Studuino 라이브러리

함수 레퍼런스 편

본 자료는 Studuino 라이브러리 함수 레퍼런스입니다.
Studuino 라이브러리의 갱신에 따라 첨삭/수정 될 수 있습니다.

목차

1. Studuino 함수 라이브러리에 대하여

Studuino 함수 라이브러리는 Arduino IDE 로 Studuino 제어 프로그램을 기술하는데 사용하는 라이브러리 입니다. 각 부품(DC 모터나 서보모터 등) 레벨의 제어 함수를 준비하고 있기 때문에 사용자는 파트제어에 대한 상세한 처리를 의식하지 않아도 쉽게 부품을 제어하는 프로그램을 작성할 수 있습니다.

2. 함수

각 함수의 설명은 다음과 같은 포맷의 표로 되어 있습니다.

함수명	[함수명]			
인수	[타입]	[변수명]	[설정값]	[설명]
리턴	[타입]	[설명]		
비고				

아래에 각 함수에 대해 기술합니다.

2.1. 초기화 함수

부품을 연결한 Studuino 포트의 초기화에 사용하는 함수를 기록합니다.

함수명	SetDCMotorCalibration			
인수	byte[2]	rate	0~100	오프셋 값
리턴	없음			

DC 모터의 속도 조절이 불필요한 경우에는 본 함수를 사용하지 않아도 됩니다.
DC 모터의 속도를 조정하는 함수입니다. 인수 offsets[0]에는 M1, offsets[1]에는 M2 의 DC 모터의 속도 비율을 백분율로 설정해 주십시오.
【사용예】
// M1 에 연결되어 있는 DC 모터의 최대 속도를 80%까지 낮춤
// M2 에 연결되어 있는 DC 모터의 최대 속도를 100%로 설정
byte calib[] = { 80, 100 };
SetDCMotorCalibration(calib); // DC 모터 속도 교정을 설정				

함수명	SetServomotorCalibration			
인수	char[8]	offsets		오프셋 값
리턴	없음			
서보모터의 각도 조정이 불필요한 경우에는, 본 함수를 사용하지 않아도 됩니다. 서보모터의 각도를 조정하는 함수입니다. 인수 offsets[0]~[7]에 서보모터 D2~D12 에 대한 조정 각도를 설정해 주십시오.				

【사용예】
// D9~D12 에 연결된 서보모터의 조정 각도를 설정합니다.
byte calib[] = { 0, 0, 0, 0, -6, 0, 12, 3 }; // 조정각도 : D9(-6°), D10(0°), D11(12°), D12(3°)
SetServomotorCalibration(calib); // DC 모터 속도 교정을 설정

함수명	InitDCMotorPort			
인수	byte	connector	PORT_M1	접속 커넥터
			PORT_M2	
리턴	없음			

DC 모터 포트를 초기화하는 함수입니다. DC 모터를 제어하기 전에 본 함수에서 사용하는 포트를 초기화 해 주십시오.
【사용예】
// Move 함수 / DCMotor 함수 / DCMotorPower 함수 / DCMotorControl 함수를 실행하기 전에 사용하고 있는 포트를 본 함수에서 초기화하십시오.
InitDCMotorPort(PORT_M1); // DC 모터를 연결한 M1 포트를 초기화

함수명	InitServomotorPort			
인수	byte	connector	* 1 참조	접속 커넥터
리턴	없음			

서보모터 포트를 초기화하는 함수입니다. 서보모터를 제어하기 전에 본 함수에서 사용하는 포트를 초기화 해 주십시오.
【사용예】
// Servomotor 함수 /SyncServomotors 함수 / AsyncServomotors 함수를 실행하기 전에 사용하고 있는 포트를 본 함수에서 초기화하십시오.
InitServomotorPort(PORT_D2); //서보모터를 연결한 D2 포트를 초기화

함수명	InitServomotorPortForLED			
인수	byte	connector	PORT_D9	접속 커넥터
			PORT_D10	
			PORT_D11	
리턴	없음			

서보모터 포트의 D9, D10, D11 을 LED 용으로 초기화하는 함수입니다. D9, D10, D11 을 사용해 LED 의 밝기를 제어하기 전에 본 함수에서 사용하는 포트를 초기화해 주십시오.
【사용예】

// Gradation 함수를 실행하기 전에 사용하고 있는 포트를 본 함수에서 초기화하십시오.
InitServomotorPortForLED (PORT_D9); // LED 를 연결한 D9 포트를 초기화
Gradation(PORT_D9, 128);

함수명	InitSensorPort			
인수	byte	connector	* 4 참조	접속 커넥터
	byte	pid	* 5 참조	접속 파트
리턴	없음			

센서/LED/버저 포트를 초기화하는 함수입니다. 센서/LED/버저 포트를 제어하기 전에 본 함수에서 사용하는 포트를 초기화해 주십시오.
【사용예】
// Buzzer 함수 / BuzzerControl 함수 / Melody 함수 / LED 함수 / Get*함수를 실행하기 전에 사용하고 있는 포트를 본 함수에서 초기화하십시오.
InitSensorPort(PORT_A0, PIDLED); // LED 를 연결한 A0 포트를 초기화				

2.2. DC 모터 제어 함수

DC 모터를 제어하는 함수에 대해 기술합니다.

