LẬP TRÌNH ANDROID CĂN BẢN

Bài 8: Networking

Ths. Trần Xuân Thanh Phúc | Trường Đại học Công Nghiệp Thực Phẩm

Nội dung

- Tổng quan
- RESTful API, Status code
- HttpURLConnection
- Retrofit2

1. Tổng quan

- Các thiết bị di động ở thời điểm hiện tại có thể quy chung về hình thức kết nối sau:
  - Mobile Internet: một dạng kết nối thông qua hình thức cung cấp dịch vụ băng thông của nhà mạng, bao gồm một số chuẩn phổ biến sau: GPRS, EDGE, 3G, 4G, LTE...
  - Wifi: kết nối không dây và truyền dữ liệu ra Internet thông các thiết bị mạng (router, switch...), ngoàira thiết bị di động Android có khả năng phát tín hiệu mạng cho các thiết bị khác.

1. Tổng quan

- Để làm việc với mạng, yêu cầu bạn cần cấp một số quyền cho ứng dụng:

```
<uses-permission android:name="android.permission.INTERNET"/>
// cho phép ứng dụng có khả năng kết nối mạng.

<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
//Cho phép ứng dụng kiểm tra trạng thái kết nối mạng của điện thoại.
```

1. Tổng quan

```
private boolean checkInternetConnection() {
    ConnectivityManager connManager = (ConnectivityManager) getSystemService(Context.CONNECTIVITY_SERVICE);
    NetworkInfo networkInfo = connManager.getActiveNetworkInfo();

    if (networkInfo == null) {
        Toast.makeText(this, "No default network is currently active", Toast.LENGTH_LONG).show();
        return false;
    }

    if (!networkInfo.isConnected()) {
        Toast.makeText(this, "Network is not connected", Toast.LENGTH_LONG).show();
        return false;
    }

    if (!networkInfo.isAvailable()) {
        Toast.makeText(this, "Network not available", Toast.LENGTH_LONG).show();
        return false;
    }
    Toast.makeText(this, "Network OK", Toast.LENGTH_LONG).show();
    return true;
}
```

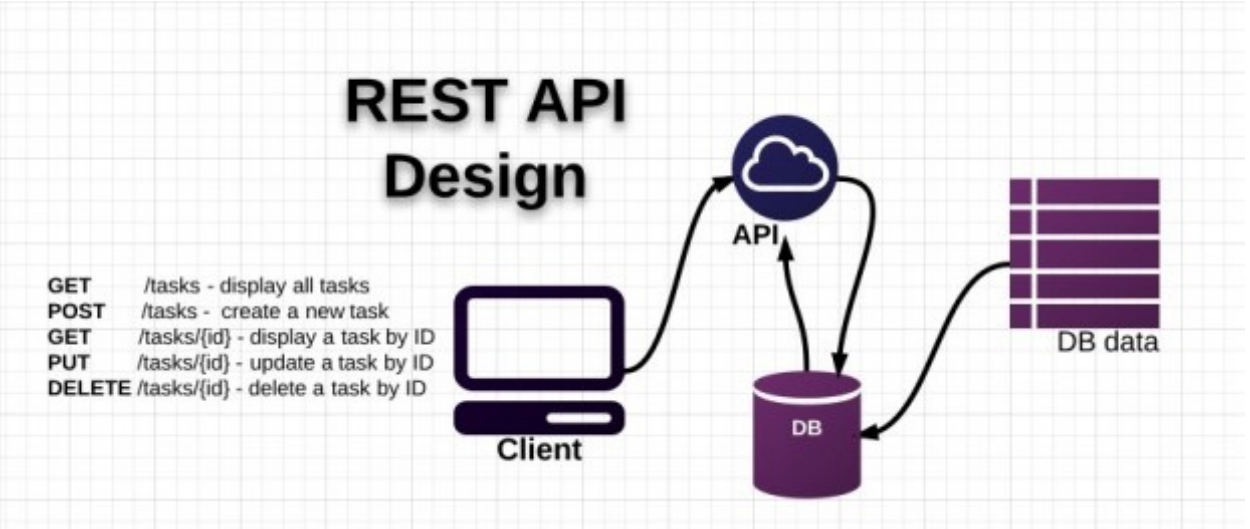
1. Tổng quan

- Theo mặc định khi làm việc với network trong Android, lập trình viên nên tạo ra một thread mới để gửi và nhận dữ liệu trả về. Nếu làm việc trên UI Thread thì khi khởi động sẽ nhận được lỗi android.os.NetworkOnMainThreadException, đây là chính sách mặc định của Android.
- Tuy nhiên có thể ghi đè chính sách này của Android để có thể làm việc với Network trên UI Thread (không khuyến khích).

```
StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().permitAll().build();
StrictMode.setThreadPolicy(policy);
```

