卒業論文

Firebaseを用いた

位置情報共有システム

提出者 半田勝之

提出年月日 平成31年1月25日

指導教員 金子邦彦 教授

Firebaseを用いた

位置情報共有システム

情報工学科 半田勝之

研究概要

　現在，スマートフォンの位置情報は，カーナビゲーションや地図などのアプリケーション，宅配の現在地表示サービスなど様々な目的で利用されている．本研究では，スマートフォンの位置情報やマーカー，チャットなどの情報をグローバルで共有し，コミュニケーションを取ることができる位置情報共有システムを試作した．Firebaseと連携したアプリケーションの機能の動作と，Firebase内のデータベースの動作を調べた．

目次

[1. まえがき 1](#_Toc536267956)

[2. Firebase 2](#_Toc536267957)

[3. 位置情報共有システム 3](#_Toc536267958)

[3.1 データベース 3](#_Toc536267959)

[3.2 iOSアプリケーション内でのGPSの取得 4](#_Toc536267960)

[3.3 iOSアプリケーション内でのFirebaseの利用 4](#_Toc536267961)

[3.4 アプリケーションの基本機能 5](#_Toc536267962)

[3.4.1 ユーザ 5](#_Toc536267963)

[3.4.2 マーカー 6](#_Toc536267964)

[3.4.3 チャット 8](#_Toc536267965)

[4. 評価実験 9](#_Toc536267966)

[4.1 基本機能の実験概要 9](#_Toc536267967)

[4.2 基本機能の実験結果 11](#_Toc536267968)

[4.3 大学構内の探索，調査への応用（概要） 15](#_Toc536267969)

[4.4 大学構内の探索，調査への応用（結果） 15](#_Toc536267970)

[5. むすび 17](#_Toc536267971)

[付録 19](#_Toc536267972)

# まえがき

現在，スマートフォンのGPS機能は，地図アプリケーションや，カーナビアプリケーション，コミュニケーションツールなど様々な用途で活躍している．地図アプリケーションは，GPSに合わせて地図上に常に自分の位置や向きが表示される．目的の建物の位置に加えて，そこに到達するまでの経路を自分の位置と合わせて知りたい場合に便利である．また，自分の位置情報だけではなく，地図上で他人と位置情報を共有してコミュニケーションを取ることができる機能やアプリケーションも複数存在している．

位置情報を共有するアプリケーションや機能を利用する場合，電話番号，LINEのID，Googleアカウントなどの個人識別符号を介して共有することが考えられる．その場合，第三者がそれらの個人情報と位置情報を結びつけるのを防ぐために，公開する相手を限定する必要がある．安全ではあるが，個人識別符号を共有したくない場合は不便である．また，限定された相手との共有であるため，個人が地図上に追加した案内マーカーなどの情報を，不特定多数と共有したい場合にも向いていない．相手を限定せず，より多くの人と地図上で様々な情報を共有できれば，さらに便利となり社会的意義がある．

本論文では，個人識別符号を必要とせず，公開された地図上で位置情報や地図上の案内マーカーなどの共有が行える，位置情報共有システムについて説明する．本論文での位置情報共有システムは，ある特定の利用者グループにのみサービスを提供するシステムである．個人識別符号や認証などでデータアクセス制限を制御することは行わない．このとき，位置情報共有システムをいかに簡単に立ち上げることができるか，そして，個々の利用者グループの要望や特性を反映したアプリケーションをいかに簡単に制作できるかが課題になる．その課題の解決を目指して，位置情報共有システムの試作に取り組んだ．このとき，リアルタイムで情報を共有できるデータベースを構築する必要がある．しかし，そのようなデータベースを一から構築しようとすると，サーバの構築も必須となり，サーバの構築後の運用やアプリケーションとの連携の複雑な設定などが負担になるという課題がある．そこで，システムを簡単に開発することに焦点を当て，データベースにMBaaSである，Google 社の提供する「Firebase」を利用した．Firebaseのデータベースが複数の利用者機器で簡単にリアルタイム共有でき，アプリケーションをいかに簡単に制作できるかの確認に取り組んだ．

# Firebase

　位置情報共有システムのデータベースとしてMBaaSであるFirebaseを利用した．MBaaSとはMobile Backend as a Serviceの略であり，モバイルアプリケーション開発向けにバックエンドで動く機能を提供するクラウドサービスのことである．これにより，アプリケーションを開発する際にサーバの構築をすることなく，通信を利用した機能を実装することができるという良さがある．

　FirebaseとはGoogleが提供しているMBaaSである．Firebaseの提供する主なサービスを以下に示す．

1. Cloud Functions

専用サーバーの管理や拡張を行うことなく，カスタムのバックエンドコードでアプリを拡張する機能(3)がある．

1. 認証

認証の機能であるFirebase Auth では，メールやパスワード，Google や Facebook などのサードパーティのプロバイダ，既存のアカウントシステムを直接使用するなど、複数の方法によって認証を行う機能(3)がある．

1. ホスティング

ウェブアプリ専用に作成されたツールを使用して，静的なWebホスティングを行う機能(3)がある．

1. クラウドストレージ

画像，音声，動画，またはその他のユーザーが生成したコンテンツを保存できるクラウドストレージの機能(3)がある．

1. Realtime Database

ホスティングされた NoSQL データベースを使用して，リアルタイムでユーザーと端末の間での同期を行う機能(3)がある．

　位置情報共有システムの試作にあたって，位置情報のほかに名前やIDなどの文字の情報を端末間で共有させるためにRealtime Databaseを利用した．また，写真を共有させるためにクラウドストレージを利用した．詳しくは，3章で説明する．

# 位置情報共有システム

位置情報共有システムは，iOSアプリケーションとデータベースから成る．

データベースとアプリケーションの詳細をそれぞれ分けて後述する．

## データベース

Firebase のRealtime Databaseを利用し，JSON形式のリアルタイムデータベースを構築した．JSON形式のデータベースは，すでに，位置情報の共有以外にも，さまざまな目的で利用されており，利用例は，Twitterに投稿されたテキストや位置情報などの複数の情報の管理(1)や，地図(2)などがある．

データベースの構築手順は以下の通りである．

1. Googleアカウントを作成．
2. Firebaseにサインイン．
3. 「プロジェクトを追加」をクリックでプロジェクトを追加．
4. 「データベースを作成」をクリックでリアルタイムデータベースを作成．
5. テストモードで運用を開始．
6. 「iOSアプリにFirebaseを追加」をクリックし，Google APIを発行する．アプリのバンドルIDを入力し，「GoogleService-Info.plist」ファイルをダウンロード．

以上は，Webブラウザで操作した．

構築したリアルタイムデータベースをiOSアプリのプログラムであるプロジェクトファイルと結びつけるために，Mac上でソースコードにGoogle APIと GoogleServie-Info.plistファイルを設定した．

詳しい手順は以下の通りである．

1. Xcodeプロジェクトを作成
2. 「pod 'Firebase/Core'」のコマンドを「ターミナル」で実行し，Podfileにポッドを追加する．
3. 「pod install」でポッドをアプリのプロジェクトにインストールする．
4. プロジェクト内にGoogleServie-Info.plistを追加する．
5. プロジェクト内のUIApplicationDelegateにimport FirebaseとFirebaseApp.configure()の２つのプログラムを追加する．

以上で，Firebaseとの連携が完了する．

図1は，Webブラウザ上のデータベースの管理画面であり，データベースの構造を示している．プロジェクト名を親とし，そこから目的ごとに複数の子に派生させ，データを格納した．「Chat」はユーザがチャットに投稿した内容を格納するノードである．「Latest」は最後に更新したユーザの位置情報を格納するノードである．「GPS」は「Latest」の緯度経度を格納するノードである．「Marker」はユーザが設置したマーカーの情報を格納するノードである．「userName」はユーザのIDや名前，コメントなどの情報を格納するノードである．各ノードのさらに深い階層については，機能ごとに後述（節3.5）する．

