

Silk Test 17.5

モバイル アプリケーションのテスト

Micro Focus
The Lawn
22-30 Old Bath Road
Newbury, Berkshire RG14 1QN
UK
<http://www.microfocus.com>

Copyright © Micro Focus 1992-2016. All rights reserved.

MICRO FOCUS, Micro Focus ロゴ及び Silk Test は Micro Focus IP Development Limited
またはその米国、英国、その他の国に存在する子会社・関連会社の商標または登録商標です。

その他、記載の各名称は、各所有社の知的所有財産です。

2016-11-03

目次

モバイル アプリケーションのテスト	4
Android	4
Android 上のモバイル アプリケーションのテストにおける前提条件	4
Android 上のモバイル アプリケーションのテスト	4
Android 上のハイブリッド アプリケーションのテスト	5
USB ドライバのインストール	6
USB デバッグの有効化	6
Android デバイスの推奨設定	7
Silk Test 用に Android エミュレータを設定する	7
iOS	8
iOS 上のモバイル アプリケーションのテストにおける前提条件	8
物理 iOS デバイス上のネイティブ モバイル アプリケーションのテスト	9
iOS シミュレータ上のネイティブ モバイル アプリケーションのテスト	10
物理 iOS デバイス上のモバイル Web アプリケーションのテスト	11
iOS シミュレータ上のモバイル Web アプリケーションのテスト	12
iOS 上のハイブリッド アプリケーションのテスト	12
iOS デバイスのテストの準備	12
iOS アプリのテストの準備	13
Silk Test Information Service を Mac にインストールする	13
Silk Test Information Service を Mac からアンインストールする	14
iOS デバイスの推奨設定	15
同じ Mac 上での複数の iOS デバイスまたはシミュレータのテスト	15
インストール済みアプリのテスト	15
モバイル アプリケーションの記録	16
テストを再生するモバイル デバイスの選択	16
モバイル デバイスの操作	17
モバイル デバイスの開放	17
記録後のモバイル デバイスの開放	17
再生後のモバイル デバイスの開放	18
リモート ロケーションの編集	18
モバイル デバイスの接続文字列	19
モバイル アプリケーションのテスト時のトラブルシューティング	22
テストの再生に Chrome for Android を使用する方法	26
モバイル Web アプリケーションのテストにおける制限事項	27
ネイティブ モバイル アプリケーションのテストにおける制限事項	28
モバイル Web サイトでのオブジェクトのクリック	29
既存のモバイル Web テストの使用法	29

モバイル アプリケーションのテスト

Silk Test では、ネイティブ モバイル アプリケーション (アプリ) およびモバイル Web アプリケーションを自動的にテストすることができます。Silk Test を使用してモバイル アプリケーションを自動的にテストすることには、次のメリットがあります。

- モバイル アプリケーションのテスト時間を大幅に減少させることができます。
- テストを一旦作成すれば、数多くの異なるデバイスやプラットフォーム上でモバイル アプリケーションをテストできます。
- エンタープライズ モバイル アプリケーションに要求される信頼性とパフォーマンスを確保できます。
- QA チームのメンバーおよびモバイル アプリケーションの開発者の効率を向上できます。
- モバイル アプリケーションは、多くのモバイル デバイスとプラットフォームで動作することを要求されるため、アジャイルにフォーカスした開発環境にとって手動テストは十分効率的とは言えない場合があります。

 **注:** Silk Test を使用してネイティブ モバイル アプリケーションやハイブリッド アプリケーションをテストするには、ネイティブ モバイル ライセンスが必要です。詳細については、「[ライセンス情報](#)」を参照してください。

 **注:** Silk Test は、Android および iOS デバイスの両方でのモバイル アプリのテストをサポートします。

モバイル アプリケーションのテストをサポートするオペレーティング システムとサポートするブラウザについての情報は、『[リリースノート](#)』を参照してください。

Android

Silk Test では、Android デバイスまたは Android エミュレータ上のモバイル アプリケーションをテストすることができます。

Android 上のモバイル アプリケーションのテストにおける前提条件

Android デバイスや Android エミュレータ上のモバイル アプリケーションをテストする前に、次の前提条件を満たしていることを確認してください。

- Silk Test を実行するマシンに Java インストールされており、Java へのパスが *Path* 環境変数に追加されていることを確認してください。Silk Test は、JRE または JDK のいずれかをモバイル テストで必要とします。Java がインストールされていない場合は、`C:\Program Files (x86)\Silk\SilkTest\ng\jre\bin` を *Path* に追加してください。
- ネイティブ モバイル アプリに Web ビューを追加したハイブリッド アプリを作成した場合は、アプリを Silk Test でテスト可能にするために、次のコードをアプリに追加してください。

```
WebView.setWebContentsDebuggingEnabled(true);  
webView.getSettings().setJavaScriptEnabled(true);
```

- Silk Test では、Android 4.4 のデバイス画面のライブ ビュー表示をサポートしていません。このため、Android 5 以降を使用することを Micro Focus は推奨しています。

Android 上のモバイル アプリケーションのテスト

物理 Android デバイスや Android エミュレータ上のモバイル アプリケーションをテストするには、次のタスクを実行します。

1. Android 上のモバイル アプリケーションのテストにおける前提条件を満たすことを確認します。
詳細については、「[Android 上のモバイル アプリケーションのテストにおける前提条件](#)」を参照してください。
2. Android エミュレータ上のモバイル アプリケーションをテストするには、Silk Test 用にエミュレータを設定します。
詳細については、「[Silk Test 用に Android エミュレータを設定する](#)」を参照してください。
3. Silk Test をインストールしたマシンで Android エミュレータを開始するか、デバイスを接続します。
4. 物理 Android デバイスを初めて使用するマシンでテストする場合、適切な Android USB ドライバをマシンにインストールします。
詳細については、「[USB ドライバのインストール](#)」を参照してください。
5. 物理 Android デバイス上でモバイル アプリケーションをテストする場合は、Android デバイスの USB デバッグを有効化します。
詳細については、「[USB デバッグの有効化](#)」を参照してください。
6. モバイル アプリケーション用の Silk Test プロジェクトを作成します。
7. モバイル アプリケーション用のテストを作成します。
8. テストで実行する操作を記録します。記録 ウィンドウを開始すると、**アプリケーションの選択** ダイアログ ボックスが開きます。
9. モバイル Web アプリケーションをテストするには：
 - a) **Web** タブを選択します。
 - b) 使用するモバイル ブラウザーを選択します。
 - c) **移動する URL の入力** テキスト ボックスに、開く Web ページを指定します。
10. ネイティブ モバイル アプリケーションまたはハイブリッド アプリケーションをテストするには：
 **注:** Silk Test を使用してネイティブ モバイル アプリケーションやハイブリッド アプリケーションをテストするには、ネイティブ モバイル ライセンスが必要です。詳細については、「[ライセンス情報](#)」を参照してください。
 - a) **モバイル** タブを選択します。
 - b) アプリをテストするモバイル デバイスをリストから選択します。
 - c) **参照** をクリックしてアプリ ファイルを選択するか、アプリ ファイルへの完全パスを **モバイル アプリ ファイル** テキスト フィールドに入力します。
このパスでは、Silk Test は HTTP および UNC 形式をサポートします。
Silk Test は、モバイル デバイスまたはエミュレータ上に指定したアプリをインストールします。
11. **OK** をクリックします。
Android デバイスまたはエミュレータの画面が、テスト中にロックされないようにしてください。マシンに接続中にデバイスがロックされないようにするには、**開発者向けオプション** を開きます。**スリープモードにしない** または **充電中に画面をスリープにしない** をチェックします。
12. **記録** ウィンドウを使用して、モバイル アプリケーションに対するテストを記録します。
詳細については、「[モバイル アプリケーションの記録](#)」を参照してください。
13. すべての操作の記録を終えたら、記録を停止します。
14. テストを再生します。
15. テスト結果を分析します。

Android 上のハイブリッド アプリケーションのテスト

ハイブリッド アプリケーション (アプリ) は、デバイス上で実行されるネイティブ アプリケーションのようなアプリですが、HTML5、CSS、JavaScript などの Web テクノロジーを使用して記述されたアプリです。

Silk Test は、ネイティブ コンテナに埋め込まれた単一の Web ビューで構成されたハイブリッド アプリのテストをサポートします。このようなハイブリッド アプリの一般的な例は、Apache Cordova アプリケーションです。

Silk Test は、Chrome リモート デバッグをサポートする Web ビューを検出できます。Silk Test は、`com.android.webview` パッケージまたは `com.google.android.webview` パッケージ（多くの Android デバイス上のデフォルト パッケージ）のいずれかを使用した Web ビューを検出できます。



注: Silk Test は、Android 4.4 以降でのハイブリッド アプリのテストをサポートします。Android 6.0 以降でハイブリッド アプリをテストするには、Android システムの WebView バージョン 51 以降が必要です。

Android 上のハイブリッド アプリをテストする手順は、モバイル ネイティブ アプリケーションをテストする手順と同じです。詳細については、「[Android 上のモバイル アプリケーションのテスト](#)」を参照してください。

USB ドライバのインストール

モバイル アプリケーションをテストするために、ローカル マシンに最初に Android デバイスに接続するには、適切な USB ドライバをインストールする必要があります。

デバイスの製造元は、そのデバイスに必要なすべてのドライバをもった実行可能ファイルを提供している可能性があります。この場合、ローカル マシンにその実行可能ファイルをインストールするだけです。製造元がこのような実行可能ファイルを提供していない場合、マシン上にデバイスに対する単一の USB ドライバをインストールできます。

Android USB ドライバをインストールするには：

1. デバイス用の適切なドライバをダウンロードします。
たとえば、Google Nexus デバイス用の USB ドライバを検索し、インストールする場合は、<http://developer.android.com/tools/extras/oem-usb.html> を参照します。
2. Android デバイスをローカル マシンの USB ポートに接続します。
3. デスクトップ、または **Windows Explorer** から、**コンピュータ** を右クリックし、**管理** を選択します。
4. 左側のペインで、**デバイス マネージャ** を選択します。
5. 右側のペインで、**その他のデバイス** を探して展開します。
6. デバイス名 (*Nexus 5x* など) を右クリックして、**ドライバ ソフトウェアの更新** を選択します。**ハードウェアの更新ウィザード** が開きます。
7. **コンピュータを参照してドライバ ソフトウェアを検索します** を選択して、**次へ** をクリックします。
8. **参照** をクリックし、USB ドライバをダウンロードしたフォルダに移動します。
9. USB ドライバを選択します。
- 10 **次へ** をクリックしてドライバをインストールします。

