平成２７年度

卒業論文

題目	Android を用いた時間割アプリの企画と開発

担当教員 （自署）		印

学籍番号　201214066

氏　　名　高木　宏基

広島経済大学

広島経済大学

要旨

第１章でスマートフォンの現状と Android、iOS のシェアについて述べている。第２章では、Android アプリの開発経緯、開発環境、制作したアプリの操作方法について述べている。詳しくは第２章で述べるが、開発環境は「eclipse」を使用し、開発したアプリは「時間割アプリ」である。第３章では、アプリの設計について述べている。制作したアプリのプログラムの解説は図を用いて述べている。第４章では、制作したアプリについてアンケートを取り、その集計結果について述べている。第５章では、本論文のまとめであり、アプリのこれからの発展や感想などを述べている。

目次

第１章　はじめに

現在、多くの人たちがスマートフォンを持っている。筆者も使っているスマートフォンであるが、そもそもスマートフォンとはどのようなものなのだろうか。スマートフォンとは、パソコンのような機能を持った高性能な携帯電話である。インターネット上でパソコン用のウェブ画面も開くことができるため、小型のパソコンともいえるだろう。現時点では、大きく分けて Android、iOS の 2 種類のスマートフォンが普及している。また、スマートフォンはインターネット上にあるマーケットから様々なアプリケーション（以降、アプリと略記する）をダウンロードすることで、自分の端末を好きなようにカスタマイズをすることができる。そのため、自分の好みにあった使い方ができるようになる。

先にも述べたが、現在のスマートフォンのシェアは Android、iOS の 2 強となっている。日本では図 1 にもあるように Android は 30.5%、iOS では 68.7%の割合となっており、圧倒的に iOS の方が人気となっている。では、海外ではどうなのだろうか。図 2 を見てみると日本では iOS が人気を誇っていたが、世界的に見れば Android の方が圧倒的に普及していることがわかる。スペインにいたっては iOS の普及率はわずか 7.2%となっており Android が圧倒的な人気となっている。このように世界と日本とではスマートフォンの普及率が正反対であることがわかる。本論文ではこの Android に着目し、アプリの開発を行った。

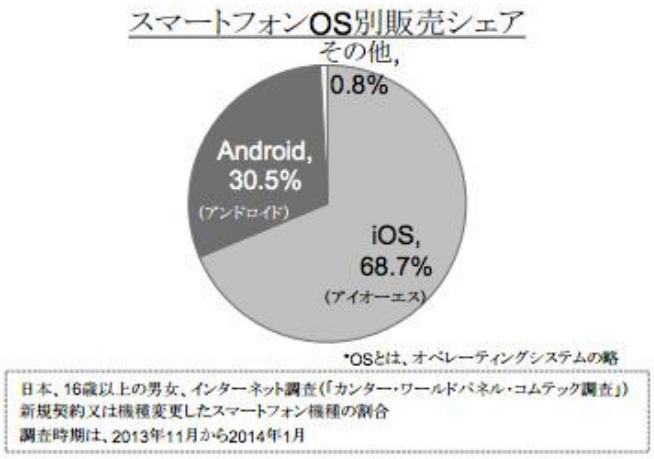

図 1　　日本でのスマートフォン普及率 [1]

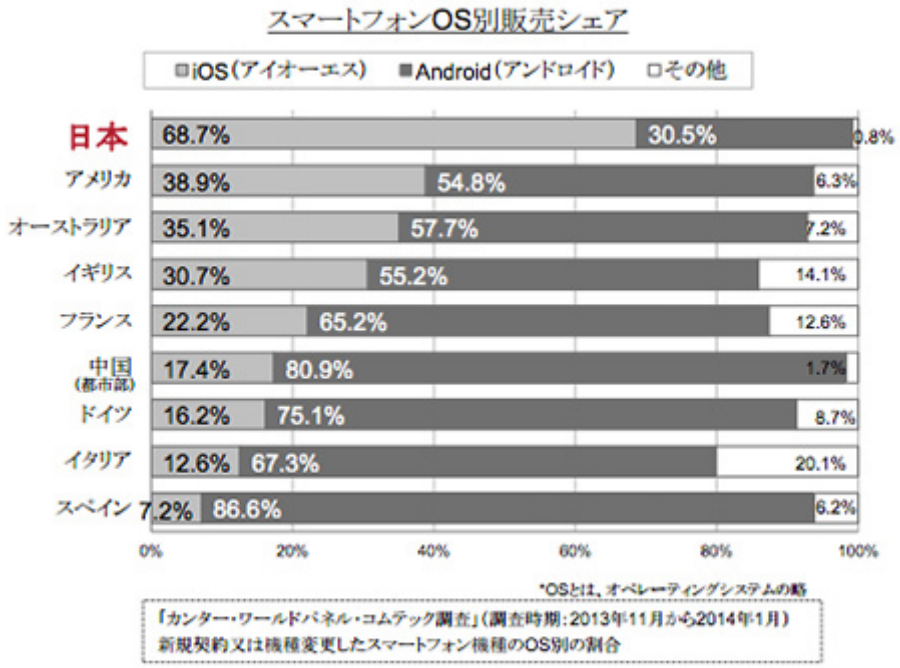

図 2　　世界でのスマートフォン普及率 [1]

