Windows.Devices.Bluetooth Namespace

The **Windows.Devices.Bluetooth** namespace defines a set of Windows Runtime API that allows UWP app and desktop apps to interact with Bluetooth devices.

Classes

CLASSES

BluetoothAdapter	Represents a Bluetooth adapter.
BluetoothClassOfDevice	Provides functionality to determine the Bluetooth Class Of Device (Bluetooth COD) information for a device.
BluetoothDevice	Represents a Bluetooth device. Sample applications that use this class include Device enumeration and pairing sample and Bluetooth RFCOMM chat sample.
BluetoothDeviceId	Represents a bluetooth device ID.
BluetoothLEAppearance	Provides functionality to determine the Bluetooth Low Energy (LE) Appearance information for a device. To get an instance of this class, call the BluetoothLEAppearance.FromRawValue method or the BluetoothLEAppearance.FromParts method.
BluetoothLEAppearanceCategories	Indicates the appearance category code of the Bluetooth LE device.
BluetoothLEAppearanceSubcategories	Indicates the appearance subcategory code of the Bluetooth LE device.
BluetoothLEDevice	Represents a Bluetooth LE device.
BluetoothSignalStrengthFilter	Groups parameters used to configure received signal strength indicator (RSSI)-based filtering.
BluetoothUuidHelper	A helper class that provides methods to convert between bluetooth device UUID and short ID.

Enums

ENUMS

BluetoothAddressType	Describes the Bluetooth address type.

BluetoothCacheMode	Indicates whether applicable Bluetooth API methods should operate on values cached in the system, or whether they should retrieve those values from the Bluetooth device. See **Remarks** in the member topic.
BluetoothConnectionStatus	Indicates the connection status of the device.
BluetoothError	Specifies common Bluetooth error cases.
BluetoothMajorClass	Indicates the Major Class code of the device, which is the general family of device with which the device is associated.
BluetoothMinorClass	Indicates the Minor Class code of the device. These are interpreted in the context of the Major Class codes.
BluetoothServiceCapabilities	Indicates the service capabilities of a device.

BluetoothAdapter Class

Definition

Namespace:

Windows.Devices.Adapter

Represents a Bluetooth adapter.

Properties

PROPERTIES

AreClassicSecureConnectionsSupported	Gets or sets a value indicating whether Secure Connections are supported for classic paired Bluetooth devices.
AreLowEnergySecureConnectionsSupported	Gets or sets a value indicating whether Secure Connections are supported for paired Bluetooth LE devices.
BluetoothAddress	Gets the device address.
DeviceId	Gets the device ID.
IsAdvertisementOffloadSupported	Gets a boolean indicating if the adapter supports advertisement offload.

IsCentralRoleSupported	Gets a boolean indicating if the adapater supports LowEnergy central role.
IsClassicSupported	Gets a boolean indicating if the adapater supports the Bluetooth Classic transport type.
IsExtendedAdvertisingSupported	Indicates whether the adapter supports the 5.0 Extended Advertising format.
IsLowEnergySupported	Gets a boolean indicating if the adapater supports LowEnergy Bluetooth Transport type.
IsPeripheralRoleSupported	Gets a boolean indicating if the adapater supports LowEnergy peripheral role.
MaxAdvertisementDataLength	Indicates the maximum length of an advertisement that can be published by this adapter.

Methods

METHODS

FromIdAsync(String)	Gets a BluetoothAdapter object identified by the given DeviceId.
GetDefaultAsync()	Gets the default BluetoothAdapter.
GetDeviceSelector()	Gets an Advanced Query Syntax (AQS) string for identifying instances of this adapter. This string is passed to the FindAllAsync or CreateWatcher method.
GetRadioAsync()	Gets the radio represented by this Bluetooth adapter.

BluetoothDevice Class

Definition

Namespace:

Windows.Devices.Bluetooth

Represents a Bluetooth device.

Sample applications that use this class include Device enumeration and pairing sample and Bluetooth RFCOMM chat sample.

Properties

PROPERTIES

BluetoothAddress	Gets the device address.
BluetoothDeviceId	Gets the bluetooth device ID.
ClassOfDevice	Gets the Bluetooth Class Of Device information of the device.
ConnectionStatus	Gets the connection status of the device.
DeviceAccessInformation	The current **DeviceAccessInformation** object for the device. Used to check and monitor access changes to the device.
DeviceId	Gets the device ID that came from the Windows.Devices.Enumeration.DeviceInformation.Id
DeviceInformation	Gets the DeviceInformation object for the Bluetooth device.
HostName	Gets the HostName of the device.
Name	Gets the Name of the device.
RfcommServices	Gets the read-only list of RFCOMM services supported by the device. **Important** The **RfcommServices** API is deprecated, and it may not be available in future versions of Windows. Instead, use **GetRfcommServicesAsync**.
SdpRecords	Gets the read-only list of Service Discovery Protocol (SDP) records for the device.
WasSecureConnectionUsedForPairing	Gets a boolean indicating whether the BluetoothDevice was paired using a Secure Connection.

Methods

METHODS

Close()	Closes the Bluetooth device.
Dispose()	Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.
FromBluetoothAddressAsync(UInt64)	Returns a BluetoothDevice object for the given BluetoothAddress.
FromHostNameAsync(HostName)	Returns a BluetoothDevice object identified by the given HostName.

FromIdAsync(String)	Returns a BluetoothDevice object identified by the given DeviceId.
GetDeviceSelector()	Gets an Advanced Query Syntax (AQS) string for identifying instances of this device. This string is passed to the FindAllAsync or CreateWatcher method.
GetDeviceSelectorFromBluetoothAddress(UInt64)	Creates an Advanced Query Syntax (AQS) filter string from a 64-bit address that represents a Bluetooth device. The AQS string is passed into the CreateWatcher method to return a collection of DeviceInformation objects.
GetDeviceSelectorFromClassOfDevice(BluetoothClassOfDevice)	Creates an Advanced Query Syntax (AQS) filter string from a BluetoothClassOfDevice object. The AQS string is passed into the CreateWatcher method to return a collection of DeviceInformation objects.
GetDeviceSelectorFromConnectionStatus(BluetoothConnectionStatus)	Creates an Advanced Query Syntax (AQS) filter string that contains a query for Bluetooth devices with the indicated BluetoothConnectionStatus. The AQS string is passed into the CreateWatcher method to return a collection of DeviceInformation objects with the indicated Bluetooth connection status.
GetDeviceSelectorFromDeviceName(String)	Creates an Advanced Query Syntax (AQS) filter string that contains a query for the Bluetooth device name. The AQS string is passed into the CreateWatcher method to return a collection of DeviceInformation objects containing the specified Bluetooth device name.
GetDeviceSelectorFromPairingState(Boolean)	Creates an Advanced Query Syntax (AQS) filter string that contains a query for Bluetooth devices that are either paired or unpaired. The AQS string is passed into the CreateWatcher method

	to return a collection of DeviceInformation objects.
GetRfcommServicesAsync()	Retrieves all Rfcomm Services on the remote Bluetooth Device.
GetRfcommServicesAsync(BluetoothCacheMode)	Retrieves all cached Rfcomm Services on the remote Bluetooth Device.
GetRfcommServicesForIdAsync(RfcommServiceId)	Retrieves all Rfcomm Services on the Remote Bluetooth Device matching the specified RfcommServiceId.
GetRfcommServicesForIdAsync(RfcommServiceId, BluetoothCacheMode)	Retrieves all cached Rfcomm Services on the Remote Bluetooth Device matching the specified RfcommServiceId.
RequestAccessAsync()	Used to request explicit access to the Bluetooth device. This invokes Device Consent and must be called on the UI thread.

Events

EVENTS

ConnectionStatusChanged	Occurs when the connection status of the device has changed.
NameChanged	Occurs when the name of the device has changed.
SdpRecordsChanged	Occurs when the list SDP records for the device has changed.

BluetoothDeviceId Class

Definition

Namespace:

Windows.Devices.Bluetooth

Represents a bluetooth device ID.

Properties

PROPERTIES

Id	Gets the bluetooth device ID.
IsClassicDevice	Gets a boolean indicating if this is a classic device.
IsLowEnergyDevice	Gets a boolean indicating if this is a LowEnergy device.

Methods

METHODS

FromId(String)	Creates a BluetoothDeviceId object from the device ID.

BluetoothError Enum

Definition

Namespace:

Windows.Devices.Bluetooth

Specifies common Bluetooth error cases.

Fields

FIELDS

ConsentRequired	8	The operation requires consent.
DeviceNotConnected	3	The operation cannot be completed because the remote device is not connected.
DisabledByPolicy	5	The operation is disabled by policy.
DisabledByUser	7	The operation is disabled by the user.
NotSupported	6	The operation is not supported on the current Bluetooth radio hardware.
OtherError	4	An unexpected error has occurred.
RadioNotAvailable	1	The Bluetooth radio was not available. This error occurs when the Bluetooth radio has been turned off.
ResourceInUse	2	The operation cannot be serviced because the necessary resources are currently in use.
Success	0	The operation was successfully completed or serviced.

TransportNotSupported	9	The transport is not supported.

BluetoothLEAppearance Class

Definition

Namespace:

Windows.Devices.Bluetooth

Provides functionality to determine the Bluetooth Low Energy (LE) Appearance information for a device. To get an instance of this class, call the BluetoothLEAppearance.FromRawValue method or the BluetoothLEAppearance.FromParts method.

Properties

PROPERTIES

Category	Gets the appearance category value of the Bluetooth LE device.
RawValue	Gets the appearance raw value of the Bluetooth LE device.
SubCategory	Gets the appearance subcategory value of the Bluetooth LE device.

Methods

METHODS

FromParts(UInt16, UInt16)	Creates a BluetoothLEAppearance object by supplying values for Category (see BluetoothLEAppearanceCategories) and Subcategory (see BluetoothLEAppearanceSubcategories) of the Bluetooth LE device.
FromRawValue(UInt16)	Creates a BluetoothLEAppearance object by supplying for raw integer values representing the Category and Subcategory of the Bluetooth LE device.

BluetoothLEAppearanceCategories Class

Definition

Namespace:

Windows.Devices.Bluetooth

Indicates the appearance category code of the Bluetooth LE device.

Properties

PROPERTIES

BarcodeScanner	Gets the barcode scanner appearance category code.
BloodPressure	Gets the blood pressure appearance category code.
Clock	Gets the clock appearance category code.
Computer	Gets the computer appearance category code.
Cycling	Gets the cycling appearance category code.
Display	Gets the display appearance category code.
EyeGlasses	Gets the eye glasses appearance category code.
GlucoseMeter	Gets the glucose meter appearance category code.
HeartRate	Gets the heart rate appearance category code.
HumanInterfaceDevice	Gets the human interface device appearance category code.
Keyring	Gets the key ring appearance category code.
MediaPlayer	Gets the media player appearance category code.
OutdoorSportActivity	Gets the outdoor sport activity appearance category code.
Phone	Gets the phone appearance category code.
PulseOximeter	Gets the pulse oximeter appearance category code.
RemoteControl	Gets the remote control appearance category code.
RunningWalking	Gets the running or walking appearance category code.

Tag	Gets the tag appearance category code.
Thermometer	Gets the thermometer appearance category code.
Uncategorized	Gets the uncategorized appearance category code.
Watch	Gets the watch appearance category code.
WeightScale	Gets the weight scale appearance category code.

BluetoothLEDevice Class

Definition

Namespace:

Windows.Devices.Bluetooth

Represents a Bluetooth LE device.

Properties

PROPERTIES

Appearance	Gets the BluetoothLEAppearance object for the Bluetooth LE device.
BluetoothAddress	Gets the device address.
BluetoothAddressType	Gets the address type for the Bluetooth LE device.
BluetoothDeviceId	Gets the bluetooth device ID.
ConnectionStatus	Gets the connection status of the device.
DeviceAccessInformation	Gets the DeviceAccessInformation.
DeviceId	Gets the device Id.
DeviceInformation	Gets the device information for the Bluetooth LE device.
GattServices	Gets the read-only list of GATT services supported by the device.

	Important The **GattServices** API is deprecated, and it may not be available in future versions of Windows. Instead, use **GetGattServicesAsync**.
Name	Gets the name of the Bluetooth LE device.
WasSecureConnectionUsedForPairing	Gets a boolean indicating whether the BluetoothLEDevice was paired using a Secure Connection.

Methods

METHODS

Close()	Closes this Bluetooth LE device. This may close the connection to the device if this is the only app with a connection.
Dispose()	Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.
FromBluetoothAddressAsync(UInt64)	Returns a BluetoothLEDevice object representing the peer Bluetooth LE device with the given address.
FromBluetoothAddressAsync(UInt64, BluetoothAddressType)	Returns a BluetoothLEDevice object representing the peer device with the given address and address type.
FromIdAsync(String)	Returns a BluetoothLEDevice object for the given Id.
GetDeviceSelector()	Gets an Advanced Query Syntax (AQS) string for identifying all Bluetooth Low Energy (LE) devices. This string is passed to the FindAllAsync or CreateWatcher method in order to get a list of Bluetooth LE devices.
GetDeviceSelectorFromAppearance(BluetoothLEAppearance)	Creates an Advanced Query Syntax (AQS) filter string from a BluetoothLEAppearance object. The AQS string is passed into the CreateWatcher method to return a collection of DeviceInformation objects with the specified appearance.
GetDeviceSelectorFromBluetoothAddress(UInt64)	Creates an Advanced Query Syntax (AQS) filter string from a 64-bit address that

	represents a Bluetooth LE device. The AQS string is passed into the CreateWatcher method to return a collection of DeviceInformation objects.
GetDeviceSelectorFromBluetoothAddress(UInt64, BluetoothAddressType)	Creates an Advanced Query Syntax (AQS) filter string from a 64-bit address and address type that represents a Bluetooth LE device. The AQS string is passed into the CreateWatcher method.
GetDeviceSelectorFromConnectionStatus(BluetoothConnectionStatus)	Creates an Advanced Query Syntax (AQS) filter string that contains a query for Bluetooth LE devices with the indicated BluetoothConnectionStatus. The AQS string is passed into the CreateWatcher method to return a collection of DeviceInformation objects with the indicated Bluetooth connection status.
GetDeviceSelectorFromDeviceName(String)	Creates an Advanced Query Syntax (AQS) filter string that contains a query for the Bluetooth LE device name. The AQS string is passed into the CreateWatcher method to return a collection of DeviceInformation objects containing the specified Bluetooth LE device name.
GetDeviceSelectorFromPairingState(Boolean)	Creates an Advanced Query Syntax (AQS) filter string that contains a query for Bluetooth LE devices that are either paired or unpaired. The AQS string is passed into the CreateWatcher method to return a collection of DeviceInformation objects.
GetGattService(Guid)	Returns the GATT service with the given service Id. **Important** The **GetGattService** API is deprecated, and it may not be available in future versions of Windows. Instead, use **GetGattServicesAsync**.
GetGattServicesAsync()	Gets the GattDeviceServices for this Bluetooth LowEnergy device.
GetGattServicesAsync(BluetoothCacheMode)	Returns the GattDeviceServices for this Bluetooth LowEnergy device with the specified cache mode.