함수명	Move			
인수	byte	direct	FORWARD	전진
			BACKWARD	후퇴
			FORWARD_RIGHT	우회전(앞)
			FORWARD_LEFT	좌회전(앞)
			BACKWARD_RIGHT	우회전(뒤)
			BACKWARD_LEFT	좌회전(뒤)
			CLOCKWISE	우회전
			COUNTERCLOCKWISE	좌회전
	byte	pace	0~255	속도 (단계)
	ulong	duration	0~2^32-1	시간(msec)
	byte	brake	BRAKE	급정지 있음
			COAST	급정지 없음
리턴	없음			
DC 모터를 2 개 사용해 자동차를 만들었을 때 자동차의 이동을 실행하는 함수입니다. 본 함수의 제어는, DC 모터와 기판의 설치 방법에 의존합니다. **부록 A.Studuino 기판과 DC 모터의 연결을 참고해 자동차를 만들어 주십시오.** 【사용예】				

Move (FORWARD, 10, 1000, BRAKE); //속도 10 으로 1 초간 전진하고 정지합니다.

함수명	DCMotor			
인수	byte	connector	PORT_M1	접속 커넥터
			PORT_M2	
	byte	rotation	NORMAL	우회전
			REVERSE	좌회전
	byte	pace	0~255	속도 (단계)
	ulong	duration	0~2^32-1	시간(msec)
	byte	brake	BRAKE	급정지 있음
			COAST	급정지 없음
리턴	없음			

DC 모터를 제어하는 함수입니다.

【사용예】

// M1 에 연결되어 있는 DC 모터를 속도 10 으로 1 초간 회전, 정지합니다.

DCMotor (PORT_M1, NORMAL, 10, 1000, BRAKE);

함수명	DCMotorPower			
인수	byte	connector	PORT_M1	접속 커넥터
			PORT_M2	
	byte	pace	0~255	속도 (단계)
리턴	없음			

DC 모터의 속도를 제어하는 함수입니다.

【사용예】

// M1 에 연결되어 있는 DC 모터를 속도 10 으로 1 초간 회전하고, 그 후 100 에서 1 초간 회전한 다음 정지합니다.

DCMotorPower(PORT_M1, 10); // M1 에 연결한 DC 모터의 속도를 설정

DCMotorControl(PORT_M1, CLOCKWISE); // M1 에 연결한 DC 모터를 시계방향회전으로 움직임

Timer(1000); // 1 초 카운트

DCMotorPower(PORT_M1, 100); // M1 의 DC 모터의 속도를 변경합니다.

Timer(1000); // 1 초 카운트

DCMotorControl (PORT_M1, BRAKE); // M1 에 연결된 DC 모터를 멈춤.

함수명	DCMotorControl			
인수	byte	connector	PORT_M1	접속 커넥터
			PORT_M2	
	byte	rotation	NORMAL	우회전
			REVERSE	좌회전
			BRAKE	급정지
			COAST	감속 후 정지
리턴	없음			

DC 모터의 회전을 제어하는 함수입니다.

【사용예】

```
// M1 에 연결되어 있는 DC 모터를 속도 10 으로 1 초간 돌려 정지합니다.
DCMotorPower(PORT_M1, 10); // M1 에 연결한 DC 모터의 속도를 설정
DCMotorControl(PORT_M1, CLOCKWISE); // M1 에 연결한 DC 모터를 시계방향회전으로 움직임
Timer(1000); // 1 초 카운트
DCMotorControl (PORT_M1, BRAKE); // M1 에 연결한 DC 모터를 멈춤
```

2.3. 서보모터 제어 함수

서보모터를 제어하는 함수에 대하여 기술합니다.

함수명	Servomotor			
인수	byte	connector	* 1 참조	접속 커넥터
	byte	degree	0~180	서보모터 각도
리턴	없음			

1 개의 서보모터의 각도를 설정합니다.

서보모터가 지정각도로 이동함과 동시에 다음 처리가 실행됩니다.

【사용예】

```
// D2 에 연결된 서보모터에 90 도를 설정합니다.
Servomotor (PORT_D2, 90);
```

함수명	AsyncServomotors			
인수	byte[]	connectors	* 1 참조	접속 커넥터 배열
	byte[]	degrees	0~180	각 서보모터의 각도
	byte	number	1~8	서보모터의 수
리턴	없음			

복수의 서보모터 각도를 설정합니다. 모든 서보모터가 지정 각도로 이동함과 동시에 다음 처리가 실행됩니다.

