

Salesforce Touch Platform

モバイル開発ガイド

salesforce
platform™

Salesforce Touch Platform

モバイル開発ガイド

Salesforce Touch Platform

© Copyright 2000-2012 salesforce.com, inc. All rights reserved. Salesforce.com is a registered trademark of salesforce.com, inc., as are other names and marks. Other marks appearing herein may be trademarks of their respective owners.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher.

ISBN: 978-1-4276-9555-0

著者: Mario Korf、**協力者:** Michael Alderete、Alex Berg、Sandeep Bhanot、Josh Birk、Steve Bobrowski、Jack Cai、Cory Cowgill、Anita Dennis、Tom Gersic、Akhilesh Gupta、Kevin Hawkins、Mike Jacobsen、Wolfgang Mathurin、Eugene Oksman、Pat Patterson、Adam Seligman、Richard Whitley、Quinton Wall、Clive Wong、Rob Woollen

Salesforce Touch Platform に関する最新情報は、以下のページでご確認ください。

http://wiki.developerforce.com/page/Salesforce_touch_platform (英語)

目次

はじめに	1
Salesforce Touch Platform	3
Force.com for Touch	3
モバイルコンテナ (Salesforce Mobile SDK 1.3)	4
Salesforce Identity	4
Dreamforce アプリケーション	5
このドキュメントについて	6
章の構成	6
バージョン	6
ご意見、ご質問について	6
最新情報	7
Mobile SDK GitHub リポジトリ	7
第 1 章: Salesforce Touch Platform でのモバイル開発の概要	9
ネイティブアプリケーション、HTML5 アプリケーション、ハイブリッドアプリケーションの開発	10
マルチデバイス戦略	14
開発の前提条件	18
Database.com と Force.com	18
Force.com へのサインアップ	18
Database.com へのサインアップ	19
サポートされるブラウザ	20
では、始めましょう	21
第 2 章: モバイルアプリケーションの認証、セキュリティ、ID	23
OAuth2 認証フロー	24
OAuth 2.0 ユーザエージェント認証フロー	24
OAuth 2.0 トークン更新フロー	26
scope パラメータの値	26
ID URL の使用	27
OAuth トークンの取り消し	32
リモートアクセスアプリケーションの作成	33
第 3 章: 接続済みアプリケーション	35
接続済みアプリケーションの開発と管理	36

開発者の作業	36
接続済みアプリケーションの作成	36
接続済みアプリケーションの基本情報	37
接続済みアプリケーションの API 統合	38
接続済みアプリケーションのモバイル統合	39
接続済みアプリケーションの IP 範囲	39
接続済みアプリケーションの公開	40
接続済みアプリケーションの削除	41
接続済みアプリケーションの更新	41
管理者の作業	41
接続済みアプリケーションのインストール	41
接続済みアプリケーションの管理	42
PIN コード保護について	45
接続済みアプリケーションのアンインストール	45
接続済みアプリケーションのアップグレード	46
接続済みアプリケーションのエラーコード	46
第 4 章: ネイティブ iOS アプリケーションの開発	47
ネイティブ iOS アプリケーションのクイックスタート	48
ネイティブ iOS の開発要件	48
Mobile SDK for iOS のインストール	48
Xcode でのネイティブ iOS アプリケーションの作成	48
Xcode プロジェクトテンプレートアプリケーションの実行	49
既存のプロジェクトへの Mobile SDK の統合	49
iOS 用の RestAPIExplorer サンプルアプリケーション	50
第 5 章: ネイティブ Android アプリケーションの開発	51
ネイティブ Android アプリケーションのクイックスタート	52
ネイティブ Android の開発要件	52
Mobile SDK for Android のインストール	53
Android プロジェクトの作成	53
Android テンプレートアプリケーション	54
Eclipse でのプロジェクトの設定	54
Android プロジェクトファイル	55
Eclipse によるクリーンとビルド	55
Android 用の RestExplorer サンプルアプリケーション	55

第 6 章: ハイブリッドアプリケーションの開発.....	57
ハイブリッドアプリケーションのクイックスタート	58
ハイブリッドアプリケーションの開発要件	58
iOS 向けハイブリッドアプリケーションプロジェクトの作成.....	59
Android 向けハイブリッドプロジェクトの作成.....	59
サンプルハイブリッドアプリケーションの実行	59
サンプルアプリケーションの仕組み.....	62
一覧情報を表示するモバイルページの作成	63
詳細情報を表示するモバイルページの作成	67
Chatter によるソーシャルコラボレーション	69
アプリケーションのビュー の変更 (index.html)	70
アプリケーションのコントローラの変更 (inline.js).....	70
アプリケーションの動作確認.....	74
iOS 用のハイブリッドサンプルアプリケーション	75
Android 用のハイブリッドサンプルアプリケーション	75
第 7 章: Mobile Components for Visualforce ライブラリを使用したハイブリッド開発	77
Mobile Components for Visualforce のアーキテクチャ	78
Visualforce のオープンソースモバイルコンポーネント	78
Visualforce App コンポーネント	79
Visualforce Navigation コンポーネント	79
Visualforce SplitView テンプレート	79
Visualforce Page コンポーネント	80
Visualforce Header コンポーネント	80
Visualforce Content コンポーネント	80
Visualforce List コンポーネント	81
Visualforce Detail コンポーネント	81
Visualforce Footer コンポーネント	81
コンポーネントのインストール	81
タブレットページの作成	83
容易な統合	85
Visualforce モバイルコンポーネントの作成	86
第 8 章: HTML5 アプリケーションの開発.....	97
HTML5 の開発要件	98
JavaScript を使用したデータアクセス	98

第 9 章: オフラインでのデータの安全な格納.....	101
ハイブリッドアプリケーションでの SmartStore へのアクセス.....	102
ハイブリッドアプリケーションのオフライン開発	102
MockSmartStore の使用.....	102
Soup の登録.....	104
Soup からのデータの取得.....	106
カーソルの操作	108
データの操作.....	109
SmartStore の拡張	112
第 10 章: 上級レベルのトピック	113
カメラを使用するためのハイブリッドサンプルアプリケーションのカスタマイズ..	114
アプリケーションの実行.....	115
デモアプリケーションの動作の仕組み.....	117
バーコードと QR コードのスキャニング	119
地理位置情報とモバイルアプリケーション	122
ハイブリッドアプリケーションでの NFC の活用.....	123
NFC の要件.....	124
Force.com と NFC モバイルアプリケーションのアーキテクチャ.....	124
NFC PhoneGap プラグインのインストール.....	124
JavaScript による NFC プラグインの起動.....	125
Mobile SDK を使用した Force.com への情報の更新/挿入 (upsert)	127
まとめ - モノのインターネットと NFC の未来.....	128
第 11 章: AppExchange を利用したモバイルアプリケーションの配信.....	129
AppExchange for Mobile: エンタープライズ向けモバイルアプリケーション	130
AppExchange パートナープログラムへの参加.....	131
配信組織の作成	132
プロバイダプロファイルの作成	133
AppExchange のセキュリティレビュー.....	134
第 12 章: リファレンス	137
REST API リソース	138
iOS アーキテクチャ	139
ネイティブ iOS オブジェクト	140
Android アーキテクチャ	141

用語集.....	147
索引.....	161

はじめに

5 MHz、16 ビットのプロセッサを搭載し MS-DOS を実行する IBM PC がリリースされたのは、1981 年のことです。その 30 年後の 2012 年には、1 GHz デュアルコアプロセッサを搭載した Apple iPhone 4S がリリースされました。ほんの数十年で、ハードウェアとソフトウェアのパフォーマンスが 1000 倍になり、ポケットに入れて持ち運べるようになったのですから驚きです。今日のインターネット人口は 23 億人、携帯電話のユーザ数は 61 億人にも達しています。

携帯電話に代表されるモバイルデバイスの普及により、生活や仕事の進め方は大きく変わりました。たとえば、ソーシャルネットワークやアプリケーションを利用して、顧客や同僚とコミュニケーションを取ったり、業務に関する情報をやり取りしたりすることができます。さまざまなデバイスでデータを取得、作成、共有することも簡単です。

ほとんどの企業は業務用アプリケーションを採用していますが、その大半はモバイルデバイスに対応していません。このため社員は、人事アプリケーション、ERP アプリケーション、イントラネットアプリケーション、その他のカスタムアプリケーションを必要とするときに利用できないことがあります。こうした従来のアプリケーションは使い勝手が悪く、コンシューマ向けアプリケーションのようにソーシャルグラフにも組み込まれていません。

従来のプラットフォームは、モバイル時代の変化に対応していません。

多くの企業がアプリケーション開発に利用してきた従来のプラットフォームは、モバイル時代のニーズに対応していません。大規模で融通の利かないアプリケーションスタックや、柔軟性に欠ける統合スタイルは、もう過去のものなのです。企業は急速にクラウドアプリケーションへの移行を進めています。

企業向けモバイルアプリケーションには、1990 年代に生まれた PC 向け Web アプリケーションの技術とは異なる、まったく新しいアーキテクチャとアプリケーション設計が必要となります。それを実現するのが、モバイルアプリケーション開発のニーズを踏まえた新しいプラットフォームです。

表 1: PC 向け Web アプリケーションとモバイルアプリケーションの比較

評価項目	従来の PC 向け Web アプリケーション	最新モバイルアプリケーション
接続、可用性	<ul style="list-style-type: none"> 高速で信頼性の高い LAN 低レイテンシ 高帯域幅 オンライン接続が前提 	<ul style="list-style-type: none"> さまざまな接続方式 高レイテンシ 低帯域幅 オフライン操作は必須
ユーザの操作	<ul style="list-style-type: none"> キーボード、マウス デスクトップでの長時間利用 	<ul style="list-style-type: none"> タッチスクリーン 短い集中的なアクション
ネットワークセキュリティ	<ul style="list-style-type: none"> 企業 VPN や LAN からアプリケーションにアクセス 	<ul style="list-style-type: none"> モバイルデバイスから企業 VPN にアクセスするのは面倒 パブリックモバイルネットワークでは IP 制限機能は無効
デバイスの標準化	<ul style="list-style-type: none"> 主に IT 部門が購入、管理 	<ul style="list-style-type: none"> 個人所有デバイスの利用が一般的 複数のプラットフォーム
フォームファクタ	<ul style="list-style-type: none"> PC の大型ディスプレイ 	<ul style="list-style-type: none"> 携帯電話、タブレット、PC に対応したアプリケーションが必要
ソーシャルメディア	<ul style="list-style-type: none"> サイロ化したアプリケーション メールによるコラボレーション 	<ul style="list-style-type: none"> ユーザ同士のコラボレーションが容易 直感的なデータの共有、コラボレーションが可能
マルチデバイス	<ul style="list-style-type: none"> サーバ (Web) にデータを格納するクライアントサーバーキテクチャ 	<ul style="list-style-type: none"> デバイス間でのデータ共有が容易 デバイス間のデータ転送が可能
デバイスの利用	<ul style="list-style-type: none"> 電話通信、カメラなどはほとんど使用しない 	<ul style="list-style-type: none"> モバイルデバイスのカメラ、連絡先、カレンダー、位置情報サービスなどを活用
位置情報	<ul style="list-style-type: none"> ほとんど使用しない 	<ul style="list-style-type: none"> データと位置情報を関連付けるため、および位置情報を基に使用可能なデータやサービスを絞り込むために使用

Salesforce Touch Platform

企業の IT 部門は、モバイルデバイスから社内のデータやサービスを利用できるようになるという難しい課題に直面しています。実はこれは、セールスフォース・ドットコムがエンタープライズ向け CRM およびサービスアプリケーションをモバイル化する際に直面した課題でもあります。この課題を解決するには、従来のテクノロジを根本から見直し、エンタープライズクラスの信頼性、可用性、セキュリティを実現しながら、複数のプラットフォーム (iPad、Android、iPhone)、複数のデバイス (スマートフォン、タブレット、PC) で Salesforce アプリケーションを利用できるようにする必要がありました。セールスフォース・ドットコムでは、この実践から得た知識やテクノロジをお客様にも提供したいと考えています。

Salesforce Touch Platform は、モバイルアプリケーションの課題に対応した初のエンタープライズ向けプラットフォームです。

Salesforce Touch Platform は、Salesforce のモバイルアプリケーションの基盤となる次世代のプラットフォームです。このプラットフォームを利用して、iPad、Android、iPhone 向けの独自のエンタープライズアプリケーションを開発できます。最新のアジャイル開発手法を取り入れ、Salesforce プラットフォームの特長であるエンタープライズアプリケーションに適したセキュリティ、信頼性、拡張性を実現しています。

Salesforce Touch Platform は、次の主要コンポーネントで構成されています。

- Force.com for Touch
- モバイルコンテナ (Salesforce Mobile SDK 1.3)
- Salesforce Identity

Force.com for Touch

Force.com for Touch は、エンタープライズモバイルアプリケーションの開発と管理に特化した、Salesforce Platform の新しいサービスです。

- Mobile REST API により、標準 Web プロトコルを使用して企業のデータやサービスにアクセスできます。開発者はビジネスデータを REST API として迅速に公開し、さまざまなスマートフォン、タブレット、Web ユーザインターフェースで、共通の API として利用できます。あらゆる種類のデバイスに関するアクセス、セキュリティ、共通のポリシー、プロセスの管理を一元的に行うことができます。
- 開発者は、ソーシャル化を実現する Chatter REST API を利用して、アプリケーションにソーシャルネットワーク機能とコラボレーション機能を迅速に組み込むことができます。Chatter REST API は、フィードだけでなく、ユーザのつながりを示すソーシャルグラフへのアクセスも提供します。これにより、モバイルアプリケーションから容易に、ユーザやグループと情報の受け渡しをしたり、ソーシャルグラフを利用してユーザ間のコラボレーションを即座に実現したりすることができます。
- モバイルポリシー管理機能により、ファイアウォールなどの境界セキュリティを利用せずに、モバイルアプリケーションにエンタープライズセキュリティポリシーを適用できます。二要素認証、デバイスの PIN コード保護、パスワードローテーションなどのセキュリティ機能の有効化や、モバイルアプリケーションへのユーザアクセス制御も可能です。

- 地理位置情報機能により、オンラインビジネスプロセスに位置データを付加できます。Winter' 13 リリース以降、Salesforce のすべてのオブジェクトに地理位置情報の項目が追加されます。これにより、プラットフォーム全体が位置情報に対応し、半径をベースとした近隣地の検索など、位置情報検索が可能になります。

モバイルコンテナ (Salesforce Mobile SDK 1.3)

モバイルコンテナは、iOS 向けネイティブアプリケーション (Objective-C) や Android 向けネイティブアプリケーション (Java) の開発に利用できるほか、HTML5 ベースのハイブリッドアプリケーションのネイティブコンテナとしても利用できます。iOS 向けウィザードや Android 向けツールが用意されており、ネイティブアプリケーションやハイブリッドアプリケーションの開発も簡単に始められます。Salesforce Mobile SDK で実装されるモバイルコンテナでは、次のような機能を提供します。

- ネイティブデバイスサービス: 開発者は、iPhone、iPad、Android をはじめとする多種多様なデバイスに対応する共通の API を使用して、カメラ、位置情報、連絡先の機能をアプリケーションに簡単に追加できます。
- 安全なオフラインストレージ: ネットワーク接続が限られた環境や利用できない環境でも機能するアプリケーションを開発できます。デバイスに保存されたデータは暗号化されているため、デバイスの盗難、紛失時にも安心です。
- クライアント OAuth 認証: モバイルアプリケーション用のログインページや認証処理を開発する手間が省けます。エンタープライズ向けのセキュリティ管理を、モバイルアプリケーションに迅速かつ簡単に組み込むことができます。

Salesforce Identity

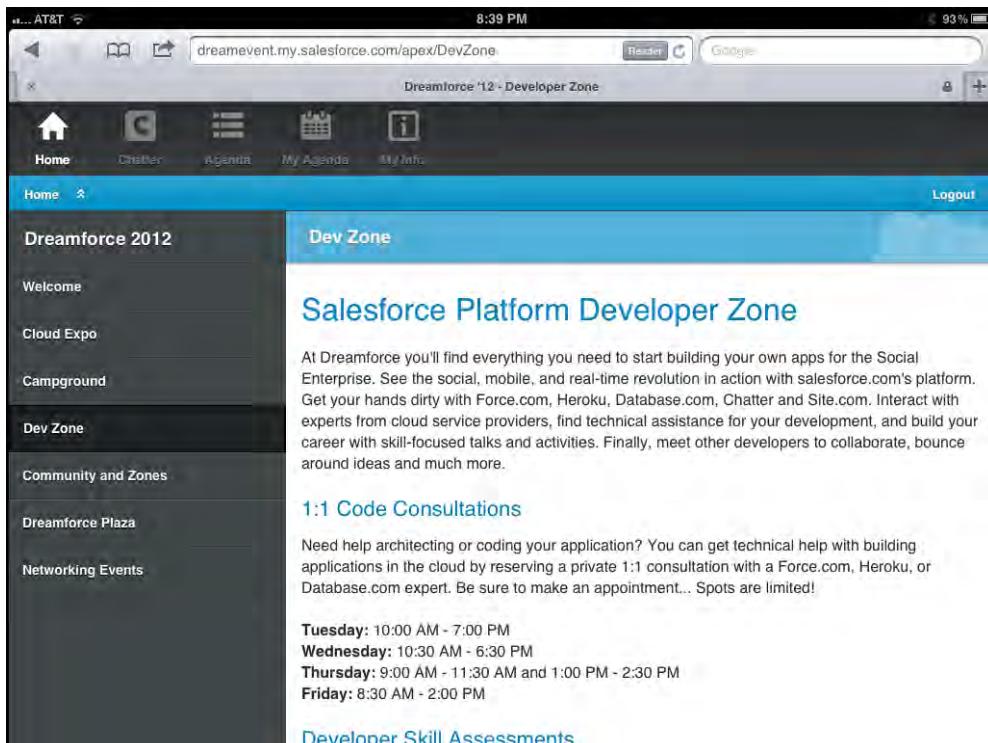
Salesforce Identity では、単一のエンタープライズ ID を発行することにより、モバイルデバイスからエンタープライズデータやサービスへのシングルサインオンを実現します。次のようなメリットがあります。

- 複数のアプリケーションやデバイスをシングルサインオンで利用できるため、複数のユーザ名やパスワードを管理する必要がありません。
- エンタープライズ向けのプラットフォームやアプリケーションに適した信頼性の高い ID を提供できます。
- クラウドディレクトリにより、カスタムのアイデンティティサービスや、企業固有のデザイン、ブランディングの提供が可能です。
- Facebook のようなコンシューマ向けアイデンティティサービスを利用して、顧客対応アプリケーションに顧客のソーシャルデータを迅速に取り込むことができます。

Dreamforce アプリケーション

Dreamforce カンファレンスで使用する Dreamforce アプリケーションは、PC のブラウザ、タブレット、各種モバイルデバイスからの数千のアクセスに対応できなければなりません。1 週間にわたるカンファレンス期間中には、モバイルデバイスからのリアルタイムアクセス数は数万にも達します。このため、モバイルファーストでの開発が必要でした。セールスフォース・ドットコムは、このドキュメントで紹介するテクノロジを活用して、Salesforce Touch Platform 上で Dreamforce 2012 アプリケーションを開発しました。

Dreamforce 2012 アプリケーションには、Dreamforce の開催前はブラウザやタブレットから、Dreamforce の開催中はスマートフォンからアクセスします。どちらの場合も、ユーザは同じ情報を取得できます。これは、Dreamforce アプリケーションが Responsive デザインを採用し、複数のディスプレイに対応しているためです。



Responsive デザインでは、使用するデバイスの種類を問わず最高のユーザエクスペリエンスが得られます。PC のブラウザでも、タブレットでも、その他のモバイルデバイスでも、サイズ変更、パンニング、スクロールの手間を最小限に抑えてアプリケーションを表示、利用できます。同時に、使用するデバイス固有の機能も活用できます。これについては、複数のデバイスから Dreamforce アプリケーションにアクセスしたことがあるお客様なら、もうご存知でしょう。

このドキュメントについて

このドキュメントでは、Salesforce Touch Platform の概要と、クラウド向けモバイルアプリケーションの設計、開発、管理方法について説明します。



メモ: このドキュメントのオンライン版は、Developer Force サイト (http://wiki.developerforce.com/page/Force.com_Books) からダウンロードできます。

章の構成

このドキュメントは、アプリケーションの実際の開発手順に沿って構成されています。アーキテクチャの決定とセキュリティに関する重要事項を確認してから、開発に入っていきます。

最初にネイティブアプリケーションの開発手順を見ていきますが、これはネイティブアプリケーションのほうが優先されるからではなく、ネイティブアプリケーションの開発の設定をハイブリッドアプリケーションの開発に応用できるからです。要するに、説明のしやすい構成となっているだけです。ネイティブアプリケーションの開発の設定さえできれば、後は難しくありません。

続いて、データをオフラインでキャッシュする方法、カメラにアクセスする方法、ソーシャルコラボレーションを実現する方法など、やや複雑なタスクについて説明します。最後に、開発したモバイルアプリケーションを AppExchange 上で公開するためのパートナープログラムについて紹介します。

バージョン

このドキュメントの最終改訂は 2012 年 9 月 7 日です。本バージョンは、Salesforce Winter' 13 リリースと Mobile SDK バージョン 1.3 に対応しています。

ご意見、ご質問について

このドキュメントの内容に関するご意見やご質問、今後取り上げてほしいトピックのご提案などについては、Force.com のディスカッションボード (<http://boards.developerforce.com>) までお寄せください。メールでも受け付けています (developerforce@salesforce.com)。

最新情報

常に最新の情報を把握しておくため、次の点に留意してください。

- 新しいバージョンの Salesforce がリリースされると、リリースノートのモバイルのセクションに重要事項が掲載されます。
- このドキュメントは頻繁に更新されています。最新版は http://wiki.developerforce.com/page/Force.com_Books にて確認できます。このドキュメントを紙ベースで参照している場合は、「バージョン」(ページ 6) でバージョンを確認してください。
- 最新の Mobile SDK は、Mobile SDK Git Hub リポジトリに格納されています。
- 最新の記事、ブログ投稿記事、チュートリアル、オンラインセミナーの情報については、<http://wiki.developerforce.com/page/Mobile> を参照してください。
- ディスカッションボード (<http://boards.developerforce.com/t5/Mobile/bd-p/mobile>) への参加も歓迎します。

Mobile SDK GitHub リポジトリ

Mobile SDK 開発チームは、GitHub を使用して <https://github.com/forcedotcom> で Mobile SDK のソースコードを管理しています。

Mobile SDK の最新リリースは、GitHub の公開リポジトリで公開されています。

- <https://github.com/forcedotcom/SalesforceMobileSDK-iOS>
- <https://github.com/forcedotcom/SalesforceMobileSDK-Android>



メモ: 最新記事、ブログ投稿記事、チュートリアル、オンラインセミナーの情報については、Mobile SDK ホームページ (http://wiki.developerforce.com/page/Mobile_SDK) を参照してください。

第1章

Salesforce Touch Platform でのモバイル開発の概要

トピック:

- ネイティブアプリケーション、HTML5 アプリケーション、ハイブリッドアプリケーションの開発
- マルチデバイス戦略
- 開発の前提条件
- サポートされるブラウザ
- では、始めましょう

Force.com プラットフォームは、シンプルで使いやすく生産性の高い、クラウドコンピューティングに最適のプラットフォームです。マウスで操作できる Web インターフェースツールを使って、カスタムオブジェクト、カスタム項目、ワークフロールール、Visualforce ページ、Apex クラス、Apex トリガなどのアプリケーションコンポーネントを定義し、組み合わせることにより、非常に魅力的で洗練されたアプリケーションを開発することができます。モバイル開発者ももちろん、この利点を享受できます。

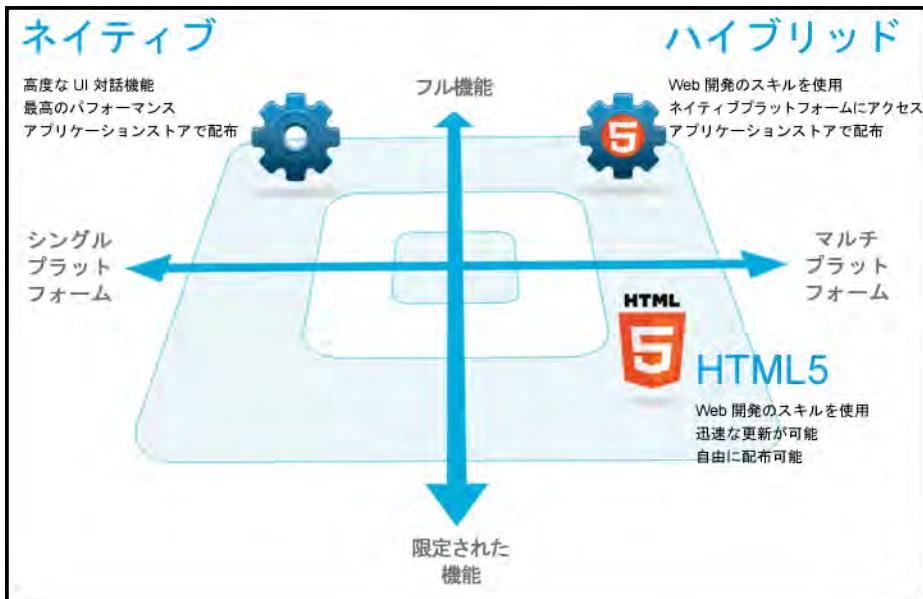
Salesforce Touch Platform には、Force.com のクラウドアーキテクチャにシームレスに統合可能なネイティブまたはハイブリッドモバイルアプリケーションをすばやく開発するために欠かせないライブラリのほか、開発の簡素化に役立つ次の機能が用意されています。

- デバイスアクセス
- ハイブリッドアプリケーション用のエンタープライズ コンテナ
- 地理位置情報サービス
- HTML5 開発
- ネイティブの REST API ラッパー
- OAuth アクセストークン管理
- 安全なオフラインストレージ
- ソーシャル & モバイル API

ネイティブアプリケーション、HTML5 アプリケーション、ハイブリッドアプリケーションの開発

モバイル戦略には、チームの開発スキル、デバイスの機能、セキュリティの重要性、オフライン機能、相互運用性など、多くの要素が関わっています。つまり、アプリケーションの機能だけでなく、それを実現するプロセスが重要なのです。Salesforce Touch Platform では、次の 3 種類のモバイルアプリケーションを開発できます。

- **ネイティブアプリケーション:** 特定のモバイルプラットフォーム (iOS、Android) に対応したアプリケーション。そのプラットフォームでサポートされる開発ツールと言語 (iOS なら Xcode と Objective-C、Android なら Eclipse と Java) を使って開発します。外観、パフォーマンスの面ですぐれています。
- **HTML5 アプリケーション:** アプリケーションストアではなく、Web サーバから各デバイスに配布され、モバイル Web ブラウザで動作するアプリケーション。アプリケーションの配布には、HTML5、JavaScript、CSS などの標準 Web テクノロジを使用します。「一度書けばどのプラットフォームでも実行できる」という特長により、複数のデバイスで動作するクロスプラットフォームモバイルアプリケーションを開発できます。HTML5 と JavaScript のみを使用して洗練されたアプリケーションを作成できる反面、セッション管理、安全なオフラインストレージ、デバイスのネイティブ機能へのアクセスに関しては重大な制限が課されます。
- **ハイブリッドアプリケーション:** Salesforce モバイルコンテナに Web アプリケーションをラップすることにより、HTML5 開発の利便性とネイティブプラットフォームの強力な機能を両立させたアプリケーション。デバイスのネイティブ機能の利用が可能で、アプリケーションストアから配布できます。ハイブリッドアプリケーションは Visualforce ページを使って開発することもできます。本ドキュメントでは、両方の開発手法を紹介します。



ネイティブアプリケーション

ネイティブアプリケーションは、操作性、機能、全体的なモバイルエクスペリエンスの面で最もすぐれています。ネイティブアプリケーションでしか利用できない次のような機能もあります。

- **マルチタッチ** — ダブルタップ、ピンチ、スプレッドなど、2本以上の指を使ったUI操作が可能です。
- **高速グラフィック API** — ネイティブプラットフォームでは、非常に高速なグラフィック処理が可能です。要素数の少ない静的な画面を表示する場合は関係ありませんが、データの量が多く、スピーディな画面更新が必要な場合には、この機能が重要になります。
- **滑らかなアニメーション** — 高速グラフィック API と関連する機能。滑らかなアニメーションを実現できます。ゲーム、インタラクティブなレポート、写真や音声の変換に使用する計算量の多いアルゴリズムで重視されます。
- **組み込みコンポーネント** — カメラ、アドレス帳、地理位置情報などのネイティブデバイス機能を、モバイルアプリケーションにシームレスに統合できます。重要な組み込みコンポーネントの1つに暗号化されたストレージがありますが、これについては後述します。
- **使いやすさ** — 慣れ親しんだネイティブプラットフォームの操作性と、ユーザが期待するネイティブ機能を組み合わせることで、非常に使いやすいアプリケーションを実現できます。

通常、ネイティブアプリケーションの開発には、開発、デバッグ、プロジェクト管理、バージョン管理など、プロの開発者が必要とするツールが揃った統合開発環境 (IDE) を使用します。これは、ネイティブアプリケーションの開発がやや難しく、他の開発シナリオより高いレベルの経験が必要になるからです。プロの開発者の方であれば、実績ある API やフレームワーク、既存のコンポーネントを使って特殊効果を簡単に実装する機能、コードを一元管理する機能の重要性は十分にご存知のことでしょう。

HTML5 アプリケーション

HTML5 モバイルアプリケーションは、基本的に、小さな画面向けに設計された単一、または一連の Web ページです。デバイスの制約を受けずに、あらゆるモバイルプラウザで実行することができます。コンテンツが Web 上に置かれるため、検索がしやすく、ショッピングアプリケーションなどに最適です。

ネイティブアプリケーションやハイブリッドアプリケーションに比べて技術的な難易度は低めで、初めてモバイル開発に挑戦する場合にも取り組みやすい開発手法ですが、デバイスごとに画面サイズや画面解像度が異なるため、さまざまなデバイスでテストを実施する必要があります。特に、Android デバイスでは、プラウザの互換性に関する問題が発生することが多いため、注意が必要です。

「一度書けばどのプラットフォームでも実行できる」を特長とする HTML5 開発は、ネイティブアプリケーションより配布やサポートが容易で、バグの修正や新機能の追加をすばやく一気に配信することができます。これに対しネイティブアプリケーションは、開発からテストまでのサイクルが長く、最新のアプリケーションやバグ修正がリリースされた場合、ユーザ自身がストアにログインしてダウンロードする必要があります。

開発、サポートが容易で、幅広いデバイスに対応する HTML5 アプリケーションですが、まったく欠点がないわけではありません。

- **オフラインストレージ** — ファイルをデバイスにキャッシュすることで、オフライン機能に近いものを実現することは可能ですが、たとえ基盤となるデータベースが暗号化されていたとしても、開発者の証明書で個々のアプリケーションを保護するネイティブのキー chaining アクセスほど高レベルなセグメント化は実現できず、安全性の面で劣ります。
- **セキュリティ** — HTML5 モバイルアプリケーションでは多くの場合、ネイティブプラットフォームにセキュリティ対策を実装すると、Web 開発者とユーザ双方の負担を増やすこととなります。たとえばユーザは、デスクトップから認証付き Web アプリケーションを起動した場合、アプリケーションがバックグラウンドに切り替わるたびにユーザ情報を入力する必要があります。
- **ネイティブ機能** — カメラ、アドレス帳、その他のネイティブ機能にアクセスできません。
- **ネイティブと同じ外観、操作性** — HTML5 は、ネイティブの外観をエミュレートしますが、ネイティブのような 2 本以上の指を使った操作には対応しません。

ハイブリッドアプリケーション

ハイブリッドアプリケーションの開発には、ネイティブプラットフォーム機能へのアクセスを提供するコンテナにラップされた HTML5 と JavaScript を使用します。一般的に、ハイブリッドアプリケーションは、HTML5 アプリケーションの開発のしやすさとネイティブアプリケーションの機能性を兼ね備えています。

ネイティブアプリケーションはデバイスにインストールされ、HTML5 アプリケーションは Web サーバ上で保管されますが、ハイブリッドアプリケーションではそのどちらも可能で、次のように 2通りの実装方法から選択できます。

- ローカル** — ネイティブアプリケーションの構築と同様に、モバイルアプリケーション バイナリ内に HTML コードと JavaScript コードをパッケージ化します。デバイスとクラウド間のデータのやりとりには、REST API と Ajax を使用します。
- サーバ** — Web アプリケーションのフル機能をサーバから実装することも可能です (オプションでキャッシュを利用するとパフォーマンスが向上します)。コンテナアプリケーションがサーバからアプリケーション全体を取得し、ブラウザウィンドウに表示します。Visualforce ハイブリッドアプリケーションはこの方法で実装されます。

本ドキュメントでは、章を分けて両方の開発方法を紹介します。

ネイティブアプリケーション、HTML5 アプリケーション、ハイブリッド アプリケーションの概要

次の表に、3種類のモバイル開発シナリオの違いを示します。

	ネイティブ	HTML5	ハイブリッド
グラフィック	ネイティブの API	HTML、Canvas、SVG	HTML、Canvas、SVG
パフォーマンス	高速	低速	低速
外観と操作性	ネイティブ	エミュレーション	エミュレーション
配布形態	アプリケーションストア	Web	アプリケーションストア
カメラ	可	不可	可
通知	可	不可	可
連絡先、カレンダー	可	不可	可
オフラインストレージ	安全なファイルシステム	共有 SQL 領域	安全なファイルシステム、共有 SQL 領域
地理位置情報	可	可	可
スワイプ	可	可	可
ピンチ、スプレッド	可	不可	可
接続	オンライン、オフライン	ほぼオンライン	オンライン、オフライン
開発スキル	Objective-C、Java	HTML5、CSS、JavaScript	HTML5、CSS、JavaScript

マルチデバイス戦略

モバイルデバイスが普及した「ポスト PC 時代」のアプリケーションは、さまざまなプラットフォーム、フォームファクタ、デバイス機能をサポートする必要があります。Force.com を使用してデバイスに依存しないアプリケーションを開発するときには、次の点を考慮してください。

- どのようなデバイスとフォームファクタをサポートするか
- どのようにしてデバイスのタイプを検出するか
- どのようにして複数タイプのデバイスをサポートする Force.com アプリケーションを設計するか

どのようなデバイスとフォームファクタをサポートするか

この考慮事項に対する回答は、ユースケースとエンドユーザーの要件によって異なります。しかし、サポートする必要があるデバイス、プラットフォーム、フォームファクタについて考える時間をとることは大切です。「すべてのプラットフォーム、デバイス、フォームファクタをサポートする」のか、あるいは「デスクトップと iPhone だけをサポートする」のか決定することで、残りの 2 つの考慮事項に対する回答が見えてきます(これは一例に過ぎません)。

この決定には重要なトレードオフがあります。複数のフォームファクタをサポートすれば、アプリケーションのリーチが拡大するのは明らかです。しかし、その一方で、アプリケーションの開発や長期間にわたる管理が複雑になります。

クロスデバイスアプリケーションの開発とは、単に複数のフォームファクタやデバイス(たとえばデスクトップ、スマートフォン、タブレット)で Web ページの表示を最適化することではありません。デバイスまたはフォームファクタごとに、ユーザエクスペリエンスを検討し、カスタマイズする必要があります。たとえば、スマートフォンやタブレット向けのアプリケーションには、デスクトップ向けの Web ページでサポートされる便利な機能(ファイルのアップロードや、多くのクリック操作が必要なユースケースに対応する機能)は不要な場合がほとんどです。

それよりも、地理位置情報サービス、カメラなど、デスクトップ環境にはない機能が必要になります。LinkedIn、Flipboard のようなすぐれたデザインのアプリケーションでは、スマートフォン向けとタブレット向けの機能もそれぞれ異なります(例: タブレット向けでは水平ナビゲーション、スマートフォン向けでは片手で操作できる垂直スクロールなど)。touch.salesforce.com でも、フォームファクタ別にユーザエクスペリエンスがカスタマイズされています。アプリケーションでどのようなデバイスやフォームファクタをサポートするかを決定するときは、このようなきめ細かな対応と、それに要する時間とコストを考慮する必要があります。

サポートするデバイスを決めたら、次に、特定のユーザがどのデバイスから Web アプリケーションにアクセスしているかを検出する必要があります。

クライアントサイドのデバイス検出

クライアントサイドの検出アプローチでは、クライアントのブラウザ上で動作する JavaScript(または CSS メディアクエリ)を使ってデバイスタイプを特定します。次の 2 通りの方法でデバイスタイプを検出できます。

- ユーザエージェント (User-Agent) ヘッダーによる特定 — JavaScript を使って、User-Agent という HTTP ヘッダーを解析し、その情報にもとづいてデバイスタイプを特定します。このために独自の JavaScript を作成することも可能ですが、既存の JavaScript を再利用することをお勧めします。インターネットを検索すると、User-Agent ヘッダーからデバイスタイプを検出できる再利用可能な JavaScript コードが多数見つかります。しかし、それと同時に、この方法の危険性にも気付かされることでしょう。使用される User-Agent の数は非常に多く、現在も増え続けているため、デバイスの検出に User-Agent ヘッダーを使用する方法は、やや信頼性に欠けると考えられます。
- 画面サイズやデバイス機能による特定 — JavaScript で User-Agent 文字列を取得する方法よりお勧めなのは、デバイスの画面サイズや機能 (タッチ機能など) からデバイスタイプを特定する方法です。たとえば、Visualforce で開発されたオープンソースの HTML5 モバイルアプリケーション Contact Viewer HTML5 は、この方法でデバイスタイプを特定します。MobileAppTemplate.page のページ上部には、スマートフォンとタブレットをデバイスの画面サイズで識別する単純な JavaScript コードが含まれています。このほかに、Device.js や Modernizr のようなライブラリを使ってデバイスタイプを検出する方法もあります。これらのライブラリは、CSS メディアクエリと機能検出 (タッチ機能など) の組み合わせを利用するため、より高い信頼性でデバイスタイプを検出できます。Modernizr ライブラリを使用する単純な例は、

<http://www.html5rocks.com/static/demos/cross-device/feature/index.html> を参照してください。Device.js ライブラリを使用し、Visualforce と連携する少し複雑な例については、GitHub リポジトリ <https://github.com/sbhanot-sfdc/Visualforce-Device.js> を参照してください。以下は、GitHub リポジトリ内の DesktopVersion.page の抜粋です。

```
<apex:page docType="html-5.0" sidebar="false" showHeader="false"
standardStylesheets="false" cache="false" >

<head>
    <!--Every version of your webapp should include a list of all
    versions. -->
    <link rel="alternate" href="/apex/DesktopVersion" id="desktop"
media="only screen and (touch-enabled: 0)"/>
    <link rel="alternate" href="/apex/PhoneVersion" id="phone" media="only
screen and (max-device-width: 640px)"/>
    <link rel="alternate" href="/apex/TabletVersion" id="tablet"
media="only screen and (min-device-width: 641px)"/>

    <meta name="viewport" content="width=device-width, user-scalable=no"/>

    <script src="{!URLFOR($Resource.Device_js)}"/>
</head>

<body>
    <ul>
        <li><a href="?device=phone">Phone Version</a></li>
        <li><a href="?device=tablet">Tablet Version</a></li>
    </ul>
    <h1> This is the Desktop Version</h1>
</body>
</apex:page>
```

