Vol.2019-CE-151 No.3
2019/10/5

高等学校「情報 I」の研修資料を JavaScript 言語で実施するうえでの検討

岡本雄樹[1]　辰己丈夫[2]

概要：　JavaScript は教科「情報」の教科書でも採用されており，教育現場での実績が豊富な言語である．文部科学省が 2019 年 5 月に一時的に公開した教員研修資料でも掲載が決まっているが，プログラミング部分は Python のみが先行公開され，JavaScript による研修教材の公開は延びている．研修教材の公開が更に遅れたり，万が一不十分な形で公開されたりした場合，既に JavaScript 言語でプログラミング教育を行っている学校や地域は不利益を被ってしまう．そこで，研修資料を JavaScript 言語で実施する上でのサンプルコード作成や課題と解決方法などを検討して論じたい．

キーワード：プログラミング教育，高等学校，情報 I , JavaScript，教員研修

1. はじめに

2019 年 5 月に文部科学省より高等学校情報科「情報 I」教員研修用教材[1]が公開された．この資料は 4 章立ての構成となっており，また 24 個の学習活動が設定されている（図 1）．

第1章	情報社会の問題解決	学習1	情報やメディアの特性と問題の発見・解決
		学習2	情報セキュリティ
		学習3	情報に関する法規，情報モラル
		学習4	情報社会におけるコミュニケーションのメリット・デメリット
		学習5	情報技術の発展
第2章	コミュニケーションと情報デザイン	学習6	デジタルにするということ
		学習7	コミュニケーションを成立させるもの
		学習8	メディアとコミュニケーション，そのツール
		学習9	情報をデザインすることの意味
		学習10	デザインするための一連の進め方
第3章	コンピュータとプログラミング	学習11	コンピュータの仕組み
		学習12	外部装置との接続
		学習13	基本的プログラム
		学習14	応用的プログラム
		学習15	アルゴリズムの比較
		学習16	確定モデルと確率モデル
		学習17	自然現象のモデル化とシミュレーション
第4章	情報通信ネットワークとデータの活用	学習18	情報通信ネットワークの仕組み
		学習19	情報通信ネットワークの構築
		学習20	情報システムが提供するサービス
		学習21	さまざまな形式のデータとその表現形式
		学習22	量的データの分析
		学習23	質的データの分析
		学習24	データの形式と可視化

図 1 「情報 I」教員研修用教材で取り扱う内容

プログラミングに関する内容は主に 3 章で盛り込まれている．また，4 章にも一部，プログラムのコード例が掲載されている．研修資料では Python のコードが記載されているが，別のプログラム言語として JavaScript, VBA, Swift，ドリトルのコードが記載された PDF を用意することが 11 ページ目に記載されている．しかし，2019 年 9 月 10 日現在，PDF の公開は順次公開予定となっており，また，研修資料自体も準備中となってしまった．

研修資料の WG 委員に名を連ねている兼宗が 2019 年 8 月に情報教育シンポジウムで行った発表[2]によると，JavaScript 版の研修教材では「WebAPI」や「グラフ描画」や「グラフィックス」に対応していないことが述べられている（図 2）．その理由として，兼宗は‘JavaScript 自体はこれらを記述することは可能だが,「テキストに掲載できるような短い記述では実現できない」「専用のライブラリを入手して設定する必要がある」等の理由から，掲載を見送ったものと考えている’と述べている．しかし，全体で 200 ページを超える研修資料においてコードの長さが理由というのは疑問が残る，また，専用のライブラリを使わなければグラフを描けないことは Python も同様であり，研修資料では matplotlib というライブラリが利用されているため，理由としては弱いと考える．

学習	内容	PY	DL	JS	VB	SW
11	コンピュータの仕組み	○	○	○	○	○
12	外部装置との接続	○	○	○	×1	○
13	基本的プログラム	○	○	○	○	○
14	応用的プログラム	○	○	△2	○	○
15	アルゴリズムの比較	○	○	○	○	○
16	確定モデルと確率モデル	△4	○	△3	△4	△4
17	自然現象のモデル化とシミュレーション	△4	○	×5	△4	△4

1.micro:bit に対応していない。
2.WebAPI に対応していない。
3. グラフ描画に対応していない。
4. アニメーションとして描画されない。
5. グラフィックスに対応していない。

