<TeachMeSkills/>

Школа программирования
teachmeskills.com

курс

Android разработчик

Занятие 15. Введение в Android. Часть 2

Агенда занятия

Темы и стили

Resources

Локализация

Базовые Layout

Базовые View

▶ Темы и стили

Resources

Локализация

Базовые Layout

Базовые View

Темы и стили

Тема — это набор параметров, которые применяются ко всему приложению, Activity или View-компоненту. Она содержит базовые цвета приложения, стили для отрисовки ***всех*** *компонентов приложения и различные настройки.*

Темы и стили. Темы

```
<style name="MyStyleOrTheme">
    <item name="key">value</item>
</style>
```

Для создания используется тег style. У каждого стиля есть имя и он хранит в себе параметры key-value.

Темы и стили

В теме могут быть переопределены основные цвета приложения, стиль для текста или некоторых стандартных компонентов.

Для того чтобы стиль стал полноценной темой, необходимо наследовать от дефолтной реализации темы.

Темы и стили. Тема

```
<style name="BaseAppTheme" parent="Theme.MaterialComponents.NoActionBar">
    <item name="colorPrimary">@color/colorPrimary</item>
    <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
    <item name="colorAccent">@color/colorPrimaryDark</item>
    <item name="android:windowLightStatusBar">false</item>
    <item name="android:windowBackground">@color/colorBackground</item>
    <item name="tabStyle">@style/AppTheme.TabLayout.Tab</item>
    <item name="android:fontFamily">@font/museo_sans_cyrl_500</item>
    <item name="android:statusBarColor">@android:color/transparent</item>
</style>
```

Темы и стили

Стиль — это набор параметров для стилизации одного View-компонента.

Темы и стили. Стиль

```
<!-- Base Button -->
<style name="BaseButtonStyle">
    <item name="android:layout_height">wrap_content</item>
    <item name="android:paddingTop">@dimen/button_padding_top</item>
    <item name="android:paddingBottom">@dimen/button_padding_bottom</item>
    <item name="android:textSize">@dimen/button_text_size</item>
    <item name="android:textAlignment">center</item>
    <item name="android:fontFamily">sans-serif-medium</item>
    <item name="android:maxLines">1</item>
    <item name="android:textAllCaps">true</item>
</style>
```

Темы и стили. Наследование

Как и в ООП, при создании темы и стилей, у нас есть возможность перенимать функционал существующей реализации. Для этого существует 2 способа:

- Explicit (явно)
- Implicit (неявно)

Темы и стили. Наследование

*При явном наследовано мы указываем родителя с помощью ключевого слова **parent:***

```
<style name="SnackbarStyle" parent="Widget.MaterialComponents.Snackbar">
    <!-- ... -->
</style>
```

Темы и стили. Наследование

*При неявном наследовании используется **dot-notation** с указанием родителя:*

```
<style name="SnackbarStyle.Green">
    <!-- ... -->
</style>
```

