Apple Notification Center Service (ANCS) Specification

v1.0



Contents

Introduction 4 Dependencies 4 Endianness and String Encoding 4 Terminology 4 **The Apple Notification Center Service** 5 Service Characteristics 5 Notification Source 5 Control Point and Data Source 7 Sessions 10 Attribute Fetching and Caching 10 Error Codes 10 Example Diagrams 12 **Appendix** 14 CategoryID Values 14 **EventID Values 15** EventFlags 15 CommandID Values 15 NotificationAttributeID Values 15 AppAttributeID Values 16

Document Revision History 17

Figures and Tables

The Apple I	Notification Center Service 5
Figure 2-1	The format of a GATT notification delivered through a Notification Source characteristic 6
Figure 2-2	The lifetime of an iOS notification 7
Figure 2-3	The format of a Get Notification Attribute command 8
Figure 2-4	The format of a response to a Get Notification Attributes command 8
Figure 2-5	The format of a Get App Attributes command 9
Figure 2-6	The format of a response to a Get App Attributes command 9
Figure 2-7	Service setup example 12
Figure 2-8	Notification attribute retrieval example 13
Appendix	14
Table 3-1	CategoryID values 14
Table 3-2	EventID values 15
Table 3-3	EventFlags 15
Table 3-4	CommandID values 15
Table 3-5	NotificationAttributeID values 15
Table 3-6	AppAttributeID values 16

Introduction

The purpose of the Apple Notification Center Service (ANCS) is to give Bluetooth accessories (that connect to iOS devices through a Bluetooth low-energy link) a simple and convenient way to access many kinds of notifications that are generated on iOS devices.

The ANCS is designed around three principles: simplicity, efficiency and scalability. As a result, accessories ranging from simple LEDs to powerful "companion" devices with large displays can find the service useful.

Dependencies

The ANCS has no dependencies, apart from the standard set of Generic Attribute Profile (GATT) sub-procedures. An accessory acting as a GATT client is free to access and use other services provided by the iOS device while using the ANCS.

Endianness and String Encoding

Unless specified otherwise, all numerical values transmitted through the ANCS shall be little endian.

Unless specified otherwise, all string values transmitted through the ANCS shall be composed of unicode characters encoded with UTF-8.

Terminology

The Apple Notification Center Service shall be referred to as the ANCS.

The publisher of the ANCS service (the iOS device) shall be referred to as the Notification Provider (NP).

Any client of the ANCS service (an accessory) shall be referred to as a *Notification Consumer (NC)*.

A notification displayed on an iOS device in the iOS Notification Center shall be referred to as an iOS notification.

A notification sent by a GATT characteristic as an asynchronous message shall be referred to as a *GATT* notification.

The Apple Notification Center Service

The Apple Notification Center Service is a primary service whose service UUID is 7905F431–B5CE–4E99–A40F–4B1E122D00D0.

Only one instance of the ANCS may be present on an NP. Due to the nature of iOS, the ANCS is not guaranteed to always be present. As a result, the NC should look for and subscribe to the Service Changed characteristic of the GATT service in order to monitor for the potential publishing and unpublishing of the ANCS at any time.

Service Characteristics

In its basic form, the ANCS exposes three characteristics:

- Notification Source: UUID 9FBF120D-6301-42D9-8C58-25E699A21DBD (notifiable)
- Control Point: UUID 69D1D8F3-45E1-49A8-9821-9BBDFDAAD9D9 (writeable with response)
- Data Source: UUID 22EAC6E9-24D6-4BB5-BE44-B36ACE7C7BFB (notifiable)

Support for the Notification Source characteristic is mandatory, whereas support for the Control Point characteristic and Data Source characteristic is optional.

Note: There may be more characteristics present in the ANCS than the three listed above. That said, an NC may ignore any characteristic it does not recognize.

Notification Source

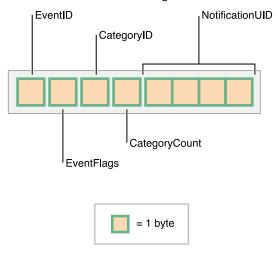
The Notification Source characteristic is the characteristic upon which an NC is informed of:

- the arrival of a new iOS notification on the NP
- the modification of an iOS notification on the NP
- the removal of an iOS notification on the NP

GATT notifications may be delivered as soon as the NC subscribes to the Notification Source characteristic. As a result, the NC should be in a state where it can properly accept and process these messages before subscribing to this characteristic.

