Apache Cordovaを用いたハイブリッドアプリケーション開発教材

Apache Cordova を用いたハイブリッドアプリケーション開発教材

Educational Material for Development of Hybrid Application using Apache Cordova

幸谷智紀*
Tomonori KOUYA

Abstract: Hybrid application is a kind of single page applications built with HTML5 techique and rendering engine used in Web browsers, which has a benefit that is possible to run on any OS environment such as Windows, macOS, iOS and Android. Although it is almost slower than a native application developped by compiler languages such as C/C++, it is not necessary to rebuild its source codes written in HTML and JavaScript in order to be applied to various OSs. In this paper, we describe our educational material to develop hybrid application using Apache Cordova, one of development framework to build hybrid application.

1. 初めに

ハイブリットアプリケーションとは，HTML5 技術を用いて構築された Single Page application の一種で，ブラウザに内包されているレンダリングエンジンと組み合わせてパッケージングすることで，どんな OS 環境でも動作するという特徴がある．コンパイラ言語で作るネイティブアプリケーションに比べると動作は遅いが，HTML と JavaScript だけで構築できるハイブリットアプリケーションは OS ごとに作り替える必要がなく，メンテナンス性に優れている．

本稿では，Apache Cordova を用いて，主として Android の環境下で動作するハイブリットアプリケーションを構築する学生実験について述べる．

2. Web アプリケーションとハイブリッドアプリケーション

幸谷研究室では，卒業研究の一環として Web アプリケーション開発を行ってきた．既に 10 年以上の歴史があり，代々の卒業研究制作物の一部は，本研究室の Web サーバ (`https://cs-tklab.na-inet.jp/` もしくは `https://cs-tklab.sist.ac.jp/`) で閲覧することができる．担当教員は本来，科学技術シミュレーションにおける高性能計算法を研究テーマとしているが，過去に Ethernet を用いた分散メモリ型の PC クラスタを構築してきた経験から，ネットワーク技法を卒研テーマの一つにしてきた時期があり，その延長上で Web アプリケーションの開発を主テーマの一つとするに至った．

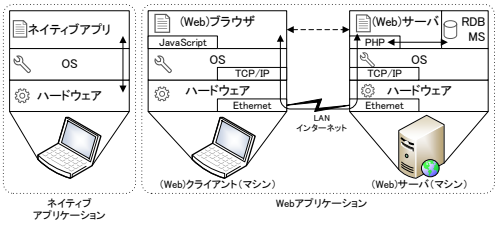

Fig. 1: ネイティブアプリケーションと Web アプリケーションのソフトウェア階層

Fig.1 に示すように，Web アプリケーション開発のためには，クライアント側 (ブラウザ) のインターフェースを担当する HTML，CSS，JavaScript を学び，サーバ側ではデータを保持するためのデータベースとのやり取りを行う PHP 等の言語を学ぶ必要がある．これらを網羅的に学習することは現実的ではないため，本研究室では卒業研究の前哨戦に当たる 3 年生後期の「コンピュータシステム実践演習 2」(旧「情報セミナー 2」) で独自の教材「Web アプリケーション開発入門」[1] を使用している．詳細については紀要論文 [2] を参照されたい．これはサーバ側の PHP にも HTML や CSS を埋め込み，サーバ側とクライアント側とでネットを介した協調動作を行うアプリケーションの開発を主眼としている．

それに対し，近年はクライアント側はもっぱら HTML+CSS で静的インターフェースを固め，JavaScript でユーザの要求に応じた画面更新やデータのやり取りを極力単独の HTML ファイルだけで完結させる SPA(Single Page Application) が主流となっている．SPA のための JavaScript フレームワークとしては，Angular[8]，React[9]，Vue[10] などがあり，Node.js[6] が提供するブラウザのレンダリングエンジンと組み合わせることで，単独のネイティブアプリケーションとしても利用できるようになっている．Atom[11] のように，Electron[12] というフレームワークで構築されたネイティブアプリケーションは，動作的に重たく感じることもあるものの，広く利用されている．

