平成２７年度

卒業論文

題目	Android アプリケーション「Grouping You」の開発

担当教員 （自署）		印

学籍番号　201214059

氏　名　常光　笙

広島経済大学

広島経済大学

要旨

第 1 章では、なぜ“Grouping You”を開発しようと考えたのか、教育機関のデータも用いて私の考えを述べています。

第 2 章では、“Grouping You”をどのように使うのかを述べています。

第 3 章では、“Grouping You”の主要プログラムコードを書き出し解説しています。

第 4 章では、Google Play に投稿されている類似のグループ分けアプリをダウンロードし、使ってみて比較して意見を述べています。

第 5 章 5-1 ではアンケート結果を載せ、総評を述べています。5-2 では、今後の課題、作成しての感想を述べています。

目次

第1章　　はじめに

なぜ“Grouping You”を開発しようと考えたのか。それは、グループ分けやペア分けは手間がかかりすぎるからです。皆さんは、小・中・高校生の頃にクラスや部活でグループ分けをしたことはありますか。少人数ならともかく、大人数になればなるほど決めるまで時間がかかってきたはずです。また、毎回同じ組み合わせだとマンネリ化してしまうので変えようと思うとかなり悩んだはずです。

図 1 は平成 27 年度の初等教育機関等の学校数、在学者数及び教員数の表になります。小学生 6,543,104 人を 1 クラス 30 人とした場合、全体のほぼ半数である約 218,104 人の教員がクラス担任をしなければなりません。小学校といえば、理科や音楽は担当教員がいたりしますが、基本的に主要 5 科目の英語以外の教科及び副教科のほぼ全てを教えています。よって、授業の準備やテストの採点等を考えるとかなりの時間を必要とするので、グループ分けに時間をかけるわけにはいかないはずです。同じように、中学生 3,465,215 を 1 クラス 30 人とした場合も、ほぼ半数の約 115,508 人の教員が担任をしなければなりません。中学教員は担当する科目は 1 つでも 3 学年全てに教えなければならなかったりします。また、部活動の顧問等もあるので小学生教諭以上に時間が無いでしょう。

今回、私が作成したアプリケーション“Grouping You”の開発コンセプトは「グループ分けを「素早く」かつ「簡単」に行えるものにする」です。なので、番号でそれぞれのグループに振り分けを行います。

区　分	学校数（校）				在学者数（人）				教員数（人）		
	計	国立	公立	私立	計	国立	公立	私立	計	うち女性	女性の比率(%)
幼稚園	(-1,231)	(－)	(-393)	(-838)	(-155,013) (イ)	(-104)	(-26,527)	(-128,382)	(-9,562)	(-8,879)	(0.1)
	11,674	49	4,321	7,304	1,402,448 (ア)	5,510	238,036	1,158,902	101,497	94,769	93.4
幼保連携型認定こども園	1,943	－	374	1,569	281,136 (ウ)	－	43,928	237,208	37,461	35,337	94.3
小学校	(-251)	(－)	(-256)	(5)	(-56,902) (オ)	(-799)	(-55,642)	(-461)	(677)	(149)	(-0.1)
	20,601	72	20,302	227	6,543,104 (エ)	40,268	6,425,754	77,082	417,152	260,024	62.3
中学校	(-73)	(－)	(-70)	(-3)	(-39,119) (キ)	(-194)	(-36,515)	(-2,410)	(-128)	(394)	(0.2)
	10,484	73	9,637	774	3,465,215 (カ)	31,026	3,190,799	243,390	253,704	108,542	42.8
高等学校	(-24)	(－)	(-24)	(－)	(-14,905) (ケ)	(10)	(-18,223)	(3,308)	(-336)	(761)	(0.3)
	4,939	15	3,604	1,320	3,319,114 (ク)	8,623	2,268,162	1,042,329	234,970	73,591	31.3
中等教育学校	(1)	(－)	(1)	(－)	(818)	(-18)	(1,042)	(-206)	(77)	(57)	(1.2)
	52	4	31	17	32,317	3,142	21,466	7,709	2,509	854	34.0
特別支援学校	(18)	(－)	(19)	(-1)	(2,277) (サ)	(-14)	(2,311)	(-20)	(1,625)	(1,208)	(0.3)
	1,114	45	1,056	13	137,894 (コ)	3,019	134,092	783	80,905	49,274	60.9
専修学校	(-5)	(-1)	(-2)	(-2)	(-3,346)	(-39)	(-292)	(-3,015)	(143)	(120)	(0.1)
	3,201	9	193	2,999	656,106	411	25,963	629,732	40,917	21,496	52.5
うち高等課程を置く学校	(-5)	(－)	(－)	(-5)	(38)	(-4)	(-18)	(60)	(-2)	(5)	(0.2)
	432	1	7	424	40,095	19	519	39,557	2,749	1,534	55.8
うち専門課程を置く学校	(9)	(-1)	(-2)	(12)	(-705) (シ)	(-34)	(-275)	(-396)	(345)	(154)	(-0.1)
	2,823	9	190	2,624	588,183 (タ)	301	25,422	562,460	37,063	19,709	53.2
各種学校	(-47)	(－)	(-2)	(-45)	(-4,119)	(－)	(-53)	(-4,066)	(-204)	(-34)	(0.5)
	1,229	－	6	1,223	117,727	－	585	117,142	8,619	3,536	41.0

図 1　平成 27 年度初等教育機関等の学校数、在学者数及び教員数[1]

第2 章　“Grouping You”の企画

ここでは、実際にアプリを起動させてどのように使用するか説明していきたいと思います。まず図 2 が“Grouping You”の処理手順を図に表したものとなります。アプリを機動したら、まずは人数の設定をします。次にグループ数の設定をします。そして、「シャッフルを押す」で分岐になっていますが、人数とグループ数が設定したものでよければシャッフルを押す、全て戻したい時はリセットを押す、というふうに分かれます。

シャッフルを押した場合、初めにグループ 1 の結果が表示されます。ここで次の分岐となり、「次のグループ結果を表示」するかどうか決めます。表示する場合は「→」を押せばグループ 2 の画面になります。最初の画面に戻る時は、「戻る」を押せば戻ります。

グループ 2 を表示したらここでも分岐が発生します。「他のグループの結果を表示」するかどうか決めます。表示する場合、次のグループを表示するなら「→」を、前のグループを表示する時は「←」を押します。

