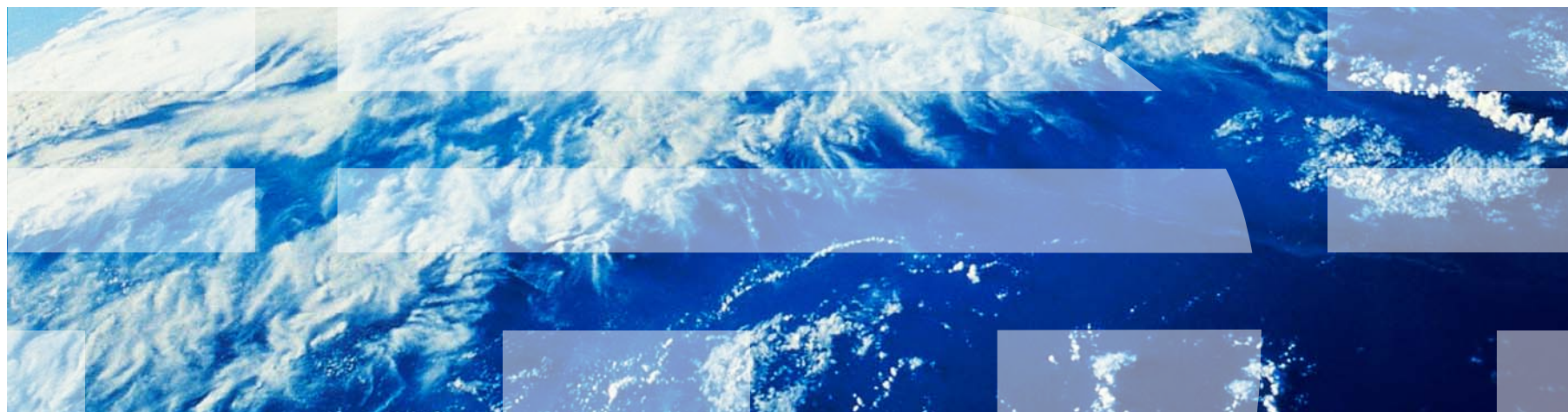# *IBM Worklight V6.1.0*
Getting Started

Integrating with SiteMinder

Trademarks

- IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Worklight is a trademark or registered trademark of Worklight, an IBM Company. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

- Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

- Other company products or service names may be trademarks or service marks of others.

- This document may not be reproduced in whole or in part without the prior written permission of IBM.

About IBM®

See http://www.ibm.com/ibm/us/en/

Agenda

- Background

- Creating the challenge handler

- Creating the native page handler

- Creating the native page handler (Android)

- Creating the native page handler (iOS)

- Examining the result

- Troubleshooting

- Exercise

Background

- As with other form-based authentication methods, SiteMinder requires a custom challenge handler to detect when authentication is required.

- A *native page handler* is required for this module because it displays the SiteMinder authentication form to the user directly.

- This native page handler is required to propagate SiteMinder cookies to your IBM® Worklight® application.

- Though this module targets SiteMinder specifically, the concepts that it describes can apply to other form-based authentication methods.

Agenda

- Background

- Creating the challenge handler

- Creating the native page handler

- Creating the native page handler (Android)

- Creating the native page handler (iOS)

- Examining the result

- Troubleshooting

- Exercise

Creating the challenge handler

- Start the implementation of the challenge handler with the following code, which is used to detect the SiteMinder login page:

```javascript
var LoginFormChallengeHandler = WL.Client.createChallengeHandler("SampleAppRealm");

LoginFormChallengeHandler.isCustomResponse = function(response) {
    if (!response || response.responseText === null) {
        return false;
    }

    if ((response.status == 200) && (response.responseText.indexOf("login.fcc") !== -1)) {
        return true;
    } else {
        return false;
    }
};
```

Creating the challenge handler

- This challenge handler makes some assumptions about your SiteMinder configuration:

  - **PreservePostData** is set to **yes**. This value is the default one.

