《3. 아두이노 스케치》

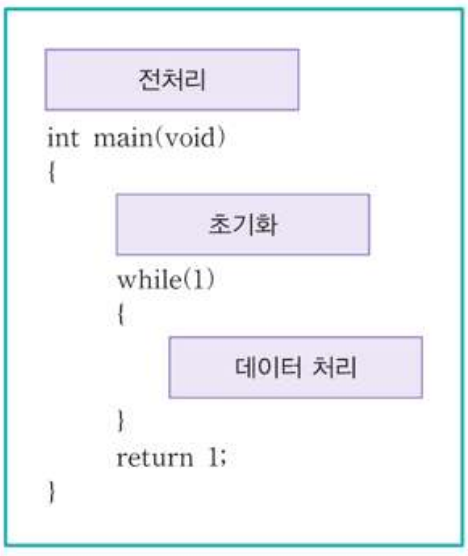
1. 스케치란? (통합 개발 환경 IDE)

- 아두이노 IDE
  아두이노에서 프로그램을 그림 그리듯 쉽게 작성할 수 있다는 의미로 스케치라 불린다.
  아두이노 IDE는 작고 간단한 통합개발환경을 가지고 있다.
  초보자를 위해 꼭 필요한 기능들로만 구성되어 있다.
  예제 소스들도 포함하고 있기 때문에 간단한 예제는 아두이노 스케치 내에서 가져다 쓸 수 있다
  한번의 클릭으로 컴파일과 업로드를 할 수 있어서 쉽게 사용할 수 있다.
  JAVA로 구현되어 OS 간 이식성이 뛰어나다. 실제로 Windows 뿐만 아니라 Linux에서도 사용이 가능하다.

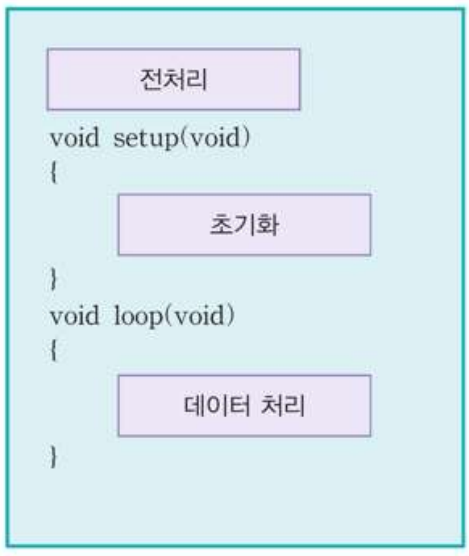
- 아두이노 SW 장점
  서로 다른 마이크로 컨트롤러를 사용하는 아두이노 보드를 똑같은 코드로 동일한 동작을 구현할 수 있다.
  아두이노에 꽂힌 ATmega328p와 같은 AVR 시리즈에서는 DDRx와 PORTx 레지스터에 대해서 직접 제어해야 하는데
  아두이노에서 사용하는 pinMode 나 digitalWrite는 모든 아두이노 보드에서 동일하다.
  라이브러리의 종류에 대해서도 AVR 시리즈는 헤더파일이 iom128, iom328p처럼 다르지만
  아두이노는 기본 라이브러리가 아두이노 IDE와 함께 기본적으로 설치되고
  필요에 따라 확장 라이브러리는 개별적으로 설치해서 사용할 수 있다는 장점이 있다.