表示しない時は、「アプリを終了する」かどうかを決めます。終了しない場合は「戻る」を押せば最初の画面に戻ります。終了する場合はアプリを終了するようになります。

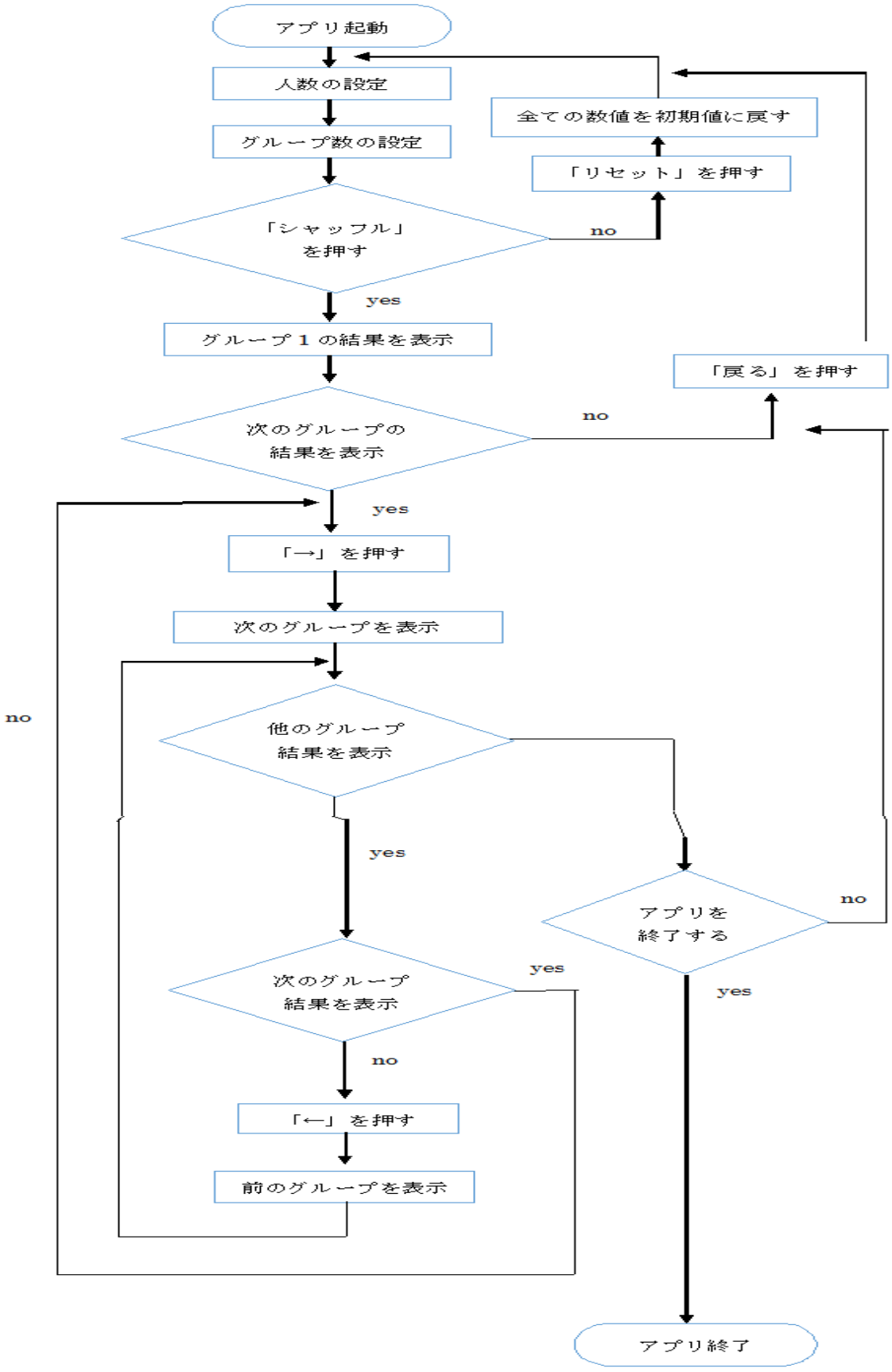

図 2“Grouping You”の処理手順

図 3 がアプリ起動後のメイン画面となります。　“ Number Picker ”がボタンの上にあるものとなります。左側が人数、右側がグループの設定になります。次は画面中央にあるボタンの説明となります。左側の「シャッフル」は設定した人数とグループ数で振り分けするものとなります。右側の「リセット」は人数とグループ数を初期値に戻すものとなります。

“ Number Picker ”はスクロールする、もしくはタップして打ち込む事によって数値を変更することができます。なお、タップして数値を打ち込む場合、端末によっては Enter を押さなければ打ち込んだ数値が決定しない場合があります。

図 3　アプリ起動画面

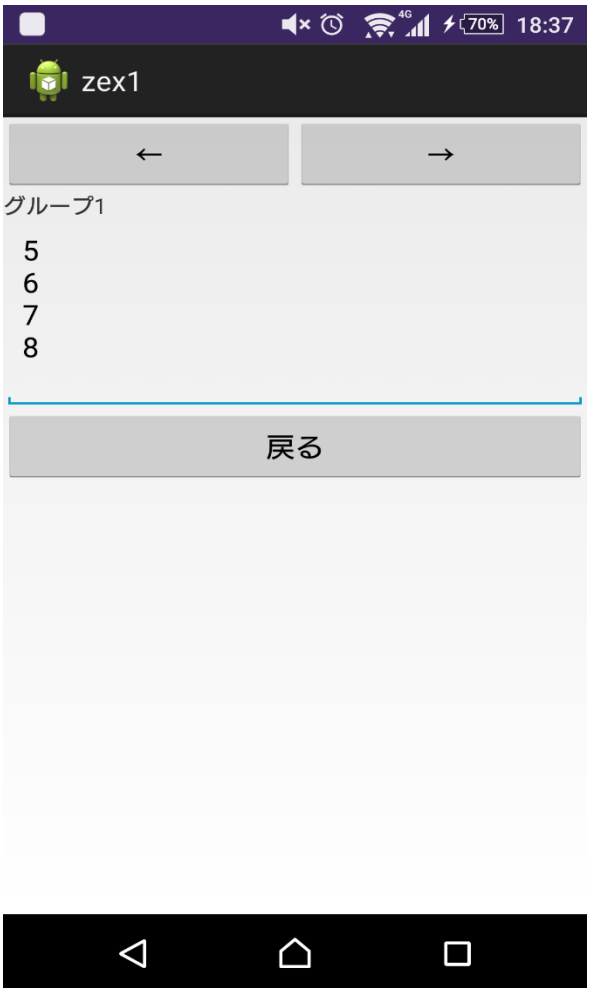

図 4　シャッフル後画面

次に、図 4 が図 3 の「シャッフル」を押した後の画面となります。一番上の矢印は、最初に設定したグループ数だけ画面を移動するものとなります。ただし、移動できるのは第 2 章で解説したように振り分けする計算式は人数÷グループ数となり、この計算の余りも同じ計算式で再振り分けします。よって、11人を 3 グループに分けるときは、 1,2 グループが 4 人、 3 グループが 3 人となるように振り分けられます。「戻る」のボタンを押すと、最初の画面に戻ります。

第 3 章　アプリの主要プログラム設計

今回作成した “Grouping You” のコードを書き出し、どのような動きをしているのかを説明します。開発環境は Eclipse 及び Android SDK を使用しました。図 5 は Eclipse の開発画面となります。今回、エラーが起こっているのですが、その場合、画面中央左の行番号の左横に×印が付き、該当箇所には赤の波線がひかれるので、わかりやすくなっています。なお、エラーが起きている時は図 6 のようにデバックをしようとしても出来ないようになっています。