アプリケーションがサポートするデバイスタイプごとに <link> タグを記述しておくと、Device.js ライブラリにより、検出されたデバイスタイプに対応する Visualforce ページにユーザが自動的にリダイレクトされます。?device=xxx というフォーマット（上記参照）を利用して、デフォルトの Device.js リダイレクトを上書きする方法もあります。

サーバサイドのデバイス検出

デバイスタイプをサーバで（Apex コントローラ、拡張クラスを使って）検出する方法もあります。サーバサイドのデバイス検出は、User-Agent HTTP ヘッダーを解析して行います。以下は、Visualforce ページが iPhone クライアントから参照されていることを検出するコードです。

```
<apex:page docType="html-5.0" sidebar="false" showHeader="false"
cache="false"
    standardStylesheets="false"
controller="ServerSideDeviceDetection"
    action="{!detectDevice}">
    <h1> This is the Desktop Version</h1>
</apex:page>

public with sharing class ServerSideDeviceDetection {
    public boolean isiPhone {get;set;}

    public ServerSideDeviceDetection() {
        String userAgent =
System.currentPageReference().getHeaders().get('User-Agent');
        isiPhone = userAgent.contains('iPhone');
    }

    public PageReference detectDevice(){
        if (isiPhone)
            return Page.PhoneVersion.setRedirect(true);
        else
            return null;
    }
}
```

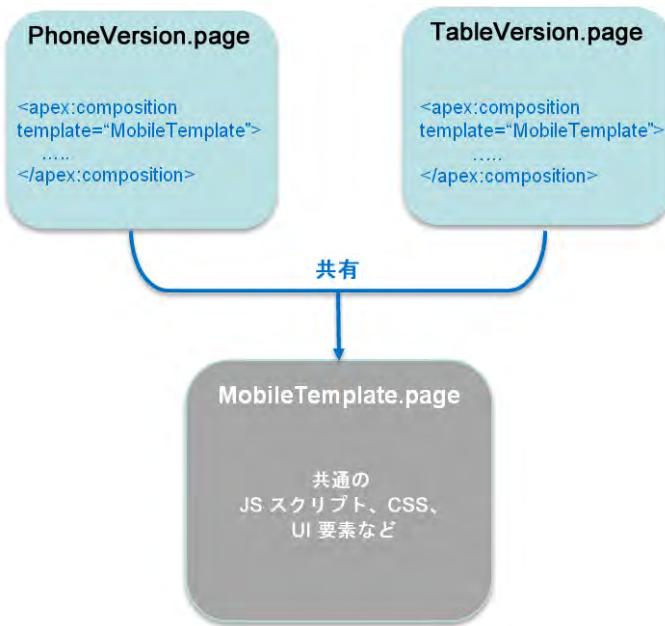
ここで行われている User-Agent 解析はすべてのデバイスをカバーするものではないため、実際には、サポート対象のすべてのデバイスを正規表現マッチに基づいて検出する、より強力な手法を実装する必要があります。まずは、detectmobilebrowsers.com サイトのコード内で使われている正規表現を調べてみることをお勧めします。

どのようにして複数タイプのデバイスをサポートする Force.com アプリケーションを設計するか

サポート対象のデバイスと、それらを検出する方法が決まつたら、最後に、デバイスまたはフォームファクタごとにカスタムのユーザエクスペリエンスを提供する最適なアプリケーション設計を決定します。ここでもいくつかの選択肢があります。

同じ Visualforce ページをさまざまなフォームファクタで適切に表示するだけでよいなら、Responsive デザインが適しています。この手法では、CSS3 のメディアクエリを使って、クライアントブラウザのフォームファクタに合うように動的にページを再フォーマットします。Twitter Bootstrap のような Responsive デザインフレームワークも利用できます。

特定のフォームファクタ向けの Visualforce ページを複数設計し、前のセクションで説明したような手法を使って適切なページにユーザをリダイレクトする方法もあります。別々の Visualforce ページを用意するといつても、重複するコードや機能をその都度記述するわけではありません。すぐれたアーキテクチャを持つソリューションでは、クライアントサイドでもサーバサイドでもコードを最大限に再利用できます。たとえばクライアントサイドでは、コンポーネント、テンプレートなどの Visualforce 機能を使用します。サーバサイドでは、共通のビジネスロジックを Apex クラスにカプセル化して、複数のページコントローラで呼び出せるようにします。こうした設計例は、前述のオープンソースの Contact Viewer アプリケーションで確認できます。このアプリケーションではスマートフォン用、タブレット用に別々のページ(スマートフォン用は ContactsAppMobile.page、タブレット用は ContactsApp.page)を用意していますが、これらは同じテンプレート(MobileAppTemplate.page)を共有しているため、コードとアーティファクトを最大限に再利用することが可能です。以下の図は、Contact Viewer アプリケーションの設計概念を示しています。



最後に紹介するのは、単一の Visualforce ページから複数のフォームファクタにサービスを提供する方法です。サーバサイドのデバイス検出を行い、ほとんどの Visualforce コンポーネントに含まれている rendered 属性(より直接的には、<div> タグの「display:none/block」CSS プロパティ)を利用することにより、ページ要素を選択的に表示または非表示することができます。ただしこの方法はコード量が多くなり、管理が難しいため、慎重に選択する必要があります。

開発の前提条件

Database.com または Force.com の利用経験があると、スムーズに開発を行うことができます。開発には、Database.com アカウントか Force.com Developer Edition 組織を使用します。さらに、次のプラットフォームおよびテクノロジを使用するため、これらに関する程度の知識を持っていることが前提です。

- iOS アプリケーションを開発するには、Mac OS X Snow Leopard または Lion と Xcode 4.2 以上が必要です。
- Android アプリケーションを開発するには、Java JDK 6、Eclipse、Android ADT プラグイン、Android SDK が必要です。
- ハイブリッドアプリケーションを開発するには、Visualforce を使用可能な組織が必要です。
- ほとんどのリソースは、ソーシャルコーディングコミュニティ GitHub 上にアップロードされています。セールスフォース・ドットコムの公開リポジトリにあるすべてのファイルには無料でアクセスできますが、MobileSDK のインストールスクリプトには git コマンドが使用されているため、Git 環境を構築しておくことをお勧めします (<https://github.com/forcedotcom>)。

Database.com と Force.com

本ドキュメントで開発するモバイルアプリケーションは、Database.com または Force.com 組織で動作します。各手順の説明は、便宜上、Force.com Developer Edition を使用していることを前提に書かれています。Force.com には、通常、login.salesforce.com などのログインページからログインします。Database.com を使用する場合は、Database.com のログインページから Database.com の資格情報を使ってログインしてください。



メモ: Database.com を使用する場合、Visualforce ベースのアプリケーションは開発できません。

Force.com へのサインアップ

1. ブラウザで <http://developer.force.com/join> にアクセスします。
2. ユーザ情報に入力します。
3. [Email Address] フィールドに、Web ブラウザから簡単にメールを確認できる、公開のメールアドレスを指定します。
4. [Username] フィールドに、メールアドレス形式の一意のユーザ名を指定します。前の手順で指定したメールアドレスと同じものにする必要はありません (通常は別のアドレスを指定したほうがよいでしょう)。このユーザ名は、developer.force.com へのログインに使用する ID となるため、職務 (develop@workbook.org など) や、名前 (firstname.lastname@example.com など) を含む形をお勧めします。

5. [Master Subscription Agreement] を読み、利用規約への同意を示すチェックボックスをオンにします。
6. Captcha を入力し、[Submit] をクリックします。
7. しばらくすると、ログイン用のリンクを記載したメールが届きます。リンク先でパスワードを変更してください。

Database.com へのサインアップ

1. ブラウザで www.database.com/jp にアクセスします。
2. [サインアップ] をクリックします。
3. ユーザ情報に入力します。
4. [メールアドレス] フィールドに、Web ブラウザから簡単に確認できる、公開のメールアドレスを指定します。
5. [ユーザ名] フィールドに、メールアドレスの形式でユーザ名を指定します。実際のメールアドレスである必要はありません。Database.com 組織の用途を示すユーザ名にすると便利です。このワークブックでは、admin-user@workbook.db を使用します。
6. Captcha を入力します。
7. 「主登録契約」と「Database.com 利用規約追補」を確認し、チェックボックスをオンにします。
8. [Sign Up] をクリックします。
9. アカウント確認用のメールが届いたら、メールに記載されたリンク先に移動します。
10. パスワードを設定し、パスワード確認用の質問と回答を入力します。

サポートされるブラウザ

Salesforce は次のブラウザをサポートします。

ブラウザ	説明
Microsoft® Internet Explorer® 7、8、9、10	<p>Internet Explorer を使用する場合は、最新バージョンを使用することをお勧めします。マイクロソフト提供の修正プログラムをすべて適用してください。次の制限事項があります。</p> <ul style="list-style-type: none"> Internet Explorer の互換表示モードはサポートされていません。 Internet Explorer 10 の Metro 版はサポートされていません。 <p>設定の推奨事項については、Salesforce オンラインヘルプの「Internet Explorer の設定」を参照してください。</p>
Mozilla® Firefox® の最新安定版	<p>セールスフォース・ドットコムでは、Firefox の最新バージョンについて、徹底したテストとサポートを実施しています。設定の推奨事項については、Salesforce オンラインヘルプの「Firefox の設定」を参照してください。</p>
Google Chrome™ の最新安定版	<p>Google Chrome では更新が自動的に適用されます。セールスフォース・ドットコムは、Google Chrome の最新バージョンについて、徹底したテストとサポートを実施しています。Chrome の設定に関する推奨事項はありません。Chrome は [コンソール] タブまたは [Add Google Doc to Salesforce] ブラウザボタンではサポートされていません。</p>
Microsoft® Internet Explorer® 6 および 7 用の Google Chrome Frame™ プラグイン	<p>Internet Explorer 6 および 7 のみでサポートされているプラグインです。Google Chrome Frame では、更新が自動的に適用されます。セールスフォース・ドットコムは、最新バージョンのみをサポートしています。設定の推奨事項については、Salesforce オンラインヘルプの「Microsoft® Internet Explorer® 用 Google Chrome Frame のインストール」を参照してください。Chrome Frame プラグインは Service Cloud コンソールまたは売上予測機能ではサポートされていません。</p>
Mac OS X で動作する Apple® Safari® バージョン 5.1.x	<p>Safari の設定に関する推奨事項はありません。iOS の Safari はサポートされていません。Salesforce CRM コールセンターの CTI ツールキットまたは Service Cloud コンソールでは、Safari はサポートされていません。</p>



メモ: セールスフォース・ドットコムでは、次のドメインを使用してコンテンツを配信します。標準的なインターネットアクセスが許可されている場合は、特に操作は必要ありません。ドメインをホワイトリストに登録する場合は、許可済みドメインのリストに追加する必要があります。サードパーティの Cookie をブロックしている場合、Salesforce を正しく機能させるには、「受け入れる」に変更する必要があります（主要なブラウザのデフォルトの設定は、通常、「受け入れる」になっています）。

- *.staticforce.com
- *.content.force.com
- *.force.com
- *.salesforce.com



重要: どのブラウザを使用する場合も、JavaScript、Cookie、SSL 3.0 を有効にする必要があります。

一部のサードパーティ Web ブラウザのプラグインや拡張機能は、Chatter の機能に干渉する可能性があります。Chatter が正しく機能しない場合や、整合性のない動作をする場合は、Web ブラウザのプラグインと拡張機能をすべて無効にしてから、もう一度試してみてください。

セールスフォース・ドットコムでは、ユーザの操作性を最大限に高めるため、1024 x 768 以上の画面解像度を使用することをお勧めしています。画面解像度が 1024 x 768 に満たない場合、レポートビルダーやページレイアウトエディタなどの Salesforce 機能の表示に問題が生じる可能性があります。

詳細については、個々の製品のドキュメントを参照してください。その他の要件については、Salesforce システム要件を参照してください。

では、始めましょう

詳細の確認は後にして、すぐ開発に入りたい場合は、ネイティブまたはハイブリッド開発シナリオのクイックスタートに進んでください。プラットフォーム別、開発シナリオ別に、単純なモバイルアプリケーションをダウンロードして実行できる『Mobile SDK Workbook』もぜひご利用ください。

- ネイティブ iOS アプリケーションのクイックスタート (ページ 48)
- ネイティブ Android アプリケーションのクイックスタート (ページ 52)
- ハイブリッドアプリケーションのクイックスタート (ページ 58)
- 『Mobile SDK Workbook』のダウンロード
(<https://github.com/forcedotcom/SalesforceMobileSDK-Samples>)

第 2 章

モバイルアプリケーションの認証、セキュリティ、ID

トピック:

- OAuth2 認証フロー
- リモートアクセスアプリケーションの作成

モバイルデバイスで実行するエンタープライズアプリケーションには、安全な認証が不可欠です。OAuth2 は、ユーザ名やパスワードを使わず、安全にユーザデータへのアクセスを認証できる業界標準のプロトコルです。その機能はバレットキーに例えられます。バレットキーとは、自動車の特定の部分のみ開けられる鍵のことです、トランクやグローブボックスを開けることはできません。

Salesforce のモバイルアプリケーションには、OAuth2 認証を簡単に組み込むことができます。OAuth2 を使用して HTML ページからユーザ名とパスワードを収集し、サーバに送信すると、セッショントークンが返され、今後の使用に備えてデバイス上に安全に保存されます。

OAuth2 認証フロー

認証フローはデバイスの認証状況により異なります。

初回認証のフロー

1. ユーザがモバイルアプリケーションを開きます。
2. 認証用のダイアログ、またはウィンドウやフロート表示が表示されます。
3. ユーザ名とパスワードを入力します。
4. アプリケーションがセッション ID を受信します。
5. ユーザがアプリケーションにアクセスを許可します。
6. アプリケーションが起動します。

継続認証

1. ユーザがモバイルアプリケーションを開きます。
2. セッション ID が有効な場合、アプリケーションはただちに起動します。セッション ID が無効な場合、アプリケーションは初回認証時の更新トークンを使用して、更新されたセッション ID を取得します。
3. アプリケーションが起動します。

PIN コード認証

1. ユーザがモバイルアプリケーションを開きます（前回の利用からしばらく時間が経過しているとします）。
2. 経過時間が、設定された PIN タイムアウト値を超えている場合、PIN 入力画面が表示されます。PIN を入力します。



メモ: PIN はモバイルポリシーの機能です。ユーザがオンラインかオフラインかにかかわらず、前回アプリケーションを使用してから所定の時間が経過した場合に表示されます。

3. アプリケーションは既存のセッション ID を再利用します。
4. アプリケーションが起動します。

OAuth 2.0 ユーザエージェント認証フロー

ユーザエージェント認証フローは、ユーザのモバイルデバイス上のクライアントアプリケーションで使用される、ユーザエージェントの同一生成源ポリシーを利用した認証です。

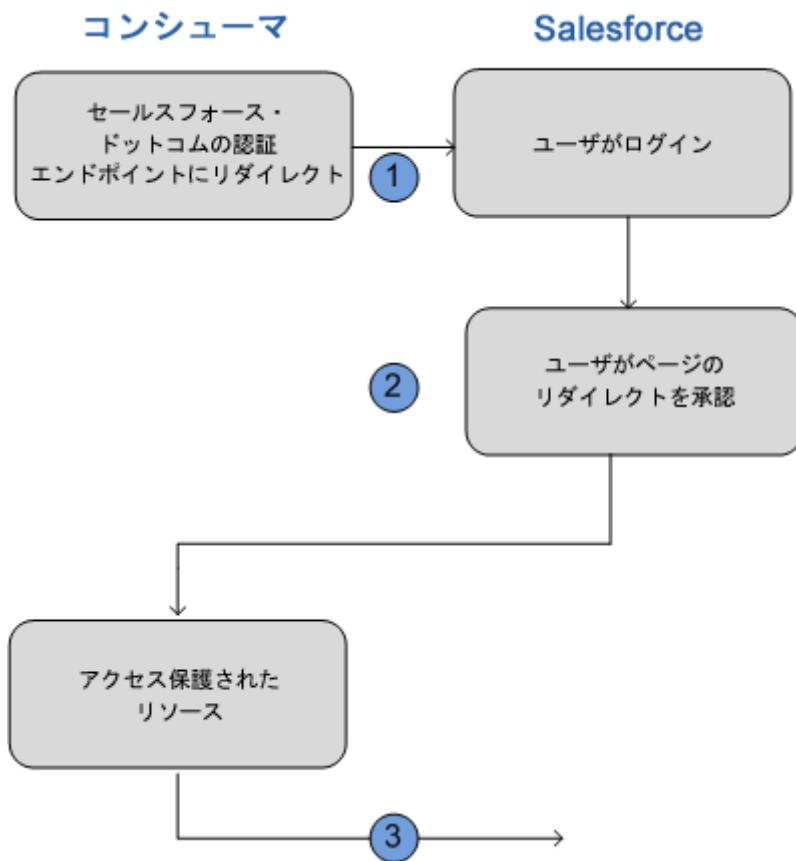
この認証フローでは、クライアントアプリケーションは HTTP リダイレクトの形式でアクセストークンを受信します。クライアントアプリケーションが、ユーザエージェントにアクセスできるローカルリソースまたは他の Web サーバにユーザエージェントをリダイレクトするよう認証サーバに要求すると、レスポンスからアクセストークンが抽出され、クライアントアプリケーションに渡されます。トークンレスポンスは、URL でハッシュ (#) フラグメントとして指定されます。これはセキュリティ保護のためであり、そのサーバだけでなく、参照ヘッダーで指定されている他のサーバにもトークンが渡されないようにします。

このユーザエージェント認証フローでは、クライアントの秘密は使用しません。これは、クライアントの実行可能ファイルがエンドユーザのコンピュータ上またはデバイス上にあり、クライアントの秘密にアクセスしたり、探索したりすることができるためです。



警告: アクセストークンはエンコードされ、リダイレクト URI になっているため、エンドユーザや、コンピュータ上またはデバイス上の他のアプリケーションに公開できます。

JavaScript を使用して認証する場合、`window.location.replace();` をコールして、ブラウザの履歴からコールバックを削除します。



1. クライアントアプリケーションでは、アプリケーションを認証および承認するため、ユーザを Salesforce へ移動させます。
2. ユーザは常に、この認証フローへのアクセスを承認する必要があります。アクセス承認後、アプリケーションは Salesforce からコールバックを受信します。

コンシューマはアクセストークンの入手後、そのアクセストークンを使用して、エンドユーザの代わりにデータにアクセスできます。何らかの理由でアクセストークンが無効になった場合は、更新トークンを使用して新しいアクセストークンを取得できます。

OAuth 2.0 トークン更新フロー

コンシューマは、アクセスを認証された後、更新トークンを使って新しいアクセストークン(セッション ID)を取得できます。これは、コンシューマが Web サーバまたはユーザエージェントフローのいずれかを使って更新トークンをすでに受信している場合にのみ実行可能です。アクセストークンを無効にするタイミングや、新しいトークンを取得するタイミングを決定するのは、コンシューマです。ペアーラフローは、コンシューマが更新トークンフローを受信した後で使用できます。

更新トークン認証フローの手順は次のとおりです。各手順の詳細は後述します。

1. コンシューマは既存の更新トークンを使用して、新しいアクセストークンを要求します。
2. 要求の確認後、Salesforce からクライアントにレスポンスが送信されます。

scope パラメータの値

scope パラメータを使って、Salesforce 組織でクライアントアプリケーションがアクセスできる項目を細かく設定できます。scope の有効な値は次のとおりです。

値	説明
api	REST API や Bulk API などの API によって、現在ログインしているユーザの取引先へのアクセスを許可します。これには、Chatter API リソースへのアクセスを許可する chatter_api も含まれます。
chatter_api	Chatter API リソースへのアクセスのみを許可します。
full	ログインユーザがアクセスできるすべてのデータへのアクセスを許可します。full は更新トークンを返しません。更新トークンを取得するには、refresh_token の範囲を明示的に要求する必要があります。
id	ID URL サービスへのアクセスのみを許可します。Salesforce オンラインヘルプの「ID URL の使用」を参照してください。
refresh_token	更新トークンを受信できる場合、それを返すことを許可します。
visualforce	Visualforce ページへのアクセスを許可します。

値	説明
web	Web 上で access_token を使用することを許可します。これには visualforce も含まれ、Visualforce ページへのアクセスが許可されます。

ID URL の使用

id パラメータでは、アクセストークンのほかに、トークンレスポンスの一部として ID URL も返されます。

ID URL は、ユーザを一意に識別する文字列であると同時に、有効なアクセストークンを使用してユーザの詳細情報をクエリするために使用する RESTful API でもあります。Salesforce は、ユーザの基本的な個人設定情報、クライアントが通信する重要なエンドポイント（ユーザの写真など）、アクセス可能な API エンドポイントの情報を返します。

URL の形式は、<https://login.salesforce.com/id/orgID/userID> です。ここで、orgId はユーザが所属する Salesforce 組織の ID、userID は Salesforce のユーザ ID です。



メモ: Sandbox では、login.salesforce.com の代わりに test.salesforce.com を使用します。

URL は必ず HTTPS である必要があります。

ID URL パラメータ

次のパラメータは、アクセストークンや ID URL で使用します。認証要求ヘッダーや、oauth_token パラメータを使用する要求で使用されます。詳細は、Salesforce オンラインヘルプの「アクセストークンの使用」を参照してください。

パラメータ	説明
アクセストークン	Salesforce オンラインヘルプの「アクセストークンの使用」を参照してください。
表示形式	<p>このパラメータはオプションです。返される出力の形式を指定します。有効な値は、次のとおりです。</p> <ul style="list-style-type: none"> • urlencoded • json • xml <p>表示形式パラメータを使用する代わりに、クライアントは要求の Accept ヘッダーで次のいずれかを使用して、返される形式を指定することもできます。</p> <ul style="list-style-type: none"> • Accept: application/json • Accept: application/xml • Accept: application/x-www-form-urlencoded

パラメータ	説明
	<p>次の点に注意してください。</p> <ul style="list-style-type: none"> Accept ヘッダーではワイルドカードを使用できます。 「<code>/*</code>」と指定すると、JSON が返されます。 値のリストも使用可能で、左から順に確認されます。たとえば、「<code>application/xml,application/json,application/html,/*</code>」と指定すると、XML が返されます。 Accept ヘッダーより、表示形式パラメータが優先されます。
バージョン	このパラメータはオプションです。SOAP API のバージョン番号またはリテラル文字列 <code>latest</code> を指定します。この値が指定されていないと、返される API URL に、バージョン番号ではなく、クライアントが文字列置換を行うリテラル値 <code>{version}</code> が含まれます。値が <code>latest</code> として指定されると、最新バージョンの API が使用されます。
PrettyPrint	このパラメータはオプションで、ヘッダー内でのみ使用できます。URL パラメータとしては使用できません。指定すると、出力フォーマットが最適化されます。たとえば、ヘッダーに <code>X-PrettyPrint:1</code> と指定します。この値を指定しないと、返される XML または JSON は、読みやすさではなくサイズで最適化されます。
コールバック	このパラメータはオプションです。有効な JavaScript 関数名を指定します。このパラメータは、形式として JSON を指定した場合にのみ使用します。出力はこの関数名 (JSONP) でラップされます。たとえば、 https://server/id/orgid/userid/ に対する要求が <code>{"foo":"bar"}</code> を返す場合、 https://server/id/orgid/userid/?callback=baz に対する要求は <code>baz({"foo":"bar"});</code> を返します。

ID URL レスポンス

有効な要求を作成すると、インスタンス URL への **302 リダイレクト** が返されます。後続の要求では、JSON 形式で次の情報が返されます。

- `id` – ID URL (クエリされた URL と同じ)
- `asserted_user` – 指定されたアクセストークンがこの ID に発行されたかどうかを示す Boolean 値
- `user_id` – Salesforce のユーザ ID
- `username` – Salesforce ユーザ名
- `organization_id` – Salesforce 組織 ID
- `nick_name` – クエリされたユーザのコミュニティのニックネーム
- `display_name` – クエリされたユーザの表示名 (フルネーム)

- email – クエリされたユーザのメールアドレス
- status – ユーザの現在の Chatter 状況
 - ◊ created_date: 2010-05-08T05:17:51.000Z など、ユーザの前回の投稿の作成日時の xsd datetime 値
 - ◊ body: 投稿の本文
- photos – ユーザプロファイル写真への URL の対応付け



メモ: これらの URL にアクセスするには、アクセストークンを渡す必要があります。Salesforce オンラインヘルプの「アクセストークンの使用」を参照してください。

- ◊ picture
- ◊ thumbnail
- urls – 指定されたユーザで使用できる、さまざまな API エンドポイントを含む対応付け



メモ: REST エンドポイントにアクセスするには、アクセストークンを渡す必要があります。Salesforce オンラインヘルプの「アクセストークンの使用」を参照してください。

- ◊ enterprise (SOAP)
- ◊ metadata (SOAP)
- ◊ partner (SOAP)
- ◊ profile
- ◊ feeds (Chatter)
- ◊ feed-items (Chatter)
- ◊ groups (Chatter)
- ◊ users (Chatter)
- ◊ custom_domain – 組織にカスタムドメインが設定され、プロパゲーションが完了していない場合、この値は省略されます (Salesforce オンラインヘルプの「マイドメインの概要」を参照)。
- active – クエリされたユーザが有効であるかどうかを指定する Boolean 値
- user_type – クエリされたユーザのタイプ
- language – クエリされたユーザの言語
- locale – クエリされたユーザのロケール
- utcOffset – クエリされたユーザのタイムゾーンの UTC からのオフセット (ミリ秒単位)
- last_modified_date – 2010-06-28T20:54:09.000Z など、ユーザの前回の変更の xsd datetime 形式

XML 形式のレスポンスは次のようにになります。

```
<?xml version="1.0" encoding="UTF-8"?>
<user xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<id>http://na1.salesforce.com/id/00Dx0000001T0zk/005x0000001S2b9</id>
<asserted_user>true</asserted_user>
<user_id>005x0000001S2b9</user_id>
<organization_id>00Dx0000001T0zk</organization_id>
<nick_name>admin1.2777578168398293E12foofoofoo</nick_name>
<display_name>Alan Van</display_name>
<email>admin@2060747062579699.com</email>
<status>
    <created_date xsi:nil="true" />
    <body xsi:nil="true" />
</status>
<photos>
    <picture>http://na1.salesforce.com/profilephoto/005/F</picture>
    <thumbnail>http://na1.salesforce.com/profilephoto/005/T</thumbnail>
</photos>
<urls>

<enterprise>http://na1.salesforce.com/services/Soap/c/{version}/00Dx0000001T0zk

</enterprise>

<metadata>http://na1.salesforce.com/services/Soap/m/{version}/00Dx0000001T0zk

</metadata>

<partner>http://na1.salesforce.com/services/Soap/u/{version}/00Dx0000001T0zk

</partner>
<rest>http://na1.salesforce.com/services/data/v{version}/
</rest>

<sobjects>http://na1.salesforce.com/services/data/v{version}/sobjects/

</sobjects>
<search>http://na1.salesforce.com/services/data/v{version}/search/

</search>
<query>http://na1.salesforce.com/services/data/v{version}/query/
</query>
<profile>http://na1.salesforce.com/005x0000001S2b9
</profile>
</urls>
<active>true</active>
<user_type>STANDARD</user_type>
<language>en_US</language>
<locale>en_US</locale>
<utcOffset>-28800000</utcOffset>
<last_modified_date>2010-06-28T20:54:09.000Z</last_modified_date>
</user>
```

JSON 形式のレスポンスは次のようにになります。

```
{"id": "http://na1.salesforce.com/id/00Dx0000001T0zk/005x0000001S2b9",
"asserted_user": true,
"user_id": "005x0000001S2b9",
"organization_id": "00Dx0000001T0zk",
"nick_name": "admin1.2777578168398293E12foofoofoofoo",
"display_name": "Alan Van",
"email": "admin@2060747062579699.com",
"status": {"created_date": null, "body": null},
"photos": {"picture": "http://na1.salesforce.com/profilephoto/005/F",
"thumbnail": "http://na1.salesforce.com/profilephoto/005/T"},
"urls":
{
"enterprise": "http://na1.salesforce.com/services/Soap/c/{version}/00Dx0000001T0zk",
"metadata": "http://na1.salesforce.com/services/Soap/m/{version}/00Dx0000001T0zk",
"partner": "http://na1.salesforce.com/services/Soap/u/{version}/00Dx0000001T0zk",
"rest": "http://na1.salesforce.com/services/data/v{version}/",
"sobjects": "http://na1.salesforce.com/services/data/v{version}/sobjects/",
"search": "http://na1.salesforce.com/services/data/v{version}/search/",
"query": "http://na1.salesforce.com/services/data/v{version}/query/",
"profile": "http://na1.salesforce.com/005x0000001S2b9"
},
"active": true,
"user_type": "STANDARD",
"language": "en_US",
"locale": "en_US",
"utcOffset": -28800000,
"last_modified_date": "2010-06-28T20:54:09.000+0000"}
```

無効な要求を作成した場合、Salesforce から次のレスポンスが返される可能性があります。

要求の問題	エラーコード
HTTP	403 (forbidden) – HTTPS_Required
アクセストークンがない	403 (forbidden) – Missing_OAuth_Token
アクセストークンが無効	403 (forbidden) – Bad_OAuth_Token
異なる組織のユーザ	403 (forbidden) – Wrong_Org
ユーザ ID、組織 ID が無効または間違っている	404 (not found) – Bad_Id
無効になったユーザまたは無効な組織	404 (not found) – Inactive
ユーザに組織または情報への適切なアクセス権がない	404 (not found) – No_Access
サイトのエンドポイントへの要求	404 (not found) – No_Site_Endpoint
無効なバージョン	406 (not acceptable) – Invalid_Version
無効なコールバック	406 (not acceptable) – Invalid_Callback

OAuth トークンの取り消し

ユーザがアプリケーションからログアウトしたり、アプリケーションがタイムアウトなどで無効になったりした場合、ログインユーザの資格情報がモバイルアプリケーションから消去され、サーバとの接続が切断されます。一方、サーバ上のトークンを明示的に取り消すこともできます。

ユーザが外部アプリケーション(コンシューマのページ)からデータを要求すると、ユーザが認証されますが、取り消し機能により、そのアクセストークン、または更新トークンとすべての関連するアクセストークンを取り消すことができます。開発者は、アプリケーションのログアウトボタンを設定するときに、この機能を使用できます。

トークンの取り消し

OAuth 2.0 トークンを取り消すには、次の取り消しエンドポイントを使用します。

```
https://login.salesforce.com/services/oauth2/revoke
```

HTTP 要求のエンティティ本体 (entity-body) に、application/x-www-form-urlencoded の形式で、次のパラメータを含む POST 要求を記述します。たとえば、次のようにになります。

```
POST /revoke HTTP/1.1
Host: https://login.salesforce.com/services/oauth2/revoke
Content-Type: application/x-www-form-urlencoded

token=currnettoken
```

アクセストークンが含まれる場合、それを無効化してトークンを取り消します。更新トークンが含まれる場合、そのトークンと関連するすべてのアクセストークンを取り消します。

要求が正常に処理されると、認証サーバは HTTP ステータスコード 200 を返し、エラーが発生すると、ステータスコード 400 と次のいずれかのエラーレスポンスを返します。

- unsupported_token_type – サポートされていないトークンのタイプ
- invalid_token – トークンが無効

Sandbox では、login.salesforce.com の代わりに test.salesforce.com を使用します。

リモートアクセスアプリケーションの作成

モバイルデバイスをサービスに接続する前に、リモートアクセスアプリケーションを作成する必要があります。リモートアクセスアプリケーションには、コンシューマ鍵が含まれます。これは、本ドキュメントで紹介するすべての開発シナリオの前提条件となります。

1. Database.com または Force.com インスタンスにログインします。
2. [アプリケーションの設定] > [開発] > [リモートアクセス] の順にクリックします。
3. [新規] をクリックします。
4. [アプリケーション] に「Test Client」のようなアプリケーション名を入力します。
5. [コールバック URL] に「sfdc://success」と入力します。



メモ: [コールバック URL] には、コールバックされる URL を指定します。有効な URL である必要はありません。sfdc:///など、任意のカスタムプレフィックスを使用できます。

6. [取引先責任者 メール] に、自分のメールアドレスを入力します。
7. [保存] をクリックします。



ヒント: [リモートアクセスの詳細] ページに、コンシューマ鍵が表示されます。後で使用するので、テキストファイルにコピーしておくとよいでしょう。モバイルアプリケーションはコンシューマの秘密を使用しないので、この値は無視してかまいません。

Remote Access

[« Back to List: Remote Access](#)

[Printable View](#) | [Help for this Page](#)

Remote Access Detail

[Edit](#) [Delete](#)

Basic Information

Application	My Hybrid App
Description	Demo of the Salesforce Mobile SDK.
Logo Image URL	https://example.com/images/logo.png
Info URL	https://example.com/hybridapp/info.html
Contact Phone	
Contact Email	user@example.com

= Required Information

Integration

Callback URL	https://login.salesforce.com/services/oauth2/success
--------------	-------------------------------------------------------------------------------------------------------------------------

= Required Information

Policies

No user approval required for users in this organization	
----------------------------------------------------------------	--

= Required Information

Authentication

My App uses digital signatures for login	
Consumer Key	3MVG9Km_cBLhsuPzTtGhsZpj9Bzs2Uk4M6eW8YPP_l3Uwrid56s15QBgRkrY5y8BpMqU1XUk2LgZGx6xo79
Consumer Secret	Click to reveal

Created By [Andy Admin](#)

= Required Information

第 3 章

接続済みアプリケーション

トピック:

- 接続済みアプリケーションの開発と管理
- 開発者の作業
- 管理者の作業

接続済みアプリケーションとは、ID とデータ API を使用して Salesforce に接続できるアプリケーションのことです。接続済みアプリケーションでは、認証やシングルサインオン、Salesforce API で使用するアクセストークンの取得に、標準の OAuth 2.0 プロトコルを使用します。管理者は、標準の OAuth 機能を利用できるだけでなく、制御レベルを追加して、アプリケーションの利用を許可するユーザやさまざまなセキュリティポリシーを明示的に制御できます。

接続済みアプリケーションは、すべての新規 Developer Edition 組織で有効です。既存の Developer Edition 組織とその他の Salesforce 組織では、パイロットプログラムの一部として接続済みアプリケーションを有効にすることができます。



メモ: 現在、接続済みアプリケーション機能はパイロットプログラムとして提供されています。組織内で接続済みアプリケーション機能を有効にする方法については、セールスフォース・ドットコムの担当者までお問い合わせください。本ドキュメント、その他のプレスリリース、公開資料に記載しているリリース前のサービスおよび機能は、現時点ではまだご利用いただけません。また、将来の提供を確約するものではありません。弊社のサービスのご購入にあたっては、現在利用可能な機能を基準にご判断いただきますようお願いいたします。

接続済みアプリケーションの開発と管理

最初に、開発者がアプリケーションの OAuth メタデータを定義します。次のようなメタデータがあります。

- 接続済みアプリケーションの簡単な説明と連絡先情報
- 接続済みアプリケーションの OAuth の範囲とコールバック URL
- 接続済みアプリケーションを実行する IP 範囲 (オプション)
- 接続済みアプリケーションに適用するモバイルポリシーの情報 (オプション)

開発者は、OAuth クライアント ID とクライアントの秘密、接続済みアプリケーションのインストール URL を受け取り、この URL を Salesforce 管理者に渡します。

管理者は、接続済みアプリケーション、ユーザプロファイル、権限セット、IP 範囲の制限を組織にインストールして、アプリケーションにアクセスできるユーザを管理します。管理は、接続済みアプリケーションの詳細ページから行います。接続済みアプリケーションをアンインストールして、新しいバージョンをインストールすることもできます。新しいバージョンの準備ができると、開発者は管理者に通知します。管理者は、既存のインストール URL から新しいバージョンをインストールできます。

開発者の作業

接続済みアプリケーションのライフサイクルは、次のフェーズで構成されます。

- 接続済みアプリケーションの作成
- 接続済みアプリケーションの公開
- 接続済みアプリケーションの削除
- 接続済みアプリケーションの更新

接続済みアプリケーションの作成

次の手順を実行して、接続済みアプリケーションを作成します。

- あなたの名前>[設定]>[作成]>[アプリケーション] の順にクリックします。
- [接続済みアプリケーション] セクションの [新規] をクリックします。

The screenshot shows the 'Connected Apps' section of the Salesforce 'Apps' page. At the top, there's a note: 'Custom apps work in conjunction with User Profile Tab Visibility settings. View User Profiles now.' Below this is a table with columns: Action, App Label, Service Cloud Console, Custom, and Description. A message at the bottom says 'No Apps found.'

