Javascript First Step

2022 年 4 月 12 日

概要

プログラミングの初心者向けに、Javascript を利用して説明します。

目次

1　プログラミングするにあたっての心構え

1.1　プログラミングって何?

さて、プログラミングってなんだ?

コンピュータに処理をさせるために、プログラムを作ること

だよね。なので、ほんの少しだけ、プログラマーはコンピュータに考え方を合わせてあげる必要がある。

ただ、前に比べたらはるかに人間の思考を理解するようになっているので、そこまで難しく考える必要はないよ。 ここ数十年、コンピュータのスピードは飛躍的に伸び、いろいろなことができるように準備がされている。

それをうまく利用してあげて、組み合わせればいろんなことがそんなに難しくなく書けるようになるよ。

1.2　私、理系じゃないし...

コンピュータだから理系でないとうまくいかない、と思う人もいるかも知れないけど、実は大きなところではそうではなかったりするよ。

言語能力って、自分が思ってることを正しく相手に伝えるやり方だよね。

コンピュータ言語、っていうくらいなので、実は、自分が指示したいことを正しく相手に伝えるのがプログラミングなんだ。

もちろん、たまには理数系の頭が必要になることもある。でも、大きな意味では、それを必要としない部分もかなりある、と思って欲しい。

実際、文系の人が SE になるケースは数十年前からすっごくあるんだ。

人は、類推してくれるかもしれないけど、コンピュータは類推はしてくれなくて、正しく書かないと「わかりませーん」っていうだけ。

1.3　まずはじめに覚えなくてはいけないこと

コンピュータに何かさせたいわけだから、コンピュータになんらかの情報を入力して、計算させて、それの出力を得る、っていうのが基本的な考え方だよ。

自分が何をしているかわからなくなったら、今何をやっているか、ということを認識しよう。

あと、完璧主義者はそんなに向いてない気がする。「とりあえずやってみたら動いたー!」と気楽にしてた方が、伸びると思うよ。

一つプログラミング言語を覚えると、後は、方言みたいなものだから、次にいくのはそこまで難しくない。 ので、最初につまづかないようにするのが大事だね。

1.4　英語...

日本語のプログラミング言語も一応は存在している (た) んだけど、やっぱり、英語が圧倒的に使われているね。結局そっちの方が書きやすいんだよな... ので、怖がらないでね。

1.5 全角スペース・半角スペース

コンピュータは基本的に、日本語を表示する時以外は、半角の英数、記号を使うよ。間違えないようにしよう。

1.6 インデント

プログラミングはその書いている有効範囲を示すために、「始まり」「終わり」をマークしていくことが多い。で、それがみづらいと非常に読みにくくなってしまう。TAB をうまく使って、読みやすく書くように気をつけよう。

インデントをつけすぎた場合には、Shift+TAB で元に戻せるよ。

1.7 アルゴリズム

コンピュータにさせたいことの処理手順をアルゴリズムというのだけど、実際にはこれを考えるのが一番大変かな。例えば、A 地点から B 地点にいくのに短距離でいくにはどうしたらよいか?ということを計算させるのに、どうやれば処理手順が短くてすむか、というのを考えるのが一番大事なんだ。

2 超基本的なこと

Javascript をベースにプログラミングを学んでいこう。

今日は、何かが動いたりあまりしないのでつまらないかもだけど、英語の基本文法を覚える感じでやってみよう。

2.1 変数

プログラミングするときに、値を保存しておく必要があるので、これのことを「変数」と呼ぶよ。

```
let myvalue = 10;
```

等と書けば OK。文字の場合には

```
let myvalue = "Hello world!";
```

の様にダブルクオーテーションでくくってあげることで、「文字列だよ」と表現するよ。

言語によっては、型宣言というのをすることもあって、変数の型が整数なのか小数なのかなどを定義最初からしてしまうやり方。Javascript では型という考え方はあるけれども、明示はしないよ。

2.2 つーか、どうやって実行すればいいの?

