

研究ノート

楽しさで引っ張るプログラミング入門講座Ⅱ —Monacaを使ったスマホで動くプログラムの作成—

室谷 心

A Playful Learning Class for Programming II:
Making Smart Phone Applications Using "Monaca"

MUROYA Shin

要 旨

プログラミング教育が近年特に注目を集めているが、我々は学習者の動機を維持するために、“楽しさで引っ張る”ような教材の開発を進めている。マルチメディアに対応した楽しいアプリを目標とすることで、HTML5+JavaScriptという組み合わせでのスマートフォンアプリの作成はその一つの解答である、本稿ではスマートフォンアプリの作成というゴールに話を絞り、アシアル社が提供するMonacaを使う可能性について検討した。

キーワード

プログラミング教育 JavaScript スマートフォンアプリ

目 次

- I. はじめに
- II. Monaca
- III. JavaScriptを使ったプログラミング教育
- IV. JavaScriptからスマホのデバイスへのアクセス
- V. Monacaを使ったJavaScriptプログラミングの授業
- VI. まとめ

文献

I. はじめに

次期小学校学習指導要領や高等学校の情報Ⅰに取り入れられるという中教審の答申以来、プログラミング教育に対する社会の注目度が増加している。次期指導要領との関係で小学校でのプログラミング教育が特に注目されているが、例えば、「中学校においては、技術・家庭科(技術分野)においてプログラミング、情報セキュリティに関する内容を充実「計測・制御のプログラミング」に加え、「ネットワークを利用した双方向性のあるコンテンツのプログラミング」等について学ぶ。【技術・家庭科(技術分野)】¹⁾という記述にみられるように、中学校や高等学校においてもプログラミング教育の充実が盛り込まれている。

プログラミング学習導入段階での、スクラッチ²⁾のようなタイルプログラミングシステムの利用は、タイプミスや文法エラーの可能性を無くし、学習者は作成プログラムのアルゴリズムに集中することが可能となり、プログラミング学習の初期段階での敷居を大きく下げる。しかしながら、次の段階に展開する場合には、キー入力を伴うプログラミング言語の活用に進まざるを得ないのが現状である。その際には、タイルプログラミングでは起らなかった、入力ミスや文法エラーなどに遭遇し、学習者はそれまで経験したことのないデバッグ作業に対する忍耐を要求される。

我々は、まづもと広域ものづくりフェアにおいて、小学生を対象としたキッズプログラミング教室と³⁾、中高生を対象としたヤングプログラミング教室の2種類の公開講座を開催してきている。キッズプログラミング教室では、タイルプログラミングのスクラッチを使っているが、中高生対象のヤングプログラミング教室では、対象が中高生なのでより発展的なテーマとして、Raspberry Piを使った物理コンピューティング^{4,5)}とJavaScriptを使ったゲームの作成の2テーマを用意した。Raspberry PiはLinuxのOS操作があ

り、JavaScriptはプログラミング言語なので、いずれもキー入力や作業の複雑化は避けられない。学習者が飽きてしまわないように、興味で引っ張り続ける必要がある⁶⁾。そこで我々はJavaScriptを使ったプログラミング教室では、特に中高生の興味を引くために、学習者自身の持つスマートフォン(以下スマホと記す)での動作をゴールとした⁷⁾。

自分の持つスマホでの動作は、学習者の興味を大いに引くものであるが、文献6)で報告したように、所有スマホとしてはアンドロイドとiPhoneとが半々であり、いずれのOSのスマホであっても動作するような教材でなければ、全員が興味をもつような題材にはならない。また、特にiPhoneの場合には、保安のために、個人が作成したプログラムを簡単には実行できないようなシステムがとられている。

ヤングプログラミング教室では、このような状況の解決策としてJavaScriptを用いたシューティングゲームの作成を題材とした。文献7)において議論されているように、JavaScriptでゲームを作る場合には、画像スプライトの扱いをスクラッチでのプログラム作成と並行した形で行うことが可能であり、JavaScriptはポストタイルプログラミング段階での活用が期待されている。JavaScriptはHTMLの拡張機能から始まった、ブラウザ上で動作するスクリプト言語なので、基本的にはOSによらない言語である。しかし、そのスマホでの利用のためにはhttpのサーバーから利用する必要がある。文献6)の講座では自前のhttpサーバーを学内に設置して利用したが、一般的にはプログラミングの授業だけのために自前のサーバーを立て維持することは現実的ではない。また、レンタルサーバーの利用も、費用の問題やCGI利用の可否、ファイルのアップロードの自由度、同一ドメインでのアダルト系サイトとの共存など様々な問題がある。本稿では上記問題の一つの解決策として、Monacaの利用を検討することにする⁸⁾。

II. Monaca

Monacaはアシアル株式会社が提供する、クラウド上で動作する総合開発環境である⁹⁾。Webブラウザから利用し、エディットやデバッグに加えて、各種スマホに対応したエミュレーションによる動作確認ができる(図1)。さらに、AndroidとiOSのそれぞれに対応したデバッガーがアプリとして提供されており、google playやapple storeからMonacaデバッガーをダウンロードしインストールすれば、クラウド上のエディタにあるコードをデバッガーで動かすことによって、スマホ上での動作を実現することができる(図2)。すなわち、Monacaを利用することによって、JavaScriptで作成したプログラムをAndroidスマホとiPhoneのいずれでも動作させることが

