

Android クイックスタート

2017/05/15 作成

このページでは、mobile backendをAndroidアプリを連携させる手順を紹介します

目次

- アプリの新規作成
- SDKのダウンロード
- SDKのインストール
- AndroidManifest.xml の編集
- SDKの読み込み
- APIキーの設定とSDKの初期化
- サンプルコードの実装
 - サンプルコード（データストア）
 - アプリを実行してmBaaSのダッシュボードを確認する

アプリの新規作成

mBaaS ダッシュボード

- ニフティクラウドmobile backendに[ログイン](#)します
- ダッシュボードが表示されたら、「アプリの新規作成」を行います
 - すでに別のアプリを作成済みの場合は、ヘッダーの「+新しいアプリ」をクリックします



アプリの新規作成

アプリ名

QuickStart

半角英数字もしくはアンダースコア

戻る 新規作成

- 「アプリ名」を入力し「新規作成」をクリックすると、APIキー（アプリケーションキーとクライアントキー）が発行されます



✓ 新しいアプリが作成されました

QuickStartアプリが作成されました。

SDKを利用して、mobile backendと連携したアプリを開発してみましょう！
詳しくはクイックスタートをご覧ください ([iOS](#) / [Android](#) / [javascript](#))

APIキー

アプリケーションキー * cdd5d587cc5f7eb1ac60b0713cc143aa7b2949bc58477f コピー

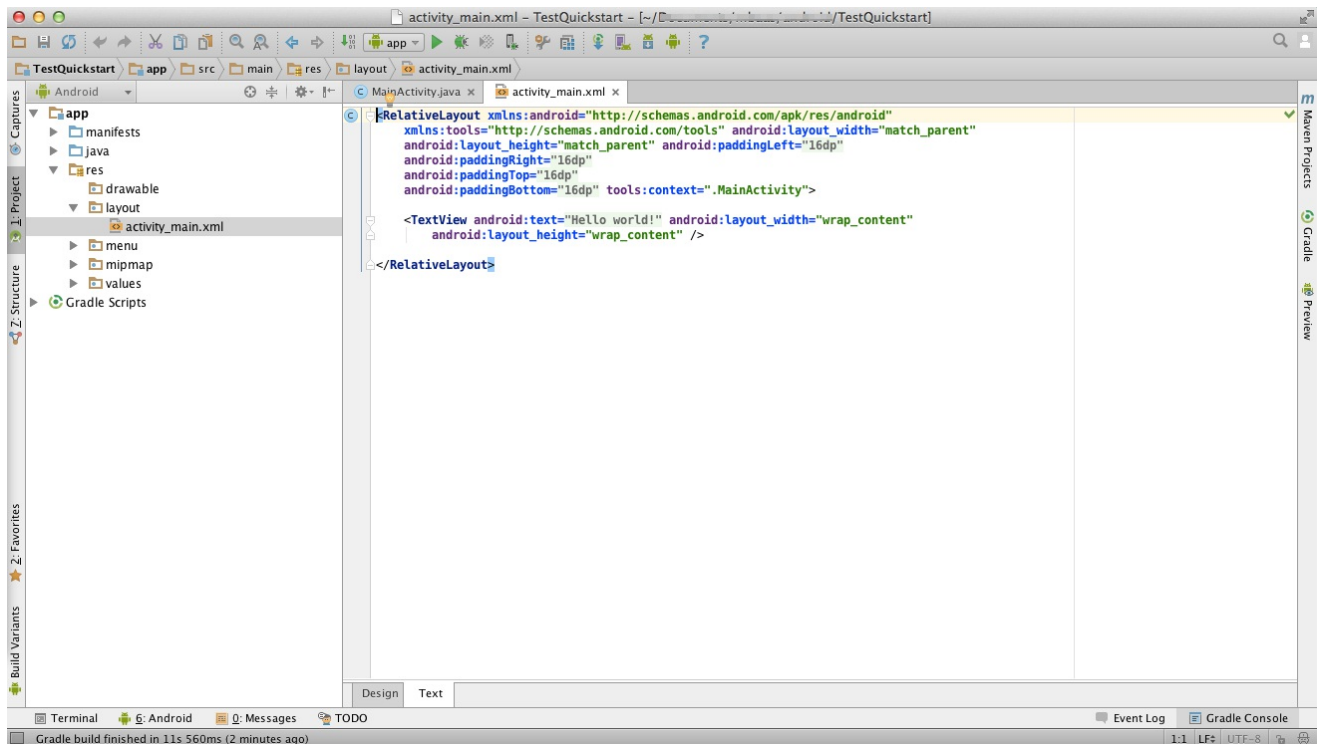
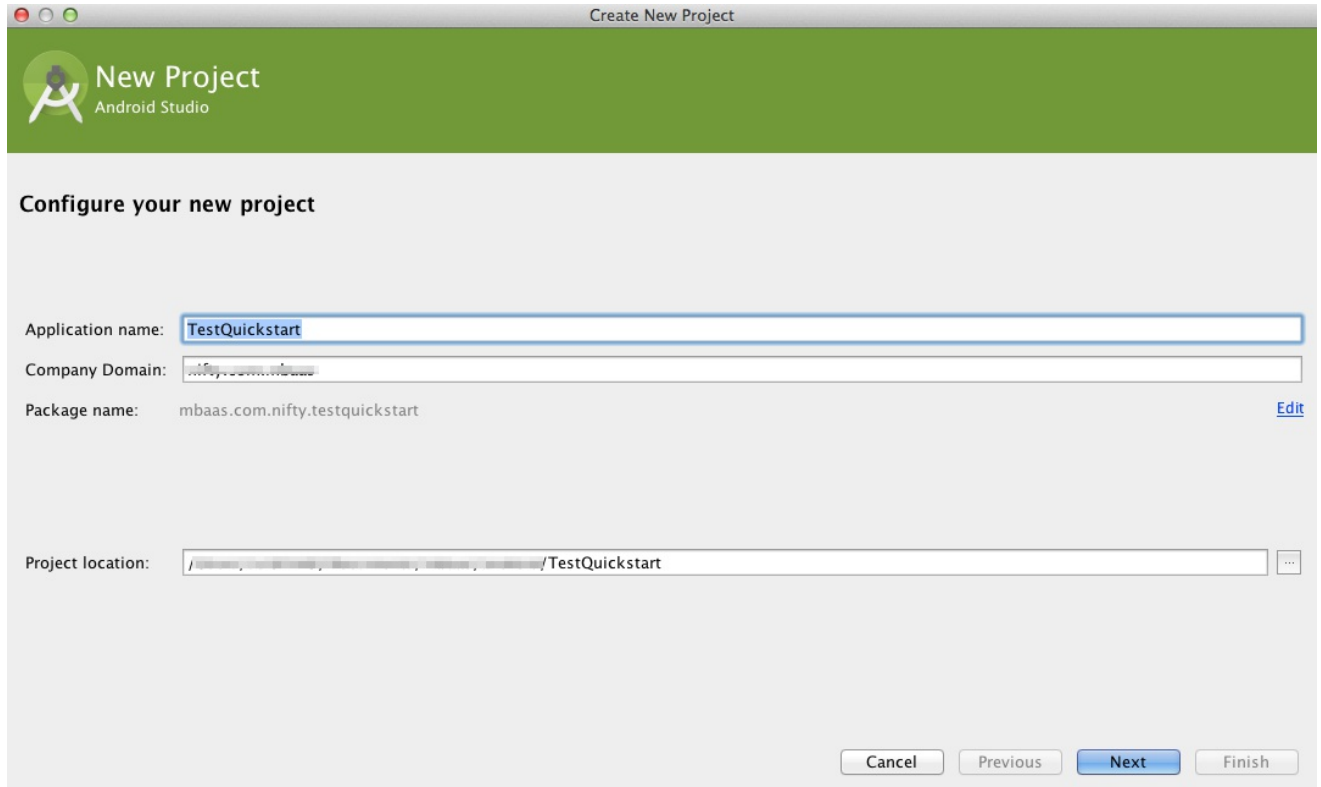
クライアントキー * 478fa200404c00fb19a064044ab6dacedf0e5a4577621e7c コピー