【사용예】

```
// D2, D9, D10 에 연결된 서보모터에 각각 90, 180, 45 도를 설정합니다.
byte myConectors[] = { PORT_D2, PORT_D9, PORT_D10 };
byte myDegrees[] = {90, 180, 45};
ASyncServomotor (myConectors, myDegrees, 3);
```

함수명	SyncServomotors			
인수	byte[]	connector	* 1 참조	접속 커넥터 배열
	byte[]	degree	0~180	각 서보모터의 각도
	byte	number	1~8	서보모터의 수
	byte	time	0~255	1 도 당 이동 시간(ms)
리턴	없음			

복수의 서보모터의 각도를 설정합니다. 모든 서보모터가 지정 각도에 도달할 때까지 다른 처리는 실행되지 않습니다. 인수 time 값을 높여 서보모터의 이동 속도를 늦출 수 있습니다. 단, 서보모터의 1 도당 가장 빠른 이동 시간이 3ms 를 위해 time 인수에 3 미만을 설정해도 서보모터의 이동 속도는 바뀌지 않습니다(최고 속도로 이동합니다).

【사용예】

```
// D2, D9, D10 에 연결된 서보모터에 각각 90, 180, 45 도를 설정합니다.
byte myConectors[] = { PORT_D2, PORT_D9, PORT_D10 };
byte myDegrees[] = { 90, 180, 45};
SyncServomotor (myConectors, myDegrees, 3, 5);
```

2.4. 버저 제어 함수

버저를 제어하는 함수에 대하여 기술합니다.

함수명	Buzzer			
인수	byte	connector	* 3 참조	접속 커넥터
	word	pitch	* 6 참조	음 높이
	ulong	duration	0~2^32-1	출력 시간(msec)
리턴	없음			

버저에서 지정된 높이의 소리를 지정된 시간 동안 출력합니다.

【사용예】

```
Buzzer (PORT_A0, BZR_C4, 1000); // A0 에 연결된 버저에서 "도" 를 1 초간
출력합니다
```

함수명	BuzzerControl			
인수	byte	connector	* 3 참조	접속 커넥터
	boolean	onoff	ON	소리를 출력
			OFF	소리 정지
	byte	pitch	* 6 참조	음 높이
리턴	없음			

인수 onoff 에 ON 을 지정한 경우, 버저에서 지정된 높이(pitch)의 음을 출력합니다. OFF 를 지정한 경우 출력한 음을 정지합니다. (인수 pitch 의 값은 무시됩니다).

【사용예】

```
// A0 에 연결된 버저에서 "도"를 1 초간 출력합니다.
BuzzerControl(PORT_A0, ON, BZR_C4);
Timer(1000);
BuzzerControl(PORT_A0, OFF, 0);
```

함수명	Melody			
인수	byte	connector	* 3 참조	접속 커넥터
	word[]	pitches	* 6 참조	음 높이
	float[]	beats	0~	박
	byte	number	멜로디의 수(0~255)	음표의 수
	byte	tempo	TEMPO60	템포
			TEMPO90	
			TEMPO120	
			TEMPO150	
리턴	없음			

버저에서 지정된 멜로디를 출력합니다.

【사용예】

```
// A0 에 연결된 버저에서 "도레미파미레도" 를 출력합니다.
word myPitches[] = { BZR_C3, BZR_D3, BZR_E3, BZR_F3,
                     BZR_E3, BZR_D3, BZR_C3 };
float myBeats[] = { 1, 1, 1, 1, 1, 1, 1 };
byte num = 7;   // 요소 수
Melody (PORT_A0, myPitches, myBeats, num, TEMPO90);
```

긴 멜로디의 경우(출력하는 음의 수가 많은 경우), Studuino 기판의 SRAM 에 인수 scales 이나 notes 의 데이터가 들어가지 않을 수 있습니다. SRAM 용량을 넘는 데이터를 사용하고 싶은 경우에는 PROGMEM 키워드로 Flash 메모리에 데이터를 넣어 사용할 수

있습니다.

2.5. LED 제어 함수

LED 를 제어하는 함수에 대하여 기술합니다.

함수명	LED			
인수	byte	connector	* 3 참조	접속 커넥터
	boolean	onoff	ON	LED ON/OFF
			OFF	
리턴	없음			
LED 를 ON/OFF 합니다. 【사용예】 LED (PORT_A0, ON); // A0 에 연결된 LED 를 ON 합니다.				