可能となる。

アシアルはMonacaをOSを問わない共通言語(HTML/CSS/JavaScript)を使った、ワンソースで複数のOS向けのアプリを開発することのできる、クラウド型のハイブリッドアプリ開発プラットフォームと位置付けており、さらにAndroidとiOSそれぞれのネイティブアプリのビルドも可能なシステムとして提供している¹⁰⁾。アシアルホームページには実際にMonacaで開発されたアプリがリストアップされている。Monacaは商用のシステムであり、オフラインで動作するネイティブアプリのビルドを行うためには、有料の会員になる必要がある。本稿では、教育用に許されているfree会員の範囲での、教材利用の可能性を検討する。アプリのビルドを行わず、オンラインという制限付きであっても、ブラウザ上ではなく



図1. ブラウザGoogle Chromeで開いたMonacaの画面



図2. Monacaデバッガーのアイコン



図3. Androidタブレット上でのMonacaデバッガー画面

Monaca デバッガーで動かすことによって、実際にネイティブコードでの実行形式を動かしているのと変わらない使用感を体験することができる。

Ⅲ. JavaScriptを使ったプログラミング教育

プログラミング教育においてどの言語を利用するかは、教育目標と合わせてさまざまな議論のある問題である。文献6)のように、学習者が自身の持つ実機で動作させることを目指した場合、HTML5とJavaScriptという組み合わせはその一つの解答となる。

近年のネット環境とスマホの普及により、学習者のスマホ保有率はほぼ100%でありスマホでのブラウザ利用は標準的なものである。したがって、HTML5とJavaScriptを組み合わせたコードに関してはOSや機種によらず、インターネットの利用を前提としたスマホであるならば、間違いなく動作することが期待できる。実際にはブラウザによる対応のばらつきが考えられるが、現状ではブラウザとしてGoogle Chromeを利用することによって、よほど古いOSの機種を除き、ほぼすべてのスマホで同一の動作環境を実現することができる。逆に言えば、HTML5とJavaScriptを使ったコードならば、Google Chromeさえ動けばハードウェアやOSには拠らずに実行させることができる。

一方、JavaScriptの弱みとしては、HTMLと共に使用しブラウザで動作するスクリプト言語なので、動作が速くはないことに加えて、ローカルなマシンのハードウェアへのアクセスは基本的に想定されていない。すなわち、ファイルの読み書きやローカルマシンに接続されたデバイスの利用などが問題として考えられる。しかし現在のJavaScriptではCordovaやAPIの利用によって、デバイスへのアクセスやコントロール、ファイル操作もある程度可能となっている¹¹⁾。

文献6)でも報告したように、プログラミングを“楽しむ”という視点では、学習者が自分のスマホで動作させることは重要で、その場合スマホが持つ高度なデバイスを活用できることが、プログラム作成のさらなる満足感に直結することが予想される。パソコンではなく、スマホに固有のデバイスとしては「加速度センサー」、「GPS」、「カメラ」、「振動モータ」などが考えられる。これらを使うアプリが容易に作れるとしたら、作って楽しいプログラミング教材の可能性が大きく広がると期待できる。

Ⅳ. JavaScriptからスマホのデバイスへのアクセス

1. 加速度センサーへのアクセス

加速度センサーへのアクセスは、すでに2016年からのヤングプログラミング教室でも使っているが、`window.addEventListener("devicemotion", function(evt) {`によって取り出すことができる^{12,13)}(図4)。加速度センサーが返す値には、重力を除いた加速度を与える`event.acceleration`、重力を含めた加速度を与える`event.accelerationIncludingGravity`、回転速度を与える`event.rotationRate`の3種類のデータがある。iPhoneはデータの符号がアンドロイドとは逆になっているため、プログラムの中で使う際に符号を反転させておく必要がある。

```

20 // 加速度センサーの準備
21 ax = 0;
22 ay = 0;
23 az = 0;
24
25 window.addEventListener("devicemotion", function(evt) {
26   ax = evt.accelerationIncludingGravity.x; // 横方向の傾斜
27   ay = evt.accelerationIncludingGravity.y; // 縦方向の傾斜
28   az = evt.accelerationIncludingGravity.z; // 上下方向の傾斜
29   if (navigator.userAgent.indexOf('iPhone') > 0) {
30     ax=ax*(-1);
31     ay=ay*(-1);
32     az=az*(-1);
33   }
34 }, true); // 加速度センサーの値を (ax, ay, az) の中に入れる
35

```

図4. JavaScriptからの加速度センサーの利用

このWebAPIインタフェースを利用したコードはヤングプログラミング教室の教材の中ですでに利用しており、文献6)の教材の中にすでに組み込んである。後述するようにMonacaでの利用においても、正しく動作する。スマホの8軸センサーに関しては、“devicemotion”による加速度センサーの情報に加えて、“deviceorientation”を引数にしてジャイロセンサー(+地磁気センサー)の情報を取得できる¹⁴⁾。

