

# Monacaとクラウドデータベースを 連携してみよう！

富士通クラウドテクノロジーズ株式会社



NIFTY Cloud  
ニフティ クラウド

Monacaで作成したアプリとデータストアを通して、  
クラウドデータベースへの保存・検索の方法を学んでいきます

## <使用するアプリについて>

- ・問い合わせ内容の送信
- ・問い合わせの一覧表示
- ・問い合わせの検索

が出来るお問い合わせアプリです



ニフティクラウド mobile backendを使ってMonacaと連携させると・・・

- ・問い合わせ内容の送信／一覧表示／検索が**簡単に実装**できます
- ・動作確認が**素早く手軽**にできます



.....  
連携



## ニフティクラウドmobile backendとは・・・？

**mobile Backend as a Service = mBaaS**

スマートフォンアプリのバックエンド機能が

**開発不要**となるクラウドサービス

**基本無料**  
で使えます！

※詳しくは[こちら](#)



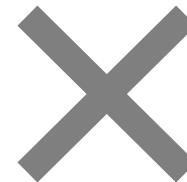
## mBaaSのここがすごい！

- ・**基本無料**で使える
- ・バックエンド機能は**開発不要**
- ・サーバーも**運用不要**
- ・クラウド上の機能を**APIを呼び出すだけ**で使用可能
- ・APIを使うための**充実したSDK**を提供  
⇒ iOS／Android／Monaca／Unity

→ 運用負担大幅削減！  
→ 開発スピードUP！



mBaaSとMonacaの組み合わせで、  
複雑な機能も**簡単かつスピーディーに実現**できます！



# <Demo1>データの保存：完成イメージ①

- ✓ 各項目に入力し送信ボタンを押すと、入力したデータがmBaaSのデータストアに保存されます

**Monaca**



Demo1 : フォーム

お名前 (必須)

メールアドレス (必須)

お住まい ご年齢

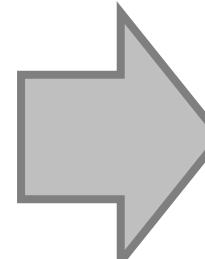
--選択-- --選択--

タイトル (必須)

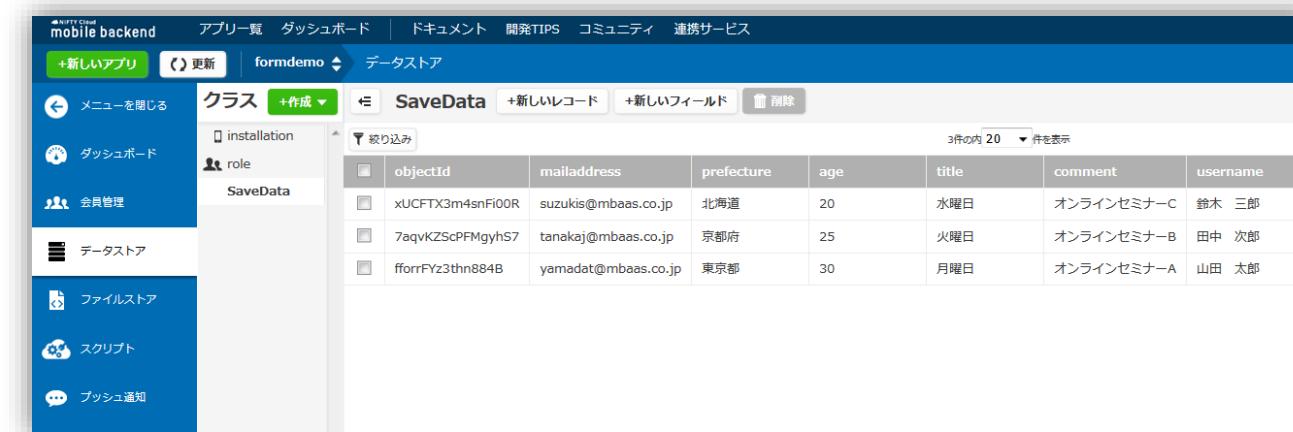
お問い合わせ内容 (必須)

送信

Demo1 Demo2 Demo3



**mBaaS**



mobile backend

+新しいアプリ

アプリ一覧 ダッシュボード ドキュメント 開発TIPS コミュニティ 連携サービス

更新

formdemo データストア

クラス +作成

installation role SaveData

データストア

ファイルストア スクリプト プッシュ通知

objectID	mailaddress	prefecture	age	title	comment	username
xUCFTX3m4snFi00R	suzukis@mbaas.co.jp	北海道	20	水曜日	オンラインセミナーC	鈴木 三郎
7aqvKZScPFMgyhS7	tanakaj@mbaas.co.jp	京都府	25	火曜日	オンラインセミナーB	田中 次郎
fforrFYz3thn884B	yamadat@mbaas.co.jp	東京都	30	月曜日	オンラインセミナーA	山田 太郎

# <Demo1>データの保存：完成イメージ②

- ✓ お名前、メールアドレス等の各データはデータストアにフィールドごとに保存されます

The screenshot shows the NIFTY Cloud mobile backend application interface. On the left, there's a sidebar with various navigation options like 'mobile backend', 'アプリ一覧', 'ダッシュボード', etc. The main area shows a table of data with columns: objectId, mailaddress, prefecture, age, title, comment, username, createDate, and updateDate. Three rows of data are visible. An orange box highlights the 'username' column in the table. A red speech bubble points to this column with the text: 'お名前のデータの場合はフィールド「username」の中にデータが保存されます'. Below this, a larger table shows the same data with the 'username' column highlighted by a red dashed border. The columns are labeled: メールアドレス, お住まい, ご年齢, タイトル, コメント, お名前.

mailaddress	prefecture	age	title	comment	username
suzukis@mbaas.co.jp	北海道	20	水曜日	オンラインセミナーC	鈴木 三郎
tanakaj@mbaas.co.jp	京都府	25	火曜日	オンラインセミナーB	田中 次郎
yamadat@mbaas.co.jp	東京都	30	月曜日	オンラインセミナーA	山田 太郎

# <Demo2> データの取得：完成イメージ

- ✓ フッターのDemo2をクリックすると、保存済みのデータを一覧として表示します
- ✓ データは送信日時が新しいものから順に表示されます



# <Demo3> データの条件検索：完成イメージ

- ✓ メールアドレス、お住まい、日付のいずれかで検索を行うと、Demo2画面に遷移し、条件に一致したデータを一覧として表示します



## 動作環境

- Google Chrome (推奨)
- Firefox

## 準備作業

- 上記ブラウザの最新バージョンをインストールして下さい
- MonacaとmBaaSの会員登録を行い、アカウントを作成して下さい
  - [Monaca会員登録](#) (無料)
  - [ニフティクラウド mobile backend会員登録](#) (無料)

## 作業

1. mBaaSでアプリの作成
2. Monacaでプロジェクトのインポート
3. SDKのインストール
4. APIキーの設定

## コード解説と動作確認

5. <Demo1>データの保存
6. <Demo2>データの取得
7. <Demo3>データの条件検索

# 1. mBaaSでアプリの作成

- ・ [mBaaS](#)にログインして下さい
- ・ アプリの新規作成画面が出たらアプリ名を入力し、「新規作成」をクリックします
- ・ 既にmBaaSをご利用の方は「+新しいアプリ」からアプリの新規作成が可能です

**mBaaS**

アプリの新規作成

アプリ名  
formdemo

半角英数字もしくはアンダースコア

戻る 新規作成

「新規作成」をクリック

アプリ名：  
「formdemo」

＜既にご利用の方＞

+新しいアプリ

メニューを閉じる 更新

ダッシュボード 会員管理 データストア ファイルストア スクリプト プッシュ通知

ステータス 2017年7月

登録会員数	APIリクエスト	プッシュ通知	ストレージ	スクリプト
0人	1,372回	0回	0.31MB	0回

グラフ APIリクエスト数 2017/06/26 - 2017/07/26

250  
150  
50  
0

06月26日 06月29日 07月02日 07月05日 07月08日 07月11日 07月14日 07月17日 07月20日 07月23日 07月26日 日

+新しいアプリ ボタン

# 1. mBaaSでアプリの作成

- ・ アプリが作成されると2種類のAPIキーが発行されます
- ・ 「OK」をクリックするとアプリの作成が完了し、ダッシュボード画面が表示されます



クリック

# 1. mBaaSでアプリの作成

- APIキーはダッシュボードの画面右上にある「アプリ設定」からいつでも確認できます

The screenshot shows the NIFTY Cloud mobile backend dashboard. On the left, there's a sidebar with various management options like Member Management, Data Store, and Script. The main area displays a 'Status' section for August 2017, showing metrics such as Registered Members (0), API Requests (0), Push Notifications (0), Storage (0 MB), and Scripts (0). Below this is a graph section titled 'Graph' which currently shows no data. At the bottom, there are links to Fujitsu Cloud Technologies homepage, contact information, and privacy policies.

On the right side of the dashboard, there's a smaller window titled 'App Settings'. This window has a blue header with 'お知らせ' (Notification) and 'アプリ設定' (App Settings). A large orange callout bubble with the text 'クリック' (Click) points to the 'App Settings' button, which is highlighted with an orange oval.

