Networking

Mobile App Programming

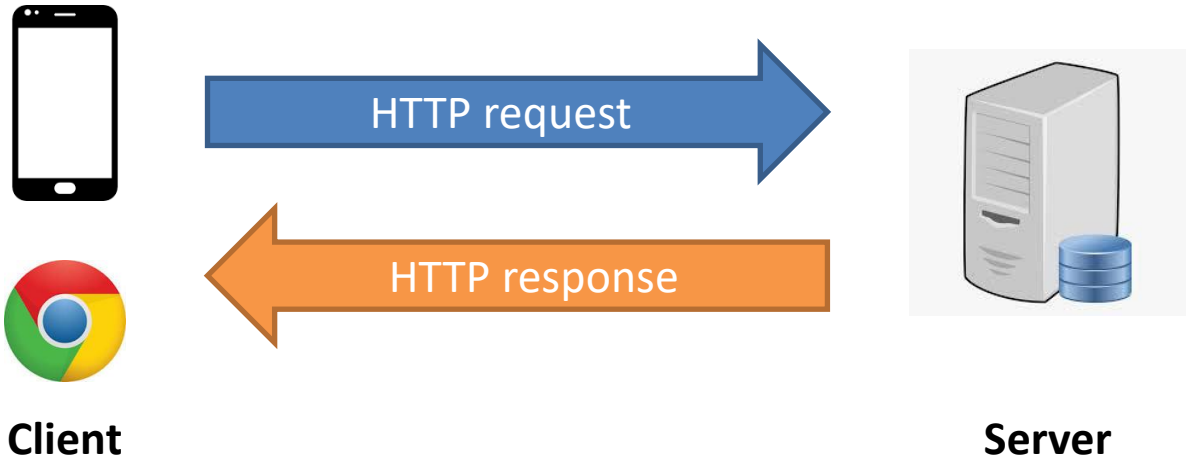
What we learn today?

- **Let's learn the Android Network Connection.**
  - HTTP Networking in Android
    - HTTP using "okhttp"
    - Parsing JSON Using GSON

- **edu.skku.MAP.week10**
  - Create project which contains Empty Views Activity

HTTP – Hypertext Transfer Protocol

- **What is HTTP?**
  - HTTP is an application layer protocol for web service.
  - Client usually send **HTTP request** to server and receive the HTTP response.
  - There are many HTTP methods (**GET, POST, PUT, DELETE**)



HTTP – GET and POST

- **We will learn two HTTP methods, GET and POST**
- **GET** is used to **request data** from server
  - Ex) Get "student's name" whose id is "2018711586".
- Using GET method, you can get data from server.

- **POST** is used **to send data to a server to create/update a resource.**
  - Ex) Add/Update a user information on the server DB.
- Using POST method, you can send a data and create a new resource.
- To send data to server, you just put data on **request body.**
  - Usually use formatted data like **JSON** (similar with dictionary).

Let’s send HTTP request in android

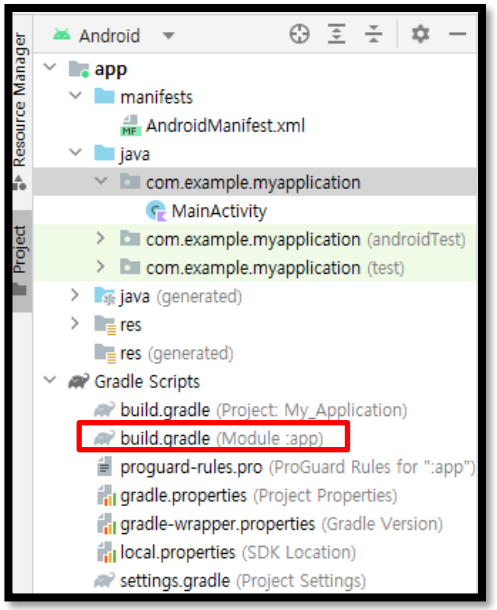
- There is a developer page for HTTP communication, named HTTPUrlConnection.
- However, in this class, we will use simple open-source API named **okhttp3** (which is easier to use.)
- https://square.github.io/okhttp/

