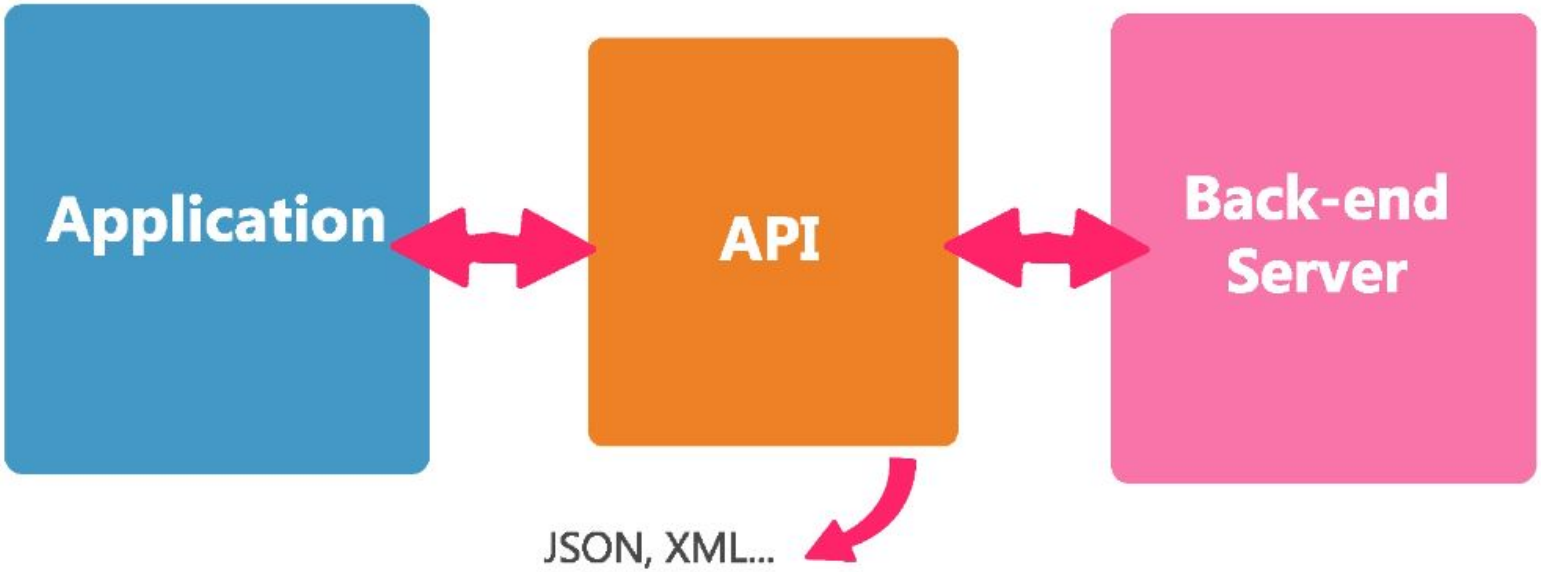
Лекция №8
«Web, JSON»

Москва 2019

Android API



Библиотека Volley

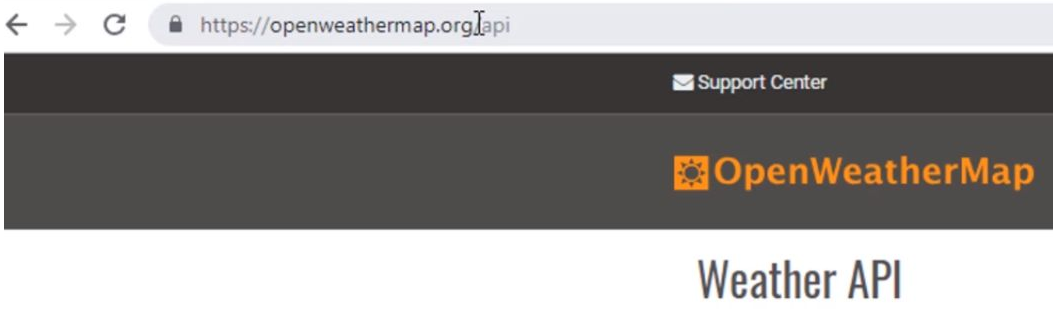
Volley - это HTTP-библиотека, которая упрощает и ускоряет работу в сети для приложений Android. Доступна на GitHub .