2. 스케치의 기본 구조 알아보기

아두이노 스케치는 C/C++을 기반으로 한다.
2개의 기본 함수인 setup과 loop 함수로 구성되어 있다.
setup 함수는 초기화 함수이고 스케치가 실행될 때 한 번만 실행된다.
loop 함수는 반복 실행 함수라서 마이크로 컨트롤러를 위한 프로그램에서 메인/이벤트 루프에 해당된다.
그래서 일반 프로그래밍 구조와 다른 것을 알 수 있다.
전처리 → int main (void) → 초기화 → while(1) → 반복될 데이터처리 → return → 완료
전처리 → void setup → 초기화 → void loop → 데이터처리 → 완료



uC를 위한 프로그램의 구조



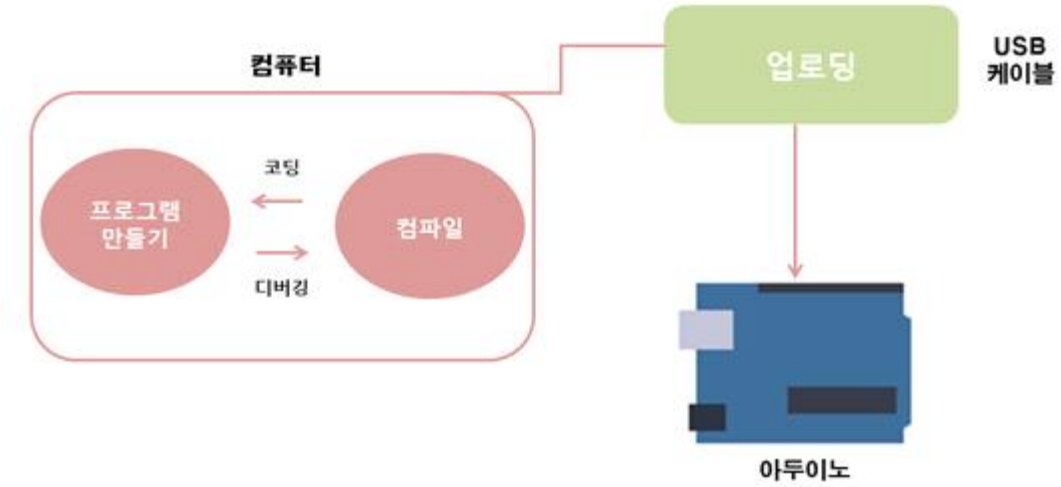
아두이노를 위한 스케치의 구조

아두이노는 비전공자들을 위한 마이크로 컨트롤러 플랫폼이다.

표준화되고 공개된 하드웨어와 소프트웨어를 사용한다.
간단하게 사용할 수 있고 최소한의 기능을 제공하고 있다.
확장 라이브러리를 통해서 소프트웨어 확장할 수 있고 쉴드보드를 통해서 하드웨어도 확장할 수 있다.

3. 스케치 구조, 업로드 방법

아두이노와 PC에 USB 케이블을 이용하여 연결하고 USB장치 드라이버가 정상적으로 설치 완료되면
윈도우 PC 장치 관리자에 Arduino Uno(COMx) 포트가 올라온 것을 확인할 수 있다.
“COMx”의 x는 사용자 PC마다 다른 번호로 나타날 수 있고 같은 PC라도 USB 포트에 따라 숫자가 달라진다.

