Mobile/Web Service Programming

Chapter 20
안드로이드 포토 블로그 Part01

Kiok Ahn

Image Processing Lab

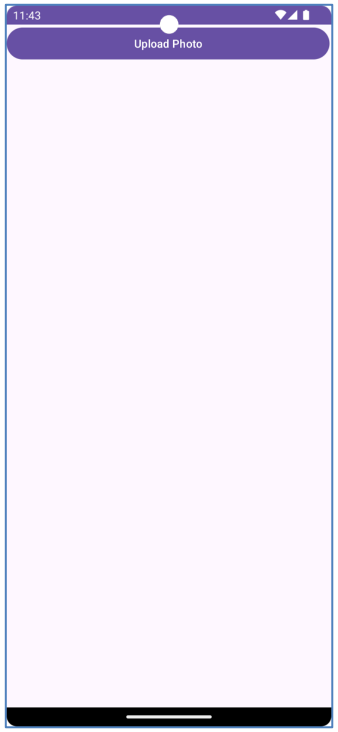
경희대학교
KYUNG HEE UNIVERSITY

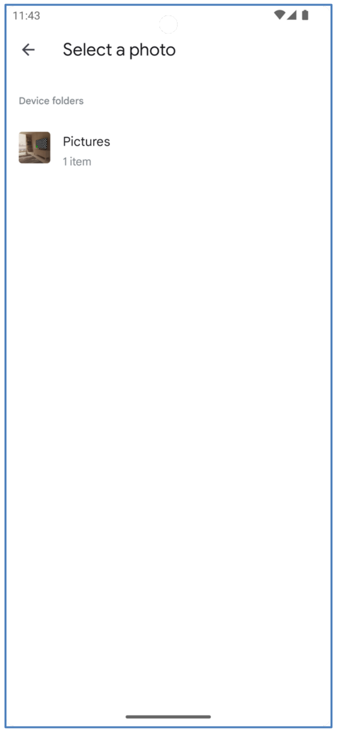
목차

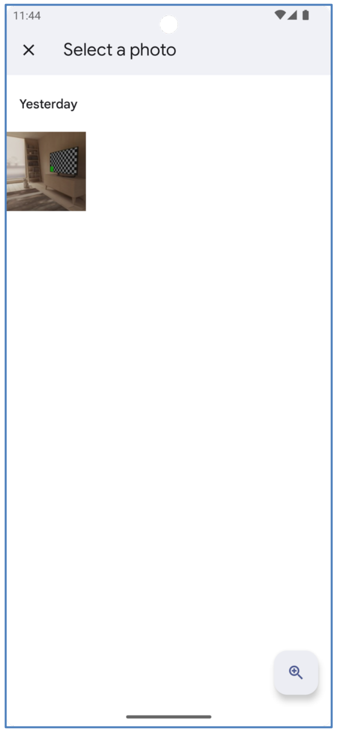
개요

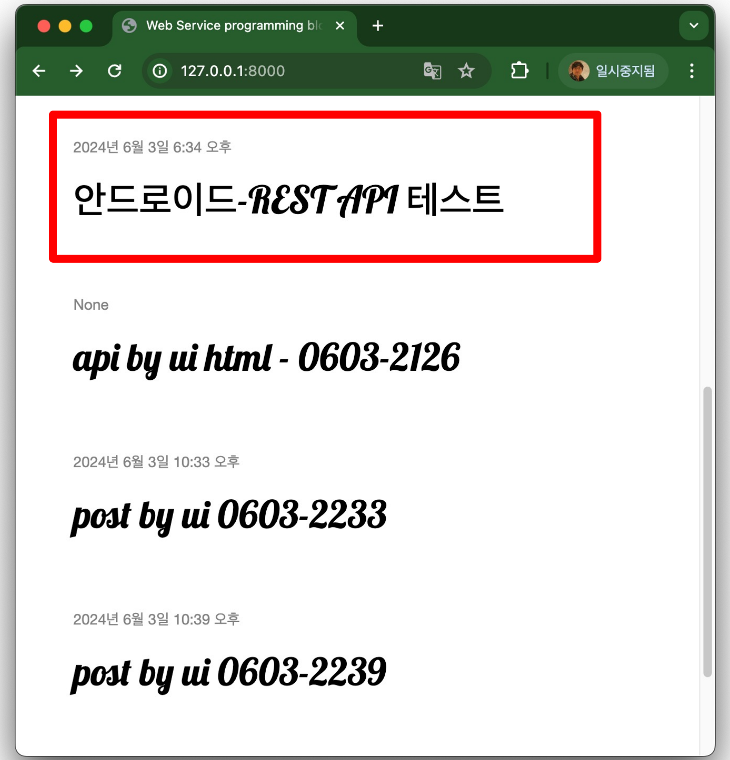
❖ 안드로이드 포토 블로그

- 안드로이드 모바일 폰의 사진에 접근
- 사진을 Web Blog(DJANGO 서버) REST API를 이용하여 업로드









2024년 6월 3일 6:34 오후

안드로이드-REST API 테스트

None

api by ui html - 0603-2126

2024년 6월 3일 10:33 오후

post by ui 0603-2233

2024년 6월 3일 10:39 오후

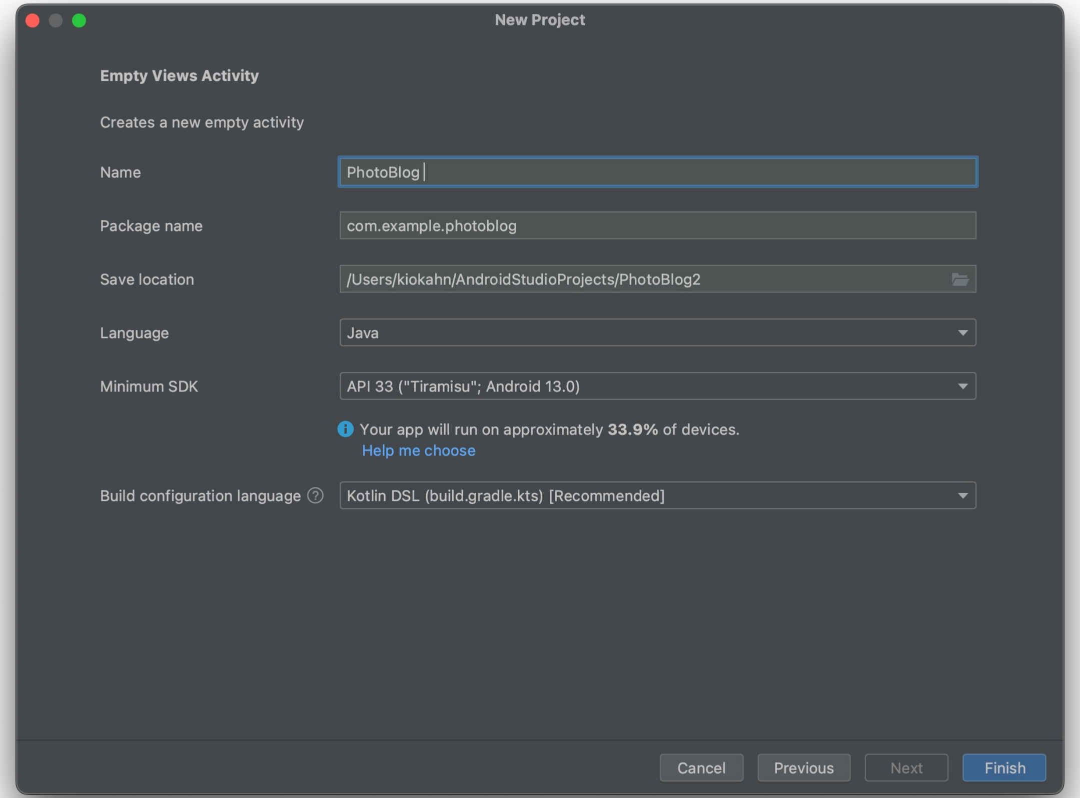
post by ui 0603-2239

프로젝트 생성

❖ PhotoBlog 프로젝트 생성

❖ "API 33"

- 다른 수준의 API의 경우 보안 이슈가 다를 수 있음



Empty Views Activity

Creates a new empty activity

Name	PhotoBlog
Package name	com.example.photoblog
Save location	/Users/kiokahn/AndroidStudioProjects/PhotoBlog2
Language	Java
Minimum SDK	API 33 ("Tiramisu"; Android 13.0)
	Your app will run on approximately **33.9%** of devices. Help me choose
Build configuration language	Kotlin DSL (build.gradle.kts) [Recommended]

Manifest 수정

경희대학교 KYUNG HEE UNIVERSITY

❖ 실행을 위한 권한 적용

Manifest/ AndroidManifest.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.READ_MEDIA_IMAGES" />

    <application
        android:allowBackup="true"
        …
        tools:targetApi="31"
        android:usesCleartextTraffic="true">
        <activity
```

인터넷 접근 권한

사진/이미지 접근 권한

API 33이전 버전은 “READ_EXTERNAL_STORAGE”

HTTP 접근 권한

HTTP 접근 허용, API 28이상 버전은 HTTPS만 허용 함

UI 레이아웃

❖ 사용자 인터페이스

layout/activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context=".MainActivity">

   <LinearLayout
      android:layout_width="409dp"
      android:layout_height="729dp"
      android:orientation="vertical"
      tools:layout_editor_absoluteX="1dp"
      tools:layout_editor_absoluteY="1dp">

      <Button
         android:id="@+id/uploadButton"
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:text="Upload Photo" />
   </LinearLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

이미지 업로드를 위한 버튼만 사용 함

Activity

❖ import

com/example/photoblog/MainActivity.java

