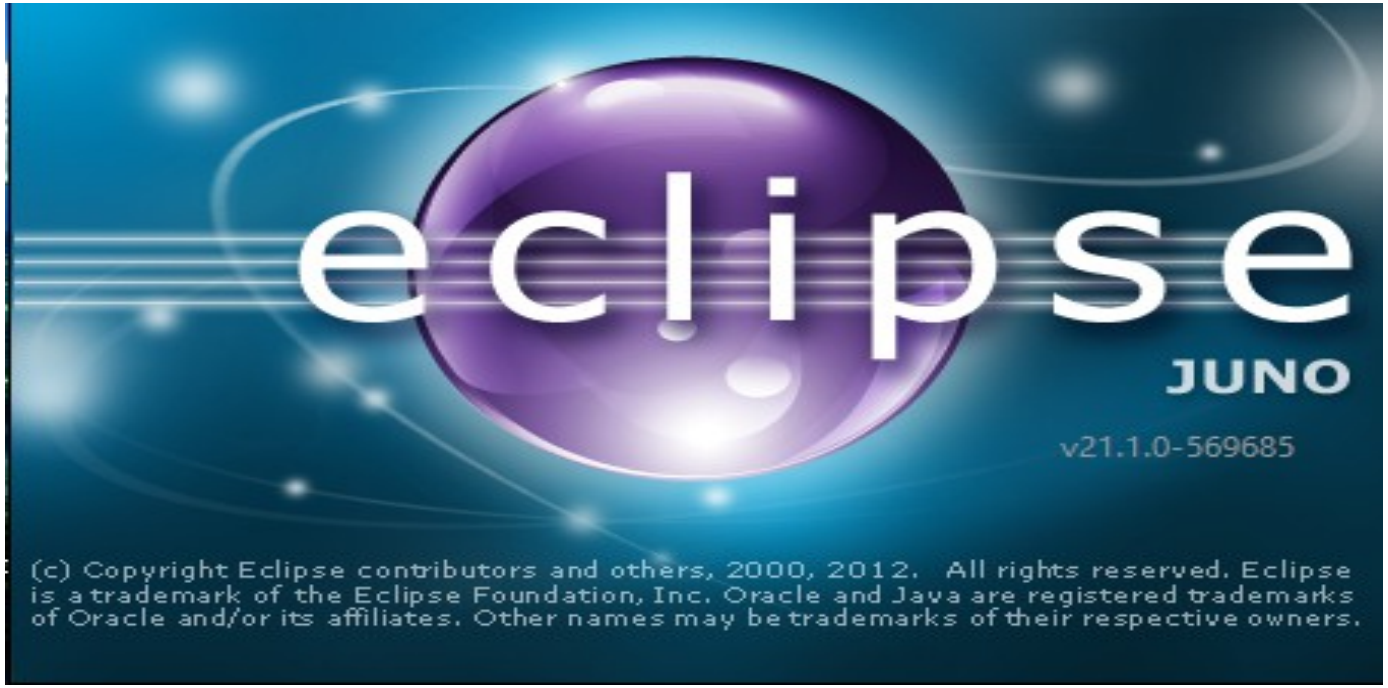
Основы программирования под Android с примерами