GetGattServicesForUuidAsync(Guid)	Returns the GattDeviceServices for the Bluetooth LowEnergy device with the specified UUID.
GetGattServicesForUuidAsync(Guid, BluetoothCacheMode)	Returns the GattDeviceServices for the Bluetooth LowEnergy device with the specified UUID.
RequestAccessAsync()	Requests access to the Bluetooth LowEnergy device.

Events

EVENTS

ConnectionStatusChanged	Occurs when the connection status for the device has changed.
GattServicesChanged	Raised when the list of GATT services supported by the device has changed.
NameChanged	Occurs when the name of the device has changed.

BluetoothSignalStrengthFilter Class

Definition

Namespace:

Windows.Devices.Bluetooth

Groups parameters used to configure received signal strength indicator (RSSI)-based filtering.

Remarks

The BluetoothSignalStrengthFilter class only accepts a limited range for its properties. However, depending on how this class is used by an app, additional restrictions may apply. For example, the valid range for RSSI values differs between Bluetooth LE and Bluetooth BR/EDR devices.

The valid range for these properties are as follows:

- InRangeThresholdInDBm - The minimum threshold for an RSSI event to be considered in range. The valid range is -128 to 127.
- OutOfRangeThresholdInDBm - The minimum threshold for an RSSI event to be considered out of range. The valid range is -128 to 127.
- OutOfRangeTimeout - Timeout for an RSSI event to be considered out of range. The valid range is equal or greater than 1 second.

- SamplingInterval - The interval at which RSSI events are sampled. The valid range is equal or greater than 0. Any sampling interval greater or equal to 25.5 seconds will disable sampling entirely. In that special case, the filtering is trigger-based. There are two possible states for filtering RSSI values for any device:
- In range.
- Out of range. Any RSSI events are propagated if they are considered in range. This includes events with RSSI values lower or equal than OutOfRangeThresholdInDBm as long as OutOfRangeTimeout has not expired.

This class has additional limitations on its properties when used by the BluetoothLEAdvertisementWatcher class. An exception will be thrown when the BluetoothLEAdvertisementWatcher is started with parameters outside of the valid range for BluetoothLEAdvertisementWatcher.

Constructors

CONSTRUCTORS	
BluetoothSignalStrengthFilter()	Create a new BluetoothSignalStrengthFilter object.

Properties

PROPERTIES	
InRangeThresholdInDBm	The minimum received signal strength indicator (RSSI) value in dBm on which RSSI events will be propagated or considered in range if the previous events were considered out of range.
OutOfRangeThresholdInDBm	The minimum received signal strength indicator (RSSI) value in dBm on which RSSI events will be considered out of range.
OutOfRangeTimeout	The timeout for a received signal strength indicator (RSSI) event to be considered out of range.
SamplingInterval	The interval at which received signal strength indicator (RSSI) events are sampled.

Windows.Devices.Bluetooth.Advertisement Namespace

Allow apps to send and receive Bluetooth Low Energy (LE) advertisements.

Classes

CLASSES

BluetoothLEAdvertisement	A representation of a Bluetooth LE advertisement payload.
BluetoothLEAdvertisementBytePattern	A Bluetooth LE advertisement byte pattern for filters to match.
BluetoothLEAdvertisementDataSection	A Bluetooth LE advertisement section. A Bluetooth LE advertisement packet can contain multiple instances of these BluetoothLEAdvertisementDataSection objects.
BluetoothLEAdvertisementDataTypes	Some of the Bluetooth LE advertisement types defined in the Generic Access Profile (GAP) by the Bluetooth Special Interest Group (SIG).
BluetoothLEAdvertisementFilter	Groups parameters used to configure payload-based filtering of received Bluetooth LE advertisements.
BluetoothLEAdvertisementPublisher	An object to send Bluetooth Low Energy (LE) advertisements.
BluetoothLEAdvertisementPublisherStatusChangedEventArgs	Provides data for a StatusChanged event on a BluetoothLEAdvertisementPublisher.
BluetoothLEAdvertisementReceivedEventArgs	Provides data for a Received event on a BluetoothLEAdvertisementWatcher. A BluetoothLEAdvertisementReceivedEventArgs instance is created when the Received event occurs on a BluetoothLEAdvertisementWatcher object.
BluetoothLEAdvertisementWatcher	An object to receive Bluetooth Low Energy (LE) advertisements.
BluetoothLEAdvertisementWatcherStoppedEventArgs	Provides data for a Stopped event on a BluetoothLEAdvertisementWatcher. A BluetoothLEAdvertisementWatcherStoppedEventArgs instance is created when the Stopped event occurs on a BluetoothLEAdvertisementWatcher object.
BluetoothLEManufacturerData	A Bluetooth LE manufacturer-specific data section (one particular type of LE advertisement section). A Bluetooth LE advertisement packet can contain multiple instances of these BluetoothLEManufacturerData objects.

Enums

ENUMS

BluetoothLEAdvertisementFlags	Defines constants that specify flags used to match flags contained inside a Bluetooth LE advertisement payload.
BluetoothLEAdvertisementPublisherStatus	Represents the possible states of the BluetoothLEAdvertisementPublisher.
BluetoothLEAdvertisementType	Specifies the different types of Bluetooth LE advertisement payloads.
BluetoothLEAdvertisementWatcherStatus	Represents the possible states of the BluetoothLEAdvertisementWatcher.
BluetoothLEScanningMode	Defines constants that specify a Bluetooth LE scanning mode.

Remarks

The Windows.Devices.Bluetooth. Advertisement namespace provides an app with a simple but powerful set of methods that allow the following:

- Receive advertisement data from Bluetooth LE peripherals with configurable filtering capabilities.
- Send out Bluetooth LE advertisements allowing the app to operate as a source of beacon advertisements.

This namespace has two sets of classes used for the following:

- Advertisement watcher for receiving.
- Advertisement publisher for sending.

BluetoothLEAdvertisement Class

Definition

Namespace:

Windows.Devices.Bluetooth.Advertisement

A representation of a Bluetooth LE advertisement payload.

Remarks

This class is used to represent an advertisement received by the system, an advertisement pattern to filter for, or an advertisement payload that needs to be published. In the case it is generated by the system to represent an advertisement received, certain properties are automatically populated.

For more information about its usage, refer to BluetoothLEAdvertisementWatcher class for receiving advertisements and the BluetoothLEAdvertisementPublisher class for sending advertisements.

Constructors

	CONSTRUCTORS
BluetoothLEAdvertisement()	Creates a new BluetoothLEAdvertisement object.

Properties

	PROPERTIES
DataSections	Gets the list of raw data sections.
Flags	Bluetooth LE advertisement flags.
LocalName	The local name contained within the advertisement.
ManufacturerData	Gets the list of manufacturer-specific data sections in a BluetoothLEAdvertisement.
ServiceUuids	Gets the list of service UUIDs (in 128-bit GUID format) in a BluetoothLEAdvertisement.

Methods

	METHODS
GetManufacturerDataByCompanyId(UInt16)	Return a list of all manufacturer data sections in the BluetoothLEAdvertisement payload matching the specified company ID.
GetSectionsByType(Byte)	Return a list of advertisement data sections that matches a given advertisement section type in a BluetoothLEAdvertisement.

BluetoothLEAdvertisementPublisher Class

Definition

Namespace:

Windows.Devices.Bluetooth.Advertisement

An object to send Bluetooth Low Energy (LE) advertisements.

Remarks

The BluetoothLEAdvertisementPublisher class allows the configuration and advertising of a Bluetooth LE advertisement packet. The advertising request is serviced on a best effort basis and the advertising parameters are defined by an advertising policy.

The payload of the advertisement is configured when the BluetoothLEAdvertisementPublisher is constructed. The payload can be retrieved by the Advertisement property of the BluetoothLEAdvertisementPublisher. The BluetoothLEAdvertisement class must be managed by the caller if used this way. For example, it is not recommended to directly reuse a BluetoothLEAdvertisement obtained from a BluetoothLEAdvertisementReceivedEventArgs because it contains duplication of actual payload in order to facilitate the access to common data sections. If a received BluetoothLEAdvertisement needs to be duplicated, only the DataSections property should be copied, as they represent the advertisement data sections in their raw form.

The local advertisement resource is limited at the protocol layer. Usage of this resource is best suited for controlled scenarios such as enterprise use cases. Other apps may compete for this limited resource and regular broadcast of advertisements for any specific use case cannot be guaranteed.

The following advertisement types are allowed:

- Manufacturer Specific Information (0xFF)
- Any non-standard type not reserved by the system, as shown in the list below. For more information on non-standard types, see the Assigned numbers and GAP Bluetooth specification.

The following advertisement types are system-reserved and are not allowed:

- Flags (0x01)
- Incomplete List of 16-bit Service UUIDs (0x02)
- Complete List of 16-bit Service UUIDs (0x03)
- Incomplete List of 32-bit Service UUIDs (0x04)
- Complete List of 32-bit Service UUIDs (0x05)
- Incomplete List of 128-bit Service UUIDs (0x06)
- Complete List of 128-bit Service UUIDs (0x07)
- Shortened Local Name (0x08)
- Complete Local Name (0x09)
- Tx Power Level (0x0A)
- Class of Device (0x0D)
- Simple Pairing Hash C192 (0x0E)
- Simple Pairing Randomizer R192 (0x0F)
- Security Manager TK Values (0x10)
- Security Manager Out-of-Band Flags (0x11)
- Slave Connection Interval Range (0x12)
- List of 16-bit Service Solicitation UUIDs (0x14)
- List of 32-bit Service Solicitation UUIDs (0x1F)
- List of 128-bit Service Solicitation UUIDs (0x15)
- Service Data 16-bit UUID (0x16)
- Service Data 32-bit UUID (0x20)
- Service Data 128-bit UUID (0x21)
- Public Target Address (0x17)
- Random Target Address (0x18)
- Appearance (0x19)
- Advertising Interval (0x1A)
- LE Bluetooth Device Address (0x1B)
- LE Role (0x1C)
- Simple Pairing Hash C256 (0x1D)
- Simple Pairing Randomizer R256 (0x1E)
- 3D Information Data (0x3D)

Constructors

CONSTRUCTORS

BluetoothLEAdvertisementPublisher()	Creates a new BluetoothLEAdvertisementPublisher object.
BluetoothLEAdvertisementPublisher(BluetoothLEAdvertisement)	Creates a new BluetoothLEAdvertisementPublisher object with the Bluetooth LE advertisement to publish.

Properties

PROPERTIES

Advertisement	Gets a copy of the Bluetooth LE advertisement to publish.
IncludeTransmitPowerLevel	Specifies whether the transmit power level is included in the advertisement header. Defaults to False.
IsAnonymous	Specifies whether or not the device address is included in the advertisement header. By default, the address is included.
PreferredTransmitPowerLevelInDBm	If specified, requests that the radio use the indicated transmit power level for the advertisement. Defaults to Null.
Status	Gets the current status of the BluetoothLEAdvertisementPublisher.
UseExtendedAdvertisement	Specifies that the advertisement publisher should use the Extended Advertising format.

Methods

METHODS

Start()	Start advertising a Bluetooth LE advertisement payload.
Stop()	Stop the publisher and stop advertising a Bluetooth LE advertisement payload.

Events

EVENTS

StatusChanged	Notification that the status of the BluetoothLEAdvertisementPublisher has changed.

BluetoothLEAdvertisementPublisherStatus Enum

Definition

Namespace:

Windows.Devices.Bluetooth.Advertisement

Represents the possible states of the BluetoothLEAdvertisementPublisher.

Fields

FIELDS

Aborted	5	The publisher is aborted due to an error.
Created	0	The initial status of the publisher.
Started	2	The publisher is being serviced and has started advertising.
Stopped	4	The publisher has stopped advertising.
Stopping	3	The publisher was issued a stop command.
Waiting	1	The publisher is waiting to get service time.

BluetoothLEAdvertisementReceivedEventArgs Class

Definition

Namespace:

Windows.Devices.Bluetooth.Advertisement

Provides data for a Received event on a BluetoothLEAdvertisementWatcher. A BluetoothLEAdvertisementReceivedEventArgs instance is created when the Received event occurs on a BluetoothLEAdvertisementWatcher object.

Properties

PROPERTIES

Advertisement	Gets the Bluetooth LE advertisement payload data received.
AdvertisementType	Gets the type of the received Bluetooth LE advertisement packet.
BluetoothAddress	Gets the Bluetooth address of the device sending the Bluetooth LE advertisement.

BluetoothAddressType	Retrieves the Bluetooth Address Type of the received advertisement.
IsAnonymous	Indicates whether a Bluetooth Address was omitted from the received advertisement.
IsConnectable	Indicates whether the received advertisement is connectable.
IsDirected	Indicates whether the received advertisement is directed.
IsScannable	Indicates whether the received advertisement is scannable.
IsScanResponse	Indicates whether the received advertisement is a scan response.
RawSignalStrengthInDBm	Gets the received signal strength indicator (RSSI) value, in dBm, for this received Bluetooth LE advertisement event. This value could be the raw RSSI or a filtered RSSI depending on filtering settings configured through BluetoothSignalStrengthFilter.
Timestamp	Gets the timestamp when the Received event occurred.
TransmitPowerLevelInDBm	Represents the received transmit power of the advertisement.