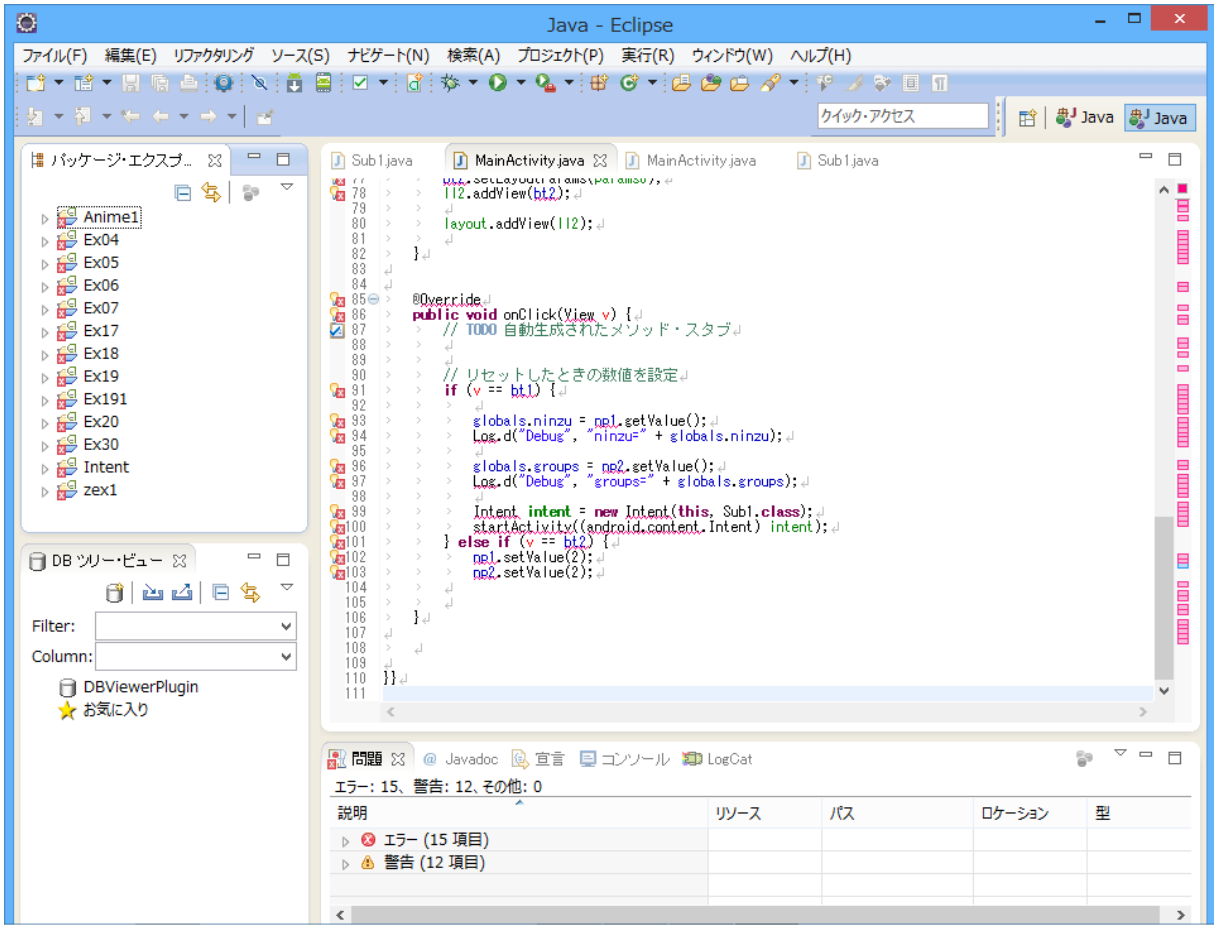

図 5　開発中の Eclipse 画面 1

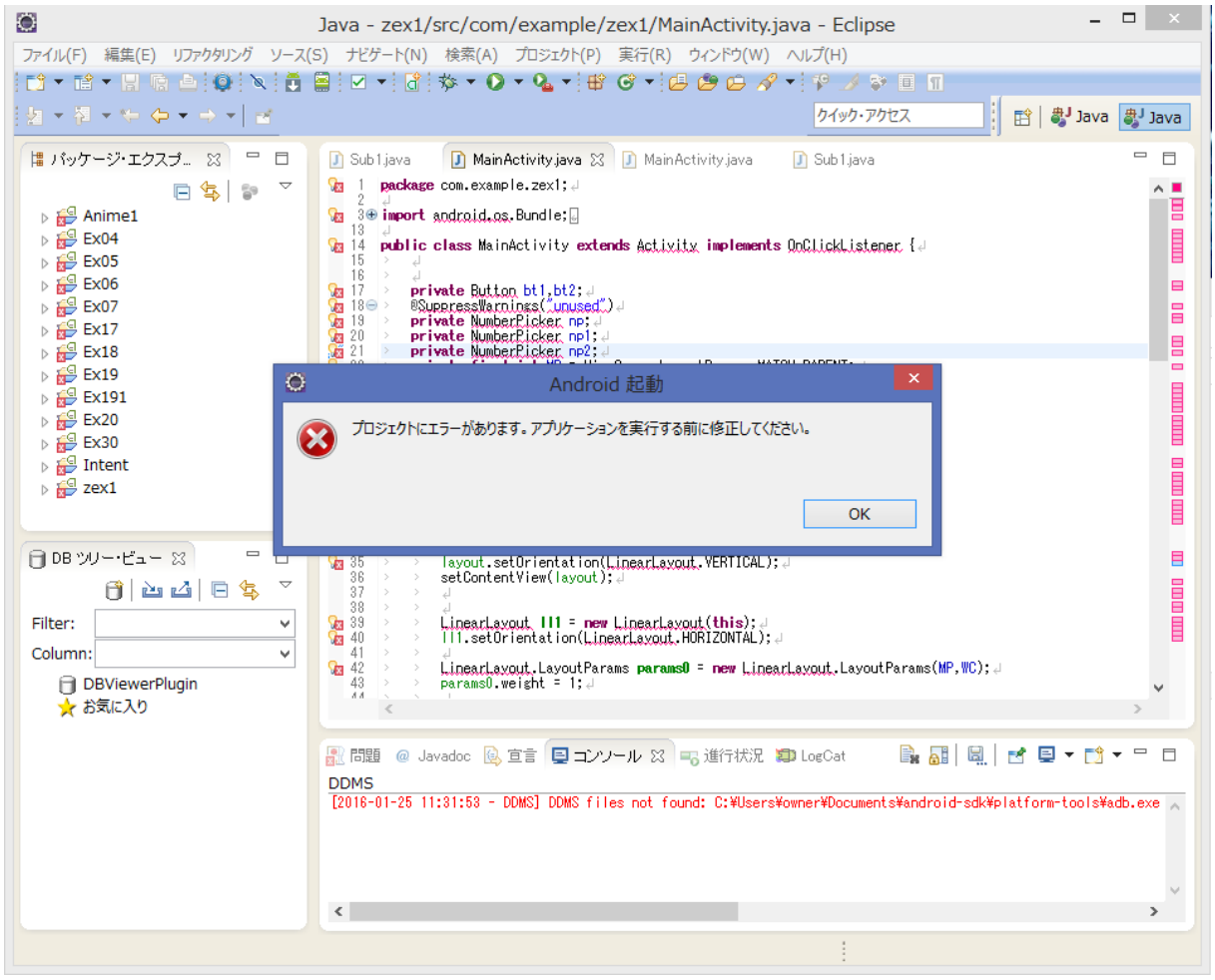

図 6　開発中の Eclipse 画面 2

```
1.LinearLayout layout = new LinearLayout(this);
2.layout.setOrientation(LinearLayout.VERTICAL);
3.setContentView(layout);
4.
5.
6.LinearLayout lll = new LinearLayout(this);
7.lll.setOrientation(LinearLayout.HORIZONTAL);
8.
9.LinearLayout.LayoutParams params0 = new LinearLayout.LayoutParams(MP,WC);
10.params0.weight = 1;
11.
12.np1 = new NumberPicker(this);
```

図　7 プログラムコード 1

```
1.LinearLayout ll2 = new LinearLayout(this);
2.ll2.setOrientation(LinearLayout.HORIZONTAL);
3.
4.bt1 = new Button(this);
5.bt1.setText("シャッフル");
6.bt1.setOnClickListener(this);
7.bt1.setLayoutParams(params0);
8.ll2.addView(bt1);
9.
10.bt2 = new Button(this);
11.bt2.setText("リセット");
12.bt2.setOnClickListener(this);
13.bt2.setLayoutParams(params0);
14.ll2.addView(bt2);
15.
16.layout.addView(ll2);
```

図 8　プログラムコード 2

```
1.np1 = new NumberPicker(this);
2.np1.setLayoutParams(params0);
3.np1.setMaxValue(50);
4.np1.setMinValue(2);
5.np1.setValue(INITIAL_VALUE);
6.ll1.addView(np1);
7.
8.np2 = new NumberPicker(this);
9.np2.setLayoutParams(params0);
10.np2.setMaxValue(10);
11.np2.setMinValue(2);
12.np2.setValue(INITIAL_VALUE);
13.ll1.addView(np2);
```

図 9　プログラムコード 3

```
1. if (v == bt1) {
2.
3.    globals.ninzu = np1.getValue();
4.    Log.d("Debug", "ninzu=" + globals.ninzu);
5.
6.    globals.groups = np2.getValue();
7.    Log.d("Debug", "groups=" + globals.groups);
8.
9.    Intent intent = new Intent(this, Sub1.class);
10.   startActivity((android.content.Intent) intent);
11.} else if (v == bt2) {
12.   np1.setValue(2);
13.   np2.setValue(2);
```

図　10　プログラムコード 4

図 7 のコードはアプリの入れ物を作成するコードとなります。 1 行目～ 3 行目までが縦方向に入れ物を作り、 6 行目～ 7 行目そして図 8 の 1 行目～ 2 行目が横方向に入れ物を作るコードになります。図 7 の 2 行目と 7 行目、図 8 の 2 行目にある“VERTICAL”、“HORIZONTAL” が縦、横を決めています。このままではボタン等も全て左寄りで表示されてしまうので、図 7 の 9 行目～ 10 行目のコードで均等になるようにしています。これは、結果表示画面でも同じものを使用しています。

ボタンを作成するコードは図 8 のものとなります。 5 行目及び11行目の ”bt○ .setText(” ・・・ ”)” でボタンにテキストとして「・・・」の部分に「シャッフル」あるいは「リセット」を入れています。

このアプリで重要になるのが数を選択する“ Number Picker ”になります。実装するためのコードが図 7 の 12 行目、そして図 9 が“ Number Picker ”の主要コードになります。実装は最初に”private NumberPicker np;”と定義してからになりますが、このアプリでは人数とグループ数で別々のものが必要になるので“ np1 ”“ np2 ”というものも作っています。また最大値と最小値はそれぞれ 3~4 行目と 10~11 行目の max 及び min で設定しています。今回は人数の最大値を 50 、最小値を 2 、そしてグループ数の最大値を 10 、最小値を 2 としています。