# 1. mBaaSでアプリの作成

- 「アプリ設定」を開くと、「APIキー」という項目の中に2種類のキーが表示されています

The screenshot shows the 'App Settings' screen for the 'formdemo' application. On the left sidebar, under the 'API Key' section, there are two main sections: 'Application Key' and 'Client Key'. Each section contains a key value and a 'Copy' button. A large orange arrow points from the text 'APIキー' in the list above to the 'Application Key' section.

APIキー

Application Key

Client Key

Application Key / Client Keyの再生成

FormAppアプリのApplication Key / Client Keyを再生成します。

Application Keyを再生成する Client Keyを再生成する

REST APIツール

## 2. Monacaでプロジェクトのインポート

- ・[Monaca](#)にログインし、画面左上の「Import Project」をクリックします
- ・プロジェクトのインポート画面が開くので、プロジェクト名を入力します
- ・「URLを指定してインポート」を選択し、下記URLを貼り付け後「インポート」をクリックします

### ＜インポート用URL＞

<https://github.com/NIFTYCloud-mbaas/MonacaFormApp/archive/master.zip>

Monaca

The screenshot shows the Monaca User Dashboard. At the top, there's a green button labeled "新規プロジェクトの作成" (Create New Project) and a grey button labeled "Import Project" with a circular arrow icon. Below these are tabs for "オンライン" (Online) and "アーカイブ" (Archive). A search bar with "Tags" and "Filter:" options is present. On the left, there are project cards for "MonacaAdvancePush" and "jQuery TODO App". On the right, there's a preview of a project card titled "Monaca×ゲーム手軽に" with a snippet of code.



プロジェクトのインポート

プロジェクト名\*

説明

インポート方法

URLを指定してインポート

プロジェクトのパッケージをアップロード

GitHubのレポジトリからインポート

http://example.com/monaca-project.zip

参照... ファイルが選択されていません。

GitHubアカウントと連携することで、GitHubレポジトリからプロジェクトインポートが可能です。[GitHub連携設定](#)

インポート キャンセル

The dialog box has a title "プロジェクトのインポート". It contains fields for "プロジェクト名\*" (Project Name) and "説明" (Description). Under "インポート方法" (Import Method), the radio button for "URLを指定してインポート" (Specify URL to import) is selected. The URL "http://example.com/monaca-project.zip" is entered in the input field. There are also options for "プロジェクトのパッケージをアップロード" (Upload project package) and "GitHubのレポジトリからインポート" (Import from GitHub repository). At the bottom are "インポート" (Import) and "キャンセル" (Cancel) buttons.

## 2. Monacaでプロジェクトのインポート

- ・インポート後、Monacaのダッシュボード上にformdemoプロジェクトが作成されます
- ・「開く」をクリックで、プロジェクトを開きます



### 3. SDKのインストール

- formdemoプロジェクトを開いたら、上部のメニューbaruにある「設定」から「JS/CSSコンポーネントの追加と削除...」をクリックして開きます

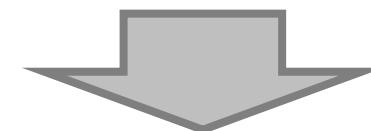


### 3. SDKのインストール

- 検索ボックスに「ncmb」と入力し、「検索」をクリックします
- ncmbのSDKが出てくるので、そのまま「追加」をクリックします

JS/CSSコンポーネント

コンポーネント	操作
Cordova (PhoneGap) Loader バージョン:1.0.0	設定 削除
jQuery (Monaca Version) バージョン:1.9.0 他のパッケージに依存しています。	設定 削除
jQuery Mobile (Monaca Version) バージョン:1.3.1	設定 削除
Monaca Core Utility バージョン:2.0.4	設定 削除



JS/CSSコンポーネント

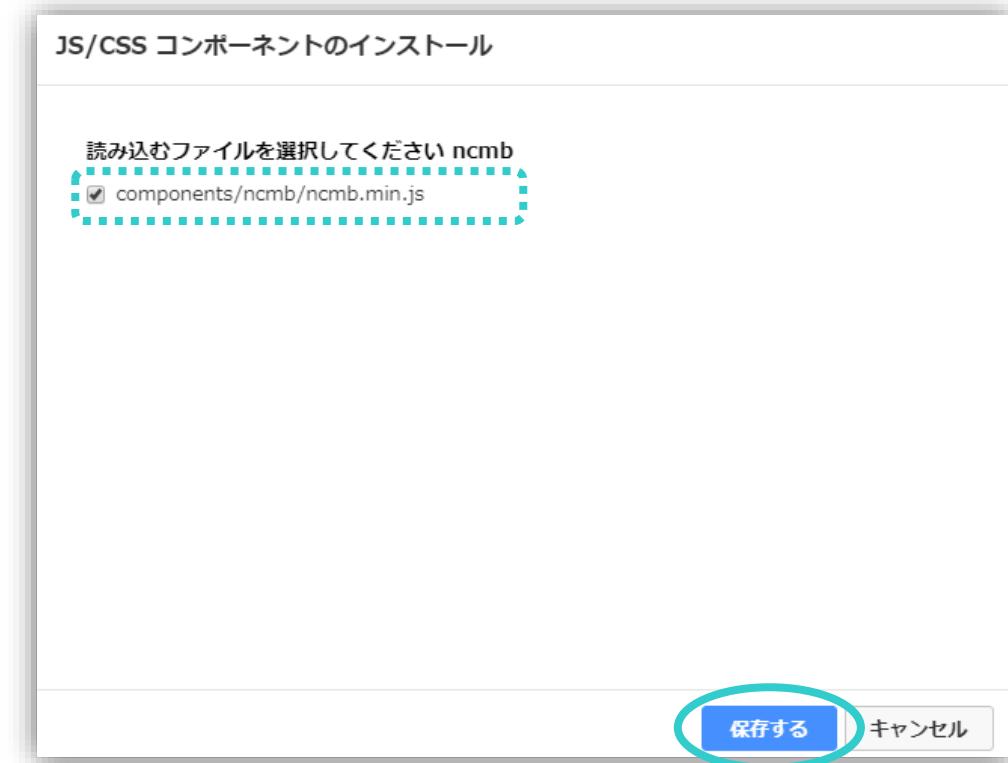
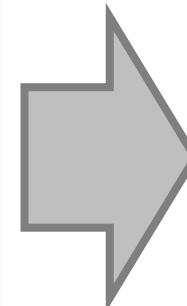
コンポーネント	操作
ncmb	追加

「ncmb」と入力し  
検索ボタンをクリック

「ncmb」が出てきたら  
追加ボタンをクリック

### 3. SDKのインストール

- ・「追加」をクリック後、インストール画面が出てきます
- ・デフォルトで最新バージョンが表示されているので、「インストール」をクリックして下さい
- ・SDKが出てくるので、チェックを入れて「保存する」をクリックして下さい



### 3. SDKのインストール

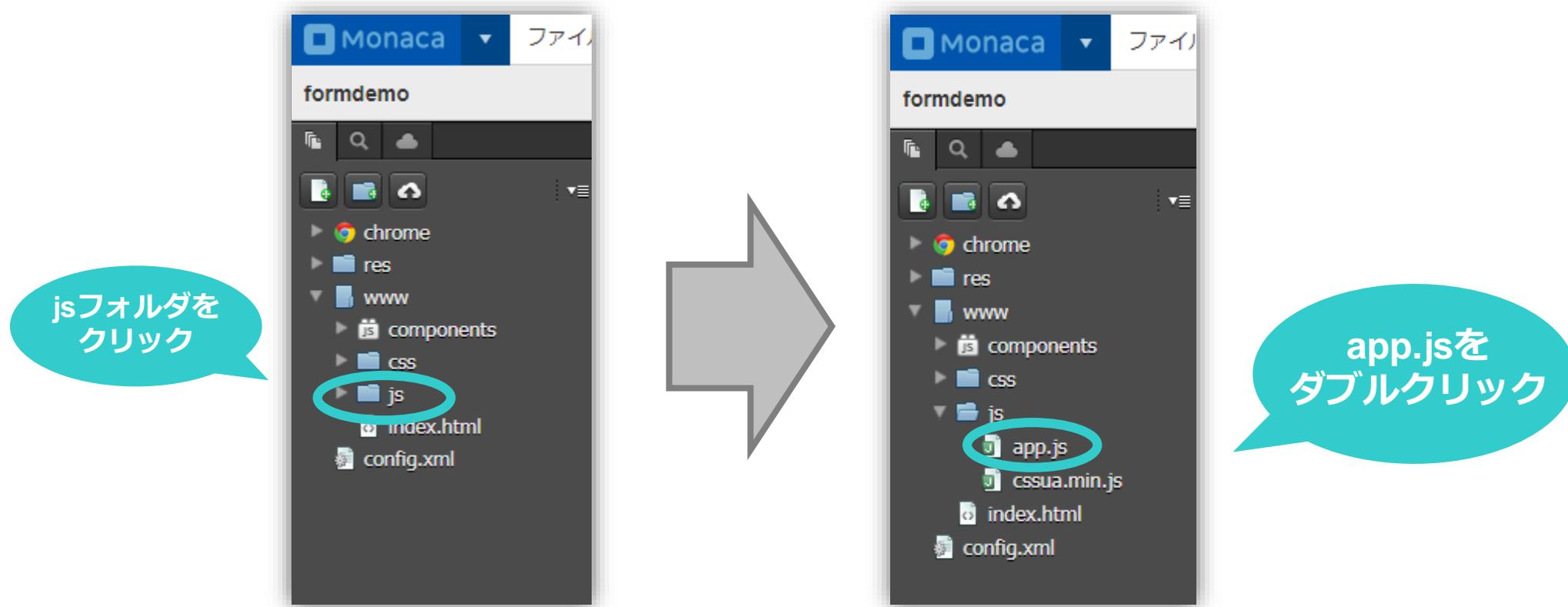
- 下記のように表示されていればOKです

JS/CSSコンポーネント

コンポーネント	コンポーネント名	検索
Cordova (PhoneGap) Loader バージョン:1.0.0	設定	削除
jQuery (Monaca Version) バージョン:1.9.0 他のパッケージに依存しています。	設定	削除
jQuery Mobile (Monaca Version) バージョン:1.3.1	設定	削除
Monaca Core Utility バージョン:2.0.4	設定	削除
ncmb バージョン:2.1.5	設定	削除
Onsen UI (Monaca Version)	追加	
WinJS (Monaca Version)	追加	

