

# Unity クイックスタート

---

このページでは、mobile backendをUnityアプリと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をインストールする
- 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
- 注意事項

# 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

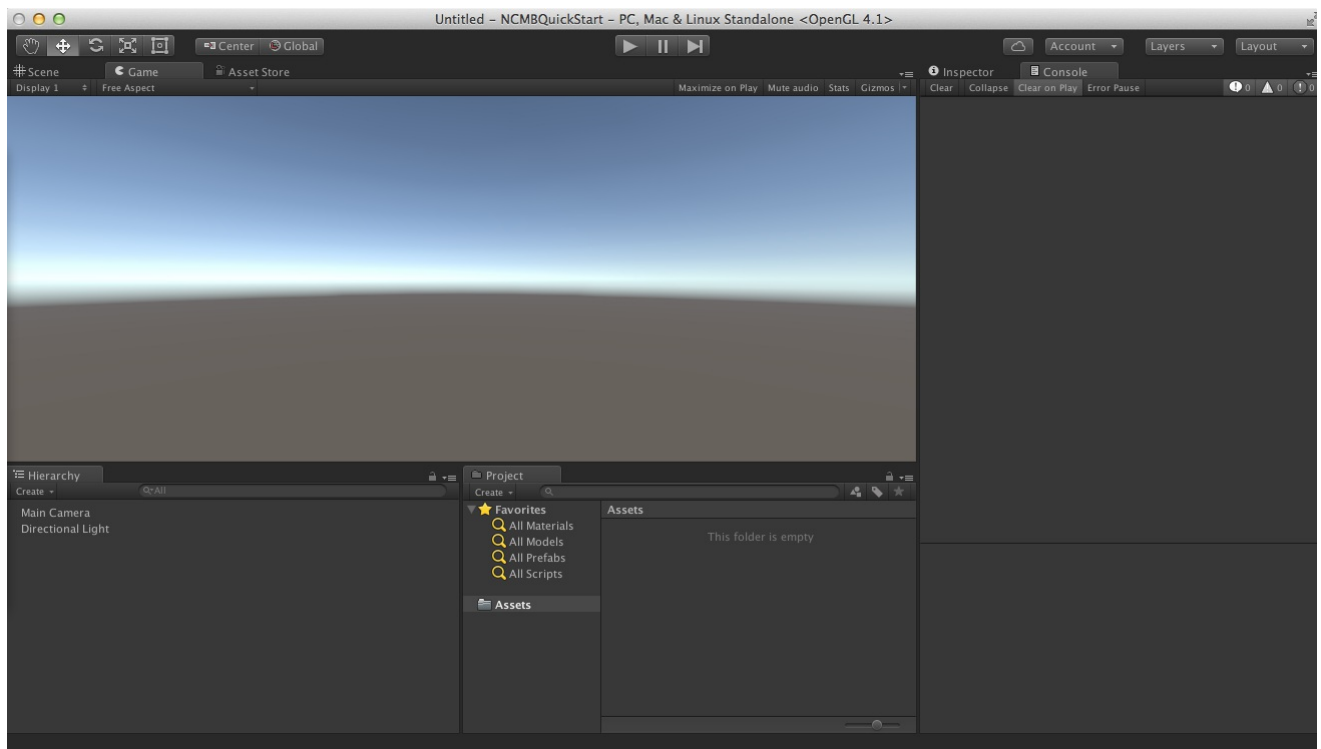
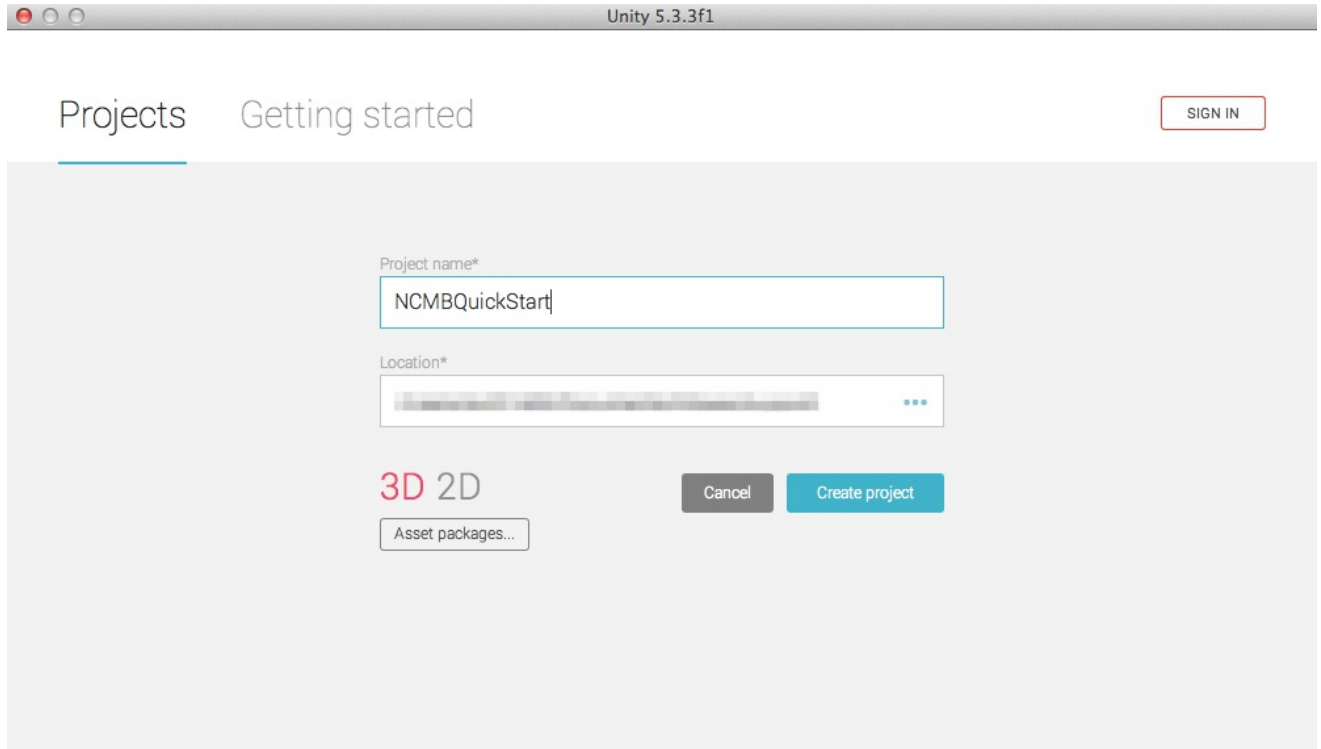
クライアントキー \* 478fa200404c00fb19a064044ab6dacedf0e5a4577621e7c コピー