The format of GATT notifications delivered through the Notification Source characteristic is shown in Figure 2-1.

Figure 2-1 The format of a GATT notification delivered through a Notification Source characteristic

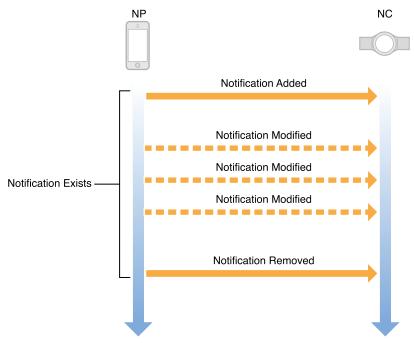


A GATT notification delivered through the Notification Source characteristic contains the following information:

- **EventID:** This field informs the accessory whether the given iOS notification was added, modified, or removed. The enumerated values for this field are defined in "EventID Values" (page 15).
- EventFlags: A bitmask whose set bits inform an NC of specificities with the iOS notification. For example, if an iOS notification is considered "important", the NC may want to display a more aggressive user interface (UI) to make sure the user is properly alerted. The enumerated bits for this field are defined in "EventFlags" (page 15).
- CategoryID: A numerical value providing a category in which the iOS notification can be classified. The
 NP will make a best effort to provide an accurate category for each iOS notification. The enumerated values
 for this field are defined in "CategoryID Values" (page 14).
- CategoryCount: The current number of active iOS notifications in the given category. For example, if two unread emails are sitting in a user's email inbox, and a new email is pushed to the user's iOS device, the value of CategoryCount is 3.
- **NotificationUID:** A 32-bit numerical value that is the unique identifier (UID) for the iOS notification. This value can be used as a handle in commands sent to the Control Point characteristic to retrieve more information about the iOS notification.

The lifetime of an iOS notification can be implicitly deduced through the sequence of Notification Source GATT notifications generated by the NP, as shown in Figure 2-2.

Figure 2-2 The lifetime of an iOS notification



Control Point and Data Source

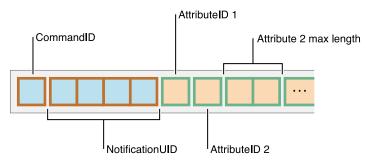
An NC may want to retrieve more information about an iOS notification, including its contents. The retrieval of these attributes is performed through the Control Point and Data Source characteristics.

An NC can issue a request to retrieve more information about an iOS notification by writing specific commands to the Control Point characteristic. If the write to the Control Point characteristic is successful, the NP will promptly respond to the request through a stream of GATT notifications on the Data Source characteristic.

Get Notification Attributes

The Get Notification Attributes command allows an NC to retrieve the attributes of a specific iOS notification. The format of a Get Notification Attribute command is shown in Figure 2-3.

Figure 2-3 The format of a Get Notification Attribute command

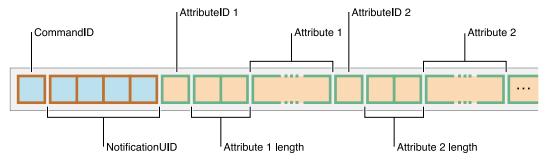


A Get Notification Attributes command contains the following information:

- CommandID: Should be set to 0 (CommandIDGetNotificationAttributes).
- **NotificationUID:** The 32-bit numerical value representing the UID of the iOS notification for which the client wants information.
- AttributeIDs: A list of attributes that the NC wants to retrieve. Some attributes may need to be followed by a 16-bit length parameter that specifies the maximum number of bytes of the attribute the NC wants to retrieve.

The format of a response to a Get Notification Attributes command is shown in Figure 2-4.

Figure 2-4 The format of a response to a Get Notification Attributes command



A response to a Get Notification Attributes command contains the following information:

- **CommandID:** Set to 0 (CommandIDGetNotificationAttributes).
- **NotificationUID:** The 32-bit numerical value that is the UID of the iOS notification the following attributes correspond to.

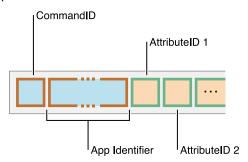
• AttributeList: A list of AttributeIDs/16-bit Length/Attribute tuples. An attribute is always a string whose length in bytes is provided in the tuple, but which is not NULL-terminated. If a requested attribute is empty or missing for the iOS notification, its length is set to 0.