次の各セクションに、接続済みアプリケーションの作成に必要な情報を入力します。

- ・ [基本情報] — アプリケーションの説明、使用可能なアプリケーションのリストに表示する画像、アプリケーションに関する連絡先など
- ・ [API 統合] — アプリケーションと Salesforce の通信方法
- ・ [モバイル統合] — モバイルアプリケーションの PIN の文字数とセッションタイムアウト値
- ・ [IP 範囲] — ユーザ認証なしでアプリケーションにアクセスできる IP アドレスのリスト（アプリケーションの作成後に設定）

情報の入力が完了したら、[保存] をクリックして新しいアプリケーションを保存します。これで、アプリケーションを公開する準備ができました。必要に応じて、アプリケーションを編集、削除することもできます。アプリケーションを保存すると、Salesforce との通信で使用するコンシューマ鍵とコンシューマの秘密が生成されます。

接続済みアプリケーションの基本情報

このセクションには、アプリケーション名、アプリケーションのロゴ、取引先責任者の連絡先などの基本情報を指定します。

1. [接続済みアプリケーション名] にアプリケーション名を入力します。接続済みアプリケーションのリストには、ここで入力したアプリケーション名が表示されます。



メモ: 組織内で一意のアプリケーション名を指定してください。既存のアプリケーション名や、削除された接続済みアプリケーションの名前を再利用することはできません。

2. 必要に応じて、[説明] にアプリケーションの説明を入力します。これも接続済みアプリケーションのリストに表示されます。
3. アプリケーションのロゴ画像を指定する場合は、[ロゴ画像 URL] に HTTPS を使用した URL を入力します。また、ロゴの大きさは最大で縦 125 ピクセル、横 200 ピクセルとします。デフォルトのロゴは雲の形のロゴです。

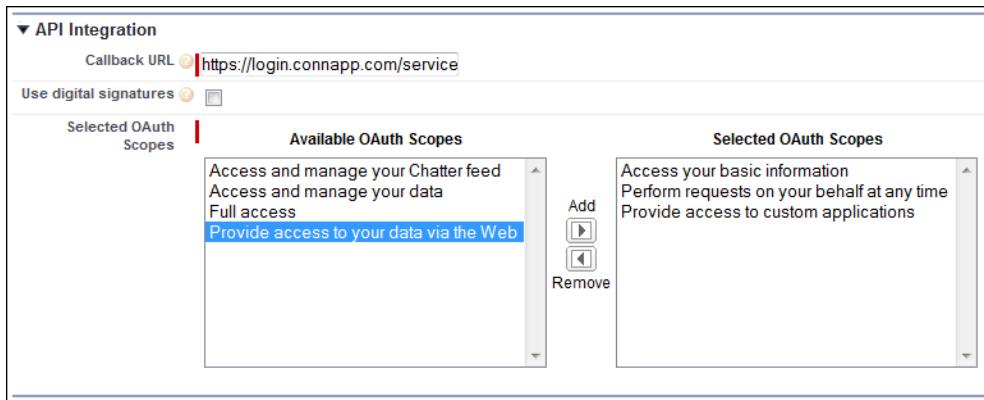
4. アプリケーションの詳細情報を表示する Web ページがある場合は、[情報 URL] にその URL を入力します。
5. [取引先責任者 電話] に、セールスフォース・ドットコムからお客様に連絡するときに使用する電話番号を入力します。この番号は、アプリケーションをインストールする管理者には公開されません。
6. [取引先責任者 メール] に、セールスフォース・ドットコムからお客様またはお客様のサポートチームに連絡するときに使用するメールアドレスを入力します。このアドレスは、アプリケーションをインストールする管理者には公開されません。

▼ Basic Information	
Connected App Name	ExampleConnApp
Description	Demonstrate Connected App creation.
Logo Image URL	(empty)
Info URL	http://www.connapp.com
Contact Phone	415-555-1234
Contact Email	Support@ConnApp.com

接続済みアプリケーションの API 統合

[API 組合] セクションには、アプリケーションと Salesforce の通信方法を指定します。

1. [コールバック URL] にコールバック URL を入力します。これは、OAuth 認証時に Salesforce からアプリケーションにコールバックされる URL (エンドポイント) で、OAuth の redirect_uri に指定されます。
2. アプリケーションが証明書を使用する場合は、[デジタル署名を使用] チェックボックスをオンにします。
3. [利用可能な OAuth 範囲] から OAuth 範囲を選択して、[選択した OAuth 範囲] に追加します。これは、接続済みアプリケーションを実行するユーザーによって付与される次のような権限を指定します。
 - Chatter フィードへのアクセスと管理 (Chatter REST API を使用)
 - データへのアクセスと管理
 - 基本情報へのアクセス
 - フルアクセス
 - ユーザに代わっていつでも要求を実行
 - カスタムアプリケーションへのアクセスの提供
 - Web 経由のデータへのアクセスを提供



Spring' 12 以前のリリースのリモートアクセスの設定で、[この組織のユーザにはユーザ承認は必要ありません] チェックボックスをオンにしている場合、アプリケーションを作成した組織内のユーザには、アプリケーションへのアクセスが自動承認されます。この場合、[この組織のユーザにはユーザ承認は必要ありません] チェックボックスがオフの状態で表示されます(変更不可)。接続済みアプリケーションの場合、アプリケーションの作成後に管理者がアプリケーションをインストールし、[許可されているユーザ] を [管理者が承認したユーザ] に設定することをお勧めします。[この組織のユーザにはユーザ承認は必要ありません] チェックボックスをオンにしていない場合、このチェックボックスは表示されません。

接続済みアプリケーションのモバイル統合

モバイルアプリケーションには、画面ロックと PIN コード保護のポリシーを適用できます。これらのポリシーは Salesforce Mobile SDK (<http://developer.force.com/mobilesdk>) によって自動的に適用されますが、ユーザの ID URL の mobile_policies オブジェクトを参照して手動で適用することもできます。ポリシーを適用する場合は、[画面ロックと固定保護を実装] チェックボックスをオンにします。これにより管理者は、接続済みアプリケーションをインストールした後、モバイルアプリケーションのセッションタイムアウトと PIN の文字数を設定できます。

接続済みアプリケーションの IP 範囲

アプリケーションの作成後、[IP 範囲] セクションの [新規] をクリックして、IP 範囲を指定できます。これは、アプリケーションへのアクセスが許可された IP アドレスのホワイトリストとして機能します。[開始 IP アドレス] に有効な IP アドレスを入力し、[終了 IP アドレス] に開始 IP アドレスより上位のアドレスを入力します。連続しない複数の範囲を指定する場合は、[新規] をクリックして、次の範囲を指定します。アプリケーションのインストール後、各組織の管理者は IP 制限を設定して、範囲を承認するかどうかを決定します。

接続済みアプリケーションの公開

接続済みアプリケーションを作成したら、他のユーザに公開します。未公開のアプリケーションは、[接続済みアプリケーション] リスト内に [公開されていない変更] として表示されます。リスト内の接続済みアプリケーション名をクリックすると、アプリケーションの編集ページが表示されます。

このページで次のような作業を行うことができます。

- 公開:** アプリケーションのインストール URL を作成します。
- 編集:** アプリケーションの作成時に指定したアプリケーション情報を変更できます。
- 削除:** アプリケーションを完全に削除します。ただし、削除したアプリケーションの名前は記録されるため、同じ名前を再利用することはできません。

[公開] をクリックすると、接続済みアプリケーションを公開してもよいか確認するメッセージが表示されます。[公開] をクリックして公開します。[バージョン] の値が 1 つ大きくなり、新しい値が生成されます。

- インストール URL** – 管理者はこの URL を使用して、接続済みアプリケーションを組織内にインストールします。
- コンシューマ鍵** – コンシューマはこの値を使用して Salesforce に自身の識別情報を示します。OAuth 2.0 では client_id として参照されます。
- コンシューマの秘密** – コンシューマがコンシューマ鍵の所有権を確立するために使用する秘密です。OAuth 2.0 では client_secret として参照されます。

Connected App Name
ExampleConnApp

[« Back to List: Custom Apps](#)

[New Version](#) [Delete](#)



Version	1
Status	Published
Description	Demonstrate Connected App creation.
Created Date	8/30/2012 4:44 PM
By	Hal User
Contact Phone	415-555-1234
Contact Email	Support@ConnApp.com
Last Modified Date	8/30/2012 4:54 PM
By	Hal User

Basic Information

Installation URL	https://login-blitz01.soma.salesforce.com/services/apps/oCID0000000000pO	Info URL	http://www.connapp.com
------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------	----------	-------------------------------------------------------------

API Integration

Consumer Key	3MVG9PhR6gB7ps5XMZugtWS5D8uDeDlwdnBaU8HBubxThdI09GQw84EACkLD90ccoJ7RtPZOMkKa42.YAMMF	Consumer Secret	Click to reveal
Selected OAuth Scopes	Provide access to custom applications Perform requests on your behalf at any time Access your basic information	Callback URL	https://login.connapp.com/services/oauth2/callback

Mobile Integration

Implements Screen Locking & PIN Protection	<input checked="" type="checkbox"/>
--------------------------------------------	-------------------------------------



メモ: テスト中などに行われた未公開の接続済みアプリケーションの OAuth 認証の承認は、最初に公開されたバージョンにも有効です。この承認は、その後に公開されるバージョンには移行しません。

接続済みアプリケーションの削除

接続済みアプリケーションを削除するには、アプリケーションリスト内の接続済みアプリケーション名をクリックします。編集ページの [削除] をクリックしてから、もう一度 [削除] をクリックして削除します。アプリケーションをリストから削除しても、削除したアプリケーションの名前を再利用することはできません。

組織にインストールされていた接続済みアプリケーションを削除した場合、組織管理者の [接続済みアプリケーション] リストにはそのアプリケーションが引き続き表示されますが、実行することはできません。実行できる操作は [削除] のみです。

接続済みアプリケーションの更新

接続済みアプリケーションはいつでも更新できます。リスト内の接続済みアプリケーション名をクリックすると、アプリケーションの編集ページが表示されます。[新しいバージョン] をクリックして、編集ページから変更を加えます。[保存] をクリックして変更を保存します。[接続済みアプリケーション] リストでは、アプリケーションの状況は [公開されていない変更] になります。アプリケーション名をクリックし、編集画面の [公開] をクリックして、アプリケーションを公開します。アプリケーションを公開すると、それ以前のバージョンは利用できなくなります。

管理者の作業

管理者は、接続済みアプリケーションに対して次の作業を行います。

- ・ 接続済みアプリケーションのインストール
- ・ 接続済みアプリケーションの管理
- ・ 接続済みアプリケーションのアンインストール
- ・ 接続済みアプリケーションのアップグレード

接続済みアプリケーションのインストール

接続済みアプリケーションの開発者から受け取ったインストール URL を使って、接続済みアプリケーションをインストールします。アプリケーションを組織にインストールするには、Salesforce 組織にログインし、ブラウザに URL を貼り付けてインストールプロセスを開始する方法が一番簡単です。確認画面が表示され、アプリケーション名、アプリケーションの説明、インストール後のユーザアクセス制御の方法が表示されます。[インストール] をクリックするとインストールが完了します。

複数の Salesforce 組織にログインしている場合、自動的に 1 つが選択されます。右上に表示されるユーザ名から、アプリケーションのインストール先の組織が正しいことを確認します。ユーザ名が正しくない場合は、ログアウトしてインストールを中止します。

組織内に、以前にインストールした旧バージョンの接続済みアプリケーションが存在する場合、エラーメッセージが表示されるので、現在のバージョンをアンインストールして最新バージョンをインストールします。

インストールが完了すると、アプリケーションの詳細ページが表示されます。このページからアプリケーションのポリシーを編集できます。

接続済みアプリケーションの管理

接続済みアプリケーションの管理は編集ページから行います。[あなたの名前](#) > [設定] > [アプリケーションを管理する] > [接続済みアプリケーションを管理する] をクリックすると、アプリケーションが表示されます。このページから、アプリケーション情報の編集、確認のほか、アプリケーションのアンインストールを行うことができます。

- ・ アプリケーションに変更を加えるには、[編集] をクリックします。

[アプリケーションポリシー] はすべての接続済みアプリケーションに適用できます。

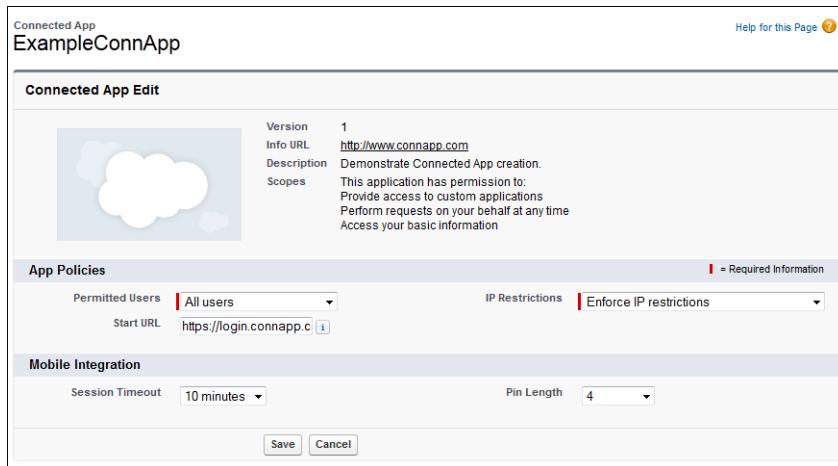
◊ [許可されているユーザ] でアプリケーションを実行できるユーザを指定します。

- デフォルトでは [すべてのユーザ] が選択されており、組織内の全ユーザがアプリケーションの認証を許可されます。各ユーザは初回アクセス時にアプリケーションを承認する必要があります。[すべてのユーザ] から [管理者が承認したユーザ] に変更すると、管理者が指定した権限セットを持つユーザ以外はこのアプリケーションにアクセスできなくなります。
- [管理者が承認したユーザ] では、指定の権限セットを持つユーザのみにアクセスが制限されます。これらのユーザは承認なしでアプリケーションにアクセスできます。アプリケーションの権限セットの管理は、詳細ページから行います。

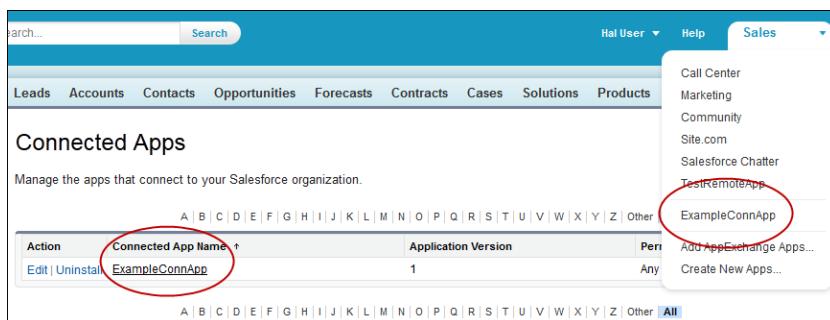
◊ [IP 制限] では、この接続済みアプリケーションのユーザに IP 制限を適用できます。管理者は次のいずれかのオプションを選択して、制限を適用または回避します。

- [IP 制限を適用] – デフォルトの設定。このアプリケーションを実行するユーザに、組織の IP 制限 (ユーザプロファイルで設定した IP 範囲など) を適用します。
- [IP 制限を 2 次要素で緩和] – このアプリケーションを実行するユーザは、次のいずれかの条件を満たしている場合、組織の IP 制限を回避できます。
 - IP 範囲のホワイトリストが存在し、アプリケーションが Web サーバの OAuth 認証フローを利用している場合。ホワイトリストに登録された IP からの要求のみが許可されます。
 - IP 範囲のホワイトリストは存在しないが、アプリケーションが Web サーバまたはユーザエージェントによる OAuth 認証フローを利用して、ユーザが ID 確認に成功した場合。
- [IP 制限の緩和] – このアプリケーションを実行するユーザには、IP 制限は適用されません。

- ◊ 接続済みアプリケーションでシングルサインオン機能を利用する場合は、[開始 URL] に認証プロセスを開始するページの URL を設定します。



この URL はアプリケーションの切り替えメニューにも表示されます。



- ◊ [モバイル統合] の設定は、モバイルアプリケーションとして利用されるすべての接続済みアプリケーションに適用できます。
 - [セッションタイムアウト] には、何も操作しない状態が何分間続くと、アプリケーションがロックされ、PIN 入力が必要な状態になるかを指定します。1 分、5 分、10 分、30 分のいずれかを選択できます。
 - [PIN の文字数] には、認証用に送信する識別番号の文字数を設定します。4 から 8 までの値を選択できます。
- 接続済みアプリケーションをアンインストールするには、[アンインストール] をクリックします。[OK] をクリックすると、アンインストールが行われます。新しいバージョンをインストールする前に、以前のアプリケーションをアンインストールする必要があります。
- 接続済みアプリケーションの詳細ページを確認するには、アプリケーション名をクリックします。確認後に変更を加えるには、詳細ページの [編集] または [アンインストール] をクリックします。詳細ページから、アプリケーションにプロファイルや権限セットを割り当てることもできます。

- ◊ [権限セットの管理] をクリックして [アプリケーション権限セットの割り当て] ページを開き、このアプリケーションのプロファイルに割り当てる権限セットを選択します。選択した権限セットに、アプリケーションへのアクセスを許可することができます。
- ◊ [プロファイルを管理する] をクリックして [アプリケーションプロファイルの割り当て] ページを開き、このアプリケーションに割り当てるプロファイルを選択します。選択したプロファイルに、アプリケーションへのアクセスを許可することができます。

Connected App
ExampleConnApp

[Printable View](#)

[« Back to List: Connected Apps](#)

Connected App Detail

[Edit](#) [Uninstall](#)

Version	1
Info URL	http://www.connapp.com
Description	Demonstrate Connected App creation.
Scopes	This application has permission to: Provide access to custom applications Perform requests on your behalf at any time Access your basic information

System Info

IP Ranges
No application-defined IP ranges

Profiles [Manage Profiles](#)

Profile	Profile Description
Warehouse User	For inventory control and shipping tasks.

Permission Sets [Manage Permission Sets](#)

Permission Set	Permission Set Description
Auth Provider Admins	Admins that can modify Auth Providers.
Invoice Approvals	Can open, approve and close invoices.

[許可されているユーザ] で [管理者が承認したユーザ] を選択した場合、選択したプロファイルまたは権限セットを持つユーザにのみアプリケーションの実行が許可されます。[すべてのユーザ] を選択した場合、プロファイルと権限セットは無視されます。

- 開発者がアプリケーションを削除した場合、管理者はアプリケーションの削除しか実行できなくなります。削除されたアプリケーションは、リストからも削除されます。

PIN コード保護について

接続済みのモバイルアプリケーションには、PIN コード保護によるセキュリティレイヤが追加されています。この PIN コード保護は、デバイスの PIN コード保護や、Salesforce 組織で提供されているログインセキュリティとは異なり、モバイルアプリケーション自体を保護するものです。

PIN コード保護を有効にするには、接続済みアプリケーションの作成時に、開発者が [画面ロックと固定保護を実装] チェックボックスをオンにする必要があります。そうすることにより、モバイルアプリケーションの管理画面に、PIN コード保護の有効化オプション、タイムアウト値の設定オプション、PIN の文字数の設定オプションが表示されます。



メモ: PIN コード保護はモバイルデバイスのオペレーティングシステムに実装されるため、適用対象はネイティブモバイルアプリケーションとハイブリッドモバイルアプリケーションのみとなります。HTML5 Web アプリケーションには適用できません。

PIN コード保護を適用した場合、モバイルアプリケーションを操作しない状態が一定時間経過すると、モバイルアプリケーションがロックされます。モバイルアプリケーションがバックグラウンド動作になっても、経過時間はカウントされます。

PIN コード保護の仕組みは次のとおりです。

1. 携帯電話の電源を入れ、デバイスの PIN コードを入力します。
2. モバイルアプリケーション(接続済みアプリケーション)を起動します。
3. Salesforce 組織のログイン情報を入力します。
4. モバイルアプリケーションの PIN コードを入力します。
5. モバイルアプリケーションの操作中に着信があつたり、別のアプリケーションを開いたりすると、モバイルアプリケーションがバックグラウンド動作になります。
6. モバイルアプリケーションがタイムアウトになります。
7. 再度モバイルアプリケーションにアクセスしようとすると、デバイスではなく、モバイルアプリケーションの PIN コード入力画面が表示されます。
8. PIN コードを入力し、作業を再開します。

接続済みアプリケーションのアンインストール

接続済みアプリケーションをアンインストールするには、**あなたの名前** > [設定] > [アプリケーションを管理する] > [接続済みアプリケーションを管理する] をクリックして、アプリケーション名の横の [アンインストール] をクリックします。確認ウィンドウで [OK] をクリックします。



メモ: 接続済みアプリケーションをアンインストールすると、このアプリケーションのすべてのユーザのアクセストークンと更新トークンが削除されます。これは、ユーザが今後、既存のアクセストークンを使用し、明示的にアプリケーションを承認しないでアプリケーションを実行するのを防ぐためです。

接続済みアプリケーションのアップグレード

新しいバージョンの接続済みアプリケーションが公開された場合、現在のバージョンをアンインストールしてから新しいバージョンをインストールする必要があります。インストールには、以前に使用したインストール URL を使用します。

接続済みアプリケーションのエラーコード

接続済みアプリケーションを使用するときに、次のエラーコードが表示される場合があります。

エラーコード	エラー	説明
1805	APP_ACCESS_DENIED	管理者に、この接続済みアプリケーションへのアクセスを許可されていないユーザ。

第 4 章

ネイティブ iOS アプリケーションの開発

トピック:

- ネイティブ iOS アプリケーションのクイックスタート
- 既存のプロジェクトへの Mobile SDK の統合
- iOS 用の RestAPIExplorer サンプルアプリケーション

iOS 用のネイティブ SDK には、次の特長があります。

- OAuth2 ログインプロセスを自動化します。また、OAuth2 をアプリケーションに簡単に組み込むことができます。
- すべてのインフラストラクチャクラス (RestKit のようなサードパーティライブラリを含む) を使って、REST API に簡単にアクセスできます。

Salesforce Mobile SDK を使って新規プロジェクトを作成する場合、テンプレートアプリケーションが自動的にインストールされます。これは、組織に接続して簡単なクエリを実行するだけの単純なアプリケーションですが、設計どおりにアプリケーションが動作することを確認し、独自のアプリケーションを開発するための基礎知識を身に付けるのに役立ちます。

ネイティブ iOS アプリケーションのクイックスタート

次の手順で開始します。

1. ネイティブ iOS の開発要件をすべて満たしていることを確認します。
2. Mobile SDK for iOS をインストールします。
3. テンプレートアプリケーションを実行します。

ネイティブ iOS の開発要件

- Xcode — Xcode 4.0 以上 (最新バージョン推奨)
- iOS 4.3 以上
- Mobile SDK
- Developer Edition 組織とリモートアクセスアプリケーション

Mobile SDK for iOS のインストール

1. ブラウザで、Mobile SDK for iOS の GitHub リポジトリにアクセスします。
<https://github.com/forcedotcom/SalesforceMobileSDK-iOS>
2. 次のコマンドを実行して、リポジトリをローカルファイルシステムへコピーします。
`git clone git://github.com/forcedotcom/SalesforceMobileSDK-iOS.git`



メモ: Mac OS X 用の GitHub アプリケーションを使用する場合は、[Clone in Mac] をクリックします。

3. Mac OS X ターミナルアプリケーションを起動し、コマンドラインから、リポジトリをコピーしたディレクトリでインストールスクリプト `./install.sh` を実行します。
4. GitHub からサンプルアプリケーションもダウンロードしておきます。
<https://github.com/forcedotcom/SalesforceMobileSDK-Samples/tree/master/iOS/CloudTunesNative>

Xcode でのネイティブ iOS アプリケーションの作成

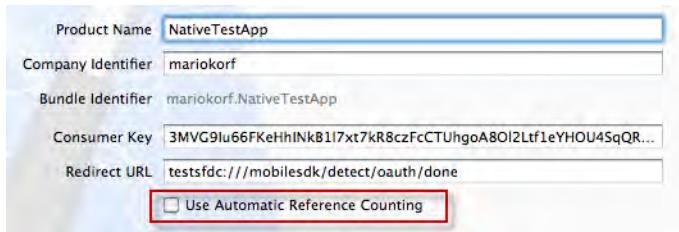
次の手順で、新しい Force.com ベースのアプリケーションプロジェクトを作成し、設定します。

1. Xcode を起動し、新しいプロジェクトを作成します (Shift + Command + N キー)。
2. [Native Force.com REST App] を選択し、[Next] をクリックします。
3. [Choose options for your new project] ダイアログで、[Product Name] に「NativeTestApp」と入力します。



メモ: Xcode を初めて使用する場合は、[Company Identifier] に、会社識別用のプレフィックスも入力する必要があります。

4. [Use Automatic Reference Counting] チェックボックスがオフになっていることを確認します。



5. [Next] をクリックします。
6. 新しいプロジェクトの場所を指定し、[Create] をクリックします。

Xcode プロジェクトテンプレートアプリケーションの実行

Xcode プロジェクトテンプレートには、すぐに実行できるサンプルアプリケーションが含まれています。

1. **Command + R** キーを押すと、デフォルトのテンプレートアプリケーションがビルドされ、iOS シミュレータで実行されます。



メモ: ビルドエラーになった場合は、以下の手順で **Automatic Reference Counting (ARC)** がオフになっていることを確認してください。

- プロジェクトナビゲータでプロジェクトを選択します。
- [Build Settings] タブで、[Objective-C Automatic Reference Counting] の値を [No] に切り替えます。

2. アプリケーションの起動時に OAuth 認証フローがスタートし、認証ページが表示されます。ログイン情報を入力して、[ログイン] をクリックします。
3. [許可] をタップして、アプリケーションにアクセス権を与えます。

これで、サンプルプロジェクトをコンパイルして実行することができます。OAuth2 を使って組織にログインし、「select Name from User」クエリを発行し、UITableView に結果を表示するシンプルなアプリケーションです。

既存のプロジェクトへの Mobile SDK の統合

既存の iOS プロジェクトで Mobile SDK を使用するには、次の手順を実行します。

- Xcode で、[native/dependencies] フォルダをプロジェクト内にドラッグします ([Create groups for any added folders] を選択します)。
- プロジェクトの [Build Settings] タブを開き、[Other Linker Flags] に 「-ObjC -all_load」を設定します。
- プロジェクトのメインターゲットの [Build Phases] タブを開き、次の必須フレームワークをリンクします。

- CFNetwork.framework
 - CoreData.framework
 - MobileCoreServices.framework
 - SystemConfiguration.framework
 - Security.framework
 - libxml2.dylib
4. 「#import “SFRestAPI.h”」を実行して、SalesforceSDK ヘッダーをインポートします。
 5. プロジェクトをビルドして、インストールが正しく行われていることを確認します。
 6. Salesforce インスタンスにログインし、REST API コールを発行するサンプルコードについては、SFRestAPI のドキュメントを参照してください。

iOS 用の RestAPIExplorer サンプルアプリケーション

上で実行した Xcode プロジェクトテンプレートは SOQL クエリを発行し、結果を返すだけの単純なサンプルアプリケーションでしたが、RestAPIExplorer サンプルアプリケーションには、独自のアプリケーションに組み込める多数の機能が用意されています。

RestAPIExplorer サンプルアプリケーションは、Mobile SDK for iOS の以下の場所に格納されています。

native/SampleApps/RestAPIExplorer

第 5 章

ネイティブ Android アプリケーションの開発

トピック:

- ネイティブ Android アプリケーションのクイックスタート
- Eclipse でのプロジェクトの設定
- Eclipse によるクリーンとビルド
- Android 用の RestExplorer サンプルアプリケーション

Android 用のネイティブ SDK には、次の特長があります。

- OAuth2 ログインプロセスを自動化します。また、OAuth2 をアプリケーションに簡単に組み込むことができます。
- アクセスを簡便化するインフラストラクチャクラスを使って REST API にアクセスします。

Android 用 Salesforce Mobile SDK には、いくつかのサンプルネイティブアプリケーションが含まれています。さらに、新しいアプリケーションをすばやく作成できる ant ターゲットも用意されています。

ネイティブAndroid アプリケーションのクイックスタート

次の手順で開始します。

1. ネイティブAndroid の開発要件をすべて満たしていることを確認します。
2. Mobile SDK for Android をインストールします。
3. コマンドラインから ant スクリプトを実行して新規Android プロジェクトを作成し、テンプレートアプリケーションを実行します。

ネイティブAndroid の開発要件

- Java JDK 6
- Apache Ant 1.8 以上
- Android SDK バージョン 20 以上。<http://developer.android.com/sdk/installing.html> の手順に従ってインストールしてください。



メモ: Android SDK のインストールでは、ターゲットバージョンだけでなく、使用可能なすべてのバージョンをインストールすることをお勧めします。

- Eclipse 3.6 以上。その他のバージョンについては、<http://developer.android.com/sdk/requirements.html> を参照してください。
- Android ADT (Android Development Tools) plugin for Eclipse バージョン 20 以上。<http://developer.android.com/sdk/eclipse-adt.html#installing> の手順に従ってインストールしてください。
- アプリケーションをエミュレータで実行するには、Android 2.2 以上 (2.2 推奨) をターゲットとする Android Virtual Device (AVD) を最低 1 つ作成する必要があります。Eclipse で AVD を作成する方法については、<http://developer.android.com/guide/developing/devices/managing-avds.html> を参照してください。
- Developer Edition 組織とリモートアクセスアプリケーション

SalesforceSDK プロジェクトのビルドには、Android 3.0 (HoneyComb) のライブラリを使用します。これは、実行時にファイルシステム暗号化機能の条件判定を行うためです。古いバージョンの Android プラットフォームでは、条件チェックは省略されます。条件チェック機能をサポートする Android 2.2 以上のライブラリ (salesforcesdk.jar) は引き続き使用可能です。

Mobile SDK for Android のインストール

1. ブラウザで、Mobile SDK for Android の GitHub リポジトリにアクセスします。
<https://github.com/forcedotcom/SalesforceMobileSDK-Android>
2. 次のコマンドを実行して、リポジトリをローカルファイルシステムへコピーします。
`git clone git://github.com/forcedotcom/SalesforceMobileSDK-Android.git`
3. コマンドプロンプトを起動し、リポジトリをコピーしたディレクトリでインストールスクリプト `./install.sh` を実行します。



メモ: Windows ユーザは `cscript install.vbs` を実行します。

次のシェル変数を作成します。

1. `ANDROID_SDK_DIR` — Android SDK ディレクトリを参照します。
2. `SALESFORCE_SDK_DIR` — Salesforce Mobile SDK リポジトリをコピーしたディレクトリ (例: `/home/jon/SalesforceMobileSDK-Android`) を参照します。
3. `NATIVE_DIR` — `$SALESFORCE_SDK_DIR/native` を参照します。
4. `TARGET_DIR` — 作成する Android プロジェクトの格納場所を参照します。



メモ: これらの変数を設定していない場合は、コード内の `$ANDROID_SDK_DIR`、`$SALESFORCE_SDK_DIR`、`$NATIVE_DIR`、`$TARGET_DIR` を実際のパスに置き換えてください。

Android プロジェクトの作成

`ant` スクリプトを使用すると、ネイティブ Android プロジェクトを簡単に作成できます。次のパラメータを指定して、スクリプトを実行します。

- `appName` — 新規アプリケーションの名前
- `targetDir` — コードを格納するディレクトリ (環境変数の `$TARGET_DIR` を定義している場合は、この変数の参照する場所)
- `packageName` — 新規アプリケーションの Java パッケージ (例: `com.acme.mobileapp`)

ネイティブ Android プロジェクトを作成するには、次の手順を実行します。

1. Salesforce Mobile SDK をインストールした場所 (変数 `$SALESFORCE_SDK_DIR`) でコマンドプロンプトを起動します。
2. 「`ant create_native -Dapp.name={appName} -Dttarget.dir={targetDir} -Dpackage.name={packageName}`」を実行します。

Android プロジェクトには、ビルドして実行できる単純なアプリケーションが付属しています。

Android テンプレートアプリケーション

Android 用のネイティブテンプレートアプリケーションでは、ログインして、クエリ、挿入などの標準的な CRM 機能を実行できます。

このアプリケーションをビルドするには、次の手順を実行します。

1. テキストエディタで \$TARGET_DIR/res/values/rest.xml を開きます。
2. OAuth クライアント ID とコールバック URL を入力して、ファイルを保存します。
3. コマンドプロンプトを起動し、次のコマンドを入力します。

```
cd $TARGET_DIR
$ANDROID_SDK_DIR/tools/android update project -p . -t 1
ant clean debug
```



メモ: -t 1 パラメータを使って、Android 11 をターゲットバージョンとして指定しています。ターゲット ID のリストを確認したい場合は、「android.bat list targets」を実行します。

4. エミュレータが起動していない場合は、Android AVD Manager を使って起動します。実際のデバイスを使用する場合は、デバイスを接続します。
5. ant installd を実行します。

Eclipse でのプロジェクトの設定

リポジトリのコピーには、他のサンプルアプリケーションも含まれています。これらを Eclipse にインポートします。

1. Eclipse を起動し、ワーカースペースディレクトリとして \$TARGET_DIR を選択します。
2. [Window] > [Preferences] を選択し、[Android] セクションを選択して、Android SDK の場所を指定します。
3. [OK] をクリックします。
4. [File] > [Import] を選択し、[General] > [Existing Projects into Workspace] を選択します。
5. [Next] をクリックします。
6. ルートディレクトリとして \$NATIVE_DIR を選択し、[Android Project Files] に表示されているプロジェクトをインポートします。
7. ナビゲータで SalesforceSDKTest を右クリックし、[New] > [Folder] を選択して、フォルダ名を「gen」に設定します。フォルダがすでに存在する場合は、[Cancel] をクリックします。
8. 同様にして、RestExplorer プロジェクトと RestExplorerTest プロジェクトにも gen フォルダを作成します。
9. SalesforceSDK プロジェクトを右クリックして、「res」というフォルダを作成します（フォルダが存在しない場合）。

Android プロジェクトファイル

\$NATIVE_DIR には、複数のプロジェクトが含まれています。

1. SalesforceSDK — SalesforceSDK — OAuth2 と REST API コールをサポートする SalesforceSDK プロジェクト
2. SalesforceSDKTest — SalesforceSDK のテストプロジェクト
3. TemplateApp — SalesforceSDK を使って新規ネイティブアプリケーションを作成するときに使用するテンプレート
4. TemplateAppTest — Templateapp のテストプロジェクト
5. RestExplorer — SalesforceSDK で REST API コールを使用するためのサンプルアプリケーション
6. RestExplorerTest — RestExplorer のテストプロジェクト
7. SampleApps/CloudTunes — SalesforceSDK を使用するサンプルネイティブアプリケーション (『Mobile SDK Workbook』を参照)

Eclipse によるクリーンとビルド

使用する Android SDK Tools のバージョンによっては、ワークスペースのクリーン ([Project] > [Clean]) で問題が発生することがあります。特に、Android ライブラリプロジェクトに依存するプロジェクトはビルド順序に正しく従わないことがあります。このような場合にすべてのプロジェクトをクリーンすると、依存プロジェクトが依存先のライブラリプロジェクトを認識しなくなります。その結果、クリーン後にすべての非ライブラリプロジェクトでビルドエラーが発生します。

すべてのプロジェクトを再ビルトする場合は、まずライブラリプロジェクト (SalesforceSDK) をクリーンまたは再ビルトしてから、依存プロジェクトのクリーンと再ビルトを行うことをお勧めします。

Android 用の RestExplorer サンプルアプリケーション

RestExplorer は、OAuth と SalesforceSDK の REST API 機能の使い方を学習できるサンプルアプリケーションです。Android Honeycomb 搭載タブレットからさまざまな REST API アクションを確認するのにも便利です。

1. アプリケーションを実行するには、**RestExplorer** プロジェクトを右クリックして、[Run As] > [Android Application] を選択します。
2. テストを実行するには、**RestExplorerTest** プロジェクトを右クリックして、[Run As] > [Android JUnit Test] を選択します。

第 6 章

ハイブリッドアプリケーションの開発

トピック:

- ・ ハイブリッドアプリケーションのクイックスタート
- ・ 一覧情報を表示するモバイルページの作成
- ・ 詳細情報を表示するモバイルページの作成
- ・ Chatter によるソーシャルコラボレーション
- ・ iOS 用のハイブリッドサンプルアプリケーション
- ・ Android 用のハイブリッドサンプルアプリケーション

ハイブリッドアプリケーションは、HTML5 Web アプリケーションの開発しやすさと、ネイティブプラットフォームの機能性、操作性を併せ持っています。

ハイブリッドアプリケーションでは、HTML ファイルと JavaScript ファイルを使用します。これらのファイルは、デバイスまたはサーバ上に保存されます。

- ・ **デバイス** — JavaScript ライブラリ「forcetk」で開発されたハイブリッドアプリケーションは、Salesforce モバイルコンテナに Web アプリケーションをラップします。この方法により、JavaScript ファイルと HTML ファイルがデバイス上に保存されます。
- ・ **Salesforce モバイルコンテナ** — Visualforce テクノロジを使って開発されたハイブリッドアプリケーションは、HTML ファイルと JavaScript ファイルをサーバ上に保存します。配布には、ネイティブコンテナである Salesforce モバイルコンテナを使用します。

ハイブリッドアプリケーションのクイックスタート

次の手順で開始します。

1. 「ハイブリッドアプリケーションの開発要件」(ページ 58) をすべて満たしていることを確認します。
2. Mobile SDK をインストールします。
 - ・ 「Mobile SDK for iOS のインストール」(ページ 48) を参照してください。
 - ・ 「Mobile SDK for Android のインストール」(ページ 53) を参照してください。
3. 「リモートアクセスアプリケーションの作成」(ページ 33) の手順に従って、リモートアクセスアプリケーションを作成します。



メモ: コールバック URL の詳細は次のように指定します。

- ・ iOS の場合、<https://login.salesforce.com/services/oauth2/success> を使用します。
 - ・ Android の場合、myapp://mobilesdk/detect/oauth/done を使用します。ここで、myapp はアプリケーションに固有のプレフィックスを示します。
4. 「サンプルハイブリッドアプリケーションの実行」(ページ 59) の手順に従って、サンプルアプリケーションを実行します。

サンプルアプリケーションの実行後、次の手順に従って機能を追加できます。

1. 「一覧情報を表示するモバイルページの作成」(ページ 63)
2. 「詳細情報を表示するモバイルページの作成」(ページ 67)
3. 「カメラを使用するためのハイブリッドサンプルアプリケーションのカスタマイズ」(ページ 114)

ハイブリッドアプリケーションの開発要件

すべてのターゲットデバイスで、以下を用意しておく必要があります。

- ・ Ant 1.8.0 以上
- ・ Git (詳細は、<http://help.github.com/set-up-git-redirect> を参照してください。)

iOS デバイス向けのアプリケーションを開発する場合は、**Xcode 4.2 以上** も必要です。

Android デバイス向けのアプリケーションを開発する場合は、以下も必要です。

- ・ Eclipse Classic
- ・ Android SDK (r20 以上)
- ・ ADT Plugin (r20 以上)

iOS 向けハイブリッドアプリケーションプロジェクトの作成

新しいプロジェクトを作成する手順は、次のとおりです。

1. Xcode で、「Hybrid Force.com App」プロジェクトを新規作成します (Command + Shift + N)。次のパラメータを指定する必要があります。
 - **Consumer Public Key:** リモートアクセスアプリケーションのコンシューマ鍵。
 - **OAuth Redirect URL:** リモートアクセスアプリケーションのコールバック URL。
 - **Company Identifier:** Apple iOS Dev Center のアカウントで作成したアプリケーション ID と一致している必要があります (「com.mycompany.foo」など)。
 - Use Automatic Reference Counting: オフにします。

Android 向けハイブリッドプロジェクトの作成

新しいプロジェクトを作成する手順は、次のとおりです。

1. Eclipse で、[File] > [Import] > [General] > [Existing Projects into Workspace] の順にクリックします。
2. **ContactExplorer** というサンプルプロジェクト (SalesforceMobileSDK-Android/hybrid/SampleApps/ContactExplorer) をルートディレクトリとして選択します。[Copy projects into workspace] が選択されていることを確認して、[Finish] をクリックします。
3. カスタマイズ前の ContactExplorer サンプルと区別するため、ContactExplorer プロジェクトを右クリックして名前を変更します。/res/values/strings.xml を開き、app_name に新しいプロジェクト名を指定します。
4. リモートアクセスパラメータを使ってアプリケーションを設定するには、assets/www/bootconfig.js を開き、次の値を編集します。
 - remoteAccessConsumerKey: リモートアクセスアプリケーションのコンシューマ鍵。
 - oauthRedirectURI: リモートアクセスアプリケーションのコールバック URL。
5. まだ Android Virtual Device を作成していない場合は、作成します。Eclipse で、[Window] > [AVD Manager] を選択し、[New] をクリックします。必要に応じて、デバイスでのカメラのサポートを有効にします。

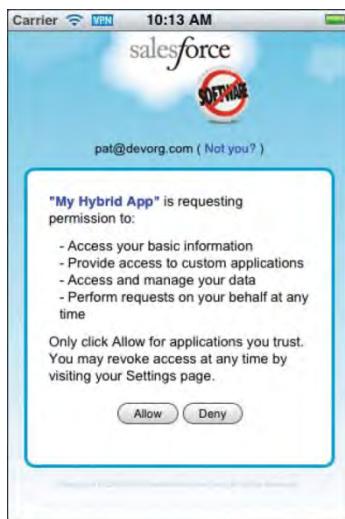
サンプルハイブリッドアプリケーションの実行

シミュレータまたは実際のデバイスでサンプルプロジェクトをコンパイルし、実行します。iOS、Android のどちらの環境でも、必要に応じてシミュレータまたは接続済みデバイスを選択してアプリケーションを実行できます。iOS デバイスを使用する場合は、『Xcode 4 User Guide』の説明に従ってプロファイルを設定する必要があります。Android デバイスを使用する場合は、Android の開発者向けドキュメントの説明に従って Android デバイスを設定する必要があります。

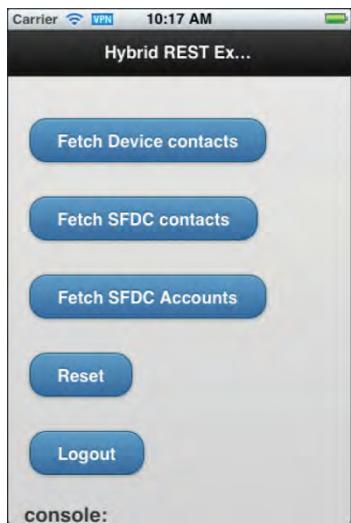
どちらの場合も、最初にスプラッシュスクリーンが表示され、続いて Salesforce ログイン画面が表示されます。



