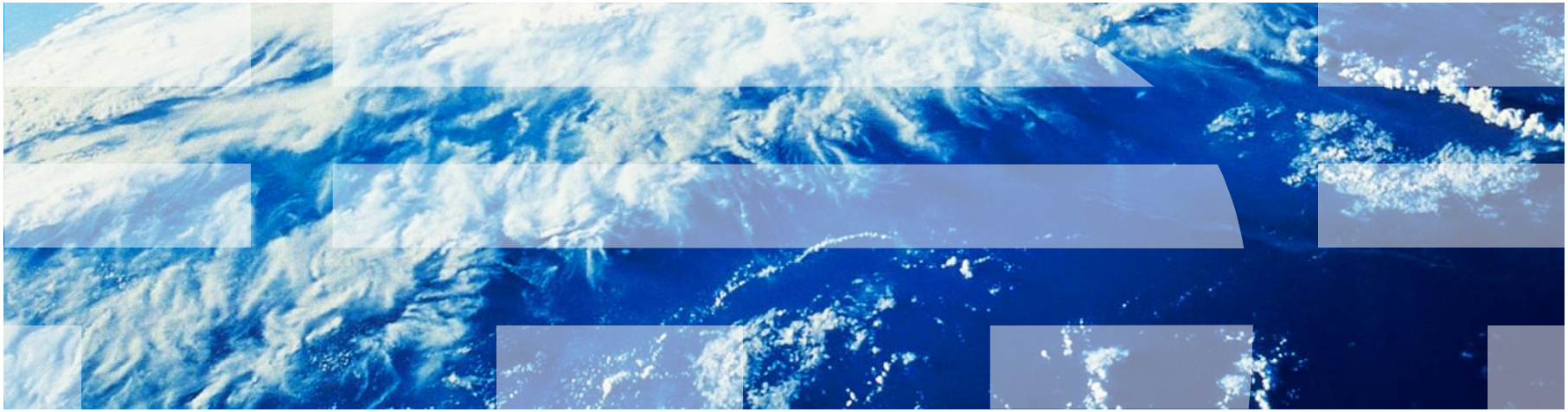
IBM Worklight V6.1.0
入門

Windows Phone 8 – Apache Cordova プラグインを使用して ネイティブ機能をハイブリッド・アプリケーションに追加する



2013 年 12 月 6 日

商標

- IBM、IBM ロゴおよび ibm.com は、世界の多くの国で登録された International Business Machines Corporation の商標です。他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。現時点での IBM の商標リストについては、http://www.ibm.com/legal/copytrade.shtml をご覧ください。
- Microsoft、Windows、Windows NT および Windows ロゴは、Microsoft Corporation の米国およびその他の国における商標です。
- Java およびすべての Java 関連の商標およびロゴは Oracle やその関連会社の米国およびその他の国における商標または登録商標です。

- この資料は、事前に IBM の書面による許可を得ずにその一部または全部を複製することは禁じられています。

IBM® について

- http://www.ibm.com/ibm/us/en/ を参照してください。

アジェンダ

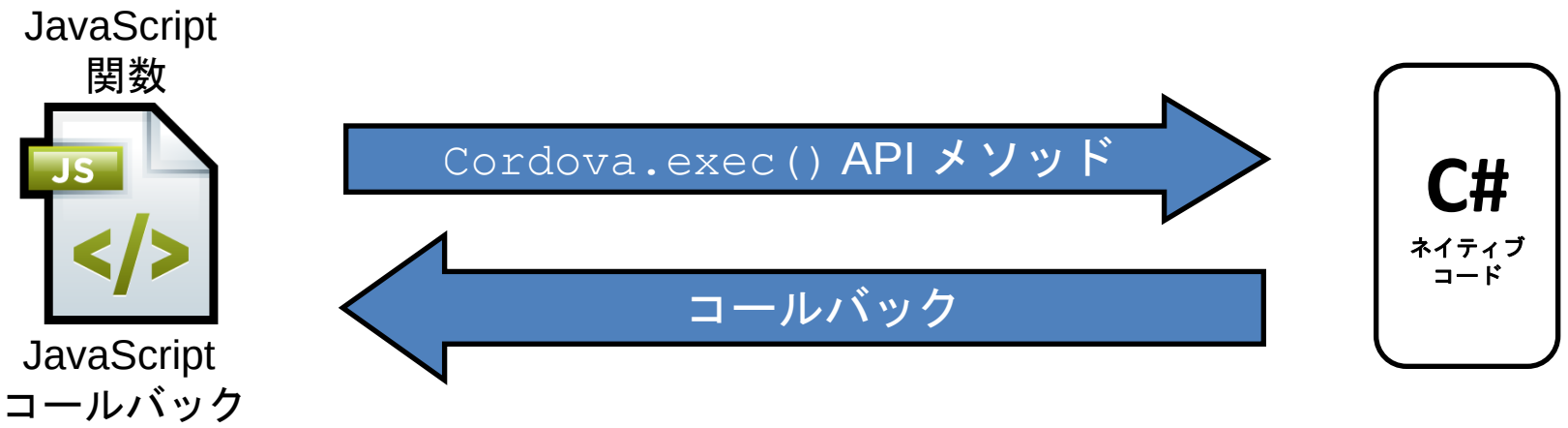
- Apache Cordova プラグイン概説
- C# コードを使用してプラグインを実装する
- プラグインを DOM に追加する
- JavaScript からプラグインを呼び出す
- Worklight プロジェクトを構成する

