



Application Note: Melody BLE db gen - Rev C

Introduction

This document explains how to use gatt_db_api library (db_gen.dll) which can be used to generate a BLE database for Melody 6.



Application Note: Melody BLE db gen - Rev C

Contents

Introduction	1
1. Initialization of the database	3
2. Adding services and characteristics	3
3. Generation of the BLE database	3
4. Example	4



Application Note: Melody BLE db gen - Rev C

Initialization of the database

The function `init_db` must be called first to initialize the BLE database.

Adding services and characteristics

1. Primary service declaration:

The function `add_primary_service` can be used to add a service with its UUID (16, 32 or 128 bits).

The number of services is limited to 20.

2. Characteristic declaration:

The function `add_characteristic` can be used to add a characteristic with its UUID (16, 32 or 128 bits) to the last service added.

The characteristic's properties (read, write, notify, indicate...) are defined in `gatt_db_api.h`.

If the value of the characteristic is set, `BLE_READ` notifications won't be received and Melody will automatically reply to a read request with the value.

The client characteristic configuration can also be added with this function.

The number of characteristics per services is limited to 40.

Generation of the BLE database

Once the services and characteristics have been added to the database, the functions `get_db_size` and `generate_db` can be used to retrieve an array of values representing the database.

Please note it is an array of bytes but Melody expects words. i.e. "01 23 AB CD" must be sent as "0123 ABCD" to Melody when using `BLE_SET_DB`.



Application Note: Melody BLE db gen - Rev C

Example

a) Creating a custom BLE database

The following code generates an array of values representing the database. It includes the GAP, GATT, HRS and BC Smart services:

```
#include "gatt_db_api.h"

uint8_t *database = NULL;
uint16_t size_database = 0;

uint8_t device_name="Melody 6";

/* Init */
init_db();

/* Add services and characteristics */
// add GAP Service
add_primary_service("1801");
add_characteristic("2A05", indicate, NULL, 0, TRUE);

//add GATT Service
add_primary_service("1800");
add_characteristic("2A00", read, strlen(device_name), device_name, FALSE);
add_characteristic("2A01", read, 0, NULL, FALSE);

// add Heart Rate Service
add_primary_service("180D");
add_characteristic("2A37", notify, 0, NULL, TRUE);
add_characteristic("2A38", read, 0, NULL, FALSE);

// add BC Smart Service
add_primary_service("bc2f4cc6aaef43519034d66268e328f0");
add_characteristic("06d1e5e779ad4a718faa373789f7d93c", write | notify, 0, NULL, TRUE);
add_characteristic("818ae3069c5b448db51a7add6a5d314d", write_without_response | notify, 0, NULL, TRUE);

/* Retrieve BLE database */
size_database = get_db_size();
database = malloc(sizeof(uint8_t) * size_database);
if(database != NULL)
{
    memset(database, 0, size_database);
    generate_db(database);
}
```



Application Note: Melody BLE db gen - Rev C

The size of the database is equal to 0x48 and here are the values that you should have in database:

```
{0x00, 0x02, 0x01, 0x18, 0x30, 0x05, 0x20, 0x03, 0x00, 0x05, 0x2A, 0x00, 0xDC, 0x00, 0x6C, 0x00, 0x00,
0x02, 0x00, 0x18, 0x30, 0x05, 0x02, 0x07, 0x00, 0x00, 0x2A, 0x00, 0xD0, 0x08, 0x4D, 0x65, 0x6C, 0x6F,
0x64, 0x79, 0x20, 0x36, 0x30, 0x05, 0x02, 0x09, 0x00, 0x01, 0x2A, 0x00, 0xDC, 0x00, 0x00, 0x02, 0x0D,
0x18, 0x30, 0x05, 0x10, 0x0C, 0x00, 0x37, 0x2A, 0x00, 0xDC, 0x00, 0x6C, 0x00, 0x30, 0x05, 0x02, 0x0F,
0x00, 0x38, 0x2A, 0x00, 0xDC, 0x00, 0x00, 0x10, 0xF0, 0x28, 0xE3, 0x68, 0x62, 0xD6, 0x34, 0x90, 0x51,
0x43, 0xEF, 0xAA, 0xC6, 0x4C, 0x2F, 0xBC, 0x30, 0x13, 0x18, 0x12, 0x00, 0x3C, 0xD9, 0xF7, 0x89, 0x37,
0x37, 0xAA, 0x8F, 0x71, 0x4A, 0xAD, 0x79, 0xE7, 0x5D, 0xD1, 0x06, 0x00, 0xCC, 0x00, 0x6C, 0x00, 0x30,
0x13, 0x14, 0x15, 0x00, 0x4D, 0x31, 0x5D, 0x6A, 0xDD, 0x7A, 0x1A, 0xB5, 0x8D, 0x44, 0x5B, 0x9C, 0x06,
0xE3, 0x8A, 0x81, 0x00, 0xCC, 0x00, 0x6C, 0x00 }
```