If the response is larger than the negotiated GATT Maximum Transmission Unit (MTU), it is split into multiple fragments by the NP. The NC must recompose the response by splicing each fragment. The response is complete when the complete tuples for each requested attribute has been received.

Get App Attributes

The Get App Attributes command allows an NC to retrieve attributes of a specific app installed on the NP. The format of the Get App Attributes command is shown in Figure 2-5.

Figure 2-5 The format of a Get App Attributes command

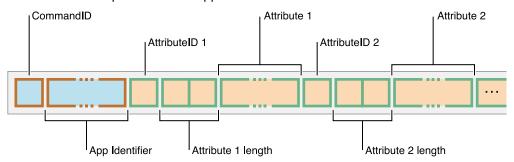


A Get App Attributes command contains the following information:

- **CommandID:** Should be set to 1 (CommandIDGetAppAttributes).
- Appldentifier: The string identifier of the app the client wants information about. This string must be NULL-terminated.
- AttributeIDs: A list of attributes the NC wants to retrieve.

The format of a response to a Get App Attributes command is shown in Figure 2-6.

Figure 2-6 The format of a response to a Get App Attributes command



A response to a Get App Attributes command contains the following information:

- CommandID: Set to 1 (CommandIDGetAppAttributes).
- **AppIdentifier:** The string identifier of the app the following attributes correspond to. This string is NULL-terminated.
- AttributeList: A list of AttributeIDs/16-bit Length/Attribute tuples. An attribute is always a string whose length in bytes is provided in the tuple, but which is not NULL-terminated. If a requested attribute is empty or missing for the app, its length is set to 0.

As with a response to a Get Notification Attributes command, if the response to a Get App Attributes command is larger than the negotiated GATT Maximum Transmission Unit (MTU), it is split into multiple fragments by the NP. The NC must recompose the response by splicing each fragment. The response is complete when the complete tuples for each requested attribute has been received.

Sessions

An ANCS session begins when an NC subscribes to the Notification Source characteristic on an NP and ends when the NC either unsubscribes from the same characteristic or disconnects from the NP. Because the ANCS is not designed to be a complete synchronization service, it does not keep track of state across sessions. As a result, all identifiers (such as NotificationUID and Appldentifier) and all data exchanged between an NC and an NP are valid only within a particular session.

When a particular session ends, the NC should remove any identifiers and data it gathered and stored during the session. When a new session begins, the NP does its best to inform the NC of any existing iOS notifications on the system. The NC can use this information to build a model to use for the remainder of the session.

Attribute Fetching and Caching

It is strongly recommended for an NC to fetch attributes only as needed and possibly in response to user actions. For example, if an NC chooses to display active iOS notifications in a simple list, and to show only details about a specific iOS notification when selected by the user, then the retrieval of this iOS notification's attributes can be triggered lazily.

During a session, it is strongly recommended that an NC build a cache of App Attributes for each app identifier it encounters. Building this cache allows the NC to avoid retrieving the same immutable App Attributes multiple times—saving time and preserving battery.

Error Codes

When writing to the Control Point characteristic, an NC may receive the following ANCS-specific error codes:

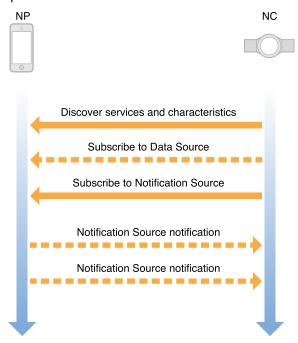
- **Unknown command** (0×A0): The commandID was not recognized by the NP.
- **Invalid command** (0×A1): The command was improperly formatted.
- **Invalid parameter** (0×A2): One of the parameters (for example, the NotificationUID) does not refer to an existing object on the NP.

If the NP replies with an error, it will not generate any GATT notification on the Data Source characteristic for the corresponding command.

Example Diagrams

The following two figures show examples of two common interactions between an NP and an NC. Figure 2-7 shows the typical sequence of commands and responses needed to set up the ANCS on an NC. Figure 2-8 (page 13) shows the typical sequence of commands and responses needed to get more information about an iOS notification in order to display it on an NC.