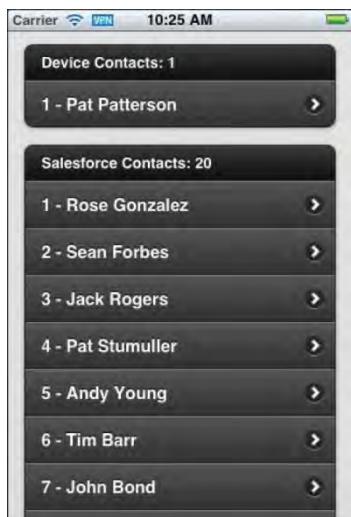
Developer Edition のユーザ名とパスワードを使ってログインすると、アプリケーションに Salesforce 内のデータへのアクセスを許可するかどうかを確認するメッセージが表示されます。



[許可] をタップすると、Developer Edition 上の取引先や取引先責任者のリストを取得できるようになります。



画面をタップして、取引先 (Account) レコードと取引先責任者 (Contact) レコードを取得します。



このアプリケーションでは、デバイスから取引先責任者情報を取得することもできます。これは Web アプリケーションにはない機能です。その仕組みを詳しく見ていきましょう。

サンプルアプリケーションの仕組み

ログインプロセスが完了すると、www フォルダ内の index.html が表示されます。ページがロードされ、モバイルフレームワークの準備が完了すると、onDeviceReady() 関数が inline.js 内の regLinkClickHandlers() をコールします。regLinkClickHandlers() は、サンプルアプリケーション内のさまざまな機能を制御する 5 つのクリックハンドラを設定します。

```
$j('#link_fetch_device_contacts').click(function() {
    SFHybridApp.logToConsole("link_fetch_device_contacts clicked");
    var options = new ContactFindOptions();
    options.filter = ""; // empty search string returns all contacts
    options.multiple = true;
    var fields = ["name"];
    navigator.contacts.find(fields, onSuccessDevice,
        onErrorDevice, options);
});
```

この #link_fetch_device_contacts ハンドラは、navigator.contacts オブジェクトに対して find() をコールし、デバイスから取引先責任者リストを取得します。onSuccessDevice() 関数により、index.html ページに取引先責任者リストが表示されます。

```
$j('#link_fetch_sfdc_contacts').click(function() {
    SFHybridApp.logToConsole("link_fetch_sfdc_contacts clicked");
    forcetkClient.query("SELECT Name FROM Contact",
        onSuccessSfdcContacts, onErrorSfdc);
});
```

#link_fetch_sfdc_contacts ハンドラは、forcetkClient オブジェクトを使ってクエリを実行します。このオブジェクトは最初の OAuth 2.0 認証プロセスで設定され、認証済みユーザのコンテキスト内で Force.com REST API へのアクセスを許可します。ここで Developer Edition アカウントのすべての取引先責任者の名前を取得すると、onSuccessSfdcContacts() により、index.html ページに取引先責任者のリストが表示されます。

```
$j('#link_fetch_sfdc_accounts').click(function() {
    SFHybridApp.logToConsole("link_fetch_sfdc_accounts clicked");
    forcetkClient.query("SELECT Name FROM Account",
        onSuccessSfdcAccounts, onErrorSfdc);
});
```

#link_fetch_sfdc_accounts ハンドラは #link_fetch_sfdc_contacts ハンドラと同様の働きをし、Force.com REST API を使用して取引先レコードを取得します。残りのハンドラ #link_reset と #link_logout は、それぞれ表示されたリストのクリアとユーザのログアウトを実行します。

一覧情報を表示するモバイルページの作成

サンプルハイブリッドアプリケーションは多くの点で有用であり、ハイブリッドモバイルアプリケーション開発の基本的な事項を学ぶのに最適です。このチュートリアルでは、カスタムの Warehouse (商品在庫管理) アプリケーションスキーマを使用して Merchandise (商品) レコードを表示するように、サンプルのハイブリッドモバイルアプリケーションに変更を加えます。

Warehouse スキーマは、次のオンラインコンテンツを使って簡単に作成できます。

http://wiki.developerforce.com/page/Developing_Cloud_Apps---Coding_Optional

アプリケーションの初期化ブロックの変更 (index.html)

このセクションでは、アプリケーションを Warehouse スキーマに合わせてカスタマイズし、Merchandise カスタムオブジェクト内のすべてのレコードを表示するように、ビューファイル (index.html) とコントローラ (inline.js) に変更を加えます。

モバイルアプリケーションのデフォルトのホームページに Merchandise レコードのリストを表示するには、まず、アプリケーションが `onDeviceReady` 関数をコールしたときに自動的に行われる処理を変更する必要があります。`index.html` 内のサンプルの `salesforceSessionRefreshed` 関数の末尾に、次のコードを追加します。

```
// log message
SFHybridApp.logToConsole("Calling out for records");
// register click event handlers --see inline.js
regLinkClickHandlers();
// retrieve Merchandise records, including the Id for links
forcetkClient.query("SELECT Id, Name, Price__c, Quantity__c
FROM Merchandise__c", onSuccessSfdcMerchandise, onErrorSfdc);
```

この JavaScript コードは forcetk ライブラリを利用し、単純な SOQL ステートメントで Force.com データベースをクエリして、Merchandise カスタムオブジェクトのすべてのレコードを取得します。成功すると、JavaScript 関数 `onSuccessSfdcMerchandise` をコールします。この JavaScript 関数は後で作成します。

アプリケーションの mainpage ビューの作成 (index.html)

標準的なタッチ式のモバイルユーザインターフェースに Merchandise レコードを表示するには、`index.html` を開いて `<body>` タグまでスクロールし、このタグ全体を次の HTML で置き換えます。

```
<!--Main page, to display list of Merchandise once app starts -->
<div data-role="page" data-theme="b" id="mainpage">
    <!--page header -->
    <div data-role="header">
        <!--button for logging out -->
        <a href='#' id="link_logout" data-role="button"
            data-icon='delete'>
            Log Out
        </a>
        <!--page title -->
        <h2>List</h2>
    </div>
    <!--page content -->
    <div id="#content" data-role="content">
        <!--page title -->
        <h2>Mobile Inventory</h2>
        <!--list of merchandise, links to detail pages -->
        <div id="div_merchandise_list">
            <!--built dynamically by function onSuccessSfdcMerchandise -->
            </div>
        </div>
    </div>
</div>
```

変更後のビューでは、魅力的なタッチインターフェースをアプリケーションで提供するため、標準 HTML タグに加えて jQuery Mobile マークアップ (data-role、data-theme、data-icon など) を使用しています。HTML、CSS、JavaScript、jQuery といった標準的な Web 開発技術を利用したことがあるなら、ハイブリッドモバイルアプリケーションの開発は簡単です。

アプリケーションのコントローラの変更 (inline.js)

前のセクションでは、初期化ブロックがコントローラの `onSuccessSfdcMerchandise` 関数に従って、`div_merchandise_list` 内に Merchandise レコードのリストを表示する HTML を動的に生成するようにビューファイルとコントローラを変更しました。このセクションでは、`onSuccessSfdcMerchandise` 関数を作成します。

`inline.js` ファイルをロードし、サンプルの関数とやや似た次のコントローラアクションを追加します。

```
// handle successful retrieval of Merchandise records
function onSuccessSfdcMerchandise(response) {
    // avoid jQuery conflicts
    var $j = jQuery.noConflict();

    // debug info to console
    SFHybridApp.logToConsole("onSuccessSfdcMerchandise: received " +
        response.totalSize + " merchandise records");

    // clear div_merchandise_list HTML
    $j("#div_merchandise_list").html("");

    // set the ul string var to a new UL
    var ul = $j('<ul data-role="listview" data-inset="true"'
        " data-theme="a" data-dividertheme="a"></ul>');
    // update div_merchandise_list with the UL
    $j("#div_merchandise_list").append(ul);
```

```

// set the first li to display the number of records found
// formatted using list-divider
ul.append($j('<li data-role="list-divider">Merchandise records: ' +
    + response.totalSize + '</li>'));
// add an li for the merchandise being passed into the function
// create array to store record information for click listener
inventory = new Array();
// loop through each record, using vars i and merchandise
$j.each(response.records, function(i, merchandise) {
    // create an array element for each merchandise record
    inventory[merchandise.Id] = merchandise;
    // create a new li with the record's Name
    var newLi = $j("<li class='detailLink' data-id='" +
merchandise.Id +
        + "'><a href='#" + merchandise.Name + "</a></li>");
    ul.append(newLi);
});

// render (create) the list of Merchandise records
$j("#div_merchandise_list").trigger( "create" );
// send the rendered HTML to the log for
debuggingSFHybridApp.logToConsole($j("#div_merchandise_list").html());

// set up listeners for detailLink clicks
$j(".detailLink").click(function() {
    // get the unique data-id of the record just clicked
    var id = $j(this).attr('data-id');
    // using the id, get the record from the array created above

    var record = inventory[id];

    // use this info to set up various detail page information
    $j("#name").html(record.Name);
    $j("#quantity").val(record.Quantity__c);
    $j("#price").val(record.Price__c);
    $j("#detailpage").attr("data-id",record.Id);

    // change the view to the detailpage
    $j.mobile.changePage('#detailpage', {changeHash: true});
});

}

```

各行の説明については、コード内のコメントをご覧ください。SFHybridApp.logToConsole();へのコールにより、コンソールログに HTML が outputされ、コードから生成された内容を確認できます。以下は、出力のサンプルです。

```

<ul data-role="listview" data-inset="true" data-theme="a"
    data-dividertheme="a" class="ui-listview ui-listview-inset
    ui-corner-all ui-shadow">
    <li data-role="list-divider" role="heading"
        class="ui-li ui-li-divider ui-btn ui-bar-a
        ui-corner-top">Merchandise records: 6
    </li>
    <li class="detailLink ui-btn ui-btn-up-a ui-btn-icon-right ui-li"
        data-id="a00E000003BzSfIAK" data-theme="a">
        <div class="ui-btn-inner ui-li">
            <div class="ui-btn-text">
                <a href="#" class="ui-link-inherit">Tablet</a>
            </div>
        </div>
    </li>

```

```
</div>
</li>
<li class="detailLink ui-btn ui-btn-up-a ui-btn-icon-right ui-li" data-id="a00E0000003BuUPIAK" data-theme="a">
<div class="ui-btn-inner ui-li">
<div class="ui-btn-text">
<a href="#" class="ui-link-inherit">Laptop</a>
</div>
</div>
</li>
...
</ul>
```

コードの次の点に注目してください。

- ・ アプリケーションのメインページに表示される Merchandise レコードのリストがどのようにして生成されるのか
- ・ Merchandise レコードの Name (名前) を表示する各リスト項目がどのようにして生成されるのか
- ・ 詳細ページの表示内容を決定する一意のリンク情報に関連付けられた各リスト項目がどのようにして生成されるのか

作成したアプリケーションのテスト

モバイルアプリケーションのシミュレータを起動します。最初のページは、次のようにになります。



Merchandise レコードをクリックしても、まだ何も起こりません。リスト機能は単独で使用しても便利ですが、詳細ビューと組み合わせると、さらに有用です。次のセクションでは、ユーザが特定の Merchandise レコードをクリックしたときに表示される詳細ページ (detailpage) を作成します。

詳細情報を表示するモバイルページの作成

前のトピックでは、サンプルハイブリッドアプリケーションに変更を加えて、起動時にすべての Merchandise レコードと詳細ページへのリンクが表示されるようにしました。ここでは、詳細ページを表示する detailpage ビューを作成し、アプリケーションのコントローラを更新します。

アプリケーションの detailpage ビューの作成 (index.html)

ユーザがアプリケーションの mainpage ビューで Merchandise レコードをクリックすると、クリックリスナーによりレコード固有の情報が生成され、この情報を表示する detailpage という名前のビューがロードされます。detailpage ビューを作成するには、mainpage の div タグの後ろに次の div タグを追加します。

```
<!--Detail page, to display details when user clicks specific
Merchandise record -->
<div data-role="page" data-theme="b" id="detailpage">
    <!--page header -->
    <div data-role="header">
        <!--button for going back to mainpage -->
        <a href="#mainpage" id="backInventory"
            class='ui-btn-left' data-icon='home'>
            Home
        </a>
        <!--page title -->
        <h1>Edit</h1>
    </div>
    <!--page content -->
    <div id="#content" data-role="content">
        <h2 id="name"></h2>
        <label for="price" class="ui-hidden-accessible">
            Price ($):
        </label>
        <input type="text" id="price" readonly="readonly"></input>
        <br/>
        <label for="quantity" class="ui-hidden-accessible">
            Quantity:
        </label>
        <!--note that number is not universally supported -->
        <input type="number" id="quantity"></input>
        <br/>
        <a href="#" data-role="button" id="updateButton"
            data-theme="b">Update</a>
    </div>
</div>
```

各部分の説明については、コード内のコメントをご覧ください。このビューは、基本的に、Merchandise レコードの Price (価格) 項目と Quantity (数量) 項目を表示するフォームになっており、ユーザは必要に応じて Quantity 項目を更新できます。

前述の inline.js に作成した onSuccessSfdcMerchandise 関数の最後の部分でコールされる jQuery を使用して、Merchandise レコードの値で詳細ページ要素が更新されます。必要に応じて、該当のコードをもう一度確認してみてください。

アプリケーションのコントローラの変更 (inline.js)

ユーザが detailpage ビューの [Update] ボタンをクリックしても、まだ何も起こりません。[Update] ボタンのクリックを処理するよう、アプリケーションのコントローラ (inline.js) を変更する必要があります。

inline.js の regLinkClickHandlers 関数の末尾に次の JavaScript を追加します。

```
// handle clicks to Update on detailpage
$( "#updateButton" ).click(function() {
    // update local information in the inventory array
    inventory[$j( "#detailpage" ).attr("data-id")].Quantity__c =
    $j( "#quantity" ).val();
    currentRecord = inventory[$j( "#detailpage" ).attr("data-id")];

    // strip out ID before updating the database
    var data = new Object();
    data.Quantity__c = currentRecord.Quantity__c;
    // update the database
    forcetkClient.update("Merchandise__c", currentRecord.Id,
        data, updateSuccess, onErrorSfdc);
});

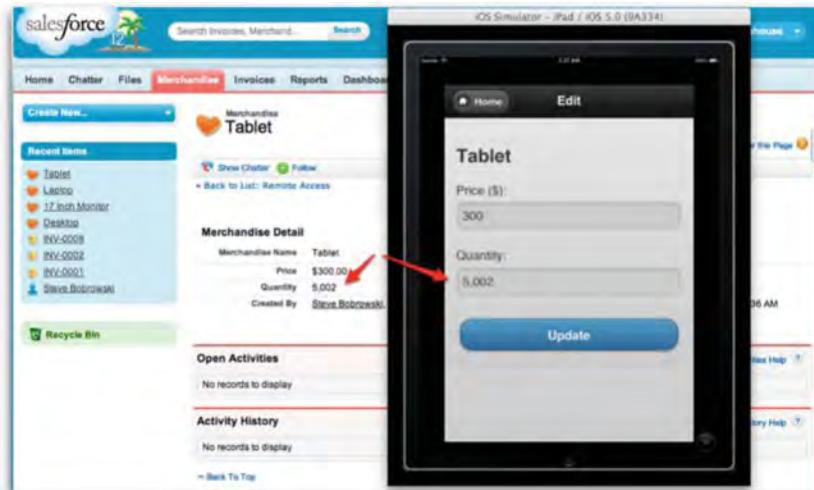
});
```

各行の説明については、コード内のコメントをご覧ください。成功すると、新しいハンドラにより updateSuccess 関数が呼び出されます。この関数は上のコードにはまだ含まれていません。次のコードを inline.js に追加します。

```
function updateSuccess(message) {
    alert("Item Updated");
}
```

アプリケーションのテスト

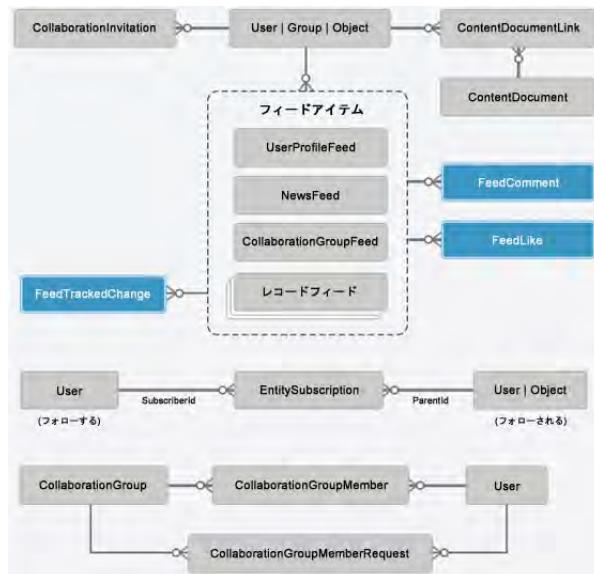
モバイルアプリケーションのシミュレータを起動します。特定の Merchandise レコードをクリックすると、次のような詳細ページが表示されます。



レコードの Quantity 項目は自由に変更できます。変更後 Developer Edition 組織にログインし、Force.com アプリケーションユーザインターフェース上の該当レコードで、Quantity 項目への変更が反映されていることを確認してください(上の図を参照)。

Chatter によるソーシャルコラボレーション

開発者の観点で言えば、Chatter は、組織内のレコードのソーシャルデータを管理する複数の標準 Force.com オブジェクトから成るデータモデルです。



- フィードアイテム:** 各フィードアイテムは、特定のレコードに対する変更や、特定のユーザまたはレコードへの投稿に対する変更を表します。ユーザがフィードにアイテムを投稿するとき、フィードアイテムの ParentId にはユーザの UserId が含まれます。クエリやステートメントによっては(たとえば、コメントの追加など)、フィードアイテムの ID が必要になります。
- FeedComment:** FeedComment オブジェクトにはコメントが格納されます。このオブジェクトは、関連するレコードフィードアイテムの子オブジェクトです。

アプリケーションは、Chatter REST API を使用して Chatter データモデルにアクセスすることができます。Chatter REST API を使用すると、標準の JSON/XML ベースの API を介して、Chatter フィードや、フィードアイテム、コメントなどのソーシャルデータにアクセスすることができます。

Chatter REST API の詳細については、ここでは取り上げません。概要は、『Chatter REST API Cheat Sheet』(http://wiki.developerforce.com/page/Cheat_Sheets) を参照してください。

アプリケーションのビューの変更 (index.html)

まず、アプリケーションのビュー (index.html) に新しいモバイルページ chatterpage を追加します。ソースコードは、<https://gist.github.com/3644284> からダウンロードできます。

```
<!--Detail page, to display Chatter Information -->
<div data-role="page" data-theme="b" id="chatterpage">
    <!--page header -->
    <div data-role="header">
        <!--button for going back to detailpage -->
        <a href="#detailpage" id="backInventory" class='ui-btn-left' data-icon='arrow-1'>
            Edit
        </a>
        <!--page title -->
        <h1>Collaborate</h1>
    </div>
    <!--page content -->
    <div id="#content" data-role="content">
        <h2 id="name"></h2>
        <div id="div_chatter_list">
            <!--built dynamically by controller function
updateChatterList -->
        </div>
    </div>
</div>
```

アプリケーションのコントローラの変更 (inline.js)

Mobile SDK に含まれる forcetk ライブラリは、Force.com REST API のラッパーです。ただし、本ドキュメントの執筆時点ではアプリケーションで Chatter REST API はサポートされていません。このため、まずは forcetk.Client オブジェクトに 3 つの新しい関数を追加する必要があります。inline.js の先頭に、以下のコードを追加します。ソースコードは、<https://gist.github.com/3644304> からダウンロードできます。

```
// add select Chatter functions to forcetk
// get feed-items
forcetk.Client.prototype.chatterFeed = function(id, callback, error) {
    this.ajax('/' + this.apiVersion + '/chatter/feeds/record/' + id +
    '/feed-items', callback, error);
}

// post feed item
forcetk.Client.prototype.postChatterItem = function(id, text, callback,
error) {
    this.ajax('/' + this.apiVersion + '/chatter/feeds/record/' + id +
'/feed-items', callback, error, "POST", '{ "body" : { "messageSegments" :
[ { "type": "Text", "text" : "' + text + '" } ] } }');
}

// post feed comment
forcetk.Client.prototype.postChatterComment = function(id, text,
callback, error) {
    this.ajax('/' + this.apiVersion + '/chatter/feed-items/' + id +
'/comments', callback, error, "POST", '{ "body" : { "messageSegments" :
[ { "type": "Text", "text" : "' + text + '" } ] } }');
}
```

これらの新しい関数は、Chatter REST API をコールして、フィードアイテム（と関連するすべての投稿およびコメント）の取得、フィードへの新しいアイテムの投稿、既存のフィードアイテムへの新しいコメントの投稿という 3 つの操作をそれぞれ実行します。

`regLinkClickHandlers()` 関数に次のコードを追加します。これから `detailpage` に追加する新しいボタンのクリックを監視し、対応するアクションを実行するためのコードです。ソースコードは、<https://gist.github.com/3644348> からダウンロードできます。

```
// handle clicks to Collaborate on detailpage
$j("#chatterButton").click(function() {
    // using the id of the current Merchandise record, get its Chatter
    feed items
    SFHybridApp.logToConsole("Getting Chatter");
    forcetkClient.chatterFeed($j("#detailpage").attr("data-id"),
    updateChatterList, onErrorSfdc);
});
```

このコードは、新しい `forcetkClient chatterFeed()` 関数をコールします。この関数は、データベースを 1 回呼び出すだけで、最新の Merchandise レコードのフィードアイテム（関連する投稿やコメントを含む）を取得できる、非常に効率的な関数です。効率性は、モバイルアプリケーションの設計上の重要な目標でもあります。ただし、この関数を使用するには、`updateChatterList()` 関数が必要です。この `updateChatterList()` 関数と、関連する `refreshChatterList()` 関数を次に追加します。

`inline.js` に次のコードを追加してください。ソースコードは、<https://gist.github.com/3644332> からダウンロードできます。

```

function refreshChatter(response) {
    forcetkClient.chatterFeed($("#detailpage").attr("data-id"),
updateChatterList, onErrorSfdc);
}
function updateChatterList(response) {
    // output debug information to the log
    SFHybridApp.logToConsole("Got Chatter");
    SFHybridApp.logToConsole(response.items.length);

    // clear div_chatter_list HTML
    $("#div_chatter_list").html("");

    // loop through all items and display UI
    $j.each(response.items, function(i, chatter) {
        // open a new div
        var newItemDiv = $j("<div class='ui-body ui-body-b'>");
        // append the item author name
        newItemDiv.append($j("<h5>" + chatter.actor.name + " said
...</h5>"));
        // append the item text
        newItemDiv.append($j("<p>" + chatter.body.text + "</p>"));

        // display item comments
        var newCommentDiv;
        $j.each(chatter.comments.comments, function(i, comment) {
            // reset newCommentDiv to open the div
            newCommentDiv = $j("<div class='ui-body ui-body-c'>");
            // append the item author name
            newCommentDiv.append($j("<h5>" + comment.user.name + " commented
...</h5>"));
            // append the item text
            newCommentDiv.append($j("<p>" + comment.body.text + "</p>"));
            // append the closing inner div tag
            newCommentDiv.append($j("</div>"));
            // append newCommentDiv to newItemDiv
            newItemDiv.append(newCommentDiv);
        });
        // append a comment button to the item
        newItemDiv.append($j("<a href='#' data-role='button' data-min='true'
class='comment' data-theme='b' data-inline='true' data-icon='plus'
data-id='" + chatter.id + "'>Your Comment</a>"));

        // append the closing outer div tag
        newItemDiv.append($j("</div>"));

        // add the final item output to the div
        $j("#div_chatter_list").append(newItemDiv);
    });
}

var newPostButtonDiv = $j("<a href='#' data-role='button'
data-min='true' class='post' data-theme='b' data-icon='plus'
data-inline='true' class='post'>New Post</a>");
$j("#div_chatter_list").append(newPostButtonDiv);

// set up listeners for chatterButton clicks
$j("a.comment").click(function() {
    SFHybridApp.logToConsole("Commenting");

    var id = $j(this).attr('data-id');

```

```
var post = prompt('Enter New Comment');
if (post != null) {
    forcetkClient.postChatterComment(id, post, refreshChatter,
onErrorSfdc);
}

});

// set up listeners for chatterButton clicks
$j("a.post").click(function() {
    SFHybridApp.logToConsole("Posting");

    var id = $j("#detailpage").attr("data-id");
    SFHybridApp.logToConsole('detailpage id.');
    SFHybridApp.logToConsole(id);
    var post = prompt('Enter New Post');
    if (post != null) {
        forcetkClient.postChatterItem(id, post, refreshChatter,
onErrorSfdc);
    }
}

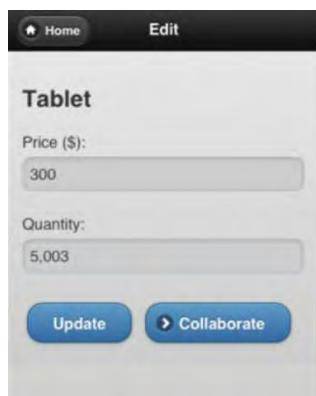
);

// render the final chatter list
$j("#div_chatter_list").trigger("create");
// log debug information
SFHybridApp.logToConsole('Item output div.');
SFHybridApp.logToConsole($j("#div_chatter_list").html());

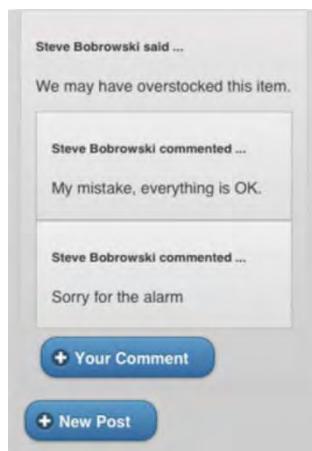
// change the view to the detailpage, tracking the location change
$j.mobile.changePage('#chatterpage', {
    changeHash: true
});
}
```

アプリケーションの動作確認

アプリケーションのビューとコントローラの更新が完了し、Chatter がサポートされるようになりました。次に、このアプリケーションの動作を確認します。アプリケーションを再ビルトし、IDE のシミュレータで実行します。次に、Chatter フィードアイテムが関連付けられている Merchandise レコードをクリックします(対象レコードを検索する必要がある場合、ブラウザで Warehouse アプリケーションを使用します)。モバイルアプリケーションの詳細ページ(detailpage)で、新しい [Collaborate] ボタンをタップします。



すると、モバイルアプリケーションの新しい chatterpage ビューが開き、関連する投稿とコメントを含む、すべての Chatter フィードアイテムが表示されます。新しい投稿または既存の投稿へのコメントを追加して、この操作が Warehouse アプリケーションにも反映されることを確認します。



これで、複数のユーザが Merchandise レコードを共有してコラボレーションできる、ソーシャルなモバイルアプリケーションが完成しました。

iOS用のハイブリッドサンプルアプリケーション

hybrid/ フォルダ内のサンプルアプリケーションは、PhoneGap SDK で設計されています。これらのアプリケーションを使用するには、PhoneGap の Web サイトから PhoneGap SDK をダウンロードしてインストールする必要があります。PhoneGap SDK の詳しいインストール手順や操作方法は、「Getting Started Guides」に記載されています。

ハイブリッドサンプルアプリケーションは、/Users/Shared/PhoneGap/Frameworks/PhoneGap.framework 内の PhoneGap iOS フレームワークを参照するよう設定されています。このフレームワークが別の場所に格納されている場合、正しくロードされない可能性があります。PhoneGap フレームワークがサンプルプロジェクトに正しくリンクされているかどうかを確認するには、次の手順を実行します。

1. Xcode でプロジェクトを開きます。
2. プロジェクトナビゲータで [Frameworks] フォルダを展開します。
3. PhoneGap.framework が設定済みフレームワークのリストに表示されている場合、プロジェクトに手を加える必要はありません。

PhoneGap フレームワークの場所がリストに表示されていない場合や、PhoneGap フレームワークが見つからないためコンパイルエラー（「Undefined symbols for architecture i386: "_OBJC_METACLASS_\$_PhoneGapDelegate"」など）が発生した場合は、次の手順で、サンプルプロジェクトに PhoneGap フレームワークを追加する必要があります。

1. サンプルアプリケーションの Xcode プロジェクトを開きます。
2. プロジェクトナビゲータで、[Frameworks] フォルダを右クリックするか Ctrl ボタンを押したままクリックして、[Add files to "Project Name..."] を選択します。
3. PhoneGap.framework フォルダ（デフォルトの場所は /Users/Shared/PhoneGap/Frameworks/PhoneGap.framework/）に移動し、[Add] をクリックします。

これでサンプルアプリケーションプロジェクトが正常にビルドされ、実行されます。

Android用のハイブリッドサンプルアプリケーション

HYBRID_DIR には、複数のプロジェクトが含まれています。

- **SampleApps/ContactExplorer:** ContactExplorer サンプルアプリケーションは、PhoneGap (コードバック) を使ってローカルデバイスの取引先責任者を取得します。また、forcetk.js ツールキットを使って、Salesforce REST API を操作する REST トランザクションを実装します。このアプリケーションは Salesforce SDK の OAuth2 サポートを使用して OAuth 認証情報を取得し、JavaScript イベントを送信して、これらの認証情報を forcetk.js に伝えます。
- **SampleApps/VFConnector:** VFConnector サンプルアプリケーションは、ネイティブコンテナに Visualforce ページをラップする方法を示します。このサンプルでは、組織内に BasicVFTest という Visualforce ページがあると仮定しています。アプリケーションはまず、Salesforce SDK の OAuth2 サポートを利用して、OAuth ログイン認証情報を取得します。次に、これらの認証情報を使って、Visualforce ページにアクセスするために使用する適切な WebView Cookie を設定します。

- **SmartStorePluginTest:** SmartStore phonegap プラグインのテストプロジェクトです。

hybrid/ フォルダ内のサンプルアプリケーションは、PhoneGap SDK で設計されています。これらのアプリケーションを使用するには、PhoneGap の Web サイトから PhoneGap SDK 1.0.0 以上をダウンロードしてインストールする必要があります。PhoneGap SDK の詳しいインストール手順や操作方法は、「Getting Started Guide」に記載されています。

これらのサンプルアプリケーションは、/Users/Shared/PhoneGap/Frameworks/PhoneGap.framework 内の PhoneGap Android フレームワークを参照するよう設定されています。このフレームワークが別の場所に格納されている場合、正しくロードされない可能性があります。PhoneGap フレームワークがサンプルプロジェクトに正しくリンクされているかどうかを確認するには、次の手順を実行します。

1. Eclipse でプロジェクトを開きます。
2. プロジェクトナビゲータでプロジェクトフォルダを展開します。
3. PhoneGap.framework が設定済みフレームワークのリストに表示されている場合、プロジェクトを変更する必要はありません。

PhoneGap フレームワークの場所がリストに表示されていない場合や、PhoneGap フレームワークが見つからないためコンパイルエラー（「Undefined symbols for architecture i386: "_OBJC_METACLASS_\$_PhoneGapDelegate"」など）が発生した場合は、次の手順で、サンプルプロジェクトに PhoneGap フレームワークを追加する必要があります。

1. サンプルアプリケーションのプロジェクトを開きます。
2. プロジェクトナビゲータで、[Frameworks] フォルダを右クリックするか Ctrl ボタンを押したままクリックして、[Add files to "Project Name..."] を選択します。
3. PhoneGap.framework フォルダ（デフォルトの場所は /Users/Shared/PhoneGap/Frameworks/PhoneGap.framework/）に移動し、[Add] をクリックします。

これでサンプルアプリケーションプロジェクトが正常にビルドされ、実行されます。

第 7 章

Mobile Components for Visualforce ライブラリを使用したハイブリッド開発

トピック:

- Mobile Components for Visualforce のアーキテクチャ
- Visualforce のオープンソースモバイルコンポーネント
- コンポーネントのインストール
- タブレットページの作成
- Visualforce モバイルコンポーネントの作成

Visualforce の柔軟なコンポーネントインフラストラクチャを利用して、低レベルのコードをラップして再利用可能なコンポーネントを作成し、Force.com プラットフォーム上でカスタムアプリケーションを開発することができます。Visualforce を利用してハイブリッドモバイルアプリケーションも作成できます。

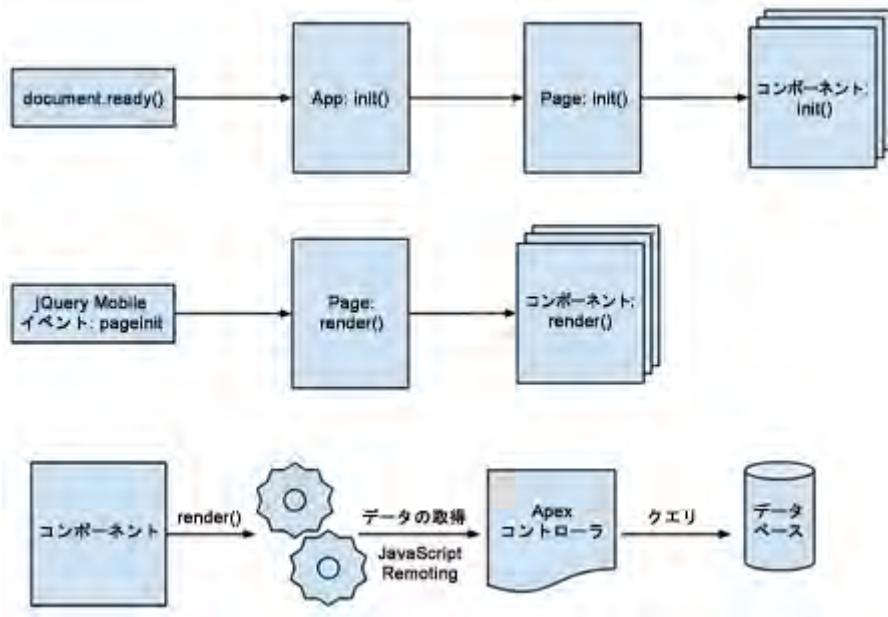
オープンソースの Mobile Components for Visualforce ライブラリには、クロスプラットフォームの HTML5 出力を生成する軽量の UI コンポーネントが含まれています。これらのアプリケーションは、ブラウザに展開することも、ハイブリッド Mobile SDK コンテナに組み込むこともできます。このライブラリのソースコードは、Github からダウンロードできます。

<https://github.com/forcedotcom/MobileComponents/tree/master/Visualforce>

このコンポーネントライブラリを利用すると、複雑なモバイルフレームワークや設計パターンにとらわれることなく、Visualforce を使って、モバイルアプリケーションを短時間で簡単に開発できます。このライブラリには、単純なコンポーネントインターフェースで使用できる、モバイル開発のフレームワークとベストプラクティスが含まれています。

Mobile Components for Visualforce のアーキテクチャ

次の図に、Mobile Components for Visualforce のアーキテクチャの概要を示します。



Visualforce のオープンソースモバイルコンポーネント

Visualforce ライブラリには、次のコンポーネントが含まれています。

- **App コンポーネント** — モバイルアプリケーション開発に使用するすべての設定と構成要素 (jQuery、jQuery Mobile を含む) を提供します。
- **Navigation コンポーネント** — このコンポーネントを使って、jQuery Mobile ページ間を移動するためのフックを生成できます。
- **SplitView テンプレート** — ページ上に分割ビューを表示するためのコンポーネントを提供します。ランドスケープモードの場合、左側にメニューセクションが、右側に幅広のメインセクションが表示されます。ポートレートモードの場合、左側のメニューはポップオーバー表示になります。
- **List コンポーネント** — このコンポーネントを使って、任意の sObject のレコードリストをすばやく表示できます。このコンポーネントの動作は、さまざまな属性や JavaScript フックを使用することにより簡単に制御できます。
- **Detail コンポーネント** — このコンポーネントを使って、任意の sObject の詳細を簡単に表示できます。さまざまな属性や JavaScript フックを使用することにより、動作を簡単に制御できます。
- **Page コンポーネント** — `data-role="page"` という属性を持つ jQuery Mobile ラッパーを提供します。

- **Header コンポーネント** — Page コンポーネント内のヘッダーセクションに、`data-role="header"` という属性を持つ jQuery Mobile ラッパーを提供します。
- **Content コンポーネント** — Page コンポーネント内のコンテンツセクションに、`data-role="content"` という属性を持つ jQuery Mobile ラッパーを提供します。
- **Footer コンポーネント** — Page コンポーネント内のフッターセクションに、`data-role="footer"` という属性を持つ jQuery Mobile ラッパーを提供します。



メモ: これらの Visualforce コンポーネントはオープンソースであり、Salesforce では公式にサポートされていません。

Visualforce App コンポーネント

このコンポーネントライブラリを使って作成されたすべてのモバイル Visualforce ページは、App コンポーネント内にラップする必要があります。App コンポーネントは、ビューポート設定、JavaScript、スタイルシートなど、モバイルアプリケーションの主要な構成要素を提供します。App コンポーネントの `debug` 属性では、開発モードと本番モードのどちらで実行するかを指定できます。後者の場合、簡易版のアセットが提供されます。

```
<c:App debug="true"></c:App>
```

Visualforce Navigation コンポーネント

Navigation コンポーネントでは、jQuery Mobile ページ間のナビゲーションに使用するフックを作成できます。

Visualforce SplitView テンプレート

SplitView テンプレートは App コンポーネント内で使用します。このテンプレートで提供される分割ビューでは、左側に細いリストビュー用セクション、右側にレコードの詳細を表示する幅広のセクションが表示されます。

```
<apex:composition template="SplitViewTemplate"></apex:composition>
```

SplitView テンプレートには、定義の必要な 2 つのセクションがあります。

- “menu” は、ランドスケープモードの分割ビューの左側のセクションです。ユーザがタブレットを回転させてポートレートモードに切り替えると、このセクションはポップオーバー表示になります。
- “main” は分割ビューの右側の幅広のセクションです。このセクションは、ポートレートモード、ランドスケープモードのどちらの場合にも表示されます。

```
<apex:define name="menu">
    <c:Page name="list">
        <c:Header>
            <h1 style="font-size: 20px; margin: 0px;">All Contacts</h1>

        </c:Header>
        <c:Content>
            <c>List sobject="Contact" labelField="Name"
subLabelField="Account.Name" sortByField="FirstName" listFilter="true"/>
        </c:Content>
    </c:Page>
</apex:define>
```

Visualforce Page コンポーネント

固定の Header コンポーネントと Footer コンポーネント、およびその間に挟まれたスクロール可能なコンテンツセクションを定義する、重要なラッパーになります。

- Header コンポーネントは、セクションの固定タイトルを記述します。
- Content コンポーネントは、スクロール可能なコンテンツセクションについて記述します。
- Content コンポーネントの本文では、List コンポーネントを使って取引先責任者リストを取得、表示します。

Visualforce Header コンポーネント

Page コンポーネント内のヘッダーセクションに、`data-role="header"` という属性を持つ jQuery Mobile ラッパーを提供します。Header コンポーネントには通常、`<H1>` タグが含まれています。Header コンポーネントと Footer コンポーネントは固定されており、その間にスクロール可能なコンテンツを挿入できます。

