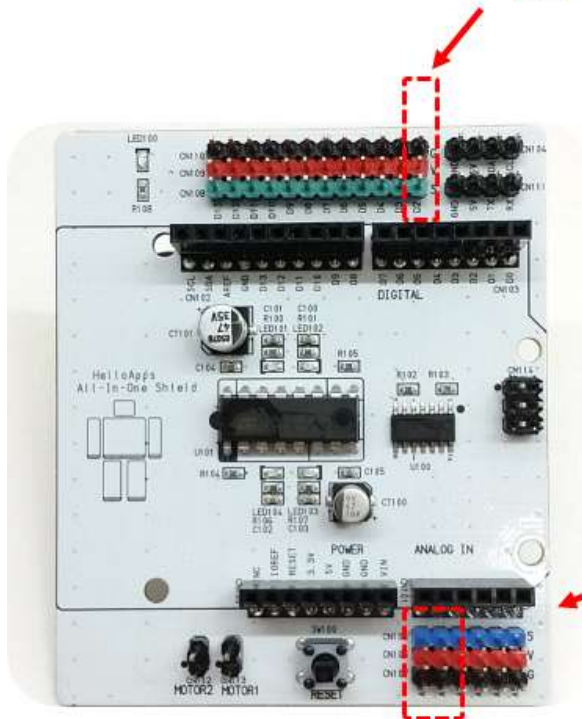


아두이노 보드에 조이스틱 연결하기

- 조이스틱 센서를 다음과 같이 아두이노 보드에 연결한다. 조이스틱 센서 중에서 가장 오른쪽에 있는 세 번째 핀은 디지털 버튼 센서이다.

디지털 버튼은 디지털 핀에 연결합니다.



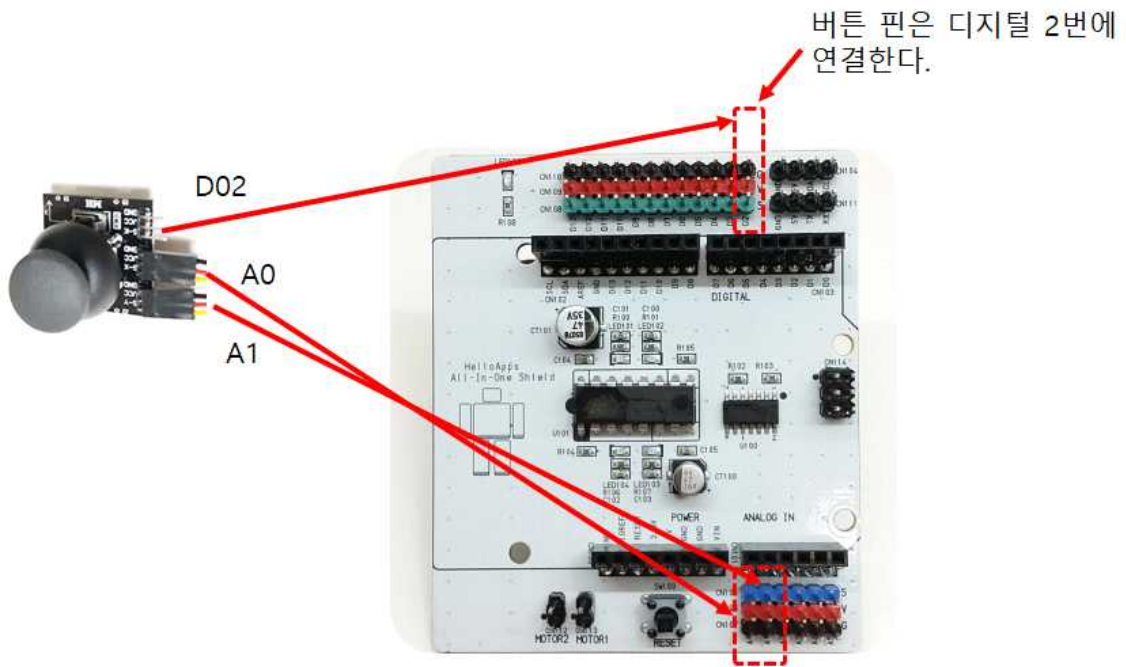
아날로그 핀에는
0번 ~ 5번 까지
번호가 표시되어 있습니다.
(A0 ~ A5)

아날로그 핀에 부품을 연결할
때에는 핀 번호를 확인해야
합니다.

아날로그 핀은 아날로그
핀에 연결합니다.



