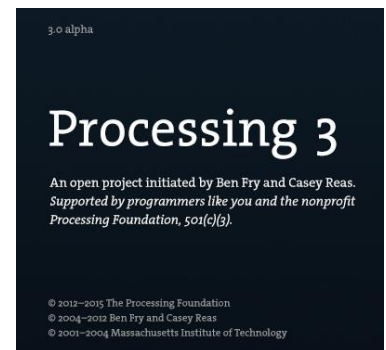


# 증강현실을 이용한 로봇팔 작업 설계



# 증강현실의 개요 및 사례

- ❖ 현실 세계의 배경위에 가상의 물체나 정보를 겹쳐 보여줌 (오버레이)
- ❖ 스마트 기기 게임 포켓몬 고
- ❖ 거리의 상점 종류 및 위치를 알려주는 기능
- ❖ 자동차의 네비게이션 정보를 전방 유리창에 전시
- ❖ 사용자 매뉴얼을 해당 물건의 위나 옆의 공간에 생성

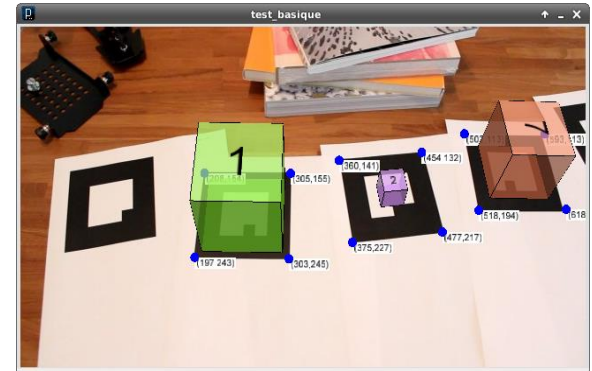


# 증강현실 구현 방법에 따른 분류

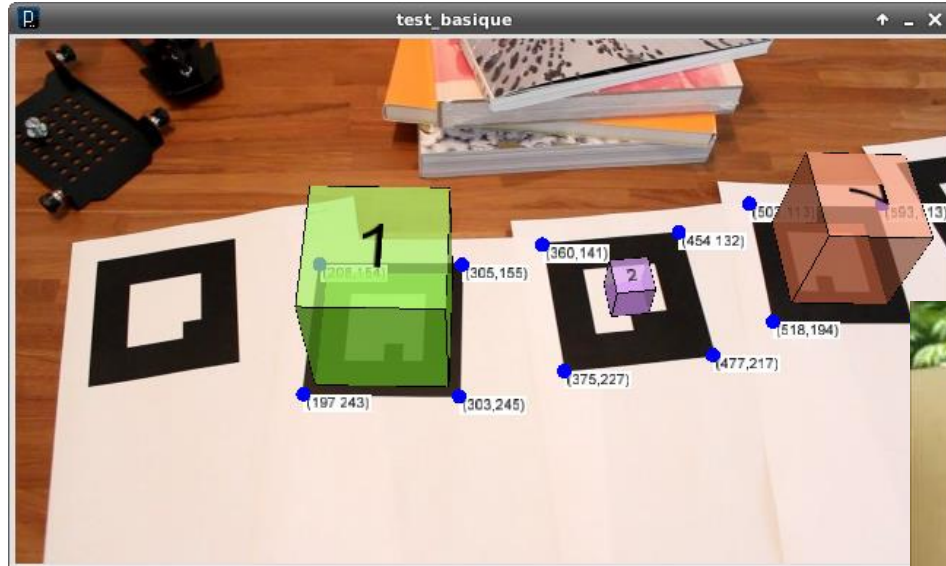
❖ GPS와 자이로 센서 등을 이용하여 사용자의 위치와 지리정보로 구현하는 위치기반형 증강현실

❖ 스테레오 비전, 3차원 스캐너 및 Depth 측정 센서 등으로 구현하는 공간 센서형 증강현실

❖ 특정 마크를 비전 프로세싱으로 인식하여 카메라의 방향 및 거리를 알아내는 마커 인식형 증강현실



# 마커 인식형 증강현실





# 증강현실 기술의 요구

- ❖ 로봇과 로봇, 사람과 로봇이 협업하기 시작
  - ↳ 로봇이 다른 로봇, 사람, 물건등과 간섭하지 않고 작업하기 위해 작업 경로 계획이 요구됨
- ❖ 초기의 경로 계획은 로봇의 각 구동부에 단순한 문법체계를 가진 위치지정 커맨드군을 사용
  - ↳ 로봇팔의 위치를 설정하는 중에도 커맨드 입력 오류로 인한 인명 재산 손실 발생 가능
- ❖ 숙련된 기술자가 로봇의 팔과 손을 직접 움직이며 동작을 한 단계씩 기억시키는 방법
  - ↳ 정밀한 위치를 잡을 수 없는 한계가 있음



➔ 증강 현실을 이용하여 가상의 로봇팔로 경로 계획을 생성하고 동작 중 간섭 여부를 시각적으로 판단

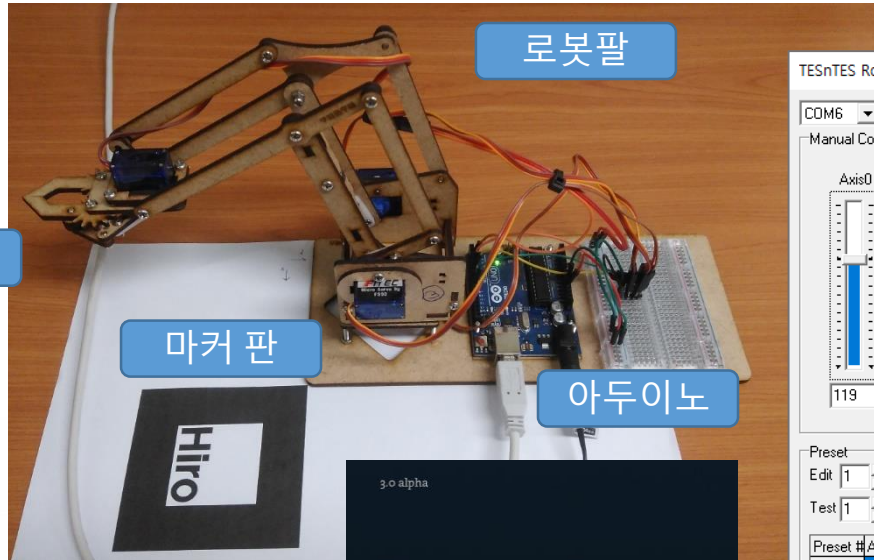
# 로봇 팔 증강현실 교보재 구성



웹캠



웹캠용 스탠드



로봇팔

마커 판

아두이노

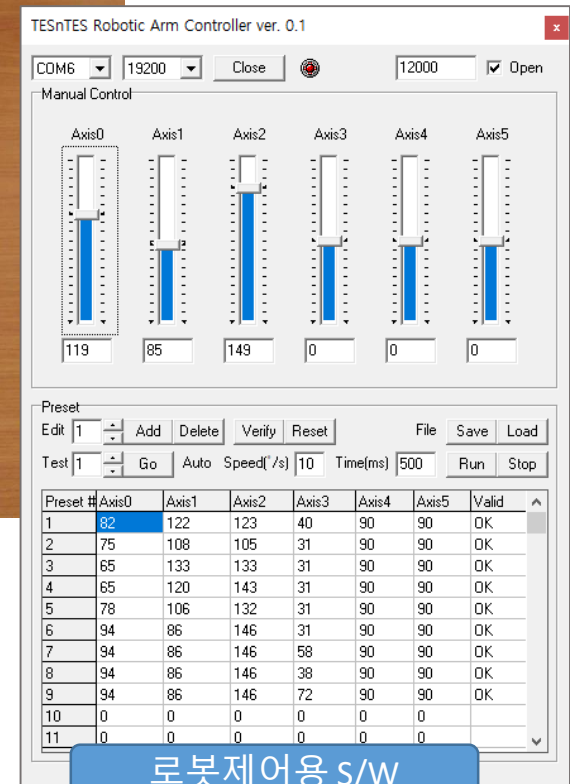
3.0 alpha

## Processing 3

An open project initiated by Ben Fry and Casey Reas.  
Supported by programmers like you and the nonprofit  
Processing Foundation, 501(c)(3).

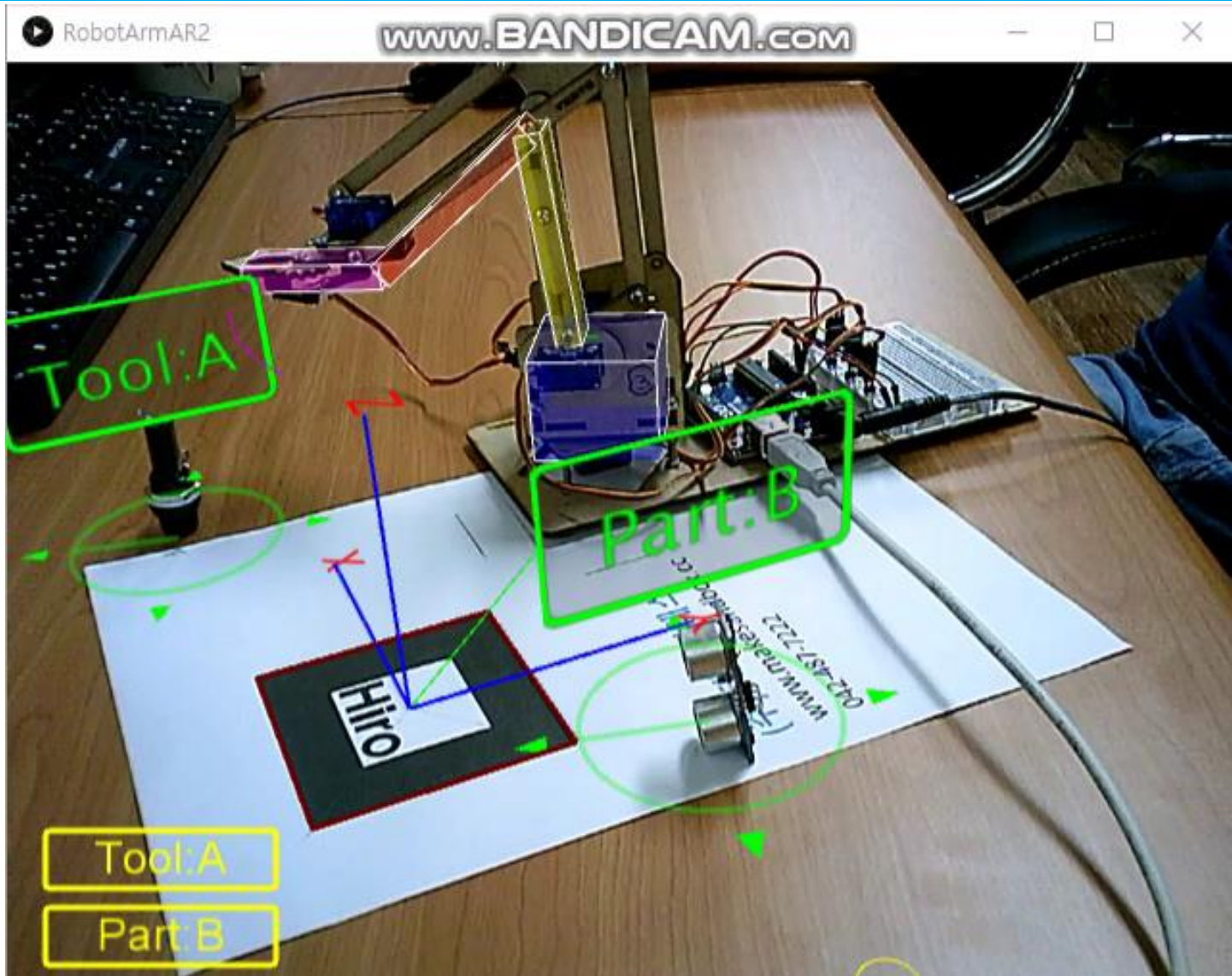
© 2012–2015 The Processing Foundation  
© 2004–2012 Ben Fry and Casey Reas  
© 2001–2004 Massachusetts Institute of Technology

