



ОНЛАЙН-ОБРАЗОВАНИЕ


Онлайн-образование

Проверить, идет ли запись!





Меня хорошо видно && слышно?

Ставьте  , если все хорошо
Напишите в чат, если есть проблемы

Преподаватель



Даниил Попов

- Занимаюсь Android разработкой с 2012 года
- Закончил МГТУ им. Баумана
- Работал в Mail.ru Group и Авито
- Сейчас работаю в Bolt Technology, Эстония



RecyclerView

Даниил Попов

План занятия

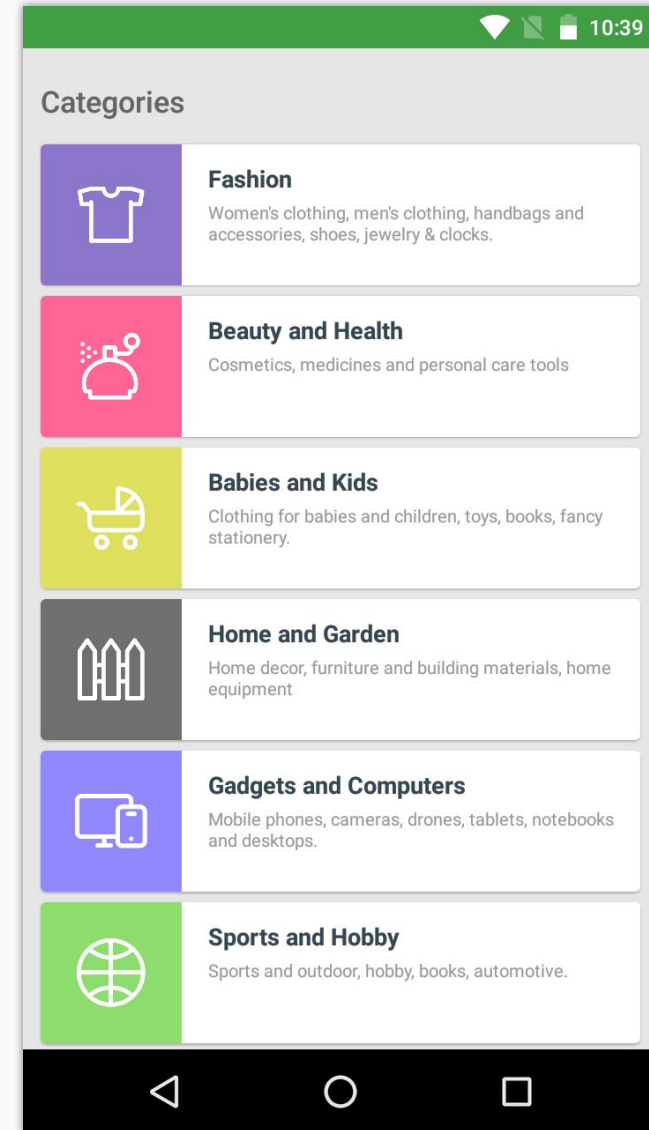
1. Принцип работы RecyclerView
2. Компоненты RecyclerView
3. ItemDecorator
4. ItemTouchHelper
5. ItemAnimator
6. SnapHelper
7. DiffUtil, AsyncListDiffer, ListAdapter