2. RESTful API

- **RESTful (REpresentational State Transfer) API** là một tiêu chuẩn dùng trong việc thiết kế API cho các ứng dụng web (thiết kế Web services) để tiện cho việc quản lý các resource. Nó chú trọng vào tài nguyên hệ thống (tệp văn bản, ảnh, âm thanh, video, hoặc dữ liệu động…), bao gồm các trạng thái tài nguyên được định dạng và được truyền tải qua HTTP.



2. RESTful API

- REST hoạt động chủ yếu dựa vào giao thức HTTP. Các hoạt động cơ bản nêu trên sẽ sử dụng những phương thức HTTP riêng.
  - GET (SELECT): Trả về một Resource hoặc một danh sách Resource.
  - POST (CREATE): Tạo mới một Resource.
  - PUT (UPDATE): Cập nhật thông tin cho Resource.
  - DELETE (DELETE): Xoá một Resource.
- Những phương thức hay hoạt động này thường được gọi là CRUD tương ứng với Create, Read, Update, Delete – Tạo, Đọc, Sửa, Xóa.

2. Status code

200 OK – Trả về thành công cho những phương thức GET, PUT, PATCH hoặc DELETE.

201 Created – Trả về khi một Resouce vừa được tạo thành công.

204 No Content – Trả về khi Resource xoá thành công.

304 Not Modified – Client có thể sử dụng dữ liệu cache.

400 Bad Request – Request không hợp lệ

401 Unauthorized – Request cần có auth.

403 Forbidden – bị từ chối không cho phép.

404 Not Found – Không tìm thấy resource từ URI

405 Method Not Allowed – Phương thức không cho phép với user hiện tại.

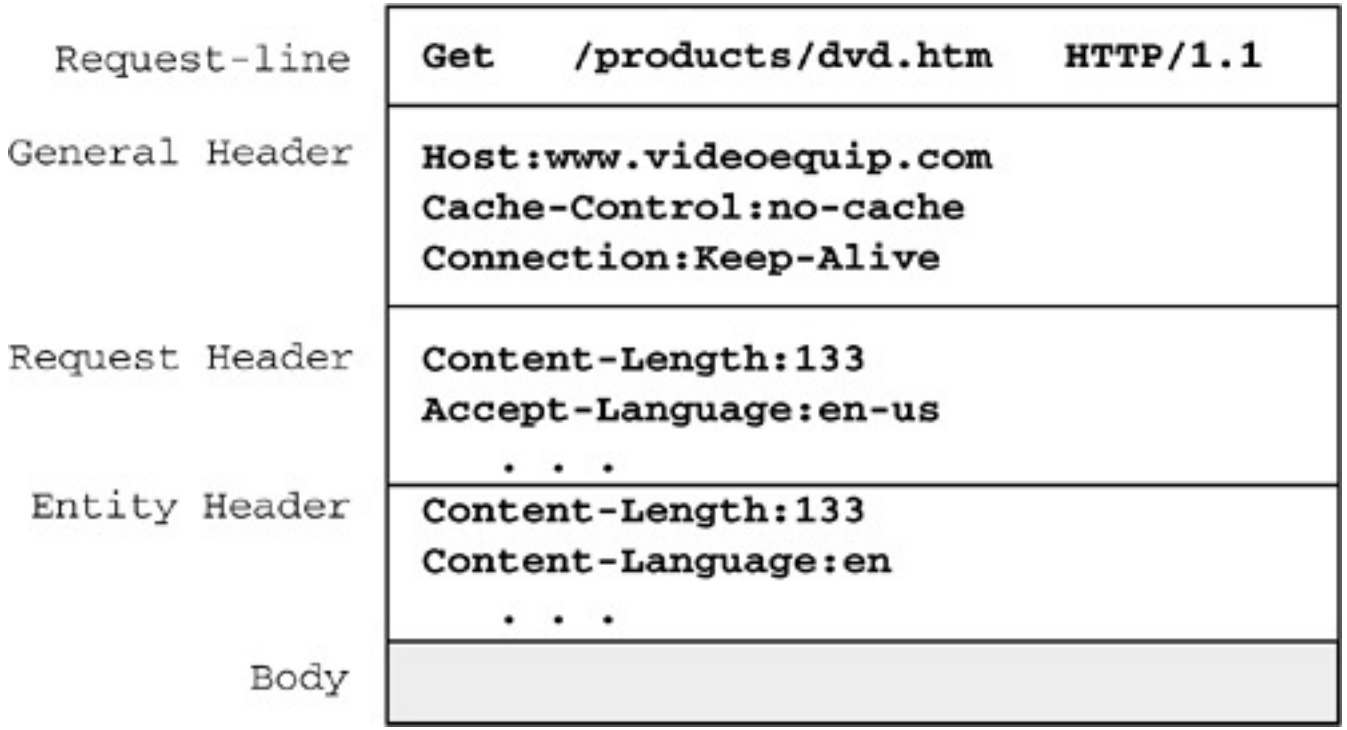
410 Gone – Resource không còn tồn tại, Version cũ đã không còn hỗ trợ.