Библиотека обладает следующими возможностями:

Автоматическое планирование сетевых запросов.
Несколько одновременных сетевых подключений.
Прозрачное кэширование отклика диска и памяти со стандартной согласованностью HTTP- кэша .
Поддержка приоритизации запросов.
Простота настройки, например, для повторных попыток и откатов.
Строгий порядок, позволяющий легко заполнять ваш пользовательский интерфейс данными, извлекаемыми асинхронно из сети.
Инструменты отладки и трассировки.
Volley не подходит для потоковых передачи видео.

JSON OBJECT

```
{
  "userId": 1,
  "id": 1,
  "title": "delectus aut autem",
  "completed": false
}
```



JSON был создан для передачи данных между сайтами или из сайта.
Во многих API есть JSON,
Facebook API, vk API и.т. д.

https://samples.openweathermap.org/data/2.5/weather?q=London,uk&appid=b6907d289e10d714a6e88b30761fae22

https://www.jsonmate.com/

Пример программы для получения запросов JSON

https://jsonplaceholder.typicode.com/todos
https://developer.android.com/training/volley/request

Шаблон синглетон Java

```
CheeseBurger cheeseBurger = new CheeseBurger();
```

Одиночка (англ. Singleton) — порождающий шаблон проектирования, гарантирующий, что в однопоточном приложении будет единственный экземпляр некоторого класса, и предоставляющий глобальную точку доступа к этому экземпляру.

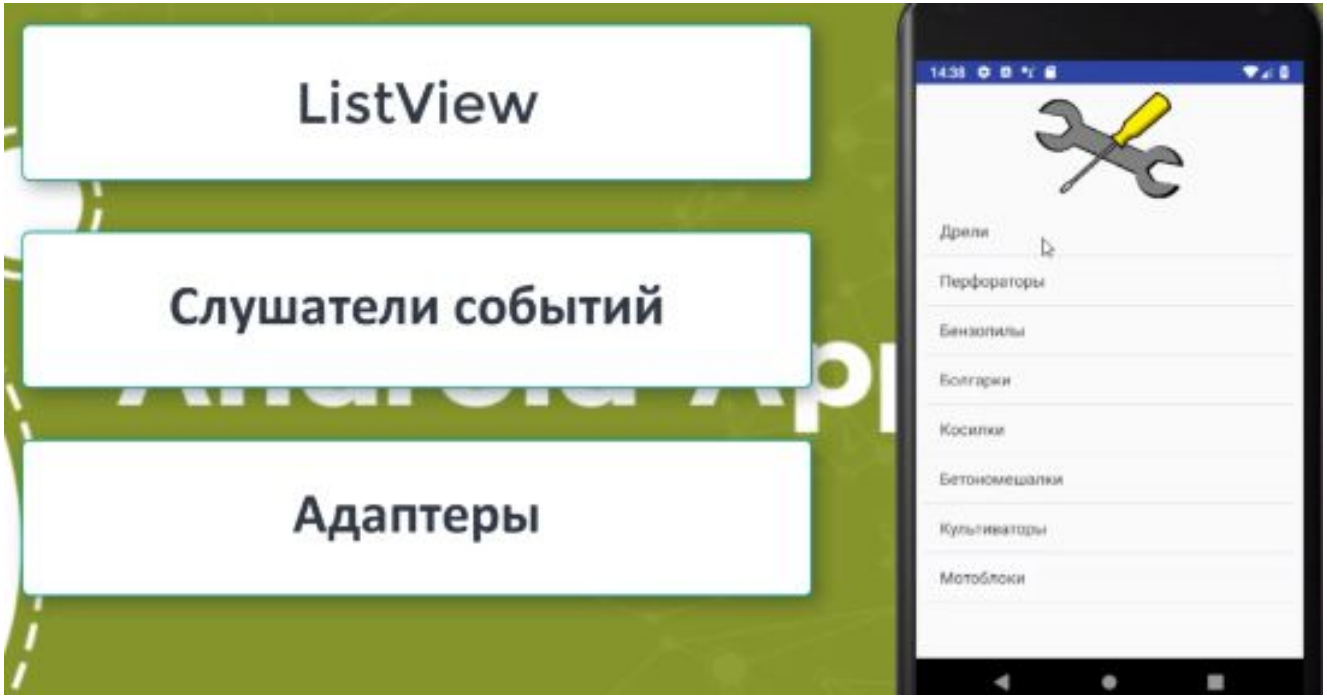
У класса есть только один экземпляр, и он предоставляет к нему глобальную точку доступа. При попытке создания данного объекта он создаётся только в том случае, если ещё не существует, в противном случае возвращается ссылка на уже существующий экземпляр и нового выделения памяти не происходит.
Например, класс настроек приложения.