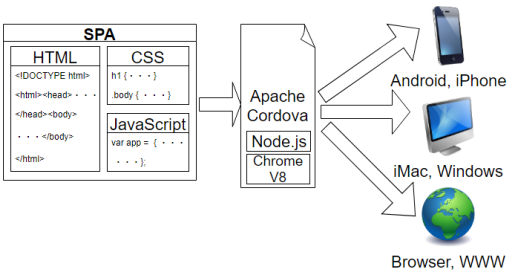

Fig. 2: Apache Cordova によるハイブリッドアプリケーション生成

Apache Cordova も，SPA によるアプリケーション開発環境の一つで，商用の開発フレームワークがオープンソースされたものである．現在はその上に iOS アプリケーションを構築できる有償の開発環境 [14] や，Onsen インターフェース [13] なども提供

2019 年 6 月 28 日 受理
*情報学部 コンピュータシステム学科

されるようになっている．2018 年度のコンピュータシステム実験で利用したのは Apache Cordova 8.0 であるが，Windows 環境下であろうと，Linux 環境下であろうと，macOS 環境下であろうと，Fig.2 に示すように，Web 上の SPA(WWW)，Android，Windows，iOS(エミュレータは macOS でのみ実行可) で動作する SPA に基づくネイティブアプリケーションが構築できる．

本実験を行うにあたり，当初は Linux 環境下での実験環境構築を目指したが，Android エミュレータの動作が不安定であるため断念し，Windows 環境下で実施することとした．iOS アプリケーションを作成したい学生向けに，iMac の開発環境も構築したかったが，予算執行時期が間に合わず，今回は iMac 環境下で Windows 同様の Apache Cordova 開発環境が構築できることのみ確認した．

3. Apache Cordova 開発教材について

Apache Cordova は Node.js 上で動作する開発フレームワークである．Android で動作する実行ファイルの動作確認のためには，Java で開発を行うための Android Studio と，エミュレータが必要となる．今回使用する開発環境では，少なくとも Android アプリケーション開発ができることを受講生に体感して欲しいと考え，Fig.3 のように必要となる環境をあらかじめ用意し，その上で Apache Cordova を組み込んだ．

加えて，ソースコードの編集には Microsoft Visual Studio Code，ブラウザは Node.js と共通のレンダリングエンジンを持つ Google Chrome を使用した．

既存の Web テキスト [2] に，第 6 章として「Apache Cordova を用いたハイブリッドアプリケーションの開発」を設け，下記の 5 節の中に，作成課題 6 件を埋め込んだ．最初の節では，既存の Web テキストの HTML, CSS，JavaScript の解説ページにリンクを張り，Web ページ作成の未経験者への配慮を行っている．

1. ハイブリッドアプリケーションと Apache Cordova ･･･ 課題 1
2. 計算アプリの作成 ･･･ 課題 2，課題 3，課題 4
3. Indexed DB を用いた住所録 ･･･ 課題 5
4. じゃんけんアプリの作成 ･･･ 課題 6
5. 課題レポート

2018 年度は 51 名の受講生がこの実験を 2 週間，計 6 コマで体験した．最後の課題 5 と 6 は重量級の内容であるため，どちらか片方だけで良いとしたが，最終的に課題 1〜4 ＋課題 5 or 課題 6 まで，時間内に完遂できたのは 1 名だけであった．おおかたの受講生は課題 4 までで時間切れとなっていた．

以下，各課題について概説する．

3.1 課題 1: SPA の作成

HTML による Web ページ作成の経験はあっても，DOM (Document Object Model) の概念については全く知らないというのが，本学の平均的なコンピュータシステム学科 3 年生である．JavaScript で自在に HTML のタグの内容を書き換えたり，タグそのものを生成したり消去したりするためには，DOM について，ある程度の理解は必要不可欠である．

そこで，簡単なテキストボックスを使った DOM の内容変更を行う SPA を最初に作成してもらった．雛形としては 23 行の HTML を用意し，そこに穴埋め形式で JavaScript を完成させてもらい (Fig.4)，各タグについた id で要素を指定することができ，その内容についても変更が可能であることを理解してもらうようにした．

Fig.5 に示すように，このやり方で順次テキストボックスに入力された文字列をメッセージボックスに出力したり，他の HTML 要素に埋め込んだりできることを理解してもらうのが，課題 1 の目的である．

