

# API – Khái niệm và cách sử dụng trong Android

## 1. API là gì?

API (Application Programming Interface) là một tập hợp các quy tắc cho phép các phần mềm giao tiếp với nhau. API xác định cách mà một phần mềm có thể tương tác và sử dụng dịch vụ của phần mềm khác, thường là qua các phương thức HTTP (như GET, POST) trong trường hợp web API.

Các thành phần cơ bản của API:

- Endpoint: Địa chỉ cụ thể trên web server mà ứng dụng có thể truy cập để thực hiện một chức năng nhất định,

Ví dụ: <https://api.openweathermap.org/data/2.5/weather>.

- HTTP Method: Xác định hành động mà API sẽ thực hiện, phổ biến gồm GET (lấy dữ liệu), POST (gửi dữ liệu), PUT (cập nhật dữ liệu), DELETE (xóa dữ liệu).

- Parameters: Các tham số kèm theo giúp API xử lý yêu cầu cụ thể, như tên thành phố (q=cityname) hay tọa độ (lat và lon).

- Response: API trả về dữ liệu dưới dạng JSON hoặc XML, dễ dàng để phân tích cú pháp và sử dụng trong ứng dụng.

## 2. Cách làm việc với API trong Android

### a. Thêm thư viện hỗ trợ

Android có một số thư viện để làm việc với API như:

- Volley: Thư viện do Google phát triển, tối ưu cho các yêu cầu HTTP. Thư viện này có ưu điểm về tốc độ và khả năng xử lý các yêu cầu không đồng bộ.

- Retrofit: Thư viện của Square hỗ trợ REST API mạnh mẽ, dễ sử dụng với các dịch vụ RESTful.

Ngoài ra, tùy thuộc vào các bài toán cụ thể mà cần thêm các thư viện tương ứng.

### b. Trình tự sử dụng API trong ứng dụng Android

b1. Chuẩn bị các endpoint và API key: Đảm bảo bạn có các endpoint, API key, và đã hiểu cách gửi yêu cầu.

b2. Thiết lập RequestQueue: Trong Volley, RequestQueue là nơi quản lý tất cả các yêu cầu HTTP được gửi đi. Bạn có thể tạo RequestQueue bằng cách gọi `Volley.newRequestQueue(context)`.

b3. Tạo yêu cầu đến API: Dùng `JsonArrayRequest` hoặc `JsonObjectRequest` để gửi các yêu cầu GET, POST,...

- Định nghĩa URL, các tham số (như tên thành phố hoặc tọa độ GPS) và callback để xử lý dữ liệu trả về.

b4. Xử lý phản hồi (Response): Dữ liệu nhận được dưới dạng JSON thường được xử lý qua các lớp `JsonObject` và `JSONArray` trong Android.

b5. Hiển thị kết quả: Sau khi xử lý, hiển thị thông tin trên giao diện người dùng, ví dụ như hiển thị thông tin thời tiết lên `TextView`.

### **3. Thực hiện các thao tác với dữ liệu API**

- Kiểm tra lỗi: Cần có xử lý lỗi khi API không phản hồi, trả về mã lỗi HTTP, hoặc khi có lỗi phân tích dữ liệu JSON.

- Chuyển đổi dữ liệu: Một số dữ liệu có thể cần chuyển đổi, ví dụ nhiệt độ từ Kelvin sang Celsius.

- Quản lý các quyền truy cập: Để lấy vị trí của người dùng, cần có quyền `ACCESS_FINE_LOCATION`.

### **4. Các bước cụ thể trong ứng dụng sử dụng API lấy thông tin thời tiết theo thành phố.**

Lấy danh sách tỉnh/thành phố

b1. Gửi yêu cầu GET đến API <https://provinces.open-api.vn/api/p/> để lấy danh sách các tỉnh/thành phố.

b2. Sử dụng `JsonArrayRequest` để nhận về một mảng JSON, sau đó trích xuất tên từng thành phố từ `JSONArray` và đưa vào `ArrayList`.

b3. Sử dụng `ArrayAdapter` để hiển thị danh sách thành phố lên `Spinner`.