2. GPSへのアクセス

GPSへのアクセスはGeolocation APIを利用して可能となる¹⁵⁾(図5)。後述のMonacaデバッガーを使ってアンドロイドタブレット上で動作させると、図6のようになる。ここで、位置精度の単

```

10  <script>
11  function test() {
12    navigator.geolocation.getCurrentPosition(test2);
13  }
14
15  function test2(position) {
16
17    var geo_text = "緯度:" + position.coords.latitude + "\n";
18    geo_text += "経度:" + position.coords.longitude + "\n";
19    geo_text += "高度:" + position.coords.altitude + "\n";
20    geo_text += "位置精度:" + position.coords.accuracy + "\n";
21    geo_text += "高度精度:" + position.coords.altitudeAccuracy + "\n";
22    geo_text += "移動方向:" + position.coords.heading + "\n";
23    geo_text += "速度:" + position.coords.speed + "\n";
24
25    var date = new Date(position.timestamp);
26
27    geo_text += "取得時刻:" + date.toLocaleString() + "\n";
28
29    alert(geo_text);
30  }
31  </script>

```

図5. JavaScriptからのGPS位置情報の利用

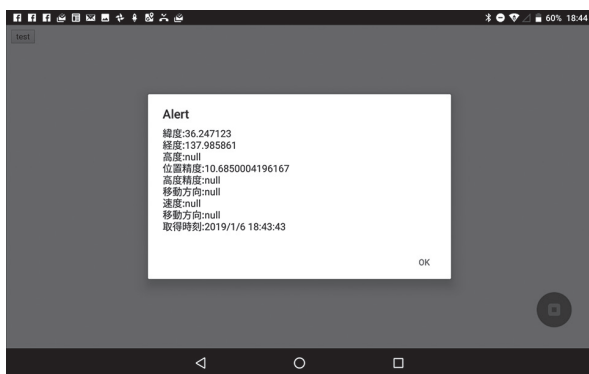


図6. Manacaデバッガーで位置情報を表示したスクリーンショット

位はmである。この緯度と経度の情報をGoogle Maps APIと組み合わせれば、現在位置を地図上に表示したり、地名や距離を表示したりするいわゆる地図アプリを作ることができる^{16,17)}。Google Map APIを使うには、APIキーが必要である。

3. カメラへのアクセス

カメラへのアクセスはMediaDevices.getUserMedia()メソッドを使う方法もあるが¹⁸⁾、MonacaはCordovaをサポートしているので、Cordovaからカメラを利用するのが容易である¹⁹⁾。CordovaはApacheが提供するオープンソースのモバイル開発フレームワークで、Webページ作成技術(HTML/CSS/JavaScript)を使って、さまざまなプラットフォーム用のアプリを開発できる²⁰⁾。Cordovaが提供するデバイス系のAPIを使用することによって、JavaScriptを使用してカメラ²¹⁾、ファイルシステム、端末側のストレージ

```

11  // 写真を撮る
12  function takePicture(){
13    var option = {
14      | sourceType: Camera.PictureSourceType.CAMERA, //
15      | saveToPhotoAlbum: true //
16    };
17
18    //
19    navigator.camera.getPicture(onSuccess, onError, option);
20
21    //
22    function onSuccess(imageURI){
23      | alert(imageURI + "に保存しました");
24    }
25    //
26    function onError(message){
27      | alert("エラー : " + message);
28    }
29  }
30  //
31  function loadPicture(){
32    var options = {
33      | sourceType: Camera.PictureSourceType.SAVEPHOTOALBUM //
34    };
35
36    //
37    navigator.camera.getPicture(onSuccess, onError, options);
38
39    //
40    function onSuccess(imageURI){
41      | document.getElementById("photo").src = imageURI;
42    }
43    //
44    function onError(message){
45      | alert("エラー : " + message);
46    }
47  }

```

図7. MonacaからCordovaを通してカメラにアクセスするJavaScriptのコード

など、端末側のネイティブ機能を使用することが可能となる²²⁾。後述するように、MonacaではCordovaをプラグインメニューから利用することができ、JavaScript経由で作成したハイブリッドアプリから容易に端末側のネイティブ機能へアクセスすることが可能である²³⁾。図7のコードを使えば、図8のように、Monacaデバッガーからカメラを起動して写真を撮り、写真フォルダーにファイルとして保存することができる。

以上のように、APIを通じてJavaScriptからスマホのデバイスに容易にアクセスすることができ、そのコードをMonacaデバッガーで動かすことによって、簡単にデバイスを利用するスマホアプリの作成を体験できる。

Monacaでは各種APIをプラグインとして追加利用できるようになっていて、基本APIとしては

- ・ バッテリー情報の取得 プラグイン
- ・ カメラ操作 プラグイン
- ・ 住所録の取得 プラグイン
- ・ 端末情報の取得 プラグイン
- ・ 端末のモーション検知 プラグイン
- ・ 端末のオリエンテーション検知 プラグイン
- ・ ダイアログの制御 プラグイン
- ・ ファイル操作 プラグイン
- ・ ファイル転送 プラグイン
- ・ 位置情報の取得 プラグイン



