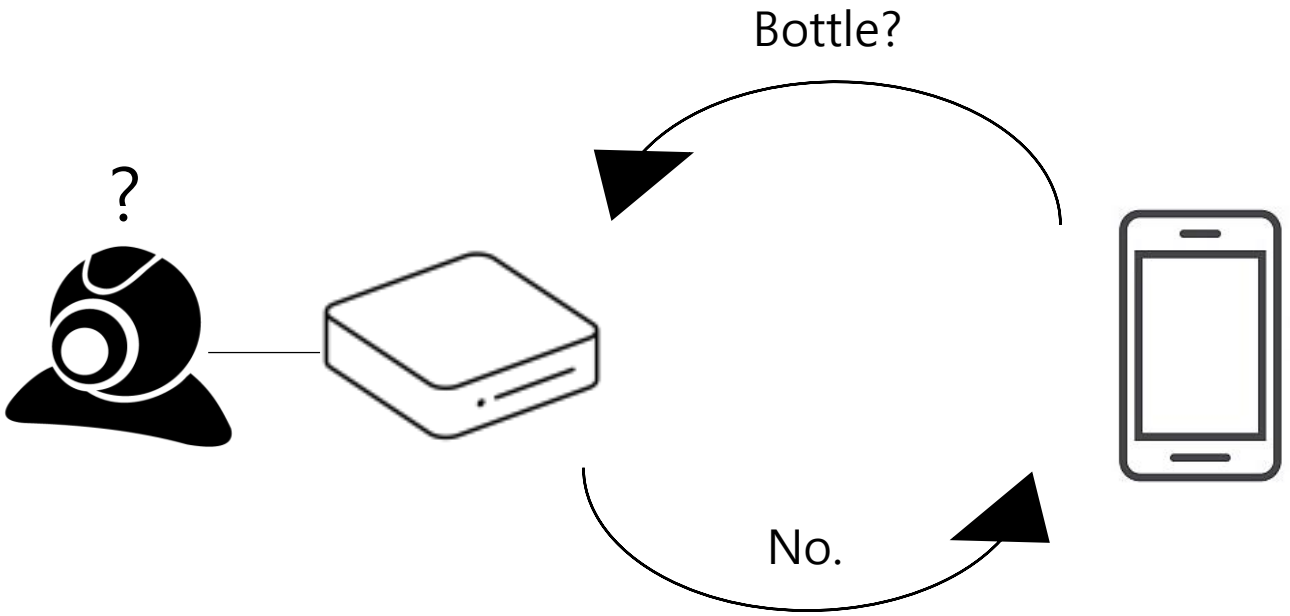
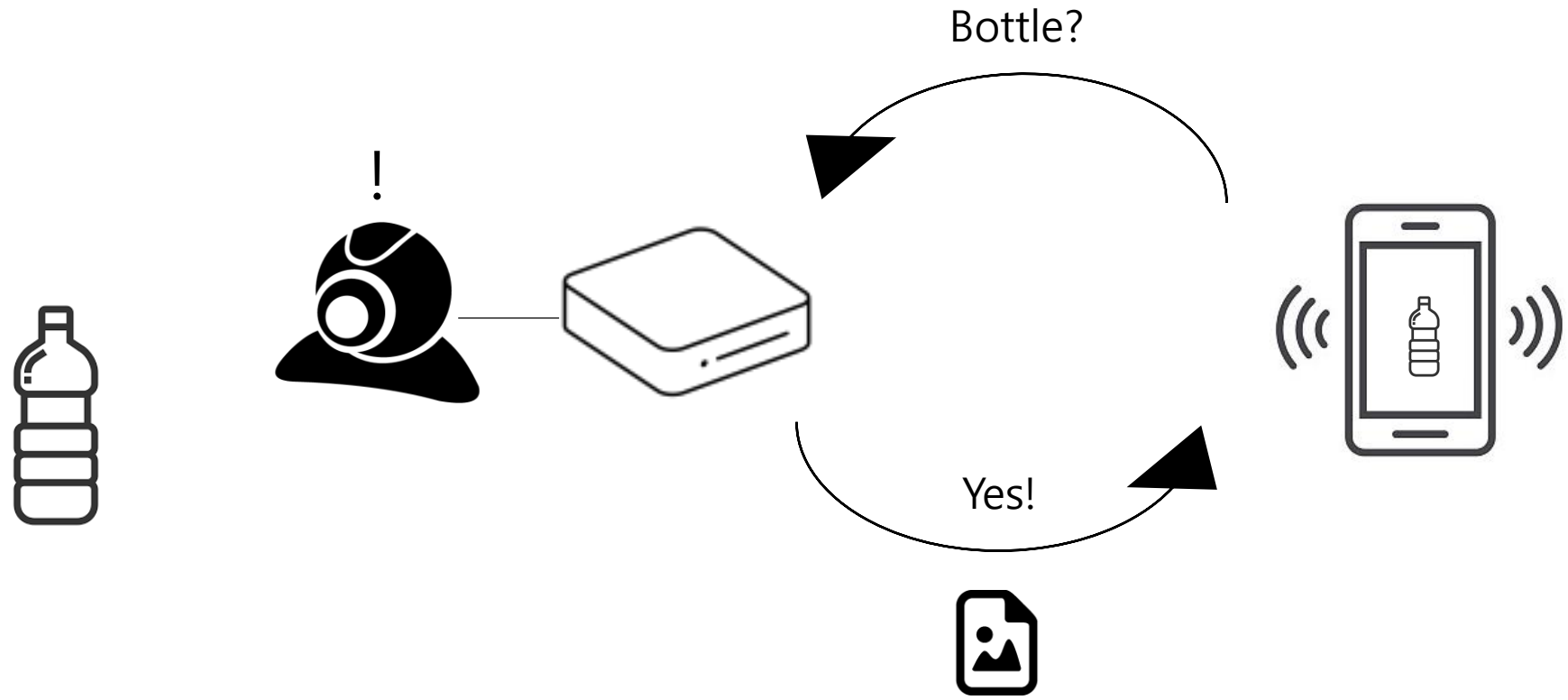
Detector 앱 흐름



Detector 앱 흐름

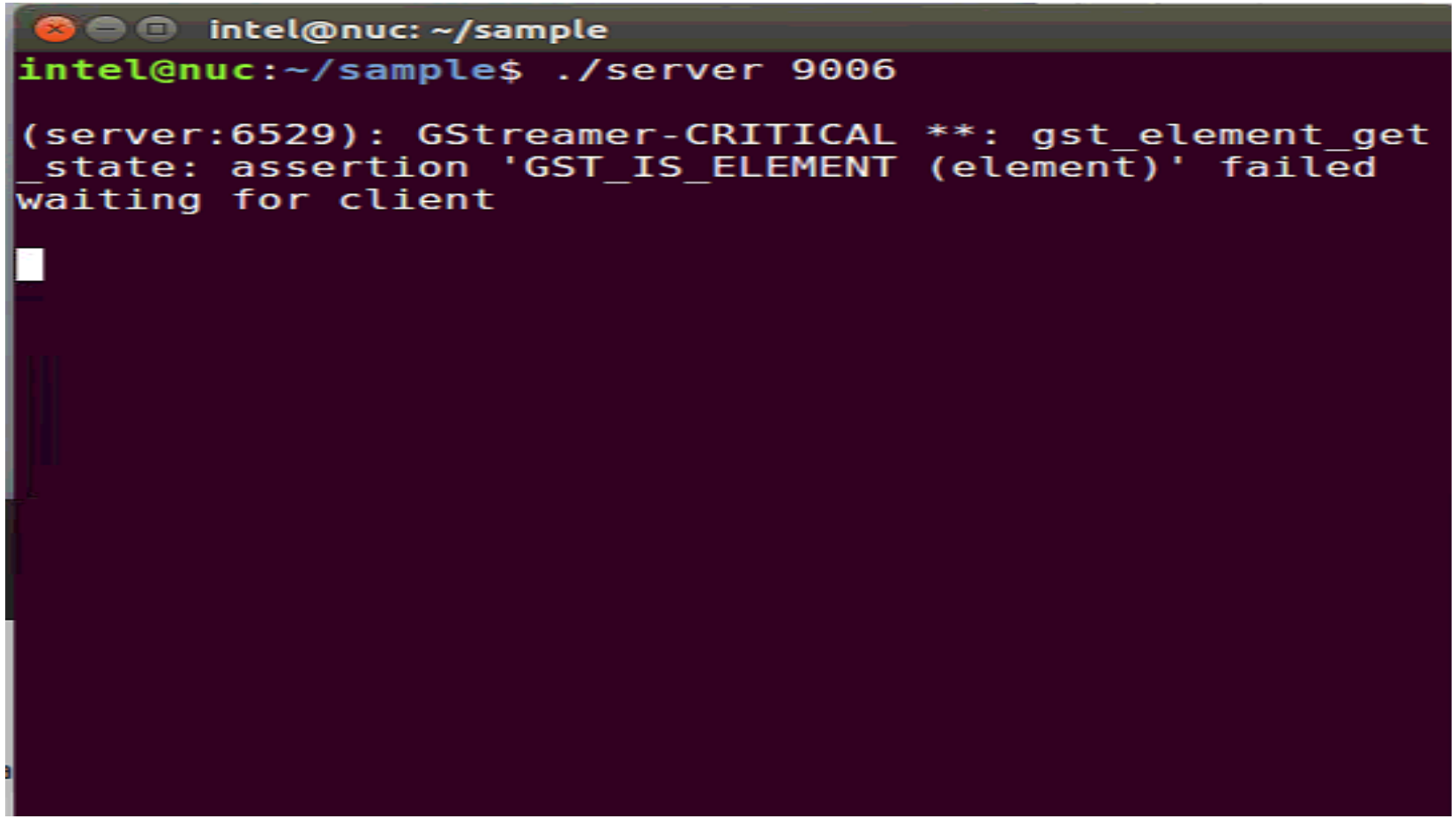


Detector 앱 설명 – 서버 실행

```
$   cd  sample
$   g++ -o server -std=c++11 main_mobilenet_server_new.cpp -
I$INTEL_CVSDK_DIR/opencv/include -
I$INTEL_CVSDK_DIR/deployment_tools/inference_engine/include -
L$IE_PLUGINS_PATH -L$INTEL_CVSDK_DIR/opencv/lib -ldl -lopencv_core -
lopencv_imgproc -lopencv_imgcodecs -linference_engine -lopencv_video -
lopencv_highgui -lopencv_videoio -lX11
```

터미널에서 sample 디렉토리로 이동후 main_mobilenet_server_new 파일을 컴파일 합니다.

