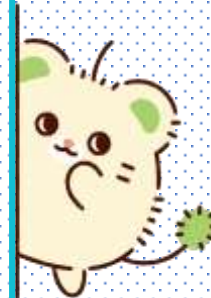




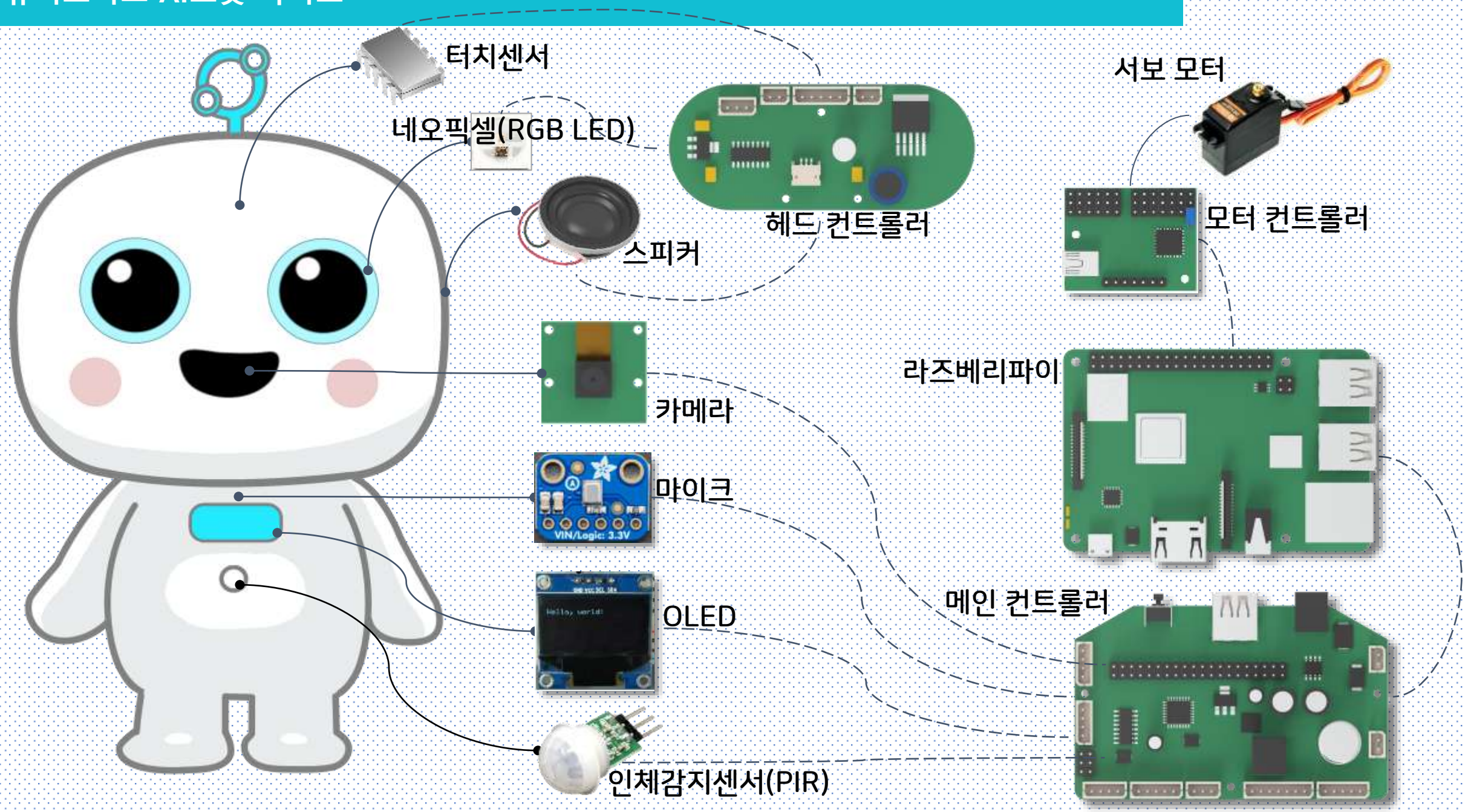
내 손으로 만드는
AI 휴머노이드 로봇비서

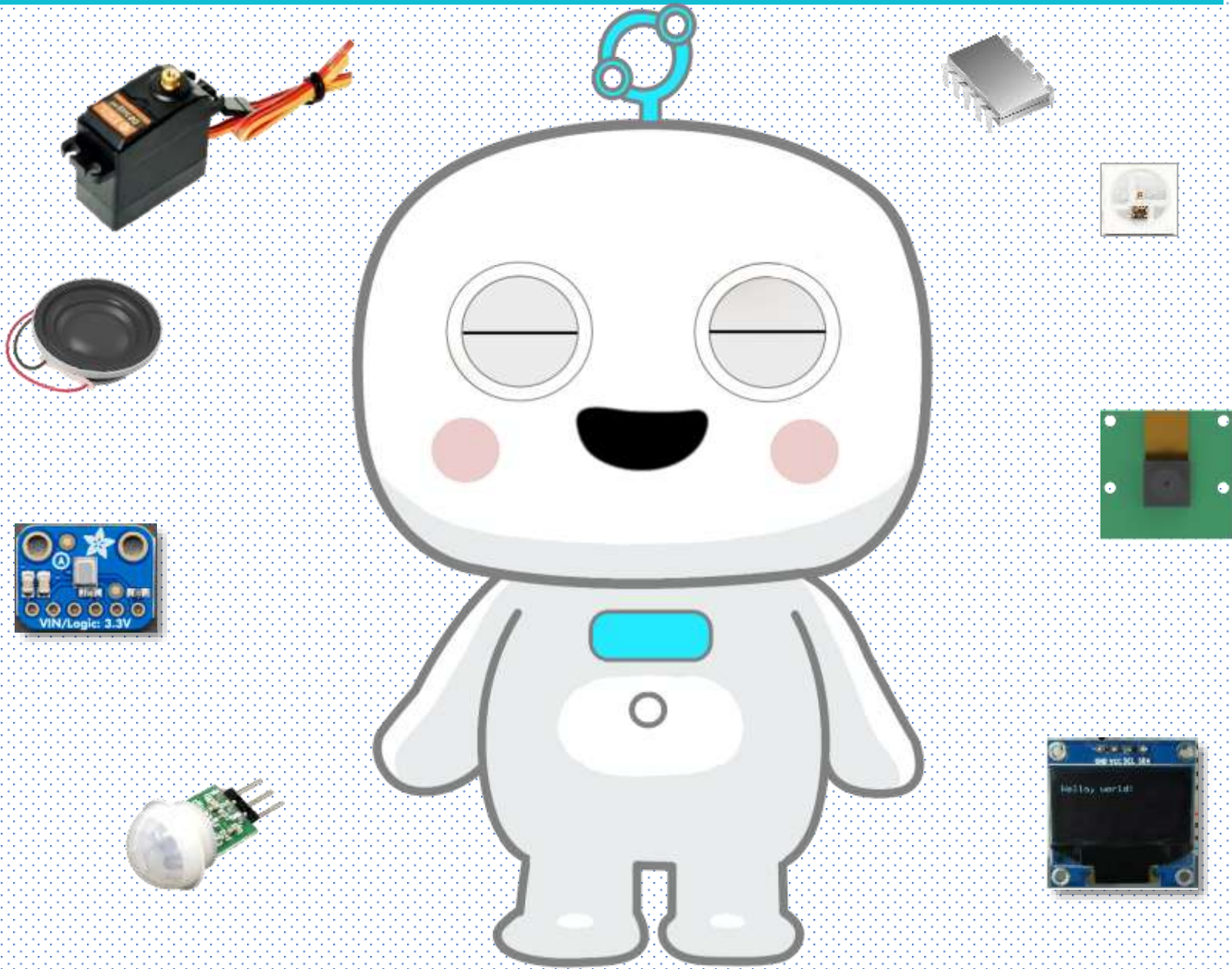


로봇 이해하기



휴머노이드 AI로봇 파이보





네오피셀

디지털신호로 LED의
빛을 조절할 수 있는 전
자부품

MIC

마이크, 소리를 전기신호로 변
환해주는 장치

OLED

전기에 자극받아 빛을 내
는 물질. 휴대폰, TV, 카메
라 화면에 쓰이는 LED

서보모터

파이보의 구동부 역할,
10개의 서보모터를 통해
움직임 구현

터치센서

인체가 닿으면 신호를
감지할 수 있는 센서

스피커

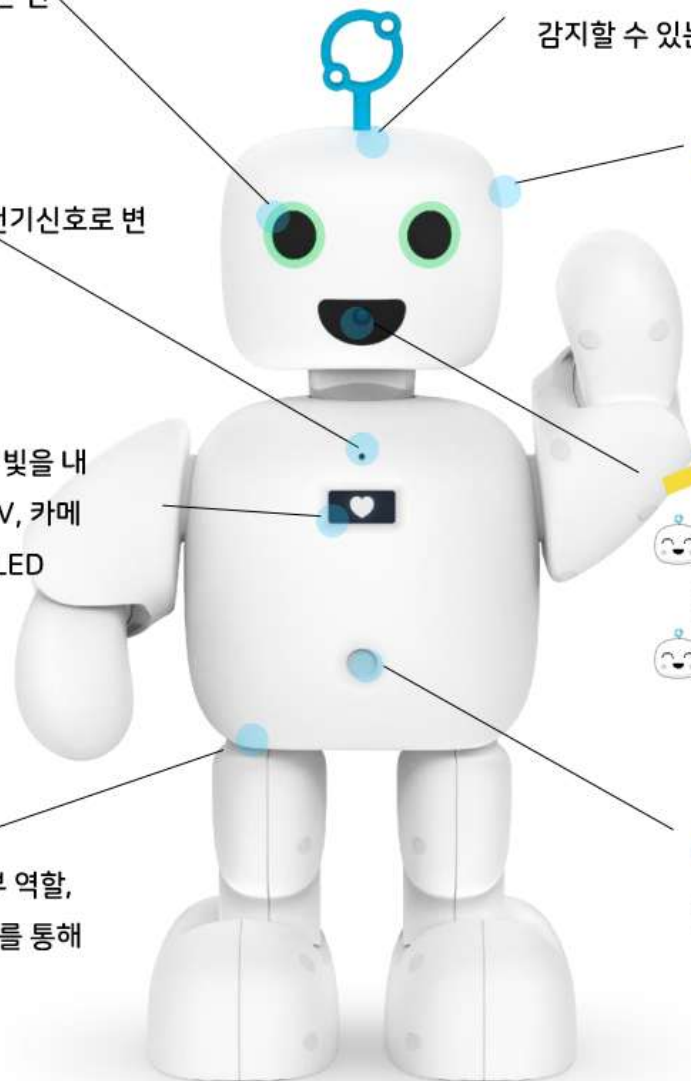
음악이나 음성합성
이 이루어질 수 있도
록 소리를 출력하는
장치

카메라

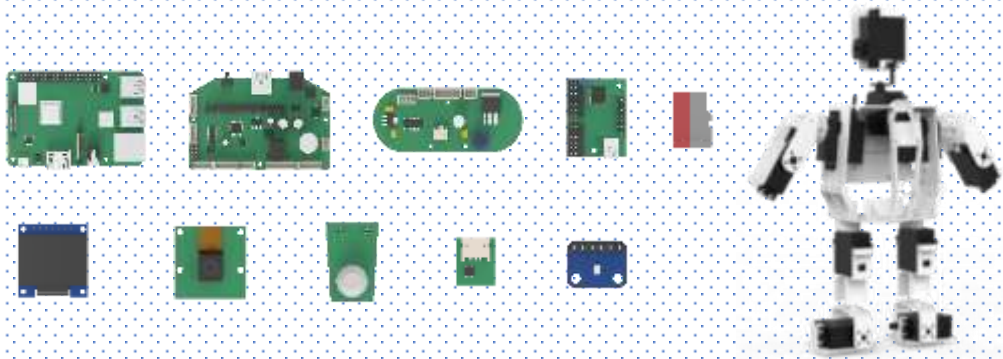
시각 정보를 읽어 전기적인
영상신호로 변환해주는 장치
컴퓨터비전 활용 가능

PIR센서

적외선을 통해 인체
를 감지하는 센서



- 로봇을 구성하는 하드웨어와 소프트웨어



하드웨어(Hardware)

로봇을 비롯한 컴퓨터/기계를 구성하는
물리적인 부품 또는 장치



소프트웨어(Software)

하드웨어를 동작하게 만들어주는
프로그래밍과 서비스 전체

- 하드웨어만 있으면 로봇이 동작할 수 있을까요?

파이보 연결하기





파이보를 꺼내 전용 충전기를 꽂는다.



파이보는 전원이 "꺼진" 상황에서만 구동부를 손으로 움직일 수 있다.
(단, 너무 세게 모터를 돌리지 않도록 유의)

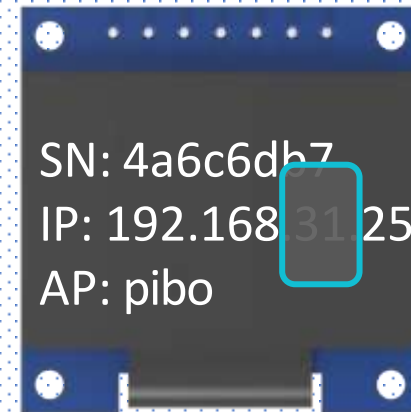
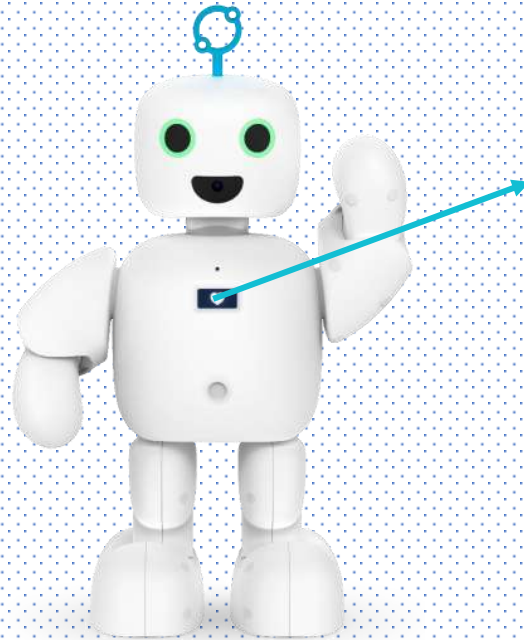
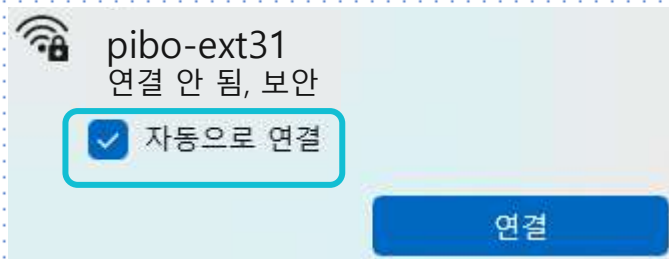


파이보의 전원은 눈에 불이 켜질 때까지 3초 이상 길게 누른다.
(파이보의 전원을 끌 때는 파이보의 눈이 **빨간색**이 될 때까지)

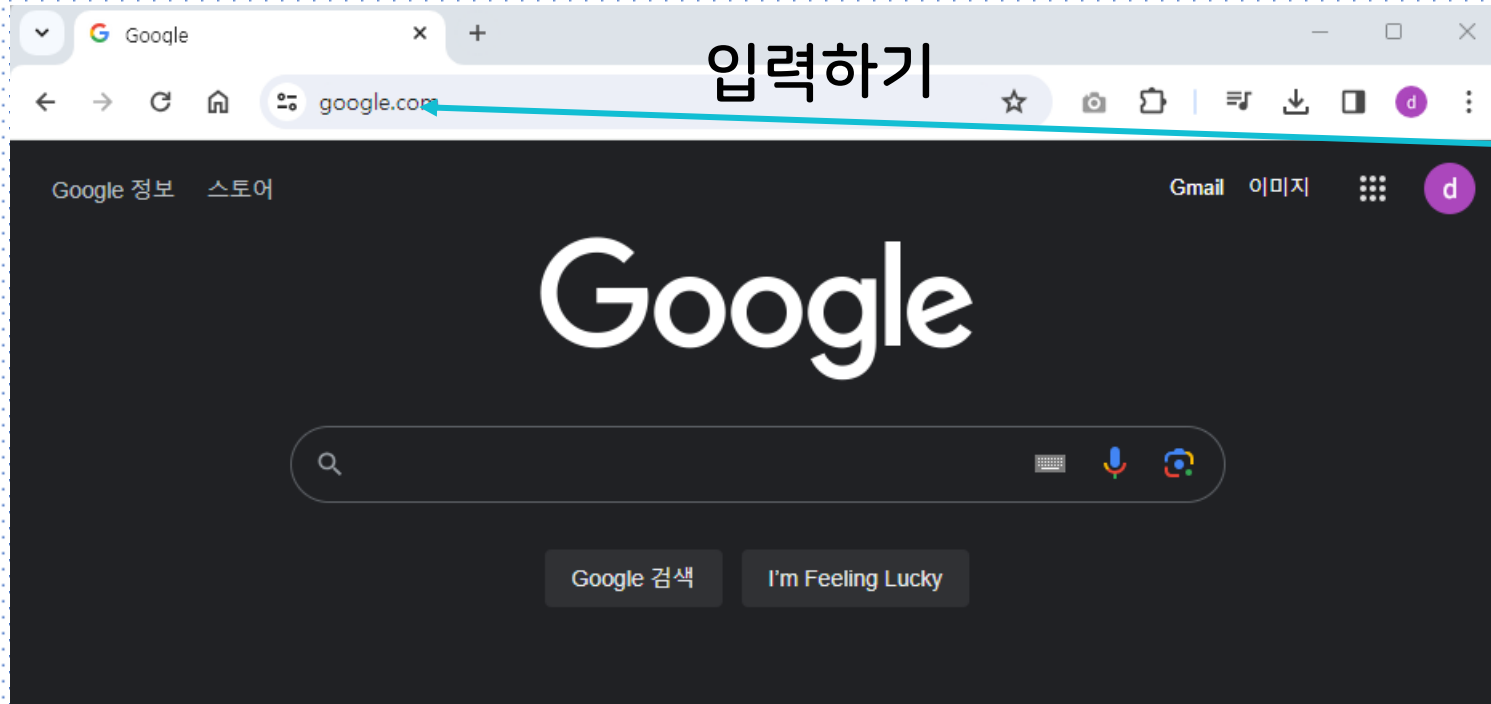


파이보의 가슴(OLED)에 시리얼 번호, IP정보가 나오면 부팅이 완료된다.