Темы и стили. Иерархия

Темы и стили

▶ **Resources**

Локализация

Базовые Layout

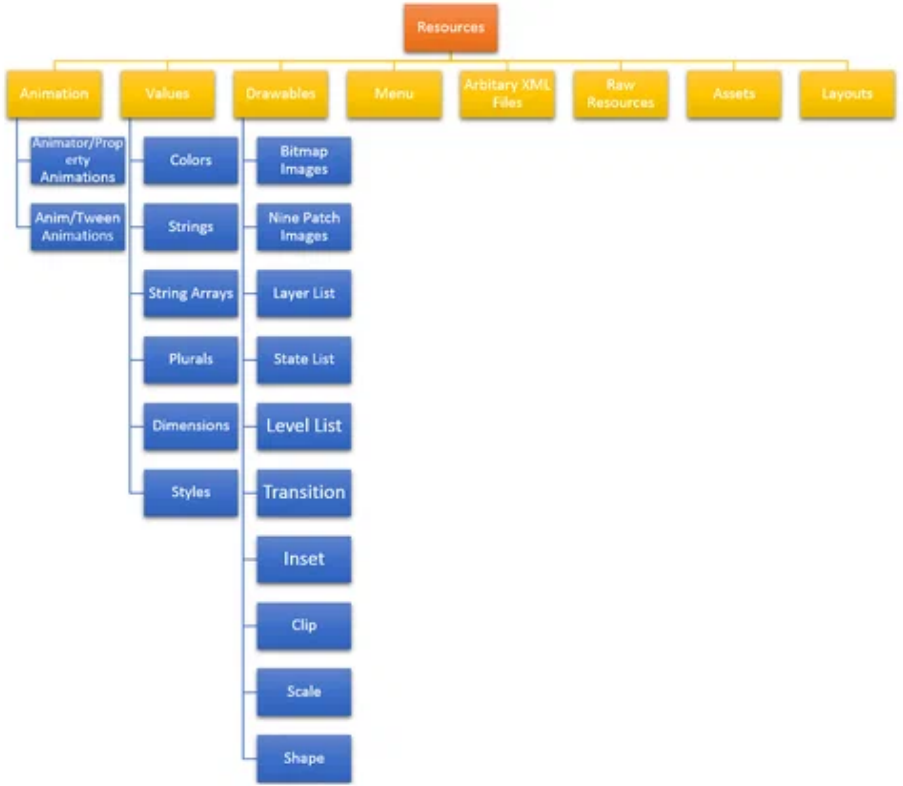
Базовые View

Ресурсы

Ресурсы - один из основных компонентов, с которыми вам придётся работать очень часто. В Android принято держать некоторые объекты - изображения, строковые константы, цвета, анимацию, стили и т.п. за пределами исходного кода. Система поддерживает хранение ресурсов в отдельных файлах. Ресурсы легче поддерживать, обновлять, редактировать.

*Ресурсы находятся в каталоге /**res**.*

Ресурсы


Resources
Animation
Values
Drawables
Menu
Arbitary XML Files
Raw Resources
Assets
Layouts
Animator/Property Animations
Anim/Tween Animations
Colors
Strings
String Arrays
Plurals
Dimensions
Styles
Bitmap Images
Nine Patch Images
Layer List
State List
Level List
Transition
Inset
Clip
Scale
Shape

Ресурсы

Для доступа к ресурсам из кода используется статический класс R, который создается во время компиляции. Может содержать следующие вложенные классы:

- *R.anim — идентификаторы для файлов из каталога res/anim/ (анимация);*
- *R.color — идентификаторы для файлов colors.xml из каталога res/values/ (цвета);*
- *R.dimen — идентификаторы для файлов dimens.xml из каталога res/values/ (размеры);*
- *R.drawable — идентификаторы для файлов из каталога res/drawable/ (изображения);*
- *R.id — идентификаторы представлений и групп представлений для файлов XML-разметки из каталога res/layout/;*
- *R.layout — идентификаторы для файлов разметки из каталога res/layout/;*
- *R.string — идентификаторы для файлов strings.xml из каталога res/values/ (строки);*
- *R.style — идентификаторы для файлов styles.xml из каталога res/values/ (стили);*

Ресурсы

Основные виды ресурсов:

- colors.xml
- dimens.xml
- strings.xml
- drawables
- layouts

Ресурсы. Цвета

Обычно находятся в файле /res/colors.xml. Чтобы объявить цвет, используется тег <color>:

```
<resources>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
</resources>
```

Доступ из кода **resources.getColor(R.color.white)**

Ресурсы. Строки

Обычно находятся в файле /res/strings.xml. Чтобы объявить строку, используется тег <string>:

```
<resources>
    <string name="app_name">MyFirstApp</string>
</resources>
```

Доступ из кода **resources.getString(R.string.app_name)**

Ресурсы. Размеры

Обычно находятся в файле /res/dimens.xml. Чтобы объявить размер, используется тег <dimen>:

```
<resources>
    <dimen name="size_20">20dp</dimen>
</resources>
```

Доступ из кода **resources.getDimension(R.dimen.size_20)**

Ресурсы. Изображения, фигур, градиентов

Обычно находятся в каталоге /res/drawable.

Доступ из кода **resources.getDrawable(R.drawable.ic_launcher_background)**

Ресурсы. Разметка

Обычно находятся в каталоге /res/layout.