第 2 章　Android アプリの企画

第 1 節　本アプリの開発経緯

普段より授業の時間割を手帳で管理をしていた。手帳で管理していると鞄の中から取り出す必要があり手間取ることが多くあった。そこでスマートフォンで時間割を管理することができればアプリを開くだけで即座に見ることができるのではないのかと思ったからだ。そういった経緯で時間割を管理するアプリを開発しようと考えた。また、このアプリを開発する上で対象とするユーザーは学生を主なターゲットとして開発を行った。下図が私の開発した「時間割アプリ」である。

図 3　　アプリの起動画面

第 2 節　開発環境

アプリを開発するにあたって Eclipse というオープンソースでの統合開発環境、Android アプリを開発するのに必要な Android　SDK を使用した。図 4 が Eclipse での開発画面になる。また、どちらもフリーソフトであるためアプリ開発は無料で行うことができる。

アプリをテストする場合は図 5 のような Eclipse に備わっている仮想デバイスやパソコンと Android を直接 USB ケーブルで接続し Android で動作の確認またはテストをする方法がある。

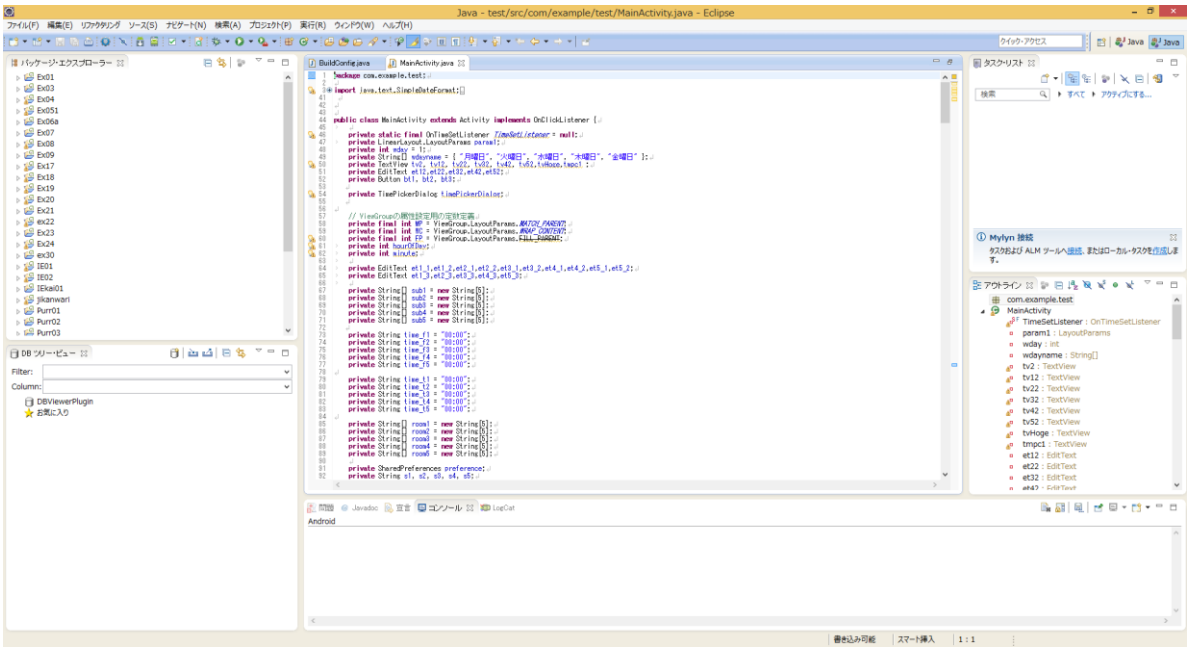

図 4　　Eclipse での開発画面

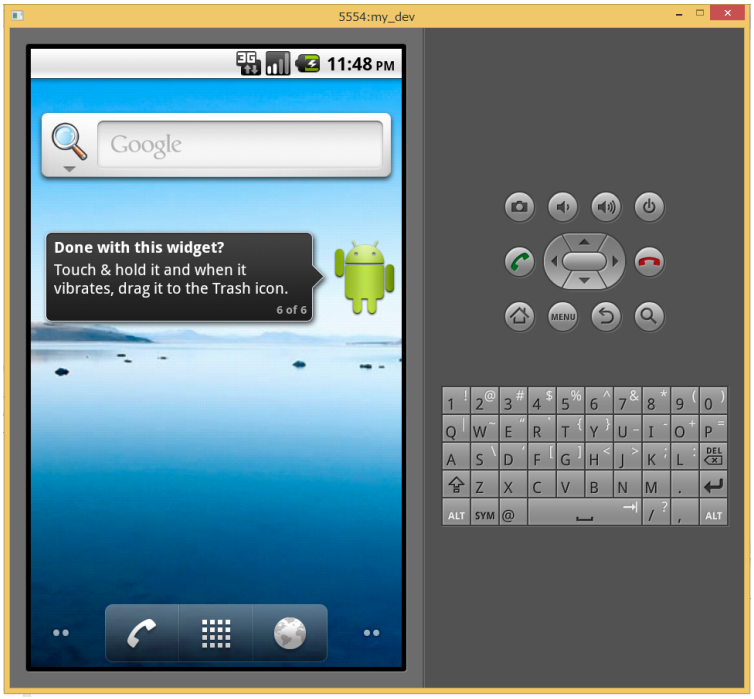

図 5　　仮想デバイス

第 3 節　アプリの企画

ここからは、アプリの企画について述べたいと思う。まず、「科目 1」と書かれているところをタップし自分が入れたい科目を入力する。例えば、月曜日の 1 限目に英語があったとすると、図 6 のように「科目 1」に英語を入力する。そして、同じようにあとは自分が入れたい科目を順々と入力していくことで､その日に何の科目があるのか把握できるようになる。