USB デバッグの有効化

Android Debug Bridge (adb) 上で Android デバイスと通信するために USB デバッグを有効化します。

1. Android デバイスで設定を開きます。
2. **開発者向けオプション** (Dev Settings) をタップします。
開発者向けオプションは、デフォルトでは表示されません。開発者向けオプションがデバイスの設定メニューに含まれていない場合：
 - a) 画面を下にスクロールさせて、デバイスが携帯電話の場合は **端末情報** を、タブレットの場合は **タブレット情報** をタップします。
 - b) 再度画面を下にスクロールさせて **ビルド番号** を 7 回タップします。
3. **開発者向けオプション** ウィンドウで、**USB デバッグ** をオンにします。
4. デバイスの USB モードをデフォルトの設定である **メディア デバイス (MTP)** に設定します。
詳細については、デバイスのドキュメントを参照してください。

Android デバイスの推奨設定

Silk Test を使用したテストを最適化するために、テストしたい Android デバイスで次の設定を行ってください。

- USB デバッグを Android デバイスで有効化します。詳細については、「[USB デバッグの有効化](#)」を参照してください。
- Android デバイスは、Open Agent を実行しているマシンに、メディアデバイスとして接続されている必要があります。Android デバイスの USB モードは、**メディアデバイス (MTP)** を設定します。
- Android デバイスまたはエミュレータの画面が、テスト中にロックされないようにしてください。マシンに接続中にデバイスがロックされないようにするには、**開発者向けオプション** を開きます。**スリープモードにしない** または **充電中に画面をスリープにしない** をチェックします。

Silk Test 用に Android エミュレータを設定する

Silk Test を使用して Android エミュレータ上でモバイル アプリケーションをテストする場合、テスト用にエミュレータを設定する必要があります。

1. Android SDK の最新版をインストールします。

Android SDK のインストールと設定についての詳細は、「[Get the Android SDK](#)」を参照してください。

2. Android Studio 2 をインストールします。



ヒント: Android Studio 2 のインストールをスキップして、Android SDK で提供されるエミュレータを使用することもできます。ただし、エミュレータのパフォーマンスを向上させるため、Android Studio 2 をインストールすることを、Micro Focus では推奨しています。このトピックの以下の手順では、Android Studio 2 がインストールされていることを前提としています。

3. Android Studio 2 から **AVD Manager** を開きます。

4. **Create Virtual Device** をクリックします。

5. 仮想デバイスを選択します。

6. **Next** をクリックします。

7. Google API を含む Android のシステム イメージをダウンロードして選択します。

8. **Next** をクリックします。

9. 要件に従って仮想デバイスを設定します。

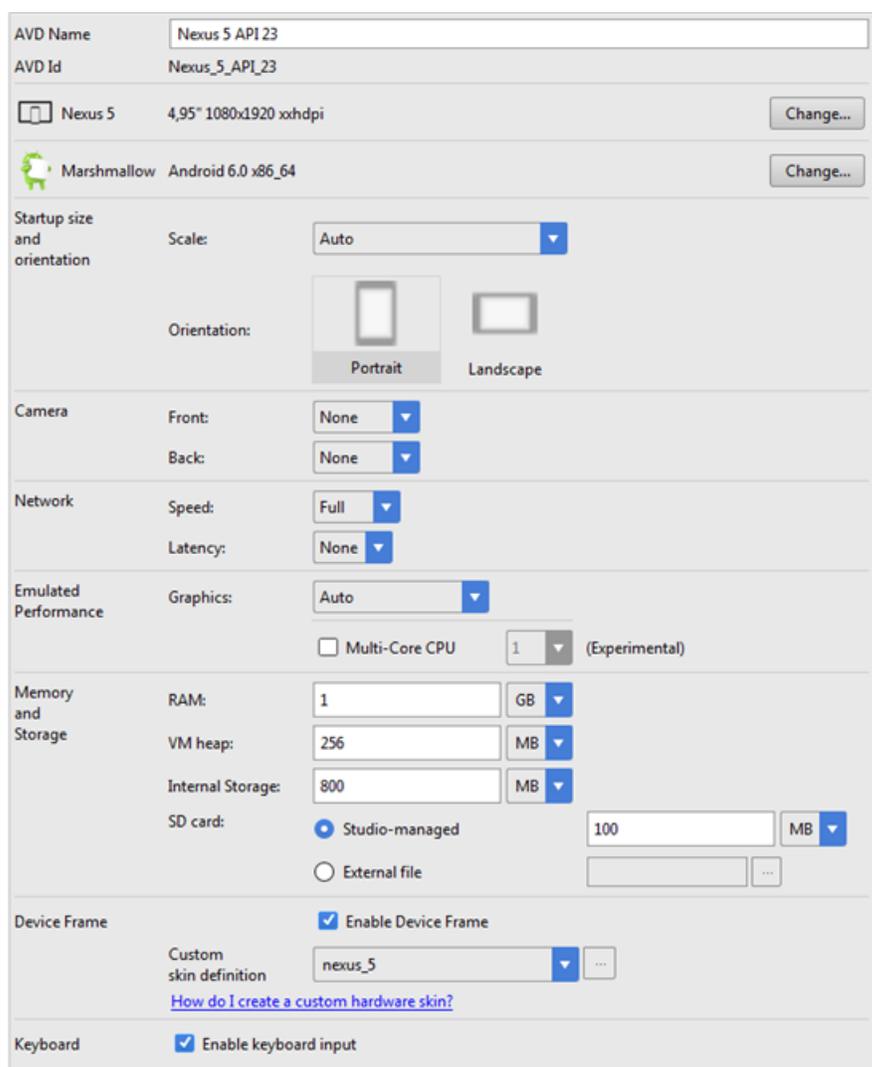
- 10 **Show Advanced Settings** をクリックします。

- 11 エミュレータが使用する RAM サイズとヒープ領域を対象のマシンで管理可能な量に調整します。



ヒント: 最低 1 GB の RAM と 256 MB のヒープ領域を使用することを、Micro Focus では推奨しています。

- 12 **Emulated Performance** 領域のリストで **Auto** を選択します。



13 **Finish** をクリックします。

iOS

Silk Test では、iOS デバイスまたは iOS シミュレータ上のモバイル アプリケーションをテストすることができます。

iOS 上のモバイル アプリケーションのテストにおける前提条件

iOS デバイスや iOS シミュレータ上のモバイル アプリケーションをテストする前に、次の前提条件を満たしていることを確認してください。

- Silk Test を実行するマシンに Java インストールされており、Java へのパスが *Path* 環境変数に追加されていることを確認してください。Silk Test は、JRE または JDK のいずれかをモバイル テストで必要とします。Java がインストールされていない場合は、`C:\Program Files (x86)\Silk\SilkTest\ng\jre\bin` を *Path* に追加し、マシンを再起動してください。
- iOS デバイスが Mac に接続されているか、または目的の iOS シミュレータ イメージが Mac 上にインストールされている。
- Information Service が Mac 上にインストールされている。詳細については、「[Silk Test Information Service を Mac にインストールする](#)」を参照してください。

- サポートするバージョンの Xcode が Mac 上にインストールされている。サポートする Xcode のバージョンについての情報は、『[リリースノート](#)』を参照してください。
- Windows マシン上に Silk Test がインストールされている。
- Mac が Windows マシンと同じネットワークにあり、Windows マシンにリモート ロケーションとして追加されている。
- iOS デバイス上でネイティブ モバイル アプリをテストする場合は、Developer Account でサインされたアプリの .ipa ファイルが作成されている。詳細については、『[iOS アプリのテストの準備](#)』を参照してください。
- iOS シミュレータ上でネイティブ モバイル アプリをテストする場合は、ZIP 形式に圧縮したアプリが作成されている。詳細については、『[iOS シミュレータ上のネイティブ モバイル アプリケーションのテスト](#)』を参照してください。
- iOS デバイスと iOS シミュレータの両方でネイティブ モバイル アプリをテストする場合は、サインした .ipa ファイルと ZIP した .app ディレクトリの両方が同じフォルダに作成されている。
- ネイティブ モバイル アプリをテストする場合は、iOS デバイスの ID がアプリをサインするのに使用した Developer Profile に関連付けられている。
- モバイル Web アプリケーションをテストする場合は、Xcode を使用して SafariLauncher.app がビルドされている。アプリのビルドに必要なファイルは、/Applications/Silk/Mobile/osx/Appium/node_modules/appium/node_modules/appium-ios-driver/node_modules/safari-launcher にあります。ビルドした app ファイルは、/Applications/Silk/Mobile/osx/Appium/node_modules/appium/node_modules/appium-ios-driver/build/SafariLauncher に保存する必要があります。
- iOS デバイスのテスト中に、スリープ モードに移行してはいけません。画面ロックとパスワードをオフにするには、**設定 > 一般 > パスコードロック** を選択します。
- テスト中に Mac の画面がオフにならないようにしてください。オフになると、**再生ステータス** ダイアログ ボックスに何も表示されなくなります。
- iOS シミュレータ上でモバイル アプリケーションをテストする場合は、iOS シミュレータがインストールされている Mac は、テスト中にディスプレイがスリープ モードにならないようにする必要があります。
- 物理 iOS デバイス上のネイティブ モバイル アプリをテストする場合は、デバイスで UI オートメーションが有効化されている。詳細については、『[iOS デバイスのテストの準備](#)』を参照してください。
- 物理 iOS デバイス上で Apple Safari を使用してのモバイル Web アプリケーションをテストする場合は、**Web インスペクタ** が有効化されている。詳細については、『[iOS デバイスのテストの準備](#)』を参照してください。
- Lightning コネクタを持つ iOS デバイスを使用することを Micro Focus は推奨しています。これは、Silk Test では、Lightning ケーブルで Mac に接続していない iOS デバイスのデバイス画面のライブ ビュー表示をサポートしていないためです。

物理 iOS デバイス上のネイティブ モバイル アプリケーションのテスト



注: Silk Test を使用してネイティブ モバイル アプリケーションやハイブリッド アプリケーションをテストするには、ネイティブ モバイル ライセンスが必要です。詳細については、『[ライセンス情報](#)』を参照してください。