Доступ из кода **resources.getLayout(R.layout.activity_main)**

Темы и стили

Resources

▶ **Локализация**

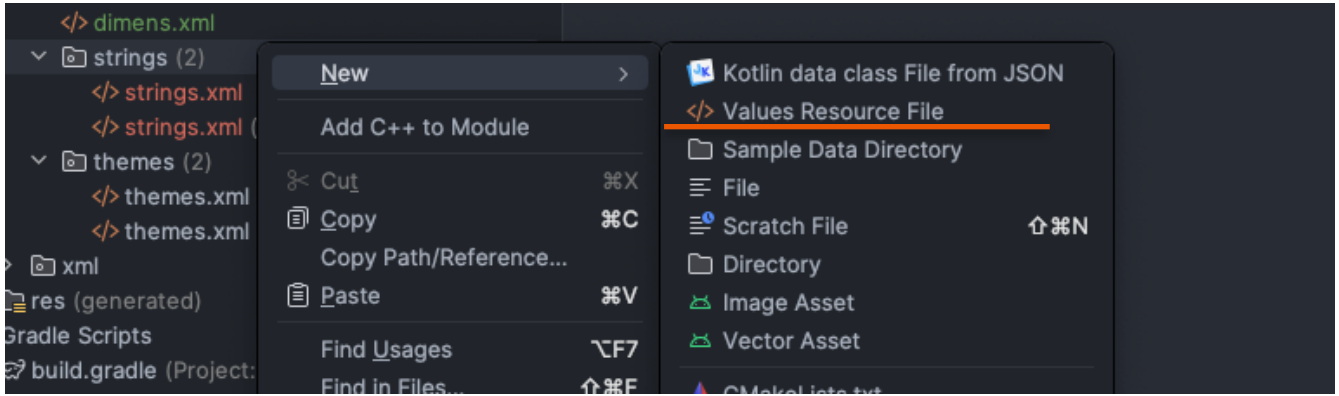
Базовые Layout

Базовые View

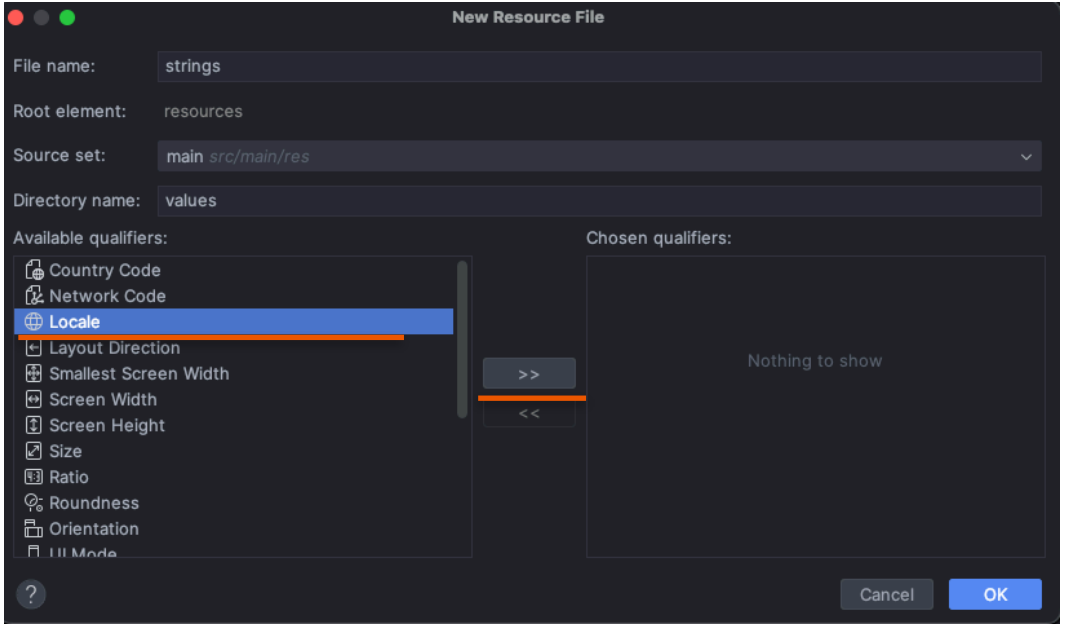
Локализация

Мобильное приложение может поддерживать несколько языков. Для каждого из них необходимо создать свой файл strings.xml и вручную прописать необходимый перевод. После этого приложение будет самостоятельно переключать языковой пакет в зависимости от языка системы.