- We don’t have any server-side program, so we will use “**reqres**” website as server in this lecture.
- https://reqres.in/

Before start (VERY IMPORTANT)

- You should add implementation in your **build.gradle(Module)** script dependencies and click **syncNow**.

```
dependencies {
        implementation ("com.squareup.okhttp3:okhttp:4.9.0“)
        implementation ("com.google.code.gson:gson:2.8.7“)
}
```



Before start (VERY IMPORTANT)

- And to use internet in your application add uses-permission in your manifest file. And Add usesCleartextTraffic as "true".

activity_main.xml | MainActivity.kt | AndroidManifest.xml | build.gradle (My Application) | build.g

```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">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="My Application"
        android:supportsRtl="true"
        android:theme="@style/Theme.MyApplication"
        android:usesCleartextTraffic="true"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>


    <uses-permission android:name="android.permission.INTERNET"/>

</manifest>
```

https://gist.github.com/RmKuma/68887e08b75fd76e0abec18476496ee4

Example 1. send GET request

```
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        val btn = findViewById<Button>(R.id.button)
        val tv = findViewById<TextView>(R.id.textView)
        val tv2 = findViewById<TextView>(R.id.textView2)

        val client = OkHttpClient()
        val host = "https://reqres.in"
        val path = "/api/users?page=2"

        btn.setOnClickListener { it: View!
            tv.text = host + path
            val req = Request.Builder().url(host + path).build()
            client.newCall(req).enqueue(object : Callback{
                override fun onFailure(call: Call, e: IOException) {
                    e.printStackTrace()
                }
                override fun onResponse(call: Call, response: Response) {
                    response.use{ it: Response
                        if (!response.isSuccessful) throw IOException("Unexpected code $response")
                        val str = response.body!!.string()
                        CoroutineScope(Dispatchers.Main).launch { this: CoroutineScope
                            tv2.text = str
                        } ^use
                    }
                }
            })
        }
    }
}
```

! IMPORTANT

```
import okhttp3.Call
import okhttp3.Callback
import okhttp3.OkHttpClient
import okhttp3.Request
import okhttp3.Response
```

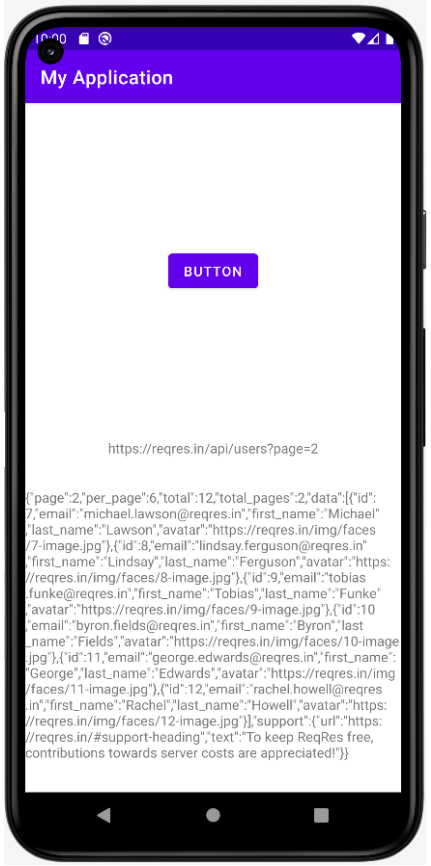
Use OkHttpClient and Request classes defined in Okhttp API

Build request with URL (host + path)