Принцип работы RecyclerView

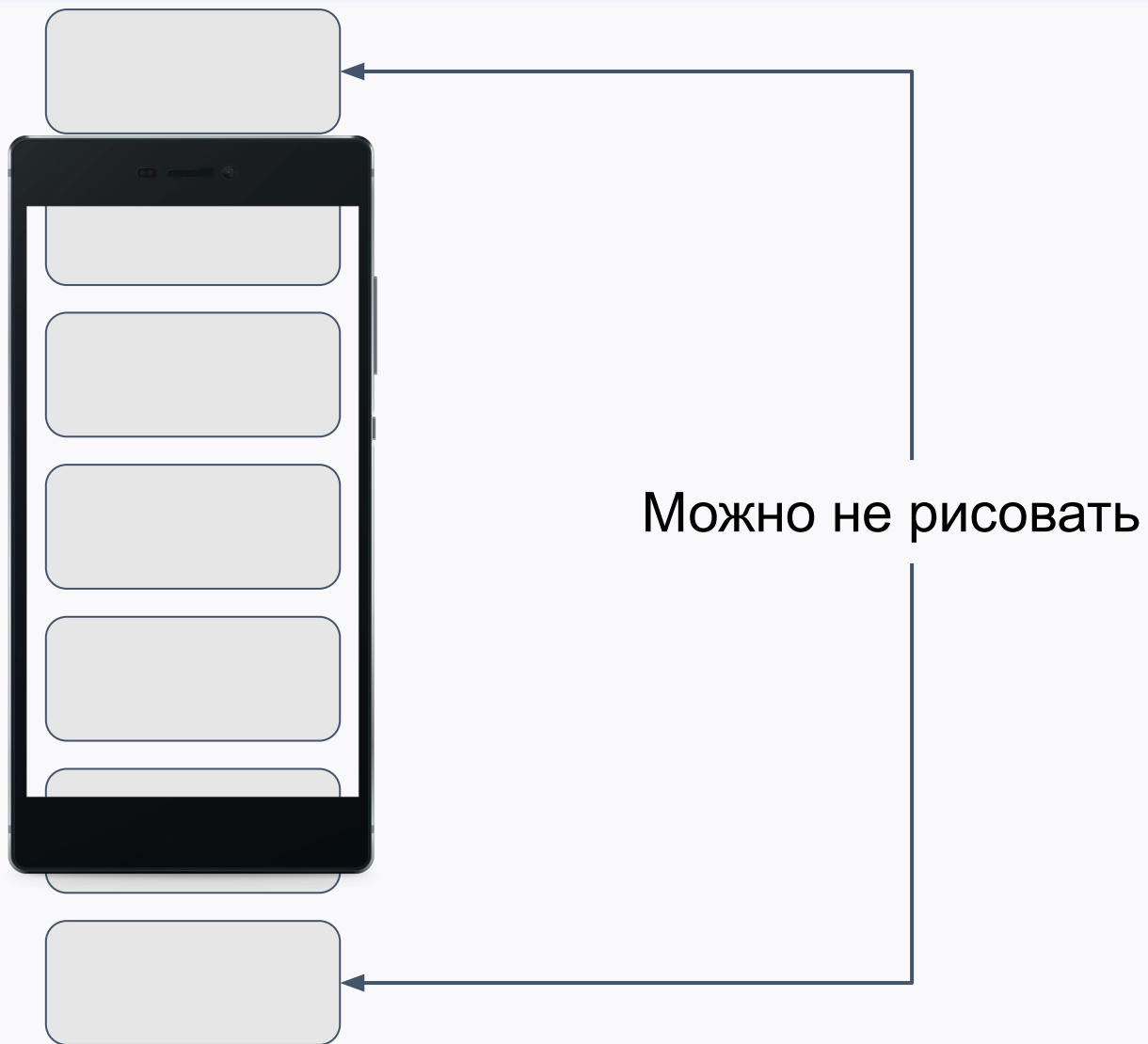
В чем, собственно, проблема?

Задача:

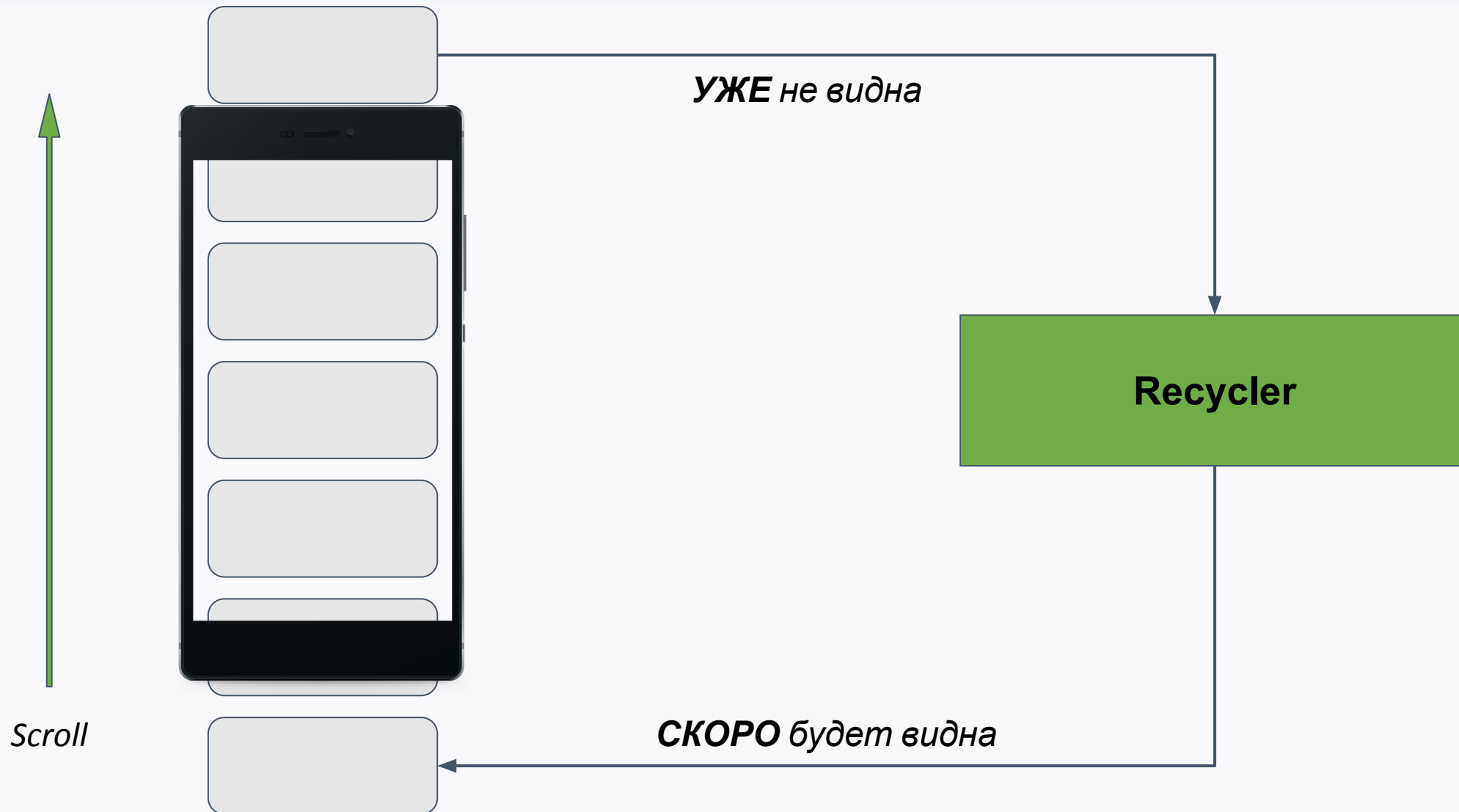
Требуется отобразить список из 10'000 продуктов в приложении интернет-магазина.



