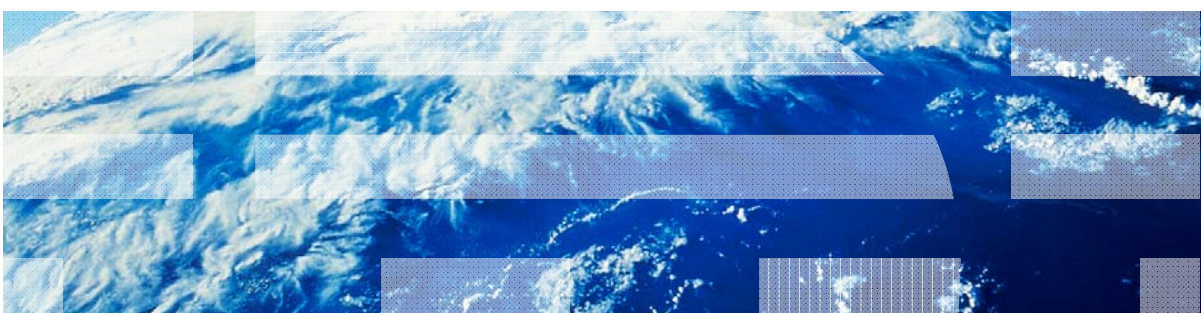
# *IBM Worklight V6.0.0*
# *Getting Started*

## iOS – Adding native functionality to hybrid application with Apache Cordova plug-in



27 August 2013

# 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*

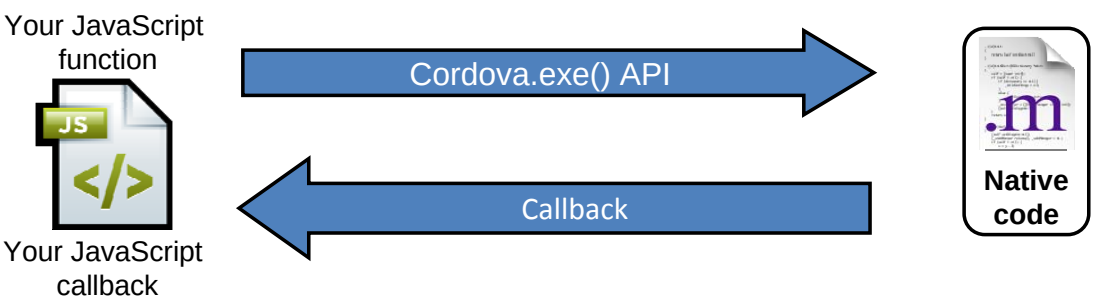
- Apache Cordova plug-in overview
- Plug-in declaration
- Implementing cordova.exec() in JavaScript
- Implementing an Objective-C code plug-in

# *Apache Cordova plug-in overview (1 of 2)*

- In some cases, developers of an IBM Worklight® application might have to use a specific third-party native library or a device function that is not yet available in Apache Cordova.
- With Apache Cordova, developers can *creating an Apache Cordova plug-in*, that is, create custom native code blocks and have them called in their applications by using JavaScript™.
- This module demonstrates how to create a simple Apache Cordova plug-in and how to use it in your code.
- More samples can be found in the Apache Cordova documentation at https://github.com/phonegap/phonegap-plugins.

# *Apache Cordova plug-in overview (2 of 2)*

- To creating and use an iOS Apache Cordova plug-in:
  - Declare the plug-in in the **config.xml** file.
  - Use **cordova.exec()** API in the JavaScript code.
  - Create the plug-in class that will be run natively in iOS.
- The plug-in performs the required action and calls a JavaScript callback method that is specified during **cordova.exec()** invocation.



# *Agenda*

- Apache Cordova plug-in overview
- Plug-in declaration
- Implementing cordova.exec() in JavaScript
- Implementing an Objective-C code of a plug-in

# Plug-in Declaration

- For Cordova to know about the plug-in that will be created in the next slides, you must also declare that plug-in in the project.
  - Add your plug-in reference to the **config.xml** file, which is in the **`native`** folder of the iOS environment.
  - Make sure to place the plug-in reference below the **<!—User—>** section.

```
  <plugin name="SignificantLocationChangeWatchPlugin" value="Significant
  <plugin name="AnalyticsConfigurator" value="AnalyticsConfigurator"/>
  <!--User-->
  <plugin name="HelloWorldPlugin" value="HelloWorldPlugin"/>
</plugins>
```

**The name that the JavaScript code uses to reference the plug-in**

**The name of the implementation class**

# *Agenda*

- Apache Cordova plug-in overview
- Plug-in declaration
- Implementing cordova.exec() in JavaScript
- Implementing an Objective-C code of a plug-in