Visualforce Content コンポーネント

Content コンポーネントには、List コンポーネント、Detail コンポーネントを含むスクロール可能なコンテンツセクションを記述します。

Content コンポーネント内の List コンポーネントと Detail コンポーネントは、オブジェクト、項目、レコードの表示設定に従います。このため、これらのコンポーネントを使用すると、既存のデータセキュリティを変更することなくアプリケーションを開発できます。これらのコンポーネントではプロジェクトに必要な外観と操作性を実装するため、簡単に CSS スタイルを上書きできます。jQuery Mobile を使用したことがある方なら、Page、Header、Footer、Content の各コンポーネントが実際には jQuery Mobile プロパティを使ってモバイルユーザエクスペリエンスを実現していることがわかるでしょう。つまり、このコンポーネントライブラリを使用する場合は、jQuery Mobile のその他の機能も簡単に利用できるということです。

Visualforce List コンポーネント

List コンポーネントは、レコードの取得と表示に使用します。List コンポーネントの sObject 属性で、リストビューを作成する必要がある sObject のタイプを指定します。labelField、subLabelField、sortByField、listFilter などのその他の属性で、このリストの動作や表示のプロパティを指定します。

```
<c>List sobject="Contact" labelField="Name" subLabelField="Account.Name"
    sortByField="FirstName" listFilter="true"/>
```

Visualforce Detail コンポーネント

Detail コンポーネントは、属性として sObject 型を取り、関連レコードの詳細を含むモバイルレイアウトを生成します。レコードのモバイルレイアウトは、現在のユーザプロファイルと関連レコードタイプに関連付けられているページレイアウトによって決定されます。これにより、管理者は、標準の Salesforce ページレイアウトマネージャを使って、コードを変更することなくモバイルレイアウトを柔軟に更新できます。

```
<apex:define name="main">
    <c:Page name="main">
        <c:Header >
            <h1 style="font-size: 22px; margin: 0px;">Contact
        Details</h1>
        </c:Header>
        <c:Content >
            <c:Detail sobject="Contact"/>
        </c:Content>
    </c:Page>
</apex:define>
```

Visualforce Footer コンポーネント

Page コンポーネント内のヘッダーセクションに、data-role="footer" という属性を持つ jQuery Mobile ラッパーを提供します。Header コンポーネントと Footer コンポーネントは固定されており、その間にスクロール可能なコンテンツを挿入できます。

コンポーネントのインストール

コンポーネントを使用するには、コンポーネントのパッケージをインストールして、リモートアクセスポイントを有効にする必要があります。

1. 次のコマンドを実行して、Git からソースコードをコピーします。

```
git clone https://github.com/forcedotcom/MobileComponents.git
```

次に、Force.com Workbench を使って、MobileComponents/Visualforce/src フォルダ内の Force.com メタデータをターゲット組織にデプロイします。次の手順を実行します。

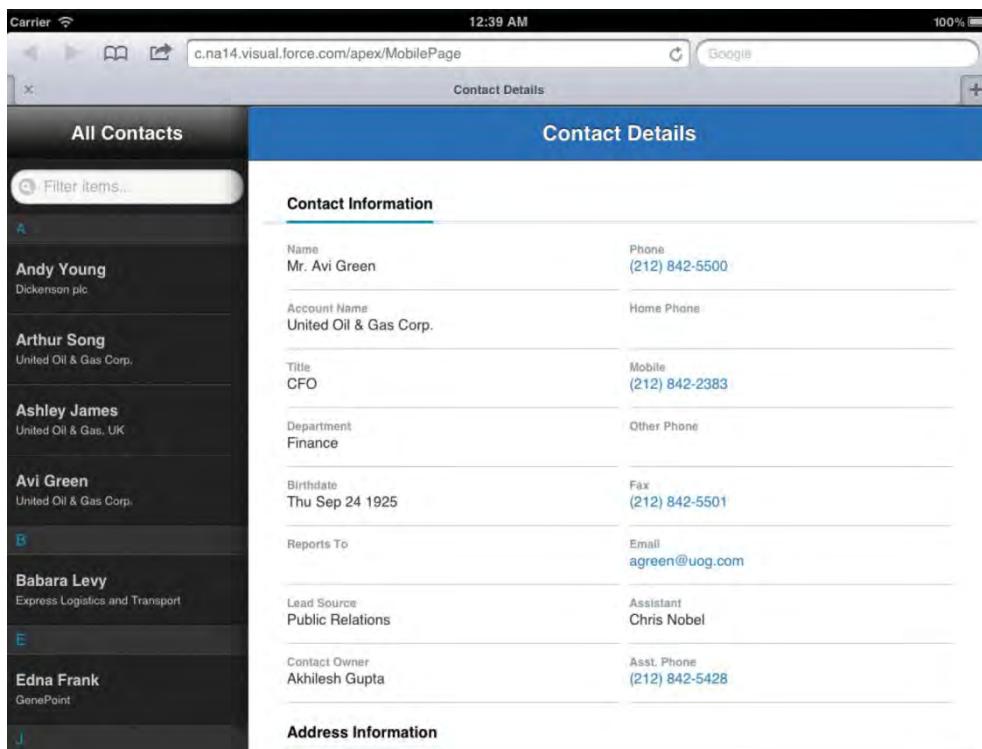
1. MobileComponents/Visualforce/src の zip アーカイブを作成します。
2. Workbench (<https://workbench.developerforce.com/>) にアクセスします。
3. Force.com のログイン情報を使ってログインし、Workbench からデータへのアクセスを許可します。
4. [migration] > [Deploy] の順にクリックします。
5. [Choose File] (ブラウザによっては [参照...]) をクリックして、先ほど作成した zip アーカイブファイルを選択します。
6. [Rollback on Error] チェックボックスをオンにします。
7. [Next] をクリックし、[Deploy] をクリックします。

最後に、Salesforce 組織内にリモートアクセスポイントを設定します。

1. Salesforce 組織にログインし、[設定] > [管理者設定] > [セキュリティのコントロール] > [リモートサイトの設定] の順にクリックします。
2. [新規リモートサイト] をクリックし、[リモートサイトの URL] に Salesforce インスタンスの URL を指定して、新しいサイトを作成します。たとえば、Salesforce インスタンスが NA1 の場合、リモートサイトの URL は <https://na1.salesforce.com> になります。

これで、Mobile Components for Visualforce ライブリヤリを使用する準備ができました。サンプルの Contact Viewer アプリケーションの動作は、次のページで確認できます。

<https://<remote site url>/apex/MobilePage>



タブレットページの作成

Mobile Components for Visualforce には、タブレットアプリケーションでのコンポーネントの使用方法を示す 2 つのサンプル Visualforce ページ (MobilePage と MobilePageWithComponents) が含まれています。

ここでは、MobilePage を取り上げ、タブレットに取引先責任者や詳細情報のリストを表示する仕組みを確認します。Mobile Components for Visualforce を使って作成されたすべてのモバイル Visualforce ページは、App コンポーネント内にラップする必要があります。App コンポーネントは、ビューポート設定、JavaScript、スタイルシートなど、モバイルアプリケーションの主要な構成要素を提供します。App コンポーネントの debug 属性では、開発モードと本番モードのどちらで実行するかを指定できます。後者の場合、簡易版のアセットを使用するかどうかも指定できます。

```
<c:App debug="true"></c:App>
```

App コンポーネント内に、モバイルアプリケーションの本文を指定します。ここでは、左側に細いリストビュー用セクション、右側にレコードの詳細を表示する幅広のセクションが配置された、分割ビューページを作成します。まず、このコンポーネントライブラリに含まれる SplitView テンプレートを使って、ページを構成します。

```
<apex:composition template="SplitViewTemplate"></apex:composition>
```

SplitView テンプレート (SplitViewTemplate) には、定義の必要な 2 つのセクションがあります。

- menu — ランドスケープモードの分割ビューの左側のセクションです。ユーザがタブレットを回転させてポートレートモードに切り替えると、このセクションはポップオーバー表示になります。
- main — 分割ビューの右側の幅広のセクションです。このセクションは、ポートレートモード、ランドスケープモードのどちらの場合にも表示されます。

次に、分割ビューを構成するこれらのセクションのコンテンツを定義します。以下は、左側のメニューセクションのコンテンツの定義です。

```
<apex:define name="menu">
    <c:Page name="list">
        <c:Header>
            <h1 style="font-size: 20px; margin: 0px;">All Contacts</h1>

        </c:Header>
        <c:Content>
            <c>List sobject="Contact" labelField="Name"
                subLabelField="Account.Name" sortByField="FirstName"
                listFilter="true" />
        </c:Content>
    </c:Page>
</apex:define>
```

左側のメニューセクションのコンテンツを定義するときは、Page コンポーネントを使用します。Page コンポーネントは、固定の Header コンポーネントと Footer コンポーネントの間にスクロール可能なコンテンツセクションを定義する、重要なラッパーになります。Page コンポーネント内で Header コンポーネントを使ってセクションのタイトルを定義します。次に、Content コンポーネントを使ってスクロール可能なコンテンツセクションを定義します。Content コンポーネントの本文では、List コンポーネントを使って取引先責任者リストを取得、表示します。List コンポーネントの sObject 属性を使って、リストビューを作成する sObject のタイプを指定します。さらに、labelField、subLabelField、sortByField、listFilter などのその他の属性を使って、このリストの動作や表示のプロパティを指定します(属性には、処理内容を想像しやすい属性名が付けられています)。

```
<c>List sobject="Contact" labelField="Name" subLabelField="Account.Name"
    sortByField="FirstName" listFilter="true" />
```

分割ビューのメインセクション内のコンテンツは次のとおりです。

```
<c:Page name="main">
    <c:Header>
        <h1 style="font-size: 22px; margin: 0px;">Contact Details</h1>
    </c:Header>
    <c:Content>
        <c:Detail sobject="Contact"/>
    </c:Content>
</c:Page>
</apex:define>
```

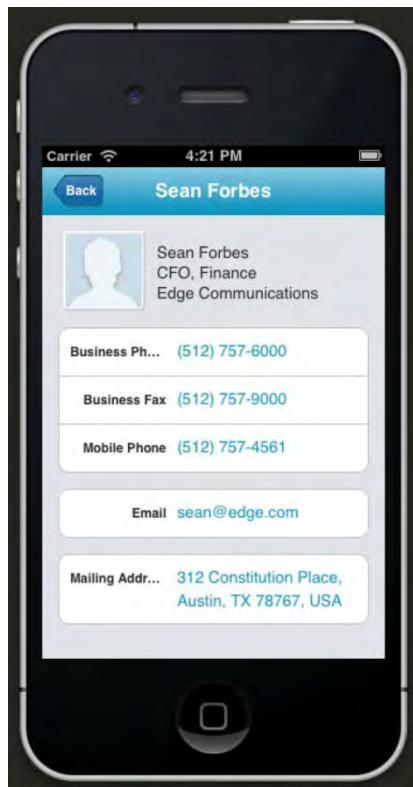
メインセクションでも、メニューセクション同様、Page コンポーネントを使ってコンテンツを定義し、Header、Footer、Content の各コンポーネントを有効にします。Header コンポーネントを使って、このセクションのタイトルを指定します。次に、Content コンポーネントを使って Detail コンポーネントを指定します。Detail コンポーネントは、属性として sObject 型を取り、関連レコードの詳細を含むモバイルレイアウトを生成します。現在のユーザプロファイルとレコードタイプに関連付けられたページレイアウトにより、レコードのモバイルレイアウトが決定されます。これにより、管理者は、標準の Salesforce ページレイアウトマネージャを使って、コードを変更することなくモバイルレイアウトを柔軟に更新できます。

```
<c:Detail sobject="Contact"/>
```

容易な統合

List コンポーネントと Detail コンポーネントは、Salesforce 組織のオブジェクト、項目、レコードの表示設定に従います。このため、これらのコンポーネントを使用すると、既存のセキュリティ設定を変更することなくアプリケーションを開発できます。これらのコンポーネントではプロジェクトに必要な外観と操作性を実装するため、簡単に CSS スタイルを上書きできます。

jQuery Mobile を使用したことがある方なら、Page、Header、Footer、Content の各コンポーネントが実際には jQuery Mobile プロパティを使ってモバイルユーザエクスペリエンスを実現していることがわかるでしょう。つまり、このコンポーネントライブラリを使用する場合は、jQuery Mobile のその他の機能も簡単に利用できるということです。



Visualforce モバイルコンポーネントの作成

どのスマートフォンやタブレットでも通常の Web ページを表示することはできますが、モバイル向けにユーザインターフェースを最適化したほうが快適なユーザエクスペリエンスを提供できます。Visualforce の開発者であれば、モバイルに最適化されたアプリケーションの開発はすでに半分終わっているようなものです。HTML5 が本格的に浸透しており、その機能を簡単に活用できる、モバイルに最適化された Web フレームワークも数多く公開されています。この章ですでに、Visualforce Mobile オープンコンポーネントで jQuery Mobile を活用し、Salesforce データを使用するモバイルアプリケーションを簡単に構築する方法を説明しました。これらのコンポーネントは開発者が自分で作成することもできます。

jQuery Mobile ページについて

jQuery Mobile ページについて簡単に説明しましょう。このフレームワークは主に `data-role` 属性を持つタグ付け要素を利用し、さまざまな値を指定することでブロック単位にページレイアウトを行います。ページの読み込み時に、jQuery Mobile フレームワークは、ドキュメント内でこれらの要素を検索、制御し、モバイルアプリケーションとして適した動作と書式をページに追加します。たとえば、`data-role` が `page` という値を持つ要素は、jQuery Mobile アプリケーションの最も基本的なビルディングブロック（モバイルに最適化された単一ページ）としてページを定義します。ページを解析するときに、jQuery Mobile では、これをモバイルアプリケーション内の単一のページとして扱い、画面にその内容を表示します。

```
<!--Ex. jQuery Mobile Page -->
<div data-role="page">
  <h1>My Page</h1>
</div><!--/page -->
```

data-role 属性は他にもあり、モバイルアプリケーション作成時に一緒に使用できます。ページセクション (`page`) 上部にヘッダーセクション (`header`)、中央にコンテンツセクション (`content`)、下部にフッターセクション (`footer`) を配置すると、モバイルに最適なレイアウトになります。これらの `data-role` 値を持つ要素が `page` 内にネストされている場合、フレームワークによって、どのデバイスでも同じように意図したレイアウトでページが表示されます。ネストされた要素だけでなく、アプリの各セクションを水平に組み合わせることもできます。`page` セクションに、**同階層の `page` セクション**がある場合、jQuery Mobile では、アプリの読み込み時に最初のページセクションのみを表示し、動作の機敏さと応答性を維持するために DOM から他のページセクションを削除します。ただし、これらのページはキャッシュされており、ページ ID を使用してハッシュリンクすることで、ページの読み込みと遷移アニメーションによって完全に表示できます。

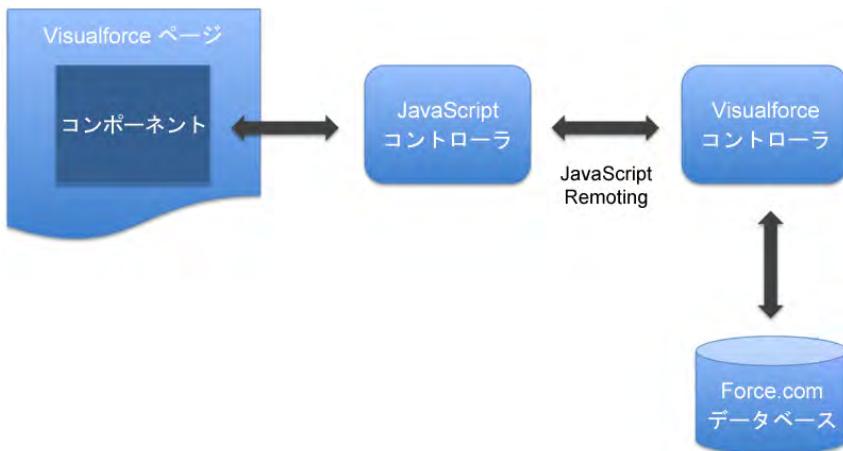
```
<!--Ex. jQuery Mobile Page 2 -->
<div data-role="page2">
  <div data-role="header">
    <h1>My Title</h1>
  </div><!--/header -->

  <div data-role="content">
    <p>Hello world</p>
  </div><!--/content -->
</div><!--/page -->
```

Mobile Components for Visualforce について

jQuery Mobile には、モバイルデバイスでのデータの表示と入力に適した、リストビュー、ナビゲーション、および各種入力フォームが用意されています。これらと Salesforce 組織などのデータソースを組み合わせることで、有用なデータを使用したモバイルアプリケーションを簡単に作成できます。このことを念頭に置いて、Mobile Components for Visualforce ライブラリの説明を進めます。オープンソースライブラリである Mobile Components for Visualforce では使いやすいコンポーネントが提供されており、Visualforce 開発者は jQuery Mobile のようなモバイルアプリケーションフレームワークを容易に扱うことができます。このライブラリは比較的新しく、このドキュメントの執筆時点では、一部のタイプのモバイルビューとウィジェットはサポートされませんが、そのような不足部分を補った新しいコンポーネントを簡単に作成できる強力なコア機能を搭載しています。

次の図に、Mobile Components for Visualforce のアーキテクチャを示します。



Visualforce ページは、Mobile Components for Visualforce が提供するモバイルコンポーネントなど、さまざまなコンポーネントで構成されています。モバイルコンポーネントと Force.com データベースとのやり取りには、JavaScript コントローラ、JavaScript Remoting ブリッジ、Visualforce コントローラが使用されます。

カスタム Visualforce モバイルコンポーネントの作成

Mobile Components for Visualforce のアーキテクチャを理解したら、実際に新しいコンポーネントを作成し、その処理を見てみましょう。オープンソース化を行った最初の段階では、Mobile Components for Visualforce プロジェクトは、リスト、ナビゲーション、詳細コンポーネントなどのわずかなコンポーネントでしか構成されていません。しかし、jQuery Mobile フレームワークが他の多くのコンポーネントをサポートしており、このプロジェクトにはまだ拡張する余地があります。

次の左側の画像は、すでに Mobile Components for Visualforce に含まれているリストコンポーネントの例です。ご覧のように、プロフィール写真を表示した右側のユーザリストに比べて、見栄えはあまりよくありません。



jQuery Mobile が提供する検索性にすぐれたこのタイプのリストは、サムネイルリストと呼ばれます。このセクションでは、このようなコンポーネントを作成するコードについて、大まかなレベルでは各コードスニペットの役割を解説し、詳細なレベルでは具体的な実装例を紹介して説明していきます。また、このようなアプローチに従い、他のタイプのコンポーネントを作成する読者も概要およびリファレンスとして利用できる情報を提供します。

作業環境の準備

まず、次の手順に従って作業環境を準備します。

1. 無料の **Developer Edition (DE)** 組織を新規作成します。
2. サンプル管理パッケージをインストールして、新しい **ThumbnailList** コンポーネントに関係するものをすべて構築します。

このパッケージには以下が含まれます。

- ページなど、提供コンポーネントをいくつか修正したバージョンが含まれている、カスタマイズされた Mobile Components for Visualforce フレームワーク
- **ThumbnailList** という名前のデモページを含む **ThumbnailList** コンポーネント

カスタム Visualforce モバイルコンポーネントの解説

次のコードは、カスタマイズされた **ThumbnailList** コンポーネントのコードです。サンプルパッケージを DE 組織にインストールしている場合は、[設定] > [Develop] > [コンポーネント] > [ThumbnailList] の順にクリックして、このコードを表示できます。

```
<apex:component controller="ThumbnailListController">
    <!--Content -->
    <apex:attribute name="sObjectType" type="String" required="true"
        assignTo="={!config.sObjectType}" description="" />
    <apex:attribute name="filter" type="String" required="false"
        assignTo="={!config.filter}" description="owner|recent|follower" />
```

```

<apex:attribute name="filterClause" type="String" required="false"
assignTo="={!config.filterClause}"
description="SOQL WHERE clause to use to filter list records."/>
<!--UI -->
<apex:attribute name="imageUrlField" type="String" required="true"
assignTo="={!config.imageUrlField}" description="" />
<apex:attribute name="labelField" type="String" required="true"
assignTo="={!config.labelField}" description="" />
<apex:attribute name="subLabelField" type="String" required="true"
assignTo="={!config.subLabelField}" description="" />
<apex:attribute name="listItemStyleClass" type="String"
assignTo=" {!config.listItemStyleClass}" description="" />
<apex:attribute name="sortByField" type="String" required="true"
assignTo=" {!config.sortByField}" description="" />
<apex:attribute name="listDividerStyleClass" type="String"
assignTo=" {!config.listDividerStyleClass}" description="" />
<apex:attribute name="listDividerStyleClass" type="String"
assignTo=" {!config.listDividerStyleClass}" description="" />
<apex:attribute name="listFilter" type="Boolean" default="false"
description="" />
<!--Behaviour -->
<apex:attribute name="nextPage" type="String"
assignTo=" {!config.nextPage}" description="" />
<apex:attribute name="jsCtrlrName" type="String"
assignTo=" {!config.jsCtrlrName}" default="$V.ThumbnailListController"
description="Custom JavaScript handler to manage client-side lifecycle
and behavior." />
<apex:attribute name="debug" type="Boolean" assignTo=" {!config.debug}"
default="false" description="" />

<!--VF/HTML -->
<apex:includeScript value=" {!URLFOR($Resource[ThumbnailListJS])}" />

<apex:includeScript value=" {!URLFOR($Resource.ICanHaz)}" />

<apex:outputPanel layout="inline" id="list">
    <ul data-role="listview" data-filter=" {!listFilter}"></ul>
</apex:outputPanel>

<script>$V.App.registerComponent(' {!$Component.list}', 
{!configAsJson});
</script></apex:component>

```

- 1 行目: コンポーネントでカスタム Visualforce コントローラ ThumbnailListController を使用しています。このコントローラについては、このすぐ後で説明します。
- 3 ~ 20 行目: このブロックは、コンポーネントが受け取るランタイムパラメータを表します。Visualforce ページがこのコンポーネントを利用する際、コンポーネントは関連付けられている属性値をそのコントローラに渡し、コントローラはその値を、参照先の JavaScript コントローラに渡します。各属性の assignTo パラメータは、Visualforce コントローラで定義されている config オブジェクトを参照していることに注意してください。この JavaScript と Visualforce のコントローラについては、後のセクションで説明します。
- 23 行目: この行には、静的リソースとしてデータベースに格納されているカスタム JavaScript コントローラが含まれています。このコントローラについては、このすぐ後で説明します。

- 24 行目: 次のセクションで説明する Visualforce ページの例には、ロジックレステンプレートツールとして知られている、いわゆる **mustache** タグを使用するリスト項目テンプレートが含まれています。したがって、そのタグを解析して、テンプレートにデータを記述することができる JavaScript ライブリヤリが必要になります。このコンポーネントで使用している、mustache をサポートする **ICanHaz** というテンプレートツールは、オープンソースコードとして自由に利用することができます。この行には、静的リソースとしてデータベースにも格納されている **ICanHaz** が含まれています。
- 26 行目～28 行目: 生成するリスト項目が追加される標準の **outputPanel**。
- 30 行目: ページ要求によってレンダリングが開始されるようにするには、すべてのコンポーネントをフレームワークに登録する必要があります。この行でそれを実行しています。今後の説明のために、次のことに注意してください。JavaScript コントローラは、Visualforce コントローラに格納されている **config** オブジェクトをここで受け取ります。この config オブジェクトは、Visualforce コントローラと JavaScript コントローラの両方で使用され、両コントローラ間で、Visualforce コントローラの名前やリストのルート要素、モバイルアプリケーションがデバッグモードになっているか、デバッグメッセージを許可するかどうかなどの重要な情報をやり取りします。この関数のパラメータは JSON 文字列の値をとるため、Visualforce コントローラには、**config** オブジェクトを JSON にシリアル化する **getConfigAsJson** というメソッドがあります。

サンプルの Visualforce ページの解説

ここで、新しい **ThumbnailList** コンポーネントを使用するサンプル Visualforce ページを詳しく見てみましょう。サンプルパッケージを DE 組織にインストールしている場合は、[設定] > [開発] > [ページ] > [ThumbnailList] の順にクリックして、このコードを表示できます。

```
<!--ThumbnailList.page -->
<apex:page showHeader="false" standardStylesheets="false" cache="false"
doctype="html-5.0">
<!--Templates -->
<script id="rowItemTempl" type="text/html">
  {{#records}}
    <li data-corners="false"
        data-shadow="false"
        data-iconshadow="true"
        data-wrapperels="div"
        data-icon="arrow-r"
        data-iconpos="right"
        data-theme="c"
        data-item-context="{{Id}}"
        class="ui-btn ui-btn-icon-right ui-li-has-arrow ui-li
ui-li-has-thumb ui-btn-up-c">

      <div class="ui-btn-inner ui-li">
        <div class="ui-btn-text">
          <a href="/ThumbnailList#userDetail" class="ui-link-inherit">

            
            <h3 class="ui-li-heading">{{Name}}</h3>
            <p class="ui-li-desc">{{Phone}}</p>
          </a>
        </div>
      <span class="ui-icon ui-icon-arrow-r ui-icon-shadow"> </span>
    </div>
  {{/records}}
</script>
</apex:page>
```

```

</li>
{{/records}}
</script>
<!--Declare a new app, with one page. -->
<c:App debug="true">
<c:Page name="list"
    theme="touch"
    debug="true">
<c:Header >
    <h1 style="font-size: 20px; margin: 0px;">All Users</h1>
</c:Header>
<c:Content >
    <c:ThumbnailList sObjectType="User"
        imageUrlField="FullPhotoUrl"
        labelField="Name"
        subLabelField="Phone"
        sortByField="Name"
        listFilter="true"
        filter="recent"
        debug="true"/>
</c:Content>
</c:Page>
</c:App>
<style>
    [data-role="panel"] [data-id="main"] [data-role="page"].ui-page
    .ui-content {
        background: white;
    }
    .ui-body-touch, .ui-overlay-touch {
        font-family: Helvetica, Arial, sans-serif
    }
</style>
</apex:page>

```

まず、3 ~ 28 行目で <script id="rowItemTempl"> タグで定義されている HTML テンプレートは、jQuery Mobile サイトにあるサンプル **Thumbnail List** の HTML を拡張したものです。このトピックの冒頭で、特定のプロパティを持つ要素が jQuery Mobile でどのように読み込まれるかについて説明しました。これらの要素がページの DOM に追加されたときに jQuery Mobile によって適切な書式が設定されるようにするには、li タグでプロパティの配列を使用する必要があります。

また、rowItemTempl テンプレートでは、JSON データを HTML 要素に変換するプロセスを簡素化するために、フィールド値のプレースホルダとして **mustache** タグ ({{records}}、{{Id}}、{{FullPhotoURL}}、{{Phone}}) を使用しています。前述した ICanHaz でこのテンプレートを解析した後、JSON データをテンプレートに埋め込むことで、Visualforce コントローラへの JavaScript Remoting 要求の結果を受け取ります。これにより、取得したレコードごとに異なるデータを含み、jQuery Mobile によって解析および処理が可能な特定の属性を持つ li タグのリストを迅速かつ簡単に作成できます。これで、Salesforce データを取得し、ネイティブモバイルアプリケーションのように表示されるページをレンダリングする方法を最後まで確認しました。

ページの本文にあたる 38 行目から 45 行目を見てください。必要なのはカスタム **ThumbnailList** コンポーネント (前述) への参照です。この参照により、コンポーネントの各属性の値が提供されます。

JavaScript コントローラの解説

次は、JavaScript コントローラについて説明します。重要なのは、JavaScript コントローラの主な機能の一つが、コンポーネントの Visualforce コントローラへデータを渡すブリッジとしての役割であるということです。これについては次のセクションで説明します。サンプルパッケージを DE 組織にインストールしている場合は、[設定] > [開発] > [静的リソース] > [ThumbnailListJS] > [ファイルを表示] の順にクリックして、次のコードを表示できます。

```
(function($) {
    $V.ThumbnailListController = $V.Component.extend({
        init: function(config) {
            this._super(config);
        },
        prepare: function() {
            this._super();
        },
        render: function() {
            this._super();
            // Load records from the server, and give it a function to handle
            // the response.
            $.mobile.showPageLoadingMsg();
            var serverRecords = this.requestRecords(this.requestRecordsHandler);

            $.mobile.hidePageLoadingMsg();
        },
        requestRecords: function(responseHandler) {
            // Specify any parameter values for the component's Visualforce
            controller.
            var configProxy = {
                sObjectType: this.config.sObjectType,
                imageUrlField: this.config.imageUrlField,
                labelField: this.config.labelField,
                subLabelField: this.config.subLabelField,
                sortByField: this.config.sortByField
            };
            $V.App.getFn(this.config.serverCtrlrName).getRecordsForConfig(
                this.config,
                // Callback
                (function(that, fn) {
                    return function(result, event) {
                        fn.apply(that, arguments);
                    };
                })(this, responseHandler)
            );
        },
        requestRecordsHandler: function(result, event) {
            // See what the response looks like.
            $V.App.log.debug(this.config.serverCtrlrName + '.getRecords response:',
                result);
            // Transform the response list of records to match the template.
            var model = {};
            model.records = [];
            for (var i = 0; i < result.length; i++) {
                var record = result[i];
                model.records.push(record);
            }
            $V.App.log.debug('Finished model: ', model);
            // Push the data into the template. This templating engine is
        }
    });
});
```

```

called
    // ICanHaz, which is a wrapper around Mustache templating.
    // This command looks for a template stored as HTML inside
    // <script id="rowItemTempl" type="text/html"><script> tags.
    var listItemsMarkup = ich.rowItemTempl(model, true);
    $V.App.log.debug('Template result: ', listItemsMarkup);
    // Render the markup to the DOM.
    $ul = this.$me.find('ul');
    $ul.html(listItemsMarkup);
    // After mark up is rendered, do other stuff, like create handlers
    for tap events.
    //this.applyEvents($ul);
}
});
})(jQuery);

```

2行目は、フレームワークの \$V.Component クラスを継承する際に使用する構文です。関連するマニュアルでも説明はされていますが、この JavaScript コントローラを実装する方法については、詳しく解説します。このフレームワーククラスを拡張する場合、init、prepare、および render メソッドを実装するための暗黙のコントラクトを実装し、すべての実装ロジックの前に this._super() メソッドを呼び出す必要があります。これは、3 ~ 25 行目に記述されています。

これらのメソッドの役割を詳しく見ていきましょう。これらのメソッドはなぜ必要であり、どのように実装すべきかを考えます。これらのメソッドはフレームワークから呼び出され、特定のアクションを実行することを要求されます。フレームワークは、オブジェクトを初めて作成するときにコンポーネントの init メソッドをコールし、コンポーネントをレンダリングする直前に prepare メソッドをコールし、レンダリングするコンポーネントをユーザが要求したときに render メソッドをコールします。このサンプルで行うような単純な実装の場合は、このオブジェクトを設定するために特に何かする必要はないため、init メソッドは空のままにしておきます。同様の理由で、prepare メソッドも空のままにしておきます。ここでは、HTML テンプレートを使用してページにリスト項目を生成する render メソッド(前のセクションを参照)に焦点を当てます。この render メソッドは次のような機能があります。

- Visualforce コントローラで JavaScript Remoting メソッド getRecordsForConfig をコールして、サーバのレコードを取得する。このメソッドについては次のセクションで説明します。
- requestRecordsHandler メソッドを使用してサーバの応答を処理する。このメソッドは、rowItemTempl テンプレートを使用してレコードデータを HTML 文字列に変換し、DOM 内の ul 要素に追加します。

Visualforce コントローラの解説

最後に、新しいThumbnailList コンポーネントをサポートする Visualforce コントローラがどのように動作しているか見てみましょう。サンプルパッケージを DE 組織にインストールしている場合は、[設定] > [開発] > [Apex クラス] > [ThumbnailListController] の順にクリックして、このコードを表示できます。

```

public class ThumbnailListController {
    public ThumbnailListConfig config {get; set;}

```

```

// Parameter object to pass between JavaScript and Visualforce
controllers.
public virtual class ThumbnailListConfig {
    public Boolean debug { get; set; }
    // The elemId is generated in VF and sent to client.
    public String elemId {get; set;}
    public String sObjectType {get; set;}
    public String imageUrlField {get; set;}
    public String labelField {get; set;}
    public String subLabelField {get; set;}
    public String sortByField {get; set;}
    public String filter { get; set; }
    public String filterClause {get; set;}
    public String listItemStyleClass { get; set; }
    public String listDividerStyleClass { get; set; }
    public String nextPage { get; set; }
    public String serverCtrlrName = 'ThumbnailListController';
    public String jsCtrlrName {get; set;}
}

// constructor
public ThumbnailListController() {
    this.config = new ThumbnailListConfig();
}

private final static String THUMBNAIL_LIST_JS = 'ThumbnailListJS';

public String getThumbnailListJS() {
    return config.debug ? THUMBNAIL_LIST_JS : (THUMBNAIL_LIST_JS
+ 'Min');
}

@RemoteAction
public static List<Sobject> getRecordsForConfig(ThumbnailListConfig
config) {
    System.debug('---config: ' + config);
    Set<String> fieldsToQuerySet = new
    Set<String>();
    fieldsToQuerySet.add(config.imageUrlField);
    fieldsToQuerySet.add(config.labelField);
    fieldsToQuerySet.add(config.subLabelField);
    fieldsToQuerySet.add(config.sortByField);

    List<Sobject> recordList = ThumbnailListController.getRecords(
        config.sObjectType,
        fieldsToQuerySet,
        UserInfo.getUserId()
    );

    return recordList;
}
public static List<Sobject> getRecords(
    String sObjectType,
    Set<String> fieldsToQuerySet,
    Id userId) {

    List<Sobject> recordList = new List<Sobject>();
    String queryString = ' ';

```

```

// Build the Select clause.
queryString += 'SELECT ';
for (String field : fieldsToQuerySet)
    queryString += (field + ',');
queryString = queryString.subString(0, queryString.length()-1);

// Build the From clause.
queryString += ' FROM ' + sObjectType;

// Build the Where clause.
queryString += ' WHERE ';
queryString += ' LastModifiedById = \'' + userId + '\'';

recordList = Database.query(queryString);
return recordList;
}

public virtual String getConfigAsJson() {
    String configStr = JSON.serialize(this.config);
    System.debug(configStr);
    return configStr;
}
}

```

- 3 ~ 27 行目: これらのメソッドで config オブジェクトを設定します。前述した JavaScript コントローラと Visualforce コントローラのどちらも、config オブジェクトを使用して必要な情報をやり取りします。
- 76 ~ 80 行目: config オブジェクトを JSON としてシリアル化するメソッドです。このメソッドは、コンポーネントの 30 行目にある \$V.App.registerComponent メソッドに渡されます。
- 29 ~ 32 行目: ThumbnailList コンポーネント内のスクリプトインクルードタグ (23 行目) は、この getThumbnailListJS メソッドを使用して JavaScript コントローラのファイル名を取得します。メソッドでこのファイルの名前を提供できるようにすることで、ソースコードのフルバージョンまたは簡易バージョンのいずれかを返すロジックを挿入できます。簡易バージョンはファイルサイズが小さく、スクリプトの読み込み時間を短縮できるため、このコードを本番環境に展開する場合に最適です。
- 34 ~ 50 行目: getRecordsForConfig メソッドは、JavaScript コントローラと Visualforce コントローラとを結びつける JavaScript Remoting のメソッドです。getRecordsForConfig メソッドは、JavaScript コントローラの requestRecords 関数によって呼び出されると、いくつかの設定を行い、Visualforce コントローラで getRecords メソッドをコールして、要求されたレコードを取得して返します。
- 51 ~ 74 行目: Visualforce コントローラの getRecords メソッドが、要求されたレコードを取得して返します。

まとめ

学習の成果をご確認ください。ここで説明したコンポーネントが使用されているサンプルページを開くことができます。DE 組織で <pod>.salesforce.com/apex/ThumbnailList に移動すると、作成したコンポーネントが使用されているデモページが表示されます。これらのコンポーネントにより、幅広い層の開発者によるモバイルアプリケーションの開発が可能になると共に、Visualforce を利用する多くの開発者はモバイルアプリケーションを簡単に作成できるようになります。

第 8 章

HTML5 アプリケーションの開発

トピック:

- HTML5 の開発要件
- JavaScript を使用したデータアクセス

HTML5 を利用すると、モバイルデバイスにソフトウェアをインストールしなくても軽量なインターフェースを作成できます。また、モバイルデバイス、タッチデバイス、デスクトップなど、あらゆるデバイスから同じインターフェースにアクセス可能です。Visualforce を通じて HTML コンテンツを提供し、JavaScript から Apex コントローラを呼び出すことによって Force.com のレコードデータを取得する HTML5 アプリケーションを作成することができます。サンプルアプリケーションでは、ユーザインターフェースに jQuery Mobile ライブラリを使用します。

HTML5 の開発要件

- リモートアクセスアプリケーションを作成しておく必要があります。作成方法の詳細については、「リモートアクセスアプリケーションの作成」を参照してください。
- Apex と Visualforce の知識が必要になります。
- Force.com 組織が必要です。



メモ: この開発シナリオでは Visualforce を使用するため、Database.com は利用できません。

JavaScript を使用したデータアクセス

HTML5 アプリケーションでは、JQuery ライブラリ用と JavaScript のリモーティング機能 (JavaScript Remoting) 用の 2 つの静的リソースファイルが必要となります。

- jQuery 静的リソースには、jQuery ライブラリと jQuery Mobile ライブラリで使用されるすべての JavaScript ファイルとスタイルシートファイルが含まれています。
- JavaScript のリモーティング機能を使用して Apex コントローラからデータを取得するメソッドが含まれた JavaScript ファイルも必要になります。このデータは適切な HTML 要素にラップされ、Visualforce ページに表示されます。

次のような JavaScript ファイルを使用します。

```
function getAlbums(callback) {
    $j('#albumlist').empty();
    CloudtunesController.queryAlbums(function(records, e)
        { showAlbums(records, callback) }, {escape:true});
}
```

- getAlbums() 内の \$j('#albumlist').empty() などの関数コールを見ると、jQuery がどのように使用されているかがわかります。この例では、jQuery は albumlist という ID を持つ HTML 要素を取得し、結果を表示するため HTML を空にします。
- 次に、getAlbums() は Apex コントローラの queryAlbums() メソッドをコールします。ここで JavaScript リモーティング機能の働きに注目してください。JavaScript から直接コントローラメソッドを呼び出すために必要な設定は、すべて Visualforce により提供されます。
- queryAlbums() の引数として渡されたコールバック関数は、Apex コントローラからレコードが返されると自動的に呼び出されます。取得されたレコードは showAlbums() 関数により表示されます。

次は、この showAlbums() 関数について説明します。

```
function showAlbums(records, callback) {
    currentAlbums.length = 0;
    for(var i = 0; i < records.length; i++) {
        currentAlbums[records[i].Id] = records[i];
    }

    $j.each(records, function() {
        $j('<li></li>')
            .attr('id', this.Id)
            .hide()
            .append('<h2>' + this.Name + '</h2>')
            .click(function(e) {
                e.preventDefault();
                $j.mobile.showPageLoadingMsg();
                $j('#AlbumName').html(currentAlbums[this.id].Name);
                $j('#AlbumPrice').html('$'+currentAlbums[this.id].Price__c);

                if($j('#AlbumPrice').html().indexOf(".") > 0
                    && $j('#AlbumPrice').html().split(".")[1].length == 1)
                {
                    $j('#AlbumPrice').html($j('#AlbumPrice').html()+"0"); //add
                    trailing zero
                }
                $j('#AlbumId').val(currentAlbums[this.id].Id);
                var onTracksLoaded = function() {
                    $j.mobile.hidePageLoadingMsg();
                    $j.mobile.changePage('#detailpage', {changeHash: true});
                }
                getTracks(currentAlbums[this.id].Id, onTracksLoaded);
            })
            .appendTo('#albumlist')
            .show();
    });

    $j('#albumlist').listview('refresh');
    if(callback != null && typeof callback == 'function') { callback(); }
}
}
```

