OTUS

Android Developer

Networking

otus.ru

Проверить, идет ли запись

```
if (видно && слышно) {
    chat.print(“+”)
}
```

Ставим “+”, если все хорошо
“-”, если есть проблемы

Тема вебинара

Networking

Николай Кочетков

Руководитель Андроид разработки FlyXO

Об опыте (например):

22 года в IT, 7 лет - играющий тренер Андроид

Телефон / эл. почта / соц. сети:
LinkedIn: https://www.linkedin.com/in/motorro/
GitHub: https://github.com/motorro/
Medium: https://medium.com/@motorro

Правила вебинара

Активно
участвуем

Off-topic обсуждаем
в Slack #**канал группы**
или #general

Задаем вопрос
в чат или голосом

Вопросы вижу в чате,
могу ответить не сразу

Условные обозначения

Индивидуально

Время, необходимое
на активность

Пишем в чат

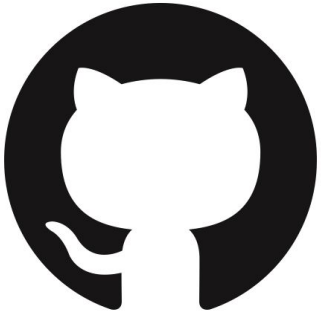
Говорим голосом

Документ

Ответьте себе или
задайте вопрос

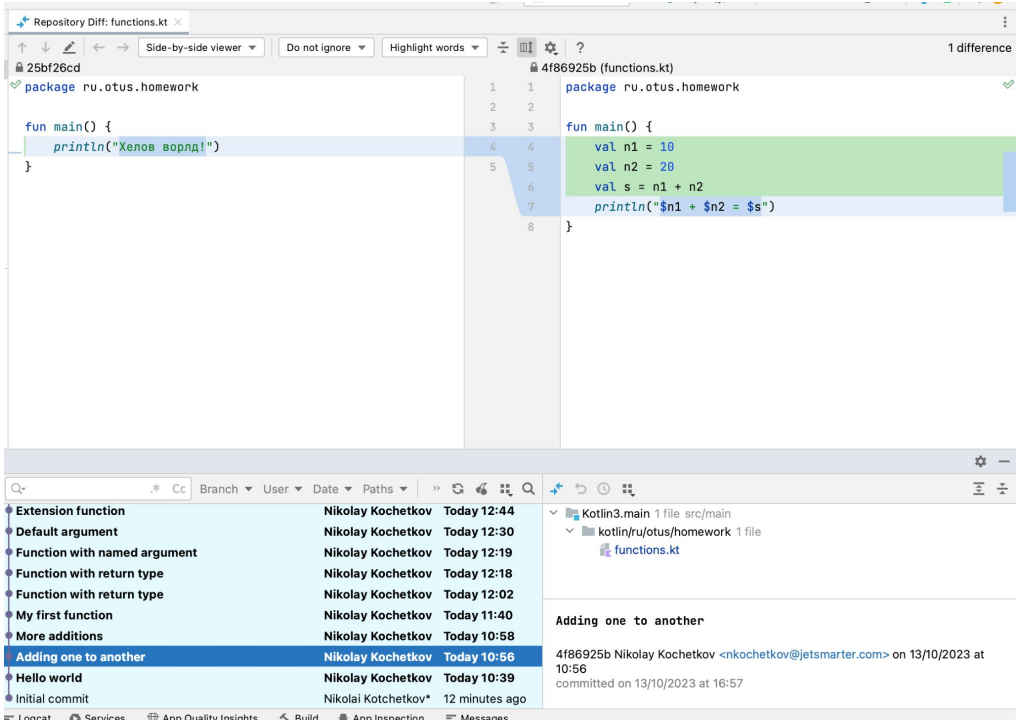
Маршрут вебинара

Репозиторий к занятию

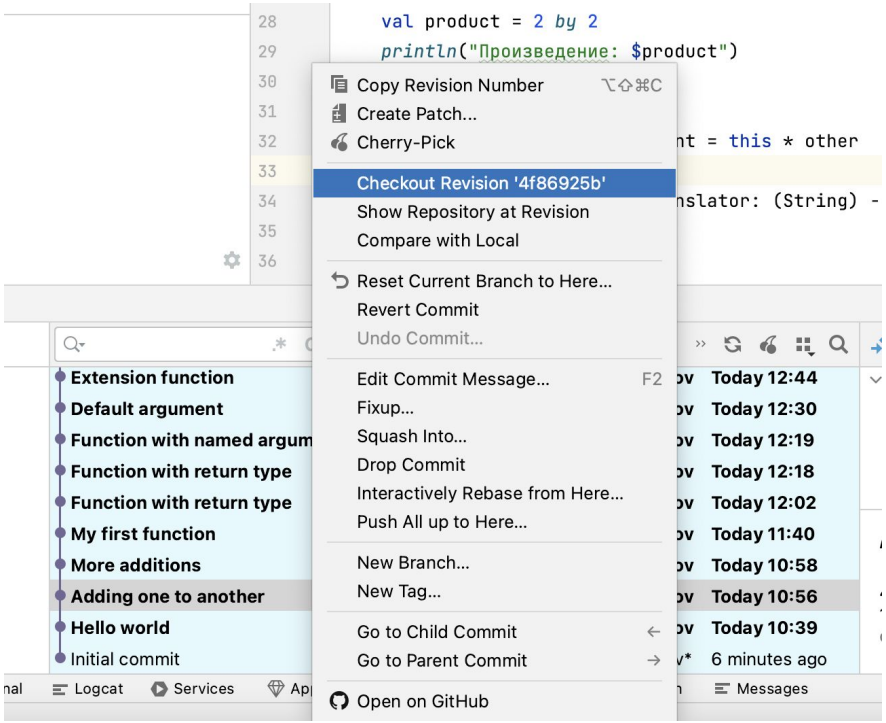


https://github.com/Android-Developer-Basic/Networking

Репозиторий к занятию - по шагам



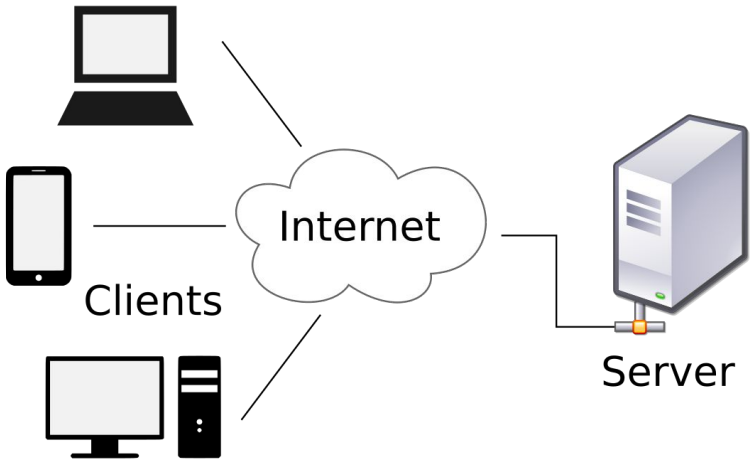
Репозиторий к занятию - по шагам



Client-server architecture

Вспомним...

Архитектура



Архитектура

- Распределенное приложение
- Безопасное исполнение бизнес-логики на сервере
- Безопасное хранение данных на сервере
- Нет ограничения на объем памяти и производительность клиента

