**Векторные элементы**

Это чертежи в формате XML, которые могут определять сложные векторные изображения, которые могут автоматически масштабироваться в зависимости от своего размера. При использовании векторных изображений вам понадобится только один файл ресурсов в отличие от растровых изображение, где вам понадобится отдельный файл ресурсов для каждого размера.

**Создание векторных чертежей**

Чтобы создать векторное изображение, вам необходимо определить синтаксис pathData, который содержит команды перемещения, линии, кривой, дуги и окончания пути (в общем координаты). В этом примере определяются детали изображения внутри XML-элемента:

<vector xmlns:android="http://schemas.android.com/apk/res/android"

android:height="256dp"

android:width="256dp"

android:viewportWidth="32"

android:viewportHeight="32">

*<!-- draw a path -->*

<path android:fillColor="#c9c10606"

android:pathData="M20.5,9.5

c-1.955,0,-3.83,1.268,-4.5,3

c-0.67,-1.732,-2.547,-3,-4.5,-3

C8.957,9.5,7,11.432,7,14

c0,3.53,3.793,6.257,9,11.5

c5.207,-5.242,9,-7.97,9,-11.5

C25,11.432,23.043,9.5,20.5,9.5z" />

</vector>



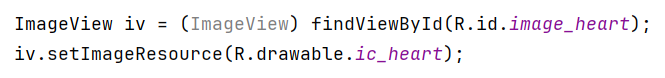
Затем вы можете использовать векторы, с помощью атрибутов app:srcCompat или android:src:

Обязательно используйте app:srcCompat для поддержки старых устройств Android. Если вы используете android:src для векторного рисования, ваше приложение может некорректно отображаться на устройствах, выпущенных до версии Lollipop.





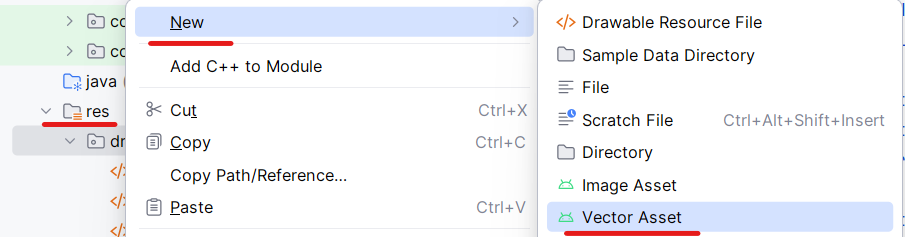
Вы также можете задать вектор в программном коде с помощью:



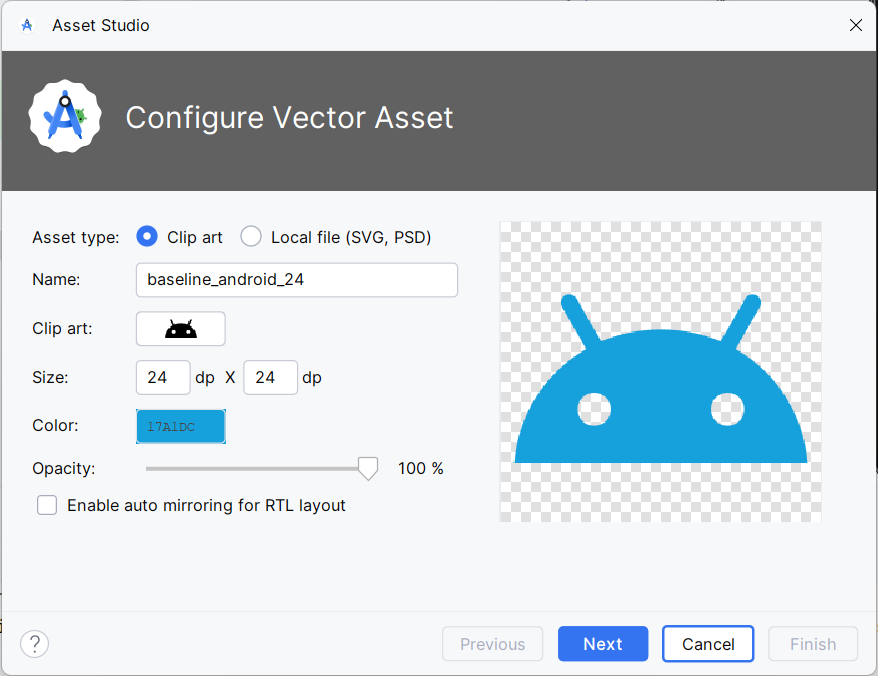
**Создание векторных изображений**

Векторные изображения можно создавать непосредственно в Android Studio.

Создайте новое векторное изображение. Чтобы создать файл, нажмите: app -> res -> new -> vector asset.