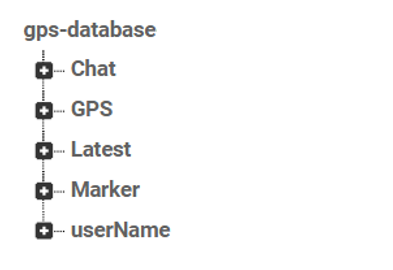


　　　　図1　データベースの管理画面

## iOSアプリケーション内でのGPSの取得

ユーザの位置情報は，iOS端末に搭載されているGPSセンサーから5秒に1回の間隔で取得している．図2は，緯度・経度を取得した後，変数latitude，longitudeに代入しているコードである．

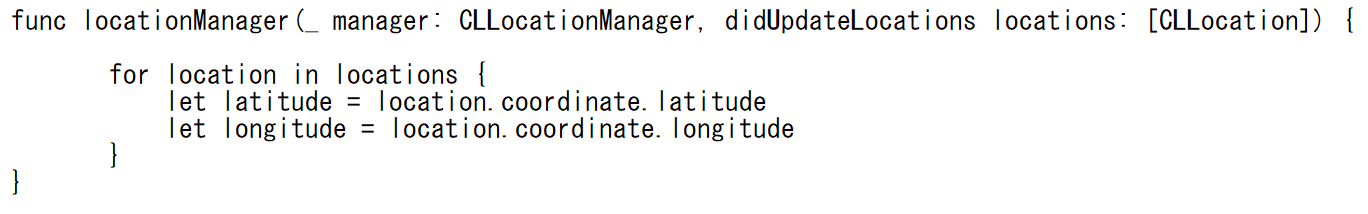


　　　　　　図2　端末の位置情報を取得するコード

## iOSアプリケーション内でのFirebaseの利用

アプリケーションとサーバの通信処理の実装は，図3，図4のようなコードで行っている．図3はFirebaseから情報をダウンロードするコードである．

pick\_jsonという変数にダウンロードしたJSONデータを代入している．図4はFirebaseに情報をアップロードするコードである．「lastnode」はデータベースのパスを示す.updateChildValues()の中にアップロードする情報である，JSONの配列を指定している．コードの詳細は付録に示している．

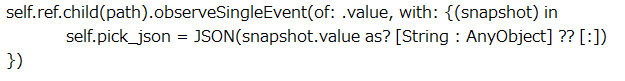


　　　　　　図3　データベースからの情報をダウンロードするコード

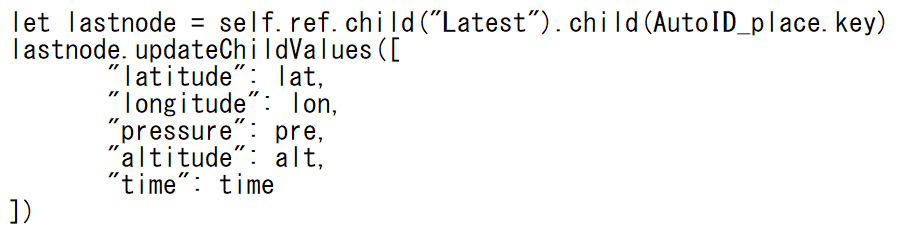


　　　　　　図4　データベースに情報をアップロードするコード

## アプリケーションの基本機能

位置情報共有システムをiOSアプリケーションとして試作した．アプリケーションで扱う情報は，データベースを介して他の端末と共有される．

### ユーザ

　次の機能を実現するにあたって，データベースの「Latest」と「userName」のノードを用いている．

* 自分自身の位置をマーカーと青い円で地図上に表示する機能．
* 他のユーザが最後に更新した位置をマーカーで地図上に表示する機能．
* マーカーの下にユーザ名を表示する機能．
* マーカーをタップすると編集可能な名前・コメントを表示する機能．

図5は「Latest」ノードの構造を示している．ユーザごとにノードを追加しており，その中に，緯度・経度，タイムスタンプなどの情報が格納されている．ここから，各端末が位置情報をダウンロード・アップロードし，地図上のユーザを示すマーカーに反映する．

図6は「userName」ノードの構造を示している．ユーザごとにノードを追加しており，その中に，ユーザの名前やコメントの情報が格納されている．端末がこのユーザの名前をデータベースからダウンロードし，地図上のユーザの位置を示すマーカーの下に表示する名前に反映する．

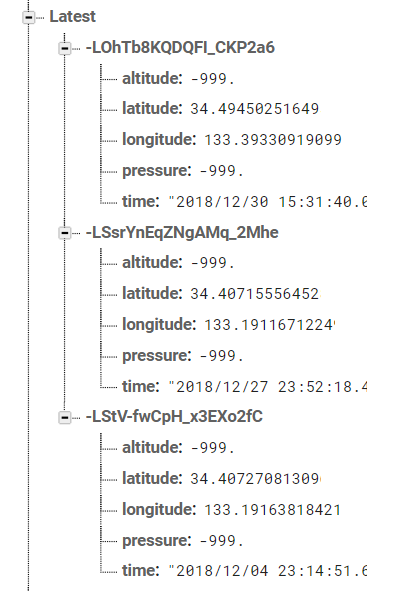




　　　　図5　Latestの構造　　　　　　　　　図6　userNameの構造

### マーカー

次の機能を実現するにあたって，データベースに「Marker」ノードを用いている．画像ファイルはJSON形式に変換せずに，Firebaseのクラウドストレージで管理している．

* タップした場所にマーカーを設置する機能．
* 設置したマーカーをタップすると編集可能な名前・紹介文・写真を表示する機能．
* マーカーに追加された写真を保存する機能．
* 削除する機能．

図7は「Marker」ノードの構造を示している．設置されたマーカーごとにノードを追加しており，各ノードには，マーカーの色をRGBで表した「color」や，掲載している画像名を表す「image\_name」のほかに緯度・経度や名前，コメントなどの情報を格納している．ここから，各端末が情報をダウンロード・アップロードし，地図上のマーカーに反映する．

図8は， Webブラウザでクラウドストレージにアクセスしている画面である．マーカーに掲載する画像を蓄積・管理する．画像ファイル名が重複すると上書きされてしまうため，ファイル名は，マーカーのIDとした．マーカーのIDはマーカーを作成するときに自動的に付与され，他と重複することはない．

