

# P5JS에서 Text, Image, Video 처리

# text

- **text(str, x, y, [x2], [y2])**

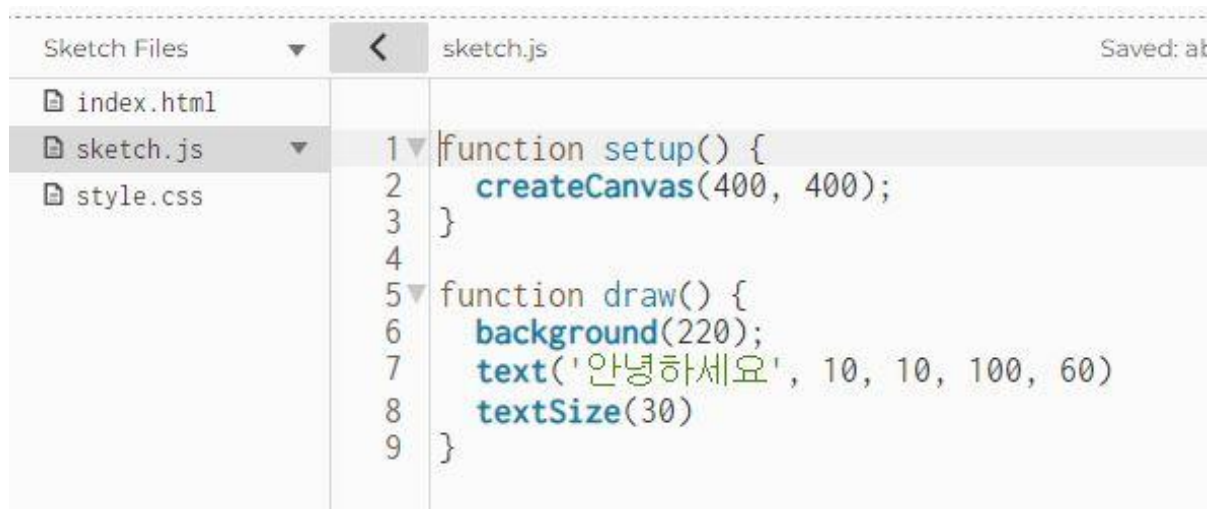
str : 표시할 문자

x : 텍스트의 x좌표값

y : 텍스트의 y좌표값

x2 : 텍스트 상자의 너비

y2 : 텍스트 상자의 높이



```
Sketch Files < sketch.js Saved: at
index.html
sketch.js
style.css

1 function setup() {
2   createCanvas(400, 400);
3 }
4
5 function draw() {
6   background(220);
7   text('안녕하세요', 10, 10, 100, 60)
8   textSize(30)
9 }
```