Обмен данными сервер - клиент

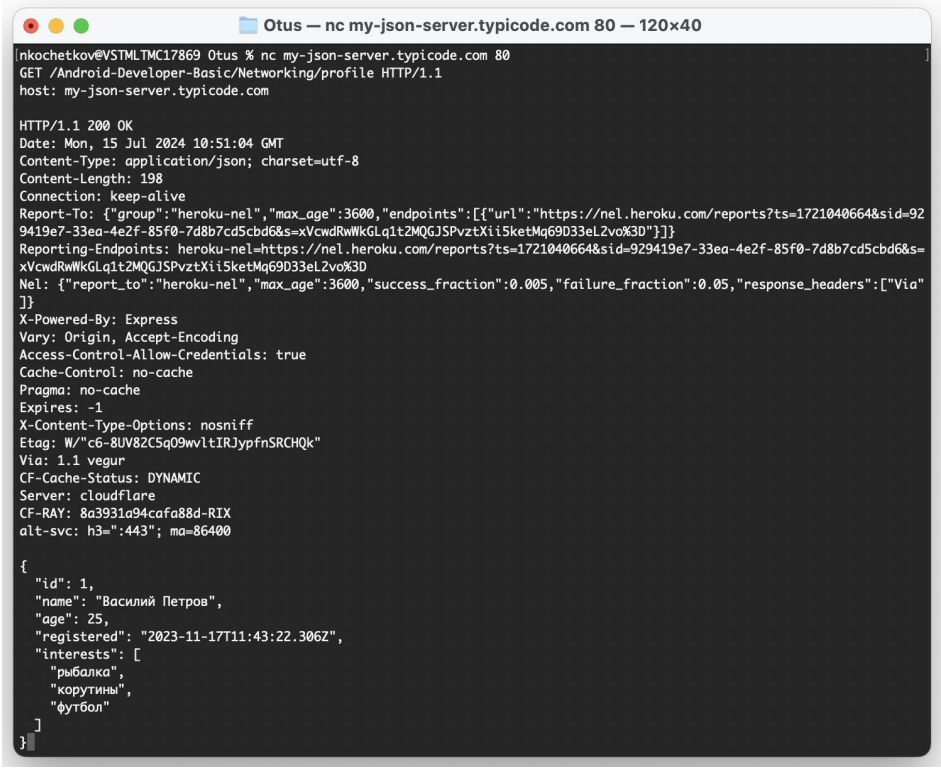
- Сетевой стек TCP/IP
- Прикладной уровень HTTP
- Формат обмена данных JSON

Обмен данными сервер - клиент

- Сетевой стек TCP/IP
- Прикладной уровень HTTP
- Формат обмена данных JSON

HTTP

1. Соединение с сервером
2. Запрос ресурса
3. Статус ответа
4. Заголовки ответа
5. Тело ответа

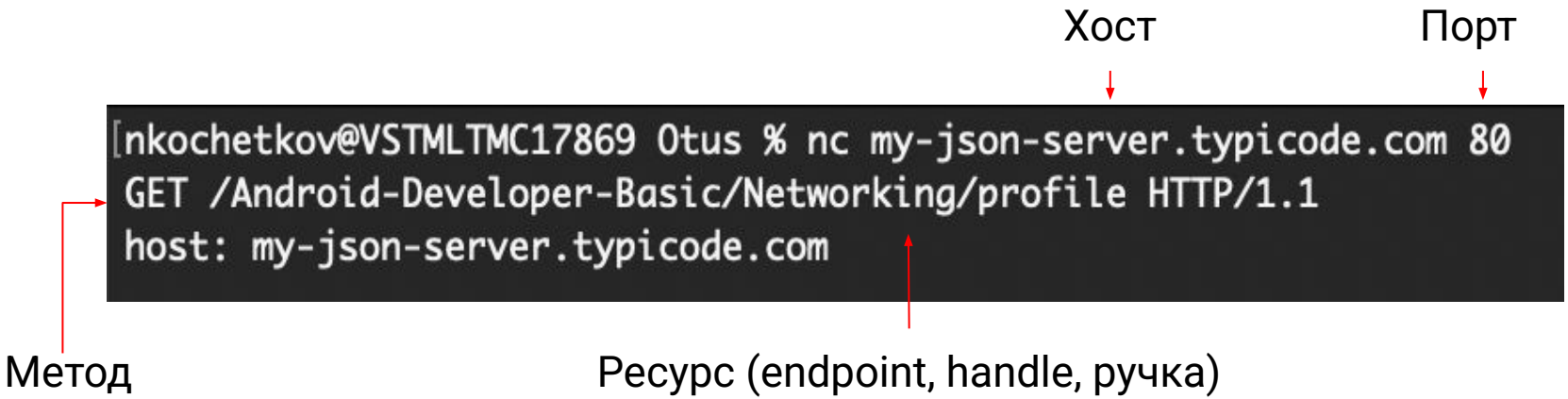


URL

```
<схема>:[//[<логин>[:<пароль>]@]<хост>[:<порт>]][/<URL - путь>][?<параметры>][#<якорь>]
```

https://user:password@myserver.com:8080/profile?id=123#name

Соединение


Хост
Порт
nkochetkov@VSTMLTMC17869 Otus % nc my-json-server.typicode.com 80
GET /Android-Developer-Basic/Networking/profile HTTP/1.1
host: my-json-server.typicode.com
Метод
Ресурс (endpoint, handle, ручка)