この段階で躓く学生はごく少ない．概ね 20 分ほどで完了していた．

3.2 課題 2: Apache Cordova の使用方法

SPA の初歩が理解できたところで，Apache Cordova によるアプリケーション構築体験を行う．

Fig.6 に示すように，最初はコマンドラインから cordova コマンドが動作することを，バージョン番号と共に確認させ，しかる後に，

(1) “firstapp”という名前で最初の Apache Cordova アプリの雛形を自動生成させる
(2) “firstapp”フォルダに移動
(3) Apache Cordova で制作できるアプリケーション環境の確認
(4) ここでは，Android(android)，ブラウザ (クライアント側)(browser)，iOS(ios)，macOS(osx)，Windows(windows)，サーバ側設置 (www) のプラットフォーム (環境) で動作するアプリケーションの開発が可能と分かる

を確認する．

コマンドラインによる最初のアプリケーション雛形ができたら，生成したファイルの一覧を確認してもらう (Fig.7)．

開発する際には，www フォルダにある HTML ファイル，CSS ファイル，JavaScript ファイルを編集する．browser や www は通常の SPA なのでそのままで良いが，他のプラットフォーム上で動作するネイティブアプリケーションとするには，実行ファイルを生成する必要がある．そのために，各プラットフォーム用のフォルダが用意されており，コンパイルするとここに必要なファイルが生成される．

例として，Android アプリケーションを生成するケースを Fig.8 に示す．手順としては

1. Android 用にビルドする（ここが一番時間がかかる）
2. 生成した Android アプリをエミュレータで起動する
3. Android エミュレータ上でアプリケーションが起動して表示される

となる (Fig.8)．

Linux 環境化 (Ubuntu 16.04) では，この Android エミュレータの安定動作が難しかった．Windows 上でもたまにハングアップすることがあり，強制終了してやり直すということを何度か経験した．

この段階まで解説した後，課題 2 としては最下行に自分の名前を入れるというものを提示した．あらかじめ HTML ファイルに記述してある address タグの中に入力するだけなので，ここでまごつく受講生は皆無であった．

3.3 課題 3: 数式計算機の作成

課題 3 は，式パーサ機能を使った数式計算機の機能を追加するというものである．HTML にテキストボックスを作り，ボタンをクリックしたタイミングで，数式として解釈できる入

