



ATID Co., Ltd

# AT288N API Reference Guide for Android Developers

Android Developer Guide

SW Team

2023-06-12

## Revision History

Version	Revision Date	Reason <sup>1</sup>	Contents of Revision <sup>2</sup>	Author
v0.1	2016-05-25	Draft	Create the new document	KJ, Min
v0.2	2017-03-13	Modify	Supplementation and correction	YJ, Cho
v0.3	2019-01-21	Modify	Modified Intro comments.	YJ, Cho
v0.4	2023-06-12	Modify	Modified Intro comments.	SW Team

<sup>1</sup> Revision Reason : Compared to the previous file, add, modify or delete the contents of enactment or revision

<sup>2</sup> Revision Content : States the page number and changed contents that where to be revised.



# AT288N API Reference Guide for Android Developers

Android Developer Guide

Company

ATID Co., Ltd

Name of Doc.

Writer

SW Team

Date


2023-06-12

Version

v0.4

## Table of Contents

Table of Contents .....	3
1. Intro .....	4
2. Reference Library Guide .....	5
2.1. Reader Class .....	5
2.1.1. Constructor .....	5
2.1.2. Event Synchronization Method .....	5
2.1.3. Device Control Method .....	8
2.1.4. Action Method .....	10
2.1.5. Property Method .....	15
2.1.6. Extended Property Method .....	23
2.1.7. Static Method .....	29
2.2. IReaderCallbackReceiver Interface .....	30
2.2.1. Method .....	30
2.3. Enumeration .....	33
2.3.1. ModelType .....	33
2.3.2. TagType .....	33
2.3.3. BankType .....	33
2.3.4. CommType .....	33
2.3.5. InventoryFormatType .....	33
2.3.6. SelectionActionType .....	34
2.3.7. StoredModeType .....	34
2.4. Constant .....	35

		AT288N API Reference Guide for Android Developers					
Android Developer Guide					Company	ATID Co., Ltd	
Name of Doc.		Writer	SW Team	Date	2023-06-12	Version	v0.4


## 1. Intro

The purpose of this document is to describe how to use the SDK Library for the Android developers who want to develop applications.

The development tool that used in this document is Android Studio, and the target platform is Android 4.3 or later.

※ From v1.10.2019012200, Library, Demo and Sample had been built by Android Studio instead of Eclipse.

Library	Description
<del>at288lib.jar</del>	<del>Library for Android to control AT288N</del>
<b>at288lib.aar</b>	From v1.10.2019012200, jar had been changed to aar.

		AT288N API Reference Guide for Android Developers					
Android Developer Guide					Company	ATID Co., Ltd	
Name of Doc.		Writer	SW Team	Date	2023-06-12	Version	v0.4

## 2. Reference Library Guide

### 2.1. Reader Class

The Reader class is the class that uses the AT288 SDK Library to connect to or disconnect from the AT288N Device, and invokes method to control AT288N Device, sets or return attribute, receives the returned event from the AT288N Device as an event.

#### 2.1.1. Constructor

Initialize a new instance of the Reader class with the Activity's context and a Handler object which will receive event.

##### ➤ Syntax

```
public Reader(Context context, IReaderCallbackReceiver receiver)
```

##### ➤ Parameters

**context:** The user Context of Application

**receiver:** The interface that implemented in the application to process event that received from AT288N.

##### ➤ Remarks

Create a new instance of Reader class as an interface to process events and objects which inherited from Context.

#### 2.1.2. Event Synchronization Method

The Activity manages Life Cycle of Activity by redefine the synchronization event of onCreate, onStart, onResume, onPause, onStop, onDestroy, onActivityResult, etc.

The Reader object can also synchronize Activity and Life Cycle by calling the Life Cycle synchronization function in the redefined function.

##### 2.1.2.1. onCreate

Calls to do synchronize onCreate event in the onCreate method which is overridden by Activity that owned by Reader object.

##### ➤ Syntax

```
public boolean onCreate()
```

##### ➤ Return value

Returns false if Bluetooth is not supported.

##### ➤ Remarks

Synchronizes the owning Activity's onCreate event with the Reader object.

## 2.1.2.2. **onStart**

Calls to do synchronize onStart event in the onStart method which is overridden by Activity that owned by Reader object.

### ➤ **Syntax**

```
public void onStart()
```

### ➤ **Remarks**

Synchronizes the owning Activity's onStart event with the Reader object.

## 2.1.2.3. **onResume**

Calls to do synchronize onResume event in the onResume method which is overridden by Activity that owned by Reader object.

### ➤ **Syntax**

```
public void onResume()
```

### ➤ **Remarks**

Synchronizes the owning Activity's onResume event with the Reader object.

## 2.1.2.4. **onPause**

Calls to do synchronize onPause event in the onPause method which is overridden by Activity that owned by Reader object.

### ➤ **Syntax**

```
public void onPause()
```

### ➤ **Remarks**

Synchronizes the owning Activity's onPause event with the Reader object.

## 2.1.2.5. **onStop**

Calls to do synchronize onStop event in the onStop method which is overridden by Activity that owned by Reader object.

### ➤ **Syntax**

```
public void onStop()
```

### ➤ **Remarks**

Synchronizes the owning Activity's onStop event with the Reader object.

## 2.1.2.6. **onDestroy**

Calls to do synchronize onDestroy event in the onDestroy method which is overridden by Activity that owned by Reader object.

### ➤ **Syntax**

```
public void onDestroy()
```

## ➤ Remarks

Synchronizes the owning Activity's onDestroy event with the Reader object.

### 2.1.2.7. **onActivityResult**

Calls the openDeviceListActivity method, which is the method that Intent is invoked on, and then calls from the onActivityResult method which had overridden of Activity that called openDeviceListActivity to synchronize the value being returned by Intent.

## ➤ Syntax

```
public void onActivityResult (int requestCode, int resultCode, Intent data)
```

## ➤ Parameters

**requestCode:** The requestCode of the onActivityResult method which is overridden of App Activity.

**resultCode:** The resultCode of onActivityResult method which is overridden of App Activity.

**data:** The data of onActivityResult method which is overridden of App Activity.

## 2.1.3. Device Control Method

Device Control methods provide a way to connect to the AT288N depends on the situation after an instance of the Reader class has been created.

