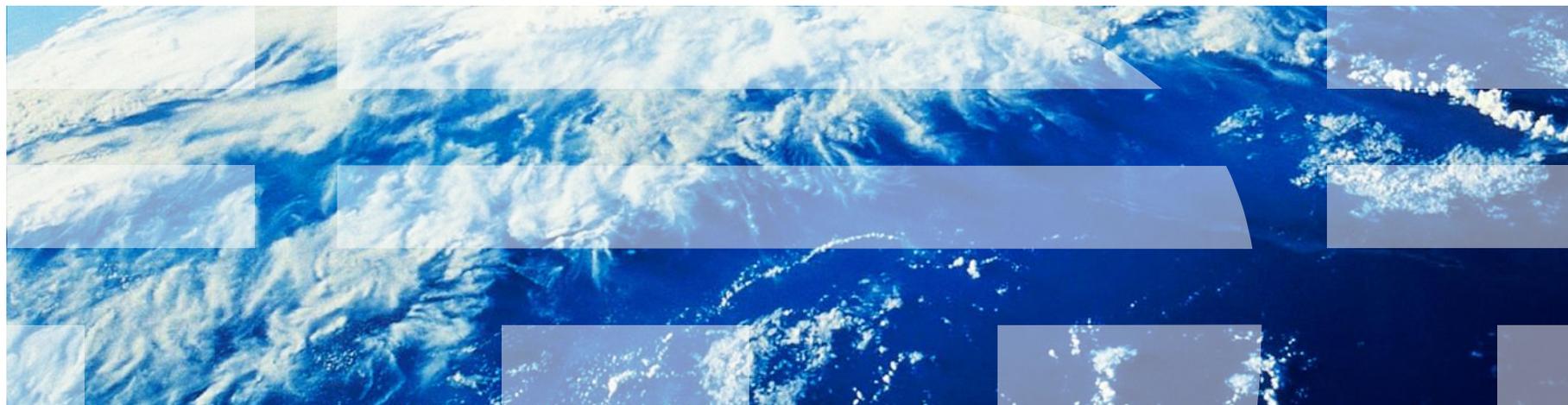


# ***IBM Worklight V6.1.0*** **入門**

## ネイティブ Java ME アプリケーションでの IBM Worklight API の使用



## 商標

- IBM、IBM ロゴおよび [ibm.com](http://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、[www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml) をご覧ください。
- 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

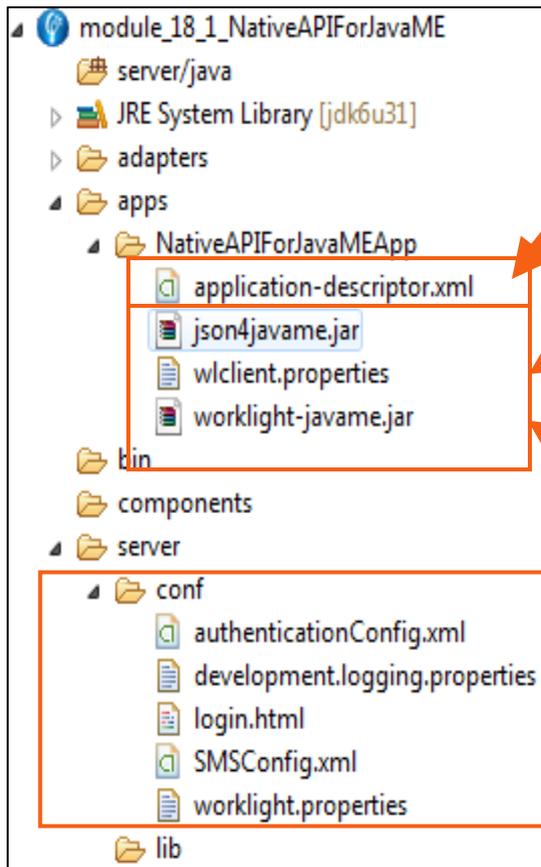
- IBM Worklight ネイティブ API の作成
- Java ME ネイティブ・アプリケーションの作成および構成
- WLCClient の初期化
- IBM Worklight プロシージャーの呼び出し
- プロシージャー応答の受信
- BlackBerry の場合

