Android Networking – Part 2

Minstrel Chiu

minstrelsy@gmail.com

Outline

- REST
- JSON
- Retrofit
- RxJava

REST (1)

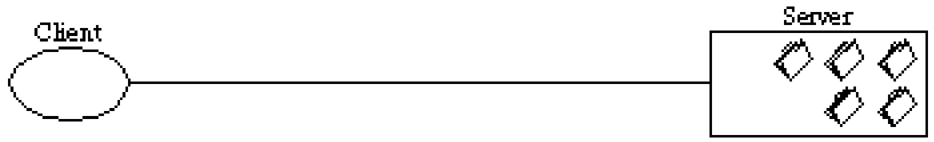
- REST (REpresentational State Transfer) is a way of providing interoperability between computer systems on the Internet for network-based software
- Year 2000, in Roy Fielding’s doctoral dissertation
- Comparison with SOAP, XML-RPC

REST (2)

- Client-Server
- Stateless
- Cache
- Uniform Interface
- Layered System
- Code-On-Demand

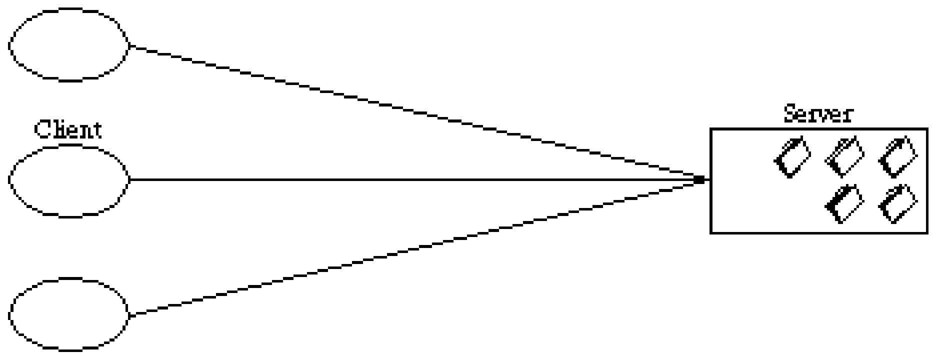
REST (3)

- Client-Server



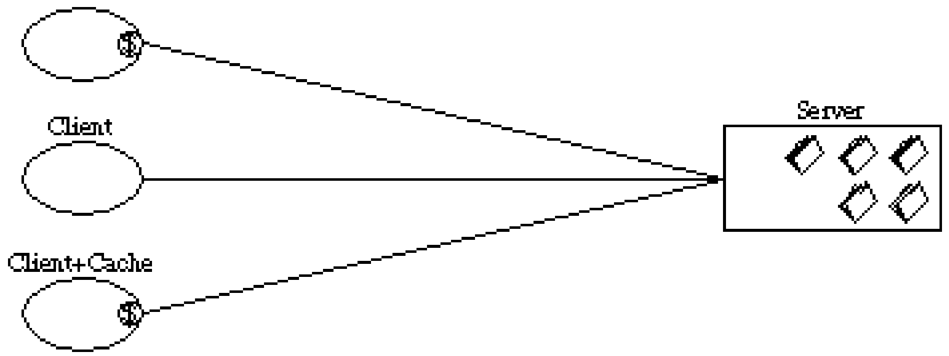
REST (4)

- Stateless
  - Communication must be stateless
  - Session state is on client side



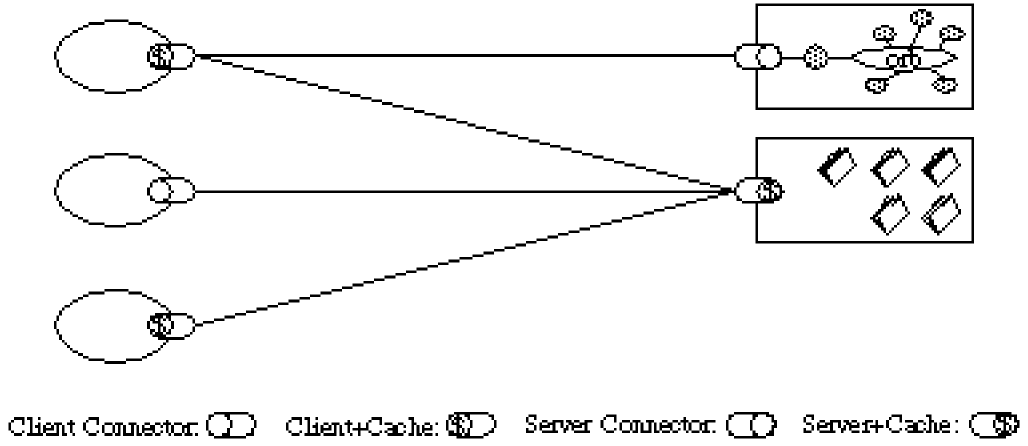
REST (5)