物理 iOS デバイス上のネイティブ モバイル アプリケーション (アプリ) やハイブリッド アプリケーションをテストするには、次のタスクを実行します。

1. テストする iOS デバイスを準備します。
詳細については、『[iOS デバイスのテストの準備](#)』を参照してください。
2. テストするアプリを準備します。
詳細については、『[iOS アプリのテストの準備](#)』を参照してください。
3. iOS デバイスが接続されている Mac を、Silk Test がインストールされている Windows マシンに、リモート ロケーションとして追加します。
詳細については、『[リモート ロケーションの編集](#)』を参照してください。

 **注:** 任意の時点で、Mac の各ユーザーは、Mac に接続されている 1 つの iOS デバイス、または、Mac 上で実行中の 1 つの iOS シミュレータに対してのみテストを実行できます。同じ Mac 上の複数の iOS デバイスや iOS シミュレータに対してテストを実行するには、Mac のユーザー セッションが各デバイスとシミュレータに対して必要となります。詳細については、「[同一 Mac 上の複数の iOS シミュレータでのテスト](#)」を参照してください。

4. モバイル アプリケーション用の Silk Test プロジェクトを作成します。
5. モバイル アプリケーション用のテストを作成します。
6. テストで実行する操作を記録します。記録 ウィンドウを開始すると、**アプリケーションの選択** ダイアログ ボックスが開きます。
7. **モバイル** タブを選択します。
8. アプリをテストするモバイル デバイスをリストから選択します。
9. **参照** をクリックしてアプリ ファイルを選択するか、アプリ ファイルへの完全パスを **モバイル アプリ ファイル** テキスト フィールドに入力します。
このパスでは、Silk Test は HTTP および UNC 形式をサポートします。
Silk Test は、モバイル デバイス上に指定したアプリをインストールします。
- 10 **OK** をクリックします。
iOS デバイスやシミュレータのテスト中に、スリープ モードに移行してはいけません。画面ロックとパスワードをオフにするには、**設定 > 一般 > パスコードロック** を選択します。
11. すべての操作の記録を終えたら、記録を停止します。
12. テストを再生します。
13. テスト結果を分析します。

 **注:** iOS デバイスと iOS シミュレータの両方でネイティブ モバイル アプリをテストする場合は、サインした .ipa ファイルと ZIP した .app ディレクトリの両方が同じフォルダに作成されている。

iOS シミュレータ上のネイティブ モバイル アプリケーションのテスト

 **注:** Silk Test を使用してネイティブ モバイル アプリケーションやハイブリッド アプリケーションをテストするには、ネイティブ モバイル ライセンスが必要です。詳細については、「[ライセンス情報](#)」を参照してください。

iOS シミュレータ上のネイティブ モバイル アプリケーション (アプリ) やハイブリッド アプリケーションをテストするには、次のタスクを実行します。

1. アプリの Xcode プロジェクトで、iOS シミュレータ用にアプリをコンパイルします。
Xcode UI からでも、コマンドラインからでもアプリをコンパイルできます。たとえば、iOS 9.2 で iOS シミュレータ用のアプリをコマンドラインでコンパイルするには、次のコマンドを実行します。

```
xcodesbuild -sdk iphonesimulator9.2
```
2. アプリの .app ディレクトリを .zip ファイルに Zip します。
デフォルトでは、.app ディレクトリは、~/Library/Developer/Xcode/DerivedData ディレクトリにあります。Xcode で **File > Project Settings** をクリックすれば、ディレクトリがある場所を確認できます。
3. iOS シミュレータがインストールされている Mac を、Silk Test がインストールされている Windows マシンに、リモート ロケーションとして追加します。
詳細については、「[リモート ロケーションの編集](#)」を参照してください。

 **注:** Mac にインストールされている 1 つの iOS シミュレータでのみテストを実行できます。複数の Silk Test ユーザーが、同じ Mac にインストールされている複数の iOS シミュレータ上で同時にテストを実行することはできません。

4. モバイル アプリケーション用の Silk Test プロジェクトを作成します。

5. モバイル アプリケーション用のテストを作成します。
6. テストで実行する操作を記録します。記録 ウィンドウを開始すると、**アプリケーションの選択** ダイアログ ボックスが開きます。
7. **モバイル** タブを選択します。
8. リストから iOS シミュレータを選択します。
9. **参照** をクリックして Zip したアプリ ファイルを選択するか、Zip したアプリ ファイルへの完全パスを **モバイル アプリ ファイル** テキスト フィールドに入力します。
このパスでは、Silk Test は HTTP および UNC 形式をサポートします。
Silk Test は、iOS シミュレータ上に指定したアプリをインストールします。
- 10 **OK** をクリックします。
iOS デバイスやシミュレータのテスト中に、スリープ モードに移行してはいけません。画面ロックとパスワードをオフにするには、**設定 > 一般 > パスコードロック** を選択します。
11. すべての操作の記録を終えたら、記録を停止します。
12. テストを再生します。
13. テスト結果を分析します。

 **注:** iOS デバイスと iOS シミュレータの両方でネイティブ モバイル アプリをテストする場合は、サインした .ipa ファイルと ZIP した .app ディレクトリの両方が同じフォルダに作成されている。

物理 iOS デバイス上のモバイル Web アプリケーションのテスト

物理 iOS デバイス上のモバイル Web アプリケーションをテストするには、次のタスクを実行します。

1. テストする iOS デバイスを準備します。
詳細については、「[iOS デバイスのテストの準備](#)」を参照してください。
2. iOS デバイスが接続されている Mac を、Silk Test がインストールされている Windows マシンに、リモート ロケーションとして追加します。
詳細については、「[リモート ロケーションの編集](#)」を参照してください。
 **注:** 任意の時点で、Mac の各ユーザーは、Mac に接続されている 1 つの iOS デバイス、または、Mac 上で実行中の 1 つの iOS シミュレータに対してのみテストを実行できます。同じ Mac 上の複数の iOS デバイスや iOS シミュレータに対してテストを実行するには、Mac のユーザー セッションが各デバイスとシミュレータに対して必要となります。詳細については、「[同一 Mac 上の複数の iOS シミュレータでのテスト](#)」を参照してください。
3. モバイル アプリケーション用の Silk Test プロジェクトを作成します。
4. モバイル アプリケーション用のテストを作成します。
5. テストで実行する操作を記録します。記録 ウィンドウを開始すると、**アプリケーションの選択** ダイアログ ボックスが開きます。
6. モバイル Web アプリケーションをテストするには：
 - a) **Web** タブを選択します。
 - b) 使用するモバイル ブラウザーを選択します。
 - c) **移動する URL の入力** テキスト ボックスに、開く Web ページを指定します。
7. **OK** をクリックします。
iOS デバイスやシミュレータのテスト中に、スリープ モードに移行してはいけません。画面ロックとパスワードをオフにするには、**設定 > 一般 > パスコードロック** を選択します。
8. すべての操作の記録を終えたら、記録を停止します。
9. テストを再生します。
10. テスト結果を分析します。

iOS シミュレータ上のモバイル Web アプリケーションのテスト

iOS シミュレータ上のモバイル Web アプリケーションをテストするには、次のタスクを実行します。

1. iOS デバイスが接続されている Mac を、Silk Test がインストールされている Windows マシンに、リモートロケーションとして追加します。

詳細については、「[リモートロケーションの編集](#)」を参照してください。



注: 任意の時点で、Mac の各ユーザーは、Mac に接続されている 1 つの iOS デバイス、または、Mac 上で実行中の 1 つの iOS シミュレータに対してのみテストを実行できます。同じ Mac 上の複数の iOS デバイスや iOS シミュレータに対してテストを実行するには、Mac のユーザーセッションが各デバイスとシミュレータに対して必要となります。詳細については、「[同一 Mac 上の複数の iOS シミュレータでのテスト](#)」を参照してください。

2. モバイル アプリケーション用の Silk Test プロジェクトを作成します。
3. モバイル アプリケーション用のテストを作成します。
4. テストで実行する操作を記録します。記録 ウィンドウを開始すると、**アプリケーションの選択** ダイアログ ボックスが開きます。
5. モバイル Web アプリケーションをテストするには：
 - a) **Web** タブを選択します。
 - b) 使用するモバイルブラウザを選択します。
 - c) **移動する URL の入力** テキスト ボックスに、開く Web ページを指定します。
6. **OK** をクリックします。

iOS デバイスやシミュレータのテスト中に、スリープ モードに移行してはいけません。画面ロックとパスワードをオフにするには、**設定 > 一般 > パスワードロック** を選択します。
7. すべての操作の記録を終えたら、記録を停止します。
8. テストを再生します。
9. テスト結果を分析します。

iOS 上のハイブリッド アプリケーションのテスト

ハイブリッド アプリケーション (アプリ) は、デバイス上で実行されるネイティブ アプリケーションのようなアプリですが、HTML5、CSS、JavaScript などの Web テクノロジーを使用して記述されたアプリです。

Silk Test は、ネイティブ コンテナに埋め込まれた単一の Web ビューで構成されたハイブリッド アプリのテストをサポートします。このようなハイブリッド アプリの一般的な例は、Apache Cordova アプリケーションです。

iOS 上のハイブリッド アプリをテストする手順は、モバイル ネイティブ アプリケーションをテストする手順と同じです。詳細については、「[物理 iOS デバイス上のネイティブ モバイル アプリケーションのテスト](#)」または「[iOS シミュレータ上のネイティブ モバイル アプリケーションのテスト](#)」を参照してください。

iOS デバイス上のハイブリッド アプリをテストする前に、デバイスで **Web インスペクタ** が有効化されていることを確認してください。

iOS デバイスのテストの準備



注: Silk Test を使用してネイティブ モバイル アプリケーションやハイブリッド アプリケーションをテストするには、ネイティブ モバイル ライセンスが必要です。詳細については、「[ライセンス情報](#)」を参照してください。

モバイル アプリケーションをテストするために iOS デバイスを準備するには：

1. Mac 上で Xcode を起動します。
2. iOS デバイスを Mac に接続します。
3. iOS デバイスで、**設定 > デベロッパ** をクリックします。



ヒント: デベロッパ メニューが iOS デバイスに表示されていない場合は、デバイスと Mac を再起動します。

4. **Enable UI Automation** をオンにします。
5. Apple Safari 上でモバイル Web アプリケーションをテストするには、**設定 > Safari > 詳細** をクリックします。
6. **Web インスペクタ** をオンにします。

iOS アプリのテストの準備

Silk Test を使用して特定の iOS デバイス上で特定の iOS アプリをテストできるようにするには、次の項目を考慮する必要があります。

- 特定の iOS デバイスに手動でインストールできる iOS アプリに対してのみテストを自動化できます。iOS アプリにサインできるようにするために、*Apple Developer Program* のメンバーシップに登録する必要があります。詳細については、「[メンバーシップの選択](#)」を参照してください。



