ゼロから始める Android プログラミング
-1 週間で携帯アプリを作れるか？-

万里川則亮* 高津和紀** 平田隆幸**

Beginners Try to Learn Android Programming
-Can They Master Android Programming in a Week? -

Noriaki MARIKAWA*, Kazuki TAKATSU and Takayuki HIRATA****

(Received September 27, 2019)

Can programming beginners create an application of mobile devices after 7 days lesson? To answer this question, we have done an experiment. One lesson was limited to an hour. Java on Android was used as a programming language. Five students who major in engineering took part in this attempt. They were beginners in Java programming and unfamiliar with making an application on mobile devices. Before and after 7 days lesson, the questionnaire surveys were carried out. The participants of the lesson succeeded in making a Java application on Android devices.

Key Words **:** Mobile Device, Android, Java, Programming beginners, Education

1. 緒 言

携帯端末は，広く普及するとともに，その処理能力も飛躍的に向上してきている．これに伴い，携帯端末は通信手段としての利用のみならず，様々な利用がなされるようになってきた[1]. 携帯端末は, 通信機能をはじめ, PC やゲーム機としても使用されている．また，携帯端末は PC と比較して，老若男女問わずより身近なものとなっている．現在，携帯端末上で動くアプリケーションには，多種多様なものが存在し，生活の一部になりつつある．また，携帯端末で動くアプリケーションは，新しい可能性を秘めているとともに，技術やサービスを低コストで広く供給することが可能である．

携帯端末を使った多種多様なアプリケーションが存在するが，これらのアプリケーションには，一般の人々が開発したものも多く含まれている. それ故, アプリケーションを製作してみたいと考えている人は多い[2]. そこで, プログラミングの初心者が携帯端末のアプリケーションを開発できるかという点に注目する．また，それらの人々が短期間でアプリケーションをプログラミングできるかの可能性について議論する．

スマートフォンと呼ばれる携帯端末には，iOS(iPhone，iPad)で動作するものと，Android OS で動作するものがあり，この 2 つのプラットフォームが主流である．代表的な開発環境として，iOS では Xcode が挙げられ, Android OS では Android Studio を挙げることができる[3]. ここでは開発環境に Android Studio を用いる．その理由として，世界的なシェアでは Android 端末が多くを占めているところにある．また，Android アプリを配信する google play では，開発者にかかるアプリ配信時のコストが App store に比べて低い．その為，だれでも簡単かつ気軽にアプリケーションを広く配布することが可能である[4].

ここではプログラミング初心者の学生に，7 日間の講習をおこない，Android OS 上のアプリケーションの製作が可能になるかを調べた．

* 機械・システム工学科
** 大学院工学研究科知能システム工学専攻
* Dept. of Mechanical and System Engineering
** Human and Artificial Intelligent Systems Course, Graduate School of Engineering

2. Andorid Programming 環境インストール

Android のアプリケーション開発は, Android アプリを用いて Android 端末上で行うことが可能である[5]．しかし，Android の携帯端末上で，プログラム開発をおこなうより, PC をプログラム開発のプラットフォームにした方が便利である．ここでは，PC 上でのアプリケーション開発をおこなう．

2.1 PC プラットフォーム

開発環境について述べる．プラットフォームは，最低限のハードウェア(Windows10 が起動するもの：CPU Core-i5, RAM 2GB 以上, HDD 最低 4ＧＢ以上), モニター(24 インチ，1280×800 以上)をもちいた．また，代表的な開発環境を表 1 にまとめた．対象の携帯端末のバージョンとして，Android4.4(KitKat)を想定した．アプリの実行は，Android Studio に搭載されている仮想デバイス上で行った．

次に，開発言語について述べる．Android アプリケーションの開発言語には，C，Kotlin，Ruby，Java などがある．図 1 に，上位 3 つのプログラミング言語の移り変わりを示す．データは，オランダの TIOBE Software 社が公開した「TIOBE Index」の，2018 年 12 月版を使った[6]. なお, このランキングは複数の検索エンジンの検索結果から，各言語がどれだけ話題になっているのかを評価したものである．