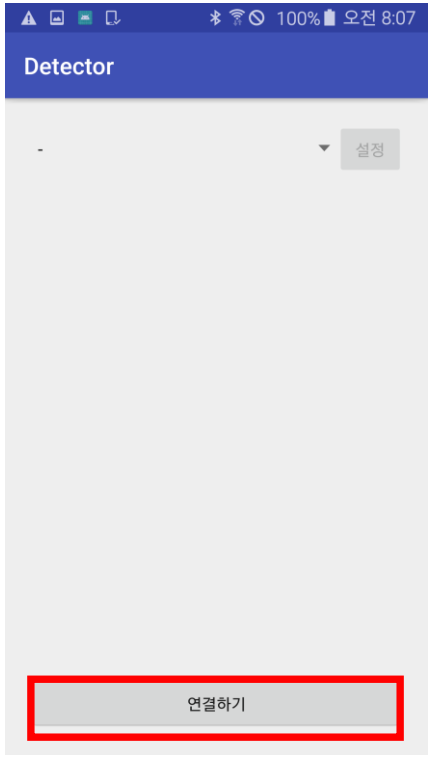
Detector 앱 설명 – 서버 실행

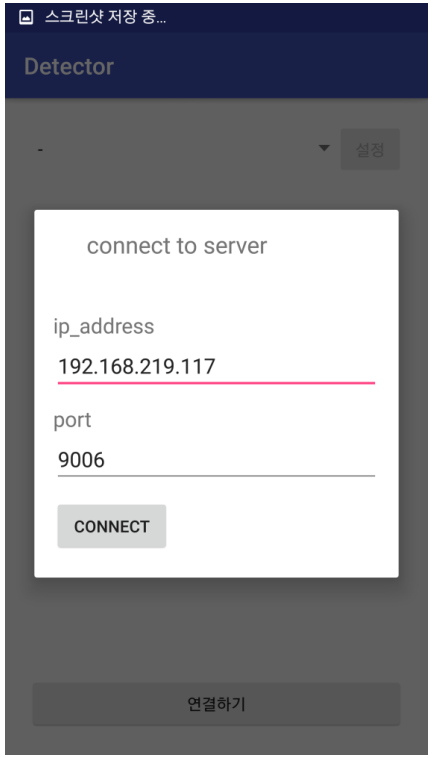


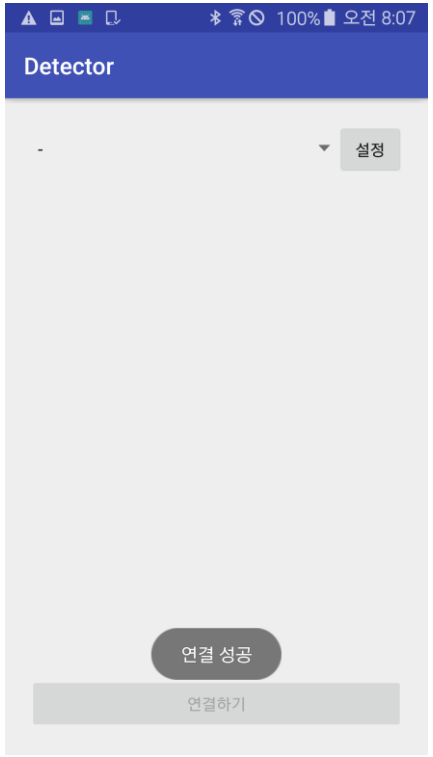
./server + (포트번호)로 서버를 실행 합니다.

waiting for client가 출력되면 클라이언트를 접속시킬 수 있습니다.

Detector 앱 설명

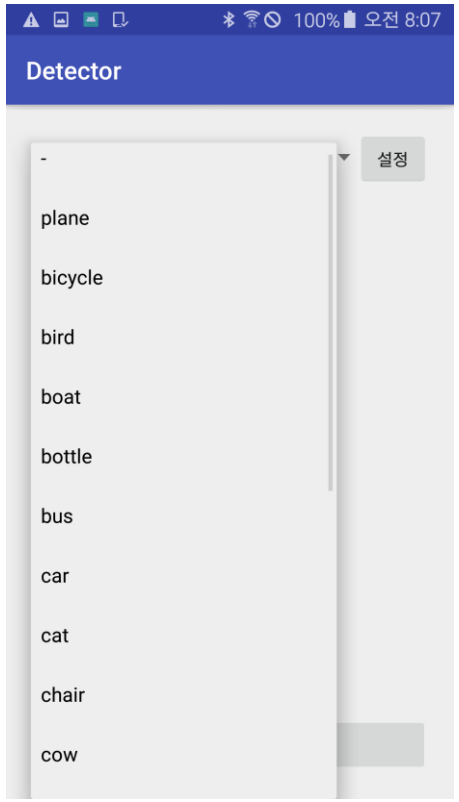


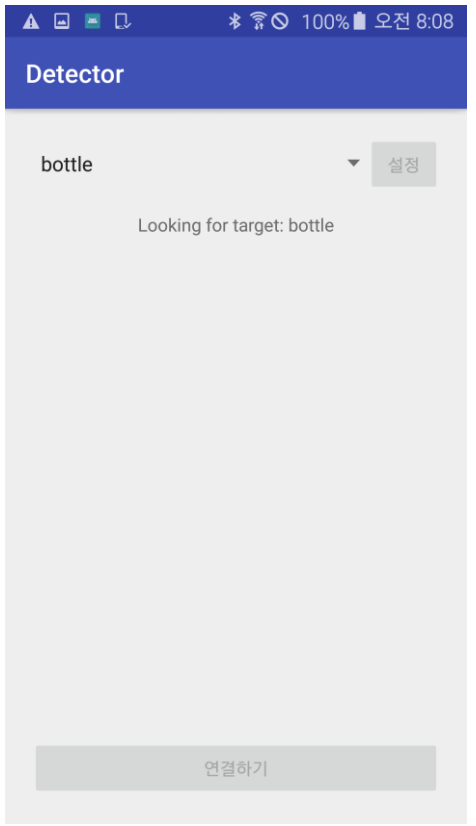




'연결하기' 클릭 후 서버 ip주소와 포트 입력

Detector 앱 설명

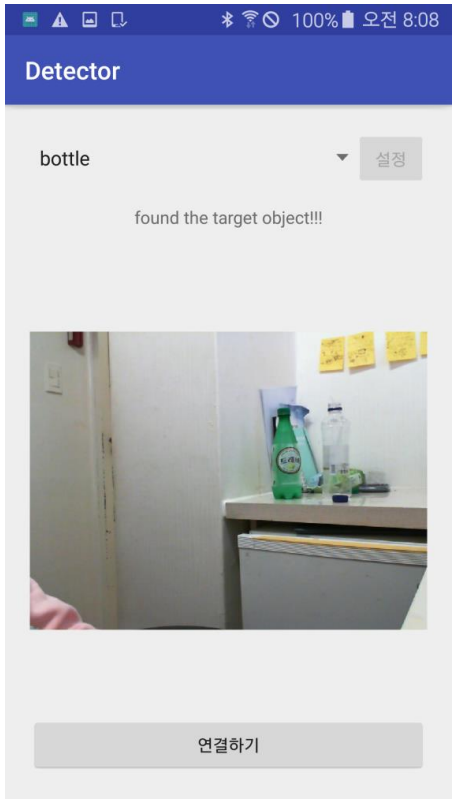


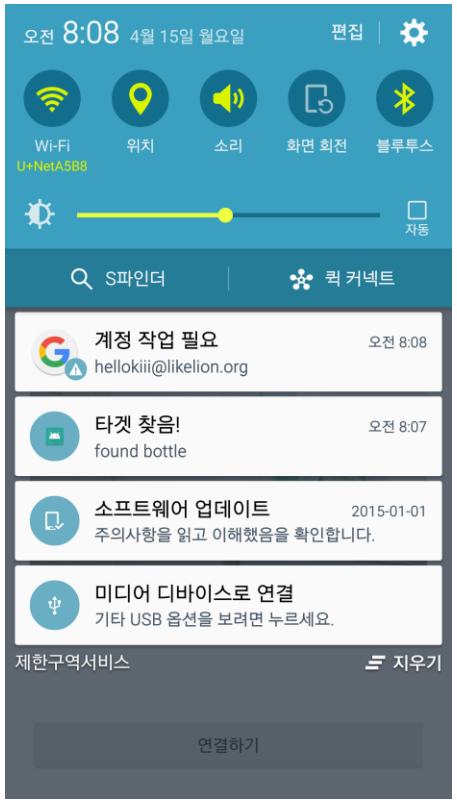


Detect 할 타겟을 설정 후 '설정' 버튼 클릭

타겟은 Detecting 중에도 수정 가능 합니다.

Detector 앱 설명

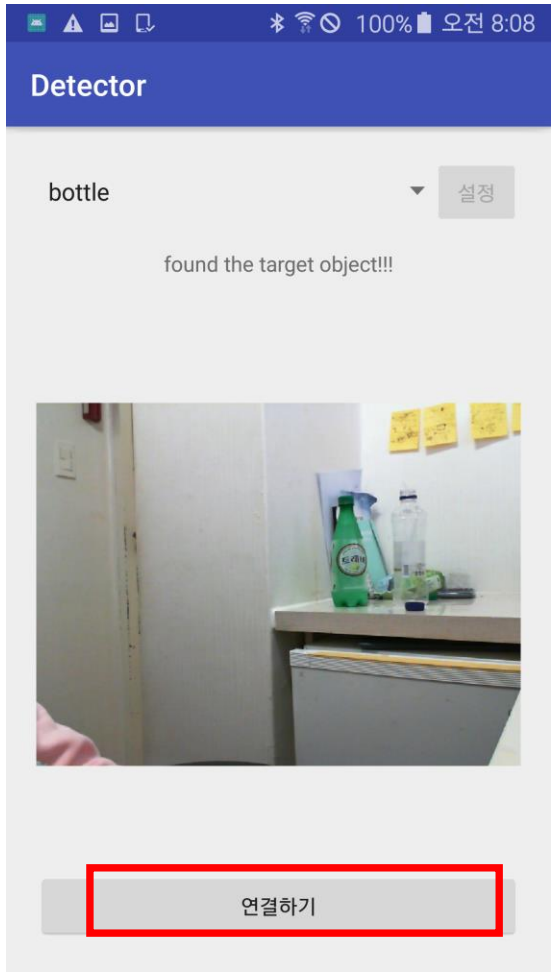


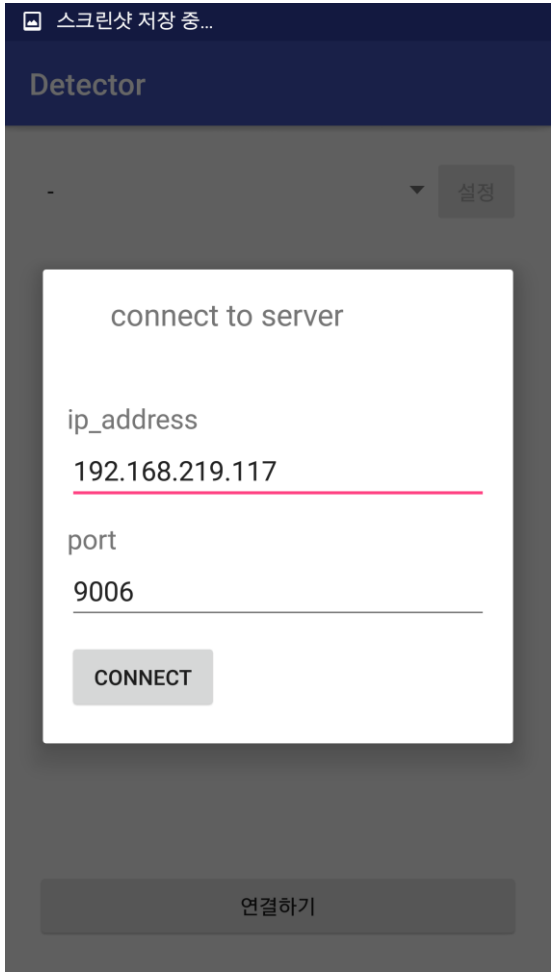


서버에서 타겟이 Detect 되면 사진 전송 합니다.

백그라운드에서 실행해도 알림이 울립니다.

Detector 앱 설명





활성화된 '연결하기' 버튼으로 서버에 재접속 가능 합니다.