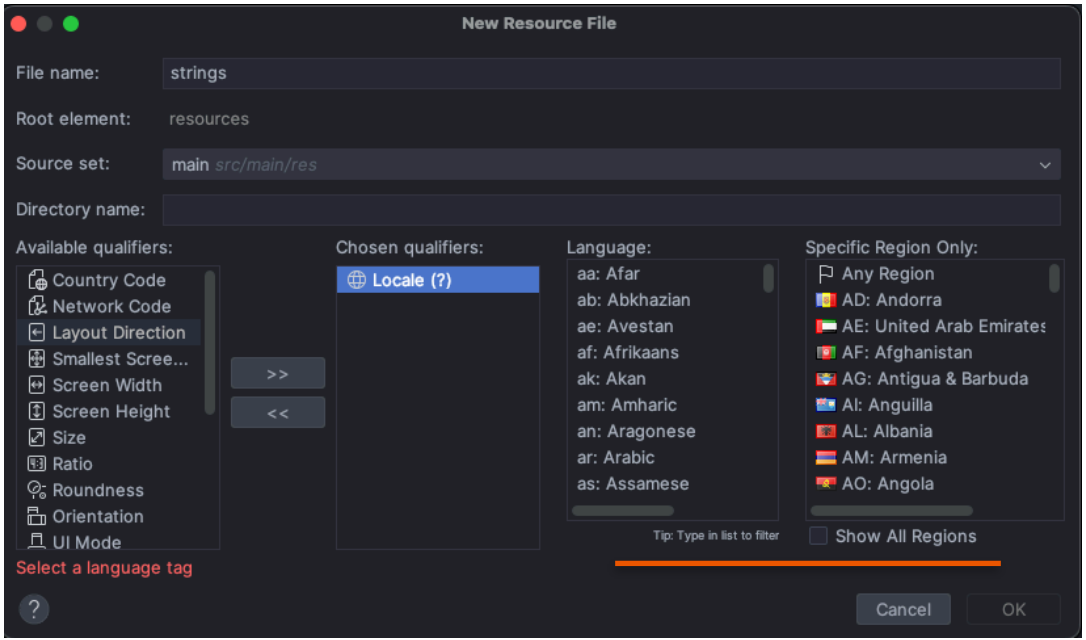
Локализация. Создание языкового файла



Локализация. Создание языкового файла



Локализация. Создание языкового файла



Локализация

Для быстрого добавления строк с вариантами перевода, можно использовать встроенный editor.

Темы и стили

Resources

Локализация

▶ **Базовые Layout**

Базовые View

Базовые Layout

Разметка для экранов или отдельных элементов интерфейса приложения представлена в виде XML кода с корневым элементов. У каждой разметки обязан быть корневой элемент (layout), который в большинстве случаев предопределяет расположение дочерних элементов.

Основные контейнеры:

- LinearLayout
- RelativeLayout
- ConstraintLayout
- FrameLayout

Базовые Layout. FrameLayout

FrameLayout является самым простым типом разметки. Обычно это пустое пространство на экране, которое можно заполнить только дочерними объектами View или ViewGroup. Все дочерние элементы FrameLayout прикрепляются к верхнему левому углу экрана.

Расположением элементов можно манипулировать с помощью атрибута ***layout_gravity.***

Базовые Layout. FrameLayout

```xml
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="Main Activity"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</FrameLayout>
```