Как работает RecyclerView



Переиспользование элементов



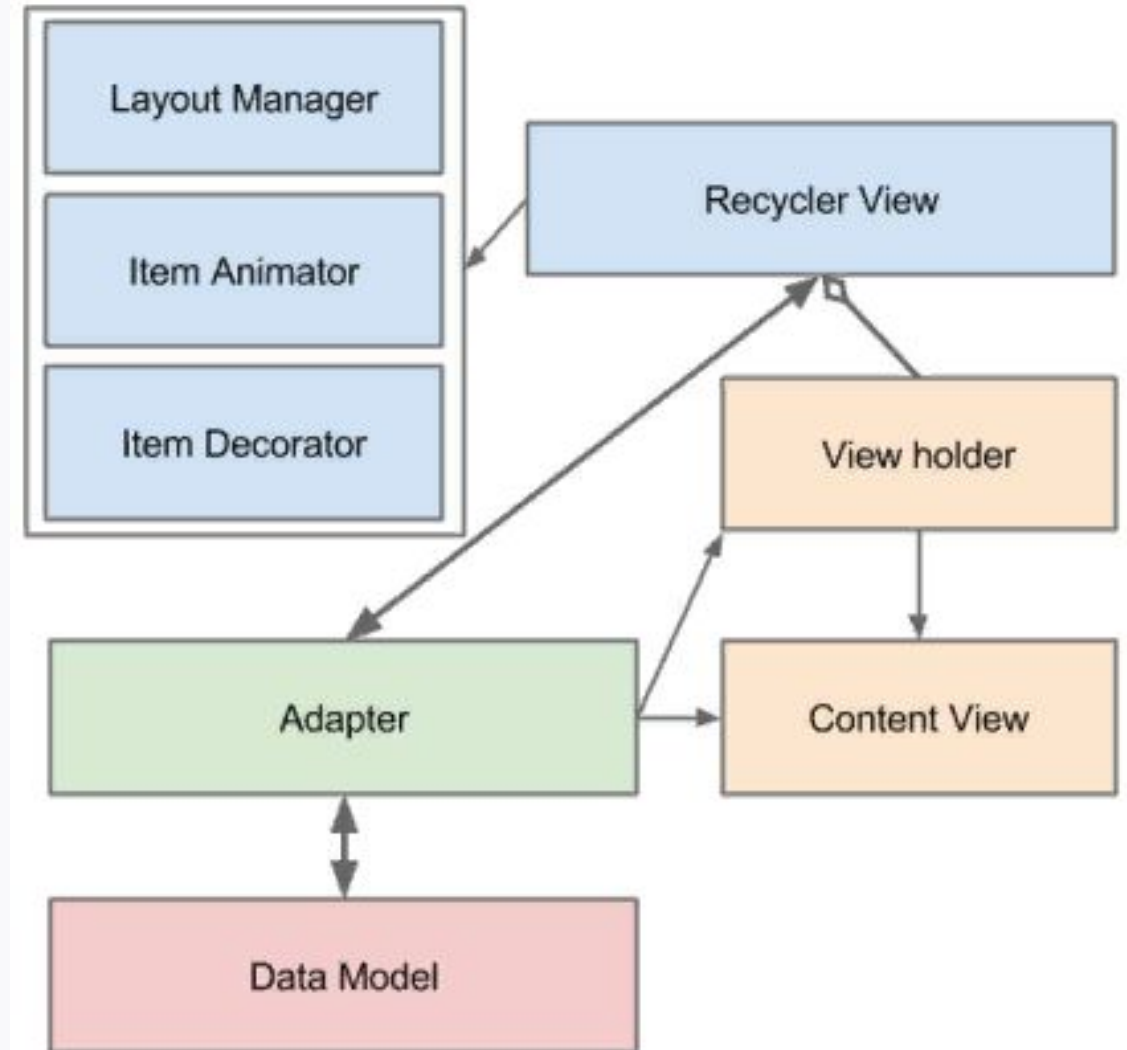
План занятия

1. Принцип работы RecyclerView ✓
2. Компоненты RecyclerView
3. ItemDecorator
4. ItemTouchHelper
5. ItemAnimator
6. SnapHelper
7. DiffUtil, AsyncListDiffer, ListAdapter

Компоненты RecyclerView

Компоненты RecyclerView

- **ViewHolder** - кеширует findViewById
- **Adapter** - создает элементы
- **LayoutManager** - размещает элементы
- **ItemAnimator** - анимирует элементы
- **ItemDecorator** - дорисовывает элементы
- **Helpers**
 - **ItemTouchHelper** - drag&drop и swipe to dismiss
