



Bluetooth® Mesh ADK 5.0.2.0

October 9, 2023

Bluetooth mesh is a new topology available for Bluetooth Low Energy (LE) devices that enables many-to-many (m:m) communication. It's optimized for creating large-scale device networks, and is ideally suited for building automation, sensor networks, and asset tracking. Our software and SDK for Bluetooth development supports Bluetooth Mesh and Bluetooth 5 functionality. Developers can add mesh networking communication to LE devices such as connected lights, home automation, and asset tracking systems. The software also supports Bluetooth beaconing, beacon scanning, and GATT connections so Bluetooth mesh can connect to smart phones, tablets, and other Bluetooth LE devices.

These release notes cover ADK version(s):

- 5.0.2.0 released on October 9, 2023 (underlying Bluetooth changes only)
- 5.0.1.0 released on July 26, 2023
- 5.0.0.0 released on June 7, 2023



KEY FEATURES

- Application Key is separated from Group. Groups are now ordinary addresses.
- Export and Import were temporarily removed from the API.

Compatibility and Use Notices

- This release is to be used with Bluetooth Mesh SDK 5.0.2.0.
- The iOS ADK supports the last three major releases of the iOS system (iOS 14, iOS 15 and iOS 16).
- The Android ADK supports the last three major releases of the Android system (Android 11, Android 12 and Android 13).

Contents

- 1 Android.....2
 - 1.1 New Items.....2
 - 1.2 Improvements2
 - 1.3 Fixed Issues.....2
 - 1.4 Known Issues in the Current Release2
 - 1.5 Deprecated Items.....2
 - 1.6 Removed Items.....2
 - 1.7 API changes between releases 4.2.1 and 5.0.0.....2
- 2 iOS.....4
 - 2.1 New Items.....4
 - 2.2 Improvements4
 - 2.3 Fixed Issues.....4
 - 2.4 Known Issues in the Current Release4
 - 2.5 Deprecated Items.....4
 - 2.6 Removed Items.....4
 - 2.7 API changes between releases 4.2.1 and 5.0.0.....4
- 3 Using This Release.....6
 - 3.1 Installation and Use.....6
 - 3.2 Support.....6

1 Android

1.1 New Items

Added in release 5.0.0.0

- Added support for virtual addresses.

1.2 Improvements

Changed in release 5.0.0.0

- Failure during sending or receiving mesh messages doesn't throw exceptions now. Mesh errors are now represented with classes and interfaces placed in `errors` package, e.g., `StackError` and `FoundationError`. `ErrorType` class has been removed.

1.3 Fixed Issues

Fixed in release 5.0.0.0

| ID # | Description |
|--------|---|
| 449189 | Application key is strictly connected with Group. |

1.4 Known Issues in the Current Release

None

1.5 Deprecated Items

None

1.6 Removed Items

Removed in release 5.0.0.0

- Temporarily removed import-related classes, with the intention of restoring them in future releases.

1.7 API changes between releases 4.2.1 and 5.0.0

- Core data classes, e.g., `Subnet` and `Node`, are rewritten to Kotlin ones.
- `AppKey` and `Group` separation related changes:
 - `Model` – `boundGroups` property was replaced with `boundAppKeys`. `isSIGModel()` function was removed. `Model` doesn't store `modelSettings` anymore. The user can use `Publication` and `Subscription` classes for managing publications and subscriptions.
 - `Element` – `name` and `location` properties were removed. `Element` doesn't store sensors anymore. The user can fetch them using methods in `ControlElement` and `ControlGroup` classes.
 - `AppKey` – `name` property was removed.
 - `Node` – `groups` property was replaced with `appKeys`. `Node` doesn't store `nodeSecurity` and `nodeSettings` anymore. The user can fetch information about secure network beacon, for example, using `ConfigurationControl` class. `overrideDeviceCompositionData()` method was added to parse elements and models.
 - `Provisioner` and `Scene` classes were removed.
 - `Group` is now a simple wrapper of multicast address.