- 파이보의 부팅 및 와이파이 연결 상태 확인
- 노트북/태블릿 와이파이를 활용하여 파이보와 연결
 - 파이보 디스플레이를 확인하여 와이파이 목록에서 pibo-ext### 선택
 - 예) IP가 192.168.31.25인 경우 → 노트북 와이파이 pibo-ext31 선택
 - 연결 비밀번호 : !pibo0314
 - 와이파이 '자동으로 연결' 설정

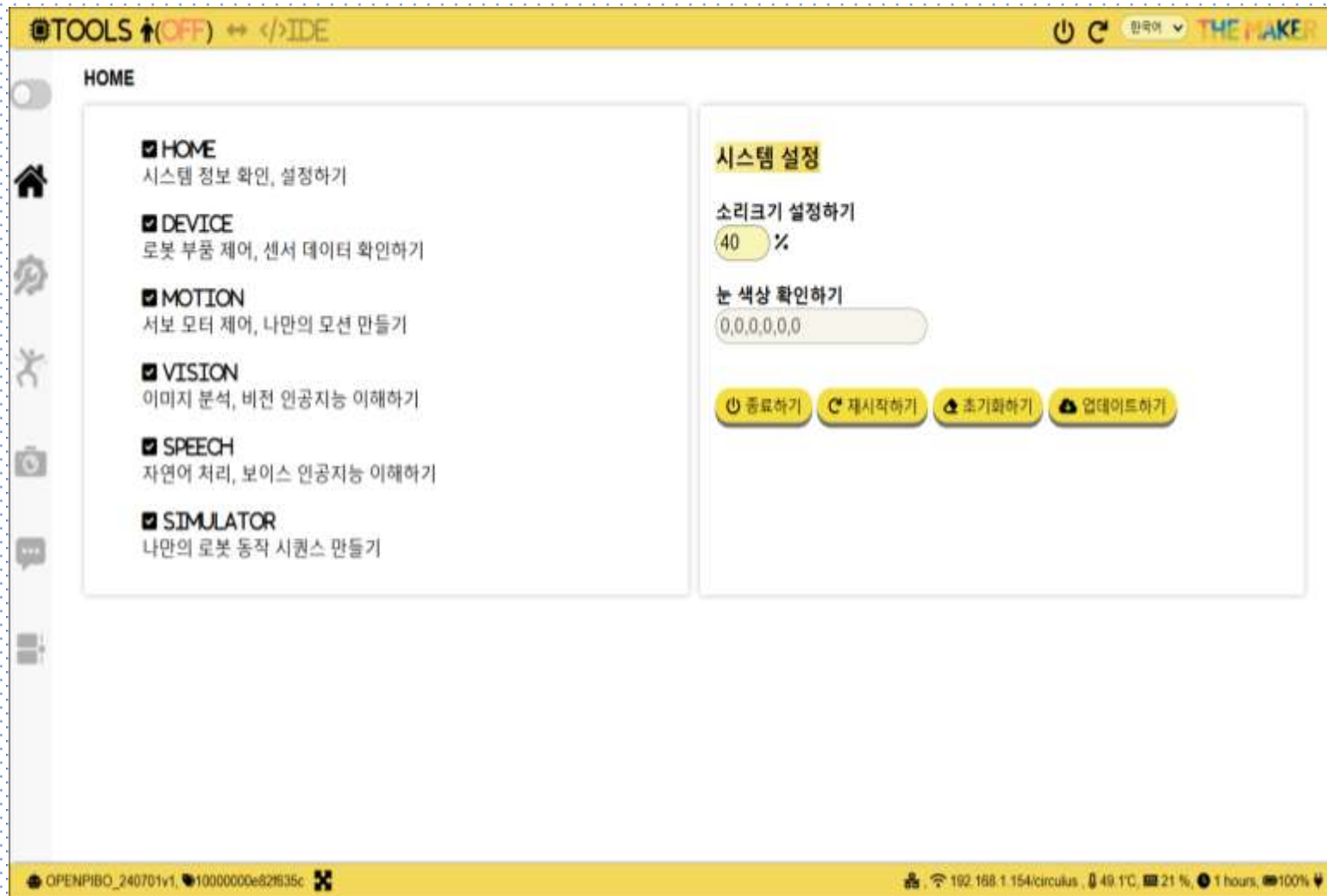


- 크롬 브라우저에서 파이보 메이커(pibo Maker) 접속

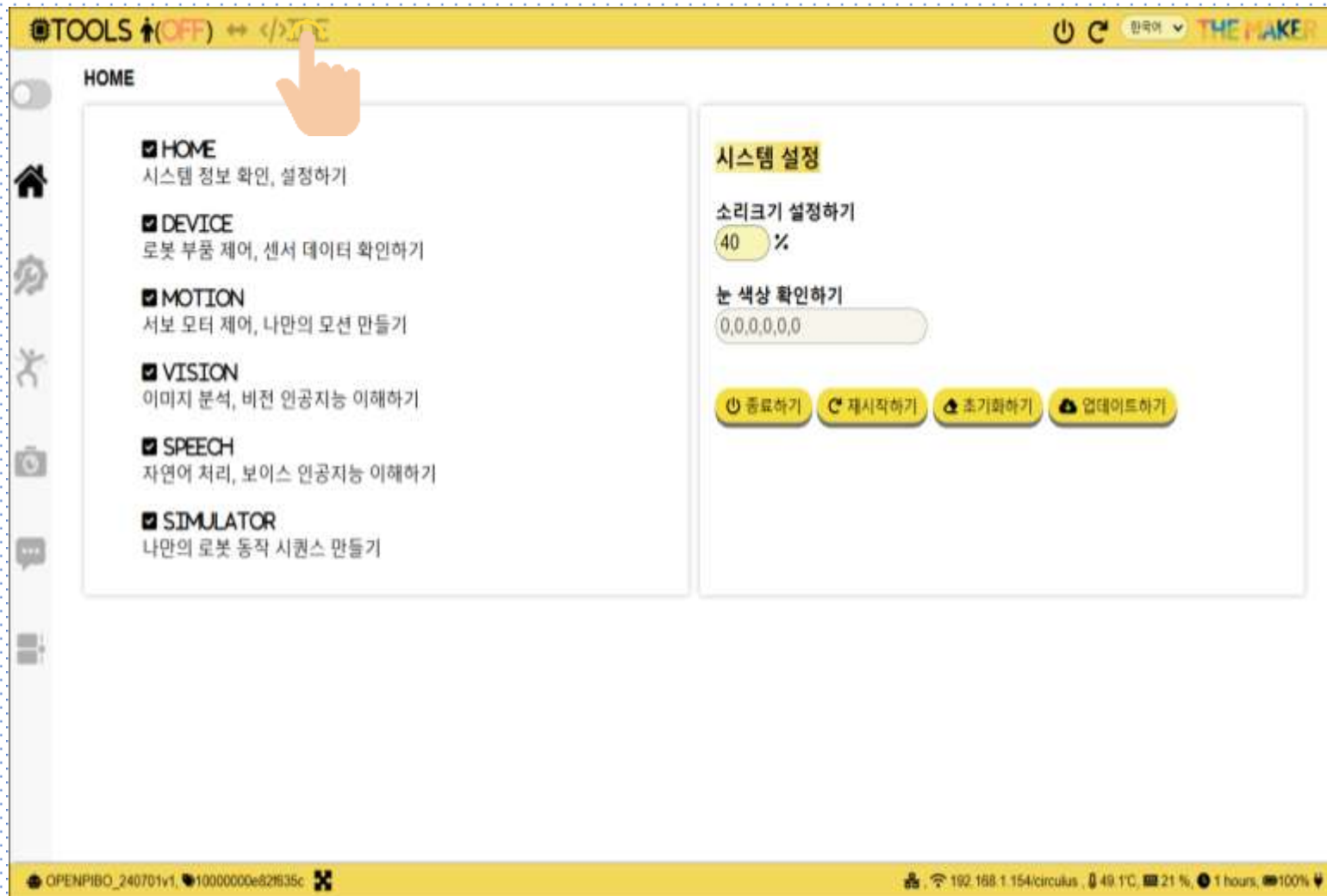


<파이보 디스플레이 정보>

- 크롬 브라우저에서 파이보 메이커(pibo Maker) 접속



- 크롬 브라우저에서 파이보 메이커(pibo Maker) 접속 – 코딩 메뉴로 이동



파이보 연결하기

The image shows the Pyboard IDE interface with several components labeled in Korean:

- 파이썬 코드 화면 테마 | 코드 창 확대**: Points to the top toolbar area.
- 블록/파이썬 가이드**: Points to the top toolbar area.
- 표준 입력**: Points to the top toolbar area.
- 파일탐색기**: Points to the file explorer on the left side.
- 결과 표시 공간**: Points to the output console on the right side.
- 코드 편집 공간 (블록코딩/파이썬)**: Points to the main code editor area.
- 이미지 뷰어, 오디오 플레이어**: Points to the bottom left area.
- 시스템 정보**: Points to the bottom right area.

The code editor displays the following Python code:

```
1 from openpiبو.device import Device
2 import time
3
4 device = Device()
5
6 # WRITE
7 device.eye_on(255,255,255)
8 time.sleep(1)
9
10 # RED BLUE
11 device.eye_on(255,0,0, 0,0,255)
12 time.sleep(1)
13
14 # OFF
15 device.eye_off()
16
```

프로그래밍과 코딩



- 사용자의 명령(요청)에 반응하고 동작하는 소프트웨어
- 주어진 명령을 해결하기 위한 처리 방법과 순서를 정리



- 프로그램을 만드는 일련의 과정(Program + ~ing)



- 컴퓨터가 우리의 명령/요청에 따라 일을 할 수 있도록 프로그램을 만드는 과정
 - 문제 인식
 - 프로그램 설계
 - 프로그램 구현
 - 테스트와 디버깅
 - 프로그램 유지보수

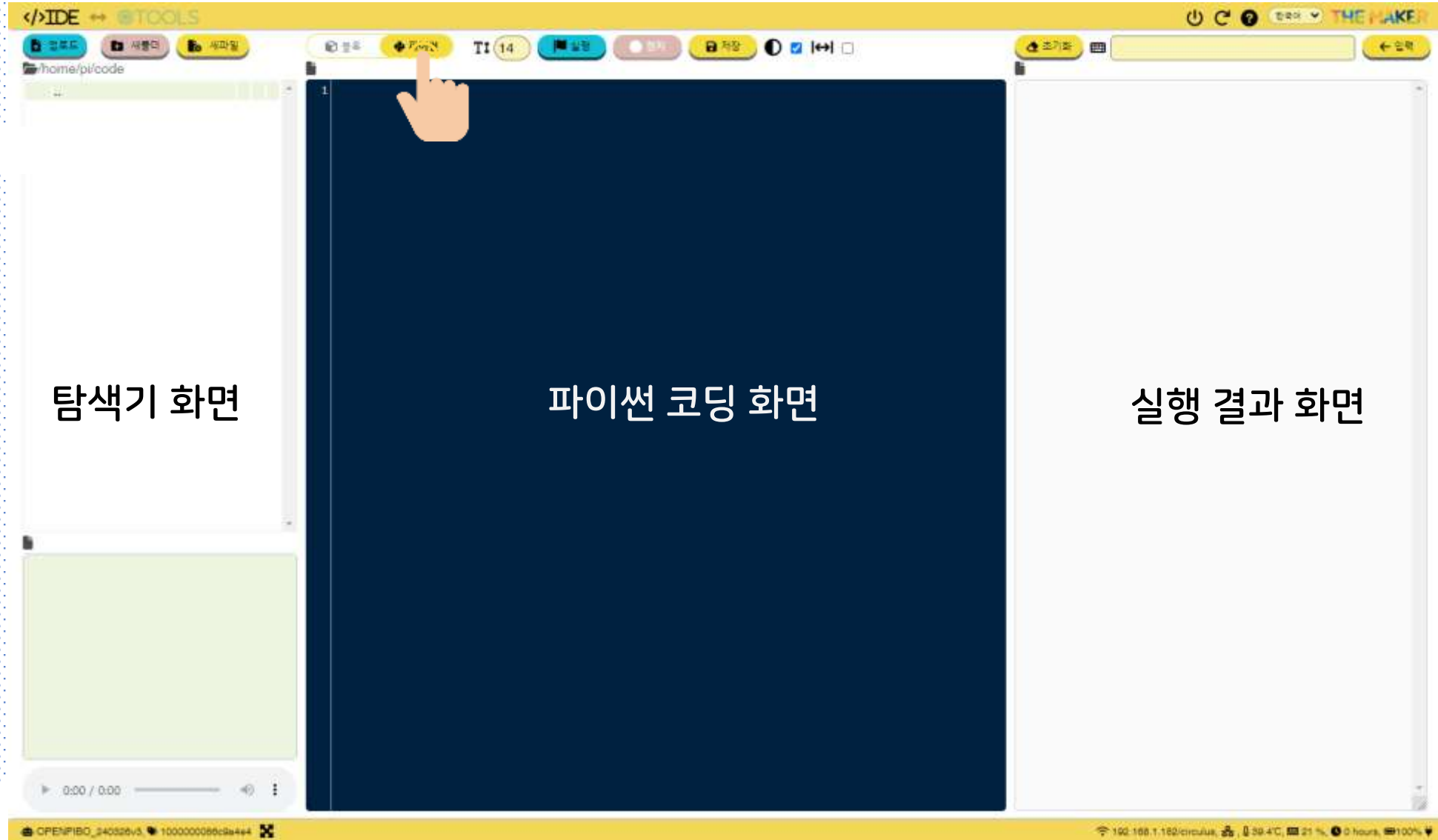
- 프로그래밍 과정 중 필요한 데이터/값을 컴퓨터가 이해할 수 있는 형태로 바꾸는 활동
 - 문제 인식
 - 프로그램 설계
 - 프로그램 구현
 - 테스트와 디버깅
 - 프로그램 유지보수