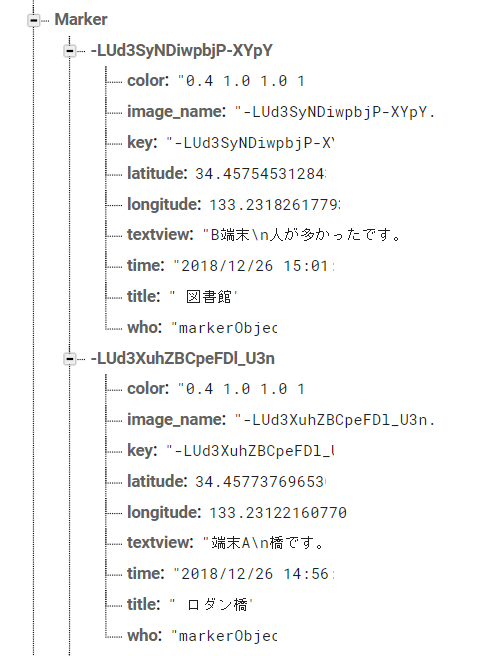


図7　Markerの構造

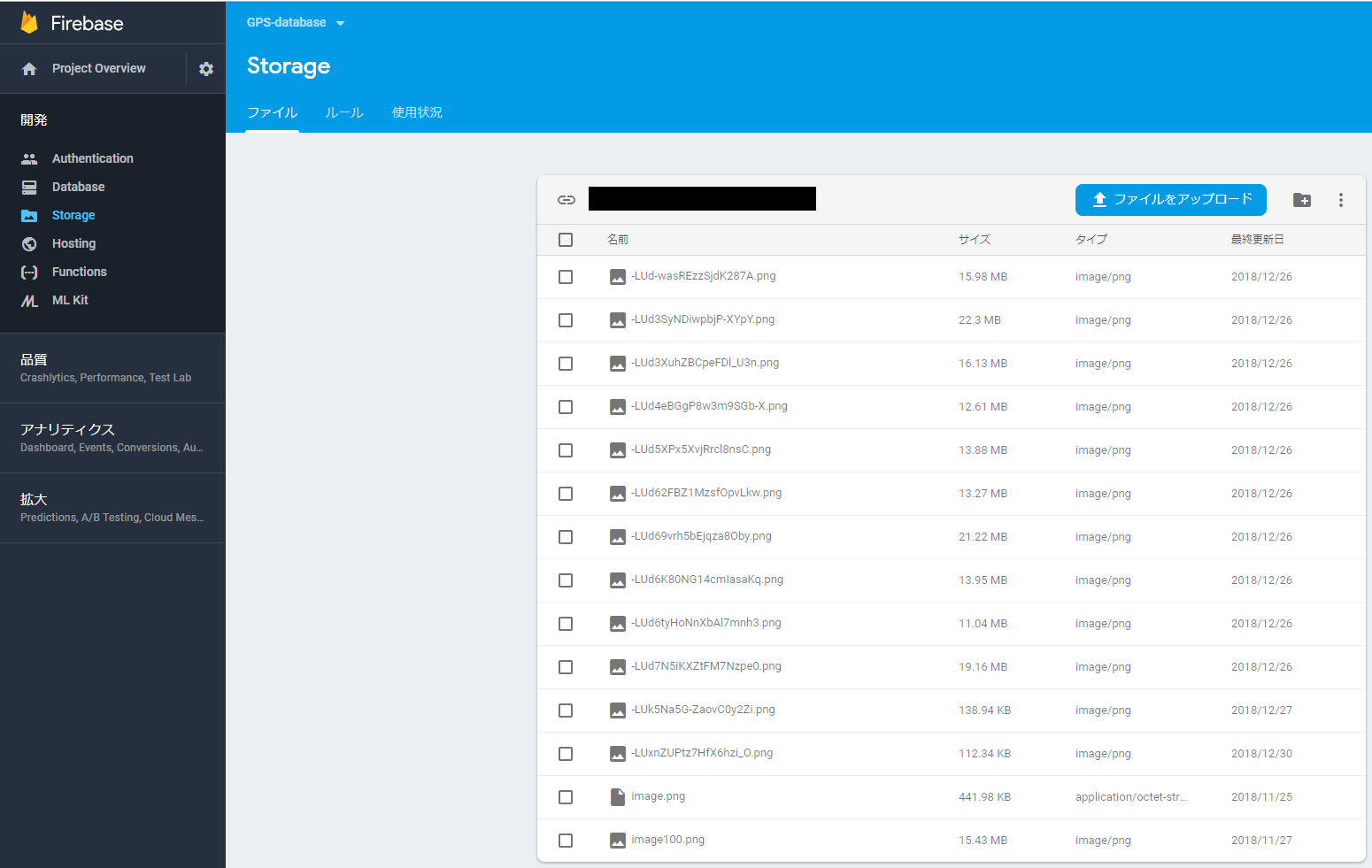


　　　　　　図8　Webブラウザで管理するStorageの画面

### チャット

次の機能を実現するにあたって，データベースの「Chat」ノードを使用している．画像ファイルはクラウドストレージで管理している．

* グローバルなチャットルームにテキスト・写真を投稿する機能．
* チャット内の投稿された写真を保存する機能．

図9は「Chat」ノードの構造を示している．チャットに投稿されたメッセージごとにノードを追加している．その中で，投稿した画像名である「image\_URL」やメッセージの内容である「message」，投稿したユーザのIDやタイムスタンプなどの情報を格納している．

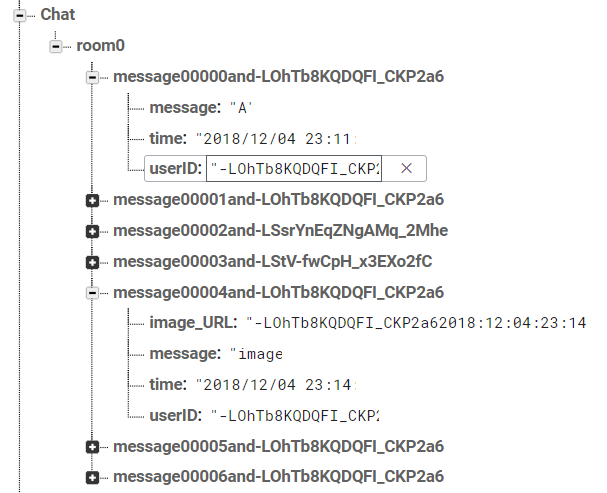


図9　Chatの構造

　これらの機能を実装したソースコードは付録の枠の部分に示した．

# 評価実験

## 基本機能の実験概要

実験は2台のiOS端末で行った．それぞれの端末で地図アプリを操作し，節3.4で説明した機能が動作するかを，アプリケーションを実行している端末とデータベースを表示しているWebブラウザの画面を通して確認した．

データベースをテストモードで運用しており，公開している状態になっている．そのため，実験では，アプリケーション内のユーザ名やコメント，チャット・マーカーのテキストや写真にプライベートな情報は付与せず，個人を特定できるような情報を位置情報と一緒にデータベースへアップロードしないように配慮した．

・**ユーザ**

①自分自身の位置をマーカーと青い円で地図上に表示する機能．

②他のユーザが最後に更新した位置をマーカーで地図上に表示する機能．

③マーカーの下にユーザ名を表示する機能．

④マーカーをタップすると編集可能な名前・コメントを表示する機能．

・**チャット**

⑤グローバルなチャットルームにテキストを投稿・表示する機能．

⑥グローバルなチャットルームに写真を投稿・表示する機能．

・**マーカー**

⑦タップした場所にマーカーを設置する機能．

⑧設置したマーカーをタップすると編集可能な名前・紹介文・写真を表示する機能．

⑨マーカーを削除する機能．

実験の手順は以下の通りである．実験に使用した端末を端末A，端末Bとする．

1. ホーム画面にある試作したアプリケーションのアイコンをタップし，起動する．
2. ①から③は自動で行われるため，操作せずに確認する．
3. 端末Aと端末Bを示すマーカーをタップして表示し，テキストを編集することで④を確認する．
4. 画面右端を左にスワイプすることで，チャットを開く．テキスト入力後，投稿ボタンを押し，⑤を確認する．
5. 写真選択ボタンを押し，端末内の写真を選択後，投稿ボタンを押すことで⑥を確認する．
6. マーカー作成ボタンを押した後，地図上の任意の場所をタップすることで⑦を確認する．
7. 作成されたマーカーをタップし，名前・紹介文・写真が表示されるか確認する．
8. 名前・紹介文のテキストを変更した．写真選択ボタンを押し，写真選択後投稿ボタンを押し，⑧を確認する．
9. マーカーをタップした時に表示される削除ボタンを押し，⑨を確認する．