Apache Cordova プラグイン概説

- IBM Worklight® アプリケーションでは、開発者は、Apache Cordova でまだ使用できない、特定のサード・パーティー・ネイティブ・ライブラリーやデバイス機能を使用しなければならない場合があります。
- Apache Cordova では、開発者は Apache Cordova プラグインを作成できます。これは、開発者がカスタム・ネイティブ・コード・ブロックを作成し、JavaScript™ を使用してアプリケーション内でこれらのコード・ブロックを呼び出すことを意味します。
- このモジュールでは、簡単な Windows Phone 8 Apache Cordova プラグインを作成し、コード内でそれを使用する方法について学習します。
- 他のサンプルが Apache Cordova 資料 (https://github.com/phonegap/phonegap-plugins) にもあります。

Apache Cordova プラグイン概説

- Apache Cordova プラグインは 2 つのパーツから構成されます。
  - Windows Phone OS 内でネイティブに実行される C# コード
  - JavaScript ラッパー
- 両方のパーツが実装されると、開発者は簡単かつ慣れている方法で、JavaScript からネイティブ・コードを呼び出せるようになります。

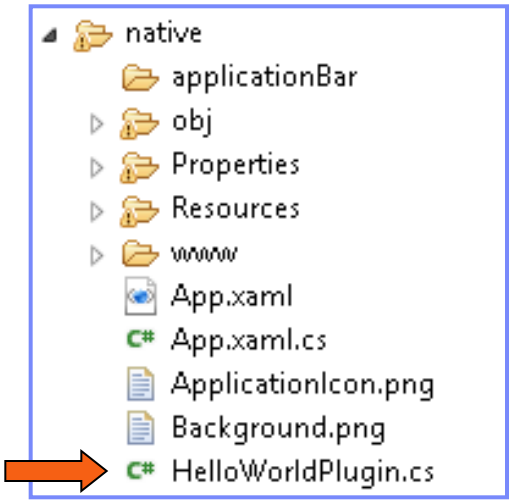


アジェンダ

- Apache Cordova プラグイン概説
- C# コードを使用してプラグインを実装する
- プラグインを DOM に追加する
- JavaScript からプラグインを呼び出す
- Worklight プロジェクトを構成する

C# コードを使用してプラグインを実装する

- まず最初に、プラグインの C# クラスを作成します。
  名前は `HelloWorldPlugin.cs` にします。



- 新しいクラスをプロジェクト名前空間に追加し、必要なインポートを追加します。

```
using WPCordovaClassLib.Cordova;
using WPCordovaClassLib.Cordova.Commands;
using WPCordovaClassLib.Cordova.JSON;

namespace WindowsPhoneCordovaPlugin
{
    public class HelloWorldPlugin : BaseCommand
```

C# コードを使用してプラグインを実装する - 続き

- HelloWorldPlugin クラスと sayHello メソッドを実装します。

```
namespace WindowsPhoneCordovaPlugin
{
    public class HelloWorldPlugin : BaseCommand
    {
        public void sayHello(string options)
        {
            string optVal = null;
            try {
                optVal = JsonHelper.Deserialize<string[]>(options)[0];
            }
            catch (Exception) {
                DispatchCommandResult(new PluginResult(PluginResult.Status.ERROR, "HelloWorldPlugin signaled an error"));
            }

            if (optVal == null)
            {
                DispatchCommandResult(new PluginResult(PluginResult.Status.ERROR, "Got null value as input"));
            }
            else
            {
                DispatchCommandResult(new PluginResult(PluginResult.Status.OK, "{result: Hello " + optVal + "}"));
            }
        }

    }
}
```