실습활동

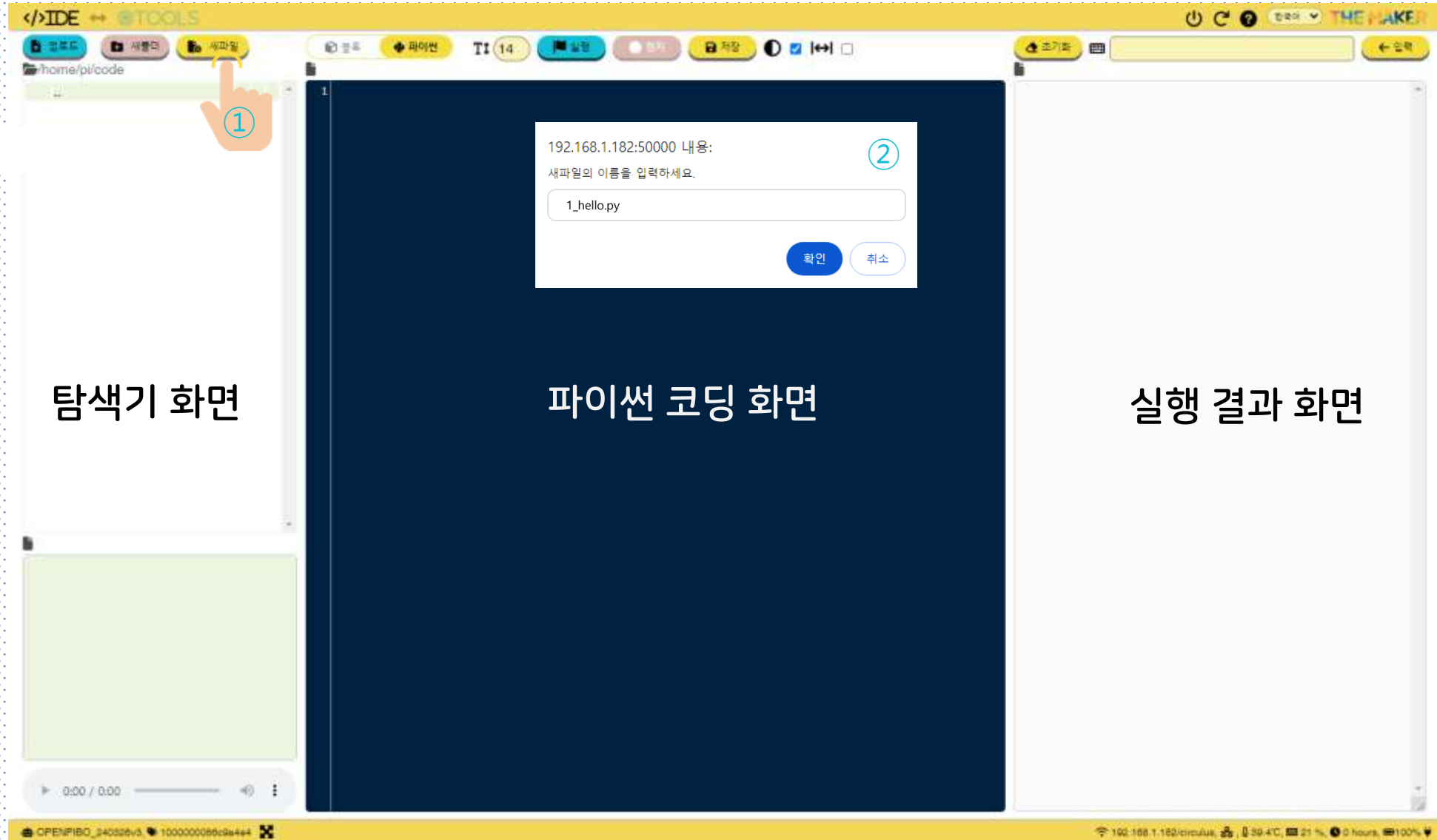
화면에 "안녕"을
출력해요!



- 파이보에게 명령을 내리기 위한 새 파일 생성



- 파이보에게 명령을 내리기 위한 새 파일 생성



- 파이보 메이커 IDE 출력창에 “안녕”을 출력합니다.

print() 안의 문자, 숫자 등을 출력창에 출력

```
print('안녕')  
  
print(123)  
  
print(2024, '년', 5, '월')
```


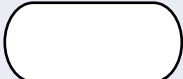

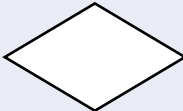


```
안녕  
123  
2024 년 5 월  
  
종료됨.
```

- 알고리즘

- 문제를 해결하기 위해 수행해야 할 과제를 순서에 맞게 나열한 절차

- 순서도

- 프로세스 과정을 순차적으로 표현한 흐름도

형태	명칭	설명
	흐름선	프로세스의 실행 순서를 나타낸다.
	터미널	하위 프로세스나 프로그램의 시작과 끝을 나타낸다.
	처리	데이터의 값, 형태, 장소를 변경하는 한 세트의 실행을 표현한다.
	판단	프로그램이 실행되는 두 가지 경로 중에 하나를 결정하는 조건부 실행을 나타낸다. 예/아니오 또는 참/거짓을 표현한다.
	입력/출력	데이터를 입력하거나 결과를 출력하는 경우와 같이 데이터의 입력과 출력을 나타낸다.
	출력	입력한 값의 출력을 나타낸다.

실습활동

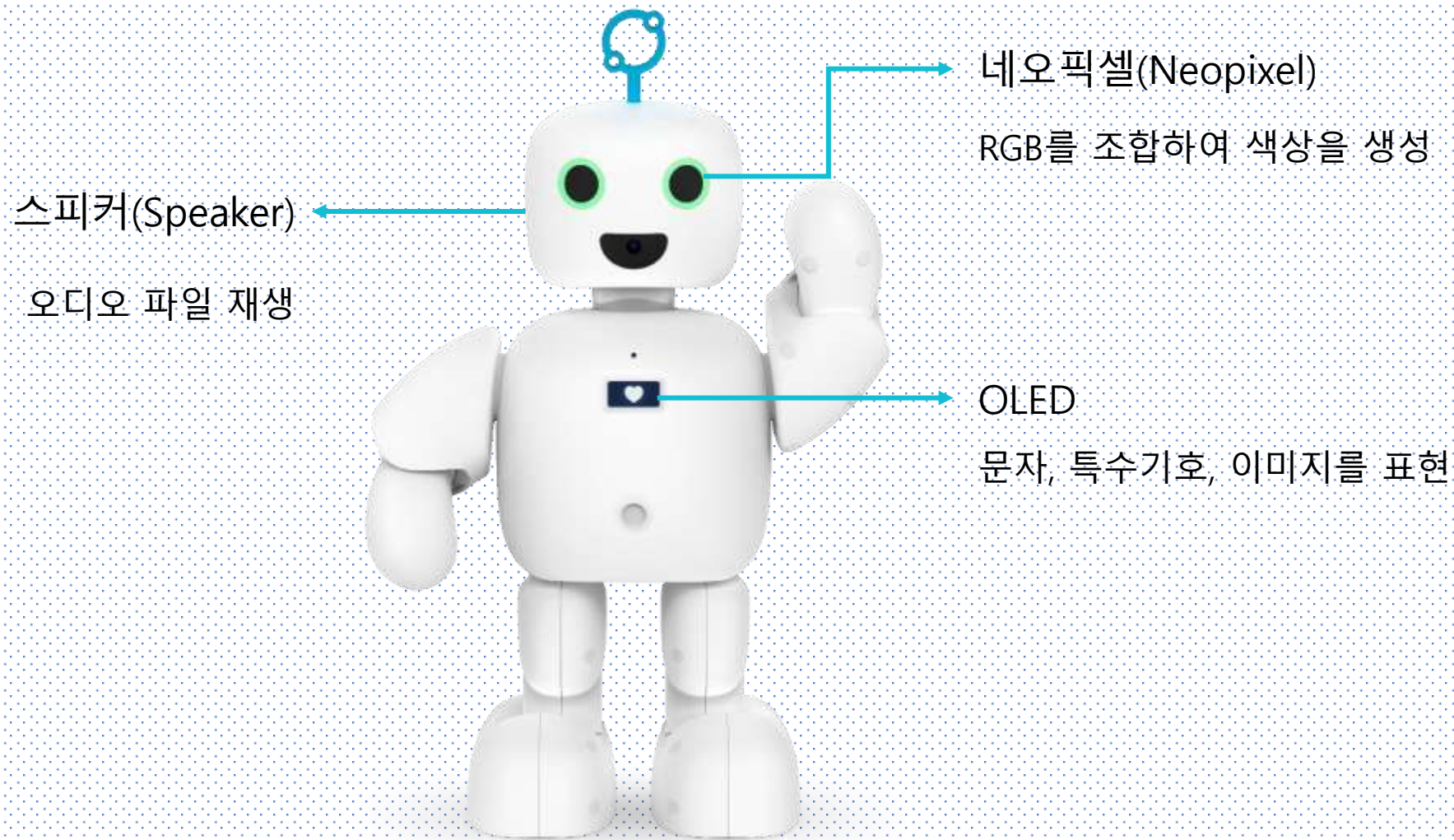
00000 과정을
순서대로 완성해요!



나를 소개하는 파이보

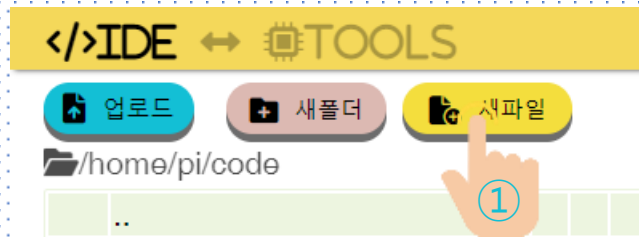


- 로봇의 전자부품을 제어하여 나만의 파이보를 만들 수 있어요!



- 처음 만난 친구들에게 파이보를 활용해 나를 소개합니다.
 - 네오픽셀 : 내가 좋아하는 색상을 표시해요.
 - OLED : 내 이름 또는 별명을 표시해요.
 - 스피커 : 파이보의 음성으로 내 소개를 해요.

- 새 파일을 생성합니다.

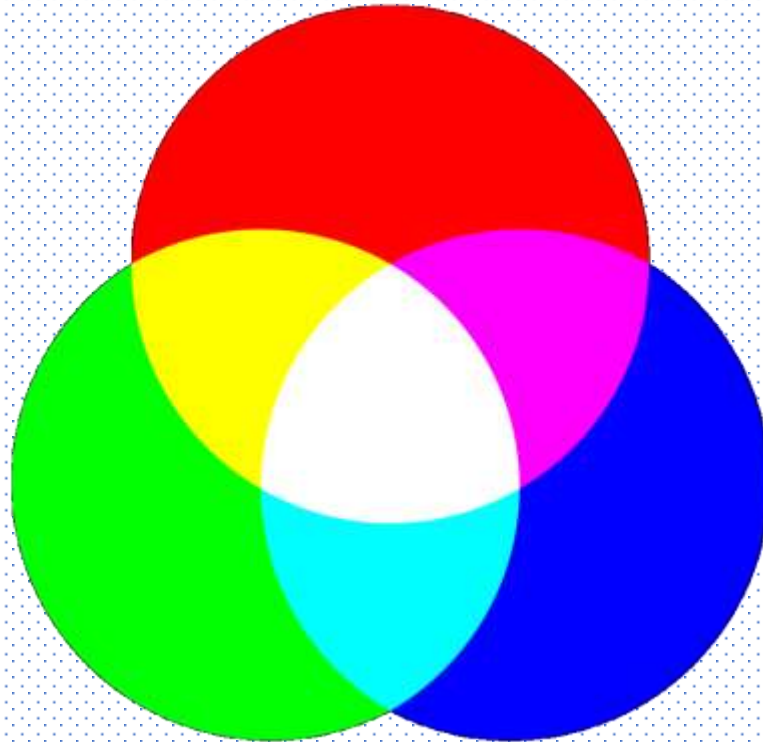


192.168.1.182:50000 내용: ②

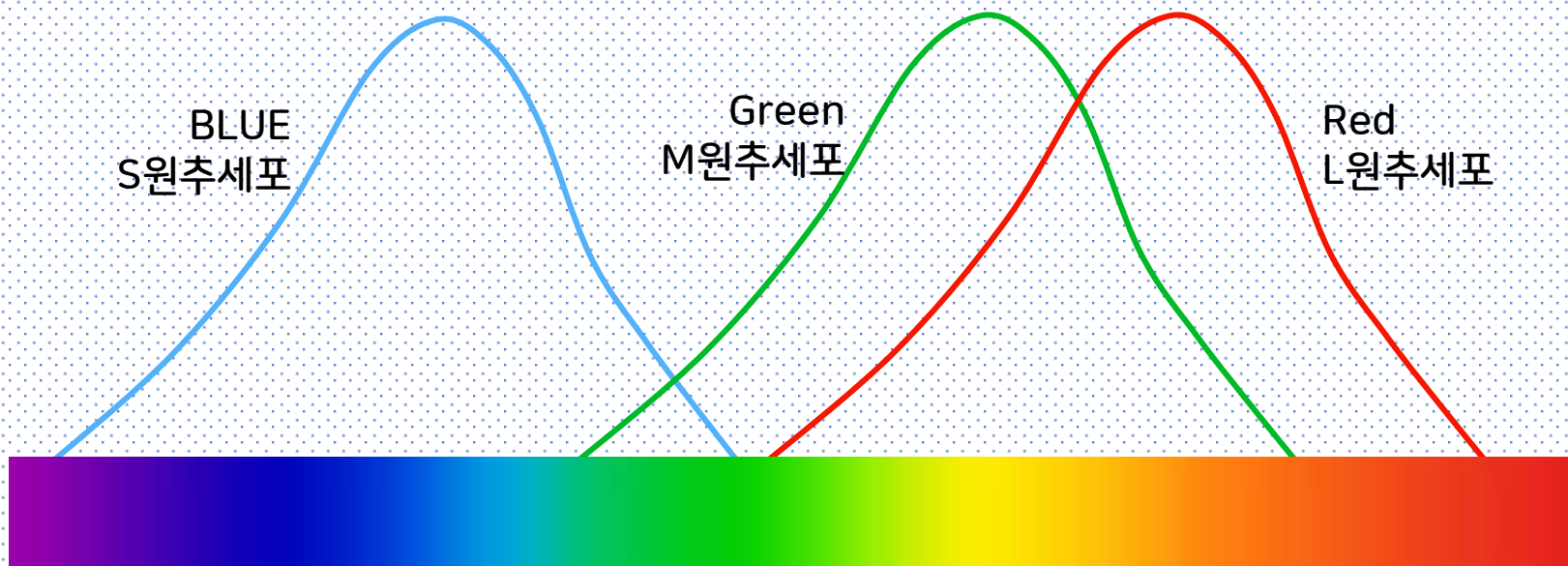
새파일의 이름을 입력하세요.