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Unityアプリで使います

- Unity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

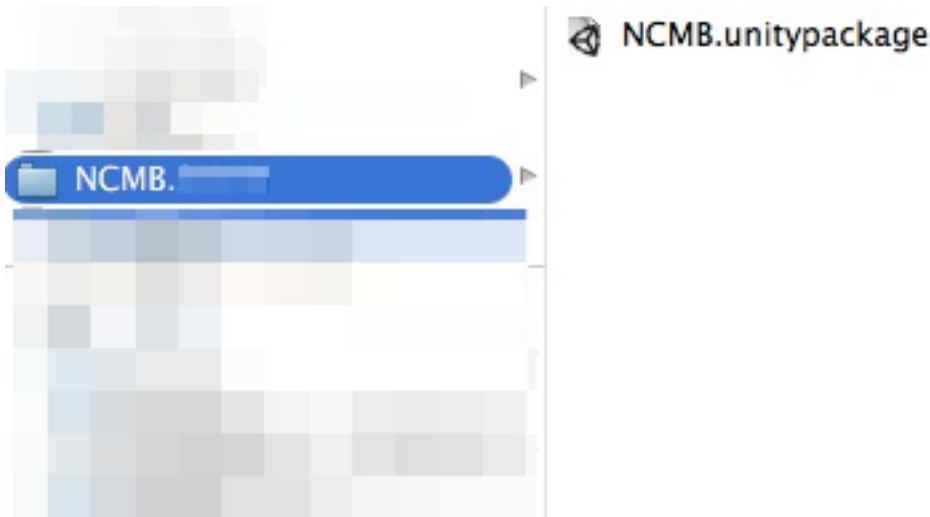


# SDKのダウンロード

---

- 以下のリンクからGithubのリリースページを開き、NCMB.2.x.x.zip(xはバージョン番号)をダウンロードしてください

## Githubのリリースページ

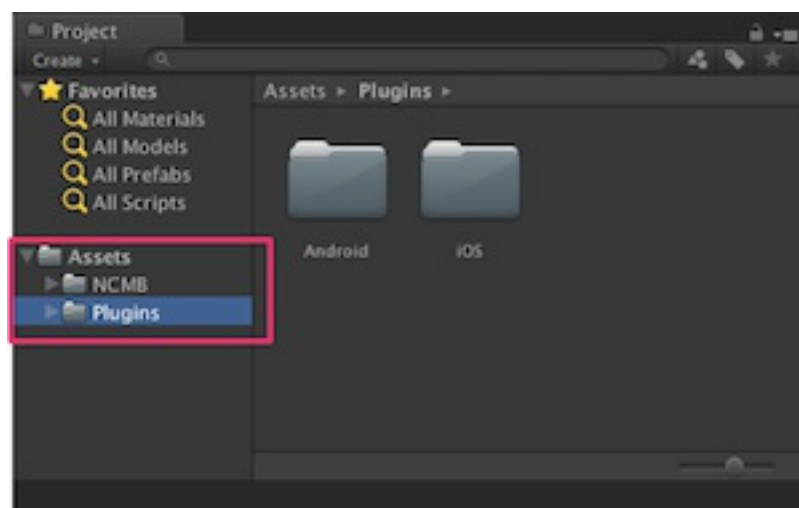
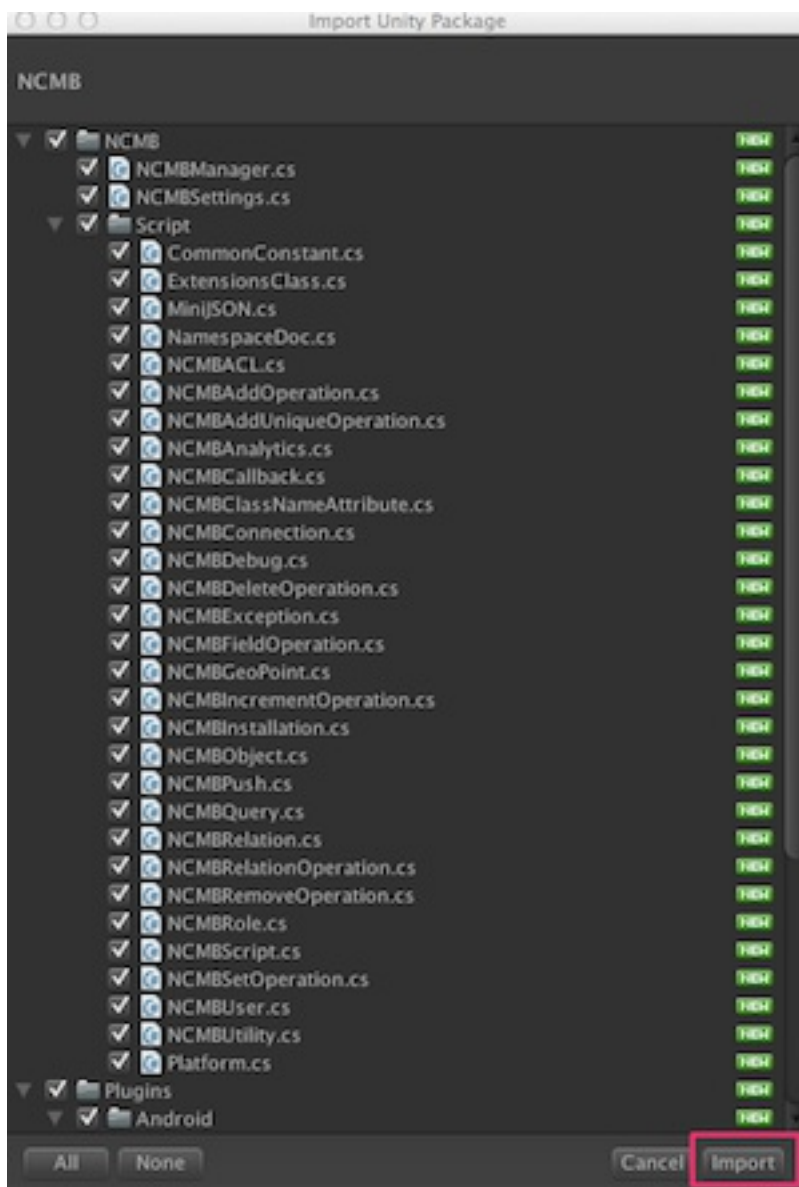


