BlackBerry Dynamics Bindings for Microsoft.iOS

Development Guide

12.0

Contents

What is the BlackBerry Dynamics SDK? 5

Reliance on the underlying BlackBerry Dynamics SDK 5
BlackBerry Dynamics API reference 5
Key features of the BlackBerry Dynamics SDK 5
Activation 6
Secure storage 7
Secure communication 7
Shared Services Framework 8
Data Leakage Prevention 9
User authentication 9
Administrative controls 11
Advanced security features with CylancePROTECT Mobile 11
Data collection and metrics with BlackBerry Analytics 11

Requirements and support for platform-specific features 12

Software requirements 12
Using an entitlement ID and version to uniquely identify a BlackBerry Dynamics app 13
Relationship between the entitlement ID and version and native identifiers 13
FIPS compliance 14
Declaring a URL type to support BlackBerry Dynamics features 14
App UI restrictions 15
Requirements and prerequisites for iOS platform features 15
Support for Touch ID 15
Support for Face ID 15
Support for WKWebView 15
Support for SFSafariViewController 20
Support for the Apple Universal Clipboard 20
Unsupported iOS features 20
Supported TLS protocols and cipher suites 21

Steps to get started with the BlackBerry Dynamics SDK 22

Install the BlackBerry Dynamics bindings for .NET 23
Using the BlackBerry Dynamics SDK framework 23
Prepare an existing BlackBerry Dynamics app to use the dynamic framework 23
Prepare an existing BlackBerry Dynamics app to use the NuGet packages 23
Configure a new or existing BlackBerry Dynamics app to use the dynamic framework 23
Developing apps in Microsoft Visual Studio on Windows with the pair to Mac feature 24
Provide the required project settings 24
Implement a BlackBerry Dynamics event listener 24
Implement the BlackBerry Dynamics Launcher 26

About the BlackBerry Dynamics SDK for Microsoft.Maui 28

Principal interfaces 28

Using the BlackBerry Dynamics SDK for Microsoft.Maui..... 28

Sample apps..... 30

Testing and troubleshooting..... 31
Troubleshooting common issues..... 31

Deploying your BlackBerry Dynamics app..... 32

Deploying certificates to BlackBerry Dynamics apps..... 33
Using Personal Information Exchange files..... 33
Configuring support for client certificates..... 34
Certificate requirements..... 34
Using Kerberos..... 35

Legal notice..... 36

What is the BlackBerry Dynamics SDK?

The BlackBerry Dynamics SDK provides a powerful set of tools that you can use to create secure productivity apps for a BlackBerry UEM domain. The SDK leverages the full capabilities of the secure BlackBerry Dynamics platform, so you can focus on building your apps rather than learning how to secure, deploy, and manage those apps.

The BlackBerry Dynamics SDK is available for all major development platforms. It allows you to leverage many valuable services, including secure communication, securing data in file systems and databases, inter-app data exchange, presence, push, directory lookup, single sign-on authentication, identity and access management, and more.

This guide will provide:

- Information about supported features
- Development requirements and prerequisites
- Instructions for installing, configuring, and using the SDK
- Considerations for key platform features
- Information about the sample apps provided with the SDK
- Testing and troubleshooting guidance
- Guidance for deploying your app

This guide is intended for intermediate and experienced developers with an understanding of how to create apps for the intended platform. It is not a basic tutorial.

Reliance on the underlying BlackBerry Dynamics SDK

Apps developed with the Microsoft .NET bindings rely on the underlying BlackBerry Dynamics SDK. The full set of the SDK APIs is bound to a Microsoft C#/.NET counterpart. For example:

- BlackBerry Dynamics SDK for iOS Objective-C: `+ (void)initializeWithClassNameConformingToUIApplicationDelegate: (NSString*)applicationDelegate;`
- Microsoft C#/.NET: `static void InitializeWithClassNameConformingToUIApplicationDelegate (string applicationDelegate);`

BlackBerry Dynamics API reference

The BlackBerry Dynamics SDK API reference describes the available interfaces, classes, methods, and much more. The API reference for each SDK platform is available at https://developers.blackberry.com/us/en/resources/api-reference.html.

Key features of the BlackBerry Dynamics SDK

This section provides more information about key features of the BlackBerry Dynamics SDK. It does not detail the complete feature set. For more information about the full list of supported features and APIs, see the BlackBerry Dynamics SDK API reference for your platform.

For more information about the requirements and prerequisites to support platform-specific features, see Requirements and support for platform-specific features.

The implementation of some of the features discussed in this section will depend on how the UEM administrator has configured your organization's servers, network, and other infrastructure components. Contact the administrator to clarify whether there are components of a feature that are configured or managed using the management server.

Activation

Infrastructure and enterprise activation

After a BlackBerry Dynamics app is installed on a user's device, the user must activate the app in order to use it. The activation process registers the app with the management server and gives the app access to the full capabilities of the BlackBerry Dynamics platform. The activation process ensures that all end users are fully authorized and permitted to use the app.

Users can activate a BlackBerry Dynamics app manually using an activation password, QR code, or access key provided by the administrator or obtained from UEM Self-Service, by using the UEM Client, through a third-party IDP, such as Active Directory or Okta, or by using the Easy Activation feature described below.

For more information about activating BlackBerry Dynamics apps, see the Activation section in the GDAndroid class reference or GDiOS class reference and Managing BlackBerry Dynamics apps in the UEM Administration content.

Easy Activation

Easy Activation simplifies the process of activating multiple BlackBerry Dynamics apps on a user's device. With Easy Activation, a device user only needs to activate the first BlackBerry Dynamics app on their device; when the user installs additional BlackBerry Dynamics apps, the user can choose to delegate the activation process to the previously activated app. Any BlackBerry Dynamics app can be an activation delegate, but priority is given to the app that is configured as the authentication delegate.

Easy Activation is automatically enabled for all BlackBerry Dynamics apps that are produced by BlackBerry. To enable Easy Activation for your custom BlackBerry Dynamics app, the UEM administrator must specify the app package ID (Android) or bundle ID (iOS) in the BlackBerry Dynamics app settings in the management console. Contact your organization's administrator to provide this information. For instructions for specifying the package ID or bundle ID for an app, see Manage settings for a BlackBerry Dynamics app in the UEM Administration content.

On the application side, Easy Activation is enabled by default by the BlackBerry Dynamics Runtime.

For more information, see the Easy Activation section in the BlackBerry Dynamics Security White Paper.

iOS user enrollment and DEP activation enhancements

The BlackBerry Dynamics SDK for iOS version 8.0 and later and UEM version 12.13 and later feature the following activation enhancements for iOS user enrollment and DEP:

- MDM enrollment and the activation of BlackBerry Dynamics apps doesn't require the UEM Client.
- After a new device is enrolled on UEM, UEM prompts the user to install the BlackBerry Dynamics app that is configured as the authentication delegate (that app must be assigned to the user). When the user opens this app for the first time, it activates automatically. The user can then activate additional BlackBerry Dynamics apps using Easy Activation.

Programmatic activation

The programmatic activation feature enables a BlackBerry Dynamics app to activate without any user interaction and without displaying activation prompts or progress screens. This can be useful when targeting your apps to a consumer audience or for developing apps for devices that have limited or no means of user input.

For more information about programmatic activation, see `programmaticActivityInit` in the BlackBerry Dynamics SDK for Android API Reference or `programmaticAuthorize` in the BlackBerry Dynamics SDK for iOS API Reference.

Note the following implementation details:

- Decide whether you want users to specify a password to unlock a BlackBerry Dynamics app after the initial activation. You can use programmatic activation while still requiring users to type a password to unlock the app. This setting is configured in a BlackBerry Dynamics profile in UEM.
- To activate the app, your application server must use the BlackBerry Web Services REST APIs to retrieve the user credentials and to generate an access key. You may need to create a new UEM user account or to lookup an existing user account. See the User resource in the BlackBerry Web Services REST API reference for the available REST APIs that can be used to create a user, lookup a user, and to generate an access key.
- Pass the user credentials to the app and call `programmaticActivityInit` or `programmaticAuthorize` with the retrieved credentials, setting `ShowUserInterface` to false.
- For Android, receive the broadcast event `GD_STATE_ACTIVATION_ACTION` to track activation progress from `NotActivated` to `InProgress` to `Activated`. You can choose to display a progress indicator during this short period.
- For iOS, observe `GDState.BBActivationState` to track activation progress from `NotActivated` to `InProgress` to `Activated`. You can choose to display a progress indicator during this short period.
- Once activation completes, the user is prompted to set a password (unless you've configured the BlackBerry Dynamics profile to not require a password). The app should wait for the `GD_STATE_AUTHORIZED_ACTION` notification.
- You can use `configureUI` (Android/iOS) to customize the UI of the password screen (for example, with a custom logo and colors).