확인 취소

- 파이보의 눈 색상을 만들어요.
 - 빛의 삼원색(Red, Green, Blue)을 활용합니다.
 - 빛의 삼원색 합성해보기



- 왜 빨강, 초록, 파랑일까요?
 - 빛의 삼원색은 사람이 느낄 수 있는 세 가지 색상
 - 색감을 감지하는 인간의 원추세포가 RGB로 구성되어 있음



- 파이보의 눈 색상을 만들어요.

```
from openpibo.device import Device  
device = Device()  
device.eyе_on(255,255,255)
```

} 파이보의 전자부품 제어를 위한 초기 설정

→ 눈 색상 켜기

- 파이보의 눈 색상을 만들어요.

```
device.eye_on(255,255,255)
```

```
device.eye_on(255,255,255,0,0,0)
```

```
device.eye_off()
```

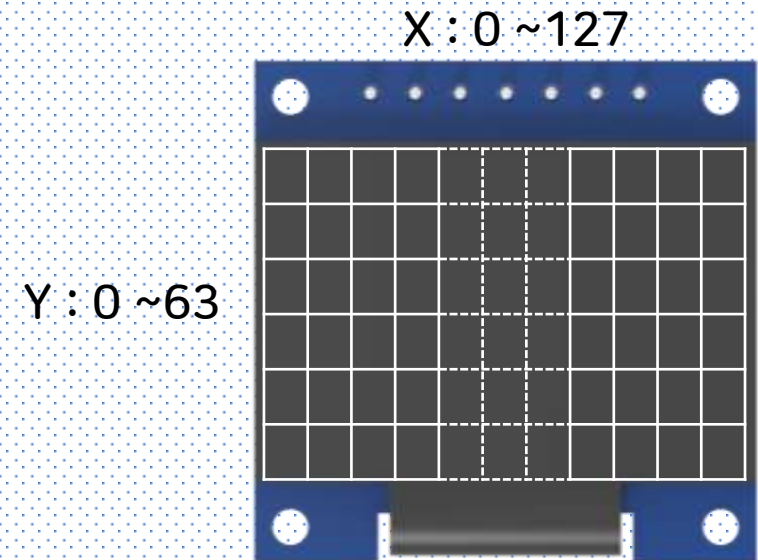
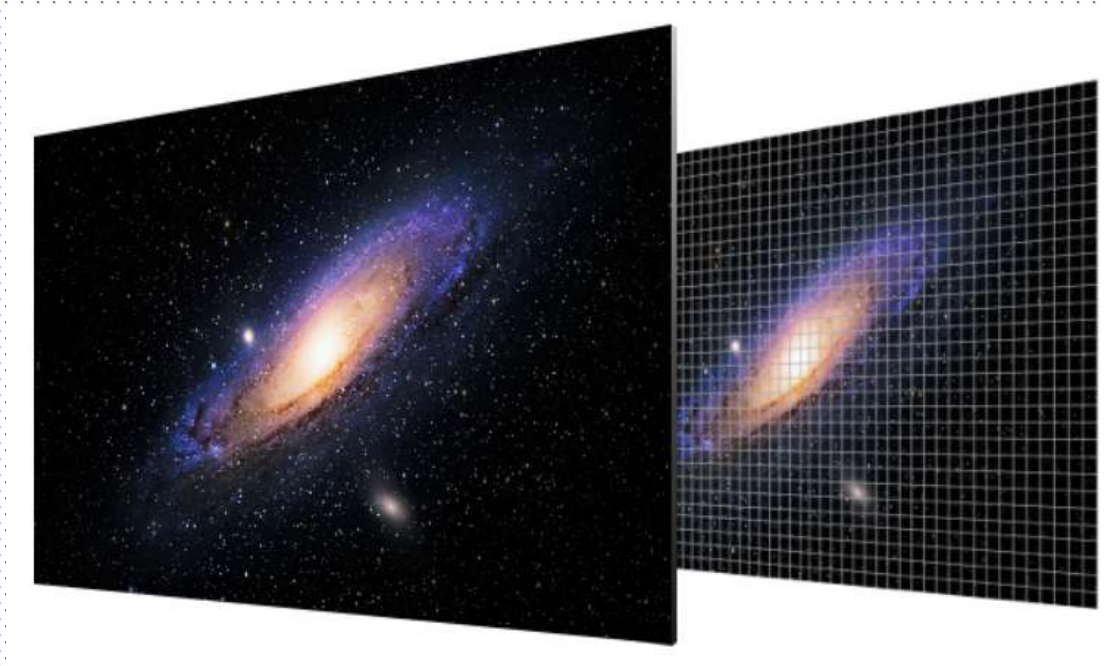
R, G, B를 조합하여 눈 색상을 만들 수 있습니다.

R, G, B를 조합하여 좌, 우 눈 색상을 만들 수 있습니다.

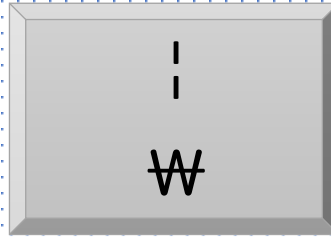
눈 색상을 끕니다.

```
== device.eye_on(0,0,0)
```

- OLED에 내 이름(별명)을 적어요.
 - OLED는 전류가 흐르면 스스로 빛을 내는 전자회로 소자
 - 픽셀(Pixel, 화면을 구성하는 단위) 각각의 밝기 제어 가능
 - 문자(한글, 영어, 숫자, 특수기호), 이미지, 도형 표시 가능



- OLED에 내 이름(별명)을 적어요.



★ 줄 바꾸기와 띄어쓰기로
글씨 위치를 조정할 수 있어요! ★

파이보WnHi



파이보Wn Hi



- OLED에 내 이름(별명)을 적어요.

```
from openpibo.oled import Oled  
  
oled = Oled()  
  
oled.set_font(size=20)  
oled.draw_text((0,0), '안녕하세요')  
oled.show()
```

파이보의 OLED 제어를 위한 초기 설정

파이보의 OLED 제어
(글씨 크기, 글씨 쓰기, 출력하기)

- OLED에 내 이름(별명)을 적어요.

```
oled.set_font(size=20)
```

```
oled.draw_text((0,0), '안녕하세요')
```

```
oled.show()
```



OLED에 표시할 문자 크기를 설정합니다.

OLED에 표시할 문자의 시작 위치와 내용을 입력합니다.

입력한 위치와 내용을 OLED에 표시합니다.

★ show()를 사용하지 않으면
입력한 내용을 출력하지 않습니다.

- OLED에 이미지를 출력해요.

```
oled.draw_image('/home/pi/openpibo-files/image/expression/smile.jpg')
```

OLED에 표시할 이미지를 준비합니다.

```
oled.show()
```

입력한 위치와 내용을 OLED에 표시합니다.

★ show()를 사용하지 않으면
입력한 내용을 출력하지 않습니다.



* 이미지 파일

1. IDE의 파일 탐색기의 "/home/pi/openpibo-files/image" 확인
2. 개별 업로드 (jpg, png)

- 스피커를 통해 내 소개를 말해요.

```
from openpibo.speech import Speech  
from openpibo.audio import Audio
```

```
speech = Speech()  
audio = Audio()
```

파이보의 음성합성, 오디오 출력을 위한 초기 설정

```
speech.tts(string='안녕하세요, 파이보입니다.', filename='voice.mp3', voice='main')  
audio.play('voice.mp3', 80)
```

음성합성 및 오디오 출력하기

- 스피커를 통해 내 소개를 말해요. – 음성 합성하기

```
speech.tts(string='안녕하세요, 파이보입니다.', filename='voice.mp3', voice='main')
```

↓
생성할 음성 내용을 입력합니다.

↓
음성을 저장할 위치와
파일 이름을 입력합니다.
★ 파일 입력만 입력한 경우
실행한 파일과 같은 위치에 저장합니다.

↓
음성 목소리를
선택합니다.

★ espeak, gtts, main, boy, girl, man1, woman1

- 스피커를 통해 내 소개를 말해요. – 오디오 출력하기

```
audio.play('voice.mp3', 80)
```

출력할 오디오 파일 경로와 이름, 볼륨 크기를 입력합니다.

★ 오디오 파일 경로를 입력하지 않으면
실행한 파일과 같은 위치에 있는 파일을 실행합니다.

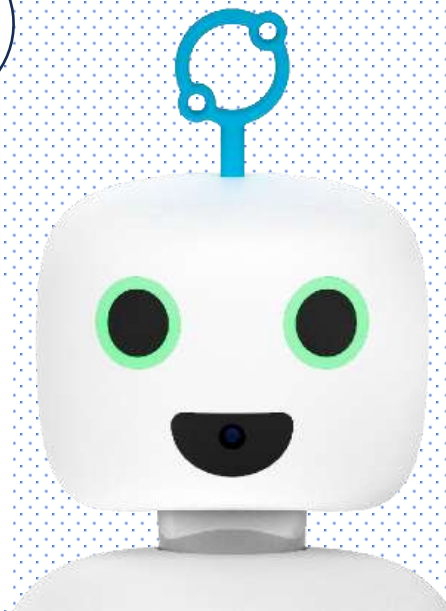
AI 휴머노이드 로봇 비서 만들기

Part 1. 아침을 깨우는 파이보



- 내 하루를 관리하는 AI 휴머노이드 로봇 비서 파이보

파이보의 알람으로
상쾌한 하루를 시작해볼까?



- 오늘의 시간 정보(연월일, 시분초 단위)를 확인해요!

```
print(time.strftime('%Y-%m-%d %H:%M:%S'))  
time_list = time.strftime('%Y,%m,%d,%H,%M,%S').split(',')  
print(time_list)
```

2024-05-13 10:01:58

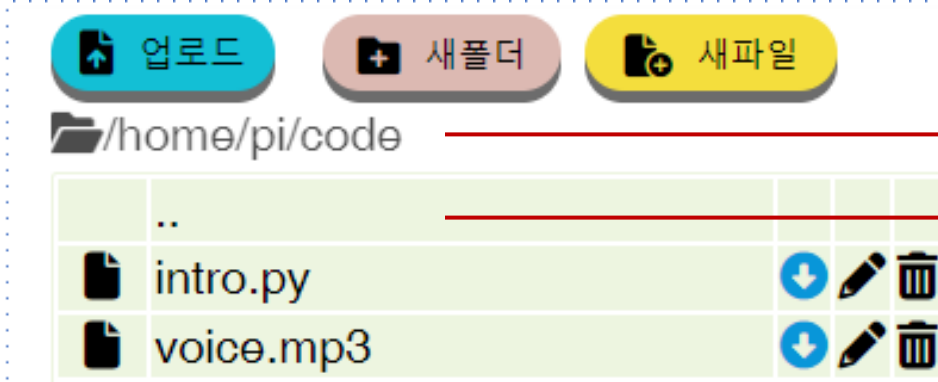
['2024', '05', '13', '10', '01', '58']

종료됨.

'연 월 일 시 분 초'를 필요한 형식의 문자로 출력
split 함수로 문자를 리스트로 변환하고 각각 항목을 가져올 수도 있어요

★ 파이보는 인터넷에 연결되어 있어야 정확한 시간 정보를 알 수 있어요!

- 파이보에서 사용할 수 있는 효과음을 찾아봅니다.

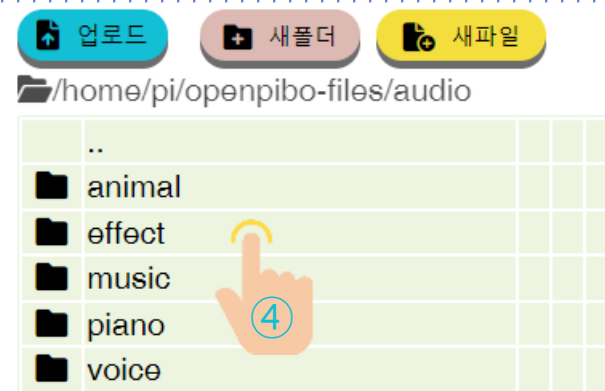
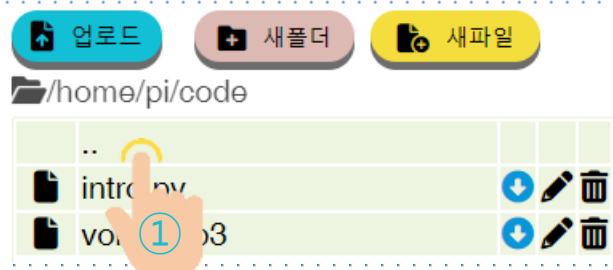


현재 위치(폴더)를 알 수 있습니다.

한 단계 위(폴더)로 이동할 수 있습니다.

현재 위치(폴더)에 저장한
파일, 폴더를 알 수 있습니다.

- 파이보에서 사용할 수 있는 효과음을 찾아봅니다.



- 파이보 알람 시계를 완성해요!

초기화

...

반복:

현재 시간 가져오기

00초 인 경우, (분 단위로 실행)
눈 색상 변경

"00시 00분 입니다." 음성 알림

연월일 시분초 OLED에 표시
1초 간격
OLED 초기화

★ 체크!

```
time_list = time.strftime('%Y,%m,%d,%H,%M,%S').split(',')
```

```
if time_list[5] == '00': # 0초 일 때만 체크 > 분 단위  
    print(f'{time_list[3]}시 {time_list[4]}분 입니다.')
```



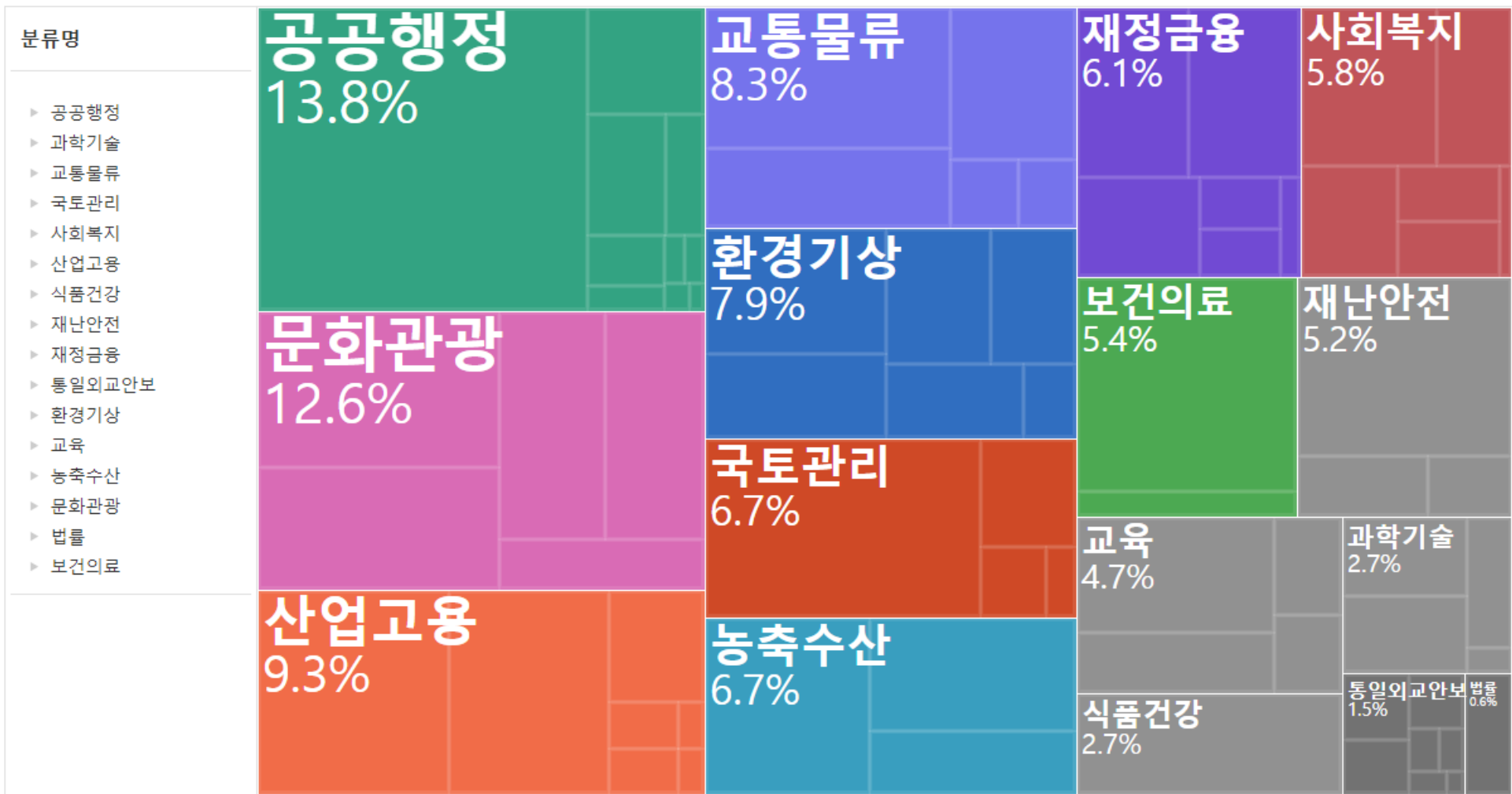
AI 휴머노이드 로봇 비서 만들기

Part 2. 날씨/뉴스 알려주는 파이보



- 데이터(Data)에 대해 알아봅니다.
 - ☐ 이론을 세우는 데 기초가 되는 사실 또는 바탕이 되는 자료
 - ☐ 관찰이나 실험, 조사로 얻은 사실이나 자료
 - ☒ 컴퓨터가 처리할 수 있는 문자, 숫자, 소리, 그림 따위의 형태로 된 자료