また，データベースは各操作の度に，アプリケーションの情報が反映されているかWebブラウザの画面を通して確認する．

## 基本機能の実験結果

2つのiOS端末の画面を通して，①から⑨までの機能が正常に動作していたのを確認した．また，位置情報やテキスト，写真などのデータがデータベースに反映されているかwebブラウザ上で確認できた．

図10は節4.1の手順の(3)を行った時の2端末の画面である．節4.1に示した，①②③④の機能が正常に動作していることが確認できた．

図11は節4.1の手順の(4)と(5)を行った時の2端末の画面である．節4.1に示した，⑤⑥の機能が正常に動作していることが確認できた．

図12は地節4.1の手順の(6)を行った時の2端末の画面である．節4.1に示した，⑦の機能が正常に動作していることが確認できた．

図13は節4.1の手順の(7)を行った時の2端末の画面である．

図14は節4.1の手順の(8)を行った時の2端末の画面である．図13，図14から節4.1に示した，⑧の機能が正常に動作していることが確認できた．

図15は節4.1の手順の(9)を行った時の2端末の画面である．節4.1に示した，⑨の機能が正常に動作していることが確認できた．

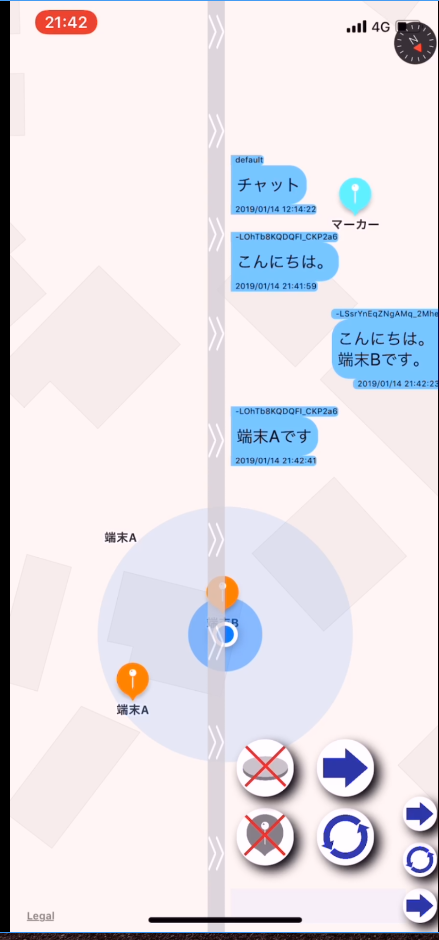
図16，17，18はFirebaseのRealtime DatabaseをWebブラウザ上で管理している画面である．各ノードにアプリケーションの機能によってアップロードされたデータが反映される様子が確認できた．