Put request into queue for asynchronous networking

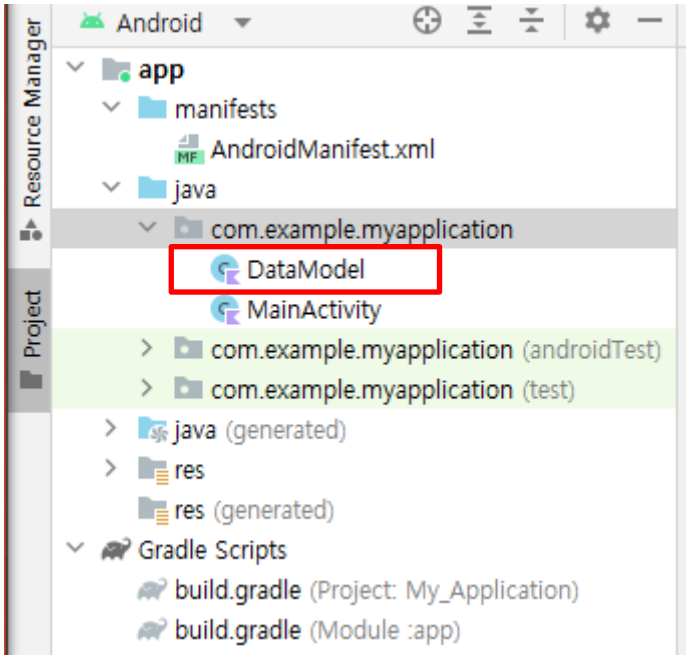
When changing text view, you must run it in **Main(UI) Thread!**

onResponse will be executed by IO thread.



Parsing with GSON

- As a response of request, You will get data with **JSON** format.
- Therefore, you should parse it with **gson API**.
  - https://github.com/google/gson
- And Let's make data's class which json will be converted to.

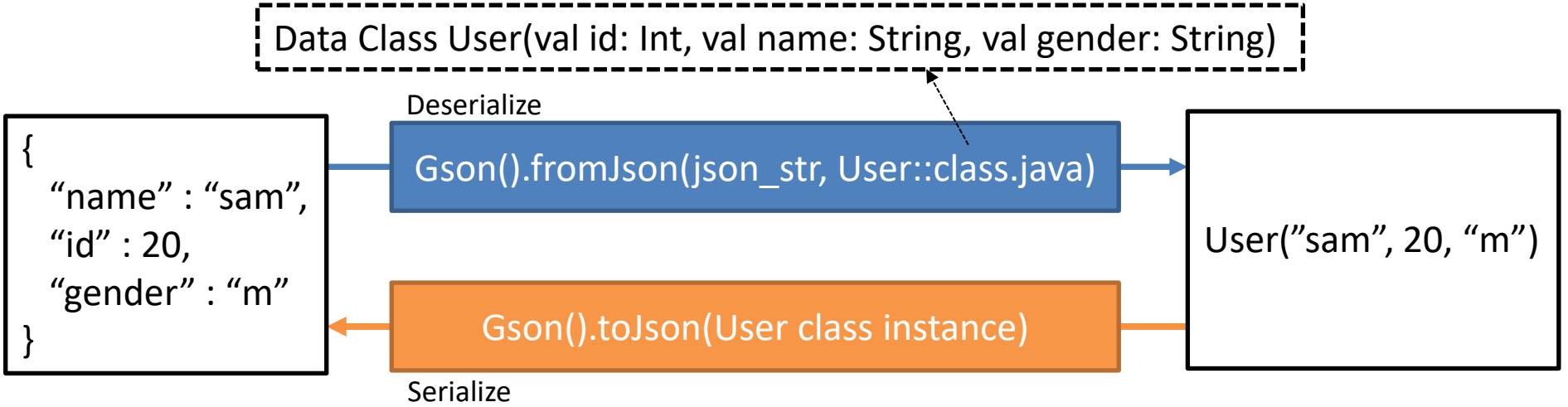


Parsing with GSON - JSON

- JSON (Javascript Object Notation)
  - JSON is data structure in the manner of JS grammer
    - Similar with **dictionary** data structure
    - Key-value pairs are represented using { }
    - **Lists** are represented using [ ]
    - An object must be composed using { }
    - Keys are strings enclosed in double quotes (")
    - Values are enclosed in double quotes (") if they are strings, but remain without double quotas if they are integers, booleans, etc.
    - Key-value pairs are separated by commas and take the form
      - key : value, key : value, key : value …
    - Values can also contain other objects or lists.