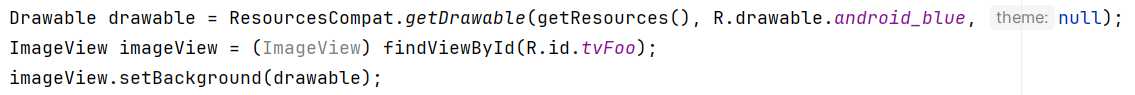
В открывшемся окне вы можете изменить его название, выбрать иконку из предустановленных из библиотеки Material Design, изменить размер и цвет.



Дополнительные иконки можно найти на [веб-сайте material design icons](https://pictogrammers.com/library/mdi/), который также включает иконки, созданные сообществом.

**Использование Drawable в программном коде.**

Вы можете получить доступ к drawable во время выполнения в Java-коде, обратившись к элементу, к которому применен чертеж.

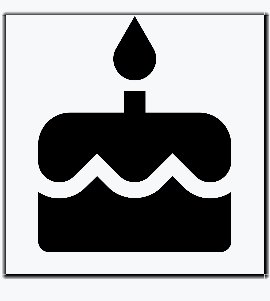


Вы можете получить доступ к drawable во время выполнения в Java-коде, обратившись к элементу, к которому применен чертеж.

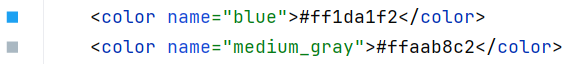
**Применение оттенков**

К drawable можно применять оттенок (hint). Это можно использовать чтобы изменять цвет drawable например при изменении темы приложения.

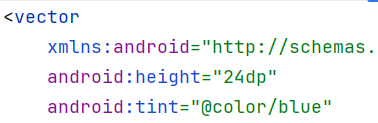
Создайте векторное изображение черного цвета на ваше усмотрение:

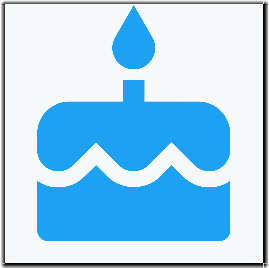


Добавьте пару цветов в colors.xml:

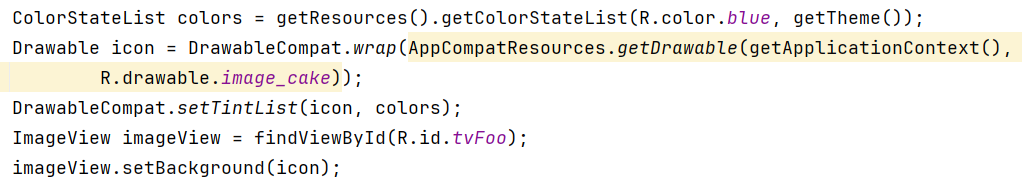


Первый способ это изменить оттенок в файле drawable с помощью атрибута android:tint:





Вы также можете изменить цвет drawable программно:

****

Также вы можете устанавливать отдельные цвета для разных состояний объекта, для этого создайте в папке res подпапку color, в папке color создайте файл:



**Создание вручную простых векторных элементов в xml**

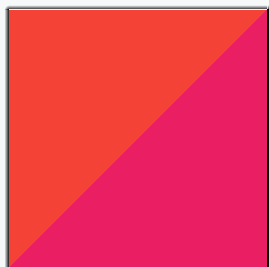
VectorDrawable - это XML представление вектора. В отличие от популярных графических форматов, таких как Bitmap, JPEG, GIF и PNG, векторные изображения не теряют качества при увеличении или уменьшении масштаба.

В многих форматах используется метод объединения множества изображений с разным размером в один файл что позволяет удобно изменять размер изображения но увеличивает объем итогового файла программы. VectorDrawables содержат команды path on (как рисовать линии и дуги). Процесс рисования VectorDrawables отнимает много времени и памяти, поэтому VectorDrawable лучше всего использовать для плоской графики.

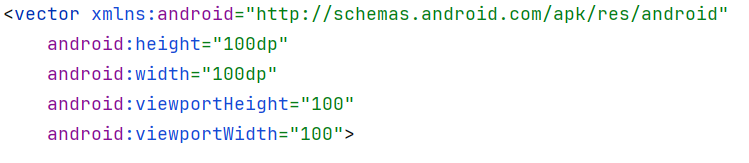
Понимание path команд может быть необходимым для дальнейшего анимирования векторных изображений.