## IBM Worklight ネイティブ API の作成 (1/3)

- IBM Worklight® は、Java™ Platform, Micro Edition (Java ME) アプリケーションが IBM Worklight ネイティブ API ライブラリーを使用して Worklight Server と通信できるようにします。
- Java ME アプリケーションでこの機能を使用できるようにするには、Worklight Server 側でこのアプリケーションが認識されている必要があります。
- IBM Worklight ネイティブ API は、IBM Worklight プロジェクトの **apps** フォルダ内にあります。
- IBM Worklight ネイティブ API フォルダは次の 2 つの目的に適います。
  - このフォルダには、ネイティブ API ライブラリーと構成ファイルが含まれており、これらを Java ME プロジェクトにコピーする必要があります。
  - このフォルダには **application-descriptor.xml** ファイルが含まれています。このファイルは、エントリー・ポイントとして機能するように Worklight Server にデプロイできます。
- このモジュールでは、IBM Worklight ネイティブ API を作成し、そのコンポーネントを Java ME アプリケーションで使用する方法について学習します。

## IBM Worklight ネイティブ API の作成 (2/3)

- IBM Worklight ネイティブ API には複数のコンポーネントが含まれています。



**application-descriptor.xml** ファイルを使用して、アプリケーション・メタデータを定義し、Worklight Server で実施されるセキュリティー設定を構成します。

**wlclient.properties** ファイルには、ネイティブ Java ME アプリケーションで使用される接続設定が含まれます。このファイルをネイティブ Java ME プロジェクトにコピーする必要があります。

**worklight-javame.jar** ファイルおよび **json4javame.jar** ファイルは IBM Worklight API ライブラリーを定義しており、ネイティブ Java ME プロジェクトにコピーする必要があります。

その他の Worklight プロジェクトの場合と同様に、**server¥conf** フォルダー内にあるファイルを変更することによって、サーバー構成を定義します。

## IBM Worklight ネイティブ API の作成 (3/3)

1. Worklight Studio で、Worklight プロジェクトを作成し、IBM Worklight ネイティブ API を追加します。
2. 「新規 Worklight ネイティブ API (New Worklight Native API)」ダイアログでネイティブ API 名を入力し、「環境 (Environment)」フィールドで「JavaME」を選択します。
3. IBM Worklight ネイティブ API フォルダーを右クリックし、「実行 (Run As)」>「ネイティブ API のデプロイ (Deploy Native API)」を選択します。

# アジェンダ

- IBM Worklight ネイティブ API の作成
- Java ME ネイティブ・アプリケーションの作成および構成
- WLCClient の初期化
- IBM Worklight プロシージャーの呼び出し
- プロシージャー応答の受信
- BlackBerry の場合

## Java ME ネイティブ・アプリケーションの作成および構成

1. Java ME ネイティブ・アプリケーションを作成します。
2. IBM Worklight ネイティブ API フォルダから Java ME ネイティブ・アプリケーション (/lib ディレクトリー内) に、`worklight-javame.jar` ファイルおよび `json4javame.jar` ファイルをコピーします。
3. IBM Worklight ネイティブ API フォルダから新しい Java ME ネイティブ・アプリケーション (/res ディレクトリー内) に、`wlclient.properties` ファイルをコピーします。

# アジェンダ

- IBM Worklight ネイティブ API の作成
- Java ME ネイティブ・アプリケーションの作成および構成
- **WLCLient の初期化**
- IBM Worklight プロシージャーの呼び出し
- プロシージャー応答の受信
- BlackBerry の場合

## WLClient の初期化

- WLClient のインスタンスを作成します。

```
private WLClient client;  
  
public JavaMENativeApp() {  
    client = WLClient.createInstance(this);  
    this.display = Display.getDisplay(this);  
}
```

- Worklight Server への接続を確立するには、**connect** メソッドを使用して、パラメーターとして **MyConnectListener** クラス・インスタンスを指定します。

```
public void commandAction(Command command, Item item) {  
    StringItem itemName = (StringItem)item;  
    if(itemName.getText().equals("1.Connect")) {  
        updateTextView("\nConnecting...");  
        client.connect(new MyConnectListener());  
    }  
}
```

- この作成方法については、次のスライドを参照してください。

## MyConnectListener

- 最初に、`WLClient` インスタンスは、`wlclient.properties` ファイルのプロパティに従って Worklight Server に接続します。
- 接続が確立すると、インスタンスにより、`MyConnectListener` クラスのいずれかのメソッドが呼び出されます。
  - `MyConnectListener` クラスには `WLResponseListener` インターフェースが実装されています。

```
public class MyConnectListener implements WLResponseListener {
```

- `WLResponseListener` インターフェースでは、以下の2つのメソッドが指定されます。
  - `public void onSuccess (WLResponse response) { }`
  - `public void onFailure (WLFailResponse response) { }`
- これらのメソッドを使用して、接続の成功または接続の失敗を処理します。