“ Number Picker ”の数値をリセットするコードが図 10 の11行目より下のコードになります。ここでは人数、グループ数ともに 2 となるようにしています。

次に選択した数値を確定させるコードが図 10 の 3 行目～ 7 行目となります。“ globals ”は全画面共通のものとなり、振り分けするためのコードを別のページで作っているので、“ globals …”と定義して、図11の 1 ～ 2 行目のように別ページで“… = globals …”と定義しなければページが違うので数値が設定されていないというエラーがおこります。

```
1.ninzu = globals.ninzu;
2.groups = globals.groups;
3.total = ninzu;
4.person =new int[total];
5.
6.for (int i = 0; i < total; i++) {
7.    person[i] = -1;
8.}
9.
10.rnd = new Random();
11.
12.// グループ番号の割り当て
13.for (int i = 1; i <= groups; i++) {
14.    for (int j = 0; j < ninzu/groups; j++) {
15.        while (true) {
16.            r = rnd.nextInt(total);
17.            if (person[r] == -1) {
18.                person[r] = i;
19.                break;
```

図11　プログラムコード5

```
1.for (int i = 1; i <= ninzu - (ninzu/groups)*groups; i++) {
2.    for (int j = 0; j < 1; j++) {
3.        while (true) {
4.            r = rnd.nextInt(total);
5.            if (person[r] == -1) {
6.                person[r] = i;
7.                break;
```

図　12　プログラムコード6

```
1.if ( v == bt1){
2.    Intent intent = new Intent(this, MainActivity.class);
3.    intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
4.    startActivity(intent);
```

図　13　プログラムコード7

先ほど確定させた数値でグループ分けをしていくプログラムが図11になります。 13 行目より下のコードで一通りグループ分けを行います。計算式は人数÷グループ数です。このままでは、余った分が全てグループ 1 に振り分けられてしまうので、図 12 のコードで余り分を再振り分けしていきます。

このアプリは起動した時の画面（以後メイン画面）と振り分けた後の画面（以後サブ画面）の 2 つを使います。これらを移動するためのコードが図 10 の 9 行目～ 10 行目及び図 13 になります。今回はスタック方式を採用しています。これによって、メイン画面で「シャッフル」を押せばサブ画面に、サブ画面で「戻る」を押せばメイン画面に行くようになっています。

ここで、先ほど述べたスタック方式について説明します。図 14 のようにA→B→Cの

順番でデータを入力した場合、最後に入力したものが先に出力される構造の事を言います[2]。例としてパソコンが挙げられます。パソコンを使用するとき、複数の画面を開いたら最後に開いたものが一番上に来ていると思います。これがスタック方式と同じものになります。

ちなみにスタック方式とは逆に、入力した順番で出力していく構造をキュー方式と言います。図 15 及び図 16 のように病院等の予約受付はこのキュー方式を使用しているため、予約した順番に呼び出しされます。

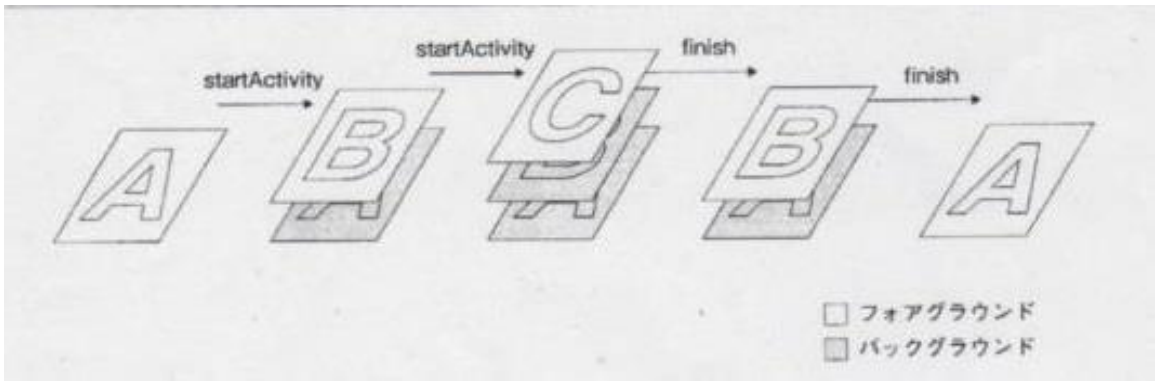

図 14　スタック方式解説図

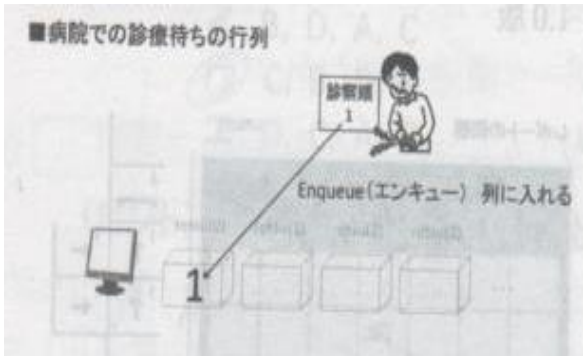

図 15　キュー方式解説図 1

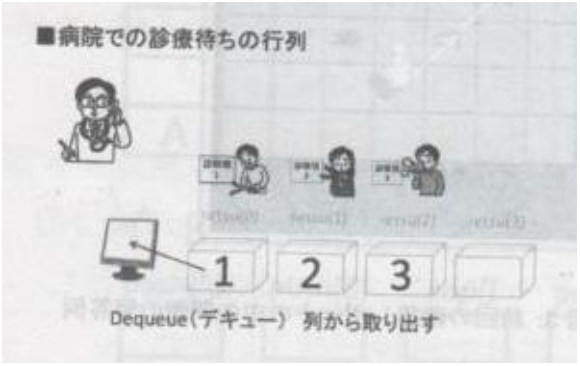

図 16　キュー方式解説図 2

```
1.if (v == bt_l) { // ←
2.   if ( now > 1) {
3.       now--;
4.       tv1.setText("グループ" + now);
5.       String kekka = "";
6.       for (int i = 1; i <= total; i++) {
7.           if (person[i-1] == now) {
8.               kekka = kekka + i + "¥n";
9.           }
10.      }
11.      et1.setText(kekka);
```

図 17　プログラムコード 8

```
1.if (v == bt_r) { // →
2.   if ( now < groups) {
3.       now++;
4.       tv1.setText("グループ" + now);
5.       String kekka = "";
6.       for (int i = 1; i <= total; i++) {
7.           if (person[i-1] == now) {
8.               kekka = kekka + i + "¥n";
9.           }
10.      }
11.      et1.setText(kekka);
```

図 18　プログラムコード 9

サブ画面はグループ数に応じてページが作成されます。それを移動させるコードが図 17 及び図 18 となります。図 17 のコードが左矢印で表示グループを一つ戻し、図 18 のコードが右矢印で表示グループを一つ進めるものになります。”now” が今見ているグループのことを指すため、図 17 、2 行目の“ now > 1 ”によりグループ 1 より前の画面には行かず、図 18 、2 行目の“ now < groups ”によって設定したグループ数より先の画面には進まないようになっています。また、図 17 及び図 18 、4 行目 ”tv1.set Text(” グループ” + now)”というコードによって、今どのグループ結果を見ているのかを表示しています。

```
1.now = 1;
2.tv1 = new TextView(this);
3.tv1.setText("グループ1");
4.layout.addView(tv1);
5.
6.String kekka = "";
7.for (int i = 1; i <= total; i++) {
8.    if (person[i-1] == 1) {
9.        kekka = kekka + i + "¥n";
10.   }
11.}
12.
13.et1 = new EditText(this);
14.et1.setText(kekka);
15.layout.addView(et1);
```

図 19　プログラムコード 10

図 19 は振り分け結果を表示するコードとなります。1 行目～4 行目の ”now = 1” から ”layout.addView(tv1)” は現在見ているのがどのグループか表示するものを作っています。6 行目～9 行目が振り分け結果を確定させるためのコードとなります。 13 行目～ 15 行目の ”et1 = new Edit Text(this)” から ”layout.addView(et1)” までが確定させた振り分け結果をEdit Textに表示させるコードとなります。