次に，開発言語の特徴をみる．C 言語は Unix とともに発展してきた．開発言語としては最も歴史があり，制御など様々な分野で使用されている．また，プログラミング言語の教育の現場で最もよく使われている言語である．Kotlin は，Java 言語をより簡潔に書くことを目指して作られた言語である．Java 言語とは直接的な互換性がある. 2017 年には Google が Kotlin を Android アプリケーション開発の推奨言語とした[7]．Ruby は，日本人の松本行弘によって開発された言語である[8]．きれいな構造的なプログラミングができ，人気がある．Java は，最も人気のあるプログラミング言語である. 人気の要因として, Java は OS に依存することなく開発することができることにある．ここでは，Java を使用して携帯アプリケーションを作成する．

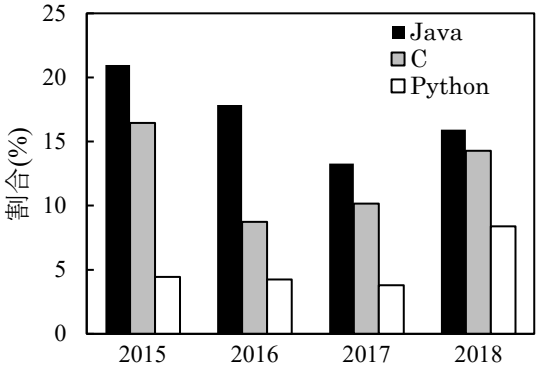

図 1　プログラミング言語の人気の変遷.

2.2 Java Android 環境

Android Studio はエディタやコンパイラ，リンカ，デバッガまたはその他の支援ツールなどを統合した開発環境である[9]．また，動作の確認には，実際の Android 端末ではなく，Android Studio に搭載された仮想デバイスを用いる．

3. 1 週間で Android アプリを作る

プログラミング初心者の大学生を対象に，短期間で Android アプリの製作が可能かを調べた. 1 日 1 コマ(60 分), 7 日でアプリ製作を目指す．なお，開発言語として Java を使用した.

3.1 被験者

Java 言語のプログラミング初心者，5 名を被験者とした．年齢は，21～23 歳の男子 4 名，女子 1 名である．福井大学工学部の知能システム工学科の 4 年生である．なお，Android Studio はプログラミング言語 Java のほかに Kotlin を用いることができる．講習を始める前に Android Studio に触れたことがあるか，特に Java を使ったことがあるかなどについてアンケートをおこなった．プログラミング経験についてのアンケートの結果を表 2，表 3 に示す（アンケートの内容は付録を参照).

被験者の携帯端末上のアプリケーションプログラ

表 1　代表的な開発環境.

ハードウェア	Dell Inspiron 3470
CPU	i5-8400CPU
RAM	8GB
ソフトウェア	Android Studio
モニター	24 インチ，1280×800

表 2　被験者の開発環境に対しての理解度*.

理解度レベル	1	2	3	4
Java	0 人	3 人	2 人	0 人
Android Studio	0 人	2 人	3 人	0 人

*1 は理解している，2 は少し理解している，3 は理解していない，4 は分からない.

表 3　被験者の経験.

	有	無	わからない
自作アプリ製作	0 人	5 人	0 人
開発環境の導入	0 人	4 人	1 人

ミングの経験についてまとめたものが表 3 である．経験の有無は，講習前のアンケートをおこなうことで調査した．表 3 から分かるように，被験者は開発環境を導入した経験がなかった．また，被験者 5 人は，アプリの製作経験もなかった．

3.2 講習内容と達成目標

教材の内容を要約する．講習は，7 日間でおこなわれ，1 日が 1 回分の講習に対応する．7 回分の講習の内容を以下に箇条書きで示す．

1 回目：準備
2 回目：出力編 ～文字列～
3 回目：出力編 ～図～
4 回目：入力編 ～ボタン～
5 回目：入力編 ～複数のボタン～
6 回目：入力編 ～文字列～
7 回目：センサの利用

なお，理解度調査のため，全 7 回のうち，偶数回目(2，4，6 回目)の講習に課題 1～3 をおこなった．課題の出力例を図 2 に示す．

受講者の達成目標について述べる．大きな目標は，Android アプリケーションを 1 週間で製作することである．目標達成のため，3 つの課題を設けた．課題 1，仮想デバイスに，複数の文字列を表示する．課題 2 ，仮想デバイスに，ボタン(インターフェース部)を配置する．課題 3，仮想デバイス上で計算機を作成する．