Lấy thông tin thời tiết theo tên thành phố

b1. Sử dụng tên thành phố từ `Spinner`, xây dựng URL với tham số `q=cityname,VN` để lấy thời tiết tại thành phố đó.

b2. Gửi yêu cầu GET đến API

<https://api.openweathermap.org/data/2.5/weather>.

b3. Phân tích cú pháp phản hồi JSON để lấy các thông tin như nhiệt độ (temp), mô tả (description), độ ẩm (humidity).

Lấy thông tin thời tiết dựa trên vị trí

b1. Sử dụng FusedLocationProviderClient để lấy vị trí hiện tại của thiết bị (tọa độ latitude và longitude).

b2. Gửi yêu cầu đến OpenWeatherMap API với các tham số lat và lon để lấy thông tin thời tiết tại vị trí hiện tại của người dùng.

## **5. Một số khái niệm quan trọng khác**

- Xử lý bất đồng bộ: Làm việc với API là bất đồng bộ, có nghĩa là các yêu cầu được gửi đi nhưng không làm gián đoạn giao diện người dùng. Điều này giúp người dùng không phải chờ đợi khi ứng dụng đang lấy dữ liệu từ server.

- Quản lý bộ nhớ và tài nguyên: Khi sử dụng RequestQueue hoặc FusedLocationProviderClient, cần phải quản lý tài nguyên và quyền truy cập để đảm bảo hiệu suất ứng dụng và không gây hao pin.

- Quyền truy cập: Việc lấy vị trí yêu cầu quyền truy cập từ người dùng (ACCESS\_FINE\_LOCATION). Việc này cần được xử lý bằng cách kiểm tra quyền và yêu cầu nếu quyền chưa được cấp.

## Hướng dẫn chi tiết

### 1. Khai báo quyền.

Khai báo quyền truy cập internet và vị trí trong AndroidManifest.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">
    <!-- Khai báo quyền truy cập internet -->
    <uses-permission android:name="android.permission.INTERNET"/>
    <!-- Khai báo quyền truy cập vị trí thiết bị -->
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />

    <application
        <!-- Các khai báo sẵn có -->
    </application>

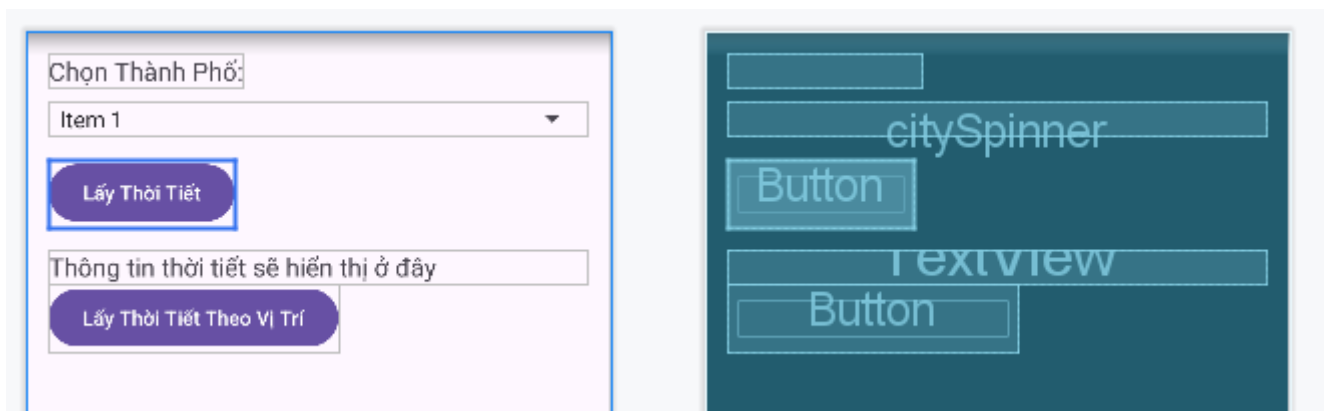
</manifest>
```

### 2. Thêm thư viện.

Thêm các thư viện trong build.gradle (Module: app).

