# <TeachMeSkills/>

Школа программирования
**teachmeskills.com**

курс

# Android разработчик

## Занятие 21. Начало работы со списками

## Агенда занятия

ListView

RecyclerView

ScrollView, NestedScrollView

## ▶ ListView

RecyclerView

ScrollView, NestedScrollView

# ListView

*Устаревший способ отображения списка данных. Работает с адаптером ArrayAdapter.*

## ListView

```
<ListView
    android:id="@+id/listView"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

# ListView

```
private val catNames = listOf(
    "Рыжик", "Барсик", "Мурзик", "Мурка", "Васька",
    "Томасина", "Кристина", "Пушок", "Дымка", "Кузя",
    "Китти", "Масяня", "Симба"
)

override fun onCreateView(
    inflater: LayoutInflater,
    container: ViewGroup?,
    savedInstanceState: Bundle?
): View? {
    viewBinding = FragmentFirstBinding.inflate(inflater, container, attachToParent: false)
    return viewBinding?.root
}

override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
    super.onViewCreated(view, savedInstanceState)
    navController = findNavController()
    setupView()
}

private fun setupView() {
    // Инициализация адаптера для ListView
    val arrayAdapter = ArrayAdapter<String>(
        requireContext(),
        android.R.layout.simple_list_item_1,
        catNames
    )
    viewBinding?.listView?.adapter = arrayAdapter
}
```

## ListView

*Получение доступа к разметке*

```
private fun setupView() {
    // Инициализация адаптера для ListView
    val arrayAdapter = object : ArrayAdapter<String>(
        requireContext(),
        android.R.layout.simple_list_item_1,
        catNames
    ) {
        override fun getView(position: Int, convertView: View?, parent: ViewGroup): View {
            val view = super.getView(position, convertView, parent)
            val item = getItem(position)
            val title = view.findViewById<TextView>(android.R.id.text1)
            title.text = item
            return view
        }
    }
    viewBinding?.listView?.adapter = arrayAdapter
}
```

## ListView

*Слушатель нажатий*

```
viewBinding?.listView?.setOnItemClickListener { parent, itemClicked, position, id ->
    val selectedCatName = catNames[position]
    log( text: "Selected cat name: $selectedCatName")
}
```

## ListView

*Для обновления данных в списке можно использовать метод адаптера notifyDataSetChange()*

ListView

▶ **RecyclerView**

ScrollView, NestedScrollView

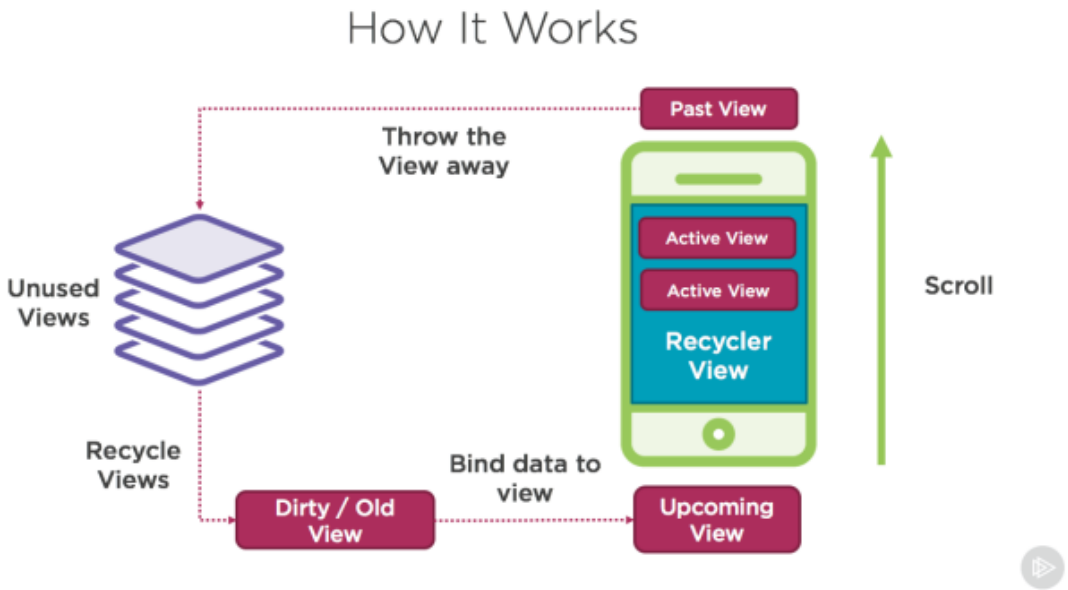
# RecyclerView

*Современный список с повторным использованием элементов. Работает в связке с адаптером, ViewHolder, LayoutManager.*

Преимущества:

- Поддерживает анимации, вложенные списки, Grid/List/Horizontal и т.п.
- Высокая производительность при больших объемах данных.

# RecyclerView



## RecyclerView. ViewHolder

*Специальный класс-обертка над View, который хранит ссылки на элементы, чтобы не вызывать findViewById каждый раз. Принимает ваш View в качестве параметра конструктора.*