次に、「00：00」をタップする。これは、入力された科目の開始時間と終了時間を知るためである。例えば、開始時間が 9 時からで終了時間が 10 時 30 分であるならば、図 7 のように入力していく。同じように最後まで入力していくことで、その日の授業が何時から始まり何時に終わるのかを把握できるようになる。

次に、どの教室で授業を行うのかを知るために「教室」をタップする。そこには、どの教室で授業を行うのかを入力していく。例えば、図 7 のように教室が 711 であるのなら「教

室」をタップしそこに 711 と入力する。これで、どの教室で授業があるのか把握することができる。

図 6　　入力画面 1

図 7　　入力画面 2

次に「次の曜日へ」をタップすると図 8 のように月曜日から火曜日へと画面が変わり、次の曜日の時間割を入力できるようになる。また、一度時間を入力すると月曜日で入力した時間が火曜日から金曜日まで全て同じになるようにした。これで、入力する時間を短縮することができるだろう。

最後に「初期化」をタップすると入力されていた時間がもとに戻り、再度設定し直すことができる。これで、科目と教室を書き直すことで再度新たに使用することができる。


5556:my_dev
11:59 PM
test
時間割
火曜日
1 科目1 00:00 ~ 00:00 教室
2 科目2 00:00 ~ 00:00 教室
3 科目3 00:00 ~ 00:00 教室
4 科目4 00:00 ~ 00:00 教室
5 科目5 00:00 ~ 00:00 教室
前の曜日へ
次の曜日へ
初期化
MENU

図 8　　入力画面 3

第３章　アプリの設計

第１節　アプリの概要

第２章でも説明したが、この「時間割アプリ」は、学校や大学などの時間割をアプリで管理するために開発を行った。このアプリ開発の中で最も重要視したところは「シンプルでわかりやすい」という点である。誰にでも使えて、かつシンプルで見やすいを目標にアプリの開発を行った。また、下図は本アプリの処理の流れとなる。