4. 実験と分析

プログラミング初心者 5 人に 7 日間の講習で Java アプリ製作できるようになるかの実験をおこなった．そして，設定した課題を達成できたかを調べた．設定した課題は，3 つである．以下に，課題の内容を述べる．

課題 1 は，複数の文字を Android 端末の画面に出力する課題である(図 2−a)参照)．課題 1 には，任意の文字サイズと文字の色に変えるという課題が含まれており，文字の設定をできるかで理解度を測る．

課題 2 は，ボタンを押すと乱数が決まり，Android 端末の画面にその結果を出力させる課題である(図 2−b)参照)．出力の結果には図や数式を用いる．インターフェース部が反応するか，乱数とその他のコンポーネントを組み合わせてあるかで理解度を測る．

課題 3 は，計算機をつくる課題である(図 2−c)参

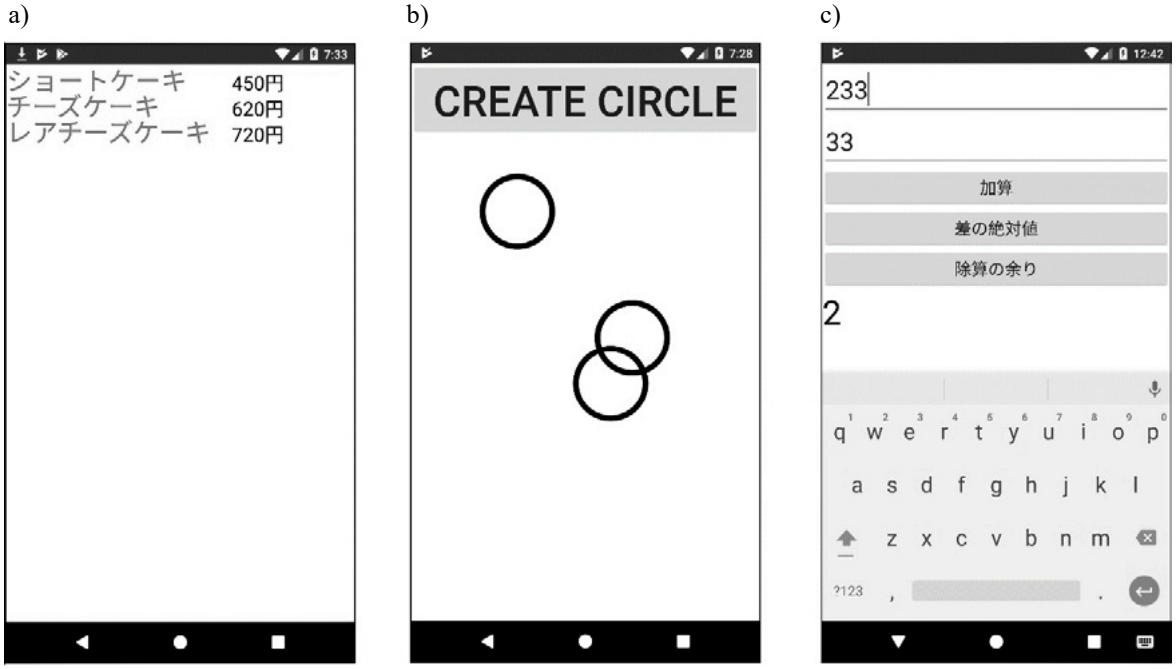

図 2　課題の出力例．a)課題 1(文字の大きさと色が異なる文字列の出力．)，b)課題 2(CREATE CIRCLE をクリックすると，ランダムな位置に円が描画されるボタン．)，c)課題 3(計算機: 上部には値の入力部分が 2 つ．中部には加算，差の絶対値，除算の余りの計算ボタン．下部には計算結果を出力させる．)．

照)．計算機をつくるために，入力部としてボタンとエディットテキストボックスをそれぞれ複数用いる．ボタンからは，加算や減算などの演算の内容を受け取り，エディットテキストボックスからは数値を受け取る．これらの 2 種類の入力部から Android 端末の画面に結果を出力できるかで理解度を測る．

被験者の自己評価および教師による評価をみていく．被験者 5 名の各課題に対して達成に要した時間を表 4 に示す．各課題には制限時間の設定をしている．課題 1 には 15 分，課題 2 には 20 分，課題 3 には 40 分である．いずれも 1 コマ(60 分)の時間の中に含まれている．

