

Worklight® 4.2

# Release Notes

November 2011



© 2006-2011 Proprietary and Confidential of Worklight Inc.

The information contained herein is the propriety and confidential information of Worklight Inc. and is not to be disclosed or used in any way except as expressly authorized by written contract with Worklight.

# Table of Contents

<b>New Features</b> .....	<b>3</b>
Application Development.....	3
Authentication and Integration.....	5
Runtime, Security, and Management.....	6
More Environments.....	7
<b>Changed Behavior</b> .....	<b>8</b>
Installation and Configuration.....	8
Building Mobile Environments.....	8
Publishing to Web Environments.....	8
Building Desktop Environments.....	8
Database Structure.....	8
<b>Upgrading from Previous Versions</b> .....	<b>9</b>
Upgrading the Development Environment.....	9
Upgrading the Production Environment.....	9
<b>Deprecated API and Unsupported Features</b> .....	<b>10</b>
Installation and Configuration.....	10
Mobile Application Development.....	10
Integration.....	10
Web Environments.....	10
Desktop Environments.....	11
<b>Bug Fixes and Closed Issues</b> .....	<b>12</b>
<b>Known Issues</b> .....	<b>18</b>

# New Features

## Application Development

### Support for PhoneGap 1.0

Release 4.2 includes PhoneGap 1.0 as part of its mobile client run time.

### Support for OS X 10.7 Lion and Xcode 4.2

Developers of iOS apps can now run the Worklight 4.2 Studio on Mac OS X 10.7 Lion as well as use Xcode 4.2 to add native Objective-C code to their native and hybrid apps, build their applications, run them in the iOS emulator, and package them for the Apple app store.

### Use of Prototype 1.7

The Worklight client run-time framework has been upgraded and can now use the Prototype 1.7 library.

### Granular Build

Release 4.2 enhances the builder functionality for hybrid mobile apps. The new builder is seamlessly integrated into the Worklight Studio and also allows developers to build only specific environments at a time, instead of all supported environments on each Build action.

### Centralized Build

The Worklight Builder is now available as a standalone application that can be integrated with a central build service such as Luntbuild, Hudson, and IBM Rational Jazz, effectively enhancing the collaboration and automation of the development process. See the *Worklight 4.2 Developer's Reference Guide* for details.

### A New Push Notifications Model

Release 4.2 includes:

- A new server-side API for pushing notifications to user devices through the Apple and Google notification services.
- A matching JavaScript and Objective-C APIs for subscribing to notifications and handling them on the client side.

The new API methods provide greater flexibility in pushing notifications to specific devices of a given user as well as allow applications to determine whether different users of the same device can subscribe to the same notifications.

For performance reasons, custom notification states are not automatically saved by the Worklight Server. Release 4.2 introduces a new API that saves these notification states.

See the *Worklight 4.2 Developer's Reference Guide* for more details.

## Simplified Templates for New App Environments

In release 4.2, the HTML, CSS and JavaScript templates that are automatically generated upon creation of a new environment have been simplified.

- The new app HTML template no longer contains the "content" and "auth" div elements. This makes it immediately compatible with third-party JavaScript toolkits such as Sencha Touch and jQuery Mobile.
- The JavaScript templates no longer require knowledge of object-oriented JavaScript development.

## Remote On-device Debugging

With Worklight 4.2, developers can use tools such as weinre to inspect their app remotely, running on an emulator or a physical mobile device. See the Worklight 4.2 training modules for additional details.

## Studio Preview Function

The Worklight 4.2 Studio now allows developers to preview an app in a specific environment directly in the Eclipse project browser by invoking a context-menu option. The environment name is also visible as the title of the page in the browser.

## Studio "Run Configuration" Screens Now Hidden

The Worklight 4.2 Studio no longer displays the Eclipse Run Configuration windows when previewing or first building an application, thereby streamlining the process of running these commands. The Run Configuration windows can still be accessed via Eclipse's Run > Run Configurations menu, if necessary.