Шаблон синглетон Java

Если ваше приложение постоянно использует сеть, вероятно, наиболее эффективно настроить один экземпляр RequestQueue, который будет работать в течение всего срока службы вашего приложения. Рекомендуемый подход заключается в реализации одноэлементного класса, который инкапсулирует RequestQueue и другие функциональные возможности Volley.

Ключевой концепцией является то, что RequestQueue должен создаваться с контекстом приложения, а не с контекстом действия. Это гарантирует, что RequestQueue будет длиться в течение всего жизненного цикла вашего приложения, а не воссоздается каждый раз, когда воссоздается действие (например, когда пользователь поворачивает устройство).

ListView, адаптеры, слушатели событий



ArrayAdapter

В Android часто используются адаптеры. Если говорить в общих чертах, то адаптеры упрощают связывание данных с элементом управления. ArrayAdapter является простейшим адаптером, который специально предназначен для работы с элементами списка типа ListView, Spinner, GridView и им подобным.

```
// определяем массив типа String
final String[] products = new String[] {
   "Телефон", "Телевизор", "Ноутбук"
};

ArrayAdapter<String> adapter = new ArrayAdapter<String>(this, R.layout.list_item,
products);

listView.setAdapter(adapter);
```

Вид приложения зависит от устройства

На телефоне две активности.
На планшете одна активность.
Один код должен выполняться в нескольких активностях.
Вместо дублирования кода используются фрагменты
Фрагменты дают возможность повторно использовать код

Фрагмент как и активность имеет макет

На телефоне две активности.
На планшете одна активность.
Один код должен выполняться в нескольких активностях.
Вместо дублирования кода используются фрагменты
Фрагменты дают возможность повторно использовать код

Фрагмент как и активность имеет макет

1. **При запуске приложение открывает активность MainActivity.**
   Активность использует макет *activity_main.xml*.

2. **Активность использует два фрагмента, WorkoutListFragment и WorkoutDetailFragment.**

3. **Фрагмент WorkoutListFragment отображает список комплексов упражнений.**
   Он использует макет *fragment_workout_list.xml*.

4. **Фрагмент WorkoutDetailFragment отображает подробное описание одного комплекса.**
   Он использует макет *fragment_workout_detail.xml*.

5. **Оба фрагмента получают свои данные из *Workout.java*.**
   Workout.java содержит массив с объектами `Workout`.

Фрагменты

Создание фрагментов.
Мы создадим два фрагмента: `WorkoutListFragment` используется для вывода списка комплексов упражнений, а `WorkoutDetailFragment` — для вывода подробного описания конкретного комплекса. Оба фрагмента будут отображаться в одной активности. Кроме того, мы добавим класс Java `Workout`, из которого фрагменты будут получать свои данные.