図19，20はクラウドストレージをWebブラウザ上で管理している画面である．アプリケーションの機能によってアップロードされた写真が反映される様子が確認できた．



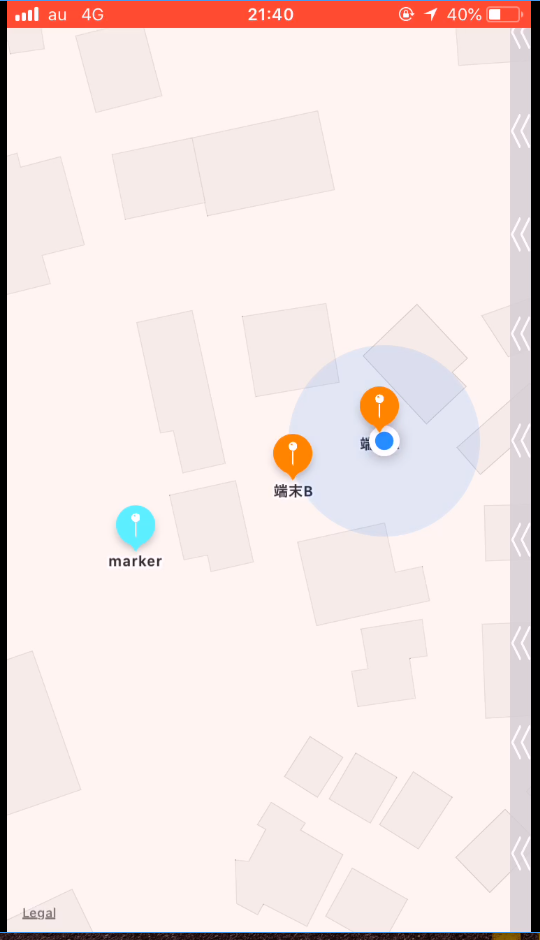
　　　　　端末Aの画面　　　　　　　　　　端末Bの画面

　　　図10　ユーザを示すマーカーをタップした時の画面



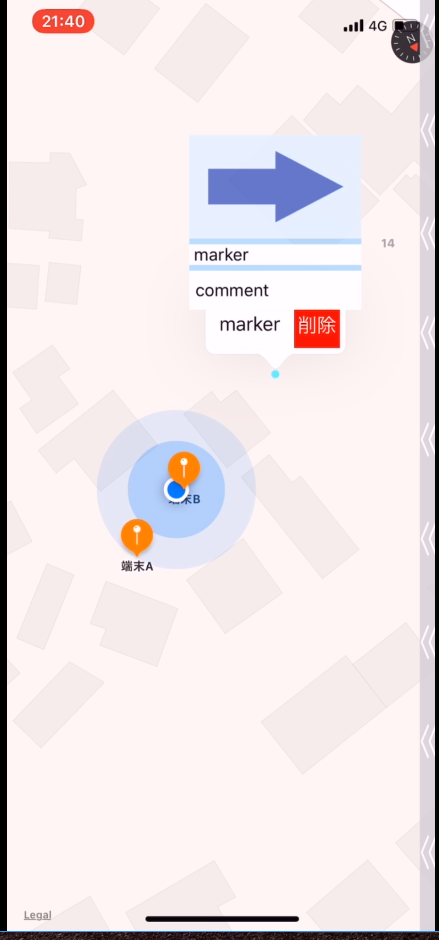
　　　　　　　端末Aの画面　　　　　　　　　　端末Bの画面

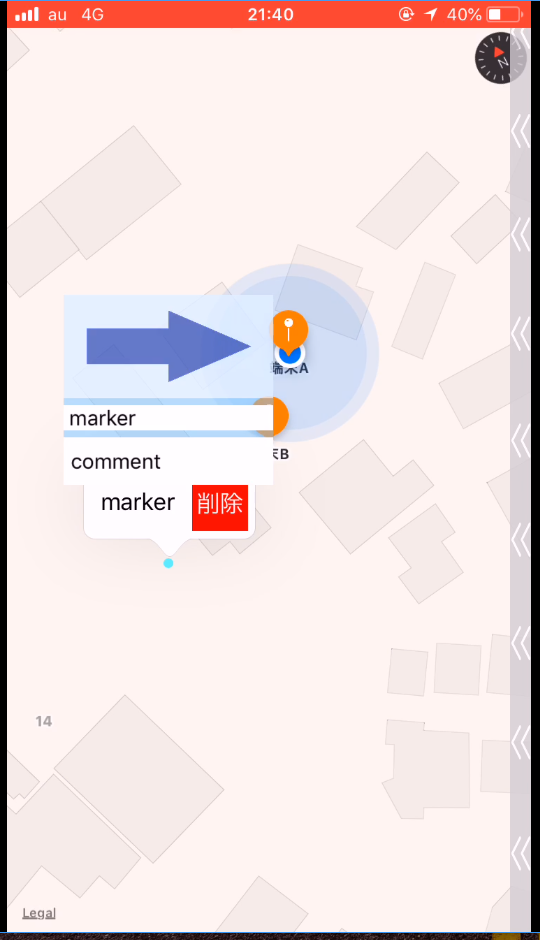
図11　チャット機能によるテキストと写真の投稿・共有



　　　　　　端末Aの画面　　　　　　　　　　　　　端末Bの画面

　　　 図12　地図をタップして，マーカーを設置した時の画面

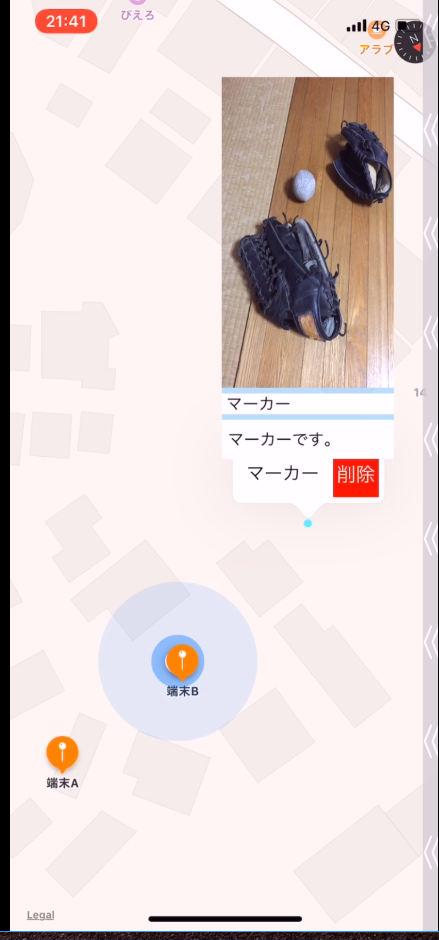




　　　端末Aの画面 　　　端末Bの画面

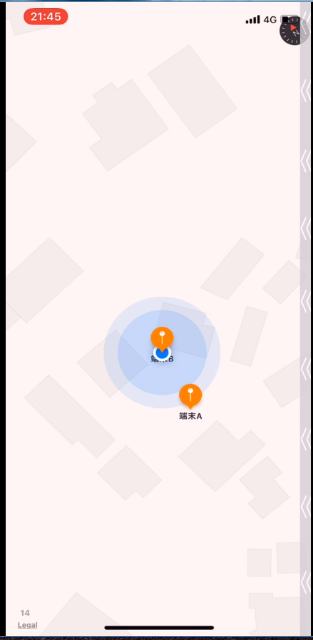
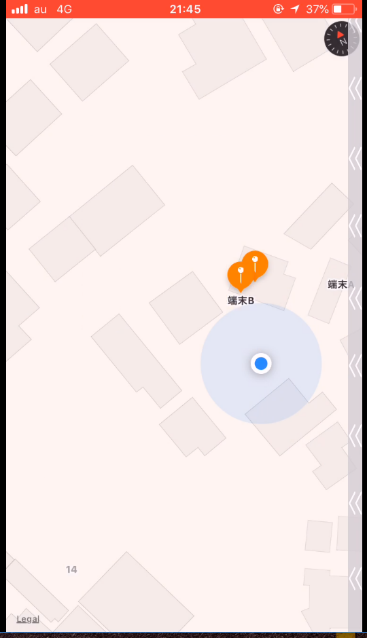
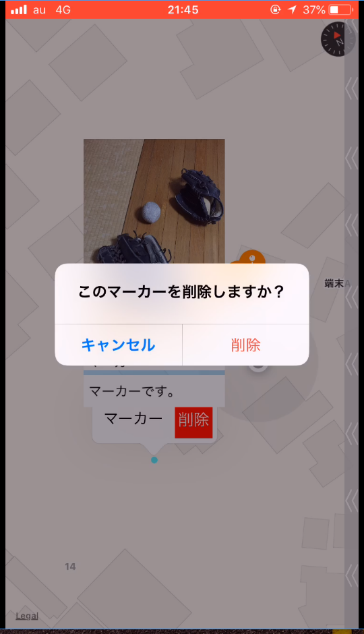
端末Aの画面　　　　　　　　　　　　　　　　端末Bの画面

　図13　設置したマーカーをタップして，名前・テキストを表示している画面



　　　　　端末Aの画面　　　　　　　　　　端末Bの画面

　　図14　名前・テキスト・写真が変更されたマーカーを表示している画像



　　　　　　　　　　端末Aの画面　　　　　　　　　　　　　端末Bの画面

　　　図15　マーカーをタップすると表示される削除ボタンで削除した画面

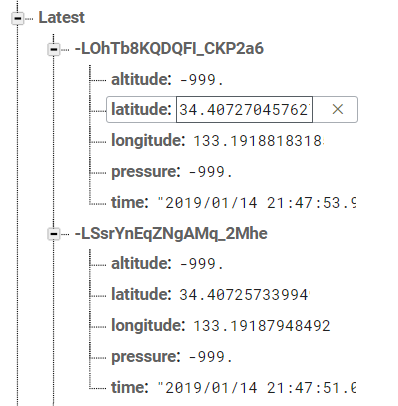


図16　Chatに投稿されたデータ　　　　図17　最新のユーザ情報



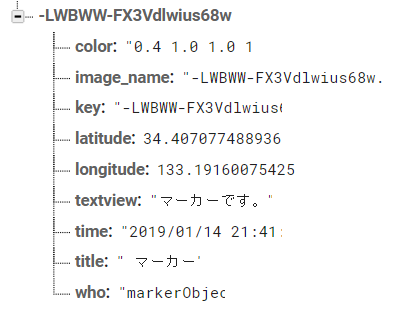
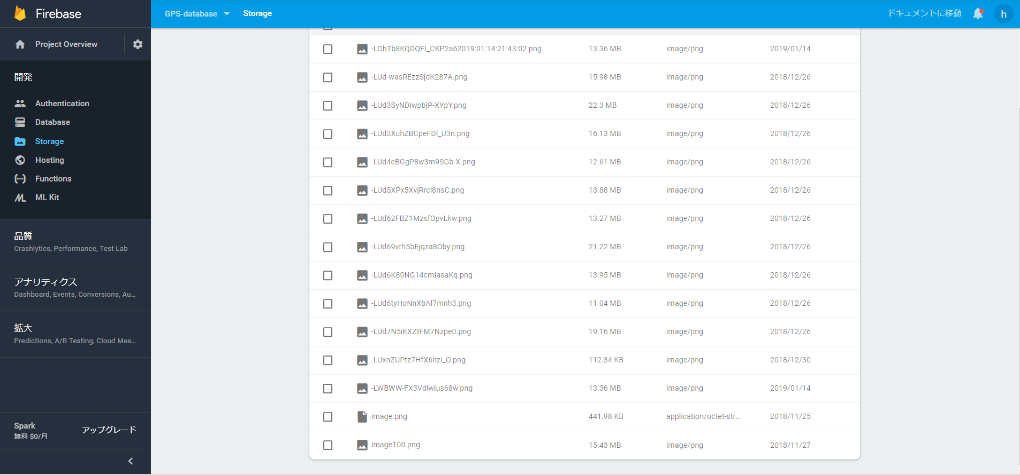