# SDKをインストールする

---

## Unity

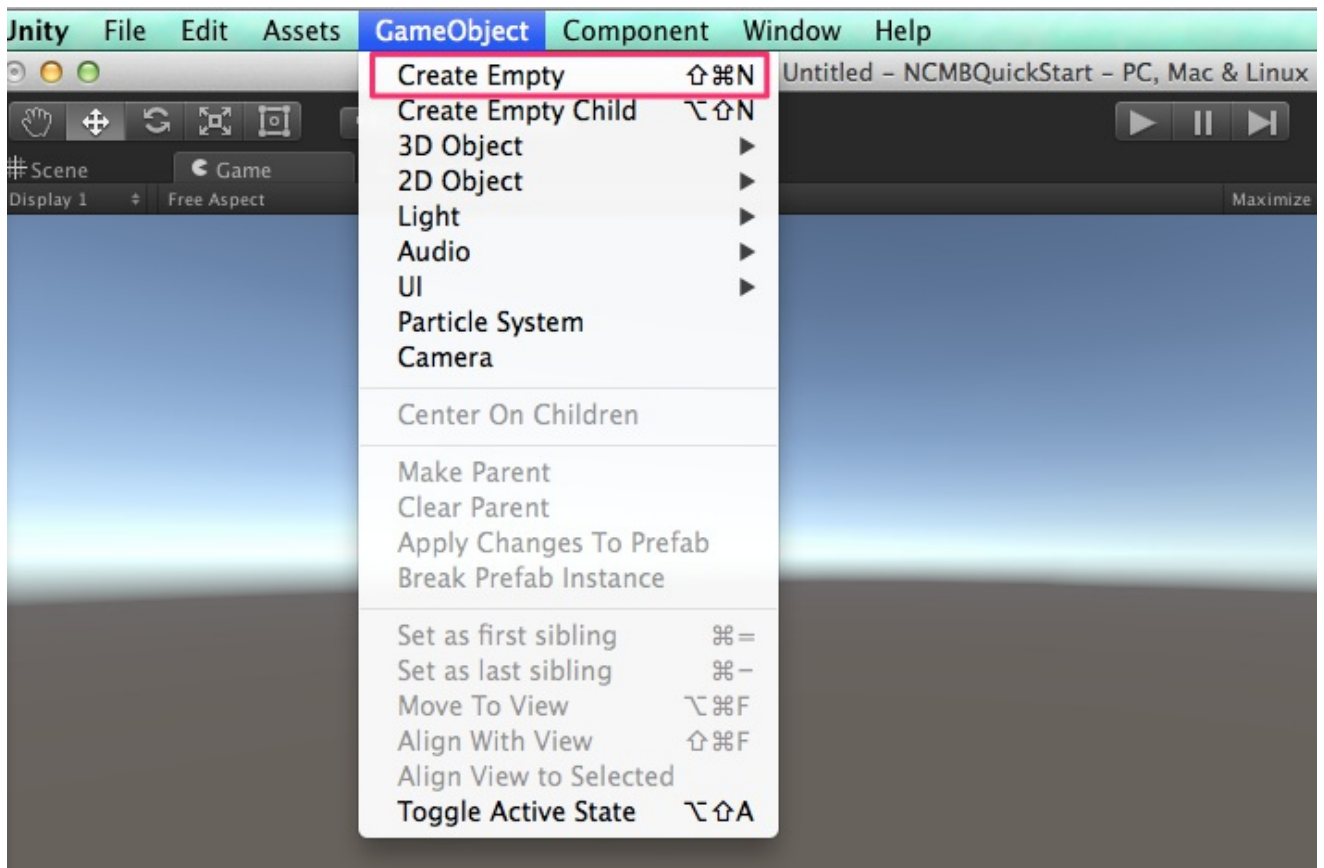
- 先ほどダウンロードしたzipファイルを解凍します
- フォルダ内にある「NCMB.unitypackage」をダブルクリックして、インポートしてください