```java
package com.example.photoblog;

import androidx.activity.result.ActivityResultLauncher;
import androidx.activity.result.contract.ActivityResultContracts;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import android.Manifest;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.database.Cursor;
import android.net.Uri;
import android.os.Build;
import android.os.Bundle;
import android.os.Handler;
import android.os.Looper;
import android.provider.MediaStore;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;
```

```java
import org.json.JSONException;
import org.json.JSONObject;

import java.io.IOException;
import java.io.OutputStreamWriter;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
```

Import 자동 추가는 해당 코드에서 “Alt+Enter” or “option+Enter”

Activity

❖ MainActivity class 구성

com/example/photoblog/MainActivity.java

```java
public class MainActivity extends AppCompatActivity {
   private static final int READ_MEDIA_IMAGES_PERMISSION_CODE = 1001;  // 상수 정의
        private static final int READ_EXTERNAL_STORAGE_PERMISSION_CODE = 1002;
   // 상수 정의

        //private static final String UPLOAD_URL = "http://127.0.0.1:8000/api_root/Post/";
   private static final String UPLOAD_URL = "http://10.0.2.2:8000/api_root/Post/";
   Uri imageUri = null;

   private final ExecutorService executorService = Executors.newSingleThreadExecutor();
   private final Handler handler = new Handler(Looper.getMainLooper());

   private final ActivityResultLauncher<Intent> imagePickerLauncher = registerForActivityResult(
         //...코드 계속
        );

   @Override
   protected void onCreate(Bundle savedInstanceState) { /*...코드 계속 */ }
   @Override
   public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults) {
         //...코드 계속
        }

   private void openImagePicker() {/*...코드 계속*/ }
   private String getRealPathFromURI(Uri contentUri) {/*...코드 계속*/ }

   private String uploadImage(String imageUrl) throws IOException, JSONException {/*...코드 계속*/ }
}
```

Activity

❖ MainActivity class 구성

com/example/photoblog/MainActivity.java

```
public class MainActivity extends AppCompatActivity {
   private static final int READ_MEDIA_IMAGES_PERMISSION_CODE = 1001;  // 상수 정의
        private static final int READ_EXTERNAL_STORAGE_PERMISSION_CODE = 1002;
   // 상수 정의

  //private static final String UPLOAD_URL = "http://127.0.0.1:8000/api_root/Post/";
   private static final String UPLOAD_URL = "http://10.0.2.2:8000/api_root/Post/";
   Uri imageUri = null;

   private final ExecutorService executorService = Executors.newSingleThreadExecutor();
   private final Handler handler = new Handler(Looper.getMainLooper());

   private final ActivityResultLauncher<Intent> imagePickerLauncher = registerForActivityResult(
         //...코드 계속
        );
   @Override
   protected void onCreate(Bundle savedInstanceState) { /*...코드 계속 */ }
   @Override
   public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults) {
         //...코드 계속
        }
   private void openImagePicker() {/*...코드 계속*/ }
   private String getRealPathFromURI(Uri contentUri) {/*...코드 계속*/ }
   private String uploadImage(String imageUrl) throws IOException, JSONException {/*...코드 계속*/ }
}
```

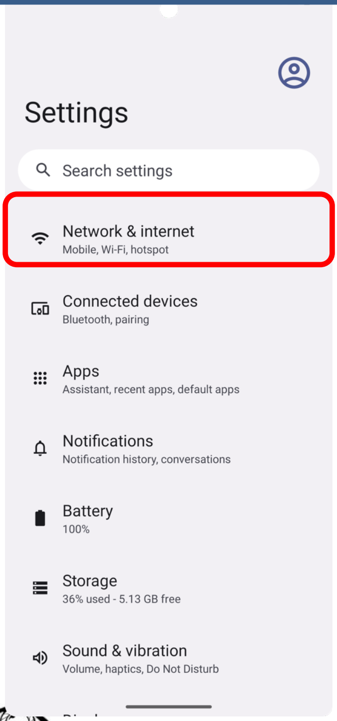
Activity

❖ AVD와 호스트 간 통신

- AVD에서 WiFi 연결 확인 (호스트와 인터넷 연결 공유)
- AVD에서 “10.0.2.2”의 IP를 사용하면 호스트의 “127.0.0.1” 로 접근함
- 단, 호스트에서 인식하는 IP는 “10.0.2.2”이므로 서버의 접근 허용 필요

./mysite/settings.py