- **textAlign( )**

함수의 매개변수에 따라 좌표를 기준으로 좌,우,가운데에 텍스트를 그릴 수 있음

- **textFont( )**

함수로 별도 폰트를 지정하지 않을 경우 기본 폰트가 사용됨

- **textSize( )**

함수로 별도 글자 크기를 지정하기 않을 경우 기본 글자 크기가 사용됨

- **fill( )**

함수로 텍스트 색상 변경 가능

\* 참고 : <https://py-edu.tistory.com/374>

# image

- **image(img, x, y, [width], [height] )**

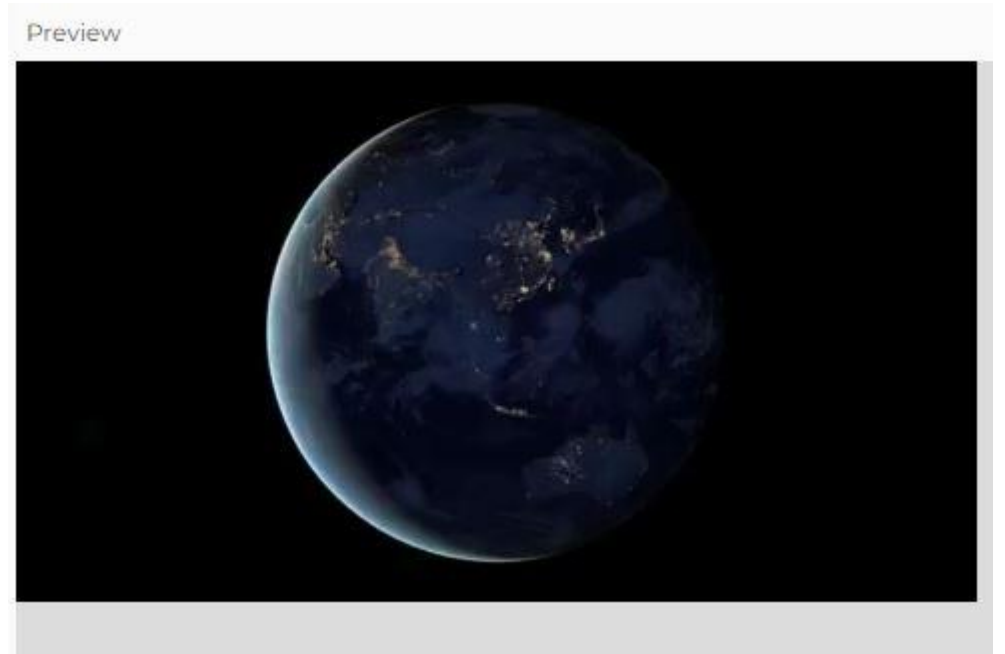
- img : p5.Image, p5.Element : 화면에 나타낼 이미지
- x : 왼쪽 위 모서리의 x 좌표값
- y : 왼쪽 위 모서리의 y 좌표값
- width : 이미지 너비값 (선택사항)
- height : 이미지 높이값 (선택사항)

```
Sketch Files  < sketch.js Saved:
▼ images
  bird.jpg
  kitten.jpg
  index.html
  sketch.js
  style.css

1 let img;
2
3 function preload() {
4   img = loadImage('images/bird.jpg')
5 }
6
7 function setup() {
8   createCanvas(400, 400);
9 }
10
11 function draw() {
12   image(img, 0, 0, 300, 300)
13   // image(img, 25, 25, 400, 400)
14 }
```

# video play

```
Sketch Files  < sketch.js Saved: about 11 h  
▼ assets  
  video.mov  
  index.html  
  sketch.js  
  style.css  
1 var vid;  
2  
3 ▼ function setup() {  
4   createCanvas(windowWidth, windowHeight);  
5   vid = createVideo("assets/video.mov");  
6   vid.position(0, 0);  
7 }  
8  
9 ▼ function draw() {  
10  background(220);  
11  image(vid, 0, 0);  
12 }  
13  
14 ▼ function mousePressed() {  
15   vid.play();  
16 }
```



# video capture : webcam

Sketch Files

index.html

sketch.js

style.css

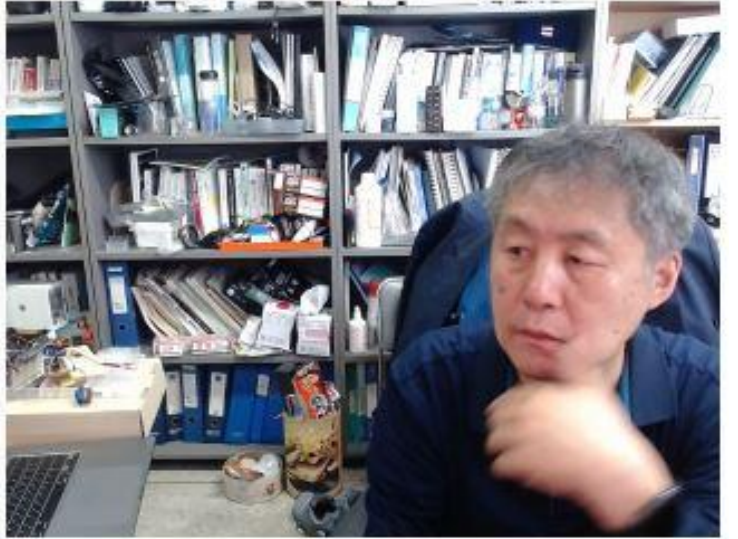
<

sketch.js

Saved: about 11 hours ago

```
1 let capture;
2
3 function setup() {
4   createCanvas(390, 240);
5   capture = createCapture(VIDEO);
6   capture.size(320, 240);
7   capture.hide();
8 }
9
10 function draw() {
11   background(255);
12   image(capture, 0, 0, 320, 240);
13 }
```

Preview



# preload() 함수

- preload() 함수가 모두 끝난 후 setup() 함수 실행

```
let img;  
function preload(){  
  img = loadImage("/assets/learn/program-flow/images/clouds.jpg");  
}
```