BluetoothLEAdvertisementType Enum

Definition

Namespace:

Windows.Devices.Bluetooth.Advertisement

Specifies the different types of Bluetooth LE advertisement payloads.

Fields

FIELDS

ConnectableDirected	1	The advertisement is directed and indicates that the device is connectable but not scannable. This advertisement type cannot carry data. This corresponds with the ADV_DIRECT_IND type defined in the Bluetooth LE specifications.

ConnectableUndirected	0	The advertisement is undirected and indicates that the device is connectable and scannable. This advertisement type can carry data. This corresponds with the ADV_IND type defined in the Bluetooth LE specifications.
Extended	5	This advertisement is a 5.0 extended advertisement. This advertisement type may have different properties, and is not necessarily directed, connected, scannable, nor a scan response. See the advertisement event properties to determine the advertisement details.
NonConnectableUndirected	3	The advertisement is undirected and indicates that the device is not connectable nor scannable. This advertisement type can carry data. This corresponds with the ADV_NONCONN_IND type defined in the Bluetooth LE specifications.
ScannableUndirected	2	The advertisement is undirected and indicates that the device is scannable but not connectable. This advertisement type can carry data. This corresponds with the ADV_SCAN_IND type defined in the Bluetooth LE specifications.
ScanResponse	4	This advertisement is a scan response to a scan request issued for a scannable advertisement. This advertisement type can carry data. This corresponds with the SCAN_RSP type defined in the Bluetooth LE specifications.

BluetoothLEAdvertisementWatcher Class

Definition

Namespace:

Windows.Devices.Bluetooth.Advertisement

An object to receive Bluetooth Low Energy (LE) advertisements.

Constructors

CONSTRUCTORS

BluetoothLEAdvertisementWatcher()	Creates a new BluetoothLEAdvertisementWatcher object.
BluetoothLEAdvertisementWatcher(BluetoothLEAdvertisementFilter)	Creates a new BluetoothLEAdvertisementWatcher object with an advertisement filter to initialize the watcher.

Properties

PROPERTIES

AdvertisementFilter	Gets or sets a BluetoothLEAdvertisementFilter object used for configuration of Bluetooth LE advertisement filtering that uses payload section-based filtering.
AllowExtendedAdvertisements	Enables reception of advertisements using the Extended Advertising format. Defaults to False.
MaxOutOfRangeTimeout	Gets the maximum out of range timeout.
MaxSamplingInterval	Gets the maximum sampling interval.
MinOutOfRangeTimeout	Gets the minimum out of range timeout.
MinSamplingInterval	Gets the minimum sampling interval.
ScanningMode	Gets or sets the Bluetooth LE scanning mode.
SignalStrengthFilter	Gets or sets a BluetoothSignalStrengthFilter object used for configuration of Bluetooth LE advertisement filtering that uses signal strength-based filtering.
Status	Gets the current status of the BluetoothLEAdvertisementWatcher.

Methods

METHODS

Start()	Start the BluetoothLEAdvertisementWatcher to scan for Bluetooth LE advertisements.
Stop()	Stop the BluetoothLEAdvertisementWatcher and disable the scanning for Bluetooth LE advertisements.

Events

EVENTS

Received	Notification for new Bluetooth LE advertisement events received.
Stopped	Notification to the app that the Bluetooth LE scanning for advertisements has been cancelled or aborted either by the app or due to an error.

BluetoothLEAdvertisementWatcherStatus Enum

Definition

Namespace:

Windows.Devices.Bluetooth.Advertisement

Represents the possible states of the BluetoothLEAdvertisementWatcher.

Fields

FIELDS

Aborted	4	An error occurred during transition or scanning that stopped the watcher due to an error.
Created	0	The initial status of the watcher.
Started	1	The watcher is started.
Stopped	3	The watcher is stopped.
Stopping	2	The watcher stop command was issued.

BluetoothLEManufacturerData Class

Definition

Namespace:

Windows.Devices.Bluetooth.Advertisement

A Bluetooth LE manufacturer-specific data section (one particular type of LE advertisement section). A Bluetooth LE advertisement packet can contain multiple instances of these BluetoothLEManufacturerData objects.

Constructors

CONSTRUCTORS

BluetoothLEManufacturerData()	Creates a new BluetoothLEManufacturerData object.
BluetoothLEManufacturerData(UInt16, IBuffer)	Creates a new BluetoothLEManufacturerData object with a company identifier code and manufacterer-specific section data.

Properties

PROPERTIES

CompanyId	The Bluetooth LE company identifier code as defined by the Bluetooth Special Interest Group (SIG).
Data	Bluetooth LE manufacturer-specific section data.

BluetoothLEScanningMode Enum

Definition

Namespace:

Windows.Devices.Bluetooth.Advertisement

Defines constants that specify a Bluetooth LE scanning mode.

Fields

FIELDS

Active	1	Specifies an active scanning mode. This indicates that scan request packets will be sent from the platform to actively query for more advertisement data of type BluetoothLEAdvertisementType.ScanResponse.
None	2	Specifies no scanning mode. The watcher will provide advertisements as the system receives them, but will not initiate its own scanning.

Passive	0	Specifies a passive scanning mode. This is the default scanning mode.

Windows.Devices.Bluetooth.GenericAttributeProfile Namespace

The **Windows.Devices.Bluetooth.GenericAttributeProfile** namespace defines Windows Runtime classes that a UWP app or a desktop application can use to communicate with Bluetooth LE devices.

Classes

CLASSES

GattCharacteristic	Represents a Characteristic of a GATT service. The GattCharacteristic object represents a GATT Characteristic of a particular service, and is obtained from the Characteristics property of the GattDeviceService object.
GattCharacteristicsResult	Contains the result of GetCharacteristicsForUuidAsync and GetCharacteristicsAsync
GattCharacteristicUuids	Represents an enumeration of the most well known Characteristic UUID values, and provides convenience methods for working with GATT characteristic UUIDs, and static properties providing characteristic UUIDs for common GATT characteristics.
GattClientNotificationResult	The result of NotifyValueAsync
GattDescriptor	Represents a Descriptor of a GATT Characteristic. The GattDescriptor object represents a GATT Descriptor of a particular characteristic, and is obtained from the Descriptors property of the GattCharacteristic object.
GattDescriptorsResult	The result of descriptor operations like GattCharacteristic.GetDescriptorsAsync
GattDescriptorUuids	Represents an enumeration of the most well known Descriptor UUID values, and provides convenience methods for working with GATT descriptor UUIDs, and

	static properties providing descriptor UUIDs for common GATT descriptors.
GattDeviceService	Represents a GATT Primary Service on a Bluetooth device. The GattDeviceService class represents a GATT service on a Bluetooth LE device. It is instantiated by using a device service instance path, obtained by finding a device using the Windows.Devices.Enumeration API.
GattDeviceServicesResult	The status of GetIncludedServicesForUuidAsync and GetIncludedServicesForUuidAsync
GattLocalCharacteristic	This class represents a local characteristic.
GattLocalCharacteristicParameters	This class contains the local characteristic descriptor parameters.
GattLocalCharacteristicResult	A result of CreateCharacteristicAsync.
GattLocalDescriptor	This class defines a descriptor of a local characteristic.
GattLocalDescriptorParameters	This class defines the parameters of a descriptor.
GattLocalDescriptorResult	The result of local characteristic descriptor operations like CreateDescriptorAsync.
GattLocalService	This class represents a GATT local service.
GattPresentationFormat	Represents the value of a single Presentation Format GATT Descriptor.
GattPresentationFormatTypes	Represents the different well-known values that the GattPresentationFormat.FormatType property can take.
GattProtocolError	This class contains byte values for GATT protocol errors.
GattReadClientCharacteristicConfigurationDescriptorResult	Represents the result of reading a GATT Client CharacteristicConfigurationClientDescriptor value.
GattReadRequest	This class represents a Bluetooth GATT read request.
GattReadRequestedEventArgs	This class contains the arguments for the StateChanged event.
GattReadResult	Represents the result of an asynchronous read operation of a GATT Characteristic or Descriptor value.
GattReliableWriteTransaction	Performs GATT reliable writes on the Bluetooth LE device, in the form of a transaction write operation.
GattRequestStateChangedEventArgs	This class defines arguments of StateChanged.

GattServiceProvider	This class is used to advertise a GATT service.
GattServiceProviderAdvertisementStatusChangedEventArgs	The event args for the GattServiceProvider.
GattServiceProviderAdvertisingParameters	This class is used to define a GATT service advertisement parameters.
GattServiceProviderResult	This class is the result of the CreateAsync operation.
GattServiceUuids	Represents an enumeration of the most well known Service UUID values, and provides convenience methods for working with GATT service UUIDs, and static properties providing service UUIDs for common GATT services. To view a list of all Bluetooth SIG-defined service UUIDs, see Bluetooth SIG-defined Service UUIDs. **Note** For UUIDs for which **GattServiceUuids** doesn't have a named property, you can use **BluetoothUuidHelper.FromShortId**.
GattSession	Represents a Generic Attribute Profile (GATT) session.
GattSessionStatusChangedEventArgs	This class represents the SessionStatusChanged event args.
GattSubscribedClient	This class represents a subscribed client of a GATT session.
GattValueChangedEventArgs	Represents the arguments received by a GattCharacteristic.ValueChanged event handler used to process characteristic value change notification and indication events sent by a Bluetooth LE device.
GattWriteRequest	This class represents a GATT write request.
GattWriteRequestedEventArgs	This class represents the event args for WriteRequested.
GattWriteResult	Contains the result of GATT write operations like WriteValueWithResultAsync.

Enums

	ENUMS
GattCharacteristicProperties	Specifies the values for the GATT characteristic properties as well as the

	GATT Extended Characteristic Properties Descriptor. Provides a collection of flags representing the GATT Characteristic Properties and if the GATT Extended Properties Descriptor is present the GATT Extended Characteristic properties of the characteristic. Represents the GATT characteristic properties, as defined by the GATT profile, and if the ExtendedProperties flag is present it also represents the properties of the Extended Characteristic Properties Descriptor.
GattClientCharacteristicConfigurationDescriptorValue	Represents the value of the GATT ClientCharacteristicConfigurationDescriptor. Indicates the state of the Client Characteristic Configuration descriptor.
GattCommunicationStatus	Represents the return status of a WinRT GATT API related Async operation. Indicates the status of the asynchronous operation.
GattOpenStatus	The status of GattDeviceService.OpenAsync.
GattProtectionLevel	Represents the desired security level. Allows the desired Security option to be requested.
GattRequestState	This enumeration defines a GATT request state.
GattServiceProviderAdvertisementStatus	This enumeration defines the advertisement status of a GattServiceProvider.
GattSessionStatus	This enumeration defines a GattSession status.
GattSharingMode	This enumeration represents the GATT sharing mode.
GattWriteOption	Indicates what type of write operation is to be performed.

GattCharacteristic Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

Represents a Characteristic of a GATT service. The GattCharacteristic object represents a GATT Characteristic of a particular service, and is obtained from the Characteristics property of the GattDeviceService object.

Properties

PROPERTIES

AttributeHandle	Gets the handle used to uniquely identify GATT-based characteristic attributes as declared on the Bluetooth LE device.
CharacteristicProperties	Gets the GATT characteristic properties, as defined by the GATT profile. If the *ExtendedProperties* flag is present it also represents the properties of the Extended Characteristic Properties Descriptor.
PresentationFormats	Gets the list of presentation format descriptors associated with this GattCharacteristic, in the order specified by the Aggregate Format Descriptor. The list shall be empty if no PresentationFormat or Aggregate Format descriptors are found.
ProtectionLevel	Gets or sets the desired GATT security options for over the air communication with the device. Windows will negotiate the maximum security possible with the device as part of the pairing process, and specifying a lower level of security won't degrade the existing security level.

Service	Gets the GattDeviceService of which this characteristic is a member.
UserDescription	Get the user friendly description for this GattCharacteristic, if the User Description Descriptor is present, otherwise this will be an empty string.
Uuid	Gets the GATT Characteristic UUID for this GattCharacteristic.

Methods

METHODS

ConvertShortIdToUuid(UInt16)	Converts a Bluetooth SIG defined short Id to a full GATT UUID. **Important** The **ConvertShortIdToUuid** API is deprecated, and it may not be available in future versions of Windows. Instead, use **BluetoothUuidHelper.FromShortId**.
GetAllDescriptors()	Gets the collection of all descriptors belonging to this GattCharacteristic instance. **Important** The **GetAllDescriptors** API is deprecated, and it may not be available in future versions of Windows. Instead, use **GetDescriptorsAsync**.
GetDescriptors(Guid)	Returns a vector of descriptors, that are identified by the specified UUID, and belong to this GattCharacteristic instance. **Important** The **GetDescriptors** API is deprecated, and it may not be available in future

	versions of Windows. Instead, use **GetDescriptorsForUuidAsync**.
GetDescriptorsAsync()	Returns the descriptors for this GattCharacteristic instance.
GetDescriptorsAsync(BluetoothCacheMode)	Returns the descriptors with the specified cache mode for this GattCharacteristic instance.
GetDescriptorsForUuidAsync(Guid)	Returns the descriptors whose UUIDs match descriptorUuid.
GetDescriptorsForUuidAsync(Guid, BluetoothCacheMode)	Returns the descriptors whose UUIDs match descriptorUuid with the specified cacheMode.
ReadClientCharacteristicConfigurationDescriptorAsync()	Reads the current value of the ClientCharacteristicConfigurationDescriptor.
ReadValueAsync()	Performs a Characteristic Value read from the value cache maintained by Windows.
ReadValueAsync(BluetoothCacheMode)	Performs a Characteristic Value read either from the value cache maintained by Windows, or directly from the device.
WriteClientCharacteristicConfigurationDescriptorAsync(GattClientCharacteristicConfigurationDescriptorValue)	Writes the ClientCharacteristicConfigurationDescriptor to the Bluetooth LE device, and if the value to be written represents an indication or a notification and a ValueChanged event handler is registered, enables receiving ValueChanged events from the device.