함수명	Gradation			
인수	byte	connector	PORT_D9	접속 커넥터
			PORT_D10	
			PORT_D11	
	byte	ratio	0~255	밝기(수치가 클수록 밝아집니다.)
리턴	없음			
포트 D9, D10, D11 에 연결한 경우 LED 의 밝기를 조절할 수 있습니다. 【사용예】 Gradation (PORT_D9, 128); // D9 에 연결된 LED 의 밝기를 지정합니다.				

2.6. 입력함수

Studuino 의 푸시버튼 또는 센서를 제어하는 함수에 대하여 기술합니다.

함수명	GetPushSwitchValue			
인수	byte	connector	* 2 참조	커넥터
리턴	byte	0: 푸시, 1: 릴리즈		

푸시버튼의 상태를 취득합니다.
【사용예】
// A0 에 연결한 푸시버튼의 수치를 얻음
byte val = GetPushSwitchValue (PORT_A0);				

함수명	GetTouchSensorValue			
인수	byte	connector	* 3 참조	커넥터
리턴	byte	0: 푸시, 1: 릴리즈		

터치 센서 상태를 취득합니다.
【사용예】
// A0 에 연결한 터치 센서 수치를 얻음
byte val = GetTouchSensorValue (PORT_A0);				

함수명	GetLightSensorValue			
인수	byte	connector	* 4 참조	커넥터
리턴	int	0~1023		
광 센서 수치를 취득합니다. 【사용예】 int val = GetLightSensorValue (PORT_A0); // A0 에 연결한 광 센서 수치를 취득				

함수명	GetSoundSensorValue			
인수	byte	connector	* 4 참조	커넥터
리턴	int	0~1023		
사운드 센서 수치를 취득 【사용예】 int val = GetSoundSensorValue (PORT_A0); // A0 에 연결한 사운드 센서 수치를 얻음				

함수명	GetIRPhotoreflectorValue			
인수	byte	connector	* 4 참조	커넥터
리턴	int	0~1023		

적외선 반사 센서 수치를 취득합니다.
【사용예】
// A0 에 연결한 적외선 반사 센서 수치를 취득
int val = GetIRPhotoreflectorValue (PORT_A0);				

함수명	GetAccelerometerValue			
인수	byte	axis	X_AXIS	측정하는 가속도의 방향
			Y_AXIS	
			Z_AXIS	

리턴	int	0~1023
가속도 센서 수치를 취득합니다. 가속도 센서는 I2C 를 사용하므로 A4, A5 만 대응합니다. 【사용예】 int val = GetAccelerometerValue (X_AXIS); //가속도 센서 X 축 방향의 기울기를 취득		

2.7. 타이머 함수

지정 시간 대기 처리하는 함수에 대하여 기술합니다.

함수명	Timer			
인수	ulong	time	0~2^32-1	시간(msec)
리턴	없음			
지정시간을 카운트합니다. 【사용예】 Timer (1000); // 1 초 카운트 합니다.				

* 1 서보모터의 포트 설정

설정값	포트
PORT_D2	D2
PORT_D4	D4
PORT_D7	D7
PORT_D8	D8
PORT_D9	D9
PORT_D10	D10
PORT_D11	D11
PORT_D12	D12

* 2 푸시 스위치의 포트의 설정값

설정값	ポート
PORT_A0	A0
PORT_A1	A1
PORT_A2	A2
PORT_A3	A3

* 3 Digital 입출력계의 설정값

설정값	포트
PORT_A0	A0
PORT_A1	A1
PORT_A2	A2
PORT_A3	A3
PORT_A4	A4
PORT_A5	A5

* 4 Analog 입출력계의 설정값

설정값	포트
PORT_A0	A0
PORT_A1	A1
PORT_A2	A2
PORT_A3	A3
PORT_A4	A4
PORT_A5	A5
PORT_A6	A6
PORT_A7	A7

* 5 부품

설정값	파트
PIDOPEN	미접속
PIDLED	LED
PIDBUZZER	버저
PIDLIGHTSENSOR	광 센서
PIDSOUNDSENSOR	사운드 센서
PIDIRPHOTOREFLECTOR	적외선 반사 센서
PIDACCELEROMETER	가속도 센서
PIDTOUCHSENSOR	터치 센서
PIDPUSHSWITCH	푸시 스위치

* 6 음계의 설정값

설정값	음계	Hz
BZR_C3	도	130
BZR_CS3	도 #	139
BZR_D3	레	147
BZR_DS3	레 #	156
BZR_E3	미	165
BZR_F3	파	175
BZR_FS3	파 #	185
BZR_G3	솔	196
BZR_GS3	솔 #	208
BZR_A3	라	220
BZR_AS3	라 #	233
BZR_B3	시	247
BZR_C4	도	262
BZR_CS4	도 #	277
BZR_D4	레	294
BZR_DS4	레 #	311
BZR_E4	미	330
BZR_F4	파	349
BZR_FS4	파 #	370
BZR_G4	솔	392
BZR_GS4	솔 #	415
BZR_A4	라	440
BZR_AS4	라 #	466
BZR_B4	시	494

