

ЛЕКЦИЯ I © 2021 Нет Ит

Android: Retrofit

Теодор Костадинов



SOFTWARE
ACADEMY



Съдържание

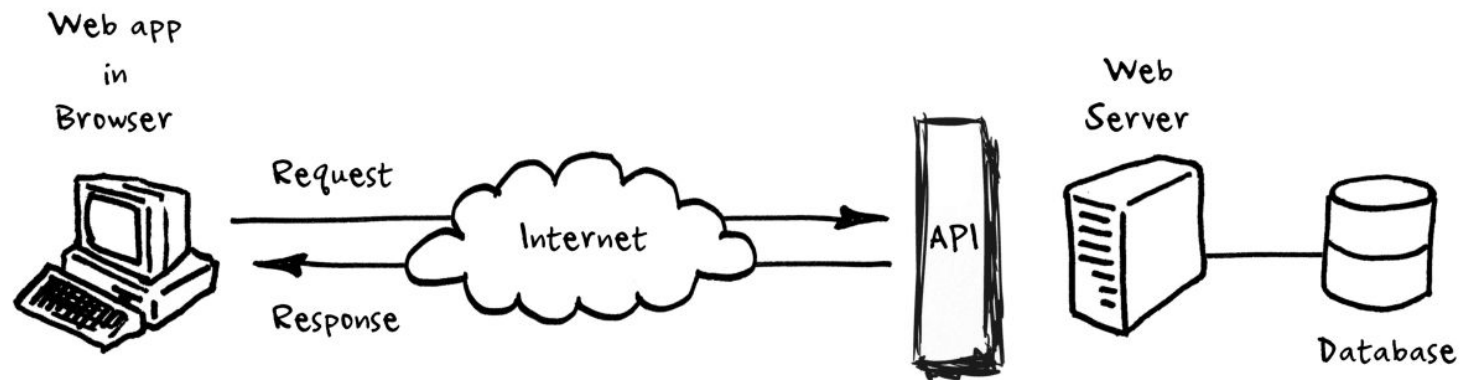
1. Получаване на данни от интернет
2. Какво е Retrofit и как да го използваме?
3. JSON и GSON
4. Предимствата на Postman
5. Домашно



APIs

Какво е API?

- Application Program Interface (API) е публичен интерфейс, обикновено към приложение, работещо на сървър (back-end / бекенд).
- Клиентите (уебсайтове, мобилни апп-ове, умни хладилници и т.н.) могат да правят заявки към този интерфейс.
- Бекенд-а получава заявките, извършва работа (взима данни от базата, генерира картинки и т.н.) и връща отговор към клиента.
- Една заявка се нарича request или API call.



Видове заявки

- GET: цялата информация на заявката е закодирана в нейния URL.

Пример: <https://opentdb.com/api.php?amount=1&category=22&type=multiple>

- Правим заявка към opentdb.com/api.php
- Заявката има три параметъра: amount, category и type. Всеки параметър има зададена стойност.
- POST: в URL-а се задава само към кой метод от интерфейса се обръщаме. Данните се предават като байтове, скрито. POST е по-сигурният метод за комуникация.

Пример: <https://deckofcardsapi.com/api/deck/new/shuffle/>

- Правим заявка към горното URL
- Към заявката прикачваме обект, съдържащ данните.

```
Deck.class
```

```
int deck_count = 1;
```




POSTMAN

Какво е Postman?

- Използва се, когато искаш да тестваме дали дадена заявка връща очакваната информация
- Тази информация след това ни се визуализира
- Лесно можем да разберем дали заявките, които пращаме, са грешни, или всъщност има проблем с кода

Кратък tutorial:

<https://www.guru99.com/postman-tutorial.html>



CONNECTIVITY MANAGER

Старият начин за връзка с интернет

```
private static boolean wifiConnected = false;
private static boolean mobileConnected = false;

ConnectivityManager cm =
    (ConnectivityManager)context.getSystemService(Context.CONNECTIVITY_SERVICE);
NetworkInfo activeInfo = connMgr.getActiveNetworkInfo();

if (activeInfo != null && activeInfo.isConnected()) {
    wifiConnected = activeInfo.getType() == ConnectivityManager.TYPE_WIFI;
    mobileConnected = activeInfo.getType() == ConnectivityManager.TYPE_MOBILE;
}
```