Parsing with GSON - JSON

- If you want to make JSON to store user information (name, id, gender)
  - Name, id, gender will be a keys of JSON object
    - Ex) {"name": "James", "id": 123, "gender": "m"}
- GSON() is a **serializing & deserializing tool** between JSON and Kotlin object.
  - GSON().fromJson() convert JSON string to kotlin instance
  - GSON().toJson() convert kotlin instance to JSON string
  - When defining a kotlin class for deserialize, you don't have to define all keys in json string. Define only the required key-value pairs.



https://gist.github.com/RmKuma/f9d47b780fe8679fee6cf7b6c16b0250

Example 2. GSON for parsing

- From response body, how to get all users last name?
  -> **Parsing** the data use GSON API

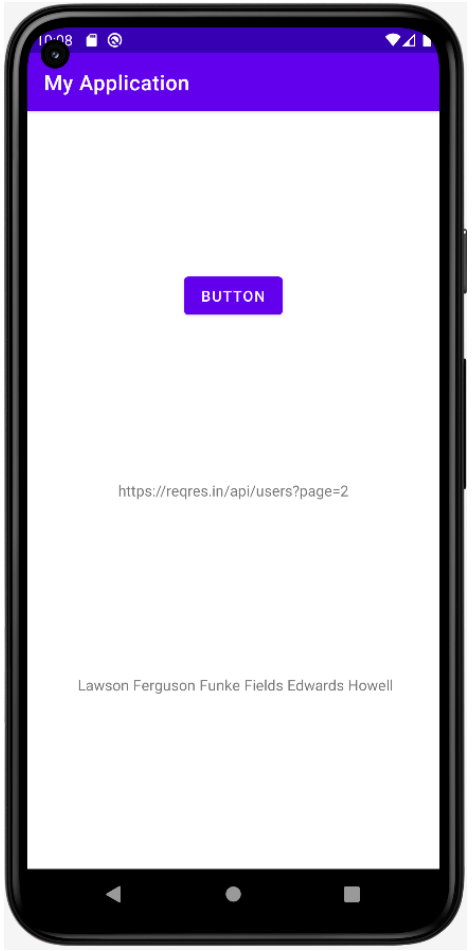
DataModel.kt

```
package com.example.myapplication

data class DataModel(var page: Int ?= null, var data: ArrayList<User>){
   data class User(var id: Int ?= null, var last_name: String ?= null){
   }
}
```

MainActivity.kt. -> In onResponse function

```
response.use{ it: Response
    if (!response.isSuccessful) throw IOException("Unexpected code $response")
    val str = response.body!!.string()
    val data = Gson().fromJson(str, DataModel::class.java)
    CoroutineScope(Dispatchers.Main).launch { this: CoroutineScope
        var concat : String = ""
        for (i:Int in 0 ≤ .. ≤ data.data.size -1)
            concat += " " + data.data[i].last_name
        tv2.text = concat
    } ^use
}
```