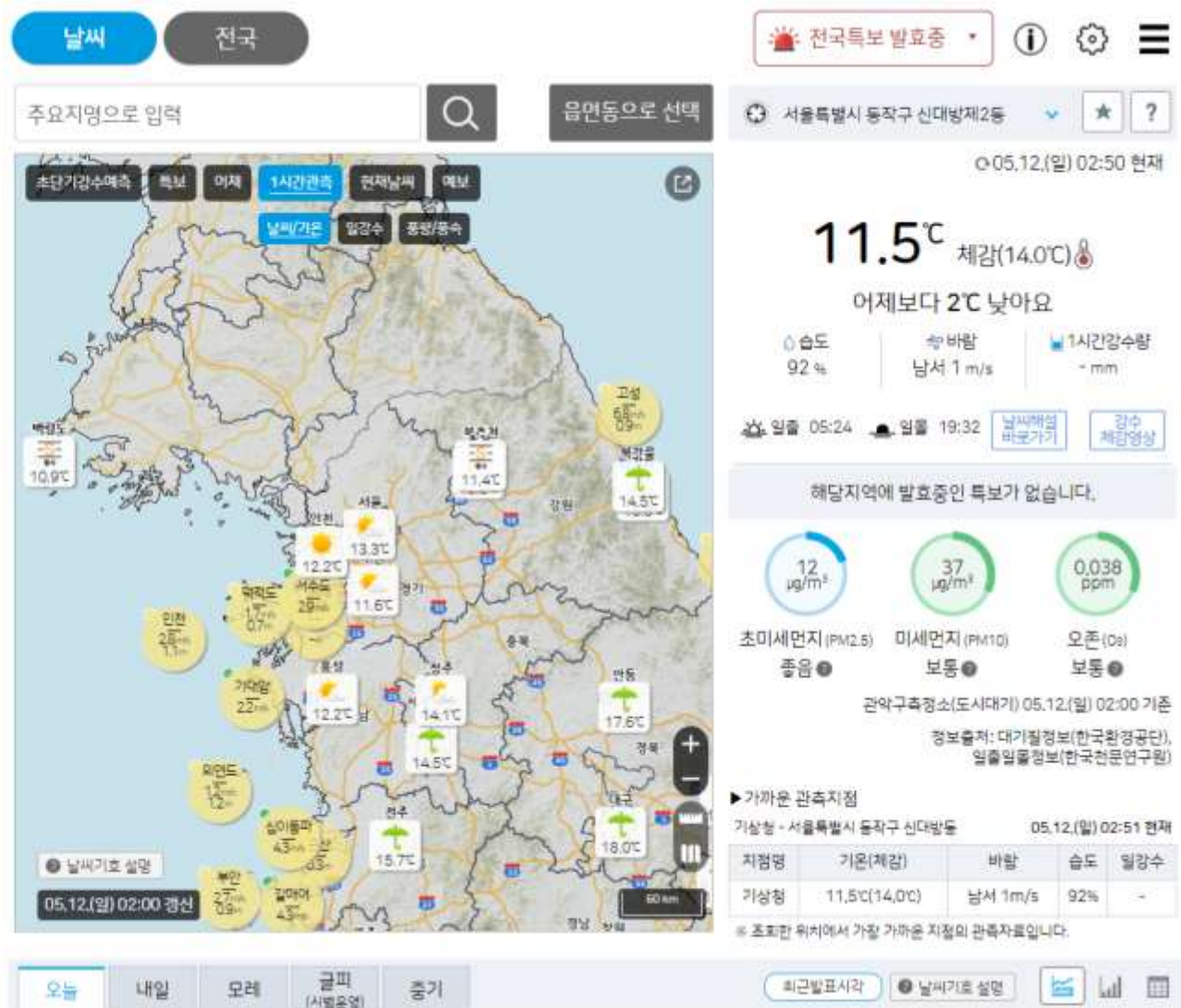
- 일상에서 활용하는 데이터가 무엇이 있을까요?



- 공공데이터에 대해 알아봅니다.
 - 공공기관이 생성 또는 취득하여 관리하고 있는 데이터
 - 누구든지 공공데이터를 이용할 수 있음
 - 공공데이터포털이 제공 중인 데이터는 별도의 신청없이 이용 가능
 - 공공데이터포털에서 제공하고 있지 않은 데이터의 경우 제공신청 필요
 - 공공데이터를 활용해 실제 서비스를 운영할 경우 공공누리 유형에 따른 저작권 부착 필요



- 날씨 데이터를 확인합니다.
- 기상청 날씨누리



- 날씨 데이터를 확인합니다.
 - 기상청에서 제공하는 날씨 예보를 시각화로 보기 쉽게 표시
 - 지역별 단기예보부터 장기전망까지 확인 가능
 - 날씨 외 미세먼지, 황사, 지진, 태풍 등 날씨 관련 정보 확인 가능

- 지역별 날씨를 검색합니다.
 - 설정한 지역 기준 날씨 정보를 시각화로 표시

기상청 날씨누리

날씨

기상특보 +

예보 -

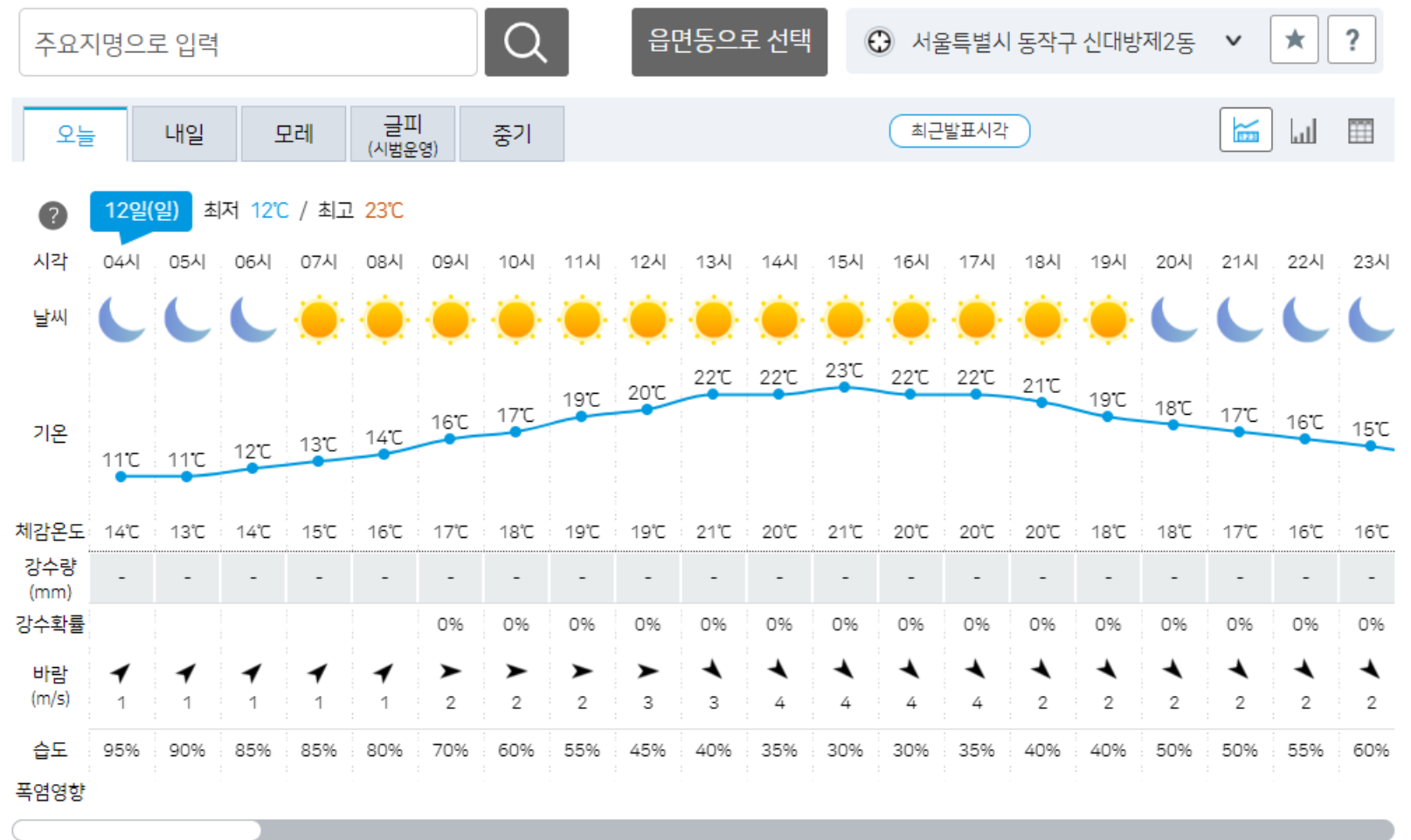
단기예보

중기예보

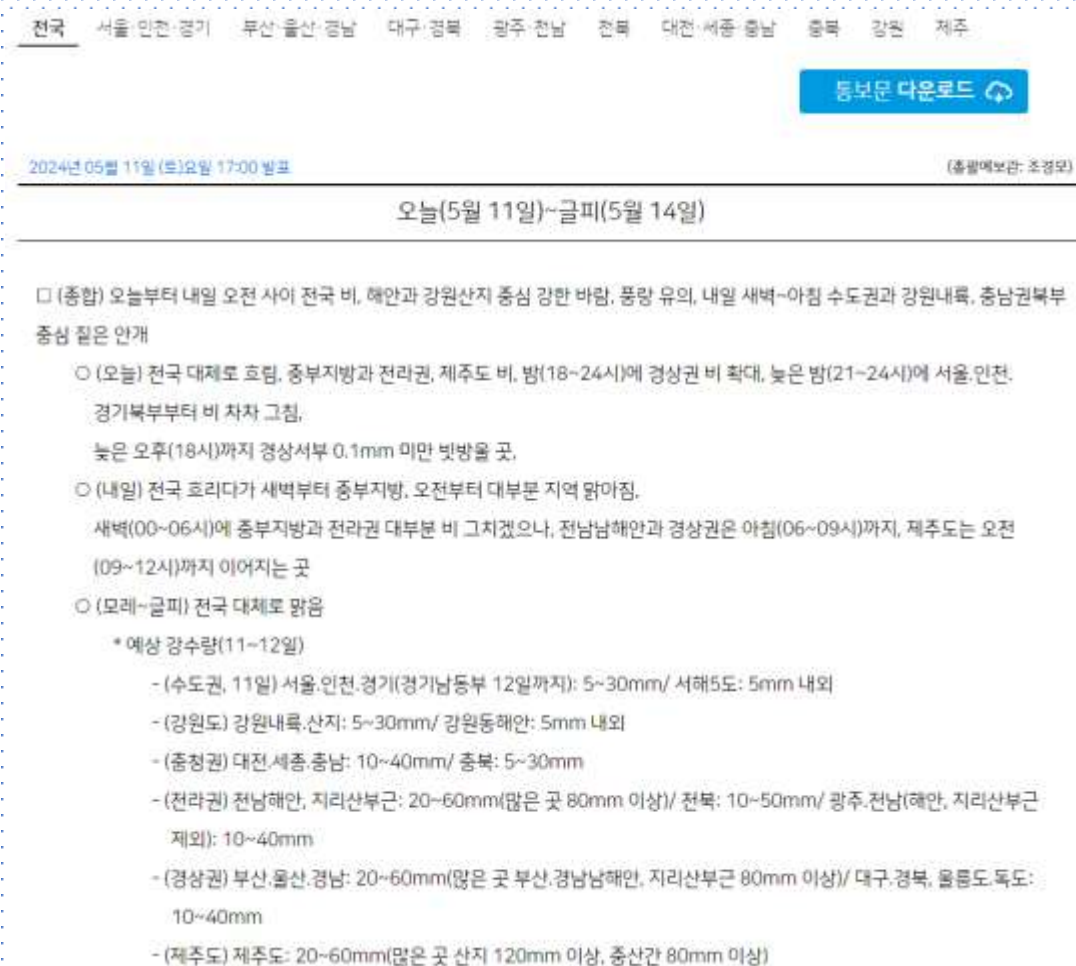
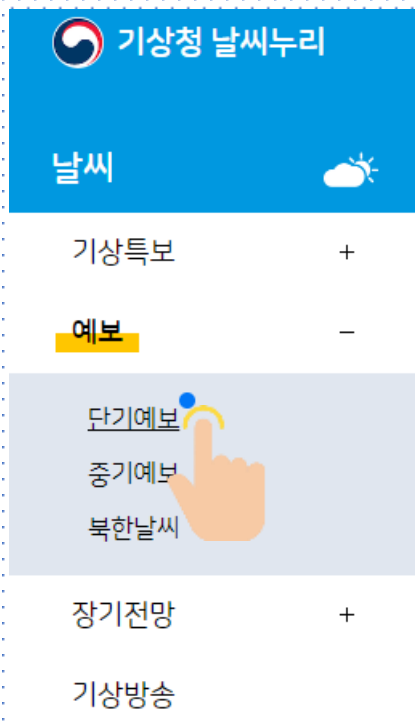
북한날씨

장기전망 +

기상방송



- 지역별 날씨를 검색합니다.
- 선택한 지역 기준 날씨 정보를 문자로 표시



- 지역별 날씨를 검색합니다.

```
from openpibo.collect import Weather  
weather = Weather()  
  
result = weather.search('서울')  
print(result)
```

파이보의 날씨 정보 수집을 위한 초기 설정

지역에 따른 날씨 정보 수집 및 출력

- 지역별 날씨를 검색합니다. – 예보 전체

```
result = weather.search('서울')
```

입력한 지역 날씨 정보를 수집합니다.

★ 전국, 서울, 인천, 경기, 부산, 울산, 경남, 대구, 경북, 광주, 전남, 전북, 대전, 세종, 충남, 충북, 강원, 제주

```
print(result)
```

수집한 날씨 정보를 파이보 메이커 화면에 출력합니다.

📄 /home/pi/code/weather.py

[Sun May 12 2024 03:20:10 GMT+0900 (대한민국 표준시)]:

```
{'forecast': '오늘 비, 경기남동부 내일 새벽까지 비 이어지는 곳, 오늘 강풍과 풍랑 유의, 내일 새벽~아침 짙은 안개 유의', 'today': {'weather': '대체로 흐림, 비 내리다 늦은 밤(21~24시)에 서울.인천.경기북부부터 차차  
그침 \n          서해5도 대체로 흐리다 차차 맑아짐, 늦은 오후(~18시)까지 비', 'minimum_temp': '7.8 ~ 15.4', 'highst_temp': '20.5 ~ 26.3'}, 'tomorrow': {'weather': '대체로 맑음\n* 예상 강수량(11일)- 서울.인천.  
경기도(경기남동부 12일 새벽까지): 5~30mm- 서해5도: 5mm 내외', 'minimum_temp': '9 ~ 12', 'highst_temp': '19.4 ~ 24.7'}, 'after_tomorrow': {'weather': '대체로 맑음\n* 예상 강수량(11일)- 서울.인천.경기도  
(경기남동부 12일 새벽까지): 5~30mm- 서해5도: 5mm 내외', 'minimum_temp': '7 ~ 12', 'highst_temp': '20 ~ 24'}}
```

종료됨.

- 지역별 날씨를 검색합니다. – 오늘 날씨만

```
result = weather.search('서울')
```

입력한 지역 날씨 정보를 수집합니다.

★ 전국, 서울, 인천, 경기, 부산, 울산, 경남, 대구, 경북, 광주, 전남, 전북, 대전, 세종, 충남, 충북, 강원, 제주

```
print(result['today'])
```

수집한 날씨 정보를 파이보 메이커 화면에 출력합니다.

📄 /home/pi/code/weather.py

[Sun May 12 2024 03:21:25 GMT+0900 (대한민국 표준시)]:

```
{'weather': '대체로 흐림, 비 내리다 늦은 밤(21~24시)에 서울.인천.경기북부부터 차차 그침 \n'20.5 ~ 26.3'}
```

종료됨.

서해5도 대체로 흐리다 차차 맑아짐, 늦은 오후(~18시)까지 비, 'minimum_temp': '7.8 ~ 15.4', 'highst_temp':

- 뉴스 데이터를 확인합니다.
 - RSS는 미디어 배포 뉴스나 블로그 사이트에서 주로 사용하는 콘텐츠 표현 방식
 - 웹사이트에 직접 방문하지 않아도 업데이트 되는 정보/뉴스를 쉽게 접근할 수 있음
 - 파이썬을 이용하여, 최신 뉴스를 RSS 형식으로 읽어올 수 있고 가공할 수 있음



- 주제 별 뉴스를 검색합니다. - 전체

```
from openpibo.collect import News  
  
News = News()  
  
result = news.search('속보')  
print(result)
```

파이보의 뉴스 정보 수집을 위한 초기 설정

종류 별 뉴스를 수집하고, 출력

★ 뉴스 수집 결과

종류: '속보', '정치', '경제', '사회', '국제', '문화', '연예', '스포츠', '뉴스랭킹'

데이터 구조:

```
[  
  뉴스1: {title(제목), link(링크), description(내용), pubDate(날짜)}  
  뉴스2  
  ...  
  뉴스n  
]
```

- 주제 별 뉴스를 검색합니다. – 개별

```
result = news.search('경제')
```

‘경제’ 뉴스 정보를 수집합니다.