프로세싱



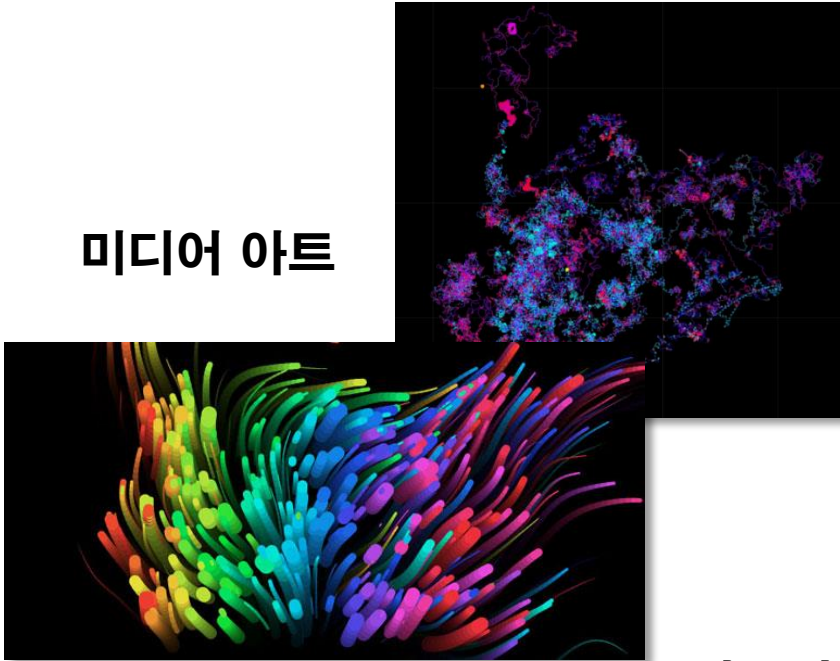
로봇제어용 s/w

# 로봇 팔 증강현실 구동 영상

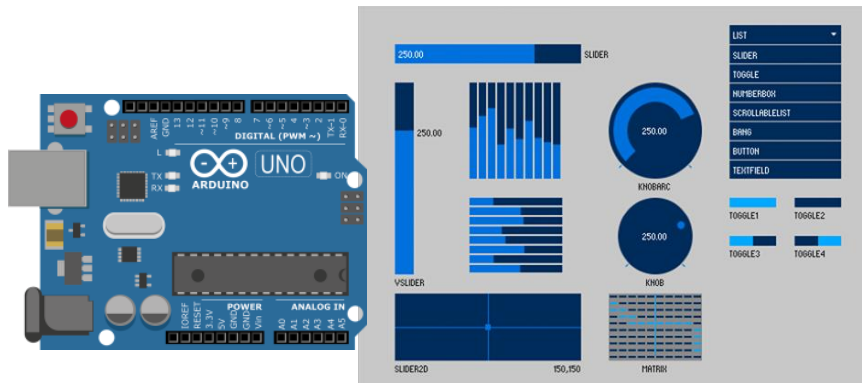


# 프로세싱으로 할 수 있는 것

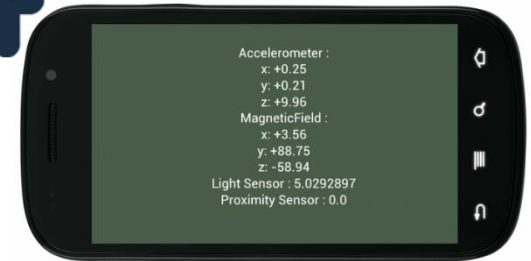
## 미디어 아트



## 아두이노 제어

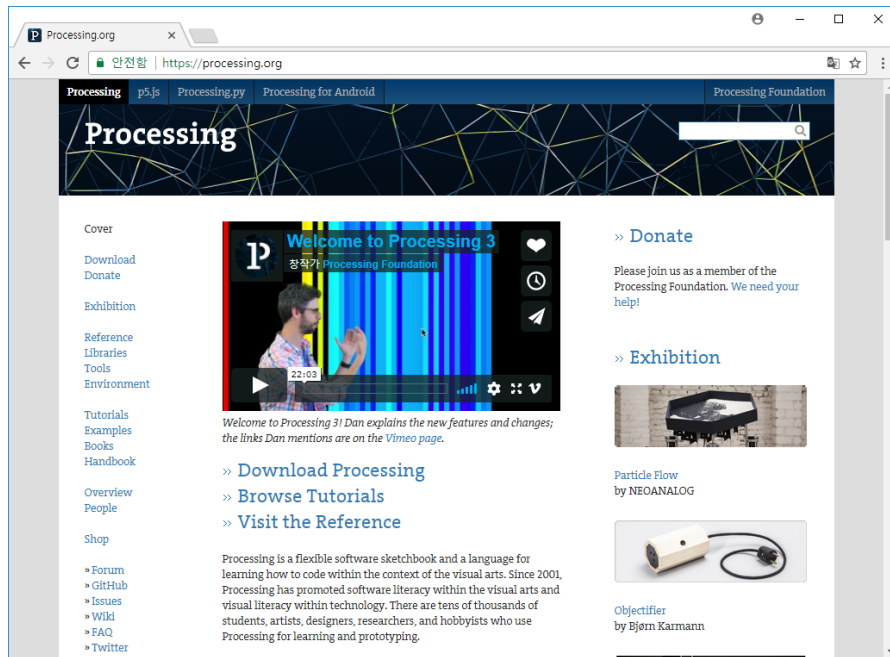


## 안드로이드 앱





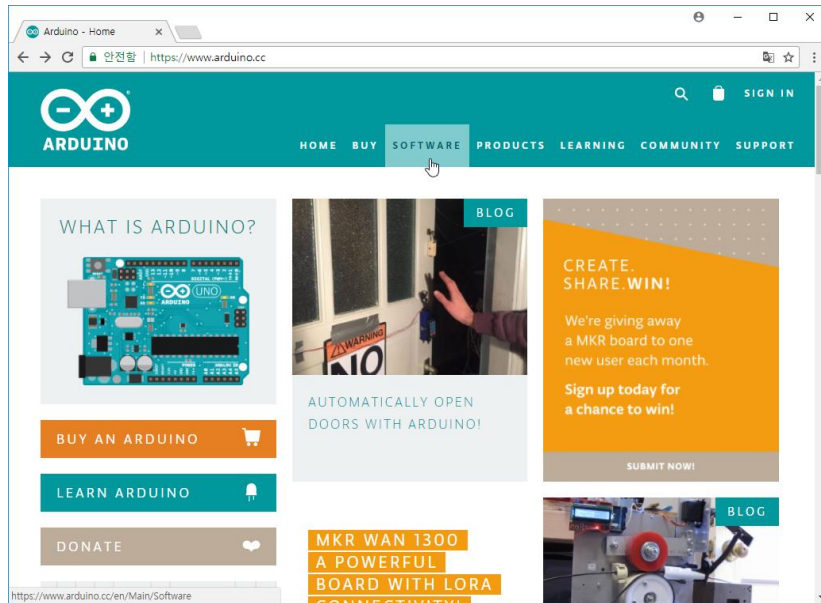
# 프로세싱 개발환경 설치



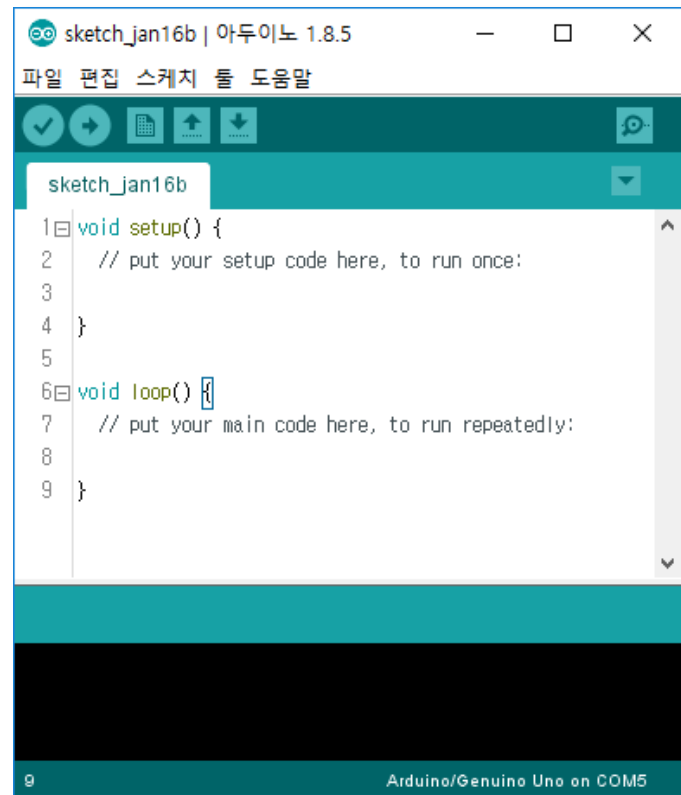
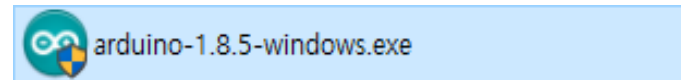
www.processing.org > Download



# 아두이노 개발환경 설치



www.arduino.cc > software > download

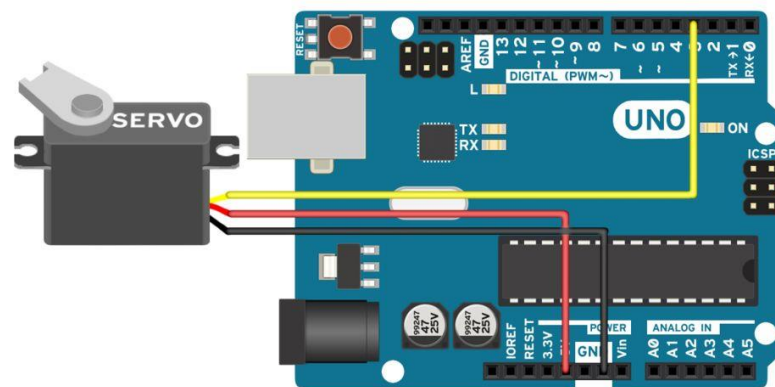


# 통신으로 서보모터 제어하기

```
1 #include <Servo.h>
2 #include <MsTimer2.h>
3
4 Servo axis;
5 struct {
6     int now, dst;
7     int spd;
8 } servo = {85, 90, 1};
9
10 void setup(){
11     Serial.begin(57600);
12     axis.attach(6);
13     MsTimer2::set(25, ServoControl);
14     MsTimer2::start();
15 }
```