APIキーは[アプリ設定](#)からいつでも参照できます。

OK

- APIキーは後ほどAndroidアプリで使います

- AndroidStudioでプロジェクトを作成します
 - 既存のプロジェクトを利用する場合はこの作業は不要です



SDKのダウンロード

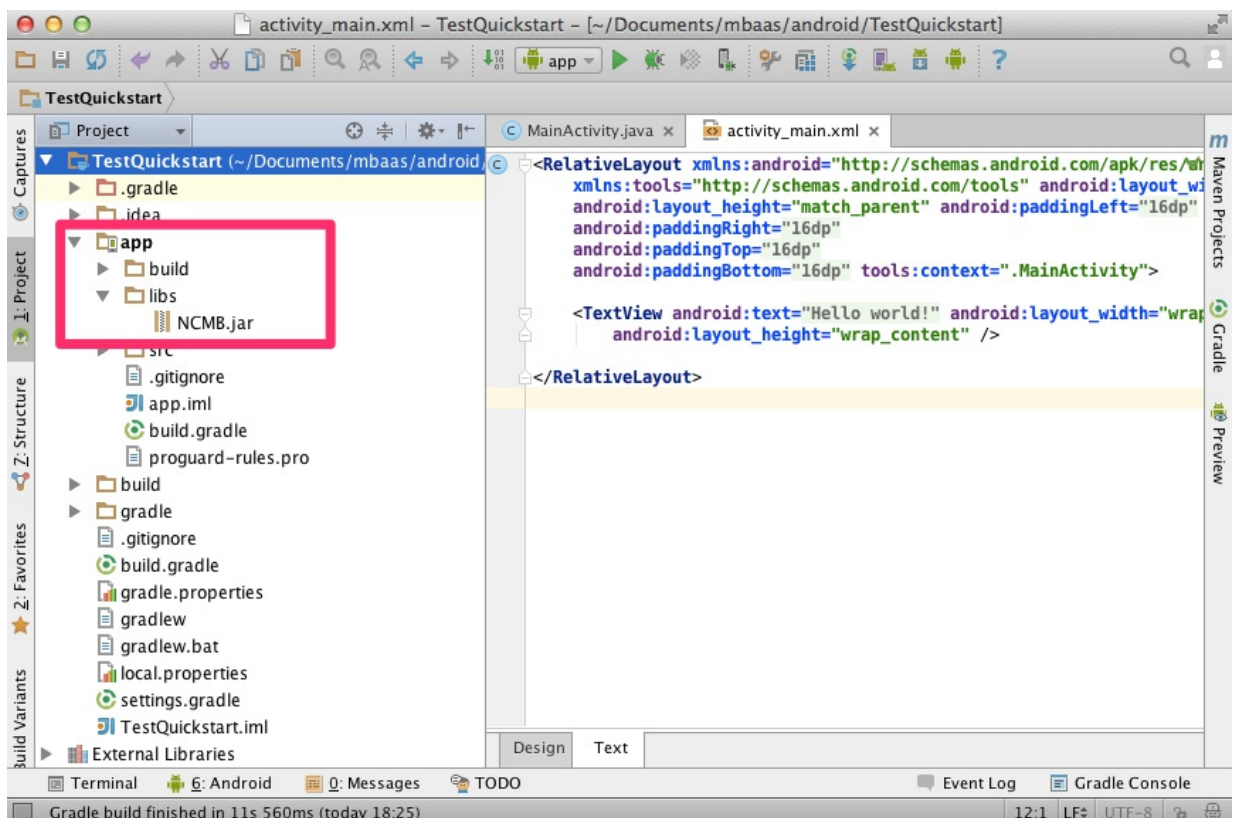
- [Githubリリースページ](#)の NCMB.x.x.x.zip ボタンからダウンロードしてください
 - 最新版をダウンロードしてください。
 - zipファイルの中身に、NCMB.jarがあります。

