Networking

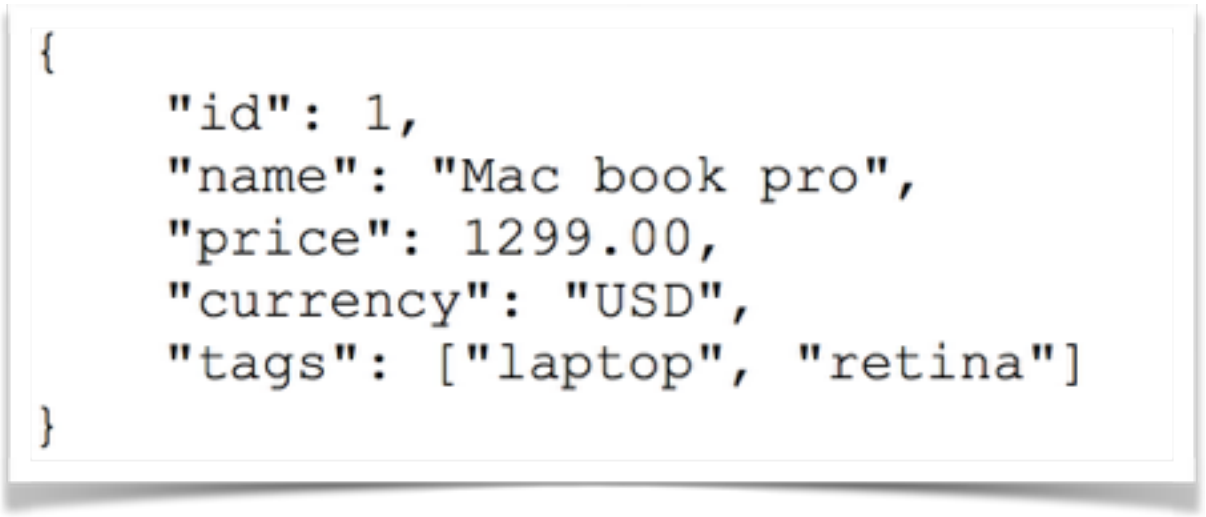
HyperText Transfer Protocol (HTTP)

- Used in client-server model
  - browser - server hosting a website
  - mobile app - REST API or WebService
- Types: GET, POST, PUT, PATCH, DELETE
- Response Codes: 1xx (info), 2xx (OK), 3xx (redirect), 4xx (client error), 5xx (server error)
- http://www.w3.org/Protocols/rfc2616/rfc2616.html