Example 2. GSON for parsing

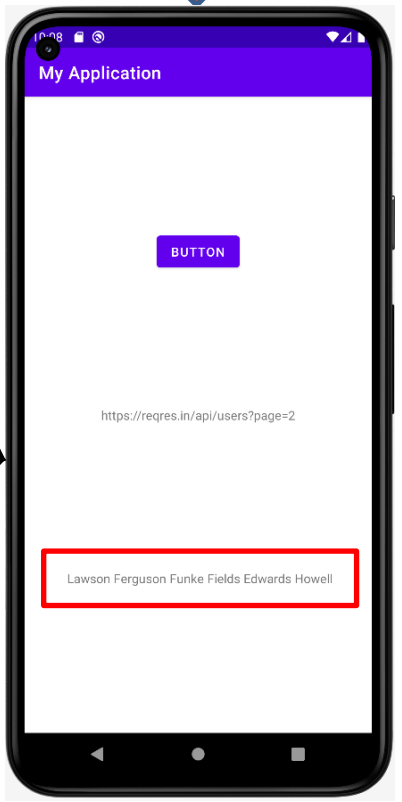
Result


```
{
  "page": 2,
  "per_page": 6,
  "total": 12,
  "total_pages": 2,
  "data": [
    {
      "id": 7,
      "email": "michael.lawson@reqres.in",
      "first_name": "Michael",
      "last_name": "Lawson",
      "avatar": "https://reqres.in/img/faces/7-image.jpg"
    },
    {
      "id": 8,
      "email": "lindsay.ferguson@reqres.in",
      "first_name": "Lindsay",
      "last_name": "Ferguson",
      "avatar": "https://reqres.in/img/faces/8-image.jpg"
    },
    {
      "id": 9,
      "email": "tobias.funke@reqres.in",
      "first_name": "Tobias",
      "last_name": "Funke",
      "avatar": "https://reqres.in/img/faces/9-image.jpg"
    },
    {
      "id": 10,
      "email": "byron.fields@reqres.in",
      "first_name": "Byron",
      "last_name": "Fields",
      "avatar": "https://reqres.in/img/faces/10-image.jpg"
    },
    {
      "id": 11,
      "email": "george.edwards@reqres.in",
      "first_name": "George",
      "last_name": "Edwards",
      "avatar": "https://reqres.in/img/faces/11-image.jpg"
    },
    {
      "id": 12,
      "email": "rachel.howell@reqres.in",
      "first_name": "Rachel",
      "last_name": "Howell",
      "avatar": "https://reqres.in/img/faces/12-image.jpg"
    }
  ]}
```


```
package com.example.myapplication

data class DataModel(var page: Int ?= null, var data: ArrayList<User>){
    data class User(var id: Int ?= null, var last_name: String ?= null){
    }
}
```

The name of variable must be same to result key!



Example 3. Send POST request

- In the case of POST, your application send request with data!

DataModel.kt

```
data class Register(var email: String ?= null,
                    var password: String ?= null)
data class RegisterResponse(var id: String ?= null,
                    var createdAt : String ?= null)
```

MainActivity.kt

```
super.onCreate(savedInstanceState)
setContentView(R.layout.activity_main)

val btn = findViewById<Button>(R.id.button)
val tv = findViewById<TextView>(R.id.textView)
val tv2 = findViewById<TextView>(R.id.textView2)

val client = OkHttpClient()
val host = "https://reqres.in"
val path = "/api/register"

btn.setOnClickListener { it: View!
    tv.text = host + path
    val json = Gson().toJson(Register        ( email: "eve.holt@reqres.in",  password: "pistol"))
    val mediaType = "application/json; charset=utf-8".toMediaType()

    val req = Request.Builder().url(host + path).post(json.toString().toRequestBody(mediaType)).build()
    client.newCall(req).enqueue(object : Callback{
        override fun onFailure(call: Call, e: IOException) {
            e.printStackTrace()
        }
        override fun onResponse(call: Call, response: Response) {
            response.use{ it: Response
                if (!response.isSuccessful) throw IOException("Unexpected code $response")
                val str = response.body!!.string()
                val data = Gson().fromJson(str, RegisterResponse::class.java)
                CoroutineScope(Dispatchers.Main).launch { this: CoroutineScope
                    tv2.text = data.id.toString()
                } ^use
            }
        }
    })
}
```


https://gist.github.com/RmKuma/2aa3a4fead7c7811c4538f8a08c8dc60

[Lab-Practice #10] TV show information

- Make the application which show the episode list of TV show!
- Input TV **show name** and print the episode information(**Season, number, name, airdate**) list. If the result has multiple tv shows, use first one.