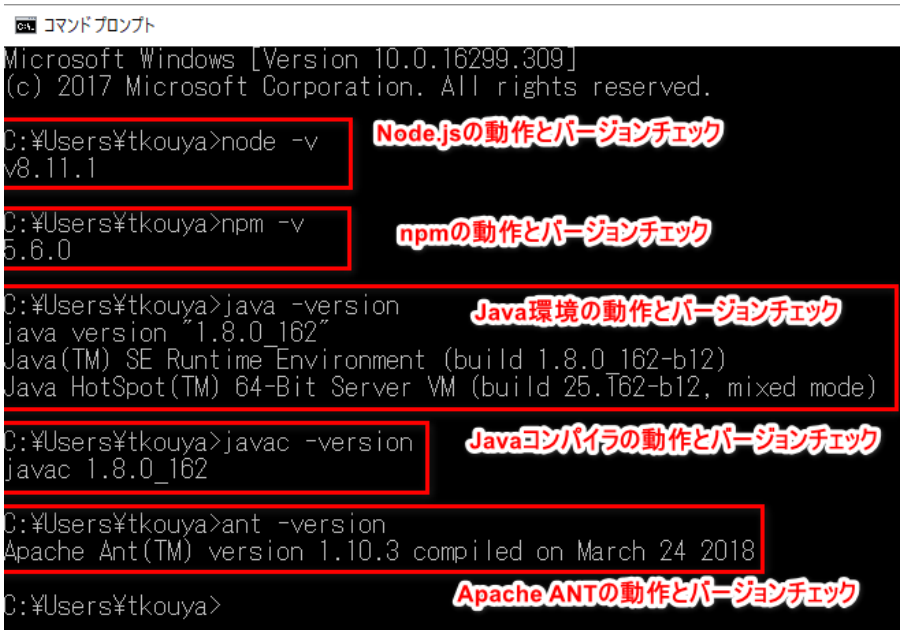

Fig. 3: Apache Cordova に必要なソフトとコマンド

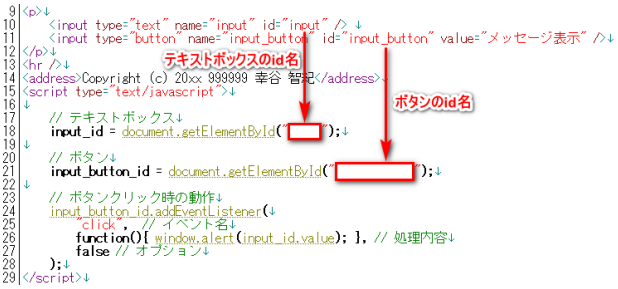

Fig. 4: HTML の雛形に埋め込んだ JavaScript

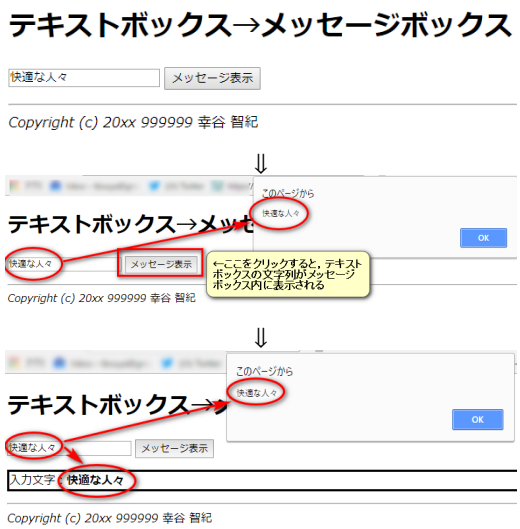

Fig. 5: 最初の SPA

Fig. 6: 最初の Apache Cordova アプリ作成

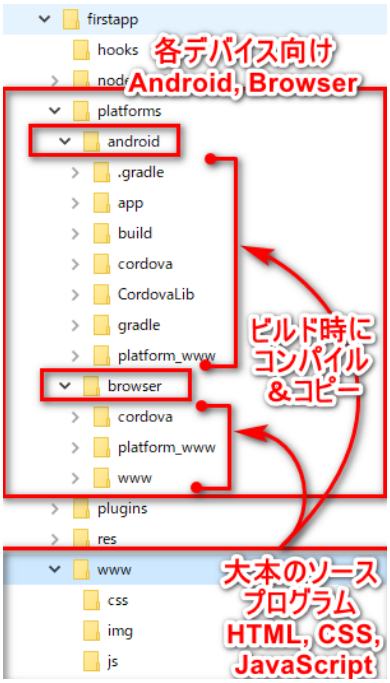

Fig. 7: Apache Cordova アプリのディレクトリ構造

1. Android 用にビルド

```
C:¥Users¥tkouya¥firstapp>cordova build android
Android Studio project detected
ANDROID_HOME=C:¥Users¥tkouya¥AppData¥Local¥Android¥sdk
JAVA_HOME=C:¥Program Files¥Java¥jdk1.8.0_162
studio
Starting a Gradle Daemon (subsequent builds will be faster)

BUILD SUCCESSFUL in 8s
```

⇓

2. 生成した Android アプリをエミュレータで起動

```
C:¥Users¥tkouya¥firstapp>cordova emulate android
Android Studio project detected
ANDROID_HOME=C:¥Users¥tkouya¥AppData¥Local¥Android¥sdk
JAVA_HOME=C:¥Program Files¥Java¥jdk1.8.0_162
studio
Subproject Path: CordovaLib
```

⇓

3. Android エミュレータ上でアプリケーションが起動

Fig. 8: Android エミュレータによる動作確認

力文字列を JavaScript で受け取り，計算結果を表示するというものである．ここで初めて指定 ID を持つフッタ HTML 要素を作成し，そこから文字列を受け取って処理を行うというハイブリッドアプリケーションを一から作成することになる．