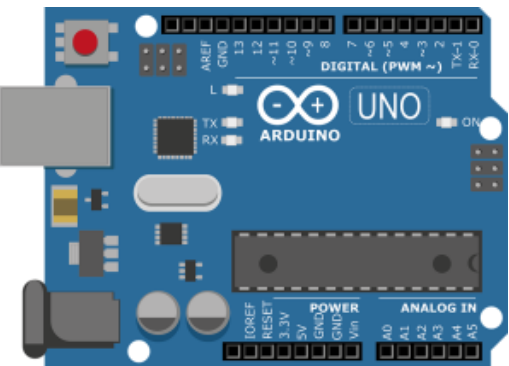
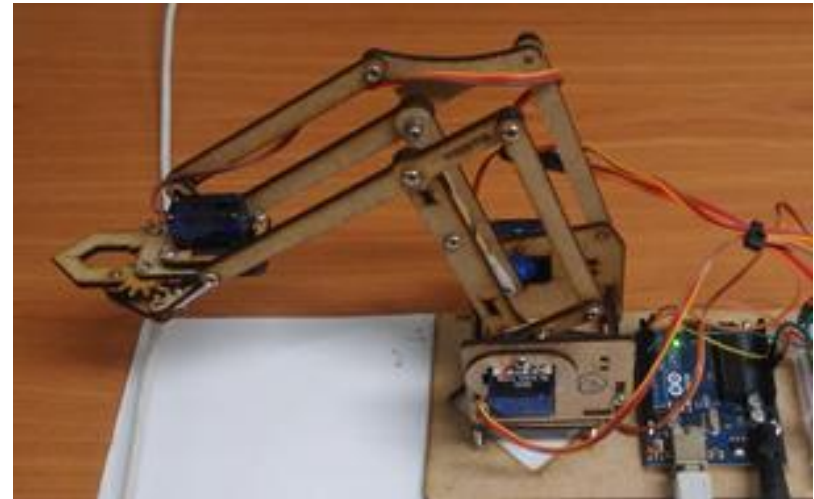
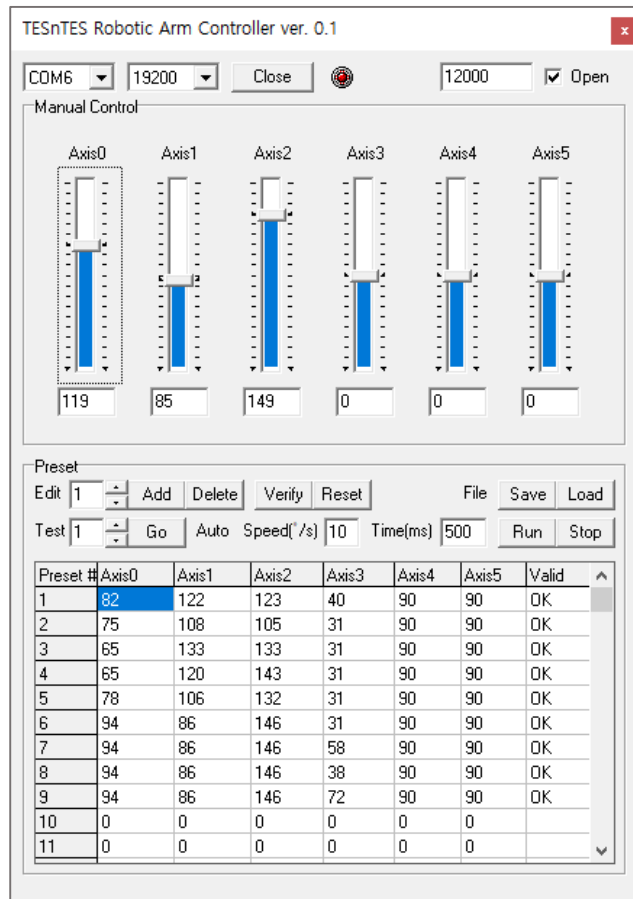
```
17 void loop(){
18     if(Serial.available()) {
19         char buf[64] = {0};
20         int len = Serial.readBytesUntil(']', buf, 64);
21
22         if(buf[0] == '[' && buf[1] == 'I') {
23             int id, pos, spd;
24             char *p;
25
26             p = strtok(&buf[1], ",");
27             if(p && *p == 'I') id = atoi(p+1);
28             p = strtok(NULL, ",");
29             if(p && *p == 'P') pos = atoi(p+1);
30             p = strtok(NULL, ",");
31             if(p && *p == 'S') spd = atoi(p+1);
32
33             servo.dst = pos;
34             servo.spd = spd;
35         }
36     }
37 }
```

```
39 void ServoControl(void)
40 {
41     if(servo.dst < servo.now){
42         servo.now -= servo.spd;
43         axis.write(servo.now);
44     }
45     else if(servo.dst > servo.now){
46         servo.now += servo.spd;
47         axis.write(servo.now);
48     }
49 }
50
```



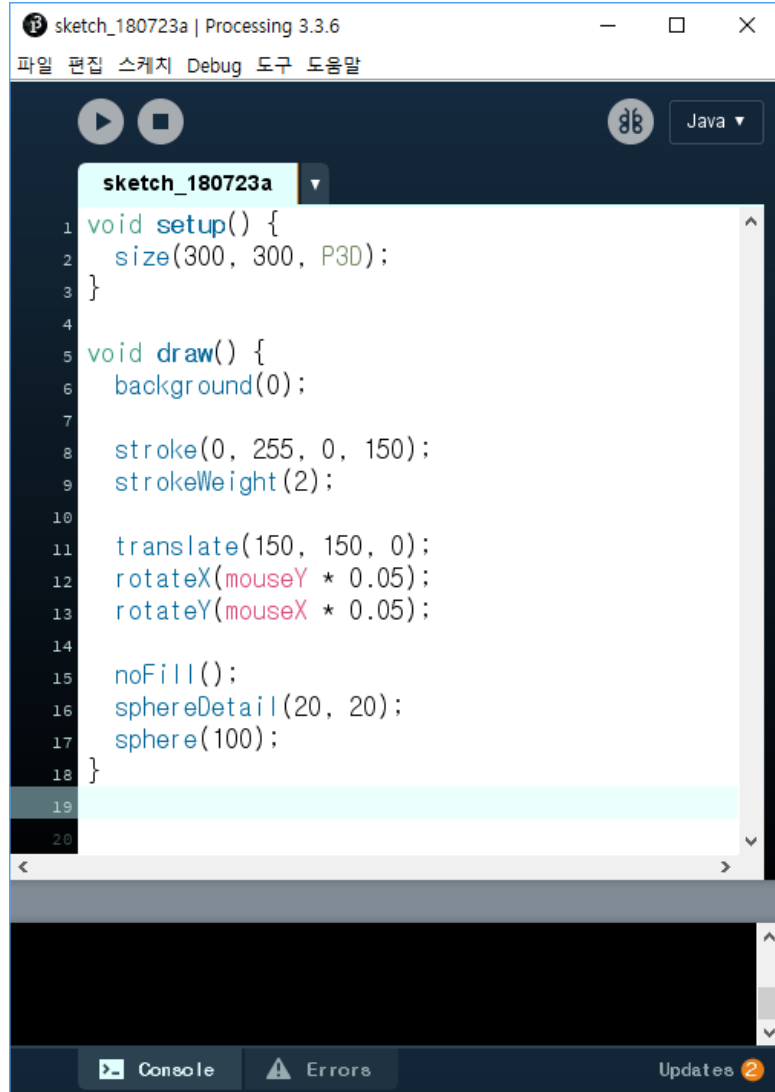
# 로봇팔 컨트롤러로 제어 해 보기

Manual Control에서 각각의 모터를 움직여 보고 Preset에 Add해 보세요



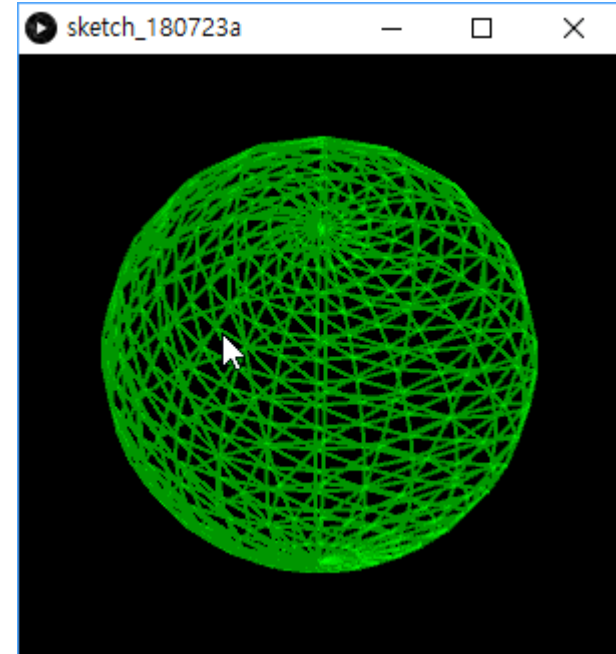


# 프로세싱으로 그려보고 움직여보기



```
sketch_180723a | Processing 3.3.6
파일 편집 스케치 Debug 도구 도움말

sketch_180723a
1 void setup() {
2   size(300, 300, P3D);
3 }
4
5 void draw() {
6   background(0);
7
8   stroke(0, 255, 0, 150);
9   strokeWeight(2);
10
11  translate(150, 150, 0);
12  rotateX(mouseY * 0.05);
13  rotateY(mouseX * 0.05);
14
15  noFill();
16  sphereDetail(20, 20);
17  sphere(100);
18 }
19
20
```



# 프로세싱에 비디오 라이브러리 설치하기

The image shows the Processing 3.3.6 interface. The 'Sketch' menu is open, and the 'Add Library...' option is selected. The 'Contribution Manager' window is also open, showing the 'Libraries' tab. The search filter is set to 'video'. The 'Video' library by The Processing Foundation is highlighted, and its details are shown at the bottom, including the version 1.0.1 and the 'Install' button.