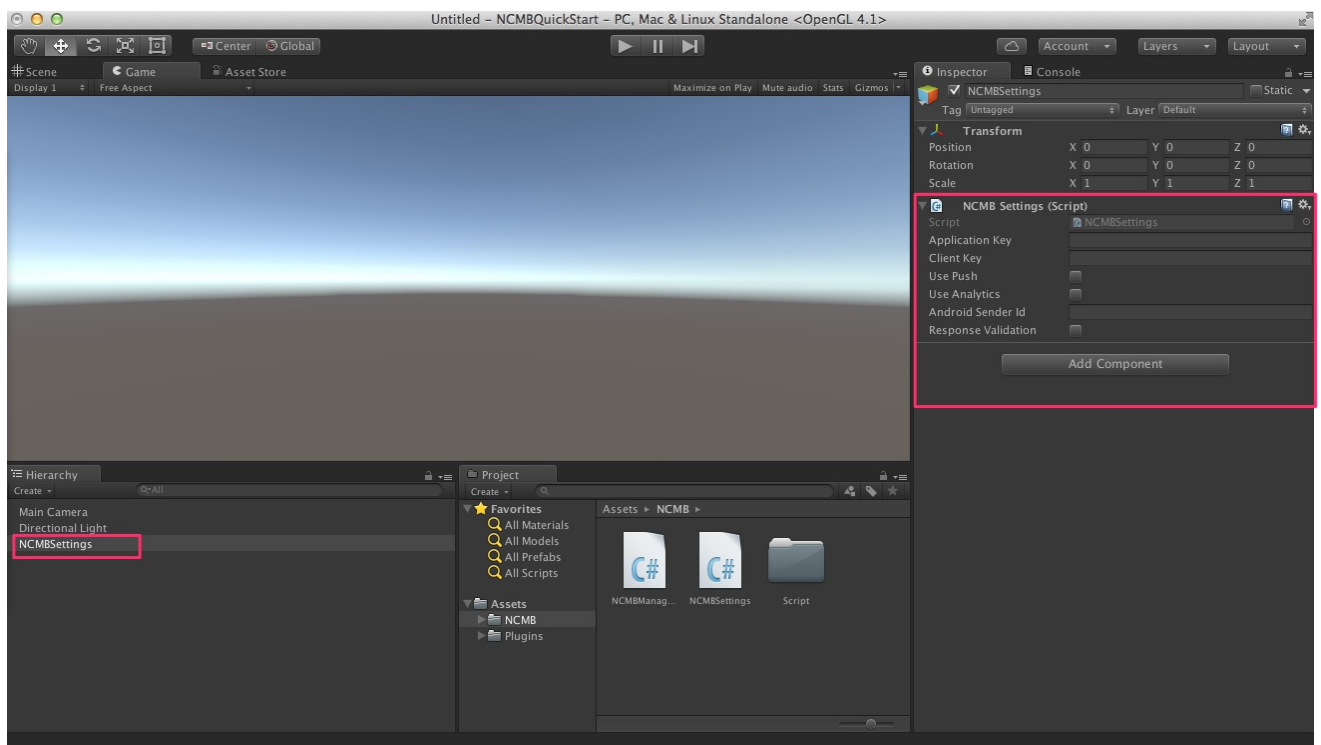
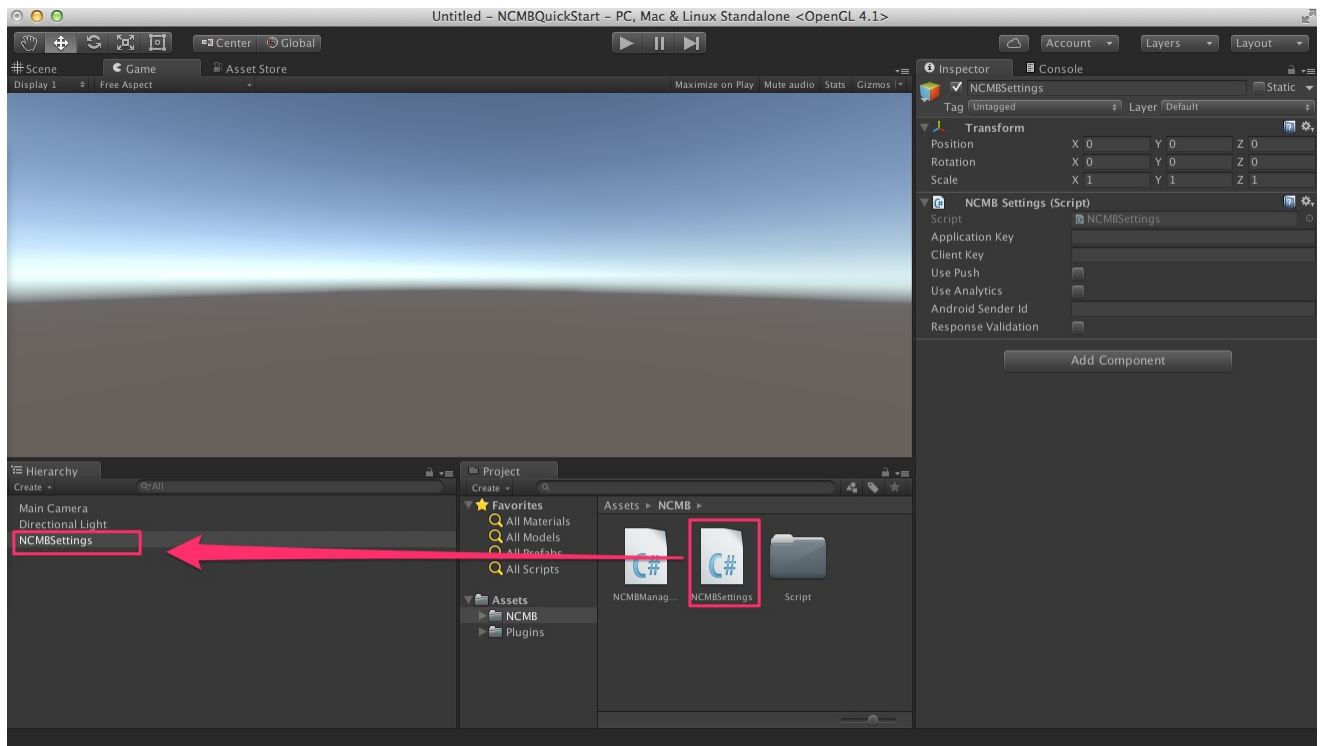
# SDKの読み込み