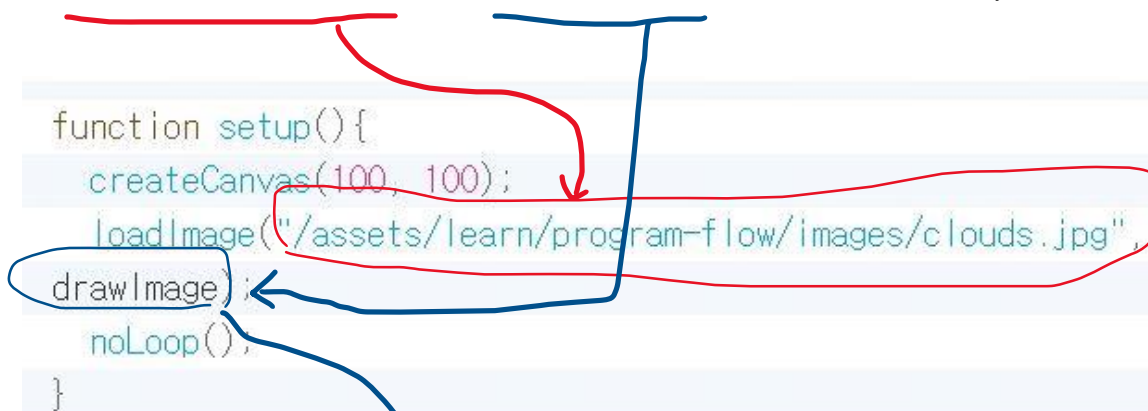
```
function setup(){  
  createCanvas(100, 100);  
  noLoop();  
}
```

```
function draw(){  
  background(200);  
  image(img,0,0);  
}
```

# callback 함수

- 함수를 순차적으로 실행하기 만듦
- 첫번째 함수 인자를 두번째 함수로 전달하고, 두번째 함수가 실행

```
function setup(){  
  createCanvas(100, 100);  
  loadImage("/assets/learn/program-flow/images/clouds.jpg",  
  drawImage);  
  noLoop();  
}
```

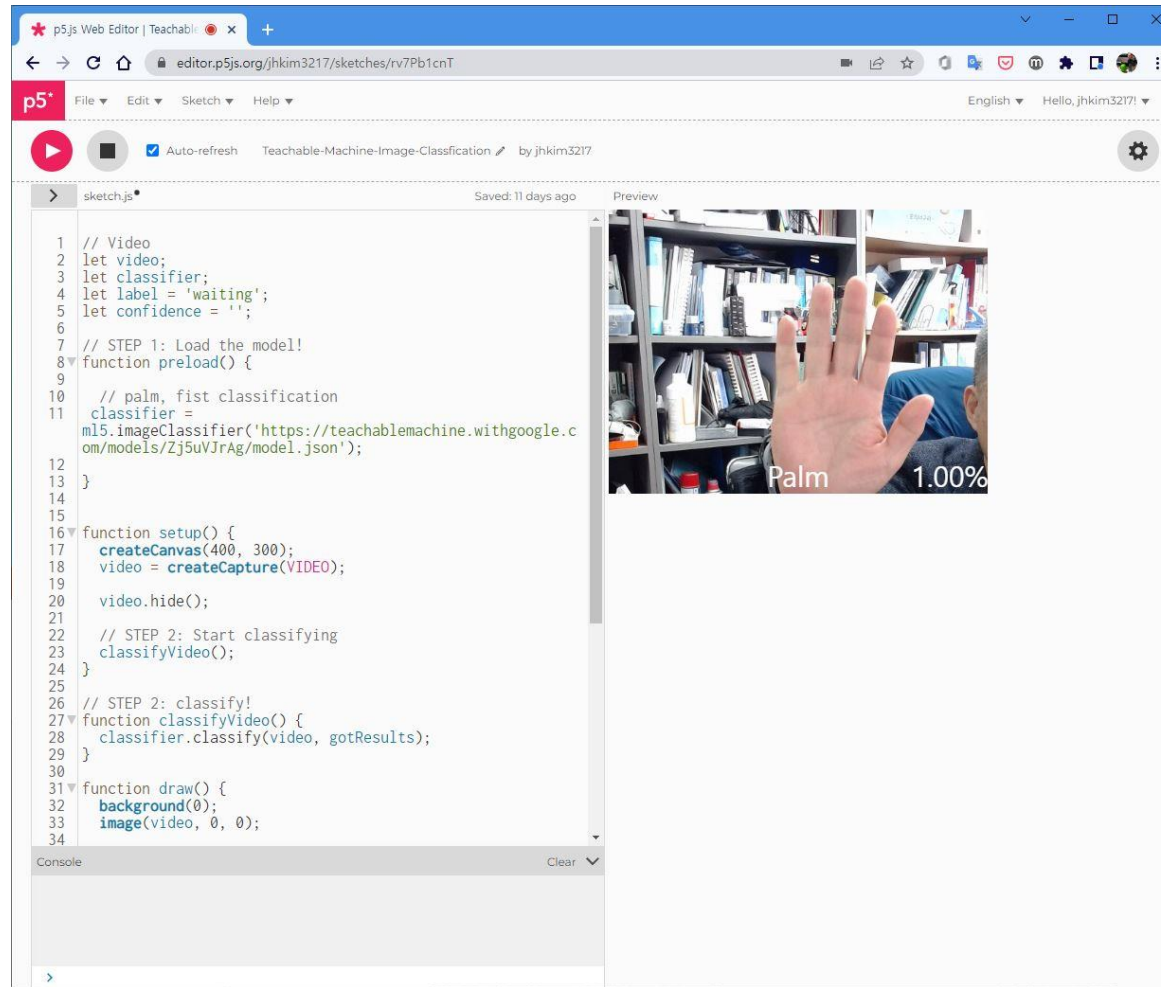


```
function draw(){  
  background(200);  
}
```

```
function drawImage(img){  
  image(img, 0, 0);  
}
```