WriteClientCharacteristicConfigurationDescriptorWithResultAsync(GattClientCharacteristicConfigurationDescriptorValue)	Writes the ClientCharacteristicConfigurationDescriptor to the Bluetooth LE device, and if the value to be written represents an indication or a notification and a ValueChanged event handler is registered, enables receiving ValueChanged events from the device.
WriteValueAsync(IBuffer)	Performs a Characteristic Value write to a Bluetooth LE device.
WriteValueAsync(IBuffer, GattWriteOption)	Performs a Characteristic Value write to a Bluetooth LE device.
WriteValueWithResultAsync(IBuffer)	Performs a Characteristic Value write to a Bluetooth LE device.
WriteValueWithResultAsync(IBuffer, GattWriteOption)	Performs a Characteristic Value write to a Bluetooth LE device.

Events

EVENTS

ValueChanged	An App can register an event handler in order to receive events when notification or indications are received from a device, after setting the Client Characteristic Configuration Descriptor.

GattCharacteristicProperties Enum

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

Specifies the values for the GATT characteristic properties as well as the GATT Extended Characteristic Properties Descriptor.

Provides a collection of flags representing the GATT Characteristic Properties and if the GATT Extended Properties Descriptor is present the GATT Extended Characteristic properties of the characteristic.

Represents the GATT characteristic properties, as defined by the GATT profile, and if the ExtendedProperties flag is present it also represents the properties of the Extended Characteristic Properties Descriptor.

This enumeration has a FlagsAttribute attribute that allows a bitwise combination of its member values.

Fields

FIELDS

AuthenticatedSignedWrites	64	The characteristic supports signed writes
Broadcast	1	The characteristic supports broadcasting
ExtendedProperties	128	The ExtendedProperties Descriptor is present
Indicate	32	The characteristic is indicatable
None	0	The characteristic doesn’t have any properties that apply.
Notify	16	The characteristic is notifiable
Read	2	The characteristic is readable
ReliableWrites	256	The characteristic supports reliable writes
WritableAuxiliaries	512	The characteristic has writable auxiliaries
Write	8	The characteristic is writable
WriteWithoutResponse	4	The characteristic supports Write Without Response

GattCharacteristicsResult Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

Contains the result of GetCharacteristicsForUuidAsync and GetCharacteristicsAsync

Properties

PROPERTIES

Characteristics	Gets the characterisitics.
ProtocolError	Gets the protocol error, if there is one.
Status	Gets the status.

GattCharacteristicUuids Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

Represents an enumeration of the most well known Characteristic UUID values, and provides convenience methods for working with GATT characteristic UUIDs, and static properties providing characteristic UUIDs for common GATT characteristics

Properties

PROPERTIES

AlertCategoryId	Gets the Bluetooth SIG-defined AlertCategoryId characteristic UUID.
AlertCategoryIdBitMask	Gets the Bluetooth SIG-Defined AlertCategoryIdBitMask characteristic UUID. To view a list of all Bluetooth SIG-defined characteristic UUIDs, see Bluetooth SIG-defined Characteristic UUIDs.

AlertLevel	Gets the Bluetooth SIG-defined AlertLevel characteristic UUID.
AlertNotificationControlPoint	Gets the Bluetooth SIG-defined AlertNotificationControlPoint characteristic UUID.
AlertStatus	Gets the Bluetooth SIG-defined AlertStatus characteristic UUID.
BatteryLevel	Gets the Bluetooth SIG-defined Battery Level Characteristic UUID.
BloodPressureFeature	Gets the Bluetooth SIG-defined Blood Pressure Feature Characteristic UUID.
BloodPressureMeasurement	Gets the Bluetooth SIG-defined Blood Pressure Measurement Characteristic UUID.
BodySensorLocation	Gets the Bluetooth SIG-defined Body Sensor Location Characteristic UUID.
BootKeyboardInputReport	Gets the Bluetooth SIG-defined BootKeyboardInputReport characteristic UUID.
BootKeyboardOutputReport	Gets the Bluetooth SIG-defined BootKeyboardOutputReport characteristic UUID.
BootMouseInputReport	Gets the Bluetooth SIG-defined BootMouseInputReport characteristic UUID.
CscFeature	Gets the Bluetooth SIG-defined Csc Feature Characteristic UUID.
CscMeasurement	Gets the Bluetooth SIG-defined Csc Measurement Characteristic UUID.
CurrentTime	Gets the Bluetooth SIG-defined CurrentTime characteristic UUID.
CyclingPowerControlPoint	Gets the Bluetooth SIG-defined CyclingPowerControlPoint characteristic UUID.
CyclingPowerFeature	Gets the Bluetooth SIG-defined CyclingPowerFeature characteristic UUID.
CyclingPowerMeasurement	Gets the Bluetooth SIG-defined CyclingPowerMeasurement characteristic UUID.
CyclingPowerVector	Gets the Bluetooth SIG-defined CyclingPowerVector characteristic UUID.
DateTime	Gets the Bluetooth SIG-defined DateTime characteristic UUID.
DayDateTime	Gets the Bluetooth SIG-defined DayDateTime characteristic UUID.
DayOfWeek	Gets the Bluetooth SIG-defined DayOfWeek characteristic UUID.

DstOffset	Gets the Bluetooth SIG-defined DstOffset characteristic UUID.
ExactTime256	Gets the Bluetooth SIG-defined ExactTime256 characteristic UUID.
FirmwareRevisionString	Gets the Bluetooth SIG-defined FirmwareRevisionString characteristic UUID.
GapAppearance	Gets the Bluetooth SIG-defined GapAppearance characteristic UUID.
GapDeviceName	Gets the Bluetooth SIG-defined GapDeviceName characteristic UUID.
GapPeripheralPreferredConnectionParameters	Gets the Bluetooth SIG-defined GapPeripheralPreferredConnectionParameters characteristic UUID.
GapPeripheralPrivacyFlag	Gets the Bluetooth GapPeripheralPrivacyFlag characterisitc UUID.
GapReconnectionAddress	Gets the Bluetooth SIG-defined GapReconnectionAddress characteristic UUID.
GattServiceChanged	Gets the Bluetooth SIG-defined GattServiceChanged characteristic UUID.
GlucoseFeature	Gets the Bluetooth SIG-defined Glucose Feature Characteristic UUID.
GlucoseMeasurement	Gets the Bluetooth SIG-defined Glucose Measurement Characteristic UUID.
GlucoseMeasurementContext	Gets the Bluetooth SIG-defined Glucose Measurement Context Characteristic UUID.
HardwareRevisionString	Gets the Bluetooth SIG-defined HardwareRevisionString characterisitc UUID.
HeartRateControlPoint	Gets the Bluetooth SIG-defined Heart Rate Control Point Characteristic UUID.
HeartRateMeasurement	Gets the Bluetooth SIG-defined Heart Rate Measurement Characteristic UUID.
HidControlPoint	Gets the Bluetooth SIG-defined HidControlPoint characteristic UUID.
HidInformation	Gets the Bluetooth SIG-defined HidInformation characteristic UUID.
Ieee1107320601RegulatoryCertificationDataList	Gets the Bluetooth SIG-defined Ieee1107320601RegulatoryCertificationDataList characteristic UUID.
IntermediateCuffPressure	Gets the Bluetooth SIG-defined Intermediate Cuff Pressure Characteristic UUID.

IntermediateTemperature	Gets the Bluetooth SIG-defined Intermediate Temperature Characteristic UUID.
LnControlPoint	Gets the Bluetooth SIG-defined LnControlPoint characteristic UUID.
LnFeature	Gets the Bluetooth SIG-defined LnFeature characteristic UUID.
LocalTimeInformation	Gets the Bluetooth SIG-defined LocalTimeInformation characteristic UUID.
LocationAndSpeed	Gets the Bluetooth SIG-defined LocationAndSpeed characteristic UUID.
ManufacturerNameString	Gets the Bluetooth SIG-defined ManufacturerNameString characteristic UUID.
MeasurementInterval	Gets the Bluetooth SIG-defined Measurement Interval Characteristic UUID.
ModelNumberString	Gets the Bluetooth SIG-defined ModelNumberString UUID.
Navigation	Gets the Bluetooth SIG-defined Navigation characteristic UUID.
NewAlert	Gets the Bluetooth SIG-defined NewAlert characteristic UUID.
PnpId	Gets the Bluetooth SIG-defined PnpId characteristic UUID.
PositionQuality	Gets the Bluetooth SIG-defined PositionQuality characteristic UUID.
ProtocolMode	Gets the Bluetooth SIG-defined ProtocolMode characteristic UUID.
RecordAccessControlPoint	Gets the Bluetooth SIG-defined Record Access Control Point Characteristic UUID.
ReferenceTimeInformation	Gets the Bluetooth SIG-defined ReferenceTimeInformation characteristic UUID.
Report	Gets the Bluetooth SIG-defined Report characteristic UUID.
ReportMap	Gets the Bluetooth SIG-defined ReportMap characteristic UUID.
RingerControlPoint	Gets the Bluetooth SIG-defined RingerControlPoint characteristic UUID.
RingerSetting	Gets the Bluetooth SIG-defined RingerSetting characteristic UUID.
RscFeature	Gets the Bluetooth SIG-defined Rsc Feature Characteristic UUID.

RscMeasurement	Gets the Bluetooth SIG-defined Rsc Measurement Characteristic UUID.
ScanIntervalWindow	Gets the Bluetooth SIG-defined ScanIntervalWindow characteristic UUID.
ScanRefresh	Gets the Bluetooth SIG-defined ScanRefresh characteristic UUID.
SCControlPoint	Gets the Bluetooth SIG-defined SC Control Point Characteristic UUID.
SensorLocation	Gets the Bluetooth SIG-defined Sensor Location Characteristic UUID.
SerialNumberString	Gets the Bluetooth SIG-defined SerialNumberString characteristic UUID.
SoftwareRevisionString	Gets the Bluetooth SIG-defined SoftwareRevisionString characteristic UUID.
SupportedNewAlertCategory	Gets the Bluetooth SIG-defined SupportedNewAlertCategory characteristic UUID.
SupportUnreadAlertCategory	Gets the Bluetooth SIG-defined SupportUnreadAlertCategory characteristic UUID.
SystemId	Gets the Bluetooth SIG-defined SystemId characteristic UUID.
TemperatureMeasurement	Gets the Bluetooth SIG-defined Temperature Measurement Characteristic UUID.
TemperatureType	Gets the Bluetooth SIG-defined Temperature Type Characteristic UUID.
TimeAccuracy	Gets the Bluetooth SIG-defined TimeAccuracy characteristic UUID.
TimeSource	Gets the Bluetooth SIG-defined TimeSource characteristic UUID.
TimeUpdateControlPoint	Gets the Bluetooth SIG-defined TimeUpdateControlPoint characteristic UUID.
TimeUpdateState	Gets the Bluetooth SIG-defined TimeUpdateState characteristic UUID.
TimeWithDst	Gets the Bluetooth SIG-defined TimeWithDst characteristic UUID.
TimeZone	Gets the Bluetooth SIG-defined TimeZone characteristic UUID.
TxPowerLevel	Gets the Bluetooth SIG-defined TxPowerLevel characteristic UUID.
UnreadAlertStatus	Gets the Bluetooth SIG-defined UnreadAlertStatus characteristic UUID.

GattClientCharacteristicConfigurationDescriptorValue Enum

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

Represents the value of the GATT ClientCharacteristicConfigurationDescriptor.

Indicates the state of the Client Characteristic Configuration descriptor.

Fields

FIELDS

Indicate	2	Characteristic indications are enabled.
None	0	Neither notification nor indications are enabled.
Notify	1	Characteristic notifications are enabled.

GattClientNotificationResult Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

The result of NotifyValueAsync

Properties

PROPERTIES

BytesSent	Gets the bytes that were sent.
ProtocolError	Gets the protocol error.
Status	Gets the GATT communication status.
SubscribedClient	Gets the subscribed client.

GattCommunicationStatus Enum

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

Represents the return status of a WinRT GATT API related Async operation.

Indicates the status of the asynchronous operation.

Fields

FIELDS

AccessDenied	3	Access is denied.
ProtocolError	2	There was a GATT communication protocol error.
Success	0	The operation completed successfully.
Unreachable	1	No communication can be performed with the device, at this time.

GattDescriptor Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

Represents a Descriptor of a GATT Characteristic. The GattDescriptor object represents a GATT Descriptor of a particular characteristic, and is obtained from the Descriptors property of the GattCharacteristic object.

Properties

PROPERTIES

AttributeHandle	Gets the GATT Attribute handle used to uniquely identify this attribute on the GATT Server Device.
ProtectionLevel	Gets or sets the desired GATT security options for over the air communication with the device. Windows will negotiate the maximum security possible with the device as part of the pairing process, and specifying a lower level of security won't degrade the existing security level.
Uuid	Gets the GATT Descriptor UUID for this GattDescriptor.

Methods

METHODS

ConvertShortIdToUuid(UInt16)	Converts a Bluetooth SIG defined short Id to a full GATT UUID. **Important** The **ConvertShortIdToUuid** API is deprecated, and it may not be available in future versions of Windows. Instead, use **BluetoothUuidHelper.FromShortId**.
ReadValueAsync()	Performs a Descriptor Value read from a value cache maintained by Windows.
ReadValueAsync(BluetoothCacheMode)	Performs a Descriptor Value read either from the value cache maintained by Windows, or directly from the device.
WriteValueAsync(IBuffer)	Performs a Descriptor Value write to a Bluetooth LE device.
WriteValueWithResultAsync(IBuffer)	Performs a Descriptor Value write to a Bluetooth LE device.

GattDescriptorsResult Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

The result of descriptor operations like GattCharacteristic.GetDescriptorsAsync

Properties

PROPERTIES

Descriptors	Gets a vector of the GATT descriptors.
ProtocolError	Gets the GATT protocol error.
Status	Gets the status of the operation.

GattDeviceService Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

Represents a GATT Primary Service on a Bluetooth device. The GattDeviceService class represents a GATT service on a Bluetooth LE device. It is instantiated by using a device service instance path, obtained by finding a device using the Windows.Devices.Enumeration API.

Properties

PROPERTIES

AttributeHandle	Gets the handle used to uniquely identify GATT-based service attributes as declared on the Bluetooth LE device.
Device	Gets the BluetoothLEDevice object describing the device associated with the current GattDeviceService object.