b) Setting the database in Melody 6

You can set the new database using the command BLE_SET_DB. Note that the input buffer has a limited size so it is preferable to send it into small chunks of data (inferior to 150 characters):

```
BLE_SET_DB 48<CR>
PENDING
0002 0118 3005 2003 0005 2A00 DC00 6C00 0002 0018<CR>
PENDING
3005 0207 0000 2A00 D008 4D65 6C6F 6479 2036 3005<CR>
PENDING
0209 0001 2A00 DC00 0002 0D18 3005 100C 0037 2A00<CR>
PENDING
DC00 6C00 3005 020F 0038 2A00 DC00 0010 F028 E368<CR>
PENDING
62D6 3490 5143 EFAA C64C 2FBC 3013 1812 003C D9F7<CR>
PENDING
8937 37AA 8F71 4AAD 79E7 5DD1 0600 CC00 6C00 3013<CR>
PENDING
1415 004D 315D 6ADD 7A1A B58D 445B 9C06 E38A 8100<CR>
PENDING
CC00 6C00
OK
```



Application Note: Melody BLE db gen - Rev C

c) Understanding the database and using its handles

Here is a more readable representation of this database:

```
[1] 0002 0118
[2] 3005 2003 0005 2A00
[3] DC00
[4] 6C00
[5] 0002 0018
[6] 3005 0207 0000 2A00
[7] D008 4D65 6C6F 6479 2036
[8] 3005 0209 0001 2A00
[9] DC00
[A] 0002 0D18
[B] 3005 100C 0037 2A00
[C] DC00
[D] 6C00
[E] 3005 020F 0038 2A00
[F] DC00
[10] 0010 F028 E368 62D6 3490 5143 EFAA C64C 2FBC
[11] 3013 1812 003C D9F7 8937 37AA 8F71 4AAD 79E7 E5D1 0600
[12] CC00
[13] 6C00
[14] 3013 1415 004D 315D 6ADD 7A1A B58D 445B 9C06 E38A 8100
[15] CC00
[16] 6C00
```

Note that for each line, the first byte indicates the type of data and the second indicates the length of the following data.

[1]	* GAP service (0x1801)
[2]	• Service Changed characteristic (0x2A05)
[3]	characteristic value
[4]	client characteristic configuration
[5]	* GATT service (0x1800)
[6]	• Device Name characteristic (0x2A00)
[7]	characteristic value ("Melody 6")
[8]	• Appearance characteristic (0x2A01)
[9]	characteristic value
[A]	* Heart Rate service (0x180D)
[B]	• Heart Rate Measurement characteristic (0x2A37)
[C]	characteristic value
[D]	client characteristic configuration
[E]	• Body Sensor Location characteristic (0x2A38)
[F]	characteristic value
[10]	* BC Smart service (0xBC2F4CC6AAEF43519034D66268E328F0)
[11]	• BC Smart Data characteristic (0x06D1E5E779AD4A718FAA373789F7D93C)
[12]	characteristic value
[13]	client characteristic configuration
[14]	• BC Smart Command characteristic (0x818AE3069C5B448DB51A7ADD6A5D314D)
[15]	characteristic value
[16]	client characteristic configuration



Application Note: Melody BLE db gen - Rev C

The line number in that case is equivalent to the handle values used by Melody.

So in order to notify a new value of the Heart Rate Measurement characteristic, you would have to use the handle corresponding to its value, 0x0C:

```
// starts advertising
ADVERTISING ON
OK

// remote device connects
OPEN_OK 14 BLE

// remote device enable notifications on Heart Rate Measurement characteristic
BLE_WRITE 14 000D 2 0100

// notify a new value for the Heart Rate Measurement characteristic
BLE_NOTIFICATION 14 C 2
PENDING
{04}{00}1OK
```

¹ hexadecimal values