Базовые Layout. RelativeLayout

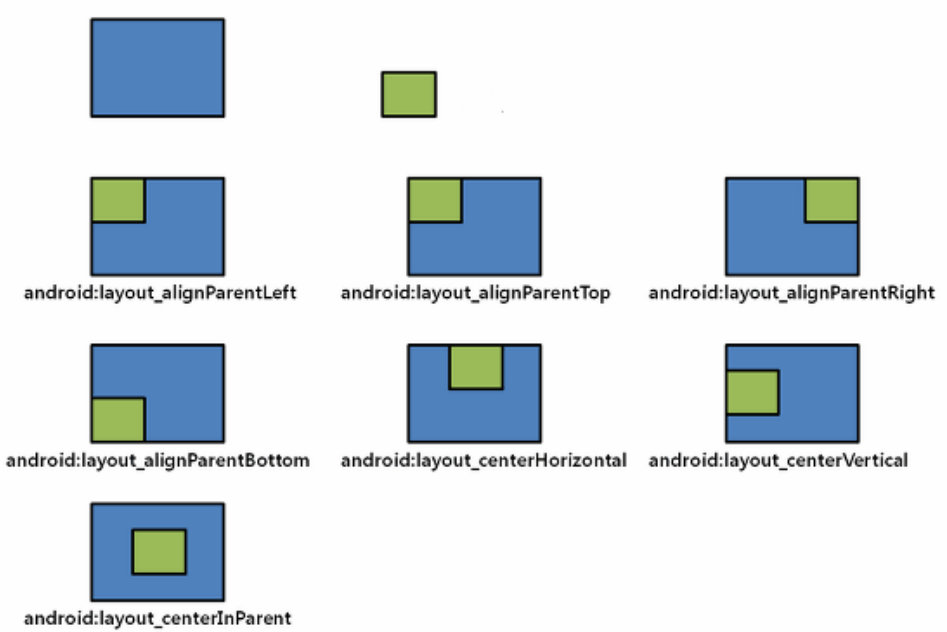
ReiativeLayout *(относительная разметка) находится в разделе* ***Layouts*** *и позволяет дочерним компонентам определять свою позицию относительно родительского компонента или относительно соседних дочерних элементов (по идентификатору элемента).*

В ***RelativeLayout*** *дочерние элементы расположены так, что если первый элемент расположен по центру экрана, другие элементы, выровненные относительно первого элемента, будут выровнены относительно центра экрана. При таком расположении, при объявлении разметки в XML-файле, элемент, на который будут ссылаться для позиционирования другие объекты представления, должен быть объявлен раньше, чем другие элементы, которые обращаются к нему по его идентификатору.*

Базовые Layout. RelativeLayout

- android:layout_alignParentBottom - выравнивание относительно нижнего края родителя
- android:layout_alignParentLeft - выравнивание относительно левого края родителя
- android:layout_alignParentRight - выравнивание относительно правого края родителя
- android:layout_alignParentTop - выравнивание относительно верхнего края родителя
- android:layout_centerInParent - выравнивание по центру родителя по вертикали и горизонтали
- android:layout_centerHorizontal - выравнивание по центру родителя по горизонтали
- android:layout_centerVertical - выравнивание по центру родителя по вертикали

Базовые Layout. RelativeLayout

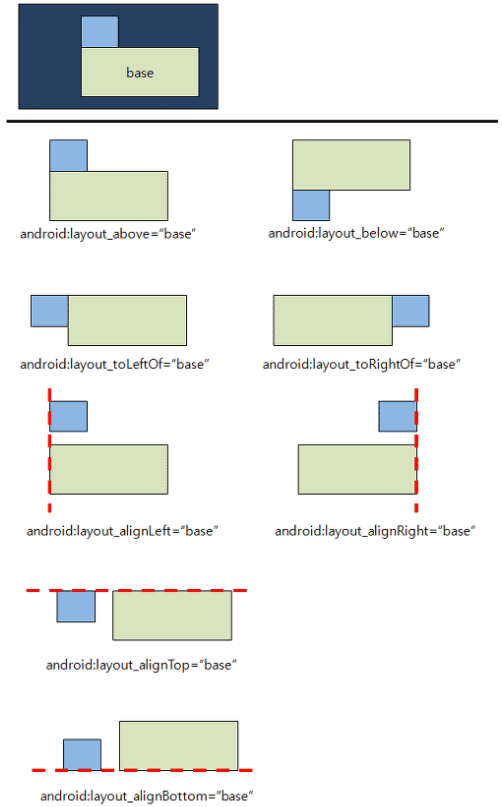


Базовые Layout. RelativeLayout

Компонент можно размещать не только относительно родителя, но и относительно других компонентов. Для этого все компоненты должны иметь свой идентификатор, по которому их можно будет отличать друг от друга. В этом случае вы можете задействовать другие атрибуты.