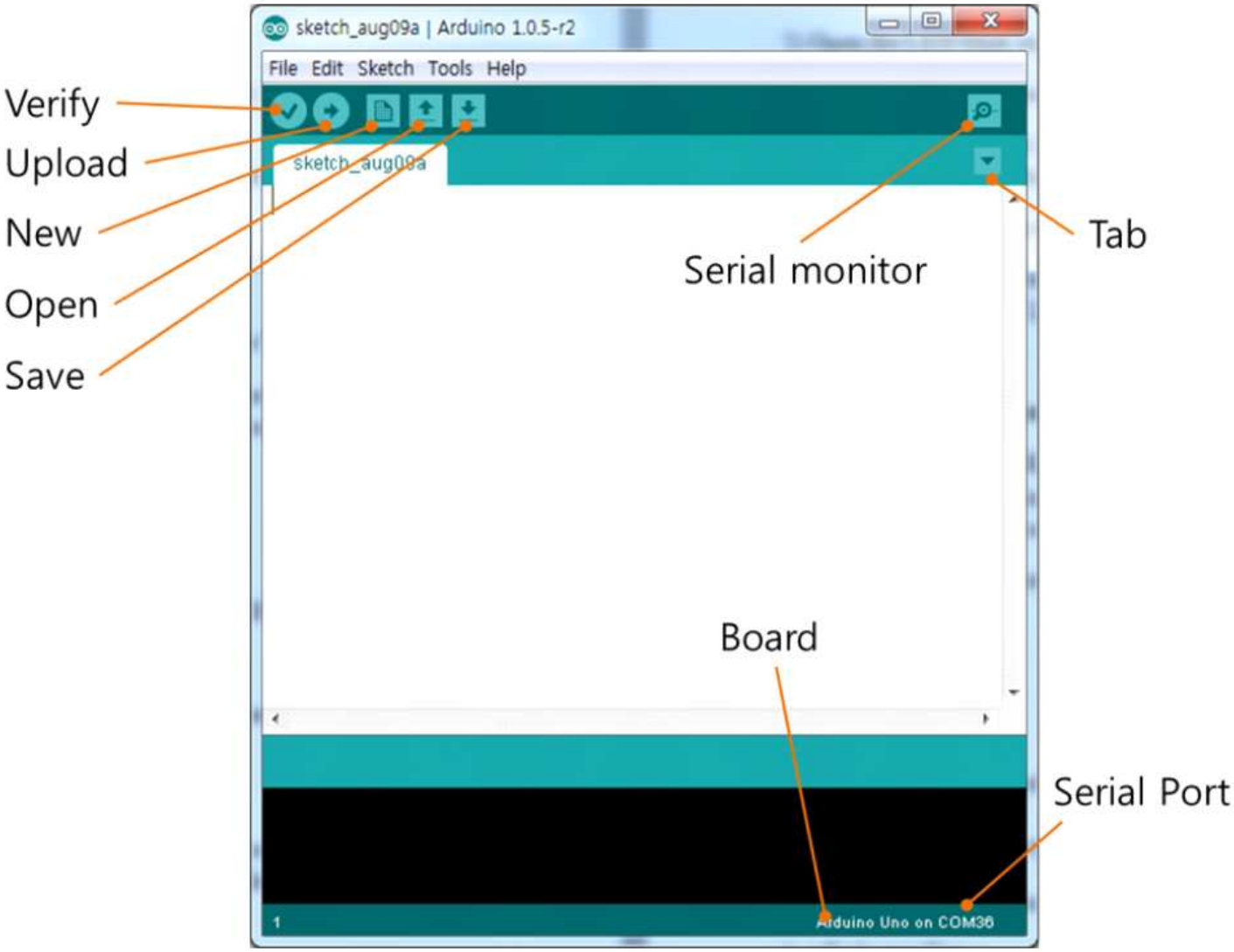


1. 프로그래밍(프로그램을 만드는 과정, 명령을 만드는 과정)을 하거나 예제에서 예제파일을 불러온다.
2. 도구 - 보드 - 보드선택 : 사용하고자하는 아두이노 보드를 선택한다.
   만약 프로세서도 설정이 필요한 보드라면 프로세서 설정도 함께 해준다.

3. 도구 - 포트 : 아두이노 보드에 할당된 가상 COM 포트를 선택한다.
   아두이노 보드에 할당된 포트는 장치관리자에서 확인한다.

4. 툴바를 사용해서 확인과 업로드를 한다.
   확인 : 스케치 컴파일
   시리얼 모니터 : 컴퓨터와의 데이터 송수신을 위한 프로그램 진행
   만일 컴파일 시 에러가 발생했다면 정보 윈도우에서 에러가 발생한 라인 번호와 원인을 알려준다.
   에러를 수정한 후에 컴파일 해서 에러가 없어졌다면 다시 아두이노 하드웨어에 업로드 할 준비가 완료된 것이다.