図8. MonacaデバッガーからJavaScriptを使って起動させたタブレットのカメラ

- ・ 表記の国際化対応 プラグイン
- ・ InAppBrowser プラグイン
- ・ メディア操作 プラグイン
- ・ メディアキャプチャー プラグイン
- ・ ネットワーク情報の取得 プラグイン
- ・ スプラッシュスクリーンの制御 プラグイン
- ・ バイブレーションの制御 プラグイン
- ・ ステータスバーの制御 プラグイン
- ・ ホワイトリストへの登録 プラグイン (Android専用)

が上げられている。例えば、バイブレーションはweb APIとしてvibration APIが提供されているが、現在のブラウザの対応状況は十分ではない²⁴⁾。一方Monacaを使った場合には、Cordova APIをプラグインとして利用できる(図9)。Cordova APIを利用する際には、通常はhtmlサーバーに組み込む必要があり、自前のサーバーで使うには一段階よけいな準備が必要となるので、MonacaがCordova APIをプラグインとして提供してくれることは、教材準備の立場からは重要なポイントである。これらは、Monacaデバッガーという統一環境を使う大きなご利益である。もちろんこれでハードウェアの壁が完全に消せたわけではなく、実際バイブレーションの場合には、Androidでは可能な振動時間の制御が、iOSではできないというような差異が残っている。とはい

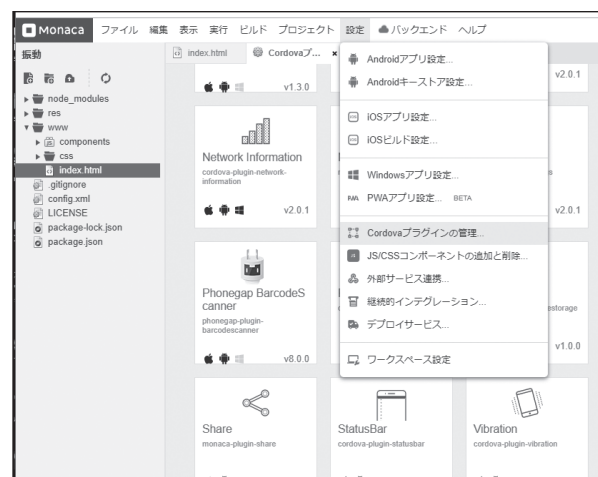


図9. Monacaプラグイン設定画面

え、Monacaの環境は上記のようなデバイス関連プログラムを容易に作ることができ、スマホアプリの開発環境として、十分機能することが期待できる。

V. Monacaを使ったJavaScriptプログラミングの授業

1. ヤングプログラミング教室

文献6)で報告したように、我々はヤングプログラミング教室として2016年2017年の2回の「まつもと広域ものづくりフェア」において、延べ4日間にわたって「1日プログラミング教室」を開講した。題材はJavaScriptライブラリーenchant.jsを使ったシューティングゲームの作成²⁵⁾で最後に、自前のサーバーにアップデートしスマホでの実行を行う計画であった。2016年は2日間とも全員がサーバーへのアップデートまで進み、スマホでの動作確認を行ったが、2017年は学内のネットワークが不調で、サーバーへのアップロードができず不満の残る終わり方であった。2018年は一日だけの開講でMonacaを使ってスマホで実行することを目指した。

2018年の講座でのプログラム作成のステップは、大きな流れは2016年同様であったが、作業工程を大幅に減らしプログラムの作成は主にコピー&ペーストで行い、時間をとって内容理解を目指したパラメータの変更作業を色々行った。2018年の作業工程は下記のようなものであった。

ヤングプログラミング教室2018の作業工程

0. テンプレートの利用

Webページの基本設定とJavaScriptの利用enchant.jsの読み込み部分

加速度センサーの準備

キャラクターの画像の読み込みと生成

ステージとアニメーションの設定

エディターによる修正と保存

ブラウザによる実行や再読み込み

1. キャラクターの動作

タッチパネルと加速度センサーによる動き

x、yの座標と移動

2. 敵登場

Spriteクラスから敵クラスの生成

敵の属性や機能の定義

敵を敵レイヤーの子プロセスとして生成

ローカル変数とグローバル変数の違い

3. 発射弾の作成

Spriteクラスから弾クラスの生成。

弾の属性(絵や動き)

Spriteクラスから敵弾クラスの生成

4. お互いの攻撃

弾の動きや発射タイミングの設定

敵弾との間の接触判定

子プロセスの消去の仕方

5. 敵に命中

弾と敵の接触判定

親プロセスに対して子プロセスの排除命令

6. 効果音

敵の爆発音を入れる。サウンドデータの扱い

7. 新しいクラスを定義して使う

新しく定義したクラスを使って新しい敵と敵弾の定義を簡単に書き直す

上記の7まで終えたのち、Monacaクラウドへのアップロードを行った(図10)。2018年ヤングプログラミング教室では、Monacaの利用アカウントはあらかじめこちらで作成しておいたものを参加者に配布した。必要なライブラリーや音データ画像データもあらかじめアップロードして準備を終えておいた。講座ではパソコン上のGoogle Chromeで動作確認をしながらプログラム作成を行っているため、同じくGoogle ChromeからMonacaにログインし、ファイルのアップデート