  - The template file that is being used is named **login.fcc**.

- When SiteMinder is configured with **PreservePostData** set to **yes**:

  - The login page is not displayed immediately.

  - Instead, a temporary page is displayed to collect POST parameters that are then redirected to the login page.

  - The challenge handler then searches for some text in this temporary page.

    - Generally, a string to search for is the name of the FCC template file. In this case: **login.fcc**

Creating the challenge handler

- Continue implementation of the challenge handler with the following code, which is used to start an appropriate native page handler:

```
LoginFormChallengeHandler.handleChallenge = function(response) {
    var nativeClass = "LoginFormNativePage";
    if (WL.Client.getEnvironment() == WL.Environment.ANDROID)
        nativeClass = "com.TestSiteminderApp." + nativeClass;

    if (WL.NativePage){
        WL.NativePage.show(nativeClass, LoginFormChallengeHandler.backFromNative, {});
    } else {
        alert("Functional on iOS/Android devices only.");
    }
};
```

- The native page handler is native code that starts a browser instance that handles the remote authentication form.

- For this module, the class names that are used are **LoginFormNativePage** for iOS and **com.TestSiteMinderApp.LoginFormNativePage** for Android.

  - Your values might differ.

Creating the challenge handler

- Complete the implementation of the challenge handler with the following code, used to detect a successful login from the native page handler:

```
LoginFormChallengeHandler.backFromNative = function(data){
    if (data.status == "success") {
        LoginFormChallengeHandler.submitSuccess();
    } else  {
        LoginFormChallengeHandler.submitFailure();
    }
};
```

Agenda

- Background

- Creating the challenge handler

- Creating the native page handler

- Creating the native page handler (Android)

- Creating the native page handler (iOS)

- Examining the result

- Troubleshooting

- Exercise

Creating the native page handler

- The primary task of the native page handler is to load a SiteMinder remote authentication form.

- You cannot reuse the remote authentication form that caused this native page handler to be created.

- Therefore, an arbitrary URL is loaded that returns a SiteMinder remote authentication form.

- For the purposes of this module, assume that the Worklight Server itself is protected by SiteMinder.

- Therefore, the URL that is accessed is: **/apps/services/reach**

  - This URL exists within the context of the Worklight Server and is always available.

Agenda

- Background

- Creating the challenge handler

- Creating the native page handler

- Creating the native page handler (Android)

- Creating the native page handler (iOS)

- Examining the result

- Troubleshooting

- Exercise

Creating the native page handler (Android, 1 of 4)

- Start the implementation of the native page handler by creating the class and its **onCreate()** method:

```
public class LoginFormNativePage extends Activity {
    private WebView webView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        webView = new WebView(this);
        webView.setWebViewClient(new MyWebViewClient());
        webView.getSettings().setJavaScriptEnabled(true);
        webView.addJavascriptInterface(new HTMLExtractor(), "HTMLExtractor");

        // load ANY url that will trigger login form
        WLConfig cfg = new WLConfig(this);
        webView.loadUrl(cfg.getRootURL() + "/apps/services/reach");
        setContentView(webView);
    }
```

- Complete the implementation of the native page handler with the following code:

```java
private class MyWebViewClient extends WebViewClient{
    @Override
    public void onPageFinished(WebView view, String url) {
        Log.d("","onPageFinished :: " + url);
        view.loadUrl("javascript:window.HTMLExtractor.extract(document.body.innerHTML)");
        super.onPageFinished(view, url);
    }
}

private class HTMLExtractor {
    public void extract(String html){
        // In case response does not contain login form - return to hybrid
        if (!html.contains("USER")){
            Intent i = new Intent();
            i.putExtra("status", "success");
            setResult(RESULT_OK, i);
            finish();
        }
    }
}
}
```

Creating the native page handler (Android, 3 of 4)

- Register the **HTMLExtractor.extract()** method as a JavaScript™ interface that you run after the page for the native page handler is loaded.

- In the **HTMLExtractor.extract()** method, verify that you are dealing with a SiteMinder login form.