   컴파일 (편집, 번역) : 마이크로 컨트롤러에 프로그램을 넣어도 이상이 없는지, 문법에 이상이 없는지 검사하고
   마이크로 컨트롤러에 넣을 수 있는 형태로 바꾸는 과정
   코딩 (프로그램 작성) : 컴퓨터 작업의 흐름에 따라 프로그램의 명령문을 이용하여 프로그램 작성하는 일
   디버깅 (오류수정) : 프로그램의 잘못을 찾아내고 고치는 작업
   업로드 : 스케치 컴파일 및 생성된 기계어 파일을 마이크로 컨트롤러에 업로드 (한번의 클릭으로 업로드까지 진행된다)
   스케치 상의 프로그램이 아두이노로 잘 전송이 되었다면 아두이노 상의 RX/TX가 깜빡일 것이다.
   RX/TX의 LED가 꺼지고 정보 윈도우에서 프로그램이 업로드 마쳤다는 것을 확인했다면 프로그램이 올라간 것이다.



확인 (verify) : 코드를 보내기 전에 보드가 이해할 수 있도록 명령어를 변경한다.
업로드 (Upload) : 확인, 번역한 후 USB 케이블로 보드에 전송한다.
새파일 (New) : 새 스케치를 만들기 위해 새창 띄운다.
열기 (Open) : 컴퓨터에 있는 다른 스케치 파일을 연다
저장 (Save) : 현재 작업을 스케치로 변환하여 저장한다.
시리얼 모니터 (Serial monitor) : 아두이노에서 매우 유용한 기능이다. USB를 통해 보드로 메시지를 보내거나 읽기가능
탭(Tab) : 여러 개의 스케치 작업을 할 수 있게 해준다. 좀 더 고급 프로그래밍을할 때 주로 사용된다.

4. 아두이노 프로그래밍 구조

아두이노 프로그래밍은 구조, 값(변수나 상수), 함수로 이루어진다.

1. 구조 : 아두이노 코드의 요소

스케치 : setup(), loop()
제어구조 : break, continue, do...while, else, for, goto, if...else, return, switch...case, while
추가문법 : #define, #include, /* */, //, ; , { }
산술연산자 : %, *, +, -, /, =
비교연산자 : !=, <, <=, ==, >, >=
부울연산자 : !(논리NOT), &&(논리AND), ||(논리OR)
포인터 액세스 연산자 : &(참조 연산자), *(역참조 연산자)
비트연산자 : &(비트AND), <<(왼쪽쉬프트), >>(오른쪽쉬프트), ^(XOR), |(비트OR), ~(비트NOT)
복합연산자 : &=(복합비트AND), *=(복합곱셈), ++(증가), +=(복합 덧셈), --(감소), -+(복합뺄셈)
/=(복합나눗셈), ^=(복합XOR), |=(복합OR)

2. 변수 : 아두이노 자료형 및 상수

상수 : 부동 소수점 상수, 상수, 정수 상수
변환 : byte(), char(), float(), int(), long(), word()
자료형 : String(), boolean, byte, char, double, float, int, long, short, string
unsigned char, unsigned int, unsigned long, void, word, 배열
변수 scope 및 한정어 : const, scope, static, volatile
유틸리티 : PROGMEM, sizeof()