Processing 3.3.6 Interface:

- Menu: Sketch > Add Library...
- Library: Video

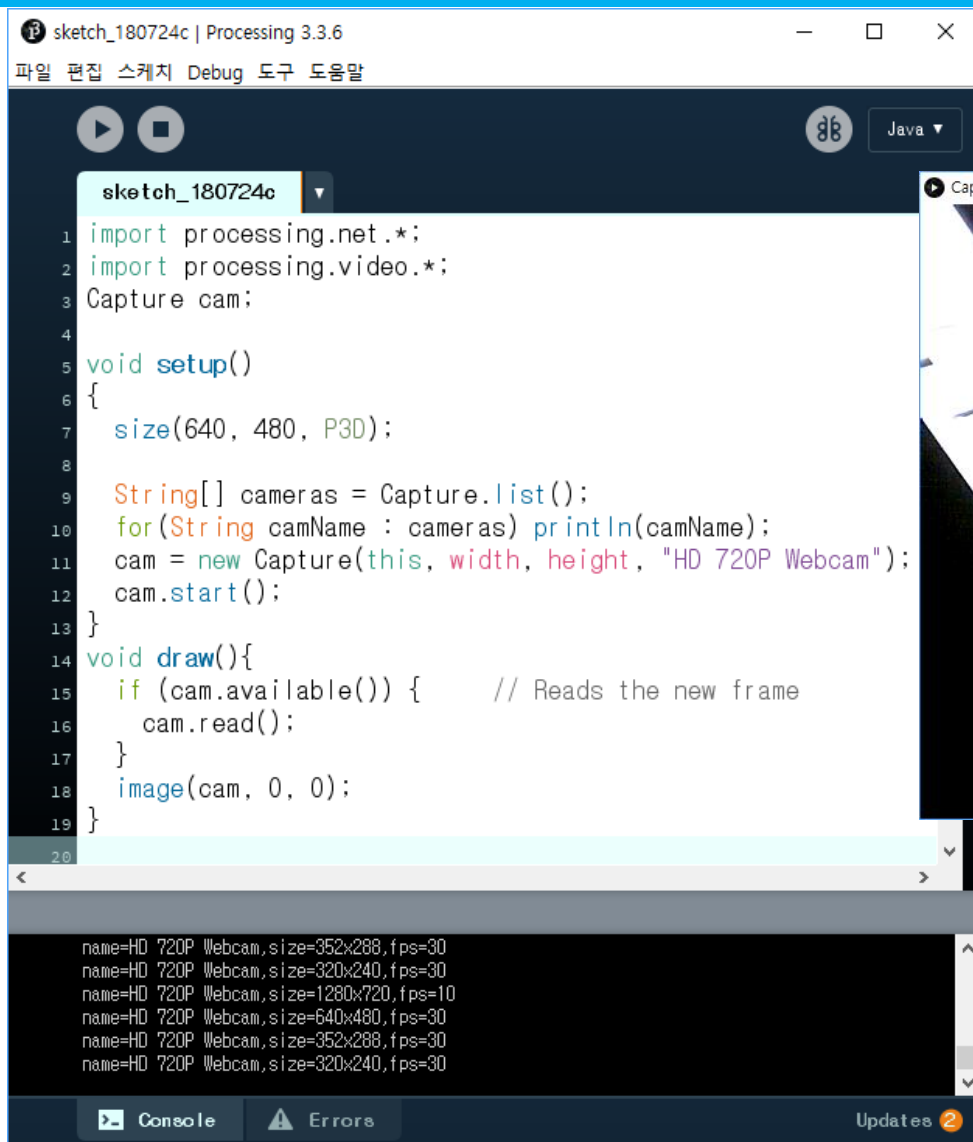
Contribution Manager - Libraries Tab:

Status	Name	Author
	Camera 3D   Alter P3D Rendering to produce St...	Jim Schmitz
	GL Video   Hardware accelerated video on the ...	Gottfried Haider
	Image processing algorithms   Implementation...	Nick 'Milchreis' Müller
	IPCapture   Acquisition of MJPEG video stream...	Stefano "singintime" Baldan
	Syphon   This library allows to share frames be...	Andres Colubri
✓	Video   GStreamer-based video library for Proc...	The Processing Foundation
	Video Export   Simple video file exporter.	Abe Pazos

Video 1.0.1  
The Processing Foundation  
GStreamer-based video library for Processing.

Buttons: Install, 1.0.1 installed, Update, Remove

# 카메라 영상 플레이 하기



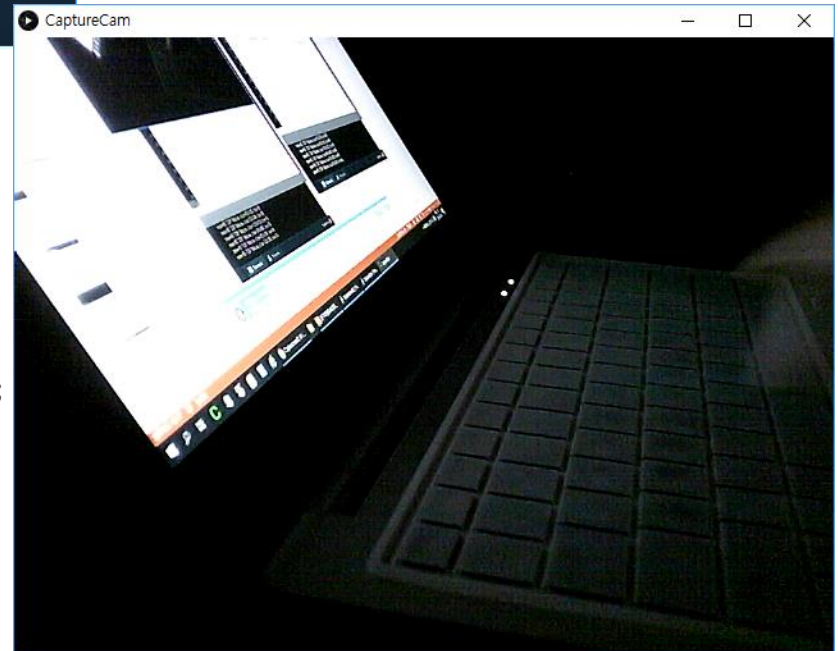
The screenshot shows the Processing IDE window titled "sketch\_180724c | Processing 3.3.6". The menu bar includes "파일", "편집", "스케치", "Debug", "도구", and "도움말". The toolbar has buttons for play, stop, and a Java dropdown menu. The sketch name "sketch\_180724c" is selected in the dropdown. The code in the editor is as follows:

```
1 import processing.net.*;
2 import processing.video.*;
3 Capture cam;
4
5 void setup()
6 {
7   size(640, 480, P3D);
8
9   String[] cameras = Capture.list();
10  for(String camName : cameras) println(camName);
11  cam = new Capture(this, width, height, "HD 720P Webcam");
12  cam.start();
13 }
14 void draw(){
15   if (cam.available()) { // Reads the new frame
16     cam.read();
17   }
18   image(cam, 0, 0);
19 }
20
```

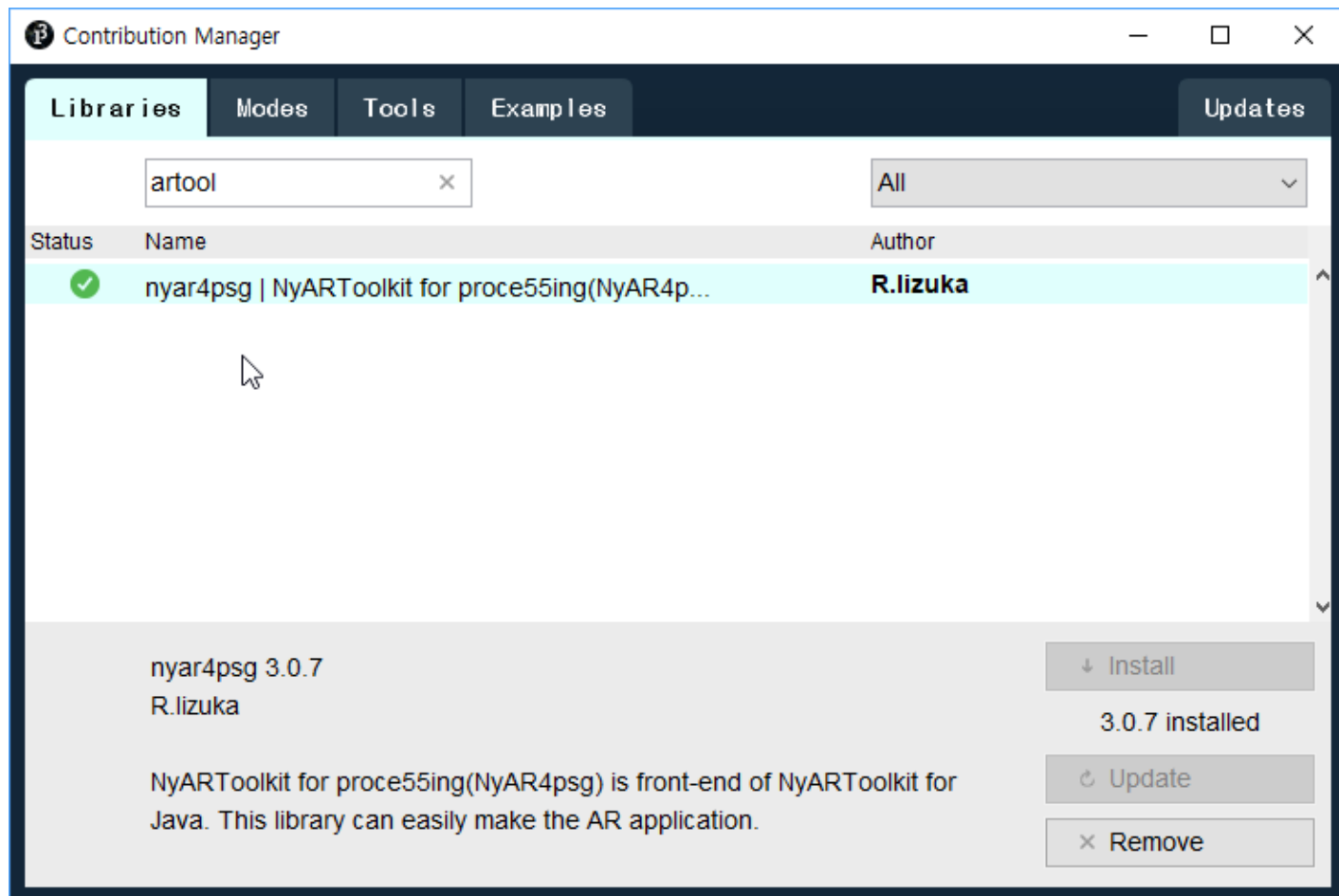
The console at the bottom shows the output of the `println` statements:

```
name=HD 720P Webcam,size=352x288,fps=30
name=HD 720P Webcam,size=320x240,fps=30
name=HD 720P Webcam,size=1280x720,fps=10
name=HD 720P Webcam,size=640x480,fps=30
name=HD 720P Webcam,size=352x288,fps=30
name=HD 720P Webcam,size=320x240,fps=30
```

The bottom status bar shows "Console", "Errors", and "Updates 2".