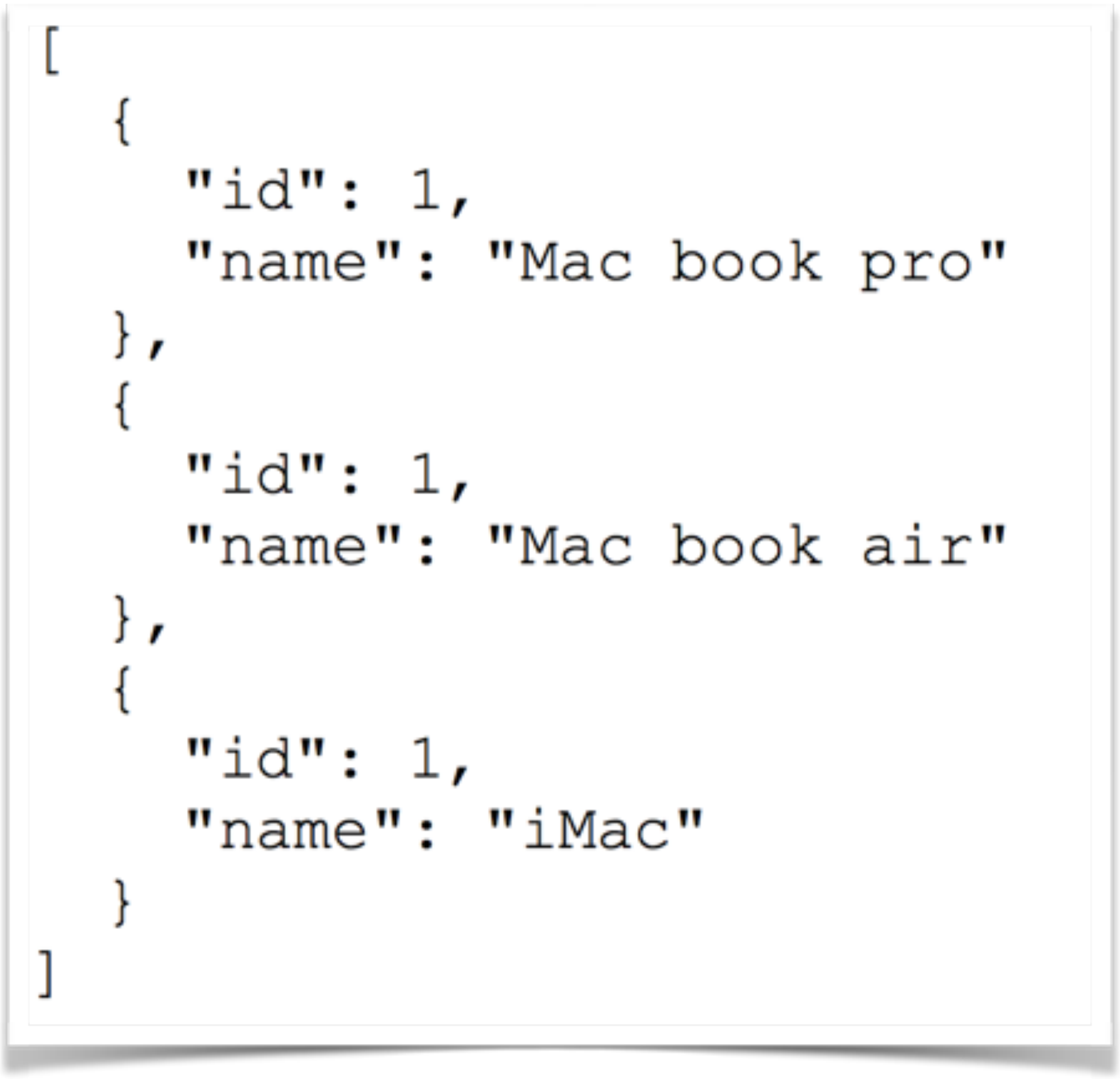
JSON

- JSON - Javascript Object Notation
- lightweight
- 2 structures:
  - object - with key value pairs
  - array - object list
- http://json.org/

JSON



```
{
    "id": 1,
    "name": "Mac book pro",
    "price": 1299.00,
    "currency": "USD",
    "tags": ["laptop", "retina"]
}
```



```
[
  {
    "id": 1,