図 2 言語毎の対応

JavaScript は現行の教科書にも掲載されている言語であり，また，和歌山県では「きのくに ICT 教育プロジェクト」のもと県下の高校において JavaScript による 20 時間程度のプログラミング教育を今年度からスタートしているため，研修資料の内容は JavaScript でも早急に実施可能な状態になることが望ましい．

なお，研修資料の公開がこのような状況におちいった原因も検討したい．文部科学省の令和２年度概算要求のポイント[3]によると，「現職教員の情報教育に係る指導力向上事業」に１５百万円が計上されており，高等学校「情報 I」

1 放送大学大学院　修士課程
2 放送大学

の教員研修用教材の作成が記載されている．つまり，来年度も文部科学省では「情報Ⅰ」の研修教材開発を行うということである．そのため，今年度は予算の問題で JavaScript や他の言語での教材開発まで至れなかったのではないかと推測できる．少なくとも，文部科学省では研修資料の修正版を印刷して各都道府県に配付するためには追加の予算が必要になるため，予算の問題を抱えている可能性は大きい．

来年度になれば明らかになることではあるが，研修資料によって方向性は既に示されている．本検討により課題と解決方法が明らかになった暁には，JavaScript で研修を行うための互換教材を開発し，研修の実施に役立てたいと考えている．

2. 研修における JavaScript の実行環境の検討

JavaScript の実行環境として，主にブラウザを実行環境にする方法とパソコンに実行環境を構築する方法の 2 つが考えられる．

2.1 ブラウザで実行する

JavaScript はブラウザの上で動作させることができる．ブラウザでプログラミング研修を行えれば，環境構築の手間を省くことができるため，今回はブラウザでの実行を中心に検討を行った．

なお，研修資料では WebAPI を利用したプログラムが含まれていたため，インターネット環境があることを前提と判断し，クラウド型のプログラミングツールである Monaca も利用することとした．

2.1.1 Monaca の活用

Monaca はブラウザだけで利用できるプログラミングツールである．ブラウザ上に高機能な WebIDE を展開できエディタ部分とプレビュー部分を 1 画面で表示できるため，実習をスムーズに行うことができる．スマートフォンやタブレット向けのデバッガーアプリも用意されており，スマートフォン上にプログラムの実行結果を表示したり，発生したエラーを WebIDE 上に表示したりできる．また，スマートフォンに備わっている加速度センサーや GPS，カメラなどが利用できる．そして， HTTPS 及び CORS に非対応の WebAPI を呼び出すこともできる．

2.2 パソコンに Node.js インストールして実行する

Node.js はサーバーサイドやパソコン上などで JavaScript を動かすための実行環境である．Windows OS や Mac OS の標準では入っていないためインストール作業が必要となるが，インストールは比較的簡単である．公式サイトでは安定版と最新版の 2 つが提供されている．こちらも，HTTPS 及び CORS に非対応の WebAPI を呼び出すことができる．ただし，外部のライブラリが必要となる．

3. 研修資料の JavaScript 対応の検討

3.1 学習 11 コンピュータの仕組み

学習 11 ではオーバーフローと計算誤差を扱っている．オーバーフローの発生を JavaScript で記述したソースコードと実行結果を図 3 オーバーフローの発生 (JavaScript)に示す．

```
var x = Number.MAX_VALUE;
console.log(x);
x = 1.79769313486231579999e+308;
console.log(x);
x = 1.8e+308;
console.log(x);
```

```
1.7976931348623157e+308
1.7976931348623157e+308
Infinity
```

図 3 オーバーフローの発生 (JavaScript)

また，計算誤差を JavaScript で記述したソースコードと実行結果を図 4 計算誤差 (JavaScript)に示す．研修資料に示されている Python の実行結果と殆ど同じ結果となった．

```
var x = 28 - 27;
console.log(x);
var y = 0.28 - 0.27;
console.log(y);
```

```
1
0.010000000000000009
```

図 4 計算誤差 (JavaScript)

3.2 学習 12 外部装置との接続

学習 12 では外部装置の例として加速度センサーや LED を備えた装置である micro:bit を活用したプログラムが掲載されている．実行環境としては Mu や MicroPython が上げられている．JavaScript で micro:bit を活用したプログラミングを行う場合，micro:bit 公式サイトで掲載されているツールの一つである Microsoft MakeCode の利用が考えられる．MakeCode であればブロック型で記述したプログラムをテキスト型である JavaScript と相互に変換しながら学習できるため，小中でブロック型を学んできた生徒にテキスト型との違いを視覚的に教えることができる．