```
#ALLOWED_HOSTS = []
ALLOWED_HOSTS = ['127.0.0.1', '10.0.2.2','.pythonanywhere.com']
```



Settings

Search settings

Network & internet
Mobile, Wi-Fi, hotspot

Connected devices
Bluetooth, pairing

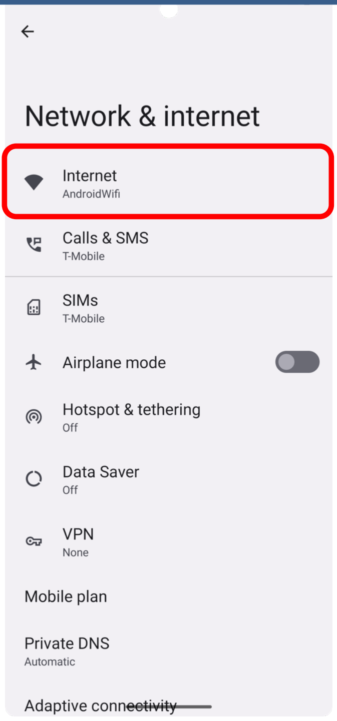
Apps
Assistant, recent apps, default apps

Notifications
Notification history, conversations

Battery
100%

Storage
36% used - 5.13 GB free

Sound & vibration
Volume, haptics, Do Not Disturb



Network & internet

Internet
AndroidWifi

Calls & SMS
T-Mobile

SIMs
T-Mobile

Airplane mode

Hotspot & tethering
Off

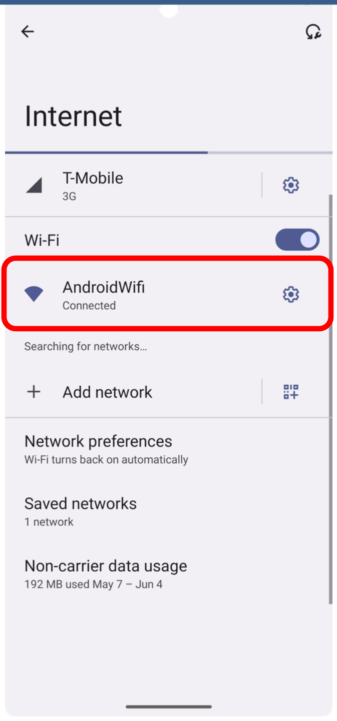
Data Saver
Off

VPN
None

Mobile plan

Private DNS
Automatic

Adaptive connectivity



Internet

T-Mobile
3G

Wi-Fi

AndroidWifi
Connected

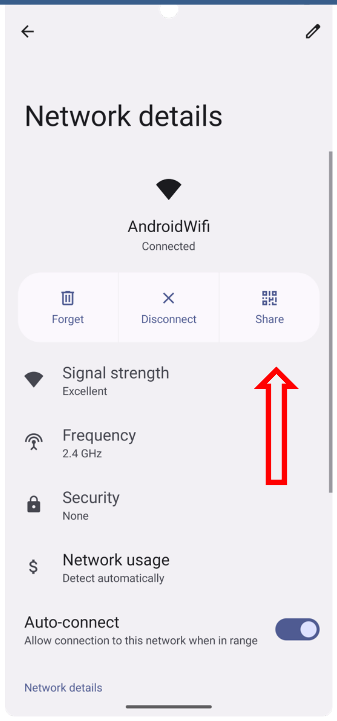
Searching for networks...

Add network

Network preferences
Wi-Fi turns back on automatically

Saved networks
1 network

Non-carrier data usage
192 MB used May 7 – Jun 4



Network details

AndroidWifi
Connected

Forget | Disconnect | Share

Signal strength
Excellent

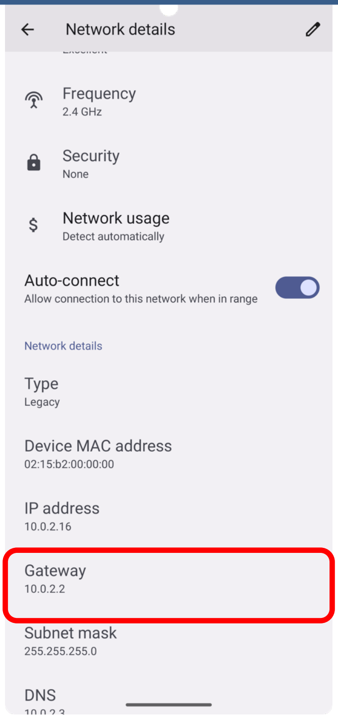
Frequency
2.4 GHz

Security
None

Network usage
Detect automatically

Auto-connect
Allow connection to this network when in range

Network details



Network details

Frequency
2.4 GHz

Security
None

Network usage
Detect automatically

Auto-connect
Allow connection to this network when in range

Network details

Type
Legacy

Device MAC address
02:15:b2:00:00:00

IP address
10.0.2.16

Gateway
10.0.2.2

Subnet mask
255.255.255.0

DNS

Activity

❖ private final ActivityResultLauncher<Intent> imagePickerLauncher

com/example/photoblog/MainActivity.java