- 조이스틱에 있는 디지털 버튼은 버튼이 눌러지지 않으면 값이 1이 출력되며, 버튼이 눌러지면 값이 0이 출력된다.



가운데 X축은 A0에, 맨 왼쪽 Y축은 A1에 연결

아두이노에서 조이스틱 센서값 읽기

- 이전 활동에 이어서, 아두이노 보드에 연결된 조이스틱 센서의 값을 확인하는 기능을 수행해 보자. 다음과 같이 아날로그 0번과 1번 핀에서 값을 읽어서 화면에 출력해 보자.

The image shows a Scratch-style code editor with the following code blocks:

- Function Setup**
 - 원쪽에 있는 명령어 아이콘을 이곳에 마우스로 드래그하여 넣어 주세요.
- Function Loop**
 - `d2` = `DigitalRead` `2` (Pin)
 - `a0` = `AnalogRead` `0` (Pin)
 - `a1` = `AnalogRead` `1` (Pin)
 - if** `d2 == 0`
 - `Print` `d2`
 - `Print` `a0`
 - `Print` `"/"`
 - `PrintLine` `a1`
 - `Delay` `100` (ms)

```
void setup()
{
}

void loop()
{
    d2 = DigitalRead(2)
    a0 = analogRead(0)
    a1 = analogRead(1)

    if (d2 == 0)
    {
        Print(d2)
    }

    Print(a0)
    Print(" / ")
    PrintLine(a1)

    Delay(100)
}
```

아두이노에서 문자로 정보 보내기

- 다음과 같이 센서와 버튼의 값을 비교하여 조건이 만족되면 문자가 전송되도록 if 조건문을 추가해 본다.

The image shows a Scratch-style code editor window for an Arduino project. The function is named "Loop". The code consists of the following blocks:

- DigitalRead**: d2 = DigitalRead 2 (Pin)
- AnalogRead**: a0 = AnalogRead 0 (Pin)
- AnalogRead**: a1 = AnalogRead 1 (Pin)
- if**: if d2 == 0
 - Print**: "F"
- if**: if a0 < 100
 - Print**: "-X"
- if**: if a0 > 900
 - Print**: "X"
- if**: if a1 < 100
 - Print**: "-Y"
- if**: if a1 > 900
 - Print**: "Y"
- Delay**: 100 (ms)

```
void setup()
{

}

void loop()
{
    d2 = DigitalRead(2)
    a0 = analogRead(0)
    a1 = analogRead(1)

    if (d2 == 0)
        Print("F")

    if (a0 < 100)
        Print("-X")

    if (a0 > 900)
        Print("X")

    if (a1 < 100)
        Print("-Y")

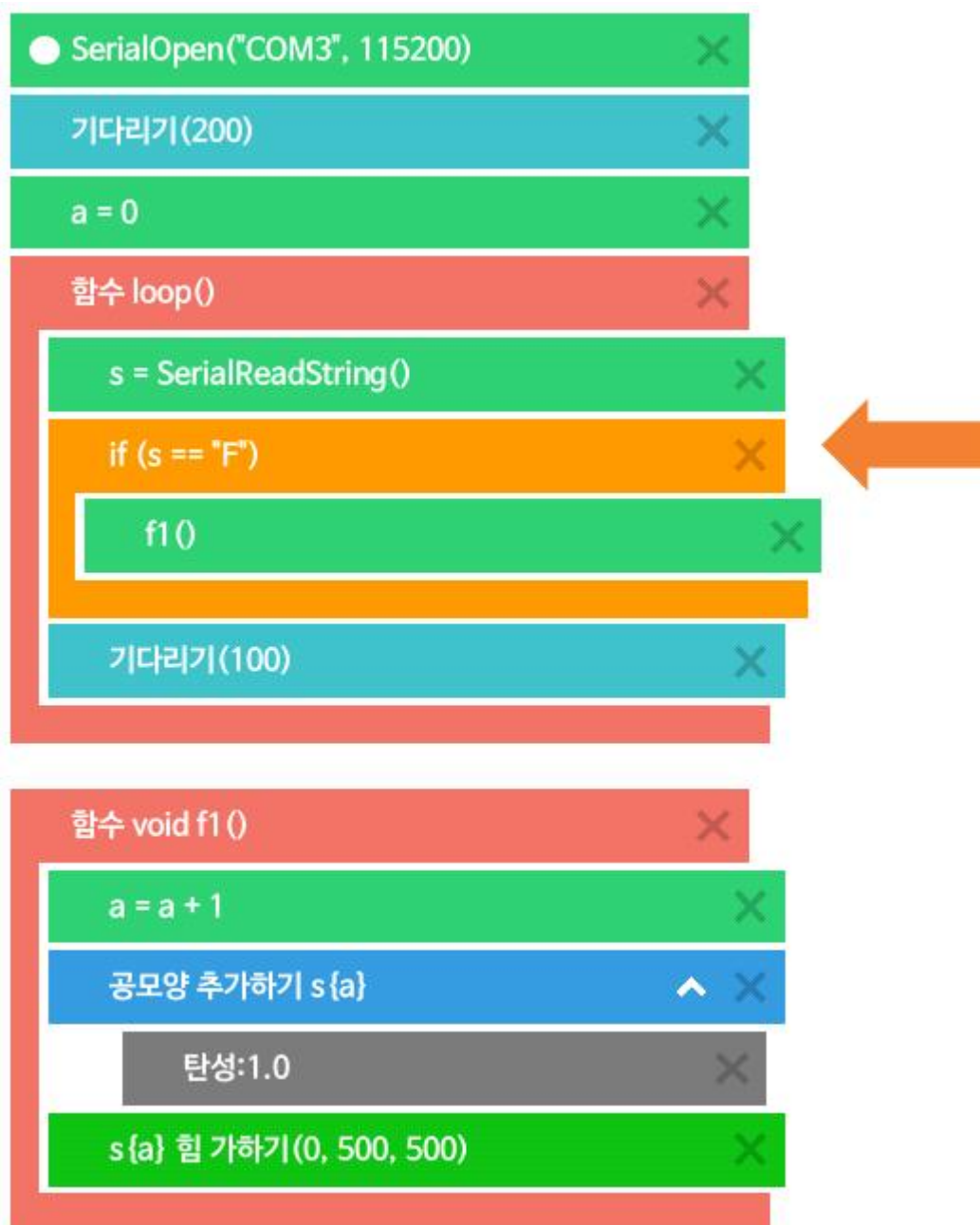
    if (a1 > 900)
        Print("Y")

    delay(100)
}
```