3. 함수 : 아두이노 보드를 제어하고 계산을 수행

디지털 입출력 : digitalRead(), digitalWrite(), pinMode()
아날로그 입출력 : analogRead(), analogReference(), analogWrite()
고급 입출력 : noTone(), pulseIn(), pulseInLong(), shiftIn(), shiftOut(), tone()
시간 : **delay()**, delayMicroseconds(), micros(), millis()
수학 : abs(), constrain() map(), max(), min(), pow(), sq(), sqrt()
삼각법 : cos(), sin(), tan()
문자 : isAlpha(), isAlphaNumeric(), isAscii(), isControl(), isDigit(), isGraph(), isHexadecimalDigit()
isLowerCase(), isPrintable(), isPunct(), isSpace(),isUpperCase(), isWhitespace()
난수 : random(), randomSeed()
비트 및 바이트 : bit(), bitClear(), bitRead(), bitSet(), bitWrite(), highByte(), lowByte()
외부 인터럽트 : attachInterrupt(), detachinterrupt()
인터럽트 : interrupts(), nointerrupts()
통신 : **Serial.begin()**, Serial.end(), **Serial.available(), Serial.read()**, Serial.peek(), Serial.flush()
Serial.print(), Serial.println(), write, stream
USB : 마우스, 키보드

아두이노 예제 Blink 프로그램 구조
LED를 깜빡이게 하는 간단한 예제 프로그램을 스케치에서 찾을 수 있다.
1초 상간으로 LED가 켜졌다가 꺼졌다가 하는 Blink 프로그램을 만들어 볼 것이다.
LED는 아두이노의 13번 핀에 연결한다.
setup()은 아두이노 스케치에서 한번만 실행되고 loop()는 아두이노 스케치가 종료할 때까지 반복실행 된다.

```
int LED = 13;                   // 아두이노 디지털 입출력핀 13번을 int형 변수 LED에 넣는다는 것이다.

void setup() {                  // 초기화 작업
  pinMode(LED, OUTPUT);         // LED(13번핀)을 출력으로 설정하겠다는 것이다.
}

void loop() {                   // 반복 작업
  digitalWrite(LED, HIGH);      // 전압레벨을 올려서 LED(13번핀)를 켠다.
  delay(1000);                  // 1초 = 1000, 1초동안 기다린다.
  digitalWrite(LED, LOW);       // 전압레벨을 낮춰서 LED를 끈다.
  delay(1000);                  // 1초동안 기다린다. (끄는 delay가 없으면 계속 켜진 것처럼 보인다)
}
```

아두이노 에서는 C/C++ 언어를 사용하고 setup()과 loop() 함수로 구성되어 있다.

- setup()
  스케치(아두이노 프로그램)가 시작될 때 호출
  변수 초기화, 핀 모드설정, 라이브러리 초기화 작업
  보드의 전원을 켜거나 reset되었을 때 한번 실행

- loop()
  setup() 함수에서 초기화를 수행한 후에 loop() 함수 호출
  연속적으로 반복하여 실행
  아두이노 보드의 동작을 제어하는 프로그램 실행 함수

setup과 loop 함수는 둘 다 어떤 값도 리턴하지 않는다.
리턴값이 필요없는 함수 이름 앞에는 void라는 키워드를 붙여 리턴 값이 없다는 것을 나타내야 한다.
함수 이름 다음에 오는 괄호 안에는 인수가 포함될 수도 있다. 인수가 없어도 괄호는 반드시 붙여야 한다.
함수를 호출하는 것이 아니라 함수를 정의하기 때문에 세미콜론은 붙일 필요가 없다.
함수가 호출되면 어떠한 작업이 수행되어야 하는지를 알려주는 것이다.
함수가 호출되었을 때 수행해야 하는 작업은 { } 중괄호로 묶어야 하고 이 안에 들어있는 코드를 합쳐 코드블록이라 한다.

pinMode 함수는 어떤 핀을 입력모드나 출력모드로 설정한다.
pinMode는 핀을 사용한다는 초기화 작업이므로 한 번만 수행되면 되기에 setup에서 실행한다.
digitalWrite 함수를 사용하려면 핀 번호와 핀 상태(HIGH 또는 LOW)를 지정해야 한다.
이 두가지 정보를 인수(argument)라고 부른다. 인수는 함수가 호출될 때 함수에 전달된다.
전체 괄호안에서 하나하나를 쉼표로 구분한다.
delay(1000)에서 delay는 밀리초로 계산하기 때문에 1초를 사용할 때는 1000으로 설정해야 하는 것이다.
하나의 행이 끝나면 세미콜론을 붙여서 한 명령의 끝을 알려준다.