を行った。

作ったプログラムを各人のスマホで実行するには、学習者にMonacaデバッガーをインストールしてもらわなければならない、この作業には少し時間がかかった。また、パソコンの作業フォルダーからGoogle Chromeで実行確認をする際には、拡張子がhtmlであればファイル名は何でもよいが、Monacaデバッガーでの実行には、ターゲットファイルをindex.htmlと名前を変えてアップロードする必要がある。公開参加型のヤングプログラミング教室の場合には、参加者の既習知識がまちまちで、名前の変更のような基本的ファイル操作の怪しい参加者もいて、授業についてこれているかどうか注意が必要であった。パソコン上で動作確認をしたうえでアップロードしているので、各人のスマホのMonacaデバッガーでも当然のごとく正しく動作しヤングプログラミング教室は無事に終了した。

2. 大学授業での活用

2018年度総合経営学部2年配当の「マルチメディア活用論」においても、JavaScriptのプログラミングを行い、JavaScriptライブラリーenchant.jsを使ったスマホアプリの作成を行った。この授業の受講者は30人程度であったが、全員がスマホを

保有しており各自のスマホでの作成プログラムの動作確認を前提とした授業設計で問題はなかった。ネットワーク接続に関しても、授業中のMonacaサーバーサイトへの同時アクセスには現在のところ特に問題は生じていない。

大学の授業では、メールを使ったMonacaアカウント取得も、freeライセンスでの利用設定も学生各人で行ったので、パソコンのブラウザからのMonaca利用の導入とスマホへのデバッガーのインストールに授業一コマ分を費やしたが、結果としては、大学生の場合全員が時間内に終了し、その前回の授業までにパソコン上で作成したコードをスマホ上で動かすことに成功した(図11右下段)。

半期の授業の構成方針は、文献6)のヤングプログラミング教室と基本的には同様でありライブラリーとテンプレートの入ったフォルダーを学生に配布し、テラパッドでJavaScriptを記述したhtmlファイルを作成し、パソコン上のGoogle Chromeで実行・動作確認を行った。エラーが発生した場合には、テラパッドに戻りhtmlファイルの修正・保存の後、再びGoogle Chromeで実行させることになる。パソコン上でのJavaScriptプログラムの作成作業に一通り慣れた、7回目の授業でMonacaの利用を開始した(図11)。

Monacaデバッガーではプロジェクトのターゲットファイルはindex.htmlでなければならない、さらに、freeライセンスでの利用の場合、プロジェクトを同時には3つしか作れない。このため、授業では課題プログラムを漸進的に作っていても、Monacaでの動作確認のたびにプロジェクト単位での作成・消去を頻繁に行う必要がある。そこで、今年度の授業では、パソコン上では作業フォルダー内に階層を作らずに、必要なライブラリーや音や画像ファイルを平面的に配置し、Monacaへのファイル転送を一度の操作で済ませられるようにした(図11左下段)。

7回目以降の授業では、パソコンでのコーディング作業に加えて、毎時間いくつかのプログラム



図10. ヤングプログラミング教室での配布資料のMonaca関連ページ

のMonacaでの実行を行った。加速度センサーやタッチパネルの動作はスマホでなければ確認できないので、作成ゲームの操作に組み込んであることはMonaca利用の理由になる。アップデート作業をくりかえすことは、この作業に含まれる確認動作を通じて、ファイルのフォルダー管理や、

ファイル名の変更、ライブラリーやデータファイルの存在確認といった、パソコンの基本作業のトレーニングをしていることになる。前節で報告したヤングプログラミング教室の場合には、ファイルの名前変更やフォルダー間のファイル移動に問題のある受講生が見られたが、大学生の場合には

2018年第7回

Monaca 無料トライアルを使って スマホアプリを作る

プログラムファイルの名前を index.html にしないといけない。
ファイルをコピーしてから名前の変更をする

Monaca無料トライアルを使ってスマホアプリを作る 2018年第7回

課題の確認にしたいので
パスワードは学籍番号に
してください
17k□□□

アカウント作成

大学メールアドレス
14k990@s.matsu.ac.jp

パスワード
14k990

アカウント作成ボタンをクリックすると、利用規約に同意したとみなされます。

アカウント新規作成

GitHubアカウントで作成

既にアカウントをお持ちですか？ ログイン

スタート画面

Shin MUROYA
14k990@s.matsu.ac.jp
アカウント設定 ログアウト

トライアル期間中です (残り13日)
Pro

プラン管理

プランの種類 個人向けプラン

現在のプラン 月額 Pro
次回の契約更新日: 2018/11/17

プラン変更

アカウントのプラン管理から Free プランに切り替える

Freeプラン
個人で利用される方を対象としています。
- 最新の機能に先行して利用することができます。
- 請求書発行には対応していません。

Monaca Dashboard

新しいプロジェクトを作る

新しいプロジェクトを作る

テンプレートの種類

- サンプル アプリケーション
- フレームワークテンプレート
- 最小限のテンプレート

名前を付ける
説明を入れる
作成する

プロジェクトの情報を
11月6日

プロジェクトを選んで開く

アップロード

フォルダー www を選んでファイルのアップロード

必要なファイルを全部選んでドロップする
プログラムファイルは index.html という名前にしておく

確認

Confirm File Overwrite

The file "www/index.html" already exists.
Would you like to overwrite it?