### 2.1.3.1. **openDeviceListActivity**

Outputs a dialog box to search for Peripheral Bluetooth device and by selecting the AT288N, which is searched, try the Bluetooth connection.

#### ➤ **Syntax**

```
public void openDeviceListActivity()
```

#### ➤ **Remarks**

Bluetooth search Activity is created, perform Bluetooth inquiry to search for AT288N Device which peripheral device of Host device.

If you select the searched AT288N Device, attempts to connect to the selected device and the connection result will be passed to the onReaderStateChange method of the IReaderCallbackReceiver interface. See 2.4 Constants (Connection State).

### 2.1.3.2. **connectMostRecentDevice**

ConnectMostRecentDevice method is the method to try connecting to the AT288N Device which connected with the Reader recently.

#### ➤ **Syntax**

```
public void connectMostRecentDevice()
```

#### ➤ **Remarks**

Trying to connect to AT288N Device which recently connected by Bluetooth, the connection result will be passed to the onReaderStateChange method of the IReaderCallbackReceiver interface. See 2.4 Constants (Connection State).

### 2.1.3.3. **ensureDiscoverable**

Sets as connection wait state to be able to allow the AT288N Device to connect to the Reader.

#### ➤ **Syntax**

```
public void ensureDiscoverable()
```

#### ➤ **Remarks**

When AT288N Device is in Bluetooth mode, when the power is turned on, search the Host device that recently connected to try Bluetooth connection. At this time, it is possible to connect to the Reader only when it is in connection wait state.



When the AT288N Device is connected, the result will be passed to onReaderStateChange method of the IReaderCallbackReceiver interface. See 2.4 Constants (Connection State).

## 2.1.3.4. **activate**

Synchronizes the state of the Reader object and the AT288N Device.

### ➤ **Syntax**

```
public void activate()
```

### ➤ **Remarks**

If attempts to connect, the result will be passed to onReaderStateChange of IReaderCallbackReceiver interface. If the received state is STATE\_CONNECTED, synchronize the application and the AT288N Device by calling the activate method. When you call the activate method, pass the attribute value to onReaderExtendedProperty method of IReaderCallbackReceiver interface. See 2.4 Constants (Extended Property Codes).

## 2.1.3.5. **start**

Prepares to communicate to the AT288N Device.

### ➤ **Syntax**

```
public void start()
```

### ➤ **Remarks**

This method is the method that being called the inside the SDK when you try to connect by using openDeviceListActivity or connectMostRecentDevice, so you should not call it arbitrarily.

## 2.1.3.6. **stop**

Immediately end the connection with the AT288N and the Host.

### ➤ **Syntax**

```
public void stop()
```

### ➤ **Remarks**

This method is executed when onDestroy is called and use it when end the connection with the Host.

## 2.1.4. Action Method

Provides Tag operation function.

### 2.1.4.1. inventory

Performs the Inventory function that based on what was set as setTagType, setContinueModeEx method.

#### ➤ Syntax

```
public void inventory(String mask)
```

#### ➤ Parameters

**mask:** Masking pattern that for select the Tag (Hexadecimal string).

#### ➤ Remarks

Tag data that became the Inventory will be passed to the onReaderReadTag method of IReaderCallbackReceiver interface. To use the Selection function, the location where to apply the masking pattern should be set by SetSelectionBank, SetSelectionOffset, SetSelectionAction, etc method.

### 2.1.4.2. inventory6b6cAnyone

Performs Inventory function without classification of ISO 18000-6B/6C.

#### ➤ Syntax

```
public void inventory6b6cAnyone()
```

#### ➤ Remarks

Can use only if the call result of the IsAT288N\_MA method is true. Tag data that became the Inventory will be passed to the onReaderReadTag method of IReaderCallbackReceiver interface.

### 2.1.4.3. inventoryMemory

Performs Inventory operation and read specific memory Bank value of the Tag together.

#### ➤ Syntax

```
public void inventoryMemory(BankType bank, int offset, int length, String mask)
```

#### ➤ Parameters

**bank:** Memory Bank of the Tag to access

**offset:** Start address of Memory Bank of the Tag to access

**length:** Memory length to access

**mask:** Masking pattern to select the Tag (Hexadecimal string)

#### ➤ Remarks

Tag data that became the Inventory will be passed to the onReaderReadTag method of

IReaderCallbackReceiver interface. Cannot use if the call result of the IsAT288N\_MA method is true.

## 2.1.4.4. **inventoryMultiple**

Performs Inventory function as Multiple mode.

### ➤ **Syntax**

```
public void inventoryMultiple()
```

### ➤ **Remarks**

Tag data that became the Inventory will be passed to the onReaderReadTag method of IReaderCallbackReceiver interface.

## 2.1.4.5. **inventorySelection**

Performs Inventory function of the specific Tags.

### ➤ **Syntax**

```
public void inventorySelection(String mask)
```

### ➤ **Parameters**

**mask:** Masking pattern to select the Tag (Hexadecimal string).

### ➤ **Remarks**

Performs Inventory only for the Tag that has the data which start from pattern entered as mask, Tag data that became the Inventory will be passed to the onReaderReadTag method of IReaderCallbackReceiver interface.

## 2.1.4.6. **inventorySingle**

Performs Inventory function as Single mode.

### ➤ **Syntax**

```
public void inventorySingle()
```

### ➤ **Remarks**

Stop after performing Inventory once, and the Tag data that became the Inventory will be passed to the onReaderReadTag method of IReaderCallbackReceiver interface.

## 2.1.4.7. **inventoryFiltering**

Performs Inventory function.

### ➤ **Syntax**

```
public void inventoryFiltering(String action, String mask1, String mask2)
```

### ➤ **Remarks**

Not use.

## 2.1.4.8. **inventoryVLC**

Performs Inventory function.

### ➤ **Syntax**

```
public void inventoryVLC()
```

### ➤ **Remarks**

Not use.

## 2.1.4.9. **stopOperation**

Stop operation of the Action series that currently acting.

### ➤ **Syntax**

```
public void stopOperation()
```

### ➤ **Remarks**

Cancels the action that being performed, and forward that it has been stop as onReaderActionChange method of IReaderCallbackReceiver interface.

Action: ACTION\_STOP

## 2.1.4.10. **readMemory**

Read specific memory Bank value of the Tag.