Secure storage

Secure file system

BlackBerry Dynamics apps store data in a secure, encrypted file system. For more information, see BlackBerry Dynamics File I/O Package in the BlackBerry Dynamics SDK for Android API reference, or GDFileManager and GDFileHandle in the BlackBerry Dynamics SDK for iOS API reference.

Core data

BlackBerry Dynamics apps can store Core Data objects in a secure, encrypted store. For more information, see GDPersistentStoreCoordinator in the API reference.

Secure SQL database

BlackBerry Dynamics apps can leverage a secure SQL database that stores and encrypts data on the user's device. The secure SQL database is based on the SQLite library.

For more information, see the BlackBerry Dynamics SQL Database page in the BlackBerry Dynamics SDK API reference (Android/iOS).

Secure communication

The BlackBerry Dynamics platform enables secure data exchange between a BlackBerry Dynamics app on an end user's device and a back-end application server on the Internet or behind the enterprise firewall. Any communication through the enterprise firewall uses the secure BlackBerry Dynamics proxy infrastructure. One app can communicate with multiple application servers.

To learn more about the programming interfaces for secure communication, see GDSocket and GDHttpClient in the BlackBerry Dynamics SDK for Android API reference, or GDSocket, GDURLLoadingSystem, and NSURLSession Support in the BlackBerry Dynamics SDK for iOS API reference.

AppKinetics

AppKinetics, or Inter-Container Communication (ICC), is a method for securely exchanging data and commands between two BlackBerry Dynamics apps on the same device. The exchange uses a consumer-provider model: one app initiates a service request that the other app receives and responds to as a service provider.

For more information about AppKinetics, see the Inter-Container Communication Package in the BlackBerry Dynamics SDK for Android API reference, or GDService and GDServiceClient in the BlackBerry Dynamics SDK for iOS API reference.

Shared Services Framework

BlackBerry Dynamics apps can communicate with each other and application servers using the Shared Services Framework, a collaboration system that is defined by two components: one that provides a service and another that consumes the service.

The provider can be a client-side service, which is a BlackBerry Dynamics app that uses the GDService APIs (Android/iOS), or a server-side service that is provided by an application server or other remote system. The service is consumed by a BlackBerry Dynamics app that communicates with the provider using AppKinetics (a proprietary BlackBerry ICC protocol) for client-side services or a protocol such as HTTPS for server-side services.

The typical steps that are required to consume a service:

1. Service discovery: The BlackBerry Dynamics app (the consumer) queries for service providers using the GDAndroid.getServiceProvidersFor API or the GDiOS.getServiceProvidersFor API. Service discovery is optional but recommended for both types of services because it respects user entitlements and permissions.
2. Provider selection: The consuming app selects the provider. This is handled by the app code.
3. Service request: The consuming app sends a service request to the provider using the GDServiceClient API (Android/iOS) for client-side services or TCP sockets or HTTP over BlackBerry Dynamics secure communication (Android/iOS) for server-side services.
4. Service response: The consuming app receives the provider response using the same interface that was used for the request (the GDServiceClient API (Android/iOS) for client-side services or BlackBerry Dynamics secure communication (Android/iOS) for server-side services).

Client-side services can be used offline and are ideal if the service requires specific user interaction.

Server-side services can be provided by a clustered application server and are ideal if the server software already exists outside of the BlackBerry Dynamics platform.

Client-side and server-side services both require user entitlement in the management console.

If you want your custom BlackBerry Dynamics app to use the Shared Services Framework, the UEM administrator must specify the app package ID (Android) or bundle ID (iOS) in the BlackBerry Dynamics app settings in the management console. Contact your organization's administrator to provide this information. For instructions for specifying the package ID or bundle ID for an app, see Manage settings for a BlackBerry Dynamics app in the *UEM Administration Guide*.

Sample apps that are included with the SDK demonstrate how to use the Shared Services Framework. For more information about how to use the Shared Services Framework, see the following resources:

- GDAndroid.getServiceProvidersFor
- GDiOS.getServiceProvidersFor
- GDService (Android/iOS)
- GDServiceClient (Android/iOS)
- BlackBerry Dynamics Service Definition (Android/iOS)
- Definitions and descriptions of published services

Server-side services can use the Push Channel API (Android/iOS) to send notifications to BlackBerry Dynamics apps. The channel is end-to-end secure at the same level as BlackBerry Dynamics secure communication. As

a result, the BlackBerry Dynamics app does not need to poll the application server, which decreases the load on both the app and the application server. Any application server that is a service provider can use the Push Channel.

Data Leakage Prevention

The BlackBerry UEM administrator can use Data Leakage Prevention (DLP) settings in BlackBerry Dynamics profiles (UEM) to configure data protection standards, including enabling or disabling copy and paste between BlackBerry Dynamics apps and non-BlackBerry Dynamics apps, screen captures, dictation, FIPS, and more.

Contact your organization's administrator to configure DLP standards as necessary for your custom BlackBerry Dynamics apps.

Note the following for the different platforms of the SDK:

SDK platform	Notes
BlackBerry Dynamics SDK for iOS	The BlackBerry Dynamics Runtime can secure or block text in transit to or from the clipboard, depending on the configuration of the DLP settings. The Runtime secures text by encrypting it when it is cut or copied to the clipboard and decrypting it when it is pasted. These operations are handled automatically by the Runtime, so no development changes are required.

User authentication

BlackBerry UEM offers the following options to adjust the user experience for accessing BlackBerry Dynamics apps.

Fingerprint and biometric authentication

Various forms of biometric authentication are supported by the BlackBerry Dynamics SDK, including fingerprint authentication and for Android and Touch ID and Face ID for iOS. The BlackBerry UEM administrator can use a BlackBerry Dynamics profile (UEM) to enable biometric authentication. Contact your organization's administrator to enable and configure these features.

For more information, see BlackBerry Dynamics and Fingerprint Authentication.

Authentication delegation

The BlackBerry UEM administrator can configure up to three BlackBerry Dynamics apps on users' devices to act as an authentication delegate (a primary, secondary, and tertiary delegate). When a user opens any BlackBerry Dynamics app, the device will display the login screen of the authentication delegate app. After the user logs in successfully, all of the BlackBerry Dynamics apps on the device are unlocked. The user does not need to enter a password again until the idle timeout is reached.

If you want your custom BlackBerry Dynamics app to be an authentication delegate, the UEM administrator must specify the app package ID (Android) or bundle ID (iOS) in the BlackBerry Dynamics app settings in the management console. Contact your organization's administrator to provide this information. For instructions for specifying the package ID or bundle ID for an app, see Manage settings for a BlackBerry Dynamics app in the *UEM Administration Guide*.

The administrator configures one or more authentication delegate using a BlackBerry Dynamics profile. It is a best practice to configure the most commonly used app as the authentication delegate. Contact your organization's administrator to configure one or more authentication delegates.

Note: If the administrator configures a secondary authentication delegate, the administrator must notify users that if they delete the primary authentication delegate app, the user must unlock the secondary delegate app and

set the app password again so that it can be used to authenticate any additional BlackBerry Dynamics apps. The same requirement applies if a tertiary delegate is configured and the primary and secondary delegate apps are deleted.

Do not require a password

Enabled using a BlackBerry Dynamics profile, this setting removes the password login for BlackBerry Dynamics apps. Users cannot choose whether to use a password.

Do not enable authentication delegation and this setting in the same profile or policy set. This feature is supported in UEM 12.7 or later. If the setting is enabled and then disabled at a later date, users are prompted to create a password the next time they log in to a BlackBerry Dynamics app.

You can use the GDAndroid.getInstance().canAuthorizeAutonomously() or [GDiOS sharedInstance].canAuthorizeAutonomously method to check if this feature is enabled. See the GDInteraction sample app (Android) or the SecureStore sample app (iOS) for examples of this method.

