

# 부록 C

---

## 프로그래밍 환경 설치 및 기초 동작 실습

### 내용

[C.1 프로그래밍 환경 설치](#)

[C.2 시리얼 모니터 문자 출력 실습](#)

[C.3 고찰](#)

- 본 자료는 Ardu-Ez의 제조사인 ㈜알앤유의 제공 자료로 구성되었습니다.

## C.1 프로그래밍 환경 설치

### 1) 스케치 다운로드 및 설치

아두이노 프로그래밍을 위해서는 Sketch IDE를 설치해야 한다. 이 프로그램은 다음과 같이 아두이노 공식 사이트(<http://arduino.cc>)에서 다운받을 수 있다.

EXE 수행파일로 다운 받아 설치

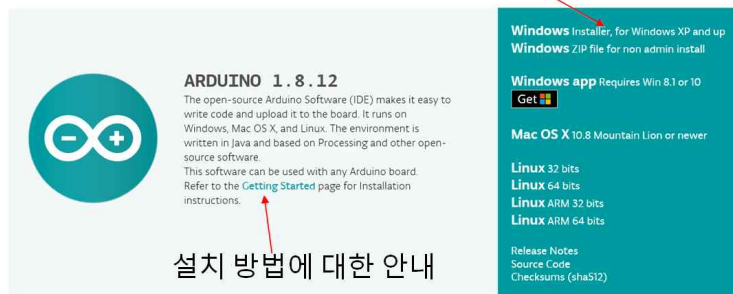


그림 C.1.1(a) 아두이노 공식 사이트 Sketch IDE 다운로드([링크](#))

Windows installer를 선택하면 EXE 파일을 다운로드 받아 설치할 수 있다<sup>1)</sup>. 입문단계에서는 꼭 필요한 내용은 아니지만 필요하다면 "Getting Started([링크](#))"를 방문하여 설치 방법에 대한 그림 C.1.1(b), (c)에 보인 바와 같이 자세한 안내를 볼 수 있다<sup>2)</sup>.

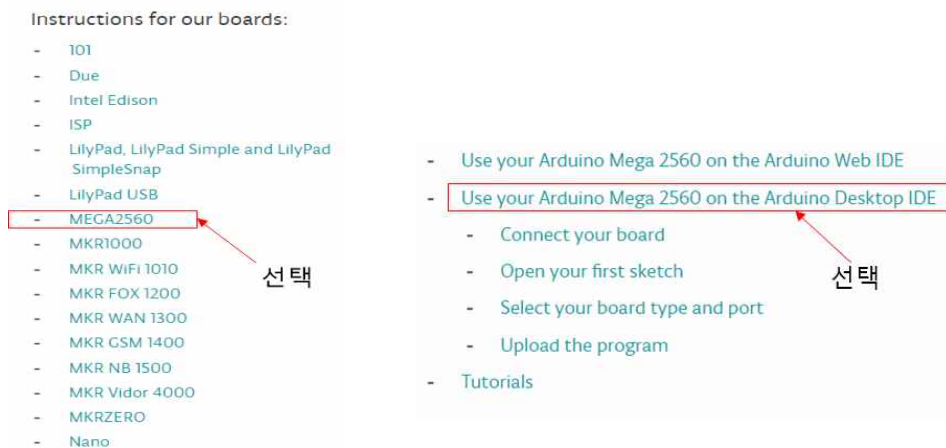


그림 C.1.1(b) 보드선택

그림 C.1.1(c) 데스크탑용 IDE 설치 과정

1) IDE 설치 만으로도 바로 USB 선을 연결하여 드라이버가 자동 설치되어 바로 실험을 수행할 수 있다.

2) 해당 페이지는 참고사항이며 설치에 꼭 필요한 요소는 아니다.

## 2) 드라이버 설치(윈도10 운영체제)

운영체제가 윈도 10이라면 별도의 설치과정이 필요치 않다. USB 선을 연결하는 것으로 자동 검색되어 사용할 수 있도록 드라이버가 자동 설치된다.

정상적으로 확인된 것을 확인하려면 다음과 같이 "제어판->시스템->장치관리자->포트(COM & LPT)"에 새로운 가상 COM 포트가 추가된 것을 확인하면 된다. 아두이노의 USB를 연결하지 않았을 때는 없었던 새로운 COM 포트(여기서는 COM8)가 나타나면 성공적으로 설치된 것이다. 현재 이 시스템은 가상 COM3 포트로 아두이노 메가 보드와 통신한다. COM 포트 번호는 상황에 따라 바뀔 수 있다.