Связывание фрагментов.
Когда пользователь выбирает комплекс упражнений в `WorkoutListFragment`, подробное описание этого комплекса должно появиться в `WorkoutDetailFragment`.

Создание макетов для устройств.
Наконец, мы изменим приложение так, чтобы оно по-разному выглядело и работало в зависимости от типа устройства, на котором оно выполняется. Если приложение запущено на устройстве с большим экраном, то фрагменты будут размещаться рядом друг с другом. На устройствах с малыми экранами фрагменты будут находиться в разных активностях.

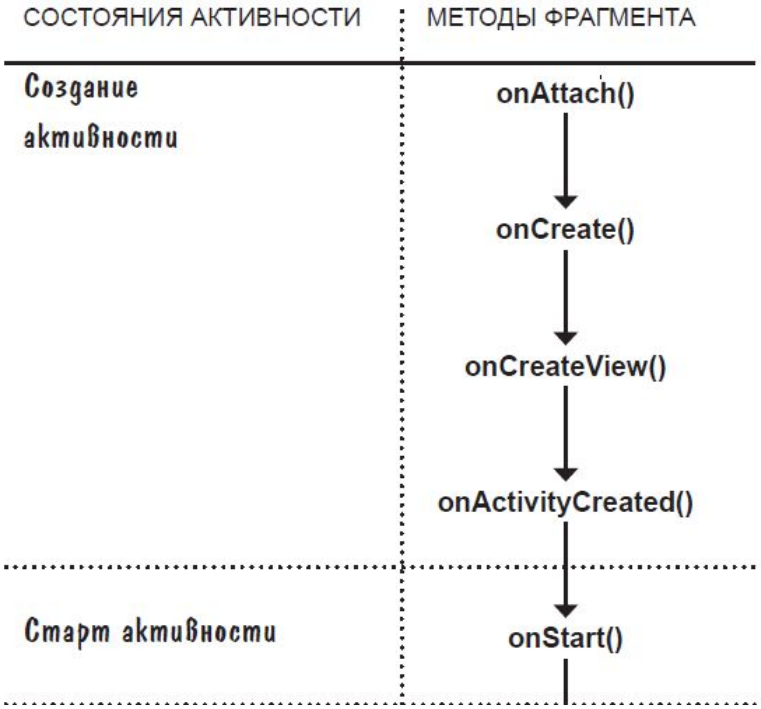
Фрагменты

Этот метод является аналогом метода setContentView() активностей в мире фрагментов.

У каждого фрагмента должен быть определен открытый конструктор без аргументов

Аргумент container передается активностью, использующей фрагмент

Жизненный цикл фрагментов



onAttach(Activity)
Вызывается при связывании фрагмента с активностью.

onCreate(Bundle)
Метод очень похож на метод `onCreate()` активности; используется для выполнения инициализации фрагмента.

onCreateView(LayoutInflater, ViewGroup, Bundle)
Фрагменты используют объект LayoutInflater для создания своего представления в этой точке.

onActivityCreated(Bundle)
Метод вызывается при завершении метода `onCreate()` активности.

onStart()
Метод `onStart()` вызывается перед тем,

Жизненный цикл фрагментов

onResume()
Вызывается, когда фрагмент виден и активно работает.

onPause()
Вызывается, когда фрагмент перестает взаимодействовать с пользователем.

onStop()
Вызывается, когда фрагмент перестает быть видимым.

onDestroyView()
Дает фрагменту возможность освободить любые ресурсы, связанные с его представлением.

onDestroy()
В этом методе фрагмент может освободить любые другие ресурсы, созданные им.

onDetach()
Вызывается при окончательном разрыве связи между фрагментом и активностью.

Списковый фрагмент

ListFragment — разновидность Fragment, специализированная для работы со списковым представлением. В макете по умолчанию этого фрагмента содержится компонент ListView