HTTP метод

Метод - последовательность ЛЮБЫХ* символов, определяющих действие с ресурсом
Стандартные методы:

- **GET** - получить ресурс
- **POST** - создать ресурс
- **PUT** - изменить ресурс
- **DELETE** - удалить ресурс

HTTP статус (код состояния)

1xx - информационный
2xx - запрос успешно обработан
3xx - перенаправление (redirect)
4xx - ошибка со стороны клиента
5xx - ошибка со стороны сервера

HTTP заголовки

- Позволяют клиенту и серверу обмениваться дополнительной информацией
- Заголовки могут быть стандартными и пользовательскими

HTTP заголовки

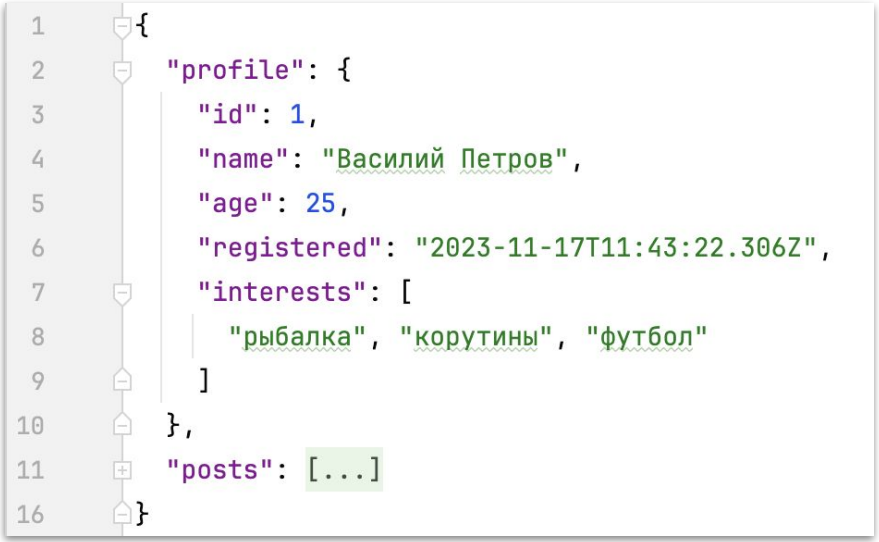
Запрос клиента:

```
GET /wiki/страница HTTP/1.1
Host: ru.wikipedia.org
User-Agent: Mozilla/5.0 (X11; U; Linux i686; ru; rv:1.9b5) Gecko/2008050509 Firefox/3.0b5
Accept: text/html
Connection: close
```

Ответ сервера:

```
HTTP/1.1 200 OK
Date: Wed, 11 Feb 2009 11:20:59 GMT
Server: Apache
X-Powered-By: PHP/5.2.4-2ubuntu5wm1
Last-Modified: Wed, 11 Feb 2009 11:20:59 GMT
Content-Language: ru
Content-Type: text/html; charset=utf-8
Content-Length: 1234
Connection: close
```

JSON



```
{
  "profile": {
    "id": 1,
    "name": "Василий Петров",
    "age": 25,
    "registered": "2023-11-17T11:43:22.306Z",
    "interests": [
      "рыбалка", "корутины", "футбол"
    ]
  },
  "posts": [...]
}
```

- JSON - JavaScript object notation
- Текстовый формат обмена данными
- Человекочитаемый
- Лаконичный

Вопросы?