第 4 章　他のグループ決めアプリとの比較

ここでは、 Google Play にアップロードされている他の類似アプリケーションとの比較をしていこうと思います。そのアプリケーションを実際にダウンロードして使用した比較です。

Google Play で「グループ分け」と検索した結果、出てきたのは Keeping,cir. さんの“グループ分け”というアプリだけでした。類似アプリとしては、テニスのペア分けや電話帳のグループ分けはありましたが、“Grouping You”のように大人数を分けるアプリはほとんどありませんでした。

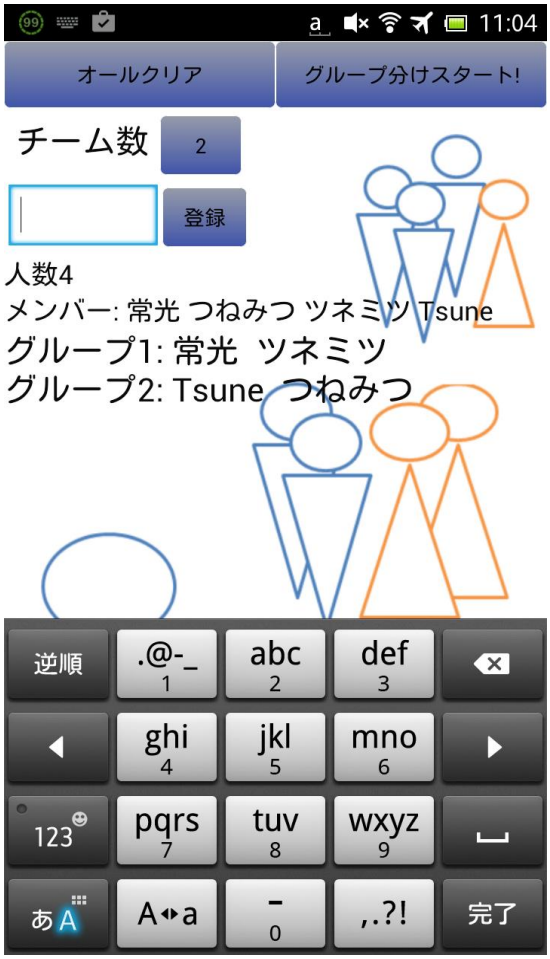

図 20　グループ分け使用画面

図 21　TeamMaker 使用画面 1

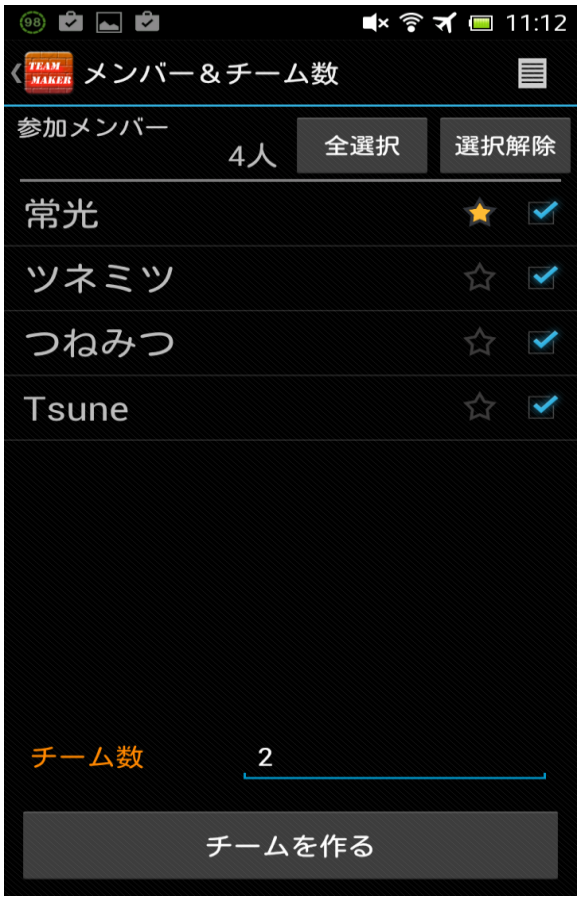

図 22　TeamMaker 使用画面 2

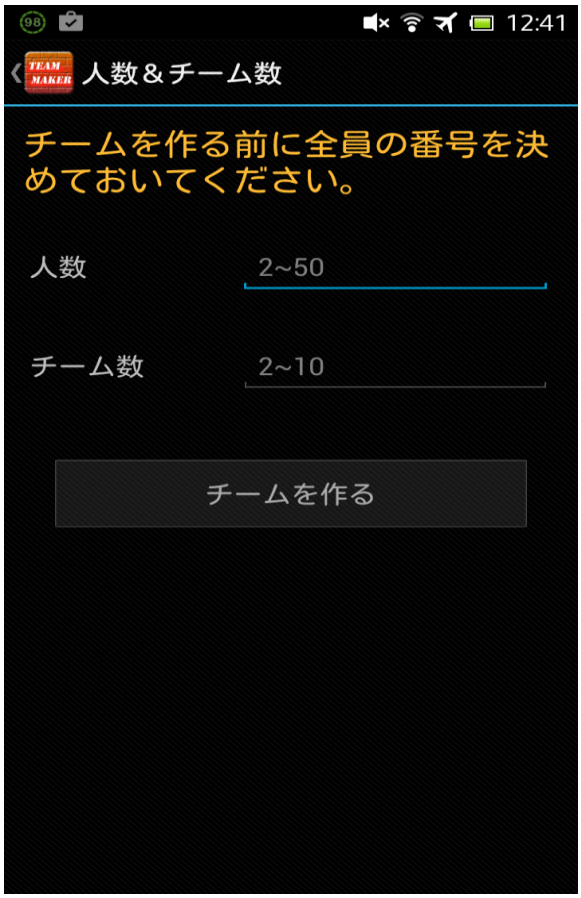

図 23　TeamMaker 使用画面 3

まずは、 Keeping,cir. さんの“グループ分け”というアプリと比較していきます[3] 。使用した画像が図 20 となります。異なる点としては、名前を打ち込む事によって正確に誰がどのグループかがわかります。しかし、名前を打ち込む時間が必要となるので、速度の面から見れば少し時間がかかると感じました。

次に HAL Take さんの“TeamMaker –簡単チーム分け - ”と比較していきます[4] 。こちらを使用した画像が図 21 ～図 25 となります。名前の横にある星マークは、グループのリーダーになる人のマークになっています。その横のチェックマークは参加メンバーを表すものとなっています。グループリーダーはグループ数より多く設定していると、「リーダーの人数は、チーム数以下にしてください 。」 という表示が出てきて振り分けが行えません。同じようにグループ数を人数より多く設定すると「チーム数は、人数以下にしてください 。」 という表示が出てきて振り分けをすることができませんでした。

名前を打ち込むという点では“グループ分け”と同じですが、メンバーリストを作成しておけるので事前にリストを作成しておけばかなり便利だと思います。またリストを使用しない分け方もあるようで、これは“Grouping You”とほとんど同じものでした。しかし、振り分け後のグループ内人数はランダム決定になっていました。