그림 C.1.2 가상 COM 포트

## 3) 스케치 프로그램 내부의 설정

아두이노스케치 IDE 프로그램을 가동하면 그림 C.1.3과 같은 메뉴를 볼 수 있다.

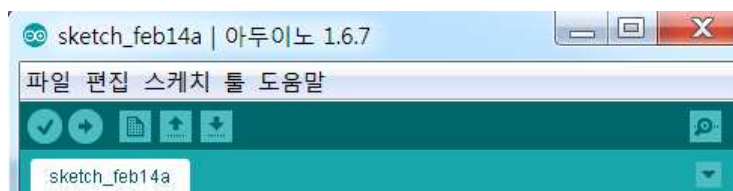


그림 C.1.3 스케치의 메뉴

### ① 보드 사양의 결정

먼저 스케치 프로그램에게 본 프로그램이 어떤 H/W 장치를 사용하는지 설정하는 작업이 필요하다. 그림 C.1.4에 보인 바와 같이 Ardu-Ez의 경우라면 스케치 프로그램의 메뉴에서 "[도구(혹은 툴)]-> [보드]-> [Arduino Mega 2560 or Mega ADK]"를

선택하여 설정 작업을 수행한다.

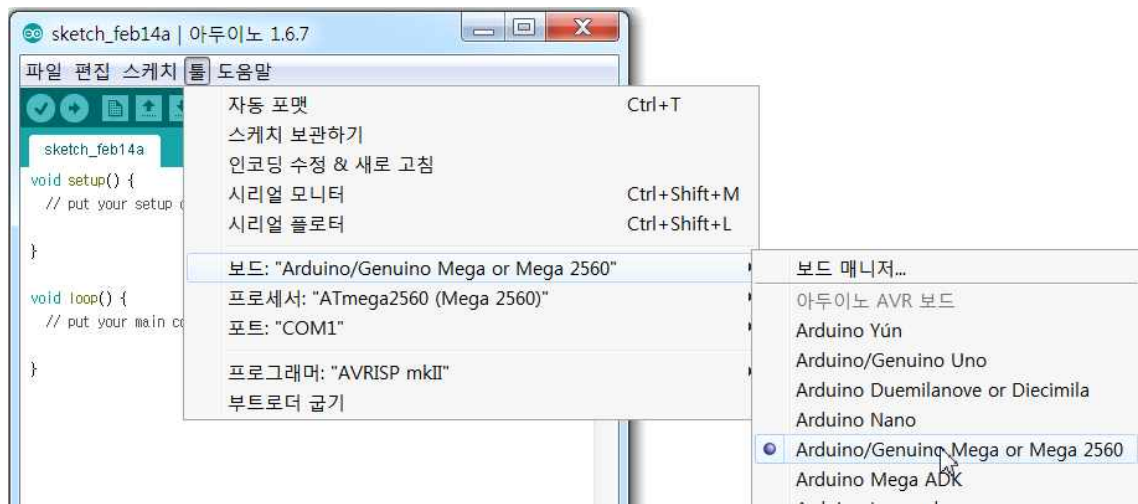


그림 C.1.4 보드 사양의 설정

## ② COM 포트 확인

과거 버전의 IDE에서는 수동으로 사용하는 COM 포트를 설정해야 했다. 최근 버전에서는 이같은 수고를 거치지 않아도 IDE 프로그램에서 COM 포트 번호를 자동으로 설정하는 듯하다. C.1.5의 경우는 본 실험 키트에서 사용하는 COM 포트 번호가 3번으로 설정되어 있음을 나타내고 있다.