안드로이드 스튜디오 – MainActivity.java

```
37
38  public class MainActivity extends AppCompatActivity {
39      private Socket socket;  //소켓생성
40      BufferedInputStream in;  //서버로부터 들어오는 input stream
41      PrintWriter out;        //서버로 데이터 전송 Writer
42      Button button;
43      Button btn_connect;
44      TextView tv_message;
45      String data;
46      ImageView imageView;
47      Spinner spinner;
48      String target = "0";
49      String targetName;
50      String ip;
51      String port;
52      Intent intent;
53      Thread socketThread;
54      RunSocket runSocket;
55      NotificationManager notificationManager;
56      NotificationCompat.Builder builder;
57
58      public void onCreate(Bundle savedInstanceState) {
59          super.onCreate(savedInstanceState);
60          setContentView(R.layout.activity_main);
61
62          // View 초기화
63          intent = new Intent(Intent.ACTION_MEDIA_SCANNER_SCAN_FILE);
64          spinner = findViewById(R.id.spinner);
65          button = findViewById(R.id.button);  //설정 버튼.
66          button.setEnabled(false);
67          tv_message = findViewById(R.id.tv_message);
68          tv_message.setVisibility(View.INVISIBLE);
69          imageView = findViewById(R.id.iv);
70
71
72          button.setOnClickListener(new View.OnClickListener() {
73              public void onClick(View v) {
74                  // 설정 버튼이 클릭되면 소켓에 타겟 번호를 출력한다.
75                  tv_message.setVisibility(View.VISIBLE);
76                  out.print(target);
77                  out.flush();
78                  button.setEnabled(false);
79
80              }
81          });
82
83          btn_connect = findViewById(R.id.btn_connect);  //연결하기 버튼
84          btn_connect.setOnClickListener(new View.OnClickListener() {  //연결하기 버튼을 누르면 dialog view를 띄운다
85              @Override
86              public void onClick(View view) {
87                  showConncetDialog();
88              }
89          });
90
```

```
90
91
92          // 타겟을 설정하는 spinner
93          // spinner에서 새 값을 선택하면 target을 업데이트 한다.
94          spinner.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
95              @Override
96              public void onItemSelected(AdapterView<?> adapterView, View view, int index, long id) {
97                  target = String.valueOf(index-1);
98                  targetName = spinner.getSelectedItem().toString();
99                  tv_message.setText("Looking for target: " + targetName);
100             }
101
102             @Override
103             public void onNothingSelected(AdapterView<?> adapterView) {
104
105             }
106         });
107
108
```

뷰와 관련된 변수, 함수 초기화 합니다.

안드로이드 스튜디오 – MainActivity.java

```
108
109         // 알림 설정
110         Uri alarmSound = RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);
111         builder = new NotificationCompat.Builder( context: this,  channelId: "default");
112         builder.setSmallIcon(R.mipmap.ic_launcher)
113         .setContentTitle("타겟 찾음!")
114         .setDefaults(Notification.DEFAULT_SOUND)
115         .setAutoCancel(true)
116         .setSound(alarmSound);
117
118         notificationManager = (NotificationManager) this.getSystemService(Context.NOTIFICATION_SERVICE);
119         if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
120             notificationManager.createNotificationChannel(new NotificationChannel( id: "default",
121                     name: "기본 채널", NotificationManager.IMPORTANCE_DEFAULT));
122         }
123
124
125
126     }
127
```

알림 설정을 합니다.

안드로이드 스튜디오 – MainActivity.java

```java
    // 서버 ip주소와 포트 설정하는 창 띄워주는 함수
    public void showConncetDialog() {
        final AlertDialog.Builder builder = new AlertDialog.Builder( context: this);
        LayoutInflater inflater = getLayoutInflater();
        View view = inflater.inflate(R.layout.dialog_connect,  root: null);
        builder.setView(view);
        final Button submit = view.findViewById(R.id.btn_submit);
        final EditText et_ip = view.findViewById(R.id.et_ip);
        final EditText et_port = view.findViewById(R.id.et_port);

        // 기존에 저장되어있는 ip, 포트 값이 있으면 가져온다.
        final SharedPreferences pref = getSharedPreferences( name: "connection", MODE_PRIVATE);
        et_ip.setText(pref.getString( s: "ip",  s1: ""));
        et_port.setText(pref.getString( s: "port",  s1: ""));


        final AlertDialog dialog = builder.create();

        // connect 버튼 눌렀을 때
        submit.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                ip = et_ip.getText().toString();
                port = et_port.getText().toString();

                // 새 값을 저장한다.
                SharedPreferences.Editor editor = pref.edit();
                editor.putString( s: "ip", ip);
                editor.putString( s: "port", port);
                editor.commit();
                btn_connect.setEnabled(false);

                // 소켓 스레드가 돌아가고 있으면 끄고 새로 시작한다.
                if(socketThread != null && socketThread.isAlive()) socketThread.interrupt();
                runSocket = new RunSocket();
                socketThread = new Thread(runSocket);
                socketThread.start();

                // 창 닫기
                dialog.dismiss();

            }
        });

        dialog.show();
    }
```

소켓통신 연결창을 띄우는 함수

안드로이드 스튜디오 – MainActivity.java

```java
    // 서버 ip주소와 포트 설정하는 창 띄워주는 함수
    public void showConncetDialog() {
        final AlertDialog.Builder builder = new AlertDialog.Builder( context: this);
        LayoutInflater inflater = getLayoutInflater();
        View view = inflater.inflate(R.layout.dialog_connect,  root: null);
        builder.setView(view);
        final Button submit = view.findViewById(R.id.btn_submit);
        final EditText et_ip = view.findViewById(R.id.et_ip);
        final EditText et_port = view.findViewById(R.id.et_port);

        // 기존에 저장되어있는 ip, 포트 값이 있으면 가져온다.
        final SharedPreferences pref = getSharedPreferences( name: "connection", MODE_PRIVATE);
        et_ip.setText(pref.getString( s: "ip",  s1: ""));
        et_port.setText(pref.getString( s: "port",  s1: ""));


        final AlertDialog dialog = builder.create();

        // connect 버튼 눌렀을 때
        submit.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                ip = et_ip.getText().toString();
                port = et_port.getText().toString();

                // 새 값을 저장한다.
                SharedPreferences.Editor editor = pref.edit();
                editor.putString( s: "ip", ip);
                editor.putString( s: "port", port);
                editor.commit();
                btn_connect.setEnabled(false);

                // 소켓 스레드가 돌아가고 있으면 끄고 새로 시작한다.
                if(socketThread != null && socketThread.isAlive()) socketThread.interrupt();
                runSocket = new RunSocket();
                socketThread = new Thread(runSocket);
                socketThread.start();

                // 창 닫기
                dialog.dismiss();

            }
        });

        dialog.show();
    }
```

소켓통신 연결창을 띄우는 함수

안드로이드 스튜디오 – MainActivity.java

```
177     class RunSocket implements Runnable {
178
179         // 소켓통신 스레드 실행
180         public void run() {
181
182             // 소켓 통신 connect
183             try {
184                 socket = new Socket(ip, Integer.parseInt(port));
185                 out = new PrintWriter(socket.getOutputStream(), autoFlush: true); //데이터를 전송시 stream 형태로 변환하여 전송
186                 in = new BufferedInputStream(socket.getInputStream());
187
188                 runOnUiThread(new Runnable() {
189                     @Override
190                     public void run() {
191                         btn_connect.setEnabled(false);
192                         button.setEnabled(true);
193                         imageView.setVisibility(View.INVISIBLE);
194                         tv_message.setVisibility(View.INVISIBLE);
195                         Toast.makeText( context: MainActivity.this,  text: "연결 성공", Toast.LENGTH_LONG).show();
196                     }
197                 });
198
199             } catch (IOException e) {
200                 runOnUiThread(new Runnable() {
201                     @Override
202                     public void run() {
203                         btn_connect.setEnabled(true);
204                         button.setEnabled(false);
205                         Toast.makeText( context: MainActivity.this,  text: "연결 실패", Toast.LENGTH_LONG).show();
206                     }
207                 });
208
209             }
210
```

서버 소켓에 connect

안드로이드 스튜디오 – MainActivity.java

```
210
211         // 소켓통신 Loop
212         try {
213             while (true) {
214
215                 // 데이터가 들어오면 읽음
216                 int temp;
217                 if((temp = in.read()) > 0) {
218                     data = String.valueOf((char)temp);
219                         if (data.equals("b")) {  // 서버에서 아직 찾지 못하면 "b"를 받는다.
220                         out.print(target);
221                         out.flush();
222                     } else if(data.equals("a")) {  // 서버에서 타겟을 발견함
223                         tv_message.post(new Runnable() {
224                             public void run() {
225                                 // 메세지 표시하고 알림을 띄운다.
226                                 tv_message.setText("found the target object!!!");
227                                 builder.setContentText("found " + targetName);
228                                 notificationManager.notify( id: 1, builder.build());
229                             }
230                         });
231
```

서버와 통신하는 루프 타겟을 발견하면 서버로부터 “a”, 아니면 “b”를 받습니다.

매 루프마다 타겟을 다시 전송해주고, 타겟을 발견하면 메세지와 알림을 띄움니다.

안드로이드 스튜디오 – MainActivity.java

```
231
232
233             // 사진을 저장할 임시 파일을 만든다.
234             File storage = getApplicationContext().getCacheDir();
235             String fileName = "tempFile.jpeg";
236
237             final File tempFile = new File(storage, fileName);
238             tempFile.createNewFile();
239
240             // 이미지 데이터 수신해 파일에 저장
241             BufferedOutputStream bos = new BufferedOutputStream(
242                     new FileOutputStream(tempFile),  size: 1024);
243             while (true) {
244                 int x = in.read();
245                 if(x == -1) break;
246                 bos.write(x);
247             }
248
249             in.close();
250             bos.close();
251
```

사진을 저장할 임시파일을 만들고, 서버로부터 사진 데이터를 받아옵니다.

안드로이드 스튜디오 – MainActivity.java

```
252                     // 이미지 띄우기
253                     imageView.post(new Runnable() {
254                         @Override
255                         public void run() {
256                             Bitmap bitmap = BitmapFactory.decodeFile(tempFile.getAbsolutePath());
257                             imageView.setImageBitmap(bitmap);
258                             tempFile.delete();
259
260                             imageView.setVisibility(View.VISIBLE);
261                             btn_connect.setEnabled(true);
262                             button.setEnabled(false);
263                             try {
264                                 // 소켓 닫고 스레드 정지
265                                 socket.close();
266                                 socketThread.interrupt();
267                             } catch (IOException e) {
268                                 e.printStackTrace();
269                             }
270                         }
271                     });
272                     break;
273                 }
274
```

저장된 이미지를 뷰에 띄웁니다.