Сериализация
Преобразование JSON-DTO

JSON-DTO (Data Transfer Object)

```json
{
  "profile": {
    "id": 1,
    "name": "Василий Петров",
    "age": 25,
    "registered": "2023-11-17T11:43:22.306Z",
    "interests": [
      "рыбалка", "корутины", "футбол"
    ]
  },
  "posts": [...]
}
```

JSON

```kotlin
@Serializable
data class Profile(
    @SerialName("id")
    val userId: Int,
    val name: String,
    val age: Int,
    val registered: Instant,
    val interests: List<String> = emptyList()
)
```

DTO

JSON-DTO (Data Transfer Object)

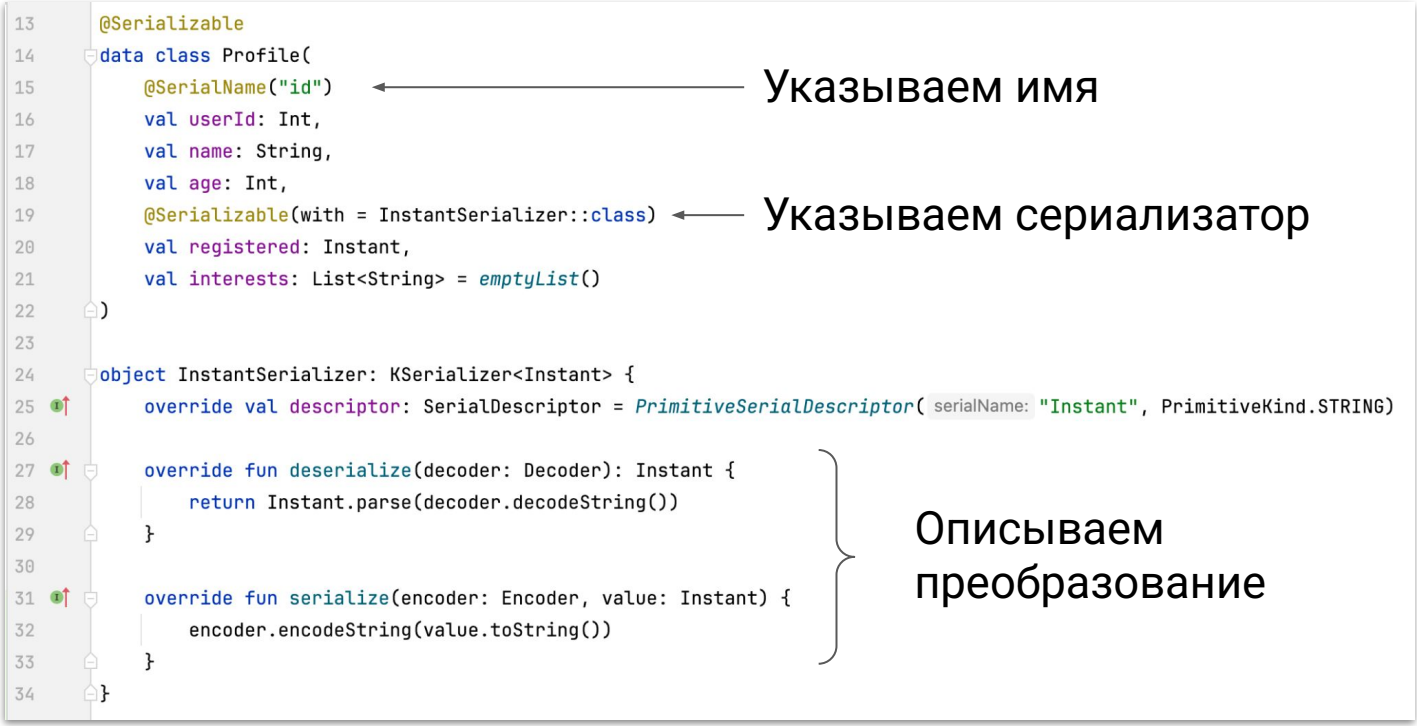
```
@Test
fun serializesProfile() {
    assertEquals(
        expected: """{"id":1,"name":"Vasya","age":25,"registered":"2023-11-17T11:43:22.306Z","interests":["рыбалка","корутины","футбол"]}""",
        Json.encodeToString(Profile.serializer(), profile)
    )
}

@Test
fun deserializesProfile() {
    assertEquals(
        profile,
        Json.decodeFromString(
            string: """{"id":1,"name":"Vasya","age":25,"registered":"2023-11-17T11:43:22.306Z","interests":["рыбалка","корутины","футбол"]}"""
        )
    )
}
```

Библиотеки конвертации

- GSON
- Moshi
- Kotlin Serialization

JSON-DTO (Data Transfer Object)



Изменение параметров преобразования

Вопросы?