Figure 2-7 Service setup example



NΡ NC Incoming call notification added UID=3 Get notification attributes UID=3 Attributes=Title, Message, AppID Get notification attributes UID=3 Title="William T. Riker" Message="Incoming FaceTime" AppID="com.apple.facetime" com.apple.facetime? Yes Display Memory Get App attributes ID="com.apple.facetime" Attributes=DisplayName No Get App attributes ID="com.apple.facetime" DisplayName="FaceTime" Display

Figure 2-8 Notification attribute retrieval example

Appendix

The following tables list important values used in the ANCS.

CategoryID Values

Table 3-1 CategoryID values

CategoryIDOther	= 0,
CategoryIDIncomingCall	= 1,
CategoryIDMissedCall	= 2,
CategoryIDVoicemail	= 3,
CategoryIDSocial	= 4,
CategoryIDSchedule	= 5,
CategoryIDEmail	= 6,
CategoryIDNews	= 7,
CategoryIDHealthAndFitness	= 8,
CategoryIDBusinessAndFinance	= 9,
CategoryIDLocation	= 10,
CategoryIDEntertainment	= 11,
Reserved CategoryID values	= 12-255

EventID Values

Table 3-2 EventID values

EventIDNotificationAdded	= 0,
EventIDNotificationModified	= 1,
EventIDNotificationRemoved	= 2,
Reserved EventID values	= 3-255

EventFlags

Table 3-3 EventFlags

EventFlagSilent	= (1 << 0),
EventFlagImportant	= (1 << 1),
Reserved EventFlags	= (1 << 2)-(1 << 7)

CommandID Values

Table 3-4 CommandID values

CommandIDGetNotificationAttributes	= 0,
CommandIDGetAppAttributes	= 1,
Reserved CommandID values	= 2-255

NotificationAttributeID Values

 Table 3-5
 NotificationAttributeID values

NotificationAttributeIDAppIdentifier	= 0,
--------------------------------------	------

NotificationAttributeIDTitle	= 1, (Needs to be followed by a 2-bytes max length parameter)
NotificationAttributeIDSubtitle	= 2, (Needs to be followed by a 2-bytes max length parameter)
NotificationAttributeIDMessage	= 3, (Needs to be followed by a 2-bytes max length parameter)
NotificationAttributeIDMessageSize	= 4,
NotificationAttributeIDDate	= 5,
Reserved NotificationAttributeID values	= 6-255

Note: The format of the NotificationAttributeIDMessageSize constant is a string that represents the integral value of the message size. The format of the NotificationAttributeIDDate constant is a string that uses the Unicode Technical Standard (UTS) #35 date format pattern yyyyMMdd'T'HHmmSS. The format of all the other constants in Table 3-5 are UTF-8 strings.

AppAttributeID Values

Table 3-6 AppAttributeID values

AppAttributeIDDisplayName	= 0,
Reserved AppAttributeID values	= 1–255

Document Revision History

This table describes the changes to Apple Notification Center Service (ANCS) Specification.

Date	Notes
2013-10-22	Added information about an ANCS session.
	Added the "Sessions" (page 10) section, which describes an ANCS session.
2013-09-18	Changes made for version v1.0:
	Added format information for the NotificationAttributeID constants in Table 3-5 (page 15).
2013-11-21	Made format and value changes for Version v0.2.
	Changed the format of GATT notifications delivered through the Notification Source characteristic, as shown in Figure 2-1 (page 6) and the bulleted-list that follows.
	Updated the CategoryID values in Table 3-1 (page 14).
	Added reserved values for the CategoryID, EventID, EventFlags, CommandID, NotificationAttributeID, and AppAttributeID attributes in "Appendix" (page 14).
	New document that describes the specification for building Bluetooth accessories that communicate with Apple Notification Center Service (ANCS).

Apple Inc. Copyright © 2013 Apple Inc. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Apple-labeled computers.

Apple Inc. 1 Infinite Loop Cupertino, CA 95014 408-996-1010

Apple and the Apple logo are trademarks of Apple Inc., registered in the U.S. and other countries.

iOS is a trademark or registered trademark of Cisco in the U.S. and other countries and is used under license.

Even though Apple has reviewed this document, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS PROVIDED (AS IS," AND YOU, THE READER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.

IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.

THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.