注: App Store で配布するように作成された iOS アプリや、任意の iOS デバイスに手動でインストールできるアプリを自動的にテストできません。

- 特定の iOS デバイスで iOS アプリをインストールして実行する前に、iOS デバイスを Apple Developer アカウントを使用して登録する必要があります。
- iOS デバイスに iOS アプリをインストールするには、Xcode を使用してアプリの IPA ファイルを作成する必要があります。特定の iOS デバイスでのテスト用に IPA ファイルを作成するには、Apple Developer Program のメンバーとして Xcode の *Archive* 機能を使用します。この機能には 2 つのオプションがあります。
 - *Apple Developer Enterprise Program* のメンバーの場合、**Save for Ad Hoc Deployment** オプションを使用できます。
 - Apple Developer Enterprise Program のメンバー以外の Apple Developer Program のメンバーの場合、**Save for Development Deployment** オプションを使用できます。

詳細については、「[Exporting Your App for Testing \(iOS, tvOS, watchOS\)](#)」を参照してください。

iOS シミュレータ上の特定の iOS アプリを Silk Test を使用してテストできるようにするには、Xcode を使用して iOS アプリの Zip ファイルを作成してから iOS シミュレータ上にインストールします。詳細については、Xcode のドキュメントを参照してください。

Silk Test Information Service を Mac にインストールする



注: Information Service を Mac にインストールするには、Mac の管理者権限が必要です。

Mac 上の Apple Safari や、Mac に接続されている iOS や Android デバイス上のモバイル アプリケーションに対するテストを作成して実行するには、Mac に Silk Test Information Service (Information Service) をインストールしてから、**リモート ロケーション** ダイアログ ボックスを使用して、Silk Test をインストールした Windows マシンと Mac を接続する必要があります。

Information Service を Mac にインストールするには :

1. Java JDK が Mac 上にインストールされていることを確認します。
2. iOS デバイス上でモバイル アプリケーションをテストする場合は、Xcode が Mac 上にインストールされていることを確認します。

3. Information Service セットアップ ファイル (SilkTestInformationService<バージョン>-<ビルド番号>.pkg にアクセスします。
 - Silk Test のインストール時に Information Service セットアップ ファイルをダウンロードした場合は、Silk Test インストール ディレクトリ (C:¥Program Files (x86)¥Silk¥SilkTest など) の macOS フォルダを開きます。
 - Silk Test のインストール時に Information Service セットアップ ファイルをダウンロードしなかった場合は、[Micro Focus SupportLine](#) からセットアップ ファイルをダウンロードできます。
4. SilkTestInformationService<バージョン>-<ビルド番号>.pkg ファイルを Mac にコピーします。
5. SilkTestInformationService<バージョン>-<ビルド番号>.pkg を実行して、Information Service をインストールします。
6. インストール ウィザードの指示に従います。
7. パスワードを尋ねられた場合、現在サインインしている Mac ユーザーのパスワードを入力します。
8. Apple Safari が開き、SafariDriver を信頼するかどうかを尋ねるメッセージ ボックスが表示されたら、**信頼** をクリックします。

インストールを完了するために、現在の Mac ユーザーをログアウトします。Information Service が正しくインストールされていることを確認するには、Mac にログインし、画面の右上隅にある Silk Test アイコンをクリックして、利用可能なデバイスとブラウザを表示させます。

 **ヒント:** Silk Test アイコンが表示されない場合は、Mac を再起動してください。

Silk Test Information Service を Mac からアンインストールする

Mac 上の Apple Safari に対するテストを実行する必要がなくなった場合など、次の手順で Silk Test Information Service を Mac からアンインストールすることができます。

1. uninstallInfoService.sh のような新しいシェル ファイルを作成します。
2. 新しいファイルに以下のコードを入力します。

```
#!/bin/sh

if launchctl list | grep com.borland.infoservice ; then
    launchctl unload /Library/LaunchAgents/com.borland.infoservice.plist
    echo "unloading Launch Daemon"
fi

if [ -d "/Applications/Silk" ]
then
    sudo rm -rf /Applications/Silk
fi

if [ -f "/Library/LaunchAgents/com.borland.infoservice.plist" ]
then
    sudo rm /Library/LaunchAgents/com.borland.infoservice.plist
fi

if [ -f "/usr/local/bin/ideviceinstaller" ]
then
    sudo rm /usr/local/bin/ideviceinstaller
fi

exit 0
```

3. コマンドラインで `chmod +x uninstallInfoService.sh` を実行し、シェル ファイルの実行可能にします。

4. コマンドラインからシェル ファイルを実行します。

iOS デバイスの推奨設定

Silk Test を使用したテストを最適化するために、テストしたい iOS デバイスで次の設定を行ってください。

- 実際にユーザーが行った操作をテストに反映させるために、Apple Safari の自動入力とパスワードの保存を無効化します。設定 > Safari > パスワードと自動入力をタップし、ユーザー名とパスワード 設定をオフにします。
- iOS デバイスのテスト中に、スリープ モードに移行してはいけません。画面ロックとパスワードをオフにするには、設定 > 一般 > パスコードロック を選択します。

同じ Mac 上での複数の iOS デバイスまたはシミュレータのテスト

デフォルトでは、Silk Test は同じ Mac に接続されている複数の iOS デバイスまたは iOS シミュレータのテストをサポートしません。いかなるときでも、ユーザーは Mac に接続されている 1 台の iOS デバイスまたは 1 つの iOS シミュレータ上でのみテストを実行できます。しかし、Mac 上で複数のユーザー セッションを作成すると、Mac に接続されている複数の iOS デバイス、または Mac で実行されている複数の iOS シミュレータ上で並列にテストを実行できます。

1. iOS デバイス、または iOS シミュレータそれぞれに対して、Mac 上の管理者権限をもつユーザーを作成します。
たとえば、Mac に接続されている 2 台の iOS デバイスと、Mac で実行されている 2 つの iOS シミュレータ上でテストを実行するには、管理者権限を持つ 4 つのユーザー セッションが Mac で必要になります。
2. それぞれのユーザーに対して、Silk Test Information Service をインストールします。
Information Service のインストールに関する詳細は、「[Silk Test Information Service を Mac にインストールする](#)」を参照してください。
3. それぞれのユーザーで、.silk/silktest/conf/ ディレクトリに移動し、infoservice.properties.sample ファイルの名前を infoservice.properties に変更します。
4. それぞれのユーザーで、infoservice.properties ファイルを編集して、使用されていないポートを指定します。
デフォルトのポートは 22901 です。通常、それぞれのユーザーに対してデフォルトのポート番号から 1 つずつ増加させた値を指定します。
たとえば、4 つのユーザー セッションに対して、22901、22902、22903、22904 のポートを割り振ります。
5. それぞれのユーザーで、一旦ログアウトし、再びログインすることで、指定したポートで Information Service を再起動します。



ヒント: ユーザー セッションに対応した Information Service を使用するには、そのユーザーが Mac にログインしている必要があります (Mac を再起動した後など)。

インストール済みアプリのテスト

デバイス、エミュレーター、またはシミュレータ上に既にインストールされているネイティブ モバイル アプリをテストするには、接続文字列でアプリを指定します。

1. ネイティブ モバイル アプリをテストする既存のプロジェクトを開きます。
2. **アプリケーション構成の編集** ダイアログ ボックスを開きます。
3. 次のいずれかの方法で、既存のアプリの指定部分を接続文字列で置き換えます。

- iOS アプリをテストする場合は、`bundleId` を使ってアプリを指定します。たとえば、`app=MyApp.ipa` を `bundleId=silktest.InsuranceMobile` で置き換えます。
- Android アプリをテストする場合は、`appActivity` と `appPackage` を使用してアプリを指定します。たとえば、`app=MyApp.apk` を `appActivity=.LoginActivity;appPackage=silktest.insurancemobile` で置き換えます。

詳細については、「[接続文字列](#)」を参照してください。

モバイル アプリケーションの記録

Silk Test とモバイル デバイスまたはエミュレータとの間の接続が一旦確立すると、デバイス上で実行する操作を記録できます。モバイル アプリケーションを記録するために、Silk Test では次の機能を持つ **記録** ウィンドウを使用します。

- テストするモバイル デバイスまたは Android エミュレータの画面を表示します。
- **記録** ウィンドウで操作を実行すると、モバイル デバイス上でも同じ操作が実行されます。
- 画面上のコントロールを操作すると、**記録** ウィンドウはデフォルトの操作を事前に選択します。
 - デフォルトの操作が Click の場合、コントロール上で左クリックすると、その操作が実行されます。右クリックすると、コントロールに対して利用可能な操作のリストが表示されます。この場合は、実行する操作を選択し、**OK** をクリックします。
 - デフォルトの操作が Click ではない場合、コントロールに対して有効なすべての操作がリストで表示されるので、実行したい操作を選択するか、単に **OK** をクリックして事前に選択された操作を受け入れます。

リストから操作を選択する場合、選択した操作のパラメータの値をパラメータ フィールドに入力することができます。Silk Test は自動的にパラメータを検証します。

- 記録を一時停止すると、画面上での操作は記録されないため、デバイスを記録を続けたい状態に変更することができます。
- 記録を停止すると、記録した操作でスクリプトが生成されるため、続いてテストの再生を行うことができます。

テストを再生するモバイル デバイスの選択

テストを再生するために使用するモバイル デバイスを定義できます。

- スクリプトをコマンド ラインや CI サーバーから実行する場合は、スクリプトのアプリケーション構成で接続文字列を指定します。
- Silk Central からテストを実行する場合、接続文字列を指定する代わりに、Silk Central の実行定義の **配置** タブにある **モバイル デバイスの選択** 領域でモバイル デバイスを指定します。詳細については、『[Silk Central ヘルプ](#)』を参照してください。

デバイス プールなどがある場合に、特定のモバイル デバイスを指定したり、利用可能なデバイス群のサブセットをフィルタするために接続文字列を使用できます。最初に一致したデバイスが再生に使用されません。特に指定がない場合には、次のルールに従って一致したモバイル デバイスが使用されます（高い優先度順）。