## Unity

- 空のGame Objectを作成します
  - わかりやすいように、作成したGame Objectをここでは「NCMBSettings」という名前に変更します。



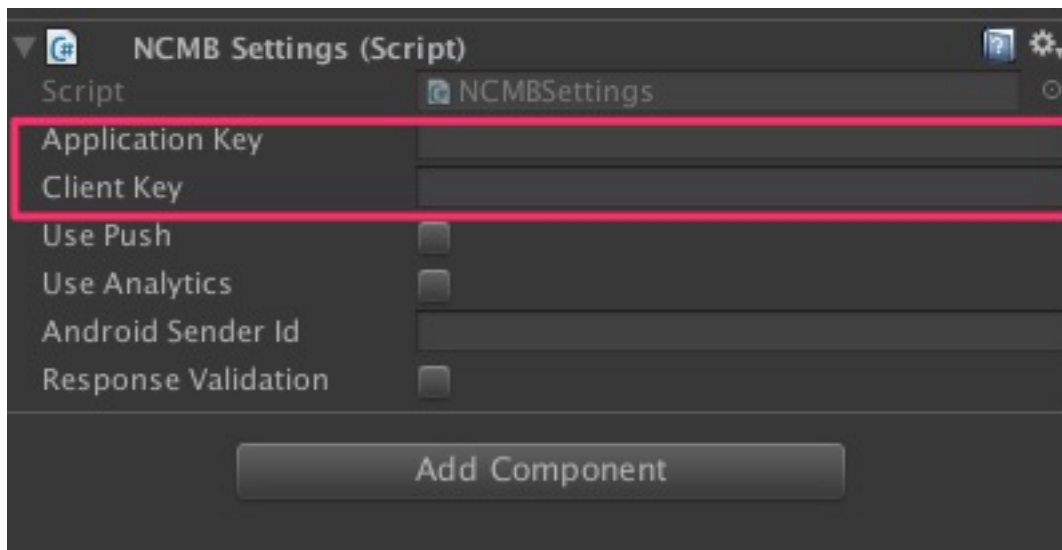
- インポートした「NCMB」のフォルダ内にある「NCMBSettings.cs」を、先ほど作成した「NCMBSettings」にドラッグ&ドロップでアタッチします



# APIキーの設定とSDKの初期化

## Unity

- コードを書いていく前に、必ずmBaaSで発行されたAPIキーの設定を行う必要があります
- アタッチした状態でヒエラルキー（Hierarchy）の「NCMBSettings」をクリックすると、インスペクタ（Inspector）に図のようなアプリケーションキーとクライアントキーの入力欄が表示されます



- プッシュ通知機能を使用する場合、Use Pushにチェックを入れてください
  - Android向けにプッシュ通知を行う場合、さらにAndroid Sender IDを設定します。
  - 開封通知を行う場合、Use Analyticsにチェックを入れてください。
  - 詳しくは[プッシュ通知の基本的な使い方](#)をご覧ください。
- レスポンスシグネチャの検証を行う場合はResponse Validationにチェックを入れてください