[Lab-Practice #10] TV show information

- This week we will make tv show searching application using TV MAZE API.
- https://www.tvmaze.com/api
- You can search the TVShow by name
  - Host: https://api.tvmaze.com
  - URL: **/search/shows?q=:query**
  - Example: https://api.tvmaze.com/search/shows?q=infinite+challenge
  - Result


```
[{"score":1.1961879,"show":{"id":3480,"url":"https://www.tvmaze.com/shows/3480/infinite-challenge","name":"Infinite
Challenge","type":"Variety","language":"Korean","genres":["Comedy"],"status":"Running","runtime":90,"averageRuntime":90,"premiered":"2005-04-
23","ended":null,"officialSite":"http://www.imbc.com/broad/tv/ent/challenge/main.html","schedule":{"time":"18:20","days":["Saturday"]},"rating":
{"average":null},"weight":81,"network":{"id":166,"name":"MBC","country":{"name":"Korea, Republic
of","code":"KR","timezone":"Asia/Seoul"},"officialSite":null},"webChannel":null,"dvdCountry":null,"externals":
{"tvrage":36048,"thetvdb":284209,"imdb":"tt4546568"},"image":
{"medium":"https://static.tvmaze.com/uploads/images/medium_portrait/18/46328.jpg","original":"https://static.tvmaze.com/uploads/images/original_untouched/18/46328.jpg"}
,"summary":"<p><b>Infinite Challenge</b> is a Korean television entertainment program that is distributed and syndicated by MBC. It is recognized as the first \"real
variety\" show in Korean television history. The program is largely unscripted and follows a similar format of challenge-based reality television programs, familiar to
some audiences in the West.</p><p>The challenges are often silly, absurd, or impossible to achieve, so the program takes on the aspect of a satirical comedy variety
show rather than a more standard reality or contest program. In earlier episodes, the show's six hosts and staff would continuously proclaim that, in order to achieve
its comedic purposes, the program was 3-D: Dirty, Dangerous, and Difficult.</p>","updated":1607184511,"_links":{"self":
{"href":"https://api.tvmaze.com/shows/3480"},"previousepisode":{"href":"https://api.tvmaze.com/episodes/1466220"}}}}]
```

[Lab-Practice #10] TV show information

- You can search the episodes of TV Show by show id
  - Host: https://api.tvmaze.com
  - URL: **/shows/:id/episodes.**
  - Example: https://api.tvmaze.com/shows/3480/episodes
  - Result


```
[{"id":231041,"url":"https://www.tvmaze.com/episodes/231041/infinite-challenge-
1x01-tug-of-war-challenge","name":"Tug-of-War
Challenge","season":1,"number":1,"type":"regular","airdate":"2005-04-
23","airtime":"18:30","airstamp":"2005-04-
23T09:30:00+00:00","runtime":90,"rating":
{"average":null},"image":null,"summary":"","_links":{"self":
{"href":"https://api.tvmaze.com/episodes/231041"},"show":
{"href":"https://api.tvmaze.com/shows/3480"}}},
{"id":231042,"url":"https://www.tvmaze.com/episodes/231042/infinite-challenge-
1x02-100-meter-sprint","name":"100-Meter
Sprint","season":1,"number":2,"type":"regular","airdate":"2005-04-
30","airtime":"18:30","airstamp":"2005-04-
30T09:30:00+00:00","runtime":90,"rating":
{"average":null},"image":null,"summary":"","_links":{"self":
{"href":"https://api.tvmaze.com/episodes/231042"},"show":
{"href":"https://api.tvmaze.com/shows/3480"}}},
{"id":231043,"url":"https://www.tvmaze.com/episodes/231043/infinite-challenge-
1x03-motorboat-vs-paddle-boat","name":"Motorboat vs. Paddle
Boat","season":1,"number":3,"type":"regular","airdate":"2005-05-
07","airtime":"18:30","airstamp":"2005-05-
07T09:30:00+00:00","runtime":90,"rating":
{"average":null},"image":null,"summary":"","_links":{"self":
{"href":"https://api.tvmaze.com/episodes/231043"},"show":
{"href":"https://api.tvmaze.com/shows/3480"}}},
{"id":231044,"url":"https://www.tvmaze.com/episodes/231044/infinite-challenge-
1x04-public-bath-challenge","name":"Public Bath
Challenge","season":1,"number":4,"type":"regular","airdate":"2005-05-
14","airtime":"18:30","airstamp":"2005-05-
14T09:30:00+00:00","runtime":90,"rating":
{"average":null},"image":null,"summary":"","_links":{"self":
{"href":"https://api.tvmaze.com/episodes/231044"},"show":
{"href":"https://api.tvmaze.com/shows/3480"}}},
```