  – Here, consider that you are dealing with a SiteMinder login form if the page contains a field that is called **USER**, as most non-customized SiteMinder login forms do.

- Do this check in the **HTMLExtractor.extract()** method to ensure that the native page handler does not run before a SiteMinder login form is presented.

Creating the native page handler (Android, 4 of 4)

- Finally, because the native page handler is an Android activity, add the appropriate definition in your application **AndroidManifest.xml** file:

```xml
<activity android:name=".TestSiteminderApp"
          android:label="@string/app_name"
          android:configChanges="orientation|keyboardHidden"
          android:launchMode="singleTask">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
    <intent-filter>
        <action android:name="com.TestSiteminderApp.TestSiteminderApp.N
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>

<activity android:name=".LoginFormNativePage"></activity>
<!-- Preference Activity -->
```

Agenda

- Background

- Creating the challenge handler

- Creating the native page handler

- Creating the native page handler (Android)

- Creating the native page handler (iOS)

- Examining the result

- Exercise

■ Start the implementation of the native page handler by creating the **LoginFormNativePage.m** file and its **viewDidLoad** method:

```objc
#import "LoginFormNativePage.h"
#import "NativePage.h"

@interface LoginFormNativePage ()

@end

@implementation LoginFormNativePage

- (id)initWithNibName:(NSString *)nibNameOrNil bundle:(NSBundle *)nibBundleOrNil
{
    self = [super initWithNibName:nibNameOrNil bundle:nibBundleOrNil];
    if (self) {
    }
    return self;
}

- (void)viewDidLoad
{
    [super viewDidLoad];

    NSUserDefaults *prefs = [NSUserDefaults standardUserDefaults];

    // hit the reachability url, which will trigger the login form to load
    NSString *serverURL = [NSString stringWithFormat:@"%@/apps/services/reach",[prefs stringForKey:@"WLDefaultServerURL"]];

    // Creating a full screen UIWebView to display login form received from a backend
    UIWebView *webView = [[UIWebView alloc] initWithFrame:CGRectMake(0, 0, 320, 480)];
    [webView setDelegate:self];
    [self.view addSubview:webView];
    [webView loadRequest:[NSURLRequest requestWithURL:[NSURL URLWithString:serverURL]]];
    [webView release];
}
```

Creating the native page handler (iOS, 2 of 3)