OkHTTP

Реализация HTTP протокола

OkHTTP

https://square.github.io/okhttp/

OkHTTP

- Реализация протокола HTTP
- Управление TCP/IP
- Гибкая поддержка транспорта
- Поддержка TLS, certificate pinning
- Кэширование запросов в соответствии с заголовками от сервера

OkHTTP - создаем клиент

```
43  @Module
44  @InstallIn(ViewModelComponent::class)
45  class MainModuleProvider {
46      @Provides
47      fun okHttp(): OkHttpClient = OkHttpClient.Builder().build()
48  }
```

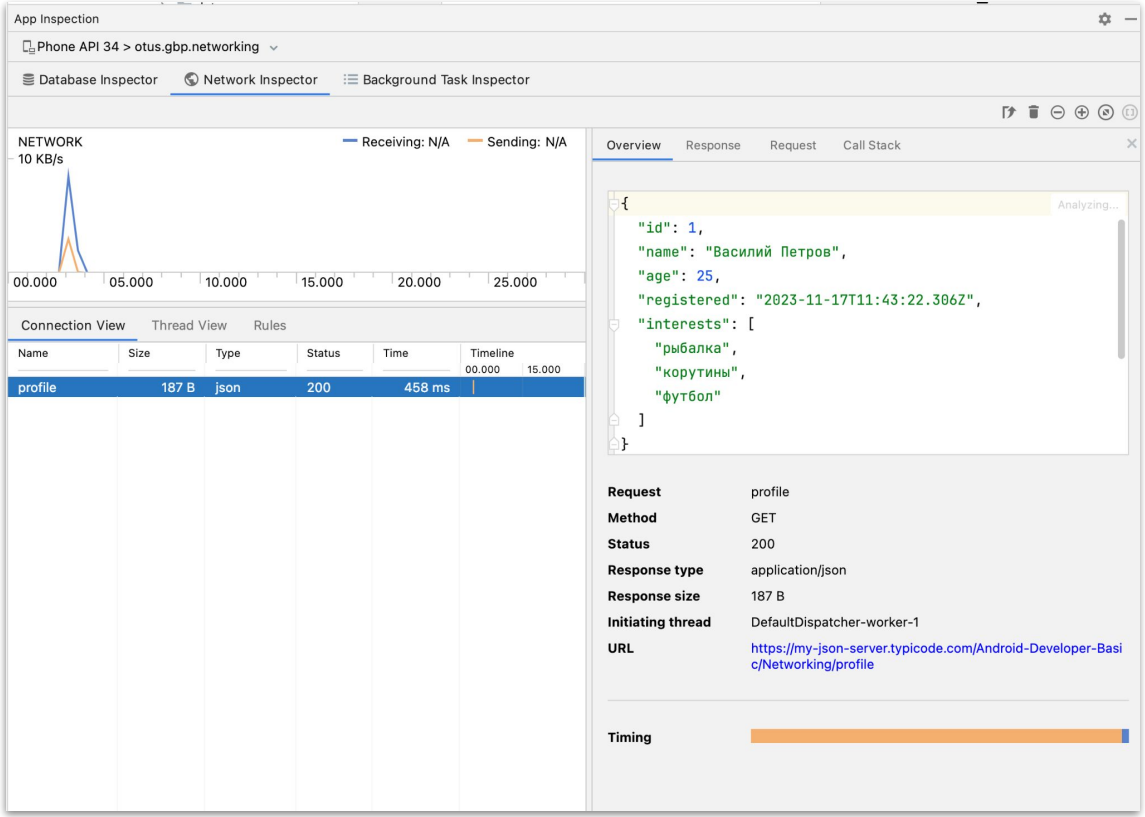
OkHTTP - идем в сеть напрямую

Создаем запрос

Вызываем синхронно

```
15  class Impl @Inject constructor(private val okHttpClient: OkHttpClient) : GetProfile {
16      override suspend fun invoke(): Profile = withContext(Dispatchers.IO) {
17          val request = Request.Builder()
18              .url("https://my-json-server.typicode.com/Android-Developer-Basic/Networking/profile")
19              .get()
20              .build()
21
22          okHttpClient.newCall(request).execute().let { response ->
23              if (!response.isSuccessful) throw IOException("Unexpected code $response")
24              val body = response.body ?: throw IOException("Empty body $response")
25              Json.decodeFromString(Profile.serializer(), body.string())
26          }
27      }
28  }
```

OkHTTP - идем в сеть напрямую



OkHTTP - настройка

```
44  @Module
45  @InstallIn(ViewModelComponent::class)
46  class MainModuleProvider {
47      @Provides
48      fun okHttp(): OkHttpClient = OkHttpClient.Builder()