- Cache
  - Client side cache
  - Server side cache



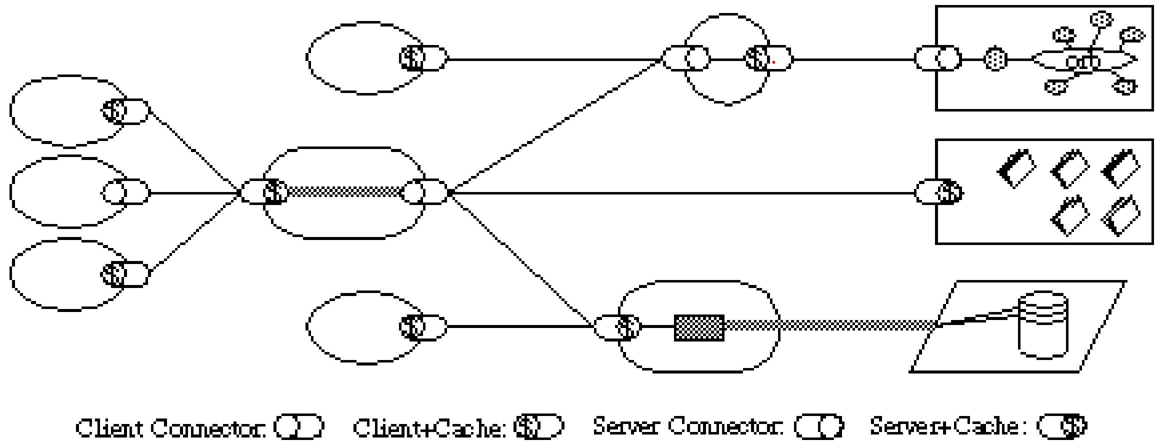
REST (6)

- Uniform Interface



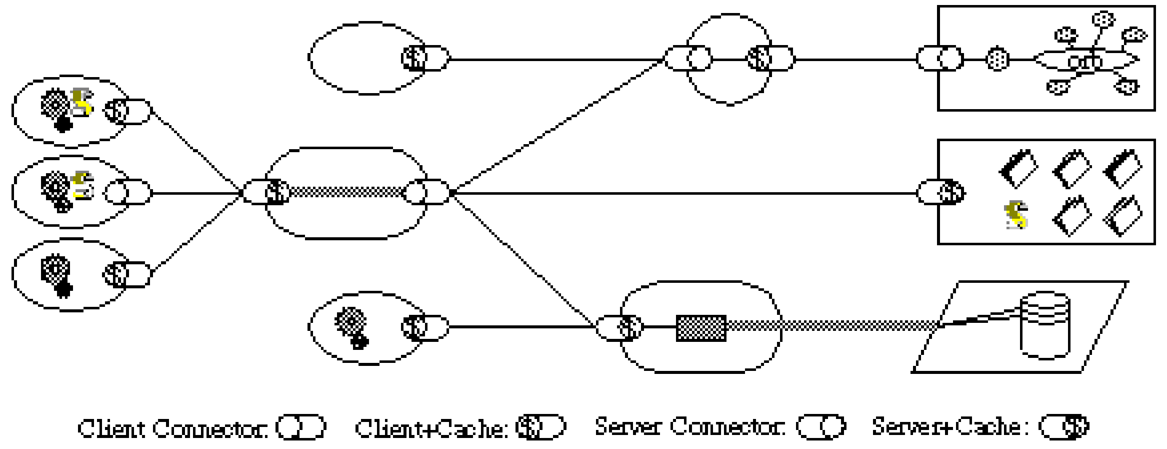
REST (7)