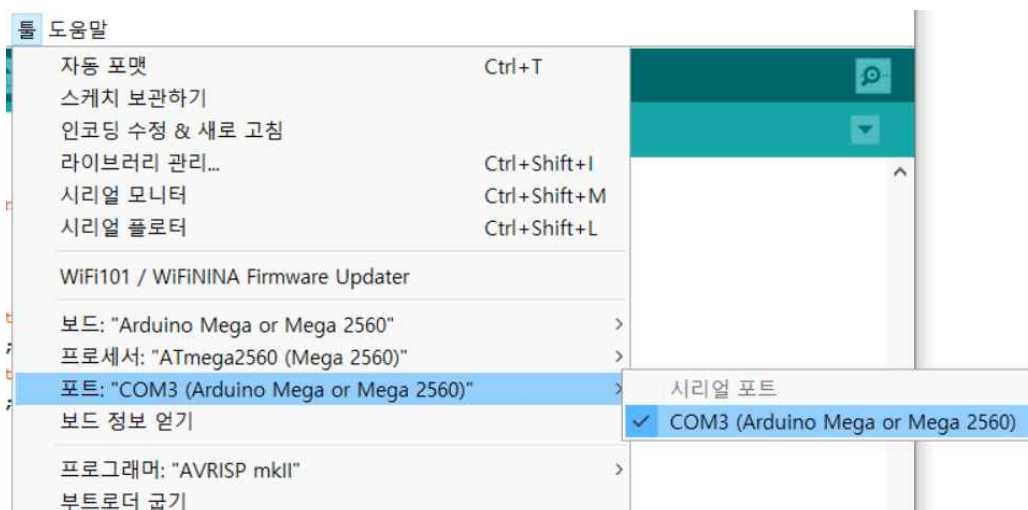


그림 C.1.5 직렬포트의 설정 상태

## C.2 시리얼 모니터 문자 출력 실습

본격적인 실습에 앞서 우선 간단히 보드가 제대로 작동하는 것인지 확인하기 위해 아두이노가 제공하는 시리얼 모니터(serial monitor)를 이용해 간단한 문자 출력 테스트를 통해 이를 점검해 보고자 한다.

### 1) 시리얼 모니터란?

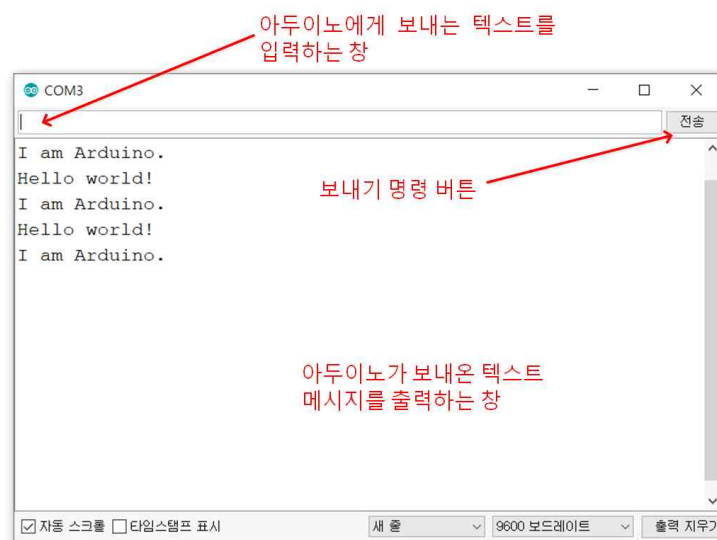


그림 C.2.1 시리얼 모니터의 수행 사례

시리얼 모니터(serial monitor)은 그림 C.2.1에 보인 바와 같이 아두이노와 PC간의 문자 데이터 통신 데이터 창구를 말한다. 그림에서는 아두이노 보드가 출력한 문자를 PC에 출력해 주고 있다. 필요하다면 이 창을 통해 PC의 사용자가 아두이노 보드에 텍스트 문자를 전송할 수 있다.

이 같은 직렬통신은 아두이노의 직렬통신 포트를 통해 USB 제어를 통해 USB 선로로 PC에게 전달된다. PC는 USB 통신 선로를 마치 전통 직렬통신장치(RS-232) COM port로 들어온 것으로 간주한다. 시리얼 모니터는 이런 문자 데이터 교신을 담당하는 IDE의 한 구성 요소이다.

### 2) 시리얼 모니터의 가동

시리얼 모니터 창을 가동하기 위해서는 그림 C.2.2(a)에 보인 바와 같이 [메뉴=> 툴=>시리얼 모니터]을 선택하거나 hot key(Control+Shift+M)를 입력하면 된다. 또


다른 방법으로는 그림 C.2.2(b)에 보인 바와 같이 메뉴 우측 상단의  아이콘을 클릭함으로써 가동할 수 있다.