- 위의 코드를 아두이노 보드로 업로드 한 후, 아두이노 코딩 SW를 종료한다.
- 아두이노 보드와 PC 사이에 연결된 USB 케이블을 뽑다가 다시 연결하여 연결을 초기화 해준다.
- 이제 VR 코딩 편집기 프로그램을 실행한다.

VR 화면에서 공 발사하는 함수 만들기

- 이전에 작성한 공 발사하기 기능을 다음과 같이 함수로 수정한 후, 공이 발사되는 지 확인해 본다. if 조건문의 내용이 수정되었다.



```
SerialOpen("COM3", 115200)
delay(200)
a = 0
void loop()
{
    s = SerialReadString()
    if (s == "F")
    {
        f1()
    }

    delay(100)
}

void f1()
{
    a = a + 1
    AddSphere s{a}
        /Bounciness:1.0
    s{a}.AddForce(0, 500, 500)
}
```

카메라 방향으로 공 발사하기

- 이전의 코드는 공이 항상 일정한 방향으로만 발사되도록 되어 있다. 공을 발사하는 AddForce 명령어에서 1개의 값이 입력하면 항상 카메라가 바라보는 전방 방향으로만 공이 발사된다. 아래와 같이 함수 f1()의 내용을 수정해 본다.

```
함수 void f1 ()  
  a = a + 1  
  공모양 추가하기 s{a}  
    탄성:1.0  
  s{a} 힘 가하기(5000)
```



```
SerialOpen("COM3", 115200)
delay(200)
a = 0
void loop()
{
    s = SerialReadString()
    if (s == "F")
    {
        f1()
    }

    delay(100)
}

void f1()
{
    a = a + 1
    AddSphere s{a}
        /Bounciness:1.0
    s{a}.AddForce(5000)
}
```

카메라 방향 회전 시키기

- 기존 명령어에 각각 수신값을 비교하여 카메라를 회전시키는 기능을 다음과 같이 추가해 준다.



```
SerialOpen("COM3", 115200)
delay(200)
a = 0
void loop()
{
    s = SerialReadString()
    if (s == "F")
    {
        f1()
    }

    if (s == "-X")
    {
        CameraRotateY(-1)
    }

    if (s == "X")
    {
        CameraRotateY(1)
    }

    if (s == "-Y")
    {
        CameraRotateX(-1)
    }

    if (s == "Y")
    {
        CameraRotateX(1)
    }
}
```

```
    delay(100)
}

void f1()
{
    a = a + 1
    AddSphere s{a}
        /Bounciness:1.0
    s {a}.AddForce(5000)
}
```