- android:layout_above - размещается над указанным компонентом
- android:layout_below - размещается под указанным компонентом
- android:layout_alignLeft - выравнивается по левому краю указанного компонента
- android:layout_alignRight - выравнивается по правому краю указанного компонента
- android:layout_alignTop - выравнивается по верхнему краю указанного компонента
- android:layout_alignBottom - выравнивается по нижнему краю указанного компонента
- android:layout_toLeftOf - правый край компонента размещается слева от указанного компонента
- android:layout_toRightOf - левый край компонент размещается справа от указанного компонента

Базовые Layout. RelativeLayout



Базовые Layout. LinearLayout

LinearLayout представлен двумя вариантами - Horizontal и Vertical.
Макет LinearLayout выравнивает все дочерние объекты в одном направлении — вертикально или горизонтально. Направление задается при помощи атрибута ориентации android:orientation:

- android:orientation="horizontal"
- android:orientation="vertical"

Все дочерние элементы помещаются в стек один за другим, так что вертикальный список компонентов будет иметь только один дочерний элемент в ряду независимо от того, насколько широким он является. Горизонтальное расположение списка будет размещать элементы в одну строку с высотой, равной высоте самого высокого дочернего элемента списка.

Базовые Layout. LinearLayout

*Важным атрибутом в LinearLayout является **layout_weight** у дочерних элементов:*

- Разделяет дополнительное свободное место между элементами
- Элементы с 0dp по ширине/высоте растягиваются по весу, а элементы с wrap_content остаются своего размера

Базовые Layout. ConstraintLayout

ConstraintLayout — это мощный ViewGroup, который помогает создавать гибкие и адаптивные интерфейсы без вложенных LinearLayout и RelativeLayout, улучшая производительность.

Базовые Layout. ConstraintLayout

Фишка этого layout в том, что можно привязывать стороны элемента как к родителю (самому ConstraintLayout), так и к другим элементам, тем самым создавая гибкую разметку. Чтобы привязать элемент, необходимо использовать специальные атрибуты constraint:

```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Темы и стили

Resources

Локализация

Базовые Layout

▶ **Базовые View**

Базовые View. TextView

Элемент для отображения текста. Атрибуты:

android:text — выставление текста

android:textSize — размер текста (всегда в sp!)

android:textColor — цвет текста

android:maxLines — макс. кол-во линий

Базовые View. TextView

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Hello World!"
    android:textSize="18sp"
    android:textColor="@android:color/black"
    android:maxLines="1"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

Базовые View. EditText

Используется для ввода текста пользователем. Атрибуты:

android:hint — подсказка

android:inputType — тип ввода данных

```
<EditText
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    android:hint="Введите текст"
    android:inputType="textCapWords" />
```

Базовые View. Button

Используется для обработки нажатий, поддерживает обработку нажатий. Атрибуты:

android:text — текст

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    android:text="Нажми меня" />
```

Базовые View. ImageView

Используется для отображения картинки. Атрибуты:

android:src — ссылка на изображение

android:scaleType — режим масштабирования

```
<ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    android:src="@drawable/ic_launcher_background" />
```

Задачи

Задача 1: Создай стиль для кнопки

Создай стиль для Button, который задаёт:

Фон #FF6200EE

Белый текст

Задачи

Задача 2: Создай светлую и тёмную тему

Добавь поддержку светлой и тёмной темы.

Задачи

Задача 3: Добавь строковые ресурсы

Вынеси строку "Привет, мир!" в ресурсы и используй её в TextView.

Задачи

Задача 4: Создай цветовые ресурсы

Определи цвет #FF5722 в ресурсах и используй его в TextView.

Задачи

Задача 5: Переведи приложение на английский

Добавь поддержку английского языка.

Задачи

Задача 6: Форматирование строк

Добавь строковый ресурс "Привет, %s!", чтобы передавать имя пользователя в TextView.

Задачи

Задача 7: Используй LinearLayout с весами

Создай LinearLayout, где:

- Один TextView занимает 70% ширины
- Второй Button – 30%

Задачи

Задача 8: Используй ConstraintLayout

Создай ConstraintLayout, где Button будет по центру экрана.

Задачи

Задача 9: Создай кнопку с обработчиком нажатия

Добавь кнопку, которая при нажатии покажет Toast с текстом "Кнопка нажата!".

Задачи

Задача 10: EditText с вводом текста

Добавь поле EditText, кнопку и TextView, который покажет введённый текст.

Q&A

Домашнее задание

Задачи

Задача 1: Темы и стили: Изменение цвета кнопки через стиль

Создай стиль для кнопки, которая:

- Имеет фиолетовый фон #6200EE,
- Белый текст,
- Закруглённые углы 8dp.

Задачи

Задача 2: ImageView с загруженной картинкой

Добавь ImageView, который покажет картинку из ресурсов.

Q&A

Ваши вопросы

Спасибо