### ➤ **Syntax**

```
public void readMemory(BankType bank, int location, int length)
public void readMemory(BankType bank, int offset, int length, String mask)
```

### ➤ **Parameters**

**bank:** Memory Bank of the Tag to access

**location, offset:** Start address of Memory Bank of the Tag to access

**length:** Memory length to access

**mask:** Masking pattern to select the Tag (Hexadecimal string)

### ➤ **Remarks**

The read Tag data will be passed to the onReaderReadTag method of IReaderCallbackReceiver interface, and if it is failed, it will forward a fail code to onReaderResponse method.

See 2.2.1.3 onReaderResponse.

Cannot use if the call result of IsAT288N\_MA method is true and the result of getTagType is ISO18000\_6B.

## 2.1.4.11. **writeMemory**

Records the value to specific memory Bank of the Tag.

### ➤ **Syntax**

```
public void writeMemory(BankType bank, int location, String data)
public void writeMemory(BankType bank, int offset, String data, String mask)
```

## ➤ Parameters

**bank:** Memory Bank of the Tag to access

**location, offset:** Start address of Memory Bank of the Tag to access

**data:** Data to record to the Tag (Hexadecimal string)

**mask:** Masking pattern to select the Tag (Hexadecimal string)

## ➤ Remarks

The perform result will be passed to onReaderResponse method of IReaderCallbackReceiver interface. See 2.2.1.3 onReaderResponse.

Cannot use if the call result of IsAT288N\_MA method is true and the result of getTagType is ISO18000\_6B.

### 2.1.4.12. lock

Perform locks or unlocks to prevent access to the Tag

## ➤ Syntax

```
public void lock(String bitMask, String action)
public void lock(String bitMask, String action, String mask)
```

## ➤ Parameters

**bitMask:** A value that determines whether apply the entered value by action (4-digit hexadecimal string).

**action:** A value that to lock the set location (4-digit hexadecimal string).

**mask:** Masking pattern to select the Tag (Hexadecimal string).

## ➤ Remarks

The bitmask and action value are calculated by referring to the following table.

Mask/Action State

Offset	9	8	7	6	5	4	3	2	1	0
Mask	Kill Pwd		Access Pwd		EPC		TID		User	
	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write

Offset	9	8	7	6	5	4	3	2	1	0
Action	Kill Pwd		Access Pwd		EPC		TID		User	
	pwd read/ write	perma lock	pwd read/ write	perma lock	pwd write	perma lock	pwd write	perma lock	pwd write	perma lock

The perform result will be passed to onReaderResponse method of IReaderCallbackReceiver interface. See 2.2.1.3 onReaderResponse.

Cannot use if the call result of IsAT288N\_MA method is true and the result of getTagType is ISO18000\_6B.

## 2.1.4.13. kill

Makes the Tag no longer available.

### Syntax

```
public void kill(String password)
public void kill(String password, String mask)
```

### ➤ Parameters

**password** : The kill password which saved in the Tag that will conduct kill.

**mask** : Masking pattern to select the Tag (Hexadecimal string).

### ➤ Remarks

The perform result will be passed to onReaderResponse method of IReaderCallbackReceiver interface and should be careful to use the kill method because the Tag which successes kill cannot revert to use again. See 2.2.1.3 onReaderResponse. Cannot use if the call result of IsAT288N\_MA method is true and the result of getTagType is ISO18000\_6B.

## 2.1.5. Property Method

Provides the function of module property setting and the property of Tag operation.

### 2.1.5.1. getModelType

Returns the model type of AT288N that connected to Host.

#### ➤ Syntax

```
public int getModelType()
```

#### ➤ Return value

The value of AT288N's type (Reference the value of ModelType Enumeration).

#### ➤ Remarks

It must be used after the activate method call.

### 2.1.5.2. getFirmwareVersion

Reads AT288N's UHF module's F/W version.

#### ➤ Syntax

```
public void getFirmwareVersion()
```

#### ➤ Remarks

The result will be passed to parameter of onReaderProperty of IReaderCallbackReceiver.  
Code: PROPERTY\_VERSION

### 2.1.5.3. getGlobalBand

Reads the country set value of UHF module.

#### ➤ Syntax

```
public void getGlobalBand()
```

#### ➤ Remarks

- The result will be passed to parameter of onReaderProperty of IReaderCallbackReceiver.  
Code: PROPERTY\_GLOBAL\_BAND

### 2.1.5.4. getState

Returns the connection status of the AT288N and the Host.

#### ➤ Syntax

```
public int getState()
```

#### ➤ Return value

The value that indicates connection status. See 2.4 Constants (Connection State).

### 2.1.5.5. getPower

Reads the Antenna transmit power level.

➤ **Syntax**

```
public void getPower()
```

➤ **Remarks**

- The result will be passed to parameter of onReaderProperty of IReaderCallbackReceiver.  
Code: PROPERTY\_POWER\_GAIN

## 2.1.5.6. **setPower**

Sets the Antenna transmit power level.

➤ **Syntax**

```
public void setPower(int value)
```

➤ **Parameters**

**value:** Transmit power level (0~19)

➤ **Remarks**

If the value is 0, it is the maximum output (30dBm), and if increase the value, the output will decrease by 1dBm. For example, if the value is set to 5, the output power level will be 25dBm.

## 2.1.5.7. **getAccessPassword**

Reads Access password that set on the AT288N.

➤ **Syntax**

```
public void getAccessPassword()
```

➤ **Remarks**

- The result will be passed to parameter of onReaderProperty of IReaderCallbackReceiver.  
Code: PROPERTY\_ACCESS\_PASSWORD

## 2.1.5.8. **setAccessPassword**

Sets Access password on the AT288N.

➤ **Syntax**

```
public void setAccessPassword(String password)
```

➤ **Parameters**

**password:** access password (Hexadecimal string).

## 2.1.5.9. **getContinueMode**

Reads whether Inventory mode is multiple.

➤ **Syntax**

```
public void getContinueMode()
```

➤ **Remarks**



The result will be passed to parameter of onReaderProperty of IReaderCallbackReceiver.

Code: PROPERTY\_CONTINUE\_MODE

## 2.1.5.10. setContinueMode

Sets or release Inventory mode as multiple mode.