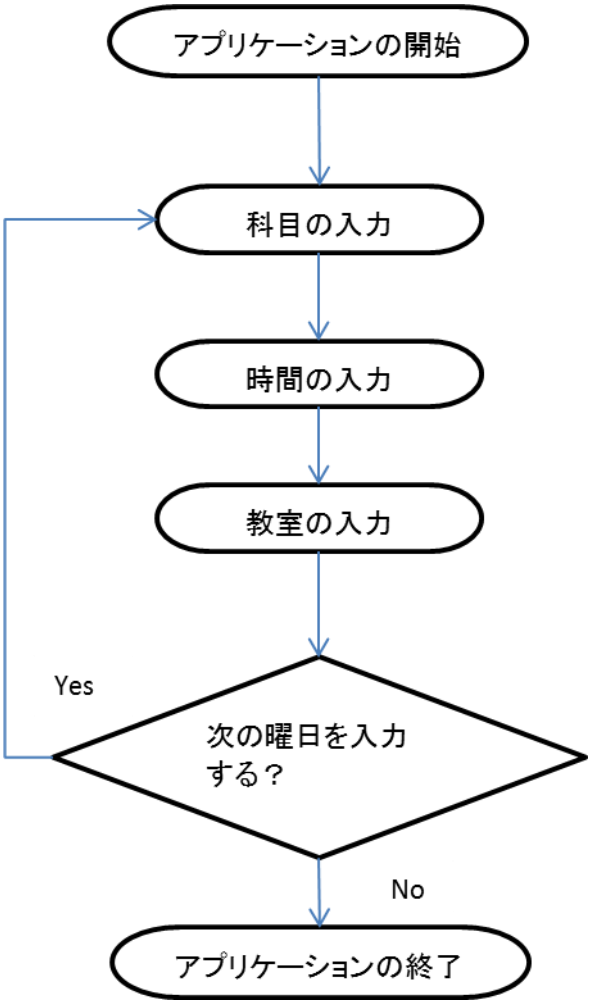

図９　アプリの処理の流れ

第２節　プログラムの説明

```
98      @Override
99      protected void onCreate(Bundle savedInstanceState) {
100         super.onCreate(savedInstanceState);
101         //setContentView(R.layout.activity_main);
102
103         // テキスト編集ボックスの幅は画面の余白に合わせる
104         param1 = new LinearLayout.LayoutParams(MP, WC);
105         param1.weight = 1.0f;
106
107         LinearLayout ll = new LinearLayout(this);
108         ll.setOrientation(LinearLayout.VERTICAL);
109         setContentView(ll);
110
111         LinearLayout l0 = new LinearLayout(this);
112         l0.setOrientation(LinearLayout.HORIZONTAL);
113         ll.addView(l0);
114
115         TextView tv0 = new TextView(this);
116         String ttl = "時間割 ";
117         tv0.setText(ttl);
118         l0.addView(tv0);
119
120         LinearLayout l2 = new LinearLayout(this);
121         l2.setOrientation(LinearLayout.HORIZONTAL);
122         ll.addView(l2);
123
124         tv2 = new TextView(this);
125         tv2.setText("月曜日");
126         l2.addView(tv2);
127
128         // 1限目
129         LinearLayout l11 = new LinearLayout(this);
130         l11.setOrientation(LinearLayout.HORIZONTAL);
131         ll.addView(l11);
132
133         TextView tv11 = new TextView(this);
134         tv11.setText("1　");
135         l11.addView(tv11);
136
137         et12 = new EditText(this);
138         et12.setText("科目1");
139         l11.addView(et12);
140
141         TextView tv_1_1 = new TextView(this);
142         tv_1_1.setText("　　");
143         l11.addView(tv_1_1);
144
145         et1_1 = new EditText(this);
146         et1_1.setText("00:00");
147         l11.addView(et1_1);
148
149         TextView tv_1_2 = new TextView(this);
150         tv_1_2.setText("～");
151         l11.addView(tv_1_2);
152
```

図 10　　ソースコード 1

```
153         et1_2 = new EditText(this);
154         et1_2.setText("00:00");
155         l11.addView(et1_2);
156
157         et1_3 = new EditText(this);
158         et1_3.setText("教室");
159         l11.addView(et1_3);
160
```

図 11　　ソースコード 1-2

ここからは、アプリのプログラムについて説明していきたいと思う。

98～100 行目にあるコードはアプリが起動時に最初に実行される処理となる。107～109 行目にあるコードは､アプリの画面全体に対応する LinearLayout を ll という名前で作成している。この LinearLayout は、パーツを縦一列もしくは横一列に並べる場合に使用するレイアウトである。つまり、ボタンやテキストボックスなどのコンポーネントを縦や横一列に並べるためのビューである。また、LinearLayout.VERTICAL が垂直方向、LinearLayout.HORIZONTAL が水平方向という機能を持つ。

次に、111～118 行目にあるコードは「時間割」というタイトルを表示するためのプログラムである。ここでは、テキストビューを生成することで「時間割」という文字列を表示した。テキストビューは TextView クラスを用いて表示することができる。また、120～126 行目も同じように「月曜日」というタイトルを表示するために TextView クラスを用いている。

次に、129 行目～159 行目までは 1 限目の科目、時間、教室を表示するためのプログラムである。ここでは、科目、時間、教室を表示するプログラムはすべて同様のプログラムで作成している。そこで、科目を例にして説明していきたいと思う。科目は、EditText クラスを用いた。このエディットテキストは文字列が編集可能なコンポーネントである。上図では、et12.setText("科目 1");とあるが、アプリ上ではこの科目 1 は編集可能であり英語や数学といったように文字列を後から変更することができる。

```
        // 幅と行数の設定
        et12.setWidth(101);
        et12.setMaxLines(1);
        et1_1.setWidth(100);
        et1_1.setMaxLines(1);
        et1_2.setWidth(100);
        et1_2.setMaxLines(1);
        et1_3.setWidth(90);
        et1_3.setMaxLines(1);

        et22.setWidth(101);
        et22.setMaxLines(1);
        et2_1.setWidth(100);
        et2_1.setMaxLines(1);
        et2_2.setWidth(100);
        et2_2.setMaxLines(1);
        et2_3.setWidth(90);
        et2_3.setMaxLines(1);

        et32.setWidth(101);
        et32.setMaxLines(1);
        et3_1.setWidth(100);
        et3_1.setMaxLines(1);
        et3_2.setWidth(100)
        et3_2.setMaxLines(1);
        et3_3.setWidth(90)
        et3_3.setMaxLines(1);

        et42.setWidth(101);
        et42.setMaxLines(1);
        et4_1.setWidth(100);
        et4_1.setMaxLines(1);
        et4_2.setWidth(100);
        et4_2.setMaxLines(1);
        et4_3.setWidth(90);
        et4_3.setMaxLines(1);
```