- この関数はコールバックからレコードを取得し、各レコードをループ処理して、 albumlist セクション内に表示される 要素のリストを新規作成します。
- この関数は、各リストアイテムに新しいイベントを動的に付加し、ユーザがリストアイテムをクリックしたとき、そのアルバムに関連付けられたトラックのリストが表示されるようにします。トラックリストは、getTracks() を使って取得されます。

最後に、この getTracks() について説明します。このコードは、getAlbums() や showAlbums() のコードと機能的に非常によく似ています。アルバムを処理するコードとの大きな違いは、使用する Apex コントローラメソッドと、取得した結果をもとにページを更新するためのコールバック関数です。

```
function getTracks(albumid, callback) {  
    $('#tracklist').empty();  
    CloudtunesController.queryTracks(albumid, function(records, e) {  
        showTracks(records,callback) }, {escape:true} );  
    return true;  
}
```

アルバム名をクリックすると、新しいトラックデータのセットが取得され、アイテムリストが書き換えられます。トラック名をクリックすると、トラック情報を表示する要素のHTMLが書き換えられ、jQuery Mobileにより、そのトラックのページに移動します。実際のアプリケーションでは、もちろんこの情報もキャッシュすることができます。

第 9 章

オフラインでのデータの安全な格納

トピック:

- ハイブリッドアプリケーションでの SmartStore へのアクセス
- ハイブリッドアプリケーションのオフライン開発
- MockSmartStore の使用
- Soup の登録
- Soup からのデータの取得
- カーソルの操作
- データの操作
- SmartStore の拡張

モバイルデバイスは、いつでも接続が切断される可能性があります。また、病院や飛行機内では接続を禁止される場合もあります。こうした状況に備えて、オフラインでもアプリケーションが機能するようにしておくことが重要です。

Mobile SDK では、安全なオフラインストレージソリューションである SmartStore をデバイスで使用します。SmartStore を使用すると、デバイスがインターネットに接続していないときでも作業を続けることができます。SmartStore は、Soup と呼ばれる形式でデータを JSON ドキュメントとして格納します。Soup は、データベースのエントリをさまざまな方法でインデックス化し、各種メソッドでクエリできるようにする、シンプルな 1 つのテーブルで構成されるデータベースです。



メモ: HTML5 のみで構築されたアプリケーションは、オフライン情報をブラウザのキャッシュに格納します。ブラウザのキャッシュ機能は Mobile SDK には含まれていないため、ここでは説明しません。SmartStore はデバイスのストレージ機能を使用します。このアプローチには、ネイティブまたはハイブリッドの開発手法が要求されます。

サンプルオブジェクト

この章のコードスニペットで使用されている 2 つのオブジェクト (取引先、商談) は、Salesforce 組織が提供するすべての組織であらかじめ定義されているオブジェクトです。取引先オブジェクトと商談オブジェクトは主従関係にあり、取引先には複数の商談があります。

ハイブリッドアプリケーションでの SmartStore へのアクセス

ハイブリッドコンテナは、JavaScript を使用して、カメラ、アドレス帳、ファイルストレージなどのネイティブのデバイス機能にアクセスします。SmartStore もまた、JavaScript からアクセスします。ハイブリッドモバイルアプリケーションのオフラインアクセスを有効にするには、次のような JavaScript ファイルと CSS ファイルを Visualforce (HTML) ページ内に追加する必要があります。

- cordova-1.8.1.js — Cordova ライブラリ (旧名 PhoneGap)。
- SFHybridApp.js — オフラインかどうかを判断するなど、ユーティリティタスクを実行するメソッドを含む。
- SFSmartStorePlugin.js – SDK のオフライン機能のコア実装。

SmartStore では、1つまたは複数の Soup にオフラインデータを格納します。Soup は、概念的に言えば、JSON オブジェクトとして表される、オフラインで保存やクエリを行うデータレコードの論理的な集合です。Force.com では、Soup は通常オフラインで保存する標準またはカスタムのオブジェクトに対応付けますが、これは厳格なルールではありません。アプリケーションで必要な数だけ Soup を保存できますが、Soup は自己完結しているデータセットであり、Soup どうしには直接の相関関係はありません。データそのものを格納するだけでなく、データのクエリの使いやすさとカスタマイズ性を向上するために、データ内の項目に対応付けるインデックスを指定することもできます。



メモ: SmartStore のデータは本質的に、認証されたユーザに関連付けられています。アプリケーションからユーザがログアウトすると、SmartStore は、そのユーザに関連付けられている Soup データを削除します。

ハイブリッドアプリケーションのオフライン開発

コンテナ内でハイブリッドアプリケーションの開発を行うと、変更のたびにビルドと展開のステップが必要になります。そのため、ブラウザで直接ハイブリッドアプリケーションを開発し、コンテナ内のコードはテストの最終段階でのみ実行することをお勧めします。JavaScript の開発は、ブラウザで行うとコンパイルとビルドのステップを踏まずに済むため、より簡単です。コードを変更したら、ブラウザを更新するだけで変更の反映を確認できます。

Google Chrome には、Web アプリケーションの内部へアクセスできる開発者ツールがバンドルされているため、Chrome ブラウザの使用をお勧めします。詳細については、「Chrome Developer Tools: Overview」を参照してください。

MockSmartStore の使用

コンテナの外で開発を進める場合、SmartStore を使用するコードの開発とテストを容易にするために、SmartStore をエミュレートすることができます。MockSmartStore は、ローカルストレージ (または必要に応じて、メモリ内) にデータを格納する、SmartStore の JavaScript 実装です。



メモ: MockSmartStore はデータを暗号化しないため、本番環境のアプリケーションで使用するものではありません。

PhoneGap ディレクトリ内には、次のファイルを含むローカルディレクトリがあります。

- MockSmartStore.js — SmartStore の JavaScript 実装。コンテナの外で開発とテストを行う場合にのみ使用します。
- MockSmartStorePlugin.js — JavaScript ヘルパークラス。SmartStore Cordova プラグインのコールをインターフェーストし、MockSmartStore を使用して処理します。
- CordovaInterceptor.js — JavaScript ヘルパークラス。Cordova プラグインのコールをインターフェーストします。

アプリケーションの index.html 内で、cordova-1.8.1.js と SFSmartStorePlugin.js の JavaScript インクルードの後に次のコードを追加し、MockSmartStore を使用できるようにします。

```
$(function() {
    //Add event listener
    SFHybridApp.logToConsole("onLoad: jquery ready");
    document.addEventListener("deviceready", onDeviceReady, false);

    // Initialize mock smartstore
    // if true, use local storage, false, in memory only
    MockSmartStore.init(true);
    // Without a container, we have to do some initialization
    cordova.completeInitialization();
});

// When this function is called, Cordova has been initialized
function onDeviceReady() {
    // that's where your application really starts
}
```

MockSmartStore の動作を確認するには、ブラウザで Cordova/local/test.html にアクセスします。

同一生成源 (Same-Origin) ポリシー

同一生成源ポリシーとは、同一のサイトを生成源とするそれぞれのページにスクリプトの実行を許可し、特定の制約を設けずに互いのメソッドやプロパティにアクセスできるようするポリシーです。これにより、異なるサイトのページにまたがるほとんどのメソッドやプロパティへのアクセスはブロックされることになります。コンテナ内でコードを実行する場合には、同一生成源ポリシーの制約は問題にはなりません。コンテナによって WebView では同一生成源ポリシーは無効にされるからです。ただし、リモート API を呼び出す場合は、同一生成源ポリシーの制約を考慮する必要があります。

ブラウザには同一生成源ポリシーを無効にするオプションがあり、ご使用のブラウザでのポリシーの適用方法を調べることができます。ローカルファイルシステムから読み込まれる JavaScript ファイルから Force.com に対して XHR コールを実行する場合は、同一生成源ポリシーを無効にしてからブラウザを起動する必要があります。一般的に使用されるブラウザで同一生成源ポリシーを無効にする方法については、「Getting Around Same-Origin Policy in Web Browsers」の記事を参照してください。

認証

MockSmartStore で認証を行うには、実際のセッションでアクセストークンと更新トークンを取得し、JavaScript アプリケーションでそれらをコードに渡す必要があります。ForceTk JavaScript ツールキットを初期化する場合にもこれらのトークンは必要になります。

Soup の登録

Soup にアクセスするには、まず Soup を登録する必要があります。次のように、Soup の名前、インデックスの指定、および成功の場合とエラーの場合のコールバック関数の名前を指定します。

```
navigator.smartstore.registerSoup(soupName, indexSpecs, successCallback,  
errorCallback)
```

既存の Soup がない場合、この関数で作成されます。既存の Soup がある場合は、登録することでその Soup へアクセスできます。Soup がすでに存在するかどうかを調べるには、次の関数を使用します。

```
navigator.smartstore.soupExists(soupName, successCallback,  
errorCallback);
```

Soup は、そのエントリ内にある 1 つ以上の項目にインデックス化されます。Soup エントリの挿入、更新、および削除操作は、Soup のインデックスで追跡されます。Soup を登録する際には、必ずインデックス項目を 1 つ以上指定します。たとえば、単純なキーバリューストアとして Soup を使用している場合、文字列型のインデックスの指定を 1 つ使用します。

indexSpecs

indexSpecs 配列は、あらかじめ定義されたインデックスを持つ Soup の作成に使用します。indexSpecs 配列のエントリで、Soup をインデックス化する方法を指定します。各エントリは、path と type のペアで構成されます。path にはインデックス項目の名前、type には「string」または「integer」を指定します。インデックスのパスは大文字小文字を区別し、複合パス (Owner.Name など) を含めることもできます。



メモ: インデックスのパスが深すぎると処理に時間がかかることがあります。インデックスのエントリで、特定の indexSpec に記載されたいずれかの項目が欠落している場合、そのインデックスでは追跡されません。

```
"indexSpecs": [
  {
    "path": "Name",
    "type": "string"
  },
  {
    "path": "Id",
    "type": "string"
  },
  {
    "path": "ParentId",
    "type": "string"
  },
  {
    "path": "lastModifiedDate",
    "type": "integer"
  }
]
```



メモ: 現在のところ、Mobile SDK でサポートされるのは、「string」と「integer」の 2 種類のインデックスのみです。これらのデータ型は、インデックス自体にのみ適用され、データの格納や取得の方法には適用されません。インデックス列に null 項目を使用することもできます。

successCallback

registerSoup の成功コールバック関数の引数は 1 つだけです (Soup の名前)。

```
function(soupName) { alert("Soup " + soupName + " was successfully created"); }
```

Soup の作成に成功すると、Soup の準備が整ったことを示す successCallback が返されます。トランザクションが完了してコールバックが返されてから、次の操作を開始します。渡された名前で Soup を登録した場合、成功コールバック関数はその Soup を返します。

errorCallback

registerSoup のエラーコールバック関数の引数は 1 つだけです (エラーを説明する文字列)。

```
function(err) { alert ("registerSoup failed with error:" + err); }
```

Soup の作成中に、次のようなさまざまな理由でエラーが発生することがあります。

- Soup の名前が無効、または正しくない
- インデックスがない (インデックスは少なくとも 1 つ必要)
- その他の予期しないエラー (データベースのエラーなど)

Soupからのデータの取得

SmartStore は、クエリ文字列を構築するヘルパー・メソッドのセットを提供します。特定のレコードセットのクエリを行うには、クエリの仕様に合った `build*` メソッドをコールします。次の表に示すように、必要に応じて、インデックス項目、ソート順、および絞り込みに使用するその他のメタデータを定義することができます。

パラメータ	説明
<code>indexPath</code>	名前、取引先番号、または日付などの検索対象です。
<code>beginKey</code>	省略可能。クエリ範囲の開始を指定するために使用します。
<code>endKey</code>	省略可能。クエリ範囲の終了を指定するために使用します。
<code>order</code>	省略可能。「 <code>ascending</code> 」(昇順) または「 <code>descending</code> 」(降順) のいずれかです。
<code>pageSize</code>	省略可能。指定しない場合、ネイティブ・プラグインは、最終的な <code>Cursor.pageSize</code> に合わせたページサイズで返します。



メモ: クエリはすべて、1つの項目で検索を行います。複数の検索項目はサポートされません。

全件対象クエリ

`buildAllQuerySpec(indexPath, order, [pageSize])` は順不同で、Soup 内のすべてのエントリを返します。このクエリは Soup 内のすべてのエントリを対象とする場合に使用します。

`order` と `pageSize` は省略可能です。デフォルトで、`ascending` (昇順) と 10 がそれぞれ設定されています。次のようなパターンで指定できます。

- `buildAllQuerySpec(indexPath)`
- `buildAllQuerySpec(indexPath, order)`
- `buildAllQuerySpec(indexPath, order, [pageSize])`

ただし、`buildAllQuerySpec(indexPath,[pageSize])` という指定はできません。

ページサイズの詳細については「カーソルの操作」を参照してください。



メモ: 基本ルールとして、`pageSize` には画面上に表示するエントリの数を設定します。スクロール表示を滑らかにするには、実際に表示されるエントリの数の 2 倍から 3 倍の値にします。

完全一致クエリ

`buildExactQuerySpec(indexPath, matchKey, [pageSize])` は、`indexPath` 値に指定された `matchKey` に完全に一致するエントリを検索します。指定された ID の子エンティティを検索する場合に使用します。たとえば、商談を状態で検索することができます。ただし、検索結果の順序を指定することはできません。

ID で子を取得するサンプルコード:

```
var querySpec = navigator.smartstore.buildExactQuerySpec( "sfid" ,
  "some-sfdc-id" );
navigator.smartstore.querySoup( "Catalogs" , querySpec, function(cursor)
{
  // we expect the catalog to be in:
  cursor.currentPageOrderedEntries[0]
});
```

親 ID で子を取得するサンプルコード:

```
var querySpec = navigator.smartstore.buildExactQuerySpec( "parentSfid" ,
  "some-sfdc-id" );
navigator.smartstore.querySoup( "Catalogs" , querySpec, function(cursor)
{ });
```

範囲指定クエリ

`buildRangeQuerySpec(indexPath, beginKey, endKey, [order, pageSize])` は、`indexPath` 値が `beginKey` と `endKey` で指定された範囲内にあるエントリを検索します。整数値で保存された日付の範囲など、数字による範囲指定で検索する場合にこの関数を使用します。

`order` と `pageSize` は省略可能です。デフォルトで、`ascending` (昇順) と 10 がそれぞれ設定されています。次のようなパターンで指定できます。

- `buildRangeQuerySpec(indexPath, beginKey, endKey)`
- `buildRangeQuerySpec(indexPath, beginKey, endKey, order)`
- `buildRangeQuerySpec(indexPath, beginKey, endKey, order, pageSize)`

ただし、`buildRangeQuerySpec(indexPath, beginKey, endKey, pageSize)` という指定はできません。

次のように `beginKey` と `endKey` に `null` 値を渡すことで、範囲制限のない検索を実行できます。

- `endKey` に `null` 値を渡すと、`indexPath` が `beginKey` 以上である項目を持つレコードをすべて検索します。
- `beginKey` に `null` 値を渡すと、`indexPath` が `endKey` 以下である項目を持つレコードをすべて検索します。
- `beginKey` と `endKey` の両方に `null` 値を渡すと、全件検索と同じになります。

Like を使用したクエリ

`buildLikeQuerySpec(indexPath, likeKey, [order, pageSize])` は、指定した `likeKey` に類似する `indexPath` 値を持つエントリを検索します。検索キーワードが先頭にある語句を検索するときには "foo%" を使用し、検索キーワードが末尾にある語句を検索するときには "%foo" を使用し、`indexPath` 値の任意の位置でキーワードを検索する場合は "%foo%" を使用できます。全般的な検索や部分的に一致する名前を検索する場合にこの関数を使用します。`order` と `pageSize` は省略可能です。デフォルトで、`ascending` (昇順) と 10 がそれぞれ設定されています。



メモ: Like を使用したクエリは、時間が一番長くかかるクエリです。

クエリの実行

クエリは非同期で実行され、JavaScript コールバックにカーソルが返されます。成功コールバックは、`function(cursor)` の形式にする必要があります。`querySoup` メソッドにクエリ仕様を渡すには、`querySpec` パラメータを使用します。

```
navigator.smartstore.querySoup(soupName, querySpec, successCallback, errorCallback);
```

プライマリキーによる個々の Soup エントリの検索

すべての Soup エントリには、自動的に一意の内部 ID (Soup 内のすべてのエントリを保持する内部テーブルのプライマリキー) が割り当てられます。その ID 項目は、Soup エントリで `_soupEntryId` 項目として利用できます。Soup エントリは、`retrieveSoupEntries` メソッドを使用して `_soupEntryId` で検索できます。返される順序は保証されず、エントリが削除されている場合、結果の配列からは認識されなくなりますので注意が必要です。このメソッドは Soup エントリを取得するための最速の方法ですが、`_soupEntryId` を知っていなければ使用できません。

```
navigator.smartStore.retrieveSoupEntries(soupName, indexSpecs, successCallback, errorCallback)
```

カーソルの操作

クエリの結果セットが長くなると、データのサイズが大きすぎて読み込めなくなる可能性があります。この場合には、クエリ結果の小さなサブセット(単一ページ)のみを、任意の時点でネイティブレベルから JavaScript レベルにコピーします。クエリを実行すると、ネイティブレベルからカーソルオブジェクトが返され、クエリ結果のリスト全体をページングできるようになります。これにより、JavaScript コードでページを前後に移動し、JavaScript レベルにページをコピーできます。



メモ: (上級ユーザ向け) カーソルはデータのスナップショットではなく、動的です。Soup に変更を加えてから、カーソルを使用してページングを開始すると、それらの変更は反映されています。カーソルが保持している情報は、クエリ自体と結果セット内の現在の位置のみです。カーソルを移動すると、クエリが再実行されます。このようにして、新しく作成された Soup エントリが返されます(元のクエリを満たしている場合)。

クエリ結果内を移動するには、次のカーソル関数を使用します。

- `navigator.smartstore.moveCursorToPageIndex(cursor, newIndex, successCallback, errorCallback)` — 指定されたページのインデックスにカーソルを移動します。最初のページは 0、最後のページは `totalPages - 1` で指定します。
- `navigator.smartstore.moveCursorToNextPage(cursor, successCallback, errorCallback)` — 次のエントリページに移動します(ページが存在する場合)。

- `navigator.smartstore.moveCursorToPreviousPage(cursor, successCallback, errorCallback)` — 前のエントリページに移動します(ページが存在する場合)。
- `navigator.smartstore.closeCursor(cursor, successCallback, errorCallback)` — カーソルの操作が完了した際に、カーソルを閉じます。



メモ: これらの関数の `successCallback` には、1 個の引数(最新のカーソル)が必要です。

データの操作

Soup エントリの挿入、更新、削除を追跡するために、SmartStore では、各エントリに次の項目を追加します。

- `_soupEntryId` — 指定された Soup のテーブル内の Soup エントリを示すプライマリキー。
- `_soupLastModifiedDate` — 1970 年 1 月 1 日を起点とするミリ秒数。
 - ◊ JavaScript の日付に変換するには、`new Date(entry._soupLastModifiedDate)` を使用します。
 - ◊ 1970 年 1 月 1 日を起点とする、対応するミリ秒数に日付を変換するには、`date.getTime()` を使用します。

Soup エントリを挿入または更新すると、SmartStore では自動的にこれらの項目を設定します。特定のエントリを削除または取得する場合は、`_soupEntryId` を使用してエントリを参照できます。

Soup エントリの挿入または更新

指定した Soup エントリにすでに `_soupEntryId` スロットが設定されている場合は、そのスロットで識別されるエントリが更新されます。エントリに `_soupEntryId` スロットがない、またはスロットの値が Soup 内の既存のエントリに一致しない場合は、その Soup にエントリが追加(挿入)されて、`_soupEntryId` スロットが上書きされます。



メモ: `_soupEntryId` または `_soupLastModifiedDate` の値を手動で操作してはいけません。

エントリを挿入または更新するには、`upsertSoupEntries` メソッドを次のように使用します。

```
navigator.smartStore.upsertSoupEntries(soupName, entries[], successCallback, errorCallback)
```

`soupName` が対象の Soup の名前であり、`entries` は Soup のデータ構造と一致する 1 つまたは複数のエントリの配列です。`successCallback` と `errorCallback` のパラメータは、`registerSoup` の場合とほとんど同じように機能します。ただし、`upsertSoupEntries` の成功コールバックは、新しいレコードが挿入されたか、または既存のレコードが更新されたことを示します。

外部 ID による挿入/更新 (upsert)

Soup エントリを使用して外部システムのデータをミラーリングする場合、外部システムの ID (プライマリキー) でそれらのエンティティを参照することが必要になる場合があります。このため、外部 ID による更新/挿入 (upsert) がサポートされています。upsert を実行するときは、外部 ID 項目として任意のインデックス項目を指定できます。SmartStore では指定された項目に同じ値を持つ既存の Soup エントリが検索され、次のいずれかの結果になります。

- 同じ値を持つ項目が存在しない場合は、新しい Soup エントリが作成されます。
- 外部 ID 項目が存在する場合、その項目が更新されます。
- 外部 ID と一致する複数の項目が存在する場合、エラーが返されます。

ローカルで新規エントリを作成するときは、通常の upsert を使用します。サーバに新しいエントリをアップロードするときに後で参照できる値を、外部 ID 項目に設定します。

サーバからのデータを使用してエントリを更新する場合は、外部 ID による upsert を行います。こうすることで、同一のリモートエンティティに対して Soup エントリが重複しないようにすることができます。

次のサンプルコードでは、レコードがサーバ上に存在しないため、id 項目に値 new を設定しています。オンラインになったときに、ローカルにのみ存在するレコードをクエリして (id が "new" であるレコードを検索する方法で)、サーバにそれらをアップロードすることができます。サーバからレコードの実際の ID が返されたら、ローカルでその id 項目を更新します。サーバ上にまだ作成されていないカタログに属する製品を作成するような場合は、parentSoupEntryId 項目を介することでカタログとの関係を取得できるようになります。サーバ上にカタログが作成されたら、ローカルレコードの parentExternalId 項目を更新します。

次のコードには、サンプルのシナリオが含まれています。最初に、新しい Soup エントリを作成する upsertSoupEntries をコールします。成功コールバックで、新たに割り当てた Soup エントリ ID を使用して新しいレコードを取得します。次に、説明を変更した後、ForceTK メソッドをコールしてサーバ上に新しい取引先を作成し、それを更新します。最後のコールは、外部 ID による upsert です。コードを読みやすくするために、エラーコールバックは指定されていません。また、すべての SmartStore コールは非同期であるため、実際のアプリケーションでは、各ステップは、前のステップのコールバックで実行する必要があります。

```
// Specify data for the account to be created
var acc = {id: "new", Name: "Cloud Inc", Description: "Getting started"};

// Create account in SmartStore
// This upsert does a create because the acc has no _soupEntryId field
navigator.smartstore.upsertSoupEntries("accounts", [ acc ],
function(accounts) {
    acc = accounts[0];
    // acc should now have a _soupEntryId field and _lastModifiedDate
});

// Update account's description in memory
acc["Description"] = "Just shipped our first app ";

// Update account in SmartStore
// This does an "update" because acc has a _soupEntryId field
```

```

navigator.smartstore.upsertSoupEntries("accounts", [ acc ],
function(accounts) {
    acc = accounts[0];
});

// Create account on server
forcetkClient.create("account", { "Name": acc["Name"], "Description": acc["Description"] }, function(result) {
    acc["id"] = result["id"];
    // Update account in SmartStore
    Navigator.smartstore.upsertSoupEntries("accounts", [ acc ]);
});

// Update account's description in memory
acc["Description"] = "Now shipping for iOS and Android";

// Update account's description on server
// Sync client -> server for entities existing on server
forcetkClient.update("account", acc["id"], {"Description": acc["Description"]});

// Later, there is an account (id: someSfdcId) you want to get locally

// There might be an older version in the SmartStore already

// Update account on client
// sync server/client for entities that might not exist on client
forcetkClient.retrieve("account", someSfdcId, "id,Name,Description",
function(result) {
    // Create or update account in SmartStore
    Navigator.smartstore.upsertSoupEntriesWithExternalId("accounts",
        [ result ], "id");
});

```

Soup エントリの削除

エントリは非同期的に Soup から削除され、成功またはエラーのコールバックが呼び出されます。soupEntryIds は、削除するエントリに含まれる _soupEntryId 値のリストです。

```
navigator.smartStore.removeFromSoup(soupName, soupEntryIds,
successCallback, errorCallback)
```

Soup の削除

Soup を削除するには、removeSoup() をコールします。ユーザがサインアウトした場合も、Soup は自動的に削除されます。

```
navigator.smartstore.removeSoup(soupName, successCallback, errorCallback);
```

SmartStore の拡張

アプリケーションによっては、さまざまな方法で SmartStore API を拡張することでメリットが得られる場合があります。

- セキュアな localStorage — W3C 策定の localStorage は、SmartStore の上に容易に実装できる、シンプルなキーバリューストレージです。たとえば、各プラットフォームの SmartStore プラグインで 1 つの localStorage Soup を作成することができ、matchKey を localStorage メソッドに渡すキーとして使用できます。これは、すでに localStorage について精通し、その制限事項にも対応できる開発者にとっては役に立つレイヤです。この実装の大きな特徴は、Cordova の形式で JavaScript コールバックを使用する必要があり、すべての localStorage メソッドが非同期であるという点です。
- ファイルとラージバイナリオブジェクト — アプリケーションによっては、ビデオ、PDF、および PPT ファイルのようなサイズの大きなバイナリオブジェクトを格納する必要があります。現在のところ Cordova には、このようなアプリケーションで利用できる一貫性のある安全なストレージメカニズムはありません。

第 10 章

上級レベルのトピック

トピック:

- カメラを使用するためのハイブリッドサンプルアプリケーションのカスタマイズ
- バーコードと QR コードのスキャニング
- 地理位置情報とモバイルアプリケーション
- ハイブリッドアプリケーションでの NFC の活用

前章まで、アプリケーションの基本的な作成方法を説明し、目的に応じてサンプルアプリケーションをカスタマイズし機能を追加する方法を紹介してきました。これまで習得した知識だけで十分、プロジェクトを作成してサンプルアプリケーションを構築し、組織に合わせて独自のカスタマイズを行い、組織データを操作することが可能でしょう。以降のセクションでは、デバイスならではの機能を追加することで、アプリケーションの機能をさらに拡張していく方法について説明します。

カメラを使用するためのハイブリッドサンプルアプリケーションのカスタマイズ

このセクションでは、ハイブリッドアプリケーションから主にカメラなどのデバイスのネイティブ機能へアクセスする方法について説明します。すぐに作業を始められるように、サンプルプロジェクトを用意しました。

1. ブラウザから GitHub にアクセスし、新規ディレクトリに GlueCon 2012 Salesforce Mobile SDK Demo プロジェクトをダウンロードします。

```
git clone
https://github.com/metadaddy-sfdc/GlueCon2012-Salesforce-Mobile-SDK-Demo.git
```

2. GlueCon2012-Salesforce-Mobile-SDK-Demo/www ディレクトリからファイルや画像を PC のハイブリッドアプリケーションの www ディレクトリにコピーし、forcetk.js、index.html、および inline.js ファイルは上書き、画像フォルダは新規作成します。

アプリケーションを実行する前に、DE 組織の取引先責任者の標準オブジェクトの設定をいくつかカスタマイズする必要があります。画像の ID を保持するためのカスタム項目と、取引先責任者のページレイアウトに画像を表示するためのカスタム項目をそれぞれ 1 つずつ作成します。このアプリケーションでは、ContentVersion レコードへ画像をアップロードし、新しい ContentVersion レコードの ID を使用して取引先責任者レコードを更新します。

1. DE アカウントにログインし、**あなたの名前** > [設定] > [アプリケーションの設定] > [カスタマイズ] > [取引先責任者] > [項目] の順にクリックします。
2. 下方向にスクロールして、[取引先責任者カスタム項目 & リレーション] セクションを表示し、[新規] をクリックします。
3. [データ型] で [テキスト] を選択し、[次へ] をクリックします。
4. 続く画面で、次のように入力します。
 - **項目の表示ラベル:** 画像 ID
 - **文字数:** 18
5. [次へ] をクリックします。項目レベルのセキュリティはデフォルトのままの設定で、再び [次へ] をクリックします。続く画面で、ページレイアウトのすべての選択を解除し、[保存 & 新規] をクリックします。



メモ: 画像項目はユーザーに表示する必要がありますが、画像 ID はあまりその必要がありません。ここで選択を解除すると、ページレイアウトに表示されなくなります。

次は、数式の項目を作成します。

1. [データ型] で [数式] を選択し、[次へ] をクリックします。
2. 続く画面で、次のように入力します。
 - **項目の表示ラベル:** 画像
 - **数式の戻り値のデータ型:** テキスト

- [次へ] をクリックして、次の数式を入力します。

```
IF( Image_ID_c != null ,
    IMAGE( '/sfc/servlet.shepherd/version/download/' &
Image_ID_c, '' ) ,
    IMAGE('' , '' , 0,0))
```

- [次へ] をクリックします。項目レベルのセキュリティはデフォルトのままの設定で、再び [次へ] をクリックします。ページレイアウトもデフォルトのままの設定で [保存] をクリックします。



メモ: この設定では、画像項目がすべてのページレイアウトに表示されるようになります。

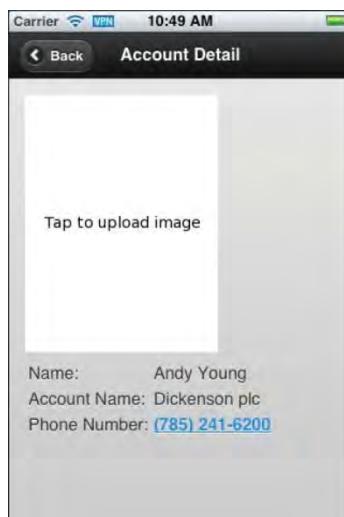
アプリケーションの実行

プロジェクトと取引先責任者の標準オブジェクトの構成が完了したので、これでデモアプリケーションを実行して、取引先責任者に画像をアップロードする準備が整いました。

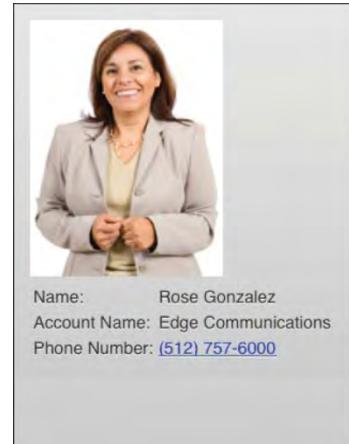


メモ: iOS シミュレータにはカメラがないので、実際のデバイスでアプリケーションを実行する必要があります。

- アプリケーションを実行し、必要があればログインして、[Fetch SFDC contacts] をクリックします。
- このデモアプリケーションでは、リスト内の取引先責任者をクリックすると、その取引先責任者の写真が表示されるプレースホルダが配置された、取引先責任者の詳細ページが開きます。



- プレースホルダをタップすると、カメラが起動します。
- 写真を撮ると、取引先責任者の詳細ページが更新されて写真が挿入され、画像データが ContentVersion レコードにアップロードされて取引先責任者に関連付けられます。



5. この処理が実際に行われているかどうかを確認するために、DE 組織にログインします。
6. [取引責任者] タブをクリックすると、その取引責任者の名前が [最近の取引責任者] のリストに表示されます。
7. 取引責任者の名前をクリックすると、取引責任者の標準データと共に写真が表示されます。

Ms. Rose Gonzalez

[Show Chatter](#) [Follow](#)

[Customize Page](#) | [Edit Layout](#) | [Print](#)

[« Back to List: Remote Access](#)

[Opportunities](#) | [Cases](#) | [Open Activities](#) | [Activity History](#) | [Campaign History](#) | [Notes & Attachments](#) | !

Contact Detail

[Edit](#) [Delete](#) [Clone](#) [Request Update](#) [Work with Portal](#) [Mailing Label](#)

Image

ImageId	068500000000VTG1		
Contact Owner	Andy Admin [Change]	Phone	(512) 757-6000
Name	Ms. Rose Gonzalez	Home Phone	
Account Name	Edge Communications	Mobile	(512) 757-9340
Title	SVP, Procurement	Other Phone	
Department	Procurement	Fax	(512) 757-9000

デモアプリケーションの動作の仕組み

index.html と inline.js を詳しく見てみると、カスタマイズ前のサンプルアプリケーションとの違いが数多くあります。デバイスの取引先責任者リストと取引先リストが index.html から削除され、画像と名前、取引先名、電話番号の項目を表示する新しい [Contact Detail] ページがあります。

まず inline.js の onSuccessSfdcContactList() から見ていきましょう。

```
function onSuccessSfdcContactList(response) {
    var $j = jQuery.noConflict();

    SFHybridApp.logToConsole("onSuccessSfdcContactList: received " +
        response.totalSize + " contacts");

    $j("#div_sfdc_contact_list").html("");
    var ul = $j('<ul data-role="listview" data-inset="true" data-theme="a" data-dividertheme="a"></ul>');
    $j("#div_sfdc_contact_list").append(ul);

    ul.append($j('<li data-role="list-divider">Salesforce Contacts: ' +
        response.totalSize + '</li>'));
    $j.each(response.records, function(i, contact) {
        var id = contact.Id;
        var newLi = $j("<li><a href='#" + (i+1) + "-" + contact.Name +
        "</a></li>");
        newLi.click(function(e){
            e.preventDefault();
            $j.mobile.showPageLoadingMsg();
            forcetkClient.query("SELECT Id, Name, Account.Name, Phone,
Image_ID__c " +
                "FROM Contact WHERE Id = '" + id + "'",
                onSuccessSfdcSingleContact, onErrorSfdc);
        });
        ul.append(newLi);
    });
    $j("#div_sfdc_contact_list").trigger( "create" )
}
```

渡された各レコードに対し、応答内で、その名前のリストアイテムを作成し、リストアイテムにクリックハンドラを設定して、取引先責任者の名前、取引先名、電話番号、画像を取得できるようにします。

クエリコールバック onSuccessSfdcSingleContact() で、取引先責任者の詳細ページを表示します。取引先責任者の画像を表示するコードに注目してください。

```
//Set up image
$j('#Image').attr('data-id', contact.Id);
$j('#Image').attr('data-name', contact.Name);
if (contact.Image_ID__c) {
    // Load image data
    $j('#Image').attr('src', "images/loading.png");
```

```
$j.mobile.changePage('#jqm-detail');

forcetkClient.retrieveBlobField("ContentVersion",
    contact.Image_ID_c, "VersionData", function(response) {
        var base64data = base64ArrayBuffer(response);
        $j('#Image').attr('src', "data:image/png;base64,"+base64data);

        $j.mobile.hidePageLoadingMsg();
    }, onErrorSfdc);
} else {
    // Display a default image
    $j.mobile.hidePageLoadingMsg();
    $j('#Image').attr('src', "images/blank.png");
    $j.mobile.changePage('#jqm-detail');
}
```

取引先責任者の ID と名前は属性として画像要素に格納されており、Image_ID_c カスタム項目に ID がある場合は、"読み込み中" の画像が表示され、画像データが retrieveBlobField() によって取得されます。

ユーティリティ関数 base64ArrayBuffer() は、JavaScript ArrayBuffer オブジェクトを base64 でエンコードされた文字列データに変換し、コールバックで画像データをデータ URI として設定します。

regLinkClickHandlers() を見てみると、この関数からは #link_fetch_device_contacts と #link_fetch_sfdc_accountshandlers が削除されていますが、次のコードが記述されています。

```
$j('#Image').click(function() {
    getPhotoAndUploadToContact($j(this).attr('data-name'),
        $j(this).attr('data-id'));
});
```

画像をクリックすると、getPhotoAndUploadToContact() 関数がコールされ、画像要素に格納されている取引先責任者の名前と ID 属性が渡されます。

```
function getPhotoAndUploadToContact(name, contactId) {
    SFHybridApp.logToConsole("in capturePhoto, contactId = "+contactId);

    $j('#Image').attr('data-old-src', $j('#Image').attr('src'));
    $j('#Image').attr('src', "images/camera.png");

    navigator.camera.getPicture(function(imageData){
        onPhotoDataSuccess(imageData, name, contactId);
    }, function(errorMsg){
        // Most likely error is user cancelling out of camera
        $j('#dialog-text').html(errorMsg);
        $j.mobile.changePage('#jqm-dialog');
        $j('#Image').attr('src', $j('#Image').attr('data-old-src'));
        $j('#Image').removeAttr('data-old-src');
    }, {
        quality: 50,
        sourceType: Camera.PictureSourceType.CAMERA,
        destinationType: Camera.DestinationType.DATA_URL
    });
}
```

`getPhotoAndUploadToContact()` では、ハイブリッドアプリケーションの利点を活用しています。`navigator.camera` の `getPicture()` 関数を使用すると、デバイスのカメラに簡単にアクセスできます。`getPicture()` へ渡すオプションで、必要とする画質、ソース (この場合は、デバイスの写真ライブラリではなくカメラ)、および返される写真の形式を指定します。`Camera.DestinationType.DATA_URL` は、base64 でエンコードされた文字列として画像を返し、`Camera.DestinationType.FILE_URI` は、デバイス上のファイルへの URI を返します。

写真が正常に撮られると、`onPhotoDataSuccess()` コールバックが呼び出されます。

```
function onPhotoDataSuccess(imageData, name, contactId) {
    var $j = jQuery.noConflict();

    SFHybridApp.logToConsole("in onPhotoDataSuccess, contactId =
"+contactId);

    // Update the image on screen
    $j('#Image').attr('src', "data:image/jpeg;base64," + imageData);

    // Upload the image data to Content
    $j.mobile.showPageLoadingMsg();
    forcetkClient.create('ContentVersion', {
        "PathOnClient" : name + ".png",
        "VersionData" : imageData
    }, function(data){
        // Now update the Contact record with the new ContentVersion
        Id
        SFHybridApp.logToConsole('Created ContentVersion ' + data.id);

        forcetkClient.update('Contact', contactId, {
            "Image_ID__c" : data.id
        }, function(){
            $j.mobile.hidePageLoadingMsg();
            SFHybridApp.logToConsole('Updated Contact '+contactId);
        }, onErrorSfdc);
    }, onErrorSfdc);
}
```

画面上の画像を更新すると、`ContentVersion` レコードが作成されます。`PathOnClient` 項目では拡張子を付けることで、データの種類を示しています。`ContentVersion` レコードの作成が成功すると、新しいレコードの ID で取引先責任者が更新されます。

バーコードと QR コードのスキヤニング

ハイブリッドアプリケーションは、モバイル開発における Web 領域とネイティブ領域の中間に位置します。Web アプリケーションと同様に、ハイブリッドモバイルアプリケーションの開発には、主に HTML5 や JavaScript、CSS などの Web テクノロジが採用されています。そこで、Salesforce Mobile SDK コンテナ (オープンソースの PhoneGap プロジェクトに基づく) を使用して、Web アプリケーションを簡単なネイティブラッパーでパッケージし、カメラ、マイクなどのネイティブのデバイス機能にアクセスします。ハイブリッドアプリケーションではデバイスのネイティブ機能にアクセスできるので、モバイルデバイスで撮った写真を Salesforce の商品レコードに追加することも可能です。ハイブリッドアプリケーション開発の別の典型的なユースケースとして、バーコードや QR コードのスキヤニングが挙げられます。