Bypass the app unlock screen

Enabled in the UEM Client settings for a specific BlackBerry Dynamics app (UEM), this setting allows an app to completely bypass the password login screen.

For more information and programming guidance, see the Bypass Unlock Developer Guide.

Background Authorize for iOS

Background Authorize is a restricted API that allows a recently locked BlackBerry Dynamics app to use the principal BlackBerry Dynamics APIs (such as secure storage and secure communication) when the app is running in the background.

This feature can be useful in scenarios where the app has stopped unexpectedly and is started in the background in response to an APNS message (for example, a new email). If Background Authorize is enabled, the app can download new data and store it in the secure container. When the user brings the app to the foreground they can authorize and immediately access the data (for example, messages).

To access this restricted API, submit a request to the BlackBerry Dynamics Registrar program at BlackBerryDynamicsRegistrar@blackberry.com.

For more information about this feature, see the Background Authorize Developer Guide.

Background Authorize for Android

GDAndroid.canAuthorizeAutonomously allows BlackBerry Dynamics apps to background unlock, receive state callback, and use credential-protected storage. The app can use canAuthorizeAutonomously() to check if it is possible to use background unlock, and if possible, authorize with serviceInit().

Web Authentication

The SDK supports ASWebAuthenticationSession. The BlackBerry Dynamics implementation of ASWebAuthenticationSession utilizes BlackBerry Dynamics secure communication and secure storage for cookies. To protect enterprise credentials from being stored in the iOS keychain, the device user will not be able to use the Safari saved passwords feature in the embedded webview.

Initialize an instance of ASWebAuthenticationSession in your app to allow user authentication through a web service, including those operated by a third party. The page will open in a secure, embedded webview in iOS, or the users default browser (if it supports web authentication sessions) on macOS. For more information, see Authenticating a User Through a Web Service.

You can use Single Sign-On (SSO) with ASWebAuthenticationSession applications by enabling keychain group sharing and using the com.good.gd.data group. The prefersEphemeralWebBrowserSession property will be set to YES by the SDK.

Administrative controls

The BlackBerry UEM administrator can use various server settings, policies, and profiles to manage BlackBerry Dynamics apps and ensure that app usage meets the organization's security standards. Consult with your organization's administrator to ensure that your custom apps adhere to the configured settings in the management console.

For more information, see Controlling BlackBerry Dynamics on users devices, Enforcing compliance rules for devices, and Managing BlackBerry Dynamics apps in the *UEM Administration Guide*.

You also have the option to add new management policies and settings that are specific to your custom BlackBerry Dynamics app to the UEM management console so that they can be configured and applied to users by UEM administrators. For more information, see Adding custom policies for your app to the UEM management console.

Advanced security features with CylancePROTECT Mobile

The BlackBerry Dynamics SDK integrates the CylancePROTECT library to support CylancePROTECT Mobile for UEM. CylancePROTECT is a licensed service that offers a suite of features that enhances UEM's ability to detect, prevent, and resolve security threats without disrupting the productivity of your workforce. CylancePROTECT is configured and managed by the UEM administrator. No additional development or integration effort is required if your organization wants to leverage the CylancePROTECT features for custom BlackBerry Dynamics apps.

CylancePROTECT uses a combination of advanced technologies, including:

- The cloud-based CylanceINFINITY service that uses sophisticated AI and machine learning to identify malware and unsafe URL
- The UEM server that provides a complete device management and compliance infrastructure for your organization
- BlackBerry apps that monitor and enforce security standards at the device and user level

The seamless integration of these technologies establishes a secure ecosystem where data is protected and malicious activities are identified at all endpoints and eliminated proactively.

CylancePROTECT includes the following features:

- Malware detection for Android apps (including BlackBerry Dynamics apps) that are uploaded to UEM for internal deployment
- Malware detection on Android devices
- Sideloaded app detection on iOS and Android devices
- Safe browsing with BlackBerry Dynamics apps
- Insecure network detection on iOS and Android devices.
- Insecure Wi-Fi access point detection on Android devices.
- Integrity checking for BlackBerry Dynamics apps on iOS devices using the Apple DeviceCheck framework
- Hardware certificate attestation for BlackBerry Dynamics apps on Android devices

For more information about CylancePROTECT, see the BlackBerry Protect documentation.

Data collection and metrics with BlackBerry Analytics

BlackBerry Analytics is a cloud-based portal that you can use to view information about the BlackBerry Dynamics apps and devices that are used in your organization's environment. Previously, the BlackBerry Analytics functionality was offered in a separate SDK that you could integrate with your BlackBerry Dynamics apps. In SDK version 8.0 and later, BlackBerry Analytics functionality is now included in the BlackBerry Dynamics SDK.

For more information about BlackBerry Analytics, see the BlackBerry Analytics documentation. For more information about integrating BlackBerry Analytics with your BlackBerry Dynamics apps, see Integrating BlackBerry Analytics.

Requirements and support for platform-specific features

This section provides the software requirements for using the SDK, as well as prerequisites that are required to support platform-specific features. For more information, see the BlackBerry Dynamics SDK for iOS Development Guide.

Software requirements

Item	Requirement
Compatibility with previous releases of the BlackBerry Dynamics Bindings for Microsoft.iOS	This release of the BlackBerry Dynamics Bindings for Microsoft.iOS is compatible with the 10.1.x, 11.0.x, and 11.1.x releases of the SDK. However, the project must be migrated from Xamarin.iOS to Microsoft.iOS. For more information, see Upgrade from Xamarin to .NET in the Microsoft documentation.
BlackBerry Dynamics SDK for iOS	The latest compatible version of the BlackBerry Dynamics SDK for iOS is bundled with the SDK.
Deployment target	iOS 15 or later
Microsoft.iOS	16.4.1707 or later
Microsoft Visual Studio for macOS	17.6.6 or later
macOS	12.5 or later
Xcode	14 or 15
Character encoding for build files	Build files (for example, settings.json) must use UTF-8 character encoding. Verify that the editor that you plan to use does not add non UTF-8 characters or headers. In general, Java does not work with UTF-8- BOM (byte order mark).
BlackBerry Dynamics Launcher Library	The BlackBerry Dynamics Launcher is a user-friendly interface that allows users to easily access and switch between BlackBerry Dynamics apps, configure app settings, and take advantage of other useful features. For more information, see the BlackBerry Dynamics Launcher Framework documentation. The BlackBerry Dynamics SDK and the BlackBerry Dynamics Launcher Library are mutually dependent. See the BlackBerry Dynamics SDK for iOS Release Notes for the required version of the BlackBerry Dynamics Launcher Library.

Note: The key prefix "blackberry" is reserved by BlackBerry and should not be used for key values, key attributes, or key elements. For more information and examples, see the Application Policies Definition in the appendix of the API Reference.

Using an entitlement ID and version to uniquely identify a BlackBerry Dynamics app

BlackBerry Dynamics apps are uniquely identified by a BlackBerry Dynamics entitlement ID (GDApplicationID) and entitlement version (GDApplicationVersion). The entitlement ID and entitlement version are used to manage end-user entitlement in the management console. The values are also used for app publishing and service provider registration.

These values are specified in the assets/settings.json file for Android or in the Info.plist file for iOS.

You must define both the entitlement ID and the entitlement version for all of your BlackBerry Dynamics apps, regardless of whether you use the Shared Services Framework. The same entitlement ID must be used to represent the app across all platforms.

For more information about setting and checking the entitlement ID and version, the proper format to use for these values, and other requirements and considerations:

- See the GDAndroid Class reference in the BlackBerry Dynamics SDK for Android API Reference, especially these sections:
  - Identification
  - Indirect Authorization Initiation
  - void authorize (GDAppEventListener eventListener) throws GDInitializationError
- See the GDiOS Class reference in the BlackBerry Dynamics SDK for iOS API Reference, especially these sections:
  - Identification
  - Authorization
  - - (void) authorize: (id< GDiOSDelegate > _Nonnull) delegate

Relationship between the entitlement ID and version and native identifiers

The entitlement ID (GDApplicationID) and entitlement version (GDApplicationVersion) of a BlackBerry Dynamics app are different from the native identifiers that are required by the app OS platform. The native identifiers for Android are the packageName and packageVersion values in the AndroidManifest.xml file. The native identifiers for iOS are the CFBundleIdentifier and CFBundleVersion in the Info.plist file. The values of the entitlement ID and entitlement version and the platform native identifiers can be the same, but do not have to be.