C# コードを使用してプラグインを実装する - 続き

- HelloWorldPlugin クラスと sayHello メソッドを実装します。

```
namespace WindowsPhoneCordovaPlugin
{
    public class HelloWorldPlugin : BaseCommand
    {
        public void sayHello(string options)
        {
            string optVal = null;
            try {
                optVal = JsonHelper.Deserialize<string[]>(options)[0];
            }
            catch (Exception) {
                DispatchCommandResult(new PluginResult(PluginResult.Status.ERROR, "HelloWorldPlugin signaled an error"));
            }

            if (optVal == null)
            {
                DispatchCommandResult(new PluginResult(PluginResult.Status.ERROR, "Got null value as input"));
            }
            else
            {
                DispatchCommandResult(new PluginResult(PluginResult.Status.OK, "{result: Hello " + optVal + "}"));
            }
        }

    }
}
```

JavaScript ラッパーは **sayHello** メソッドを呼び出し、1 つのパラメーターを受け渡します。これは、ストリングを JavaScript に返します。

C# コードを使用してプラグインを実装する - 続き

- HelloWorldPlugin クラスと sayHello メソッドを実装します。

```
namespace WindowsPhoneCordovaPlugin
{
    public class HelloWorldPlugin : BaseCommand
    {
        public void sayHello(string options)
        {
            string optVal = null;
            try {
                optVal = JsonHelper.Deserialize<string[]>(
            }
            catch (Exception) {
                DispatchCommandResult(new PluginResult(PluginResult.Status.ERROR, "HelloWorldPlugin signaled an error"));
            }

            if (optVal == null)
            {
                DispatchCommandResult(new PluginResult(PluginResult.Status.ERROR, "Got null value as input"));
            }
            else
            {
                DispatchCommandResult(new PluginResult(PluginResult.Status.OK, "{result: Hello " + optVal + "}"));
            }
        }

    }
}
```

DispatchCommandResult は、成功か失敗かにかかわらず、結果を JavaScript に返します。

アジェンダ

- Apache Cordova プラグイン概説
- C# コードを使用してプラグインを実装する
- プラグインを DOM に追加する
- JavaScript からプラグインを呼び出す
- Worklight プロジェクトを構成する

プラグインを DOM に追加する

- プラグイン実装の第 2 ステップでは、DOM でプラグインを宣言し、そのためのラッパーを作成します。

```
function HelloWorldPlugin(){
}

HelloWorldPlugin.prototype.sayHello = function(onSayHelloSuccess, onSayHelloFailure, name){
    cordova.exec(onSayHelloSuccess,
                 onSayHelloFailure,
                 "WindowsPhoneCordovaPlugin.HelloWorldPlugin",
                 "sayHello",
                 [name]
    );

};

cordova.addConstructor(function() {
    if (!window.plugins) window.plugins = {};
    window.plugins.helloWorldPlugin = new HelloWorldPlugin();
});
```

** `cordova.exec` コマンドでは、プラグイン・クラス名にその名前空間が含まれなければならないことに注意してください。*

プラグインを DOM に追加する - 続き

- プラグイン実装の第 2 ステップでは、DOM でプラグインを宣言し、そのためのラッパーを作成します。

```
function HelloWorldPlugin(){
}

HelloWorldPlugin.prototype.sayHe      function(onSayHelloSuccess, onSayHelloFailure, name){
    cordova.exec(onSayHelloSucce
                 onSayHelloFailure,
                 "WindowsPhoneCordova      n.HelloWorldPlugin",
                 "sayHello",
                 [name]
    );

};

cordova.addConstructor(function() {
    if (!window.plugins) window.plugins = {};
    window.plugins.helloWorldPlugin = new Hel
});
```