### ➤ Syntax

```
public void setAccessPassword(boolean enabled)
```

### ➤ Parameters

**enabled:** Whether to set multiple mode

### ➤ Remarks

If the enabled is true, it is multiple mode and if the enabled is false, it is single mode.

## 2.1.5.11. getSelectionBank

Reads Memory Bank of the Tag that will perform the Select.

### ➤ Syntax

```
public void getSelectionBank()
```

### ➤ Remarks

The result will be passed to parameter of onReaderProperty of IReaderCallbackReceiver.

Code: PROPERTY\_SELECTION\_BANK

## 2.1.5.12. setSelectionBank

Sets Memory Bank of the Tag that will perform the Select.

### ➤ Syntax

```
public void setSelectionBank(BankType type)
```

### ➤ Parameters

**type:** Memory bank

### ➤ Remarks

About type, see BankType Enumeration.

## 2.1.5.13. getSelectionOffset

Reads the address where comparison begins of the masking pattern when the Select is conducted.

### ➤ Syntax

```
public void getSelectionOffset()
```

### ➤ Remarks

The result will be passed to parameter of onReaderProperty of IReaderCallbackReceiver.

Code: PROPERTY\_SELECTION\_OFFSET

## 2.1.5.14. **setSelectionOffset**

Sets the address where comparison begins of the masking pattern when the Select is conducted.

### ➤ **Syntax**

```
public void setSelectionOffset(int offset)
```

### ➤ **Parameters**

**offset:** Start address

### ➤ **Remarks**

Sets the start address as bit unit.

## 2.1.5.15. **getSelectionAction**

Reads the action that will be applied when the Select is conducted.

### ➤ **Syntax**

```
public void getSelectionAction()
```

### ➤ **Remarks**

The result will be passed to parameter of onReaderProperty of IReaderCallbackReceiver.

Code: PROPERTY\_SELECTION\_ACTION

## 2.1.5.16. **setSelectionAction**

Sets action that will be applied when the Select is conducted.

### ➤ **Syntax**

```
public void setSelectionAction(SelectionAction type)
```

### ➤ **Parameters**

**type:** The action that will apply

### ➤ **Remarks**

About type, see SelectionAction Enumeration.

## 2.1.5.17. **getSelectionSession**

Reads session that will be applied when the Select is conducted.

### ➤ **Syntax**

```
public void getSelectionSession()
```

### ➤ **Remarks**

The result will be passed to parameter of onReaderProperty of IReaderCallbackReceiver.

Code: PROPERTY\_SELECTION\_SESSION

## 2.1.5.18. **setSelectionSession**

Sets the session that will be applied when the Select is conducted.

➤ **Syntax**

```
public void setSelectionSession(int value)
```

➤ **Parameters**

**value:** The session that will apply

➤ **Remarks**

Value: 0(S0), 1(S1), 2(S2), 3(S3)

## 2.1.5.19. **getBuzzer**

Reads whether the AT288N's buzzer is used.

➤ **Syntax**

```
public void getBuzzer()
```

➤ **Remarks**

The result will be passed to parameter of onReaderProperty of IReaderCallbackReceiver.

Code: PROPERTY\_BUZZER

## 2.1.5.20. **setBuzzer**

Sets the On/Off status of the AT288N's buzzer.

➤ **Syntax**

```
public void setBuzzer(boolean enabled)
```

➤ **Parameters**

**enabled:** on/off

➤ **Remarks**

enabled: true(on), false(off)

## 2.1.5.21. **getAutoPowerOffTime**

Reads the latency time value until the AT288N is power off when it is not connected to the Host and no input is received from the user.

➤ **Syntax**

```
public void getAutoPowerOffTime()
```

➤ **Remarks**

The result will be passed to parameter of onReaderProperty of IReaderCallbackReceiver.

Code: PROPERTY\_AUTO\_POWEROFF

## 2.1.5.22. **setAutoPowerOffTime**

Sets the latency time value until the AT288N is power off when it is not connected to the host and no input is received from the user.

➤ **Syntax**

```
public void setAutoPowerOffTime(int value)
```

➤ **Parameters**

**value:** The latency time until power off.

➤ **Remarks**

The unit is minute and if set as 0, it will not power off until the battery runs out.

## 2.1.5.23. **getResponseTimeout**

Returns the command execution time.

➤ **Syntax**

```
public int getResponseTimeout()
```

➤ **Return value**

The Command execution time (1000 ~ 10000).

➤ **Remarks**

The unit is millisecond.

## 2.1.5.24. **setResponseTime**

Sets the command execution time.

**Syntax**

```
public void setResponseTime(int nTimeout)
```

➤ **Parameters**

**nTimeout:** The command execution time.

➤ **Remarks**

The unit is millisecond.

## 2.1.5.25. **getDeviceName**

Returns the name of friendly Bluetooth of AT288N which connected with the Host.

➤ **Syntax**

```
public String getDeviceName()
```

➤ **Return value**

Returns the name of friendly Bluetooth of AT288N as String

## 2.1.5.26. **getPortInventoryTime**

Reads the time that the antenna is activated among the Inventory Round time.

➤ **Syntax**

```
public void getPortInventoryTime()
```

➤ **Remarks**

The result will be passed to parameter of onReaderProperty of IReaderCallbackReceiver.

Code: PROPERTY\_ANTENNA\_SWITCHING\_TIME

## 2.1.5.27. **setPortInventoryTime**

Sets the time that the antenna is activated among the Inventory Round time.

### ➤ **Syntax**

```
public void setPortInventoryTime(int value)
```

### ➤ **Parameters**

**value:** Activation time.

### ➤ **Remarks**

The unit is millisecond.

## 2.1.5.28. **getPortIdleTime**

Reads idle time of the antenna among the Inventory Round time.

### ➤ **Syntax**

```
public void getPortIdleTime()
```

### ➤ **Remarks**

The result will be passed to parameter of onReaderProperty of IReaderCallbackReceiver.

Code: PROPERTY\_POWER\_IDLE\_TIME

## 2.1.5.29. **setPortIdleTime**

Sets idle time of the antenna among the Inventory Round time.

### ➤ **Syntax**

```
public void setPortIdleTime(int value)
```

### ➤ **Parameters**

**value:** Idle time

### ➤ **Remarks**

The unit is millisecond.

## 2.1.5.30. **getLBTChannel**

Reads the LBT Channel information.