To take advantage of BlackBerry Dynamics features such as Easy Activation, authentication delegation, certificate sharing, the Shared Services Framework, and more, the UEM administrator must specify the entitlement ID and version and the native identifier (package name or bundle ID) for a custom BlackBerry Dynamics app in the management console. For more information, see Add an internal BlackBerry Dynamics app entitlement and Manage settings for a BlackBerry Dynamics app in the *UEM Administration Guide*.

The native identifiers for your custom BlackBerry Dynamics app should be unique, especially with respect to apps that are available through public app stores. Duplicate native identifiers can prevent the proper installation or upgrade of an app.

You must change the native app version if you want to distribute a new version of the app. You only need to change the entitlement version if the app starts to provide a new shared service or shared service version, or if the app stops providing a shared service or shared service version.

FIPS compliance

It is a best practice to make your BlackBerry Dynamics apps compliant with U.S. Federal Information Processing Standards (FIPS) 140-2.The BlackBerry Dynamics SDK distribution contains FIPS canisters and tools.

The BlackBerry UEM administrator enables FIPS compliance using a BlackBerry Dynamics profile (UEM). If enabled, BlackBerry Dynamics apps must start in FIPS-compliant mode. The SDK determines whether a service is running in FIPS mode when the app communicates with the server to receive policies.

FIPS compliance enforces the following constraints:

- The use of MD4 and MD5 are prohibited. As a result, access to NTLM-protected or NTLM2-protected web pages and files is blocked.
- In secure socket key exchanges with ephemeral keys, with servers that are not configured to use Diffie-Hellman keys of sufficient length, BlackBerry Dynamics retries with static RSA cipher suites.

Note:

- When you enable FIPS compliance, user certificates must use encryption that meets FIPS standards. If a user tries to import a certificate with encryption that is not compliant, the user receives an error message indicating that the certificate is not allowed and cannot be imported.
- For iOS, when you build for testing with the x86 64-bit simulator, FIPS mode is not enforced. As a result, you might see a difference in behavior with the simulator compared to actual operation. BlackBerry recommends that you always test your app on actual iOS hardware and not rely exclusively on the simulation.

Declaring a URL type to support BlackBerry Dynamics features

A BlackBerry Dynamics app for iOS devices must declare a URL type so that it can be discovered by other apps on the same device. This enables AppKinetics, which is required for many BlackBerry Dynamics features. The URL type and schemes are declared in the app's Info.plist file.

The URL type must be the same as the app's native bundle ID. Within the URL type declaration, the following URL schemes must be declared. For example, if the native bundle ID of the app is `com.example.gd.myapp` and its entitlement version (GDApplicationVersion) is 1.0.0.0, then the declared URL type is `com.example.gd.myapp` and the schema declarations are as follows:

Format	Description	Example
`com.good.gd.discovery.enterprise`	Always required for enterprise apps (not required for ISV apps)	Exactly as shown
`<bundle_ID>.sc3`	Always required	`com.example.gd.myapp.sc3`
`<bundle_ID>.sc2`	Enables an app to use authentication delegation and is required for all BlackBerry Dynamics apps	`com.example.gd.myapp.sc2`
`<bundle_ID>.sc2.<GDApplicationVersion>`	Required only if your app provides a discoverable service	`com.example.gd.myapp.sc2.1.0.0.0`

App UI restrictions

The BlackBerry Dynamics Runtime monitors the app UI to enforce the configuration of enterprise profiles or policies from BlackBerry UEM. For example, a BlackBerry Dynamics profile may require the user to enter a password when the app transitions from the background to the foreground, or it may lock the app UI after a certain period of inactivity.

For a complete explanation of the restrictions and requirements that the app UI must follow, see Application User Interface Restrictions in the API Reference.

Requirements and prerequisites for iOS platform features

This section provides specific requirements or considerations to support features of the iOS platform in your BlackBerry Dynamics apps.

Support for Touch ID

Touch ID is a fingerprint recognition system for some iOS devices.

Touch ID can be allowed for user authentication in BlackBerry Dynamics apps in addition to standard password authentication. For more information about Touch ID, see BlackBerry Dynamics and Fingerprint Authentication in the BlackBerry Developers for Enterprise Apps portal.

Support for Face ID

The BlackBerry Dynamics SDK for iOS version 4.0 and later supports Face ID. An administrator can enable or disable the feature in a BlackBerry Dynamics profile.

Each app must add the NSFaceIDUsageDescription key to the Info.plist file. For more details about Face ID, see the Build-Time Configuration appendix in the API Reference.

Support for WKWebView

The BlackBerry Dynamics SDK for iOS version 4.2 and later supports secure WKWebView for displaying interactive web content.

Note the following support details:

- The SDK supports multiple WKWebView instances. The instances must be created programmatically.
- The SDK supports loading WKWebView from UIStoryBoard. To avoid any possible data leaks, you must load UIStoryBoard with the WKWebView component after the SDK is initialized.
- The supported versions of iOS require JavaScript injection by the BlackBerry Dynamics Runtime.
- The secure Fetch API is supported.
- Synchronous XMLHttpRequests are supported for iOS 12.2 and later, but the GET method is supported for iOS 13.1 and later only.
- The SDK supports the use of the cache to search for valid cached data for resources loaded by WKWebView.

The SDK's implementation of secure WKWebView currently supports:

- Loading HTTP and HTTPS data
- Redirection
- Basic, Digest, NTLM, Kerberos, and ClientCertificate authentication
- Cookies
- Video and audio playback

- Asynchronous XHR requests
- HTML5 non-persistent local storage
- HTML5 persistent local storage
- Sending the following types of body data using XMLHttpRequest: ArrayBuffer, Blob, FormData, URLSearchParams, USVString
- WebRTC
- HTTP/2

Note:

WebSockets are supported in WKWebView with the following limitations:

- The permessage-deflate extension is not supported.

The SDK's implementation of secure WKWebView does not currently support:

- The JavaScript sendBeacon API
- The following Data Leakage Prevention (DLP) settings from BlackBerry UEM for long-press or 3D touch actions:
  - Do not allow copying data from BlackBerry Dynamics apps into non BlackBerry Dynamics apps
  - Do not allow copying data from non BlackBerry Dynamics apps into BlackBerry Dynamics apps
- HTML attributes for a link tag (for example, preconnect)

WKWebView: Unsupported methods and properties

Class: NSAttributedString (NSAttributedStirngWebKitAdditions)

Method	Details
loadFromHTMLWithFileURL:options:completionHandler:	Files can be loaded from a bundle or from the App folder, but can't be loaded from an encrypted, secured container that is protected by the SDK.

Class: WKContentWorld

WKContentWorld is new in iOS 14 and is not yet supported by the SDK. The SDK supports only WKContentWorld.pageWorld:.

Class: WKDownload

WKDownload is new in iOS 14.5 and is not yet supported by the SDK. If a developer attempts to use WKDownload via a 3rd party framework, the console will produce the following error messages:

- `WKGD: Attempt to use WKDownload interface detected! Dynamics SDK does not support WKDownload.`
- `WKGD: Attempt to start download via WKDownload interface detected! Dynamics SDK does not support WKDownload.`
- `WKGD: Attempt to resume download via WKDownload interface detected! Dynamics SDK does not support WKDownload.`

Class: WKDownloadDelegate

WKDownloadDelegate is new in iOS 14.5 and is not yet supported by the SDK.

Class: WKFindConfiguration

WKFindConfiguration is new in iOS 14 and is not yet supported by the SDK.

Class: WKFindResult

WKFindResult is new in iOS 14 and is not yet supported by the SDK.

Class: WKNavigationAction

Method or property	Details
shouldPerformDownload	This property is new in iOS 14.5 and is not supported by the SDK.

Class: WKNavigationDelegate

Enumeration case or method	Details
WKNavigationActionPolicyDownload	This enumeration case is new in iOS 14.5 and is not supported by the SDK.
WKNavigationResponsePolicyDownload	This enumeration case is new in iOS 14.5 and is not supported by the SDK.
webView(_:authenticationChallenge:shouldAllowDeprecatedTLS:)	This method is new in iOS 14 and is not yet supported by the SDK.