## Connectivity Settings Screen

Release 4.2 introduces a Settings screen that is automatically added to iOS and Android apps. The Settings screen allows developers to view and change the address of the Worklight Server with which the app communicates. This troubleshooting feature is valuable for internal testing, beta testing, and demo phases, where multiple versions of the app are circulated among different teams and multiple server environments are available.

This feature is also available for native iOS and Android apps, using specific API and instructions provided by Worklight.

## Disconnect from the Worklight Server

In release 4.2, the `WL.Client.init()` function now allows developers to initialize the application without attempting to connect to the Worklight Server (instead of trying to connect and handling a connectivity failure). This improves the application's startup performance in offline mode and simplifies the application code.

## Ability to Regress Multiple Pages at Once in the App's Page Stack

The `WL.Page.back()` method now allows specifying the number of pages to go back in the application's page stack. Refer to the *Worklight 4.2 Developer's Reference Guide* for details.

## Support for Android Widgets

In release 4.2, developers can use Worklight's Java API for Android to write widgets that communicate with the Worklight Server and retrieve backend corporate data. Refer to the *Worklight 4.2 Java API for Android* guide for details.

## JavaScript Access to Native BlackBerry Dialog Boxes

In release 4.2, the `WL.SimpleDialog` client-side API now renders native dialog boxes on BlackBerry devices.

## Windows Phone 7 Fragments

Fragments in Windows Phone 7 are now supported by the Worklight Studio. Developers can now split WP7 apps into multiple HTML files and use the fragments and page mechanism to load them in run-time, thereby minimizing application load time, facilitating simultaneous development of the application, and improving code maintainability.

## Control Over Windows Phone Application Bar Opacity Upon Initialization

Developers can now set the opacity of the Windows Phone application bar upon initialization, to better control its appearance.

# Authentication and Integration

## Enhanced JavaScript Framework for Flexible Implementation of Complex Login Scenarios

Release 4.2 introduces new server-side components that allow developers to implement complex, multi-step login scenarios in server-side JavaScript adapters.

In addition, Release 4.2 contains a simplified template for development of in-app login screens and authentication sequences. Refer to the *Worklight 4.2 Developer's Reference Guide* and training modules for details.

## Out-of-the-box support for Kerberos, NTLM, Basic, and Digest authentication for HTTP integration

Release 4.2 greatly simplifies integration with HTTP-based services that require authentication. Integration with Kerberos, NTLM, Basic, and Digest authentication can be easily achieved by simple configuration of the HTTP adapter, without having to write server-side code.

## Simplified Proxy Configuration for HTTP Integration

Release 4.2 introduces easy and flexible configuration of HTTP-based services that must be accessed via proxy. Proxy settings can be defined separately for each adapter, including support for authenticating proxies and HTTPS proxies.

## **SOAP Envelopes Signing with X509 Certificates**

Release 4.2 includes a new server-side API that can be used within the HTTP adapter to sign SOAP envelopes with X509 certificates, enabling secure exchange of messages between the Worklight Server and back-end applications that support X509 certificates.

## **Support for Radius Authentication**

Release 4.2 includes a new login module for integrating with Radius servers for user authentication.

## **Support for Device SSO**

Release 4.2 supports device SSO for mobile applications. When an application that is configured to use device SSO connects to the Worklight Server, the Server checks whether there is an existing authenticated session from that device. If there is one, the Worklight Server copies the authentication details from the existing session to the new one, making the new session automatically authenticated. Device SSO is particularly useful in employee-facing apps, where users do not have to repeatedly re-enter credentials each time launching a corporate app.

## **Server-side JavaScript Debugging**

The Worklight 4.2 Studio enables developers to fully debug their server-side JavaScript code using Eclipse Helios's JavaScript debugger. See the Worklight 4.2 training modules for further details.

# Runtime, Security, and Management

## **Support for WebSphere Application Server 7 and Higher**

With Release 4.2, administrators can package the Worklight Server, together with its customer-specific properties and code, to create a deployable on WebSphere Application Server versions 7 and higher.

## **Remote Disable: Ability to direct User to Application Stores**

When disabling an app remotely via the Worklight Console, administrators can now specify a link to the app store or marketplace, where a new version of the app is available. When such a link is specified, Worklight automatically displays a button that opens the applicable store at the specified location, allowing the user to immediately download the new app version.