**Основы**

Для примера создадим прямоугольник состоящий из двух соседних треугольников



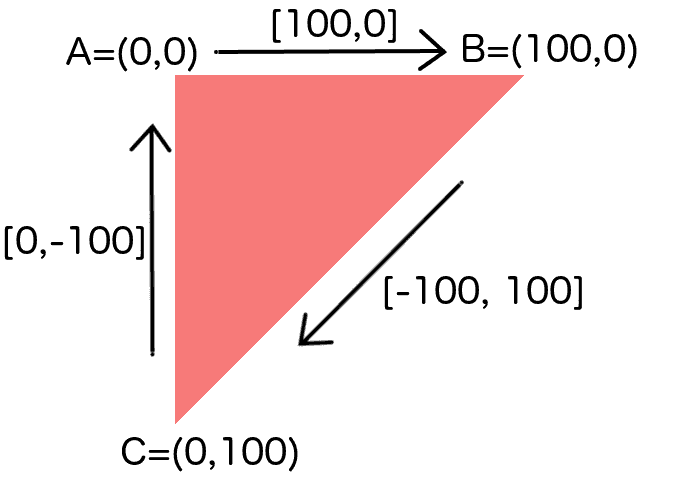
Создайте новый векторный файл и настройте ему размерность:



Свойства android:width и android:height определяют внутренний размер drawable, который используется только если в элементе в котором будет размещен drawable значением ширины и высоты будет wrap\_content.

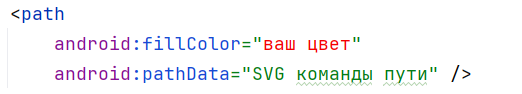
Свойства android:viewportWidth и android:viewportHeight определяют текущий размер холста для рисования контуров.

Давайте заполним половину квадратного холста первым треугольником, проведя контур через его вершины: A, B и C.



Обратите внимание на координаты — верхний левый угол холста равен (0,0), а нижний правый угол равен (100,100), потому что размер элемента равен android:viewportWidth="100" и android:viewportHeight="100".

Вы можете нарисовать вектор, добавив контур внутри элемента:



Основные команды path состоят из алфавита, за которым следует одно или несколько чисел.

Числа можно разделять запятыми, но это необязательно.

M100,100 L300,100 L200,300 z

или

M 100 100 L 300 100 L 200 300 z

или

M1100,100L300,100L200,300z

Алфавит может быть заглавным или строчным. Заглавный означает абсолютное положение, строчный - относительное.

**Команды**

M или m (X, Y) : переместить курсор в нужное положение (move to), верхний регистр - абсолютный, нижний - относительный. По сути это начальная точка от которой начинается рисование За командой moveto следуют координаты X, Y. После команды M может быть более одного набора координат, они обрабатываются как неявные команды lineto.

L или l (X, Y) : рисует линию от текущей позиции до позиции (line to), указанной X, Y. После команды L может быть более одного набора координат, это означает, что вы создаете ломаную линию.

Z или z : рисует линию от текущего положения курсора до начальной позиции пути (closepath). Не имеет никаких параметров.

H или h (X): Horizontal lineto рисует горизонтальную линию от текущего положения курсора до положения, указанного параметром X. Если после команды следует несколько координат X, это обрабатывается как ломаная линия. Координата Y остается неизменной.

V или v (Y): Vertical lineto рисует вертикальную линию от текущего положения курсора до положения, указанного символом Y. Если после команды следует несколько координат Y, это обрабатывается как ломаная линия. Координата X остается неизменной.

Построение треугольника состоит из:

* начинаем путь с A=(0,0): M 0,0
* проводим линию до B=(100,0): L 100,0
* проводим линию до C=(0,100): L 0,100
* закрываем путь: z

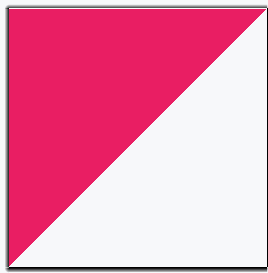
Так же вместо этого вы могли бы использовать команду относительного lineto (l) и указать векторы, соединяющие точки следующим образом:

* A до B: l 100,0
* B до C: l -100,100

Если вы используете одну и ту же команду несколько раз подряд, буква команды может быть удалена в последующих командах. Таким образом, итоговая команда выглядит следующим образом:

M 0,0 L 100,0 0,100 z





Нарисуйте второй треугольник, его вершины равны:

B=(100,0)

C=(0,100)

D=(100,100)

Путь выглядит следующим образом:

M 100,0 L 0,100 100,100 z

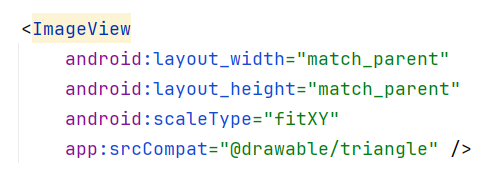
Вы можете присвоить контурам имена, чтобы их было легче найти позже, если ваш рисунок усложнится.





Установите drawable в ImageView

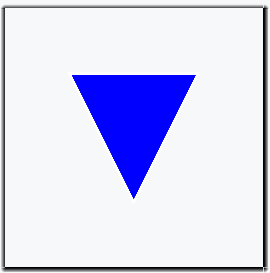
Вы можете попробовать масштабировать drawable, используя свойства android:layout\_width, android:layout\_height и android:scaleType.





**Задание для самостоятельной работы:**

1. Разработайте drawable файл соответствующий следующему макету.



1. Разработайте drawable файл соответствующий следующему макету.