[Mon May 13 2024 11:00:01 GMT+0900 (대한민국 표준시)]:

```
{'title': '"기후플레이션' 현실로... 최악 가뭄에 올리브유 가격 치솟아', 'link': 'https://news.jtbc.co.kr/article/article.aspx?news_id=NB12195793', 'description': '[앵커]코코아, 김에 이어 이번엔 올리브유 가격도  
샘표는 이달 초 대형마트에서 판매하는 올리브유 제품 가격을 각각 30% 넘게 인상했습니다. 올리브 최대 생산국인 스페인이', 'pubDate': '2024.05.13'}, {'title': '1년째 오르는 서울 아파트 전셋값...역대 최고  
'https://news.jtbc.co.kr/article/article.aspx?news_id=NB12195751', 'description': '[앵커]서울의 아파트 전셋값이 1년째 상승하면서 역대 최고가의 84% 수준까지 도달했습니다. 매매가도 덩달아 7주 연속  
기자입니다.[기자]서울 아파트 전셋값이 1년째 오르', 'pubDate': '2024.05.12'}, {'title': '주유소 기름값 주춤...휘발유 7주 만에 하락세', 'link': 'https://news.jtbc.co.kr/article/article.aspx?news_id=NB12195704',  
가격이 함께 떨어졌습니다.한국석유공사 유가정보시스템에 따르면 이번 주 전국 주유소 휘발유 평균 판매가는 전주 대비 리터당 1.2원 내린 1711.7원으로 7주 만에 내림세로 돌아섰습니', 'pubDate': '2024  
표명했는데...네이버 '매각 가능성' 시사', 'link': 'https://news.jtbc.co.kr/article/article.aspx?news_id=NB12195644', 'description': '[앵커]일본 정부와 기업이 네이버가 키워 온 일본 국민 메신저 라인을 뺏으  
태양전지'가 이집트 발전소 건설에 사용될 예정'이라고 밝혔다. 이번 프로젝트는 이집트 신왕국 시대 유적 발굴을 위한 것으로, 태양전지 패널은 유적 발굴을 위한 조명과 전력 공급에 사용될 예정이다. 이번 프로젝트는 이집트 신왕국 시대 유적 발굴을 위한 것으로, 태양전지 패널은 유적 발굴을 위한 조명과 전력 공급에 사용될 예정이다.
```

```
print(result[0]['title'])
```

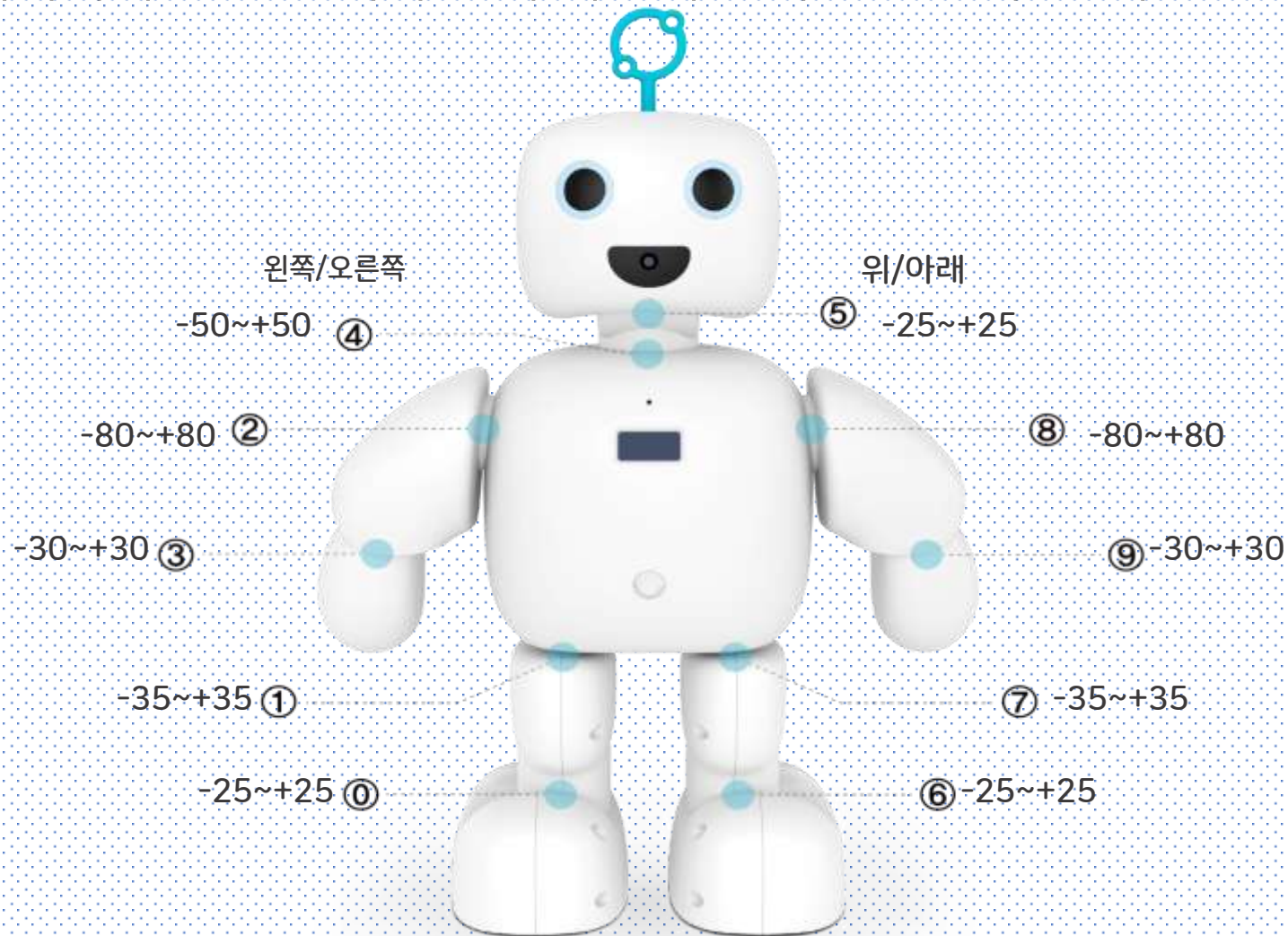
첫번째 뉴스의 제목 출력

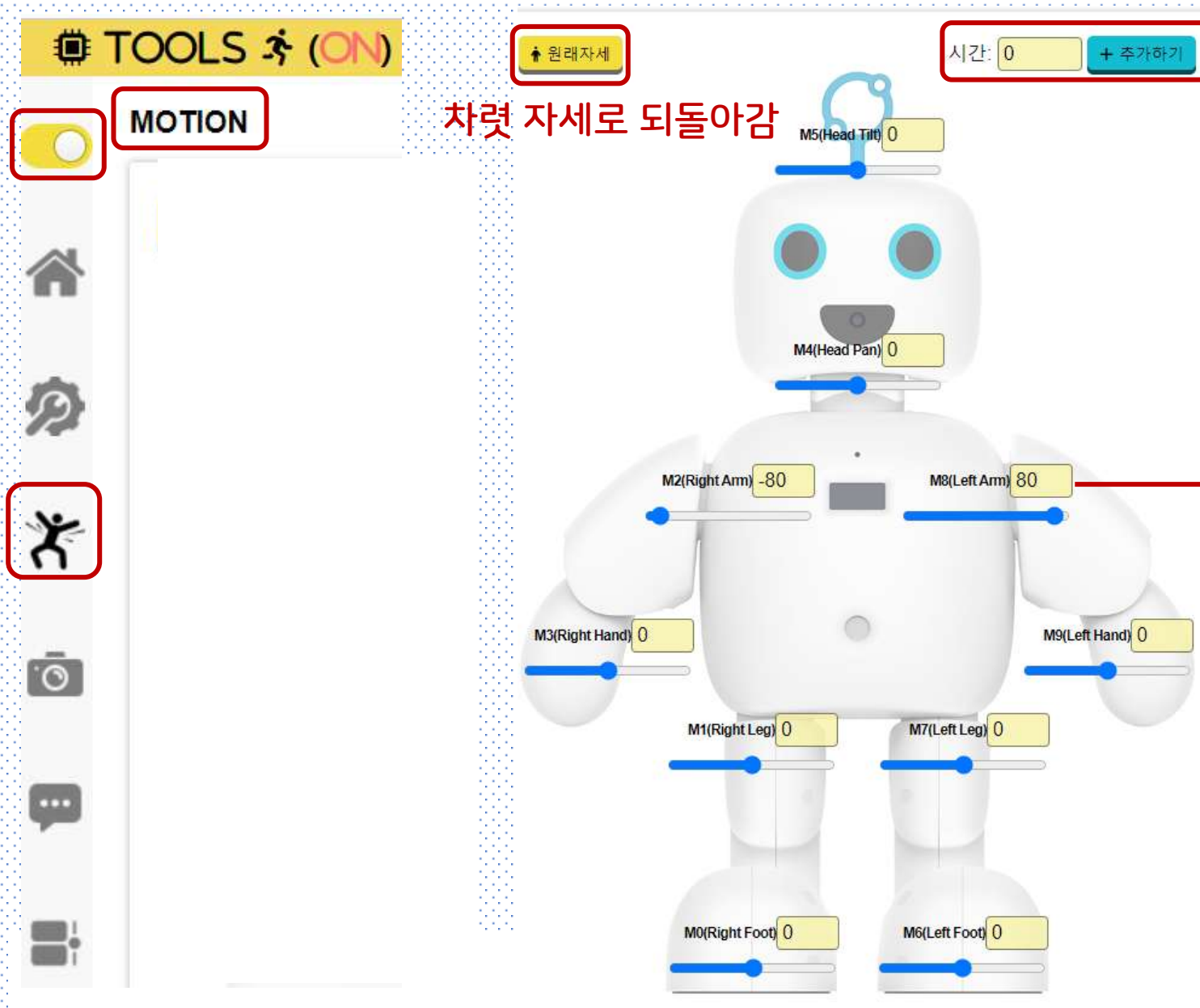
[Mon May 13 2024 10:58:44 GMT+0900 (대한민국 표준시)]:

```
'기후플레이션' 현실로...최악 가뭄에 올리브유 가격 치솟아
```

종료됨.

- 간단한 모션과 함께 하는 기상캐스터를 만듭니다.
- 파이보 메이커의 모션 툴에서 쉽게 나만의 동작을 만들 수 있어요.





2 서보모터 이동을 완료할
시간 설정

1 연결된 서보모터의 슬라이더를
움직여 동작을 먼저 설정

1,2 의 과정을 반복하여
원하는 동작을 만듦

TOOLS (ON) </>IDE

한국어 THE MAKER

MOTION

원래자세 시간: 0 + 추가하기

반복: 1

▶ 실행하기

■ 정지하기

☒ 모두 지우기

3

동작 반복 횟수 설정 후 실행

시간	M0	M1	M2	M3	M4	M5	M6	M7	M8	M9
1 초	0	0	-80	0	0	-15	0	0	80	0
2 초	0	0	-80	0	21	15	0	0	80	0
2.5 초	0	0	-80	0	-38	0	0	0	80	0
3 초	0	0	0	0	-38	0	0	0	0	0
5 초	0	0	0	30	-38	-20	0	0	0	-30

4

영어로 나만의 모션이름 정하기!

모션이름: nmotion

✓ 등록하기

↓ 불러오기

- 삭제하기

☒ 모두 지우기

📄 내보내기

📁 가져오기

원하는 모션 예제 클릭, 수정하여 사용 가능, 모션 예제명을 적어 불러오기 후 실행 가능

예제: forward, backward, left, right, welcome, foot, happy, sad, clapping, wave1, wave2, dance1, dance2

원래자세

시간: 0

+ 추가하기

M5(Head Tilt) 0

M4(Head Pan) 0

M2(Right Arm) -80

M6(Left Arm) 80

M3(Right Hand) 0

M0(Left Hand) 0

M1(Right Leg) 0

M7(Left Leg) 0

M0(Right Foot) 0

M0(Left Foot) 0

더블 클릭하면 원하는 한 줄만 삭제 가능

- 날씨를 알려주는 동작을 만들고 실행합니다.

```
from openpibo.motion import Motion  
  
motion = Motion()  
  
motion.set_mymotion('weather', 3)  
motion.stop()
```

파이보의 모션 제어를 위한 초기 설정

내가 만든 모션을 실행

- 날씨/뉴스를 알려주는 동작을 만들고 실행합니다.

```
motion.set_mymotion('weather', 3)  
motion.set_mymotion('news', 3)
```

내가 만든 모션 중 실행할 모션의 이름과 반복 횟수를 입력합니다.

```
motion.stop()
```

모션을 정지하고 차렷 자세로 변경합니다.

- 파이보에 내장된 모션을 사용할 수 있어요.

```
motion.set_motion('clapping1', 1)
```

반가움

clapping1, clapping2,
handshaking, bow,
greeting, welcome

머리 움직이기

head_h, spin_h,
yes_h, no_h

신남

wave1, wave2, wave3,
wave4, wave5, wave6,
happy1, happy2,
happy3, excite1,
excite2

침착함

stop, lookup, sleep,
breath1, breath2,
breath3

걸기

forward1,
forward2,
backward1,
backward2

리액션

cheer1, cheer2,
cheer3, think1,
wake_up1, wake_up2,
hand1, hand2, hand3,
hand4, handup_r,
handup_l, look_r,
look_l, speak1,
speak2

댄스

dance1, dance2,
dance3, dance4,
dance5

지루함

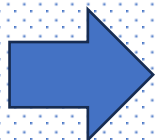
boring1, boring2,
sad1, sad2, sad3

다리 움직이기

left, left_half, right,
right_half, foot1,
foot2

- 날씨 캐스터를 만들고 실행합니다.

```
1 from openpibo.collect import Weather
2 from openpibo.device import Device
3 from openpibo.oled import Oled
4 from openpibo.speech import Speech
5 from openpibo.audio import Audio
6 from openpibo.motion import Motion
7
8 device = Device()
9 oled = Oled()
10 speech = Speech()
11 audio = Audio()
12 weather = Weather()
13 motion = Motion()
14
15 # 서울 오늘 날씨
16 result = weather.search('서울')['today']
17 print(result)
```



```
19 # 눈 색상 표시하기
20 device.eye_on(0,255,255)
21
22 # 디스플레이에 최저/최고 기온 표시
23 oled.set_font(size=20)
24 oled.draw_text((0,0), '# 날씨')
25 oled.set_font(size=15)
26 oled.draw_text((0,25), result['minimum_temp'])
27 oled.draw_text((0,45), result['highst_temp'])
28 oled.show()
29
30 # 예보 음성 재생하기
31 speech.tts(string=result['weather'], filename='voice.mp3', voice='main')
32 audio.play('voice.mp3', 80)
33
34 # 동작 실행하기
35 # motion.set_mymotion('weather')
36 motion.set_motion('speak1')
37 motion.set_motion('stop')
```

- 뉴스 캐스터를 만들고 실행합니다.

```
1 from openpibo.collect import News
2 from openpibo.device import Device
3 from openpibo.oled import Oled
4 from openpibo.speech import Speech
5 from openpibo.audio import Audio
6 from openpibo.motion import Motion
7
8 device = Device()
9 oled = Oled()
10 speech = Speech()
11 audio = Audio()
12 news = News()
13 motion = Motion()
14
15 # '경제' 뉴스 첫번째 항목 가져오기
16 result = news.search('경제')[0]
17 print(result)
```



```
19 # 눈 색상 표시하기
20 device.eye_on(0,255,255)
21
22 # 디스플레이에 뉴스 업데이트 날짜 표시
23 oled.set_font(size=20)
24 oled.draw_text((0,0), '# 뉴스')
25 oled.set_font(size=15)
26 oled.draw_text((0,25), result['pubDate'])
27 oled.show()
28
29 # 뉴스 제목 음성 재생하기
30 speech.tts(string='뉴스입니다.' + result['title'], filename='voice.mp3', voice='main')
31 audio.play('voice.mp3', 80)
32
33 # 동작 실행하기
34 # motion.set_mymotion('news')
35 motion.set_motion('speak1')
36 motion.set_motion('stop')
```

AI 휴머노이드 로봇 비서 만들기

Part 3. 이미지를 인식하는 파이보

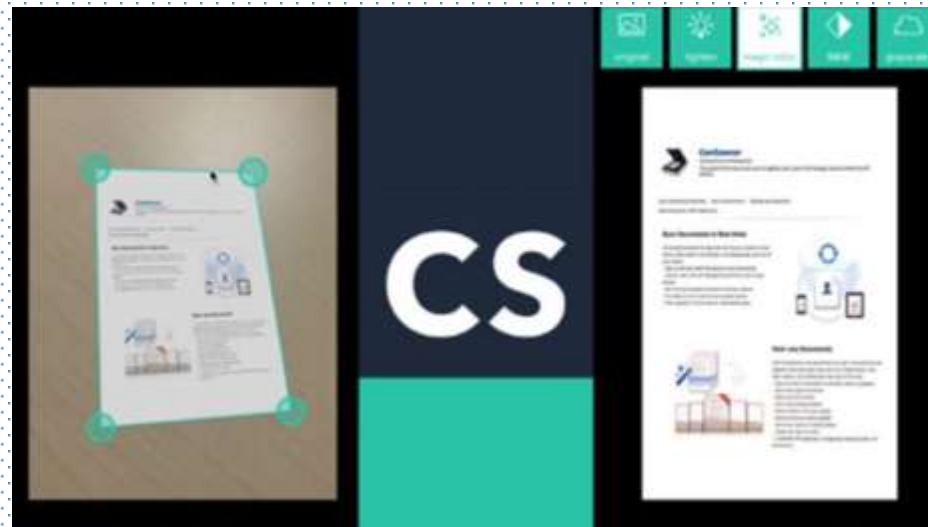


• 컴퓨터 비전

- 카메라를 통해 영상을 캡처하고 컴퓨터를 사용하여 의미 있는 정보를 추출하고 이해



9821480865132823066470938
4460955058223172535940812
8481117450284102701938521
1055596446229489549303819
6442881097566593344612847



- 컴퓨터 비전 활용 사례
자율주행 자동차



- TOOLS의 비전 메뉴에서 다양한 컴퓨터 비전을 체험할 수 있습니다.
- 토글이 'on' 상태인지 확인합니다.