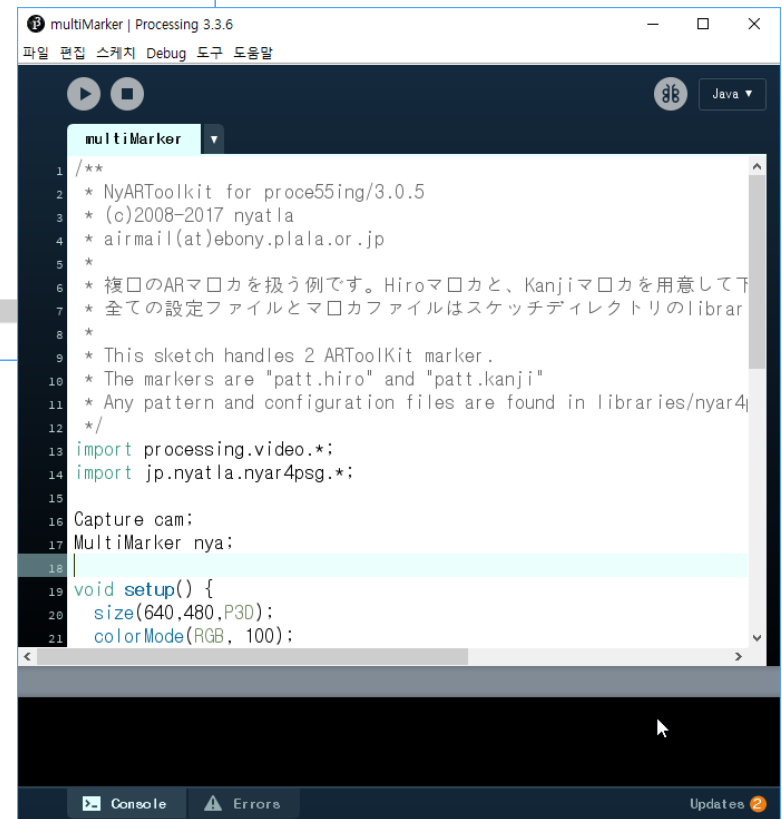
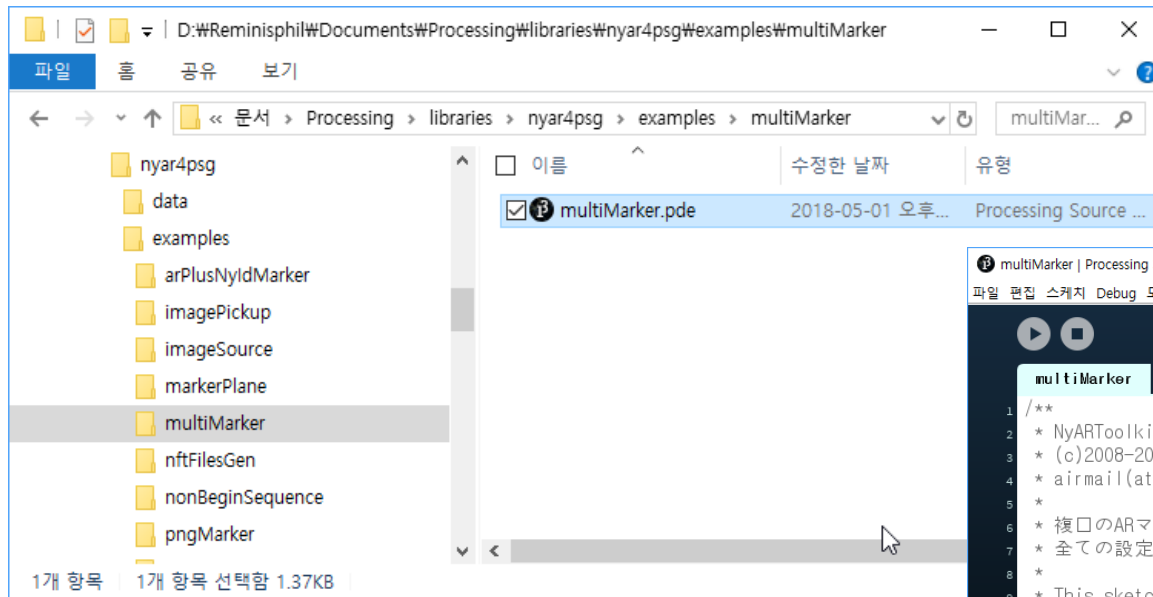


# 프로세싱에 증강현실 라이브러리 설치하기





# 증강현실 예제 실행해 보기



문서

└ Processing

└ libraries

└ nyar4psg

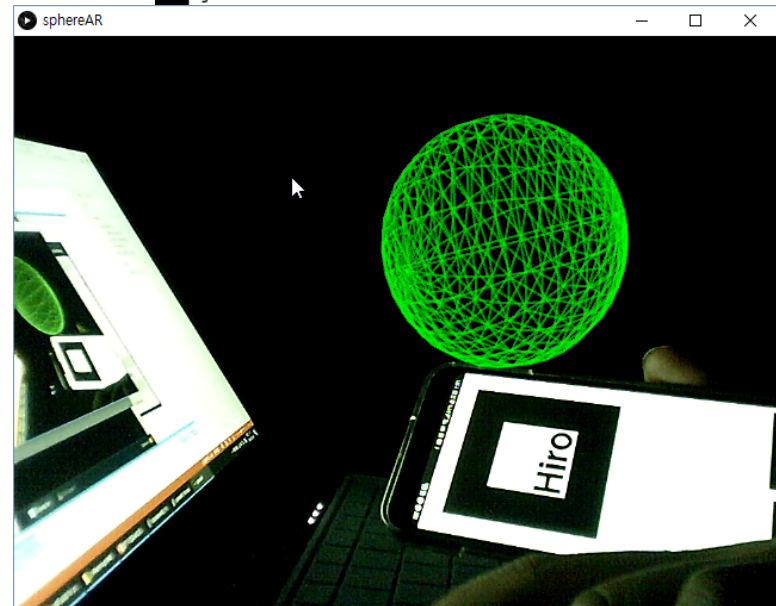
└ examples

└ multiMarker

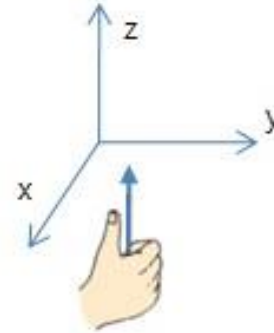
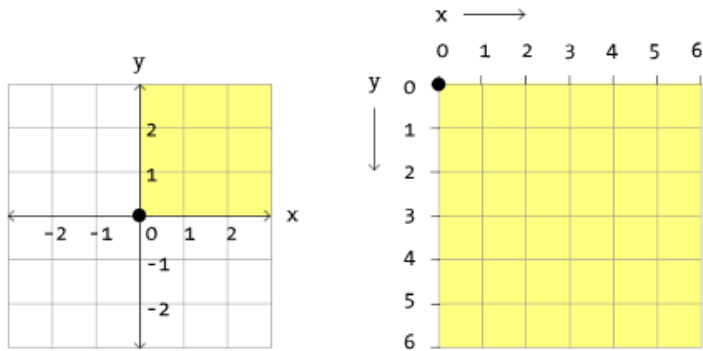
# Sphere를 증강현실로

```
1 import processing.video.*;
2 import jp.nyatla.nyar4psg.*;
3 Capture cam;
4 MultiMarker nyAR;
5
6 void setup(){
7   size(640, 480, P3D);
8   cam = new Capture(this, width, height, "HD 720P Webcam");
9   nyAR = new MultiMarker(this, width, height,
10     "data/camera_para.dat", NyAR4PsgConfig.CONFIG_PSG);
11   nyAR.addARMarker("data/patt.hiro", 80);
12   cam.start();
13 }
14
15 void draw(){
16   if (cam.available() == false) return;
17   cam.read();
18
19   nyAR.detect(cam);
20   background(0);
21   nyAR.drawBackground(cam);
22
23   if(nyAR.isExist(0)) {
24     nyAR.beginTransform(0);
25     drawSphere();
26     nyAR.endTransform();
27   }
28 }
```

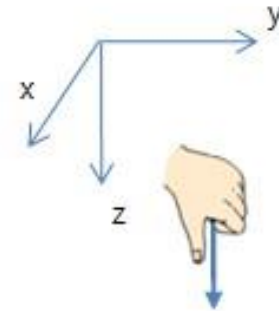
```
30 void drawSphere() {
31   stroke(0, 255, 0, 150);
32   strokeWeight(2);
33
34   pushMatrix();
35   translate(0, 0, 150);
36   rotateX(mouseY * 0.05);
37   rotateY(mouseX * 0.05);
38
39   noFill();
40   sphereDetail(20, 20);
41   sphere(50);
42   popMatrix();
43 }
```



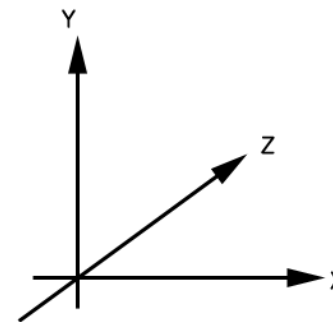
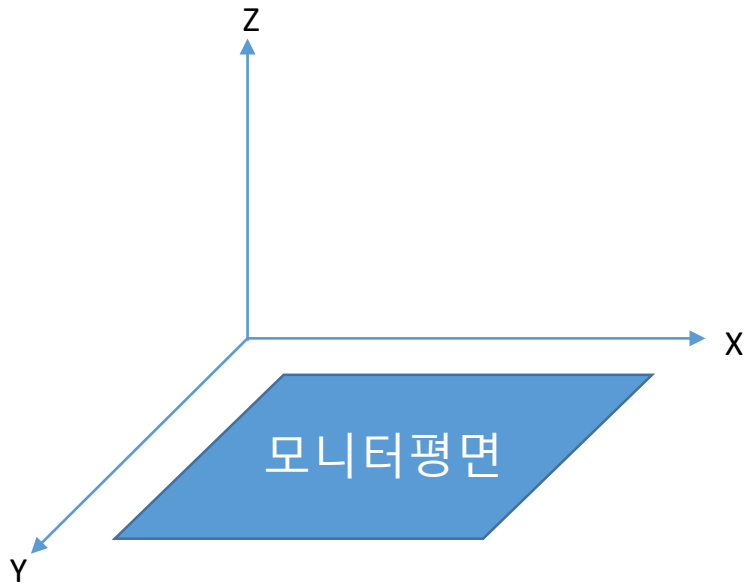
# 모니터 좌표계