### ➤ **Syntax**

```
public void getLBTChannel()
```

### ➤ **Remarks**

This method is only valid when the value that received in getModelType is MT\_AT288Japan.

The result will be passed to parameter of onReaderProperty of IReaderCallbackReceiver.

Code: PROPERTY\_LBT\_CHANNEL

## 2.1.5.31. **setLBTChannel**

Sets the LBT Channel information.

### ➤ **Syntax**

```
public void setLBTChannel( int value)
```

### ➤ **Parameters**

**value:** LBT Channel information

### **Remarks**

Creates the value by referring to the following table. In the following table, if the bit is 1, use the corresponding frequency and if it is 0, do not use it.

Bit	8Bit	7Bit	6Bit	5Bit	4Bit	3Bit	2Bit	1Bit	0Bit
Ch No.	9	8	7	6	5	4	3	2	1
Freq.(MHz)	923.4	922.8	922.2	921.6	921.0	920.4	919.2	918.0	916.8

This method is only valid when the value that received in getModelType is MT\_AT288Japan.

## 2.1.5.32. **defaultSettings**

Returns the setting value to the default value.

### ➤ **Syntax**

```
public void defaultSettings()
```

## 2.1.5.33. **saveSettings**


Saves the changed setting value.

### ➤ **Syntax**

```
public void saveSettings()
```

### ➤ **Remarks**

Sets to be applied the latest saved contents even though AT288N turns on again after it is powered off.

		AT288N API Reference Guide for Android Developers					
Android Developer Guide					Company	ATID Co., Ltd	
Name of Doc.		Writer	SW Team	Date	2023-06-12	Version	v0.4

### 2.1.6. Extended Property Method

Provides the function of module property setting and the property of Tag operation that was expanded.

#### 2.1.6.1. **getAllExtendedProperty**

Reads expanded properties.

##### ➤ Syntax

```
public void getAllExtendedProperty()
```

##### ➤ Remarks

If you call this method, the following property values will be passed to parameter of onReaderExtendedProperty of IReaderCallbackReceiver in order.

- PROPERTY\_EX\_POWER\_GAIN
- PROPERTY\_EX\_TAG\_TYPE
- PROPERTY\_EX\_COMMUNICATION\_TYPE
- PROPERTY\_EX\_CONTINUE\_MODE
- PROPERTY\_EX\_STORED\_MODE
- PROPERTY\_EX\_USE\_SERIAL\_NO

#### 2.1.6.2. **getSerialNo**

Reads AT288N's Serial Number.

##### ➤ Syntax

```
public void getSerialNo()
```

##### ➤ Remarks

The result will be passed to parameter of onReaderExtendedProperty of IReaderCallbackReceiver.

Code: PROPERTY\_EX\_SERIAL\_NO

#### 2.1.6.3. **getPowerState**

Reads the AT288N power On/Off status.

##### ➤ Syntax

```
public void getPowerState()
```

##### ➤ Remarks

The result will be passed to parameter of onReaderExtendedProperty of IReaderCallbackReceiver.

Code: PROPERTY\_EX\_POWER\_STATE

## 2.1.6.4. **getPowerEx**

Reads the Antenna transmit power level.

### ➤ **Syntax**

```
public boolean getPowerEx()
```

### ➤ **Remarks**

The result will be passed to parameter of onReaderExtendedProperty of IReaderCallbackReceiver.

Code: PROPERTY\_EX\_POWER\_GAIN

## 2.1.6.5. **setPowerEx**

Sets the Antenna transmit power level.

### ➤ **Syntax**

```
public void setPowerEx(int value)
```

### ➤ **Parameters**

**value:** Transmit power level (0~19)

### ➤ **Remarks**

If the value is 0, it is the maximum output (30dBm), and if increase the value, the output will decrease by 1dBm. For example, if the value is set to 5, the output power level will be 25dBm.

## 2.1.6.6. **getTagType**

The AT288N reads the Tag Type that set to be recognized.

### ➤ **Syntax**

```
public void getTagType()
```

### ➤ **Remarks**

The result will be passed to parameter of onReaderExtendedProperty of IReaderCallbackReceiver.

Code: PROPERTY\_EX\_TAG\_TYPE

## 2.1.6.7. **setTagType**

Sets the Tag Type that AT288N will be recognize.

### ➤ **Syntax**

```
public void setTagType(TagType type)
```

### ➤ **Parameters**

**type:** Set Tag type

### ➤ **Remarks**

About the type, see 2.3.2 TagType enumeration. ISO18000\_6B is only valid when the return



value of IsAT288N\_MA is true. And after it set to ISO18000\_6B, cannot use read, write, lock and kill.

## 2.1.6.8. **getCommunicationType**

Reads the connection method that been set between Host and AT288N currently.

### ➤ **Syntax**

```
public void getCommunicationType()
```

### ➤ **Remarks**

The result will be passed to parameter of onReaderExtendedProperty of IReaderCallbackReceiver.

Code: PROPERTY\_EX\_COMMUNICATION\_TYPE

## 2.1.6.9. **setCommunicationType**

Sets the connection method between Host and AT288N.

### ➤ **Syntax**

```
public void setCommunicationType(CommType type)
```

### ➤ **Parameters**

**type:** Connection method

### ➤ **Remarks:** About the Type, see 2.3.4 CommType enumeration.

## 2.1.6.10. **getContinueModeEx**

Reads Inventory type.

### ➤ **Syntax**

```
public void getContinueModeEx()
```

### ➤ **Remarks**

The result will be passed to parameter of onReaderExtendedProperty of IReaderCallbackReceiver.

Code: PROPERTY\_EX\_CONTINUE\_MODE

## 2.1.6.11. **setContinueModeEx**

Sets Inventory type.

### ➤ **Syntax**

```
public void setContinueModeEx(boolean enabled)
```

### ➤ **Parameters**

**enabled:** Inventory type

### ➤ **Remarks:** If Enabled is true, it is multiple and if Enabled is false, it is single.

## 2.1.6.12. **setContinueModeExN**

Sets Inventory type.

### ➤ **Syntax**

```
public void setContinueModeExN(int value)
```

### ➤ **Parameters**

**value:** inventory type

### ➤ **Remarks:** About the value, see 2.4 Constants (Continue Mode Type).

## 2.1.6.13. **getStoredMode**

Reads the enable/disable status of save mode.