VISION

카메라

사진저장하기 사진 10장 저장하기

카테고리

기능 설정

카테고리

티쳐블머신 모델 \$~/mymodel: 업로드

마커길이: 5 cm

결과

기능 설정

카테고리

카테고리

흑백

윤곽선

만화

스케치(흑백)

스케치(컬러)

선명도개선

흐리게

QR코드

얼굴분석

얼굴분석(랜드마크)

사물인식

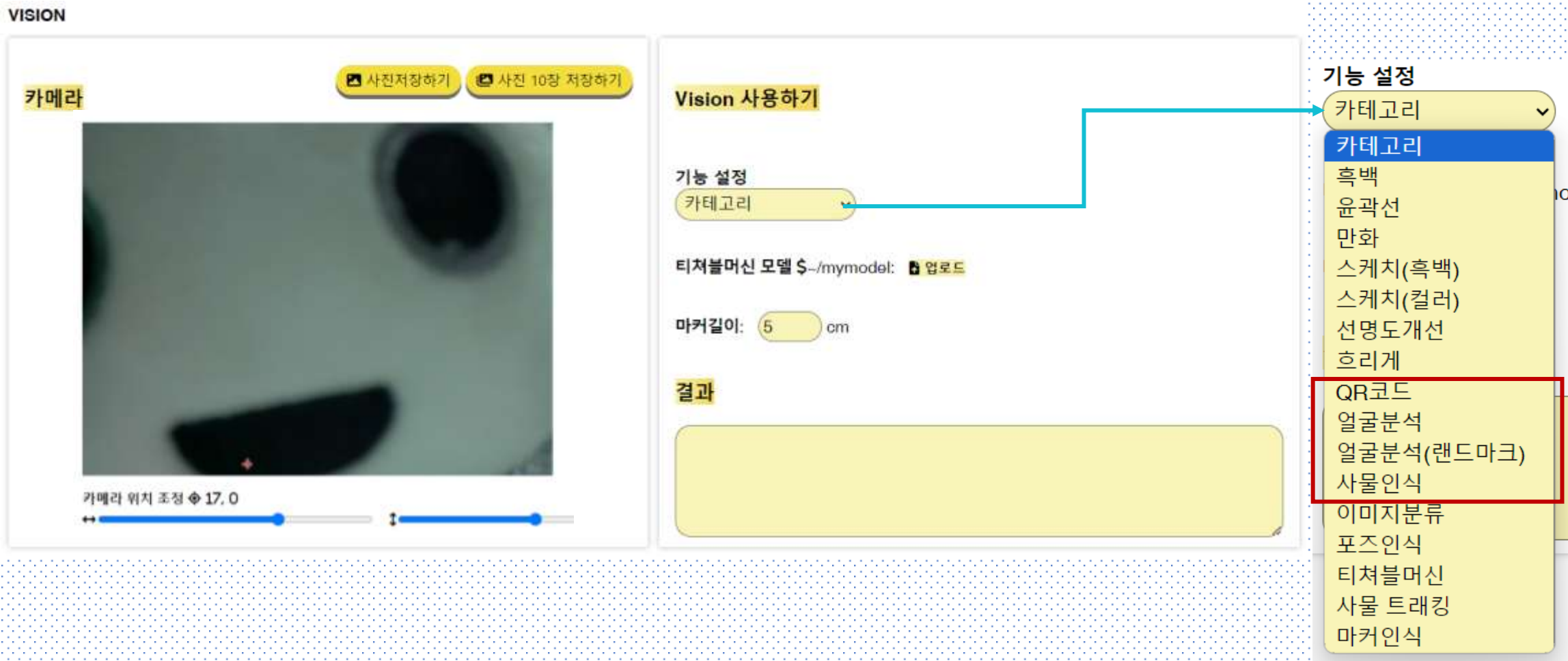
이미지분류

포즈인식

티쳐블머신

사물 트래킹

마커인식

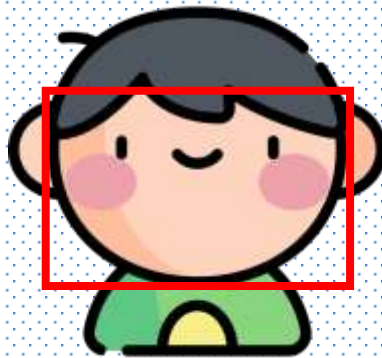


- 얼굴분석

- 이미지에서 얼굴을 검색
- 검색된 얼굴의 68개의 주요 포인트로 추출 (랜드마크)
- 랜드마크를 이용하여, 얼굴 구분, 표정, 눈 깜빡임 체크 등에 활용



1. 사진 입력



2. 얼굴 검출



3. 얼굴 랜드마크 추출

- 사물인식

- 이미지에서 사물을 인식하고, 확률/위치를 알아냅니다. (대표적인 80개 사물)



- QR코드 인식

- 바코드, QR코드를 인식하고, 데이터를 추출



- 이미지를 촬영하고, 가공해봅니다.

```
from openpibo.vision import Camera
```

```
camera = Camera()
```

```
image = camera.read() # 이미지 촬영하기
```

```
# 화면의 (100,100), (300,300) 위치에 (255,255,0) 색상, 두께 2의 사각형 그리기
```

```
image = camera.rectangle(image, (100,100), (300,300), (255,0,0), 2)
```

```
# 화면의 0, 0 위치에 글자크기 30인 "안녕하세요" 문자 쓰기
```

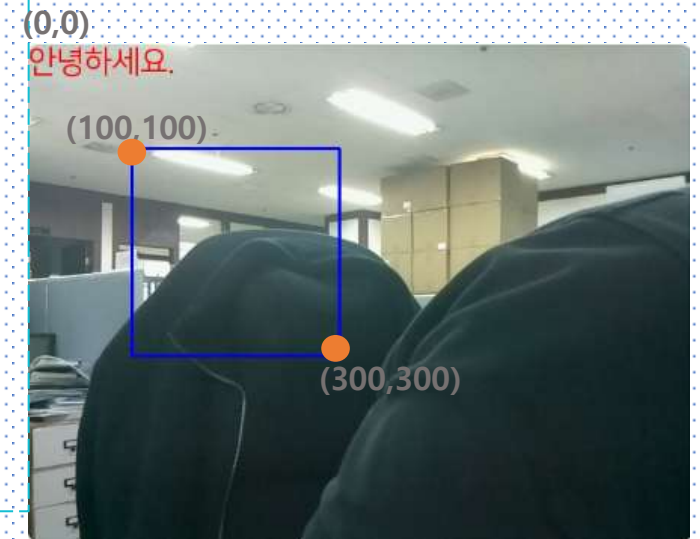
```
image = camera.putTextPIL(image, '안녕하세요.', (0, 0), 30, (0, 0, 255))
```

```
# image 변수를 test.jpg 저장
```

```
camera.imwrite("test.jpg", image)
```

```
# IDE 뷰어에 표시
```

```
camera.imshow_to_ide(image)
```



- 이미지를 촬영하고, 얼굴/사물/코드를 인식합니다.

```
from openpibo.vision import Camera
from openpibo.vision import Detect
from openpibo.vision import Face

camera = Camera()
detect = Detect()
face = Face()

# 이미지 촬영
image = camera.read()

result_face = face.detect_face(image)
result_object = detect.detect_object(image)
result_qr = detect.detect_qr(image)

print('얼굴인식:', result_face)
print('사물인식:', result_object)
print('QR코드인식:', result_qr)

# IDE에 표시
camera.imshow_to_ide(image)
```

파이보의 카메라 및 인식 기능을 위한 초기 설정

얼굴/사물/QR코드 인식

• 얼굴/사물/코드 인식 결과

[Tue Jul 16 2024 12:18:47 GMT+0900 (대한민국 표준시)]:

얼굴인식: [(66, 42, 215, 215)]

사물인식: [{'name': 'person', 'score': 84, 'position': (10, 10, 364, 476)}]

QR코드인식: {'data': 'http://en.m.wikipedia.org', 'type': 'QRCODE', 'position': (431, 287, 532, 389)}

종료됨.

* 얼굴인식 (리스트)

```
[
  얼굴1: (x,y,w,h 좌표),
  얼굴2: (x,y,w,h 좌표),
  ...
  얼굴n: (x,y,w,h 좌표)
]
```

* 사물인식 (리스트)

```
[
  사물1: {name:이름, score: 점수, position: x1,y1,x2,y2 좌표},
  사물2: {name: 이름, score : 점수, position: x1,y1,x2,y2 좌표},
  ...
  사물n: {name: 이름, score : 점수, position: x1,y1,x2,y2 좌표},
]
```

* 코드인식

```
{
  data:내용, type: 종류,
  position: x1,y1,x2,y2 좌표
}
```

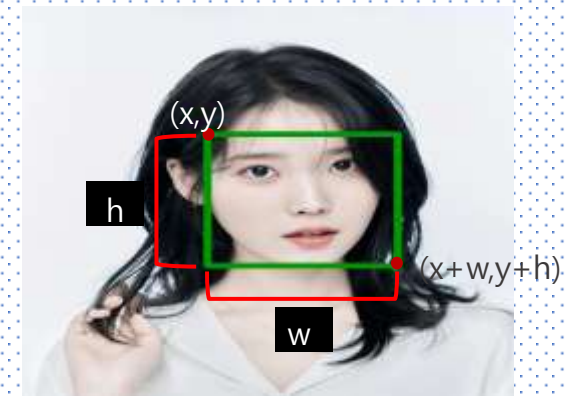
- 얼굴/사물/코드 - 표식 추가

```
print('QR코드 인식:', result_qr)
...
# visualize
if len(result_face):
    x,y,w,h = result_face[0]
    image = camera.rectangle(image, (x,y), (x+w,y+h), (0,0,255), 2)

if len(result_object):
    for item in result_object:
        x1,y1,x2,y2 = item['position']
        name = item['name']
        image = camera.putTextPIL(image, name, (x1-10, y1-10), 20, (255, 255, 255))
        image = camera.rectangle(image, (x1,y1), (x2,y2), (255,255,0), 2)

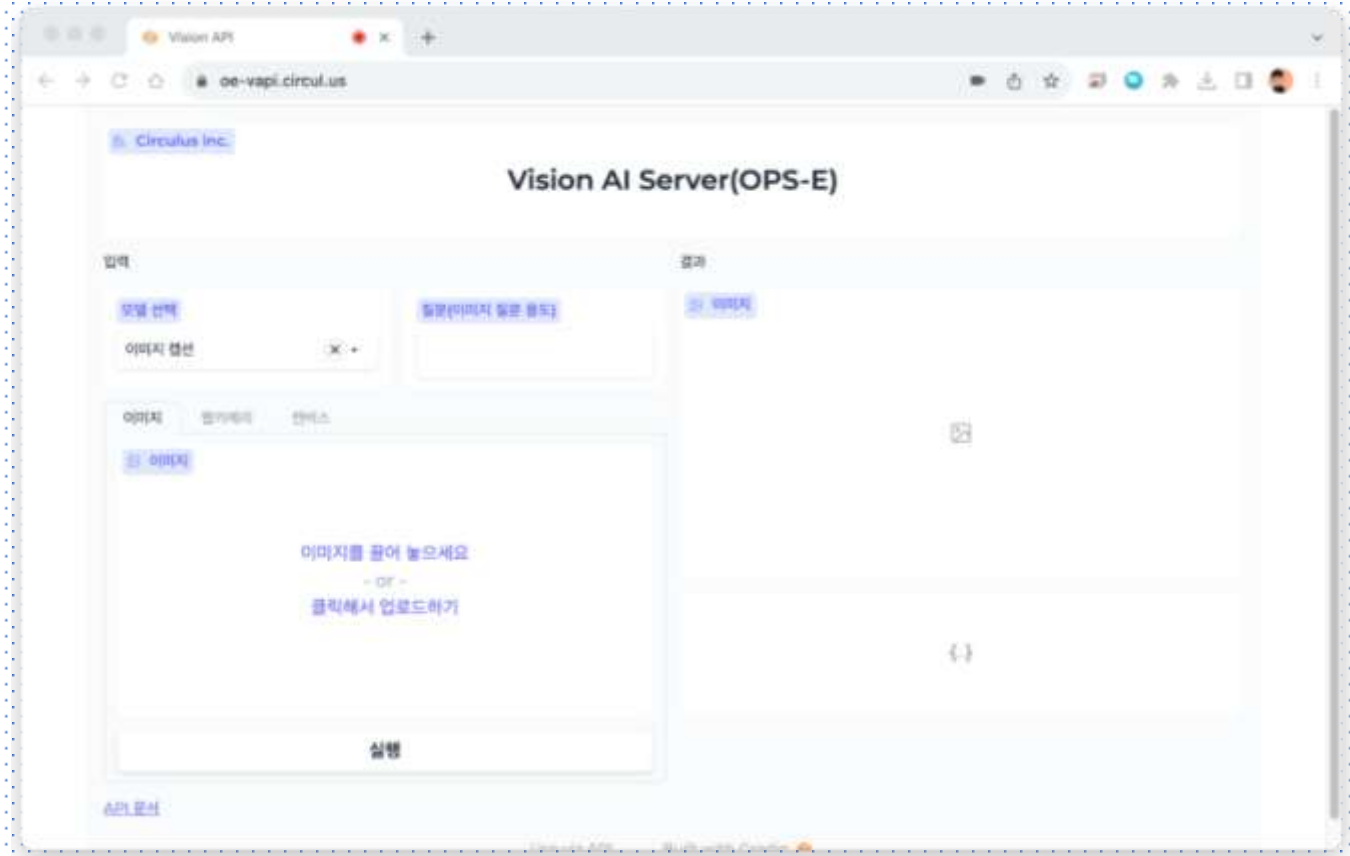
if result_qr['data'] != "":
    print(result_qr['data'])
    x1,y1,x2,y2 = result_qr['position']
    image = camera.rectangle(image, (x1,y1), (x2,y2), (0,255,0), 2)
...
# IDE에 표시
camera.imshow_to_ide(image, 1)
```

* 얼굴인식 함수의 결과는 x,y,w,h 이므로
rectangle 함수에 사용할 수 있도록,
x,y,x+w,y+h 로 변경해야함
x,y: 얼굴 범위의 좌측 상단 좌표
w,h: 인식된 얼굴의 가로길이(w), 세로길이(h)



얼굴좌표: (x,y,w,h)

- 좀 더 다양한 컴퓨터 비전 체험을 하려면?
- 크롬 또는 엣지 등 브라우저에 <https://oe-vapi.circul.us> 를 입력하여 접속
- https://docs.google.com/presentation/d/17hLPDdeH_0x7Z0SZq3yHamC8zj7xD1Fy/edit#slide=id.p3



토의활동

얼굴 인식 기술로 어떤 문제가 생길 수 있을까요?