図 12　ソースコード 2

```
        et52.setWidth(101);
        et52.setMaxLines(1);
        et5_1.setWidth(100);
        et5_1.setMaxLines(1);
        et5_2.setWidth(100);
        et5_2.setMaxLines(1);
        et5_3.setWidth(90);
        et5_3.setMaxLines(1);
```

図 13　ソースコード 2-1

上図は、1～5 限目までの科目、時間、教室を入力するエディットテキストの幅と行数の設定である。まず、setWidth について説明する。setWidth はテキストボックスとして文字を入力できるエリアの幅を設定するメソッドである。また、単位はピクセルである。つまり、et12.setWidth(101)は 101 ピクセルの幅を設定するという処理となる。

次に、setMaxLines について説明すると、setMaxLines はテキストボックスで文字を入力する際の最大行の設定をする。したがって、et12. setMaxLines(1)は最大 1 行までということになる。

```
295         LinearLayout l16 = new LinearLayout(this);
296         l16.setOrientation(LinearLayout.HORIZONTAL);
297
298         bt1 = new Button(this);
299         bt1.setText("前の曜日へ");
300         bt1.setOnClickListener(this);
301
302         l16.addView(bt1,param1);
303
304         bt2 = new Button(this);
```

図 14　ソースコード 3

```
303
304         bt2 = new Button(this);
305         bt2.setText("次の曜日へ");
306         bt2.setOnClickListener(this);
307
308         l16.addView(bt2,param1);
309
310         l1.addView(l16);
```

図 15　ソースコード 3-2

295～302 行目はボタンを生成するプログラムであり、ボタンは Button クラスを用いる。また、Button.setOnClickListerer(this);はボタンクリックイベントの処理を表す。下図はボタンがクリックされた時の処理を表す。

```
419     @Override
420     public void onClick(View v) {
421         // TODO 自動生成されたメソッド・スタブ
422         if (v == bt1 || v == bt2) {
423
424             //
425             sub1[wday - 1] = et12.getText().toString();
426             sub2[wday - 1] = et22.getText().toString();
427             sub3[wday - 1] = et32.getText().toString();
428             sub4[wday - 1] = et42.getText().toString();
429             sub5[wday - 1] = et52.getText().toString();
430
431             time_f1 = et1_1.getText().toString();
432             time_f2 = et2_1.getText().toString();
433             time_f3 = et3_1.getText().toString();
434             time_f4 = et4_1.getText().toString();
435             time_f5 = et5_1.getText().toString();
436
437             time_t1 = et1_2.getText().toString();
438             time_t2 = et2_2.getText().toString();
439             time_t3 = et3_2.getText().toString();
440             time_t4 = et4_2.getText().toString();
441             time_t5 = et5_2.getText().toString();
442
443             room1[wday - 1] = et1_3.getText().toString();
444             room2[wday - 1] = et2_3.getText().toString();
445             room3[wday - 1] = et3_3.getText().toString();
446             room4[wday - 1] = et4_3.getText().toString();
447             room5[wday - 1] = et5_3.getText().toString();
448
```

図 16　ソースコード 4

if (v == bt1 || v == bt2)はこの一文は bt1 または bt2 がクリックされた時の処理である。この if 文で囲まれた範囲がボタンをクリックした時に行われる処理になる。425～429 行目は wday の数に対応した曜日で入力された科目名を配列の該当箇所にコピーする処理となっている。431～435 行目は入力された 1～5 限目までの開始時間をコピーする処理となっている。同様に、437～441 行目も入力された 1～5 限目までの終了時間をコピーする。443～447 行目は、wday の数に対応した曜日で入力された教室名を配列の該当箇所にコピーする処理となっている。

```
451             SharedPreferences.Editor editor = preference.edit();
452
453             editor.putString("sub1_" + wday, sub1[wday - 1]);
454             editor.putString("sub2_" + wday, sub2[wday - 1]);
455             editor.putString("sub3_" + wday, sub3[wday - 1]);
456             editor.putString("sub4_" + wday, sub4[wday - 1]);
457             editor.putString("sub5_" + wday, sub5[wday - 1]);
458
459             editor.putString("tf1", time_f1);
460             editor.putString("tf2", time_f2);
461             editor.putString("tf3", time_f3);
462             editor.putString("tf4", time_f4);
463             editor.putString("tf5", time_f5);
```