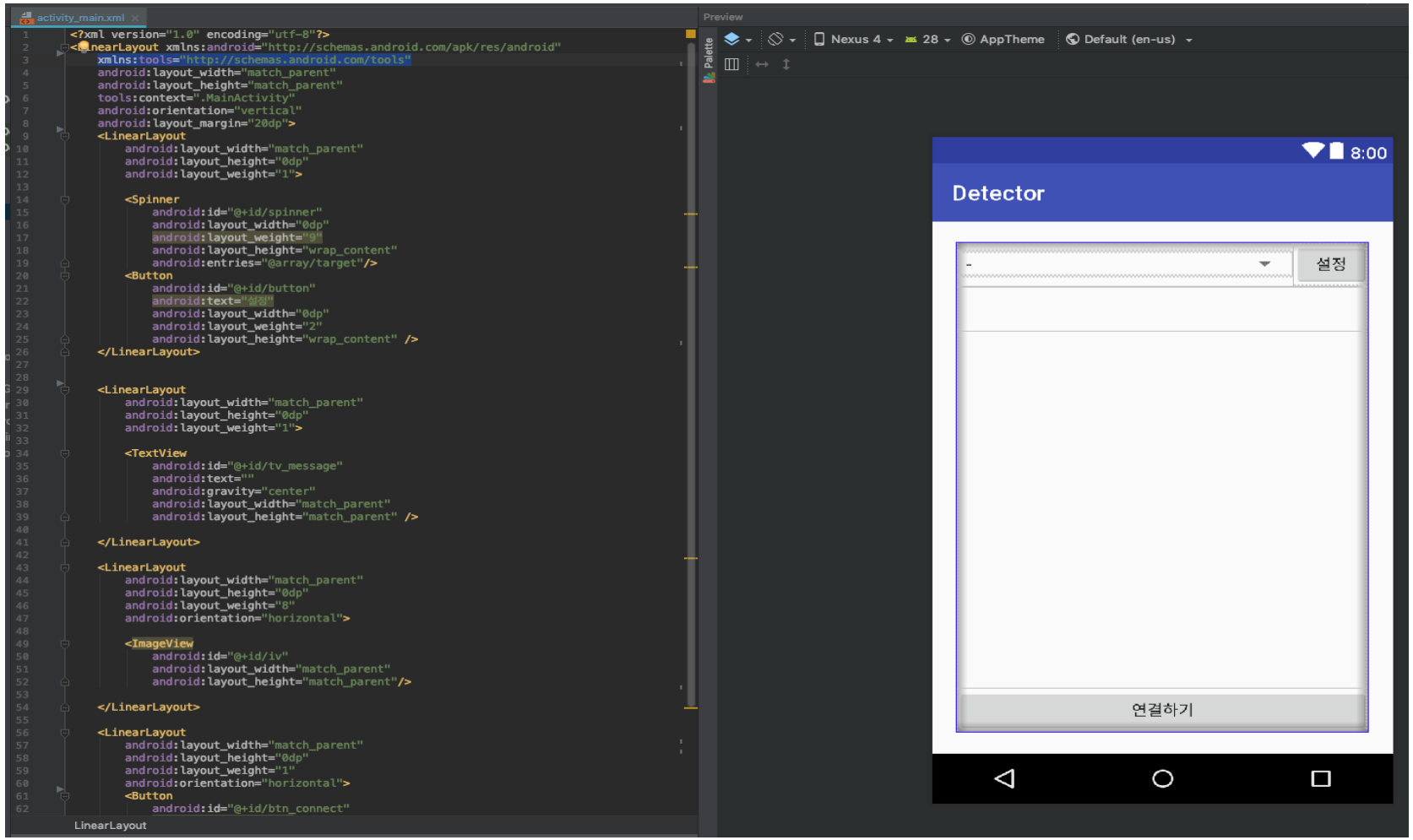
안드로이드 스튜디오 – array.xml

array.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="target">
        <item>-</item>
        <item>plane</item>
        <item>bicycle</item>
        <item>bird</item>
        <item>boat</item>
        <item>bottle</item>
        <item>bus</item>
        <item>car</item>
        <item>cat</item>
        <item>chair</item>
        <item>cow</item>
        <item>table</item>
        <item>dog</item>
        <item>horse</item>
        <item>motorcycle</item>
        <item>person</item>
        <item>plant</item>
        <item>sheep</item>
        <item>sofa</item>
        <item>train</item>
        <item>monitor</item>
    </string-array>
</resources>
```

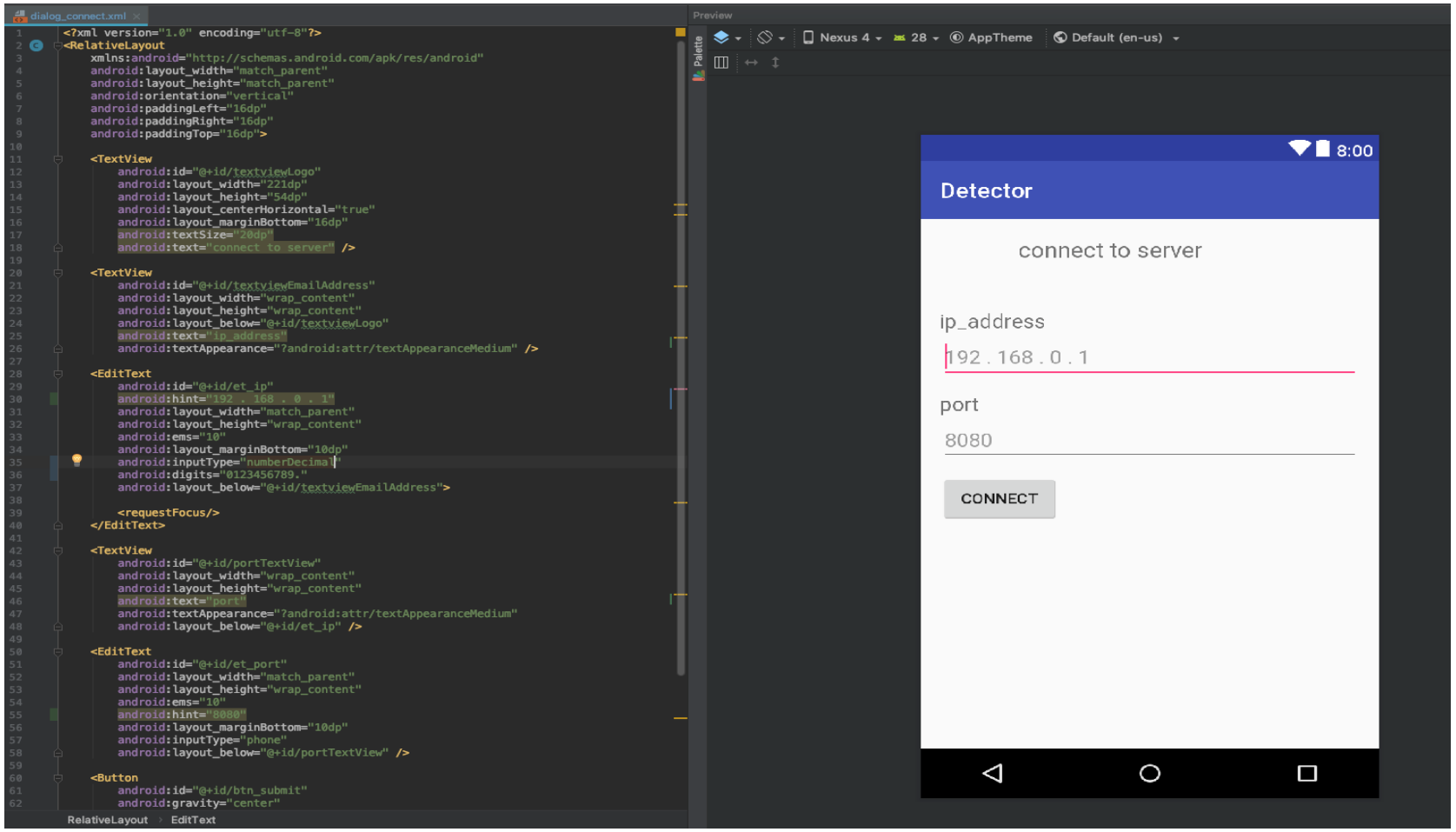
Spinner에 표시할 타겟 리스트가 저장되는 곳입니다.

안드로이드 스튜디오 – activity_main.xml



메인 뷰의 구조가 정의되는 곳입니다.

안드로이드 스튜디오 – dialog_connect.xml



ip주소와 포트설정 팝업창의 구조가 정의되는 곳입니다.

서버

```
142
143     // socket setting //
144     const int portno = atoi(argv[1]);
145     const int BUFFER_SIZE = 5;
146     int sockfd, newsockfd, n;
147
148     char buffer[BUFFER_SIZE];
149     struct sockaddr_in serv_addr, cli_addr;
150     socklen_t clilen;
151
152
153     sockfd = socket(AF_INET, SOCK_STREAM, 0);
154     if(sockfd < 0) socketError("Error opening Socket.");
155
156     bzero((char *) &serv_addr, sizeof(serv_addr));
157
158     serv_addr.sin_family = AF_INET;
159     serv_addr.sin_addr.s_addr = INADDR_ANY;
160     serv_addr.sin_port = htons(portno);
161
162     if(bind(sockfd, (struct sockaddr *) &serv_addr, sizeof(serv_addr)) < 0)
163         socketError("Binding Failed");
164
```

소켓 생성, 바인딩

서버

```
165
166     while(1) {
167
168         int found_count = 0; $
169         printf("waiting for client\n");
170         listen(sockfd, 5);
171         clilen = sizeof(cli_addr);
172         newsockfd = accept(sockfd, (struct sockaddr *) &cli_addr, &clilen);
173
174         if(newsockfd < 0) socketError("Error on Accept");
175
176         printf("accept success\n");
177
178         bzero(buffer, BUFFER_SIZE);
179         n = read(newsockfd, buffer, BUFFER_SIZE);
180
181         if(n < 0) socketError("error reading");
182
183
184         int target = atoi(buffer);  // Object index you want to detect;
185         int prev_target = target;
186         bool break_flag = false;
187         printf("looking for %s, %d\n", labels[target], target);
188
189
```

클라이언트로부터 요청이 들어오면 accept 하고
Target 정보를 받습니다.

서버

```
254
255             // check if found target
256             if ((int)label_index - 1 == target) {
257
258                 found_count++;
259
260                 if(found_count < 15) continue;
261
262                 string imgName = "target.jpeg";
263                 printf("found target object!!\n");
264
265                 imwrite(imgName, ori_image);
266                 bzero(buffer, BUFFER_SIZE);
267                 strncpy(buffer, "a", 1);
268                 write(newsockfd, buffer, strlen(buffer));
269
270
```

15 프레임 연속으로 detect 되면 클라이언트로 “a”를 전송해 타겟을 발견했음을 알림 니다.

서버

```
270
271             // send image
272             printf("sending image start\n");
273             ifstream fin;
274             fin.open(imgName, ios::in | ios::binary);
275             const int MAXSIZE = 255;
276
277             fin.seekg(0, ios::end);
278             int size = fin.tellg();
279             fin.seekg(0, ios::beg);
280
281             int left = size;
282             printf("file size: %d\n", size);
283
284             while(left > 0) {
285                 char* buf;
286                 buf = new char[MAXSIZE];
287                 fin.read(buf, MAXSIZE); //파일 읽기
288
289                 left -= MAXSIZE;
290
291                 write(newsockfd, buf, MAXSIZE);
292                 delete[] buf;
293             }
294             fin.close();
295             printf("image sending done\n");
296             close(newsockfd);
```

이미지를 바이트형태로 읽어서 클라이언트로 전송 합니다.

서버