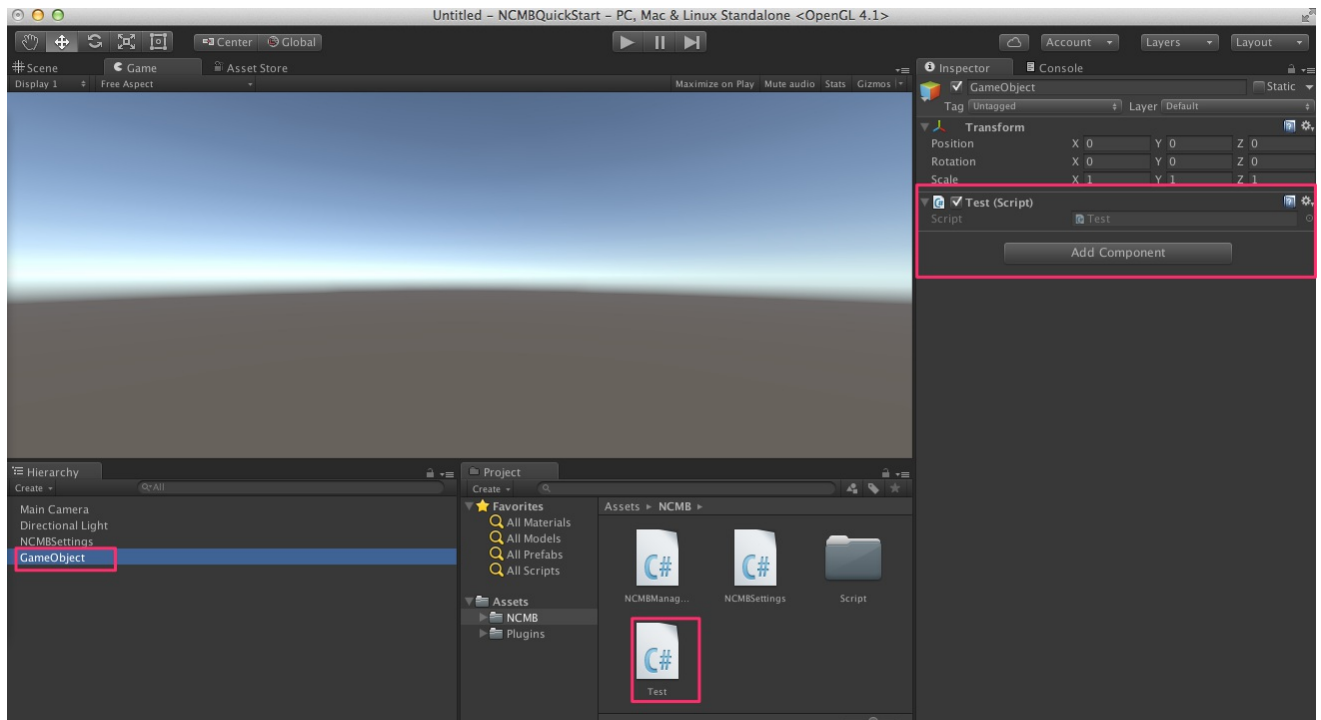
- 上のアプリケーションキーとクライアントキーは、mBaaSのダッシュボードで「アプリの新規作成」を行ったときに発行されたAPIキーに置き換えます
  - アプリ作成時のAPIキー発行画面を閉じてしまった場合は、「アプリ設定」>「基本」で確認できます。
  - 「コピー」ボタンを使用してコピーしてください。