## 4. APIキーの設定

- formdemoプロジェクトの「js」フォルダをクリックし、中から「app.js」をダブルクリックして開きます



# 4. APIキーの設定

- app.jsが開きます

The screenshot shows the Monaca IDE interface. On the left, the project structure for 'formdemo' is visible, including files like index.html, app.js, config.xml, and cssua.min.js. The main workspace shows the code for 'app.js' and a preview of the mobile application 'Demo1 : フォーム'.

**app.js Content:**

```
//APIキーの設定とSDKの初期化
var appKey = "アプリケーションキー";
var clientKey = "クライアントキー";
var ncmb = new NCMB(appKey,clientKey);

// -----[Demo1]データをmBaaSに保存する -----
function sendForm() {
    //ユーザーの入力したデータを変数にセットする
    var username = $("#form_name").val(); //お名前
    var mailaddress = $("#form_mailaddress").val(); //メールアドレス
    var prefecture = $("#form_prefecture").val(); //お住まい
    var agestr = $("#form_age").val(); //ご年齢
    var title = $("#form_title").val(); //タイトル
    var comment = $("#form_comment").val(); //内容

    //agestrを数値に変換
    var ageint = Number(agestr);

    //入力規則およびデータをフィールドにセットする
    if(username == ""){
        alert("お名前が入力されていません");
    }else if(mailaddress == ""){
        alert("メールアドレスが入力されていません");
    }else if(title == ""){
        alert("タイトルが入力されていません");
    }else if(comment == ""){
        alert("お問い合わせ内容が入力されていません");
    }else{
        //mBaaSに保存先クラスの作成
        var SaveData = ncmb.DataStore("SaveData");

        //インスタンスの生成
        var saveData = new SaveData();

        //インスタンスにデータをセットする
        saveData.set("username", username)
            .set("mailaddress", mailaddress)
            .set("prefecture", prefecture)
            .set("age", ageint)
            .set("title", title)
            .set("comment", comment)
            .save()
            .then(function(results){
                //保存に成功した場合の処理
                alert("お問い合わせを受け付けました");
                console.log("お問い合わせを受け付けました");
                location.reload();
            })
    }
}
```

**Preview of Demo1: フォーム:**

The preview shows a form with fields for 'お名前' (Required), 'メールアドレス' (Required), 'お住まい' and 'ご年齢' (both dropdown menus), 'タイトル' (Required), and 'お問い合わせ内容' (Required). Below the form, there are three buttons labeled 'Demo1', 'Demo2', and 'Demo3'.

# 4. APIキーの設定

- app.jsの最上部「APIキーの設定とSDKの初期化」の部分にカーソルを合わせます

```
//APIキーの設定とSDKの初期化
var appKey = "アプリケーションキー";
var clientKey = "クライアントキー";
var ncmb = new NCMB(appKey,clientKey);

// -----[Demo1]データをmBaaSに保存する -----
function sendForm() {
    //ユーザーの入力したデータを変数にセットする
    var username = $("#form_name").val(); //お名前
    var mailaddress = $("#form_mailaddress").val(); //メールアドレス
    var prefecture = $("#form_prefecture").val(); //お住まい
    var agestr = $("#form_age").val(); //ご年齢
    var title = $("#form_title").val(); //タイトル
    var comment = $("#form_comment").val(); //内容

    //agestrを数値に変換
    var ageint = Number(agestr);

    //入力規則およびデータをフィールドにセットする
    if(username == ""){
        alert("お名前が入力されていません");
    }else if(mailaddress == ""){
        alert("メールアドレスが入力されていません");
    }else if(title == ""){
        alert("タイトルが入力されていません");
    }else if(comment == ""){
        alert("お問い合わせ内容が入力されていません");
    }else{
        //mBaaSに保存先クラスの作成
        var SaveData = ncmb.DataStore("SaveData");

        //インスタンスの生成
        var saveData = new SaveData();

        //インスタンスにデータをセットする
        saveData.set("username", username)
            .set("mailaddress", mailaddress)
            .set("prefecture", prefecture)
            .set("age", ageint)
            .set("title", title)
            .set("comment", comment)
            .save()
            .then(function(results){
                //保存に成功した場合の処理
                alert("お問い合わせを受け付けました");
                console.log("お問い合わせを受け付けました");
                location.reload();
            })
    }
}
```

お問い合わせ内容 (必須)

Demo1 : フォーム

お名前 (必須)

メールアドレス (必須)

お問い合わせ内容 (必須)

Demo1 Demo2 Demo3

## 4. APIキーの設定

- mBaaSの「アプリ設定」からAPIキーをコピーし、それぞれ青字の部分に貼り付けて保存します
- ここでアプリの動作を確認するための準備が完了です

app.js

//APIキーの設定とSDKの初期化

```
var appKey      = "アプリケーションキー";
var clientKey  = "クライアントキー";
var ncmb       = new NCMB(appKey,clientKey);
```

ダブルクオーテーション("")  
を消さないように注意！

貼り付け

mBaaS

コピーボタン  
または  
直接ダブルクリックで  
コピー可能

APIキー

アプリケーションキー ?

クライアントキー ?

コピー  
コピー

# <Demo1>データの保存：完成イメージ①（再掲）

- ✓ 各項目に入力し送信ボタンを押すと、入力したデータがmBaaSのデータストアに保存されます

**Monaca**



Demo1 : フォーム

お名前 (必須)

メールアドレス (必須)

お住まい ご年齢

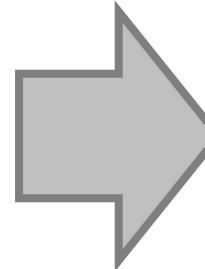
--選択-- --選択--

タイトル (必須)

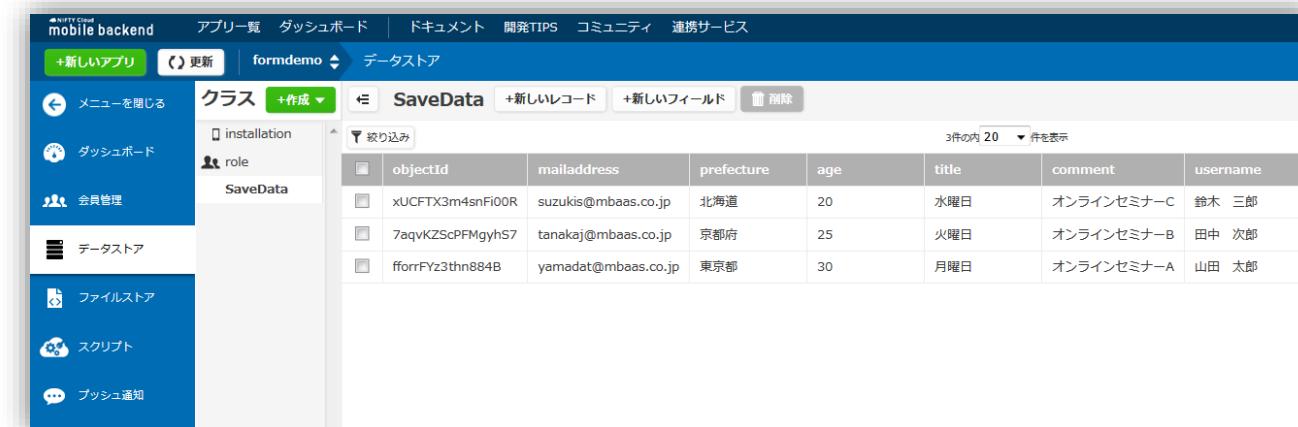
お問い合わせ内容 (必須)

送信

Demo1 Demo2 Demo3



**mBaaS**



objectID	mailaddress	prefecture	age	title	comment	username
xUCFTX3m4snFi00R	suzukis@mbaas.co.jp	北海道	20	水曜日	オンラインセミナーC	鈴木 三郎
7aqvKZScPFMgyhS7	tanakaj@mbaas.co.jp	京都府	25	火曜日	オンラインセミナーB	田中 次郎
fforrFYz3thn884B	yamadat@mbaas.co.jp	東京都	30	月曜日	オンラインセミナーA	山田 太郎