また、設定次第では１人ずつ結果を表示してから全体の結果を表示させることができるので、自分がどのグループになるのかドキドキ感が味わえるのでいいかもしれません。

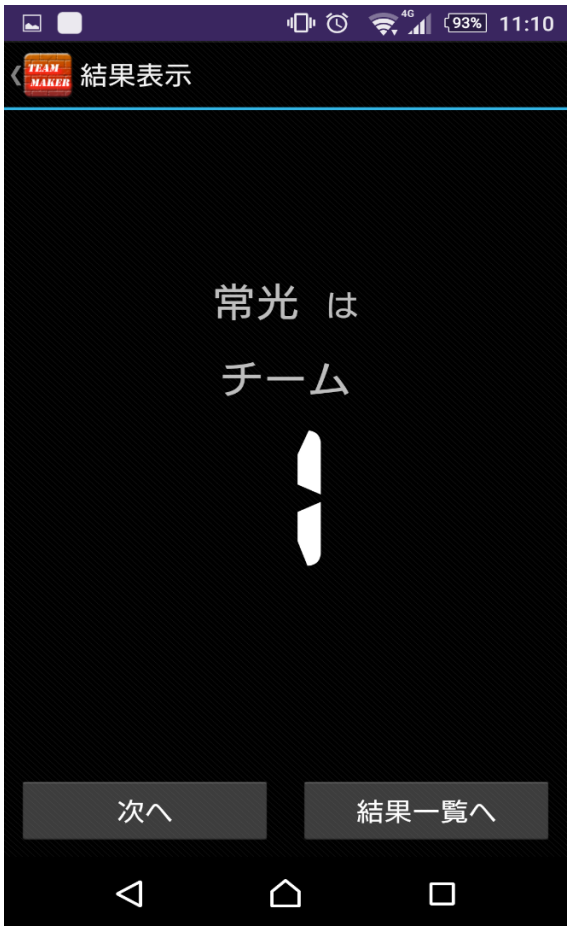

図 24　TeamMaker 使用画面 4

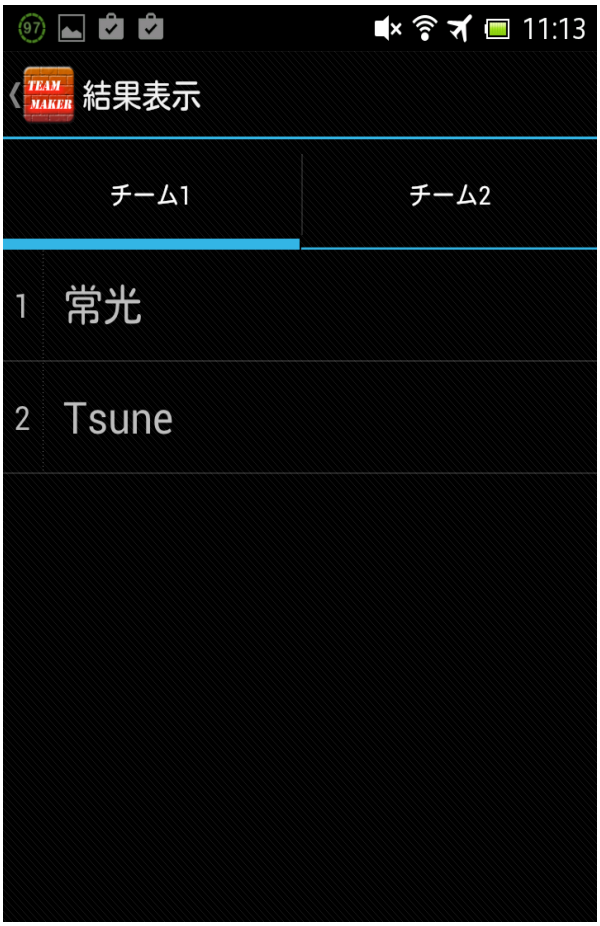

図　25　TeamMaker 使用画面 5

従って“Grouping You”は“グループ分け”と比べ操作速度の面で優れていると考えられますが、グループ分けの正確さが少し劣っていることが分かりました。

一方、“ TeamMaker ”は“グループ分け”と“Grouping You”の特徴を合わせ持ち、

今回の“Grouping You”の目標となるアプリであるかと思います。

第 5 章　考察および今後の課題

5-1　アンケートによる評価

今回、作成したアプリを周りの友人に実際に使用してもらいアンケートを実施しました。評価項目は1.デザイン　2. 分ける速さ　3. 使いやすさ　4. 独創性です。図 26 は全てのアンケート結果を集計し、各項目を 5 段階評価に直して平均したものになります。また図 27 ～図 31 は年代別に集計し、先ほどと同じく各項目を 5 段階評価に直して平均したものとなります。主にアルバイト先の人たちに使ってもらったので、 10 代及び 20 代がほとんどとなっています。

全体の結果としては2.分ける速さが他と比べて少し高い評価でしたがどの項目も高い評価をもらいました。年代別の結果として2.分ける速さと4.独創性はどの年代も高い評価でした。1.デザインと3.使いやすさは年代によって低くなっています。

集まった意見として、1.使い方がわかれば良いが人数とグループ数がわかりにくい　2.名前も入力出来たらよい等がありました。 1 はアンケート実施の時、使い方を説明しながらやってもらいましたが、その時はわかるから、どっちが何かをわかるようにしたら良いとのことでした。 2 は数字だと早いが名前を入力できるようになればもっとわかりやすいので加えてほしいということでした。また、小学校教員を目指している人に聞いてみたところ、小学校では 5 ～ 6 人の班が既に決まっているのであったら便利だが使う場面は少ないのではないかということでした。