- Complete the implementation of the native page handler with the following code:

```objc
- (void)webViewDidFinishLoad:(UIWebView *)webView{
    NSLog(@"webViewDidFinishLoad");

    // Get the received web content
    NSString *webViewContent = [webView stringByEvaluatingJavaScriptFromString:@"document.body.innerHTML"];

    // Check if it contains something that will tell us that this is a login form
    if ([webViewContent rangeOfString:@"USER" options:NSCaseInsensitiveSearch].location != NSNotFound){
        // In case it's a login form – do nothing.
        NSLog(@"Looks like we're still in a login form");
    } else {
        // In case it is not a login form looks like we've passed authentication. Need to get back to web part.
        NSLog(@"This is not a login form. Looks like auth is done");
        NSDictionary *response = [NSDictionary dictionaryWithObject:@"success" forKey:@"status"];
        [NativePage showWebView:response];
    }
}
```
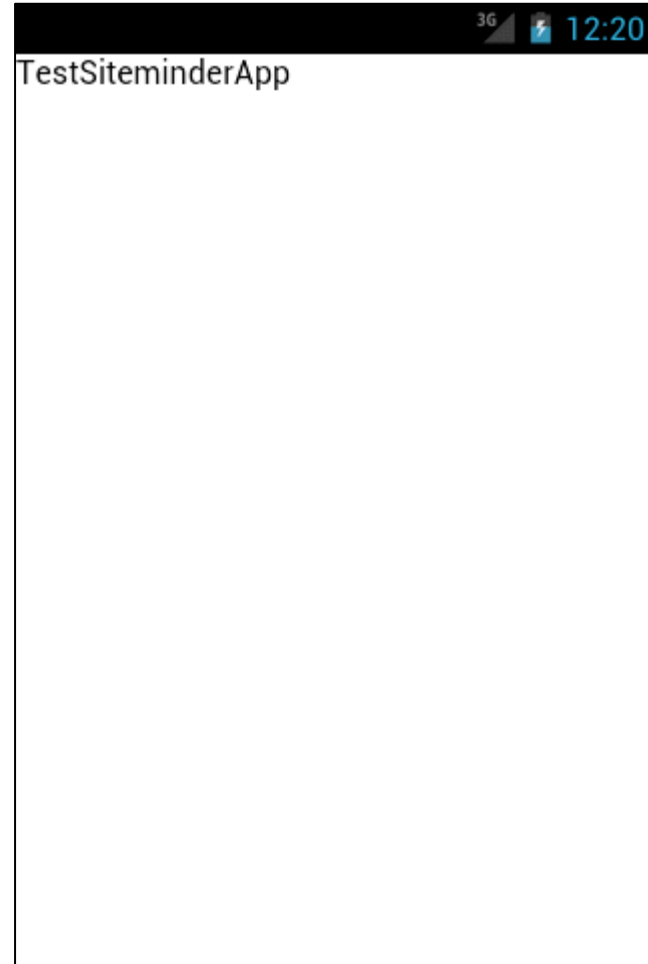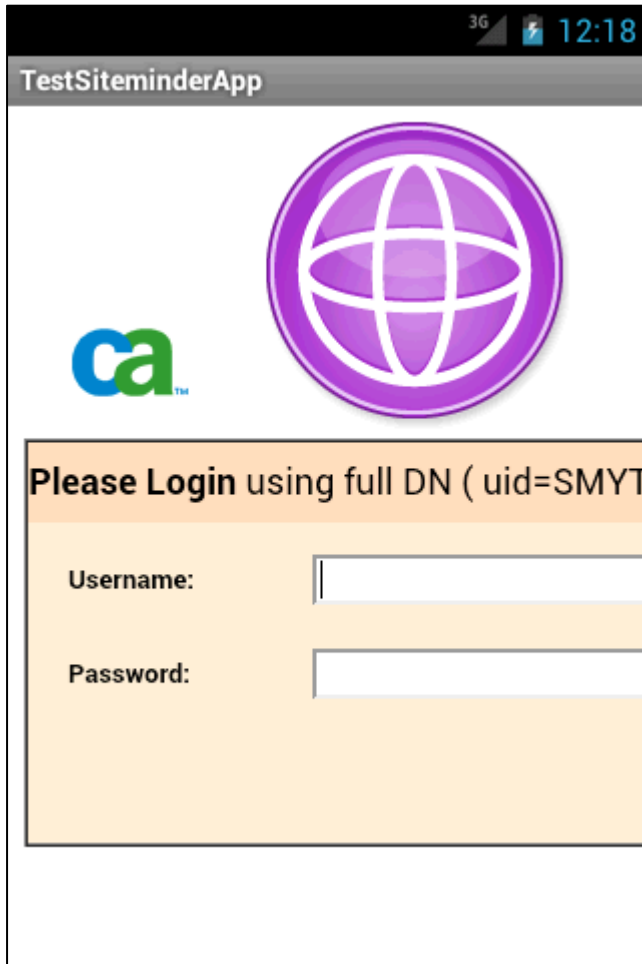
Creating the native page handler (iOS, 3 of 3)

- In the **webViewDidFinishLoad** method, verify that you are dealing with a SiteMinder login form.

  - Here, consider that you are dealing with a SiteMinder login form if the page contains a field that is called **USER**, as most non-customized SiteMinder login forms do.

- Do this check in the **webViewDidFinishLoad** method to ensure that the native page handler does not run before a SiteMinder login form is presented.