49          .callTimeout( timeout: 30, TimeUnit.SECONDS)
50          .build()
51  }
```

OkHTTP - Interceptor

Обрабатывает все запросы

```
47  @Module
48  @InstallIn(ViewModelComponent::class)
49  class MainModuleProvider {
50      @Provides
51      fun okHttp(): OkHttpClient = OkHttpClient.Builder()
52          .callTimeout( timeout: 30, TimeUnit.SECONDS)
53          .addInterceptor(HttpLoggingInterceptor().apply {
54              setLevel(HttpLoggingInterceptor.Level.BASIC)
55          })
56          .build()
57  }
```

Логирование сетевых запросов

OkHTTP - Interceptor

```
19:15:14.798  D  app_time_stats: avg=452.53ms min=0.98ms max=4476.86ms count=10
19:15:14.880  I  --> GET https://my-json-server.typicode.com/Android-Developer-Basic/Networking/profile
19:15:14.886  D  tagSocket(104) with statsTag=0xffffffff, statsUid=-1
19:15:15.538  I  <-- 200 https://my-json-server.typicode.com/Android-Developer-Basic/Networking/profile (657ms, unknown-length body)
19:15:15.969  D  Installing profile for otus.gbp.networking
```

Логирование сетевых запросов

OkHTTP - Interceptor

```
9  class AuthInterceptor @Inject constructor(private val sessionManager: SessionManager): Interceptor {
10     override fun intercept(chain: Interceptor.Chain): Response {
11         val request = chain.request()
12         val requestWithAuth = request.newBuilder()
13             .header( name: "Authorization",  value: "Bearer: ${sessionManager.getToken()}")
14             .build()
15
16         return chain.proceed(requestWithAuth)
17     }
18 }
```