- Subnet – name, network and subnetSecurity properties were removed. Groups was replaced with appKeys. canCreateGroup() method was removed as well. Creation of group was transferred to Network class.
- Added methods for creating and removing application keys. Subnet removal doesn't factory reset nodes now. It just removes subnet from the local storage.
- Network – name, uuid, provisioners, scenes, version and timestamp properties were removed. Groups are now stored in Network. Added methods for creating and removing groups and subnets. removeOnlyFromLocalStructure() and createScene() methods were removed.
- BluetoothMesh – connectableDeviceHelper and networks properties were removed. There is only one Network now. clearDatabase was replaced with deinitialize() method. activeProxyConnections property and isNewConnectionAllowed method were added.
- BluetoothMesh.initializeNetwork() method is removed. As a replacement user can provide ivIndex and provisionerAddress directly to BluetoothMeshConfiguration object.
- Unicast and multicast addresses are now represented with IntegerAddress and VirtualAddress, which extend Address sealed class.
- Sequence number handling extracted from BluetoothMesh to SequenceNumber class.
- ConnectableDeviceHelper replaced with ConnectableDevice extension functions.
- Model identifier is now 32-bit unsigned integer, big enough to correctly store vendor model identifier.
- Subscriptions are now handled via Subscription and LocalSubscription classes instead of SubscriptionControl class.
- Publications are now handled via Publication class instead of methods in SubscriptionControl class.
- Secure network beacon handling is now done via Device class instead of ProxyConnection.observeSecureNetworkBeacon.
- Generic messages are now handled via GenericClient class instead of methods in ControlElement and ControlGroup classes.
- LightControl messages are now handled via LightControlClient class instead of methods in ControlElement and ControlGroup classes.
- Time messages are now handled via TimeClient class instead of methods in ControlElement and ControlGroup classes.
- Scheduler messages are now handled via GenericClient class instead of methods in ControlElement and ControlGroup classes.
- Scene messages are now handled via SceneClient class instead of methods in ControlElement and ControlGroup classes.
- Parameters in BluetoothMeshConfigurationLimits are corrected to match changes after AppKey from Group separation.

2 iOS

2.1 New Items

Added in release 5.0.0.0

- Added support for virtual addresses.

2.2 Improvements

None

2.3 Fixed Issues

Fixed in release 5.0.0.0

| ID # | Description |
|--------|---|
| 445899 | Application key is strictly connected with Group. |

2.4 Known Issues in the Current Release

None

2.5 Deprecated Items

None

2.6 Removed Items

- Temporarily removed import-related classes, with the intention of restoring them in future releases.

2.7 API changes between releases 4.2.1 and 5.0.0

- SBMApplicationKey and SBMGroup separation related changes:
 - SBMModel – boundGroups property was replaced with boundAppKeys. isSIGModel() function was removed. SBMModel is now a base class for SBMSigModel and SBMVendorModel. Since SBMModel this change causes a lot of changes in API.
 - SBMElement – name and location properties were removed. SBMElement doesn't store sensors anymore. The user can fetch them using methods in SBMControlElement and SBMControlGroup classes.
 - SBMApplicationKey - name property was removed. Functions for creating and removing are moved to SBMSubnet.
 - SBMNode – groups property was replaced with boundAppKeys. SBMNode doesn't store nodeSecurity and nodeSettings anymore. The user can fetch information about secure network beacon, for example, using SBMConfigurationControl class. overrideDeviceCompositionData() method was added to parse elements and models.
 - SBMProvisioner and SBMScene classes were removed.
 - SBMGroup is now a simple wrapper of multicast address.
 - SBMSubnet – name and subnetSecurity properties were removed. groups was replaced with appKeys. createGroup() method was removed as well. Creation of group was transferred to SBMNetwork class. Functions for creating and removing subnet were moved to SBMNetwork.
 - Subnet removal doesn't factory reset nodes now. It just removes subnet from the local storage.

- SBMNetwork – name, uuid, provisioners, scenes, version and timestamp properties were removed. Groups are now stored in SBMNetwork. Added methods for creating and removing groups and subnets. `removeOnlyFromLocalStructure()` and `createScene()` methods were removed.
- SBMBluetoothMesh – networks property was replaced by `network`. There is only one SBMNetwork now. `clearDatabase` can throw error now. `isInitialized` property was added.
- SBMKeyRefresh – groups was replaced by `appKeys`.
- SBMBluetoothMesh.`initializeNetwork()` was changed. As a replacement user can provide `ivIndex` and `provisionerAddress` directly to SBMBluetoothMeshConfiguration object. SBMBluetoothMeshConfiguration is built using SBMBluetoothMeshConfigurationBuilder now.
- Unicast and multicast addresses are now represented with SBMIntegerAddress and SBMVirtualAddress, which extend SBMAddress class. Fields related to addresses in the API were replaced with SBMAddress classes.
- All fields related to uuid have NSUUID type now.
- SBMSubscriptionSettings and SBMNotificationSettings were removed. Use SBMPublicationSettings and SBMSubscriptionControl instead.
- Removed SBMProvisionerConfiguration class. Now SBMProvisionerConnection doesn't configure a node. The user must create a new SBMProxyConnection with a provisioned node and then configure it.
- SBMBluetoothMeshConfigurationLimits has renamed properties.
- SBMProxyConnection has removed field `acceptProvisionerAndKnownGroups`. A reject list on proxy is used by default.
- Changed nullability in callbacks:
 - SBMGetDeviceCompositionDataSuccess – `dcd` can be null,
 - SBMModelUnbindingErrorCallback – error can't be null,
 - SBMControlElementSetSuccess – response can be null,
 - SBMScenesElementPublicationSuccessHandler, SBMTimeControlElementSuccessHandler, SBMSchedulerElementPublicationSuccessHandler – all fields can be null,
 - SBMControlGroupSensorSetHandlerSuccess, SBMLightControlGroupSuccessHandler, SBMControlGroupSceneGetSuccess, SBMControlGroupModelSetSuccess, SBMSchedulerControlGroupSuccessHandler – result and element can be null.