置き換える

すべて書き換え

アップロードが終わったら、これを閉じる

ファイルアップロード

Monaca Debugger

HTMLハイブリッド テストを効率化

App Store や Google Play ストアで Monaca debugger を探してインストールする

スマホ画面もどきが動く

スマホにアプリを入れる Monaca debugger

Monaca debugger にログインして、プロジェクトを開く

図11. Monaca導入回のスライドの一部

これらの操作が問題になる学生は少なく、また、問題のある学生も毎時間のアップデート作業によって、問題なく一人でできるようになった。“楽しい”スマホでの動作が待っていることもあってか、学生は特に違和感無くパソコンでの“単調な”ファイル操作を繰り返し、習熟するようすが見て取れた。

「マルチメディア活用論」では、タッチパネル、加速度センサー、効果音を使ったゲームプログラムを作成し、スマホでの動作確認を行った。学生のスマホはOSもバージョンもまちまちであったが、スマホのカメラを起動する試験的なコードと合わせて、学生全員が自分のスマホでプログラムを動作させることができた。

表1は出席を兼ねた簡単なアンケートを取った結果である。第7回でMonacaの利用を始め第10回までは重力センサーとタッチパネルを使った鬼ごっこのようなゲームを作り、デバイスによるコントロールと、mp3による効果音の練習を行った、第11回からは、タッチパネルと乱数、分岐の練習としてもぐらたたきを作っている。第14回は課題演習を行っていた。アンケート中のmm-c8は標的の数や分岐を工夫する発展問題である。Monaca自体の動作は、受講生全員が利用できていることを授業中に確認している。表1の結果では、Monacaを使ってスマホで動作すること自体は、好意的に受け止められているが、プログラムの作成は難しいという、期待通りでありまた予想通り

の結果であった。

大学の授業であっても、おもしろさで興味を引っ張ることの重要性はヤングプログラミング教室と同じであり、難しそうなSwiftやC++を使わずに簡単にスマホアプリを作れそうな点は、学習への動機付けの効果の大きいことが期待できる。実際Monacaデバッガーを使って自分のスマホ上でプログラムが動いた時の学生の反応は、非常に良いものであった。

Ⅵ. まとめ

本稿では、Monacaとスマホを使ったプログラミングの授業の可能性を検討した。HTML5のリリース以来スタイルシート、JavaScriptとあわせてのアプリとしての利用の可能性が提案されてきた^{26,27)}。しかしながら、実際には、ブラウザによる対応が統一されておらず、また、htmlとしてオンラインの状態ですerverからの利用が基本であることなどによって、本格的なアプリの開発基盤としての活用はなかなか難しいところがあった。

本稿に述べたように、Monaca及びMonacaデバッガーの利用によってオンラインではあるが、ブラウザでの動作ではなくスマホの本当のアプリ同様の動作をさせることができる。この点は、学習者の動機付けや達成感に大きく寄与することが期待できる。もちろんアシアルは実際にビルドし

表1 授業アンケートの結果

第12回	Monacaでスマホプログラミング どうでしょう？	楽しい	まあまあ	つまらない	無意味である	
2018/12/11 (火)		8名 (20.5%)	22名 (56.4%)	2名 (5.1%)	1名 (2.6%)	
第14回	Monacaでもぐら たたきどうですか？	mm-c8 まで楽勝 です	授業は楽 ですが、 課題が大 変です	課題にな ると分か りません	授業につ いていけ ません	minacaっ てなんで すか？
2019/01/08 (火)		1名 (2.6%)	11名 (28.2%)	15名 (38.5%)	2名 (5.1%)	3名 (7.7%)

でネイティブアプリを作る環境としてMonacaを提供しているが、教育の現場での活用に関しては、本当にビルドしたアプリを作ることよりも、各種のデバイスにアクセスしたプログラムを作成し、デバッガーでの動作であっても実際に動かすという体験が、「楽しさで引っ張る」授業の重要な要素として働くと期待できる。学習者のスマホのオンラインでの利用に関しては、通信料金の問題やスマホを他の目的で使ってしまう、授業に集中しないなどの問題も考えられるが、教室のWi-Fi環境の整備やMonacaサーバーとの接続のみを許可するようなネットワークフィルターの利用によって、高校や中学での利用であっても対処は可能ではないかと考えている。