### ➤ **Syntax**

```
public void getStoredMode()
```

### ➤ **Remarks**

The result will be passed to parameter of onReaderExtendedProperty of IReaderCallbackReceiver.

Code: PROPERTY\_EX\_STORED\_MODE

## 2.1.6.14. **setStoredMode**

Sets the enable/disable status of save mode.

### ➤ **Syntax**

```
public void setStoredMode(StoredModeType mode)
```

### ➤ **Parameters**

**mode:** Save mode

### ➤ **Remarks:** About the Mode, see 2.3.7 StoredModeType enumeration.

## 2.1.6.15. **getStoredData**

Reads the Tag list that saved in the AT288N.

### ➤ **Syntax**

```
public void getStoredData()
```

### ➤ **Remarks**

The Tag data will be passed to parameter of onReaderReadTag of IReaderCallbackReceiver, and when the data transfer is finished, it will be passed that it ended by onReaderExtendedProperty method.

Code: PROPERTY\_EX\_GET\_STORED\_DATA

## 2.1.6.16. **deleteStoredData**

Deletes the Tag list that saved in the AT288N.

➤ **Syntax**

```
public void getStoredData()
```

➤ **Remarks**

Starts to delete the saved data and when it finishes, it will be passed that it ended by onReaderExtendedProperty method.

Code: PROPERTY\_EX\_DELETE\_STORED\_DATA

## 2.1.6.17. **getVersionEx**

Reads main board F/W version of AT288N.

➤ **Syntax**

```
public void getVersionEx()
```

➤ **Remarks**

The result will be passed to parameter of onReaderExtendedProperty of IReaderCallbackReceiver.

Code: PROPERTY\_EX\_VERSION

## 2.1.6.18. **getInventoryFormat**

Reads the data format that is being received as the result of Inventory.

➤ **Syntax**

```
public void getInventoryFormat()
```

➤ **Remarks**

The result will be passed to parameter of onReaderExtendedProperty of IReaderCallbackReceiver.

Code: PROPERTY\_EX\_INVENTORY\_FORMAT

## 2.1.6.19. **setInventoryFormat**

Sets the data format that is being received as the result of Inventory.

➤ **Syntax**

```
public void setInventoryFormat(InventoryFormatType type)
```

➤ **Parameters**

**type:** Inventory format type

➤ **Remarks**

About the type, see 2.3.5 InventoryFormatType enumeration.

## 2.1.6.20. **getUseSerialNo**

Reads whether the Serial number of AT288N includes in the data that is being received as the result of Inventory.

➤ **Syntax**

```
public void getUseSerialNo()
```

➤ **Remarks**

The result will be passed to parameter of onReaderExtendedProperty of IReaderCallbackReceiver.

Code: PROPERTY\_EX\_USE\_SERIAL\_NO

## 2.1.6.21. **setUseSerialNo**

Sets whether the Serial number of AT288N includes in the data that is being received as the result of Inventory.

➤ **Syntax**

```
public void setUseSerialNo(boolean enabled)
```

➤ **Parameters**

**enabled:** True (includes serial number), false (not includes serial number).

## 2.1.6.22. **getBatteryState**

Reads remaining battery level of AT288N.

➤ **Syntax**

```
public void getBatteryState()
```

➤ **Remarks**

➤ The result will be passed to parameter of onReaderExtendedProperty of IReaderCallbackReceiver.

Code: PROPERTY\_EX\_BATTERY\_STATE

## 2.1.6.23. **IsAT288N**

Returns the AT288N which connected to Host is New version or not.

➤ **Syntax**

```
public boolean IsAT288N()
```

➤ **Return value:** True if AT288N, false if AT288.

## 2.1.6.24. **IsAT288N\_MA**

Returns the AT288 which connected to Host is the New version MA type or not.

➤ **Syntax**

```
public boolean IsAT288N_MA()
```

➤ **Return value**

True if AT288N MA version, false if not.

## 2.1.7. Static Method

### 2.1.7.1. **getResponses**

Converts the code, which received to event handler, to a string.

#### ➤ **Syntax**

```
public static String getResponses(String code)
```

#### ➤ **Parameters**

**code:** The code that received to event handler.

#### ➤ **Return value**

If success, returns the result string that corresponding to the code, and if it fails, returns an empty string.

### 2.1.7.2. **isBluetoothSupported**


Returns whether support Bluetooth of the Host that running by Reader class object.

#### ➤ **Syntax**

```
public static boolean isBluetoothSupported()
```

#### ➤ **Return value**

True if Bluetooth is supported to the Host, false if not.

		AT288N API Reference Guide for Android Developers					
Android Developer Guide					Company	ATID Co., Ltd	
Name of Doc.		Writer	SW Team	Date	2023-06-12	Version	v0.4

## 2.2. IReaderCallbackReceiver Interface

For the receiving the event that occurred on AT288N, in the App, need to implement IReaderCallbackReceiver then pass it to the conductor of the Reader class.

### 2.2.1. Method

#### 2.2.1.1. onReaderStateChanged

Executes when the connection state is changed between AT288N and the Host.

##### ➤ Syntax

```
void onReaderStateChanged(int state);
```

##### ➤ Parameters

**state:** Connection state

##### ➤ Remarks

About state, see 2.4 Constants (Connection State).

#### 2.2.1.2. onReaderReadTag

Executes when read the Tag data.

##### ➤ Syntax

```
void onReaderReadTag(int event, String tag);
```

##### ➤ Parameters

**event:** Event type

**tag:** Tag data

##### ➤ Remarks

About event, see 2.4 Constants (Event Type).

#### 2.2.1.3. onReaderResponse

Executes when Access methods are completed or aborted.

##### ➤ Syntax

```
void onReaderResponse(int event, String code);
```

##### ➤ Parameters

**event:** Event type

**code:** Performance result

##### ➤ Remarks

About event, see 2.4 Constants (Event Type).

The value that received as a code is hexadecimal string but can convert to response string of string type by using getResponses method. See the table below.

code	string	Description
"00"	OtherError	Errors caused by unknown reason
"01"	Success	Result success
"02"	Undefined	Undefined error
"03"	MemoryOverrun	Out of accessible memory range
"04"	MemoryLocked	Memory is locked
"05"	Timeout	Excess the access time
"0B"	InsufficientPower	Power shortage
"0F"	NonSpecificError	Undefined error
"FF"	AckHeartbeat	Fatal communication error

## 2.2.1.4. onReaderProperty

Executes when it received the request of attribute value return.