```
dependencies {
    //Các thư viện đang có
    .....
    // Khai báo thêm các thư viện
    implementation 'com.squareup.retrofit2:retrofit:2.9.0'
    implementation 'com.squareup.retrofit2:converter-gson:2.9.0'
    implementation 'com.android.volley:volley:1.2.1'
    implementation 'com.google.android.gms:play-services-location:21.0.1'
}
```

### 3. Xây dựng layout activity\_main.xml



### 4. Xử lý dữ liệu trong MainActivity.java

```
public class MainActivity extends AppCompatActivity {
    private Spinner citySpinner; // Spinner để chọn thành phố
```

```

        private TextView weatherInfoTextView; // TextView để hiển thị thông tin
        thời tiết
        private RequestQueue requestQueue; // Hàng đợi yêu cầu HTTP

        private FusedLocationProviderClient fusedLocationProviderClient; // Client
        để lấy vị trí hiện tại của thiết bị

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

            // Ánh xạ các view từ layout
            citySpinner = findViewById(R.id.citySpinner);
            weatherInfoTextView = findViewById(R.id.weatherInfoTextView);
            Button getWeatherButton = findViewById(R.id.getWeatherButton);

            // Khởi tạo RequestQueue để thực hiện các yêu cầu HTTP
            requestQueue = Volley.newRequestQueue(this);

            // Tải danh sách các thành phố vào Spinner
            loadCities();

            // Thiết lập sự kiện nhấn nút để lấy thông tin thời tiết theo thành phố
            đã chọn
            getWeatherButton.setOnClickListener(v -> getWeatherInfo());

            // Kiểm tra quyền và lấy vị trí hiện tại
            fusedLocationProviderClient =
            LocationServices.getFusedLocationProviderClient(this);
            Button getWeatherByLocationButton =
            findViewById(R.id.getWeatherByLocationButton);

            // Thiết lập sự kiện nhấn nút để lấy thời tiết theo vị trí hiện tại
            getWeatherByLocationButton.setOnClickListener(v ->
            getLocationWeather());
        }

        // Phương thức tải danh sách thành phố từ API
        private void loadCities() {
            String apiUrl = "https://provinces.open-api.vn/api/p/"; // URL API lấy
            danh sách thành phố

            // Tạo một yêu cầu GET để lấy danh sách các thành phố
            JsonArrayRequest request = new JsonArrayRequest(Request.Method.GET,
            apiUrl, null,
            response -> {

```

```

        ArrayList<String> cities = new ArrayList<>(); // Danh sách
các tên thành phố
        for (int i = 0; i < response.length(); i++) {
            try {
                JSONObject cityObject = response.getJSONObject(i);
                String cityName = cityObject.getString("name"); //
Lấy tên thành phố từ JSON
                cities.add(cityName);
            } catch (JSONException e) {
                e.printStackTrace();
            }
        }
        // Tạo Adapter và thiết lập cho Spinner (sử dụng layout
Spinner có sẵn) để hiển thị danh sách thành phố
        ArrayAdapter<String> adapter = new ArrayAdapter<>(this,
android.R.layout.simple_spinner_item, cities);
        citySpinner.setAdapter(adapter);
    },
    error -> weatherInfoTextView.setText("Không thể tải danh sách
tỉnh/thành phố.") // Xử lý lỗi nếu không tải được danh sách
);

    // Thêm yêu cầu vào RequestQueue để thực thi
    requestQueue.add(request);
}

// Phương thức lấy thông tin thời tiết theo tên thành phố được chọn
private void getWeatherInfo() {
    String selectedCity = citySpinner.getSelectedItem().toString(); // Lấy
tên thành phố từ Spinner
    String formattedCityName = formatCityName(selectedCity); // Định dạng
tên thành phố phù hợp với API
    String apiKey = "01aa2fb1a62d65d83bd212c9ecea0674"; // API key của
OpenWeatherMap

    // URL API lấy thông tin thời tiết
    String url = "https://api.openweathermap.org/data/2.5/weather?q=" +
formattedCityName + "&appid=" + apiKey + "&lang=vi";

    // Tạo yêu cầu GET để lấy thông tin thời tiết
    JsonObjectRequest weatherRequest = new
JsonObjectRequest(Request.Method.GET, url, null,
        response -> {
            try {
                JSONObject main = response.getJSONObject("main");
                double temp = main.getDouble("temp") - 273.15; //
Chuyển đổi từ Kelvin sang độ Celsius

```