그림 C.2.2(a) 시리얼 모니터의 창 열기(메뉴 클릭)



그림 C.2.2(b) 시리얼 모니터의 창 열기(아이콘 클릭)

### 3) 시리얼 모니터를 사용하는 예제 프로그램

아래 예제 프로그램을 복사하여 IDE의 편집창에 붙여넣기하여 예제 프로그램을 만든다. 프로그램은 딱 한번만 수행하는 `setup()`부와 전원을 off할 때까지 무한 반복 수행하는 `loop()`부로 구성된다.

아래 프로그램을 작성하고 나면 다음의 컴파일링과 업로딩을 거쳐 프로그램을 수행해야 한다. 만약 성공적으로 수행된다면 그림 C.2.1과 같이 시리얼 모니터 창에 "Hello world!"와 "I am Arduino."가 1초 간격으로 출력될 것이다. 이 메시지의 확인을 통해 보드 및 환경설정은 정상적으로 이루어졌다고 간접적으로 판단할 수 있다.

□ 예제 1 : 시리얼 모니터를 통해 스트링 문자를 출력한다.

hello.ino : 아두이노 표준 클래스 및 함수 – Serial, println()

=> 이후 설명에는 파일 이름이 blink.ino인 것으로 기술되었다.

```







01 // Hello world 메시지 스트링 출력
02 void setup(){
03     Serial.begin(9600);          // 9600bps의 전송속도를 사용한다.
04 }
05
06 void loop(){
07     Serial.println("Hello world!"); // 문자열을 시리얼 모니터로 출력
08     delay(1000);                  // 1000ms 동안 지연시간을 갖는다.
09     Serial.println("I am Arduino.");
10     delay(1000);                  // 끝나면 loop의 다음으로 돌아간다.
11 }

```

## 4) 프로그램 업로드 및 수행

□ 스케치 프로그램의 툴 바 아이콘 설명



	확인	코드 오류 확인 및 소스 파일 컴파일
	업로드	아두이저(Ardu-Ez) 보드에 수행코드 올리기 컴파일하지 않은 경우에는 컴파일한 후 올린다.
	새 파일	새로운 스케치 창 띄우기
	열기	저장된 소스 코드 가져오기
	저장	소스 코드 저장
	시리얼 모니터	시리얼 통신 창 띄우기

제공된 소스 프로그램을 아두이노 스케치 창에서 입력한 후 저장하고 다음 절차에 따라 ① 컴파일링(compiling)하고 ② AVR 프로세서에 업로딩(uploading)하고 ③ 실행(execution)한다.


① **컴파일(Compile)**: 작성한 소스코드를 아두이지(Ardu-Ez)에서 실행 가능한 파일로 만드는 과정을 컴파일<sup>3)</sup>이라고 한다. 컴파일은 그림 C.2.3처럼 메뉴 아래의 왼쪽 첫 번째 아이콘  (확인)을 클릭하여 수행한다.



그림 C.2.3 컴파일 메뉴 아이콘 선택하기

컴파일이 정상적으로 수행되면 그림 C.2.3처럼 "컴파일 완료"라는 문구가 출력되고, 생성된 실행 파일 크기도 표시가 된다.

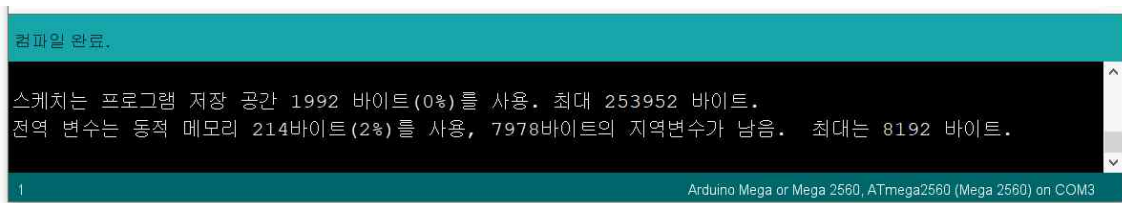


그림 C.2.3 컴파일 성공 메시지

3) 컴파일 과정은 C언어와 같은 상위언어(high level language)로 쓰인 소스 코드를 CPU가 수행할 수 있는 바이너리(binary) 형태의 기계어(machine language)로 바꾸는 과정을 말한다. 소스코드는 ASCII 텍스트 문장으로 이루어져 있는데 반해 기계어는 이를 CPU가 이해할 수 있는 0과 1로 이루어진 바이너리 코드로 되어 있다. 모든 프로그램은 기계어 형태로 구성되어 있어야 CPU가 수행 가능하다.