415 Unsupported Media Type – Không hỗ trợ kiểu Resource này.

422 Unprocessable Entity – Dữ liệu không được xác thực

429 Too Many Requests – Request bị từ chối do bị giới hạn

2. Request structure



3. HttpURLConnection

- Đây là một API chính thức của Android, nó bắt đầu được đưa vào từ phiên bản Android 2.3, trước đó Android sử dụng Apache HttpClient để làm việc với mạng.

```
URL url = new URL("https://styles.redditmedia.com/t5_49lwkx/styles/communityIcon_uh0cp7euvg571.png");
URLConnection connection = url.openConnection();
HttpURLConnection httpURLConnection = (HttpURLConnection) connection;

// Get header value by field
String headerContentType = httpURLConnection.getHeaderField("Content-Type")

// For success request
InputStream is = (InputStream) httpURLConnection.getInputStream();

// For error request
InputStream is = (InputStream) httpURLConnection.getErrorStream();
```

3. HttpURLConnection

- Sử dụng bộ thư viện Jacson Databind để chuyển đổi dữ liệu Plain Object thành JSON và ngược lại

```
// Gradle
implementation group: 'com.fasterxml.jackson.core', name: 'jackson-databind', version: '2.14.2'
```

```
// Maven
<dependency>
        <groupId>com.fasterxml.jackson.core</groupId>
        <artifactId>jackson-databind</artifactId>
        <version>2.14.2</version>
</dependency>
```

4. Retrofit2

- Được phát triển bởi Square
- Retrofit là một HTTP client type-safe cho Android và Java. Retrofit giúp dễ dàng kết nối đến một dịch vụ REST trên web bằng cách chyển đổi API thành Java Interface.
- Tất cả các yêu cầu GET, POST, PUT, PATCH, và DELETE đều có thể được thực thi.
- Chuyển đổi JSON, XML thành Plain Object

4. Retrofit2

```
// Gradle app
implementation 'com.squareup.retrofit2:retrofit:2.8.0'
implementation 'com.squareup.retrofit2:converter-jackson:2.8.0'
```

- Để làm việc với Retrofit, cần triển khai cơ bản 3 lớp:
  - Model class để ánh xạ với JSON data.
  - Một interface dùng để định nghĩa các hàm và phương thức HTTP
  - Retrofit.Builder Lớp để định nghĩa URL Endpoint cho các hoạt động liên quan tới HTTP

4. Retrofit2

```
public class DemoModel {
    protected String game;
    public String getGame() {
        return game;
    }
}
```

```
public interface DemoApi {
    @GET("/demo/game")
    Call<DemoObject> getDemo();
}
```

```
Retrofit retrofit = new Retrofit.Builder()
        .baseUrl("http://10.0.0.2:7890/")
        .addConverterFactory(JacksonConverterFactory.create())
        .build();

DemoApi api = retrofit.create(DemoApi.class);
api.getDemo().enqueue(new Callback<DemoModel>() {
    @Override
    public void onResponse(Call<DemoModel> call, Response<DemoModel> response) {

    }

    @Override
    public void onFailure(Call<DemoModel> call, Throwable t) {

    }
});
```

LẬP TRÌNH ANDROID CĂN BẢN

Kết thúc ☺️

Ths. Trần Xuân Thanh Phúc | Trường Đại học Công Nghiệp Thực Phẩm