図 17　ソースコード 5

```
465             editor.putString("tt1", time_t1);
466             editor.putString("tt2", time_t2);
467             editor.putString("tt3", time_t3);
468             editor.putString("tt4", time_t4);
469             editor.putString("tt5", time_t5);
470
471             editor.putString("room1_" + wday, room1[wday - 1]);
472             editor.putString("room2_" + wday, room2[wday - 1]);
473             editor.putString("room3_" + wday, room3[wday - 1]);
474             editor.putString("room4_" + wday, room4[wday - 1]);
475             editor.putString("room5_" + wday, room5[wday - 1]);
476
477             editor.commit();
478
479
```

図 18　ソースコード 5-2

SharedPreferences は、アプリの設定情報を保存するための機能であり、データの値の読

み書きを行うことができる。また、同じアプリ内の別コンポーネントとデータ共有をする際にも利用することができる。データ値は、キーと値という 2 つの値の組み合わせで保存される。SharedPreferences の値を保存する手順について説明していく。 [2]

①getSharedPreferences メソッドを用いて SharedPreferences オブジェクトを取得する。

②SharedPreferences オブジェクトの edit()メソッドを用いて SharedPreferences .Editor オブジェクトを取得する。

③Editor クラスの putString()メソッドなどでキーと値を指定し保存する。

④Editor クラスの commit()メソッドで保存を完了する。

ここで、注意しなければならないのが Editor クラスの putString()メソッドを呼ぶだけでは保存とはならない。Editor クラスの commit()メソッドを呼び出すことで初めて保存されるので、注意をしなければならない。

```
480
481             if (v == bt1) {
482                 wday--;
483                 if (wday == 0) {
484                     wday = 5;
485                 }
486             } else if (v == bt2) {
487                 wday++;
488                 if (wday == 6) {
489                     wday = 1;
490                 }
491             }
492
```

図 19　ソースコード 6

上図は bt1 がクリックされた時に、wday を 1 つ前の日の値にセットする処理である。同様に bt2 がクリックされた時 wday を 1 つ前の日の値にセットする。

```
493             tv2.setText(wdayname[wday - 1]);
494
495             s1 = preference.getString("sub1_" + wday,"科目1");
496             s2 = preference.getString("sub2_" + wday,"科目2");
497             s3 = preference.getString("sub3_" + wday,"科目3");
498             s4 = preference.getString("sub4_" + wday,"科目4");
499             s5 = preference.getString("sub5_" + wday,"科目5");
500
501             tf1 = preference.getString("tf1","00:00");
502             tf2 = preference.getString("tf2","00:00");
503             tf3 = preference.getString("tf3","00:00");
504             tf4 = preference.getString("tf4","00:00");
505             tf5 = preference.getString("tf5","00:00");
506
507             tt1 = preference.getString("tt1","00:00");
508             tt2 = preference.getString("tt2","00:00");
509             tt3 = preference.getString("tt3","00:00");
510             tt4 = preference.getString("tt4","00:00");
511             tt5 = preference.getString("tt5","00:00");
512
513             r1 = preference.getString("room1_" + wday,"教室");
514             r2 = preference.getString("room2_" + wday,"教室");
515             r3 = preference.getString("room3_" + wday,"教室");
516             r4 = preference.getString("room4_" + wday,"教室");
517             r5 = preference.getString("room5_" + wday,"教室");
```

図 20　ソースコード 6-2

```
519             et12.setText(s1);
520             et22.setText(s2);
521             et32.setText(s3);
522             et42.setText(s4);
523             et52.setText(s5);
524
525             et1_1.setText(tf1);
526             et2_1.setText(tf2);
527             et3_1.setText(tf3);
528             et4_1.setText(tf4);
529             et5_1.setText(tf5);
530
531             et1_2.setText(tt1);
532             et2_2.setText(tt2);
533             et3_2.setText(tt3);
534             et4_2.setText(tt4);
535             et5_2.setText(tt5);
536
537             et1_3.setText(r1);
538             et2_3.setText(r2);
539             et3_3.setText(r3);
540             et4_3.setText(r4);
541             et5_3.setText(r5);
542
543
544         }
```

図 21　ソースコード 6-3

図 20、21 は図 19 で更新された wday の新たな値に対応した曜日の時間割データをセットする処理である。

```
545         if(v == bt3){
546
547             et1_1.setText("00:00");
548             et1_2.setText("00:00");
549             et2_1.setText("00:00");
550             et2_2.setText("00:00");
551             et3_1.setText("00:00");
552             et3_2.setText("00:00");
553             et4_1.setText("00:00");
554             et4_2.setText("00:00");
555             et5_1.setText("00:00");
556             et5_2.setText("00:00");
557         }
558     }
559 }
560
```