 - **SnapHelper** - gravity для элемента



LayoutManager

- Размещает элементы в RecyclerView
- Отвечает за размер элементов (measure)
- Отвечает за то, какие элементы больше не нужны
- Отвечает за View Focusing; т.е. на каком элементе сфокусироваться.

LayoutManager



Linear



Grid



Grid + span



StaggeredGrid

Adapter

- Создание ViewHolder'ов
- Заполнение ViewHolder'ов информацией
- Уведомление RecyclerView о том какие элементы изменились
- Обработка касаний
- Частичное обновление данных
- Управление количеством ViewType'ов

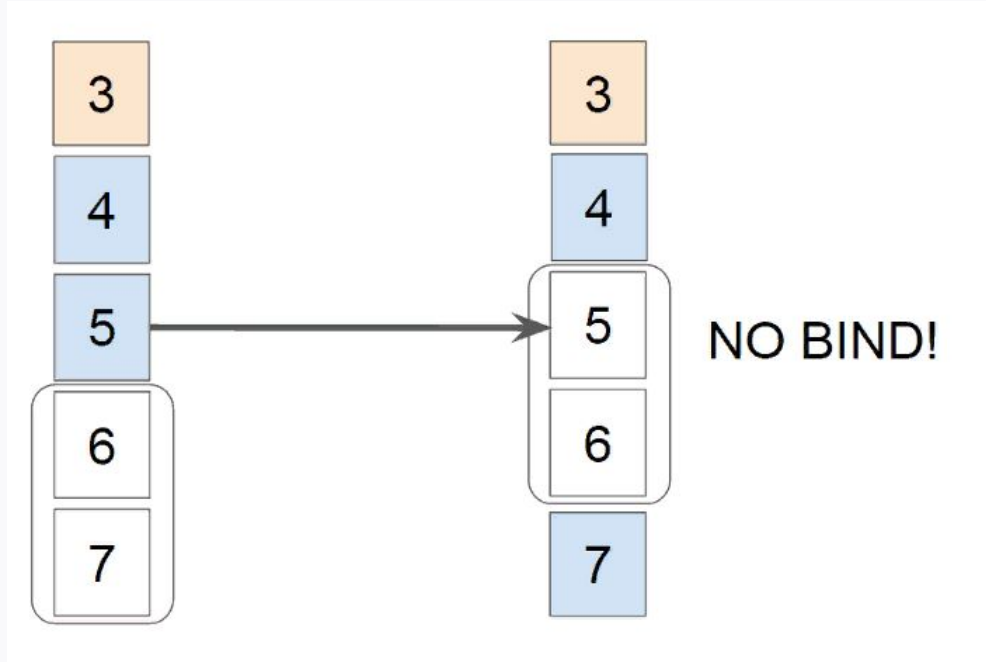
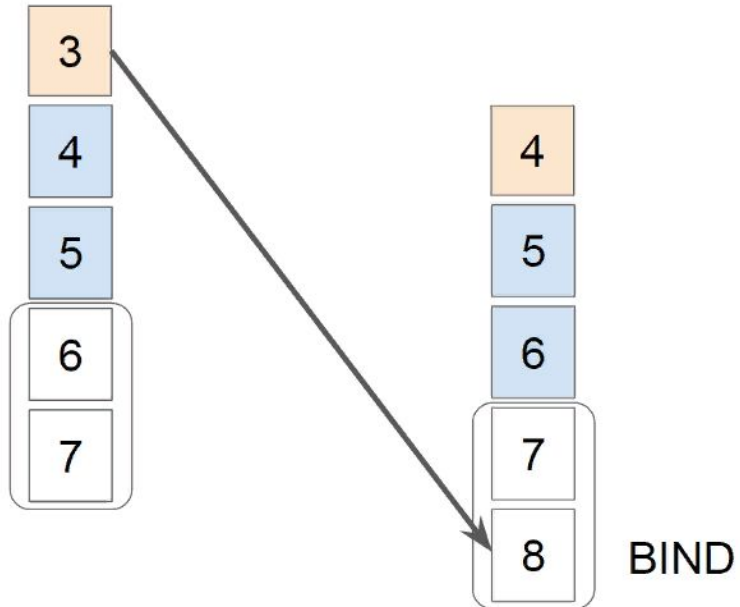
ViewHolder