まず、プラグインのラッパーとして動作する空の関数を作成します。

プラグインを DOM に追加する - 続き

- プラグイン実装の第 2 ステップでは、DOM でプラグインを宣言し、そのためのラッパーを作成します。

```
function HelloWorldPlugin(){
}

HelloWorldPlugin.prototype.sayHello = function(onSayHelloSuccess, onSayHelloFailure, name){
    cordova.exec(onSayHelloSucce
                 onSayHelloFailu
                 "WindowsPhoneCo      ugin.HelloWorldPlugin",
                 "sayHello",
                 [name]
    );

};

cordova.addConstructor(function() {
    if (!window.plugins) window.plugins = {};
    window.plugins.helloWorldPlugin = new Hell
});
```

sayHello 関数を作成します。
これは、**HelloWorldPlugin** プロトタイプと
ハードコーディングされたプラグイン・クラス名
およびアクションを使用して作成します。
この関数は、**cordova.exec()** を使用してプラグインを
呼び出します。

プラグインを DOM に追加する - 続き

- プラグイン実装の第 2 ステップでは、DOM でプラグインを宣言し、そのためのラッパーを作成します。

```
function HelloWorldPlugin(){
}

HelloWorldPlugin.prototype.sayHello = function(onSayHelloSuccess, onSayHelloFailure, name){
    cordova.exec(onSayHelloSuccess,
                 onSayHelloFailure,
                 "WindowsPhoneCordovaPlugin.HelloWorldPlugin",
                 "sayHello",
                 [name]
    );

};

cordova.addConstructor(function() {
    if (!window.plugins) window.plugins = {};
    window.plugins.helloWorldPlugin = new HelloWorldPlugin();
});
```

成功コールバック
失敗コールバック
プラグイン・クラス名
プラグイン・メソッド名
パラメーター配列

プラグインを DOM に追加する - 続き

- プラグイン実装の第 2 ステップでは、DOM でプラグインを宣言し、そのためのラッパーを作成します。

```
function HelloWorldPlugin(){
}

HelloWorldPlugin.prototype.sayHello = functi
    cordova.exec(onSayHelloSuccess,
                 onSayHelloFailure,
                 "WindowsPhoneCordovaPlugin.
                 "sayHello",
                 [name]
    );

};

cordova.addConstructor(function() {
    if (!window.plugins) window.plugins = {};
    window.plugins.helloWorldPlugin = new HelloWorldPlugin();
});
```

最終ステップでは、helloWorldPlugin プロパティーを DOM window.plugins オブジェクトに追加します。これで、**window.plugins.helloWorldPlugin.sayHello()** を使用してプラグインを呼び出すことができます。

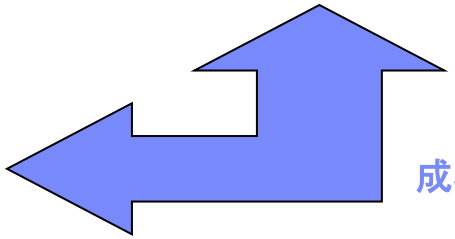
アジェンダ

- Apache Cordova プラグイン概説
- C# コードを使用してプラグインを実装する
- プラグインを DOM に追加する
- JavaScript からプラグインを呼び出す
- Worklight プロジェクトを構成する