	Important The **Device** API is deprecated, and it may not be available in future versions of Windows. Instead, use **Session**.
DeviceAccessInformation	Gets the device access information associated with the current GattDeviceService object.
DeviceId	Get the string that represents the GATT service instance path used to instantiate the GattDeviceService.
ParentServices	Gets the read-only list of parent services for this service. **Important** The **ParentServices** API is deprecated, and it may not be available in future versions of Windows.
Session	Gets the session for this GATT device service instance.
SharingMode	Gets the sharing mode for this GATT device service instance.
Uuid	Gets the GATT Service UUID associated with this **GattDeviceService**.

Methods

METHODS

Close()	Releases the resources associated with the GattDeviceService class. This allows other apps to access the resources of the GattDeviceService in question. A GattDeviceService object should not be used after Close is invoked, instead a new GattDeviceService object should be instantiated using the FromIdAsync method.
ConvertShortIdToUuid(UInt16)	Converts a Bluetooth SIG defined short Id to a full GATT UUID. **Important** The **ConvertShortIdToUuid** API is deprecated, and it may not be available in future versions of Windows. Instead, use **BluetoothUuidHelper.FromShortId**.
Dispose()	Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.

FromIdAsync(String)	Instantiates a new GattDeviceService from the device ID.
FromIdAsync(String, GattSharingMode)	Instantiates a new GattDeviceService object from the device ID.
GetAllCharacteristics()	Gets the collection of all characteristics belonging to this GattDeviceService instance. **Important** The **GetAllCharacteristics** API is deprecated, and it may not be available in future versions of Windows. Instead, use **GetCharacteristicsAsync**.
GetAllIncludedServices()	Gets the collection of all included services belonging to this GattDeviceService instance. **Important** The **GetAllCharacteristics** API is deprecated, and it may not be available in future versions of Windows. Instead, use **GetIncludedServicesAsync**.
GetCharacteristics(Guid)	Returns a vector of characteristics, that are identified by the specified UUID and belong to this GattDeviceService instance. **Important** The **GetCharacteristics** API is deprecated, and it may not be available in future versions of Windows. Instead, use **GetCharacteristicsForUuidAsync**.
GetCharacteristicsAsync()	Gets the characteristics that are part of this GattDeviceService instance.
GetCharacteristicsAsync(BluetoothCacheMode)	Gets the characteristics that are part of this GattDeviceService instance.
GetCharacteristicsForUuidAsync(Guid)	Gets the characterisitics that are part of this GattDeviceService instance and associated with the characteristicUuid.
GetCharacteristicsForUuidAsync(Guid, BluetoothCacheMode)	Gets the characterisitics that are part of this GattDeviceService instance and associated with the characteristicUuid.

GetDeviceSelectorForBluetoothDeviceId(BluetoothDeviceId)	Creates a suitable AQS Filter string for use with the CreateWatcher method, from a BluetoothDeviceId.
GetDeviceSelectorForBluetoothDeviceId(BluetoothDeviceId, BluetoothCacheMode)	Creates a suitable AQS Filter string for use with the CreateWatcher method, from a BluetoothDeviceId.
GetDeviceSelectorForBluetoothDeviceIdAndUuid(BluetoothDeviceId, Guid)	Creates a suitable AQS Filter string for use with the CreateWatcher method, from a BluetoothDeviceId and serviceUuid.
GetDeviceSelectorForBluetoothDeviceIdAndUuid(BluetoothDeviceId, Guid, BluetoothCacheMode)	Creates a suitable AQS Filter string for use with the CreateWatcher method, from a BluetoothDeviceId and serviceUuid.
GetDeviceSelectorFromShortId(UInt16)	Creates a suitable AQS Filter string for use with the CreateWatcher method, from a 16-bit Bluetooth GATT Service UUID. **Important** The **GetDeviceSelectorFromShortId** API is deprecated, and it may not be available in future versions of Windows. Instead, use **GetDeviceSelectorFromUuid**.
GetDeviceSelectorFromUuid(Guid)	Creates a suitable AQS Filter string for use with the CreateWatcher method, from a Bluetooth service UUID.
GetIncludedServices(Guid)	Returns a vector of included services, that are identified by the specified UUID and belong to this GattDeviceService instance. **Important** The **GetIncludedServices** API is deprecated, and it may not be available in future versions of Windows. Instead, use **GetIncludedServicesForUuidAsync**.
GetIncludedServicesAsync()	Gets the included services that are associated with this GattDeviceService instance.
GetIncludedServicesAsync(BluetoothCacheMode)	Gets the included services that are associated with this GattDeviceService instance.
GetIncludedServicesForUuidAsync(Guid)	Gets the included services from the serviceUuid that is associated with this GattDeviceService instance.

GetIncludedServicesForUuidAsync(Guid, BluetoothCacheMode)	Gets the included services from the serviceUuid that is associated with this GattDeviceService instance.
OpenAsync(GattSharingMode)	Opens the GATT device service with the specified sharingMode.
RequestAccessAsync()	Requests access to the GattDeviceService.

GattDeviceServicesResult Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

The status of GetIncludedServicesForUuidAsync and GetIncludedServicesForUuidAsync

Properties

PROPERTIES

ProtocolError	Gets the protocol error.
Services	Gets the services.
Status	Gets the communication status of the operation.

GattLocalCharacteristic Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

This class represents a local characteristic.

Properties

PROPERTIES

CharacteristicProperties	Gets the local characteristic properties.
Descriptors	Gets a vector list of all the descriptors for this local characteristic.
PresentationFormats	Gets the presentation formats for this local characteristic.
ReadProtectionLevel	Gets the read protection level of this local characteristic.
StaticValue	Gets the static value for this local GATT characteristic.
SubscribedClients	Gets a vector list of all clients that are subscribed to this local characteristic.
UserDescription	Gets the user-friendly description for this local characteristic.
Uuid	Gets the BluetoothSIG-defined UUID for this local characteristic.
WriteProtectionLevel	Gets the write protection level of this local characteristic.

Methods

METHODS

CreateDescriptorAsync(Guid, GattLocalDescriptorParameters)	Creates descriptor for this local characteristic.
NotifyValueAsync(IBuffer)	Send and notifies all subscribed clients a GattSubscribedClient of a value.
NotifyValueAsync(IBuffer, GattSubscribedClient)	Sends and notifies a GattSubscribedClient of a value.

Events

EVENTS

ReadRequested	An event that is triggered when a GATT client requests a read operation.
SubscribedClientsChanged	An event that triggers when the clients subscribed to this local characteristic changes.
WriteRequested	This is an event that is triggered when a write was requested.

GattLocalCharacteristicResult Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

A result of CreateCharacteristicAsync.

Properties

PROPERTIES

Characteristic	Gets the characteristic of the GATT service.
Error	Gets the error.

GattOpenStatus Enum

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

The status of GattDeviceService.OpenAsync.

Fields

FIELDS

AccessDenied	5	Access is denied.
AlreadyOpened	2	The GATT device service is already opened.
NotFound	3	The GATT device service was not found.

SharingViolation	4	There was a sharing violation.
Success	1	The GATT device service was successfully opened.
Unspecified	0	Unspecified error.

GattProtectionLevel Enum

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

Represents the desired security level.

Allows the desired Security option to be requested.

Fields

FIELDS

AuthenticationRequired	1	Require the link to be authenticated.
EncryptionAndAuthenticationRequired	3	Require the link to be encrypted and authenticated.
EncryptionRequired	2	Require the link to be encrypted.
Plain	0	Uses the default protection level.

GattReadClientCharacteristicConfigurationDescriptorResult Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

Represents the result of reading a GATT Client CharacteristicConfigurationClientDescriptor value.

Remarks

The ReadClientCharacteristicConfigurationDescriptorAsync method is used to retrieve a GattReadClientCharacteristicConfigurationDescriptorResult object.

The Status property on the GattReadClientCharacteristicConfigurationDescriptorResult returned indicates if the result of the operation was successful.

Properties

PROPERTIES

ClientCharacteristicConfigurationDescriptor	Gets the result of an asynchronous read operation.
ProtocolError	Gets the protocol error.
Status	Gets the status of an asynchronous operation.

GattReadRequest Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

This class represents a Bluetooth GATT read request.

Properties

PROPERTIES

Length	Gets the buffer length of the read request.

Offset	Gets the read request offset.
State	Gets the read request state.

Methods

METHODS

RespondWithProtocolError(Byte)	Responds to the read request with a protocol error.
RespondWithValue(IBuffer)	Responds to a read request with a value.

Events

EVENTS

StateChanged	This is an event that is triggered when the read request state was changed.

GattReadResult Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

Represents the result of an asynchronous read operation of a GATT Characteristic or Descriptor value.

Remarks

One of the ReadValueAsync methods is used to retrieve a GattReadResult object.

The Status property on the GattReadResult returned indicates if the result of the operation was successful.

Properties

PROPERTIES

ProtocolError	Gets the protocol error.
Status	Gets the status of an asynchronous operation.
Value	Gets the value read from the device.

GattReliableWriteTransaction Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

Performs GATT reliable writes on the Bluetooth LE device, in the form of a transaction write operation.

Constructors

CONSTRUCTORS

GattReliableWriteTransaction()	Creates a new GattReliableWriteTransaction object.

Methods

METHODS

CommitAsync()	Performs all the queued writes, in sequence, writing the data to the device. Once a transaction object has been committed, no further operations are possible on the GattReliableWriteTransaction object.
CommitWithResultAsync()	Performs all the queued writes, in sequence, writing the data to the device. Once a transaction object has been committed, no further operations are possible on the GattReliableWriteTransaction object.
WriteValue(GattCharacteristic, IBuffer)	Adds a new write operation to the transaction object.

GattRequestState Enum

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

This enumeration defines a GATT request state.

Fields

FIELDS

Canceled	2	The request is canceled.
Completed	1	The request if completed.
Pending	0	The request is pending.

GattRequestStateChangedEventArgs Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

This class defines arguments of StateChanged.

Properties

PROPERTIES

Error	Gets the Bluetooth error.
State	Gets the state of the read request.

GattServiceProvider Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

This class is used to advertise a GATT service.

Properties

	PROPERTIES
AdvertisementStatus	Gets the advertisement status of this GATT service.
Service	Gets the GATT service.

Methods

	METHODS
CreateAsync(Guid)	Creates a new GATT service with the specified serviceUuid
StartAdvertising()	Start advertising the GATT service.
StartAdvertising(GattServiceProviderAdvertisingParameters)	Start advertising the GATT service.
StopAdvertising()	Stop advertising the GATT service.

Events

	EVENTS
AdvertisementStatusChanged	An event that is triggered when the advertisement status has changed.

GattServiceProviderAdvertisementStatus Enum

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

This enumeration defines the advertisement status of a GattServiceProvider.

Fields

FIELDS

Aborted	3	The GATT service was aborted.
Created	0	The GATT service was created.
Started	2	The GATT service advertising has started.
StartedWithoutAllAdvertisementData	4	Indicates that the system was successfully able to issue the advertisement request, but not all of the requested data could be included in the advertisement.
Stopped	1	The GATT service is not advertising.

GattServiceProviderAdvertisementStatusChangedEventArgs Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

The event args for the GattServiceProvider.

Properties

PROPERTIES

Error	Gets the error of the GATT service.
Status	Gets the status of the GATT service.

GattServiceProviderAdvertisingParameters Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

This class is used to define a GATT service advertisement parameters.

Constructors

CONSTRUCTORS

GattServiceProviderAdvertisingParameters()	Creates a new GattServiceProviderAdvertisingParameters object.

Properties

PROPERTIES

IsConnectable	Gets or sets a boolean that indicates if the GATT service is connectable.
IsDiscoverable	Gets or sets a boolean indicating that the GATT service is discoverable.
ServiceData	For Bluetooth Low Energy, this parameter adds an additional **ServiceData** section to the advertisement payload for the service's service UUID if space is available. If the service data is added to the advertisement, then the service UUID will also be included in the same section in the advertisement.

GattServiceProviderResult Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

This class is the result of the CreateAsync operation.

Properties

PROPERTIES

Error	Gets the error.
ServiceProvider	Gets the service provider.

GattServiceUuids Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

Represents an enumeration of the most well known Service UUID values, and provides convenience methods for working with GATT service UUIDs, and static properties providing service UUIDs for common GATT services. To view a list of all Bluetooth SIG-defined service UUIDs, see Bluetooth SIG-defined Service UUIDs.

Properties

PROPERTIES

AlertNotification	Gets the Bluetooth SIG-defined AlertNotification Service UUID.
Battery	Gets the Bluetooth SIG-defined Battery Service UUID.
BloodPressure	Gets the Bluetooth SIG-defined Blood Pressure Service UUID.
CurrentTime	Gets the Bluetooth SIG-defined CurrentTime service UUID.
CyclingPower	Gets the Bluetooth SIG-defined CyclingPower service UUID.
CyclingSpeedAndCadence	Gets the Bluetooth SIG-defined Cycling Speed And Cadence Service UUID.
DeviceInformation	Gets the Bluetooth SIG-defined DeviceInformation service UUID.
GenericAccess	Gets the Bluetooth SIG-defined UUID for the Generic Access Service.
GenericAttribute	Gets the Bluetooth SIG-defined UUID for the Generic Attribute Service.

Glucose	Gets the Bluetooth SIG-defined Glucose Service UUID.
HealthThermometer	Gets the Bluetooth SIG-defined Health Thermometer Service UUID.
HeartRate	Gets the Bluetooth SIG-defined Heart Rate Service UUID.
HumanInterfaceDevice	Gets the Bluetooth SIG-defined HumanInterfaceDevice service UUID.
ImmediateAlert	Gets the Bluetooth SIG-defined ImmediateAlert service UUID.
LinkLoss	Gets the Bluetooth SIG-defined LinkLoss service UUID.
LocationAndNavigation	Gets the Bluetooth SIG-defined LocationAndNavigation service UUID.
NextDstChange	Gets the Bluetooth SIG-defined NextDstChange service UUID.
PhoneAlertStatus	Gets the Bluetooth SIG-defined PhoneAlertStatus service UUID.
ReferenceTimeUpdate	Gets the Bluetooth SIG-defined ReferenceTimeUpdate service UUID.
RunningSpeedAndCadence	Gets the Bluetooth SIG-defined Running Speed And Cadence Service UUID.
ScanParameters	Gets the Bluetooth SIG-defined ScanParameters service UUID.
TxPower	Gets the Bluetooth SIG-defined TxPower service UUID.