- Layered System



REST (8)

- Code-On-Demand



REST (9)

- An architecture for Web service which adhere to REST is called RESTful
  - Everything in REST in considered as a resource
  - A resource is identified by an URI (Uniform Resource Identifier)
  - Use uniform interfaces for handling Create, Read, Update, Delete (CRUD) operations by HTTP POST, GET, PUT, DELETE
  - Be stateless
  - Transfer representation in HTML, XML or JSON

REST (10)

Resource	**POST** ***Create***	**GET** ***Read***	**PUT** ***Update***	**Delete** ***Delete***
/users	Create a new user	List all users	Update all users entirely (in batch)	Delete all users
/users/1234	Not used, an error is returned	Retrieve an user	Update an user	Delete an user

REST (11)

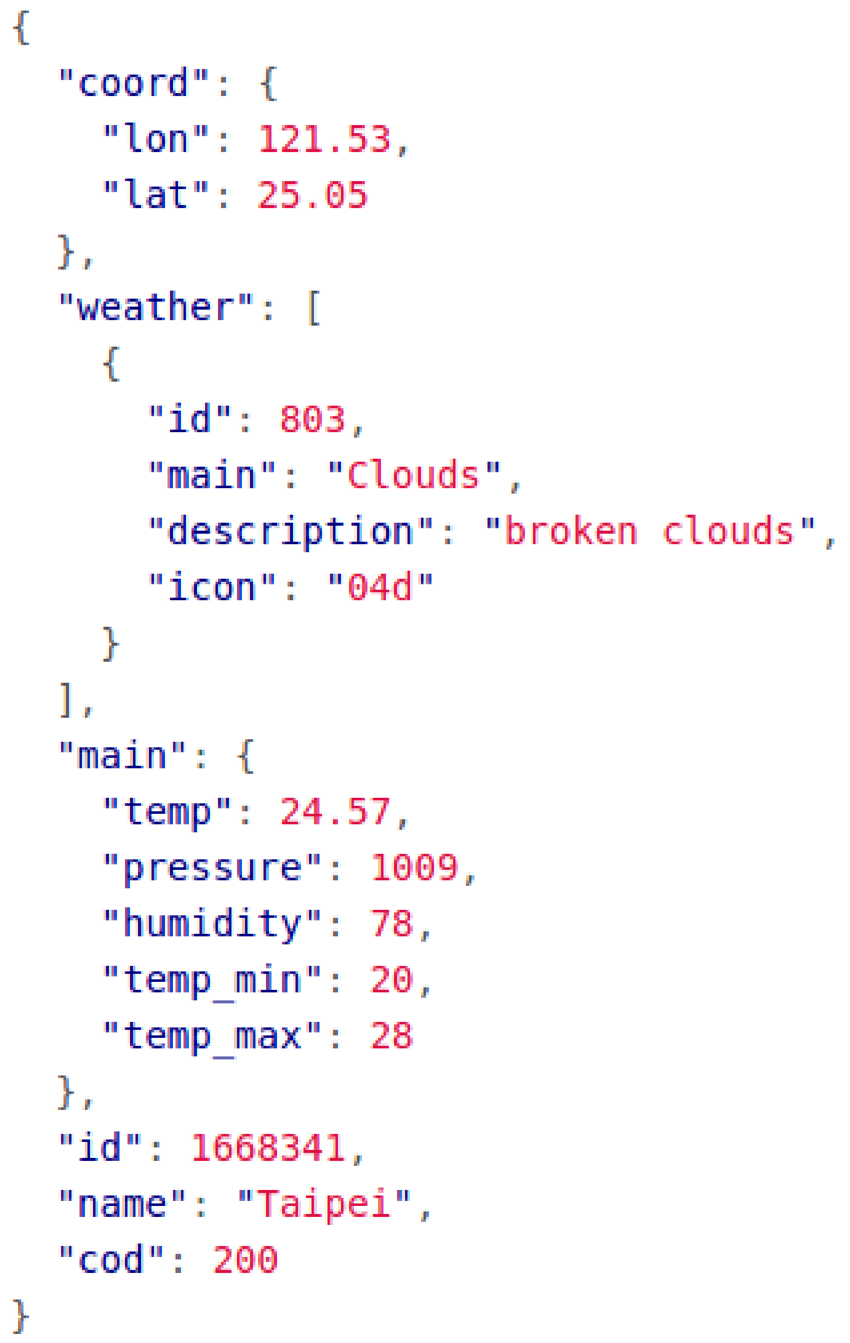
Bad	Good
/api/getAllUsers	GET /api/users
/api/getUserById?id=1234	GET /api/users/1234
/api/createUser	POST /api/users