学習 12「②条件分岐構造」に記載されたプログラムを Microsoft MakeCode のブロックで記述した例を図 5 に示す．

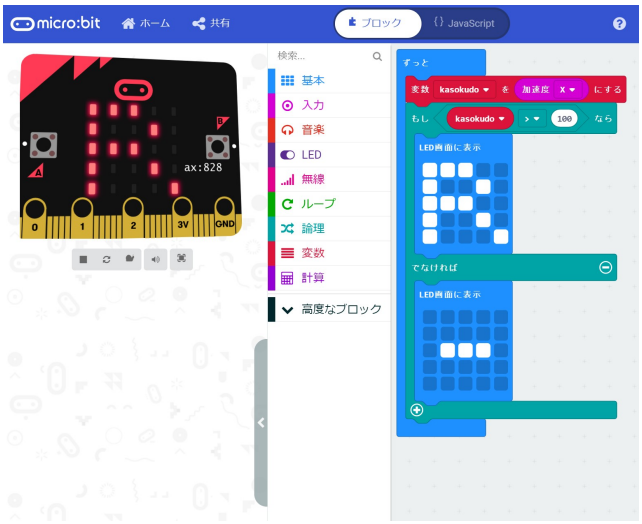

図 5 加速度と条件分岐（ブロック）

また，JavaScript に相互変換したソースコードを図 6 に示す．

```
let kasokudo = 0
basic.forever(function () {
    kasokudo = input.acceleration(Dimension.X)
    if (kasokudo > 100) {
        basic.showLeds(`
            # # # . .
            # . . # .
            # # # . .
            # . . # .
            # . . . #
            `)
    } else {
        basic.showLeds(`
            . . . . .
            . . . . .
            . # # # .
            . . . . .
            . . . . .
            `)
    }
})
```

図 6 加速度と条件分岐 (JavaScript)

また，micro:bit を使わずに外部装置の学習を行う場合，Monaca であれば Monaca デバッガーアプリを使うことでスマートフォンを外部装置にして，加速度や GPS の値を取得できる．学習者のスマートフォンを BYOD で活用できれば micor:bit を買わずに実習することも可能である．

3.3 学習 13 基本的プログラム

学習 13 は「順次」「分岐」「反復」を利用した一般的な内容となっている．そのため特に言及はしない．

3.4 学習 14 応用的プログラム

学習 14 は「リスト」「乱数」「WebAPI」が取り上げられている．それぞれ，JavaScript での対応方法を検討したい．

3.4.1 リストについて

JavaScript では Array(配列)という仕組みがあり，教員研修資料で取り上げられている「リスト」に対応が可能である．研修資料に掲載されている Python のソースコード（図 7）を JavaScript のソースコードに置き換えると図 8 のようになる．

```
var a = [56, 3, 62, 1, 17, 87, 22, 36, 83, 21, 12];
var sum = 0;
for (var i = 0; i < 10; i++) {
    sum = sum + a[i];
}
console.log(sum);
```

図 7 リストを用いたプログラム (Python)

```
a=[56, 3, 62, 17, 87, 22, 36, 83, 21, 12]
sum = 0
for i in range(0, 10, 1):
    sum = sum+a[i]
print(sum)
```

図 8 リストを用いたプログラム (JavaScript)

3.4.2 乱数について

JavaScript では組み込みオブジェクトとして Math が存在し，Math.random()命令で乱数を生成できる．ただし，Python の random.randrange()命令のように任意の範囲から数字を返すような機能ではない．そこで，互換性のある関数を作成することを提案したい．学習 16 でも 1～6 の範囲で乱数を作る機能が必要となるため，このタイミング作成すれば後の学習でも活用できる．研修資料で取り上げられている Python のソースコード（図 9）を JavaScript のソースコードに置き換えると図 10 のようになる．

```
import random
a = 5
r = random.randrange(10)
if a==r:
  print(" 当たり")
elif a>r:
  print("a の方が大きい")
elif a<r:
  print("a の方が小さい")
```

図 9 乱数を用いたプログラム (Python)

```
function random(min, max) {
    var distance = max - min + 1;
    var value = Math.floor(Math.random() * distance) + min;
    return value;
}
var a = 5;
var r = random(0, 10);
if (a == r) {
    console.log("当たり");
} else if(a > r) {
    console.log("a の方が大きい");
} else if (a < r){
    console.log("a の方が小さい");
}
```