# <Demo1>データの保存：完成イメージ②（再掲）

- ✓ お名前、メールアドレス等の各データはデータストアにフィールドごとに保存されます

The screenshot shows the NIFTY Cloud mobile backend application interface. On the left, there's a sidebar with various navigation options like 'mobile backend', 'アプリ一覧', 'ダッシュボード', etc. The main area shows a table of data with columns: objectId, mailaddress, prefecture, age, title, comment, username, createDate, and updateDate. Three rows of data are visible. An orange box highlights the 'username' column in the table. A red speech bubble points to this column with the text: 'お名前のデータの場合はフィールド「username」の中にデータが保存されます'. Below this, a larger table shows the same data with the 'username' column highlighted by a red dashed border.

mailaddress	prefecture	age	title	comment	username
suzukis@mbaas.co.jp	北海道	20	水曜日	オンラインセミナーC	鈴木 三郎
tanakaj@mbaas.co.jp	京都府	25	火曜日	オンラインセミナーB	田中 次郎
yamadat@mbaas.co.jp	東京都	30	月曜日	オンラインセミナーA	山田 太郎

メールアドレス

お住まい

ご年齢

タイトル

コメント

お名前

## 5. <Demo1>データの保存

### コード解説 (app.js : 7~56行目)

- ・ mBaaSに**保存先クラスと保存先クラスのインスタンスを生成します**

```
// -----[Demo1]データをmBaaSに保存する -----//
function sendForm() {
/* 省略 */
    //mBaaSに保存先クラスの作成
    var SaveData = ncmb.DataStore("SaveData");

    //インスタンスの生成
    var saveData = new SaveData();
```

## 5. <Demo1>データの保存

### コード解説 (app.js : 7~56行目)

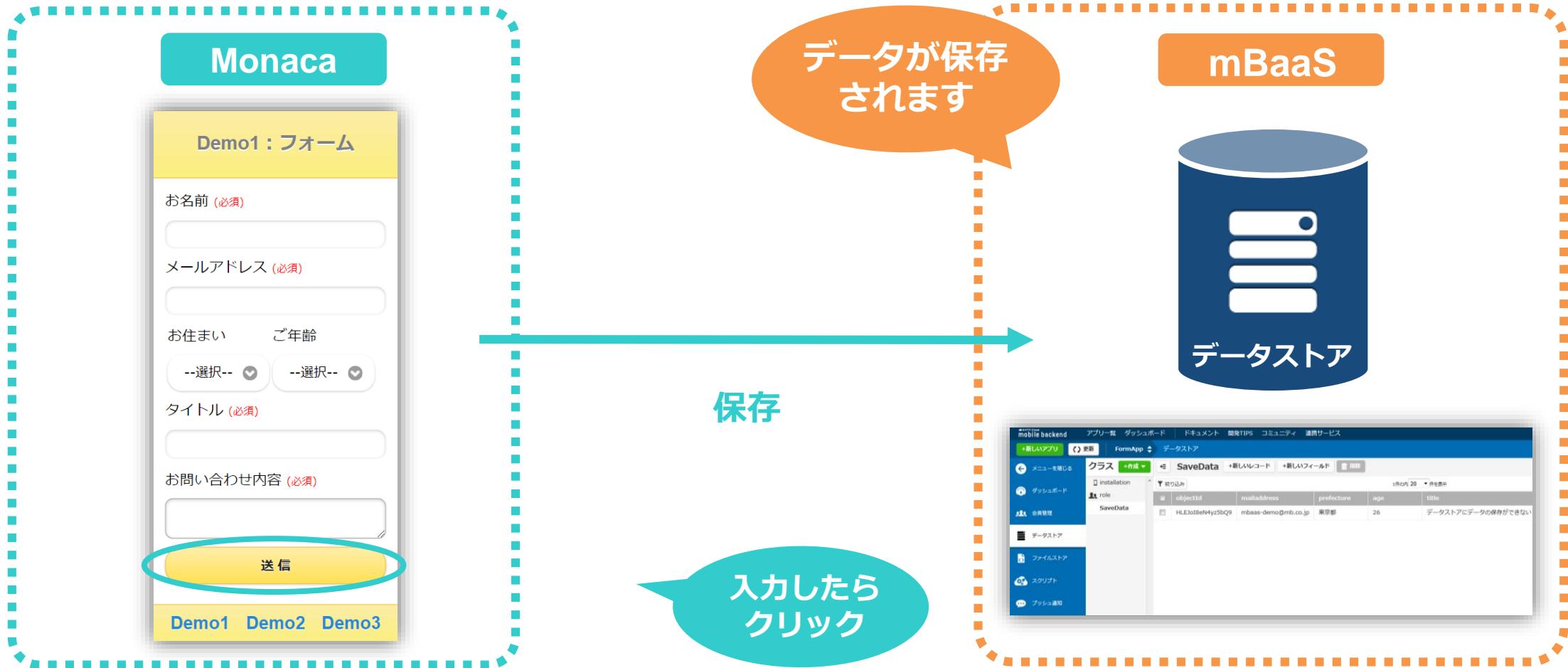
- 各フォームのデータを記録するため **set()** でデータを設定します
- データを設定後、 **save()** で保存します

```
//インスタンスにデータをセットする
saveData.set("username", username)
    .set("mailaddress", mailaddress)
    .set("prefecture", prefecture)
    .set("age", ageint)
    .set("title", title)
    .set("comment", comment)
    .save()
    .then(function(results){
        //保存に成功した場合の処理
    })
    .catch(function(error){
        //保存に失敗した場合の処理
    });
}
```

⇒ **SaveData**クラスの中に  
フィールド (**username**、**mailaddress**...etc)  
が作成され、データが保存されます！

## 5. <Demo1>データの保存

動作の仕組みはこのようになっています



# 5. <Demo1>データの保存

## 動作確認方法

- 画面上部のメニューbaruから「プレビュー」をクリックします

The screenshot shows the Monaca IDE interface. The left sidebar displays the project structure for 'formdemo' with files like 'index.html', 'app.js', 'cssua.min.js', 'config.xml', and 'js/app.js'. The main editor window shows the 'app.js' file content:

```
//APIキーの設定とSDKの初期化
var apiKey = "アプリケーションキー";
var clientKey = "クライアントキー";
var ncmb = new NCMB(apiKey,clientKey);

// -----[Demo1]データをmBaaSに保存する -----
function sendForm() {
    //ユーザーの入力したデータを変数にセットする
    var username = $("#form_name").val(); //お名前
    var mailaddress = $("#form_mailaddress").val(); //メールアドレス
    var prefecture = $("#form_prefecture").val(); //お住まい
    var agestr = $("#form_age").val(); //年齢
    var title = $("#form_title").val(); //タイトル
    var comment = $("#form_comment").val(); //内容

    //agestrを数値に変換
    var ageint = Number(agestr);

    //入力規則およびデータをフィールドにセットする
    if(username == ""){
        alert("お名前が入力されていません");
    }else if(mailaddress == ""){
        alert("メールアドレスが入力されていません");
    }else if(title == ""){
        alert("タイトルが入力されていません");
    }else if(comment == ""){
        alert("お問い合わせ内容が入力されていません");
    }else{
        //mBaaSに保存先クラスの作成
        var SaveData = ncmb.DataStore("SaveData");

        //インスタンスの生成
        var saveData = new SaveData();

        //インスタンスにデータをセットする
        saveData.set("username", username)
            .set("mailaddress", mailaddress)
            .set("prefecture", prefecture)
            .set("age", ageint)
            .set("title", title)
            .set("comment", comment)
            .save()
            .then(function(results){
                //保存に成功した場合の処理
                alert("お問い合わせを受け付けました");
                console.log("お問い合わせを受け付けました");
                location.reload();
            })
    }
}
```

# 5. <Demo1>データの保存

## 動作確認方法

- ・画面右にアプリのプレビュー画面が表示されます

The screenshot shows the Monaca IDE interface. On the left, the project structure for 'formdemo' is visible, including files like index.html, app.js, config.xml, and various CSS and JS files under 'www'. The main editor window displays the 'app.js' file content:

```
//APIキーの設定とSDKの初期化
var apiKey = "アプリケーションキー";
var clientKey = "クライアントキー";
var ncmb = new NCMB(apiKey,clientKey);

// -----[Demo1]データをmBaaSに保存する -----
function sendForm() {
    //ユーザーの入力したデータを変数にセットする
    var username = $("#form_name").val(); //お名前
    var mailaddress = $("#form_mailaddress").val(); //メールアドレス
    var prefecture = $("#form_prefecture").val(); //お住まい
    var agestr = $("#form_age").val(); //ご年齢
    var title = $("#form_title").val(); //タイトル
    var comment = $("#form_comment").val(); //内容

    //agestrを数値に変換
    var ageint = Number(agestr);

    //入力規則およびデータをフィールドにセットする
    if(username == ""){
        alert("お名前が入力されていません");
    }else if(mailaddress == ""){
        alert("メールアドレスが入力されていません");
    }else if(title == ""){
        alert("タイトルが入力されていません");
    }else if(comment == ""){
        alert("お問い合わせ内容が入力されていません");
    }else{
        //mBaaSに保存先クラスの作成
        var SaveData = ncmb.DataStore("SaveData");

        //インスタンスの生成
        var saveData = new SaveData();

        //インスタンスにデータをセットする
        saveData.set("username", username)
            .set("mailaddress", mailaddress)
            .set("prefecture", prefecture)
            .set("age", ageint)
            .set("title", title)
            .set("comment", comment)
            .save()
            .then(function(results){
                //保存に成功した場合の処理
                alert("お問い合わせを受け付けました");
                console.log("お問い合わせを受け付けました");
                location.reload();
            })
    }
}

//-----[Demo1]データをmBaaSに保存する -----
```

On the right, a preview window titled 'Demo1 : フォーム' shows a form with fields for 'お名前' (必填), 'メールアドレス' (必填), 'お住まい' (必填), 'ご年齢' (必填), 'タイトル' (必填), and 'お問い合わせ内容' (必填). Below the form, there is a footer bar with three buttons: 'Demo1', 'Demo2', and 'Demo3'.