```
304
305         if(break_flag) break;
306         bzero(buffer, BUFFER_SIZE);
307         strncpy(buffer, "b", 1);
308         write(newsockfd, buffer, strlen(buffer));
309
310         bzero(buffer, BUFFER_SIZE);
311         n = read(newsockfd, buffer, BUFFER_SIZE);
312         if(n < 0) {
313             printf("connection reset");
314             break;
315         }
316
317         target = atoi(buffer);
318         if(target != prev_target) {
319             printf("target reset to %s\n", labels[target]);
320             prev_target = target;
321         }
322         bzero(buffer, BUFFER_SIZE);
323
324
325         imshow("result", ori_image);
326
327
```

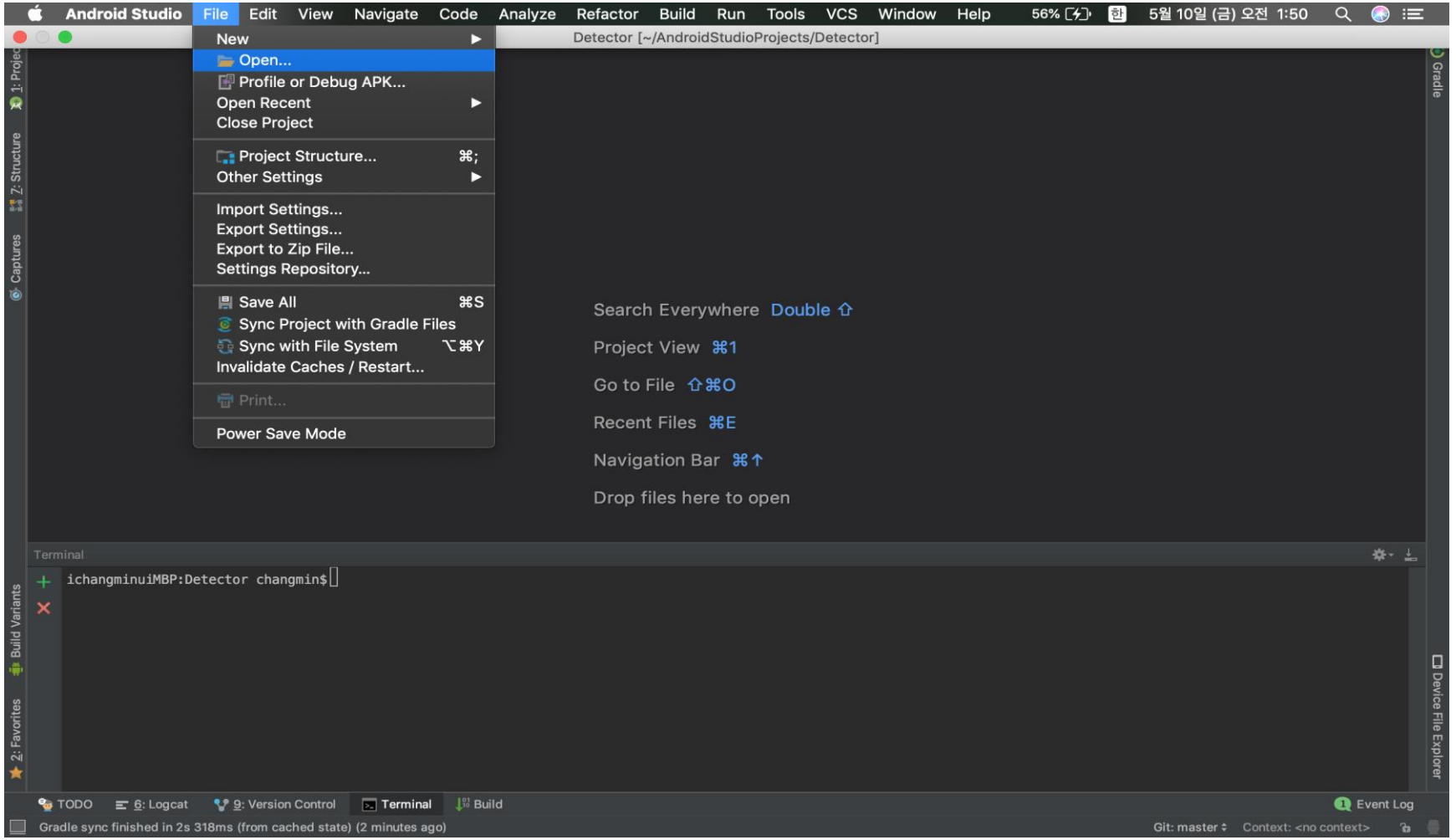
타겟을 발견하지 못하면 클라이언트로 “b” 전송하고 타겟 정보를 새로 받아옴니다.

서버 - 이미지 파일이름 관련 수정사항