설정값	음계	Hz
BZR_C5	도	523
BZR_CS5	도 #	554
BZR_D5	레	587
BZR_DS5	레 #	622
BZR_E5	미	659
BZR_F5	파	698
BZR_FS5	파 #	740
BZR_G5	솔	784
BZR_GS5	솔 #	831
BZR_A5	라	880
BZR_AS5	라 #	932
BZR_B5	시	988
BZR_C6	도	1047
BZR_CS6	도 #	1109
BZR_D6	레	1175
BZR_DS6	레 #	1245
BZR_E6	미	1319
BZR_F6	파	1397
BZR_FS6	파 #	1480
BZR_G6	솔	1568
BZR_GS6	솔 #	1661
BZR_A6	라	1760
BZR_AS6	라 #	1865
BZR_B6	시	1976

설정값	음계	Hz
BZR_C7	도	2093
BZR_CS7	도 #	2217
BZR_D7	레	2349
BZR_DS7	레 #	2489
BZR_E7	미	2637
BZR_F7	파	2794
BZR_FS7	파 #	2960
BZR_G7	솔	3136
BZR_GS7	솔 #	3322
BZR_A7	라	3520
BZR_AS7	라 #	3729
BZR_B7	시	3951
BZR_C8	도	4186
BZR_S	무음	0

3. 프로그램

아래에 Studuino 라이브러리 함수를 사용해 Studuino 제어 프로그램을 작성할 때의 주의점을 기술합니다.

3.1. Arduino 언어

Arduino 언어에서는 setup 함수와 loop 함수를 사용자가 정의해야 합니다. setup 함수는 제일 처음 한번만 실행되는 함수입니다. loop 함수는 함수에 정의한 처리가 무한 반복됩니다.

```
//처음에 한번만 실행되는 함수. 주로 초기화에 사용.
void setup() {
   //초기화 함수를 사용하여 부품을 연결하고 있는 Studuino 포트를 초기화
}

// 무한 반복 실행되는 함수. 메인 처리
void loop() {
}
```

3.2. Studuino 오브젝트

Studuino 라이브러리를 사용할 경우 Studuino 의 기판을 이미지로 한 Studuino 오브젝트를 글로벌 변수를 하나 작성해야 합니다.

```
// Studuino 의 기판을 이미지. 기판이므로 1 개의 프로그램에 1 개만 작성
Studuino board;
```

3.3. 헤더 파일의 인클루드

Studuino 라이브러리 함수는, 서보모터, 가속도 센서를 사용합니다. Studuino 헤더 파일 이외에도 이들의 헤더 파일을 인클루드해야 합니다.

```
#include <Arduino.h >          // 기본 헤더 파일
#include <Servo.h>             // 서보모터용 헤더 파일
#include <Wire.h>              // 가속도 센서용 헤더 파일
#include <MMA8653.h>           // 가속도 센서용 헤더 파일
#include “Stduino.h”// Studuino 용 헤더 파일
```

3.4. 프로그램의 예

아래에 각 부품을 사용했을 때의 프로그램의 예를 기술합니다.

3.4.1. DC 모터를 제어하는 프로그램의 예

부록 A.Studuino 기판과 DC 모터의 연결을 참고로 Studuino 의 M1, M2 포트에 DC 모터를 연결하여 Arduino IDE 에서 아래의 프로그램을 Studuino 에 전송해 주십시오.

자동차가 1 초 전진한 뒤에 1 초 후퇴하고 M1 에 연결한 DC 모터가 1 초간 시계방향으로 회전합니다.

```
#include <Arduino.h >          // 기본 헤더 파일
#include <Servo.h>             // 서보모터용 헤더 파일
#include <Wire.h>              // 가속도 센서용 헤더 파일
#include <MMA8653.h>           // 가속도 센서용 헤더 파일
#include  "Studuino.h"         // Studuino 용 헤더 파일

// Studuino 의 기판을 이미지. 기판이므로 1 개의 프로그램에 1 개만 작성
Studuino board;

//처음에 한번만 실행되는 함수. 주로 초기화에 사용
void setup() {
   //초기화 함수를 사용해 부품을 연결하고 있는 Studuino 포트를 초기화
   board.InitDCMotorPort(PORT_M1);       // DC 모터가 연결되어 있는 M1 포트를 초기화
   board.InitDCMotorPort(PORT_M2);       // DC 모터가 연결되어 있는 M2 포트를 초기화
}

// 무한 반복 실행되는 함수. 메인 처리
void loop() {
   board.Move(FORWARD, 254, 1000, BRAKE);       // 1 초 전진한 후 정지
   board.Move(BACKWARD, 254, 1000, BRAKE);      // 1 초 후퇴한 후 정지

   // M1 에 연결된 DC 모터를 1 초간 시계방향으로 회전한 후 정지
   board.DCMotorPower(PORT_M1, 254);            // M1 의 DC 모터의 회전 속도를 설정
   board.DCMotorControl(PORT_M1, NORMAL);       // M1 의 DC 모터를 시계방향 회전 개시
   board.Timer(1000);                           // 1 초간 대기
   board.DCMotorControl(PORT_M1, BRAKE);                  // M1 의 DC 모터를 정지
```

```
    for (;;) {} //처리가 맨 처음으로 돌아가지 않도록 무한반복을 넣음
}
```