では、どのような方法でハイブリッドモバイルアプリケーションにバーコードのスキャニング機能を追加するのでしょうか? PhoneGap (通称 Apache Cordova) ハイブリッドフレームワークの優れた点を、ここで活用できます。PhoneGap には、プッシュ通知、モバイル決済 (PayPal を使用)、さらにバーコードや QR コードのスキャンのような高度なユースケースをサポートするために、コミュニティによって構築されたオープンソースコンポーネントの豊富なプラグインライブラリが用意されています。実際に Salesforce Mobile SDK 自体が、OAuth ログインプロセスと安全なオフラインストレージ (SmartStore) をサポートするために、PhoneGap のプラグインを使用しています。

たとえば、ユーザがバーコードや QR コードをスキャニングして、商品レコードを検索できるようにするために、MerchandiseMobile の Visualforce ページの内容を充実させたいとします。GitHub でこのアプリケーションの最終的なコードベースを調べることができますが、ここでは、バーコードのスキャニング機能をハイブリッドモバイルアプリケーションへ追加する方法を順を追って紹介します。

1. Salesforce Mobile SDK を使用して、Visualforce ページをハイブリッドアプリケーションに改造します。詳細については、『Mobile SDK Workbook』のチュートリアル 5 (iOS 対応ハイブリッドアプリケーション) と 6 (Android 対応ハイブリッドアプリケーション) を参照してください。
2. PhoneGap プラグイン用の GitHub リポジトリをローカルマシンにダウンロードします (git クローンを使用するか、ページ右上の [Download] リンクをクリックします)。開発目標のモバイルプラットフォームに応じて、Readme ファイルの指示に従い、Android または iOS プロジェクト内にバーコードスキャナのプラグインをインポートします。
3. PhoneGap プラグインの Git リポジトリに含まれている barcodeScanner.js ファイルを Visualforce ページにインポートします。たとえば、MerchandiseMobile Visualforce ページの簡単な例では次のようにになります。

```
<apex:includeScript
value="{!!URLFOR($Resource.BarCodeScanner)}" />
```



メモ: また、Visualforce ページでコア PhoneGap JS ファイルをインポートする必要があります。

4. これで、Visualforce ページからバーコードのスキャンを開始する準備が整いました。PhoneGap の素晴らしいところは、iOS や Android 固有のコーディングを必要とせず、JavaScript すべてのデバイス機能にアクセスできることです。次に、Visualforce ページでバーコードスキャナ PhoneGap プラグインを起動する方法を示します。

```

function scanBarcode() {
    window.plugins.barcodeScanner.scan(onBarcodeScanSuccess,
    onBarcodeScanFailure);
}

function onBarcodeScanSuccess(result) {
    MerchandiseMobileController.getMerchandiseRecByBarCode(
        result.text,
        function(res,event){
            if (res != null){
                $j.mobile.showPageLoadingMsg();

                $j('#merchandiseName').html(res.Name);

                $j('#description').html(res.Description__c);

                $j('#inventory').html(res.Total_Inventory__c);
                $j('#price').html('$'+res.Price__c);

                $j('#merchandiseId').val(res.Id);
                $j.mobile.hidePageLoadingMsg();

                $j.mobile.changePage('#detailpage',
                {changeHash: true});
            }
            else{
                alert('Sorry. No matching
Merchandise records found.');
            }
        });
}

function onBarcodeScanFailure(error) {
    console.log("Barcode scanning failed: " + error);
}

```

2行目には、カスタム PhoneGap プラグインを使用したバーコードスキャニングがいかに簡単に実行できるかを示しています。バーコードまたは QR コードのスキャンが成功した場合、成功コールバック関数は、スキャンされたテキストまたは文字列(上記の `result.text`)で呼び出されます。このプラグインでは、`result.format` 変数を使用してスキャンされたバーコードの形式 (QR_CODE、UPC_A、DATA_MATRIX など) に準拠して処理されます。後は、Javascript Remoting を使用して、この Visualforce ページに添付された Apex コントローラのメソッド(6行目)をコールし、スキャンされたバーコードの値と一致している商品レコードを検索するだけです。

上記のようなハイブリッドアプリケーションでの開発アプローチに加え、ネイティブ Android や iOS アプリケーションの開発に慣れている方は、ネイティブモバイルアプリケーションにバーコードスキャニング機能を実装することも可能です。

地理位置情報とモバイルアプリケーション

複合項目の地理位置情報は、経度と緯度の 2 つのコンポーネントで構成されています。地理位置情報は、特に、任意のオブジェクト（多くの場合、取引先責任者、取引先、レストラン、店舗など、住所に関連するオブジェクト）の位置の地理座標を保持するように設計されています。地理位置情報項目がこれらのオブジェクトに追加され、座標が設定されれば、ユーザは半径をベースとした検索を実行することができます。たとえば、現在の位置から 5 マイル圏内のレストランの情報を探したり、スポーツ競技場から 15 マイルの範囲内にあるすべての家を探したりすることができます。

モバイルアプリケーションにとって、この新しい機能は、モバイルから位置情報をベースに Salesforce オブジェクトを検索する機能を提供する場合に特に便利です。通常のカスタム項目と同様に、地理位置情報項目は、オブジェクトのセットアップウィザードを使用して、オブジェクトに追加することができます。セットアップ時に小数点精度を設定することで、小数または度、分、秒表記で座標を表示できます。



メモ: この地理位置情報機能はベータリリースであり、後述するように、この機能には既知の制限があります。地理位置情報機能に関するフィードバックは、IdeaExchange にお寄せください。

地理位置情報カスタム項目の作成

1. あなたの名前>[設定]>[作成]>[オブジェクト] の順に選択し、リストからいずれかのオブジェクトを選択します。
2. [カスタム項目 & リレーション] セクションで、[新規] をクリックします。
3. [地理位置情報] を選択し、[次へ] をクリックします。
4. 地理位置情報項目の各属性を入力します。[緯度および経度表示の表記法] 属性では、Salesforce インターフェースで表示される位置情報の表記法を次のいずれかから選択します。

度、分、秒

60 を単位とする角度測定の表記法。円周を 360 等分した角度を 1 度とし、1 分は 1 度の 60 分の 1、1 秒は 1 分の 60 分の 1 を表わします。

小数

位置情報を度で表し、分と秒を小数に変換します。小数表記法では、方位起点は使用しません。北緯と東経は正の値で示し、南緯と西経は負の値で示します。

5. 手順に従ってウィザードを完了します。

半径ベースの検索とフィルタリング

地理位置情報項目を含むオブジェクトのリストビューでは、WITHIN 演算子を使用して、半径ベースの検索とフィルタリングを行います。数式項目内に DISTANCE(loc1, loc2, distance_unit) 関数を使用して、2 つの測位ポイント間の距離を計算できます。

また、SOQL は、DISTANCE 関数と GEOLOCATION 関数が強化されており、位置ベースの検索を実行する SOQL クエリを記述することができます。たとえば、カリフォルニア州サンフランシスコのマーケットストリート 1 番地の位置を地理座標で表すと 37.794915、-122.394733 になります。半径 1 マイルの範囲にあるすべてのレストランの名前と電話番号を検索するには、次のような SOQL クエリを記述します。

```
SELECT name__c, phone__c
  FROM restaurant__c
 WHERE DISTANCE(loc__c, GEOLOCATION(37.794915,-122.394733), "mi") <=
 1
```

地理位置情報項目に関する制限

地理位置情報は、緯度、経度、および内部使用に対応した 3 つのカスタム項目からなる複合項目であり、組織のタイプによって機能の制限を受けます。このベータリリースでは、Salesforce で使用している機能によって複合項目（地理位置情報）がサポートされるか、項目の各コンポーネント（緯度、経度）のみサポートされるかが異なります。たとえば、ある組織では地理位置情報項目とそのコンポーネントを表示するリストビューを作成することは可能ですが、Apex で複合項目としての地理位置情報を選択することはできません。地理位置情報項目のコンポーネントに対して SOQL クエリを実行できるだけです。

この地理位置情報のベータリリースには、その他にも以下のような機能制限があります。

- 履歴追跡は地理位置情報項目では使用できません。
- カスタム設定では地理位置情報項目はサポートされません。
- 地理位置情報項目は、レポート、ダッシュボード、入力規則、Visual Workflow、ワークフローおよび承認では利用できません。
- 地理位置情報項目は検索できません。
- 地理位置情報項目はスキーマビルダーでは使用できません。
- DISTANCE と GEOLOCATION の数式関数を使用できるのは、数式項目を作成する場合と Visual Workflow 内でのみです。
- Apex では、SOQL クエリを介した操作かコンポーネントレベルでのみ地理位置情報機能がサポートされます。

ハイブリッドアプリケーションでの NFC の活用

カンファレンス会場から、顧客と交換した名刺をすばやく簡単に Salesforce 組織にアップロードして、正規の取引先責任者レコードとして登録できたらどんなに便利か想像してみてください。スマートフォンのアプリケーションを開く必要も、QR コードをカメラで撮る必要もありません。後で名刺の写真をスキャンしたり、プロフィールを転記したりする必要もありません。名刺をスマートフォンにかざすだけで、瞬時に情報が Salesforce にアップロードされるというわけです。Salesforce Touch Platform と近距離無線通信 (NFC) 対応デバイスの組み合わせでこれが実現されます。

NFC は Near Field Communication (近距離無線通信) の略称です。至近距離 (数センチ以下) で、デバイスどうしが無線でデータを交換できる技術です。NFC でデータを送信できるデバイスを「タグ」と呼びます。丸いステッカー、名刺、クレジットカード、または名札な

ど、タグの形状や大きさはさまざまです。保存できるデータの量も数バイトから4キロバイト以上まで、デバイスによって異なります。もちろん、タグ上のデータは、さまざまな形式でエンコードすることができます。現在主流と言えるのは、NDEF (NFC Data Exchange Format) というフォーマットです。NDEF フォーマットでタグをフォーマットすると、このフォーマットを活用するシステム間でデータを簡単に通信することができます。たとえば、Android デバイスは NDEF フォーマットをサポートしており、初めて NFC を使う場合にも簡単に利用できます。

NFC の要件

屋外でこのアプリケーションを実行するには、NFC をサポートするモバイルデバイスが必要です。インターネットで「NFC スマートフォン」を検索すれば、オンラインで Android の NFC 対応機種のリストを取得することができます。また、通信するための NFC タグも必要になります。同じく、インターネットで「NFC タグ」を検索すれば、NDEF がフォーマット済みのタグや未フォーマットのタグを販売する小売業者のリストを取得できます。小売業者から NDEF フォーマット済みのタグを入手すれば、データをエンコードする手間を省くことができます。また、NFC タグが保存できるデータの容量についても必ず確認が必要です。目的によっては、データ容量の大きいタグが必要になることがあります。

NFC 対応のハードウェアを持っていない場合は、インターネットでダウンロード可能なソフトウェアの NFC エミュレータもいくつかあります。ソフトウェアエミュレータについてはここでは取扱いません。

Force.com と NFC モバイルアプリケーションのアーキテクチャ

自己流のやり方ではなく、あくまでも正しい手順に従ってアプリケーションを開発したい場合は、NFC vCard Cloud Loader アプリケーション (https://github.com/corycowgill/NFC_vCard_Loader) から Android ハイブリッドアプリケーションのソースコードをダウンロードしてください。

PhoneGap を使用する方法もあります。PhoneGap の優れた点は、そのプラグインアーキテクチャにあります。PhoneGap フレームワークでは、開発者は、デバイスのハードウェアを活用し、JavaScript を介してデバイスからモバイルアプリケーションの HTML にデータを返すプラグインを作成することができます。NFC はデバイスへのネイティブ API コールに依存するため、Chariot Solutions の Don Coleman 氏が作成したオープンソースの NFC PhoneGap プラグインをアプリケーションに実装します。この NFC プラグインは、Github サイト (<https://github.com/chariotsolutions/phonegap-nfc>) から入手できます。

NFC PhoneGap プラグインのインストール

PhoneGap プラグインのインストールは、とても簡単です。以下の数ステップでインストールを完了できます。

1. <https://github.com/chariotsolutions/phonegap-nfc/downloads> にアクセスし、phonegap-nfc-android.jar ファイルとプラグイン用の JavaScript ファイル (phonegap-nfc.js) をダウンロードします。
2. プラグイン用の JAR ファイルをアプリケーションのライブラリフォルダにコピーし、JAR ファイルがアプリケーションのクラスパスにあることを確認します。

3. JavaScript ファイル (phonegap-nfc.js) をアプリケーションの JavaScript フォルダにコピーします。

4. HTML ファイルに次のような変更を加え、プラグインの JavaScript ファイルを追加します。

```
<script type="text/javascript"
src="[view-source:https://dl-web.dropbox.com/get/SalesforceMobileSDK-
Android/
hybrid/SampleApps/ContactExplorer/assets/www/phonegap-nfc-0.2.0.js
phonegap-nfc-0.2.0.js] script>
```

5. PhoneGap プラグインの XML ファイル (plugins.xml) に次のような変更を加え、プラグインクラスを追加します。

```
<plugin name="NfcPlugin"
value="com.chariotsolutions.nfc.plugin.NfcPlugin"/>
```

6. Android マニフェストファイルに次のような変更を加え、アプリケーションデバイスに NFC の使用を許可します。

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

7. オプション – Android には、NFC タグ専用の通知システムがあります。Android のバックグラウンドで実行する NFC タグのディスパッチャを設定し、Android デバイスが NFC タグをスキャンするとアプリケーションが自動的に起動するようにします。この設定により、NFC タグが読み込まれたときにアプリケーションを即座に起動させることができます。このシステムを使用するには、適切な intent タグを使用して Android マニフェストファイルに変更を加えます。

```
<intent-filter>
<action android:name="android.nfc.action.NDEF_DISCOVERED" />
<data android:mimeType="text/pg" />
<category android:name="android.intent.category.DEFAULT" />
</intent-filter>
```

以上の簡単なステップにより、NFC プラグインを使用するためのワークスペースとアプリケーションを構成できました。それでは、次はアプリケーションのコードを詳しく見てていきましょう。

JavaScript による NFC プラグインの起動

NFC は、イベント駆動型の機能です。そのため、Android のタグディスパッチャによって開始される NFC タグのスキャンをアプリケーションに監視させ、スキャンの開始と同時にロジックを実行せらるようになります。負荷の大きい NFC タグの読み込みイベントの処理は、NFC PhoneGap プラグインに任せます。必要な作業は、NFC プラグインからのコールバックを処理するために、JavaScript でリスナーを登録するだけです。これを行うには、次のように、PhoneGap の deviceready メソッド内にプラグインオブジェクト用の NFC リスナーを登録します。

```

// When this function is called, PhoneGap is initialized
function onDeviceReady() {
    SFHybridApp.logToConsole("onDeviceReady: PhoneGap ready");
    //Call getAuthCredentials to get the initial session credentials

SalesforceOAuthPlugin.getAuthCredentials(salesforceSessionRefreshed,
                                         getAuthCredentialsError);

    //receive notifications when autoRefreshOnForeground refreshes
    //the session
    document.addEventListener("salesforceSessionRefresh",
                            salesforceSessionRefreshed, false);

    //enable buttons
    regLinkClickHandlers();

    //Use the NFC Plugin to configure the NFC Tag Reader Listener

    nfc.addNdefListener(
        onNfcRead,
        successNFCRegisterListener,
        errorNFCRegisterListener
    );
    regLinkNFCClickHandlers();
}

}

```

上記のように、NFC タグから返されるタグデータを処理するコールバックハンドラー (onNfcRead) を指定します。また、successNFCRegisterListener と errorNFCRegisterListener の 2 つのコールバックメソッドを追加しています。これらのメソッドは、JavaScript でリスナーが正常に登録された時点で実行されます。

さらに、NFC タグから返された結果の処理方法を JavaScript で指定することができます。以下のコードは、NFC タグのスキヤン時に画面にタグのデータを表示します。次に、vCard フォーマットでフォーマットされたタグデータを解析し、contact 変数に格納します。この contact 変数は最終的に Mobile SDK に渡され、データが Salesforce に挿入されます。

```

function onNfcRead(nfcEvent) {
    console.log(JSON.stringify(nfcEvent.tag)); // Debug in Console
    clearScreen(); // Clear Previously Display Tag Info
    var tag = nfcEvent.tag;
    var records = tag.ndefMessage || [],
        //Tag Content HTML Div
        display = document.getElementById("tagContents");
    display.appendChild(
        document.createTextNode(
            "Scanned an NDEF tag with " + records.length +
            " record" + ((records.length === 1) ? "" : "s")
        )
    );

    // Display Tag Info
    var meta = document.createElement('dl');
    display.appendChild(meta);

    if (device.platform.match(/Android/i)) {
        if (tag.id) {
            showProperty(meta, "Id", nfc.bytesToHexString(tag.id));
        }
    }
}

```

```

        }
        var hcard = document.createElement("div");
        hcard.innerHTML =
vCard.initialize(nfc.bytesToString(tag.ndefMessage[0].payload)).to_html();

        display.appendChild(hcard);
        var vCardVal =
vCard.initialize(nfc.bytesToString(tag.ndefMessage[0].payload));
//parse the Tag formatted vCard data into a Javascript Object
        var names = vCardVal.fn.split(" ");//Take Fullname from vCard
contact.firstName = names[0]; //Put in Firsntname for SFDC
contact.lastName = names[1]; //Put in LastName for SFDC
for(type in vCardVal.email)
{
    for(n in vCardVal.email['home'])
    {
        //Store email in contact variable for SFDC Upsert
        contact.email = vCardVal.email['home'][n];
        contact.vCard_Email__c = contact.email;
    }
}
for(type in vCardVal.tel)
{
    for(n in vCardVal.tel[type])
    {
        //Store Telephone Info in variable for SFDC Upsert
        contact.phone = vCardVal.tel[type][n];
    }
}
meta.appendChild(hcard);
}
navigator.notification.vibrate(100);
}

```

上記のコードからわかるように、JavaScript で NFC プラグインから返された NDEF タグのペイロードが処理されます。その後、タグ情報を解析し、vCard フォーマットでフォーマットし、DOM に追加します。タグ情報は、ユーザが見ている画面上に表示されます。vCard フォーマットについて補足すると、これは主に名刺データを電子的に表現する方法として、広く使われているフォーマットです。

Mobile SDK を使用した Force.com への情報の更新/挿入 (upsert)

タグを正常に読み込むことができれば、そのデータをさまざまな目的に利用できます。このサンプルアプリケーションでは、Mobile SDK を使用して、標準の取引先責任者オブジェクトに取引先責任者の情報を更新/挿入 (upsert) します。データを upsert するためのメソッドは、`forcetk.js` ファイルに記述されています。このファイルでは Force.com REST API を利用してデータを処理します。上記の NFC 処理用のコードに埋め込んだ `contact` 変数を Force.com への入力として使用します。最終的には、ユーザがページ上のボタンをクリックすることで Force.com へのアップロードが開始されますが、上記で追加した読み込み成功時のメソッドを使用して、この操作を完全に自動化することもできます。

```
$j('#insert_lead_from_nfc_tag').click(function(){
    SFHybridApp.logToConsole("Upsert vCard Contact to SFDC " +
        contact.vCard_Email__c + contact.lastName + contact.phone);
    forcetkClient.upsert("Contact", "Contact_Ext_ID__c",
        contact.contactextid, contact, onSuccessSfdcUpsertContact, onErrorSfdc);
});
```

一連の操作を最後まで実行した後に Force.com の [取引先責任者] タブを開き、[最近の取引先責任者] のリストを表示すれば、NFC タグから新しく作成された取引先責任者を見るることができます。

まとめ - モノのインターネットと NFC の未来

このシンプルなアプリケーションからもわかるように、NFC タグやスマートフォンを使用することで、「モノのインターネット (Internet of Things)」の可能性が広がります。簡単に言うと、現実世界のあらゆる「モノ」をインターネット上で表現できるようになるということです。たとえば、名刺に NFC タグが埋め込まれていれば、アプリケーションを介して、自動的に Salesforce のアプリケーションへデータを送信することができます。これにより、現実世界の名刺というモノを、Salesforce 組織内で表現することができます。

ほぼすべてのモノにタグを付け、Salesforce 内のオブジェクトとして追跡することができます。マーケティング用のキオスク端末にタグを付ければ、営業担当者がカンファレンス会場でリードを処理することもできます。商取引にタグを導入することも可能です。顧客が、カタログから製品をスキャンして買い物リストに追加したり、スマートフォンを使って支払を自動処理したりできるようになり、全体的な購買プロセスが効率化されます。都市部の公共の交通機関では、乗車カードの代わりにスマートフォンのアプリケーションを利用できるようになるでしょう。

第 11 章

AppExchange を利用したモバイルアプリケーションの配信

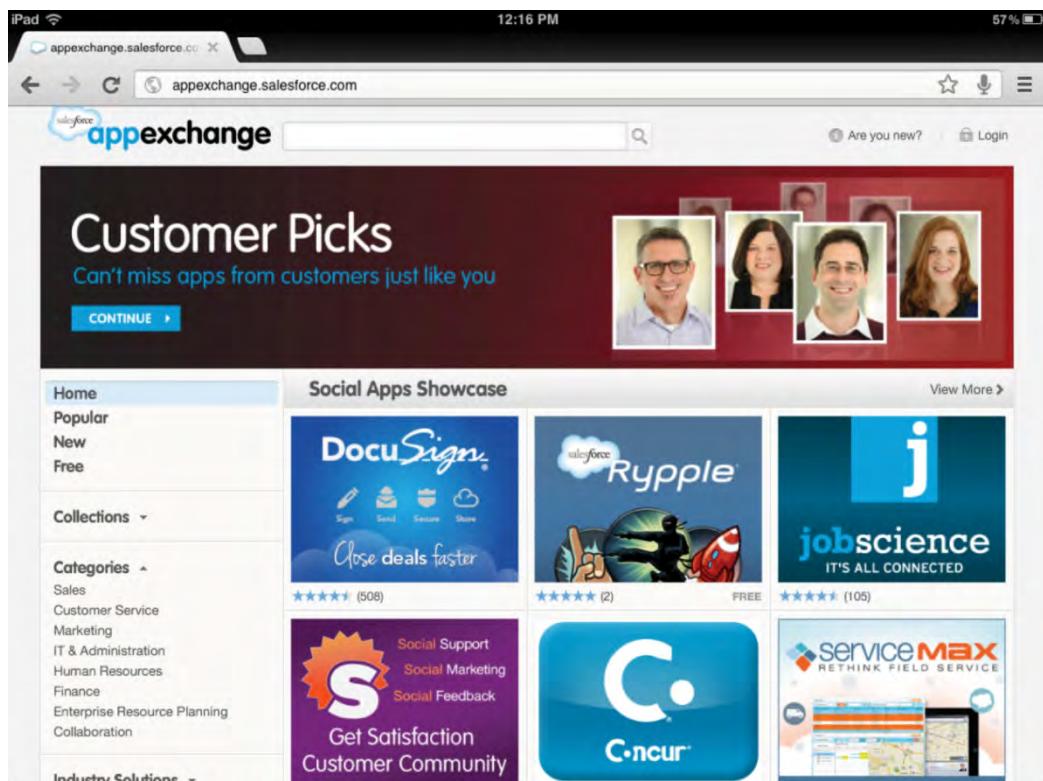
トピック:

- AppExchange for Mobile: エンタープライズ向けモバイルアプリケーション
- AppExchange パートナー プログラムへの参加
- 配信組織の作成
- プロバイダプロファイル の作成
- AppExchange のセキュリティレビュー

アプリケーションによって、モバイル利用は全く別の次元に到達しました。消費者が新しいスマートフォンやタブレットを選択するときは、必ずと言っていいほど、そのデバイスでどのようなアプリケーションが使用できるかを評価しています。それがデバイスを選択するうえで最も重要な要素になっているのです。ですから当然、モバイルアプリケーションを開発したら、それを一般に公開し、顧客や販売店が容易に見つけ、購入してインストールすることができるようにならなければなりません。Android と iOS には、モバイルアプリケーションを登録して配信できる独自のストアがありますが、それらについてはここでは触れません。セールスフォース・ドットコムにも、Salesforce に対応したモバイルアプリケーションやコンサルティングサービスを公開できる、AppExchange というパートナー向けのストアがあります。

AppExchange for Mobile: エンタープライズ向けモバイルアプリケーション

一般消費者向けのアプリケーションストアでは、モバイルアプリケーションの登録件数は 50 万件近くに達します。このようなストアで、Salesforce エコシステム内で安全に動作する信頼の置けるエンタープライズ向けアプリケーションを探すには、多大な時間と労力が必要です。お客様が目的に適ったモバイルアプリケーションを見つけ、Salesforce を利用している数百万のユーザへ開発者がアプリケーションを配信できるよう、Salesforce では、エンタープライズ向けモバイルアプリケーションに特化した世界初のクロスプラットフォームマーケットプレイス、「AppExchange」(<http://appexchangejp.salesforce.com/>)を開設しました。



AppExchange for Mobile は Salesforce ユーザと開発者とを結ぶマーケットプレイスです。

- Salesforce ユーザは、信頼性の高い最新のモバイルアプリケーションを見つけ、これを現在の Salesforce アカウントと連携させ、クラウド上の既存のデータを活用することができます。
- 独立系ソフトウェーベンダー (ISV) は、Android や iOS などのモバイルプラットフォームで動作するネイティブアプリケーションやハイブリッドアプリケーション、HTML5 アプリケーションを、一元管理されたリポジトリへアップロードすることができます。アプリケーションの提供形態は、無償、定価、サブスクリプション型など、自由に選択できます。

特定の問題を解決するための専用機能を提供するモバイルアプリケーションの開発に取り組んでいる方にも、あるいは特定の業務向けの総合ソリューションを開発している方にも、AppExchange は有用です。

AppExchange でモバイルアプリケーションを公開するには、セールスフォース・ドットコムのパートナープログラムに参加し、次の手順に従っていただく必要があります。

1. AppExchange パートナープログラムに参加します。
2. パートナー ポータルでケースを登録して、AppExchange 配信組織 (APO) を作成します。
3. AppExchange でプロバイダのプロファイルを作成します。
4. セキュリティレビューの実施をリクエストします。
5. パートナー ポータルでケースを登録し、AppExchange for Mobile へのアプリケーションのリストイングをリクエストします。



メモ : Salesforce 管理者が組織内での配布を目的としたモバイルアプリケーションを作成する場合、AppExchange でリストイングを公開する必要はありません。

AppExchange パートナープログラムへの参加

まずは、AppExchange パートナープログラムに参加する必要があります。このパートナープログラムは、独立系ソフトウェアベンダー (ISV) の皆様が、Salesforce プラットフォームをビジネスに役立てていただけるよう支援するためのものです。

1. 以下の URL にアクセスして「アプリケーションパートナープログラム」の説明を読み、ページ末尾の [2. アプリケーションパートナーになる] をクリックします。
http://www.salesforce.com/jp/partners/isv_partners.jsp
2. [アプリケーションパートナーになる] ページが開いたら、[オンラインサインアップ] をクリックしてフォームに必要事項を入力します。
3. しばらくすると、あなたのユーザ名と仮のパスワードを記載したメールが届きます。パートナー ポータル (<https://sites.secure.force.com/partners/PP2PartnerLoginPage>) へのリンクをクリックします。
4. 利用規約に同意して、次に表示されるポップアップ ウィンドウを閉じます。
5. このページを今後頻繁に使うことになるので、ブラウザのブックマークに登録しておきましょう。

パートナー ポータルには、AppExchange をすぐに活用できるよう、最もよく使用されているリソース、ドキュメント、ビデオへのリンクが掲載されています。これらの情報のほとんどは、Salesforce ユーザにアドオンやサービスを提供する ISV 向けの内容です。モバイルアプリケーションの開発や公開について、AppExchange パートナープログラムの担当者と密接に情報を交換できます。

配信組織の作成

モバイルアプリケーションの配信とサポートを管理するには、販売、マーケティング、およびサポート機能を完全に備えた Salesforce 組織が必要になります。この組織は、AppExchange Publishing Org (AppExchange 配信組織)、略して APO と呼ばれます。認定パートナーはパートナーポータルを介して、無料で配信組織を作成できます。

1. パートナーポータルにアクセスして、[Quick Links] セクションで [Create a Case] をクリックします。



図 1: [Quick Links] セクションの [Create a Case]

2. [Category] セクションで、最初のオプションを選択します。
3. 左のカテゴリボックスで、[AppExchange & Service Listings] を選択します。
4. 2番目のカテゴリボックスで、[Request CRM for Partner] を選択します。
5. [Reason] ドロップダウンボックスで、[Administration Request] を選択します。
6. [Severity] ドロップダウンボックスで、[High] を選択します。
7. [Subject] に、「Need ISV CRM」と入力します。
8. [Description] フィールドに、既存の Salesforce 組織があるか、または新しい組織が必要かどうかを記入します。既存の Salesforce 組織を使用する場合は、[Description] フィールドに組織 ID を入力します。この場合は、2つの CRM ライセンスがその組織に追加されます。既存の組織がない場合は、新しい組織が提供されます。いずれの場合も、仕事用のメールアドレスを必ず入力してください。最後に [Save] をクリックします。

Case Information

Category: *

I have a question or issue relating to a salesforce.com partner program.

I need developer or technical support.

choose a category

AppExchange & Services Listings >

Content Syndication >

Educations & Certifications >

General Portal Requests >

Joint Marketing >

Extend Org Expiration

License Management Application

Packaging/Uploading/Installing Apps

Request CRM for Partner

Request Subsequent Security Review

Security Review Questions

For more information about CRM for ISV Partners, please visit the following URL:
http://wiki.developerforce.com/index.php/CRM_for_ISV_Partners

Reason: * Administration Request

Severity: Level 3 - High

Description

Subject: * Need CRM org

Description: * To add 2 licenses to your EE org, please provide the Org ID
(apply only to existing Salesforce EE customers):
Please provide us your business mailing address:

Have an attachment? Click Save and add it on the next screen.

Save Cancel

図 2: パートナー ポータルでのケースの作成

9. しばらくすると、ログインとパスワードの変更を求めるメールが届きます。パスワードの変更が完了したら、このページもブックマークに追加しておきましょう。

プロバイダプロファイルの作成

プロバイダプロファイルを使用して、AppExchange に会社情報を公開します。これには、ビジネスを管理する Salesforce 組織にログインする必要があります。認定パートナー様には、APO 組織がすでに提供されています。APO 組織がない場合は、Developer Edition 組織を利用できます。

1. ログインページで、APO のユーザ名とパスワードを入力します。
2. プロバイダプロファイル内に情報を記入し、[保存] をクリックします。

AppExchange のセキュリティレビュー

AppExchange にアプリケーションをリストティングするには、アプリケーションを提出してセキュリティレビューを受ける必要があります。セキュリティレビューを滞りなく済ませるには、http://wiki.developerforce.com/page/JP:Security_Review で公開されているセキュリティのガイドラインとプロセスを十分理解する必要があります。

次の手順はパッケージ化されたアプリケーションを提出するためのものですが、モバイルアプリケーションの場合も手順は同じです。以下のフォームを送信すると、次のステップについて担当者から連絡が届きますのでお待ちください。

1. リスティング編集画面で、[提供] タブの [レビューの開始] をクリックします。
2. アプリケーションを有償にするかどうかを選択します。無償アプリケーションの場合は、セキュリティレビュー 자체は必須ですが、レビュー料金は免除されます。
3. アプリケーションが有償の場合は、2 営業日以内に、パートナーオペレーション担当者からメールでレビュー料金の支払方法に関する情報をお送りいたします。レビュー料金は年払いとなります。
4. アプリケーションが、Force.com 以外の Web サービス（自社のサーバも含む）と連携して動作するものであるかどうかを指定します。
5. アプリケーションが他の Web サービスと連携するものである場合、[使用する Web サービス] ボックスにすべてのサービス名を入力します。入力可能な文字数は最大 1,000 文字です。
6. アプリケーションが他の Web サービスと連携するものである場合、該当のサービスでの認証方法を選択します。参考情報があればテキストボックスに記入します。入力可能な文字数は最大 1,000 文字です。
7. アプリケーションが Salesforce のユーザ資格情報を Force.com 外部に格納するかどうかを指定します。
8. アプリケーションが Salesforce の顧客データを Force.com 外部に格納するかどうかを指定します。
9. アプリケーションが Salesforce の顧客データを Force.com 外部に保存する場合、[アクセスするオブジェクト] ボックスに、該当するすべての Salesforce オブジェクトを入力します。入力可能な文字数は最大 255 文字です。
10. アプリケーションでクライアントコンポーネント（ソフトウェア、プラグインなど）のインストールが必要かどうかを指定します。
11. アプリケーションでクライアントコンポーネントのインストールが必要な場合、[要件] ボックスに詳細を入力します。入力可能な文字数は最大 1,000 文字です。
12. [セキュリティレビューの開始] をクリックすると AppExchange セキュリティレビューが開始されます。記入した内容を破棄して前のページに戻る場合は、[キャンセル] をクリックします。



メモ: レビュー料金のお支払い後、セキュリティチームから、アプリケーションをテストする方法の詳細を確認するための質問事項が送られます。これには、次のセクションで説明するインストールリンクやテストの資格情報などの情報も記載されています。

13. 契約上、この情報は常に最新の状態にしておく必要があります。たとえば、アプリケーションをアップグレードして新しい Web サービスを使用するようになった場合などには、リストティングの編集画面で、[提供] タブの [レビューの編集] をクリックして、セキュリティレビューの提出内容を編集する必要があります。アプリケーションは定期的にレビューされます。

さらに、モバイルアプリケーションの場合は、モバイルデバイスのタイプに応じて以下の情報を提供する必要があります。

- iOS モバイルアプリケーション — アプリケーションが無償で、すでに App Store に公開されている場合は、そのインストールリンクを提供します。アプリケーションがまだ承認されていない場合、または無償でない場合は、AdHoc インストール (デバイスの UDID についてはお問い合わせください) またはアプリケーションの Testflight リンク (UDID は不要) のいずれかを提供します。Testflight の詳細については、<https://testflightapp.com/> を参照してください。Salesforce アカウントのログイン情報以外の資格情報 (連携する外部アプリケーションの資格情報など) がモバイルアプリケーションで必須またはオプションで使用される場合は、それらの情報も提供します。アプリケーションを動作させるためにサンプルデータが必要な場合は、サンプルデータの論理セットを含めてください。
- Android アプリケーション — Android アプリケーションの .APK ファイルおよび対象デバイスの情報を提供します。Salesforce アカウントのログイン情報以外の資格情報 (連携する外部アプリケーションの資格情報など) がモバイルアプリケーションで必須またはオプションで使用される場合は、それらの情報も提供します。アプリケーションを動作させるためにサンプルデータが必要な場合は、サンプルデータの論理セットも含めてください。

第 12 章

リファレンス

トピック:

- REST API リソース
- iOS アーキテクチャ
- Android アーキテクチャ

Mobile SDK 関連のリファレンスドキュメントは GitHub でホストされており、快適に表示、操作できます。

<http://forcedotcom.github.com/SalesforceMobileSDK-iOS>

<http://forcedotcom.github.com/SalesforceMobileSDK-Android>

REST API リソース

Salesforce Mobile SDK では、さまざまなマッパーを作成することで、REST API の利用を簡略化しています。メソッドを呼び出して適切なパラメータを指定すれば、残りの処理は自動的に行われます。以下の表に、利用可能なリソースとその機能を示します。詳細については、『REST API Developer's Guide』を参照してください。

リソース名	URI	説明
Versions (バージョン)	/	現在利用可能な Salesforce の各バージョンに関する概要情報(バージョン、ラベル、各バージョンのルートへのリンクなど)を取得します。
Resources by Version (バージョンに基づくリソース)	/vXX.X/	指定されたバージョンの API で利用可能なリソースのリストを取得します。リストには各リソースの名前と URI が含まれます。
Describe Global (グローバルな記述)	/vXX.X/sobjects/	組織のデータで利用可能なオブジェクト、メタデータのリストを取得します。
SObject Basic Information (Salesforce オブジェクトの基本情報)	/vXX.X/sobjects/ <i>SObject</i> /	指定されたオブジェクトの個々のメタデータを記述します。特定のオブジェクトの新規レコードの作成にも使用できます。
SObject Describe (Salesforce オブジェクトの記述)	/vXX.X/sobjects/ <i>SObject</i> /describe/	指定されたオブジェクトの個々のメタデータを、あらゆるレベルから包括的に記述します。
SObject Rows (Salesforce オブジェクトの行)	/vXX.X/sobjects/ <i>SObject</i> /id/	指定されたオブジェクト ID に基づいてレコードにアクセスし、レコードの取得、更新、削除を実行します。このリソースは、項目値の取得にも使用できます。
SObject Rows by External ID (外部 ID に基づく Salesforce オブジェクトの行)	/vXX.X/sobjects/ <i>SObjectName</i> / <i>fieldName</i> / <i>fieldValue</i>	指定されたの外部 ID 項目の値に基づいて、レコードの新規作成または更新(upsert)を実行します。

リソース名	URI	説明
SObject Blob Retrieve (Salesforce オブジェクトの Blob 項目の取得)	/vXX.X/sobjects/ SObject /id/ blobField	個々のレコードから指定された Blob 項目の値を取得します。
SObject User Password (Salesforce オブジェクトのユーザーパスワード)	/vXX.X/sobjects/User/ userid /password /vXX.X/sobjects/SelfServiceUser/ self service user id /password	ユーザーパスワードの設定、リセット、取得を実行します。
Query (クエリ)	/vXX.X/query/?q= soql	指定された SOQL クエリを実行します。
Search (検索)	/vXX.X/search/?s= sosl	指定された SOSL 検索を実行します。検索文字列は URL エンコードされている必要があります。

iOS アーキテクチャ

Salesforce のネイティブ SDK が現在ユーザに提供している主な機能は次のとおりです。

- OAuth 認証機能
- REST API 通信機能
- 安全性の高い SmartStore ストレージとアプリケーションデータの取得



メモ: この機能は現在ネイティブテンプレートアプリケーションには提供されていませんが、バイナリディストリビューションに含まれています。

Salesforce のネイティブ SDK は基本的には 1 つのライブラリですが、次の追加ライブラリに依存します(また、これらのライブラリに機能を提供します)。

- libRestKit.a — REST API のコールを容易にするサードパーティ製の基盤ライブラリ。
 - ◊ さらに、RestKit は標準の iOS 開発環境の一部である libxml2.dylib に依存します。
- libSalesforceOAuth.a — OAuth 認証を管理するための基盤ライブラリ。
- libsqlite3.dylib — SQLite の機能へのアクセスを提供するライブラリ。これもまた、標準の iOS 開発環境の一部です。
- fmdb — SQLite に関する Objective-C のラッパー。



メモ: この機能は現在ネイティブテンプレートアプリケーションには提供されていませんが、バイナリディストリビューションに含まれています。

ネイティブ iOS オブジェクト

以下のオブジェクトは、Mobile SDK の機能をアプリケーションで活用できるようにする重要なオブジェクトです。

- SFRestAPI
- SFRestAPI(ブロック)
- SFRestRequest

SFRestAPI

SFRestAPI は、REST 要求を行うためのエントリポイントであり、通常は SFRestAPI sharedInstance を介してシングルトンとしてアクセスされます。

次のように、このオブジェクトからさまざまな標準の定型クエリを簡単に作成することができます

```
SFRestRequest* request = [[SFRestAPI sharedInstance]
requestForUpdateWithObjectType:@"Contact"
objectID:contacted
fields:updatedFields];
```

以下のコードで要求を送信できます。

```
[[SFRestAPI sharedInstance] send:request delegate:self];
```

SFRestAPI(ブロック)