# アジェンダ

- IBM Worklight ネイティブ API の作成
- Java ME ネイティブ・アプリケーションの作成および構成
- WLCClient の初期化
- IBM Worklight プロシージャラーの呼び出し
- プロシージャラー応答の受信
- BlackBerry の場合

## IBM Worklight プロシーチャーの呼び出し (1/3)

- Worklight Server との接続が確立したら、`WLClient` インスタンスを使用してアダプター・プロシーチャーを呼び出すことができます。

```
}else if(itemName.getText().equals("2.Invoke Procedure")) {
    updateTextView("\nInvoking procedure...");
    String adapterName = "RSSReader";
    String procedureName = "getStoriesFiltered";

    WLProcedureInvocationData invocationData = new WLProcedureInvocationData(adapterName, procedureName);

    Object[] parameters = new Object[] {};
    invocationData.setParameters(parameters);

    WLRequestOptions options = new WLRequestOptions();
    client.invokeProcedure(invocationData, new MyInvokeListener(), options);
}
```

- アダプター名およびプロシーチャー名を指定して `WLProcedureInvocationData` オブジェクトを作成します。

## IBM Worklight プロシーチャーの呼び出し (2/3)

- Worklight Server との接続が確立したら、`WLClient` インスタンスを使用してアダプター・プロシーチャーを呼び出すことができます。

```
}else if(itemName.getText().equals("2.Invoke Procedure")) {  
    updateTextView("\nInvoking procedure...");  
    String adapterName = "RSSReader";  
    String procedureName = "getStoriesFiltered";  
  
    WLProcedureInvocationData invocationData = new WLProcedureInvocationData(adapterName, procedureName);  
  
    Object[] parameters = new Object[] {};  
    invocationData.setParameters(parameters);  
  
    WLRequestOptions options = new WLRequestOptions();  
    client.invokeProcedure(invocationData, new MyInvokeListener(), options);  
}
```

- 必須パラメーターをオブジェクト配列として追加し、要求オプションを設定します。

## IBM Worklight プロシーチャーの呼び出し (3/3)

- Worklight Server との接続が確立したら、`WLClient` インスタンスを使用してアダプター・プロシーチャーを呼び出すことができます。

```
}else if(itemName.getText().equals("2.Invoke Procedure")) {  
    updateTextView("\nInvoking procedure...");  
    String adapterName = "RSSReader";  
    String procedureName = "getStoriesFiltered";  
  
    WLProcedureInvocationData invocationData = new WLProcedureInvocationData(adapterName, procedureName);  
  
    Object[] parameters = new Object[] {};  
    invocationData.setParameters(parameters);  
  
    WLRequestOptions options = new WLRequestOptions();  
    client.invokeProcedure(invocationData, new MyInvokeListener(), options);  
}
```

- パラメーターとして `MyInvokeListener` クラス・インスタンスを指定します。
  - このようなインスタンスの定義方法については、次のスライドを参照してください。

# アジェンダ

- IBM Worklight ネイティブ API の作成
- Java ME ネイティブ・アプリケーションの作成および構成
- WLCClient の初期化
- IBM Worklight プロシージャーの呼び出し
- プロシージャー応答の受信
- BlackBerry の場合

## プロシージャー応答の受信 (1/3)

- プロシージャー呼び出しが完了すると、**WLClient** インスタンスは、**MyInvokeListener** クラスのいずれかのメソッドを呼び出します。
- **MyInvokeListener** クラスには **WLResponseListener** インターフェースが実装されています。

```
import com.worklight.wlclient.api.WLFailResponse;  
import com.worklight.wlclient.api.WLResponse;  
import com.worklight.wlclient.api.WLResponseListener;  
  
public class MyInvokeListener implements WLResponseListener {
```