3.4.2. 서보모터를 제어하는 프로그램의 예

Studuino 의 D10, D11, D12 포트에 서보모터를 연결하여 Arduino IDE 에서 아래의 프로그램을 Studuino 에 전송해 주십시오. 모든 서보모터를 90 도로 초기화하고 약 3 초 후에 D10, D11, D12 포트의 서보모터가 각각 90 도, 180 도, 0 도까지 동시에 천천히 이동, 약 3 초 후에 D10 포트의 서보모터가 180 도까지 이동합니다.

```
#include <Arduino.h >          // 기본 헤더 파일
#include <Servo.h>             // 서보모터용 헤더 파일
#include <Wire.h>              // 가속도 센서용 헤더 파일
#include <MMA8653.h>           // 가속도 센서용 헤더 파일
#include   "Studuino.h"        // Studuino 용 헤더 파일

// Studuino 의 기판을 이미지. 기판이므로 1 개의 프로그램에 1 개만 작성
Studuino board;

//처음에 한번만 실행되는 함수. 주로 초기화에 사용.
void setup() {
    //초기화 함수를 사용하여 부품을 연결하고 있는 Studuino 포트를 초기화
    board.InitServomotorPort(PORT_D10);// 서보모터가 연결되어 있는 D10 포트를 초기화
    board.InitServomotorPort(PORT_D11);// 서보모터가 연결되어 있는 D11 포트를 초기화
    board.InitServomotorPort(PORT_D12);// 서보모터가 연결되어 있는 D12 포트를 초기화
}
// 무한 반복 실행되는 함수. 메인 처리.
void loop() {
    //서보모터의 각도를 90 도로 초기화
    byte connector[] = { PORT_D10, PORT_D11, PORT_D12 };
    byte degree[] = { 90, 90, 90 };
    byte number = sizeof(connector) / sizeof(byte); //각도를 설정하는 포트의 수

    board.AsyncServomotors(connector, degree, number);
    //필요하면, 서보모터가 설정 각도에 도달할 때까지 사용자가 delay 를 넣음
    board.Timer(1000);
```

```
    board.Timer(3000);              // 3 초 대기

    // D10, D11, D12 포트에 연결된 서보모터의 각도를 90, 180, 0 도로 설정
    degree[0] = 90;
    degree[1] = 180;
    degree[2] = 0;

    // 이 함수의 완료 후 서보모터는 목적 각도까지 도달
    board. SyncServomotors(connector, degree, number, 10);

    board.Timer(3000);              // 3 초 대기

    // D10 포트에 접속된 서보모터의 각도를 180 도로 설정
    board.Servomotor(PORT_D10, 180);
    // 필요하면, 서보모터가 설정 각도에 도달할 때까지 사용자가 delay 를 넣음
    board.Timer(1000);


    for (;;) {} // 처리가 맨 처음으로 돌아가지 않도록 무한반복을 넣음
}
```