- Кеширование относительно дорогого findViewById
- Мост между LayoutManager, Animator'ами и Decorator'ами
- Основной элемент Recycling'a

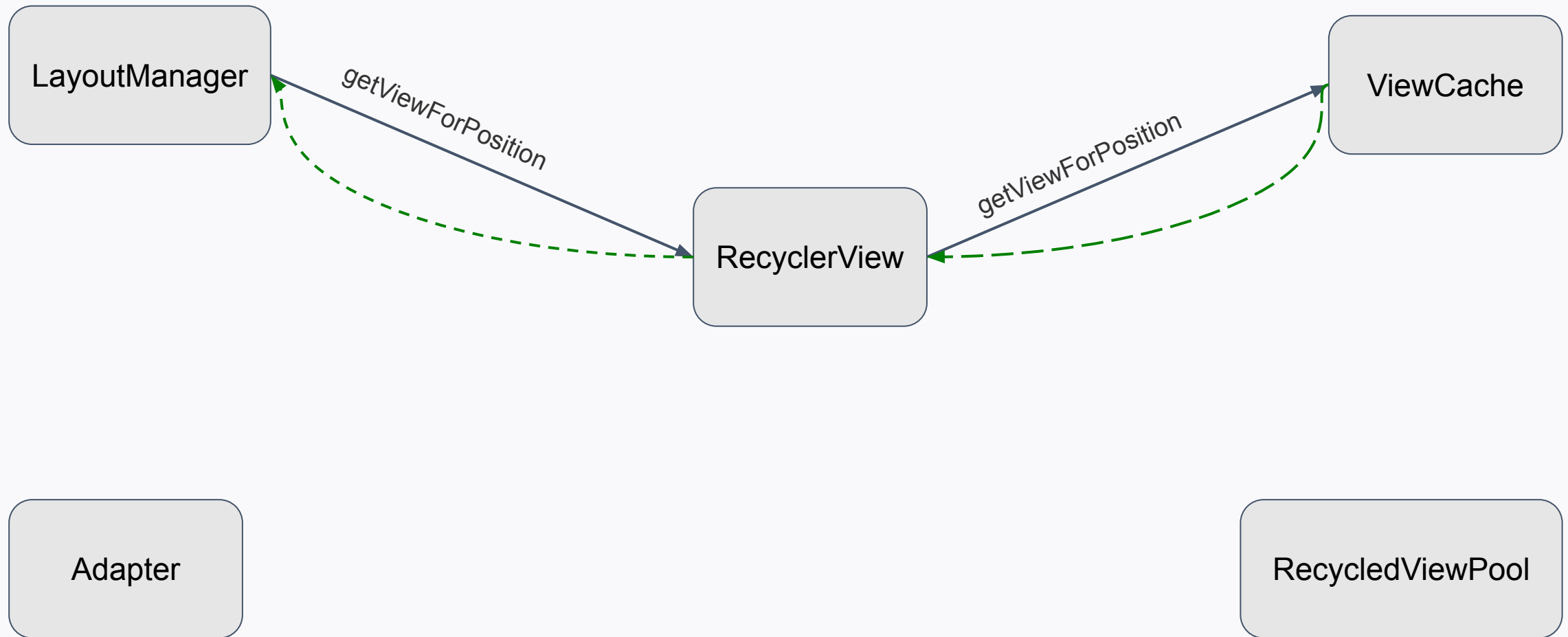
ViewHolder

Есть 3 варианта развития событий:

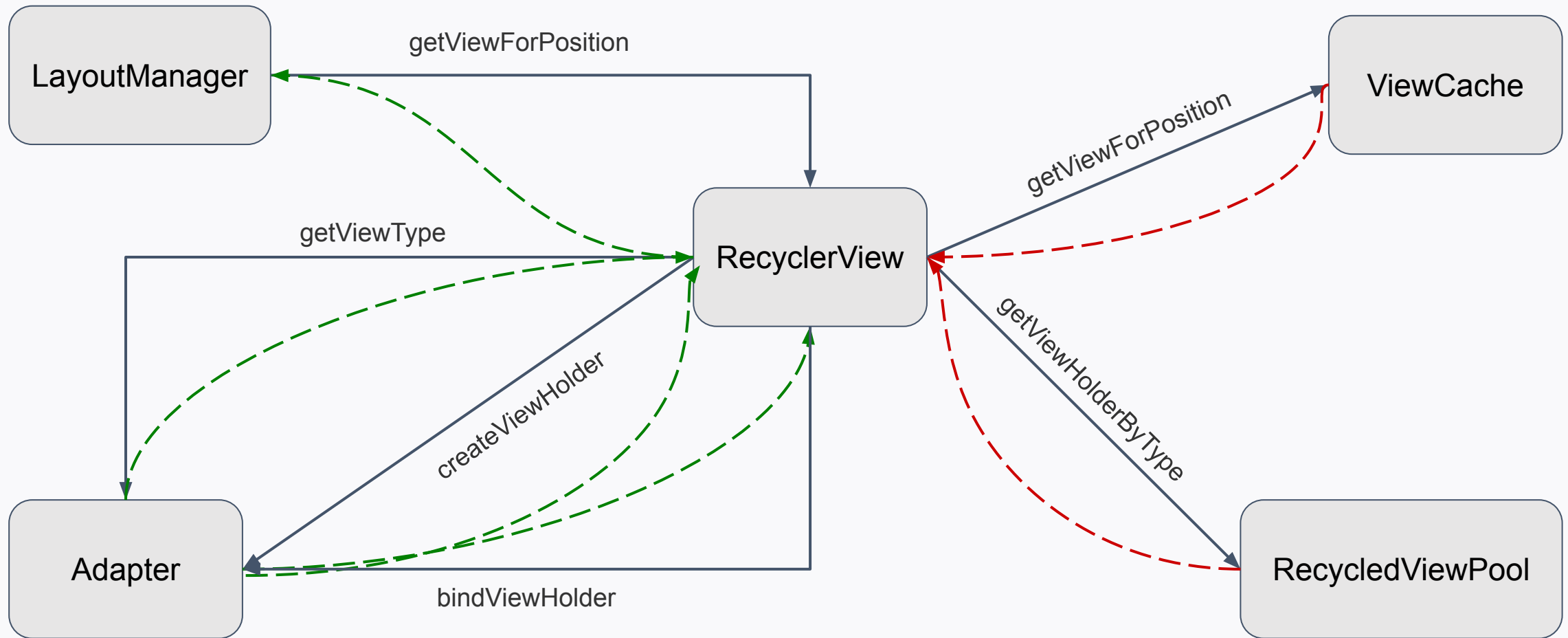
1. Элемента раньше не было: создается, привязывается
2. Элемент был в пуле: привязывается
3. Элемент был в view cache: ничего не делается



