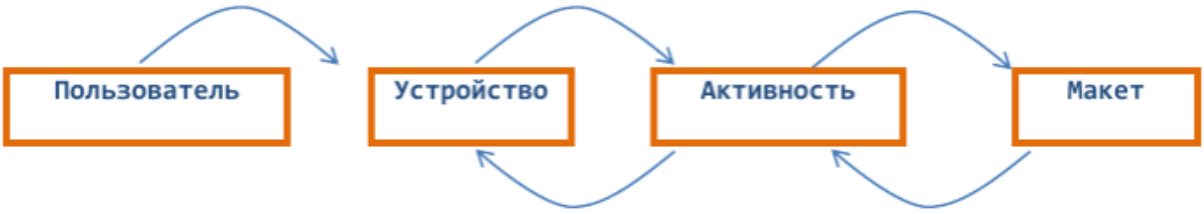
Лекция #11. Разметка ConstraintLayout

Графический интерфейс пользователя формируется с помощью объектов **View** (представление), **ViewGroup** (группа представлений). Класс **ViewGroup** является дочерним для класса **View**.

- Активность (**activity**) определяет действия
- Макет (разметка) (**layout**) – определяет способ представления интерфейса пользователя.



1. Устройство запускает приложение
2. Приложение создает объект активности
3. Активность выводит макет

4. Пользователь взаимодействует с макетом, отображаемым на устройстве
5. Активность реагирует на эти взаимодействия, выполняя код приложения

Класс **View** является основой для подклассов, которые называют виджетами. Виджеты – это элементы пользовательского интерфейса – текстовые поля, кнопки и т.д.

ViewGroup – контейнер, в который помещаются другие компоненты. Вместо термина контейнер часто используют термин макет. Это одно и то же.

Существуют несколько типов **ViewGroup**:

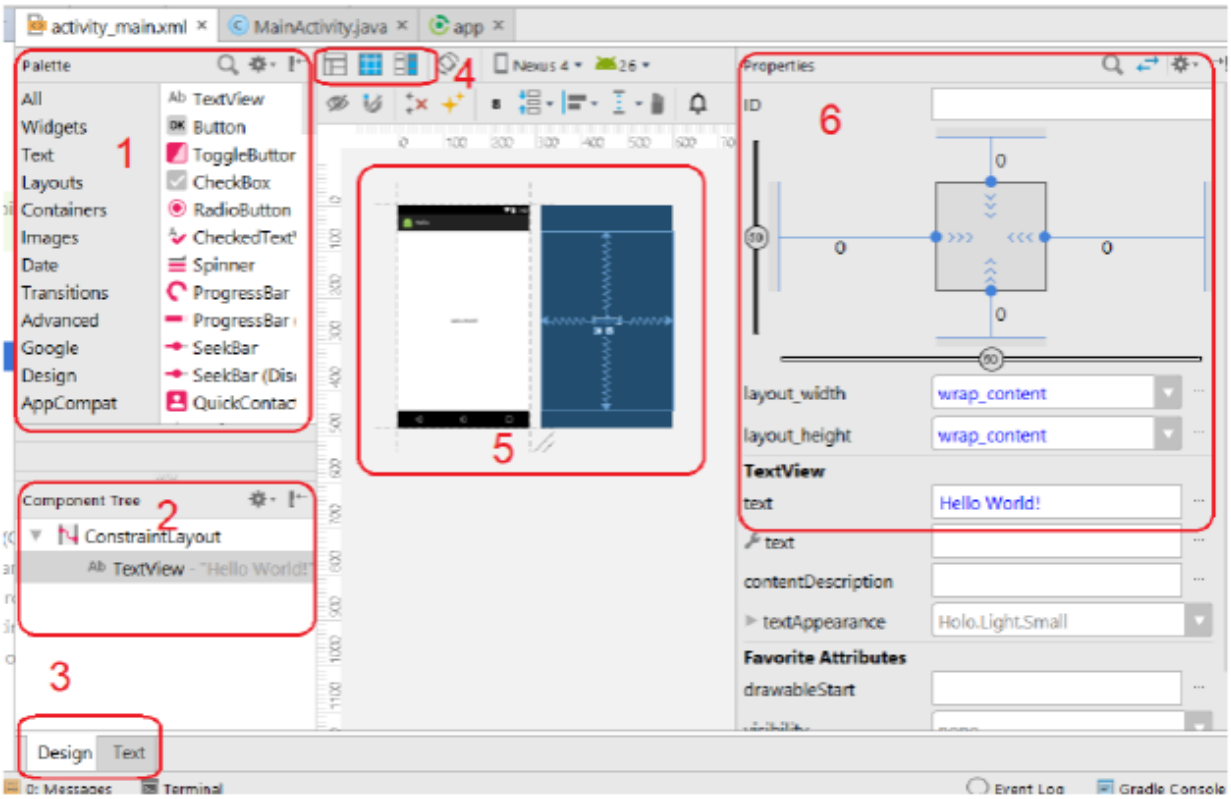
- LinearLayout,
- RelativeLayout,
- FrameLayout,
- TableLayout,
- ConstraintLayout и т.д.