　　　　　図18　Marker内のマーカー情報が更新される様子



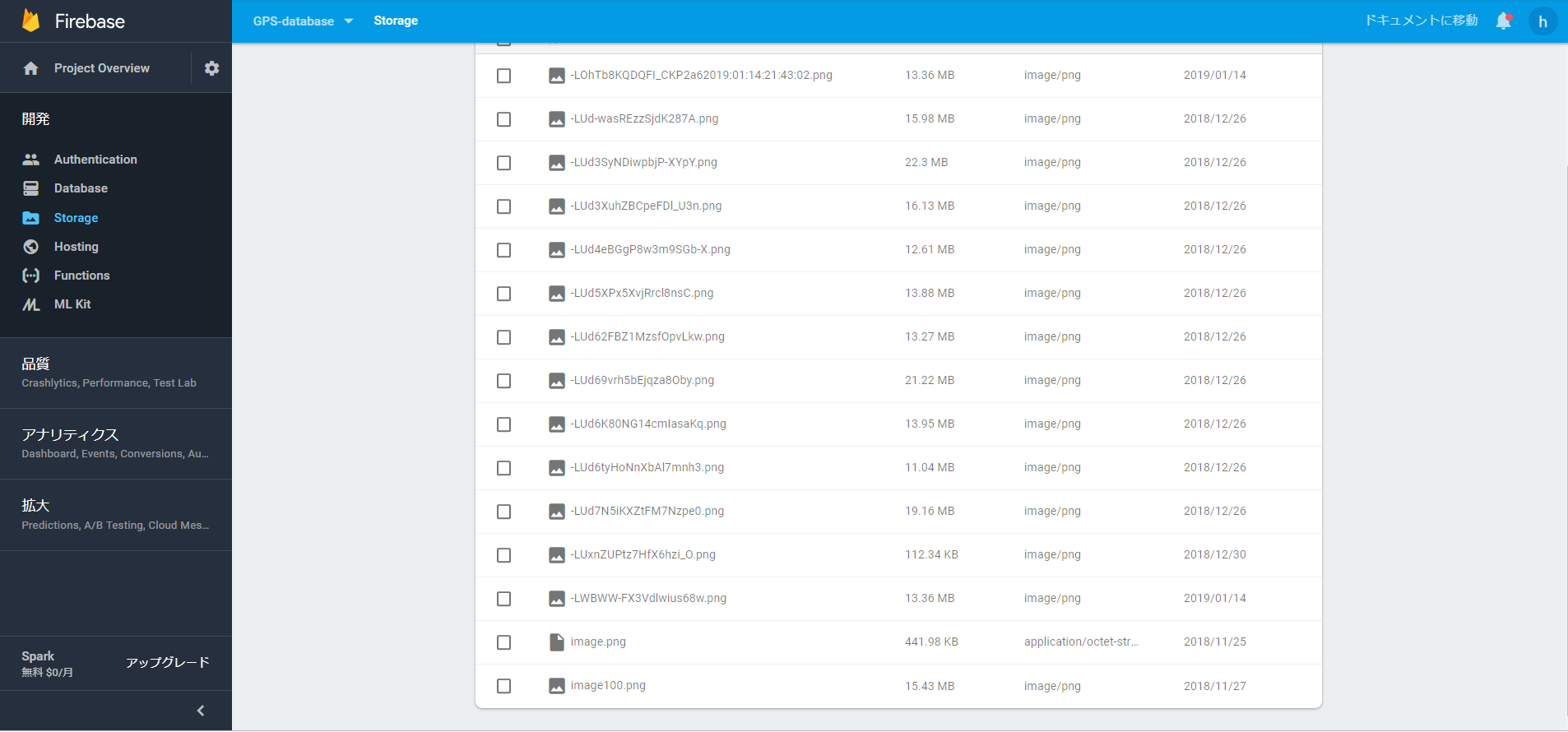


　　　　　図19　Storageの画面　　　　　　図20　マーカーの画像名と一致

## 大学構内の探索，調査への応用（概要）

福山大学構内の探索，調査を2人で実施した．このとき，既存の地図にマーカーを設置し，要所の名前・写真・紹介文の情報をマーカーに掲載する作業を行った．使用した機材は，iPhone6とiPhoneXの2端末である．1つずつ端末を持ち，別々に行動し，アプリケーションの機能を使用して地図を完成させた．目的は，実際に使う場面で意図した通りに機能するのかを確認することである．

## 大学構内の探索，調査への応用（結果）

図21のように要所にマーカーを設置し，名前・紹介文・写真を掲載する

ことができた．その情報を他の端末で確認することができた．また，事前に調査区域を分けていなかったが，お互いの位置や，追加したマーカーがリアルタイムで把握できたため，探索・調査に便利だった．このようにして，実際に地図を製作する際にも意図した通りに機能し，問題なく地図を製作することができた．



　　　　　　　端末A　　　　　　　　　　　　　　　　　　端末A



　　　　　　端末B　　　　　　　　　　　　　　　　　　　端末B

　　　　　　図21　大学構内に追加した案内マーカーの画像

# むすび

Firebaseを利用したデータベースの構築が容易であり，特に課題なくiOSアプリと連携し，位置情報共有システムである，グローバルな地図アプリケーションの製作，機能の実装ができた．チャットや位置情報の共有によるコミュニケーションと案内マーカーを使ったグローバルな地図の製作が機能的には行える．

今後の課題は，通信量を減らすために，チャットやマーカーなどで扱う画像ファイルのサイズを小さくすることである．画像ファイルをアップロードする際に，圧縮することで実現できると思われる．

また，セキュリティ面は研究していないので，データベースが安全とは言い切れない．位置情報共有システムが製作できたとはいえ，セキュリティを考慮すると世界に向けてアプリケーションをリリースするレベルには達していないので今後の課題としたい．

謝辞

本研究の実施にあたり，卒業論文指導教員の情報工学科・金子邦彦教授にご指導を賜りました．金子邦彦研究室の飯塚氏，井上氏，田坂氏には，実験の協力，研究室や実験の場での議論等を通して，知識や示唆の提供をいただきました．ここに感謝の意を表します．本研究は科研費（15H05708）の助成を受けたものである．

参考文献

1. 影澤秀明,廣井慧,奥矢淳,香取哲志,加藤朗,砂原秀樹：「Twitterを用いたセンシングシステムの提案と考察」，マルチメディア，分散，協調とモバイル（DICOMO2014）シンポジウム 平成26年7月.
2. Mingzhao Li,Zhifeng Bao,Farhana Choudhury,Timos Sellis:

“Supporting Large-scale Geographical Visualization in a Multi-granularity Way” WSDM '18 Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, pp.767-770, 2018.