4. 기초 프로그래밍

[상수]

* HIGH / LOW

HIGH	pinMode, INPUT 설정 시	pin을 읽었을 때 3V(VDD=5V) 전압이 걸리면 HIGH로 인식
		HIGH로 출력할 경우 내부 풀업저항을 설정
	pinMode, OUTPUT 설정 시	출력 값을 HIGH로 설정할 경우 pin은 5V
LOW	pinMode, INPUT 설정 시	전압이 1.5V(VDD=5V)인 경우 LOW로 인식
	pinMode, OUTPUT 설정 시	0V

* INPUT / OUTPUT / INPUT_PULLUP

  핀의 입출력 방향 설정

INPUT	pin을 입력으로 설정할 때 사용
	high-impedance 상태 설정
	센서 값을 읽기에 유용하지만 LED 제어 불가
	pull-up 및 pull-down 저항을 연결해야 할 수도 있음 (ex. pushbutton)
OUTPUT	pin을 출력으로 설정할 때 사용
	low-impedance 상태 설정
	LED 제어에 유용하나 센서 값을 읽기에 부적합
INPUT_PULLUP	pin을 입력으로 설정할 때 사용
	MCU 내부에서 pull-up 저항 연결

* TRUE / FALSE

  아두이노 언어에서 참과 거짓을 표현하는데 사용된다.

  False : 0

  True : 1, 0이 아닌 모든 정수는 논리적 의미에서 true

  HIGH, LOW, INPUT & OUTPUT과 달리 소문자를 사용한다.

* 정수 상수 (integer constants)

  프로그램에서 직접 사용

  표현할 수 있는 범위는 단어 길이에 의존

  일반적으로 정수형 상수는 10진수 정수로 처리

  정수형 상수는 4가지로 표현된다 = 2진수, 8진수, 10진수, 16진수

  2진수는 0과 1로 표현하고 숫자 앞에 B를 써서 표현한다.

  8진수는 0~7까지 표현하고 숫자 앞에 0을 써서 표현한다.

  16진수는 0~9, A~G까지 표현하고 숫자 앞에 0x를 써서 표현한다.

	2진수	8진수	10진수	16진수
123	B1111011	0173	123	0x7B
203	B11001011	0313	203	0xCB

* 부동 소수점 상수 (floating point constants)
  코드를 더 알아보기 쉽도록 하는데 사용
  과학적 표기법으로 다양하게 표현
  E와 e는 모두 유효한 지수의 지표로 사용 가능

일반적 표현	또 다른 표현	Floating-point constant
10	10	10.0
234000	2.34 * 10^5	2.34E5
0.000067	6.7 * 10^-5	6.7E-5

C코드 테스트

void setup()
{
Serial.begin(9600);
Serial.println(1234);
}
void loop()
{
}
	Serial.println은 시리얼 모니터에 메시지를 출력할 수 있는 내장함수이다.
	출력할 메시지에 해당하는 변수를 인수로 받는다.
	이렇게 setup에서 아두이노로 C로 작성된 코드의 결과를 확인할 수 있다.

void setup()
{
Serial.begin(9600);
int a = 2;
int b = 2;
int c = a + b;
Serial.println(c);
}
void loop()
{
}	

void setup()
{
Serial.begin(9600);
int degC = 20;
int degF;
degF = degC * 9 / 5 + 32;
Serial.println(degF);
}
void loop()
{
}
	섭씨온도를 화씨온도로 바꾸는 식이다.
	degC라는 int 변수를 선언했고 이 변수의 초기값을 20으로 지정했다.
	int degC; degC = 20; 이렇게 쓸 수도 있다.
	수식의 우선순위를 좀 더 명확하게 해야할 것이다.
	((degC * 9) / 5) + 32