Они различаются тем, как они будут упорядочивать компоненты внутри себя.

LinearLayout, например, выстроит их по горизонтальной или вертикальной линии, а **TableLayout** - в виде таблицы.

Контейнер **ConstraintLayout** появился в **Android 2.3**. Кроме того, **Android Studio** по умолчанию предлагает нам использовать **ConstraintLayout** при создании разметки экрана.

В файле **activity_main.xml** (**layout**-файл) мы определяем набор и расположение **View** компонентов, которые хотим видеть на экране.



1. Палитра компонентов
   Это список всех **View** компонентов, которые вы можете добавлять на ваш экран
2. Дерево компонентов
   Здесь представлена иерархия **View** компонентов вашего экрана. Корневым элементом является **ConstraintLayout**, в него вложен **TextView**.
3. Design и Text
   Desing - это графическое представление экрана. Оно сейчас открыто.
   Text - это текстовое представление – **xml**-разметка.
4. Режимы отображения экрана
   Здесь три кнопки
   - первая - **Design** - в нем мы видим **View** компоненты так, как они обычно выглядят на экране.
   - вторая - **Blueprint** - отображаются только контуры **View** компонентов на синем фоне
   - третья - **Design + Blueprint**
5. Экран
   Экран вашего приложения. Сюда мы будем добавлять различные компоненты.
6. Свойства
   Отображаются свойства текущего компонента. С помощью свойств

можно настраивать внешний вид, расположение и содержимое **View** компонента.

Размер и положение элементов

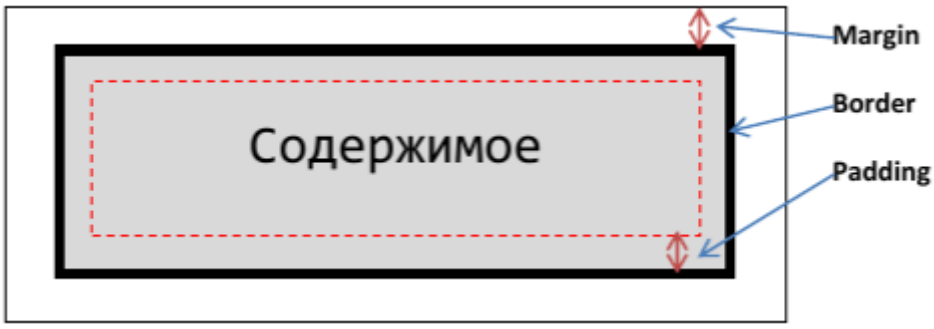
Во всех **View**, которые вкладываются в **ViewGroup**, есть атрибуты позволяющие управлять их размерами или расположением. Основные из них:

1. **android:layout_width** - ширина,
2. **android:layout_height** - высота

могут принимать следующие значения:

- **wrap_content** — элемент имеет размер по своему содержимому
- **match_parent** — элемент имеет размер такой же, как и родительский
- элемент
- **match_constraints** элемент займет пространство, доступное между
- объектами, к которым он привязан
- точное значение размера задается: **dp**, **sp**, **px**, **pt**, **in**, **mm**.

3. **Margin** – отступ и **padding** – внутренние поля
   В приведенной ниже схеме показано различие **margin** и **padding**



Разметка ConstraintLayout

Цель **ConstraintLayout** — уменьшить количество используемых в приложении разметок, улучшить производительность **layout**.

При работе с разметкой **ConstraintLayout** используется только режим **Design** - визуальный редактор проектирования интерфейса.