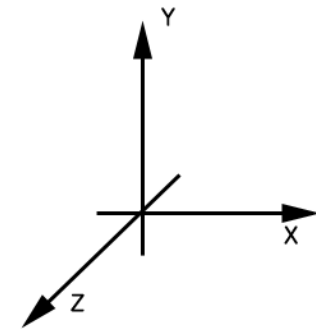
오른손 좌표계



왼손 좌표계

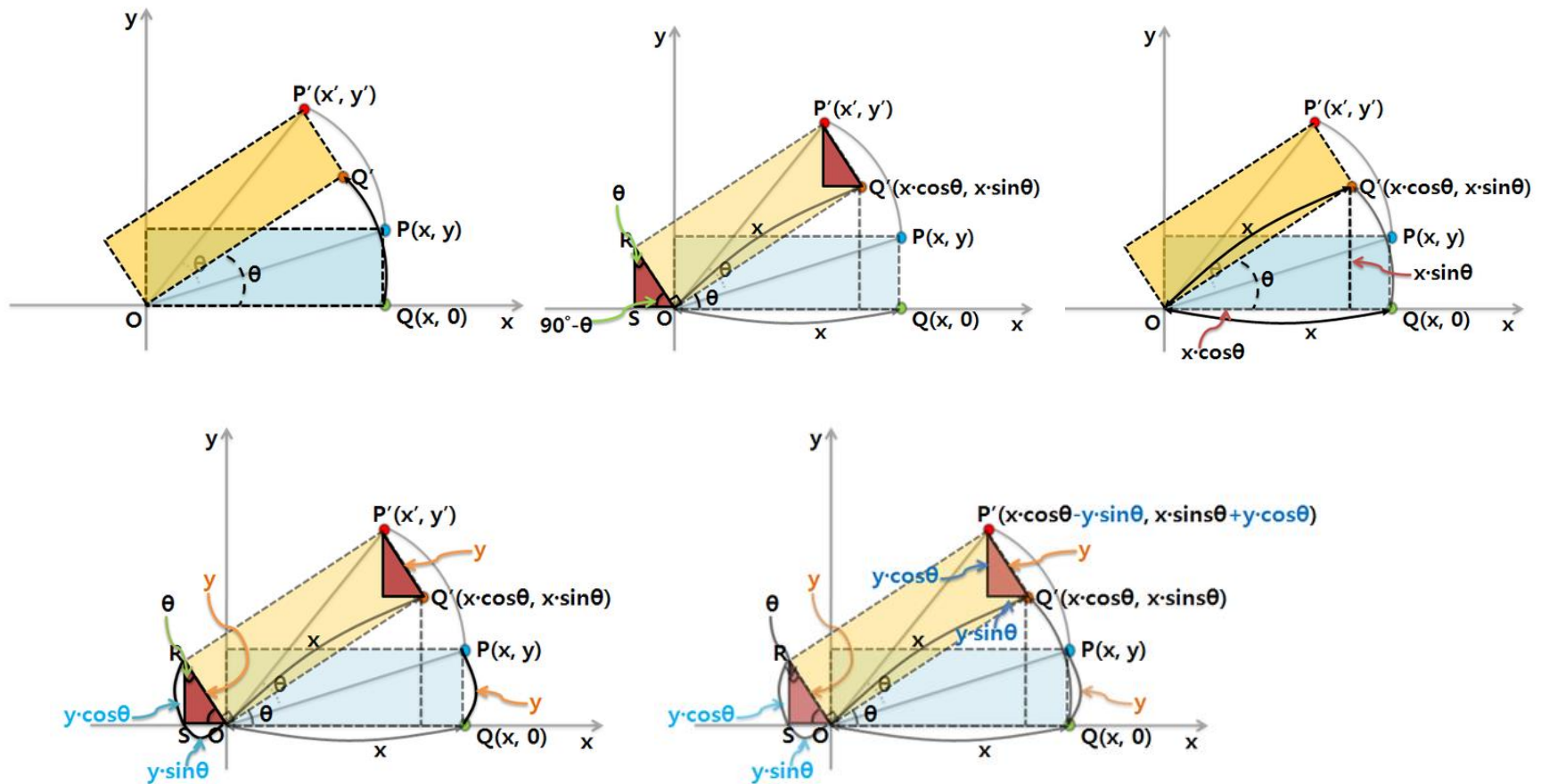


왼손 좌표계



오른손 좌표계

# 회전변환

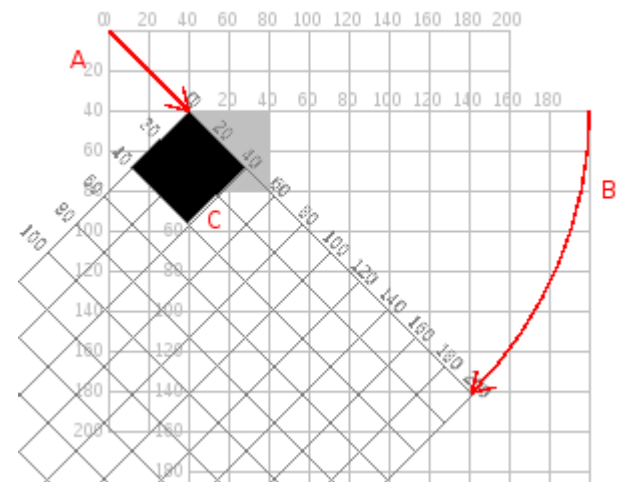
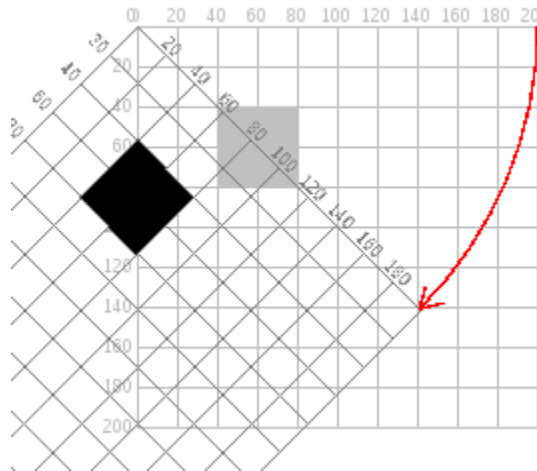
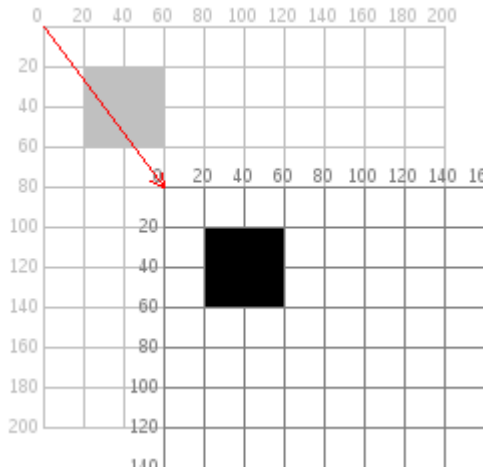


$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$



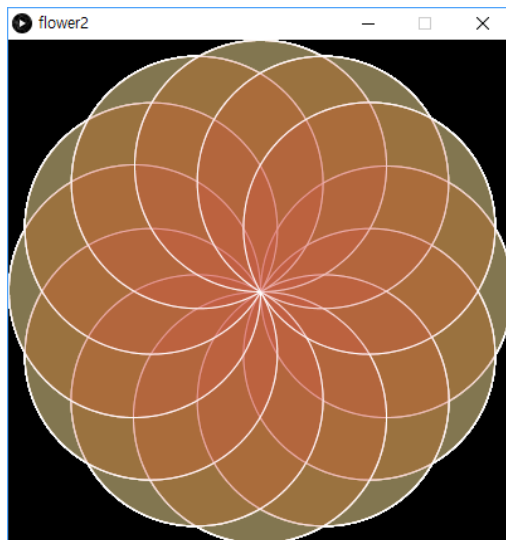
# 프로세싱의 좌표계 변환

translate(xoff, yoff)  
rotate(radian)  
pushMatrix()  
popMatrix()



# 2D 변환 비교

```
1 void setup()
2 {
3   size(400, 400);
4   ellipseMode(RADIUS);
5   background(0);
6   stroke(255);
7   frameRate(5);
8 }
9
10 void draw()
11 {
12   fill(random(255), random(255), random(255), 50);
13
14   for(int i=0; i<12; i++) {
15     float theta = radians(i*30);
16     ellipse(cos(theta)*100+200, sin(theta)*100+200, 100, 100);
17   }
18 }
```



```
1 void setup()
2 {
3   size(400, 400);
4   ellipseMode(RADIUS);
5   background(0);
6   stroke(255);
7   frameRate(5);
8 }
9
10 int i = 0;
11 void draw()
12 {
13   fill(random(255), random(255), random(255), 50);
14   pushMatrix();
15   translate(200, 200);
16
17   for(int i=0; i<12; i++) {
18     float theta = radians(i*30);
19     pushMatrix();
20     rotate(theta);
21     translate(100, 0);
22     ellipse(0, 0, 100, 100);
23     popMatrix();
24   }
25   popMatrix();
26 }
```

# 로봇팔 그리기 변환 비교

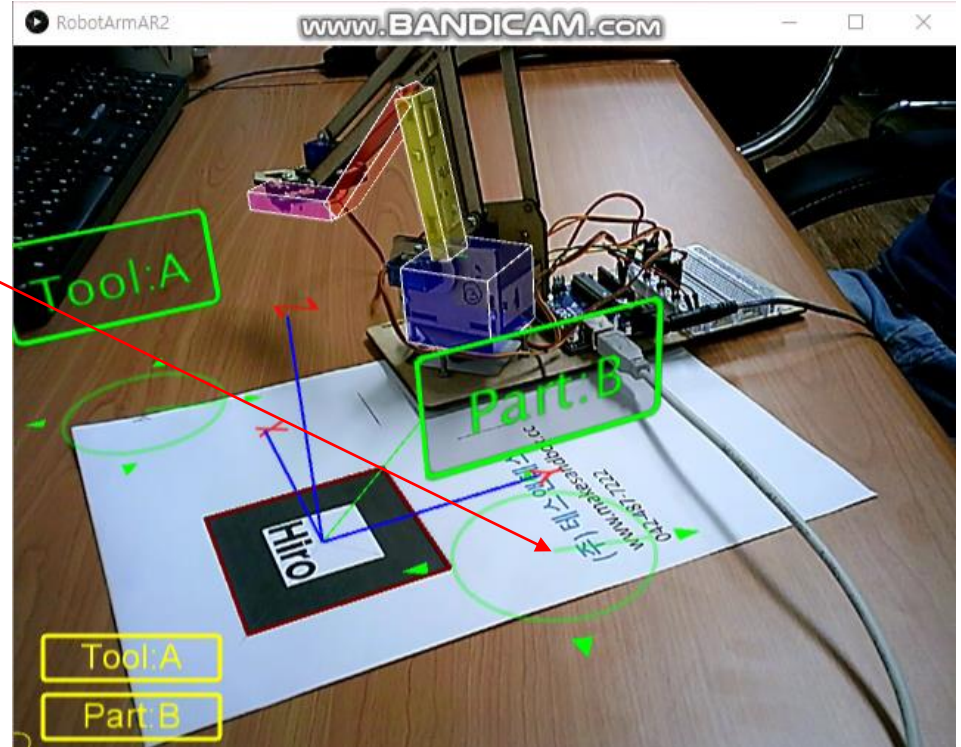
```
39 void drawArm(float A1, float A2)
40 {
41     background(100);
42     strokeWeight(5 * scale);
43     stroke(255, 255, 0, 100);
44     fill(255, 255, 0);
45
46     PVector p0 = new PVector(0.0, height-40*scale);
47     PVector p1 = new PVector(a, 0.0);
48     PVector p2 = new PVector(-b*cos(A1), -b*sin(A1));
49     PVector p3 = new PVector(-c*cos(A2), +c*sin(A2));
50     PVector p4 = new PVector(d, 0.0);
51
52     p1.x += p0.x; p1.y += p0.y;
53     p2.x += p1.x; p2.y += p1.y;
54     p3.x += p2.x; p3.y += p2.y;
55     p4.x += p3.x; p4.y += p3.y;
56
57     line(p0.x, p0.y, p1.x, p1.y);
58     line(p1.x, p1.y, p2.x, p2.y);
59     line(p2.x, p2.y, p3.x, p3.y);
60     line(p3.x, p3.y, p4.x, p4.y);
61 }
```

```
41 void drawArm(int A1, int A2)
42 {
43     background(100);
44     strokeWeight(5 * scale);
45     stroke(255, 255, 0, 100);
46     fill(255, 255, 0);
47
48     pushMatrix();
49     translate(0, height-40*scale);
50     line(0, 0, a, 0);
51     translate(a, 0);
52     rotate(radians(180+A1));
53     line(0, 0, b, 0);
54     pushMatrix();
55     translate(b, 0);
56     rotate(radians( - (A2 + A1)));
57     line(0, 0, c, 0);
58     pushMatrix();
59     translate(c, 0);
60     rotate(radians(180+A2));
61     line(0, 0, d, 0);
62     popMatrix();
63     popMatrix();
64     popMatrix();
65 }
```