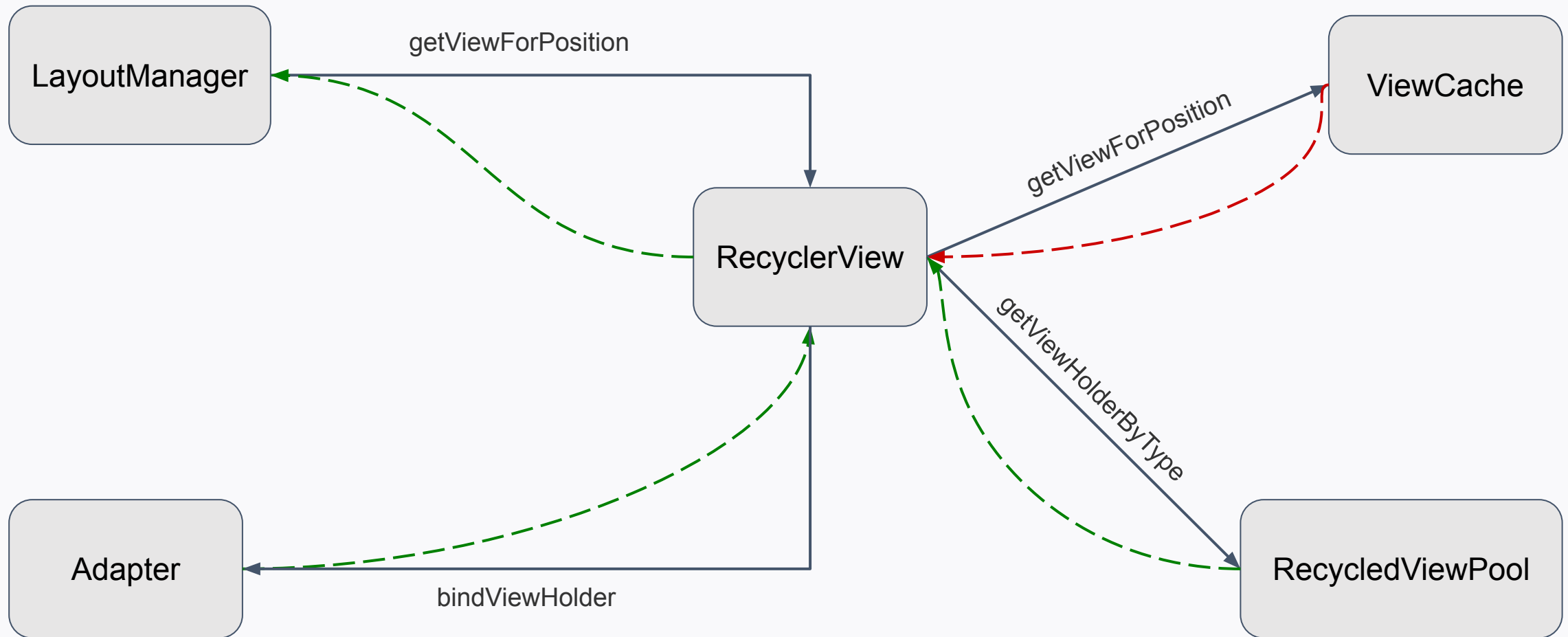
Жизненный цикл ViewHolder



Жизненный цикл ViewHolder



Жизненный цикл ViewHolder



План занятия

1. Принцип работы RecyclerView ✓
2. Компоненты RecyclerView ✓
3. ItemDecorator
4. ItemTouchHelper
5. ItemAnimator
6. SnapHelper
7. DiffUtil, AsyncListDiffer, ListAdapter



LIVE





ItemDecorator

ItemDecorator

- Добавление разделителей и отступов (`getItemOffsets`)
- Рисование под элементом списка (`onDraw`)
- Рисование над элементом списка (`onDrawOver`)

```
recyclerView.addItemDecoration(...)
```

Можно добавлять сколько угодно декораций

План занятия

1. Принцип работы RecyclerView ✓
2. Компоненты RecyclerView ✓
3. ItemDecorator ✓
4. ItemTouchHelper
5. ItemAnimator
6. SnapHelper
7. DiffUtil, AsyncListDiffer, ListAdapter