表 4 から被験者の特徴について考察する．まず，3 つの課題を達成するのに要した合計時間という観点から被験者をみてみよう．最も多くの時間を要した被験者 B と最も少ない時間で達成できた被験者 A を比べる．達成時間に約 30 分の差が生じている．これは，達成できなかった課題の達成時間を制限時間と換算してさえ, 1.6 倍以上の時間を必要としたことが分かる．また，被験者 C は，課題 2 では時間内に達成できなかったにも関わらず，課題 3 では特に優秀な時間で課題を達成している．このように，課題によるばらつきも存在していることが分かる．

次に，課題の達成割合という視点から考察する(表 5 を参照)．制限時間内に課題を達成できなかった課題を未達成とし，課題別の達成割合を達成項目数から決めている．表 5 から，課題 1 は全ての被験者が達成することができた．つまり，課題 2，3 に必要となる基礎的な部分は理解できていると考えられる．課題が達成できなかった者を注目すると，被験者 B は課題 2，3 で，被験者 D は課題 3 で達成することが出来なかった．被験者 B，D は難易度に連れて達成割合が下がったとみることができる．これに対して被験者 C は課題 2 を達成できなかったにも関わらず課題 3 を達成していることに注目する．このことからプログラミングの学習でも，一部の内容に理解がなくとも，回を重ねるごとに理解が深まる可能性があるといえる．

被験者の講習に対する理解度の自己評価という観点からみていく．自己評価は，講習終了後にアンケートによって調べた(表 6 を参照)．ここでは，課題を達成した場合は○, 達成できなかった場合は×と，二者択一で回答してもらった．なお，被験者の自己申告によるものであり，客観的に課題を達成できたかを意味しないことに注意が必要である．

表 4　課題の達成時間*.

	課題 1	課題 2	課題 3	合計
被験者 A	3	5	36	44
被験者 B	13	×(20)	×(40)	73
被験者 C	6	×(20)	21	47
被験者 D	5	7	×(40)	52
被験者 E	5	9	35	49

*時間の単位は分である．×は()内の制限時間内に達成できなかった場合である．

表 5　課題別の達成割合(%)*.

	課題 1	課題 2	課題 3	全体
被験者 A	100	100	100	100
被験者 B	100	50	33	61
被験者 C	100	50	100	83.3
被験者 D	100	100	66	88.7
被験者 E	100	100	100	100

*各課題の達成項目は課題 1 が 2 つ，課題 2 が 2 つ，課題 3 が 3 つとなっている．

表 6　被験者の自己評価.

	課題 1	課題 2	課題 3
被験者 A	○	○	○
被験者 B	○	○	×
被験者 C	○	○	○
被験者 D	○	○	×
被験者 E	○	○	○

表 7　被験者の講義に対する評価.

	項目 2	項目 3	項目 4	項目 5
被験者 A	3	3	3	3
被験者 B	2	3	2	2
被験者 C	3	2	2	3
被験者 D	2	2	2	3
被験者 E	2	2	3	3

表 8　アンケートの内容.

項目 1	被験者の自己評価(表 6 を参照)
項目 2	講習の配布テキストは適切であったか
項目 3	講習の難易度は適切であったか
項目 4	講習の分量は適切であったか
項目 5	講習に対して満足できたか

同時に，講義についても被験者に評価してもらった．調査は，講義後のアンケートを通しておこなった．評価は，0 から 3 の 4 段階である．評価基準として，評価 0 は低評価，評価 1 はやや低評価，2 はやや高評価，評価 3 は高評価としている．表 7 に評価被験者ごとの各項目に対する評価を示す．アンケートの内容をまとめたものを表 8 に示す．次に，各因子間の相関関係を調べた(相関関係を図 3 から図 7 に示す)．各因子は，達成時間，達成割合，満足度，自己評価の 4 つの因子である．

図 3 は，達成時間を横軸にとり，縦軸に達成割合をプロットしたものである．達成時間と達成割合には強い負の相関(相関係数 r = - 0.90)があった．このことから達成度の高い被験者はより短い時間でプログラムを組むことが出来ていることがわかる．合計 7 時間という短い講習でも，相関関係にあらわれるような理解度の差がうまれている．