```java
private final ActivityResultLauncher<Intent> imagePickerLauncher = registerForActivityResult( //...코드  계속
          new ActivityResultContracts.StartActivityForResult(),
    result -> {
      if (result.getResultCode() == RESULT_OK && result.getData() != null) {
        imageUri = result.getData().getData();
        String filePath = getRealPathFromURI(imageUri);
        executorService.execute(() -> {
          String uploadResult;
          try {
            uploadResult = uploadImage(filePath);
          } catch (IOException e) {
            uploadResult = "Upload failed: " + e.getMessage();
          } catch (JSONException e) {
            throw new RuntimeException(e);
          }
          String finalUploadResult = uploadResult;
          handler.post(() -> Toast.makeText(MainActivity.this, finalUploadResult, Toast.LENGTH_LONG).show());
        });
      }
    }
);
```

Activity

경희대학교 KYUNG HEE UNIVERSITY

❖ private final **ActivityResultLauncher**<Intent> **imagePickerLauncher**

com/example/photoblog/MainActivity.java

```
private final ActivityResultLauncher<Intent> imagePickerLauncher = registerForActivityResult( //...코드 계속
            new ActivityResultContracts.StartActivityForResult(),
        result -> {
            if (result.getResultCode() == RESULT_OK && result.getData() != null) {
                imageUri = result.getData().getData();
                String filePath = getRealPathFromURI(imageUri);
                executorService.execute(() -> {
                    String uploadResult;
                    try {
                        uploadResult = uploadImage(filePath);
                    } catch (IOException e) {
                        uploadResult = "Upload failed: " + e.getMessage();
                    } catch (JSONException e) {
                        throw new RuntimeException(e);
                    }
                    String finalUploadResult = uploadResult;
                    handler.post(() -> Toast.makeText(MainActivity.this, finalUploadResult, Toast.LENGTH_LONG).show());
                });
            }
        }
);
```

이미지 업로드 수행

Activity

❖ protected void **onCreate**(Bundle savedInstanceState)

com/example/photoblog/MainActivity.java

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    Button uploadButton = findViewById(R.id.uploadButton);
    uploadButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.TIRAMISU) {
                if (ContextCompat.checkSelfPermission(MainActivity.this,
                        Manifest.permission.READ_MEDIA_IMAGES) != PackageManager.PERMISSION_GRANTED) {
                    ActivityCompat.requestPermissions(MainActivity.this,
                            new String[]{Manifest.permission.READ_MEDIA_IMAGES},
                            READ_MEDIA_IMAGES_PERMISSION_CODE);
                } else {
                    openImagePicker();
                }
            } else {
                if (ContextCompat.checkSelfPermission(MainActivity.this,
                        Manifest.permission.READ_EXTERNAL_STORAGE) != PackageManager.PERMISSION_GRANTED) {
                    ActivityCompat.requestPermissions(MainActivity.this,
                            new String[]{Manifest.permission.READ_EXTERNAL_STORAGE},
                            READ_EXTERNAL_STORAGE_PERMISSION_CODE);
                } else {
                    openImagePicker();
                }
            }
        }
    });
}
```

버튼 클릭 이벤트 핸들러

이미지 경로 취득

Activity

❖ public void **onRequestPermissionsResult**()
❖ private void **openImagePicker**()

com/example/photoblog/MainActivity.java

```
@Override
public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults) {
    super.onRequestPermissionsResult(requestCode, permissions, grantResults);
    if (requestCode == READ_MEDIA_IMAGES_PERMISSION_CODE) {
        if (grantResults.length > 0 && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
            openImagePicker();
        } else {
            Toast.makeText(this, "Permission denied", Toast.LENGTH_SHORT).show();
        }
    }
}

private void openImagePicker() {
    Intent intent = new Intent(Intent.ACTION_PICK, MediaStore.Images.Media.EXTERNAL_CONTENT_URI);
    imagePickerLauncher.launch(intent);
}
```

Activity

❖ private String **getRealPathFromURI**(Uri contentUri)

com/example/photoblog/MainActivity.java

```java
private String getRealPathFromURI(Uri contentUri) {
    String[] projection = {MediaStore.Images.Media.DATA};
    Cursor cursor = getContentResolver().query(contentUri, projection, null, null, null);
    if (cursor == null) {
        return contentUri.getPath();
    } else {
        cursor.moveToFirst();
        int columnIndex = cursor.getColumnIndexOrThrow(MediaStore.Images.Media.DATA);
        String path = cursor.getString(columnIndex);
        cursor.close();
        return path;
    }
}
```