本稿での議論では、Monacaの開発環境としての機能は使っておらず、スマホ上での実行環境としてのみMonacaを使っている。コーディングとデバッグまでは、テラパッドとGoogle Chromeを利用してパソコン上で作業を行い、プログラムが完成した段階でMonacaにアップロードしデバッガーを使ってスマホ上で実行させている。5章で議論したように、スマホのいろいろなデバイスへのアクセスを行うためにMonacaは便利であるが、プラグインを使うには、Monacaの開発環境を使うことになる。開発環境はエディターとデバッガーを備えており、iPhoneやAndroidのバージョンを指定してのプレビューが可能である。しかしながら、CordovaプラグインのAPIまでは利用できないため、各種デバイスに関しては、実機で動作確認をせざるを得ない。図12は振動モーターへのアクセスをパソコンのGoogle Chromeでプレビューした画面である。Cordova APIが使えないので、“[Browsersync] Couldn't open browser”というエラーが出ている。

このコードをAndroidタブレットのMonacaデバッガーで動かすと図13のようなエラーがブラウザ上に表示され、入力ミスの行が指摘される。図14中の右下T1001というのが、デバッガーを動か

しているタブレットのモデル番号である。このタブレットは振動機能がついていないが、それに関しては、エラーの無いコードを動かしても図14の

```

Running monaca preview
> monaca-template-minimum@ monaca:preview /project
> npm run dev

WARN invalid config loglevel="notice"
> monaca-template-minimum@ dev /project
> browser-sync start --s www/ --watch --port 8080 --ui-port 8081

[Browsersync] Access URLs:
   Local: http://localhost:8080
  External: http://172.29.0.18:8080

   UI: http://localhost:8081
  UI External: http://172.29.0.18:8081

[Browsersync] Serving files from: www/
[Browsersync] Watching files...
[Browsersync] Couldn't open browser (if you are using BrowserSync in a headless environment, you might want to set the open option to false)
  
```

図12. 振動デバイスへのアクセスをパソコンでプレビューした際のターミナル画面

```

20
21
22 document.addEventListener("deviceready",
onDeviceReady, false);
23 function onDeviceReady() {
24   console.log(navigator.vibrate);
25 }
26
27 <!--
28 // 匿名関数内で実行
29 (function () {
  
```

Uncaught SyntaxError: missing) after argument list www/index.html:24 T1001

図13. Monacaデバッガーから送られてきたエラーメッセージ2

```

22 document.addEventListener("deviceready", onDeviceReady,
false);
23 function onDeviceReady() {
24   console.log(navigator.vibrate);
25 }
26
27 <!--
28 // 匿名関数内で実行
29 (function () {
  
```

Uncaught SyntaxError: missing) after argument list www/index.html:24 T1001

図14. Monacaデバッガーから送られてきたエラーメッセージ2

ようにブラウザ上のデバッガー Debug 画面ならびに LOG 画面に警告が表示される。ブラウザと連動させて使うことによって、デバッガーのデバッグ機能の効果的な活用が期待できる。

デバイスへのアクセスでもパソコン画面の Chrome 上で使えるものもある。図16は Cordova API を使わずに、位置情報表示を表示させたもので、Google Chrome による位置表示である。パソコンのブラウザ上でも位置情報を使ったプログラムを利用することは可能であるが、タブレットに比べて位置精度は非常に悪い。

HTML5 になり自由度が大きくなって以降、プログラミングの教材として JavaScript を使うことが多くなったが、基本は文献6)で行ったような、サーバーを用意し http で提供されたコード

をブラウザで動かすというものであり、プログラミングの教育ではあっても、アプリの開発とはいいがたいものであった。API の活用によって、HTML+CSS+JavaScript という組み合わせでハイブリッドアプリ開発が行われるようになってきたが、開発環境としては API も提供できるサーバーの準備が必要で、Apache サーバーや JavaScript でかかれた node.js を立ち上げそこに組み込むことが多い。また、クロスプラットフォームアプリとしていろいろな OS に対応するためには、各 OS ごとのプラグインが必要となる。これらを教材準備として教員がハンドメイドで準備することは容易ではなく、高校や中学の現場では非現実的であろう。従ってアシアルが提供する Monaca の開発環境^{28,29)}の教育現場での利用価値は高いといえる。

ここで議論した教材としては、加速度センサーやタッチパネル、音データなどを使ったゲーム程度しか利用できていない。せっかくスマホのいろいろなデバイスへのアクセスが可能となったので、より学習者の興味を引き、将来の実用的なアプリ開発につながるような“楽しいプログラミング課題”を開発していくことが今後の課題である。