図 10 乱数を用いたプログラム (JavaScript)

3.4.3 WebAPI について

JavaScript には古くから XMLHttpRequest という仕組みがあり WebAPI を呼び出すことができるが記述は若干複雑である．そのため jQuery のような外部のライブラリを使うことも多いが，最近では fetch という仕組みが追加されたため IE 以外のブラウザであればライブラリ無しでも簡便な

記述で WebAPI を呼び出せる．

学習 14 では郵便番号を元に住所を調べる WebAPI をプログラムから呼び出している．研修資料に掲載されている Python のソースコード(図 11)を JavaScript のソースコードに置き換えると図 12 のようになる．

```
import requests
import json
url = "http://zipcloud.ibsnet.co.jp/api/search"
param = {"zipcode": "100-0013"}
res = requests.get(url,params=param)
response = json.loads(res.text)
address = response["results"][0]
print(address["address1"]    +    address["address2"]    +
address["address3"])
```

図 11 WebAPI を用いたプログラム (Python)

```
var url =
'http://zipcloud.ibsnet.co.jp/api/search?zipcode=100-0013';
fetch(url)
    .then(function(response) {
        return response.json();
    })
    .then(function(myJson) {
        var address = myJson.results[0];
        console.log(address.address1 + address.address2 +
address.address3);
    });
```

図 12 WebAPI を用いたプログラム (JavaScript)

なお，図 12 プログラムはブラウザで実行するとセキュリティの仕組みでブロックされる．これは，WebAPI 提供側のサイトである zipcloud が HTTPS 化されていないことと CORS の仕組みに対応していないためである．あくまでブラウザのセキュリティ機構に起因する問題のため，Monaca デバッガーで JavaScript を実行させる場合や Node.js で実行する場合には発生しない．Monaca デバッガーの場合はそのまま動作するが，Node.js で動かす場合には node-fetch パッケージが必要となる．

3.5 学習 15 アルゴリズムの比較

学習 15 では線形探索・二分探索・選択ソート・クイックソートのアルゴリズムが掲載されている．一般的な内容となっているため，特には言及しない．

3.6 学習 16 確定モデルと確率モデル

学習 16 では線グラフ・棒グラフ・散布図が登場する．また，Python 版ではグラフ描画ライブラリとして matplotlib を利用している．JavaScript に移植するためにはグラフライブラリの選定が必要となる．今回は plotly.js を利用することにした．plotly.js は Python 版や R 版も存在するため，研修で学んだ知識を複数の言語で活用できることが期待できる．Python のソースコード（図 13）を JavaScript のソースコードに置き換えると図 14 のようになる．なお JavaScript 版はグラフライブラリの読み込みや表示エリアの確保で HTML のソースコードが必要となるため図 15 で示す．また，matplotlib のグラフを図 16，plotly.js のグラフを図 17 に示す．

```
import matplotlib.pyplot as plt
riritsu = 0.05
yokin = [100000]
for i in range(10):
risoku = int(yokin[i]*riritsu)
yokin.append(yokin[i]+risoku)
plt.title("FUKURI KEISAN")
plt.xlabel("Year")
plt.ylabel("Yokin[YEN]")
plt.plot(yokin, marker="o")
plt.show()
```

図 13 線グラフのプログラム (Python)

```
function plot() {
    var riritsu = 0.05;
    var yokin = [100000];
    for (var i = 0; i < 10; i++) {
        var risoku = (yokin[i] * riritsu);
        yokin.push(yokin[i] + risoku);
    }
    var trace1 = {
        y: yokin,
        mode: 'lines+markers',
        type: 'scatter'
    };
    var layout = {
        title:"FUKURI KEISAN",
        xaxis: {title:"Year"},
        yaxis: {title:"Yokin[YEN]"}
    }
    var data = [trace1];
    Plotly.newPlot('myDiv', data, layout);
}
```

図 14 線グラフのプログラム (JavaScript)

```
<script
src="https://cdn.plot.ly/plotly-latest.min.js"></script>
<body onload="plot()">
    <div id="myDiv"></div>
</body>
```

図 15 線グラフのプログラム(HTML 部分)