## 5. <Demo1>データの保存

### 動作確認方法

- ・アプリのプレビュー画面を開いた時点でアプリのDemo1画面が表示されています
- ・入力フォームは6箇所で、**(必須)**と書いてあるフォームは入力必須です
- ・メールアドレスは全角文字が入力不可で、@マークが必須です

Demo1 : フォーム

お名前 **(必須)**  
**①**

メールアドレス **(必須)**  
**②**

お住まい **③** ご年齢 **④**  
--選択--  --選択--

タイトル **(必須)**  
**⑤**

お問い合わせ内容 **(必須)**  
**⑥**

送信

Demo1 Demo2 Demo3

### <入力フォーム>

- ①お名前 **(必須)**
- ②メールアドレス **(必須)**
- ③お住まい
- ④ご年齢
- ⑤タイトル **(必須)**
- ⑥お問い合わせ内容 **(必須)**

## 5. <Demo1>データの保存

### 動作確認方法

(例) アプリのDemo1画面から次のデータを入力して送信します

#### <データ①>

お名前	:	山田 太郎
メールアドレス	:	<a href="mailto:yamadat@mbaas.co.jp">yamadat@mbaas.co.jp</a>
お住まい	:	東京都
ご年齢	:	30歳
タイトル	:	月曜日
お問い合わせ内容	:	オンラインセミナーA

#### <データ②>

お名前	:	田中 次郎
メールアドレス	:	<a href="mailto:tanakaj@mbaas.co.jp">tanakaj@mbaas.co.jp</a>
お住まい	:	京都府
ご年齢	:	25歳
タイトル	:	火曜日
お問い合わせ内容	:	オンラインセミナーB

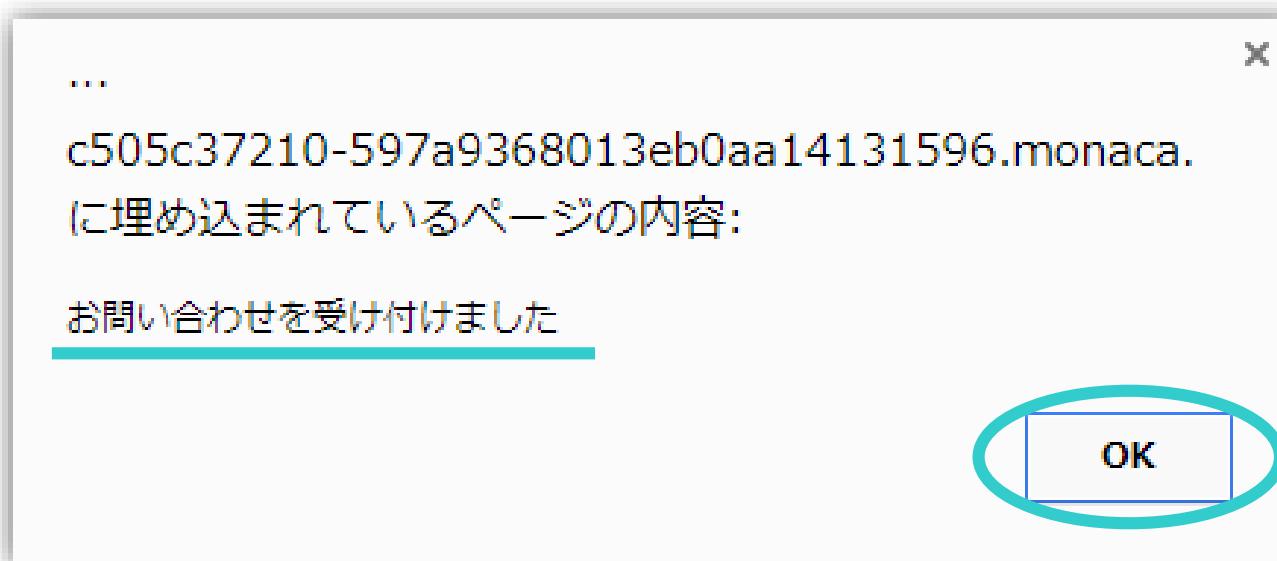
#### <データ③>

お名前	:	鈴木 三郎
メールアドレス	:	<a href="mailto:suzukis@mbaas.co.jp">suzukis@mbaas.co.jp</a>
お住まい	:	北海道
ご年齢	:	20歳
タイトル	:	水曜日
お問い合わせ内容	:	オンラインセミナーC

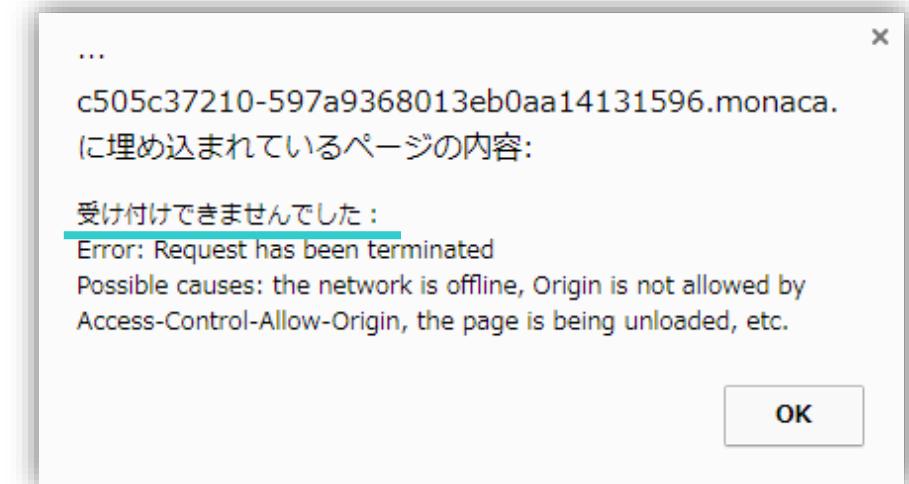
## 5. <Demo1>データの保存

### 動作確認方法

- ・送信成功の場合、「お問い合わせを受け付けました」というアラートが表示されます
- ・「OK」をクリックします



※送信失敗時はエラーの内容が表示されます



# 5. <Demo1>データの保存

## 動作確認方法

- ・mBaaSを開き、ダッシュボードからデータストアをクリックして開きます
- ・データストア内の保存先クラスをクリックして開きます

mBaaS

The screenshot shows the mBaaS dashboard with various metrics for the 'formdemo' application. On the left sidebar, the 'Datastore' icon is circled in orange. A large orange callout bubble points to this icon with the text 'データストアをクリックします' (Click Datastore).



The screenshot shows the Datastore list screen for the 'formdemo' application. The 'SaveData' class is circled in orange. A large orange callout bubble points to this class with the text '作成した保存先クラスをクリックします' (Click the created save target class).