- リモート ロケーションに接続されたモバイル デバイスよりも、ローカル マシンに接続されたモバイル デバイスが優先されます。
- ブラウザーの種類が接続文字列で指定されている場合、古いバージョンのブラウザよりも、新しいバージョンのブラウザが優先されます。
- 古いプラットフォームよりも、新しいプラットフォームが優先されます。
- 物理デバイスがエミュレータやシミュレータよりも優先されます。
- アルファベット順で後者のデバイス名のデバイスが優先されます。たとえば、"iphone 5" という名前のデバイスよりも、"iphone 6" という名前のデバイスが優先されます。

例：リモートマシンに接続されている Android デバイス上のアプリに対する接続文字列

リモートマシンに接続されている Android デバイス上で MyApp.apk アプリをテストするには、接続文字列は次のようになります。

```
"platformName=Android;deviceName=MotoG3;host=http://10.0.0.1;app=MyApp.apk"
```

例：Mac の iOS シミュレータ上のアプリに対する接続文字列

```
"platformName=iOS;platformVersion=9.2;deviceName=iPhone6;host=10.0.0.1;app=MyApp.ipa;isSimulator=true"
```

モバイルデバイスの操作

モバイルデバイス进行操作したり、テスト対象アプリケーションでスワイプのような操作を実行するには、次の手順を実行します。

1. **記録** ウィンドウで、**モバイルデバイス操作の表示** をクリックします。モバイルデバイスに対して実行できるすべての操作がリストされます。
2. リストからリストから実行したい操作を選択します。
3. Android デバイスまたはエミュレータで、スワイプを記録するには、マウスの左ボタンを押したままマウスを動かします。
4. テストの記録を続行します。

モバイルデバイスの開放

モバイルデバイスに対するテストを記録または再生する場合、Open Agent インスタンスはそのデバイスの所有権を確保します。これによって、Open Agent は、他の Silk Test ユーザーがそのデバイスを使用することを防止します。デバイス上でのテストの記録や再生が終了した後に、他の Silk Test ユーザーがデバイスを使用できるようにするために、Silk Test クライアントが閉じたとき、無人のテストプロセスが完了したとき、または Open Agent が閉じたときに、Silk Test は自動的にデバイスを開放します。また、デバイスを手動で開放することもできます。



注: モバイルデバイスを開放すると、モバイルデバイス上のテスト対象アプリケーション (AUT) は閉じられます。

記録後のモバイルデバイスの開放

他の Silk Test ユーザーがモバイルデバイスでテストできるようにするために、記録後にデバイスを開放します。

記録が完了した後にモバイルデバイスを開放するには、次のいずれかを実行します。

- システムトレイから Open Agent を停止する。
- Silk Test を閉じる。並列テストが有効な場合には、この操作でのみデバイスが開放されます。



注: モバイルデバイスを開放すると、モバイルデバイス上のテスト対象アプリケーション (AUT) は閉じられます。

再生後のモバイル デバイスの開放

他の Silk Test ユーザーがモバイル デバイスでテストできるようにするために、再生後にデバイスを開放します。

再生が完了した後に手動でモバイル デバイスを開放するために、次のいずれかを実行することもできます。

- モバイル Web アプリケーションのテストを再生した場合、BrowserApplication クラスの Close メソッド、または CloseSynchron メソッドを使用します。これらのメソッドの詳細については、API ドキュメントを参照してください。

```
webBrowser.close();
```

- モバイル ネイティブ アプリケーションのテストを再生した場合、MobileDevice クラスの CloseApp メソッドを使用します。

たとえば、次のように入力します。

```
MobileDevice mobileDevice = desktop.find("//MobileDevice");  
mobileDevice.closeApp();
```

- `desktop.detachAll()` ステートメントをテスト スクリプトに追加します。

次の条件を満たしている場合は、モバイル デバイスは自動的に開放されます。

- Open Agent を閉じる。
- 無人テスト中にテスト プロセスが停止する。並列テストが有効な場合には、この操作でのみデバイスが開放されます。
- Silk Test を閉じる。並列テストが有効な場合には、この操作でのみデバイスが開放されます。



注: モバイル デバイスを開放すると、モバイル デバイス上のテスト対象アプリケーション (AUT) は閉じられます。

リモート ロケーションの編集

リモート ロケーション ダイアログ ボックスを使用すると、リモート ロケーション上のブラウザやモバイル デバイスを、テストするアプリケーションのセットに追加できます。

1. リモート ロケーション ダイアログ ボックスを開きます。

- Silk Test Workbench を使用している場合、**ツール > リモート ロケーションの編集** をクリックします。
- Silk4J を使用している場合、**Silk4J > リモート ロケーションの編集** をクリックします。
- Silk4NET を使用している場合、**Silk4NET > リモート ロケーションの編集** をクリックします。
- Silk Test Classic を使用している場合、**オプション > リモート ロケーションの編集** をクリックします。

2. リモート ロケーションを追加するには、次の操作を実行します。

- a) **ロケーションの追加** の右側にある矢印をクリックして、Silk Test Information Service を使用してリモート ロケーションを追加するのか、Silk Central を追加するのかを指定します。



注: Silk Central は 1 つだけリモート ロケーションとして設定できます。Silk Central との統合が既に設定されている場合は、リモート ロケーション リストに Silk Central が表示されません。

- b) **ロケーションの追加** をクリックします。**ロケーションの追加** ダイアログ ボックスが表示されます。

- c) Silk Test がリモート マシン上の Information Service に接続するためのリモート ロケーションの URL とポートを、**ホスト** フィールドに入力します。

デフォルトのポートは 22901 です。

- d) 省略可能：**名前** フィールドにリモート ロケーションの名前を入力します。

3. 既存のリモート ロケーションを編集するには、**編集** をクリックします。
4. リモート ロケーションを削除するには、**削除** をクリックします。
5. 省略可能：**アプリケーションの選択** ダイアログに表示されるブラウザーやデバイスの数を減らすには、**このロケーション上のデバイスとブラウザーを非表示** をクリックします。そのリモート ロケーションにインストールされたブラウザーや接続されたデバイスが、**アプリケーションの選択** ダイアログに表示されなくなります。デフォルトでは、すべてのリモート ロケーションにインストールされたブラウザーや接続されたデバイスのすべてが、**アプリケーションの選択** ダイアログに表示されます。
6. **OK** をクリックします。

リモート ロケーションを追加すると、リモート ロケーション上にインストールされているブラウザ (Mac 上の Apple Safari など) が、**アプリケーションの選択** ダイアログ ボックスの **Web** タブで利用可能になり、リモート ロケーションに接続されているモバイル デバイスが、**アプリケーションの選択** ダイアログ ボックスの **モバイル** タブで利用可能になります。

モバイル デバイスの接続文字列

接続文字列 は、テストに使用するモバイル デバイスを指定します。モバイル テストを実行する場合、Silk Test は接続文字列を使用してモバイル デバイスに接続します。接続文字列は、アプリケーション構成の主要な一部です。テスト対象アプリケーションを構成するときに、接続文字列は設定されます。接続文字列を変更するには、**アプリケーション構成の編集** ダイアログ ボックスを使用します。

 **注:** Silk Central からテストを実行する場合、接続文字列を指定する代わりに、Silk Central の実行定義の **配置** タブにある **モバイル デバイスの選択** 領域でモバイル デバイスを指定します。詳細については、『[Silk Central ヘルプ](#)』を参照してください。

デバイス プールなどがある場合に、特定のモバイル デバイスを指定したり、利用可能なデバイス群のサブセットをフィルタするために接続文字列を使用できます。最初に一致したデバイスが再生に使用されません。特に指定がない場合には、次のルールに従って一致したモバイル デバイスが使用されます (高い優先度順)。

- リモート ロケーションに接続されたモバイル デバイスよりも、ローカル マシンに接続されたモバイル デバイスが優先されます。
- ブラウザーの種類が接続文字列で指定されている場合、古いバージョンのブラウザーよりも、新しいバージョンのブラウザーが優先されます。
- 古いプラットフォームよりも、新しいプラットフォームが優先されます。
- 物理デバイスがエミュレータやシミュレータよりも優先されます。
- アルファベット順で後者のデバイス名のデバイスが優先されます。たとえば、"iphone 5" という名前のデバイスよりも、"iphone 6" という名前のデバイスが優先されます。

次のコンポーネントが接続文字列で使用できます。

コンポーネント	説明
deviceName	モバイル デバイスの名前。物理モバイル デバイス上でテストをする場合、デバイス ID を代わりに使用します。ワイルドカードをサポートします。大文字と小文字は区別されません。
platformName	Android または iOS。必須。
deviceId	省略可能：モバイル デバイスの ID。物理モバイル デバイスでテストをする場合に、デバイス名の代わりに使用します。ワイルドカードをサポートします。大文字と小文字は区別されません。
platformVersion	省略可能：Android または iOS のバージョン。特定の Android または iOS のバージョンのモバイル デバイス上でのみテストする場合に、バージョンを指定します。ワイルドカードをサポートします。大文字と小文字は区別されません。
browserVersion	省略可能：特定のブラウザーのバージョンでのみテストする場合に、ブラウザーの種類と共に使用します。ワイルドカードをサポートします。大文字と小文字は区別されません。

コンポーネント	説明
host	省略可能：設定しない場合は、任意のリモート ロケーションがホストとして使用されます。ワイルドカードをサポートします。大文字と小文字は区別されません。
<ul style="list-style-type: none"> • app • appActivity • appPackage 	Android 上のネイティブ モバイル アプリケーションのテストでは必須です。アプリへのフルパス、または <code>appActivity</code> と <code>appPackage</code> の組み合わせで指定します。たとえば、 <code>app=MyApp.apk</code> 、 <code>appActivity=.LoginActivity</code> ； <code>appPackage=silktest.insurancemobile</code> などです。
<ul style="list-style-type: none"> • app • bundleId 	iOS 上のネイティブ モバイル アプリケーションのテストでは必須です。アプリへのフルパス、または <code>bundleId</code> で指定します。たとえば、 <code>app=MyApp.ipa</code> 、 <code>bundleId=silktest.InsuranceMobile</code> などです。
noReset	省略可能：ネイティブ モバイル アプリケーションをテストする場合に設定できます。 <code>app</code> が指定されている場合にのみ有効です。テストの前にアプリを再インストールしない場合は <code>True</code> 。テストの前にアプリを再インストールする場合は <code>False</code> を指定します。デフォルト値は、 <code>False</code> です。
isSimulator	省略可能：iOS シミュレータ上でのみテストを実行する場合に指定します。デバイス名を代わりに使用できます。
isPhysicalDevice	省略可能：物理デバイス上でのみテストを実行する場合に指定します。デバイス名を代わりに使用できます。