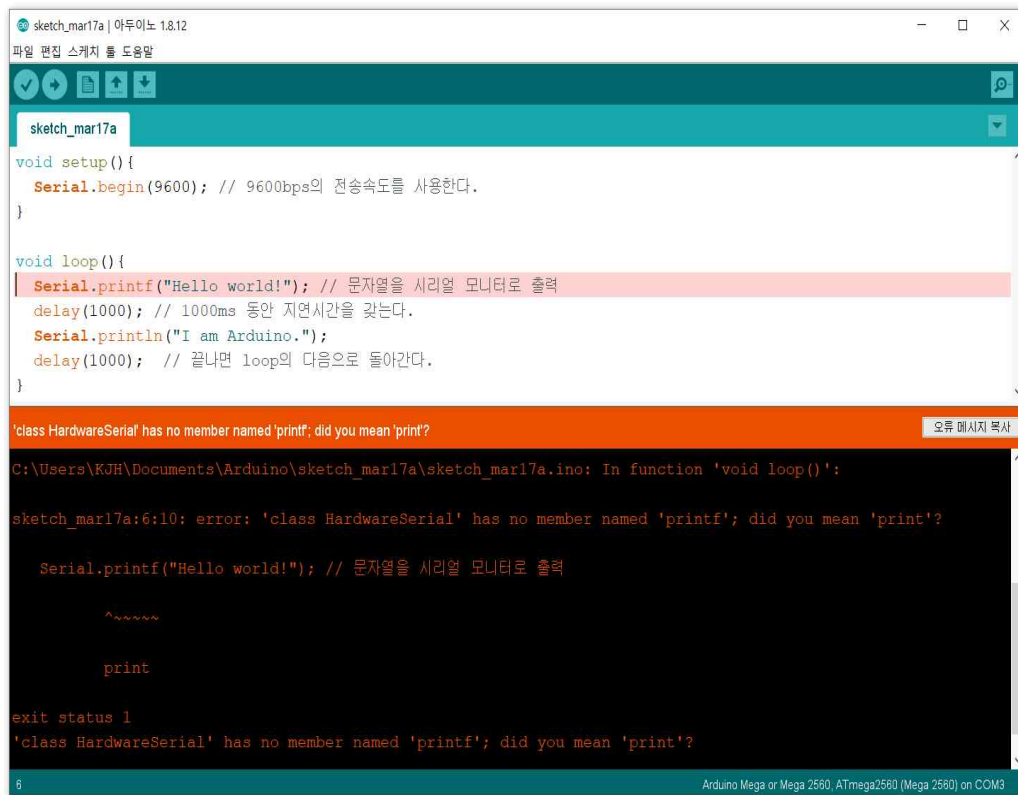



그림 C.2.4 컴파일 오류 발생 사례

만약 그림 C.2.4처럼 컴파일이 정상적으로 수행되지 않으면 주황색으로 오류에 대한 설명이 나타난다<sup>4)</sup>.

C:\Users\KJH\Documents\Arduino\sketch\_mar17a\sketch\_mar17a.ino: In function 'void loop()':  
 sketch\_mar17a:6:10: error: 'class HardwareSerial' has no member named 'printf'; did you mean 'print'?

이 경우에는 실행 가능한 수행 파일이 생성되지 않는다. 본 사례는 스케치에서 지원되지 않는 함수(printf)를 사용한 경우이다. 오류가 있으면 이를 정상적으로 정정해야 다음 업로드 과정을 행할 수 있다.

- ② **업로드(upload) 및 수행**: 아두이저(Ardu-Ez)의 AVR 프로세서안에 내장된 Flash memory 안에 프로그래머가 소스코드를 컴파일하여 만든 CPU의 실행 파일을 저장하는 과정을 아두이노에서는 업로드(Upload)라고 한다. 업로드는 그림 C.2.5와 같이  (업로드) 버튼을 클릭하여 실행한다. 개발자의 편의를 위해 콤파일을 하지 않고 바로 업로드 버튼을 클릭해도 내부에서 콤파일을 수행한 후 이상이 없으면 업로드 과정을 수행한다.