# 5. <Demo1>データの保存

## 動作確認方法

- アプリから送信した各データがフィールドごとに保存されていることが確認できます
- 動作確認は以上で完了です

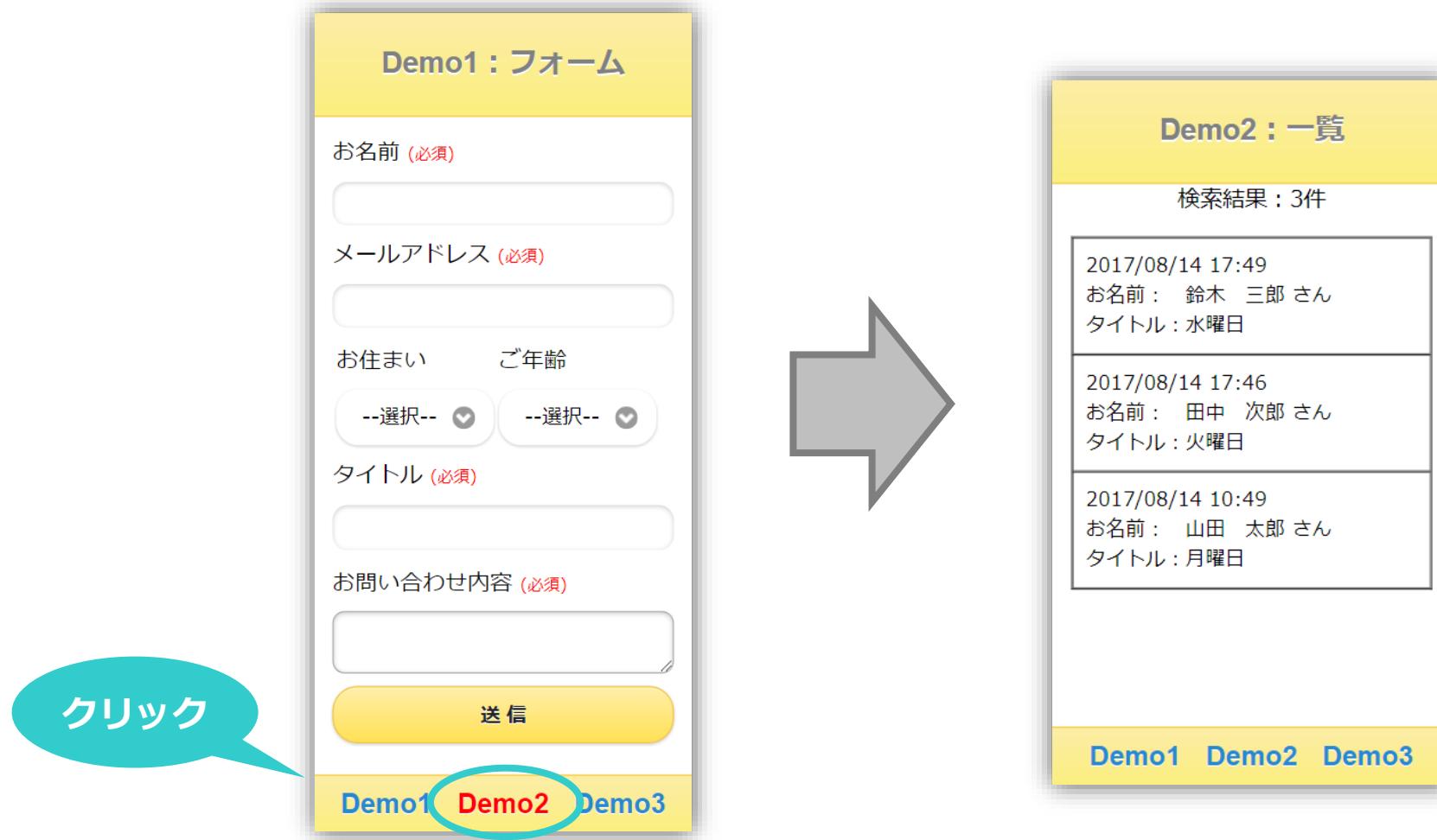
The screenshot shows the NIFTY Cloud mobile backend application interface. On the left, there's a sidebar with various navigation items like 'mobile backend', 'アプリ一覧', 'ダッシュボード', etc. The main area has a title 'SaveData' and a table with columns: objectId, mailaddress, prefecture, age, title, comment, username, createDate, and updateDate. Three rows of data are listed. Below this, a red box highlights the 'username' column in the table, and a red arrow points down to a smaller table below it. This smaller table also has columns: mailaddress, prefecture, age, title, comment, and username. It contains three rows of data, matching the ones in the larger table. A red callout bubble on the right side of the smaller table contains Japanese text explaining the data.

mailaddress	prefecture	age	title	comment	username
suzukis@mbaas.co.jp	北海道	20	水曜日	オンラインセミナーC	鈴木 三郎
tanakaj@mbaas.co.jp	京都府	25	火曜日	オンラインセミナーB	田中 次郎
yamadat@mbaas.co.jp	東京都	30	月曜日	オンラインセミナーA	山田 太郎

お名前の場合、  
フィールド「username」に  
・山田 太郎さん  
・田中 次郎さん  
・鈴木 三郎さん  
のデータが保存されています

# <Demo2>データの取得：完成イメージ（再掲）

- ✓ フッターのDemo2をクリックすると、保存済みのデータを一覧として表示します
- ✓ データは送信日時が新しいものから順に表示されます



## 6. <Demo2> データの取得

### コード解説 (app.js : 59~79行目)

- 並び替え : order()でcreateDateのデータを降順で並び替えます
  - 取得 : fetchAll()で検索条件に一致した情報を全件調べて取得します
- ※ fetchAll() は1度に最大100件まで取得可能です

```
//----- [Demo2]保存したデータを全件検索し取得する-----//  
function checkForm(){  
    /* 省略 */  
    //インスタンスの生成  
    var saveData = ncmb.DataStore("SaveData");  
  
    //データを降順で取得する  
    saveData.order("createDate",true) 並び替え  
        .fetchAll() 取得  
        .then(function(results){  
            //全件検索に成功した場合の処理  
        })  
        .catch(function(error){  
            //全件検索に失敗した場合の処理  
        });  
}
```

## 6. <Demo2>データの取得

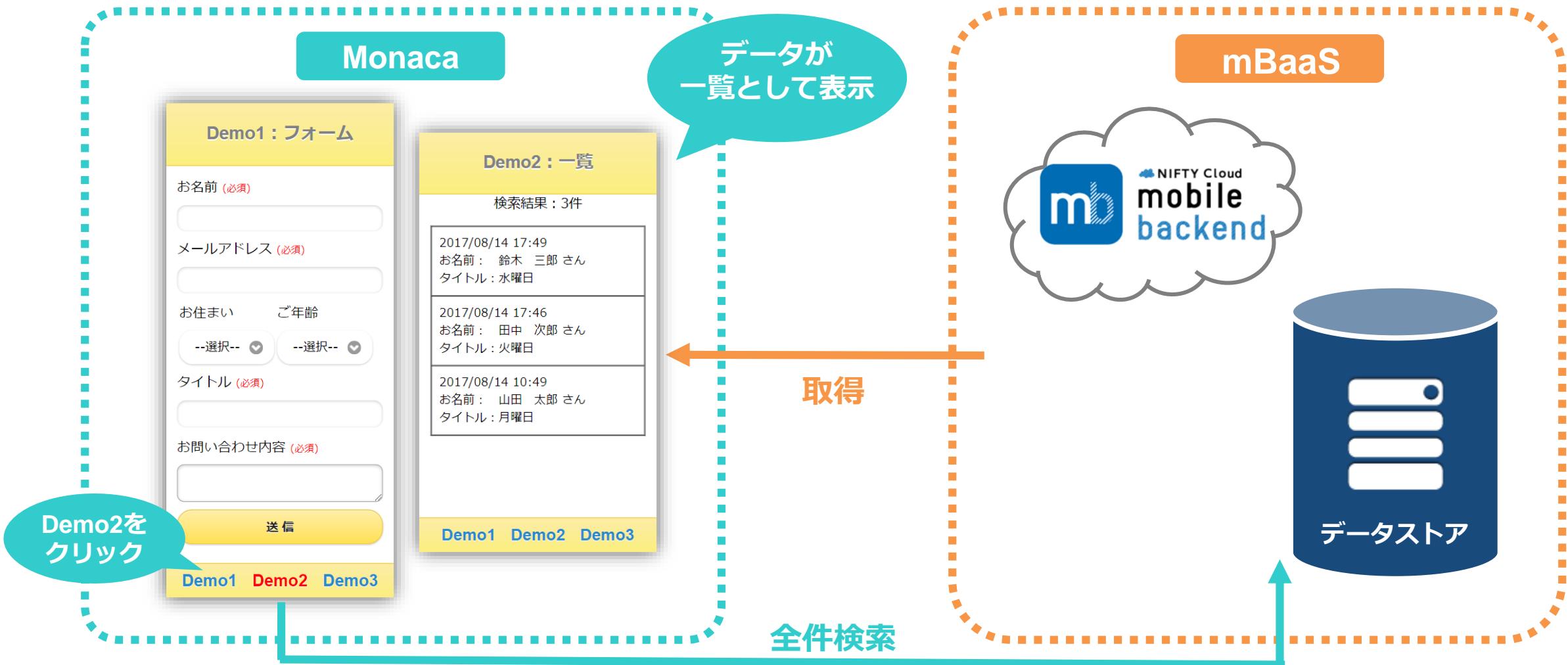
createDateはmBaaS>データストア>保存先クラスのフィールドにて確認できます

The screenshot shows the NIFTY Cloud mobile backend application interface. On the left is a sidebar with various icons and links: +新しいアプリ, アップデート, データストア (selected), ファイルストア, スクリプト, ブッシュ通知. The main area has a header with tabs: アプリ一覧, ダッシュボード, ドキュメント, 開発TIPS, コミュニティ, 連携サービス. Below the header, there's a breadcrumb trail: formdemo > データストア. The current screen displays a list of 'SaveData' records with columns: objectId, mailaddress, prefecture, age, title, comment, username, createDate, and updateDate. A red arrow points from the text 'ここ！' (Here!) to the 'createDate' column. A red double-headed vertical arrow between the main list and a zoomed-in view indicates a comparison between new and old data. The zoomed-in view shows three entries with their respective create dates.

createDate
2017-08-14T17:49:38.497+09:00
2017-08-14T17:46:55.864+09:00
2017-08-14T10:49:20.606+09:00

## 6. <Demo2>データの取得

動作の仕組みはこのようになっています



## 6. <Demo2>データの取得

### 動作確認方法

- ・アプリのプレビュー画面を開いた状態にします
- ・アプリのフッターから「Demo2」をクリックしてDemo2画面を開きます
- ・データが一覧として表示されます