Activity

❖ private String **uploadImage**()

com/example/photoblog/MainActivity.java

```
private String uploadImage(String imageUrl) throws IOException, JSONException {
    OutputStreamWriter outputStreamWriter = null;
    try {
        try {
            URL url = new URL(UPLOAD_URL);
            HttpURLConnection connection = (HttpURLConnection) url.openConnection();
            connection.setRequestMethod("POST");
            connection.setRequestProperty("Authorization", "JWT b181ce4155b7413ebd1d86f1379151a7e035f8bd");
            connection.setRequestProperty("Content-Type", "application/json");

            JSONObject jsonObject = new JSONObject();
            jsonObject.put("author", 1);
            jsonObject.put("title", "안드로이드-REST API 테스트");
            jsonObject.put("text", "안드로이드로 작성된 REST API 테스트 입력 입니다.");
            jsonObject.put("created_date", "2024-06-03T18:34:00+09:00");
            jsonObject.put("published_date", "2024-06-03T18:34:00+09:00");
            //jsonObject.put("image", imageUrl);

            outputStreamWriter = new OutputStreamWriter(connection.getOutputStream());
            outputStreamWriter.write(jsonObject.toString());
            outputStreamWriter.flush();

            connection.connect();
```

Activity

❖ private String **uploadImage**() - 계속

com/example/photoblog/MainActivity.java

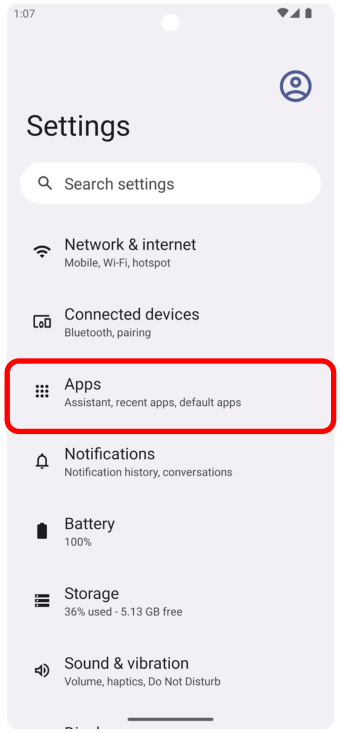
```java
if (connection.getResponseCode() == 200) {
                Log.e("uploadImage", "Success");
            }
            connection.disconnect();
        } catch (MalformedURLException e) {
            throw new RuntimeException(e);
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    } catch (Exception e) {
        Log.e("uploadImage", "Exception in uploadImage: " + e.getMessage());
    }

    Log.e("LogInTask", "Failed to login");
    throw new Error("failed to login");
  }
}
```

점검사항 – 1/2

❖ 안드로이드 클라이언트

- 접근 권한 확인 (API 버전에 따라 코드가 달라 질 수 있음)



1:07

Settings

Search settings

Network & internet
Mobile, Wi-Fi, hotspot

Connected devices
Bluetooth, pairing

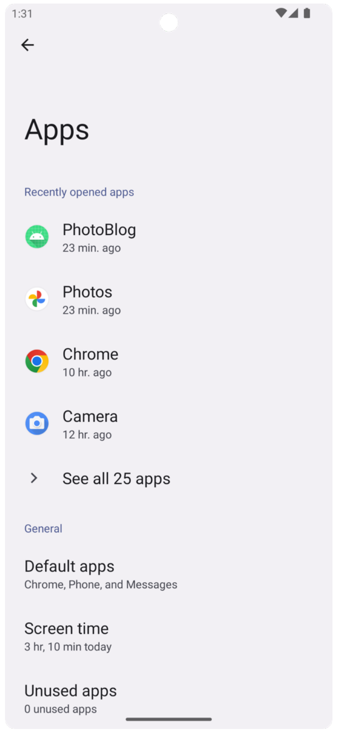
Apps
Assistant, recent apps, default apps