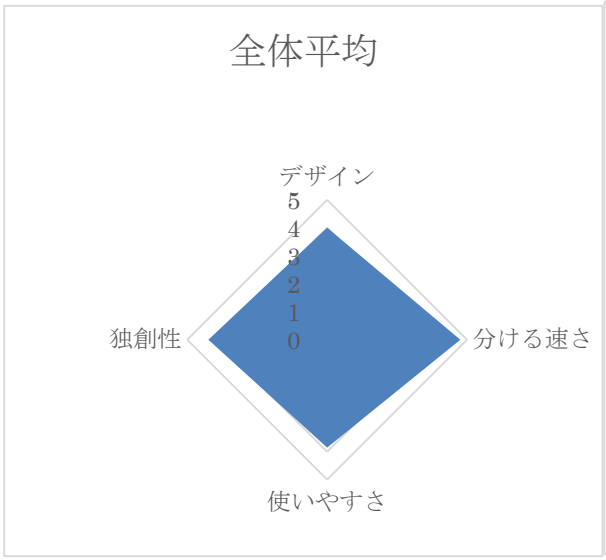

図 26　アンケート全体平均結果

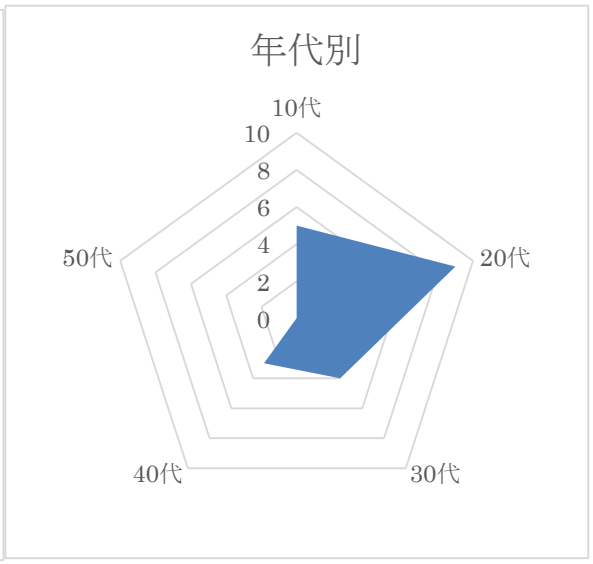

図 27　年代別アンケート集計結果

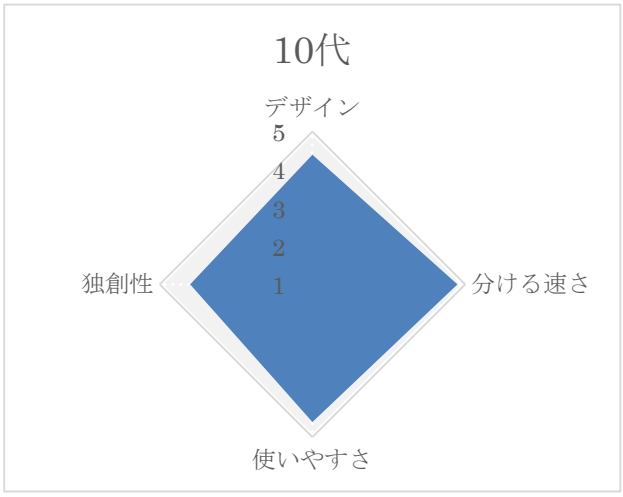

図 28　10 代アンケート結果

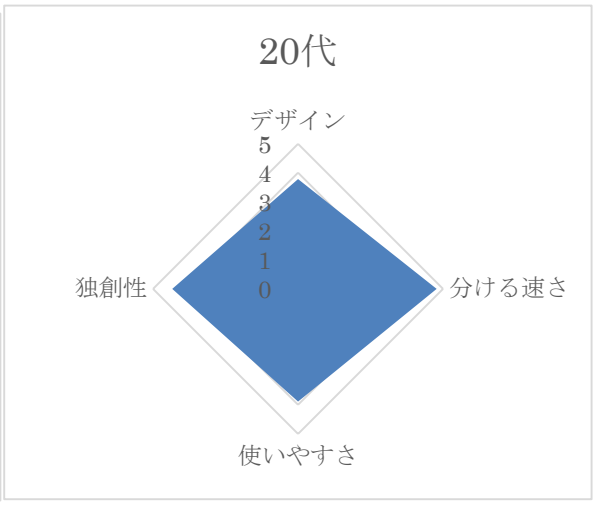

図 29　20 代アンケート結果

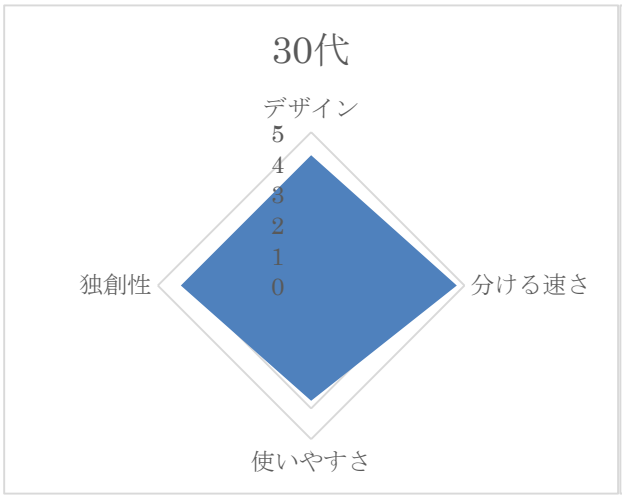

図 30　30 代アンケート結果

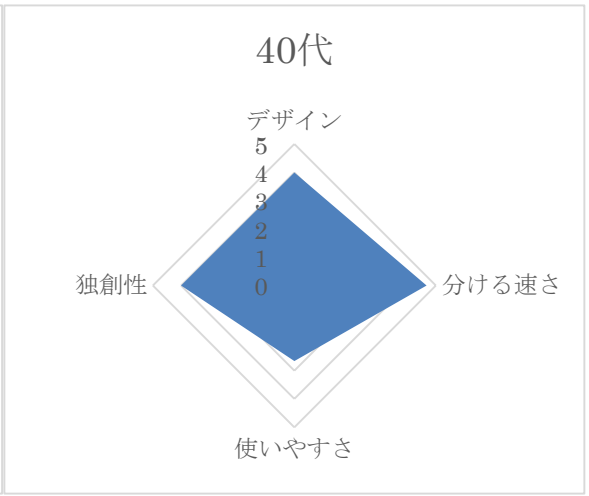

図 31　40 代アンケート結果

5-2　課題及び感想

まず作成後の改良点を挙げるとしたら、簡単にグループ分けできる分保存機能等を付けませんでした。誤って「戻る」を押したらリセットしてしまうので、今後改良していくならまずはこの機能を付けたいと思います。またアンケートでも使い方がわかれば使いやすいが、聞かないとボタンの配置がわかりにくいという意見が多かったです。したがってラベル等をつけてわかりやすくしたいと思います。

もしも 2 人を 3 グループに分けようとしたとき、本来であれば図 32 のようにエラー表示を出すべきなのですが、“Grouping You”ではエラー表示が出ずに振り分けてしまいます。よって、図 33 のようにメンバーがいないグループ 3 の画面が表示されます。こういったエラー表示も出るように改良していきます。

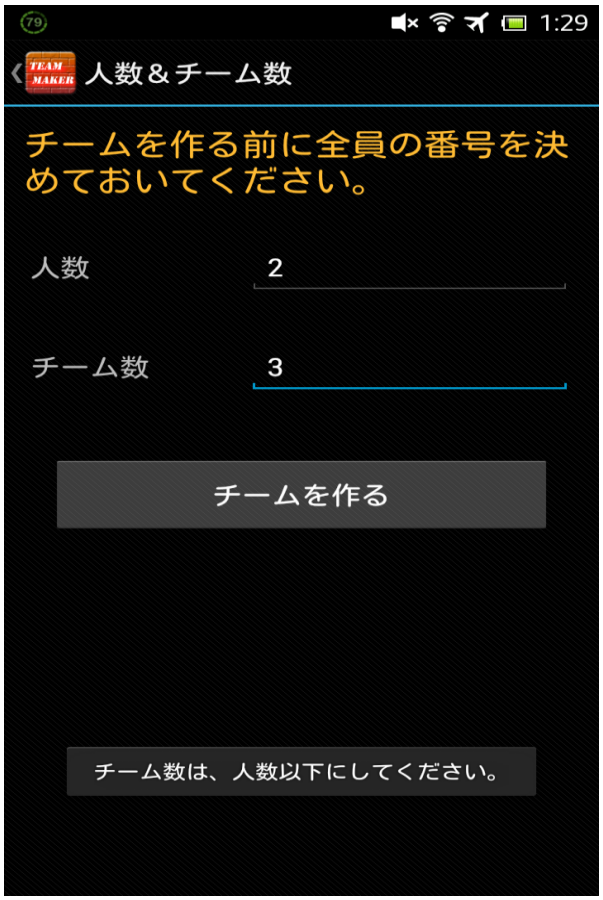

図 32　エラー画面 1

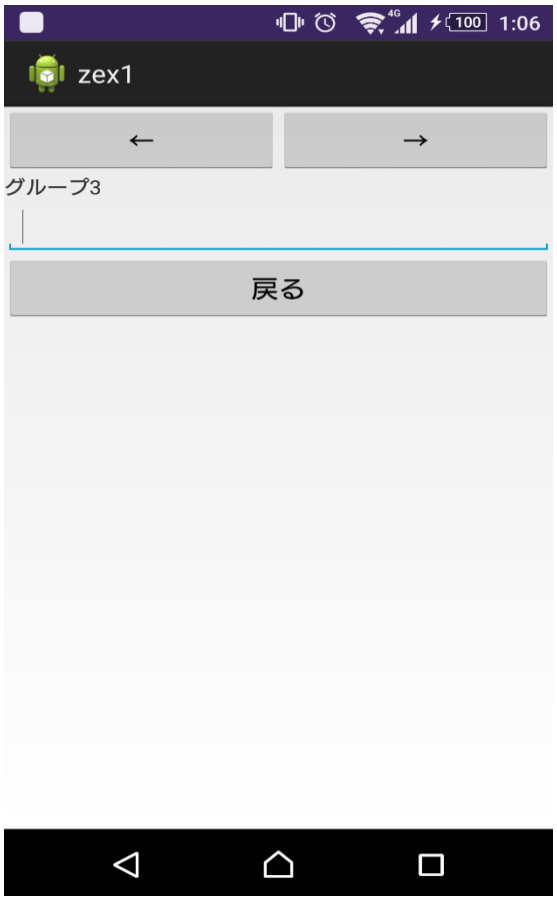

図　33 エラー画面 2

今回は Google Play には投稿せず身近な人に使ってもらってアンケート調査しました。第 4 章でも述べましたが、グループ分けアプリ自体が少ないので、ダウンロード数も少なく正確なデータは取れないかもしれませんが、投稿してから世界中の人にアンケートを実施してもよかったかもしれません。

作成してみての感想は、簡単な部分もあったけど難しい部分も多かったです。全体のボタン配置やデザイン等は過去の授業資料を元にしたのですぐにできましたが、“ Number Picker”の実装や振り分けるためのコード、ページ移行のコードは特に制作時間がかかりました。

このアプリの開発コンセプトは、グループ分けを「素早く」かつ「簡単に」するというものでした。アンケート結果を見ても、分ける速さの評価が高いことから「素早く」というのはできていますが、「簡単に」は 40 代使いやすさの結果からもわかるように達成できていないように感じます。

“Grouping You”には「ボタンや Number Picker の配置をわかりやすくする」や「エラー表示を付ける」のようにまだまだ改善する点が数多くあります。第 4 章でも述べましたが、“ TeamMaker ”がグループ分けの正確さ、及び操作速度を合わせ持つ“Grouping You”の最終目標とも言えます。この“Grouping You”を改善しつつ新しいアプリも開発していきたいと思います。

謝辞

本研究を進めるにあたり、熱心に御指導頂きました田中章司郎教授には心より御礼を申し上げます。また、同じ研究室の皆様には数々の御協力と御助言を頂きましたことを深く感謝します。