Старият начин

```
URL url = new URL("http://www.android.com/");
URLConnection urlConnection = (URLConnection) url.openConnection();
urlConnection.setReadTimeout(10000);
urlConnection.setConnectTimeout(10000);
urlConnection.setRequestMethod("GET");
urlConnection.setDoInput(true);

try {
    InputStream in = new BufferedInputStream(urlConnection.getInputStream());
    readStream(in);
} finally {
    urlConnection.disconnect();
}
```


Има и по-добри варианти

Вместо да правим всичко сами, има библиотеки, които могат да свършат работата за нас и да ни спестят много работа:

- идват с преконфигурации, които улесняват работата
- автоматично могат да parse-ват получената информация
- могат да работят както синхронно, така и асинхронно

	One Discussion	Dashboard (7 requests)	25 Discussions
AsyncTask	941 ms	4,539 ms	13,957 ms
Volley	560 ms	2,202 ms	4,275 ms
Retrofit	312 ms	889 ms	1,059 ms



RETROFIT

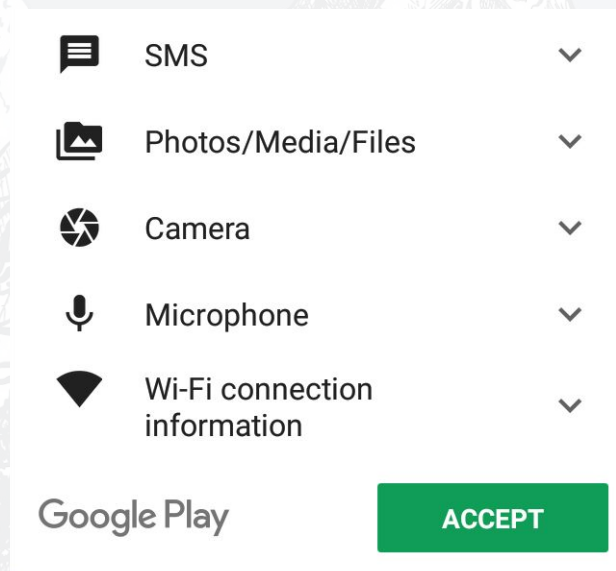
Преди да започнем:

За да можем да изпълняваме операции, свързани с Интернет, е нужно да включим следните permission-и в нашия **manifest** файл:

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

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

Когато потребителите
инсталират приложението, ще
бъдат питани за разрешение за
достъп до изисканата
информация



Какво представлява Retrofit?

- Retrofit е **HTTP клиент**, който има широка употреба за андроид приложения.
- За да можем да го използваме, трябва да включим следния ред в **dependency**-тата ни в **build.gradle** файла :

```
implementation 'com.squareup.retrofit2:retrofit:2.9.0'
```

- Трябва и да включим използването на Java версия 1.8, което става в **android** модула в **build.gradle** файла :

```
compileOptions {  
    sourceCompatibility JavaVersion.VERSION_1_8  
    targetCompatibility JavaVersion.VERSION_1_8  
}
```


#1 Create interface

Retrofit превръща нашето HTTP API в удобен за използване интерфейс

Ще замести в "users/{user}/repos" със String u

```
public interface GitHubService {  
    @GET("users/{user}/repos")  
    Call<List<Repo>> listRepos(@Path("user") String u);  
    @GET("group/{id}/users")  
    Call<List<User>> groupList(@Path("id") int groupId,  
    @Query("sort") String sort);  
}
```

Видът на заявката, която ще искаме да се изпълни

Може да имаме и query-та

#2 Create access objects

Към кой адрес да
бъдат изпращани
заявките

```
Retrofit retrofit = new Retrofit.Builder()  
    .baseUrl("https://api.github.com/")  
    .build();
```

```
GitHubService service = retrofit.create(GitHubService.class);
```

```
Call<List<Repo>> repos = service.listRepos("octocat");
```