Notifications
Notification history, conversations

Battery
100%

Storage
36% used - 5.13 GB free

Sound & vibration
Volume, haptics, Do Not Disturb



1:31

Apps

Recently opened apps

PhotoBlog
23 min. ago

Photos
23 min. ago

Chrome
10 hr. ago

Camera
12 hr. ago

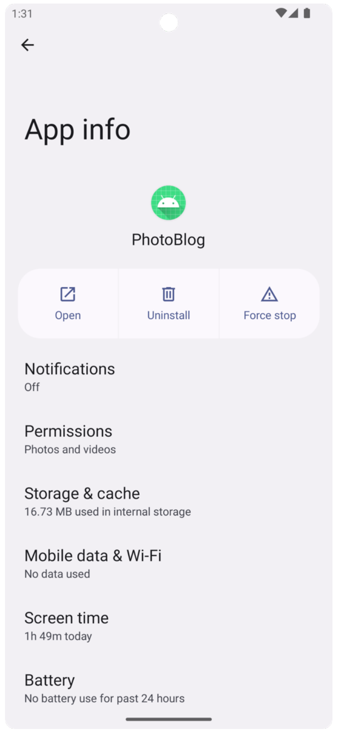
See all 25 apps

General

Default apps
Chrome, Phone, and Messages

Screen time
3 hr, 10 min today

Unused apps
0 unused apps



1:31

App info

PhotoBlog

Open | Uninstall | Force stop

Notifications
Off

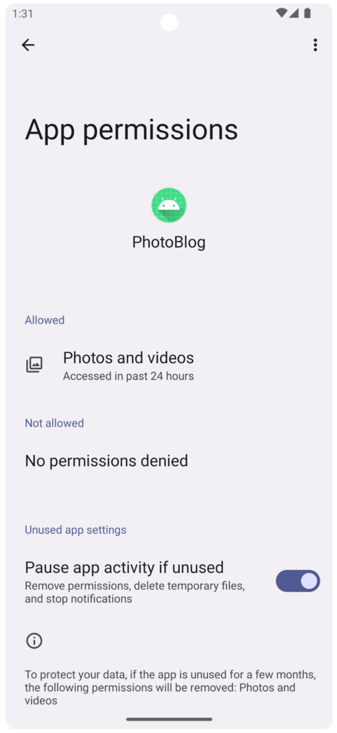
Permissions
Photos and videos

Storage & cache
16.73 MB used in internal storage

Mobile data & Wi-Fi
No data used

Screen time
1h 49m today

Battery
No battery use for past 24 hours



1:31

App permissions

PhotoBlog

Allowed

Photos and videos
Accessed in past 24 hours

Not allowed

No permissions denied

Unused app settings

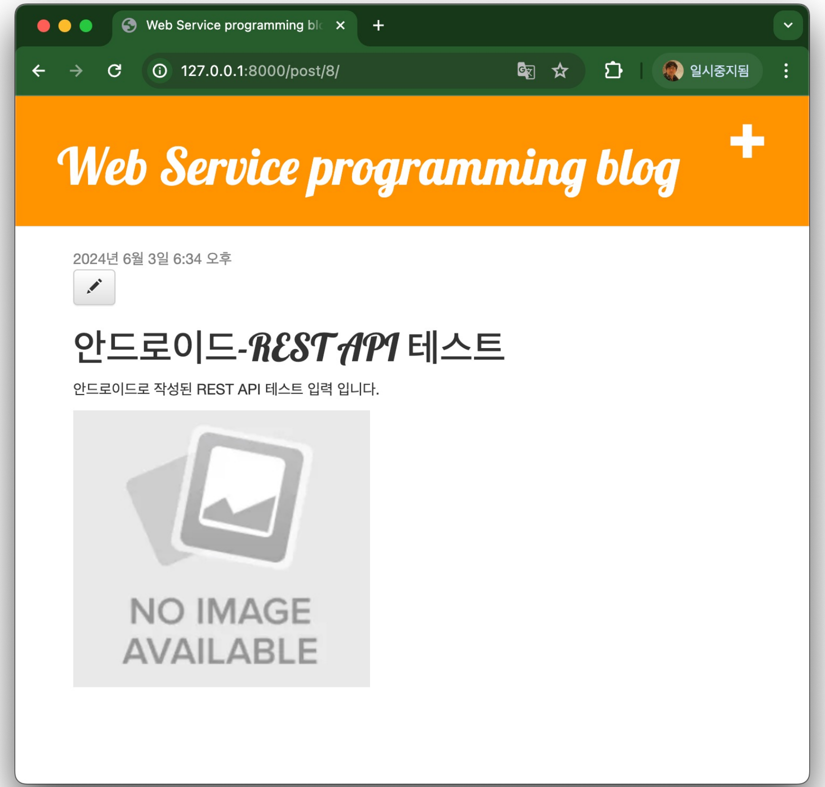
Pause app activity if unused
Remove permissions, delete temporary files, and stop notifications

To protect your data, if the app is unused for a few months, the following permissions will be removed: Photos and videos