4) 소스 프로그램에서는 본 함수를 사용하지 않았지만 편의상 오류를 가정하여 설명하기로 한다.



그림 C.2.5 업로드 메뉴 아이콘 선택하기

업로드를 실패했을 때는 아래 사항을 점검해 보기 바란다.

☞ 업로드 전 점검 사항!

- USB 케이블이 연결되었는지 확인
- 도구 -> 보드 -> Arduino Mega 2560 or Mega ADK 확인
- 도구 -> 시리얼 모니터 -> PC와 연결된 COM 포트 확인

오류가 없는 경우 그림 C.2.6처럼 “업로드 완료” 문구가 출력됨과 동시에 아두이노의 AVR CPU는 스스로 업로드한 프로그램을 실행한다<sup>5)</sup>.

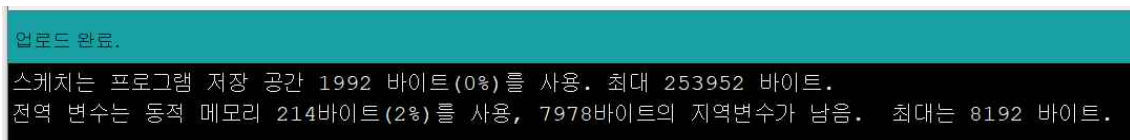



그림 C.2.6 업로드 완료 메시지

③ **시리얼 모니터 창 개방**: 업로드가 완료되면 시리얼 모니터 창을 개방한다. hot key (Control+Shift+M)을 입력하거나 우측 상단의  아이콘 혹은 “메뉴->툴->시리얼모니터” 메뉴를 클릭하여 개방할 수 있다. 성공적으로 수행된다면 그림 C.2.7과 같이 시리얼 모니터 창에 “Hello world!”와 “I am Arduino.”가 1초 간격으로 출력될 것이다.

5) 프로그램은 AVR 프로세서의 내부의 비휘발성(non volatile) 플래시 메모리에 저장되기 때문에 전원을 off해도 그 내용은 사라지지 않는다. 전원을 인가하면 따로 프로그램을 업로드하지 않아도 이미 내장된 프로그램이 자동으로 수행된다.

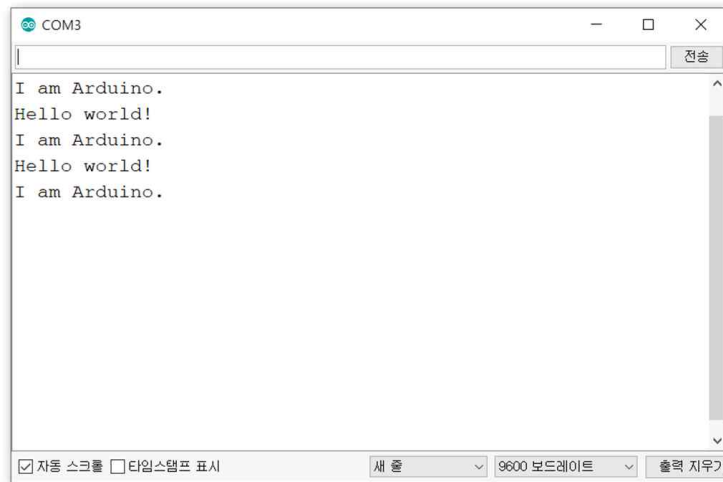


그림 C.2.7 시리얼 모니터의 출력 결과

## C.3 고찰

1. 컴파일링 오류가 발생하였는데 아두이노 실험 키트의 전원을 다시 인가하였다. 어떤 일이 발생하는가?
2. 컴파일 작업을 성공하였는데 업로딩에서 오류가 발생하였다. 업로딩 실패에 대처하기 위한 조치 사항을 3가지만 제시하시오.
3. 현재까지 실험에서 활용한 시리얼 모니터의 용도에 대하여 기술하시오.
4. 본 실험에서 사용한 COM 포트의 역할에 대하여 기술하시오.
5. 스케치를 아두이노 공식 사이트와 Ardu-Ez 제작사 사이트에서도 다운받을 수 있는데 비표준 쉘드를 제어하는 함수는 누가 제공할 것일까?