    "name": "Mac book pro"
  },
  {
    "id": 1,
    "name": "Mac book air"
  },
  {
    "id": 1,
    "name": "iMac"
  }
]
```

AndroidManifest Permissions

```xml
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="co.infinum.networking">
```

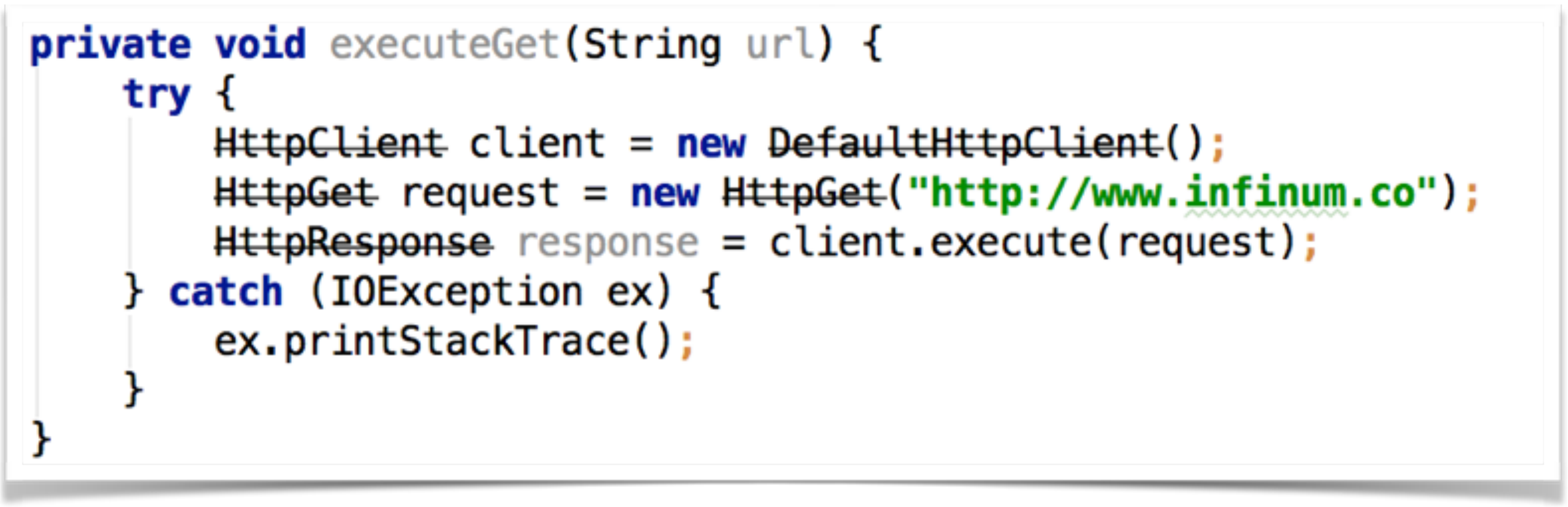
```xml
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
```