점검사항 – 2/2

❖ DJANGO 서버

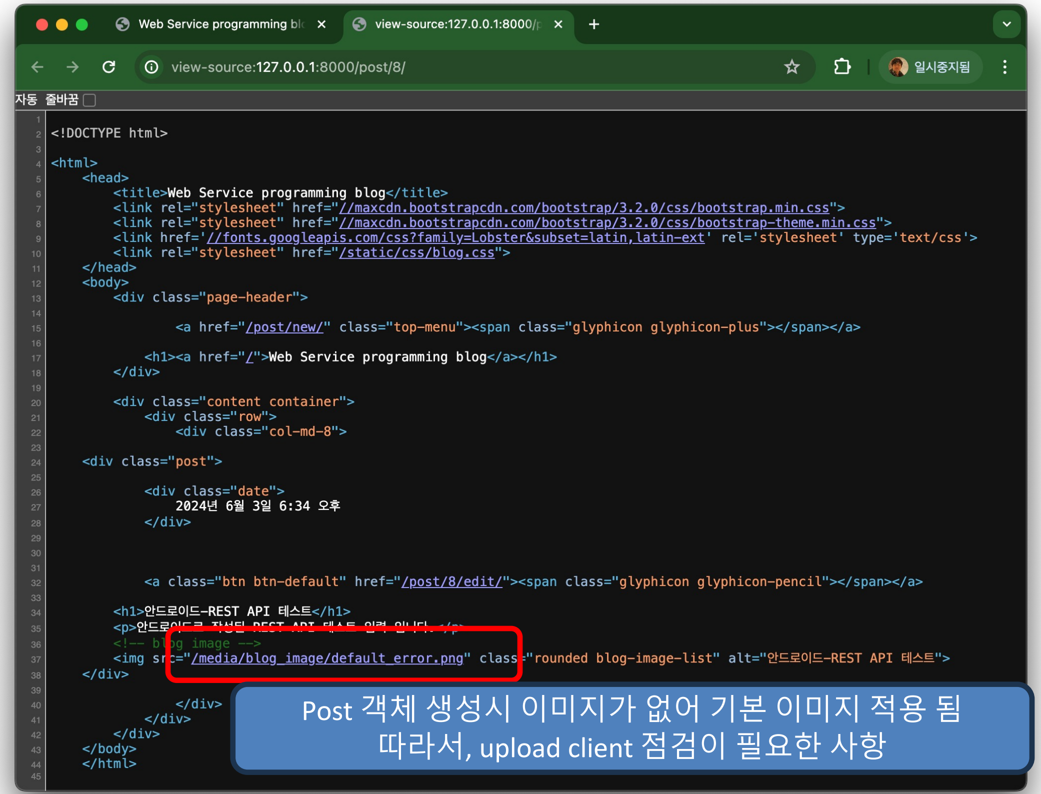
- 사용자 접근 확인 (Admin page등)
- 다양한 도구를 이용한 REST API 테스트
- 웹 클라이언트의 소스 코드 확인



2024년 6월 3일 6:34 오후

안드로이드-REST API 테스트

안드로이드로 작성된 REST API 테스트 입력 입니다.


Web Service programming bl
view-source:127.0.0.1:8000/p
view-source:127.0.0.1:8000/post/8/
일시중지됨
자동 줄바꿈

```html

<!DOCTYPE html>

<html>
    <head>
        <title>Web Service programming blog</title>
        <link rel="stylesheet" href="//maxcdn.bootstrapcdn.com/bootstrap/3.2.0/css/bootstrap.min.css">
        <link rel="stylesheet" href="//maxcdn.bootstrapcdn.com/bootstrap/3.2.0/css/bootstrap-theme.min.css">
        <link href='//fonts.googleapis.com/css?family=Lobster&subset=latin,latin-ext' rel='stylesheet' type='text/css'>
        <link rel="stylesheet" href="/static/css/blog.css">
    </head>
    <body>
        <div class="page-header">

                <a href="/post/new/" class="top-menu"><span class="glyphicon glyphicon-plus"></span></a>

            <h1><a href="/">Web Service programming blog</a></h1>
        </div>

        <div class="content container">
            <div class="row">
                <div class="col-md-8">

    <div class="post">

            <div class="date">
                2024년 6월 3일 6:34 오후
            </div>



            <a class="btn btn-default" href="/post/8/edit/"><span class="glyphicon glyphicon-pencil"></span></a>

        <h1>안드로이드-REST API 테스트</h1>
        <p>안드로이드로 작성된 REST API 테스트 입력 입니다.</p>
        <!-- blog image -->
        <img src="/media/blog_image/default_error.png" class="rounded blog-image-list" alt="안드로이드-REST API 테스트">
    </div>

                </div>
            </div>
        </div>
    </body>
</html>
```

Post 객체 생성시 이미지가 없어 기본 이미지 적용 됨
따라서, upload client 점검이 필요한 사항