JSON (1)

- JSON (JavaScript Object Notation) is an open standard data format for human readable data interchange
  - Text based
  - Human readable
  - Light weighted
  - Language independent
- Comparison with XML

JSON (2)

- Data Types
  - Number – double
  - String
  - Boolean – true or false
  - Array – []
  - Object – {}
  - null



```
{
  "coord": {
    "lon": 121.53,
    "lat": 25.05
  },
  "weather": [
    {
      "id": 803,
      "main": "Clouds",
      "description": "broken clouds",
      "icon": "04d"
    }
  ],
  "main": {
    "temp": 24.57,
    "pressure": 1009,
    "humidity": 78,
    "temp_min": 20,
    "temp_max": 28
  },
  "id": 1668341,
  "name": "Taipei",
  "cod": 200
}
```

JSON (3)

- Sample-JSON-1 – JSON parsing demo

Retrofit (1)

- Retrofit is a type-safe HTTP and REST client for Android and Java by Square
- Supports multiple convertors such as JSON and XML
- Uses OkHttp for HTTP requests

Retrofit (2)

- GET /users → *getUsers()*

```
@GET("/users")
List<User> getUsers();
```

- GET /users/1234 → *getUser(1234)*

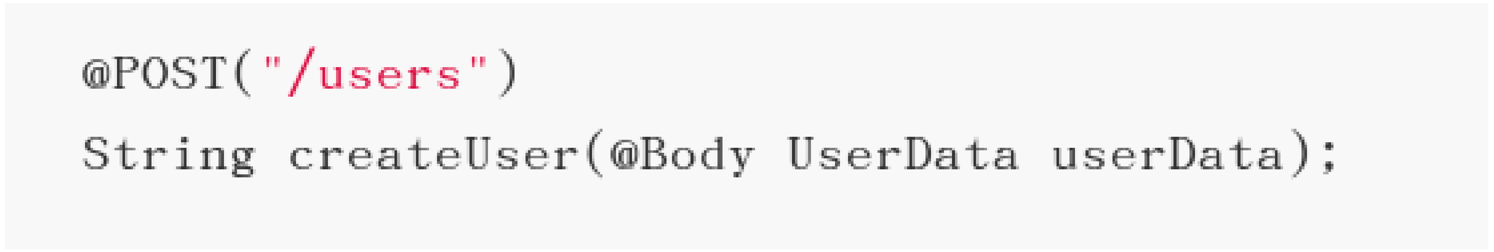
```
@GET("/users/{id}")
User getUser(@Path("id") int userId);
```

- GET /users?sort=desc → *getUsers(“sort”)*

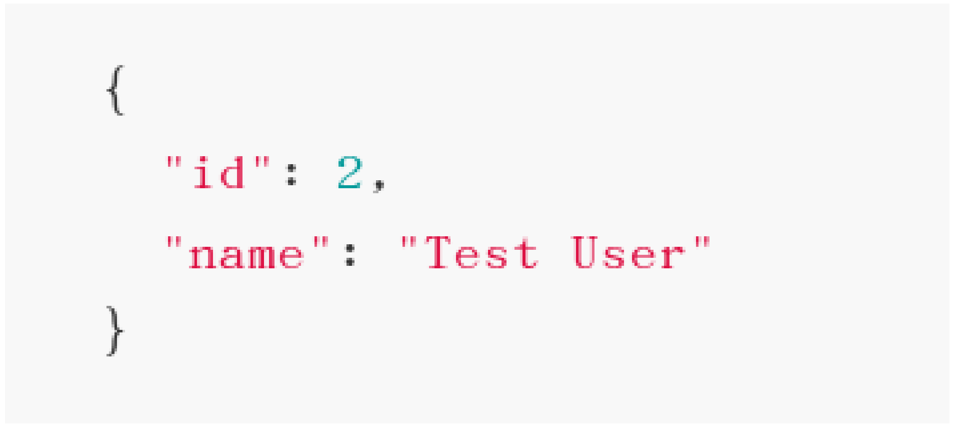
```
@GET("/users")
String getUsers(@Query("sort") String sort);
```