- 봇 카드 인식 – QR코드 인식과 동일

```
from openpibo.vision import Camera
from openpibo.vision import Detect

camera = Camera()
detect = Detect()

image = camera.read()
result_qr = detect.detect_qr(image)

print("QR Detect:", result_qr)
camera.imshow_to_ide(image, 1)
```

QR코드 인식에서 봇카드는
type: CARD로 인식하여, 카드 이름 출력

```
QR Detect: {'data': '댄스', 'type': 'CARD', 'position': (124, 139, 289, 303)}
QR Detect: {'data': '댄스', 'type': 'CARD', 'position': (117, 163, 275, 322)}
QR Detect: {'data': '댄스', 'type': 'CARD', 'position': (95, 165, 260, 329)}
QR Detect: {'data': '댄스', 'type': 'CARD', 'position': (88, 166, 257, 330)}
```

- 로봇 비서에서 사용하는 QR코드 카드



AI 휴머노이드 로봇 비서 만들기

Part 4. 대화하는 파이보



- 표준입력을 이용하여, 파이보와 대화합니다.

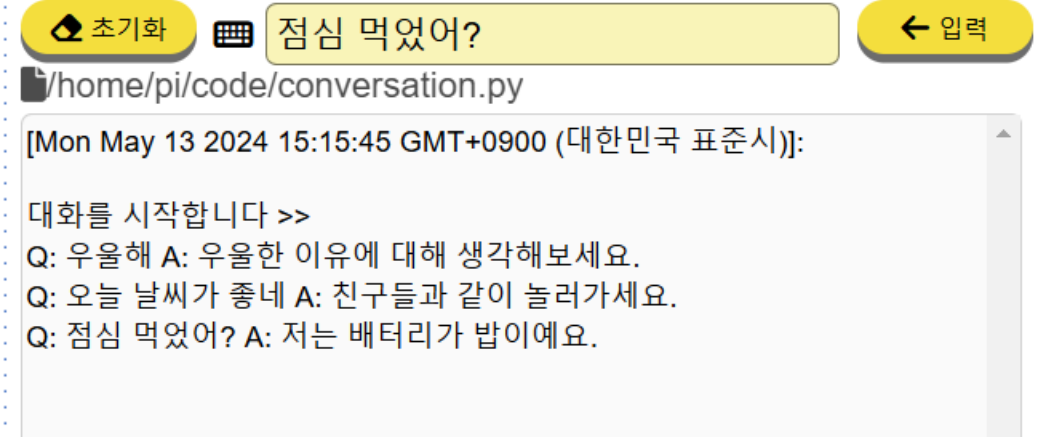
```
from openpibo.speech import Dialog
from openpibo.audio import Audio

dialog = Dialog()
audio = Audio()

print('대화를 시작합니다 >>')
while True:
    question = input("")

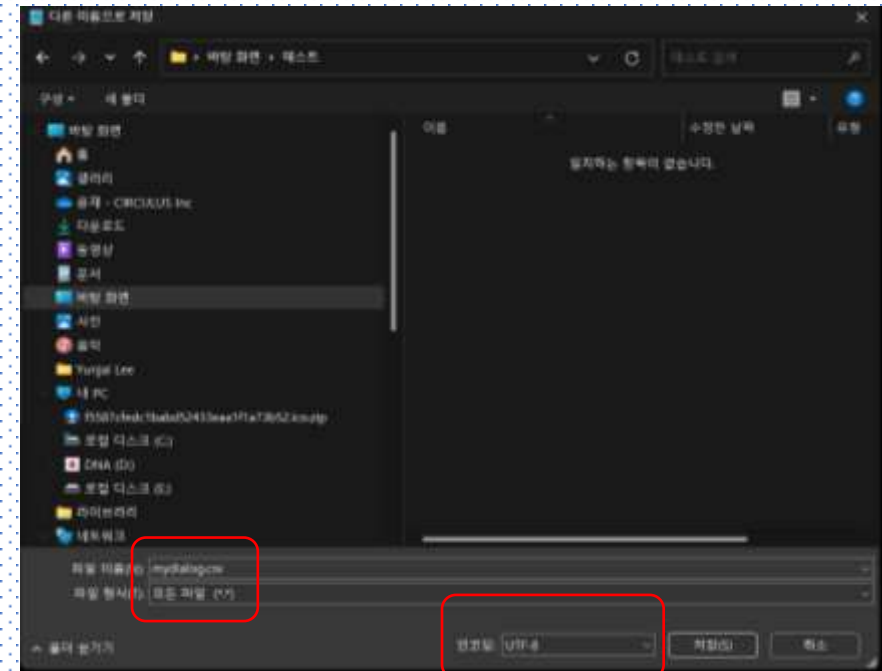
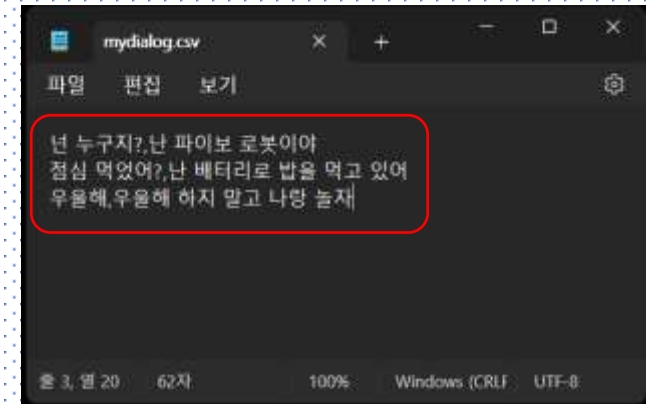
    if question == '그만':
        print('즐거운 하루 보내세요.')
        break

    answer = dialog.get_dialog(question)
    print('Q:', question, 'A:', answer)
```



- 약 5000쌍의 질문-대답 데이터셋을 기반, N-Gram(n=2) 방식으로 유사 질문을 찾고, 해당하는 답을 도출
 - 입력칸을 이용하여, 질문을 입력하고 답을 확인
- * N-Gram
두 문장을 특정 수(n)의 문자 단위로 쪼개어, 비교하여 문장의 유사도를 측정하는 측정 방법

- 나만의 데이터셋을 만들어 봅니다.
- 메모장을 이용하여, 나만의 대화파일을 만듭니다.
 - 아래와 같은 형식으로 mydialog.csv 파일을 만들고 저장합니다.
 - 질문1,답변1
 - 질문2,답변2
 - ...
- IDE의 code 폴더에 업로드합니다.



• 나만의 데이터셋으로 대화합니다.

```
from openpibo.speech import Dialog
from openpibo.audio import Audio
```

```
dialog = Dialog()
audio = Audio()
```

```
# 대화 파일 교체
```

```
dialog.load('mydialog.csv')
```

```
print('대화를 시작합니다 >>')
```

```
while True:
```

```
    question = input('')
```

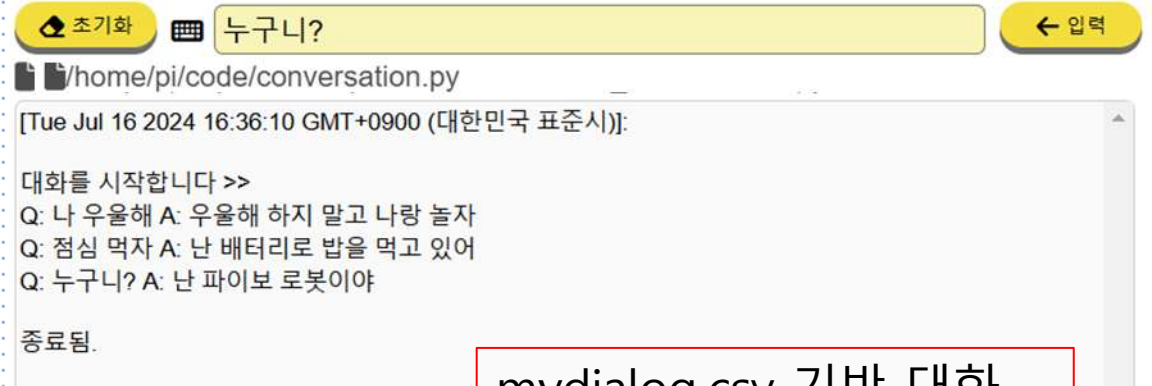
```
    if question == '그만':
```

```
        print('즐거운 하루 보내세요.')
```

```
        break
```

```
    answer = dialog.get_dialog(question)
```

```
    print('Q:', question, 'A:', answer)
```



[Tue Jul 16 2024 16:39:35 GMT+0900 (대한민국 표준시)]:

대화를 시작합니다 >>

Q: 나 우울해 A: 우울한 이유에 대해 생각해 보세요.

Q: 점심 먹자 A: 즐거운 시간 보내시길 바랍니다.

Q: 누구니? A: 저는 위로해드리는 로봇이에요.

기본 대화

AI 휴머노이드 로봇 비서 만들기

Part 4. AI 로봇 비서 파이보



- 팀 빌딩을 위한 활동

팀을 구성하기 위한 활동을 진행합니다.
특화형은 서로 모르는 친구들이 올 확률이 높기 때문에
아이스브레이킹과 팀 빌딩을 통해
분위기를 풀어주고
남은 수업을 잘 할 수 있도록 지원해주세요.

제공하는 물품 : 전지, 사인펜, 색연필

- 나만의 AI 로봇 비서를 만들어봅니다.
 - AI 로봇 비서가 어떤 일을 하면 좋을까요?
 - AI 로봇 비서를 기획하고 일 처리에 대해 순서도로 표시해봅니다.

- 프로젝트 예시

- 통합 기능으로 구현된 프로젝트 예시 코드를 참고해서, 나만의 AI 로봇 비서를 구현

- 반복

- 1분 마다, 시간 알람 재생 기능
- 사람의 얼굴을 찾았을 때, 인사하는 기능
- 카메라로 이미지를 촬영하고, 뷰어로 스트리밍하는 기능
- 봇카드를 인식했을 때, 기능 수행
 - ✓ 날씨 - 날씨 데이터를 수집하고, OLED, LED, TTS, Motion 결합한 날씨 알림 기능
 - ✓ 뉴스 - 뉴스 데이터를 수집하고, OLED, LED, TTS, Motion 결합한 뉴스 알림 기능
 - ✓ 체조 - 체조 음악과 로봇 체조를 실행하는 기능
 - ✓ 대화 - 표준 입력 창을 이용하여, 대화하는 기능
 - ✓ 카메라 - 사진 촬영해서 파일로 저장하는 기능
 - ✓ ... 추가로 기능을 구현해보세요.

```
from openpibo.vision import Camera, Detect, Face
from openpibo.speech import Speech, Dialog
from openpibo.audio import Audio
from openpibo.oled import Oled
from openpibo.motion import Motion
from openpibo.collect import Weather, News
from openpibo.device import Device
import time
```

```
camera = Camera()
detect = Detect()
face = Face()
speech = Speech()
dialog = Dialog()
audio = Audio()
oled = Oled()
motion = Motion()
weather = Weather()
news = News()
device = Device()
```

```
IMAGE_DIR = '/home/pi/openpibo-files/image/'
AUDIO_DIR = '/home/pi/openpibo-files/audio/'
VOLUME = 50
min_text = '-1'
```

```
oled.set_font(size=30)
speech.tts(string='사진 찍을게요.', filename='photo.mp3', voice='main')
```

- 사용 패키지 import 및 초기화
- 이미지, 오디오 폴더 이름 사전 정의
- VOLUME = 오디오 음량
- min_text = 현재 시간 중 '분' 에 해당하는 문자
- photo.mp3는 계속 생성할 필요없으니, 시작시 한번 생성

```
while True: # 반복
    # 현재 시간 확인
    # time_list = ['2024','07','15','10','10','5'] / 2024년 7월 15일 10시 10분 5초 라면,
    time_list = time.strftime('%Y,%m,%d,%H,%M,%S').split(',')
    print(time_list)

    # qr, 얼굴 인식
    image = camera.read()
    result = detect.detect_qr(image)
    items = face.detect_face(image)

    # 분 단위 알람 기능
    if time_list[4] != min_text:
        oled.draw_image(IMAGE_DIR + 'machine/clock.jpg')
        oled.show()
        speech.tts(string=f'{time_list[3]}시 {time_list[4]}분 입니다.', filename='voice.mp3', voice='main')
        audio.play('voice.mp3', VOLUME)
        min_text = time_list[4]
```

* '분' 단위 알람 기능

- time_list[4]는 '분'에 해당하는 문자이고 min_text는 이전에 저장된 '분' 단위 문자열
- time_list[4] != min_text 조건은 '분'이 바뀔 때, 1분에 한번 주기 의미

```
while True: # 반복문 안
```

```
...
```

```
# 얼굴을 찾았을 때, 인사
```

```
if len(items) > 0:
```

```
    device.eye_on(0,255,255)
```

```
    x,y,w,h = items[0]
```

```
    image = camera.rectangle(image, (x,y), (x+w, y+h), (255,255,255), 3)
```

```
    oled.draw_image(IMAGE_DIR + 'expression/smile.jpg')
```

```
    oled.show()
```

```
    speech.tts(string='안녕하세요.', filename='voice.mp3', voice='main')
```

```
    audio.play('voice.mp3', VOLUME)
```

```
    motion.set_motion('greeting')
```

```
else:
```

```
    device.eye_off()
```

* 얼굴 찾았을 때,

- items는 찾은 얼굴의 목록 이므로 len(items)은 얼굴의 수
- if len(items) > 0 - 얼굴의 수가 0보다 클 때
- 얼굴 위치에 네모 표시 및 LED 표시 / 인사말/ greeting 동작 실행

```
while True: # 반복문 안
```

```
...  
# 봇카드 인식 시, 기능 구현
```

```
if result['type'] == 'CARD':
```

```
    print('봇카드를 인식했습니다.')
```

```
    if result['data'] == '날씨':
```

```
        comment = weather.search('서울')['forecast']
```

```
        oled.draw_image(IMAGE_DIR + 'weather/cloud.jpg')
```

```
        oled.show()
```

```
        speech.tts(string='날씨를 알려드리겠습니다. ' + comment, filename='voice.mp3', voice='main')
```

```
        audio.play('voice.mp3', VOLUME)
```

```
        motion.set_motion('speak1')
```

```
    elif result['data'] == '뉴스':
```

```
        comment = news.search('뉴스랭킹')[0]['description']
```

```
        oled.draw_image(IMAGE_DIR + 'etc/star.jpg')
```

```
        oled.show()
```

```
        speech.tts(string='뉴스를 알려드리겠습니다. ' + comment, filename='voice.mp3', voice='main')
```

```
        audio.play('voice.mp3', VOLUME)
```

```
        motion.set_motion('speak1')
```

```
    elif result['data'] == '체조':
```

```
        oled.draw_image(IMAGE_DIR + 'etc/person.jpg')
```

```
        oled.show()
```

```
        audio.play(AUDIO_DIR+'music/exercise.mp3', VOLUME)
```

```
        motion.set_motion('dance4')
```

```
        audio.stop()
```

- QR코드가 카드로 인식되었을 때, 해당하는 봇카드에 대한 기능 구현

- 날씨 (화면 / 음성 / 모션)

- 뉴스 (화면 / 음성 / 모션)

- 체조 (화면 / 음성 / 모션)


```
while True: # 반복문 안
```

```
...
```

```
# 봇카드 인식 시, 기능 구현
```

```
...
```

```
elif result['data'] == '대화':
```

```
    oled.draw_image(IMAGE_DIR + 'expression/joke.jpg')
```

```
    oled.show()
```

```
    comment = dialog.get_dialog(input("Q>"))
```

```
    speech.tts(string='답변드리겠습니다. ' + comment, filename='voice.mp3', voice='main')
```

```
    audio.play('voice.mp3', VOLUME)
```

```
    motion.set_motion('speak1')
```

```
elif result['data'] == '카메라':
```

```
    oled.draw_image(IMAGE_DIR + 'game/scissors.jpg')
```

```
    oled.show()
```

```
    audio.play('photo.mp3', VOLUME)
```

```
    time.sleep(3)
```

```
    for n in ['- 3 -', '- 2 -', '- 1 -', '찰칵!']:
```

```
        oled.clear()
```

```
        oled.draw_text((30,20), n)
```

```
        oled.show()
```

```
        time.sleep(1)
```

```
    camera.imwrite('photo.jpg', camera.read())
```

```
else:
```

```
    pass
```

```
motion.set_motion('stop')
```

```
camera.imshow_to_ide(image)
```

- QR코드가 카드로 인식되었을 때, 해당하는 봇카드에 대한 기능 구현
 - 대화 (화면 / 음성 / 모션) / 입력칸을 이용하여 질문
 - 카메라 (화면 / 음성 / 모션) / 사진 촬영해서 photo.jpg 로 저장
 - 체조 (화면 / 음성 / 모션)

감사합니다