# MARK class

위치 : x, y  
높이 : z  
반지름 : r  
판 크기 : w, h

```
1 class MARK {  
2     float x = 0, y = 0, z = 0;  
3     float r = 1, theta = 0;  
4     float w = 1, h = 1;  
5     float scale = 1.0;  
6     String textStr = "";  
7  
8     PGraphics pg;  
9  
10    MARK(){  
11        init(0, 0, 0, 1, 1, 1, "");  
12    }  
13  
14    MARK(float _x, float _y, float _z, float _r,  
15         float _w, float _h, String _text) {  
16        init(_x, _y, _z, _r, _w, _h, _text);  
17    }  
}
```





# 더블버퍼링 : PGraphics

```
void init(float _x, float _y, float _z, float _r,
float _w, float _h, String _text) {
  x = _x;    y = _y;    z = _z;
  r = _r;
  w = _w;    h = _h;
  textStr = _text;
}
```

```
pg = createGraphics(int(w)+4, int(h)+4);
pg.beginDraw();
pg.pushMatrix();
pg.translate(w/2, h/2);
```

```
pg.stroke(0, 255, 0);
pg.strokeWeight(3);
```

```
pg.fill(0, 0, 0, 50);
pg.rectMode(CENTER);
pg.rect(0, 0, w, h, 5);
```

```
pg.fill(0, 255, 0);
pg.textAlign(CENTER, CENTER);
pg.textSize(24);
pg.text(textStr, 0, 0);
```

```
pg.popMatrix();
pg.endDraw();
}
```

```
void show() {
  pushStyle();

  stroke(0, 255, 0, 90);
  strokeWeight(3);

  pushMatrix();
  translate(x*scale, y*scale, 0);

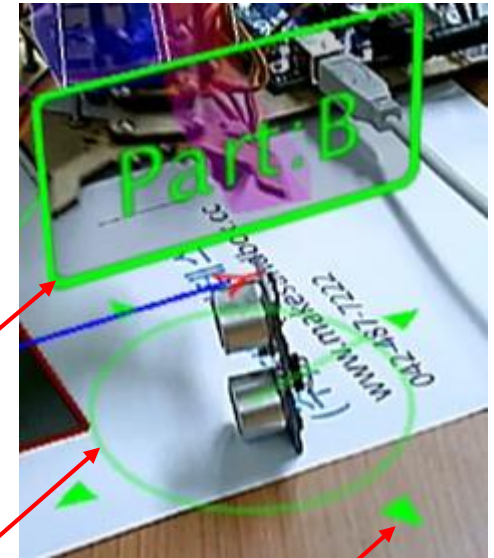
  pushMatrix();
  translate(0, 0, z*scale);
  rotateX(-radians(90));
  rotateY(-radians(90));

  imageMode(CENTER);
  image(pg, 0, 0, w*scale, h*scale);

  popMatrix();

  rotate(radians(theta++));

  noFill();
  ellipse(0, 0, r*scale, r*scale);
  line(0, 0, r/2*scale, 0);
}
```



```
final int interval = 4;
noStroke();
fill(0, 255, 0);
for(int i=0; i<interval; i++) {
  rotate(i * radians(360/interval));
  pushMatrix();
  translate(r/2 + 5, 0, 0);
  triangle(15, 0, 5, 4, 5, -4);
  popMatrix();
}

popMatrix();
popStyle();
}
```



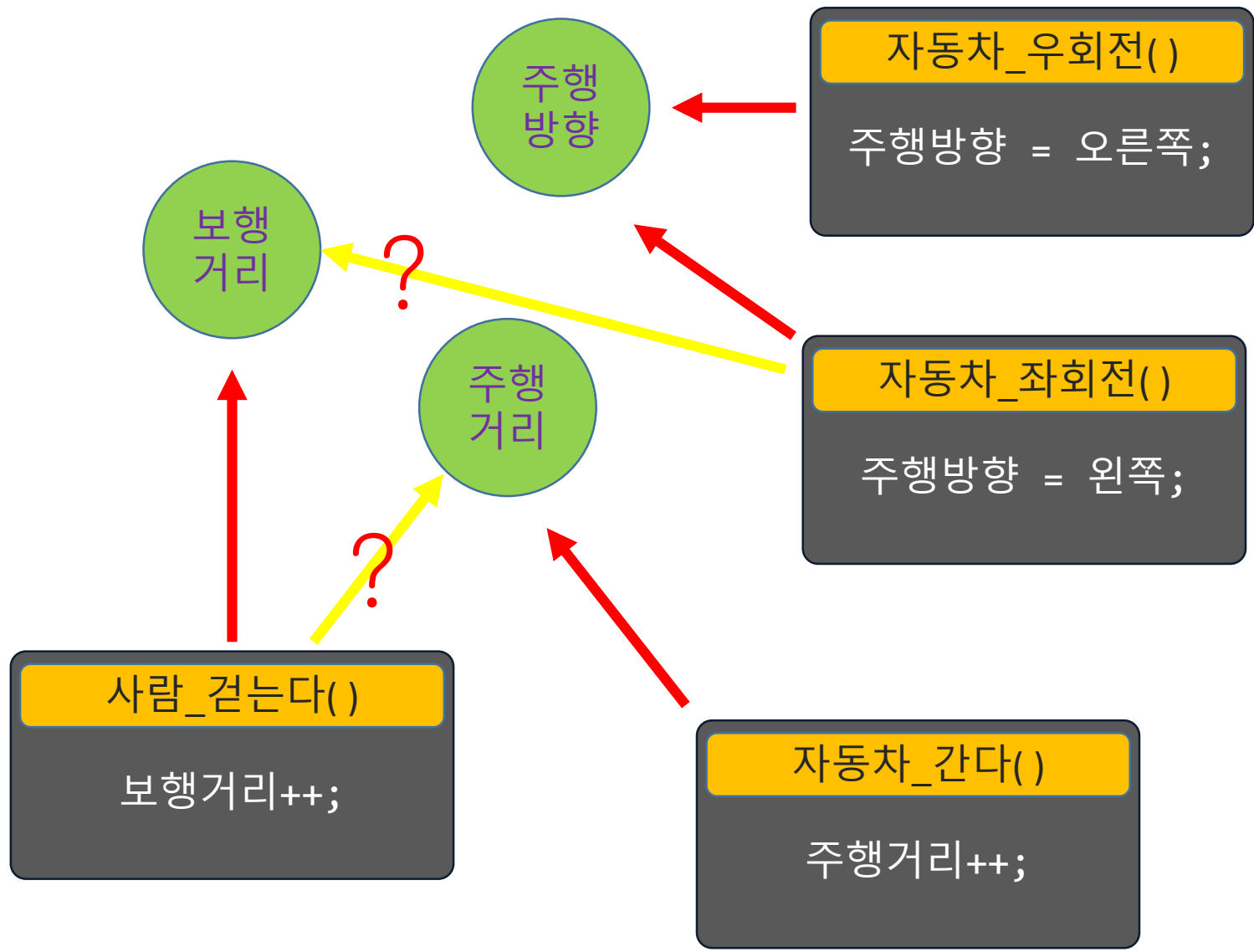
## Augmented Reality - Industrial Robotics

Fachgebiet Industrielle Automatisierungstechnik  
[lambrecht@iwf.tu-berlin.de](mailto:lambrecht@iwf.tu-berlin.de)