さて、特徴的な被験者をみることで，よりよい講習に改善できる可能性があるかもしれない．ここでは，被験者 C に注目する(表 5 を参照)．課題 2 が設定時間内に達成出来ていないことに対し，課題 3 では他の被験者と達成時間に差があることがわかる．各課題では講座時間にして 2 時間の講習がなされている．このことに起因して，理解度に十分な変化をもたらしていることが考えられる．また，図 3 からわかるように時間を最大限使って課題をすべて達成した被験者は現れなかった．図 4 は，達成時間を横軸にとり，縦軸に満足度をプロットしたものである．達成時間と満足度には強い負の相関(相関係数 r = - 0.95)があった．短い実習時間で課題を達成できた被

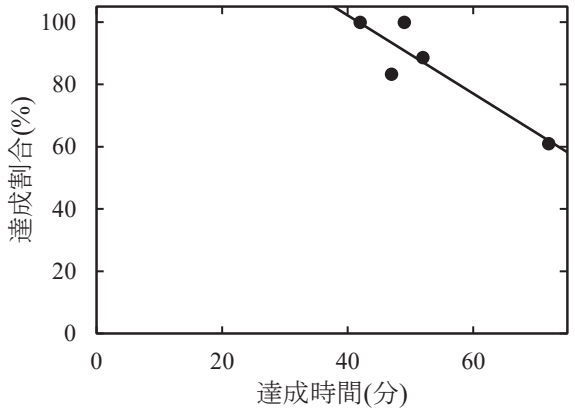

図 3　達成時間と達成割合．

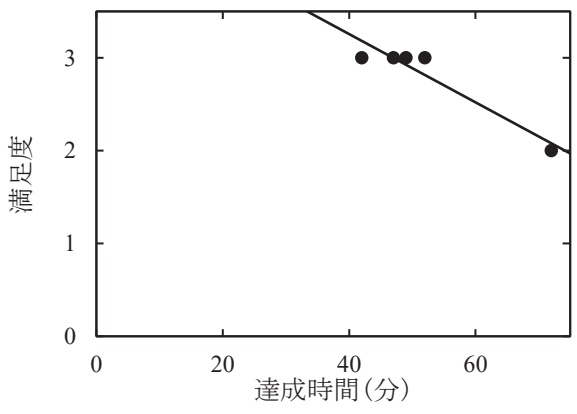

図 4　達成時間と満足度．

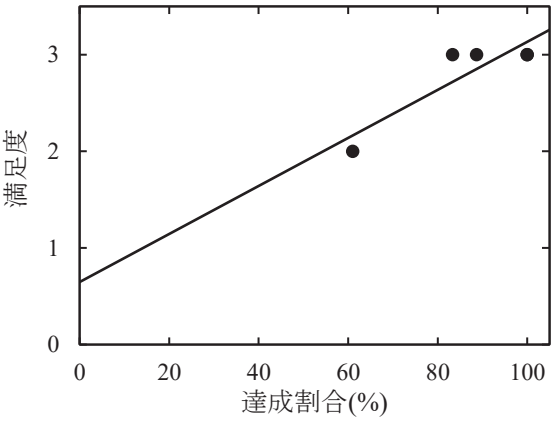

図 5　達成割合と満足度．

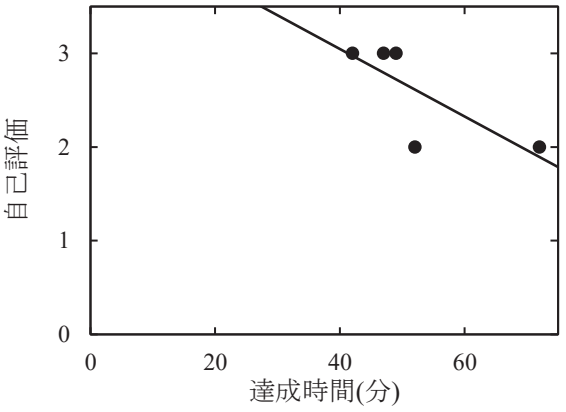

図 6　達成時間と自己評価．

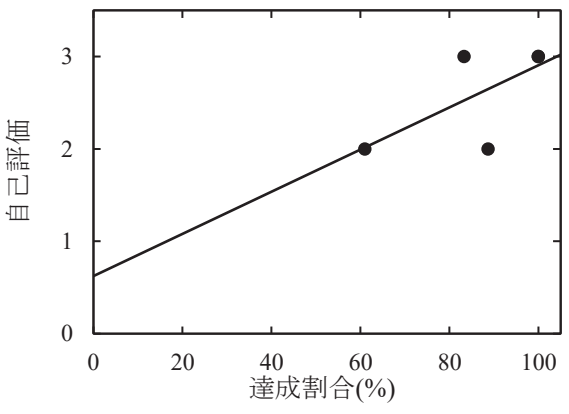

図 7　達成割合と自己評価．

験者が満足度の高い傾向にある．図 4 から，満足度の低い被験者は他の被験者に比べて達成時間に大きな差がうまれている．

図 5 は，達成割合を横軸にとり，縦軸に満足度をプロットしたものである．達成割合と満足度には強い相関(相関係数 r＝0.89)があった．ここで図 3，4 の結果を考える．達成割合と満足度の両方に対し，達成時間は強い負の相関をもっていた．

次に，達成時間と達成割合に対して，被験者の自己評価はどのような関係があるかをみていく．図 6 は，達成時間を横軸にとり，縦軸に自己評価度をプロットしたものである．達成時間と自己評価には負の相関(相関係数 r = - 0.76)があった．しかしデータ数が 5 と少ないため，相関係数 - 0.76 は相関関係として十分ではないと考えられる．

図 7 は，達成割合を横軸にとり，縦軸に自己評価をプロットしたものである．達成割合と自己評価には強い相関関係(相関係数 r = 0.67)がみられなかった．プログラミングにおける自己評価は曖昧なものであると考えられる．