Agenda

- Background

- Creating the challenge handler

- Creating the native page handler (Android)

- Creating the native page handler (iOS)

- Examining the result

- Troubleshooting

- Exercise

Examining the result

Agenda

- Background

- Creating the challenge handler

- Creating the native page handler (Android)

- Creating the native page handler (iOS)

- Examining the result

- Troubleshooting

- Exercise

Troubleshooting

- Because SiteMinder can be customized, you might also customize this scenario.

- Focus on the following two areas when troubleshooting:

  - Was the login form properly detected?

    - Check the content of **/apps/services/reach** to make sure that you are receiving the login form as expected.
    - Ensure that the returned content contains the string that the challenge handler and the native page handler are looking for. You might have to look at the network response directly because, from the browser, you might not be able to identify a redirect for post data.

  - Did authentication complete successfully?

    - Check the network traffic to see whether cookies were set on the response.
    - Add logging to make sure that **submitSuccess** is being called locally.

Agenda

- Background

- Creating the challenge handler

- Creating the native page handler (Android)

- Creating the native page handler (iOS)

- Examining the result

- Troubleshooting

- Exercise

Exercise

- Implement the remote form-based authentication that is described in this module.

- The sample for this training module can be found in the Getting Started page of the IBM Worklight documentation website at http://www.ibm.com/mobile-docs

Notices

- Permission for the use of these publications is granted subject to these terms and conditions.

- This information was developed for products and services offered in the U.S.A.

- IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

- IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

  – IBM Director of Licensing
    IBM Corporation
    North Castle Drive
    Armonk, NY 10504-1785
    U.S.A.

- For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

  – Intellectual Property Licensing
    Legal and Intellectual Property Law
    IBM Japan Ltd.
    1623-14, Shimotsuruma, Yamato-shi
    Kanagawa 242-8502 Japan

- **The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

- This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

- Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

- IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

- Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

  – IBM Corporation
    Dept F6, Bldg 1
    294 Route 100
    Somers NY 10589-3216
    USA

- Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

- The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

- Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

COPYRIGHT LICENSE:

- This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

- Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

  – © (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

Privacy Policy Considerations

- IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

- Depending upon the configurations deployed, this Software Offering may use session cookies that collect session information (generated by the application server). These cookies contain no personally identifiable information and are required for session management. Additionally, persistent cookies may be randomly generated to recognize and manage anonymous users. These cookies also contain no personally identifiable information and are required.

- If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent. For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at http://www.ibm.com/privacy and IBM's Online Privacy Statement at http://www.ibm.com/privacy/details the sections entitled "Cookies, Web Beacons and Other Technologies" and the "IBM Software Products and Software-as-a-Service Privacy Statement" at http://www.ibm.com/software/info/product-privacy.

Support and comments

- For the entire IBM Worklight documentation set, training material and online forums where you can post questions, see the IBM website at:
  - http://www.ibm.com/mobile-docs
- **Support**
  - Software Subscription and Support (also referred to as Software Maintenance) is included with licenses purchased through Passport Advantage and Passport Advantage Express. For additional information about the International Passport Advantage Agreement and the IBM International Passport Advantage Express Agreement, visit the Passport Advantage website at:
    - http://www.ibm.com/software/passportadvantage
  - If you have a Software Subscription and Support in effect, IBM provides you assistance for your routine, short duration installation and usage (how-to) questions, and code-related questions. For additional details, consult your IBM Software Support Handbook at:
    - http://www.ibm.com/support/handbook
- **Comments**
  - We appreciate your comments about this publication. Please comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this document. The comments you send should pertain to only the information in this manual or product and the way in which the information is presented.
  - For technical questions and information about products and prices, please contact your IBM branch office, your IBM business partner, or your authorized remarketer.
  - When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you. IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you state.
  - Thank you for your support.
  - Submit your comments in the IBM Worklight Developer Edition support community at:
    - https://www.ibm.com/developerworks/mobile/worklight/connect.html
  - If you would like a response from IBM, please provide the following information:
    - Name
    - Address
    - Company or Organization
    - Phone No.
    - Email address

Thank You