SDKのインストール

- このSDKでは、以下のライブラリを使用しています。
 - **Gson**

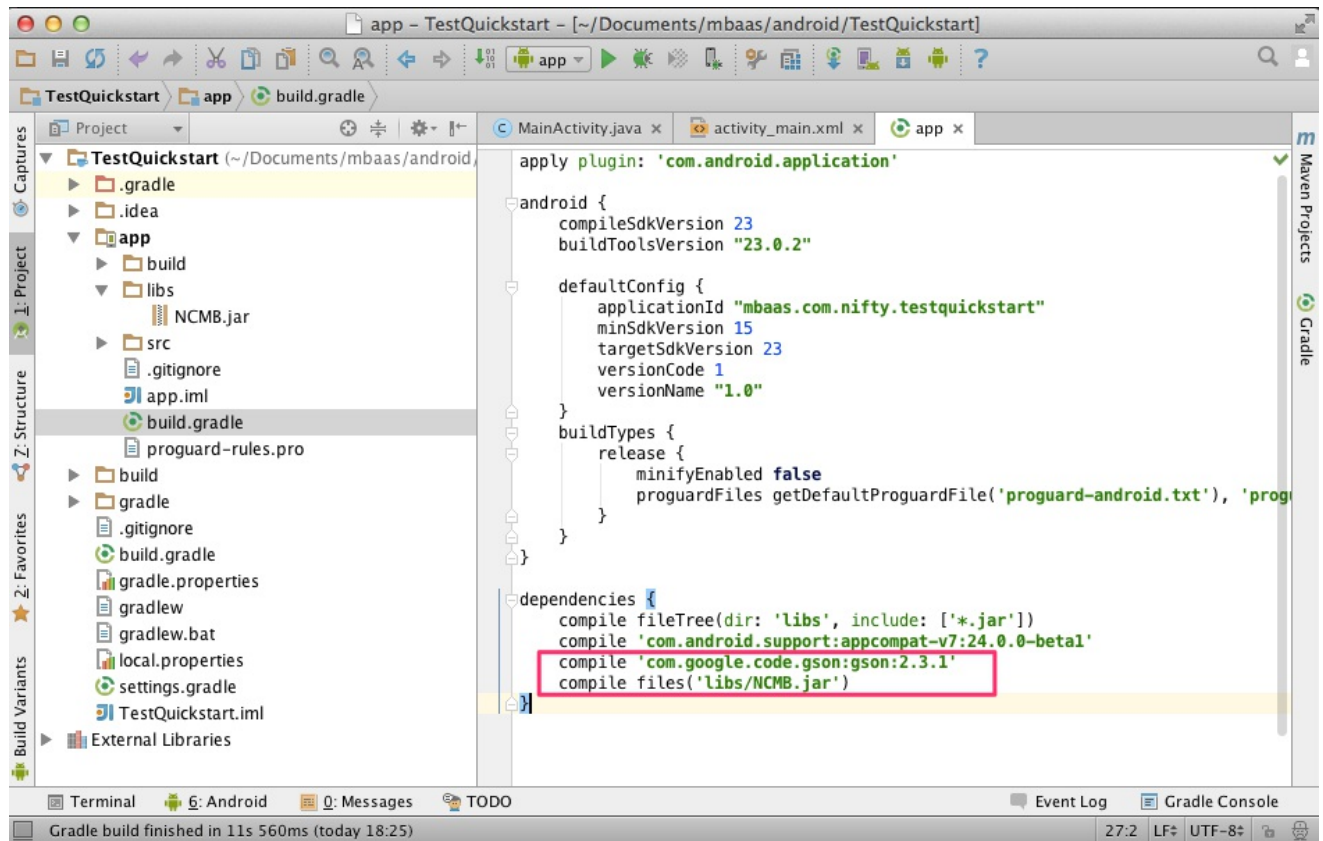
Android Studio

- Android Studioでプロジェクトを開き、以下の手順でSDKをインストールしてください
 - app/libsフォルダにNCMB.jar をコピーします



- app/build.gradleファイルに以下を追加します

```
dependencies {  
    compile 'com.google.code.gson:gson:2.3.1'  
    compile files('libs/NCMB.jar')  
}
```



AndroidManifest.xmlの編集

Android Studio

- <application>タグの直前に以下のpermissionを追加します

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
```



- プロセスが存在しない状態からサービスなどでアプリを起動しAPIリクエストを行うためには、<application> タグに以下を追加します。

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
```

