No.	Blueinno BLE 함수 및 기타 정의			2015년 10월 09일
	#include<RFduinoBLE.h> ; // 블루투스 함수 사용을 위한 헤더파일 추가			
1	**RFduinoBLE.begin(); // BLE Stack & Advertising 시작**			
2	MAC : D6:C1:..... // chip maker에 의한 자동 발급되는 고유 ID			
3	RSSI : -43			
4	Scan Record			
5	Tx Power : 4			
6	Advertisement Data : 73 6B 65 ... "sketch" // RFduinoBLE.cpp 파일내에서 정보 변경 가능함 서비스명으로 활용가능함 데이터 패키지=rgb +Data = 31Byte			
7	void RFduinoBLE_onAdvertisement() { }; // 모든 BLE & iBeacon 송신시에 실행		void RfduinoBLE_onConnect() { }	스마트폰과 연결이 되었을때 실행
8			void RfduinoBLE_Disconnect() { }	스마트폰과 연결이 끊어졌을때 실행
9	**iBeacon** (페어링없이 iBeacon advertising)		**BLE** (페어링 후 BLE advertisement, 데이터 송수신)	
10	**RFduinoBLE.iBeacon = true**	iBeacon advertising 활성화	RFduinoBLE.advertisementData = "ledbtn";	페어링 후 서비스명을 알림
11	**iBeacon - 고정 정보만 변경**		RFduinoBLE_onRSSI(int rssi)	페어링 후 스마트폰의 수신신호 세기
12	**unit8_t uuid[16]={0xE2,..... }**	UUID 설정	RFduinoBLE.send(); RFduinoBLE.send(const char* data, int len)	BLE를 통해 스마트폰으로 문자 데이터 전송
13	memcpy(RFduinoBLE.iBeaconUUID,uuid, sizeof(RFduinoBLE.iBeaconUUID));	사용자 iBeacon 정보 갱신	RFduinoBLE.sendByte()	Byte 데이터 전송
14	RFduinoBLE.iBeaconMajor = 1234;	Major 설정	RFduinoBLE.sendInt()	INT형 데이터 전송
15	RFduinoBLE.iBeaconMinor = 5678;	Minor 설정	RFduinoBLE.sendFloat()	Float형 데이터 전송
16	RFduinoBLE.iBeaconMeasurePower=0xC5;	1미터에서 측정된 iBeacon 출력 default = 0xC5 = -58dBm (0xD5=-42dBm, 0xB5=-74dBm)	RFduinoBLE.radioAcitve()	Radio가 액티브인 상태에서 실행
17	int advertisementInterval;	default - 80ms	void RfduinoBLE_Receive() { }	스마트폰으로 받은 데이터 리턴
18	int txPowerLevel;	default - +4dBm		
19	**센서 비콘이나 사용자 비콘 만들기**			
20	unit8_t advdata[]={}	비콘 데이터의 정보 입력		
21	0x05, // length			
22	0x09, // complete local name type **(사용자 이름 갱신이 가능함)**		Device Name : Blueinno2 // RFduinoBLE.cpp 파일내에서 정보 변경 가능함	
23	0x41, // 'A'			
24	0x42, // 'B'			

25	0x43, // 'C'	
26	0x44, // 'D'	
27	0x02, // length	
28	0x01, // flags type	
29	0x06, // le general discovery mode | br edr not supported	
30	0x02, // length	
31	0x0A, // tx power level	
32	0x04, // +4dBm	
33	0x03, // length	
34	0x03, // 16 bit service uuid (complete)	
35	0x20, // uuid low	
36	0x22, // uuid hi	
37	RFduinoBLE_advdata = advdata;	센서비콘의 사용을 위한 설정
38	RFduinoBLE_advdata_len = sizefo(advdata);	
40	RFduinoBLE.txPowerLevel = -20; // 모든 BLE & iBeacon 송신시에 송신출력 설정 (-20 ~ +4dBm, step = 4dB)	
41	RFduinoBLE.advertisementInterval = 20; // 모든 BLE & iBeacon 송신시에 송신 ms 시간 설정 (20ms ~ 10.24s, step = 0.625ms)	
42	Rfduino_ULPDelay(); // 설정 시간동안 ultra low power delay 실행	
43	Rfduino_temprature(); // on-chip temperature sensor value	
44	Rfduino_pin_Wake(5, HIGH); // 설정된 핀이 HIGH일때 ULP에서 깨어남	
45	Rfduino_pinWoke(); //??	
46	Rfduino_resetPinWake(); // ??	
47	Rfduino_pinWakeCallback(6, HIGH, myPinCallback); // pin 6에 깨어나서 myPinCallback을 실행	
48	Rfduino_systemReset(); // 시스템 리셋	
49	Rfduino_systemOff(); // 시스템을 ULP로 전환하며, Wake pin에 의해서 깨어남	
50	**RFduinoBLE.end(); // BLE Stack & Advertising 종료**	
51	UART 통신	Serial.begin(baud) ; USB 포트 출력 , PC와 통신시 사용 Serial.begin(baud, RX pin, TX pin) ; GPIO 핀에 사용자 할당
52	I2C 통신	Wire.begin() ; 기본설정 SCL = 6번, SDA = 5번 Wire.beginOnPins(SCL pin, SDA pin) ; GPIO 핀에 사용자 할당
53	SPI 통신	기본설정 MISO = 3번, SCK =4번 , MOSI =5번 , SS/CS = 6번 사용자 할당시는 variant.h 파일을 수정해야 함