GattValueChangedEventArgs Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

Represents the arguments received by a GattCharacteristic.ValueChanged event handler used to process characteristic value change notification and indication events sent by a Bluetooth LE device.

Properties

PROPERTIES

CharacteristicValue	Gets the new Characteristic Value.
Timestamp	Gets the time at which the system was notified of the Characteristic Value change.

GattWriteOption Enum

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

Indicates what type of write operation is to be performed.

Fields

FIELDS

WriteWithoutResponse	1	The Write Without Response procedure shall be used.
WriteWithResponse	0	The default GATT write procedure shall be used.

GattWriteRequest Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

This class represents a GATT write request.

Properties

PROPERTIES

Offset	Gets the offset.

Option	Gets the write request option.
State	Gets the state of the write request.
Value	Gets the buffer value of the write request.

Methods

METHODS

Respond()	Responds to the write request.
RespondWithProtocolError(Byte)	Responds with a protocol error.

Events

EVENTS

StateChanged	An event that is triggered when the state of the write request has changed.

GattWriteRequestedEventArgs Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

This class represents the event args for WriteRequested.

Properties

PROPERTIES

Session	Gets the session.

Methods

METHODS

GetDeferral()	Gets a deferral.
GetRequestAsync()	Gets the write request.

GattWriteResult Class

Definition

Namespace:

Windows.Devices.Bluetooth.GenericAttributeProfile

Contains the result of GATT write operations like WriteValueWithResultAsync.

Properties

PROPERTIES

ProtocolError	Gets the protocol error.
Status	Gets the status of the write result.

Windows.Devices.Enumeration Namespace

Provides classes for enumerating devices. The following are typical uses of the Windows.Devices.Enumeration API.

- Building a user interface for selecting the device to be used by an application. For example, a voice chat application may present a list of microphones or webcams for the user to select from, or a photo import application may present a list of removable storage devices for the user to import photos from.
- Get general information about devices connected to or discoverable by the system.
- Device discovery and notifications about devices for apps that use them.

Classes

CLASSES

DeviceAccessChangedEventArgs	Provides data for the AccessChanged event.
DeviceAccessInformation	Contains the information about access to a device.
DeviceConnectionChangeTriggerDetails	Provides information about the device that caused this trigger to fire.

DeviceDisconnectButtonClickedEventArgs	Provides data for the DisconnectButtonClicked event on the DevicePicker object.
DeviceInformation	Represents a device. This class allows access to well-known device properties as well as additional properties specified during device enumeration.
DeviceInformationCollection	Represents a collection of DeviceInformation objects.
DeviceInformationCustomPairing	Represents a custom pairing for a DeviceInformation object.
DeviceInformationPairing	Contains information and enables pairing for a device.
DeviceInformationUpdate	Contains updated properties for a DeviceInformation object.
DevicePairingRequestedEventArgs	Provides data for the PairingRequested event.
DevicePairingResult	Contains information about the result of attempting to pair a device.
DevicePicker	Represents a picker flyout that contains a list of devices for the user to choose from.
DevicePickerAppearance	Represents the appearance of a device picker.
DevicePickerFilter	Represents the filter used to determine which devices to show in the device picker. The filter parameters are OR-ed together to build the resulting filter.
DeviceSelectedEventArgs	Provides data for the DeviceSelected event on the DevicePicker object.
DeviceThumbnail	Represents the thumbnail image for a device.
DeviceUnpairingResult	Contains information about the result of attempting to unpair a device.
DeviceWatcher	Enumerates devices dynamically, so that the app receives notifications if devices are added, removed, or changed after the initial enumeration is complete.
DeviceWatcherEvent	Triggered when the list of devices is updated after the initial enumeration is complete.
DeviceWatcherTriggerDetails	Provides details about the device updates that invoked the trigger.
EnclosureLocation	Describes the physical location of a device in its enclosure.

Interfaces

INTERFACES

IDevicePairingSettings	Identifies a collection of settings for device pairing. WiFiDirectConnectionParameters implements **IDevicePairingSettings**.

Enums

ENUMS

DeviceAccessStatus	Indicates the status of the access to a device.
DeviceClass	Indicates the type of devices that the user wants to enumerate.
DeviceInformationKind	Represents the kind of DeviceInformation object.
DevicePairingKinds	Indicates the kinds of pairing supported by your application or requested by the system. As an input value, use this value to indicate what kinds of pairing your application supports. When this datatype is received as an output value, it indicates the kind of pairing requested by the system. In this case, your code will need to respond accordingly.
DevicePairingProtectionLevel	The level of protection for pairing.
DevicePairingResultStatus	The result of the pairing action with an Association Endpoint (AEP) device object. For more information about AEP objects, see DeviceInformationKind.
DevicePickerDisplayStatusOptions	Indicates what you'd like the device picker to show about a given device. Used with the SetDisplayStatus method on the DevicePicker object.
DeviceUnpairingResultStatus	The result of the unpairing action.
DeviceWatcherEventKind	The type of event.
DeviceWatcherStatus	Describes the state of a DeviceWatcher object.
Panel	Indicates the location of a panel on a computer. This enumeration is used for indicating the physical location of a device.

DeviceAccessChangedEventArgs Class

Definition

Namespace:

Windows.Devices.Enumeration

Provides data for the AccessChanged event.

Properties

PROPERTIES	
Id	Gets the DeviceInformation ID of the device referenced by DeviceAccessChangedEventArgs.
Status	The new status of access to a device.

DeviceAccessInformation Class

Definition

Namespace:

Windows.Devices.Enumeration

Contains the information about access to a device.

Properties

PROPERTIES	
CurrentStatus	The current device access status.

Methods

METHODS	
CreateFromDeviceClass(DeviceClass)	Initializes a DeviceAccessInformation object based on a given DeviceClass.

CreateFromDeviceClassId(Guid)	Initializes a DeviceAccessInformation object based on a device class id.
CreateFromId(String)	Initializes a DeviceAccessInformation object based on a device id.

Events

EVENTS

AccessChanged	Raised when access to a device has changed.

DeviceAccessStatus Enum

Definition

Namespace:

Windows.Devices.Enumeration

Indicates the status of the access to a device.

Fields

FIELDS

Allowed	1	Access to the device is allowed.
DeniedBySystem	3	Access to the device has been disallowed by the system.
DeniedByUser	2	Access to the device has been disallowed by the user.
Unspecified	0	The device access is not specified.

DeviceClass Enum

Definition

Namespace:

Windows.Devices.Enumeration

Indicates the type of devices that the user wants to enumerate.

Fields

FIELDS

All	0	Indicates that the user wants to enumerate all devices.
AudioCapture	1	Indicates that the user wants to enumerate all audio capture devices.
AudioRender	2	Indicates that the user wants to enumerate all audio rendering devices.
ImageScanner	5	Indicates that the user wants to enumerate all scanning devices.
Location	6	Indicates that the user wants to enumerate all location aware devices.
PortableStorageDevice	3	Indicates that the user wants to enumerate all portable storage devices.
VideoCapture	4	Indicates that the user wants to enumerate all video capture devices.

DeviceConnectionChangeTriggerDetails Class

Definition

Namespace:

Windows.Devices.Enumeration

Provides information about the device that caused this trigger to fire.

Properties

PROPERTIES

DeviceId	Gets the device Id of the device that caused this trigger to fire.

DeviceInformation Class

Definition

Namespace:

Windows.Devices.Enumeration

Represents a device. This class allows access to well-known device properties as well as additional properties specified during device enumeration.

Remarks

A DeviceInformation object is composed of an identity (@"Windows.Devices.Enumeration.DeviceInformation.Id?text=DeviceInformation.Id"), a kind (@"Windows.Devices.Enumeration.DeviceInformation.Kind?text=DeviceInformation.Kind "), and a property bag (@"Windows.Devices.Enumeration.DeviceInformation.Properties?text=DeviceInformation.Properties"). All of the other properties of a DeviceInformation object are derived from the Properties property bag. For example, Name is derived from **System.ItemNameDisplay**.

Successful completion of FindAllAsync results in a DeviceInformationCollection containing DeviceInformation objects.

If a call to CreateWatcher succeeds, a DeviceInformation object is passed to the added event for each device that is found.

The Name property should only be used for display purposes only and not for finding a device because the Name can change due to localization or a user assigning a name.

CreateFromIdAsync creates a DeviceInformation object if successful.

The DeviceInformation class provides device information, but more specifically, it provides properties of the device interface, the interface that represents functionality that the device exposes. Multi-function devices may have more than one device interface. The physical object that a user sees as a device, is known as the device container, and has properties such as **Manufacturer** and **ModelID**. For more information about enumerating devices and recovering properties, see Enumerate devices.

Properties

PROPERTIES

EnclosureLocation	The physical location of the device in its enclosure. For example, it may describe the location of a webcam inside a laptop.
Id	A string representing the identity of the device.
IsDefault	Indicates whether this device is the default device for the class.
IsEnabled	Indicates whether this device is enabled.
Kind	Gets the type of DeviceInformation represented by this object.
Name	The name of the device. This name is in the best available language for the app.
Pairing	Gets the information about the capabilities for this device to pair.
Properties	Property store containing well-known values as well as additional properties that can be specified during device enumeration.

Methods

METHODS

CreateFromIdAsync(String)	Creates a DeviceInformation object from a DeviceInformation ID.
CreateFromIdAsync(String, IEnumerable<String>)	Creates a DeviceInformation object from a DeviceInformation ID and a list of additional properties.
CreateFromIdAsync(String, IEnumerable<String>, DeviceInformationKind)	Creates a DeviceInformation object from a DeviceInformation ID, a list of additional properties, and a DeviceInformationKind parameter.
CreateWatcher()	Creates a DeviceWatcher for all devices.
CreateWatcher(DeviceClass)	Creates a DeviceWatcher for devices matching the specified DeviceClass.
CreateWatcher(String)	Creates a DeviceWatcher for devices matching the specified Advanced Query Syntax (AQS) string.

CreateWatcher(String, IEnumerable<String>)	Creates a DeviceWatcher for devices matching the specified Advanced Query Syntax (AQS) string and the specified collection of properties.
CreateWatcher(String, IEnumerable<String>, DeviceInformationKind)	Creates a DeviceWatcher for devices matching the specified Advanced Query Syntax (AQS) string, the specified collection of properties, and the kind of devices.
FindAllAsync()	Enumerates all DeviceInformation objects.
FindAllAsync(DeviceClass)	Enumerates DeviceInformation objects of the specified class.
FindAllAsync(String)	Enumerates DeviceInformation objects matching the specified Advanced Query Syntax (AQS) device interface selector string.
FindAllAsync(String, IEnumerable<String>)	Enumerates DeviceInformation objects matching the specified Advanced Query Syntax (AQS) device interface selector string and including the specified collection of properties.
FindAllAsync(String, IEnumerable<String>, DeviceInformationKind)	Enumerates DeviceInformation objects matching the specified Advanced Query Syntax (AQS) device interface selector string, the device kind, and including the specified collection of properties.
GetAqsFilterFromDeviceClass(DeviceClass)	Creates a filter to use to enumerate through a subset of device types.
GetGlyphThumbnailAsync()	Gets a glyph for the device.
GetThumbnailAsync()	Returns a thumbnail image for the device.
Update(DeviceInformationUpdate)	Updates the properties of an existing DeviceInformation object.

DeviceInformationCustomPairing Class

Definition

Namespace:

Windows.Devices.Enumeration

Represents a custom pairing for a DeviceInformation object.

Methods

METHODS

PairAsync(DevicePairingKinds)	Attempts to pair the device.
PairAsync(DevicePairingKinds, DevicePairingProtectionLevel)	Attempts to pair a device using a minimum protection level.
PairAsync(DevicePairingKinds, DevicePairingProtectionLevel, IDevicePairingSettings)	Attempts to pair a device using a minimum protection level and custom settings.

Events

EVENTS

PairingRequested	Raised when a pairing action is requested.

DeviceInformationKind Enum

Definition

Namespace:

Windows.Devices.Enumeration

Represents the kind of DeviceInformation object.

Fields

FIELDS

AssociationEndpoint	5	The DeviceInformation object represents a device association endpoint (AEP). AEPs usually represent a device discovered over a wireless or network protocol. Some of these endpoints can be used without needing to first pair the device. Use CanPair to determine if a device supports pairing and IsPaired to determine if a device is already paired with the system. Only *AssociationEndpoint* objects have a non-NULL value for Pairing. For more information about pairing over a network, see Enumerate devices