Потребителското име, което
ще бъде заместено в
заявката

#3 Make and receive requests

```
service.listRepos("octocat", new Callback<Repo>() {
```

```
    @Override
```

```
    public void onResponse(Response<Repo> response, Retrofit retrofit) {
```

```
        Repo repo = response.body();
```

```
    }
```

```
    @Override
```

```
    public void onFailure(Throwable throwable) {
```

```
    }
```

```
}).execute();
```

Какво да стане, ако ни се върне резултат

Какво да правим, ако има грешка и не можем да изпълним заявката



DEMO

Почивка
до 20:20





JSON и GSON

Какво представлява JSON?

- Формат за пренос на данни
- Използва се най-вече за изпращане и получаване на данни по мрежата от сървър
- Повечето API-та ще връщат именно този формат
- Лесен за разбиране както от хората, така и от машините
- Поддържа следните типове данни:

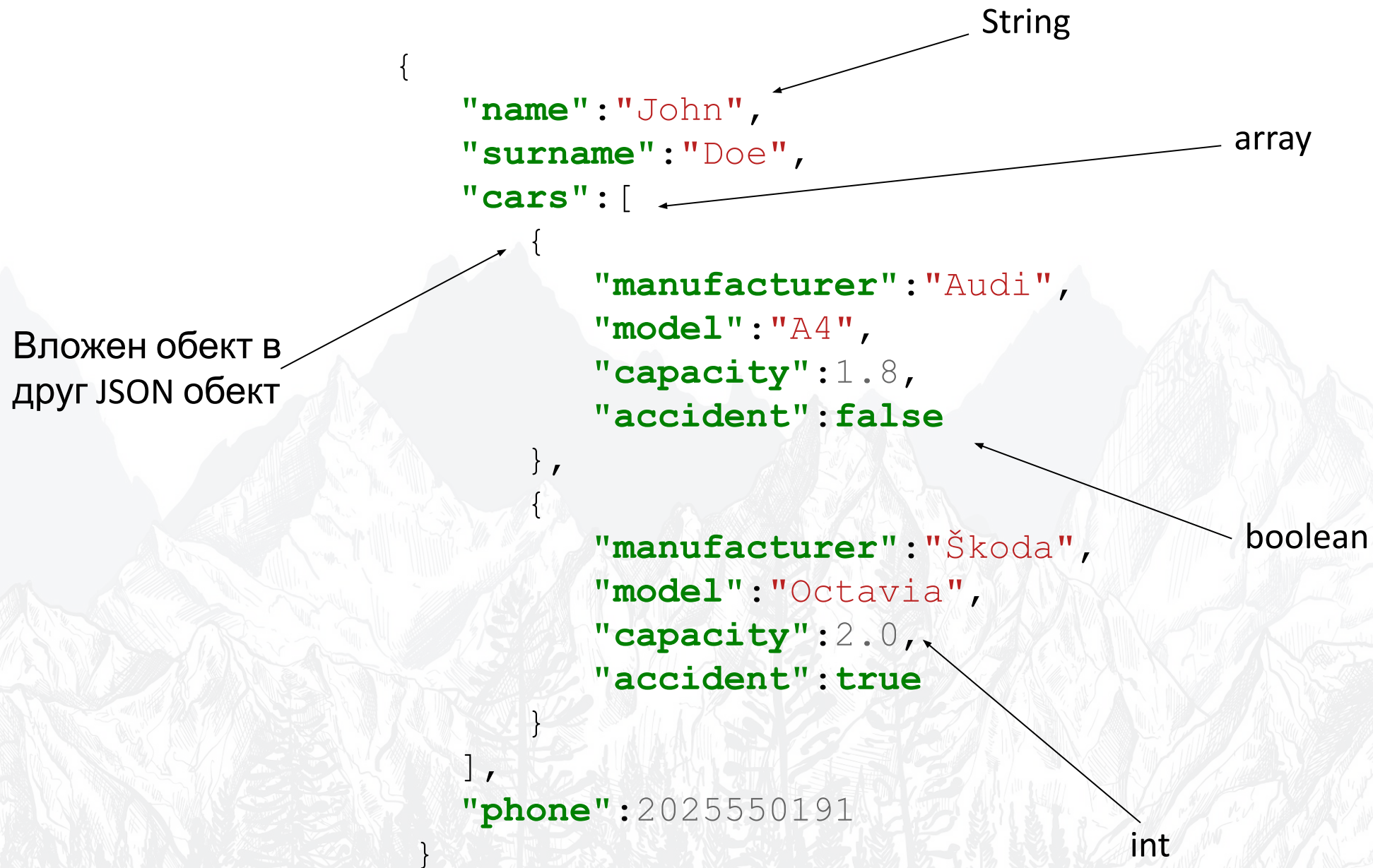
int

String

boolean

array

null



```
{
  "name": "John",
  "surname": "Doe",
  "cars": [
    {
      "manufacturer": "Audi",
      "model": "A4",
      "capacity": 1.8,
      "accident": false
    },
    {
      "manufacturer": "Škoda",
      "model": "Octavia",
      "capacity": 2.0,
      "accident": true
    }
  ],
  "phone": 2025550191
}
```

String

array

Вложен объект в
друг JSON объект

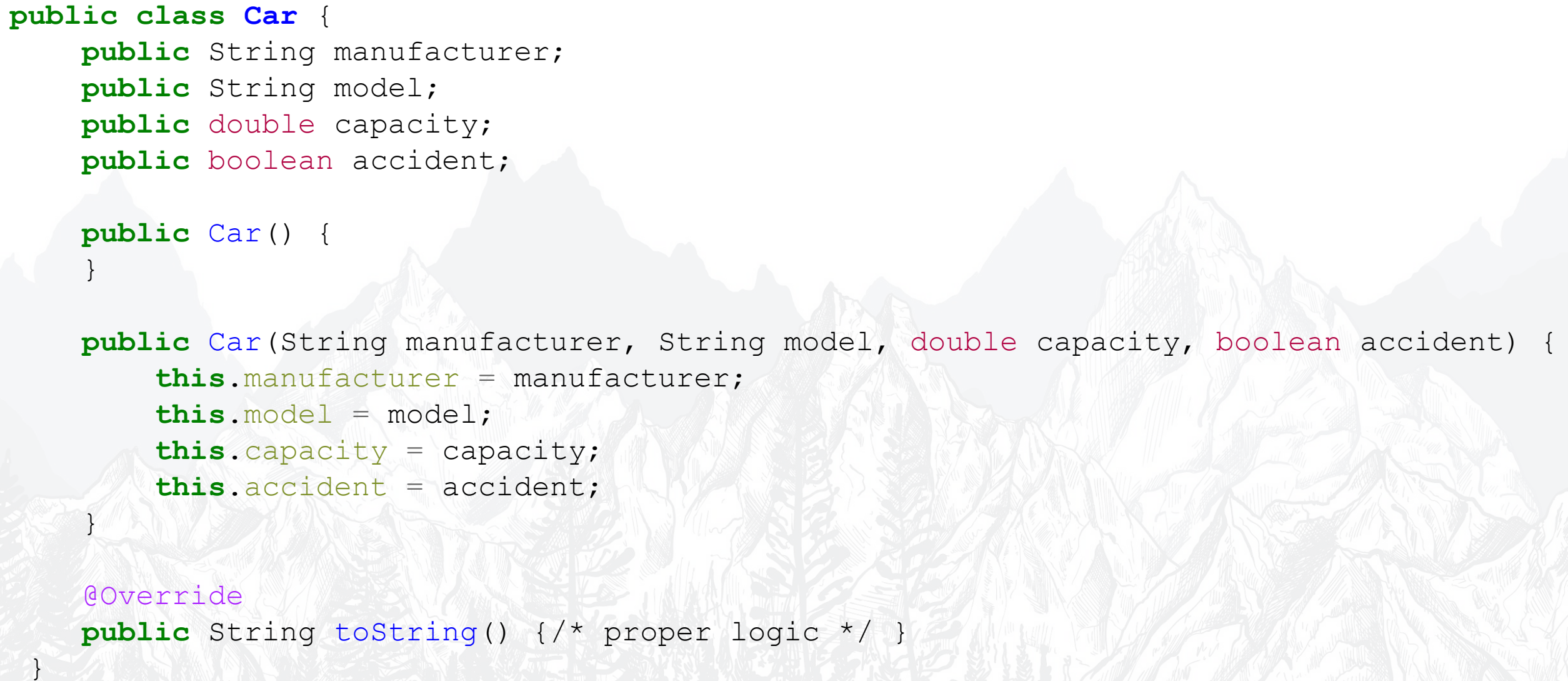
boolean

int

Какво представлява GSON?