3.4.3. 버저를 제어하는 프로그램의 예

Studuino 의 A0 포트에 버저를 연결하여 Arduino IDE 에서 아래의 프로그램을 Studuino 에 전송해 주십시오. 도 음을 1 초 출력하고, 솔 음을 1 초 출력한 뒤 “반짝반짝 작은 별”을 출력합니다.

```
#include <Arduino.h >          // 기본 헤더 파일
#include <Servo.h>             // 서보모터용 헤더 파일
#include <Wire.h>              // 가속도 센서용 헤더 파일
#include <MMA8653.h>           // 가속도 센서용 헤더 파일
#include  "Studuino.h"         // Studuino 용 헤더 파일

// Studuino 의 기판을 이미지. 기판이므로 1 개의 프로그램에 1 개만 작성
Studuino board;

//처음에 한번만 실행되는 함수. 주로 초기화에 사용.
```

```
void setup() {
    // 초기화 함수를 사용하여 부품을 연결하고 있는 Studuino 포트를 초기화
    board.InitSensorPort(PORT_A0, PIDBUZZER);// 버저가 연결된 A0 포트를 초기화
}

// 무한 반복 실행되는 함수. 메인 처리
void loop() {
    // 버저에서 1 초간 소리를 출력한 후 정지
    board.Buzzer(PORT_A0, BZR_C5, 1000);

    board.Timer(1000);              // 1 초 대기

    // 버저에서 1 초간 소리를 출력한 후 정지
    board.BuzzerControl(PORT_A0, ON, BZR_G5);
    board.Timer(1000);
    board.BuzzerControl(PORT_A0, OFF, 0);// OFF 의 경우, 마지막 인수는 무시됩니다

    board.Timer(1000);              // 1 초 대기

    // 버저에서 멜로디를 출력합니다.
    word myPitches[] = { BZR_C5, BZR_C5, BZR_G5, BZR_G5, BZR_A5, BZR_A5, BZR_G5 };
    byte number = sizeof(myScales) / sizeof(word); // 출력하는 음의 수
    float myBeats[] = { 1, 1, 1, 1, 1, 1, 1 }; // 박자수
    board.Melody (PORT_A0, myPitches, myBeats, number, TEMPO90);

    for (;;) {} // 처리가 맨 처음에 돌아가지 않도록 무한반복을 넣음
}
```

3.4.4. LED 를 제어하는 프로그램의 예

Studuino 의 A1, D9 포트에 LED 를 연결하여 Arduino IDE 에서 아래의 프로그램을 Studuino 에 전송해 주십시오.
A1 의 LED 가 3 번 점멸한 다음 D9 의 LED 를 천천히 점등합니다.

```
#include <Arduino.h >          // 기본 헤더 파일
#include <Servo.h>             // 서보모터용 헤더 파일
#include <Wire.h>              // 가속도 센서용 헤더 파일
```

```
#include <MMA8653.h>        // 가속도 센서용 헤더 파일
#include  "Studuino.h"        // Studuino 용 헤더 파일

// Studuino 의 기판을 이미지. 기판이므로 1 개의 프로그램에 1 개만 작성
Studuino board;

//처음에 한번만 실행되는 함수. 주로 초기화에 사용.
void setup() {
  //초기화 함수를 사용하여 부품을 연결하고 있는 Studuino 포트를 초기화
  board.InitSensorPort(PORT_A1, PIDLED);// LED 가 연결되어 있는 A0 포트를 초기화
  board.InitServomotorPortForLED(PORT_D9);// LED 가 연결되어 있는 D9 포트를 초기화
}

//무한 반복 실행되는 함수. 메인 처리
void loop() {
  // A1 에 연결된 LED 를 3 회 점멸
  for (int i = 0;i < 3;i++) {
    board.LED(PORT_A1, ON);  // A1 의 LED 를 점등
    board.Timer(1000);          // 1 초간 대기
    board.LED(PORT_A1, OFF); // A1 의 LED 를 소등
    board.Timer(1000);          // 1 초간 대기
  }

  // D9 에 연결된 LED 를 천천히 점등
  board.Gradation(PORT_D9, 0);
  for (int i = 0;i < 255;i++) {
    board.Gradation(PORT_D9, i);
    board.Timer(100);
  }

  for (;;) {} //처리가 맨 처음으로 돌아가지 않도록 무한반복을 넣음
}
```

3.4.5. 센서를 사용하는 프로그램의 예

Studuino 의 A1 에 터치 센서, A2 에 사운드 센서, A3 에 적외선 반사 센서, A4/A5 에 가속도 센서, A6 에 광 센서를 연결해 Arduino IDE 에서 아래의 프로그램을 Studuino 에 전송해 주십시오. 전송 후, Arduino IDE 메뉴의[도구]→[시리얼 모니터]를 선택해 시리얼 모니터를 표시해 주십시오. 시리얼 모니터에 각 센서 수치가 표시됩니다.