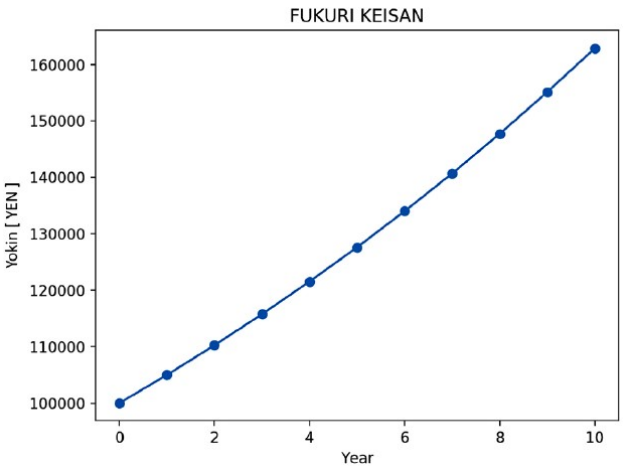

図 16 matplotlib のグラフ

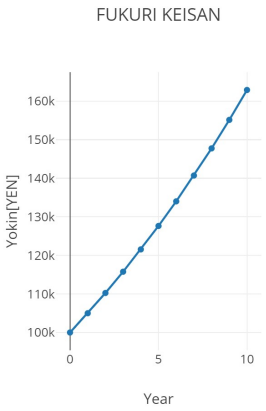

図 17 plotly.js のグラフ

plotly.js の方がソースコードは若干長くなるが、JavaScript でもグラフ描画を行えることが確認できた。

3.7 学習 21 さまざまな形式のデータとその表現形式

4 章の学習 21 では「(4) キー・バリュー形式のデータの処理・蓄積」にて Python のソースコードが掲載されている．こちらは JavaScript にはオブジェクトというデータ構造で表現できる．研修資料に掲載されている Python のソースコード（図 18）を JavaScript のソースコードに置き換えると図 19 のようになる．

```
members = [ { "name" : "A 田" ,
        "homephone" : "0**-***-1111" ,
        "mobilemail" : "ata@example.ne.jp"
    } ,
    { "name" : "B 川" ,
        "mobilephone" : "0**-***-2222"
    }
    ]
members.append( { "name" : "C 山" ,
   "mobilephone" : "090-****-3333" ,
   "email" : "cyama@example.com"
} )
for member in members:
    for key , value in member.items():
        print( key + ' : ' + value )
    print()
```

図 18 キーとバリューのプログラム (Python)

```
var members = [
    {
        name: "A 田",
        homephone: "0**-****-1111",
        mobileemail: "ata@example.ne.jp"
    },
    {
        name: "B 川",
        mobilephone: "080-****-2222",
    }
];
members.push(
    {
        name: "C 山",
        mobilephone: "090-****-3333",
        email: "cyama@example.com"
    }
);
console.log(members);
```

図 19 キーとバリューのプログラム (JavaScript)

4. おわりに

教員用研修教材で示されている内容を JavaScript で実施するうえで課題となるポイントを改めて確認したい．「micro:bit 対応」「グラフ描画」と「WebAPI」の 3 つは外部の装置やライブラリ，WebAPI サーバーに依存するため課題になると考えられる．現時点で「micro:bit 対応」に関しては Microsoft MakeCode を使えば対応できる．「グラフ描画」は plotly.js で対応できることが確認できた．Python で matplotlib を利用する場合に比べてソースコードは少し長くなったが教材に掲載できる範囲であると考える．

「WebAPI」も fetch を利用したソースコードは十分に短い．また，ブラウザのセキュリティ機構により Monaca デバッガーか Node.js が必要となる問題については HTTPS 通信と CORS に対応した教育および実験用の WebAPI サーバーを用意して対策することを今後検討したい．

参考文献

[1] 文部科学省, 高等学校情報科「情報 I」教員研修用教材, http://www.mext.go.jp/a_menu/shotou/zyouhou/detail/1416746.htm, (参照 2019-5-15)
[2] 兼宗 進, 本多 佑希, 高等学校「情報 I」の研修資料におけるプログラミング言語の扱い, 情報教育シンポジウム論文集, 2019, 168 – 175
[3]文部科学省, 令和 2 年度概算要求のポイント, http://www.mext.go.jp/component/b_menu/other/__icsFiles/afieldfile/2019/08/29/1420671_01-1.pdf, (参照 2019-9-10)