[Lab-Practice #10] TV show information

- Hint
  - After you send first get request and receive the id of TV show. And if the first request success, you send second get request to receive episode list of TV show.
  - You should make 2 layout files (activity_main.xml, episode_item.xml)
  - You should make ListViewAdapter.
  - The JSON response looks like below, not dictionary, it's list.
    - To deserialize the list json, we use **type token** of list<element>.

```
val typeToken = object : TypeToken<List<Show>>() {}.type
val id = Gson().fromJson<List<Show>>(data, typeToken)[0].show?.id
```

```
val typeToken = object : TypeToken<List<Episode>>() {}.type
val episodeList = Gson().fromJson<List<Episode>>(data, typeToken)
```

https://gist.github.com/RmKuma/6cacbefd7ecc01dbd840f8346086b429

[Lab-Practice #10] TV show information

- Hint
  - To deserialize the list JSON, we use **type token** of list<element>.

```
val typeToken = object : TypeToken<List<Show>>() {}.type
val id = Gson().fromJson<List<Show>>(data, typeToken)[0].show?.id
```

```
data class Show( val score: Double? = null, val show: Showinfo? = null){
    data class Showinfo(val id: Int?, val name: String?)
}
```


[{"score":1.1961879,"show":{"id":3480,"url":"https://www.tvmaze.com/shows/3480/infinite-challenge","name":"Infinite challenge","type":"Variety","language":"Korean","genres":["Comedy"],"status":"Running","runtime":90,"averageRuntime":90,"premiered":"2005-04-23","ended":null,"officialSite":"http://www.imbc.com/broad/tv/ent/challenge/main.html","schedule":{"time":"18:20","days":["Saturday"]},"rating":{"average":null},"weight":81,"network":{"id":166,"name":"MBC","country":{"name":"Korea, Republic of","code":"KR","timezone":"Asia/Seoul"},"officialSite":null},"webChannel":null,"dvdCountry":null,"externals":{"tvrage":36048,"thetvdb":284209,"imdb":"tt4546568"},"image":{"medium":"https://static.tvmaze.com/uploads/images/medium_portrait/18/46328.jpg","original":"https://static.tvmaze.com/uploads/images/original_untouched/18/46328.jpg"},"summary":" Infinite Challenge is a Korean television entertainment program that is distributed and syndicated by MBC. It is recognized as the first \"real variety\" show in Korean television history. The program is largely unscripted and follows a similar format of challenge-based reality television programs, familiar to some audiences in the West. The challenges are often silly, absurd, or impossible to achieve, so the program takes on the aspect of a satirical comedy variety show rather than a more standard reality or contest program. In earlier episodes, the show's six hosts and staff would continuously proclaim that, in order to achieve its comedic purposes, the program was 3-D: Dirty, Dangerous, and Difficult. ","updated":1607184511,"_links":{"self":{"href":"https://api.tvmaze.com/shows/3480"},"previousepisode":{"href":"https://api.tvmaze.com/episodes/1466220"}}}}]

Show class

https://gist.github.com/RmKuma/6cacbefd7ecc01dbd840f8346086b429