# <종합설계 앱 개발 : 카메라 캡쳐 기능 구현 정리>

1. 카메라 캡쳐 및 저장 기능을 수행하는 간단한 앱들 분석

   https://github.com/android/camera

⇨ 구글 안드로이드에서 제공해주는 가장 기본적인 카메라 예제

⇨ Camera2 API를 사용하여 카메라 기능을 수행함.

⇨ 우리가 현재 진행중인 프로젝트는 Camera2 API가 아닌 Camera API를 사용하고 있음

⇨ 두 API가 호환이 되지 않아 다른 예제를 알아봄. (확실하진 않음. 될 수도 있는데 시간이 너무 오래 걸릴 것 같아 굳이 찾아보지 않음)

https://github.com/yyc9920/SurfaceViewCamera

⇨ SurfaceView에 Camera2 API를 이용해 카메라를 구현한 예제.

⇨ 대부분의 카메라 앱들은 TextureView를 사용하는데, 이는 카메라 기능을 구현하기에 가장 적합하고 간단하게 구현할 수 있는 View가 TextureView이기 때문

⇨ 그러나 이 예제 코드 역시 Camera2 API를 사용하고 있어 호환이 되지 않아 다른 예제를 알아봄.

https://youtu.be/bTT8hVoobuE

⇨ SurfaceView에 Camera API를 이용하여 아주 간단하게 카메라 기능을 구현한 예제.

⇨ 프로젝트에 클래스가 MainActivity 하나뿐이고 코드 길이가 100줄이 채 되지 않을 정도로 아주 간단한 카메라 예제이다.

⇨ 아주 간단하지만 카메라 캡쳐 기능을 잘 수행한다.

⇨ 코드를 보면 Camera API에 기본적으로 있는 메소드인 camera.takePicture 메소드를 사용한다. 이 메소드를 사용해서 카메라 캡쳐를 하고 백그라운드에서 생성된 사진 파일을 Bitmap 데이터로 임시 저장하고, storePhotoToStorage 메소드를 만들어 Bitmap 파일을 안드로이드 내장 메모리의 DCIM/photo 폴더로 저장한다. (영상에 나오는 코드 참고)

⇨ 이 외에 surfaceCreated, surfaceChanged 등은 카메라 캡쳐 기능과는 무관한 어플의 중단, 시작, 종료 시의 SurfaceView 동작을 구현하기 위한 메소드인데, 이 부분은 기존 진행하던 프로젝트에 이미 구현이 되어 있다.

2. 예제 코드 분석 내용을 토대로 기존 앱에 카메라 캡쳐 기능을 구현한 내용 정리.

https://github.com/yyc9920/CapstoneDesign_MLKit/blob/master/app/src/main/java/com/google/firebase/samples/apps/mlkit/CameraSource.java

⇨ 기존 앱의 CameraSource 클래스에 카메라 기능이 구현되어 있기 때문에 CameraSource 클래스에 카메라 캡쳐 기능을 적절히 추가하여 구현하였다. (위 코드 참고)

⇨ 기존의 CameraSource 클래스에 다음 코드를 추가했다. (라인 152~188)

```
pictureCallback = new Camera.PictureCallback() {
        @Override
        public void onPictureTaken(byte[] data, Camera camera) {
          Bitmap bmp = BitmapFactory.decodeByteArray(data, 0, data.length);
          Bitmap cbmp = Bitmap.createBitmap(bmp, 0,0, bmp.getWidth(), bmp.getHeight(),
                    null, true);

          String pathFileName = currentDateFormat();
          storePhotoToStorage(cbmp, pathFileName);

          camera.startPreview();
        }
      };

@SuppressLint("WrongThread")
  private void storePhotoToStorage(Bitmap cbmp, String pathFileName) {
    File outputFile = new File(Environment.getExternalStorageDirectory(), "/DCIM/"+"photo"+pathFileName+".jpg");
    currentPhotoPath = "/DCIM/"+"photo"+pathFileName+".jpg";

    try {
      FileOutputStream fileOutputStream   = new FileOutputStream(outputFile);
      cbmp.compress(Bitmap.CompressFormat.JPEG, 100, fileOutputStream);
      fileOutputStream.flush();
      fileOutputStream.close();
    }catch (FileNotFoundException e){
      e.printStackTrace();
    }catch (IOException e){
      e.printStackTrace();
    }
```

```
}

private String currentDateFormat() {
  SimpleDateFormat dateFormat = new SimpleDateFormat("yyyyMMdd_HH_mm_ss");
  String currentTime = dateFormat.format(new Date());
  return currentTime;
}
```

⇨ 위에 올린 유튜브 영상에 나오는 코드와 거의 똑같다. 사진이 저장되었을 때 사진이 저장됐다는 메시지를 띄워주는 Toast 함수만 빼고 구현을 했다.