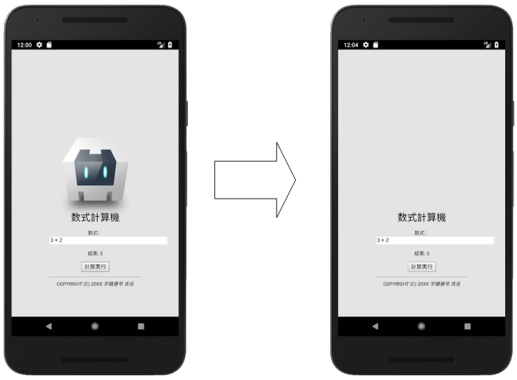

Fig. 9: 数式計算機

Fig.9 にその実行画面を示す．中央のロゴは不要なのでカットし，数式計算機として必要なものだけを画面に残してもらうようにした．この場合は「3 + 2」という数式が入力されているので，テキストボックスの下に「結果 : 5」と表示されている．

教材には HTML の変更箇所や JavaScript の例まで張り付けてあり，どちらも 30 行未満の修正で済むことから，ここでもまごつく受講生は殆どいなかった．

3.4 課題 4: 2 次方程式の解計算

思考力を試すため，初めて JavaScript のコードを自力で穴埋めしてもらうよう，課題 4 を設定した．2 次方程式 $ax^2+bx+c=0$ の係数 a, b, c をテキストボックスから入力し，解を公式に基づいて計算して表示するというものである (Fig.10).

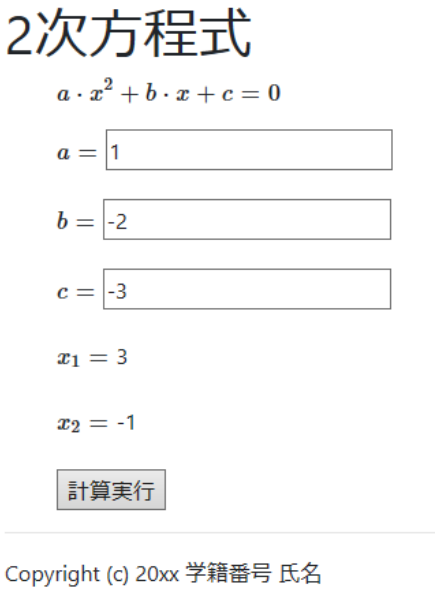

Fig. 10: 2 次方程式の解計算

2 次方程式を解くことができることを確認した後は，$a = 0$ のケースにも対応できるよう，一次方程式 $bx + c = 0$ を解く部分を穴埋め式に記述してもらうこととした．

さほど難しくないと考えたが，まず最初の 2 次方程式の解を解く JavaScript の作成で手間取り，ソースコードは全て教材に提示してあるにも関わらず，正しく動作させるまでに手間取る受講生が少数ながら見受けられた．入力コードにミスがあっても，どこがおかしいかを特定するまでに時間かかったようで，まずブラウザから実行を確認させ，「デベロッパーツール」からコンソールの表示と，JavaScript のエラーが確認できることを教える必要があった．

また，2 次方程式は解けるようになっても，1 次方程式を解く処理を記述できずに時間切れになる受講生も多かった．JavaScript の文法に慣れていないことを考慮しても，頭を使ってアルゴリズムを考え，それに従ってコーディングするという，平均的なプログラマなら常に行っている筈の作業に習熟していないと，筆者には感じられた．また，実験時間の終わりが近づいており，焦って余裕がなくなっているせいであるとも考えられる．この課題の前に，もう少し易しめの課題を追加することが望ましいのかもしれない．

3.5 課題 5: Indexed Database を用いた住所録

前述した通り，課題 5 と課題 6 は選択制とし，それぞれ各ステップを踏んで徐々に完成形に近づけていく内容とした．

課題 5 は，データを保持するためのデータベースを作り，住所録として動作させるというものである (Fig.11)．ブラウザ側でデータベースを扱うための仕組みは幾つか存在しているが，今回は最新のレンダリングエンジンのみを使うことを想定しているため，indexed database を使うことにした．