Аутентификация всех запросов

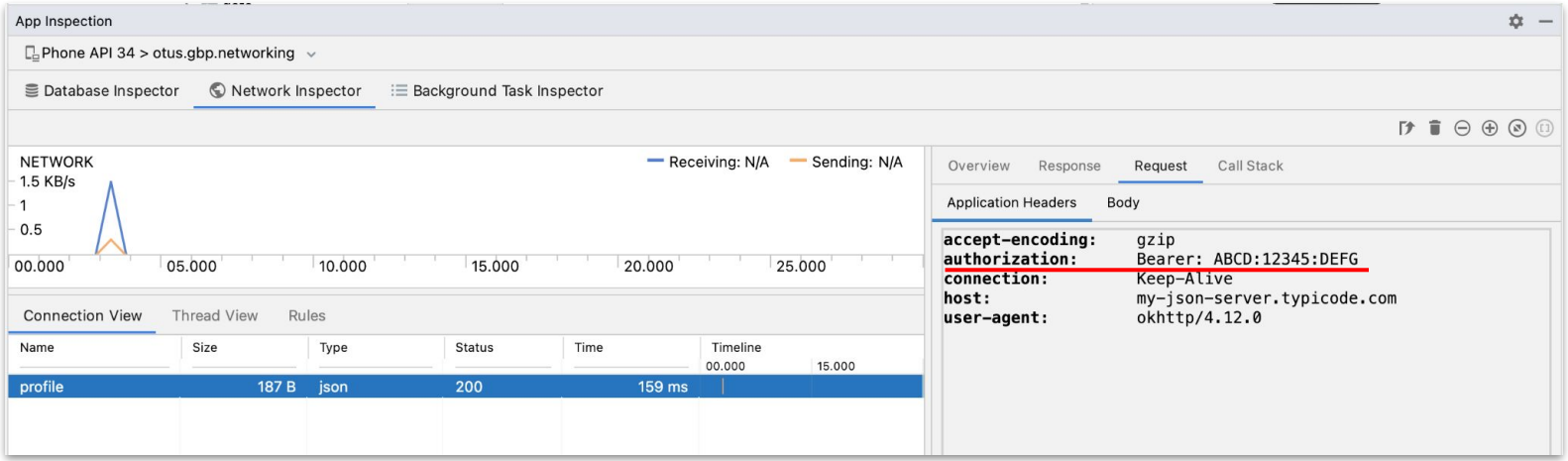
OkHTTP - Interceptor

Подключаем

```
50  @Module
51  @InstallIn(ViewModelComponent::class)
52  class MainModuleProvider {
53      @Provides
54      fun okHttp(authInterceptor: AuthInterceptor): OkHttpClient = OkHttpClient.Builder()
55          .callTimeout( timeout: 30, TimeUnit.SECONDS)
56          .addInterceptor(authInterceptor)
57          .addInterceptor(HttpLoggingInterceptor().apply {
58              setLevel(HttpLoggingInterceptor.Level.BASIC)
59          })
60          .build()
61  }
```

Аутентификация всех запросов

OkHTTP - Interceptor



Аутентификация всех запросов

Вопросы?

Retrofit

Автоматизация построения запроса

Retrofit

https://square.github.io/retrofit/

Retrofit

- Избавляет от написания однообразного кода
- Декларативное построение API описанием интерфейса
- Сериализация

Retrofit service

```
14  interface Api {
15      @GET("profile")
16      suspend fun getProfile(): Response<Profile>
17  }
```

Метод запроса

Retrofit builder

```
19  fun buildRetrofit(okHttpClient: OkHttpClient): Retrofit = Retrofit.Builder()
20      .baseUrl(baseUrl)
21      .client(okHttpClient)
22      .addConverterFactory(Json.asConverterFactory("application/json".toMediaType()))
23      .build()
```

Базовый путь

Конвертер

Retrofit API creator

```
@Provides
fun api(retrofit: Retrofit): Api = retrofit.create(Api::class.java)
```

Retrofit используем

```
15    class Impl @Inject constructor(private val api: Api) : GetProfile {
16        override suspend fun invoke(): Profile {
17            val response = api.getProfile()
18            if (!response.isSuccessful) {
19                throw IOException("Unexpected code $response")
20            }
21            return response.body() ?: throw IOException("Empty body $response")
22        }
23    }
```


Тело ответа уже нужного класса

Retrofit параметры

```
@GET("profile")
suspend fun getProfile(@Query("id") id: Int): Response<Profile>
```


Query параметр

Retrofit параметры

```
@POST("profile")
suspend fun setProfile(@Body profile: Profile): Response<Profile>
```


Передача данных на сервер

Вопросы?

Рефлексия

Тестовый проект

https://github.com/Android-Developer-Basic/Networking

Маршрут вебинара

Рефлексия

С какими впечатлениями уходите с вебинара?

Как будете применять на практике то,
что узнали на вебинаре?

Заполните, пожалуйста, опрос о занятии по ссылке в чате

Спасибо за внимание!

Приходите на следующие вебинары

Николай Кочетков

Руководитель Андроид разработки FlyXO

Об опыте (например):

22 года в IT, 7 лет - играющий тренер Андроид

Телефон / эл. почта / соц. сети:
LinkedIn: https://www.linkedin.com/in/motorro/
GitHub: https://github.com/motorro/
Medium: https://medium.com/@motorro