⇨ Toast함수도 구현이 가능하지만, 굳이 넣으려면 CameraSource클래스의 생성자 함수가 들어간 코드들을 다 수정해줘야 해서 그냥 구현하지 않았다.

https://github.com/yyc9920/CapstoneDesign_MLKit/blob/master/app/src/main/java/com/google/firebase/samples/apps/mlkit/LivePreviewActivity.java

⇨ 우리가 진행중인 프로젝트의 MainActicity 격인 LivePreviewActivity 클래스.

⇨ MainActivity : 안드로이드 프로젝트에서 앱을 실행했을 때 사용자가 직접 접하는 카메라 버튼, 메뉴 버튼 등이 수행하는 작업, 앱 시작 시 띄워지는 화면 등이 구현된 클래스를 일반적으로 MainActivity라고 한다.

⇨ 이 메인액티비티에 서브 클래스들의 메소드를 불러와서 세부적인 동작을 구현한다. (라인 117~122)

```
captureBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
      cameraSource.camera.takePicture(null, null, cameraSource.pictureCallback);
    }
  });
```

⇨ captureBtn이라는 버튼 객체의 setOnClickListener를 설정해준다.

⇨ 버튼을 클릭했을 때 cameraSource(CameraSource 클래스의 객체)의 camera.takePicture 메소드를 수행한다.

⇨ 프로젝트의 서브 클래스인 CameraSource에 카메라 기능이 구현되어 있기 때문에 CameraSource 클래스의 객체를 생성해 해당 객체에서 메소드를 가져와 카메라 캡쳐 기능을 구현한다.

⇨ 찍은 사진을 갤러리에 저장하려면 CameraSource에 있는 비트맵 파일을 메인액티비티로 가져와야 하는데, 자바에서 클래스 내부 변수는 공유가 안되기 때문에 다른 방법을 찾아봐야 하는데, 해당 부분은 꼭 필요한 것은 아니기 때문에 추후에 구현 가능하면 구현을 하면 될 것이라고 생각된다.

```
  private class SurfaceCallback implements SurfaceHolder.Callback {
    @Override
    public void surfaceCreated(SurfaceHolder surface) {
      surfaceAvailable = true;
      try {
        startIfReady();
      } catch (IOException e) {
        Log.e(TAG, "Could not start camera source.", e);
      }

//    Camera.Parameters parameters;
//    parameters = camera.getParameters();
//    parameters.setPreviewFrameRate(20);
//    parameters.setPreviewSize(352, 288);
//    camera.setParameters(parameters);
    }

    @Override
    public void surfaceDestroyed(SurfaceHolder surface) {
      surfaceAvailable = false;
    }

    @Override
    public void surfaceChanged(SurfaceHolder holder, int format, int width, int height) {}
  }
```

⇨ **surfaceCreated, surfaceDestroyed, surfaceChanged가 구현된 내용.**

https://github.com/yyc9920/CapstoneDesign_MLKit/blob/master/app/src/main/java/com/google/firebase/samples/apps/mlkit/CameraSourcePreview.java

⇨ **SurfaceView가 구현된 클래스는 CameraSourcePreview클래스이다. (라인 129~153)**

⇨ **이곳에서 화면에 보이는 View를 구현하고 관리한다.**

⇨ **앱을 실행하고 얼굴 인식을 해보면 네모난 박스와 눈을 감은 정도, 행복도, 좌표값 등의 정보가 함께 나타나는데, 이러한 데이터들의 정보 자체는 GraphicOverlay라는 클래스에 구현이 되어있고 해당 클래스에서 관리를 한다.**

⇨ **CameraSourcePreview는 해당 정보와 데이터들을 SurfcaeView에서 사용자가 볼 수 있게 구현하는 역할을 한다. 쉽게 얘기하면, GraphicOverlay가 백엔드이고, CameraSourcePreview가 프론트 엔드이다.**

⇨ **백엔드와 프론트엔드에 대한 설명(학원광고이긴 한데 설명은 잘되있다) :**

https://post.naver.com/viewer/postView.nhn?volumeNo=16600589&memberNo=36090254