Retrofit (3)

- POST /users → *createUser()*



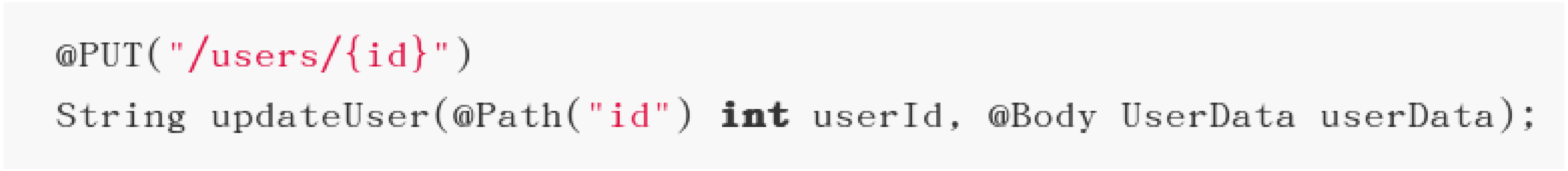
```
@POST("/users")
String createUser(@Body UserData userData);
```



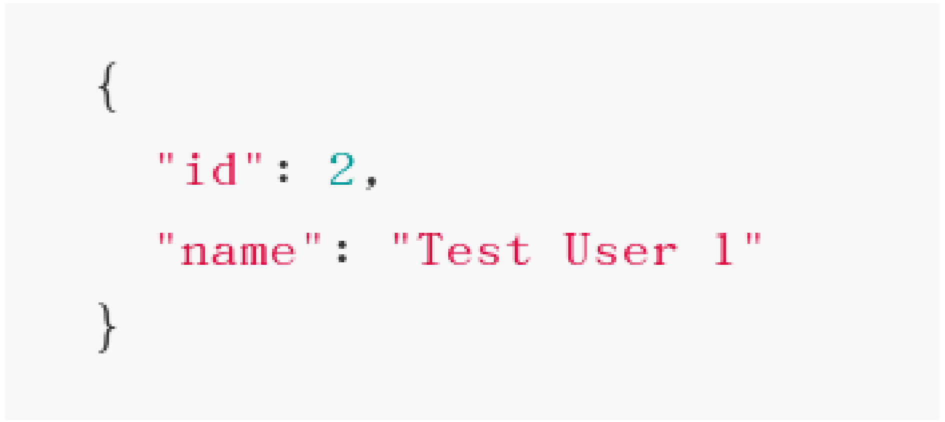
```
{
  "id": 2,
  "name": "Test User"
}
```

Retrofit (4)

- PUT /users/1234 → *updateUser(1234)*



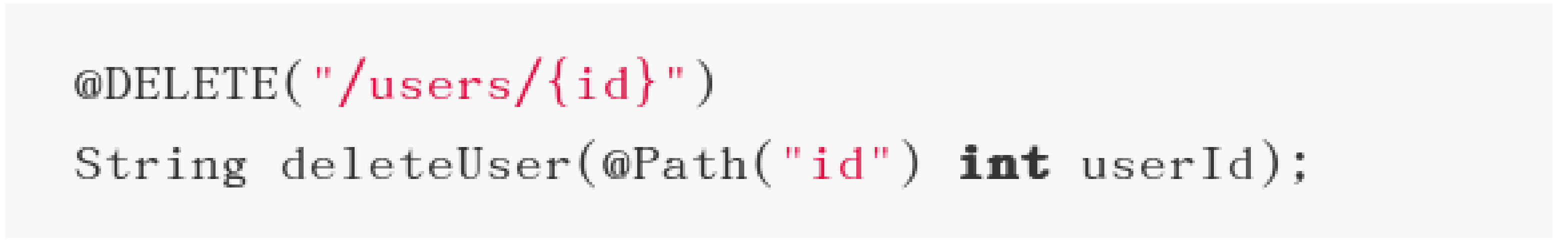
```
@PUT("/users/{id}")
String updateUser(@Path("id") int userId, @Body UserData userData);
```



```
{
  "id": 2,
  "name": "Test User 1"
}
```