## **Remote Disable: Multi-line Message**

When disabling an app remotely via the Worklight Console, the message displayed to the user can now contain newline characters.

## Determine App behavior before entering background mode in iOS

iOS takes a screenshot of an application just before it enters the background mode. While the app is retrieved from the background, the image is presented to the user to enhance his experience. In 4.2, developers can control how the application behaves upon entering background mode, selecting from a set of predefined options, or writing JavaScript code to implement custom behavior. Refer to the *Worklight 4.2 Developer's Reference Guide* for details.

## Serving Gzipped Resources to Web Environments

The Worklight Server automatically gzips application resources (HTML, CSS, and JavaScript files) when serving them to web environments, i.e. mobile web apps, embedded web apps, iGoogle, and Facebook.

## More Environments

The following new mobile operating systems are supported by Worklight 4.2:

- iOS 5
- Android 4.0 Ice Cream Sandwich
- Windows Phone Mango

The native projects generated by the Worklight Studio are compatible with these operating systems. Developers of mixed hybrid apps can write native code that utilizes new APIs available with these new mobile OS versions. Developers can use the applicable native SDKs to compile their apps, run them in emulators, and package their apps for distribution in the different application stores .

# Changed Behavior

## Installation and Configuration

- Installation location in Mac OS X has changed to the current user's Applications Folder rather than to the Root's Applications Folder.
- Proxy server configuration is no longer done in the `worklight.properties` file, but in the XML configuration file of each adapter that needs proxy access.

## Building Mobile Environments

- To comply with Xcode 4.2, and as instructed by Apple, the Xcode project generated by the Worklight Builder for iOS environments now uses the "LLVM GCC 4.2" compiler, instead of the former "GCC 4.2" compiler. This means that Xcode projects generated with Worklight 4.2 are no longer compatible with Xcode versions 3.x.

## Publishing to Web Environments

- Publishing to iGoogle: To publish a widget to iGoogle, administrators should now download the widget's descriptor via the Worklight Console and place it in a public place in their web site. See the 4.2 Administration Guide for more information.

## Building Desktop Environments

- Signing AIR applications: The certificate must now be installed within the application resources, and not in the configuration folder of the Worklight Server. See the 4.2 Worklight Developer's Reference Guide for details.
- Signing Windows 7 and Vista gadgets is longer done while building the applications, but as a manual process. The Worklight 4.2.1 Administration Guide will contain instructions for signing such gadgets for production.

## Database Structure

- The table `GADGET_ACTIVITY_REPORT` was renamed `APP_ACTIVITY_REPORT`.

# Upgrading from Previous Versions

## Upgrading the Development Environment

Worklight apps that were developed on previous releases must be upgraded before they can be built with version 4.2. To upgrade an app, you should simply install the new Worklight 4.2 Server and Worklight Studio 4.2 Eclipse Plug-in, and then build your application.

- If your apps are compatible with version 4.1.1, the upgrade process is completely automatic
- If your apps are compatible with version 4.0 or 4.1, you must perform some manual steps to upgrade your application. The Worklight Studio automatically detects the version of your app and instructs you as needed.

## Upgrading the Production Environment

Upgrading a production environment will be part of version 4.2.1. The Worklight 4.2.1 Administration Guide will contain instructions for upgrading production environments.

# Deprecated API and Unsupported Features

## Installation and Configuration

The `devmode` property in the `worklight.properties` file is no longer needed.

## Mobile Application Development

The `WL.Client.isConnected` method is no longer supported. Use `WL.Device.getNetworkInfo` instead.

Property `WL.AppProperty.WLCLIENT_TIMEOUT_IN_MILLIS` is no longer supported. App developers can set the timeout for communication with the Worklight Server when initializing the connectivity to the Server using `WL.Client.init ({timeout: x});`.

In the Objective-C API for iOS, `wlInitWithDelegate` is deprecated. Use `wlConnectWithDelegate` instead.

In the Java API for Android, `WLClient.init` is deprecated. Use `WLClient.connect` instead.