### ➤ Syntax

```
void onReaderResponse(char code, String value);
```

### ➤ Parameters

**code:** Attribute value type

**value:** Attribute value

### ➤ Remarks

About code, see 2.4 Constants (Property Codes).

## 2.2.1.5. onReaderExtendedProperty

Executes when it received the request of expanded attribute value return.

### ➤ Syntax

```
void onReaderExtendedProperty(char code, String value);
```

### ➤ Parameters

**code:** Expanded attribute value type

**value:** Attribute value

### ➤ Remarks

About code, see 2.4 Constants (Extended Property Codes).

## 2.2.1.6. onReaderActionChange

Executes when invoking the Action method and receives action type that correspond to the invoked method.

### ➤ Syntax

```
void onReaderActionChange(char action);
```

## ➤ Parameters

**action:** Action type

## ➤ Remarks

About action, see 2.4 Constants (Action Codes).

### 2.2.1.7. **onReaderTimeout**

Executes when Access method is not able to complete within time that set by setResponseTime.

## ➤ Syntax

```
void onReaderTimeout();
```

## ➤ Remarks

About the code, see 2.4 Constants (Extended Property Codes).

### 2.2.1.8. **onReaderMessage**

Executes when the Bluetooth connection is failed or lost the connection.


## ➤ Syntax

```
void onReaderMessage(String msg);
```

## ➤ Parameters

**msg:** Error string



		AT288N API Reference Guide for Android Developers					
Android Developer Guide					Company	ATID Co., Ltd	
Name of Doc.		Writer	SW Team	Date	2023-06-12	Version	v0.4

## 2.3. Enumeration

### 2.3.1. ModelType

Defines AT288N Device type.

Flag	Value	Description
None	0	Unknown
AT288	1	Standard
AT288Japan	2	Japanese

### 2.3.2. TagType

Defines Tag type that to recognizes as AT288N.

Flag	Value	Description
ISO18000_6B	0	ISO 18000-6B Tag
ISO18000_6C_GEN1	1	Not used
ISO18000_6C_GEN2	2	ISO 18000-6C GEN2 Tag

### 2.3.3. BankType

Defines Memory Bank of the Tag

Flag	Value	Description
Reserve	0	Reserved memory
EPC	1	EPC memory
TID	2	TID memory
User	3	User memory

### 2.3.4. CommType


Defines the connection method between AT288N and the Host.

Flag	Value	Description
Bluetooth	0	Connection using Bluetooth
USB	1	Connection using USB

### 2.3.5. InventoryFormatType

Defines Format of Inventory result.

Flag	Value	Description
PC_EPC	0	PC + EPC
SERIAL_PC_EPC	1	Serial Number + PC + EPC

		AT288N API Reference Guide for Android Developers					
Android Developer Guide					Company	ATID Co., Ltd	
Name of Doc.		Writer	SW Team	Date	2023-06-12	Version	v0.4

<b>ONLY_EPC</b>	2	EPC
<b>SERIAL_EPC</b>	3	Serial Number + EPC

### 2.3.6. SelectionActionType


Defines the comparison method of mask value in Action method.

Flag	Value	Description
<b>Matching</b>	0	Works if Mask matches the Tag
<b>NonMatching</b>	1	Works if Mask does not match the Tag

### 2.3.7. StoredModeType

Defines operating method of Stored mode.

Flag	Value	Description
<b>NotStored</b>	0	Do not save data
<b>Stored</b>	1	Save data
<b>StoredSerialNo</b>	2	Not used

		AT288N API Reference Guide for Android Developers					
Android Developer Guide					Company	ATID Co., Ltd	
Name of Doc.		Writer	SW Team	Date	2023-06-12	Version	v0.4

## 2.4. Constant

Defines AT288N Device type.

플래그	값	설명
<b>Model Type</b>		
MT_NONE	0	Unknown model type
MT_AT288	1	Standard AT288N
MT_AT288Japan	2	Japanese AT288N
<b>Connection State</b>		
STATE_NONE	0	No event occurs
STATE_LISTEN	1	Waiting for the connection from outside.
STATE_CONNECTING	2	Connection in progress
STATE_CONNECTED	3	Connection completed
<b>Event Type</b>		
ET_NONE	0	No event occurs
ET_TIMEOUT	3	Timeout about command
ET_INVENTORY	5	Working on Inventory
ET_READMEMORY	6	Working on Read Memory
ET_WRITEMEMORY	7	Working on Write Memory
ET_LOCK	8	Working on Lock
ET_KILL	9	Working on Kill
ET_GETPROPERTIES	10	Attribute is returned
ET_COMMAND	11	Processing the command
<b>Tag Type</b>		
ISO18000_6B	0	ISO 18000-6B Tag
ISO18000_6C_GEN1	1	ISO 18000-6C GEN1 Tag
ISO18000_6C_GEN2	2	ISO 18000-6C GEN2 Tag
<b>Continue Mode Type</b>		
CT_MULTIPLE	0	Multiple Tag reading method
CT_SINGLE	1	Single Tag reading method
CT_FILTERING	2	It is a multi-Tag reading method but does not read the same Tag more than twice. .
<b>Memory Bank Type</b>		