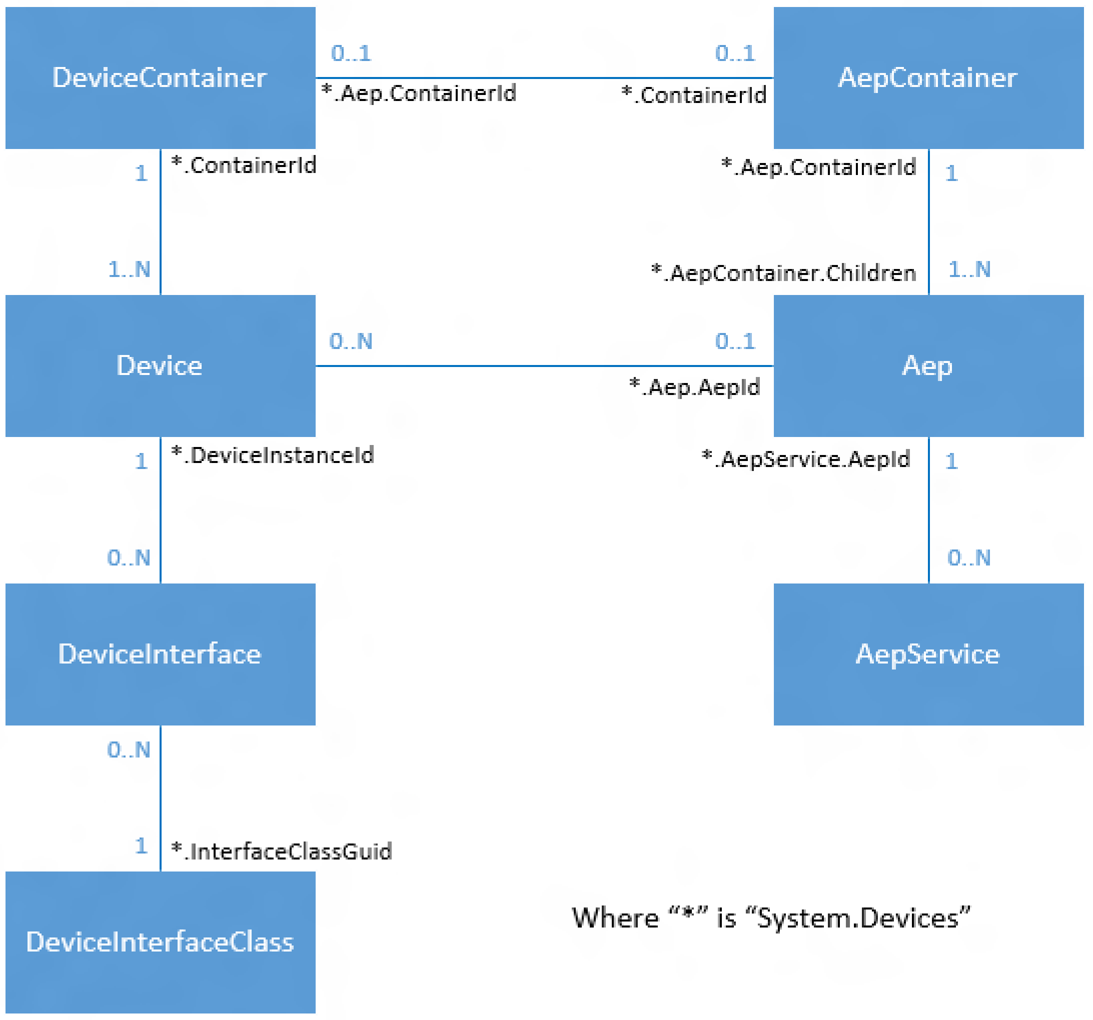
		over a network. An *AssociationEndpoint* object is a child of a single *AssociationEndpointContainer* object and can contain 0 or more *AssociationEndpointService* objects. The Id for this object.
AssociationEndpointContainer	6	The DeviceInformation object represents an association endpoint (AEP) container. An *AssociationEndpointContainer* object represents a single physical device that might have more than one *AssociationEndpoint* objects associated with it. For example, if a television supports two different network protocols, the *AssociationEndpointContainer* would be the television. It would also have two *AssociationEndpoint* objects to represent each protocol. The Id for this object is the AEP container GUID as a string.
AssociationEndpointService	7	The DeviceInformation object represents an association endpoint (AEP) Service. An *AssociationEndpointService* object represents a functional service contract exposed by the device. Not all protocols support AEP services. An *AssociationEndpointService* can have a single parent *AssociationEndpointContainer* object. The Id for this object as a string.
Device	3	The DeviceInformation object represents a device object. This could also be referred to as a devnode. These devices are objects that represent a piece of the device functionality and optionally have drivers loaded on them. When a physical device is paired with windows, multiple *Device* objects are created for it. A device contains 0 or more *DeviceInterface* objects, is a child to one *DeviceContainer* object, and is related to 0 or 1 *AssociationEndpoint* objects. The Id for this object is the device instance id for this object.
DeviceContainer	2	The DeviceInformation object represents a device container. *DeviceContainer* objects describe a collection of device objects that exist in the same physical device. For example, a multi-function printer may have several different devices included, but they would all exist as part of the parent *DeviceContainer* object. The Id for this object is the device container GUID as a string.
DeviceInterface	1	The DeviceInformation object represents a PnP device interface. These are exposed by device drivers to enable an app to talk to the device, typically using device IOCTLS (input output controls). This type of DeviceInformation object contains all the properties of the PnP *DeviceInterface* object plus some properties from the parent PnP *DeviceContainer* object. This combination of properties can be

		presented to the user when the user needs to select a device to complete an app scenario. A *DeviceInterface* object can only be a child to one *DeviceContainer* object. This is the default DeviceInformationKind when enumerating devices without specifying a specific kind. The Id for this object is the device interface id, or the device interface path.
DeviceInterfaceClass	4	The DeviceInformation object represents a device interface class. Every *DeviceInterface* object belongs to a certain *DeviceInterfaceClass*. This is similar to a contract definition. These contracts are represented with an id and properties, just like every other PnP object. *DeviceInterfaceClass* objects contain little more information than a unique identifier and possibly a name. The Id for this object is the device interface class GUID as a string.
DevicePanel	8	The DeviceInformation object represents a single physical face of a device enclosure. The Id for this object as a string.
Unknown	0	The object is of an unknown type. This value is not used.

Remarks

A DeviceInformationKind value can be passed to the various enumeration APIs to indicate the specific kind of device you want to enumerate. In most scenarios, you will receive the DeviceInformation from the device selector. To see how to enumerate over specific kinds of devices using DeviceInformationKind, see Enumerate devices.

The following diagram show how the different device types interact with each other. It also indicates the properties that tie the relationships together. For performance reasons, many of the relationship connections only go in one direction. The exception to this is **AepContainer**. It contains a list of its child objects thanks to **AepContainer.Children**.



DeviceInformationPairing Class

Definition

Namespace:

Windows.Devices.Enumeration

Contains information and enables pairing for a device.

Properties

PROPERTIES

CanPair	Gets a value that indicates whether the device can be paired.

Custom	Gets the DeviceInformationCustomPairing object necessary for custom pairing.
IsPaired	Gets a value that indicates whether the device is currently paired.
ProtectionLevel	Gets the level of protection used to pair the device.

Methods

METHODS

PairAsync()	Attempts to pair the device.
PairAsync(DevicePairingProtectionLevel)	Attempts to pair the device using a provided level of protection.
PairAsync(DevicePairingProtectionLevel, IDevicePairingSettings)	Attempts to pair a device object with a specified protection level and custom settings.
TryRegisterForAllInboundPairingRequests(DevicePairingKinds)	Registers the application to handle all inbound pairing requests.
TryRegisterForAllInboundPairingRequestsWithProtectionLevel(DevicePairingKinds, DevicePairingProtectionLevel)	Registers the application to handle all inbound pairing requests with the specified minimum level of protection.
UnpairAsync()	Attempts to unpair the device.

DeviceInformationUpdate Class

Definition

Namespace:

Windows.Devices.Enumeration

Contains updated properties for a DeviceInformation object.

Remarks

The Updated and Removed events of the DeviceWatcher class receive a **DeviceInformationUpdate** object.

For more information about **DeviceInformationUpdate**, see Enumerate and watch devices.

Properties

PROPERTIES

Id	The DeviceInformation ID of the updated device.
Kind	Gets the type of device for the updated device.
Properties	The changed properties in a update to a DeviceInformation object.

DevicePairingKinds Enum

Definition

Namespace:

Windows.Devices.Enumeration

Indicates the kinds of pairing supported by your application or requested by the system. As an input value, use this value to indicate what kinds of pairing your application supports. When this datatype is received as an output value, it indicates the kind of pairing requested by the system. In this case, your code will need to respond accordingly.

This enumeration has a FlagsAttribute attribute that allows a bitwise combination of its member values.

Fields

FIELDS

ConfirmOnly	1	The application must confirm they wish to perform the pairing action. You can present an optional confirmation dialog to the user. With a value of **ConfirmOnly**, call Accept from the event args of the PairingRequested event handler if you want the pairing to complete.
ConfirmPinMatch	8	The application must display the given PIN to the user and ask the user to confirm that the PIN matches the one show on the target device. With a value of **ConfirmPinMatch**, call Accept from the event args of the PairingRequested event handler if you want the pairing to complete.
DisplayPin	2	The application must display the given PIN to the user. The user will then need to enter or confirm that PIN on the device that is being paired. With a value of **DisplayPin**, call Accept from the event args of the PairingRequested event handler if you want the pairing to complete. If your application cancels the pairing at this point, the device might still be paired. This is because the system and the target device don't need any confirmation for this DevicePairingKinds value.
None	0	No pairing is supported.
ProvidePasswordCredential	16	The application must request a user name and password from the user. With a value of **ProvidePasswordCredential**, call AcceptWithPasswordCredential from the event args of the PairingRequested event handler to accept the pairing. Pass in the PasswordCredential that encapsulates the user name and password as a parameter.
ProvidePin	4	The application must request a PIN from the user. The PIN will typically be displayed on the target device. With a value of **ProvidePin**, call Accept from the event args of the PairingRequested event handler if you want the pairing to complete. Pass in the PIN as a parameter.

Remarks

This value is a collection of flags. It is possible to have multiple values set for a single pairing action.

DevicePairingProtectionLevel Enum

Definition

Namespace:

Windows.Devices.Enumeration

The level of protection for pairing.

Fields

FIELDS

Default	0	The default value. This should not be used.
Encryption	2	Pair the device using encryption.
EncryptionAndAuthentication	3	Pair the device using encryption and authentication.
None	1	Pair the device using no levels of protection.

DevicePairingRequestedEventArgs Class

Definition

Namespace:

Windows.Devices.Enumeration

Provides data for the PairingRequested event.

Properties

PROPERTIES

DeviceInformation	Gets the DeviceInformation object associated with this pairing request.
PairingKind	Gets the kind of pairing associated with this pairing event.
Pin	Gets the pin associated with a pairing request.

Methods

METHODS

Accept()	Accepts a PairingRequested event and pairs the device with the application.
Accept(String)	Accepts a PairingRequested event and pairs the device with the application. Requires a pin for pairing purposes.

AcceptWithPasswordCredential(PasswordCredential)	Accepts a PairingRequested event and pairs the device with the application. Requires a user name and password for pairing purposes.
GetDeferral()	Requests a Deferral to support asynchronous operations.

DevicePairingResult Class

Definition

Namespace:

Windows.Devices.Enumeration

Contains information about the result of attempting to pair a device.

Properties

PROPERTIES

ProtectionLevelUsed	Gets the level of protection used to pair the device.
Status	Gets the paired status of the device after the pairing action completed.

DevicePairingResultStatus Enum

Definition

Namespace:

Windows.Devices.Enumeration

The result of the pairing action with an Association Endpoint (AEP) device object. For more information about AEP objects, see DeviceInformationKind.

Fields

FIELDS

AccessDenied	12	Your application does not have the appropriate permissions level to pair the device object.
AlreadyPaired	3	The device object has already been paired.
AuthenticationFailure	9	Authentication failed, so the device is not paired. Either the device object or the application rejected the authentication.
AuthenticationNotAllowed	8	The authentication protocol is not supported, so the device is not paired.
AuthenticationTimeout	7	The authentication process timed out before it could complete.
ConnectionRejected	4	The device object rejected the connection.
Failed	19	An unknown failure occurred.
HardwareFailure	6	The device object indicated there was a hardware failure.
InvalidCeremonyData	13	The ceremony data was incorrect.
NoSupportedProfiles	10	There are no network profiles for this device object to use.
NotPaired	2	The device object is not currently paired.
NotReadyToPair	1	The device object is not in a state where it can be paired.
OperationAlreadyInProgress	15	The device object is already attempting to pair or unpair.
Paired	0	The device object is now paired.
PairingCanceled	14	The pairing action was canceled before completion.
ProtectionLevelCouldNotBeMet	11	The minimum level of protection is not supported by the device object or the application.
RejectedByHandler	17	The application handler rejected the pairing.
RemoteDeviceHasAssociation	18	The remove device already has an association.
RequiredHandlerNotRegistered	16	Either the event handler wasn't registered or a required DevicePairingKinds was not supported.
TooManyConnections	5	The device object indicated it cannot accept any more incoming connections.

DevicePicker Class

Definition

Namespace:

Windows.Devices.Enumeration

Represents a picker flyout that contains a list of devices for the user to choose from

Constructors

CONSTRUCTORS

DevicePicker()	Creates a DevicePicker object.

Properties

PROPERTIES

Appearance	Gets the colors of the picker.
Filter	Gets the filter used to choose what devices to show in the picker.
RequestedProperties	Gets a collection of properties for the returned device information object.

Methods

METHODS

Hide()	Hides the picker.
PickSingleDeviceAsync(Rect)	Shows the picker UI and returns the selected device; does not require you to register for an event. The picker flies out from the edge of the specified rectangle.
PickSingleDeviceAsync(Rect, Placement)	Shows the picker UI and returns the selected device; does not require you to register for an event. The picker flies out from the specified edge of the specified rectangle.
SetDisplayStatus(DeviceInformation, String, DevicePickerDisplayStatusOptions)	Updates the picker UI to reflect the provided status and display options for a specified device.
Show(Rect)	Shows the picker UI. The picker flies out from the edge of the specified rectangle.
Show(Rect, Placement)	Shows the picker UI. The picker flies out from the specified edge of the specified rectangle.

Events

EVENTS

DevicePickerDismissed	Indicates that the device picker was light dismissed by the user. Light dismiss happens when the user clicks somewhere other than the picker UI and the picker UI disappears.
DeviceSelected	Indicates that the user selected a device in the picker.
DisconnectButtonClicked	Indicates that the user clicked or tapped the disconnect button for a device in the picker.

DevicePickerFilter Class

Definition

Namespace:

Windows.Devices.Enumeration

Represents the filter used to determine which devices to show in the device picker. The filter parameters are OR-ed together to build the resulting filter.

Properties

PROPERTIES

SupportedDeviceClasses	Gets a list of supported device classes to show in the picker. This defaults to an empty list (no filter). You can add device classes to this vector and filter the devices list to those that are in one or more of the provided classes.
SupportedDeviceSelectors	Gets a list of AQS filter strings. This defaults to empty list (no filter). You can add one or more AQS filter strings to this vector and filter the devices list to those that meet one or more of the provided filters.

DeviceSelectedEventArgs Class

Definition

Namespace:

Windows.Devices.Enumeration

Provides data for the DeviceSelected event on the DevicePicker object.

Properties

PROPERTIES

SelectedDevice	The device selected by the user in the picker.

DeviceThumbnail Class

Definition

Namespace:

Windows.Devices.Enumeration

Represents the thumbnail image for a device.

Properties

PROPERTIES

CanRead	Gets a value that indicates whether the stream can be read from.
CanWrite	Gets a value that indicates whether the stream can be written to.
ContentType	Returns the content type of the thumbnail image.
Position	Gets the byte offset of the stream.
Size	Gets or sets the size of the device thumbnail image.