図 22　ソースコード 7

上図は bt3 がクリックされた時、アプリ上で設定されていた時間が初期設定に戻す処理である。

第４章　アプリの評価及び改善点

第１節　アプリの評価

ここで、アプリについての評価をしていきたいと思う。ゼミメンバーを対象にアプリのアンケートを実施した。（計 6 名。）アンケートを実施した目的は、実際にアプリを使ってみて操作性はどうなのか、改善するべき点があるのかといった客観的な評価、意見を得たいと思ったからである。アンケートの内容はそれぞれ「操作性」、「追加してほしい機能」、「アプリの感想」の 3 つとしアンケートを集計した。

まず、「操作性」に関しての評価は、図 23 にもあるように、「良かった」、「まあまあ良かった」と回答した人が 67%であり、「良くない」、「あまり良くない」という回答が 33%であった。良かったと回答した人は、操作しやすい、わかりやすいといった評価が多かった。良くないと回答した人の意見は、土曜日の授業も追加したほうが良い、自ら入力していくのではなく科目や時間を選べるようにしたほうが良いといった評価を受けた。

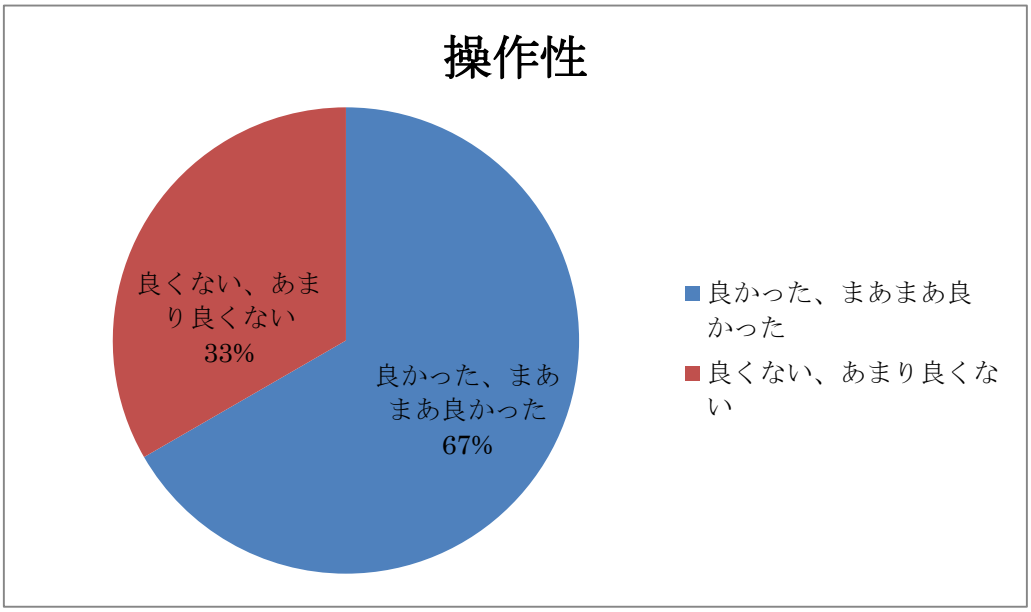

図 23　アンケート 1

「良かった」、「まあまあ良かった」を選んだ理由
● わかりやすくて良かった
● 時間入力の全体反映が便利であった
● 操作しやすくて良い

表 1　アンケート 1-2

「良くない」、「あまり良くない」を選んだ理由
● 自分で入力するより選べるようにしたほうが良い
● 土曜日の追加
● 開始時間の表記をするべき

表 2　アンケート 1-3

次に「追加してほしい機能」では、図 24 のような評価を得ることができた。入力した画面の一覧表示（メニュー画面）があったほうが良いという意見や時間を手動で入力するのではなくリストボックスを用いることで時間を選べるようにした方が良いのではないかといった回答を得ることができた。特に、入力した時間割を一覧表示にするは最も多かったため追加するべき項目の 1 つであると考える。また、リストボックスの追加についても同様に追加するべき項目であると考える。