Class: WKPDFConfiguration

WKPDFConfiguration is new in iOS 14 and is not yet supported by the SDK.

Class: WKPreferences

Method	Details
fraudulentWebsiteWarningEnabled	The SDK always sets this value to NO and the user can't change this value. The SDK provides an alternative mechanism to validate URLs. For more information, see Safe browsing with BlackBerry Dynamics apps.

Class: WKUIDelegate

Method	Details
webView:requestDeviceOrientationAndMotionPermissionForOrigin: initiatedByFrame:decisionHandler:	This method is not supported by the SDK.

Class: WKUserContentController

Method	Details
addScriptMessageHandler:contentWorld:name:	This method is new in iOS 14. The SDK supports only WKContentWorld.pageWorld.
addScriptMessageHandlerWithReply: contentWorld:name:	This method is new in iOS 14. The SDK supports only WKContentWorld.pageWorld.
removeAllScriptMessageHandlersFromContentWorld:	New in iOS 14. You cannot use this method with WKContentWorld.pageWorld. If you need to remove the script message handler from WKContentWorld.pageWorld, remove the handler by name using removeScriptMessageHandlerForName:contentWorld:.
removeAllScriptMessageHandlers	New in iOS 14. You should remove the handler by name using removeScriptMessageHandlerForName:contentWorld:.
removeAllUserScripts()	The SDK injects its own scripts, so calling this method will break how the SDK supports WKWebView.

Class: WKWebSiteDataStore

Method	Details
getCookiePolicy	This method is new in iOS 17 and is not yet supported by the SDK.
getCookiePolicy:completionHandler	This method is new in iOS 17 and is not yet supported by the SDK.
proxyConfigurations	This method is new in iOS 17 and is not yet supported by the SDK.

Class: WKWebView

Method or property	Details
evaluateJavaScript:inFrame:inContentWorld: completionHandler:	This method is new in iOS 14. The SDK supports only WKContentWorld.pageWorld.
callAsyncJavaScript:arguments:inFrame:inContentWorld: completionHandler:	This method is new in iOS 14. The SDK supports only WKContentWorld.pageWorld.
createPDFWithConfiguration:completionHandler:	After calling this function, the user gets NSData in the completion handler. To store data in the secure container, use GDFileManager. To securely send data through the network, use NSURLSession.
createWebArchiveDataWithCompletionHandler:	After calling this function, the user gets NSData in the completion handler. To store data in the secure container, use GDFileManager. To securely send data through the network, use NSURLSession.
findString:withConfiguration:completionHandler:	This method is new in iOS 14 and is not yet supported by the SDK.
loadFileURL(_:allowingReadAccessTo:)	Files can be loaded from a bundle or from the App folder, but they can't be loaded from an encrypted, secured container that is protected by the SDK.
loadFileRequest:allowingReadAccessToURL:	Files can be loaded from a bundle or from the App folder, but can't be loaded from an encrypted, secured container that is protected by the SDK.
mediaType	This property is new in iOS 14 and is not yet supported by the SDK.
printOperationWithPrintInfo:	This method is for macOS only and is not currently supported by the SDK.
resumeDownloadFromResumeData:completionHandler:	This method is new in iOS 14.5 and is not supported by the SDK. The SDK generates a warning when it is called.
serverTrust	The SDK returns nil for this property.
startDownloadUsingRequest:completionHandler:	This method is new in iOS 14.5 and is not supported by the SDK. The SDK generates a warning when it is called.

Method or property	Details
uiDelegate	The SDK returns its own internal delegate. If you set a custom delegate the property will work as expected.

Class: WKWebViewConfiguration

Method or property	Details
limitsNavigationsToAppBoundDomains	This property is new in iOS 14 and is not currently supported by the SDK.

Support for SFSafariViewController

The BlackBerry Dynamics SDK for iOS supports secure SFSafariViewController for displaying web interfaces within BlackBerry apps.

Note the following support details:

- The use of SFSafariViewController to load a preview of a web page is secured with BlackBerry Dynamics secure communication. You must add SafariServices.framework to your Xcode project.
- The SDK does not support signing in with SFSafariViewController to Microsoft Entra ID Connect (using MSAL) is not supported.

Support for the Apple Universal Clipboard

The SDK supports secure cut, copy, and paste operations using the Apple Universal Clipboard. Users can copy and move data between BlackBerry Dynamics apps and devices that follow Apple Continuity system requirements. The apps and devices must be activated for the same user account on the same instance of BlackBerry UEM.

For more information about the Apple Universal Clipboard, see support.apple.com to read Set up Universal Clipboard.

Unsupported iOS features

The following features are not supported by the BlackBerry Dynamics SDK:

Feature	Details
BitCode	BitCode is an intermediate, architecture-independent binary object format that allows developers to submit a "machine neutral" app to Apple for a final, architecture-dependent build. The intermediate nature of BitCode is not compatible with the cryptographic requirements of the BlackBerry Dynamics SDK, which relies on the delivery of libraries or other modules that are cryptographically signed at build-time. The signatures of the libraries and modules are validated at runtime to ensure integrity. If you enable BitCode and try to build an app, a build error will indicate that the SDK does not support BitCode.

Feature	Details
App Extensions	The BlackBerry Dynamics SDK does not support app extensions such as Sirikit.

Supported TLS protocols and cipher suites

The SDK uses CURL 8.0.0, supporting the TLS protocols and cipher suites of TLS version 1.3.

Steps to get started with the BlackBerry Dynamics SDK

Step	Action
1	• Install the BlackBerry Dynamics bindings for .NET • Using the BlackBerry Dynamics SDK dynamic framework **Note:** You no longer need to install the BlackBerry Dynamics SDK for iOS static framework.
2	If applicable, see Developing apps in Microsoft Visual Studio on Windows with the pair to Mac feature.
3	If applicable, Configure a new or existing BlackBerry Dynamics app to use the dynamic framework.
4	Provide the required project settings.
5	Optionally, implement an event listener. For more information about an alternative method to implement the life cycle of events, see "Add the event-handler skeleton using notifications" in the BlackBerry Dynamics SDK for iOS Development Guide. The GreetingsServer sample app demonstrates this alternative.
6	Optionally, implement the BlackBerry Dynamics launcher.
7	Consult the appropriate BlackBerry Dynamics SDK API reference for instructions for implementing the desired features of the BlackBerry Dynamics platform. See the sample apps included in the SDK package for examples of how to implement key features. If you want to use Microsoft.Maui, see About the BlackBerry Dynamics SDK for Microsoft.Maui and the corresponding API reference.
8	Test and debug your app.
9	Deploy your app.
10	Optionally, deploy certificates to the BlackBerry Dynamics apps on users' devices.

Install the BlackBerry Dynamics bindings for .NET

The bindings are delivered as a local NuGet package distribution with the required resources. The package includes .NET/Microsoft.iOS projects for the BlackBerry Dynamics bindings, for each of the sample apps, and for the BlackBerry Dynamics Launcher Library.

Before you begin: Visit BlackBerry Developer Downloads to download the BlackBerry Dynamics Bindings package for the desired platform. When you click the link, you are prompted to log in to the Developer site with your BlackBerry Online Account. If you don't already have an account, you can register and create one.

Extract the contents of the BlackBerry Dynamics Bindings SDK package (the projects for the bindings package, sample apps, and if desired, BlackBerry Dynamics Launcher Library).

After you finish: It is a best practice to use the delivered NuGet packages and corresponding resources directory that are included with every sample app instead of creating your own binding project.

Using the BlackBerry Dynamics SDK framework

The BlackBerry Dynamics SDK is currently offered as a dynamic framework, consisting of the BlackBerryDynamics.xcframework and two BlackBerryCerticom xcframeworks packed as NuGet packages. Follow the tasks below to configure your existing apps and new apps to use the BlackBerry Dynamics SDK.

Prepare an existing BlackBerry Dynamics app to use the dynamic framework

Complete the steps below if you have an existing BlackBerry Dynamics app that uses the static BlackBerry Dynamics SDK library (GD.Framework).