デバイス プールを使用して、テストで実際に使用されるデバイスを確認するには、`MobileDevice` クラスの `generateConnectionString` メソッドの戻り値を使用できます。

モバイル デバイスまたは Android エミュレータ上のモバイル Web アプリケーションのテスト

モバイル デバイスまたは Android エミュレータ上でモバイル Web アプリケーションをテストする場合、接続文字列は次の要素から構成されます。

1. モバイル デバイス名 (MotoG3 など)、またはデバイス ID (11111111 など)。



注: デバイス ID は可読性に欠けるため、デバイス名がユニークであれば、デバイス名を接続文字列に使用することを Micro Focus では推奨します。

2. プラットフォーム名。
3. ブラウザー バージョン。ブラウザーの種類の設定との組み合わせでのみ使用されます。
4. 特定のリモート マシンの IP アドレスまたはホスト名 (10.0.0.1 など)。**リモート ロケーションの編集** ダイアログ ボックスで指定したリモート ロケーションの名前をホスト名として使用することもできます (`MyRemoteLocation` など)。リモート ロケーション名を使用する場合に、ワイルドカードを使用することもできます。ローカル マシンに接続されている Android デバイスをテストする場合は、ローカル マシンの IP アドレスまたはホスト名を指定します。

例：利用可能な任意の Android デバイスの接続文字列

```
"platformName=Android"
```

例：ローカル マシンに接続されている Android デバイス上のブラウザーに対する接続文字列

ローカル マシンに接続されている Android デバイス上でモバイル ブラウザーをテストするには、接続文字列は次のようになります。

```
"deviceName=MotoG3;platformName=Android;host=localhost"
```

または

```
"platformName=Android;deviceId=11111111;host=localhost"
```

例：リモートマシンに接続されている Android デバイス上のブラウザーに対する接続文字列

リモート Android デバイス上のモバイル ブラウザーをテストするには、接続文字列は次のようになります。

```
"deviceName=MotoG3;platformName=Android;host=10.0.0.1"
```

```
"deviceName=MotoG3;platformName=Android;host=MyRemoteLocation*"
```

例：Mac に接続されている iOS デバイス上のブラウザーに対する接続文字列

リモート iOS デバイス上のモバイル ブラウザーをテストするには、接続文字列は次のようになります。

```
"deviceName=myiPhone6;platformName=iOS;host=10.0.0.1"
```

モバイル デバイスまたは Android エミュレータ上のネイティブ モバイル アプリケーションのテスト

モバイル デバイスまたは Android エミュレータ上でネイティブ モバイル アプリケーションをテストする場合、接続文字列は次の要素から構成されます。

1. モバイル デバイス名 (MotoG3 など)、またはデバイス ID (11111111 など)。



注: デバイス ID は可読性に欠けるため、デバイス名がユニークであれば、デバイス名を接続文字列に使用することを Micro Focus では推奨します。

2. プラットフォーム名。
3. 特定のリモート マシンの IP アドレスまたはホスト名 (10.0.0.1 など)。**リモート ロケーションの編集** ダイアログ ボックスで指定したリモート ロケーションの名前をホスト名として使用することもできます (MyRemoteLocation など)。リモート ロケーション名を使用する場合に、ワイルドカードを使用することもできます。ローカル マシンに接続されている Android デバイスをテストする場合は、ローカル マシンの IP アドレスまたはホスト名を指定します。
4. テストするアプリのファイルの名前または、ファイルが Web サーバー上にある場合には、ファイルの URL。たとえば、C:/MyApp.apk や MyApp.ipa など。
 - Android アプリは、常に .apk ファイルを指定します。
 - 物理デバイス上の iOS アプリは、常に .ipa ファイルを指定します。
 - シミュレータ上の iOS アプリは、ZIP ファイルまたは、app という名前のディレクトリを指定します。

例：リモートマシンに接続されている Android デバイス上のアプリに対する接続文字列

リモート マシンに接続されている Android デバイス上で MyApp.apk アプリをテストするには、接続文字列は次のようになります。

```
"platformName=Android;deviceName=MotoG3;host=http://10.0.0.1;app=MyApp.apk"
```

例：Mac に接続されている iOS デバイス上のアプリに対する接続文字列

リモート マシンに接続されている iOS デバイス上で MyApp.ipa アプリをテストするには、接続文字列は次のようになります。

```
"platformName=iOS;deviceName=MyiPhone;host=http://10.0.0.1;app=MyApp.ipa"
```

iOS シミュレータ上のモバイル Web アプリケーションのテスト

iOS シミュレータ上でモバイル Web アプリケーションをテストする場合、接続文字列は次の要素から構成されます。

1. プラットフォーム名 (iOS)。
2. プラットフォームのバージョン (9.2 など)。
3. モバイル デバイス名 (iPhone6 など)。
4. iOS シミュレータを実行している Mac の IP アドレスまたはホスト名。

例 : Mac の iOS シミュレータ上のブラウザーに対する接続文字列

```
"platformName=iOS;platformVersion=9.2;deviceName=iPhone6;host=10.0.0.1;isSimulator=true"
```

iOS シミュレータ上のネイティブ モバイル アプリケーションのテスト

Mac の iOS シミュレータ上でネイティブ モバイル アプリケーションをテストする場合、接続文字列は次の要素から構成されます。

1. プラットフォーム名 (iOS)。
2. プラットフォームのバージョン (9.2 など)。
3. モバイル デバイス名 (iPhone6 など)。
4. リモート マシンの IP アドレスまたはホスト名 (10.0.0.1 など)。
5. テストするアプリの名前 (MyApp.ipa など)。

例 : Mac の iOS シミュレータ上のアプリに対する接続文字列

```
"platformName=iOS;platformVersion=9.2;deviceName=iPhone6;host=10.0.0.1;app=MyApp.ipa;isSimulator=true"
```

モバイル アプリケーションのテスト時のトラブルシューティング

[アプリケーションの選択] ダイアログ ボックスにモバイル ブラウザーが表示されない理由

Silk Test が、次の何れかが原因でモバイル デバイスまたはエミュレータを認識していない可能性があります。

原因	解決策
エミュレータが実行されていない。	エミュレータを開始します。
Android Debug Bridge (adb) がモバイル デバイスを認識しない。	モバイル デバイスが adb によって認識されているかどうか確認するには： <ol style="list-style-type: none">1. Android SDK をインストールしたフォルダで、Android Debug Bridge (adb) がある場所に移動します。Android SDK がインストールされていない場合、C:\Program Files (x86)\Silk\SilkTest\ng\agent\plugins\com.microfocus.silktest.adb_17.5.0.<ビルド番号>\bin に移動して、Silk Test がインストールした adb を使用できます。

原因	解決策
	<ol style="list-style-type: none"> 2. Shift を押しながら、ファイル エクスプローラ ウィンドウで右クリックします。 3. コマンド ウィンドウをここで開く を選択します。 4. コマンド ウィンドウで、adb devices を入力して、アタッチしたすべてのデバイスのリストを得ます。 5. デバイスがリストされない場合、USB デバッグがデバイスで有効化されていること、および適切な USB ドライバがインストールされていることを確認します。 6. 「adb server is out of date」というエラーが表示される場合、他の adb サーバーが実行中の可能性があります。詳細については、「<i>Open Agent</i> とデバイスとの接続が不安定な場合の対処方法」を参照してください。
デバイスのオペレーティング システムのバージョンを Silk Test がサポートしていない。	サポートするモバイル オペレーティング システムのバージョンについては、 リリース ノート を参照してください。
デバイスの USB ドライバがローカル マシンにインストールされていない。	デバイスの USB ドライバをローカル マシンにインストールしてください。詳細については、「 <i>USB ドライバをインストールする</i> 」を参照してください。
USB デバッグがデバイスで有効化されていない。	USDB デバッグをデバイスで有効化してください。詳細については、「 <i>USB デバッグの有効化</i> 」を参照してください。

URL に移動せずに Silk Test が Chrome for Android で URL を検索する理由

アドレスバーに入力された URL を、Chrome for Android が検索として解釈する場合があります。回避策として、URL に移動するコマンドをスクリプトに手動で追加できます。

adb サーバーが正しく起動しない場合にすべきこと

Android Debug Bridge (adb) サーバーが開始するとき、ローカル TCP ポート 5037 にバインドし、adb クライアントから送信されてくるコマンドをリッスンします。すべての adb クライアントは、ポート 5037 を使用して、adb サーバーと通信します。adb サーバーは、5555 から 5585 の範囲 (エミュレータやデバイスで使用される範囲) で奇数のポートをスキャンしてエミュレータやデバイス インスタンスを探します。adb はこれらのポートの変更を許しません。adb 開始中に問題が発生した場合、これらの範囲のポートの 1 つが、他のプログラムによって既に使用されているかどうか確認します。

詳細については、<http://developer.android.com/tools/help/adb.html> を参照してください。

Open Agent とデバイスとの接続が不安定な場合の対処方法

Android SDK、または Android Debug Bridge (adb) を使用するその他のツールをインストールしている場合、Silk Test が使用する adb サーバー以外のサーバーが実行中の可能性があります。バージョンの異なる adb サーバーが実行中の場合、Open Agent とデバイスとの接続が不安定になったり、接続できない場合があります。

このようなバージョンの不一致によるエラーを避けるには、環境変数 `SILKTEST_ANDROID_HOME` に Android SDK ディレクトリへのパス (C:¥Users¥<ユーザー>¥AppData¥Local¥Android¥android-sdk など) を指定してください。環境変数が設定されていない場合は、Silk Test に付属したバージョンの adb が使用されます。