The `wlclientTimeout` property that governed the client behavior has been removed from the `worklight.properties` file. Timeout can be specified by application developer when initializing the application or when issuing specific requests.

The `<iphone type>` attribute, denoting whether an app is native or hybrid, is no longer required.

## Integration

Sub-element `<displayName>` of `<adapter>` in the adapter configuration file is no longer supported. This element will be removed from the adapter validating schema in the upcoming release.

As part of fixing bug 2254, authentication requests to the Worklight Server are now made using HTTP POST. The Worklight Server still supports receiving such requests issued with HTTP GET, but this will be discontinued in the next major release.

The `WL.Server.submitNotification` method is no longer supported and has been replaced by `WL.Server.notifyDeviceSubscription`.

## Web Environments

Worklight no longer generates different instance numbers for web widgets. This affects the following features:

- Authenticated provisioning (preventing viral distribution) is no longer supported. As such, the `userWidgetGallery` and `userWidgetGalleryCaption` properties, necessary for non-viral distribution, are no longer supported.
- When distributing a widget virally, it is no longer possible for one user to distribute his preferences to the destination users
- User preferences now depend on the user's identity only. It is not possible to place multiple copies of an iGoogle widget within the same iGoogle page, each with different preferences.
- The public resource server is no longer supported. Public resources of web widgets can be placed on the public part of the customer web site, and made available to the widget container as necessary.

The `WL.Client.makeRequest` method, allowing developers of web apps to use the Worklight Server as a proxy, is no longer supported. You can use the adapter framework instead.

Full-screen authentication is no longer supported. Login forms can be invoked in a separate browser window or be shown within the widget.

The widget's Welcome page is no longer generated by the Worklight Server. A template Welcome page is available as a developer resource on the [worklight.com](http://worklight.com) web site.

## Desktop Environments

Building Mac Dashboard widgets is temporarily not supported. Support for these widgets will be restored in version 4.2.1.

## Bug Fixes and Closed Issues

ID	Applicable To	Description
24	3.3	In Remember Me scenarios, username is not saved on Android devices
43	4.1	Different adapters with the same Display Name can be deployed to the Worklight Server.  In 4.1.3, the Display Name property has been deprecated, leaving only the Name property, which must be unique.
58	3.2	Reloading an AIR app sometimes creates an error and the app gets stuck
297	3.3	Android app may crash after immediately logging in after logging out.
344	4.1	Adapter XML files created by the Worklight Studio did not contain a reference to their validating schema.
492	3.3	When logging out from an app after the session had already expired, the app required the user to login prior to successfully logging out.
550	3.3	Errors displayed after stopping the Worklight Server using wl-stop.sh
664	4.0	When creating an adapter or app using the Worklight Studio and providing a name with unsupported characters, the error message is not informative enough.
665	3.3	The JavaScript error message output upon missing client-side onBeforeLogin() is unclear.
1041	4.1	Upon creating a new adapter in the Worklight Studio, the auto-complete for the Server-side JavaScript API duplicates the WL.Adapter entry.
1147	3.3	Connectivity error dialog is not displayed in BlackBerry apps.
1175	4.0	When installing the Worklight Studio Eclipse plug-in, the Group check box must be cleared before continuing the installation.
1345	3.1	The Remote Disable dialog box cannot be closed when an iPhone is connected to a Mac running Xcode.
1346	4.1	The Catalog allows saving and deleting Remote Disable rules after the session with the Server expires.
1415	4.0	When using the native Objective-C API for iOS, some app activities are not recorded in the reporting data.