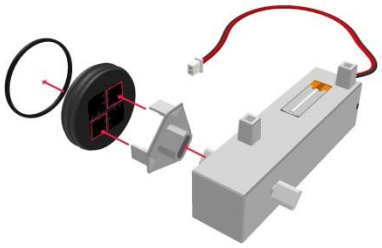
```
#include <Arduino.h >          // 기본 헤더 파일
#include <Servo.h>             // 서보모터용 헤더 파일
#include <Wire.h>              // 가속도 센서용 헤더 파일
#include <MMA8653.h>           // 가속도 센서용 헤더 파일
#include   "Studuino.h"        // Studuino 용 헤더 파일

// Studuino 의 기판을 이미지. 기판이므로 1 개의 프로그램에 1 개만 작성
Studuino board;

//처음에 한번만 실행되는 함수. 주로 초기화에 사용.
void setup() {
    // 초기화 함수를 사용하여 부품을 연결하고 있는 Studuino 포트를 초기화
    board.InitSensorPort(PORT_A0, PIDPUSHSWITCH);
    board.InitSensorPort(PORT_A1, PIDTOUCHSENSOR);
    board.InitSensorPort(PORT_A2, PIDSOUNDSENSOR);
    board.InitSensorPort(PORT_A3, PIDIRPHOTOREFLECTOR);
    board.InitSensorPort(PORT_A4, PIDACCELEROMETER);
    board.InitSensorPort(PORT_A5, PIDACCELEROMETER);
    board.InitSensorPort(PORT_A6, PIDLIGHTSENSOR);

    // 시리얼 출력 초기화
    Serial.begin(9600);
}

//무한 반복 실행되는 함수. 메인 처리.
void loop() {
    // 100msec 단위로 광 센서 수치를 취득해 시리얼 모니터에 출력함
    for (;;) {
      byte pVal = board.GetPushSwitchValue(PORT_A0);
```

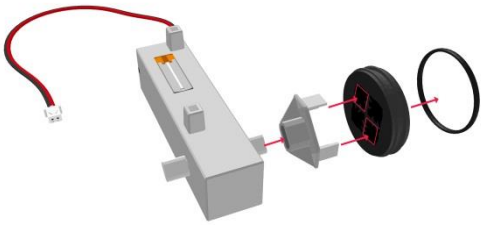
```
        byte tVal = board.GetTouchSensorValue(PORT_A1);
        int  sVal = board.GetSoundSensorValue(PORT_A2);
        int  iVal = board.GetIRPhotoreflectorValue(PORT_A3);
        int  xVal = board.GetAccelerometerValue(X_AXIS);
        int  yVal = board.GetAccelerometerValue(Y_AXIS);
        int  zVal = board.GetAccelerometerValue(Z_AXIS);
        int  lVal = board.GetLightSensorValue(PORT_A6);
        Serial.print("button:");        Serial.print(pVal);    Serial.print("₩t");
        Serial.print("touch:");         Serial.print(tVal);    Serial.print("₩t");
        Serial.print("sound:");         Serial.print(sVal);    Serial.print("₩t");
        Serial.print("ir:");  Serial.print(iVal);    Serial.print("₩t");
        Serial.print("x:");   Serial.print(xVal);    Serial.print("₩t");
        Serial.print("y:");   Serial.print(yVal);    Serial.print("₩t");
        Serial.print("z:");   Serial.print(zVal);    Serial.print("₩t");
        Serial.print("light:");         Serial.print(lVal);    Serial.println();
        board.Timer(100);
    }
}
```

부록 A. Studuino 기판과 DC 모터의 연결

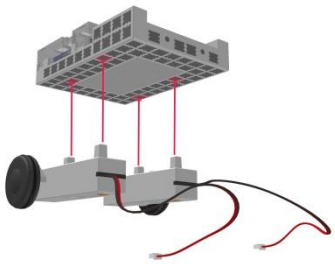
아래의 순서로 자동차를 조립합니다.

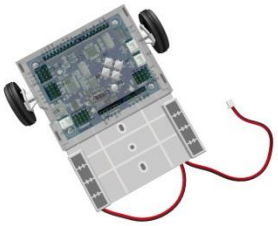
(1) DC 모터에 다음과 같이 타이어를 부착합니다.
※좌우 대칭으로 2 개 조립



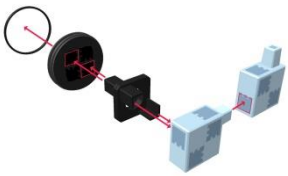


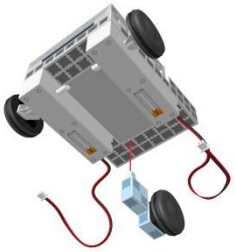
(2) DC 모터를 기판 베이스의 뒷면에 부착합니다.



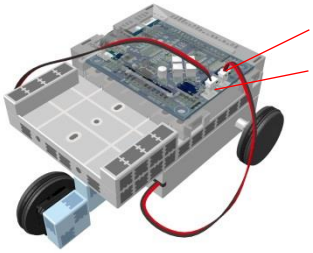


(3) 블록으로 뒷바퀴를 만듭니다.



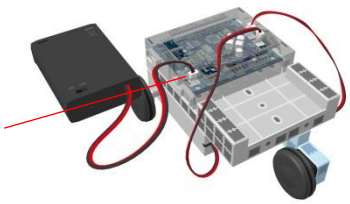


(4) DC 모터 및 배터리 박스를 각각 Studuino 기판에 연결합니다.



M1 : 오른쪽 DC 모터
M2 : 왼쪽 DC 모터

POWER : 전지 박스



(5) 전지 박스를 기판 베이스에 고정합니다.