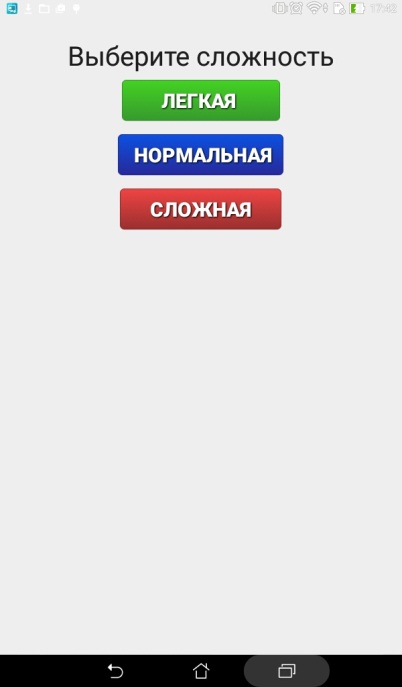
Глава 2.

Для написания программы решено было выбрать среду разработки Android studio, так как эта среда имеет большой функционал и удобный интерфейс. Открыв среду разработки мы сразу видим наш начальный экран. Он состоит из файла разметки – activity\_main.xml и из программной части – MainActivity. Первое что нам ну сделать, это добавить несколько кнопок в activity\_main – будущие сложности нашего пазла. Чтобы по нажатию на кнопки происходило какое-нибудь действие надо сначала объявить их в MainActivity.

Button button1 = (Button) findViewById(R.id.***button1***);

А затем присвоить им обработчик нажатия onClickListner.

button1.setOnClickListener(**onClickListener**);

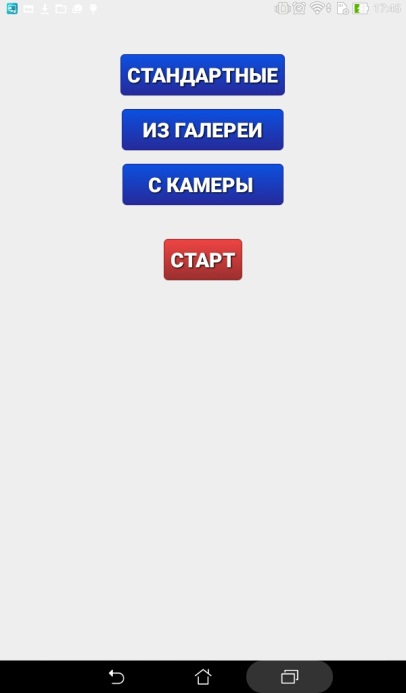
В обработчике нажатия, в зависимости от выбранной кнопки, мы создаем намерение (intent), а затем запускаем его. Это намерение будет переходить от главного экрана к экрану выбора картинки и самому пазлу.

Intent intenteasy = **new** Intent(MainActivity.**this**, easy.**class**);

startActivity(intenteasy);

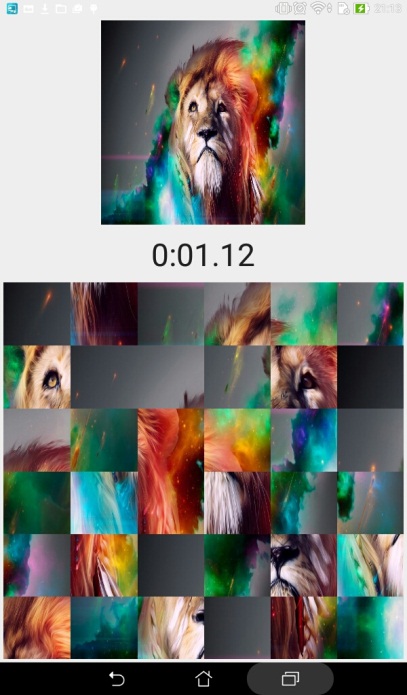
Чтобы иметь возможность переходить к другому экрану, нужно его создать. Нам нужно создать два файла. Файл разметки и activity. Созданное activity необходимо записать в файле манифеста. Делаем это так:

<**activity  
 android:name=".easy"  
 android:label="@string/app\_name"  
 android:theme="@style/AppTheme.NoActionBar"** >  
</**activity**>

В манифесте можно указать некоторые другие параметры, но в данном случае нам этого достаточно. В файле разметки (activity\_easy в данном случае) мы создаем 4 кнопки – выбор картинки из стандартного набора, галереи, фото с камеры и кнопка старта. Также нам необходимо создать несколько ImageView, в которых в дальнейшем будут находиться наши картинки. Нам потребуется 1 основное ImageView, в котором будет находиться целая картинка, а также места для нашего пазла. Например легкий режим моего приложения – пазл 4x4, нормальный - 6х6 и т.д. Поэтому для легкого режима мне потребуется дополнительных 16 ImageView, для нормального – 36 и т.д. Для удобства разместим их в таблице. Чтобы картинки занимали все предоставленное им место им надо задать ширину и высоту

**android:layout\_width="match\_parent"  
android:layout\_height="match\_parent"**

Для строк таблицы лучше задать не “match\_parent”, а “wrap\_content”.