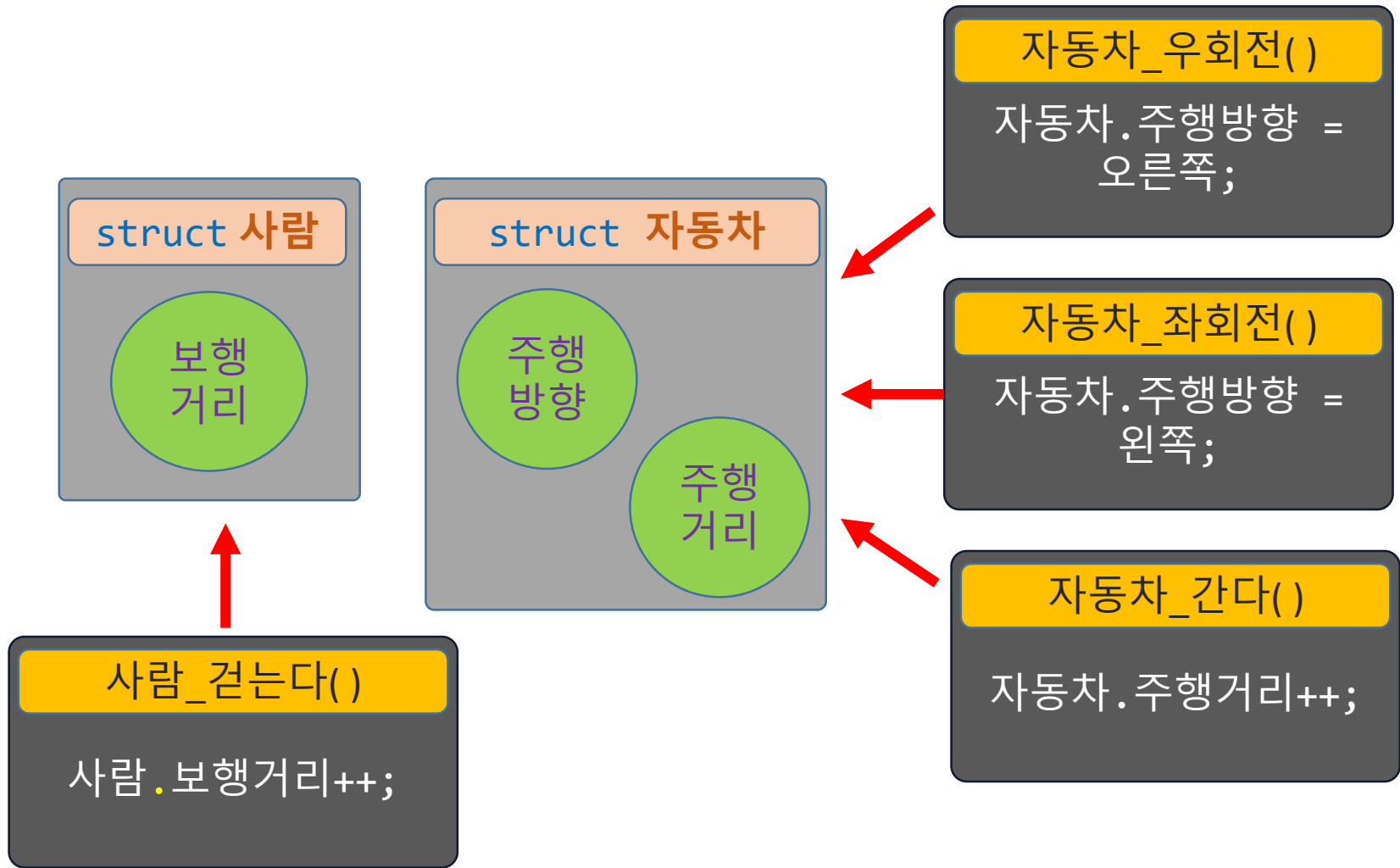
ARToolKitで  
距離を表示してみた

class

# 함수와 변수의 혼돈

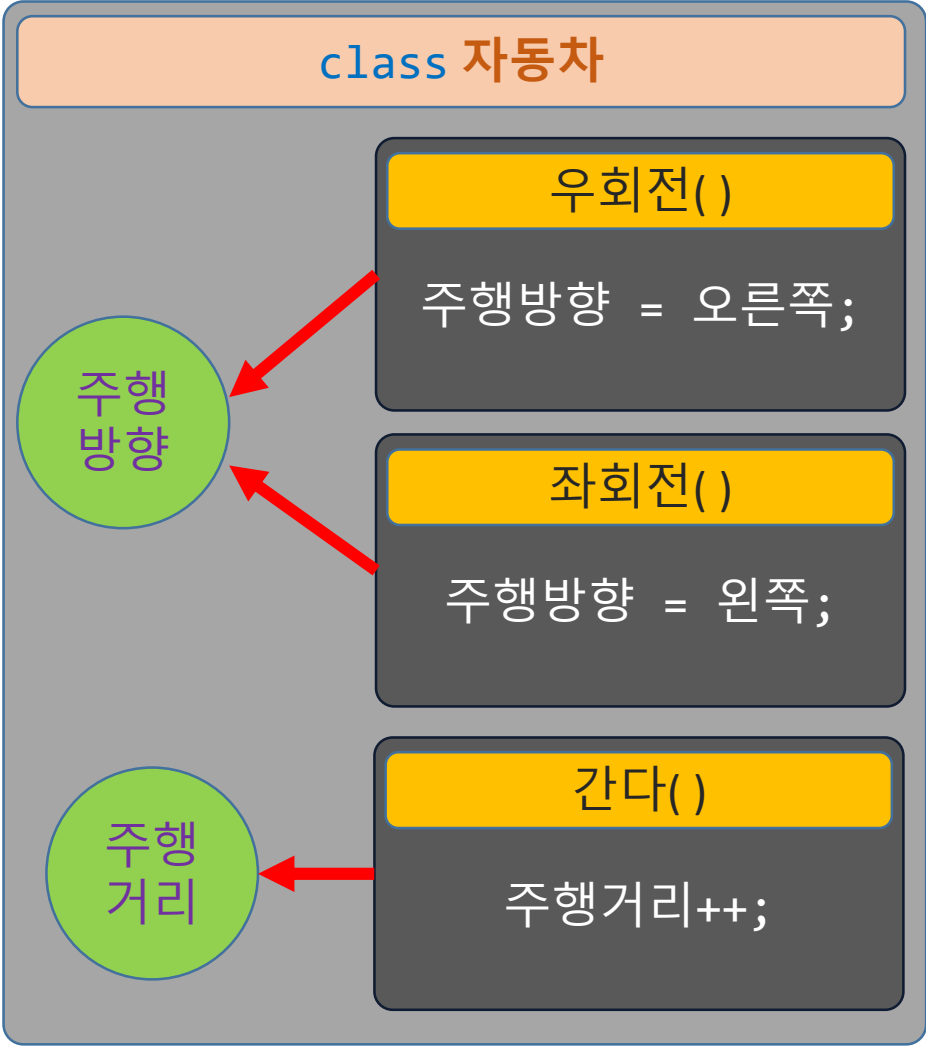
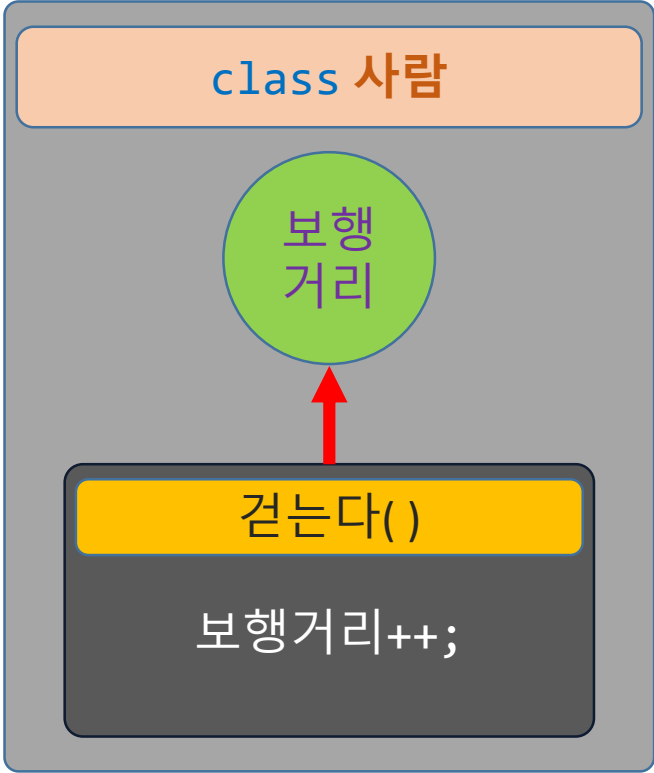


# 구조적 프로그래밍

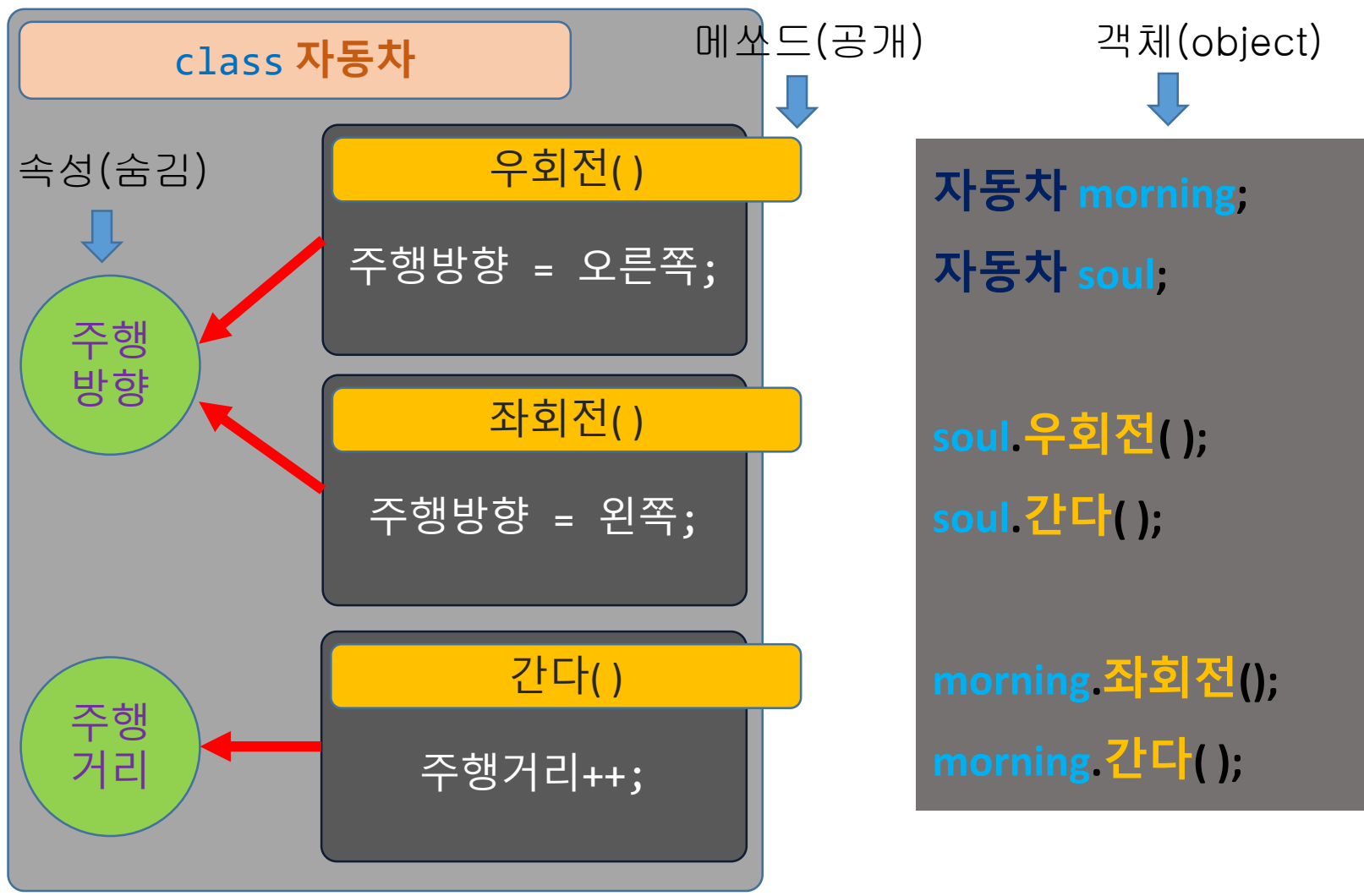




# 객체 지향 프로그래밍



# 숨길것과 보일것



# 클래스와 객체

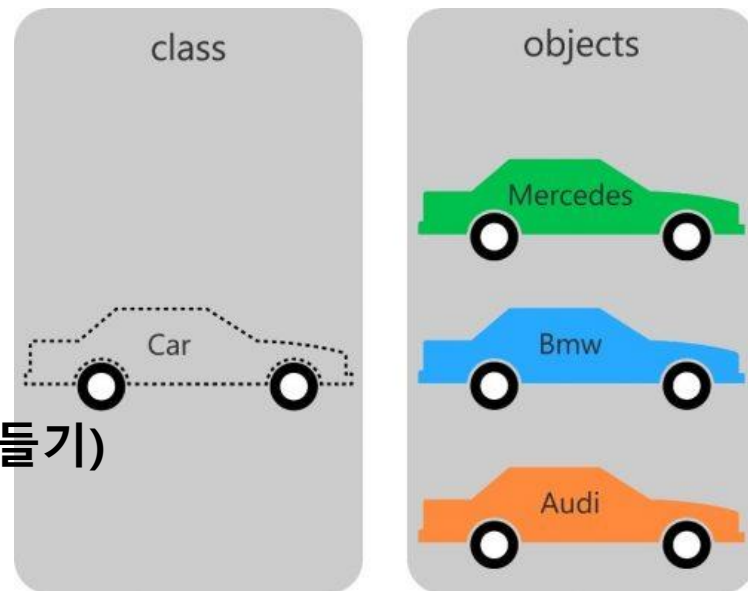
객체를 만들기 위해 반드시 클래스를 먼저 만들어야 한다.  
클래스는 객체를 만들기 위한 일종의 틀이다.

- ✓ 붕어빵이 객체라면, 붕어빵 틀은 클래스
- ✓ 자동차 클래스 선언(붕어빵 틀 만들기)

```
class Car{  
    멤버변수;  
    멤버함수;  
}
```

Car class로 객체 생성하기 (진짜 자동차 만들기)

```
Car c1 = new Car();  
Car c2 = new Car();
```



- ✓ new 연산자는 메모리에 객체를 생성. 객체를 인스턴스(instance)라고도 한다.
- ✓ 이렇게 만들어진 객체를 변수 c1 , c2 가 참조한다(메모리의 주소를 저장한다).