Javascript はブラウザで実行できるので、VS Code でやりましょうかね。うまく動かない時にはブラウザをリロードしてください。

```
<!DOCTYPE html>
<html>
```

```
        <head>
        </head>
        <body>
                <script type="text/javascript">
                        let myvalue = "hello world";
                        document.write(myvalue);
                </script>
        </body>
</html>
```

を js.html とでもして保存して実行してみよう (作業用のフォルダ作って、ワークスペース開いてから)。このように、script タグ内に Javascript を書くことができるよ。

2.3 足し算をやってみよう。

```
<html>
        <head>
        </head>
        <body>
                <script type="text/javascript">
                        let a = 1;
                        let b = 2;
                        document.write(a+b);
                </script>
        </body>
</html>
```

かんたん!じゃ、文字列を足すとどうなるんだろう?

```
 <html>
        <head>
        </head>
        <body>
                <script type="text/javascript">
                        let a = "hello ";
                        let b = "world";
                        document.write(a+b);
                </script>
        </body>
</html>
```

文字列がつながったね。文字列の連結と呼ぶよ。

他の数字の計算も試してみて

＋　足し算
-　引き算
＊　掛け算
／　割り算
％　剰余

あと、プログラミングでは数学ではありえない変な書き方もするよ。

```
a = a + 1;
```

「=」というのは、

数学　左と右が等しい
プログラミング　右の値を計算して、左に代入する

という意味になるよ。

```
<html>
        <head>
        </head>
        <body>
                <script type="text/javascript">
                        let a = 1;
                        a = a+1;
                        document.write(a);
                </script>
        </body>
</html>
```

とすると、2 と表示されたかな？

2.4　配列

これは聞き慣れないんじゃないかな?データを整理するときに非常に便利な考え方。

```
let a1 = new Array();
let a2 = new Array(111,222,1000);
```

とかって書くよ。a2[0]=111,a2[1]=222,a2[2]=1000 の様になります。同じ型でなくても、文字とかなんでも代入できるよ。

```
let myarray = new Array();
myarray[0] = 7;
myarray[1] = "hello";
document.write(myarray[1]);
```

2.5 条件分岐

条件によって処理を変えることを「条件分岐」と呼ぶよ。例えば、ゲームなら、ゲームオーバーかどうか、等の判定に利用できるね。

```
<html>
        <head>
        </head>
        <body>
                <script type="text/javascript">
                        let a = 1;
                        if(a > 10){
                                document.write(a+"は 10 より大きいです。");
                        } else {
                                document.write(a+"は 10 以下です。");
                        }
                </script>
        </body>
</html>
```

正しく計算されたら、「let a=20;」として試してみよう。

どちらが大きいかの比較ではなくて、等しいかどうかを知りたいときは「=」でなく「==」「===」を利用するよ。「=」だとさっきの説明通り代入しちゃった結果を評価して、いつでも正しくなっちゃうから。=の数が 2 個と 3 個では全く同じ意味ではないけれども、これはかなり詳しい話しになるので、ここでは省略。

2.6 繰り返し

コンピュータは単純作業の繰り返しが非常に得意！(この辺から、html のタグははずして Javascript だけを記載するよ。)

2.6.1 for 文

```
for(初期化；繰り返し条件；変数の更新){
        繰り返す処理
}
```

と書くことで同一の処理を何度もできるよ。

```
        let i;
        for(i=1;i<10;i++){
                 document.write(i+"<br/>");
        }
```

i++ っていうのは「i=i+1」を短く書いたもの。i-であれば「i=i-1」の意味になるよ。

2.6.2 while 文

while では、条件が正しい時にはずっと処理を繰り返すよ。

```
while(繰り返し条件){
        繰り返す処理
}
```

2.6.3 continue,break

continue はループは続行しつつ、繰り返す処理をそこで止める。break はループを抜けるよ。

```
let i;
for(i=1;i<10;i++){
        if(i%2==0){
                continue;
        }
        document.write(i+' ');
}
document.write("<br/>");
for(i=1;i<10;i++){
        if(i==5){
                break;
        }
        document.write(i+' ');
}
```