JavaScript からプラグインを呼び出す

- これで JavaScript からプラグインを呼び出す準備ができました。

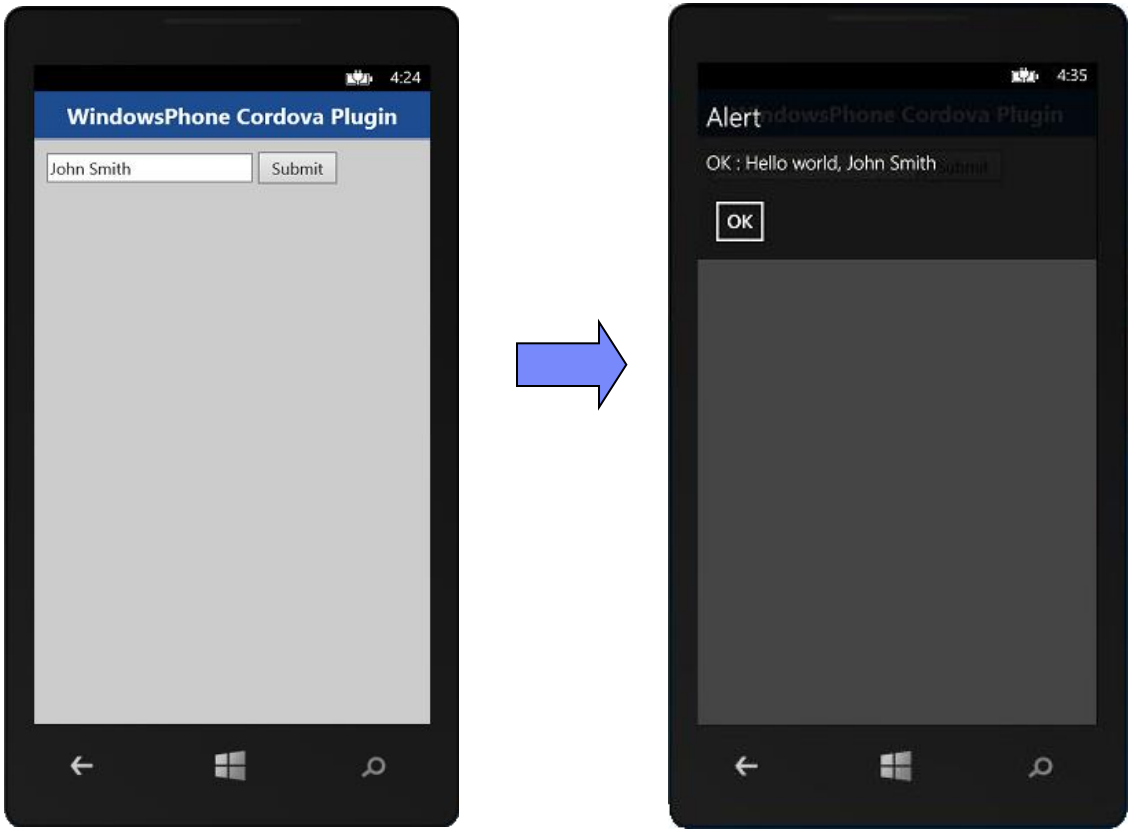
```
function greetMe(){
    window.plugins.helloWorldPlugin.sayHello(sayHelloSuccess, sayHelloFailure, $("#NameInput").val());
}

function sayHelloSuccess(data){
    alert("OK : " + data);
}

function sayHelloFailure(data){
    alert("FAIL : " + data);
}
```



成功コールバックと失敗コールバック

JavaScript からプラグインを呼び出す - 続き

- このトレーニング・モジュールのサンプルは、IBM Worklight 文書 Web サイト (http://www.ibm.com/mobile-docs) の「**入門**」 ページにあります。



アジェンダ

- Apache Cordova プラグイン概説
- C# コードを使用してプラグインを実装する
- プラグインを DOM に追加する
- JavaScript からプラグインを呼び出す
- Worklight プロジェクトを構成する

Worklight プロジェクトを構成する

- アプリケーション内でプラグインが作動するためには、さらにいくつかのステップが必要です。
  - プラグインはネイティブ・フォルダーに入れる必要があります。
  - ネイティブ・フォルダーにある `config.xml` ファイルの **User** セクションで、以下のようにプラグインを宣言する必要があります。

```
<feature name="WindowsPhone8CordovaPlugin">
    <param name="wp-package" value="HelloWorldPlugin" />
</feature>
```

特記事項

- これらの資料は、以下のご使用条件に同意していただける場合に限りご使用いただけます。
- 本書は米国 IBM が提供する製品およびサービスについて作成したものです。
- 本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、または サービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の 製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作または サービスの 評価および検証は、お客様の責任で行っていただきます。
- IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。
  - 〒103-8510
    東京都中央区日本橋箱崎町19番21号
    日本アイ・ビー・エム株式会社
    法務・知的財産
    知的財産権ライセンス渉外
- **以下の保証は、国または地域の法律に沿わない場合は、適用されません。IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態で提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含む すべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が 禁じられる場合、強行規定の制限を受けるものとします。**
- この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。
- 本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部では ありません。それらの Web サイトは、お客様の責任でご使用ください。
- IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、 自ら適切と信ずる方法で、使用もしくは配布することができるものとします。
- 本プログラムのライセンス保持者で、(i) 独自に作成したプログラムと その他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。
  - IBM Corporation
    Dept F6, Bldg 1
    294 Route 100
    Somers NY 10589-3216
    USA
- 本プログラムに関する上記の情報は、適切な使用条件の下で使用すること ができますが、有償の場合もあります。
- 本書で説明されているライセンス・プログラムまたはその他のライセンス資 料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。
- IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。IBM は、それらの製品のテストは行っておりません。したがって、 他社製品に関する実行性、互換性、またはその他の要求については確証できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。

著作権使用許諾:

- 本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに 準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性が あることをほのめかしたり、保証することはできません。
- それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生的創作物にも、次の ように、著作権表示を入れていただく必要があります。
  - © (お客様の会社名) (西暦年).このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。© Copyright IBM Corp. _年を入れる_. All rights reserved.