Methods

METHODS

CloneStream()	Creates a new instance of a IRandomAccessStream over the same resource as the current stream.
Close()	Closes the current stream and releases system resources.

Dispose()	Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.
FlushAsync()	Flushes data asynchronously in a sequential stream.
GetInputStreamAt(UInt64)	Returns a pointer to an input stream starting at the specified location.
GetOutputStreamAt(UInt64)	Returns a pointer to an output stream starting at the specified location.
ReadAsync(IBuffer, UInt32, InputStreamOptions)	Returns an asynchronous byte reader object.
Seek(UInt64)	Sets the position of the stream to the specified value.
WriteAsync(IBuffer)	Writes data asynchronously in a sequential stream.

DeviceUnpairingResult Class

Definition

Namespace:

Windows.Devices.Enumeration

Contains information about the result of attempting to unpair a device.

Properties

PROPERTIES

Status	Gets the paired status of the device after the unpairing action completed.

DeviceUnpairingResultStatus Enum

Definition

Namespace:

Windows.Devices.Enumeration

The result of the unpairing action.

Fields

FIELDS

AccessDenied	3	The caller does not have sufficient permissions to unpair the device.
AlreadyUnpaired	1	The device object was already unpaired.
Failed	4	An unknown failure occurred.
OperationAlreadyInProgress	2	The device object is currently in the middle of either a pairing or unpairing action.
Unpaired	0	The device object is successfully unpaired.

DeviceWatcher Class

Definition

Namespace:

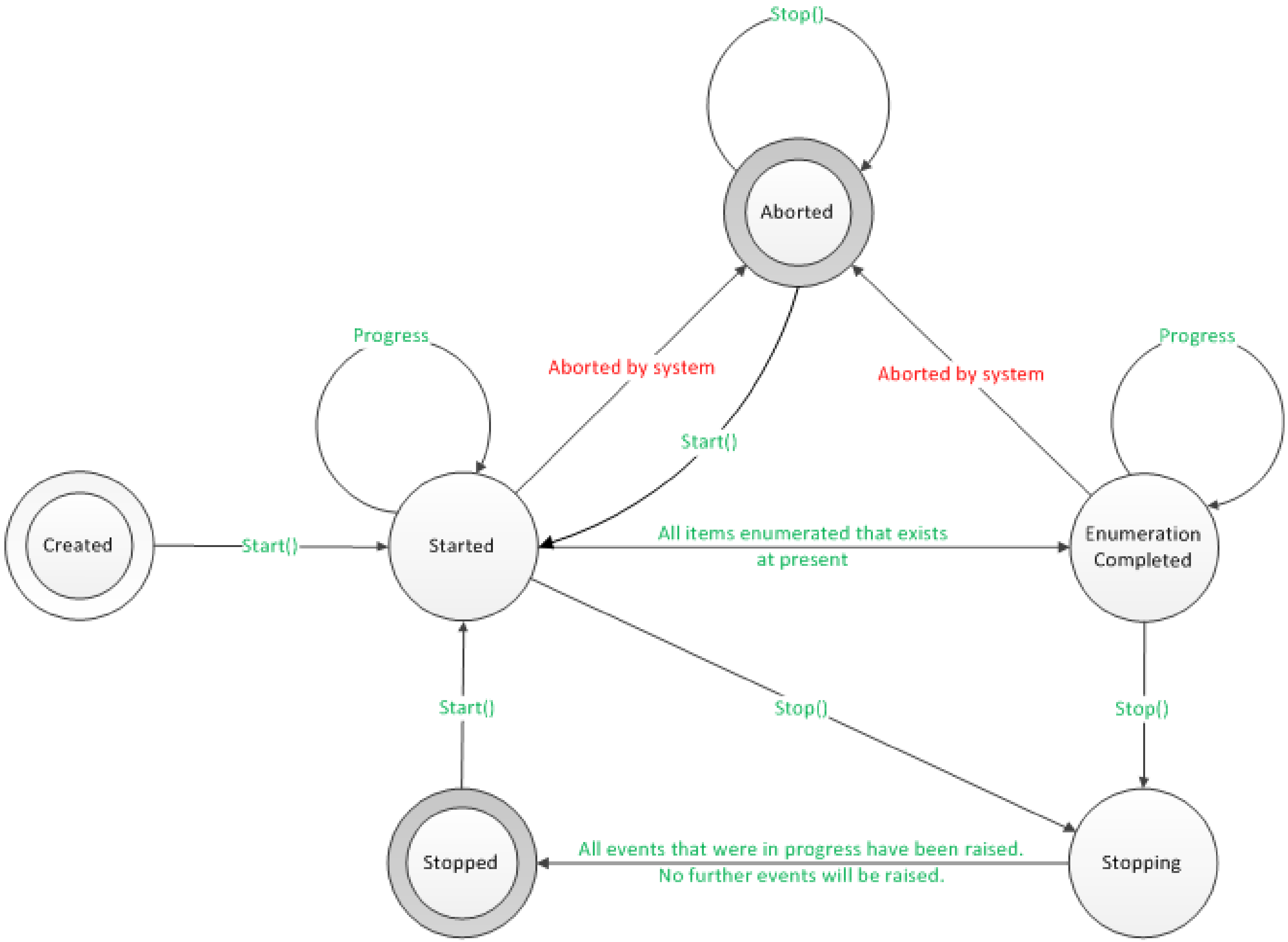
Windows.Devices.Enumeration

Enumerates devices dynamically, so that the app receives notifications if devices are added, removed, or changed after the initial enumeration is complete.

Remarks

An app calls Start to begin the search for devices. During this initial enumeration, the DeviceWatcher raises an Added event for each device that's found, until all devices are found. The DeviceWatcher raises an EnumerationCompleted event when the initial enumeration is complete, and continues to raise events if a device is added, updated, or removed.

The following diagram shows how the DeviceWatcher transitions between the states represented by DeviceWatcherStatus enumeration.



The Start method can only be called when the DeviceWatcher is in the **Created**, **Stopped** or **Aborted** state. The Status property indicates the DeviceWatcher state. When re-starting the watcher, wait for the Stopped event before calling Start.

Stop transitions the DeviceWatcher to the **Stopping** state and completes immediately. The watcher will transition to the **Stopped** state once all events that are already in the process of being raised have completed.

Apps may wait for the Stopped event if they need to know when the DeviceWatcher has stopped. Callers must wait for the Stopped event before they can call Start to restart the watcher. Callers may unsubscribe from events if they do not want to receive any additional events after Stop but do not want to wait for the Stopped event.

Note

An app must subscribe to all of the **added**, **removed**, and **updated** events to be notified when there are device additions, removals or updates. If an app handles only the **added** event, it will not receive an update if a device is added to the system after the initial device enumeration completes.

Properties

PROPERTIES

Status	The status of the DeviceWatcher.

Methods

METHODS

GetBackgroundTrigger(IEnumerable<DeviceWatcherEventKind>)	Gets a DeviceWatcherTrigger object monitoring for changes to the list of devices.
Start()	Starts a search for devices, and subscribes to device enumeration events.
Stop()	Stop raising the events that add, update and remove enumeration results.

Events

EVENTS

Added	Event that is raised when a device is added to the collection enumerated by the DeviceWatcher.
EnumerationCompleted	Event that is raised when the enumeration of devices completes.
Removed	Event that is raised when a device is removed from the collection of enumerated devices.
Stopped	Event that is raised when the enumeration operation has been stopped.
Updated	Event that is raised when a device is updated in the collection of enumerated devices.

DeviceWatcherEvent Class

Definition

Namespace:

Windows.Devices.Enumeration

Triggered when the list of devices is updated after the initial enumeration is complete.

Properties

PROPERTIES

DeviceInformation	Gets the information for the device associated with the DeviceWatcherEvent.
DeviceInformationUpdate	Gets updated information for a device updated by the DeviceWatcherEvent.
Kind	Gets the type for the device indicated by the DeviceWatcherEvent.

DeviceWatcherEventKind Enum

Definition

Namespace:

Windows.Devices.Enumeration

The type of event.

Fields

FIELDS

Add	0	The event is an add action.
Remove	2	The event is a remove action.
Update	1	The event is an update action.

DeviceWatcherStatus Enum

Definition

Namespace:

Windows.Devices.Enumeration

Describes the state of a DeviceWatcher object.

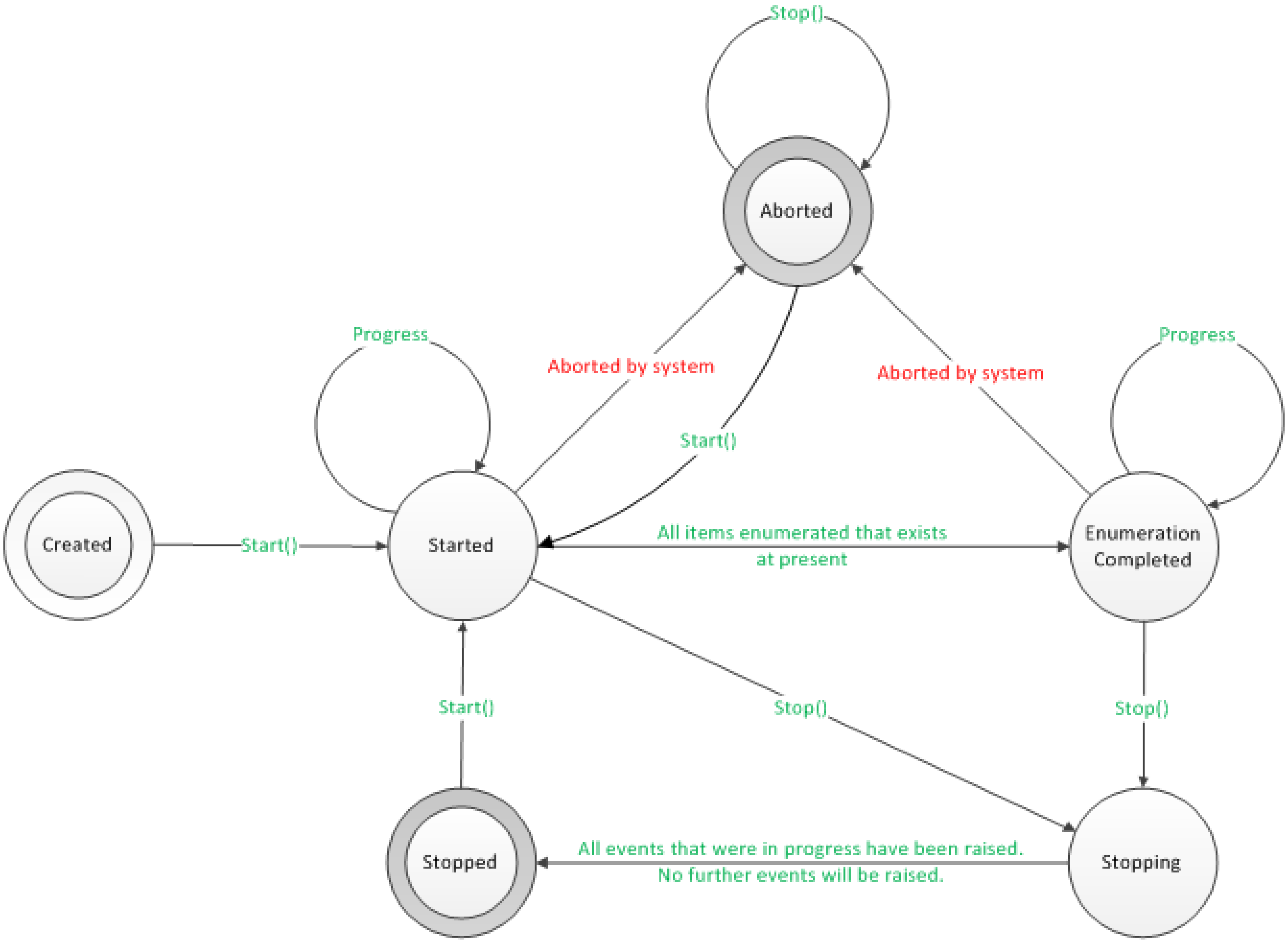
Fields

FIELDS

Aborted	5	The watcher has aborted operation. No subsequent events will be raised.
Created	0	This is the initial state of a Watcher object. During this state clients can register event handlers.
EnumerationCompleted	2	The watcher has completed enumerating the initial collection. Items can still be added, updated or removed from the collection.
Started	1	The watcher transitions to the Started state once Start is called. The watcher is enumerating the initial collection. Note that during this enumeration phase it is possible to receive Updated and Removed notifications but only to items that have already been Added.
Stopped	4	The client has called Stop and the watcher has completed all outstanding events. No further events will be raised.
Stopping	3	The client has called Stop and the watcher is still in the process of stopping. Events may still be raised.

Remarks

The following state diagram shows how the DeviceWatcher transitions between the states represented by DeviceWatcherStatus enumeration.



DeviceWatcherTriggerDetails Class

Definition

Namespace:

Windows.Devices.Enumeration

Provides details about the device updates that invoked the trigger.

Properties

	PROPERTIES
DeviceWatcherEvents	Gets the events that activated the trigger.

EnclosureLocation Class

Definition

Namespace:

Windows.Devices.Enumeration

Describes the physical location of a device in its enclosure.

Remarks

The EnclosureLocation property returns this object.

If the physical location of a device within its enclosure is unknown, the inDock and inLid properties are false and the Panel property has the value **Unknown**.

Properties

PROPERTIES

InDock	Indicates whether the device is physically located in the docking station of the portable computer.
InLid	Indicates whether the device is physically located in the lid of the portable computer.
Panel	Indicates which panel of the computer the device is physically located on.
RotationAngleInDegreesClockwise	Gets the physical degree of rotation for a device in its enclosure.

Panel Enum

Definition

Namespace:

Windows.Devices.Enumeration

Indicates the location of a panel on a computer. This enumeration is used for indicating the physical location of a device.

Fields

FIELDS

Back	2	The back panel of the computer.
Bottom	4	The bottom panel of the computer.
Front	1	The front panel of the computer.
Left	5	The left panel of the computer.
Right	6	The right panel of the computer.
Top	3	The top panel of the computer.
Unknown	0	The location of the panel is unknown.