ID	Applicable To	Description
1501	4.1	An exception was issued when deploying an app containing a .svn folder. .wlapp Application bundles can no longer be created manually, so this bug is no longer relevant.
1531	3.1	When the app code selects a non-existent tab in a tab bar, no error message is returned to the developer.
1572	3.1	In Android, the tab bar was automatically set visible after init(), instead of only after an explicit call to setVisible().
1575	3.1	After re-initializing the Android tab bar, the tabs remain instead of being cleared.
1615	4.0	When passing "null" from native code to JavaScript, it is passed as a JSON object instead of as "null".
1637	4.1	Web resources that were deleted from the source code still appear in the auto-generated native project
1709	4.1	The Worklight Studio allows developers to create long application and adapter names.  Application and adapter names are now limited to 255 characters.
1785	4.0	Calls to procedures time out after a very long time if the adapter is configured to work with a proxy while the Worklight Server is not.
2032	4.0	Date arguments sent from client apps are not received by the Worklight Server as date objects.
2176	3.1	Login activity is not reported in the raw reporting table
2265	4.0.1	localStorage.clear() deletes WorkLight cookies.
2275	4.0.2	Error message is unclear when the push notification adapter code does not specify badge when using APNS.
2283	4.0.2	On Android 2.1, the example skinLoad.js code issues a warning message to the Android log cat.
2424	4.0.2	Sample authentication code does not work for mobile devices.
2430	4.1	WL.OptionsMenu does not appear in the auto-complete of Worklight Studio.
2455	4.1	WL.SimpleDialog does not display well on unsupported browsers, such as IE6. This issue is closed without a fix.
2473	4.0.1	The code handling the mobile device's Back button does not catch exceptions.
2482	4.1	Apps with specific optimizations for iPad could not receive push notifications on the iPad.

ID	Applicable To	Description
2601	4.0.1	The session against the Worklight Server is not always shared between the web and native code of an app.
2640	4.1	The default public resource server may cause long delays while deploying apps.  By default, the Worklight Server no longer accesses the default public resource server (or any public resource server).
2769	4.0	Session between a BlackBerry app and the Worklight Server does not close when the app quits, if the BlackBerry device has no connection to the Worklight Server.
2802	4.1	Deploying an application, whose resources are being served during Direct Update, fails.
2849	4.1.1	WL.App.close() does not appear in the auto-complete of Worklight Studio.
2852	4.1	If connectivity to the Worklight Server is not available while connecting to it, a white screen is displayed for a long time.
2862	4.1	When the application updates its web resources after coming back to the foreground, resources used by native components provided by Worklight (such as the tab bar or options menu) are not automatically updated
2867	4.1	When deploying an application to the Worklight Server and the folder used by the Server is used by another process, the error message shown in the Worklight Studio is unclear.
2869	4.1.1	Automatic upgrade is performed to an application depends on the settings of the project it belongs to, rather than on the application itself.
2870	4.1	When launching an iOS app for the second time from Xcode on a real iOS device, direct update occurs without prior notification
2904	4.0.1	Application crashes if exiting it while trying to connect to the Worklight Server.
2919	4.1	Fragments do not work on Windows Phone 7.
2933	4.1	The training app on Fragments does not work in Windows Phone 7.
2957	4.1	The default conversion from XML to JSON in the HTTP adapter returns an escaped string.
2982	4.1.1	Android apps do not receive notifications while in the background

ID	Applicable To	Description
2984	4.1	Warning message for using local.bindAddress=all in the worklight.properties file is not clear enough.
3020	4.1.2	When auto-upgrading an iOS app from 4.1.x, newly added files are copied without checking if they already exist
3036	4.1.2	Android developer does not get an indication in case the 'ACCESS_WIFI_STATE' privilege is missing and a network problem was detected.
3039	4.1.2	WL.Device.getNetworkInfo should return NULL for unsupported properties, instead of returning the string "Not available".
3054	4.1.2	When an iOS app, the error message is not informative enough if the upgrade fails on a corrupted pbxproj file.
3056	4.1.2	When the upgrader fails to upgrade a specific app environment, there may be problems while restoring the application backup.  In 4.1.3, the upgrade is an atomic action: failing to upgrade one environment causes the entire upgrade process to fail and all environments to be restored.
3058	4.1.2	When the upgrader fails to create a backup for an app, the error message is not informative enough.
3061	4.1.2	If the application folder contains environment folders that are not mentioned in the application descriptor, the upgrade fails.  The upgrade process now requires the application descriptor and the application folder to match, before it starts upgrading the app
3065	4.1.2	Sometimes when clicking the Back button in an Android app, a Cookie Double Submit error occurs
3073	4.1.2	Remote Notification is displayed also when the app returns to the foreground, not only when it starts up.
3077	4.1.2	The set-customization.bat command does not work when the Worklight Server is installed within a path with spaces
3092	4.1.3	Pages and fragments sometimes do not work in Adobe AIR.
3093	4.1.1	Running an XSL transformation during procedure invocation from the Worklight Studio fails.
3099	4.1.2	In the Worklight Studio, the "Run as > Xcode project" command opens Xcode but does not actually load the project
3102	4.0	Proxy settings are not taken from worklight.properties file.
3114	4.1.1	See 2869