```

        String description =
response.getJSONArray("weather").getJSONObject(0).getString("description"); //
Lấy mô tả thời tiết

        int humidity = main.getInt("humidity"); // Lấy độ ẩm

        // Tạo chuỗi hiển thị thông tin thời tiết
        String weatherInfo = "Thời tiết tại " + selectedCity +
":\n" +
        "Nhiệt độ: " + String.format("%.2f", temp) +
"°C\n" +
        "Mô tả: " + description + "\n" +
        "Độ ẩm: " + humidity + "%";
        weatherInfoTextView.setText(weatherInfo);
    } catch (JSONException e) {
        e.printStackTrace();
        weatherInfoTextView.setText("Lỗi khi phân tích thông
tin thời tiết.");
    }
},
    error -> weatherInfoTextView.setText("Không thể lấy thông tin
thời tiết.") // Xử lý lỗi nếu không thể lấy thông tin thời tiết
);

    // Thêm yêu cầu vào RequestQueue để thực thi
    requestQueue.add(weatherRequest);
}

    // Phương thức định dạng tên thành phố (bỏ tiền tố "Tỉnh", "Thành phố" và
thêm mã quốc gia ",VN")
    private String formatCityName(String cityName) {
        if (cityName.startsWith("Thành phố ")) {
            return cityName.replace("Thành phố ", "").trim() + ",VN";
        } else if (cityName.startsWith("Tỉnh ")) {
            return cityName.replace("Tỉnh ", "").trim() + ",VN";
        }
        return cityName.trim() + ",VN"; // Nếu không có tiền tố nào
    }

    // Phương thức lấy thông tin thời tiết theo vị trí hiện tại
    private void getLocationWeather() {
        // Kiểm tra quyền truy cập vị trí, nếu chưa được cấp quyền thì yêu cầu
quyền từ người dùng
        if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED
&&

```

```

        ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {
            ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.ACCESS_FINE_LOCATION}, 100);
            return;
        }

        // Lấy vị trí cuối cùng biết được của thiết bị
        fusedLocationProviderClient.getLastLocation().addOnSuccessListener(this
, location -> {
            if (location != null) {
                double latitude = location.getLatitude(); // Lấy vĩ độ
                double longitude = location.getLongitude(); // Lấy kinh độ
                getWeatherInfoByCoordinates(latitude, longitude); // Lấy thời
tiết theo tọa độ
            } else {
                weatherInfoTextView.setText("Không thể lấy vị trí của thiết
bị.");
            }
        });
    }

    // Phương thức lấy thông tin thời tiết theo tọa độ (latitude, longitude)
    private void getWeatherInfoByCoordinates(double latitude, double longitude)
{
        String apiKey = "01aa2fb1a62d65d83bd212c9ecea0674"; // API key của
OpenWeatherMap
        String url = "https://api.openweathermap.org/data/2.5/weather?lat=" +
latitude + "&lon=" + longitude + "&appid=" + apiKey + "&lang=vi";

        // Tạo yêu cầu GET để lấy thông tin thời tiết từ tọa độ
        JsonObjectRequest weatherRequest = new
JsonObjectRequest(Request.Method.GET, url, null,
            response -> {
                try {
                    JSONObject main = response.getJSONObject("main");
                    double temp = main.getDouble("temp") - 273.15; //
Chuyển đổi từ Kelvin sang độ Celsius
                    String description =
response.getJSONArray("weather").getJSONObject(0).getString("description"); //
Lấy mô tả thời tiết
                    int humidity = main.getInt("humidity"); // Lấy độ ẩm

                    // Tạo chuỗi hiển thị thông tin thời tiết
                    String weatherInfo = "Thời tiết tại vị trí hiện tại:\n"
+

```



```

        "Nhiệt độ: " + String.format("%.2f", temp) +
        "°C\n" +
        "Mô tả: " + description + "\n" +
        "Độ ẩm: " + humidity + "%";
        weatherInfoTextView.setText(weatherInfo);
    } catch (JSONException e) {
        e.printStackTrace();
        weatherInfoTextView.setText("Lỗi khi phân tích thông
tin thời tiết.");
    }
},
error -> weatherInfoTextView.setText("Không thể lấy thông tin
thời tiết.") // Xử lý lỗi nếu không thể lấy thông tin thời tiết
);

// Thêm yêu cầu vào RequestQueue để thực thi
requestQueue.add(weatherRequest);
}
}

```