SFRestAPI クラスのカテゴリを拡張したもので、コールバックメカニズムとしてブロックを指定することができます。たとえば、次のように指定します。

```
NSMutableDictionary *fields = [NSMutableDictionary
dictionaryWithObjectsAndKeys:
    @"John", @"FirstName",
    @"Doe", @"LastName",
    nil];
[[SFRestAPI sharedInstance] performCreateWithObjectType:@"Contact"
fields:fields
failBlock:^(NSError *e) {
    NSLog(@"Error: %@", e);
}
completeBlock:^(NSDictionary *d) {
    NSLog(@"ID value for object: %@", [d objectForKey:@"id"]);
}];
```

SFRestRequest

SFRestAPI が提供する定型の REST 要求に加えて、次のような要求を独自に作成できます。

```
NSString *path = @"/v23.0";
SFRestRequest* request = [SFRestRequest
requestWithMethod:SFRestMethodGET path:path queryParams:nil];
```

その他のオブジェクト

以下のオブジェクトを直接活用する機会はあまりないと思われますが、SDKにおけるこれらのオブジェクトの役割は覚えておいても損はありません。

- RKRequestDelegateWrapper — SFRestAPI と RestKit ライブラリを仲介します。RKRequestDelegateWrapper は RestKit の通信機能をラップするラッパーです。HTTP POST のタイプの識別、データ変換、応答の解釈などを行うための便利なメソッドを提供します。
- SFSessionRefresher — SFRestAPI と密接に連携します。セッションの有効期限切れが原因で REST 要求が失敗した場合に、自動的にセッションを更新する機能を集めたラッパーです。

Android アーキテクチャ

SalesforceSDK は、Java クラスの JAR ファイルとして提供され、native/Salesforce ディレクトリ内のライブラリとリソースファイルのセットと連携します。

Java コード

Java ソースは /src ディレクトリに格納されています。

Java コード

パッケージ名	説明
com.salesforce.androidsdk.app	SDK アプリケーションクラス (ForceApp)
com.salesforce.androidsdk.auth	OAuth サポートクラス
com.salesforce.androidsdk.phonegap	Salesforce Mobile SDK PhoneGap プラグインのネイティブ実装
com.salesforce.androidsdk.rest	REST の要求およびレスポンスで使用されるクラス
com.salesforce.androidsdk.security	セキュリティ関連のヘルパークラス (パスコードマネージャなど)
com.salesforce.androidsdk.store	SmartStore およびサポートクラス
com.salesforce.androidsdk.ui	活動 (ログインなど)
com.salesforce.androidsdk.util	その他のユーティリティクラス

com.salesforce.androidsdk.app

クラス	説明
ForceApp	アプリケーションの抽象サブクラス。 プロジェクトでは具象サブクラスを指定する必要があります

com.salesforce.androidsdk.auth

クラス	説明
AuthenticatorService	認証を管理するサービス
HttpAccess	汎用的な HTTP アクセスレイヤ
OAuth2	共通の OAuth2 要求のヘルパークラス

com.salesforce.androidsdk.phonegap

クラス	説明
SalesforceOAuthPlugin	Salesforce OAuth 用 PhoneGap プラグイン
SmartStorePlugin	SmartStore 用 PhoneGap プラグイン
TestRunnerPlugin	コンテナ内で Javascript のテストを実行する PhoneGap プラグイン

com.salesforce.androidsdk.rest

クラス	説明
ClientManager	RestClient のファクトリ。必要に応じてログインフローを開始
RestClient	Force.com サーバと通信することを承認されたクライアント
RestRequest	Force.com REST 要求ラッパー
RestResponse	REST レスポンスラッパー

com.salesforce.androidsdk.security

クラス	説明
Encryptor	暗号化、復号化、ハッシュ計算用のヘルパークラス
PasscodeManager	非活動タイムアウトマネージャ。必要に応じてパスコード画面を表示します

com.salesforce.androidsdk.store

クラス	説明
Database	暗号化された、または通常の SQLite データベースラッパー

クラス	説明
DBOpenHelper	通常のデータベースの作成とバージョン管理を管理するヘルパークラス
DBOperations	DBOpenHelper、EncryptedDBOpenHelper のラッパー
EncryptedDBOpenHelper	暗号化データベースの作成とバージョン管理を管理するヘルパークラス
SmartStore	JSON ドキュメントを格納する安全で検索可能なストレージ

com.salesforce.androidsdk.ui

クラス	説明
CustomServerUrlEditor	ユーザが別のログインホストを選択できるカスタムダイアログ
LoginActivity	ログイン画面
OAuthWebviewHelper	Oauth ログインプロセス中の WebView インスタンスを管理するヘルパークラス
PasscodeActivity	パスコード (PIN) 画面
SalesforceDroidGapActivity	ハイブリッドアプリケーションの主な活動
SalesforceGapViewClient	ハイブリッドアプリケーションで使用される WebView クライアント
SalesforceR	SDK の外部で定義されたリソースへの参照を可能にするクラス
ServerPickerActivity	ログインホスト画面の選択

com.salesforce.androidsdk.util

クラス	説明
EventsObservable	SDK によって生成されたイベントの登録と受信に使用されるクラス（主にテストで使用）
EventsObserver	SDK イベントのオブザーバ
UriFragmentParser	URI のクエリ文字列を解析するためのヘルパークラス

ライブラリ

ライブラリは /libs ディレクトリに格納されています。

ライブラリ名	説明
phonegap-1.2.0.jar	オープンソースのモバイル開発フレームワーク。ハイブリッドアプリケーションに使用される (*)
sqlcipher.jar	256ビット AES によるデータベースファイルの透過的な暗号化を提供する SQLite のオープンソース拡張ライブラリ (**)
armeabi/*.so	sqlcipher の使用に必要なネイティブライブラリ (**)
commons-code.jar, guava-r09.jar	sqlcipher の使用に必要な Java ライブラリ

(*) はハイブリッドアプリケーションに必須のファイルです。

(**) は SmartStore に必須のファイルです。

リソース

リソースは /res ディレクトリに格納されています。

drawable-hdpi, drawable-ldpi

ファイル	用途
edit_icon.png	サーバの選択画面
glare.png	ログイン画面
icon.png	アプリケーションのアイコン

drawable

ファイル	用途
header_bg.png	ログイン画面
progress_spinner.xml	ログイン画面
toolbar_background.xml	ログイン画面

drawable-xlarge, drawable-xlarge-port

ファイル	用途
header_bg.png	ログイン画面(タブレット)
header_drop_shadow.xml	ログイン画面(タブレット)
header_left_border.xml	ログイン画面(タブレット)
header_refresh.png	ログイン画面(タブレット)
header_refresh_press.png	ログイン画面(タブレット)
header_refresh_states.xml	ログイン画面(タブレット)
header_right_border.xml	ログイン画面(タブレット)
login_content_header.xml	ログイン画面(タブレット)
nav_shadow.png	ログイン画面(タブレット)
oauth_background.png	ログイン画面(タブレット)
oauth_container_dropshadow.9.png	ログイン画面(タブレット)
oauth_background.png	ログイン画面(タブレット)
progress_spinner.xml	ログイン画面(タブレット)
refresh_loader.png	ログイン画面(タブレット)
toolbar_background.xml	ログイン画面(タブレット)

layout

ファイル	用途
custom_server_url.xml	サーバの選択画面
login.xml	ログイン画面
passcode.xml	PIN 画面
server_picker.xml	サーバの選択画面

layout-xlarge

ファイル	用途
login_header.xml	ログイン画面(タブレット)
login.xml	ログイン画面(タブレット)
server_picker_header.xml	サーバの選択画面(タブレット)
server_picker.xml	サーバの選択画面(タブレット)

menu

ファイル	用途
clear_custom_url.xml	接続の追加ダイアログ
login.xml	ログインメニュー(スマートフォン)

values

ファイル	用途
sdk.xml	ログイン、サーバの選択、PIN の各画面用 のローカライズされた文字列
strings.xml	その他の文字列(アプリケーション名)

values-xlarge

ファイル	用途
styles.xml	スタイル(タブレット)

xml

ファイル	用途
authenticator.xml	アプリケーションが使用するアカウントの 設定
plugins.xml	PhoneGap 用のプラグイン設定ファイル。 ハイブリッドアプリケーションに必須

用語集

A

Apex

Apex は、開発者が Force.com プラットフォームサーバでフローとランザクションの制御ステートメントを Force.com API に対するコードと組み合わせて実行できるようにした、強く型付けされたオブジェクト指向のプログラミング言語です。Java に似た構文を使い、データベースのストアドプロシージャのように動作する Apex を使用して、開発者は、ボタンクリック、関連レコードの更新、および Visualforce ページなどのほとんどのシステムイベントに対しビジネスロジックを追加できます。Apex コードは、Web サービス要求、およびオブジェクトのトリガから開始できます。

Apex コントローラ

「コントローラー — Visualforce」を参照してください。

AppExchange

AppExchange はセールスフォース・ドットコムの共有インターフェースであり、Force.com プラットフォームのアプリケーションやサービスを参照および共有できます。

AppExchange のアップグレード

アプリケーションのアップグレードは、新しいバージョンをインストールするプロセスです。

AppExchange のセキュリティレビュー

顧客がアプリケーションを安全にインストールできることを確認する検査。アプリケーションを AppExchange 上で公開するには、このセキュリティレビューに合格する必要があります。プロバイダは、アプリケーションの更新があるたびにアプリケーションを提出してセキュリティレビューを受けることを義務付けられます。

AppExchange のリスティング

AppExchange で公開されているアプリケーションやサービスに関する説明。アプリケーションやサービスを AppExchange コミュニティに売り込む際の主要なマーケティングツールになります。

AppExchange 配信組織

APO (AppExchange Publishing Organization) と略されることもあります。セールスフォース・ドットコムのパートナーが AppExchange 上でリストティングを公開するために使用するマスター組織。アプリケーションの開発を行う子組織は APO にリンクさせることができます。リストティングを単一のプロバイダエンティティの下に集約することで、顧客に一貫性のあるメッセージを提供できます。

C

Chatter Mobile

モバイルデバイスを使って Chatter でコラボレーションを行えるようにする無料のモバイルアプリケーション。Chatter で投稿やコメントの追加を行ったり、フォローする他のユーザ、レコード、ファイルやグループに関する更新情報を受け取ることができます。

Chatter フィード

Salesforce の最近の活動のリスト。Chatter フィードは、次の場所に表示されます。

- [Chatter] タブまたは [ホーム] タブ。自分の投稿、フォローしている人からの投稿、フォローしているレコードの更新、および自分がメンバーになっているグループへの投稿が表示されます。
- プロファイル。参照しているプロファイルのユーザが行った投稿が表示されます。
- レコード。参照中のレコードへの更新を確認できます。
- Chatter グループ。参照しているグループへの投稿が表示されます。

D

Developer Edition

開発者が Force.com プラットフォームを使用して拡張、統合、開発を行えるよう設計された無料でフル機能の Salesforce。Developer Edition のアカウントは、developer.force.com で登録できます。

Developer Force

Developer Force Web サイト (developer.force.com) では、サンプルコード、ツールキット、オンライン開発者コミュニティなど、プラットフォーム開発者向けの幅広いリソースを提供しています。開発向けの Force.com プラットフォーム環境も、ここから入手できます。

E

Enterprise Edition

より大規模で、複雑な構造を持つ企業向けに設計された Salesforce のエディション。

Enterprise WSDL

顧客が Salesforce 組織のみでインテグレーションを構築する場合や、パートナーが Tibco、webMethods などのツールを使って強い型付けが必要なインテグレーションを構築する場合に使用する強く型付けされた WSDL。Enterprise WSDL の欠点は、組織のデータモデルに存在するすべての一意のオブジェクトおよび項目にバインドされているため、1 つの Salesforce 組織のスキーマしか扱えないという点です。

F

Force.com

アプリケーションを構築するための Salesforce プラットフォーム。強力なユーザインターフェース、オペレーティングシステムおよびデータベースを組み合わせて設計された Force.com を使用することで、クラウドアプリケーションを全社的に展開し、クラウド上でカスタマイズできます。

Force.com IDE

開発者が Eclipse 開発環境で Force.com アプリケーションを管理、作成、デバッグおよびリリースできる Eclipse プラグイン。

Force.com アプリケーションメニュー

カスタマイズ可能なアプリケーションをワンクリックで切り替えることができるメニュー。Force.com アプリケーションメニューは、Salesforce ユーザインターフェースの各ページの上部に表示されます。

G

Group Edition

ユーザ数が限られた小規模企業やワークグループ用に設計された Salesforce のエディション。

I

IdeaExchange

セールスフォース・ドットコムのユーザが新しい商品のコンセプトを提案したり、お気に入りの拡張機能を勧めたり、製品マネージャや他のユーザと対話したり、今後のリリースが予定されるセールスフォース・ドットコム製品のレビューを行ったりすることができるフォーラム。ideas.salesforce.com からアクセスできます。

M

MVC (Model-View-Controller)

アプリケーションをデータを示すコンポーネントに分割する設計パラダイム (モデル)、ユーザインターフェースでデータを表示する手

段（ビュー）、およびビジネスロジックデータを使用してデータを処理する手段（コントローラ）。

R

REST API

HTTP の GET、POST、PUT メソッドを使用してサーバ上のリソースを更新する、シンプルな軽量 API。

S

Salesforce SOA (サービス指向アーキテクチャ)

Apex 内から外部 Web サービスへのコールを実行できる Force.com の強力な機能。

Sandbox 組織

Salesforce 本番組織のほぼ同一コピー。テストやトレーニングなど様々な目的のために、本番組織のデータとアプリケーションに影響を与えることなく、複数の Sandbox をそれぞれの環境に作成できます。

SOAP (Simple Object Access Protocol)

XML 符号化データを渡す一定の方法を定義するプロトコル。

SOQL (Salesforce Object Query Language: Salesforce オブジェクトクエリ言語)

単純で強力なクエリ文字列を構築し、Force.com データベースからデータを選択するための検索条件を指定できるクエリ言語。

SOSL (Salesforce Object Search Language: Salesforce オブジェクト検索言語)

Force.com API を使用して、テキストベースの検索を実行できるクエリ言語。

T

Test メソッド

特定のコードが適切に動作しているかを確認する Apex クラスメソッド。Test メソッドは引数を取らず、データをデータベースにコミットしません。また、コマンドラインまたは Force.com IDE のような Apex IDE で runTests() システムメソッドによって実行できます。

U

Unlimited Edition

セールスフォース・ドットコムの最上位ソリューション。CRM のメリットを最大化し、Force.com プラットフォームを通して企業全体の成功に貢献します。

URL (Uniform Resource Locator)

Web サイトやドキュメントなどのインターネット上のリソースのグローバルアドレス(例: <http://www.salesforce.com>)。

V

Visualforce

開発者が、プラットフォームに作成されたアプリケーションのカスタムページおよびコンポーネントを容易に定義できる、シンプルなタグベースのマークアップ言語。各タグが、ページのセクション、関連リスト、または項目など、大小さまざまなコンポーネントに対応しています。標準の Salesforce ページと同じロジックを使用してコンポーネントを制御できます。また、開発者が独自のロジックを Apex で記述されたコントローラと関連付けることもできます。

W

Web サービス

異なるプラットフォームで実行される場合や、異なる言語で記述された場合、またはお互い地理的に離れている場合であっても、2 つのアプリケーションがインターネットを経由してデータを容易に交換できるメカニズム。

Web サービス API

Salesforce 組織の情報へのアクセスを提供する Web サービスアプリケーションプログラミングインターフェース。

WebService メソッド

サードパーティのアプリケーションのマッシュアップなど、外部システムによって使用できる Apex クラスメソッドまたは変数。Web サービスマソッドは、グローバルクラスで定義する必要があります。

WSDL (Web Services Description Language) ファイル

Web サービスと送受信するメッセージの形式を説明する XML ファイル。開発環境の SOAP クライアントは、Salesforce Enterprise WSDL または Partner WSDL を使用して、SOAP API で Salesforce と通信します。

X

XML (Extensible Markup Language: 拡張可能マークアップ言語)

構造化データの共有と移動を可能にするマークアップ言語。メタデータ API を使用して取得またはリリースされるすべての Force.com コンポーネントは、XML 定義に従って記述されます。

あ

アクセストークン

保護されたリソースへのユーザのアクセスを許可するため、ユーザの Salesforce ログイン情報の代わりに、コンシューマにより使用される値。アクセストークンはセッション ID であり、直接使用できます。

アプリケーション

「app」と略されることもあります。特定のビジネス要件を扱うタブ、レポート、ダッシュボードおよび Visualforce ページなどのコンポーネントの集合。Salesforce では、セールスおよびコールセンターなどの標準アプリケーションを提供しています。これらの標準アプリケーションはニーズに合わせてカスタマイズできます。また、アプリケーションをパッケージ化して、カスタム項目、カスタムタブ、カスタムオブジェクトなどの関連コンポーネントと共に AppExchange にアップロードできます。そのアプリケーションを AppExchange から他の Salesforce ユーザが利用できるようにすることもできます。

アプリケーションプログラミンインターフェース (API)

コンピュータシステム、ライブラリ、またはアプリケーションが提供するインターフェースであり、その他のコンピュータプログラムによるサービスの要求やデータの交換に使用されます。

い

インスタンス

組織のデータをホストし、アプリケーションを実行する単一の論理サーバとして機能するソフトウェアおよびハードウェアのクラスタ。Force.com プラットフォームは複数のインスタンスで稼動しますが、1 つの組織のデータは常に 1 つのインスタンスで一元管理されています。

インポートウィザード

Salesforce 組織にデータをインポートするツール。[設定] からアクセスできます。

お

オブジェクト

Salesforce 組織に情報を保存するために使用される単位。オブジェクトは、保存する情報の種類の全体的な定義です。たとえば、ケースオブジェクトでは、顧客からの問い合わせに関する情報を保存できます。組織では、各オブジェクトに複数のレコードが含まれ、各レコードには、その種類のデータの特定のインスタンスに関する情報が格納されます。たとえば、ケースオブジェクトでは、佐藤次郎

さんから寄せられたトレーニングに関する問い合わせに関する情報を保存するケースレコードや、山田花子さんから寄せられた設定の問題に関する情報を保存するケースレコードが保存されます。

か

開発環境

本番組織のユーザに影響を与えることなく設定変更を行える Salesforce 組織。開発環境には、Sandbox 組織と Developer Edition 組織の 2 種類があります。

カスタムオブジェクト

組織固有の情報を保存できるカスタムレコード。

カスタム項目

組織の必要に応じて Salesforce をカスタマイズするために標準項目の他に追加できる項目。

活動 — Chatter

個々のユーザの Chatter での活動を示す指標。Chatter 活動統計では、ユーザが Chatter で行った投稿およびコメントの数と、受信したコメントと「いいね!」の数が示されます。

管理者 (システム管理者)

アプリケーションの設定およびカスタマイズができる組織内の 1 人以上のユーザ。システム管理者プロファイルに割り当てられているユーザは、管理者権限を持ちます。

管理パッケージ

ユニットとして AppExchange に投稿され、名前空間とライセンス管理組織に関連付けられるアプリケーションコンポーネントの集合。アップグレードをサポートするには、管理パッケージであることが必要です。組織は、他の多くの組織でダウンロードおよびインストールできる単一の管理パッケージを作成できます。管理パッケージは、未管理パッケージとは異なり、コンポーネントの一部がロックされており、後でアップグレードできます。未管理パッケージには、ロックされたコンポーネントは含まれておらず、アップグレードはできません。また、管理パッケージでは、開発者の知的財産保護のため、登録組織では特定のコンポーネント (Apex など) が隠されます。

き

共有

ユーザが、他のユーザが所有する情報を参照したり編集したりできるようにする機能。データの共有にはさまざまな方法があります。

- ・ 共有モデル: ユーザがお互いの情報に対して持つデフォルトの組織共有のアクセスレベルや、データへのアクセス権を決定する場合に階層を使用するかどうかを定義します。
- ・ ロール階層: 組織の共有モデル設定に関係なく、上位レベルのユーザが、ロール階層で下位のユーザが所有または共有する情報を表示および編集できるよう、ユーザの様々なレベルを定義します。
- ・ 共有ルール: 管理者が指定したグループまたはロール内のユーザが作成したすべての情報が自動的に別のグループまたはロールのメンバーに共有されるようにします。
- ・ 共有の直接設定: 各ユーザが他のユーザまたはグループとレコードを共有できるようにします。
- ・ Apex による共有管理: 開発者がプログラムを使用して、アプリケーションの動作をサポートする共有を操作できるようにします。

共有モデル

ユーザの様々な種類のレコードに対するデフォルトのアクセス権を指定する、管理者が定義する動作。



クライアントアプリケーション

Salesforce ユーザインターフェースの外部で実行し、Force.com API または Bulk API のみを使用するアプリケーション。通常、デスクトップまたはモバイルデバイス上で稼動します。これらのアプリケーションは、プラットフォームをデータソースとして扱い、設計されたツールおよびプラットフォームの開発モデルを使用します。

クラウドコンピューティング

インターネットをベースとした、ソフトウェアの開発および配布のモデル。データも含めたサービスの技術インフラストラクチャが、インターネット上でホストされます。こうすることにより、利用者は、ハードウェア、ソフトウェア、メンテナンスなどに投資しなくても、プラウザその他のシンクライアントを使用して、サービスの開発や利用を行えます。

クラス — Apex

Apex オブジェクトの作成でベースとして使用する一種のテンプレート。他のクラス、ユーザ定義メソッド、変数、例外型、および static 初期設定化コードで構成されます。多くの場合、Apex クラスは、対応する Java クラスに類似しています。

グループ

一連のユーザから成るグループ。グループには、個人ユーザ、他のグループ、特定のロールに属するユーザが含まれます。Connect for Outlook または Connect for Lotus Notes を使用する場合、グループを使

用してデータへの共有アクセスを定義したり、同期するデータを指定したりできます。

ユーザは自分の個人グループを定義できます。管理者は、組織内の全員が使用する公開グループを作成できます。

け

権限

Salesforce で特定の機能をユーザが実行するために必要とされる設定。権限は、権限セットとプロファイルで有効にできます。たとえば、カスタムオブジェクトの「編集」権限や「すべてのデータの編集」権限などがあります。

権限セット

ユーザに特定のツールと機能へのアクセスを提供する一連の権限と設定。

こ

更新トークン

新しいアクセストークンを取得するためにコンシューマが使用するトークン。エンドユーザがアクセスを再度承認する必要があります。

項目

テキストまたは通貨の値など、情報の特定の部分を保持するオブジェクトの構成要素。

項目セット

項目をグループ化したもの。たとえば、あるユーザの名、ミドルネーム、姓、役職を示す項目を 1 つの項目セットにすることができます。項目セットは、Visualforce ページで動的に参照できます。そのページを管理パッケージに追加すれば、システム管理者は項目セット内の項目の追加、削除、並び替えを行って、コードを変更せずに Visualforce ページ上に表示する項目を変更できます。

項目の運動関係

別の項目の値に基づいて、選択リストの内容を変更できるフィルタ。

項目レベルセキュリティ

項目に対するユーザのアクセス (非表示、表示、参照のみ、または編集可能) を決定する設定。使用可能なエディションは、Enterprise Edition、Unlimited Edition、および Developer Edition です。

子リレーション

別の sObject を一对多リレーションの「一」側として参照する sObject に定義されたリレーション。たとえば、取引先責任者、商談および行動には取引先との子リレーションが定義されています。

コンシューマ鍵

Salesforce で、コンシューマを識別するために使用される値。client_id として参照されます。

コントローラ — Visualforce

Visualforce ページに実行する必要のあるデータおよびビジネスロジックを提供する Apex クラス。Visualforce ページは、デフォルトですべての標準オブジェクトまたはカスタムオブジェクトに付属する標準コントローラのほか、カスタムコントローラを使用できます。

コンポーネント — Visualforce

<apex:detail> などの一連のタグを使用して Visualforce ページに追加できる要素。Visualforce には、多くの標準コンポーネントが含まれていますが、独自のカスタムコンポーネントを作成することもできます。

コンポーネントの参照ライブラリ (Component Reference) — Visualforce

組織で使用できる Visualforce の標準コンポーネントおよびカスタムコンポーネントの説明。Visualforce ページの開発フッターまたは『Visualforce 開発者ガイド』からアクセスできます。

し

システムログ

開発者コンソールの一部であり、コードスニペットのデバッグに使用できる独立したウィンドウ。ウィンドウの下部にテストするコードを入力して、[Execute (実行)] をクリックします。システムログの本文には、実行する行の長さや、作成されたデータベースコール数などのシステムリソース情報が表示されます。コードが完了しなかった場合は、コンソールにデバッグ情報が表示されます。

せ

セッション ID

ユーザが Salesforce に正常にログインした場合に返される認証トークン。セッション ID を使用すると、ユーザが Salesforce で別のアクションを実行するときに毎回ログインする必要がなくなります。Salesforce レコードの一意の ID を示すレコード ID または Salesforce ID とは異なります。

セッションタイムアウト

ログインしてからユーザが自動的にログアウトするまでの時間です。セッションは、前もって決定された非活動状態の期間の後、自動的に終了します。非活動状態の期間の長さは、あなたの名前 > [設定] > [セキュリティのコントロール] をクリックすることによって Salesforce で設定できます。デフォルト値は 120 分 (2 時間) です。ユーザが Web インターフェースでアクションを実行するか、API コールを作成すると、非活動状態タイマーが 0 にリセットされます。

設定

Force.com アプリケーションをカスタマイズおよび定義できる管理領域。[設定] には、Salesforce ページの上部にある、あなたの名前 > [設定] リンクからアクセスできます。

そ

組織

定義済みのライセンスユーザセットを提供する Salesforce のリリース。組織は、セールスフォース・ドットコムの各お客様に提供される仮想スペースです。組織には、すべてのデータおよびアプリケーションが含まれており、他のすべての組織から独立しています。

組織の共有設定

ユーザが組織で持つデータアクセスのベースラインレベルを指定できる設定。たとえば、組織の共有設定として、「オブジェクト権限によって有効化されている特定のオブジェクトの任意のレコードを参照できるが、編集するには別の権限が必要となる」というデフォルト設定を指定できます。

た

単体テスト

テスト可能なアプリケーションの最小単位 (多くの場合、1 つのメソッド) に対するテスト。単体テストは、コードの一部が正常に動作することを検証するために実施します。「Test メソッド」も参照してください。

て

データベース

構造化された情報の集合。Force.com プラットフォームの基盤となるアーキテクチャには、データを格納するデータベースが含まれています。

と**統合ユーザ**

クライアントアプリケーションまたは統合のみに定義された Salesforce ユーザ。SOAP API では「ログインユーザ」とも呼ばれます。

動的 Visualforce バインド

レコードに関する情報を表示する際に必ずしも表示する項目を指定しない汎用的な Visualforce ページを記述する方法。つまり、ページ上に表示する項目はコンパイル時ではなく実行時に決定されます。

トランスレーションワークベンチ

トランスレーションワークベンチを使用して、翻訳する言語を指定し、翻訳者を言語に割り当て、Salesforce 組織に作成したカスタマイズ機能の翻訳を作成し、管理対象パッケージから表示ラベルと翻訳を上書きすることができます。カスタム選択リスト値からカスタム項目にいたるすべてが翻訳され、海外のユーザが Salesforce のすべての機能を自分の母語で使用できるようになります。

トリガ

データベースの特定の種別のレコードが挿入、更新、または削除される前後で実行する Apex スクリプトです。各トリガは、トリガが実行されたレコードへのアクセスを提供する一連のコンテキスト変数で実行し、すべてのトリガは一括モードで実行します。つまり、一度に 1 つずつレコードを処理するのではなく、複数のレコードを一度に処理します。

取引先

ユーザが状況を把握したい組織、会社、または消費者。たとえば、顧客、パートナー、競合会社など。

な**名前空間**

パッケージコンテキストで、ドメイン名と同様、AppExchange にある自社パッケージとその内容を他の開発者のパッケージと区別するために使用される、1 ~ 15 文字の英数字で構成される識別子。Salesforce では、Salesforce 組織のすべての一意のコンポーネント名に自動的に名前空間接頭辞とそれに続く 2 つのアンダースコア (_) を追加します。

に

認証コード

エンドユーザによって付与されるアクセスを表す、有効期間の短いトークン。認証コードは、アクセストークンと更新トークンを取得するために使用されます。

は

パッケージ

AppExchange を介して他の組織に提供される Force.com のコンポーネントおよびアプリケーションのグループ。AppExchange にまとめてアップロードできるように、パッケージを使用してアプリケーションおよび関連するコンポーネントをバンドルします。

ひ

標準オブジェクト

Force.com プラットフォームに含まれる組み込みオブジェクト。アプリケーション独自の情報を格納するカスタムオブジェクトを作成することもできます。

ふ

フィードフィルタ — Chatter

フィードフィルタを使用して、[Chatter] タブの Chatter フィードに投稿のサブセットを表示できます。

フィード追跡 — Chatter

Chatter フィードでフォローできるレコードおよび追跡可能な項目を指定する管理者設定。オブジェクトのフィード追跡を有効化すると、そのオブジェクトのレコードをフォローできるようになります。項目のフィード追跡を有効にすると、フォローしているレコードでそれらの項目が変更された場合に、Chatter フィード上で確認できます。

フィード添付 — Chatter

Chatter フィードでの投稿に添付されたファイルまたはリンク。

プロファイル

Salesforce 内でユーザがさまざまな機能を実行するための権限を定義します。たとえば、ソリューション管理者プロファイルでは、ソリューションを作成、編集、および削除するためのアクセス権がユーザに付与されます。

ほ

本番組織

本番データとそれらにアクセスする実際のユーザを持つ Salesforce 組織。

み

未管理パッケージ

開発者がアップグレードまたは制御できないパッケージ。

め

メタデータ

組織および組織のいずれかの部分の構造、外観、機能に関する情報。 Force.com では、メタデータを記述するのに XML を使用します。

メタデータ WSDL

Force.com Metadata API コールを使用するための WSDL。

メタデータベースの開発

アプリケーションを宣言的な「設計図」として定義できるアプリケーション開発モデル。コードは必要ありません。プラットフォームに構築されたアプリケーション、およびそれらのデータモデル、オブジェクト、フォーム、ワークフローなどはメタデータで定義されます。

り

リモートアクセスアプリケーション

OAuth プロトコルを使用して Salesforce ユーザとアプリケーションの両方を検証する、Salesforce の外部のアプリケーション。

れ

レコード

Salesforce オブジェクトを構成する各インスタンス。たとえば、「John Jones」は取引先責任者レコードの名前となります。

レコードレベルセキュリティ

データを制御するメソッドの 1 つ。特定のユーザにオブジェクトの参照および編集を許可し、ユーザが参照できるレコードを制限できます。

連動項目

対応する制御項目で選択された値に基づいて使用可能な値が表示される、カスタムの選択リスト項目または複数選択の選択リスト項目。

索引

Force.com for Touch 3

A

Android – NFC 123
 Android – アーキテクチャ 141, 144
 Android – 開発 51, 54–55
 Android – サンプル 75
 Android – サンプルアプリケーション 54, 55
 Android – ハイブリッドサンプルアプリケーション 75
 Android – ハイブリッドプロジェクト 59
 Android – プロジェクト 53
 Android – 要件 18, 52
 Ant バージョン 58
 API, ソーシャル 69
 App コンポーネント 78–79
 AppExchange 129–135
 Apple Safari 20

B

Blob 114, 138

C

Chatter 69
 Chrome ブラウザ 20
 Content コンポーネント 78, 80

D

Database.com 18–19
 Describe Global (グローバルな記述) 138
 Detail コンポーネント 78, 81
 Developer Edition 18
 developer.force.com 18
 Dreamforce アプリケーション 5

F

Firefox ブラウザ 20
 Footer コンポーネント 78, 80–81
 Force.com 18

G

GitHub 7
 Google Chrome 20

H

Header コンポーネント 78, 80
 HTML アプリケーションの開発 97
 HTML5 97–98
 HTML5 アプリケーションの開発 98
 HTML5 開発 10, 14, 21, 113

I

ID URL 27
 Internet Explorer 20
 iOS – Xcode テンプレート 49
 iOS – アーキテクチャ 48, 52, 139
 iOS – 開発 47
 iOS – サンプルアプリケーション 49
 iOS – ハイブリッドサンプルアプリケーション 75
 iOS – ハイブリッドプロジェクト 59
 iOS – 要件 18
 iOS アプリケーション, 作成 48, 49
 IP 範囲 35–36
 ISV 131–133

J

JavaScript 57, 98

L

List コンポーネント 78, 81
 localStorage 112

M

Mobile Components for Visualforce 77–78, 81–83, 86
 Mobile Components for Visualforce のアーキテクチャ 78
 Mobile SDK 3
 Mobile SDK Workbook 21
 Mobile SDK のインストール 48, 53
 Mobile SDK リポジトリ 7
 MobileComponents.git 81–83
 MockSmartStore 102
 Mozilla Firefox 20

N

Navigation コンポーネント 78, 79
 NFC 123

O

OAuth 24, 26
 OAuth トークン, 取り消し 32
 OAuth2 23–24

P

Page コンポーネント 78, 80
 PIN コード保護 45

Q

QR スキャニング 119
 Query (クエリ) 138
 querySpec 104, 106, 108–109

R

registerSoup 104, 106, 108–109
 Responsive デザイン 5, 14
 REST 138
 REST リソース 138
 RestAPIExplorer 50

S

Safari ブラウザ 20
 Salesforce Touch Platform 3, 9
 scope パラメータ 26
 SDK のインストール 48, 53
 SDK の要件 18
 Search (検索) 138
 SmartStore 101–102
 SmartStore の拡張 102–103, 112
 SmartStore の関数 104, 106, 108–109
 SObject Basic Information (Salesforce オブジェクトの基本情報) 138
 Soup 104, 106, 108–109
 Soup のクエリ 104, 106, 108–109
 Soup の削除 104, 106, 108–109
 SplitView テンプレート 78, 79

U

upsertSoupEntries 104, 106, 108–109
 URL, ID 27

V

Versions (バージョン) 138
 Visualforce 77–78
 Visualforce App コンポーネント 79
 Visualforce Content コンポーネント 80
 Visualforce Detail コンポーネント 81
 Visualforce Footer コンポーネント 81
 Visualforce Header コンポーネント 80
 Visualforce List コンポーネント 81
 Visualforce Navigation コンポーネント 79
 Visualforce Page コンポーネント 80
 Visualforce SplitView テンプレート 79
 Visualforce コンポーネント 77–78
 Visualforce コンポーネントのインストール 81
 Visualforce コンポーネントの作成 86
 Visualforce タブレットページの作成 83
 Visualforce のアーキテクチャ 78
 Visualforce を使用したタブレットページ 83

W

Warehouse スキーマ 63, 67
 Workbook, Mobile SDK 21

X

Xcode プロジェクト 48, 49
Xcode プロジェクトテンプレート 49

あ

アーキテクチャ, Android 141, 144
アイデンティティサービス 3
アプリケーションの配信 129
安全なストレージ 101

い

位置情報 122
位置情報サービス 3
一覧情報を表示するページ 63

え

エンタープライズ ID 3

お

オープンソースのモバイルコンポーネント
77–78
オブジェクトのリストを取得 138
オフラインストレージ 101–102
オフラインのデータキャッシュ 101

か

開発 18
開発, Android 51, 54–55
開発, ハイブリッド 57
開発環境 18
開発シナリオ 10
開発を始める 21
各開発シナリオの比較 10
カメラ 114

き

近距離無線通信 123

く

クリックスタート 21
クリックスタート, ハイブリッド 58
クライアントサイドの検出 14

こ

ご意見、ご提案 6
更新トークンフロー 26
コールバック URL 33
このドキュメントについて 6
このドキュメントのオンライン版 6
このドキュメントの構成 6
このドキュメントのバージョン 6
コラボレーション 69
コンシューマ鍵 33
コンシューマの秘密 33
コンテナ 57

さ

サーバサイドの検出 14
在庫管理 63, 67
最終改定日 6
最新情報 7
最新リリースの情報 7
サインアップ 18–19
座標 122
サポートされるプラウザ 20
サンプル iOS アプリケーション 49
サンプルアプリケーション, Android 54, 55
サンプルアプリケーション, iOS 50
サンプルハイブリッドアプリケーション 59

し

上級レベルのトピック 113
詳細ページ 67
章の構成 6

す

スキャニング 119
ストア 129–130

せ

セールスフォース・ドットコム社内での
Salesforce Touch Platform の活用 5
セキュリティ 23
セキュリティ, PIN 45
セキュリティレビュー 134
接続済みアプリケーション 23, 35–36
接続済みアプリケーション – IP 制限 42
接続済みアプリケーション – IP 範囲 39
接続済みアプリケーション – IP 範囲のホワイ
トリストの指定 39
接続済みアプリケーション – Salesforce との
統合 38
接続済みアプリケーション – アクセス制御 42
接続済みアプリケーション – アップグレード
46
接続済みアプリケーション – アンインストー
ル 45
接続済みアプリケーション – インストール 41
接続済みアプリケーション – エラー 46
接続済みアプリケーション – 開始 URL 42
接続済みアプリケーション – 開発者の作業 36
接続済みアプリケーション – 管理 42
接続済みアプリケーション – 管理者の作業 41
接続済みアプリケーション – 基本情報の指定
37
接続済みアプリケーション – 公開 40
接続済みアプリケーション – 更新 41
接続済みアプリケーション – コールバック
URL 38
接続済みアプリケーション – 削除 41
接続済みアプリケーション – 作成 36
接続済みアプリケーション – バージョン管理
41
接続済みアプリケーション – モバイルアプリ
ケーションとの統合 39
接続済みアプリケーション – 連絡先情報 37
接続済みアプリケーション – ロゴ画像 37
前提条件 18

そ

ソーシャル API 69
ソーシャルコラボレーション 69
ソースコード 7

た

対象読者 6

ち

地理位置情報 3, 122
て
ディスカッションボード 7
データの格納 102–103
デバイスのカメラ 114
デバイスのネイティブ機能の利用 113

と

トークンの取り消し 32

に

認証 23, 45
認証フロー 24

ね

ネイティブ iOS アーキテクチャ 48, 52, 139
ネイティブ iOS アプリケーション 48, 49
ネイティブ iOS アプリケーションの開発 47
ネイティブ iOS プロジェクトテンプレート 49
ネイティブ開発 10, 14, 21, 113

は

バーコードのスキヤニング 119
バージョン 6–7
パートナープログラム 131–133
ハイブリッド – クイックスタート 58
ハイブリッド – Android 向けプロジェクト 59
ハイブリッド – iOS サンプル 75
ハイブリッド – iOS 向けプロジェクト 59
ハイブリッド – Visualforce 開発 77–78, 81–
83, 86
ハイブリッド – オフライン開発 102
ハイブリッド – 開発 10, 14, 21, 57–59, 63, 67,
113–114
ハイブリッド – サンプルアプリケーション 59
ハイブリッド – 要件 58
はじめに 1
パスワード 138
パラメータ, scope 26

ふ

- ファイルの格納 112
- フィード 69
- フィードバック 6
- フィードバックの送付 6
- プラウザの互換性 20
- フロー 24, 26
- ブログ 7
- プロジェクト, Android 53

ま

- マルチデバイス戦略 14

め

- メタデータ 138

も

- モバイル開発 9, 48
- モバイルコンテナ 3, 48, 57
- モバイルと PC の比較 1
- モバイルの在庫管理アプリケーション 63, 67

モバイルポリシー 3, 35–36**ゆ**

- ユーザアクセスの制限 35–36
- ユーザエージェントフロー 24

よ

- 要件, Android 52
- 要件, ハイブリッド 58

り

- リソースのリストを取得 138
- リファレンスドキュメント 137
- リモートアクセス 23
- リモートアクセスアプリケーション 33
- リリース 7
- リリースノート 7

ろ

- ローカルストレージ 102–103