Retrofit (5)

- DELETE /users/1234 → *deleteUser(1234)*



```
@DELETE("/users/{id}")
String deleteUser(@Path("id") int userId);
```

Retrofit (6)

- JSON Data

```json
{
  "username" : "Test User",
  "age" : 20
}
```

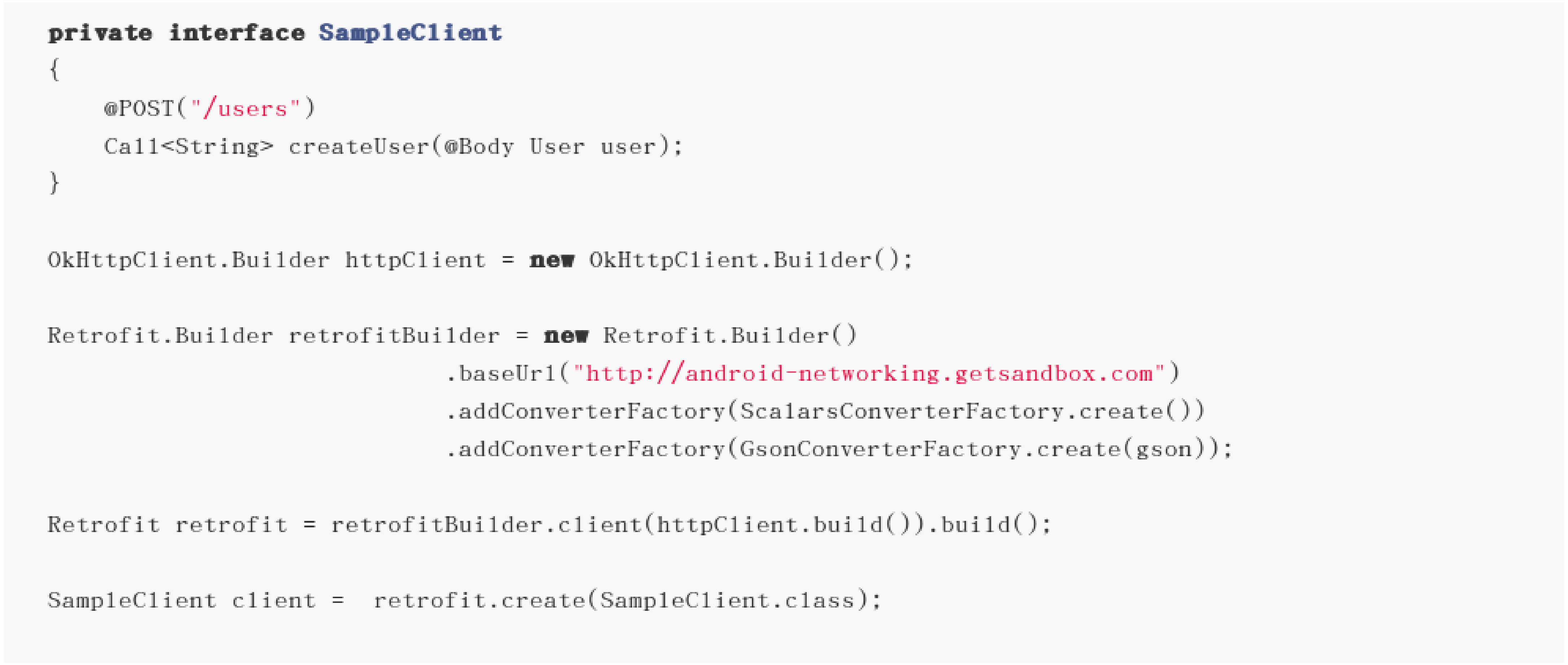
- Java Class

```java
public class UserData
{
  @SerializedName("username")
  String userName;

  @SerializedName("age")
  int userAge;

  public UserData(String name, int age)
  {
    this.userName = name;
    this.userAge = age;
  }
}
```