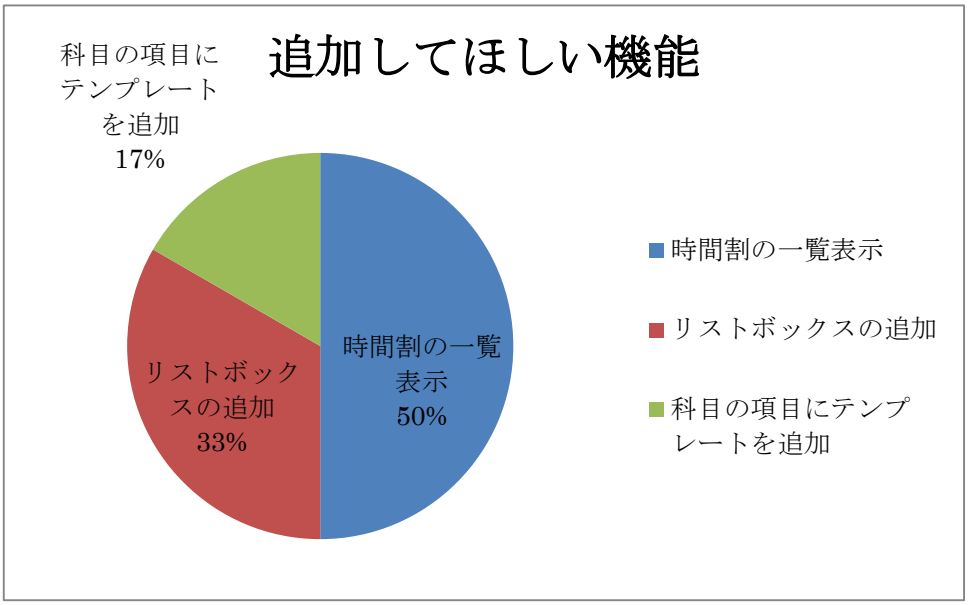

図 24　アンケート 2

最後に「アプリの感想」では、アプリの空白部分が目立つので１つ１つのボタンを大きくする、一覧表示の追加を検討したほうが良い、時間割の部分を増やしたり、減らしたりできたら良かったという回答が多かった。今回、アンケートを実施したことによって、「時間割アプリ」は全体的に改善するべき点が多くあることがわかった。これらの評価をもとにアプリを見直し、改善していくことが今後の課題といえるだろう。

アプリの感想
● 見やすいが操作性が悪く、使いにくい
● アプリの空白部分が目立つため、ボタンを大きくする、一覧表示の追加などを検討したほうが良い
● 開始時間、終了時間を時間入力の頭上につけわかりやすくする
● 学生用ということで使いやすい
● 時間割項目を追加または削除できるようにすると良い

表 3　アンケート 3

第 2 節　改善点

さきの評価を受けてアプリを改善する点が主に 2 つあることがわかった。1 つは、入力した時間割の一覧表示の追加である。確かに、画面を一覧で表示すればどの曜日に何があるのかすぐに見つけることができ、わかりやすく改善するべきだと考えた。2 つめは、時間を手動で入力するのではなく選べるようにすることである。私自身は自ら入力したほうが良いかと考えていたが、入力するよりも自分で選べた方が操作しやすいという意見が多かったため改善するべき項目の 1 つであるだろう。

以上の 2 つ以外にも今回のアンケートによって得られた結果は可能な限り反映できるようにしていきたいと考えている。また、アンケートを実施したことによって、新たに改善するべき点や自分がアプリを開発する上での目的の再確認をすることができた。また、自分では考えもしなかった発想を知ることができ、アンケートの重要性を理解することができた。

第５章　おわりに

アプリを開発する際に「シンプルでわかりやすい」を目標として開発してきた。第４章のアンケート結果にもあったようにシンプルで操作性は良いという意見は多かったが、改善するべき点がいくつか見受けられた。最初に立てた目標は達成できたといえるだろうが、アプリに関してはまだまだ改良するべき点が多くある。今後の課題はアンケートによって得られた結果をもとにしてアプリを見直していく必要があるだろう。特に、アンケートで要望が多かった、「記入した時間割を一覧で表示する」、「科目や時間を選べるようにする」は優先的に作っていきたいと考えている。

また、今回のアプリ開発で私が良いと思った機能でも他人から見れば、たいしたことはなかったり、むしろ使いにくいと思われたりすることがわかった。これは、アプリだけのことではないと思うが、やはり他人の意見や評価は大切なことなのだと知ることができた。

アプリ開発は初めてで難しいこと、苦しいことも多々あったが、大学生活の中で最も良い勉強になったと思う。アプリ開発で得た経験を活かし今後に繋げていきたい。

謝辞

本論文を進めるにあたって、指導教員の田中章司郎教授から、熱心なご指導を賜りました。ここに、感謝の意を表します。また、前任の伊藤則之教授にもご協力頂き感謝致します。

なお、本論文、本研究で作成したプログラム及び、データ、並びに関連する発表資料などの知的財産権を本研究の指導教員の田中章司郎教授に譲渡致します。

引用文献

[1] 掲載 HP（http://www.hivelocity.co.jp/blog/30247）

[2] 掲載 HP（http://techbooster.org/Android/application/11103/）