## 6. <Demo2> データの取得

### 動作確認方法

- データは新しいものから順に表示されるので  
データストアにデータが複数保存されている場合は**一番上が最新**のデータです
- 動作確認は以上で完了です

Demo2 : 一覧

検索結果 : 3件

2017/08/14 17:49 お名前： 鈴木 三郎 さん タイトル：水曜日
2017/08/14 17:46 お名前： 田中 次郎 さん タイトル：火曜日
2017/08/14 10:49 お名前： 山田 太郎 さん タイトル：月曜日

← 最新！

Demo1 Demo2 Demo3

2017/08/14 17:49 お名前： 鈴木 三郎 さん タイトル：水曜日
2017/08/14 17:46 お名前： 田中 次郎 さん タイトル：火曜日
2017/08/14 10:49 お名前： 山田 太郎 さん タイトル：月曜日

# <Demo3> データの条件検索：完成イメージ（再掲）

- ✓ メールアドレス、お住まい、日付のいずれかで検索を行うと、Demo2画面に遷移し、条件に一致したデータを一覧として表示します

The diagram illustrates the workflow for performing a search. On the left, the 'Demo3 : 条件検索' (Condition Search) screen is shown. It features three search sections: '1. メールアドレスを検索' (Search by Email Address), '2. お住まいを検索' (Search by Address), and '3. 日付を検索' (Search by Date). Each section has a yellow '検索' (Search) button. A large grey arrow points from the Demo3 screen to the right, leading to the 'Demo2 : 一覧' (List) screen. The Demo2 screen displays a list of 3 search results, each containing a date and name. At the bottom of both screens, there are blue navigation buttons labeled 'Demo1', 'Demo2', and 'Demo3'.

検索結果	日付	名前	タイトル
1	2017/08/14 17:49	鈴木 三郎 さん	水曜日
2	2017/08/14 17:46	田中 次郎 さん	火曜日
3	2017/08/14 10:49	山田 太郎 さん	月曜日

## 7. <Demo3>データの条件検索

### コード解説：①メールアドレスを検索（app.js : 82~104行目）

- 並び替え : <Demo2>データの取得と同様に設定します
- 検索条件 : equalTo()は入力したメールアドレスと一致するデータを調べる条件です
- 取得 : <Demo2>データの取得と同様に実行します

```
// -----[Demo3-1]メールアドレスを指定して検索し取得する----- //
function checkAddress(){
    /* 省略 */
    //インスタンスの生成
    var saveData = ncmb.DataStore("SaveData");

    //データの取得
    saveData.order("createDate",true) 並び替え
        .equalTo("mailaddress",mailaddress) 検索条件
        .fetchAll() 取得
        .then(function(results){
            //メールアドレスの検索に成功した場合の処理
        })
        .catch(function(error){
            //メールアドレスの検索に失敗した場合の処理
        });
}
```

## 7. <Demo3> データの条件検索

### コード解説：②お住まいを検索 (app.js : 107~129行目)

- ・並び替え : <Demo2> データの取得と同様に設定します
- ・検索条件 : equalTo()は選択した都道府県と一致するデータを調べる条件です
- ・取得 : <Demo2> データの取得と同様に実行します

```
//----- [Demo3-2]お住まいを指定して検索し取得する-----//  
function checkPrefecture(){  
    /* 省略 */  
    //インスタンスの生成  
    var saveData = ncmb.DataStore("SaveData");  
  
    //データの取得  
    saveData.order("createDate",true) 並び替え  
        .equalTo("prefecture",prefecture) 検索条件  
        .fetchAll() 取得  
        .then(function(results){  
            //お住まいの検索に成功した場合の処理  
        })  
        .catch(function(error){  
            //お住まいの検索に失敗した場合の処理  
        });  
}
```

## 7. <Demo3> データの条件検索

### コード解説：③日付を検索 (app.js : 132~161行目)

- ・検索条件1 : lessThanOrEqualTo()は入力日時より**以前と一致するデータを調べる条件**です
- ・検索条件2 : greaterThanOrEqualTo()は入力日時より**以後と一致するデータを調べる条件**です
- ・並び替え : <Demo2> データの取得と同様に設定します
- ・取得 : <Demo2> データの取得と同様に実行します

```
//----- [Demo3-3]日付を指定して検索し取得する -----//  
function checkDate(divider){  
    /* 省略 */  
    //三項演算子(条件 ? 真:偽)によって以前と以後の処理を分ける  
    (divider ? saveData.lessThanOrEqualTo("createDate", { "__type": "Date", "iso": date.toISOString() }) :  
        saveData.greaterThanOrEqualTo("createDate", { "__type": "Date", "iso": date.toISOString() }))  
            .order("createDate",true) 並び替え  
            .fetchAll() 取得  
            .then(function(results){  
                //日付の検索に成功した場合の処理  
            })  
            .catch(function(error){  
                //日付の検索に失敗した場合の処理  
            });  
}
```

**検索条件1**

**検索条件2**

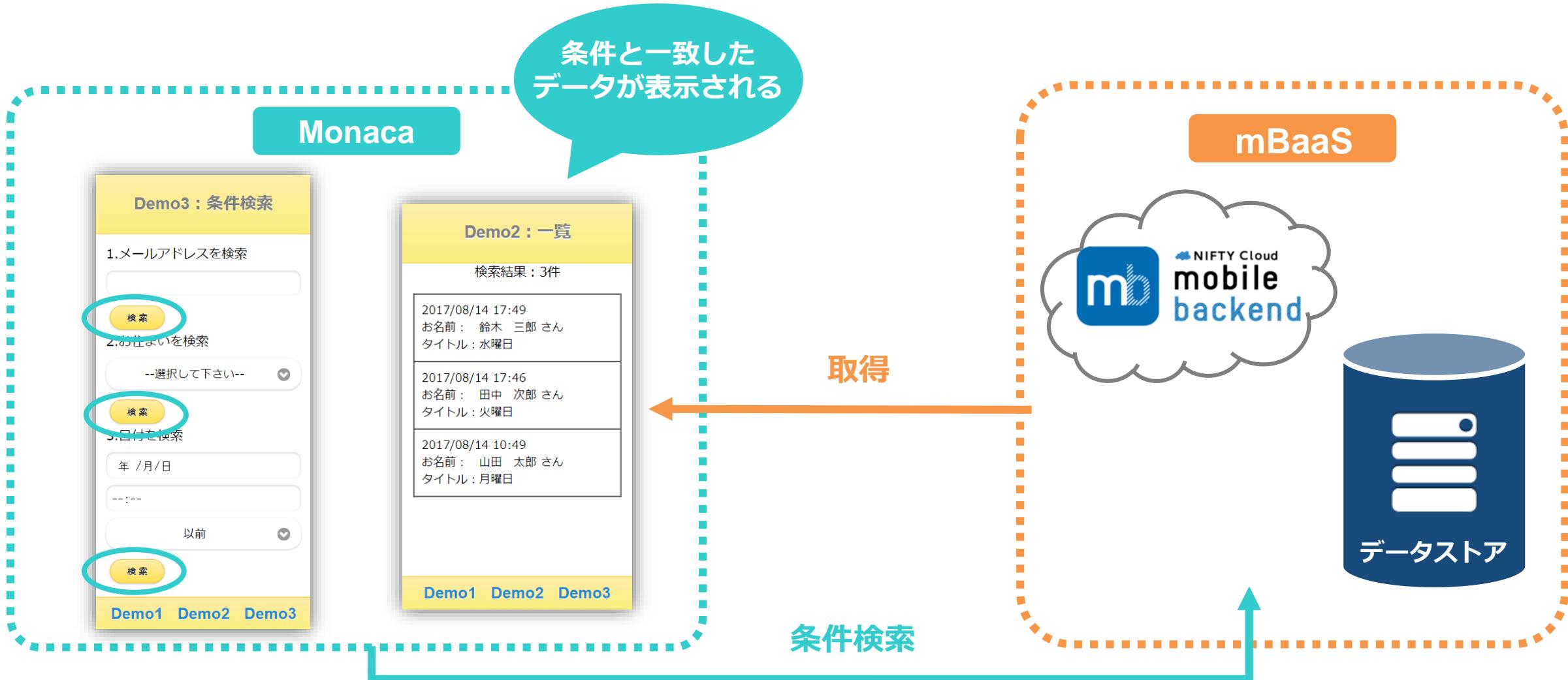
**三項演算子(条件 ? 真:偽)**

dividerの値が**true (真)** ⇒ **以前 (検索条件1)** の条件設定

dividerの値が**false (偽)** ⇒ **以後 (検索条件2)** の条件設定

## 7. <Demo3>データの条件検索

動作の仕組みはこのようになっています



## 7. <Demo3>データの条件検索

### 動作確認方法

- ・アプリのプレビュー画面を開いた状態にします
- ・アプリのフッターから「Demo3」をクリックしてDemo3画面を開きます
- ・条件検索画面が表示されます



## 7. <Demo3> データの条件検索

### 動作確認方法

- 条件検索用フォームは3箇所あります
- メールアドレスの場合は**完全一致検索のみ**可能です
- 日付の場合は**年月日**のほか、**時間と以前／以後**も選択します

The screenshot shows a mobile-style search form titled "Demo3 : 条件検索". It consists of three main sections, each with a yellow "検索" button:

- 1. メールアドレスを検索**: A text input field with a dashed green border. A yellow "検索" button is below it.
- 2. お住まいを検索**: A dropdown menu with the placeholder "--選択して下さい--". A yellow "検索" button is below it.
- 3. 日付を検索**: A date input field with a dashed green border. It includes fields for "年 / 月 / 日" and "以前" (Before). A yellow "検索" button is below it.