Retrofit (7)



```
private interface SampleClient
{
    @POST("/users")
    Call<String> createUser(@Body User user);
}

OkHttpClient.Builder httpClient = new OkHttpClient.Builder();

Retrofit.Builder retrofitBuilder = new Retrofit.Builder()
                            .baseUrl("http://android-networking.getsandbox.com")
                            .addConverterFactory(ScalarsConverterFactory.create())
                            .addConverterFactory(GsonConverterFactory.create(gson));

Retrofit retrofit = retrofitBuilder.client(httpClient.build()).build();

SampleClient client =  retrofit.create(SampleClient.class);
```

Retrofit (8)

```
Call<String> call = client.createUser(newUser);

call.enqueue(new Callback<String>()
{
    @Override
    public void onResponse(Call<String> call, Response<String> response)
    {
    }

    @Override
    public void onFailure(Call<String> call, Throwable t)
    {
    }
});
```

Retrofit (9)

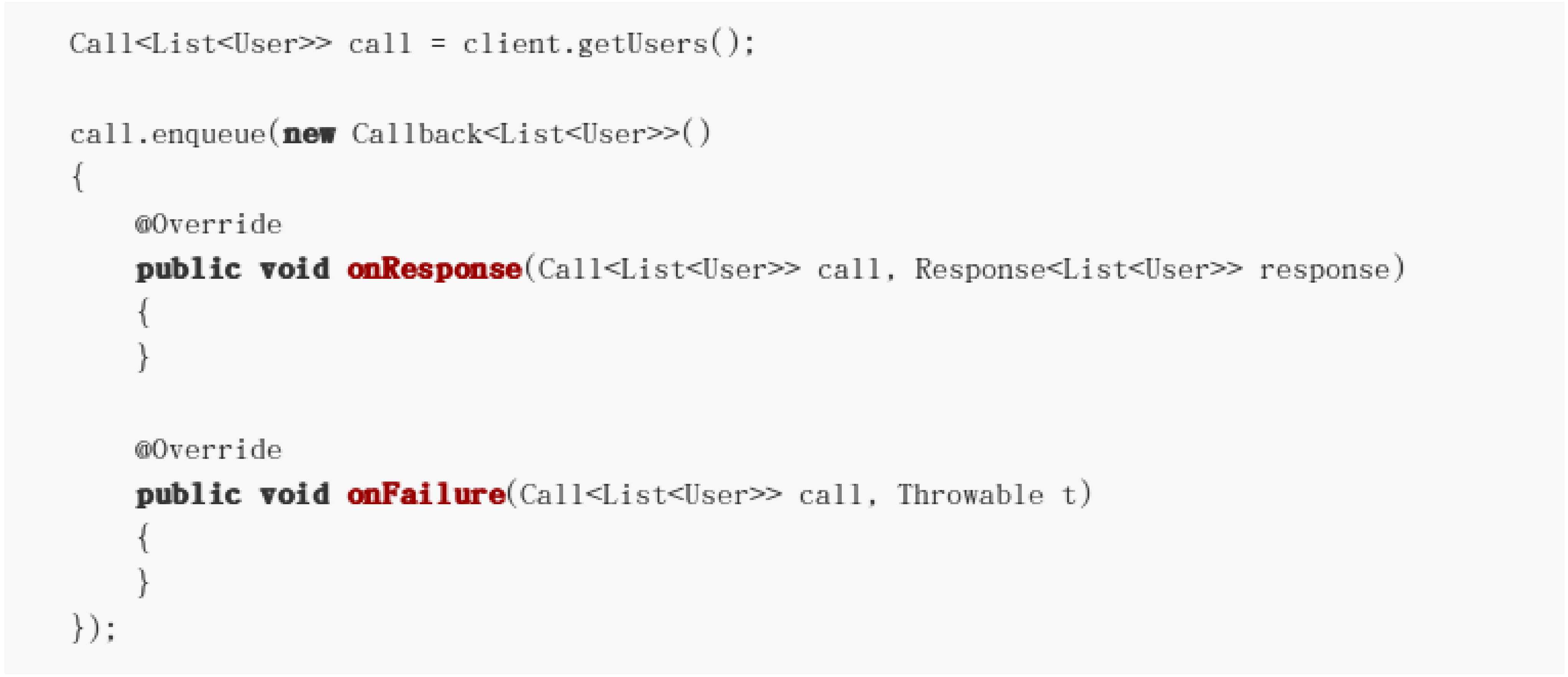
- Sample-Retrofit-1 – Retrofit GET/POST demo