プライバシー・ポリシーの考慮事項

- サービス・ソリューションとしてのソフトウェアも含めた IBM ソフトウェア製品 (「ソフトウェア・オファリング」) では、製品の使用に関する情報の収集、エンド・ユーザーの使用感の向上、エンド・ユーザーとの対話またはその他の目的のために、Cookie はじめさまざまなテクノロジーを使用することがあります。多くの場合、ソフトウェア・オファリングにより個人情報が収集されることはありません。IBM の「ソフトウェア・オファリング」の一部には、個人情報を収集できる機能を持つものがあります。ご使用の「ソフトウェア・オファリング」が、これらの Cookie およびそれに類するテクノロジーを通じてお客様による個人情報の収集を可能にする場合、以下の具体的事項を確認ください。
- このソフトウェア・オファリングは、展開される構成に応じて、(アプリケーション・サーバーが生成する) セッション情報を収集するセッションごとの Cookie を使用する場合があります。これらの Cookie は個人情報を含まず、セッション管理のために要求されるものです。加えて、匿名ユーザーの認識および管理のために持続的な Cookie が無作為に生成される場合があります。これらの Cookie も個人情報を含まず、要求されるものです。
- この「ソフトウェア・オファリング」が Cookie およびさまざまなテクノロジーを使用してエンド・ユーザーから個人を特定できる情報を収集する機能を提供する場合、お客様は、このような情報を収集するにあたって適用される法律、ガイドライン等を遵守する必要があります。これには、エンドユーザーへの通知や同意の要求も含まれますがそれらには限られません。このような目的での Cookie を含む様々なテクノロジーの使用の詳細については、IBM の『IBM オンラインでのプライバシー・ステートメント』(http://www.ibm.com/privacy/details/jp/ja/) の『クッキー、ウェブ・ビーコン、その他のテクノロジー』および『IBM Software Products and Software-as-a-Service Privacy Statement』(http://www.ibm.com/software/info/product-privacy) を参照してください。

サポートおよびコメント

- IBM Worklight の一連の文書、トレーニング資料、および質問をポストできるオンライン・フォーラムはすべて、次の IBM Web サイトからご覧になれます。
  - http://www.ibm.com/mobile-docs
- **サポート**
  - ソフトウェア・サブスクリプション & サポート (ソフトウェア・メンテナンスと呼ばれる場合もあります) は、パスポート・アドバンテージおよびパスポート・アドバンテージ・エクスプレスから購入されたライセンスに含まれています。International Passport Advantage Agreement および IBM International Passport Advantage Express Agreement の追加情報については、次のパスポート・アドバンテージ Web サイトを参照してください。
    - http://www.ibm.com/software/passportadvantage
  - ソフトウェア・サブスクリプション & サポートが有効になっている場合、IBM は、インストールおよび使用法 (ハウツー) に関する短期間の FAQ に対するサポートや、コード関連の質問に対するサポートを提供します。詳しくは、次の IBM ソフトウェア・サポート・ハンドブックを参照してください。
    - http://www.ibm.com/support/handbook
- **ご意見**
  - 本資料に関するご意見をお寄せください。本資料の具体的な誤りや欠落、正確性、編成、題材、または完成度に関するご意見をお寄せください。 お寄せいただくご意見は、本マニュアルまたは製品の情報、およびその情報の提示方法に関するもののみとしてください。
  - 製品の技術的な質問および情報、および価格については、担当の IBM 営業所、IBM ビジネス・パートナー、または認定リマーケターにお問い合わせください。
  - IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。IBM またはいかなる組織も、お客様から提示された問題についてご連絡を差し上げる場合にのみ、お客様が提供する個人情報を使用するものとします。
  - どうぞよろしくお願いいたします。
  - 次の IBM Worklight Developer Edition サポート・コミュニティーにご意見をお寄せください。
    - https://www.ibm.com/developerworks/mobile/worklight/connect.html
  - IBM からの回答を希望される場合は、以下の情報をご連絡ください。
    - 氏名
    - 住所
    - 企業または組織
    - 電話番号
    - E メール・アドレス

ありがとうございました