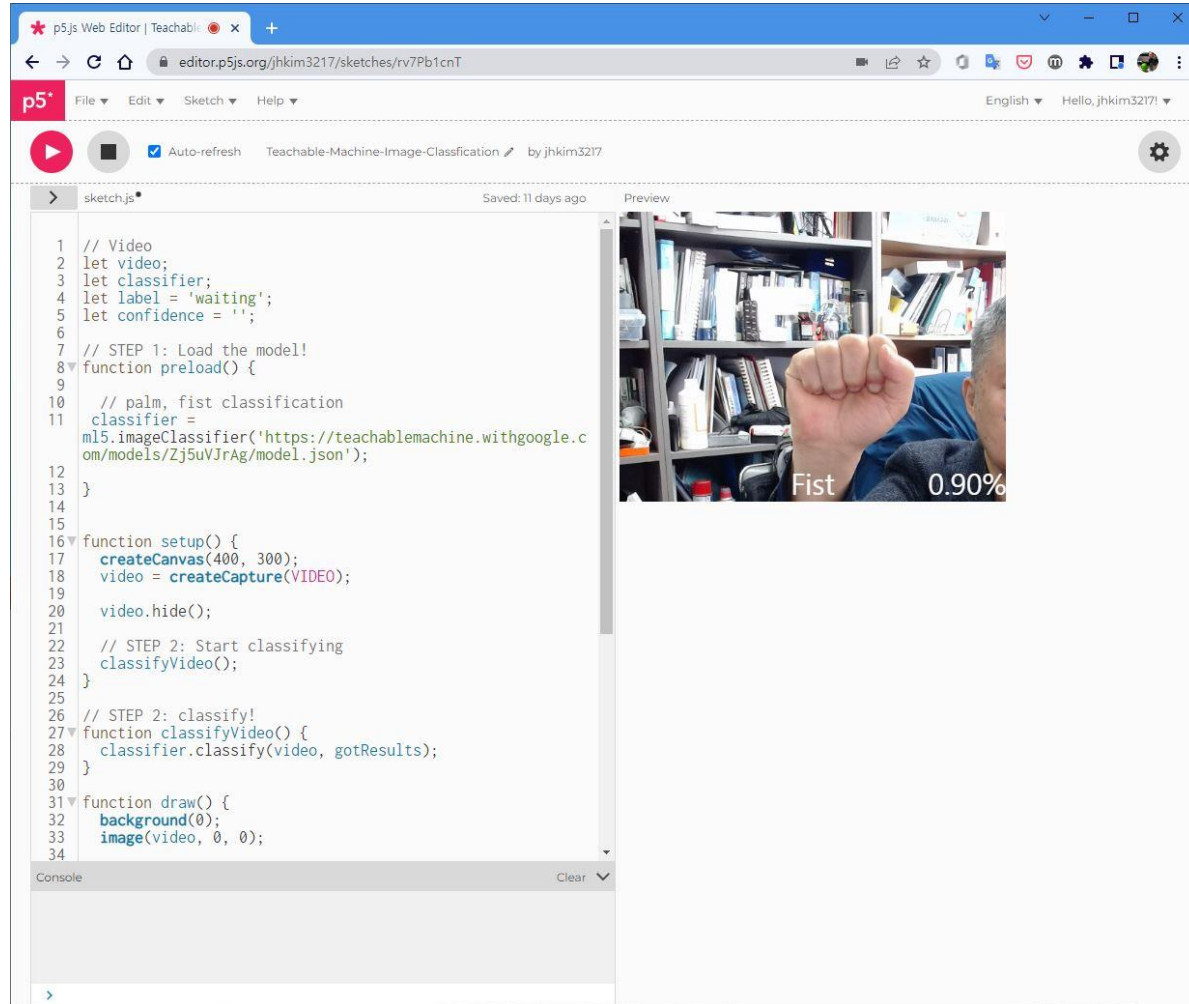
# Teachable Machine을 이용한 이미지 분류

- class 01 : palm

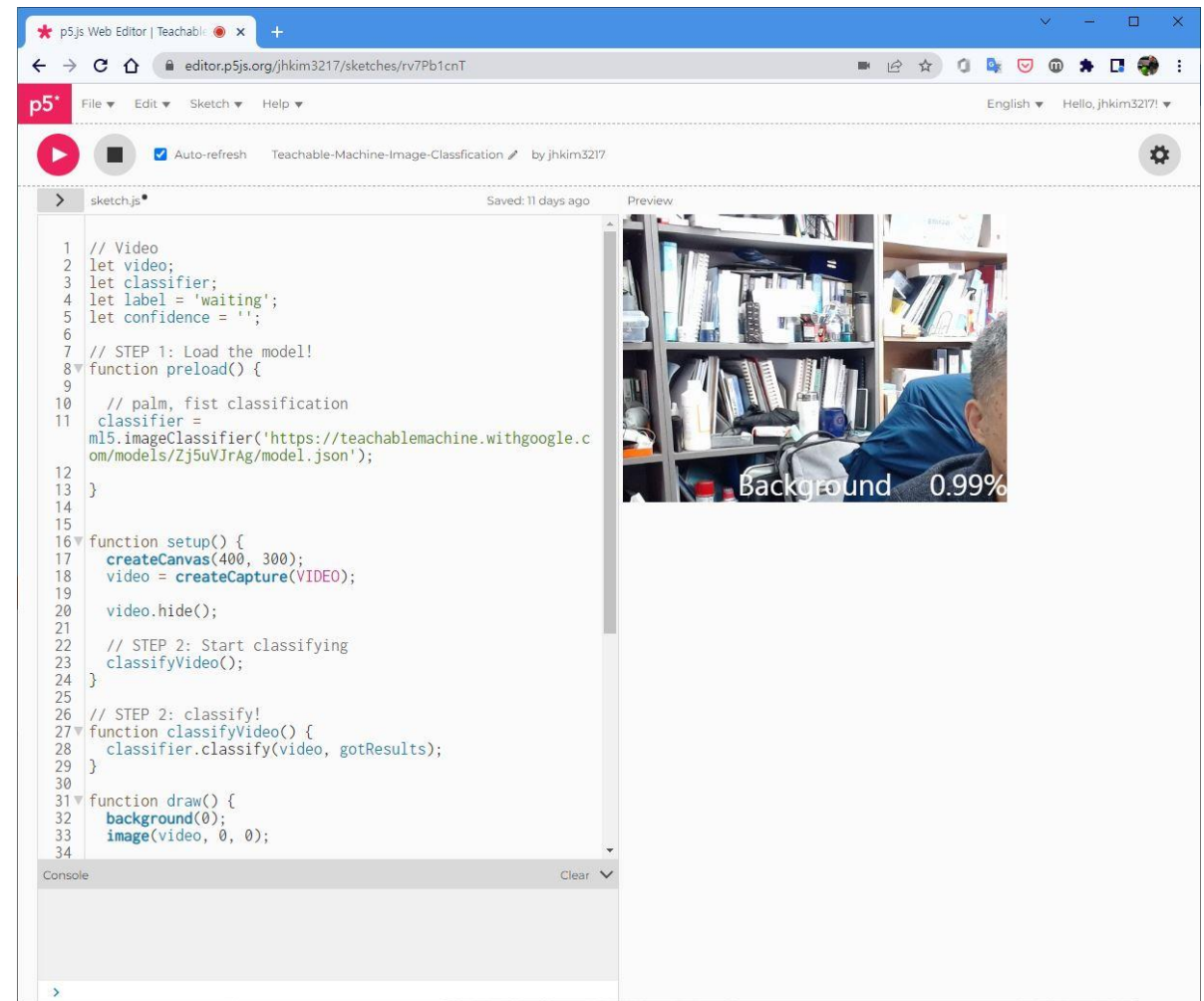




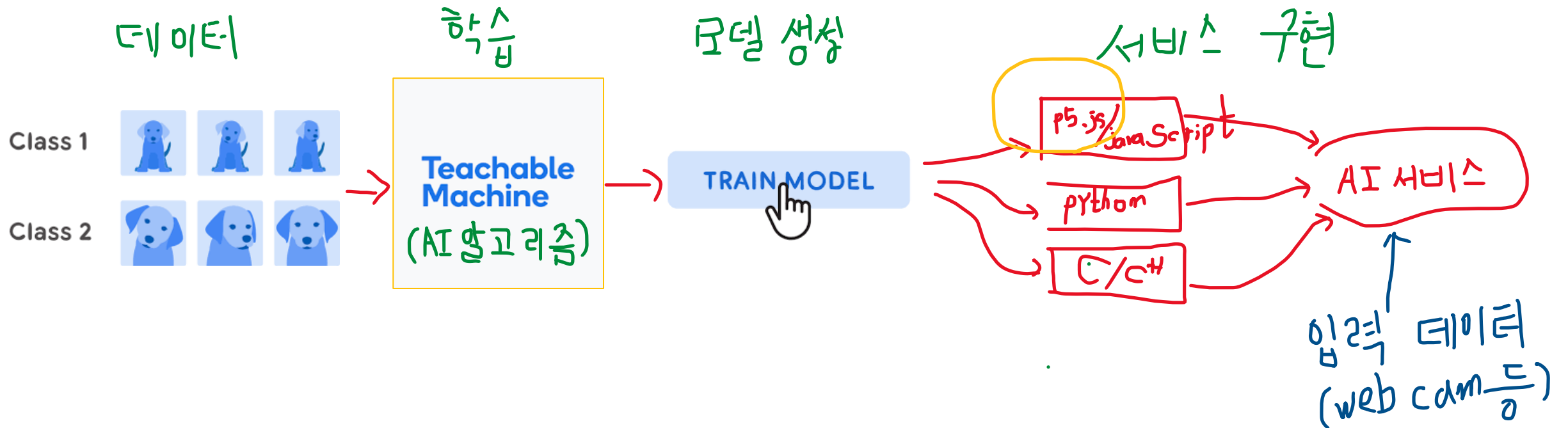
- class 02 : fist



- class 01 : background



# Teachable Machine을 이용한 AI 서비스 만들기



# 모델을 이용한 컴퓨터 프로그래밍

```
//Video;
let video;
let classifier;
let label = "waiting";
let confidence = "";

// STEP 1: Load the model!
function preload() {
  // palm, fist classification
  classifier = ml5.imageClassifier(
    "https://teachablemachine.withgoogle.com/models/Zj5uVJrAg/model.json"
  );
}

function setup() {
  createCanvas(400, 300);
  video = createCapture(VIDEO);
  video.hide();

  // STEP 2: Start classifying
  classifyVideo();
}
```

```
// STEP 2: classify!
function classifyVideo() {
  classifier.classify(video, gotResults);
}

function draw() {
  background(0);
  image(video, 0, 0);

  // STEP 4: Draw the label
  textSize(30);
  textAlign(CENTER, CENTER);
  fill(255);
  text(label, width / 2, height - 16);
  text(confidence, width - 40, height - 16);
}

// STEP 3: Get the classification!
function gotResults(error, results) {
  label = results[0].label;
  confidence = results[0].confidence;
  confidence = confidence.toFixed(2) + "%";
  classifyVideo();
  //classify video again
}
```