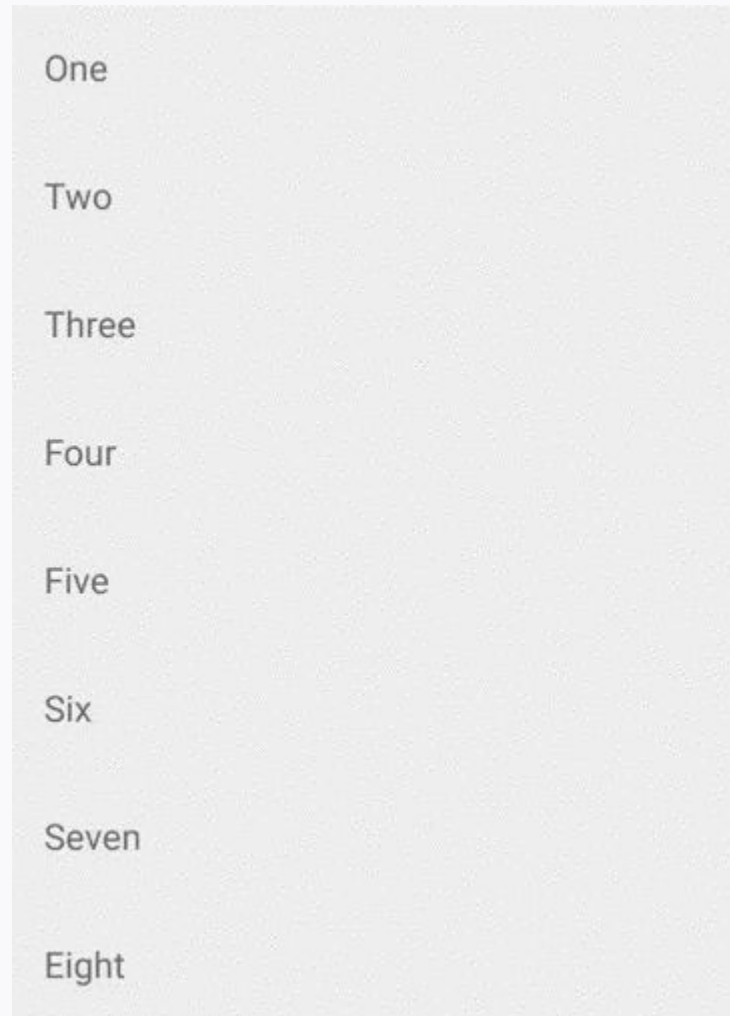


ItemTouchHelper

ItemTouchHelper

Позволяет реализовать перетаскивание и смещение элементов списка

- Drag&Drop
- Swipe to dismiss



ItemTouchHelper

Ключевые методы у Callback:

- `getMovementFlags` - управляет направлениями свайпа и перетаскивания
- `onMove` - **вызывается** при перемещении элемента
- `onSwiped` - **вызывается** при удалении элемента свайпом

План занятия

1. Принцип работы RecyclerView ✓
2. Компоненты RecyclerView ✓
3. ItemDecorator ✓
4. ItemTouchHelper ✓
5. SnapHelper
6. ItemAnimator
7. DiffUtil, AsyncListDiffer, ListAdapter



SnapHelper

SnapHelper

Позволяет настроить эффект “залипания” или “перещелкивания” элементов `RecyclerView`.

Есть две готовые реализации:

1. `LinearSnapHelper` - “прилепляет” центр элемента к центру `RecyclerView`
2. `PagerSnapHelper` - имитация постраничного скролла как у `ViewPager`

План занятия

1. Принцип работы RecyclerView ✓
2. Компоненты RecyclerView ✓
3. ItemDecorator ✓
4. ItemTouchHelper ✓
5. SnapHelper ✓
6. ItemAnimator
7. DiffUtil, AsyncListDiffer, ListAdapter



ItemAnimator

ItemAnimator

Позволяет анимировать элементы списка в случае:

- Добавления
- Удаления
- Изменения

Базовый класс - `RecyclerView.ItemAnimator`

Стандартные реализации:

- `SimpleItemAnimator`
- `DefaultItemAnimator` (по умолчанию)

```
recyclerView.itemAnimator = CustomItemAnimator()
```


ItemAnimator

Отрисовка элементов делится на два этапа:

- preLayout - элементы до изменения списка + те, что должны появиться;
- postLayout - элементы после окончания анимации. `RecyclerView` запоминает состояние `preLayout` и сравнивает его с состоянием `postLayout`. И определив эту разницу, `RecyclerView` запускает анимацию элементов.



План занятия

1. Принцип работы RecyclerView ✓
2. Компоненты RecyclerView ✓
3. ItemDecorator ✓
4. ItemTouchHelper ✓
5. SnapHelper ✓
6. ItemAnimator ✓
7. DiffUtil, AsyncListDiffer, ListAdapter



O T U S
ОНЛАЙН ОБРАЗОВАНИЕ



DiffUtil

DiffUtil

`DiffUtil` - это служебный класс, который вычисляет разницу между двумя списками и в качестве результата дает список операций обновления, преобразующих первый список во второй.

`AsyncListDiffer` - внутри использует тот же `DiffUtil`, но разница вычисляется в фоновом потоке, что позволяет обрабатывать большие списки без зависания UI.

`ListAdapter` - готовый адаптер, который скрывает работу с `AsyncListDiffer` внутри себя, упрощая работу с большими списками.

План занятия


1. Принцип работы RecyclerView ✓
2. Компоненты RecyclerView ✓
3. ItemDecorator ✓
4. ItemTouchHelper ✓
5. SnapHelper ✓
6. ItemAnimator ✓
7. DiffUtil, AsyncListDiffer, ListAdapter ✓



Для дальнейшего изучения

Комбинирование адаптеров

- [ConcatAdapter](#) - позволяет последовательно “склеивать” произвольное количество независимых адаптеров. [Пример использования](#).
- Библиотека [AdapterDelegates](#) - предлагает другой подход к декомпозиции и переиспользованию адаптеров. [Статья](#) с описанием проблемы и способом ее решения данной библиотекой.

The background of the image is an aerial photograph of a dense city skyline, likely New York City, with numerous skyscrapers. The image is overlaid with a semi-transparent blue layer. In the center of this layer, there is a network of white lines connecting various points, creating a geometric pattern. The text is written in a clean, white, sans-serif font.

Заполните, пожалуйста,
опрос о занятии по ссылке в чате

Спасибо за внимание!
Приходите на следующие вебинары



Даниил Попов

Ведущий Android инженер

Bolt Technology OÜ