RxJava (1)

- RxJava is a Java VM implementation of Reactive Extensions: a library for composing asynchronous and event-based programs by using observable sequences
- RxAndroid is usually used with RxJava
- Retrolambda is required for Lambda Expressions in Java7

RxJava (2)

- Pure Retrofit



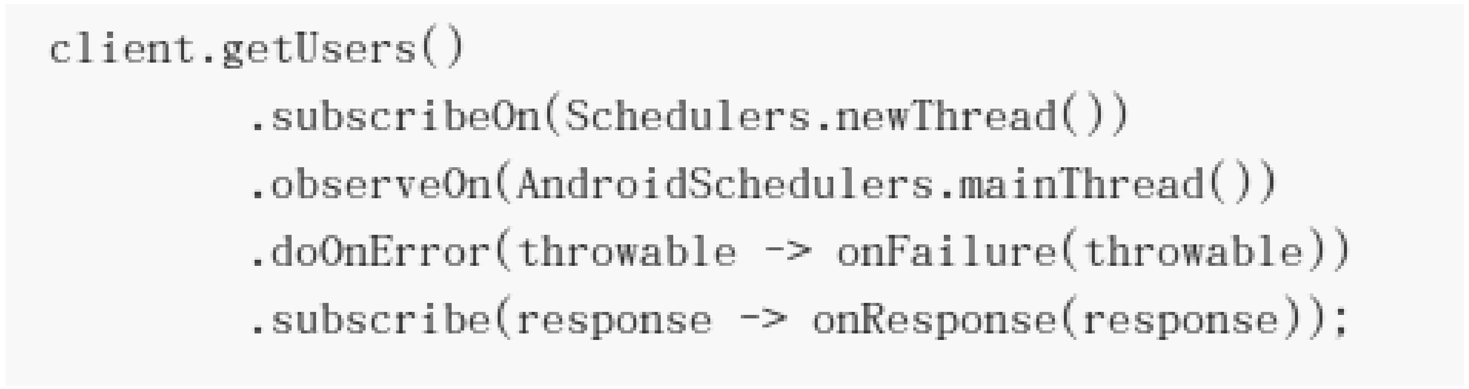
```
Call<List<User>> call = client.getUsers();

call.enqueue(new Callback<List<User>>()
{
    @Override
    public void onResponse(Call<List<User>> call, Response<List<User>> response)
    {
    }

    @Override
    public void onFailure(Call<List<User>> call, Throwable t)
    {
    }
});
```

RxJava (3)

- Retrofit + RxJava + RxAndroid + Retrolambda



```
client.getUsers()
         .subscribeOn(Schedulers.newThread())
         .observeOn(AndroidSchedulers.mainThread())
         .doOnError(throwable -> onFailure(throwable))
         .subscribe(response -> onResponse(response));
```

RxJava (4)

- Sample-RxJava-1 – RxJava+RxAndroid+Retrolambda demo

Questions?

Coursework

- GET https://api.github.com/users/minstrelsy/repos by using Retrofit
- Parse the name of repos and display them on UI
- Using RxJava + RxAndroid + Retrolambda is optional

References

- JSONObject
- JSONArray
- Representational State Transfer (REST)
- Architectural Styles and the Design of Network-based Software Architectures
- URI
- Retrofit
- Retrofit Tutorials
- RxJava
- RxAndroid
- Retrolambda
- How To Use RxJava