1. Google Firebaseホームページ,

https://firebase.google.com/?hl=ja

1. Apple Developer Documentation

https://developer.apple.com/documentation

1. 浮田 弥,山本 大介,高橋 直久：「逆進検知機能を有する案内粒度変更可能な音声経路案内システム」，DEIM Forum，2017 I3-4

# 付録

　位置情報共有システムの，地図アプリケーションのプログラムは

1. AppDelegate.swift
2. ViewController.swift
3. SideMenu.swift

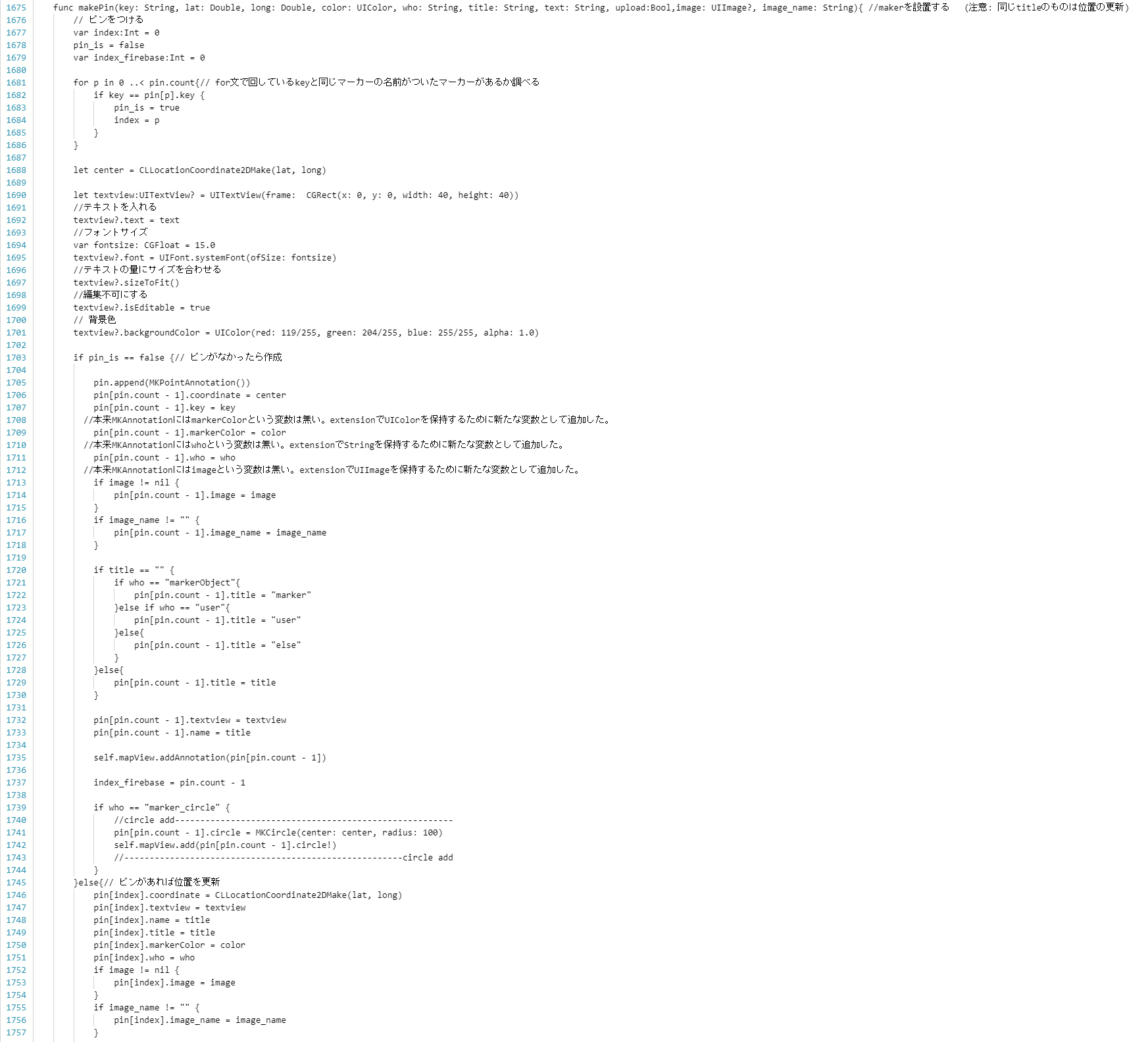