- **WLClient** は、その **onSuccess** メソッドまたは **onFailure** メソッドを呼び出します (次のスライドを参照)。

## プロシージャー応答の受信 (2/3)

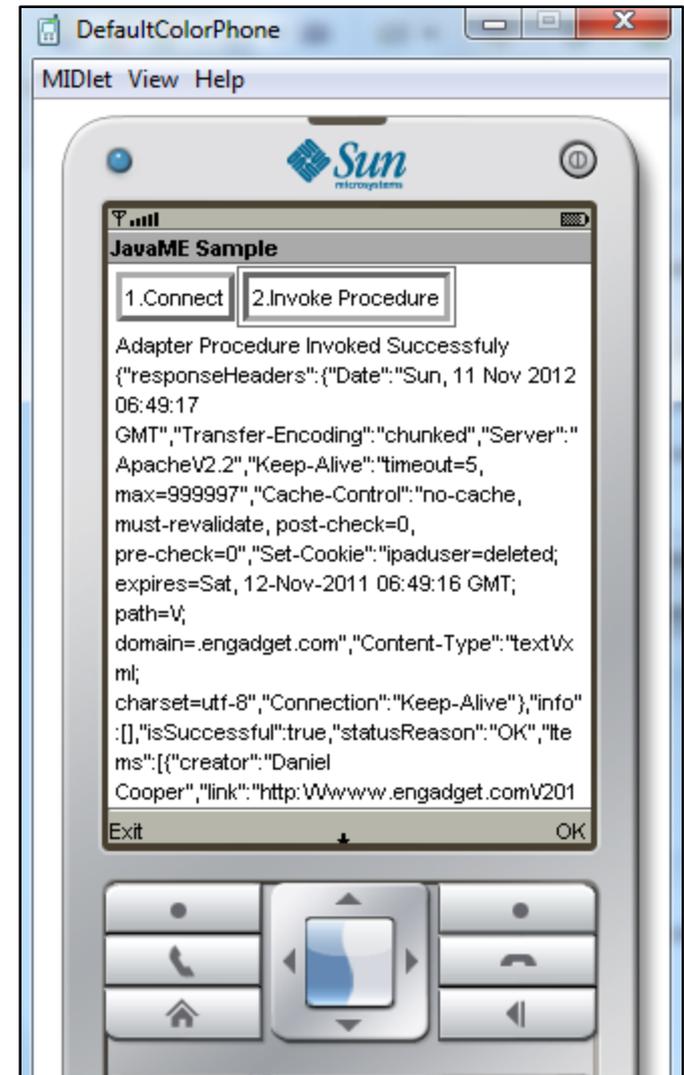
- プロシージャー呼び出しが成功した場合、**MyInvokeListener** の **onSuccess** メソッドが呼び出されます。
- このメソッドを使用して、アダプターから取り出されたデータを取得します。

```
public class MyConnectListener implements WLResponseListener {  
    public void onSuccess(WLResponse response) {  
        String responseText = response.getResponseText();  
        JavaMENativeApp.updateTextView("Connected Successfully\n" + responseText);  
    }  
  
    public void onFailure(WLFailResponse response) {  
        String responseText = response.getResponseText();  
        JavaMENativeApp.updateTextView("Connection Failure\n" + responseText);  
    }  
}
```

- 応答 オブジェクトには、応答データが含まれます。
  - このメソッドおよびプロパティを使用して、必要な情報を取得できます。

## プロシーチャー応答の受信 (3/3)

- 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。
- このサンプルには以下の 2 つのプロジェクトが含まれます。
  - **NativeAPIForJavaME.zip** ファイルには、Worklight Server にデプロイできる IBM Worklight ネイティブ API が格納されています。
  - **JavaMENativeApp.zip** ファイルには、Worklight Server と通信するために IBM Worklight ネイティブ API ライブラリーを使用する JavaME アプリケーションが格納されています。
- **重要:** 関連するサーバー設定を使用して、**JavaMENativeApp** 内の **wlclient.properties** ファイルを必ず更新するようにしてください。



# アジェンダ

- IBM Worklight ネイティブ API の作成
- Java ME ネイティブ・アプリケーションの作成および構成
- WLCClient の初期化
- IBM Worklight プロシージャーの呼び出し
- プロシージャー応答の受信
- BlackBerry の場合

## BlackBerry の場合

- BlackBerry の場合、`createInstance(String connectionString, MIDlet midlet)` メソッドを使用して、`WLClient` インスタンスを作成します。
- BlackBerry の場合、ネットワーク (HTTP またはソケット) 接続の確立方法が異なります。
  - Worklight Server への接続に使用するネットワーク接続のタイプを識別するには、`createInstance(String connectionString, MIDlet midlet)` メソッドに適切なストリング引数を渡します。
    - 例: `deviceside=true`
  - 詳しくは、BlackBerry 開発者用の知識ベース (<http://supportforums.blackberry.com/t5/Java-Development/Different-ways-to-make-an-HTTP-or-socket-connection/ta-p/445879>) を参照してください。

# 特記事項

- これらの資料は、以下のご使用条件に同意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を提供し、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認できません。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をお願いします。

## 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めめしたり、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、以下の具体的な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、または認定リマーカー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。
  - どうぞよろしく願いいたします。
  - 次の IBM Worklight Developer Edition サポート・コミュニティ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