- Библиотека, разработена от Google, за сериализиране и десериализиране на Java обекти от и към JSON формат
- Обикновено, за да се ползва, се изисква обектите да имат и конструктор без параметри
- За да можем да го използваме, е нужно да включим следната библиотека:

implementation 'com.squareup.retrofit2:converter-gson:{version}'



```
public class Car {  
    public String manufacturer;  
    public String model;  
    public double capacity;  
    public boolean accident;  
  
    public Car() {  
    }  
  
    public Car(String manufacturer, String model, double capacity, boolean accident) {  
        this.manufacturer = manufacturer;  
        this.model = model;  
        this.capacity = capacity;  
        this.accident = accident;  
    }  
  
    @Override  
    public String toString() { /* proper logic */ }  
}
```



```
public class Person {
    public String name;
    public String surname;
    public Car[] cars;
    public int phone;
    public transient int age;

    public Person() {
    }

    public Person(String name, String surname, int phone, int age, Car[] cars) {
        this.name = name;
        this.surname = surname;
        this.cars = cars;
        this.phone = phone;
        this.age = age;
    }

    @Override
    public String toString() { /* proper logic */ }
}
```

Ако изпълним следния код:

```
Gson gson = new Gson();  
Person johnDoe = gson.fromJson(json, Person.class);  
System.out.println(johnDoe.toString());
```

Ще получим следния изход:

```
Name: John Doe  
Phone: 2025550191  
Age: 0  
Car 1: Manufacturer: Audi, Model: A4, Capacity: 1.8, Accident: false  
Car 2: Manufacturer: Škoda, Model: Octavia, Capacity: 2.0, Accident: true
```




DEMO



HOMEWORK

Задача

- Създайте апп, който при отваряне показва цитат на деня. Цитатът трябва да е един и същ, независимо колко пъти се отвори апп-а съответния ден. Единствено при натискане на бутон за нов цитат, цитатът на деня се сменя.
- Потребителят да има възможност да запазва любим свой цитат. За тази цел използвайте *RecyclerView*, което да заема половината от екрана, а в другата половина да е бутонът и полето, където се показва случайният цитат.

Бонус: Запазените цитати да се пазят не на същата, а в отделна страница. Помислете как може да го направите с *фрагменти* и *BottomNavigation*.

Примерен дизайн:

Бутон за нов
цитат

Quote of the day:

"Lorem ipsum dolor sit amet,
consectetur adipiscing elit, sed
do eiusmod tempor incididunt ut
labore et dolore magna aliqua."

GET NEW QUOTE

SAVE QUOTE

Текущият цитат

RecyclerView,
пазещо любимите
цитати

My favourite quotes:

"Lorem ipsum dolor sit amet,
consectetur adipiscing elit"

"Sed do eiusmod tempor
incidunt ut labore et dolore
magna aliqua."

При натискането на
този бутон, текущия
цитат трябва да се
запазва в любимите
цитати.

Забележка: Трябва
любимите цитати да се
пазят локално, така че
при затваряне и
отваряне на апп-а да са
същите.

Полезни материали:

- <https://square.github.io/retrofit/> - как се ползва Retrofit
- <https://github.com/public-apis/public-apis> - списък от безплатни API-та, които може да ползвате
- <https://pprathameshmore.github.io/QuoteGarden/> - API-то, което ползвахме по време на лекцията

ВЪПРОСИ?



© 2020 Нет Ит

БЛАГОДАРЯ ЗА ВНИМАНИЕТО!



SOFTWARE
ACADEMY