<!—other things here—>

```xml
</manifest>
```

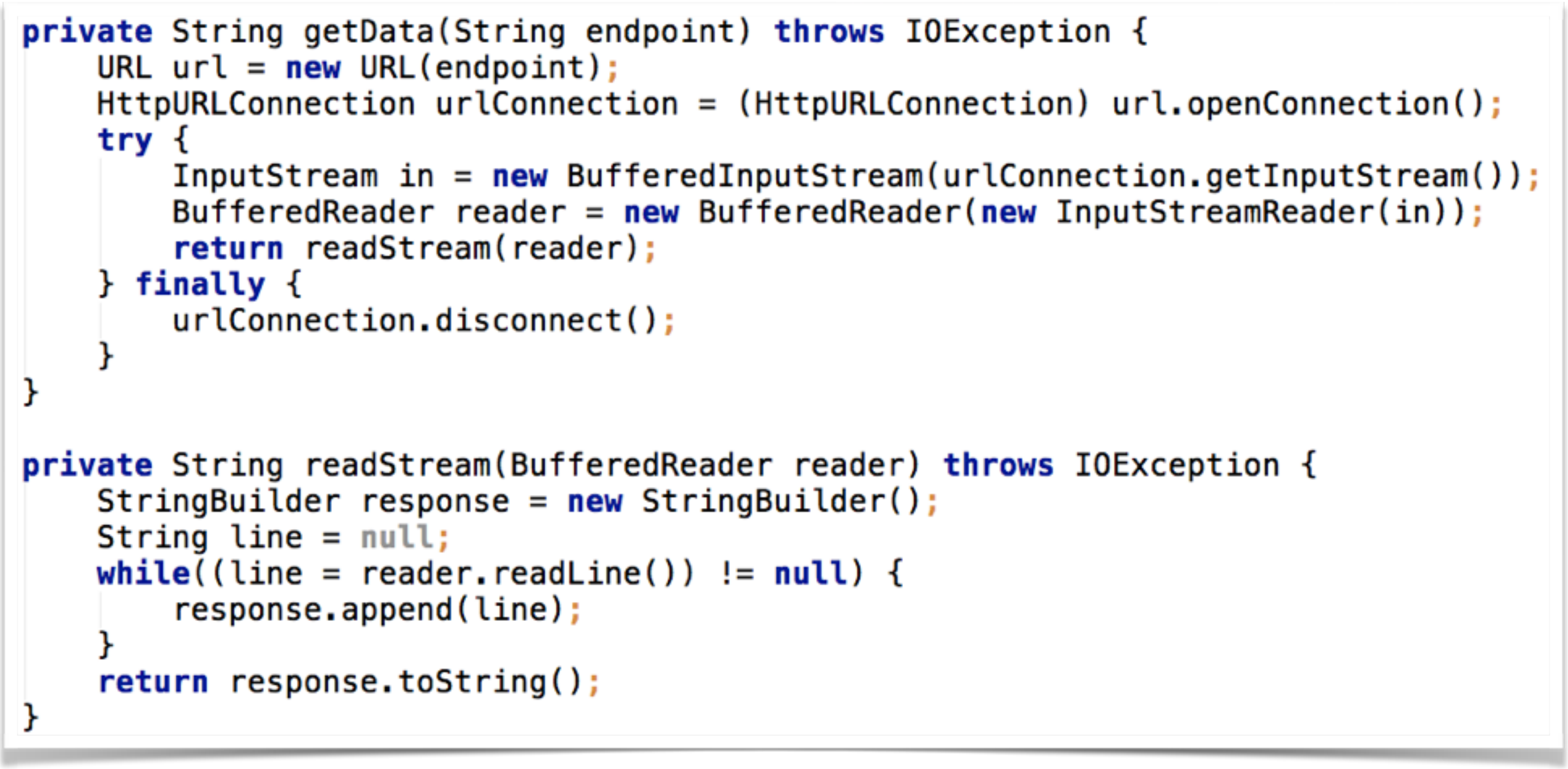
DefaultHttpClient

Deprecated - API 22



```
private void executeGet(String url) {
    try {
        ~~HttpClient~~ client = new ~~DefaultHttpClient~~();
        ~~HttpGet~~ request = new ~~HttpGet~~("http://www.infinum.co");
        ~~HttpResponse~~ response = client.execute(request);
    } catch (IOException ex) {
        ex.printStackTrace();
    }
}
```

UrlConnection



```java
private String getData(String endpoint) throws IOException {
    URL url = new URL(endpoint);
    HttpURLConnection urlConnection = (HttpURLConnection) url.openConnection();
    try {
        InputStream in = new BufferedInputStream(urlConnection.getInputStream());
        BufferedReader reader = new BufferedReader(new InputStreamReader(in));
        return readStream(reader);
    } finally {
        urlConnection.disconnect();
    }
}

private String readStream(BufferedReader reader) throws IOException {
    StringBuilder response = new StringBuilder();
    String line = null;
    while((line = reader.readLine()) != null) {
        response.append(line);
    }
    return response.toString();
}
```