```
261
262                 string timestamp = to_string(time(NULL));
263                 string imgName = labels[target] + timestamp + ".jpeg";
264                 printf("found target object!!\n");
265
```

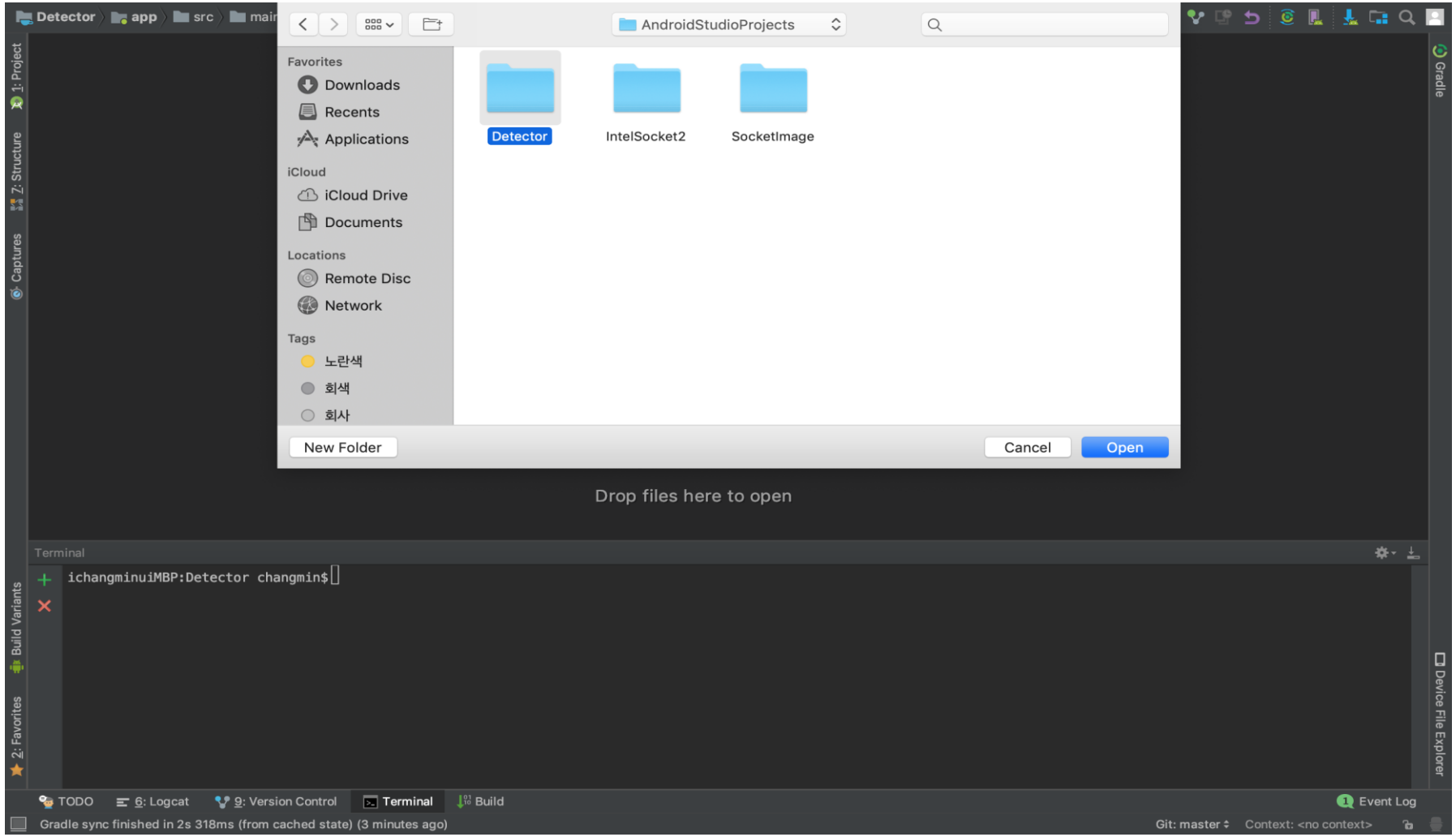
사진파일이 Target object 이름 + Timestamp 의 형태로 저장됩니다.

안드로이드 스튜디오 사용법 – 프로젝트 열기



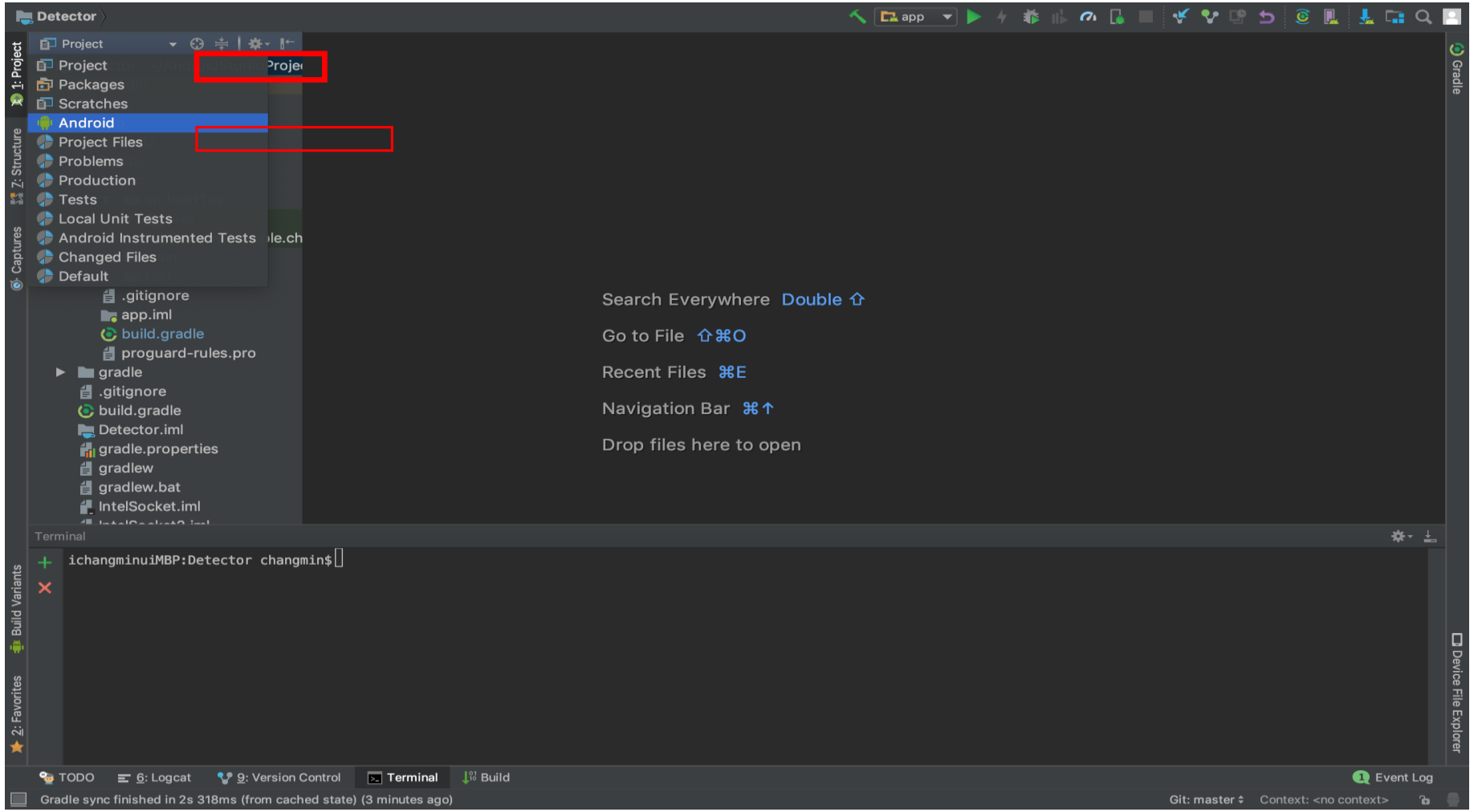
File - Open

안드로이드 스튜디오 사용법 – 프로젝트 열기



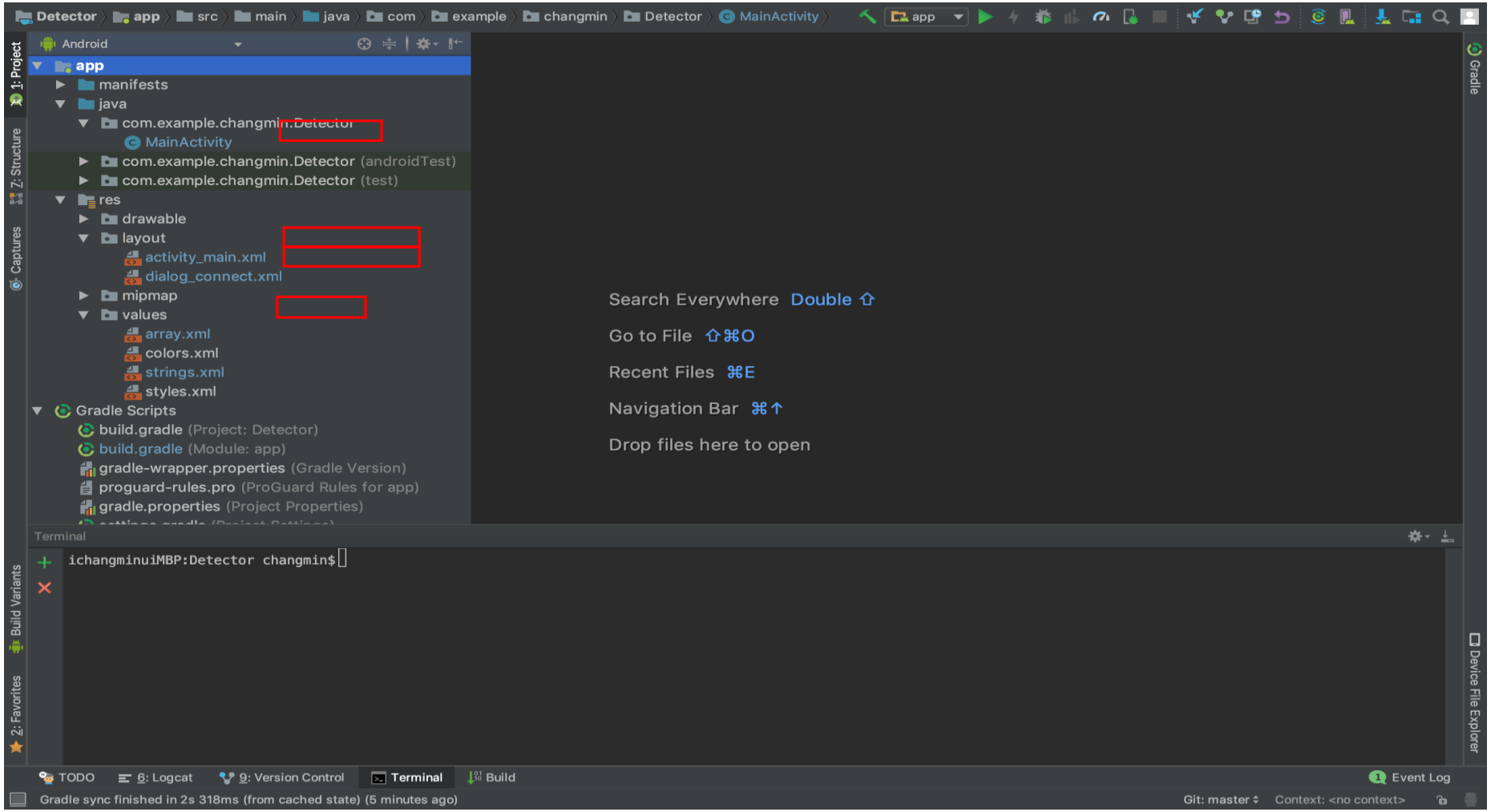
Detector 프로젝트 폴더 클릭 - Open

안드로이드 스튜디오 사용법 – 프로젝트 열기



왼쪽 파일트리 탭에서 표시 방식을 Android로 바꿔준다.