 Чтобы картинки были одинаковы по размеру зададим им и строкам таблицы (TableRow) “вес”

**android:layout\_weight="1"**

Теперь картинки равномерно займут весь объем таблицы.

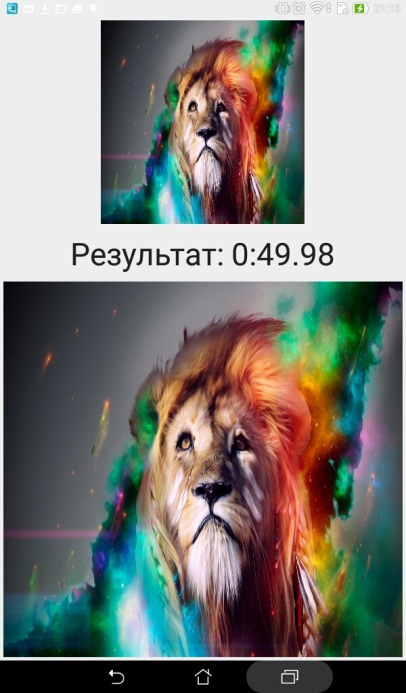
Перейдем к нашему activity. Для начала нам нужно выбрать layout файл, который будет загружаться при запуске данного activity.

setContentView(R.layout.***activity\_easy***);

Там же в onCreate мы определяем наши кнопки и компоненты ImageView. Для ImageView надо создать двумерный массив (imageView[4][4]). В данном случае лучше будет объявить массив не в onCreate, как мы делали с кнопками в MainActivity, а вне всяких методов нашего activity, чтобы иметь возможность использовать их не только в методе onCreate (который срабатывает только единожды при старте нашего activity), но и, например, в onClickListner. Кнопкам мы присваиваем обработчик нажатия. При нажатии на первую кнопку мы переходим в другое activity, с выбором картинок из стандартных, при нажатии на вторую кнопку мы переходим в галерею и выбираем фото или картинку, а при нажатию на третью кнопку, включается режим камеры и мы можем сделать фото, которое сразу поступит в наше приложение. Наша четвертая кнопка будет разрезать нашу основную фотографию на кусочки, создавать “правильную” матрицу картинок trueimageView[4][4], перемешивать их, а также делать кнопки выбора картинок и саму кнопку старта невидимыми.

Для самих картинок необходимо создать метод обработки касания onTouchListner. У этого метода есть 4 действия: ACTION\_DOWN, ACTION\_MOVE, ACTION\_UP, ACTION\_DOWN. Когда мы касаемся экрана, onTouchListner обрабатывает действие ACTION\_DOWN. В нем мы определяем номера столбца (starti) и строки (startj) того элемент нашего пазла, которого мы коснулись, и получаем из него картинку. Полученную картинку мы записываем в отдельный объект drawable, также нам необходим массив с координатами х и у для каждого компонента.

imageViewX[i][j]= j\***imageView**[i][j].getWidth();  
**imageViewY**[i][j]= i\***imageView**[i][j].getHeight();

Благодаря этому, прибавляя к этим координатам координаты перемещения, мы сможем определять в действии ACTION\_UP на каком ImageView находится наш палец. ACTION\_MOVE мы оставляем пустым. В действии ACTION\_UP мы обрабатываем события на убирание пальца с экрана. Нам надо поменять местами мы картинку, на которую мы нажали, и картинку, с которой мы отпустили. Для этого мы снова определяем на какой элемент нашего пазла мы нажали, получаем номера его столбца (i) и строки (j). После этого мы присваиваем объекту imageView[starti][startj] картинку из imageView[i][j], а объекту imageView[i][j] мы присваиваем ранее сохраненную в drawable картинку (если мы опустили и отпустили палец на одной и той же картинке, то она поменяется местами сама с собой, при этом визуально ничего не произойдет). После того, как мы поменяли картинки местами надо проверить, собралась ли готовая картинка и игра завершилась или нужно продолжать. Мы сравниваем в цикле картинки imageView[i][j] и trueimageView[i][j]. Если обнаруживается несовпадение картинок, цикл прекращается, а если цикл дошел до последних элементов и они оказались равны, то объявляется победа.