達成時間と達成割合は客観的な理解度を示す重要な項目である．一方，自己評価もしくは満足度は主観的な項目である．それゆえ将来的には，自己評価もしくは満足度から講習の効果を定量的に調べる方法の確立が必要と考えられる．

図 3〜7 までの結果から，理解度を測るには，達成時間と達成割合がともに高い相関関係を示したことより，達成時間と達成割合によって推定できることがわかった．なお，図 3〜7 の回帰直線を表 9 に，相関係数を表 10 にまとめておく．

表 9　回帰直線.

x	y	回帰直線
達成時間	達成割合	y = - 1.3x + 150
達成時間	満足度	y = - 0.037x + 4.7
達成割合	満足度	y = 0.025x + 0.65
達成時間	自己評価	y = - 0.036x + 4.5
達成割合	自己評価	y = 0.023x + 0.62

表 10　各種因子間の相関係数.

	相関係数	図
達成時間と達成割合	- 0.90	図 3
達成時間と満足度	- 0.95	図 4
達成割合と満足度	0.89	図 5
達成時間と自己評価	- 0.76	図 6
達成割合と自己評価	0.67	図 7

5. まとめ

プログラミングの初心者を対象に，1 日 1 コマ(60 分)，7 日間で Android アプリの作成ができるかという実験をおこなった．1 週間で Android アプリを製作するという目標は，概ね達成できた．その結果，被験者 5 名の内 1 名は，少し満足であるとの回答があった．そして，講習前と講習後にアンケート調査をおこなった．さらに，相関関係を調べた．

被験者は，福井大学工学部の 4 年生であった．工学部の 4 年生がプログラミングの初心者であるかどうかは, 議論の余地があるかもしれない. 一般的に，プログラムの経験が工学部の学生より少ない文科系の学生を対象に，同様の調査をおこなうと異なった結果になる可能性はある．また，被験者数が 5 人というのは実験規模としては少ないかもしれない．今後，より大人数を対象とした実験をしたいと考えている．

アプリケーションの作成において，客観的評価と講習終了後のアンケートによる自己評価を比較した．自己評価と客観評価には，十分な相関が認められなかった．被験者は，自己の課題達成に関して，正しく認識できていなかったことが分かる．このことから，アンケートの質問が不適切であったかもしれない．