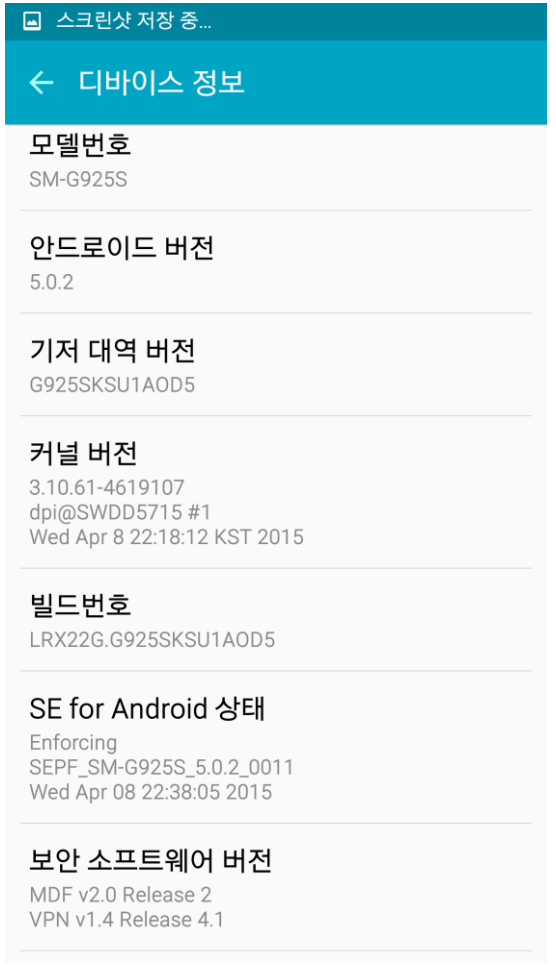
안드로이드 스튜디오 사용법 – 프로젝트 열기



앱을 구성하는 4개의 파일들이 보입니다.

안드로이드 앱 설치하기

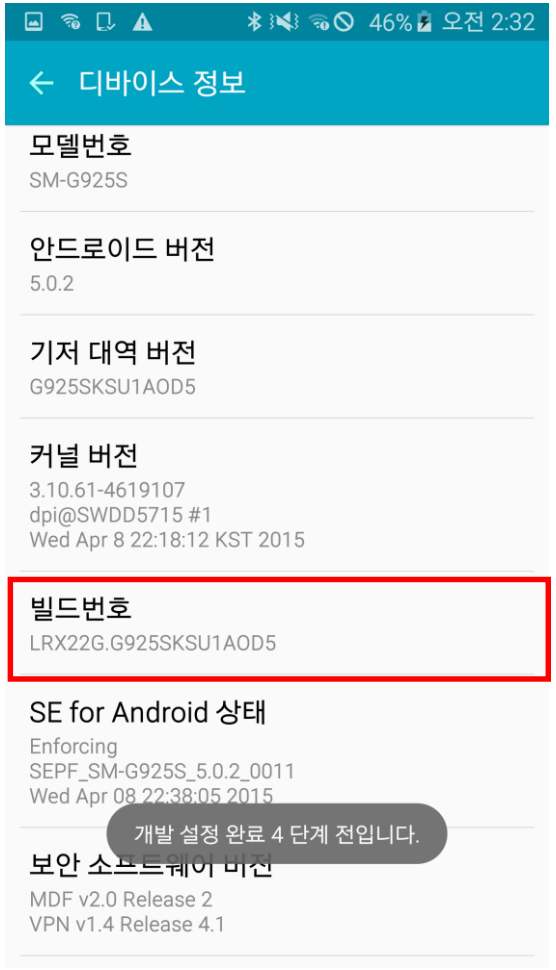
개발자 옵션

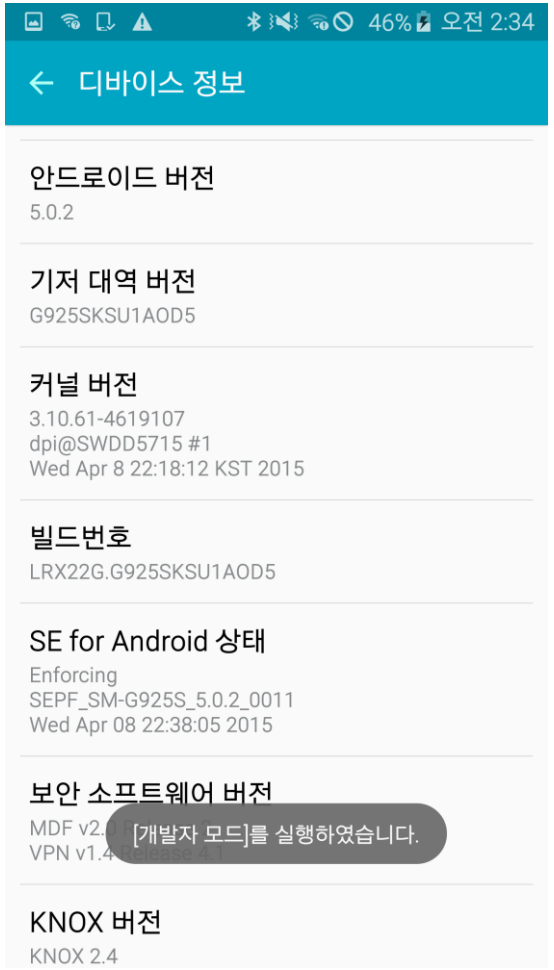


USB로 앱을 빌드하기 위해서는 휴대폰의 개발자 옵션을 활성화해야합니다.

안드로이드 앱 설치하기

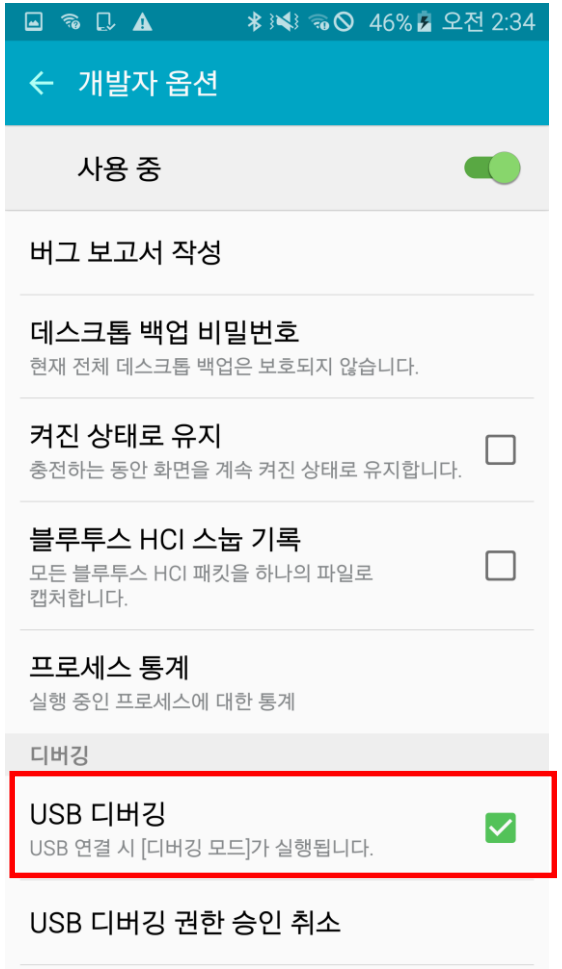
소프트웨어 정보





휴대폰의 설정 – 디바이스 정보에서 [빌드번호]를 여러 번 터치합니다.

안드로이드 앱 설치하기



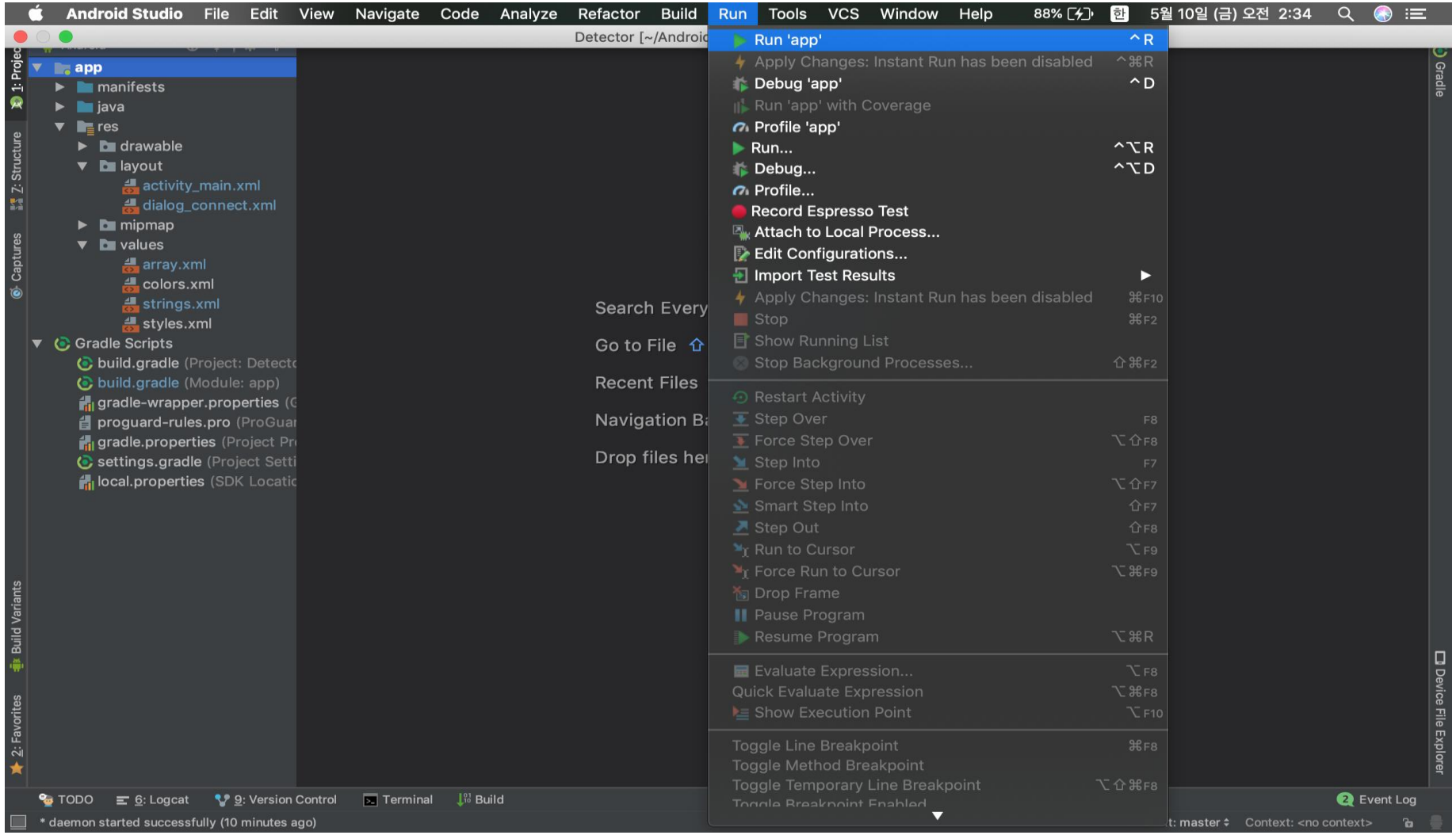
활성화된 개발자 옵션에서 USB 디버깅을 허용해야 합니다.

안드로이드 앱 설치하기

내 파일에들어가서 아이콘
클릭 합니다.

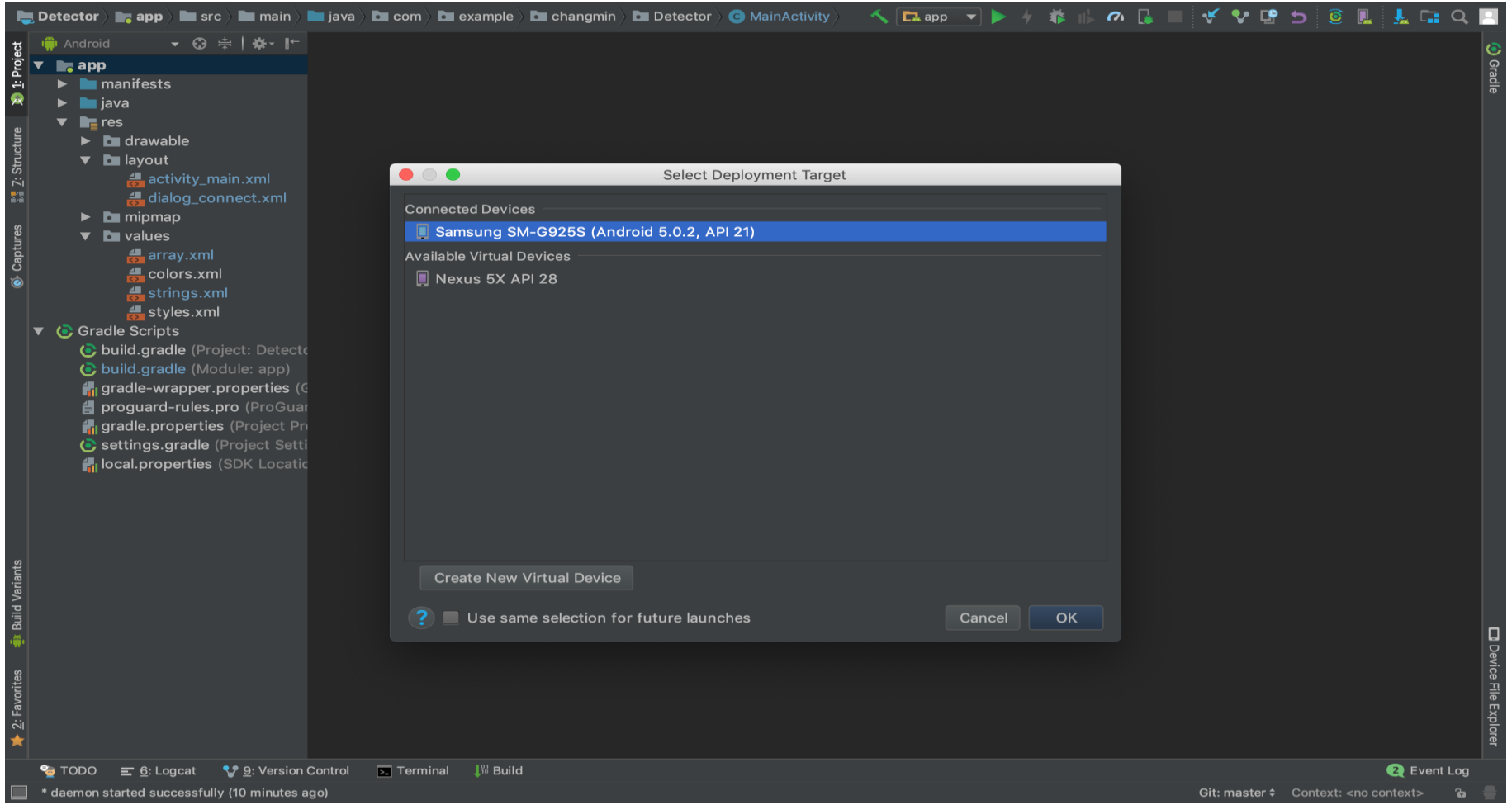
비활성화된 아이콘을