# *Implementing cordova.exec() in JavaScript (1 of 3)*

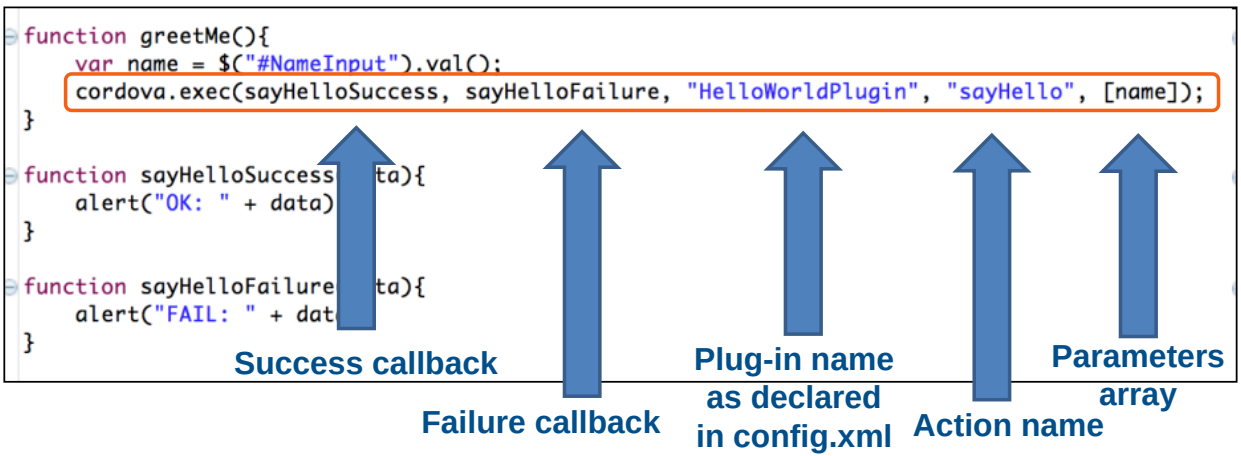
- From the JavaScript code of the application, use **cordova.exec()** to call the Cordova plug-in.

```
function greetMe(){
    var name = $("#NameInput").val();
    cordova.exec(sayHelloSuccess, sayHelloFailure, "HelloWorldPlugin", "sayHello", [name]);
}

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

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

Use **cordova.exec()** to call the plug-in.

# *Implementing cordova.exec() in JavaScript (2 of 3)*

- From the JavaScript code of the application, use **cordova.exec()** to call the Cordova plug-in.



# *Implementing cordova.exec() in JavaScript (3 of 3)*

- From the JavaScript code of the application, use **cordova.exec()** to call the Cordova plug-in.

```
function greetMe(){
    var name = $("#NameInput").val();
    cordova.exec(sayHelloSuccess, sayHelloFailure, "HelloWorldPlugin", "sayHello", [name]);
}

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

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

Success and failure callbacks will be called by the plug-in.

# *Agenda*

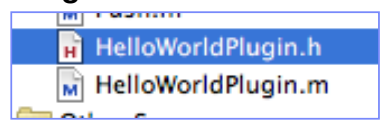
- Apache Cordova plug-in overview
- Plug-in declaration
- Implementing cordova.exec() in JavaScript
- Implementing an Objective-C code of a plug-in

# *Implementing an Objective-C Code plug-in (1 of 6)*

- Now that the plug-in is declared, and the JavaScript implementation is ready, you can also implement the Cordova plug-in.
- For this purpose, ensure that the project is built in Eclipse and opened in the Xcode IDE.

# *Implementing an Objective-C Code plug-in (2 of 6)*

- In the Xcode IDE, open the generated Xcode project that you previously built in Eclipse.
- Add an Objective-C class. Call it **HelloWorldPlugin**.
- Import the **Cordova/CDV.h** and inherit the **CDVPlugin** class.
- Declare the method signature.




```
#import <Foundation/Foundation.h>
#import <Cordova/CDV.h>

@interface HelloWorldPlugin : CDVPlugin

- (void)sayHello:(CDVInvokedUrlCommand*)command;

@end
```

# *Implementing an Objective-C Code plug-in (3 of 6)*

- Implement the method:

```
#import "HelloWorldPlugin.h"

@implementation HelloWorldPlugin

- (void)sayHello:(CDVInvokedUrlCommand*)command{

    NSString *responseString =
        [NSString stringWithFormat:@"Hello World, %@", [command.arguments objectAtIndex:0]];

    CDVPluginResult *pluginResult
        [CDVPluginResult resultWith tus:CDVCommandStatus_OK messageAsString:responseString];

    [self.commandDelegate sendPluginR t:pluginResult callbackId:command.callbackId];
}

@end
```

The **command** argument contains references to the parameters that are sent from JavaScript and callbacks.