At the bottom of the form, there are three blue buttons labeled "Demo1", "Demo2", and "Demo3".

①

②

③

### <入力フォーム>

- ①メールアドレスを検索
- ②お住まいを検索
- ③日付を検索

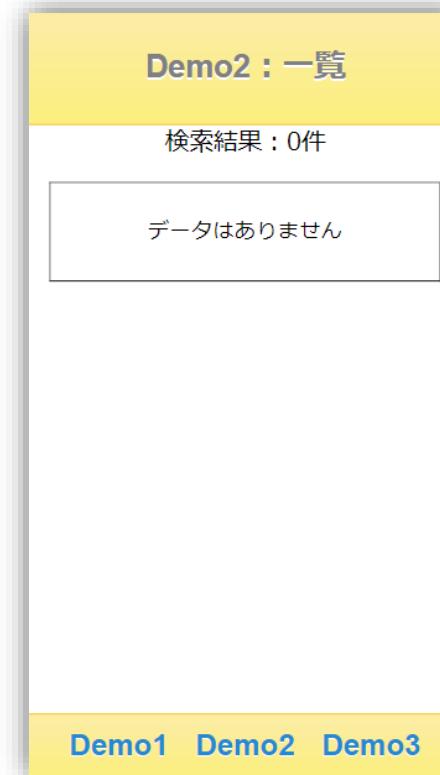
## 7. <Demo3>データの条件検索

### 動作確認方法

- ・検索後はDemo2画面に遷移します
  - ・検索したデータがあった場合は**条件に一致したデータとその件数**が表示されます
  - ・検索したデータがなかった場合は「**データありません**」と表示されます



データがあった場合



データがなかった場合

## 7. <Demo3> データの条件検索

### 動作確認方法

(例) 次のデータを各フォームに入力または選択して検索します

#### ①メールアドレスを検索

⇒ [yamadat@mbaas.co.jp](mailto:yamadat@mbaas.co.jp)

#### ②お住まいを検索

⇒ 京都府

#### ③日付を検索

⇒ (1) 2017年8月22日、16時00分、以前  
(2) 2017年8月22日、16時00分、以後

Demo3 : 条件検索

1. メールアドレスを検索  
 (highlighted with a red dashed box)
2. お住まいを検索  
 (highlighted with a red dashed box)
3. 日付を検索  
  
年 / 月 / 日  
--:--  
以前 (highlighted with a red dashed box)

検索

検索

検索

Demo1 Demo2 Demo3

## 7. <Demo3>データの条件検索

### 動作確認方法：①メールアドレスを検索

- 次のメールアドレスをフォームに入力後、「検索」をクリックします  
⇒ [yamadat@mbaas.co.jp](mailto:yamadat@mbaas.co.jp)

1.メールアドレスを検索

クリック

検索

Demo3 : 条件検索

1.メールアドレスを検索

検索

2.お住まいを検索

--選択して下さい--

検索

3.日付を検索

年 /月/日

--:--

以前

検索

Demo1 Demo2 Demo3

## 7. <Demo3> データの条件検索

### 動作確認方法：①メールアドレスを検索

- 検索結果の件数と、検索した「[yamadat@mbaas.co.jp](#)」から取得できたお名前とタイトル、送信日時のデータが表示されます

「yamadat@mbaas.co.jp」  
というメールアドレスは  
「山田 太郎」さんのデータです！

Demo2 : 一覧

検索結果：1件

2017/08/14 10:49  
お名前： 山田 太郎 さん  
タイトル：月曜日

Demo1 Demo2 Demo3

## 7. <Demo3>データの条件検索

### 動作確認方法：②お住まいを検索

- 次の都道府県をフォームのプルダウンから選択します ⇒ 京都府
- 選択後、「検索」をクリックします



Demo3 : 条件検索

1. メールアドレスを検索

検索

2. お住まいを検索

--選択して下さい--

検索

3. 曜日を検索

年 /月/日

--:--

以前

検索

Demo1 Demo2 Demo3

## 7. <Demo3> データの条件検索

### 動作確認方法：②お住まいを検索

- ・検索結果の件数と、検索した「京都府」から取得できたお名前とタイトル、送信日時のデータが表示されます

「京都府」という都道府県は  
「田中 次郎」さんのデータです！

Demo2 : 一覧

検索結果：1件

2017/08/14 17:46  
お名前： 田中 次郎 さん  
タイトル：火曜日

[Demo1](#) [Demo2](#) [Demo3](#)

## 7. <Demo3>データの条件検索

### 動作確認方法：③日付を検索 (1) & (2) 共通

- 次の年月日をフォームのカレンダーから選択します ⇒ 2017年8月22日

The screenshot illustrates the date selection process for Demo3. It shows two states of a date search form and a calendar overlay.

**<選択前>** (Left): The date input field is empty and highlighted with a red dashed border. A large red arrow points from this state to the right side of the image.

**<選択後>** (Right): The date input field now contains "2017/08/22".

A red callout bubble at the bottom left points to the number "22" in the calendar, which is also highlighted with a red circle.

**Demo3 : 検索** (Top Right): The main search interface with three tabs: 1.メールアドレスを検索, 2.お住まいを検索, and 3.日付を検索. The third tab is highlighted with a red box.

**3.日付を検索** (Bottom Right): The date search form with fields for 年 / 月 / 日, a dropdown for 以前, and a 検索 button. The entire form is highlighted with a red box.

**2017年(平成29年) 8月** (Center): A calendar overlay for August 2017. The date 2017年8月22日 is circled in red. The calendar grid is as follows:

日	月	火	水	木	金	土
30	31	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31	1	2

## 7. <Demo3> データの条件検索

### 動作確認方法 : ③日付を検索 (1) & (2) 共通

- 次の時間をフォームにて入力します ⇒ 16時00分
- 上下ボタンもしくは直接入力で時間を入力できます

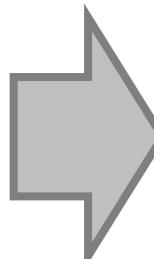
3.日付を検索

2017/08/22

--:--

以前

検索



3.日付を検索

2017/08/22

16:00

x

以前

検索

<入力前>

<入力後>

上下ボタンは  
カーソルを合わせると  
出でてきます

Demo3 : 条件検索

1.メールアドレスを検索

検索

2.お住まいを検索

--選択して下さい--

検索

3.日付を検索

年 / 月 / 日

--:--

以前

検索

Demo1 Demo2 Demo3

## 7. <Demo3> データの条件検索

### 動作確認方法：③日付を検索 (1) 以前の場合

- 「以前」がデフォルトで選択されているので、そのまま「検索」をクリックします

3.日付を検索

2017/08/22

16:00

以前

検索

Demo3 : 条件検索

1.メールアドレスを検索

検索

2.お住まいを検索

--選択して下さい--

検索

3.日付を検索

年 /月/日

--:--

以前

検索

Demo1 Demo2 Demo3

## 7. <Demo3>データの条件検索

### 動作確認方法：③日付を検索 (2) 以後の場合

- 「以前」をクリックし、プルダウンから「以後」を選択します
- 選択後、「検索」をクリックします

3.日付を検索

2017/08/22

16:00

以前

検索

<選択前>

3.日付を検索

2017/08/22

16:00

以後

検索

<選択後>

Demo3 : 条件検索

1.メールアドレスを検索

検索

2.お住まいを検索

--選択して下さい--

検索

3.日付を検索

年 /月/日

--:--

以前

検索

Demo1 Demo2 Demo3

## 7. <Demo3>データの条件検索

### 動作確認方法：③日付を検索

- 検索結果の件数と、検索した年月日、時間、以前または以後から取得できたお名前とタイトル、送信日時のデータが表示されます

Demo2 : 一覧

検索結果 : 0件

データはありません

Demo1 Demo2 Demo3

<以前の場合>

Demo2 : 一覧

検索結果 : 3件

2017/08/22 16:15 お名前 : 鈴木 三郎 さん タイトル : 水曜日
2017/08/22 16:10 お名前 : 田中 次郎 さん タイトル : 火曜日
2017/08/22 16:05 お名前 : 山田 太郎 さん タイトル : 月曜日

Demo1 Demo2 Demo3

<以後の場合>

Monacaとクラウドデータベースを連携させて、3つの動作の仕組みを以下のポイントから学びました

- **データの保存**
  - 保存先クラスの作成
  - set()でフィールドの作成 & データの設定
  - save()でデータの保存
- **データの取得**
  - order()で並び替えの設定
  - fetchAll()で全件検索および取得
- **データの条件検索**
  - equalTo()で一致したデータを調べる検索条件の設定
  - lessThanOrEqualTo() で以前のデータを調べる検索条件の設定
  - greaterThanOrEqualTo()で以後のデータを調べる検索条件の設定

mBaaSのドキュメントサイトも是非ご活用ください

- ・[イントロダクション \(Monaca\) : クイックスタート](#)
- ・[データストア \(Monaca\) : 基本的な使い方](#)



**FUJITSU**

shaping tomorrow with you