활성화 합니다.

안드로이드 스튜디오 사용법 – 앱 빌드하기



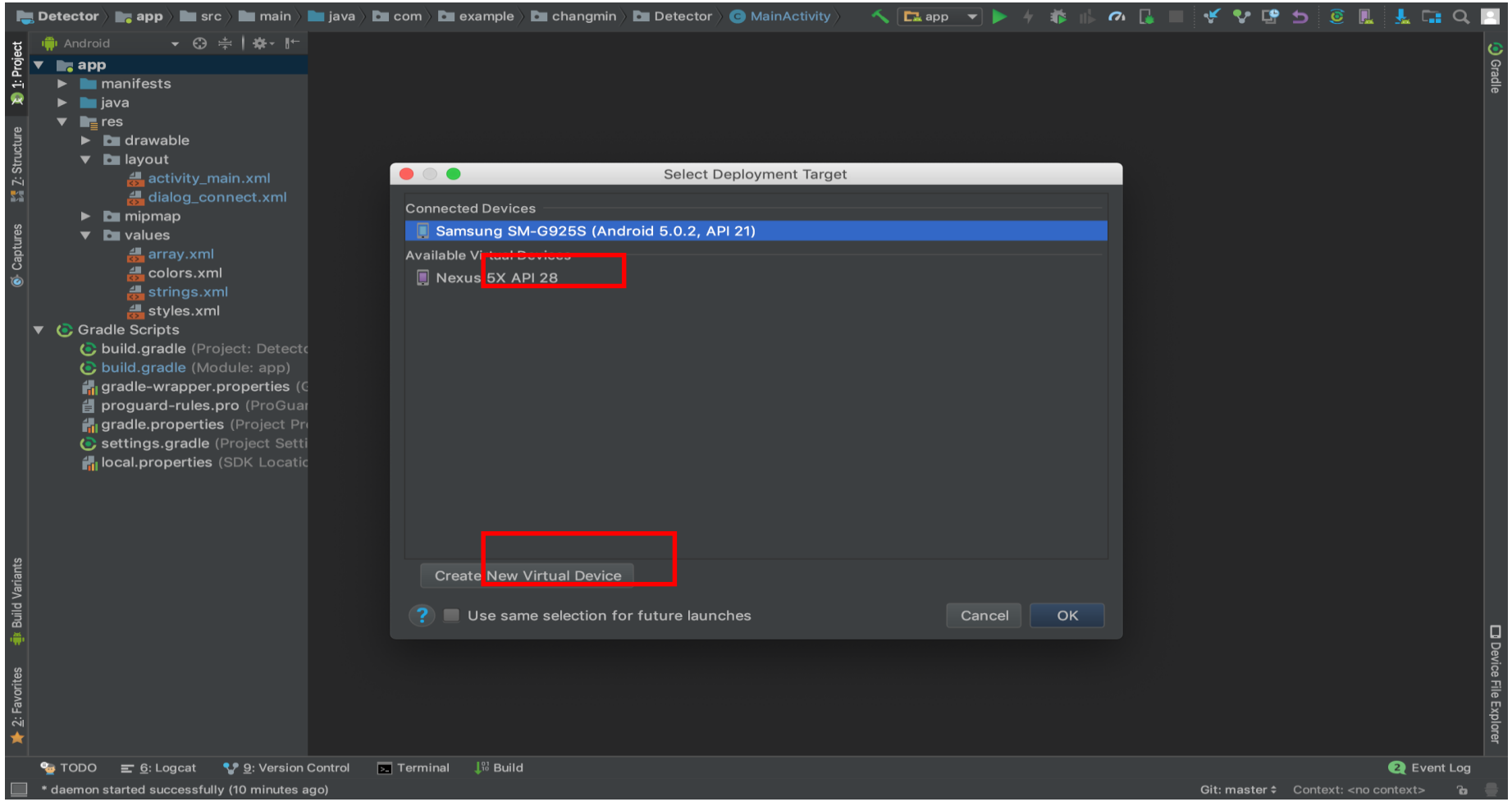
Run – Run ‘app’ 클릭

안드로이드 스튜디오 사용법 – 앱 빌드하기



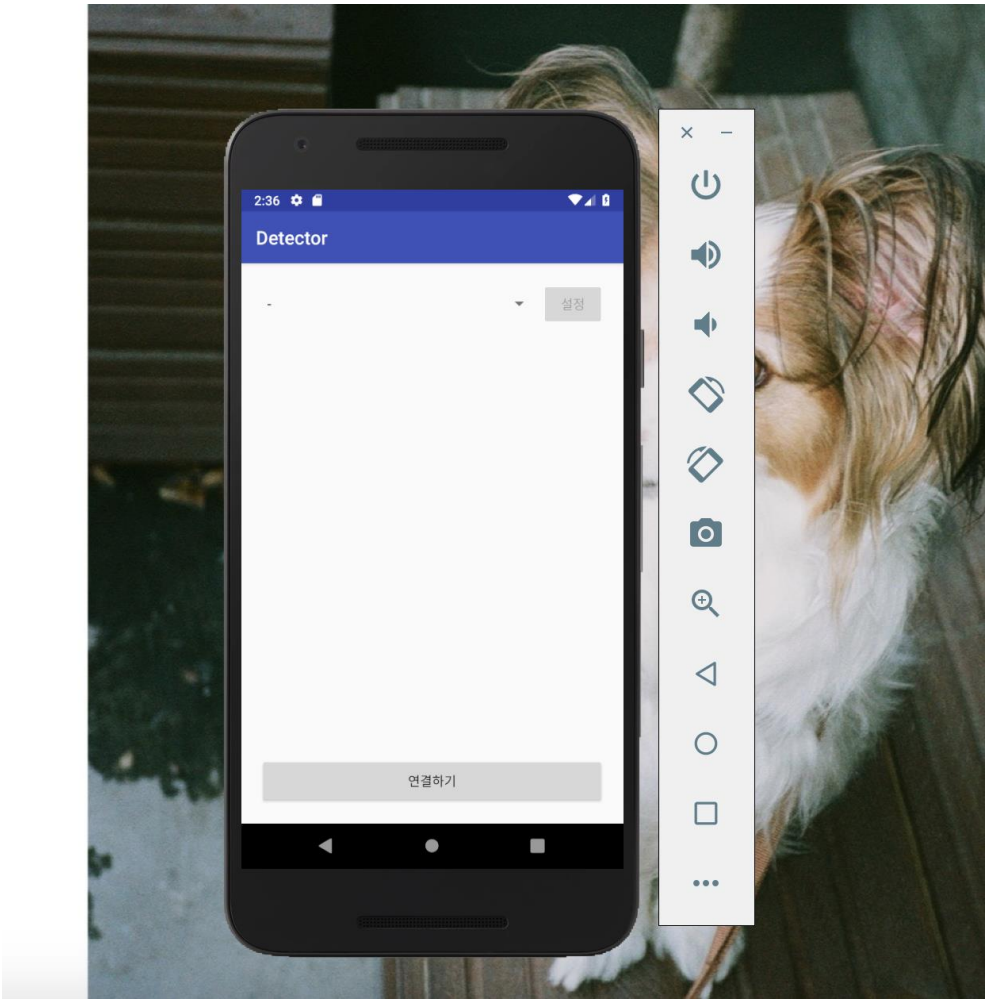
Connected Device에서 기기를 찾고 OK 클릭

안드로이드 스튜디오 사용법 – 앱 빌드하기



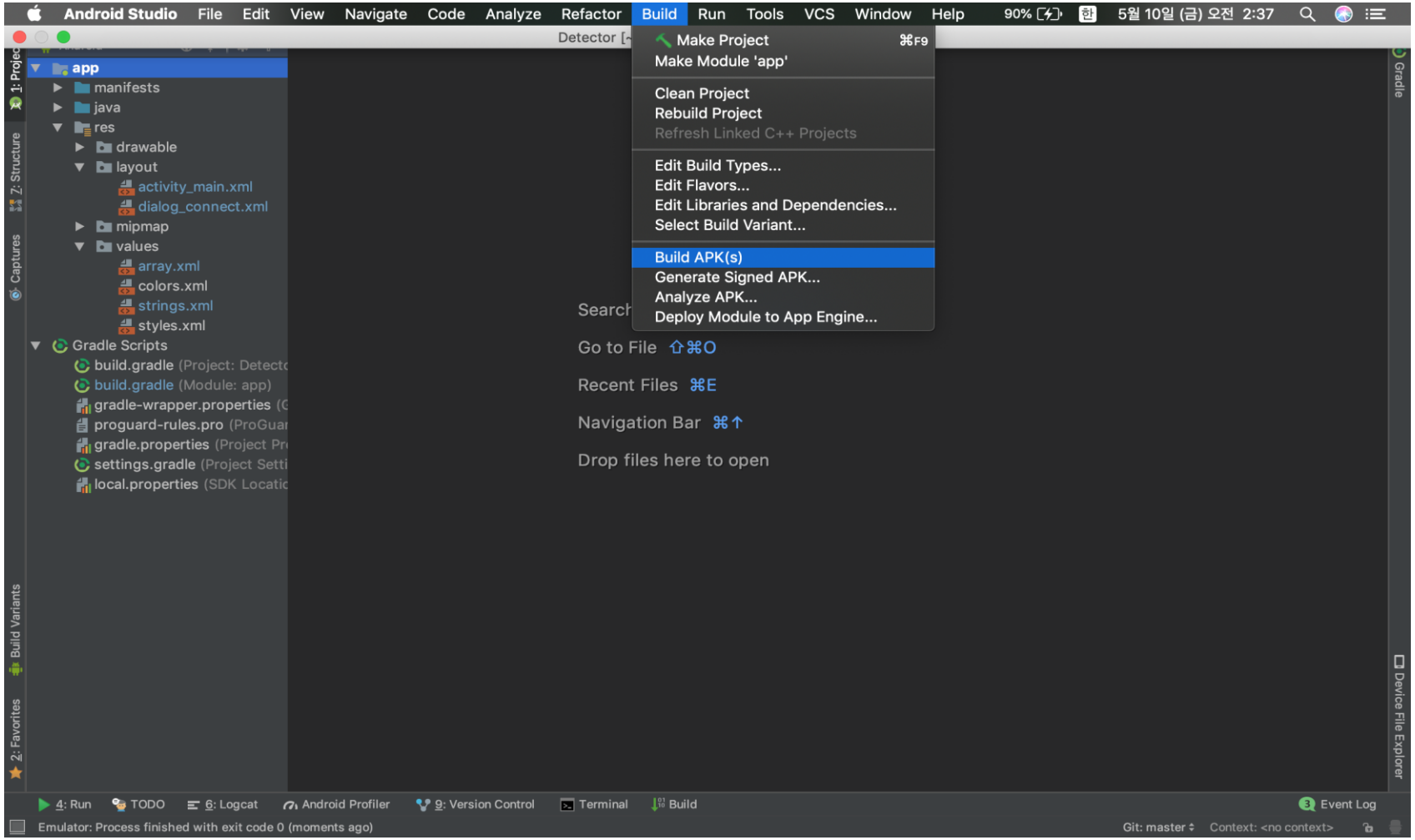
Create New Virtual Device로 가상 기기에서 앱을 실행할 수도 있습니다.

안드로이드 스튜디오 사용법 – 앱 빌드하기



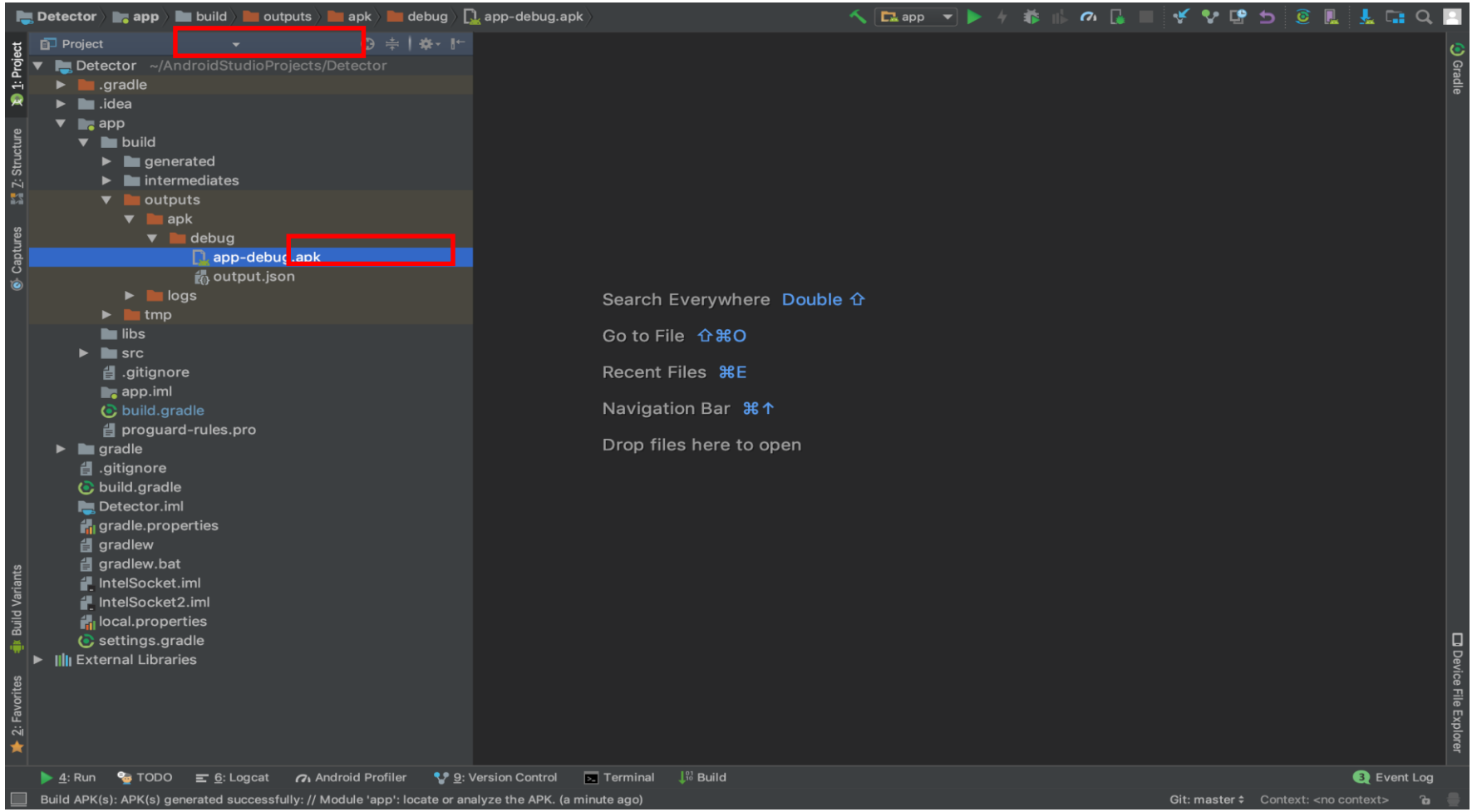
가상 디바이스 실행화면 입니다.

안드로이드 스튜디오 사용법 – 설치파일 추출하기



Build – Build APK(s) 클릭

안드로이드 스튜디오 사용법 – 설치파일 추출하기



파일 정렬 방식을 Project로 바꾸면 앱 설치파일이 생성된 것을 볼 수 있습니다.