2.7 関数

同じ処理を何度もする時には、関数として定義して、後で再利用ができるようにしておくよ。

```
function 識別子（パラメータリスト){
        関数で定義したい処理
}
```

例えば、「Hello と記載する関数」は次のように書けばいいよ。

```
function hello(){
  document.write("hello");
}
hello();
```

識別子は定義されているものとぶつからなければ、自由に設定していいです。 名前をいれて「Hello, Osamu」等としたい場合には

```
function hello2(name){
        document.write("hello, "+name+"<br/>");
}
hello2("Osamu");
hello2("Michiari");
```

のようにパラメータを定義することができるよ。関数として定義しておくと同じ処理をする時に便利でしょ？

2.8 イベント

ボタンをおした時に Javascript を起動したい場合は、

```
<a href="javascript:void(0);" onclick="hello('Sammy');return false;">Click</a>
```

とすると

ボタンが押されたよ

というイベント (出来事) が発生し、そのイベントにあらかじめ登録された関数を呼び出すことができるよ。

href は、跳び先はないよ、だけど、onclick の時に hello を呼び出してね。と言う意味。return false はおまじないだと思ってください。

おっと、HTML5 では、document.write() が非推奨となってしまった... 昔なら、これでどんどん追加できたのに。というわけで、ちょっとめんどうくさいけれども

```
<!DOCTYPE html>
<html>
        <head>
                <script type="text/javascript">
                        function hello(name){
                                document.body.innerHTML+="hello"+name+"<br/>";
                        }
                </script>
        </head>
        <body id="body">
                <a href="javascript:void(0)" onclick="hello('Sammy');return false;">Click</a>
        </body>
</html>
```

これが HTML5 のご時世では正解ですね。

3 実践！

覚えなくてはいけないことがちょっと多かったかな。でも、これらを組み合わせることでいろいろなことができるよ。(楽なので、document.write() を使います。)

3.1　1 から n までを合計する sum(n) という関数を作ってみよう

```
function sum(n){
        let a=0;
        let sum=0;
        while(a++ < n){
                sum+=a;
                document.write("a: "+a+", "+ "sum: "+sum+"<br/>");
        }
}
sum(10);
```

3.2　n が素数かどうかを判定する primenumber(n) を作ってみよう

```
function primenumber(n){
        let i= 1;
        while(i++ < n){
                document.write(i+","+n+"<br>");
                if(n % i != 0){
                        document.write("割り切れない<br/>");
                } else {
                        document.write("割り切れる<br/>");
                        break;
                }
        }
        if(i == n){
                document.write("素数!");
        } else {
                document.write("素数でない!");
        }
}
primenumber(5);
```

3.3　1 から n までの素数を表示する primenumberlist(n) という関数を作ってみよう

関数は

return 戻り値;

とすると値を返してくれます。

```
function primenumber(n){
```

```
        let i= 1;
        while(i++ < n){
                //document.write(i+","+n+"<br>");
                if(n % i != 0){
                        //document.write("割り切れない<br/>");
                } else {
                        //document.write("割り切れる<br/>");
                        break;
                }
        }
        if(i == n){
                return 1;//素数
        } else {
                return 0;//素数でない
        }
}
document.write(primenumber(5));
```

ですので、これを利用しましょう。primenumber(n) はそのままにして

```
function primenumberlist(n){
        document.write(n+"までの素数は<br/>");
        for(let i=2; i<n; i++){//素数は 1 は除外するため
                if(primenumber(i)==1){
                        document.write(i+" ");
                }
        }
        document.write("<br/>です。");
}
primenumberlist(100);
```

最後に

プログラミングの触りしかしなかったけど、組み合わせることで、面倒くさい作業を自動でやってくれるのが便利だと感じられたかな？あとは、指示を出すあなた次第なので、興味のある人はいろいろ遊んでみよう。

以上