の3つのファイルに分かれている．

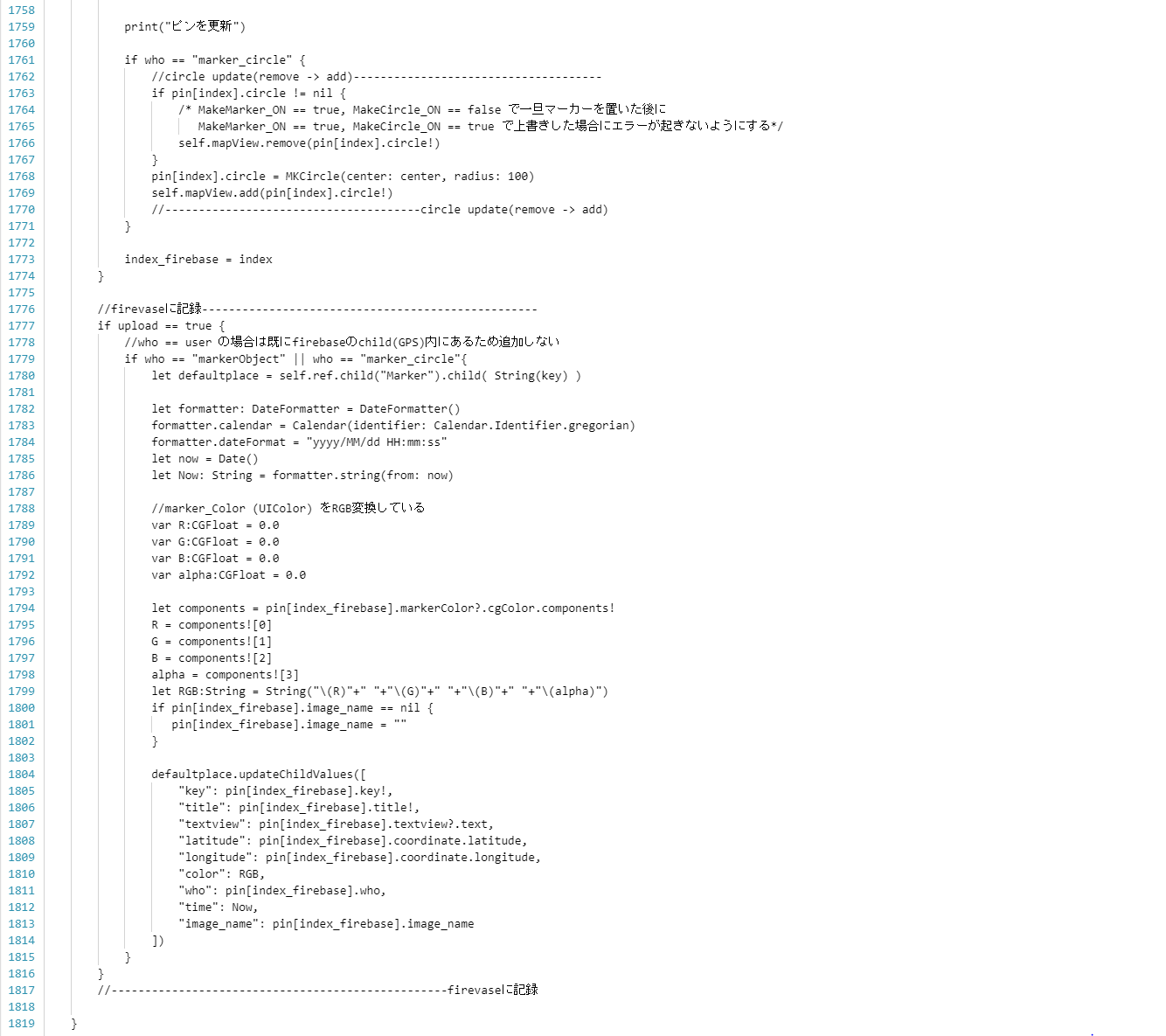
付録では，「2．」を載せる．

1. 節4.1の機能①②③の処理を実装したソースコード



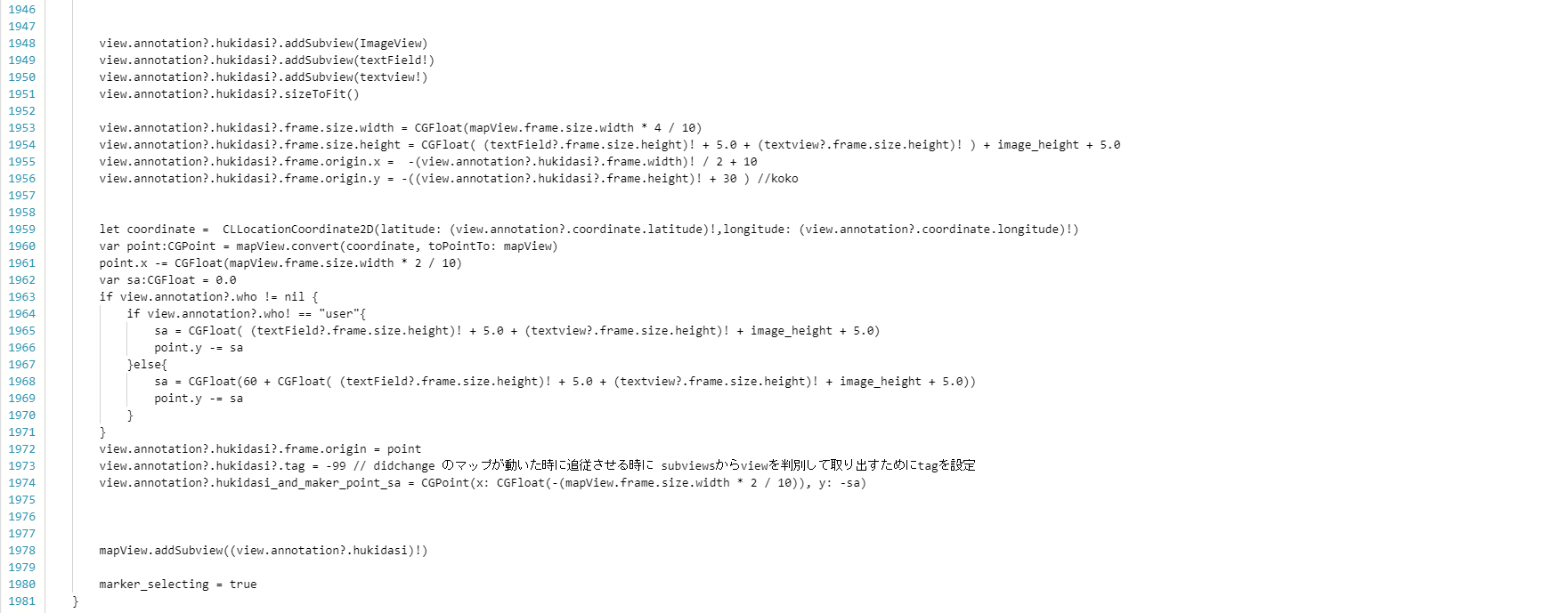






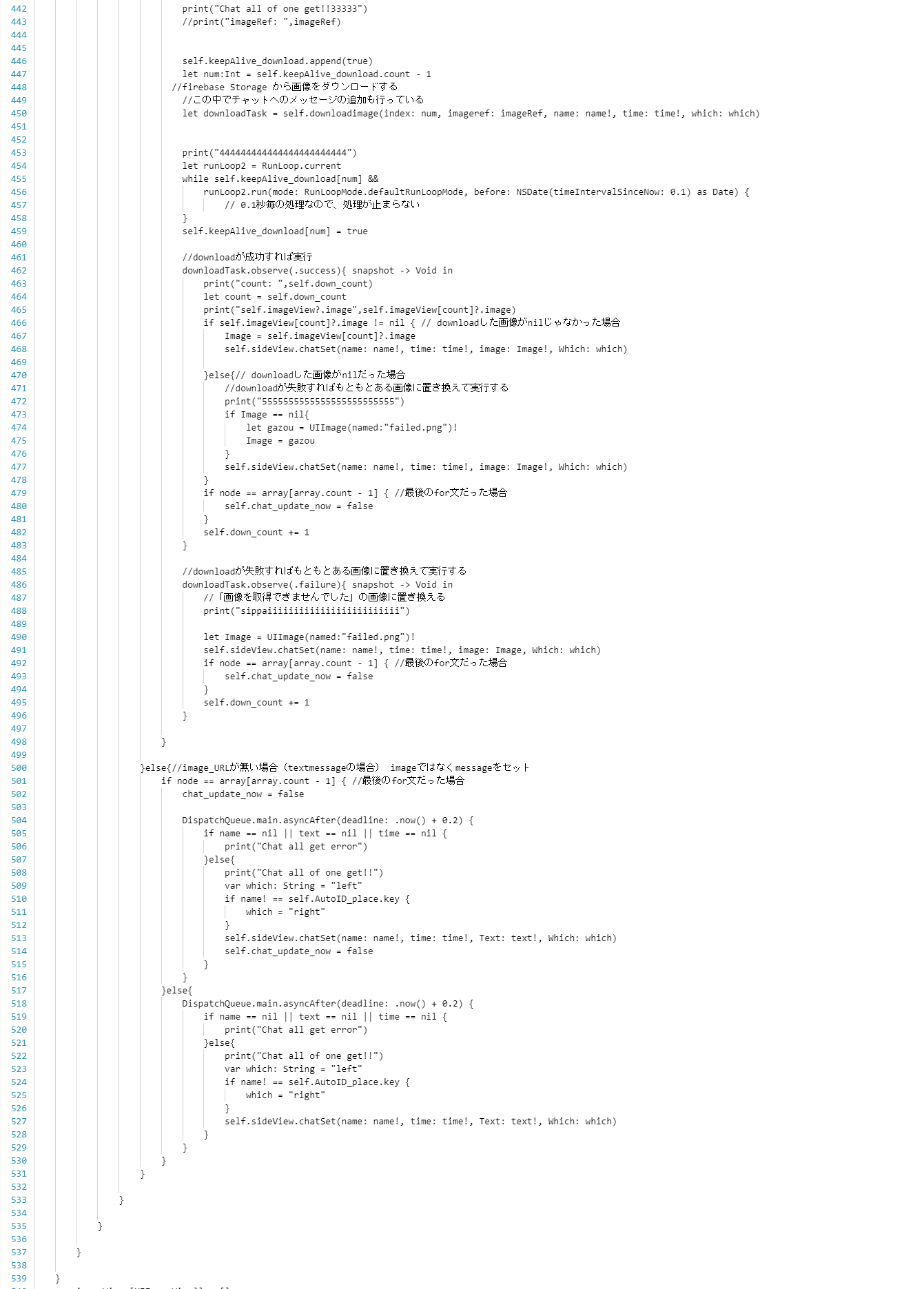
1. 節4.1の機能④⑧の処理を実装したソースコード



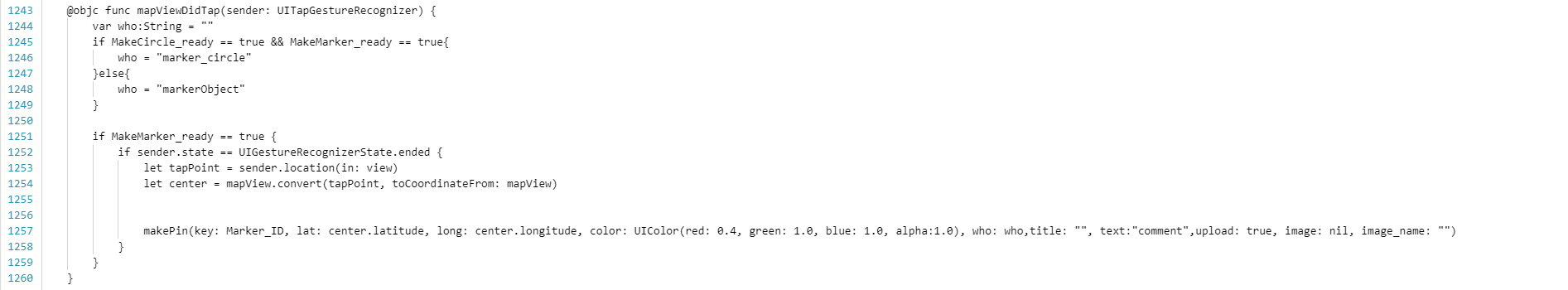


1. 節4.1の機能⑤⑥の処理を実装したソースコード





1. 節4.1の機能⑦の処理を実装したソースコード



1. 節4.1の機能⑨の処理を実装したソースコード