3 Using This Release

3.1 Installation and Use

See *AN1200.1: iOS and Android ADK for Bluetooth® Mesh SDK 2.x and Higher* for information about required tools and compatible platforms.

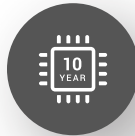
3.2 Support

Development Kit customers are eligible for training and technical support. Use the [Silicon Labs Bluetooth LE web page](#) to obtain information about all Silicon Labs Bluetooth products and services, and to sign up for product support. Contact Silicon Laboratories support at <http://www.silabs.com/support>.

Smart. Connected. Energy-Friendly.



IoT Portfolio
www.silabs.com/products



Quality
www.silabs.com/quality



Support & Community
www.silabs.com/community

Disclaimer

Silicon Labs intends to provide customers with the latest, accurate, and in-depth documentation of all peripherals and modules available for system and software implementers using or intending to use the Silicon Labs products. Characterization data, available modules and peripherals, memory sizes and memory addresses refer to each specific device, and "Typical" parameters provided can and do vary in different applications. Application examples described herein are for illustrative purposes only. Silicon Labs reserves the right to make changes without further notice to the product information, specifications, and descriptions herein, and does not give warranties as to the accuracy or completeness of the included information. Without prior notification, Silicon Labs may update product firmware during the manufacturing process for security or reliability reasons. Such changes will not alter the specifications or the performance of the product. Silicon Labs shall have no liability for the consequences of use of the information supplied in this document. This document does not imply or expressly grant any license to design or fabricate any integrated circuits. The products are not designed or authorized to be used within any FDA Class III devices, applications for which FDA premarket approval is required or Life Support Systems without the specific written consent of Silicon Labs. A "Life Support System" is any product or system intended to support or sustain life and/or health, which, if it fails, can be reasonably expected to result in significant personal injury or death. Silicon Labs products are not designed or authorized for military applications. Silicon Labs products shall under no circumstances be used in weapons of mass destruction including (but not limited to) nuclear, biological or chemical weapons, or missiles capable of delivering such weapons. Silicon Labs disclaims all express and implied warranties and shall not be responsible or liable for any injuries or damages related to use of a Silicon Labs product in such unauthorized applications.

Note: This content may contain offensive terminology that is now obsolete. Silicon Labs is replacing these terms with inclusive language wherever possible. For more information, visit www.silabs.com/about-us/inclusive-lexicon-project

Trademark Information

Silicon Laboratories Inc.[®], Silicon Laboratories[®], Silicon Labs[®], SiLabs[®] and the Silicon Labs logo[®], Bluegiga[®], Bluegiga Logo[®], EFM[®], EFM32[®], EFR, Ember[®], Energy Micro, Energy Micro logo and combinations thereof, "the world's most energy friendly microcontrollers", Redpine Signals[®], WiSeConnect, n-Link, ThreadArch[®], EZLink[®], EZRadio[®], EZRadioPRO[®], Gecko[®], Gecko OS, Gecko OS Studio, Precision32[®], Simplicity Studio[®], Telegesis, the Telegesis Logo[®], USBXpress[®], Zentri, the Zentri logo and Zentri DMS, Z-Wave[®], and others are trademarks or registered trademarks of Silicon Labs. ARM, CORTEX, Cortex-M3 and THUMB are trademarks or registered trademarks of ARM Holdings. Keil is a registered trademark of ARM Limited. Wi-Fi is a registered trademark of the Wi-Fi Alliance. All other products or brand names mentioned herein are trademarks of their respective holders.



Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

www.silabs.com