1. Use the following commands to uninstall the static library:
   a) $ cd ~/Library/Application\ Support/BlackBerry/Good.platform/iOS/
   b) $ sudo ./uninstall.sh
2. In the Microsoft Visual Studio Solution Explorer panel for your project, in the **Native References** folder, delete all the references to BlackBerryCerticom and BlackBerryCerticomSBGSE.
3. In the same location as your .csproj file, remove the Id file.

After you finish: Configure a new or existing BlackBerry Dynamics app to use the dynamic framework.

Prepare an existing BlackBerry Dynamics app to use the NuGet packages

Complete the steps below if you have an existing BlackBerry Dynamics app that uses the DLL binding libraries (GoodDynamics.iOS.dll / GoodDynamics.iOS.Launcher.dll).

1. Open your project in Microsoft Visual Studio.
2. In the **References** section, remove the GoodDynamics.iOS and GoodDynamics.iOS.Launcher references.
3. Remove the GoodDynamics.iOS.dll and GoodDyanmics.iOS.Launcher.dll files from the disc.
4. Visit https://learn.microsoft.com/en-us/dotnet/maui/migration/ and follow the instructions to upgrade from Xamarin to .NET.

Configure a new or existing BlackBerry Dynamics app to use the dynamic framework

Before you begin:

- If required, Prepare an existing BlackBerry Dynamics app to use the dynamic framework.
- If required, Prepare an existing BlackBerry Dynamics app to use the NuGet packages.
- Install the BlackBerry Dynamics bindings for .NET.

1. In Microsoft Visual Studio, open your project.
2. Configure the NuGet.Config file to point to the local source that contains the binding packages.
3. Right-click the **Dependencies** folder and click **Manage NuGet Packages...**.
4. Choose the package source that you specified in step 2, and choose the required items.
5. Follow the prompts to add the NuGet to your project.

After you finish: Provide the required project settings.

Developing apps in Microsoft Visual Studio on Windows with the pair to Mac feature

If you want to develop apps on a Windows computer using Microsoft Visual Studio with the Pair to Mac feature, visit the .NET development site and follow the instructions in Pair to Mac for iOS development.

The following table provides solutions for common issues when setting up the Pair to Mac feature:

Problem	Solution
You experience build and deployment errors.	See "Troubleshooting automatic Mac provisioning" on the Pair to Mac for iOS development page.

Provide the required project settings

1. In Microsoft Visual Studio, open your BlackBerry Dynamics-enabled Microsoft.iOS project.
2. In the project settings, select **Options** and navigate to **Build > iOS Build > Code Generation and Runtime > Linker behavior**.
3. Select **Link Framework SDKs Only** for all required configuration and platform options.
4. In the **Additional mtouch arguments** field, specify the following:

```
--registrar:static
```

5. Click **Save**. Note that more advanced examples of mtouch arguments can be found in the sample applications.

Implement a BlackBerry Dynamics event listener

The **Application** object manages the app's global app state. Many of the sample apps that are included with the BlackBerry Dynamics Bindings demonstrate the event handling lifecycle. This topic demonstrates one approach to implementing the lifecycle of events.

Before you begin: Define the BlackBerry Dynamics entitlement ID and entitlement version for your app.

1. Include the following code at the top of your **AppDelegate.cs** file to implement a skeleton `GDiOSDelegate` interface, set the app's window as a `GDWindow` (this example relies on a `_started` boolean variable), use `GDAppEvent` to process events, and move the app launch code from `FinishedLaunching` to the `HandleEvent` handler method.

```
//AppDelegate.cs
using System;
```

```
using UIKit;
using Foundation;
using System.Diagnostics
using GoodDynamics;
namespace MyApplication
{
    // The UIApplicationDelegate for the application. This class is responsible
 for launching the
    // User Interface of the application, as well as listening (and optionally
 responding) to application
    // events from iOS.
    [Register ("AppDelegate")]
    public class AppDelegate : GDiOSDelegate
    {
        private bool _started;

        public GDiOS GDLibrary { get; private set; }

        public override UIWindow Window
        {
            get;
            set;
        }

        public override bool FinishedLaunching (UIApplication application,
 NSDictionary launchOptions)
        {
            GDLibrary = GDiOS.GDSharedInstance;
            GDLibrary.Delegate = this;
            GDLibrary.Authorize();
            Window = UIApplication.SharedApplication.Delegate.GetWindow();
            Window.MakeKeyAndVisible ();
            return true;
        }

        public override void HandleEvent (GDAppEvent anEvent)
        {
            switch (anEvent.Type)
            {
                case GDAppEventType.Authorized:
                    OnAuthorized (anEvent);
                    break;
                case GDAppEventType.NotAuthorized:
                    OnNotAuthorized (anEvent);
                    break;
                case GDAppEventType.RemoteSettingsUpdate:
                    //handle app config changes
                    break;
                case GDAppEventType.ServicesUpdate:
                    Debug.WriteLine ("Received Service Update Event");
                    OnServiceUpdate (anEvent);
                    break;
                default:
                    Debug.WriteLine ("Event Not Handled");
                    break;
            }
        }

    ...

    }
```

```
}
```

2. Verify that your app can handle problems such as authorization errors or actions like remote wipe, lockout, or blocking events. Implement these events in the `OnNotAuthorized` function.

```
private void OnNotAuthorized(GDAppEvent anEvent)
{
    switch (anEvent.Code)
    {
        case GDAppResultCode.ErrorActivationFailed:
        case GDAppResultCode.ErrorProvisioningFailed:
        case GDAppResultCode.ErrorPushConnectionTimeout:
        case GDAppResultCode.ErrorSecurityError:
        case GDAppResultCode.ErrorAppDenied:
        case GDAppResultCode.ErrorAppVersionNotEntitled:
        case GDAppResultCode.ErrorBlocked:
        case GDAppResultCode.ErrorWiped:
        case GDAppResultCode.ErrorRemoteLockout:
        case GDAppResultCode.ErrorPasswordChangeRequired:
            Console.WriteLine ("OnNotAuthorized {0}", anEvent.Message);
            break;
        case GDAppResultCode.ErrorIdleLockout:
            break;
        default:
            Debug.Assert (false, "Unhandled not authorized event");
            break;
    }
}
```

3. On authorization, start the app. Initialize and start the app UI with the `OnAuthorized` function. The `GDAppResultCode.ErrorNone` event is returned by the BlackBerry Dynamics Runtime when a container is opened and no error occurs (you can test for it as an alternative to the boolean `_started` shown here).

```
private void OnAuthorized(GDAppEvent anEvent)
{
    switch (anEvent.Code)
    {
        case GDAppResultCode.ErrorNone:
            if (!_started)
            {
                _started = true;
                //Launch Application UI Here
            }
            break;
        default:
            Debug.Assert (false, "Authorized startup with an error");
            break;
    }
}
```

Implement the BlackBerry Dynamics Launcher

You have the option to implement the BlackBerry Dynamics Launcher, an intuitive front-end UI that makes it easy for device users to access and modify settings for BlackBerry Dynamics apps. For more information, see the documentation for the BlackBerry Dynamics Launcher Framework.

To implement the BlackBerry Dynamics Launcher, you add the BlackBerry Dynamics Launcher button to an activity and include its authentication logic in the appropriate place in your app.

Note: If you want to create your own version of the BlackBerry Dynamics Bindings for .NET with custom changes, open the Microsoft.iOS Launcher Bindings library project **GoodDynamics.iOS.Launcher** and check the reference to the Launcher library from the local framework path. You can find it under **Native References** in the Solution Explorer.