- 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

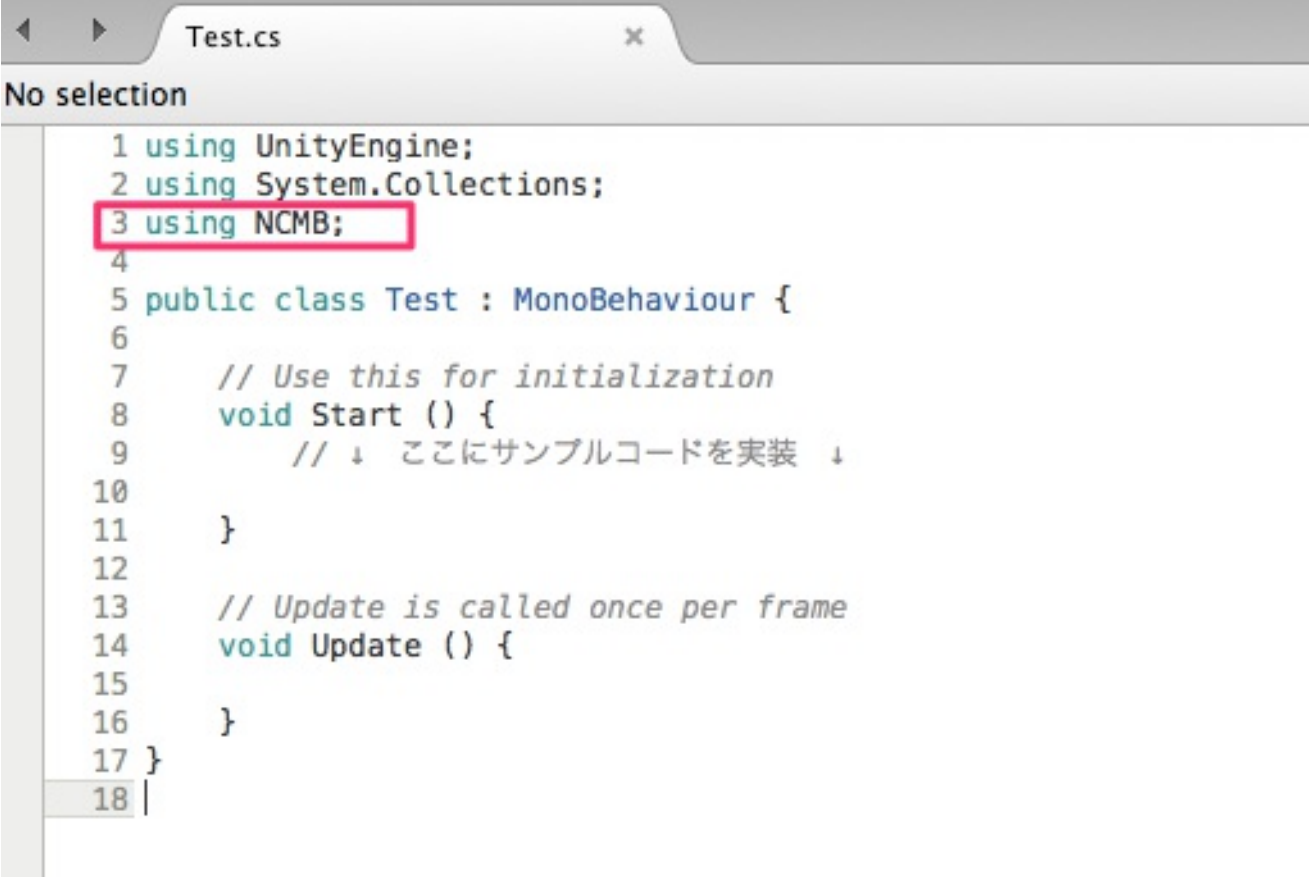
# サンプルコードの実装

- ここまでの作業を行えば、プロジェクト中のどのシーン、どのスクリプトでもmobile backendの機能を使用することができます
- 新しく「空のGame Object」と「C#スクリプト」を作成し、そのスクリプトを空のGame Objectにアタッチします



- スクリプトを開き、最上部に以下の内容を追記します

```
using NCMB;
```



```
Test.cs
No selection

1 using UnityEngine;
2 using System.Collections;
3 using NCMB;
4
5 public class Test : MonoBehaviour {
6
7     // Use this for initialization
8     void Start () {
9         // ↓ ここにサンプルコードを実装 ↓
10
11     }
12
13     // Update is called once per frame
14     void Update () {
15
16     }
17 }
18 |
```

- Startメソッド内に書いた処理は、GameObjectの起動時に実行されます
- Startメソッドの中にサンプルコードを書くと、すぐに動作確認が可能です

```
// Use this for initialization
void Start () {
    // ↓ ここにサンプルコードを実装 ↓
}
```

# サンプルコード（データストア）

## Unity

- 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 testClass = new NCMBObject("TestClass");

// オブジェクトに値を設定

testClass["message"] = "Hello, NCMB!";
// データストアへの登録
testClass.SaveAsync();
```

# 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ュミ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



## 注意事項

WindowsでMonoDevelopを使用しビルドする際、「.NET 4.0」以上を選択する必要があります。設定方法： MonoDevelop の[Project] -> [Assembly-CSharp options] -> [Build/General] -> Target Framework で Mono/.NET 4.0を選択します。