図15. パソコン上での位置情報

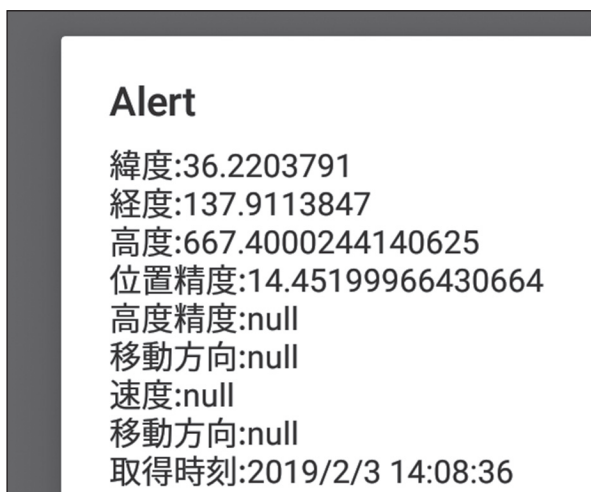


図16. Android タブレットの位置情報

文献

- 1) 文部科学省, 「新学習要領について」,
http://www.mext.go.jp/b_menu/shingi/chousa/shisetu/044/shiryo/_icsFiles/afieldfile/2018/07/09/1405957_003.pdf (2019年1月9日閲覧).
- 2) スクラッチホームページ, <https://scratch.mit.edu/> (2019年1月9日閲覧).
- 3) 室谷心, 「地域イベントで子供に教える機会を利用した情報科教職課程学生の教育」, 日本情報科教育学会誌, Vol.8, No.1, pp.61-69 (2015).
- 4) 矢野口聡, 室谷心, 「組み込みプログラミング学習教材と支援ツールの考案」, 日本教育工学会第32回全国大会論文集, 3a-B218-01.
- 5) 室谷心, 矢野口聡, 「地域イベントでの中高生向け一日プログラミング教室用教材の開発」, 第41回教育システム情報学会全国大会講演論文集, B4-2.
- 6) 室谷心, 矢野口聡, 浅見(林)大輔, 「楽しさで引張るプログラミング入門講座: JavaScriptを使った1日プログラミング教室用教材の開発と試用」, 松本大学研究紀要16, pp75-82, 2018.
- 7) 浅見(林)大輔, 室谷心, 「ScratchとJavaScriptによるプログラミング教育の実践」, 第10回全国大会講演論文集, pp113-114, 情報科教育学会, (2017).
- 8) 浅見大輔, 室谷心, 「JavaScriptによるスマートフォンアプリ開発の実践」, 日本情報科教育学会第11回全国大会講演論文集, pp.17-18情報科教育学会, (2018).
- 9) Monaca ホームページ, <http://Monaca.io/> (2019年1月9日閲覧).
- 10) Monaca プレスリリース, <https://www.asial.co.jp/pressrelease/411> (2019年1月9日閲覧).
- 11) 園田誠, 「作りながら学ぶHTML5 API入門」, 日経ソフトウェア2019年1月号第2付録
- 12) web API, DeviceMotionEvent 解説,
<https://developer.mozilla.org/en-US/docs/Web/API/DeviceMotionEvent> (2019年1月9日閲覧).
- 13) Corredor ウェブ, 「プログラミングの勉強メモ」,
<http://neos21.hatenablog.com/entry/2017/10/21/080000> (2019年1月9日閲覧).
- 14) 「スマホの8軸センサーの情報をHTML5で利用する方法」, <https://meiya.jp/?p=160> (2019年1月9日閲覧).
- 15) Geolocation API 解説,
https://developer.mozilla.org/ja/docs/Web/API/Geolocation/Using_geolocation (2019年1月9日閲覧).
- 16) Google Maps Platform, <https://cloud.google.com/maps-platform/?hl=ja> (2019年1月9日閲覧).
- 17) Google map API を使ってみた,
<https://qiita.com/Haruka-Ogawa/items/997401a2edcd20e61037> (2019年1月9日閲覧).
- 18) webRCT カメラの利用
<https://creatorsblog.nijibox.jp/browsercamera/>
(2019年1月9日閲覧).
- 19) 「Monaca で学ぶ初めてのプログラミング」, アシアル株式会社, 2016年3月.
- 20) Apache Cordova,
<https://cordova.apache.org/docs/ja/latest/guide/overview/index.html> (2019年1月9日 閲覧).
- 21) Apache Cordova, camera.getPicture,
<https://cordova.apache.org/docs/ja/3.1.0/cordova/camera/camera.getPicture.html> (2019年1月9日閲覧).
- 22) Monaca ホームページ,
https://docs.Monaca.io/ja/products_guide/Monaca_cli/dependencies/cordova_plugin/
(2019年1月9日閲覧).
- 23) Monaca Docs, https://docs.Monaca.io/ja/reference/cordova_7.1/ (2019年1月9日閲覧).
- 24) Vibration API, <https://developer.mozilla.org/ja/docs/Web/Guide/API/Vibration> (2019年1月9日閲覧).
- 25) 布留川英一, 伏見遼平, 田中諒, 「ゼロからは始める enchant.js 入門」, 株式会社アスキーメディアワークス (2012年2月).
- 26) 江見圭司, 「HTML5時代の情報教育」, 日本情報科教育学会 第7回全国大会講演論文集, pp87-88情報科教育学会, (2014).
- 27) 例えば, くじら飛行機, 「スマートフォンのためのHTML5アプリケーション開発ガイド」, ソシム株式会社 (2010年10月).
くじら 飛行機, 「基本から学ぶHTML5+JavaScript」, ソフトバンククリエイティブ株式会社 (2012年10月).
- 28) 「ハイブリッドアプリ入門[cordova ionic」,
<https://qiita.com/hironaito/items/9690c0757dd345cd5917>, (2019年2月3日 閲覧).
- 29) 「【ハイブリッドアプリ】Monaca を利用してみ
てはじめて疑問に思ったこと一覧」
<https://qiita.com/hironaito/items/a33f71d72df4fd06b191>, (2019年2月3日 閲覧).