1. Download the BlackBerry Dynamics Launcher Bindings for .NET software from BlackBerry Developer Downloads.
2. In your IDE, load the NuGet binding packages into the desired projects. For each project, right-click **Dependencies** and click **Manage NuGet Packages...** to reference the Launcher binding.
3. Add the BlackBerry Dynamics Launcher button and initialize it with the following lines in the `OnAuthorized` method of your `AppDelegate`:

```
 . . .
var launcher = new GoodDynamics.iOS.Launcher.GTLauncherViewController(root);
Window.RootViewController = launcher;
launcher.StartServicesWithOptions
(GoodDynamics.iOS.Launcher.GTLauncherServicesStartupOptions.InternalGDAuthToken
AndPushConnectionManagement);
...
```

Here is a complete example of `OnAuthorized` with the BlackBerry Dynamics Launcher code in the `case` statement:

```
...
private void OnAuthorized(GDAppEvent anEvent)
{
    switch (anEvent.Code)
    {
        case GDAppResultCode.ErrorNone:
            UINavigationBar.Appearance.TintColor = UIColor.FromRGBA(52f / 255f,
 69f / 255f, 84f / 255f, 1);
            UINavigationBar.Appearance.SetTitleTextAttributes(new
 UITextAttributes()
            {
                TextColor = UIColor.FromRGBA(33f / 255f, 50f / 255f, 65f / 255f,
 1)
            });

            var root = new UINavigationController(new ProductsView());
            var launcher = new
 GoodDynamics.iOS.Launcher.GTLauncherViewController(root);

            Window.RootViewController = launcher;

 launcher.StartServicesWithOptions(GoodDynamics.iOS.Launcher.GTLauncherServicesStartupOption
            Window.MakeKeyAndVisible();
            break;
        default:
            Debug.Assert (false, "Authorized startup with an error");
            break;
    }
}
...
```

About the BlackBerry Dynamics SDK for Microsoft.Maui

Note: Due to the deprecation and upcoming end of support for Xamarin products, Microsoft has switched from Xamarin.Forms to Microsoft.Maui.

Microsoft.Maui is a framework that developers can use to create user interfaces that can be used across platforms. For more information, visit https://learn.microsoft.com/en-us/dotnet/maui/ to see Getting Started with Microsoft.Maui, What is .NET Maui?, and instructions to upgrade from Xamarin to .NET.

Visit BlackBerry Developer Downloads to download the BlackBerry Dynamics SDK for Microsoft.Maui package. See the BlackBerry Dynamics SDK for .NET Release Notes to review the lists of fixed and known issues in each release.

Principal interfaces

The SDK provides a unified API that supports the following principal interfaces:

- Runtime Object
- Secure Storage
  - Secure SQL Database
  - Secure File System
- Secure Communication
  - Socket
  - HTTP Communication
- Push Channel
- Inter-Application Data Exchange
- Single Sign-On

For complete details about each interface, see the BlackBerry Dynamics SDK for Microsoft.Maui API reference.

Using the BlackBerry Dynamics SDK for Microsoft.Maui

- Follow the requirements, prerequisites, and setup instructions for the BlackBerry Dynamics SDK for Android or the BlackBerry Dynamics SDK for iOS (or both, if you develop apps for both platforms).
- Follow the instructions in Steps to get started with the BlackBerry Dynamics SDK.
- Use the following Microsoft.Maui NuGet libraries that are distributed in the SDK package:
  - Microsoft.Maui NuGet common libraries: BBDXamarinForms.Common.Library
  - Microsoft.Maui platform NuGet libraries:
    - BBDXamarinForms.Droid.Library
    - BBDXamarinForms.iOS.Library
  - Microsoft.Android and Microsoft.iOS binding NuGet libraries:
    - GoodDynamics.Android
    - GoodDynamics.iOS

  The NuGet libraries listed above are distributed as a local .nupkg file source (located in the Platform/nuget subdirectory) with a NuGet.Config settings file in each of the sample apps for iOS and Android. You should also use the .csproj file as a reference for NuGet project dependency configuration, as Microsoft Visual Studio Dependency Manager does not always compile with the actual state.

- Review the sample projects in the BlackBerry Dynamics SDK for Microsoft.Maui package to familiarize yourself with the project structure and configurations.
  - BlankApp: The main project that contains application views that render on both platforms and allow the execution of the unified API. Demonstrates how to implement the BlackBerry Dynamics SDK event handling lifecycle. See the **App.xaml.cs** file and Implement a BlackBerry Dynamics event listener.
  - AppKinecticsService: Demonstrates how to write a server application that uses the BlackBerry Dynamics Inter-Container Communications API (AppKinetics). AppKinetics allows BlackBerry Dynamics apps running on the same device to exchange data and commands securely.
  - SampleAppSuite: Demonstrates how to use the rest of the available APIs.

  Note: The sample apps require the following NuGet packages that can be obtained using Microsoft Visual Studio: NLog, Unity.

Sample apps

The easiest way to get started with the BlackBerry Dynamics Bindings for Microsoft.iOS is to open one of the projects for sample apps. Uncompress the desired sample and double-click either the `.csproj` or `.sln` file to open the project.

Sample app	Description
RSS reader	Demonstrates how to use the BlackBerry Dynamics Secure Communication APIs to access resources behind the enterprise firewall.

Testing and troubleshooting

This section provides guidance for testing and troubleshooting issues with your BlackBerry Dynamics apps.

Troubleshooting common issues

Problem	Possible solution
After upgrading the BlackBerry Dynamics SDK for Microsoft.Maui, you get link errors that prevent you from building your project.	Verify that you upgrade to the latest supported version of the core BlackBerry Dynamics SDK for iOS or Android. See the software requirements.
Error: `This version of Xamarin.iOS requires the tvOS 13.4 SDK`	See https://github.com/xamarin/xamarin-macios/issues/8325.

Deploying your BlackBerry Dynamics app

Before you deploy your BlackBerry Dynamics app to your organization's production environment, you should test the app and the deployment process in a BlackBerry UEM environment that is reserved for development testing and evaluation. Coordinate with your organization's administrator to get access to a dedicated test environment.

BlackBerry Dynamics apps are fully supported for BlackBerry UEM. BlackBerry UEM is the recommended enterprise management solution to implement and use going forward, because it provides advanced app and user management features, advanced connectivity and networking options, expanded compliance and integrity checking, and the most recent BlackBerry Web Services REST APIs that your apps can leverage.

See the following resources for more information about distributing and managing your app in a BlackBerry UEM environment:

Task	Resource
Add your app to BlackBerry UEM and distribute it to users	See the following topics in the *BlackBerry UEM Administration Guide*: • Apps • Add an internal BlackBerry Dynamics app entitlement • Managing BlackBerry Dynamics apps
Configure BlackBerry Dynamics profiles that impact app functionality	See the following topics in the *BlackBerry UEM Administration Guide*: • Controlling BlackBerry Dynamics on users devices • BlackBerry Dynamics profile settings • BlackBerry Dynamics connectivity profile settings • Using profiles to manage device features
Collect activity and compliance violation information for BlackBerry Dynamics apps	See the following topic in the *BlackBerry UEM Administration Guide*: Export a BlackBerry Dynamics app report to a CSV file

Deploying certificates to BlackBerry Dynamics apps

You can use any of the following options to deploy certificates to BlackBerry Dynamics apps. Each method requires configuration in the management console. Coordinate with your organization's administrator to select and configure the desired option.

Option	For more information
Personal Information Exchange files	See Using Personal Information Exchange files in this section.
CA certificate profile	See Sending CA certificates to devices and apps in the *UEM Administration Guide*.
User credential profile	See Sending client certificates to devices and apps using user credential profiles in the *UEM Administration Guide*.
SCEP profile	See Sending client certificates to devices and apps using SCEP in the *UEM Administration Guide*.
Shared certificate profile	See Sending the same client certificate to multiple devices in the *UEM Administration Guide*.

After certificates are distributed to a user's device, those certificates are shared and used by all of the BlackBerry Dynamics apps on the device. No additional programming is required by the app developer to support client certificates.

The management server and BlackBerry Dynamics apps also support the use of Kerberos for service authentication. For more information, see Using Kerberos in this section.

The SDK also provides a Crypto C language programming interface that allows an app to retrieve public key certificates that are stored in the BlackBerry Dynamics credentials store and use those certificates for signing and verification of messages and documents such as PDFs. Note that BlackBerry Infrastructure certificates cannot be retrieved from the store and that the private key will remain inaccessible. For more information, see the Crypto C Programming Interface appendix (Android/iOS) in the API reference.

Using Personal Information Exchange files

An organization can deploy corporate services that require two-way SSL/TLS authentication for users. A user is issued a password-protected Personal Information Exchange file (PKCS12 format, .p12 or .pfx) containing an SSL/TLS client certificate and a private key. This file can be provided to BlackBerry Dynamics apps to grant access to secure corporate services.

