公開の IE 9 より古い Internet Explorer にも対応し、さらに、入力が不適切だとその旨指摘するもの。

sample9.html

```
<!DOCTYPE html>
<html lang="ja">
  <head>
    <meta charset="utf-8">
    <title>sample2: squareroot</title>
    <script type="text/javascript" src="./sample9.js">
    </script>
  </head>
  <body>
     <input type=text id=in1>
  </body>
</html>
```

sample9.js

```
var in1, btn;

if (window.addEventListener) {
    window.addEventListener('load', init, false);
} else if (window.attachEvent) {
    window.attachEvent('onload', init);
}

function init (event) {
    in1 = document.getElementById('in1');
    btn = document.getElementById('btn');
    if (btn.addEventListener) {
        btn.addEventListener('click', f, false);
    } else if (btn.attachEvent) {
        btn.attachEvent('onchange', f);
    }
}

function f (event) {
     var x, squareroot;
     x = Number(in1.value);
     if (isNaN(x)) { alert("入力が有効な数ではありません"); return; }
     if (x < 0)      { alert("入力が正数ではありません"); return; }
     squareroot = Math.sqrt(x);
     alert(squareroot);
}
```

なな: 古いブラウザは、何年かすると使われなくなるわね。ブラウザや、それをサポートする OS が、メーカのサポート対象外になったりして。そういう時に、古いブラウザのサポート部分を削除しないと、プログラムの無駄が増えてしまうわね。でも、何か所も修正するのは大変ね。

先生: そういうことを考慮すると、古いブラウザをサポートする部分は関数（メソッド）に入れておき、プログラムからはその関数を呼び出すようにしておくと良いのよ。例えば、上の例だったら

```
_addEventListner(element,event,func,flag) {
    if (element.addEventListener) {
        element.addEventListener(event, func, flag);
    } else if (element.attachEvent) {
        element.attachEvent('on'+event, func);
    }
}
```

のようなのを作っておいて、プログラムでは _addEventListner を呼び出すようにする。時間がたったら

```
_addEventListner(element,event,func,flag) {
        element.addEventListener(event, func, flag);
}
```

に書き換えれば良いわ。プログラムのほうは変更不要。こういうのをたくさん集めたものを「ライブラリ」と言います。

なな: 苦労して作ったライブラリはノウハウのかたまりで、そのプログラマチームの宝になるわね。でも、JavaScript って、プログラムを第三者が見ることができるので、流用されそうでもったいない気がするわ。

先生: そういう場合には、プログラムの「難読化」ツールとか、「暗号化」ツールを使うのよ。コメントを削除したり、変数名を意味の無い名前に取り換えたり、改行を取り去ったりする道具があるの。 プログラムとしてはちゃんと動作するし、むしろ効率よく実行できるけど、人間が見ると解読困難になるのね。

なな: 解読困難だと、保守が大変なのでは？

先生: 保守は、難読化前のプログラムでやって、ウェブサーバに書き込む時に難読化処理を行うわけ。

なな: その他、気をつけることは？

先生: 好意的なユーザだけでなく、クラッカーみたいな、悪意を持ったユーザが、業務妨害やいたずらでアクセスしてくる可能性もあるので、セキュリティ対策を考慮する必要もあるわね。それから、いろいろなブラウザで動作確認テストを十分行うことも必要ね。

なな: なるほど！　自分専用のプログラムから、お客様対応版まで、品質に応じて、いろんなプログラムがあるのね。

先生: ななちゃんは、「プロフェッショナルなプログラマになるならこんなプログラムが必要」と知った上で、今は、自分専用のプログラムを、それに見合うスタイルで書いていれば良いということね。