エラー「メモリの割り当てに失敗しました : 8」が発生する理由

エミュレータを開始しているときに、システムが十分なメモリを割り当てることができない場合に、このエラーが表示されます。以下を行ってみてください。

1. エミュレータのメモリ オプションの RAM サイズを下げる
2. Intel HAXM の RAM サイズを下げる RAM サイズを下げるには、IntelHaxm.exe を再度実行して、**Change** を選択します。
3. **タスク マネージャ** を開き、十分なフリー メモリが利用可能かどうかを確認します。不足している場合、プログラムを閉じてメモリを開放してください。

iOS デバイスのテスト時に「Silk Test は指定したアプリを開始できません」というエラーが発生する理由

このエラーが発生する原因として、以下の理由が考えられます。

原因	解決策
iOS デバイスがデベロッパ モードになっていない。	次の 2 種類の方法のいずれかで、デベロッパ モードを有効化できます。 <ul style="list-style-type: none">• Xcode がインストールされている Mac にデバイスを接続し、テストするアプリをデバイスで開始します。• プロビジョニング プロファイルをデバイスに追加します。<ol style="list-style-type: none">1. Xcode を開きます。2. Window > Devices を選択します。3. iOS デバイスをクリックします。4. Show Provisioning Profiles を選択します。5. プロビジョニング プロファイルを追加します。
デバイスの iOS のバージョンを最近更新した。	<ol style="list-style-type: none">1. Xcode を開きます。2. Window > Devices を選択します。3. Xcode がシンボル ファイルを処理するまで待機します。
UI オートメーションが iOS デバイスで有効化されていない。	<ol style="list-style-type: none">1. 設定 > デベロッパ を選択します。2. Enable UI Automation をオンにします。
Web インスペクタ が iOS デバイスで有効化されていない (モバイル Web アプリケーションのテストの場合)。	<ol style="list-style-type: none">1. 設定 > Safari > 詳細 をクリックします。2. Web インスペクタ をオンにします。
テストするアプリがテストしようとしている iOS デバイスの iOS バージョン用にビルドされていない。	Xcode を使用してデバイスの iOS バージョン用にアプリをビルドします。
ソフトウェア・アップデート ダイアログ ボックスが iOS デバイス上で開いている。	ダイアログ ボックスを閉じ、ソフトウェアの自動アップデートを無効化します。 <ol style="list-style-type: none">1. 設定 > iTunes & App Store > 自動ダウンロード を選択します。2. アップデート をオフにします。

Android デバイスの動的ハードウェア コントロールに戻るボタンだけが表示される理由

テストの開始時に Android デバイスや Android エミュレータの画面がロックされると、デバイスやエミュレータが動的ハードウェア コントロールに **戻る** ボタンだけを表示する場合があります。

この問題を解決するには、Open Agent を停止し、デバイスを再起動してから、デバイスの設定を画面のロックをしないように設定してください。

Android デバイスまたはエミュレータにキーボードが表示されなくなる理由

Unicode 文字列をサポートするために、Silk Test は標準キーボードをカスタム キーボードに置き換えます。そして、テストの完了時に元のキーボードに戻します。テスト中にエラーが発生すると、カスタム キーボードが設定されたまま、元に戻らない場合があります。

この問題を解決するには、**設定 > 言語と入力 > 現在のキーボード** を開き、手動で元のキーボードに戻してください。

テスト中にデバイスが応答しなくなる理由

テストの開始時に、デバイス、エミュレータ、シミュレータの画面がロックされると、Silk Test は画面のロックを解除できず、デバイス、エミュレータ、シミュレータが操作に応答しなくなる場合があります。

この問題を解決するには、Open Agent を停止し、デバイスの設定を画面のロックをしないように設定してください。

Information Service を Mac にインストールできない理由

システム環境設定の **セキュリティとプライバシー** で、**一般** タブの **ダウンロードしたアプリケーションの実行許可** 設定が **Mac App Store と確認済みの開発元からのアプリケーションを許可** (デフォルト値) に設定されている場合、Information Service セットアップを開いているときに次のエラー メッセージが表示されます。

"SilkTestInformationService<バージョン>.pkg" は、開発元が未確認のため開けません。

この問題を解決するには、次のいずれかを行います。

- セットアップ ファイルを右クリックして、**開く** を選択します。警告メッセージが表示されても、ファイルを開くことができます。
- **ダウンロードしたアプリケーションの実行許可** 設定を **すべてのアプリケーションを許可** に設定します。
- ファイルを開いた後、システム環境設定の **セキュリティとプライバシー** の **一般** タブを開き、**このまま開く** をクリックします。

Android アプリの記録時に記録ウィンドウが真っ暗になる理由

金融取引を処理するアプリなど、高レベルなセキュリティを必要とする Android アプリでは、Silk Test がアプリのキャプチャをできないようにするために、**FLAG_SECURE** フラグが設定されている可能性があります。Silk Test は、記録時に Android でバスのスクリーンショットやビデオを必要としますが、テストする Android アプリにこのフラグが設定されていると、**記録** ウィンドウにはデバイスの真っ黒な画面が表示されます。Silk Test でこのようなアプリをテストするには、テスト中に **FLAG_SECURE** フラグを設定しないよう、アプリの開発チームに依頼してください。

Android エミュレータでのテスト時に Silk Test がビデオを表示しない理由

エミュレータがレンダリングにコンピュータのグラフィック カードを使用している場合、Silk Test のビデオキャプチャが機能しない場合があります。この問題を解決するには、ソフトウェアでグラフィックでエミュレートします。

1. **Android Virtual Device Manager** を開きます。
2. エミュレータの **Actions** 列の **Edit** をクリックします。
3. **Virtual Device Configuration** ダイアログ **Emulated Performance** 領域で、リストから **Software** を選択します。

Xcode のインストール バージョンを変更する方法

Xcode の最新版にアップグレードしてしまった場合など、使用している Xcode のバージョンを Silk Test がサポートしていない場合、iOS でテストする際にエラー メッセージが表示される場合があります。

インストールした Xcode のバージョンをサポートするバージョンで置換するには、サポートするバージョンの Xcode を <https://developer.apple.com/download/more/> からダウンロードし、サポートされていないバージョンをダウンロードしたバージョンで置換します。サポートする Xcode のバージョンについての情報は、『リリース ノート』を参照してください。

Mac のディスクの空き容量がなくなった場合の対処

Silk Test は iOS デバイスの自動化に Instruments を使用します。このツールは、/Library/Caches/com.apple.dt.instruments ディレクトリに大きなログ ファイルを生成するため、Mac のディスクの空き容量を圧迫している場合があります。この問題を解決するために、手動であるいは cron ジョブを使用して、これらのログ ファイルを定期的に削除することを Micro Focus は推奨します。たとえば、毎日同じ時間にファイルを削除するには、次の手順を実行します。

1. ターミナルで「`sudo crontab -e`」を入力します。crontab を root として編集できるエディタが開きます。
2. 次の行を crontab に追加します。

```
0 2 1 * * find /Library/Caches/com.apple.dt.instruments -mtime +10 -delete
```

3. crontab を保存します。

この例では、10 日より前のすべてのログ ファイルが、毎日午前 2 時にディレクトリから削除されます。

テストの再生に Chrome for Android を使用する方法

デフォルトでは、**ブラウザーの選択** ダイアログ ボックスを使用して、再生に使用するブラウザーを選択できます。

スクリプトをコマンド ラインや CI サーバーから実行する場合は、スクリプトのアプリケーション構成で接続文字列を指定できます。アプリケーション構成で指定したブラウザーを上書きするには、`silktest.configurationName` 環境変数を使用します。

BrowserApplication クラスの `browsertype` プロパティを使用して、再生に使用するブラウザーの種類を設定することもできます。ただし、`browsertype` は Chrome for Android の明示的な値を含みません。

テストを再生するブラウザーとして Chrome for Android を使用するよう指定するには、`browsertype` に `GoogleChrome` を設定し、Android をプラットフォームとして指定します。Android が指定されると、デスクトップ マシン上の Google Chrome の代わりに Chrome for Android を使用して、Silk Test はテストを実行します。

使用例

次のサンプル コードは、`silktest.configurationName` を使用して Nexus 7 上の Chrome for Android を使用するテストの基本状態を設定する方法を示しています。

```
SET
silktest.configurationName="platformName=Android;deviceName=Nexus
7;host=10.0.0.1 - Chrome"
```

次の Java のサンプル コードは、`browsertype` を使用して Chrome for Android を使用するテストの基本状態を設定する方法を示しています。

```
BrowserBaseState baseState = new
BrowserBaseState(BrowserType.GoogleChrome, "demo.borland.com/
InsuranceWebExtJS/");
baseState.setConnectionString("platformName=Android");
baseState.execute(desktop);
```

モバイル Web アプリケーションのテストにおける制限事項

モバイル ブラウザ上でのテストの再生とロケータの記録のサポートは、サポートされている他のブラウザほど完全なものではありません。以下のリストに、モバイル ブラウザ上でのテストの再生とロケータの記録の既知の制限事項をリストします。

- 次のクラス、インターフェイス、メソッド、プロパティは、モバイル Web アプリケーションでは現時点ではサポートされません。
 - BrowserApplication クラス。
 - CloseOtherTabs メソッド
 - CloseTab メソッド
 - ExistsTab メソッド
 - GetActiveTab メソッド
 - GetSelectedTab メソッド
 - GetSelectedTabIndex メソッド
 - GetSelectedTabName メソッド
 - GetTabCount メソッド
 - ImageClick メソッド
 - OpenTab メソッド
 - SelectTab メソッド
 - DomElement クラス。
 - DomDoubleClick メソッド
 - DomMouseMove メソッド
 - GetDomAttributeList メソッド
 - IKeyable インターフェイス。
 - PressKeys メソッド
 - ReleaseKeys メソッド
- Silk Test は、iOS 上の Apple Safari を使用した HTML フレームおよび iframe のテストをサポートしません。
- 横固定モードでの記録はシステム バーに仮想ボタンを含むエミュレータに対してサポートされません。このようなエミュレータは、回転を正しく検出せずに、横固定モードのシステム バーを画面の下部ではなく画面の右側に配置します。ただし、このようなエミュレータは縦固定モードで記録することができます。
- モバイル アプリケーションに対する XPath 式では、HTML DOM の HTML 属性だけがサポートされます。Silk Test は、XPath 式のプロパティをサポートしません。
- Chrome for Android に対するモバイルアプリケーションのテストでは、Silk Test は、拡大縮小とスクロールをサポートしません。
- BrowserWindow クラスの以下の JavaScript 警告処理メソッドが、Original Android Stock (AOSP) ブラウザー上でのテストでは機能しません。
 - AcceptAlert メソッド
 - DismissAlert メソッド
 - GetAlertText メソッド
 - IsAlertPresent メソッド
- 任意の時点で、Mac の各ユーザーは、Mac に接続されている 1 つの iOS デバイス、または、Mac 上で実行中の 1 つの iOS シミュレータに対してのみテストを実行できます。同じ Mac 上の複数の iOS シミュレータに対してテストを実行するには、Mac のユーザー セッションが各シミュレータに対して必要となります。詳細については、「[同一 Mac 上の複数の iOS シミュレータでのテスト](#)」を参照してください。