```
class RecyclerViewHolder(
    private val binding: ItemCatBinding
) : RecyclerView.ViewHolder(binding.root) {

    // Метод для связывания данных с элементами представления
    fun onBind(catName: String) {
        binding.catName.text = catName
    }
}
```

# RecyclerView. Adapter

*Adapter — это мост между данными и элементами интерфейса (ViewHolder-ами).*

- Хранит или получает список данных.
- Создаёт View'ы (onCreateViewHolder).
- Связывает данные с View'ами (onBindViewHolder).

# RecyclerView. Adapter

```
class RecyclerAdapter(private val itemList: List<String>) : RecyclerView.Adapter<RecyclerViewHolder>() {


    // Метод для создания нового элемента представления
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): RecyclerViewHolder {
        val binding = ItemCatBinding.inflate(
            LayoutInflater.from(parent.context),
            parent,
            attachToParent: false
        )
        return RecyclerViewHolder(binding)
    }

    // Метод для связывания данных с элементами представления
    override fun onBindViewHolder(holder: RecyclerViewHolder, position: Int) {
        holder.onBind(itemList[position])
    }

    // Метод для получения количества элементов в списке
    override fun getItemCount(): Int = itemList.size
}
```

# RecyclerView. LayoutManager

*LayoutManager управляет расположением элементов в RecyclerView.*

Типы:

- LinearLayoutManager — Вертикальный или горизонтальный список
- GridLayoutManager — Сетка (как в галерее)
- StaggeredGridLayoutManager — Сетка с элементами разной высоты

```
// Установка менеджера компоновки для RecyclerView
binding?.recyclerView?.layoutManager = LinearLayoutManager(requireContext())
```

# RecyclerView

Как всё работает вместе:

- RecyclerView вызывает onCreateViewHolder() → создаёт нужный View
- Вызывает onBindViewHolder() → вставляет туда данные
- LayoutManager расставляет элементы
- При прокрутке RecyclerView переиспользует ViewHolder'ы

ListView

RecyclerView

▶ **ScrollView, NestedScrollView**

## ScrollView, NestedScrollView

*ScrollView — это контейнер, который позволяет прокручивать вложенные View'ы по вертикали. Поддерживает только одного прямого потомка. Обычно это LinearLayout или ConstraintLayout.*

- Нельзя вложить RecyclerView в ScrollView, иначе скролл сломается (двойной скролл → баги и лаги).
- Никакой поддержки вложенной прокрутки или координации с другими прокручиваемыми элементами.

# ScrollView, NestedScrollView

*NestedScrollView — это улучшенная версия ScrollView. Поддерживает вложенную прокрутку, интеракции с AppBarLayout, RecyclerView, ViewPager, и т.д.*

Когда использовать:

- Когда внутри скролла есть другие прокручиваемые элементы (например, RecyclerView, ViewPager).
- Когда используешь CoordinatorLayout, AppBarLayout, и хочешь, чтобы тулбар красиво прятался при прокрутке.

# Задачи

### Задача 1: Отобразить список строк с помощью `ListView`

Отобразить список из 100 строк ("Элемент 1", "Элемент 2", ...) в ListView.

**Задачи**

## Задача 2: Использовать simple_list_item_2 в ListView

Отображать заголовок и подзаголовок в каждой строке(использовать SimpleAdapter)

**Задачи**

## Задача 3: Обработка клика по элементу `RecyclerView`

При нажатии на элемент — показать Toast.

**Задачи**

### Задача 4: Прокручиваемый layout с `ScrollView`

Сделай длинную форму с полями и кнопкой внизу.

**Задачи**

## Задача 5: Отображение RecyclerView в NestedScrollView

Отобразить блок с RecyclerView внутри прокручиваемого контента.

Q&A

# Домашнее задание

# Задачи

### Задача 1: Динамическое добавление элементов в `RecyclerView`

При нажатии на кнопку добавлять новый элемент в список.

# Задачи

### Задача 2: Пустой экран в `RecyclerView`, если нет данных

Показать заглушку (например, TextView "Нет данных") вместо списка, если список пуст.

Q&A

# Ваши вопросы

# Спасибо