The BlackBerry Dynamics SDK supports the use of Personal Information Exchange files to authenticate BlackBerry Dynamics apps and to access secure services. All of the required operations to support client certificates are carried out by the BlackBerry Dynamics Runtime, with no additional programming required to handle the authentication challenge. For more information on how this is handled, refer to *HttpViewController.swift* in the Dynamics-iOS-Swift sample app. The app can use client certificates if:

- The app uses the BlackBerry Dynamics Secure Communication Networking APIs.
- The device user's UEM account is configured to support certificates.
- The certificates satisfy the certificate requirements.

After a user activates a BlackBerry Dynamics app, the app receives the Personal Information Exchange files. For each file, the user is prompted to provide the issued password so that the files and identification material can be installed. When this process is complete, the app can access the server resources that require two-way SSL/TLS authentication.

If more than one Personal Information Exchange file is required per user, the BlackBerry Dynamics Runtime selects the appropriate certificate using the following criteria:

1. Only client certificates that are suitable for SSL/TLS client authentication are eligible to send to the server. Certificates must have no Key Usage or Extended Key Usage, or Key Usage that contains “Digital Signature" or "Key Agreement”, or Extended Key Usage that contains "TLS Web Client Authentication”. Key Usages and Extended Key Usages must not contradict allowances for SSL/TLS client authentication.
2. If the server advertises the client certificate authority in the SSL/TLS handshake, only client certificates that have been issued by that authority are considered.
3. Expired certificates and certificates that are not yet valid cannot be selected.
4. If more than one certificate satisfies the above criteria, the BlackBerry Dynamics Runtime selects the most recently issued certificate.

Configuring support for client certificates

Certificate support is configured in the management console by the administrator. Contact your organization’s administrator to configure certificate support for BlackBerry Dynamics apps.

For more information about configuring certificate support in BlackBerry UEM, see the following:

- Manage settings for a BlackBerry Dynamics app in the *UEM Administration Guide*
- Sending certificates to devices using profiles in the *UEM Administration Guide*
- Connect BlackBerry UEM to a BlackBerry Dynamics PKI Connector in the *UEM Administration Guide*

Certificate requirements

- Client certificates must be in PKCS12 format, with the Certificate Authority (CA), public key, and private key in the same file.
- The PKCS12 file must have a .p12 or .pfx extension
- The PKCS12 file must be password-protected
- The source of the certificate can be your own internal CA, a well-known public CA, or an online tool such as OpenSSL or the Java keytool. You can use the following keytool example to generate a certificate, substituting your own values as required:

```
keytool -genkeypair -alias good123 -keystore good123.pfx -storepass good123 -
validity 365 -keyalg RSA -keysize 2048 -storetype pkcs12
```

- If the organization’s security policy uses FIPS standards, Personal Information Exchange files must be encrypted with FIPS-strength ciphers. If Personal Information Exchange files use a weak cipher, which is common for third-party applications when exporting identity material, you can use a tool like OpenSSL to re-encrypt the files with a FIPS-strength cipher. See the following example:

```
openssl pkcs12 -in weak.p12 -nodes -out decrypted.pem
       <enter password>
       openssl pkcs12 -export -in decrypted.pem -keypbe AES-128-CBC -certpbe
 AES-128-CBC -out strong.p12
      <enter password>
          rm decrypted.pem
```

Using Kerberos

BlackBerry Dynamics apps support both Kerberos PKINIT with PKI certificates and Kerberos Constrained Delegation. Kerberos PKINIT and Kerberos Constrained Delegation are distinct implementations of Kerberos. You can support one or the other for BlackBerry Dynamics apps, but not both.

With Kerberos PKINIT, authentication occurs directly between the BlackBerry Dynamics app and the Windows Key Distribution Center (KDC). User authentication is based on certificates that are issued by Microsoft Active Directory Certificate Services. No additional programming is required by the app developer to use Kerberos PKINIT.

With Kerberos Constrained Delegation, authentication is based on a trust relationship between the management server (BlackBerry UEM and a KDC. The management server communicates with the service on behalf of the app.

For more information about how to configure the desired Kerberos implementation in UEM, including requirements and prerequisites, see Configuring Kerberos for BlackBerry Dynamics apps in the *UEM Administration Guide*.

Legal notice

WERE FORESEEN OR UNFORESEEN, AND EVEN IF BLACKBERRY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW IN YOUR JURISDICTION, BLACKBERRY SHALL HAVE NO OTHER OBLIGATION, DUTY, OR LIABILITY WHATSOEVER IN CONTRACT, TORT, OR OTHERWISE TO YOU INCLUDING ANY LIABILITY FOR NEGLIGENCE OR STRICT LIABILITY.

THE LIMITATIONS, EXCLUSIONS, AND DISCLAIMERS HEREIN SHALL APPLY: (A) IRRESPECTIVE OF THE NATURE OF THE CAUSE OF ACTION, DEMAND, OR ACTION BY YOU INCLUDING BUT NOT LIMITED TO BREACH OF CONTRACT, NEGLIGENCE, TORT, STRICT LIABILITY OR ANY OTHER LEGAL THEORY AND SHALL SURVIVE A FUNDAMENTAL BREACH OR BREACHES OR THE FAILURE OF THE ESSENTIAL PURPOSE OF THIS AGREEMENT OR OF ANY REMEDY CONTAINED HEREIN; AND (B) TO BLACKBERRY AND ITS AFFILIATED COMPANIES, THEIR SUCCESSORS, ASSIGNS, AGENTS, SUPPLIERS (INCLUDING AIRTIME SERVICE PROVIDERS), AUTHORIZED BLACKBERRY DISTRIBUTORS (ALSO INCLUDING AIRTIME SERVICE PROVIDERS) AND THEIR RESPECTIVE DIRECTORS, EMPLOYEES, AND INDEPENDENT CONTRACTORS.

IN ADDITION TO THE LIMITATIONS AND EXCLUSIONS SET OUT ABOVE, IN NO EVENT SHALL ANY DIRECTOR, EMPLOYEE, AGENT, DISTRIBUTOR, SUPPLIER, INDEPENDENT CONTRACTOR OF BLACKBERRY OR ANY AFFILIATES OF BLACKBERRY HAVE ANY LIABILITY ARISING FROM OR RELATED TO THE DOCUMENTATION.

Prior to subscribing for, installing, or using any Third Party Products and Services, it is your responsibility to ensure that your airtime service provider has agreed to support all of their features. Some airtime service providers might not offer Internet browsing functionality with a subscription to the BlackBerry® Internet Service. Check with your service provider for availability, roaming arrangements, service plans and features. Installation or use of Third Party Products and Services with BlackBerry's products and services may require one or more patent, trademark, copyright, or other licenses in order to avoid infringement or violation of third party rights. You are solely responsible for determining whether to use Third Party Products and Services and if any third party licenses are required to do so. If required you are responsible for acquiring them. You should not install or use Third Party Products and Services until all necessary licenses have been acquired. Any Third Party Products and Services that are provided with BlackBerry's products and services are provided as a convenience to you and are provided "AS IS" with no express or implied conditions, endorsements, guarantees, representations, or warranties of any kind by BlackBerry and BlackBerry assumes no liability whatsoever, in relation thereto. Your use of Third Party Products and Services shall be governed by and subject to you agreeing to the terms of separate licenses and other agreements applicable thereto with third parties, except to the extent expressly covered by a license or other agreement with BlackBerry.

The terms of use of any BlackBerry product or service are set out in a separate license or other agreement with BlackBerry applicable thereto. NOTHING IN THIS DOCUMENTATION IS INTENDED TO SUPERSEDE ANY EXPRESS WRITTEN AGREEMENTS OR WARRANTIES PROVIDED BY BLACKBERRY FOR PORTIONS OF ANY BLACKBERRY PRODUCT OR SERVICE OTHER THAN THIS DOCUMENTATION.

BlackBerry Enterprise Software incorporates certain third-party software. The license and copyright information associated with this software is available at http://worldwide.blackberry.com/legal/thirdpartysoftware.jsp.

BlackBerry Limited
2200 University Avenue East
Waterloo, Ontario
Canada N2K 0A7

BlackBerry UK Limited
Ground Floor, The Pearce Building, West Street,
Maidenhead, Berkshire SL6 1RL
United Kingdom

Published in Canada