MT_RESERVED	0	Reserved memory																																															
MT_EPC	1	EPC memory																																															
MT_TID	2	TID memory																																															
MT_USER	3	User memory																																															
<b>Inventory Format Type</b>																																																	
PC_EPC	0	PC + EPC																																															
SERIAL_PC_EPC	1	Serial Number + PC + EPC																																															
ONLY_EPC	2	EPC																																															
SERIAL_EPC	3	Serial Number + EPC																																															
<b>Property Codes</b>																																																	
PROPERTY_BUZZER	'b'	Buzzer mode value: 0(Disable), 1(Enable)																																															
PROPERTY_CONTINUE_MODE	'c'	When Inventory function is executes, it decides whether to read the Tag continuously or only once value: 0(Disable), 1(Enable)																																															
PROPERTY_ANTENNA_STATUS	'e'	Not used																																															
PROPERTY_GLOBAL_BAND	'f'	UHF Frequency band by country <table border="1"> <thead> <tr> <th colspan="2">value</th><th rowspan="2">Description</th></tr> <tr> <th>MA</th><th>MI</th></tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>Korea</td></tr> <tr><td>2</td><td>1</td><td>Europe</td></tr> <tr><td>3</td><td>2</td><td>USA</td></tr> <tr><td>4</td><td>3</td><td>China</td></tr> <tr><td>5</td><td>4</td><td>Taiwan</td></tr> <tr><td>6</td><td>5</td><td>Brazil</td></tr> <tr><td>7</td><td>6</td><td>Malaysia</td></tr> <tr><td>8</td><td>7</td><td>Asia</td></tr> <tr><td></td><td>8</td><td>Japan 1W</td></tr> <tr><td></td><td>9</td><td>Japan 250mW</td></tr> <tr><td></td><td>10</td><td>India</td></tr> <tr><td></td><td>11</td><td>Indonesia</td></tr> <tr><td></td><td>12</td><td>Japan 125mW</td></tr> <tr><td></td><td>13</td><td>Israel</td></tr> </tbody> </table>	value		Description	MA	MI	0	0	Korea	2	1	Europe	3	2	USA	4	3	China	5	4	Taiwan	6	5	Brazil	7	6	Malaysia	8	7	Asia		8	Japan 1W		9	Japan 250mW		10	India		11	Indonesia		12	Japan 125mW		13	Israel
value		Description																																															
MA	MI																																																
0	0	Korea																																															
2	1	Europe																																															
3	2	USA																																															
4	3	China																																															
5	4	Taiwan																																															
6	5	Brazil																																															
7	6	Malaysia																																															
8	7	Asia																																															
	8	Japan 1W																																															
	9	Japan 250mW																																															
	10	India																																															
	11	Indonesia																																															
	12	Japan 125mW																																															
	13	Israel																																															

			14	Australia
			15	New Zealand
			16	Philippines
			17	Singapore
			18	Thailand
			19	Uruguay
			20	Vietnam
			21	South Africa
PROPERTY_PACKET_OPTION	'i'	Not used		
PROPERTY_ANTENNA_SWITCHING_COUNT	'k'	Not used		
PROPERTY_ANTENNA_SWITCHING_TIME	'j'	The time Antenna is being activated when performing Inventory function (Unit: ms)		
PROPERTY_POWER_IDLE_TIME	'0'	The idle time of Antenna when performing Inventory function (Unit: ms)		
PROPERTY_POWER_GAIN	'p'	Antenna transmit power level Value: 0 ~ 19		
PROPERTY_START_Q_VALUE	'q'	Starting value of Q algorithm Value: 0 ~ 15		
PROPERTY_VERSION	'v'	RFID Module F/W version		
PROPERTY_ACCESS_PASSWORD	'w'	Password that needs for Access		
PROPERTY_READER_MODE	'x'	Not used		
PROPERTY_AUTO_COMMAND	'y'	Nor used		
PROPERTY_SELECTION_BANK	'9'	Tag memory bank that conducts Mask in the Masking Selection function		
PROPERTY_SELECTION_OFFSET	','	The address to start comparing Mask values in Masking Selection (Unit: bit)		
PROPERTY_SELECTION_ACTION	'8'	Define the behavior for Mask value in Masking Selection function		
PROPERTY_SELECTION_SESSION	's'	Inventory Session to compare Mask Value in Masking Selection		

		Value: 0(S0), 1(S1), 2(S2), 3(S3)
PROPERTY_MIN_Q_VALUE	'['	Minimum value of the Q algorithm Value: 0 ~ 15
PROPERTY_MAX_Q_VALUE	']'	Maximum value of the Q algorithm Value: 0 ~ 15
PROPERTY_LBT_CHANNEL	'B'	LBT Channel setting value
PROPERTY_PORT_ACTIVE	'e'	Not used
PROPERTY_AUTO_POWEROFF	'p'	The time until AT288N turns off (Unit: minute)
<b>Extended Property Codes</b>		
PROPERTY_EX_POWER_STATE	'1'	Power On/Off status of AT288N
PROPERTY_EX_POWER_GAIN	'2'	Antenna transmit power level Value: 0 ~ 19
PROPERTY_EX_TAG_TYPE	'3'	Tag Type that AT288N recognizes
PROPERTY_EX_COMMUNICATION_TYPE	'4'	Communication method between AT288N and the Host
PROPERTY_EX_CONTINUE_MODE	'5'	Inventory Type
PROPERTY_EX_STORED_MODE	'6'	Tag data storage mode
PROPERTY_EX_USE_SERIAL_NO	'8'	Whether includes Serial number in the Inventory data
PROPERTY_EX_BATTERY_STATE	'9'	AT288N Battery status Value: 0(high), 1(low)
PROPERTY_EX_GET_STORED_DATA	'a'	Transfer Tag list that saved in the AT288N completed to the Host
PROPERTY_EX_DELETE_STORED_DATA	'b'	Delete Tag list that saved in the AT288N completed
PROPERTY_EX_NATIONAL_CODE	'n'	Not used
PROPERTY_EX_SERIAL_NO	's'	AT288N Serial Number
PROPERTY_EX_VERSION	'v'	AT288N mainboard F/W version
PROPERTY_EX_INVENTORY_FORMAT	'8'	Inventory data type
<b>Action Codes</b>		
ACTION_INVENTORY_6B_MULTIPLE	'b'	Start Inventory about the 6B Type Tag as Multiple method

ACTION_INVENTORY_6B_SINGLE	'a'	Start Inventory about the 6B Type Tag as Single method
ACTION_INVENTORY_6C_MULTIPLE	'f'	Start Inventory about 6C Type Tag as Multiple method
ACTION_INVENTORY_6C_SINGLE	'e'	Start Inventory about 6C Type Tag as Single method
ACTION_INVENTORY_6C_SELECTION	'd'	Start Inventory about 6C Type Tag as Selection method
ACTION_INVENTORY_6C_VLC	'j'	Start Inventory about 6C Type Tag as VLC method
ACTION_READ_MEMORY	'r'	Start to read the data from Tag Memory
ACTION_WRITE_MEMORY	'w'	Start to write the data to Tag memory
ACTION_LOCK	'l'	Lock the Tag
ACTION_KILL	'k'	Discard the Tag
ACTION_STOP	'3'	Action stopped