SDKの読み込み

Android Studio

- MainActivity.javaの冒頭に次のコードを追記して、インストールしたSDKを読み込みます

```
import com.nifty.cloud.mb.core.NCMB;
```

APIキーの設定とSDKの初期化

Android Studio

- ニフティクラウド mobile backendのアプリケーションキーとクライアントキーを利用して、NCMBクラスのinitializeメソッドでAndroid SDKの初期化を行います
- アプリ起動時に表示するアクティビティのonCreateメソッドに下記の内容を追記します

```
NCMB.initialize(this.getApplicationContext(),"APP_KEY","CLIENT_KEY");
```

- 上の「`APP_KEY`」と「`CLIENT_KEY`」は、mBaaSのダッシュボードで「アプリの新規作成」を行ったときに発行されたAPIキーに置き換えます
 - アプリ作成時のAPIキー発行画面を閉じてしまった場合は、「アプリ設定」>「基本」で確認できます。
 - 「コピー」ボタンを使用してコピーしてください。



- これで連携作業は完了です！
- サンプルコードを書いて実際にmBaaSを使ってみましょう

サンプルコードの実装

Android Studio

- 最初に、ファイルの先頭に利用するライブラリを追記します

```
import com.nifty.cloud.mb.core.NCMB;  
import com.nifty.cloud.mb.core.NCMBException;  
import com.nifty.cloud.mb.core.NCMBObject;
```

- MainActivity.java の onCreate メソッド内に書いた処理は、アプリの起動時に実行されます
- APIキーの設定とSDK初期化コードの下にサンプルコードを書くと、すぐに動作確認が可能です

```
@Override  
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    // APIキーの設定とSDK初期化  
    NCMB.initialize(this.getApplicationContext(), "APP_KEY", "CLIENT_KEY");  
    // ↓ここにサンプルコードを実装↓  
  
    setContentView(R.layout.activity_main);  
}
```

サンプルコード（データストア）

- 次のコードはmBaaSのデータストアに保存先の「TestClass」というクラスを作成し、「message」というフィールドへ「Hello, NCMB!」というメッセージ（文字列）を保存するものです。

```
// クラスのNCMBObjectを作成
NCMBObject obj = new NCMBObject("TestClass");

// オブジェクトの値を設定
obj.put("message", "Hello, NCMB!");

// データストアへの登録
obj.saveInBackground(new DoneCallback() {
    @Override
    public void done(NCMBException e) {
        if(e != null){
            //保存に失敗した場合の処理

        }else {
            //保存に成功した場合の処理
        }
    }
});
```

アプリを実行してmBaaSのダッシュボードを確認する

- アプリを実機またはシミュレーターで実行します

mBaaS ダッシュボード

- アプリが起動されたら、mBaaSのダッシュボードで「データストア」から、データが保存されていることを確認できます



The screenshot shows the mBaaS mobile backend dashboard. The top navigation bar includes links for 'アプリ一覧' (App List), 'ダッシュボード' (Dashboard), 'ドキュメント' (Documentation), '開発TIPS' (Development Tips), 'コミュニティ' (Community), and '連携サービス' (Integration Services). The main header has a '+新しいアプリ' (New App) button and a 'test' dropdown menu. The left sidebar contains navigation options: 'メニューを開く' (Open Menu), 'ダッシュボード' (Dashboard), '会員管理' (Member Management), 'データストア' (Data Store), 'ファイルストア' (File Store), 'スクリプト' (Scripts), and 'プッシュ通知' (Push Notifications). The main content area is titled 'クラス' (Class) and shows a list of classes: 'installation', 'role', and 'TestClass'. The 'TestClass' is selected, and its details are shown on the right. The 'TestClass' details include a '+新しいレコード' (New Record) button, a '+新しいフィールド' (New Field) button, and a '削除' (Delete) button. Below these buttons is a table with columns: 'objectId', 'message', 'createDate', and 'updateDate'. The table contains one record with the following data:

objectId	message	createDate	updateDate
ycY4U9ZlozV5eIjQ	Hello, NCMB!	2017-01-10T18:10:18.109+09:00	2017-01-10T18:11:18.109+09:00