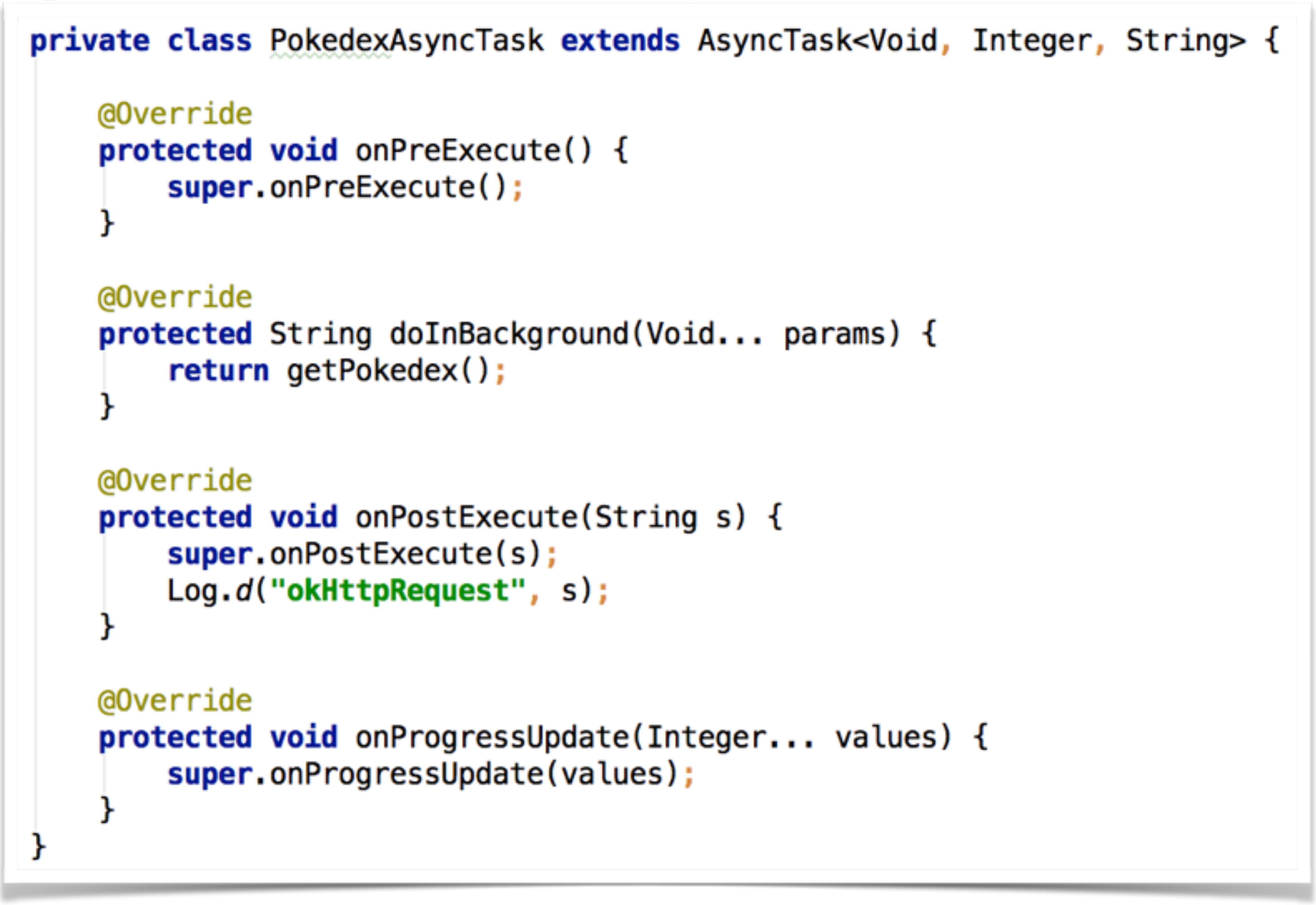
NetworkOnMainThreadException

- API 11 (Android 3.2)
- All networking operations should be in the background thread

AsyncTask

- allows to perform background operations and publish results on the UI thread without having to manipulate threads and/or handlers
- defined by 3 generic types, called Params, Progress and Result, and 4 steps, called onPreExecute, doInBackground, onProgressUpdate and onPostExecute.

AsyncTask



```
private class PokedexAsyncTask extends AsyncTask<Void, Integer, String> {

    @Override
    protected void onPreExecute() {
        super.onPreExecute();
    }

    @Override
    protected String doInBackground(Void... params) {
        return getPokedex();
    }

    @Override
    protected void onPostExecute(String s) {
        super.onPostExecute(s);
        Log.d("okHttpRequest", s);
    }

    @Override
    protected void onProgressUpdate(Integer... values) {
        super.onProgressUpdate(values);
    }
}
```

OkHttp

Don’t reinvent the wheel – part 1

OkHttp

- Setup: build.gradle dependency
    - compile ‘com.squareup.okhttp:okhttp:2.4.0’
    - compile ‘com.squareup.okhttp:okhttp-urlconnection:2.4.0’
    - compile ‘com.squareup.okio:okio:1.5.0'
- 2.0 API is designed with fluent builders and immutability.
- Supports both synchronous blocking calls and async calls with callbacks
- http://square.github.io/okhttp/

Coding session

Gotta catch ‘em all – http://pokeapi.co

Retrofit + OkHttp

Don’t reinvent the wheel – part 2

Retrofit

- Retrofit turns your REST API into a Java interface.
- Every method must have an HTTP annotation that provides the request method and relative URL
- A request URL can be updated dynamically using replacement blocks and parameters on the method
- http://square.github.io/retrofit/

Coding session

Gotta catch em all – refactored

GSON

Mapping JSON strings to data models and vice versa

GSON

- Serialize: Used to convert Java Objects into their JSON representation
- Deserialize: Used to convert a JSON string to an equivalent Java object
- https://sites.google.com/site/gson/gson-user-guide

Coding session

Let’s map our JSON to POJO (Plain Old Java Object)

Glide

load up images

Glide

- Glide supports fetching, decoding, and displaying video stills, images, and animated GIFs
- support for OkHttp
- setup:
    - compile ‘com.github.bumptech.glide:glide:3.6.0’
    - compile 'com.github.bumptech.glide:okhttp-integration:1.3.0'
- https://github.com/bumptech/glide

Coding session

Homework – part 1

https://boatit.infinum.co/api/v1/docs

Napraviti aplikaciju s 2 activity-a : LoginActivity, BoatsActivity

LoginActivity se treba spajati na REST API i ulogirati.

BoatsActivity se treba spajati na REST API i prikazati listu threadova.

Dizajn i resursi su dostupni na https://github.com/InfinumAcademy/android-materijali - boatit.zip

Login podaci:

username: admin@infinum.co

password: infinum1

Homework – part 2

Napraviti details view - dizajn je isto dostupan u prije navedenoj zip datoteci.
Napraviti da aplikacija sacuva podatke kod promjene orijentacije bez da ponovno dohvaca podatke sa REST API-a (na svakom screenu).