# *Implementing an Objective-C Code plug-in (4 of 6)*

- Implement the method:

```
#import "HelloWorldPlugin.h"

@implementation HelloWorldPlugin

- (void)sayHello:(CDVInvokedUrlCommand*)command{

    NSString *responseString =
        [NSString stringWithFormat:@"Hello World, %@", [command.arguments objectAtIndex:0]];

    CDVPluginResult *pluginResult =
        [CDVPluginResult resultWithStatus:CDVCommandStatus_OK messageAsString:responseString];

    [self.commandDelegate sendPluginResult:pluginResult callbackId:command.callbackId];
}

@end
```

Use it to retrieve the parameters that are sent from JavaScript.

# *Implementing an Objective-C Code plug-in (5 of 6)*

- Implement the method:

```
#import "HelloWorldPlugin.h"

@implementation HelloWorldPlugin

- (void)sayHello:(CDVInvokedUrlCommand*)command{

    NSString *responseString =
        [NSString stringWithFormat:@"Hello World, %@", [command.arguments objectAtIndex:0]];

    CDVPluginResult *pluginResult =
        [CDVPluginResult resultWithStatus:CDVCommandStatus_OK messageAsString:responseString];

    [self.commandDele sendPluginResult:pluginResult callbackId:command.callbackId];
}

@end
```

The **pluginResult** object is created with data retrieved from JavaScript. The **CDVCommandStatus** parameter defines whether the plug-in call was successful or not.

# *Implementing an Objective-C Code plug-in (6 of 6)*

- Implement the method:

The **sendPluginResult** method is used to return a response back to JavaScript (invoke callback).

```
#import "HelloWorldPlugin.h"

@implementation HelloWorldPlugin

- (void)sayHello:(CDVInvokedUrlCommand*)comma

    NSString *responseString =
        [NSString stringWithFormat:@"Hello, %@", [command.arguments objectAtIndex:0]];

    CDVPluginResult *pluginResult =
        [CDVPluginResult resultWithStatus:CDVCommandStatus_OK messageAsString:responseString];

    [self.commandDelegate sendPluginResult:pluginResult callbackId:command.callbackId];
}

@end
```

# End Result

- 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

## *Important*

- If you add existing Cordova plug-ins, instead of creating your own plug-ins, make sure to place their “`.m`” and “`.h`” files in the `Classes` folder in the Xcode project.
- Placing these “`.m`” and “`.h`” files in the `iphone\native\classes` folder in Eclipse is not enough to have them referenced in the Xcode project.

# Check yourself questions

- For a plug-in to be recognized in a JavaScript application, you should add it to:
  - config.xml file
  - Worklight.plist file
  - Plugins.plist file
  - Plug-in will be automatically recognized by JavaScript without adding it to any of previous files.
- When should a Cordova plug-in be used?
  - When you want to implement an application in the native code because you are not familiar with JavaScript.
  - When you want your application to look more like a native application.
  - When you want to gain access to OS APIs that you cannot access within the web container.
  - When you need to retrieve data from a remote server.
- What are the mandatory components of a Cordova plug-in?
  - A native class that implements the required functionality. After it is declared in **config.xml** file, it can be called from the application JavaScript code.
  - A native class that implements the required functionality and a JavaScript wrapper for it. The wrapper functions can be called from JavaScript.
  - A native class that implements the required functionality, a JavaScript wrapper for it, and a declaration in the application-descriptor.xml file.
  - A JavaScript wrapper only. Native classes are already provided by IBM Worklight.

# Check yourself questions

- For a plug-in to be recognized in a JavaScript application, you should add it to:
  - config.xml file
  - Worklight.plist file
  - Plugins.plist file
  - Plug-in will be automatically recognized by JavaScript without adding it to any of above files.
- When should a Cordova plug-in be used?
  - When you want to implement an application in the native code because you are not familiar with JavaScript.
  - When you want your application to look more like a native application.
  - When you want to gain access to OS APIs that you cannot access within the web container.
  - When you need to retrieve data from a remote server.
- What are the mandatory components of a Cordova plug-in?
  - A native class that implements the required functionality. After it is declared in **config.xml** file, it can be called from the application JavaScript code.
  - A native class that implements the required functionality and a JavaScript wrapper for it. The wrapper functions can be called from JavaScript.
  - A native class that implements the required functionality, a JavaScript wrapper for it, and a declaration in the application-descriptor.xml file.
  - A JavaScript wrapper only. Native classes are already provided by IBM Worklight.

# 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*