客観的評価と満足度には強い相関があった. また，達成割合と達成時間にも強い相関が得られたことから，達成割合，達成時間，満足度には相関関係があることが確認できる．ここで被験者の満足度を作為的に上げる方法について考えてみる．達成時間と満足度には負の相関があるので，プログラムの穴埋めのような時間のかからない課題に変更する方法が考えられる．達成割合と満足度には正の相関があるので，課題の難易度を下げることで，被験者の達成割合を意図的に伸ばす方法が考えられる．しかし，これらの方法はプログラミングの習得という点では好ましくないと考えられる．その理由として，実際にプログラミングをおこなう場合，前述した方法の課題よりも長いソースコードを書くことが必要となり，それに伴った複雑な構造を考えなければならないからである．

今後，Java だけでなく Android Studio に搭載されているもう 1 つのプログラミング言語である Kotlin を用いた講習をおこなうことで，言語の習得のしやすさが分かる可能性がある．さらに，Android OS と iOS の違いによる影響は，どれ程かを調べることも重要かもしれない．iOS のアプリケーションについても，本講習と同様の内容でアプリケーション製作

をおこなうことで開発のしやすさ，使い勝手の良さについて判断をおこなうことができると考えられる．

今回の実験は，受講者 5 人という小規模の被験者を対象におこなった．その結果，人数が少ない小規模授業ゆえに，きめこまやかな確認をすることが可能であった．これに対し，例えば，30 人ほどを対象とした場合を考えると，7 日間で受講者が Android アプリの作成ができるようになるのかは今後の問題である．

最後に，不特定多数の人間に講座の内容を紹介したいと考えている．INTERNET 上にテキストやサンプルプログラムを閲覧可能にした場合，閲覧者が教師なしで Android アプリの作成ができるようになるのかを調べることも今後の課題である．

謝　辞

研究をおこなうにあたり，被験者として協力してくださった研究室のメンバーに感謝いたします．また，論文を執筆するにあたり，議論および有益なコメントをしてくださった髙田宗樹教授に感謝いたします．

参考文献

[1] 濱谷英次 : 携帯電話を巡る技術社会史 : 技術的慣性から社会的慣性へ大阪大学大学院人間科学研 究科博士論文(2012).

[2]プログラミング初心者が始めるアプリ開発 <https://www.creativevillage.ne.jp/20846> (2019 年 9 月 13 日)

[3] iPhone(iOS)と Android のアプリ開発はどう違う？開発言語や開発環境の違いまとめ <https://agency-star.com/freelance/articles/314/> (2019 年 9 月 13 日)

[4] Android と iOS アプリ制作．どちらから勉強すべきか? <https://www.selva-i.co.jp/blog/archives/2531> (2019 年 9 月 13 日)

[5] AIDE- IDE for Android Java C++ <https://play.google.com/store/apps/details?id=com.aide.ui&hl=ja>(2019 年 9 月 10 日)

[6] TIOBE Software TIOBE Index(2018 年 12 月版) < https://www.tiobe.com/tiobe-index/>(2019 年 9 月 13 日)

[7] Google I/O 基調講演 (Google I/O '17) < https://events.google.com/io2017/>(2019 年 8 月 20 日)

[8] 高橋征義，後藤裕蔵，まつもとゆきひろ監修 : たのしい Ruby, ソフトバンクパブリッシング株式会社，pp.467(2002).

[9] Android Developer <https://developer.android.com/studio?hl=JA>(2019 年 9 月 13 日)

[10] 布留川英一 : Android プログラミングバイブル SDK 7/6/5/4 対応，ソシム，pp.18-164 (2017).

[11] 高橋麻奈 : やさしい Java，風工舎 (2013).

付録 Appendix

0 から始める Android プログラミング
事前アンケート

年　　月　　日

年齢

1. あなたはプログラミング言語である Java のオブジェクト指向を理解していますか？

・理解している　・少し理解している　・理解していない　・わからない

2. あなたは Android アプリの構成(基本的な Activity クラス)を理解していますか？

・理解している　・少し理解している　・理解していない　・わからない

3. Java のプログラミング経験がありますか？

・ある　・ない　・わからない

4. 自身でプログラムしたアプリを Android 端末で実行したことがありますか？

・ある　・ない　・わからない

5. あなたは一人で開発環境を整えることができますか？

・できる　・できない　・一部行ったことがある

6. あなたは開発環境を整えてある PC を持っていますか？

・ある　・ない　・わからない

情報の取り扱いについて
本アンケートで得られた個人情報に関しては，研究活動の目的のみで使用し，ご本人の同意なく第三者に開示・提供することはいたしません.