- モバイル Web アプリケーションのテストを開始する前に、ブラウザーのタブが開いていないことを確認してください。
- モバイル Web アプリケーションのテスト中に、ブラウザーのタブは 1 つだけ開くことができます。
- Silk Test は、ネイティブ モバイル アプリケーションによって開かれたモバイル Web アプリケーションのテストをサポートしません。

ネイティブ モバイル アプリケーションのテストにおける制限事項

ネイティブ モバイル アプリケーションに対するテストの再生とロケータの記録の既知の制限事項は次の通りです。

- 次のクラス、インターフェイス、メソッド、プロパティは、ネイティブ モバイル アプリケーションでは現時点ではサポートされません。
 - IKeyable インターフェイス。
 - PressKeys メソッド
 - ReleaseKeys メソッド
 - MobileDevice クラス。
 - iOS 上でのネイティブ モバイル アプリケーションのテスト時に、SetLocation メソッドは iOS シミュレータでのみサポートされ、物理 iOS デバイスではサポートされません。
 - Android 6.0 より前のバージョンの Android 上でのネイティブ モバイル アプリケーションのテスト時に、SetLocation メソッドを使用する場合は、**擬似ロケーションを許可** を有効にする必要があります。設定は、Android デバイスまたはエミュレータの設定を開き、**開発者向けオプション** をタップします。
 - Android 6.0 以降上でのネイティブ モバイル アプリケーションのテスト時に、SetLocation メソッドを使用する場合は、**Appium Settings** をアプリとして設定する必要があります。設定は、Android デバイスまたはエミュレータの設定を開き、**開発者向けオプション > 仮の現在地情報アプリを選択** をタップします。そして、**Appium Settings** を選択します。



注: Appium Settings という項目は、Android デバイスまたはエミュレータ上の Appium でテストを既に実行した場合にのみ表示されます。

- 横固定モードでの記録はシステム バーに仮想ボタンを含むエミュレータに対してサポートされません。このようなエミュレータは、回転を正しく検出せずに、横固定モードのシステム バーを画面の下部ではなく画面の右側に配置します。ただし、このようなエミュレータは縦固定モードで記録することができます。
- モバイル アプリケーションに対する XPath 式では、HTML DOM の HTML 属性だけがサポートされます。Silk Test は、XPath 式のプロパティをサポートしません。
- 任意の時点で、Mac の各ユーザーは、Mac に接続されている 1 つの iOS デバイス、または、Mac 上で実行中の 1 つの iOS シミュレータに対してのみテストを実行できます。同じ Mac 上の複数の iOS デバイスや iOS シミュレータに対してテストを実行するには、Mac のユーザー セッションが各デバイスとシミュレータに対して必要となります。詳細については、「[同一 Mac 上の複数の iOS シミュレータでのテスト](#)」を参照してください。
- Silk Test は、Android と iOS の両方とも、ネイティブ モバイル アプリケーションのテスト時にテキスト解決をサポートしません。

テキスト解決は次のメソッドを含みます。

- TextCapture
- TextClick
- TextExists
- TextRectangle
- Silk Test は、複数の Web ビューを持つネイティブ モバイル アプリケーションのテストをサポートしません。

モバイル Web サイトでのオブジェクトのクリック

自動テストの記録と再生中にオブジェクトをクリックするとき、モバイル Web サイトではデスクトップ Web サイトと比較して、次のような困難があります。

- 拡大/縮小率やデバイス ピクセル比が異なる
- さまざまなモバイル デバイスによって画面サイズが異なる
- モバイル デバイス間でのフォントとグラフィックサイズが異なる (通常、デスクトップ ブラウザの Web サイトよりも小さい)。
- さまざまなモバイルデバイスによってピクセル サイズと解像度が異なる

Silk Test は、このような困難をものともせず、モバイル Web サイトの適切なオブジェクトをクリックできます。

モバイル デバイスでテストを記録するときに、Silk Test は Click の記録時に座標を記録しません。ただし、クロスブラウザ テストの場合、再生中に座標が許されています。また、Click に座標を手動で追加することもできます。Silk Test は、これらの座標をオブジェクトの HTML 座標として解釈します。モバイル デバイスのテストの再生時に BrowserWindow の内側の適切なオブジェクトをクリックするために、Silk Test はオブジェクトの HTML 座標に現在の拡大/縮小率を適用します。デバイスのピクセル座標は、オブジェクトの HTML 座標に現在の拡大/縮小率をかけた座標です。

モバイル Web サイトの現在表示されている領域にオブジェクトが表示されていない場合、Silk Test は Web サイトの適切な位置にスクロールします。

例

HTML ページで 100 x 20 ピクセルの固定サイズの DomButton をテストするコードを以下に示します。

異なるモバイル デバイスまたは異なる拡大/縮小率で再生すると、たとえば DomButton は、デバイス画面上では実際は 10 ピクセルの幅かもしれません。しかし、現在の拡大/縮小率の影響は受けず、上記のコードを使用したときに Silk Test は要素の中央をクリックします。これは、Silk Test が座標を HTML 座標として解釈し、現在の拡大/縮小率を適用するためです。

既存のモバイル Web テストの使用方法

Silk Test 17.0 以降では、モバイル Web テストの扱いが、前のバージョンの Silk Test とは異なります。この変更により、以前のモバイル Web テストが Silk Test 17.0 以降では動作しなくなる可能性があります。このトピックでは、Silk Test 17.0 で行われた変更について説明し、既存のモバイル Web テストを Silk Test 17.0 以降で使用できるように変更する方法を説明します。

Silk Test 17.0 でモバイル Web テストに対して行われた変更は、以下の通りです。

- Silk Test の以前のバージョンでは、Windows マシンに USB で接続された iOS デバイスをテストすることができました。Silk Test 17.0 以降では、OS X マシン (Mac) に接続された iOS デバイスに対してのみテストすることができます。
- 以前のバージョンの Silk Test を使用して Android デバイス上のモバイル Web アプリケーションをテストしていた場合、Silk Test 17.0 以降で Web アプリケーションをテストするには、Android デバイスのプロキシ設定を手動で削除する必要があります。Silk Test 17.0 以降では、プロキシは使用しません。プロキシが設定されていると、「プロキシ サーバーに接続できません」というメッセージがデバイスに表示されます。

索引

A

Android

- USB デバッグの有効化 6
- USB ドライバのインストール 6
- インストール済みアプリ、テスト 15
- エミュレータを設定する 7
- 推奨設定 7
- テスト 4
- デバイスの開放 17
- デバイスの開放、記録 17
- デバイスの開放、再生 18
- トラブルシューティング 22
- ハイブリッド アプリケーション 5
- モバイル Web アプリケーション、前提条件 4
- モバイル ネイティブ アプリケーション、前提条件 4

Android エミュレータ

構成 7

Apple Safari

Information Service、インストール 13, 14

Apple Safari のテスト

Information Service、インストール 13, 14

C

Chrome for Android

ブラウザの種類、設定 26

I

Information Service

Mac、インストール 13, 14

iOS

- Information Service、インストール 13, 14
- Web アプリ、シミュレータ 12
- Web アプリ、テスト 11
- アプリ、テストの準備 13
- インストール済みアプリ、テスト 15
- 推奨設定 15
- テスト 8
- デバイス、準備 12
- デバイスの開放 17
- デバイスの開放、記録 17
- デバイスの開放、再生 18
- ネイティブ アプリ、シミュレータ 10
- ネイティブ アプリ、テスト 9
- ハイブリッド アプリケーション 12
- 複数のシミュレータ 15
- 複数のデバイス 15
- モバイル Web アプリケーション、前提条件 8
- モバイル ネイティブ アプリケーション、前提条件 8

M

Mac

Information Service、インストール 13, 14

U

USB ドライバのインストール

Android 6

X

xBrowser

Chrome for Android、設定 26

あ

アプリのアップロード

Mac 4

い

インストール

Information Service、Mac 13

インストール済みアプリ

Android、テスト 15

iOS、テスト 15

インストールする

Information Service、Mac 14

え

エミュレータ

定義、再生 16

テスト 4

き

記録

デバイスの開放 17

モバイル アプリケーション 16

く

クリック

モバイル Web 29

クロス ブラウザ テスト

リモート ロケーション、追加 18

さ

再生

デバイスの開放 18

デバイスの選択 16

し

シミュレータ

定義、再生 16

テスト 9

ネイティブ アプリ、テスト 10

モバイル Web アプリケーション、テスト 12

せ

制限事項

- ネイティブ モバイル アプリケーション 28
- モバイル Web アプリケーション 27

接続文字列

- モバイル デバイス 19

前提条件

- Android、ネイティブ モバイル アプリケーション 4
- Android、モバイル Web アプリケーション 4
- iOS、ネイティブ モバイル アプリケーション 8
- iOS、モバイル Web アプリケーション 8

て

デバイスが接続されていません

- モバイル 22

デバイスの開放

- 記録 17
- 再生 18
- モバイル テスト 17

と

トラブルシューティング

- モバイル 22

ね

ネイティブ モバイル アプリケーション

- Android、前提条件 4
- iOS、前提条件 8
- 制限事項 28

は

ハイブリッド アプリケーション

- Android 5
- iOS 12

ふ

複数のシミュレータ

- iOS 15

複数のデバイス

- iOS 15

ブラウザの種類

- Chrome for Android、設定 26

へ

編集

- リモート ロケーション 18

も

モバイル

- トラブルシューティング 22

モバイル アプリケーション

- 記録 16

- テスト 4

モバイル デバイス

- 操作する 17
- 定義、再生 16
- に対して操作を実行する 17

モバイル ネイティブ アプリケーション

- 制限事項 28

モバイル ブラウザ

- 制限事項 27

モバイル Web

- iOS 11
- 既存のテスト 29
- クリック 29

モバイル Web アプリケーション

- Android、前提条件 4
- iOS、前提条件 8
- 制限事項 27

モバイル テスト

- Android 4
- iOS 8
- Web アプリ、iOS 11
- Web アプリ、iOS シミュレータ 12
- 概要 4
- 接続文字列 19
- デバイスの開放 17
- ネイティブ アプリ、iOS シミュレータ 10
- リモート ロケーション、追加 18

モバイル テスト デバイス

- ネイティブ アプリ、iOS 9

モバイル デバイスの設定

- 再生 16

モバイルの記録

- について 16

り

リモート ロケーション

- 追加 18
- 編集 18