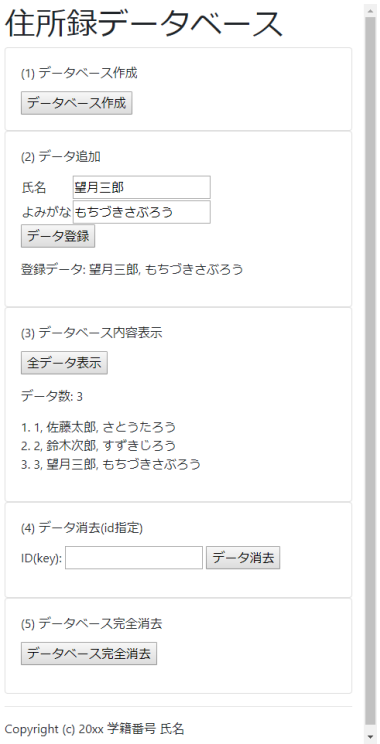

Fig. 11: Indexed Database を用いた住所録アプリ

作成手順は下記の (1)〜(5) の 5 ステップである．

(1) データベースの作成
(2) データ追加
(3) データベース内容表示
(4) データ消去 (id 指定)
(5) データベース完全消去

データベースを作成した後は，データの追記，内容表示，消去，データベースの完全消去を行うようにしてある．CRUD (Create, Read, Update, Delete) のうち，Update に相当する更新処理がないのは，他の処理に比べて手間がかかるため，実験時間内に終了しないのではという危惧があったからである．たとえ完遂しないとしても，やり方ぐらいは書いておいても良かったかもしれない．

ここまでたどり着く学生は少数派で，時間内に完成に至った学生は数人に留まった．じっくり内容確認する時間もないため，もう少しゆとりのある所で実施すべきテーマだったかもしれない．フォームと画面推移を制御する JavaScript コードを主とした index.html が 192 行で，データベースを制御する純粋 JavaScript コードが 230 行というものであったが，本学の平均レベルの学生には荷の思い内容であった．

3.6 課題 6: じゃんけんアプリ

課題 5 で実用的なアプリを作成する内容を提示したので，課題 6 はじゃんけんをするだけの簡単なゲームを作成することとした．

Fig.12 に示すように，まずテキスト表示だけでじゃんけんを行う骨組みを作り，これが動いたことを確認した後，それぞれの手の画像ファイルと勝敗画像ファイルをテキスト版のアプリにはめ込むという手順で作成するようにした (Fig.13)．

Fig. 12: じゃんけんアプリ：テキスト版

グラフィックス版の画面は Fig.14 にあるようなものになる．テキスト表示とは異なり，画像があるだけで印象が全く変わることを体感的に理解してもらうことを目的とした．

とはいえ，本テーマにチャレンジする学生はごく少数で，最後のグラフィック版まで完成させた学生は一人だけであった．

4. まとめと今後の展開

本研究室が作成してきた Web アプリケーション作成教材に追記する形で構築した Apache Cordova アプリケーション開発教材は，粗削りながら，学生の能力に応じたところまで進める内容になっていることが確認できた．

時間の関係上，Android アプリケーションはエミュレータの動作確認を行うだけにとどまったが，もう少し時間の余裕が

Fig. 13: テキスト版からグラフィックス版への作成手順の解説

Fig. 14: じゃんけんアプリ:グラフィックス版

あるところで，実機による動作確認と，iOS によるアプリケーション作成までできることを確認できるよう，本教材をブラッシュアップしていきたいと考えている．

謝辞

本教材作成は静岡理工科大学教育プロジェクト (B) の助成を受けて実施されたものである．関係各位に感謝する．また，2018 年度実験では SA として小澤皇太君と，松下孟樹君が，懇切丁寧な指導を行ってくれた．両人の献身にも感謝したい．丁寧な査読を行って頂いた未知の査読者にも感謝する．

参考文献

1) Web 教材「Web アプリケーション開発入門」, https://cs-tklab.na-inet.jp/phpdb/
2) 幸谷智紀，学生による学生のための Web プログラミング教材開発, 静岡理工科大学紀要, Vol.25, pp.169-174, 2018.
3) W3C, HTML5, https://www.w3.org/TR/html5/
4) W3C, CSS 2.x, https://www.w3.org/TR/CSS2/
5) Apache Foundation, Apache, https://www.apache.org/
6) Node.js, https://nodejs.org/
7) Apache Cordova, https://cordova.apache.org/
8) Angular, https://angular.io/
9) React, https://reactjs.org/
10) Vue.js, https://vuejs.org/
11) Atom, https://electronjs.org/apps/atom
12) Electron, https://electronjs.org/
13) Onsen UI, https://onsen.io/
14) IONIC, https://ionicframework.com/