なお、本論文、本研究で作成したプログラム及び、データ、並びに関連する資料など全ての知的財産権を本研究の指導教員である田中章司郎教授に譲渡します。

引用文献

1 掲載 HP(http://www.mext.go.jp/b_menu/toukei/chousa01/kihon/1267995.htm)
2 掲載 HP(http://e-words.jp/w/%E3%82%B9%E3%82%BF%E3%83%83%E3%82%AF.html)
3 掲載 HP(https://play.google.com/store/apps/details?id=uka.ayagi.android.divide_group&hl=ja)
4 掲載 HP(https://play.google.com/store/apps/details?id=jp.gr.java_conf.eng_takenach.teammaker&hl=ja)

[付録]“Grouping You”プログラムコード

・メイン画面プログラムコード

```
package com.example.zex1;


import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.util.Log;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.LinearLayout;
import android.widget.NumberPicker;


public class MainActivity extends Activity implements OnClickListener {



        private Button bt1,bt2;
        @SuppressWarnings("unused")
        private NumberPicker np;
        private NumberPicker np1;
        private NumberPicker np2;
        private final int MP = ViewGroup.LayoutParams.MATCH_PARENT;
        private final int WC = ViewGroup.LayoutParams.WRAP_CONTENT;
```

```
private static final int INITIAL_VALUE   = 2;
Globals globals;

@Override
protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        //setContentView(R.layout.activity_main);

        globals = (Globals) this.getApplication();

        LinearLayout layout = new LinearLayout(this);
        layout.setOrientation(LinearLayout.VERTICAL);
        setContentView(layout);


        LinearLayout ll1 = new LinearLayout(this);
        ll1.setOrientation(LinearLayout.HORIZONTAL);

        LinearLayout.LayoutParams params0 = new
LinearLayout.LayoutParams(MP,WC);
        params0.weight = 1;

        np1 = new NumberPicker(this);
```

```
// 最小値と最大値の設定
np1 = new NumberPicker(this);
np1.setLayoutParams(params0);
np1.setMaxValue(50);
np1.setMinValue(2);
np1.setValue(INITIAL_VALUE);
ll1.addView(np1);

np2 = new NumberPicker(this);
np2.setLayoutParams(params0);
np2.setMaxValue(10);
np2.setMinValue(2);
np2.setValue(INITIAL_VALUE);
ll1.addView(np2);

layout.addView(ll1);

LinearLayout ll2 = new LinearLayout(this);
ll2.setOrientation(LinearLayout.HORIZONTAL);

bt1 = new Button(this);
bt1.setText("シャッフル");
bt1.setOnClickListener(this);
bt1.setLayoutParams(params0);
ll2.addView(bt1);
```

```
        bt2 = new Button(this);
        bt2.setText("リセット");
        bt2.setOnClickListener(this);
        bt2.setLayoutParams(params0);
        ll2.addView(bt2);

        layout.addView(ll2);

}

@Override
public void onClick(View v) {
        // TODO 自動生成されたメソッド・スタブ

        // リセットしたときの数値を設定
        if (v == bt1) {

                globals.ninzu = np1.getValue();
                Log.d("Debug", "ninzu=" + globals.ninzu);

                globals.groups = np2.getValue();
                Log.d("Debug", "groups=" + globals.groups);
```

```
                Intent intent = new Intent(this, Sub1.class);
                startActivity((android.content.Intent) intent);
            } else if (v == bt2) {
                np1.setValue(2);
                np2.setValue(2);


        }


}}
```

・サブ画面プログラムコード

```
package com.example.zex1;

import java.util.Random;

import com.example.zex1.MainActivity;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.MotionEvent;
```

```
import android.view.View;
import android.view.ViewGroup;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.TextView;

public class Sub1    extends Activity implements OnClickListener {


        private int person[];
        private int ninzu, groups, total;
        private Random rnd;
        private int r, now;
        private Button bt_l, bt_r;
        private TextView tv1;
        private EditText et1;
        private final int MP = ViewGroup.LayoutParams.MATCH_PARENT;
        private final int WC = ViewGroup.LayoutParams.WRAP_CONTENT;


        private Button bt1;
        Globals globals;
```

```
@Override
protected void onCreate(Bundle savedInstanceState) {
        // TODO 自動生成されたメソッド・スタブ
        super.onCreate(savedInstanceState);

 globals = (Globals) this.getApplication();

        LinearLayout.LayoutParams params0 = new
LinearLayout.LayoutParams(MP,WC);
        params0.weight = 1;

        ninzu = globals.ninzu;
        groups = globals.groups;
        total = ninzu;
        person =new int[total];

        for (int i = 0; i < total; i++) {
                person[i] = -1;
        }

    rnd = new Random();

    // グループ番号の割り当て
    for (int i = 1; i <= groups; i++) {
         for (int j = 0; j < ninzu/groups; j++) {
```

```
                while (true) {
                        r = rnd.nextInt(total);
                        if (person[r] == -1) {
                                person[r] = i;
                                break;
                        }
                }
        }
}

// 各グループの人数が同じでない場合に、残りの人にグループ番号を割り当てる
for (int i = 1; i <= ninzu - (ninzu/groups)*groups; i++) {
        for (int j = 0; j < 1; j++) {
                while (true) {
                        r = rnd.nextInt(total);
                        if (person[r] == -1) {
                                person[r] = i;
                                break;
                        }
                }
        }
}

        LinearLayout layout = new LinearLayout(this);
```

```
        layout.setOrientation(LinearLayout.VERTICAL);
        setContentView(layout);

        LinearLayout ll = new LinearLayout(this);
        ll.setOrientation(LinearLayout.HORIZONTAL);
        layout.addView(ll);

        bt_l = new Button(this);
bt_l.setText("←");
bt_l.setOnClickListener(this);
bt_l.setLayoutParams(params0);
ll.addView(bt_l);


        bt_r = new Button(this);
bt_r.setText("→");
bt_r.setOnClickListener(this);
bt_r.setLayoutParams(params0);
ll.addView(bt_r);


now = 1;
        tv1 = new TextView(this);
        tv1.setText("グループ 1");
```

```
        layout.addView(tv1);


        String kekka = "";
            for (int i = 1; i <= total; i++) {
                    if (person[i-1] == 1) {
                            kekka = kekka + i + "¥n";
                    }
            }


        et1 = new EditText(this);
        et1.setText(kekka);
        layout.addView(et1);


        bt1 = new Button(this);
    bt1.setText("戻る");
    bt1.setOnClickListener(this);
    layout.addView(bt1);




            /*
            globals = (Globals) this.getApplication();


            LinearLayout layout = new LinearLayout(this);
```

```
        layout.setOrientation(LinearLayout.VERTICAL);
        setContentView(layout);

        TextView tv1 = new TextView(this);
        tv1.setText("グループ");
        layout.addView(tv1);

        TextView tv2 = new TextView(this);
        tv2.setText("データ");
        layout.addView(tv2);

        bt1 = new Button(this);
    bt1.setText("戻る");
    bt1.setOnClickListener(this);
    layout.addView(bt1);
    */

}

@Override
public void onClick(View v) {
        // TODO 自動生成されたメソッド・スタブ

        if ( v == bt1){
                Intent intent = new Intent(this, MainActivity.class);
```

```
            intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
            startActivity(intent);
    }

    // 左矢印で表示グループを一つ戻す
    if (v == bt_l) { // ←
            if ( now > 1) {
                    now--;
                    tv1.setText("グループ" + now);
                String kekka = "";
                    for (int i = 1; i <= total; i++) {
                            if (person[i-1] == now) {
                                    kekka = kekka + i + "¥n";
                            }
                    }
                    et1.setText(kekka);
            }
    }

    // 右矢印で表示グループを一つ進める
    if (v == bt_r) { // →
            if ( now < groups) {
                    now++;
                    tv1.setText("グループ" + now);
                String kekka = "";
```

```
                        for (int i = 1; i <= total; i++) {
                                if (person[i-1] == now) {
                                        kekka = kekka + i + "¥n";
                                }
                        }
                        et1.setText(kekka);
                }
        }



        /*
        if ( v == bt1){
                Intent intent = new Intent(this, MainActivity.class);
                intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                startActivity(intent);
        }
        */
    }

}
```