ID	Applicable To	Description
3117	4.0	The URLs created in the Worklight Console for embedded and Facebook apps should not point to the public resource server.
3119	4.0.3	Android options menu images are not rendered in the appropriate size.
3160	4.0.3	If deploying an app to the Worklight Server fails, the original app that had been previously deployed on the Server is also deleted.
3171	4.0.3	In the Android options menu, a 72-pixel image hides the option text on some devices.
3235	4.0	When previewing an iOS app, the application tab bar is not accessible.
3236	4.0	When previewing an Android app, the application options menu is not accessible.
3274	4.1.2	Name of generated Xcode project should be <ProjectName><AppName><Env>, and not WorklightPhoneGapApp
3299	4.1	GadgetsRealm should be renamed to SampleAppRealm in authenticationConfig.xml and application descriptor.
3301	4.1.3	JavaScript errors appearing in the Android logcat are missing filename and line number indication.
3325	4.0	Android apps display a white screen during startup and a flickering "Loading" indication
3330	4.1.3	The Console allows saving Remote Disable/Notify rules with empty comments.
3347	4.0	"Remember me" works by default on apps with login on demand, causing the user identity to be remembered also after user's logged out from the app.
3349	4.1.3	Applications that include JavaScript files that depend in Prototype in the <head> tag might fail to work after upgrade, as the Worklight upgrade process moves the links to the Prototype scripts to the end of the <head> tag.
3362	4.1.2	WL.BusyIndicator options have no effect on iOS.
3363	4.1.3	With the inclusion of PhoneGap 1.0, Xcode 3.2.5 is no longer supported. The minimum supported Xcode version is 3.2.6.
3392	4.1.3	In the generated Xcode project, the script buildtime.sh is missing execute permissions, making it impossible to compile the project in Xcode.

ID	Applicable To	Description
3412	4.1.3	In Android applications, a space character in the application name fails the init process when assets are stored on SDCard.
3417	4.1.3	Wrong properties are displayed for the SQL adapter in the Worklight Console.

## Known Issues

ID	Reported In	Description
2286	4.1	Setting the address of one Worklight Server in the application descriptor, and then building it and deploying it to another Server causes an exception, instead of a proper error message
2687	4.1	Skins not used by the app on the device still get downloaded when the app resources are directly updated
2715	4.1	If an app does not include skins and was installed on a device, newly added skins will not be available by directly updating the app's web resources.  Workaround: Delete the app from the device and re-install it
2905	4.1.1	It is not possible to increase the version of desktop and web widgets.
3373	4.1.2	The timestamp of raw reporting data in Oracle is always '00:00:00'.
3396	4.2	To properly create the Windows Phone environment in the Worklight Studio, Eclipse's Build Automatically option must be enabled.
3423	4.2	When remotely disabling an app and specifying a URL for the new app version, iOS's itms:// protocol is handled correctly. Http:// must be used instead.
3452	4.2	For native iOS apps, the existence of the application version property in the application's .plist file is not enforced, enabling apps to run without indicating to which version they belong.
3538	4.2	Using Xcode 4.2, the Worklight native Objective-C SDK cannot be dragged into the application project. The workaround is to drag it to the project using Xcode 4.1, and then re-open the project in Xcode 4.2.
3553	4.2	Creating a BlackBerry environment requires the BlackBerry plug-in to be pre-installed on Eclipse.
3554	4.2	On startup of mobile apps on iPhone and iPad, a white screen and/or a busy indicator are displayed between the splash screen and the full application.
3557	4.2	On Windows Vista and 7 gadgets, the dock image is surrounded by incorrect background.