Дмитрий Кузьмичев, ЛКШ.Август 2013

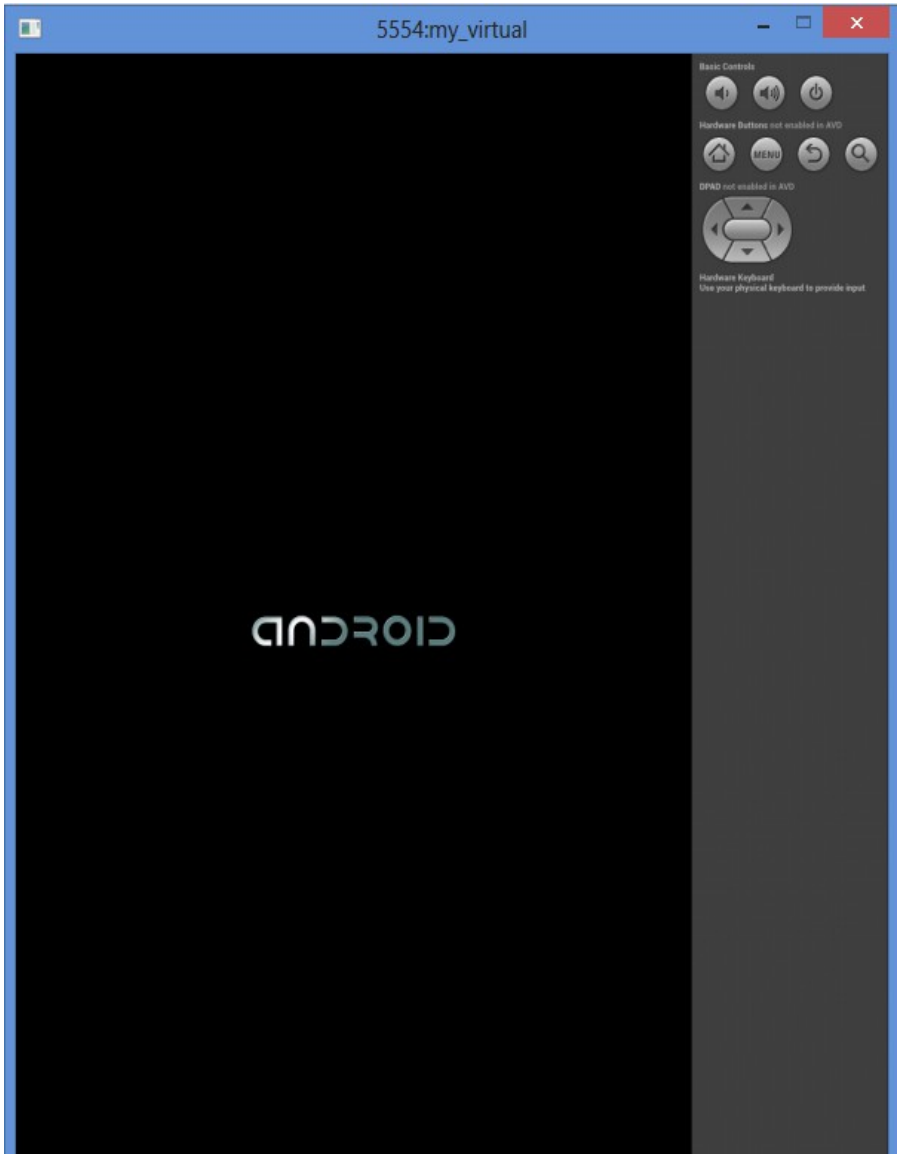
Используемый язык и среда разработки

Мы будем писать на Java, а для этого языка Google рекомендует Eclipse с плагином Android Development Tools(ADT). Им и будем пользоваться



Что такое virtual device и зачем он нужен?

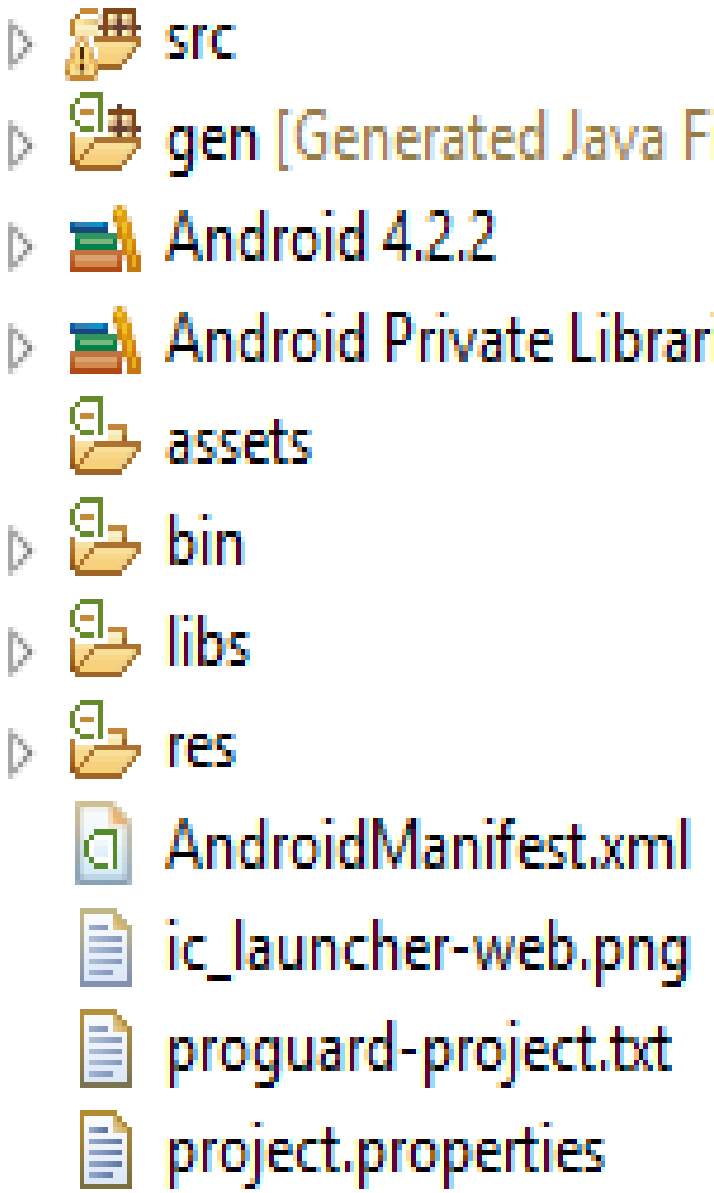
Не обязательно иметь под рукой планшет или телефон с Андроидом, чтобы проверить работу своей программы. Можно создать Android Virtual Device (AVD), который будет эмулировать работу реального устройства



Пишем первое приложение (Hello World)

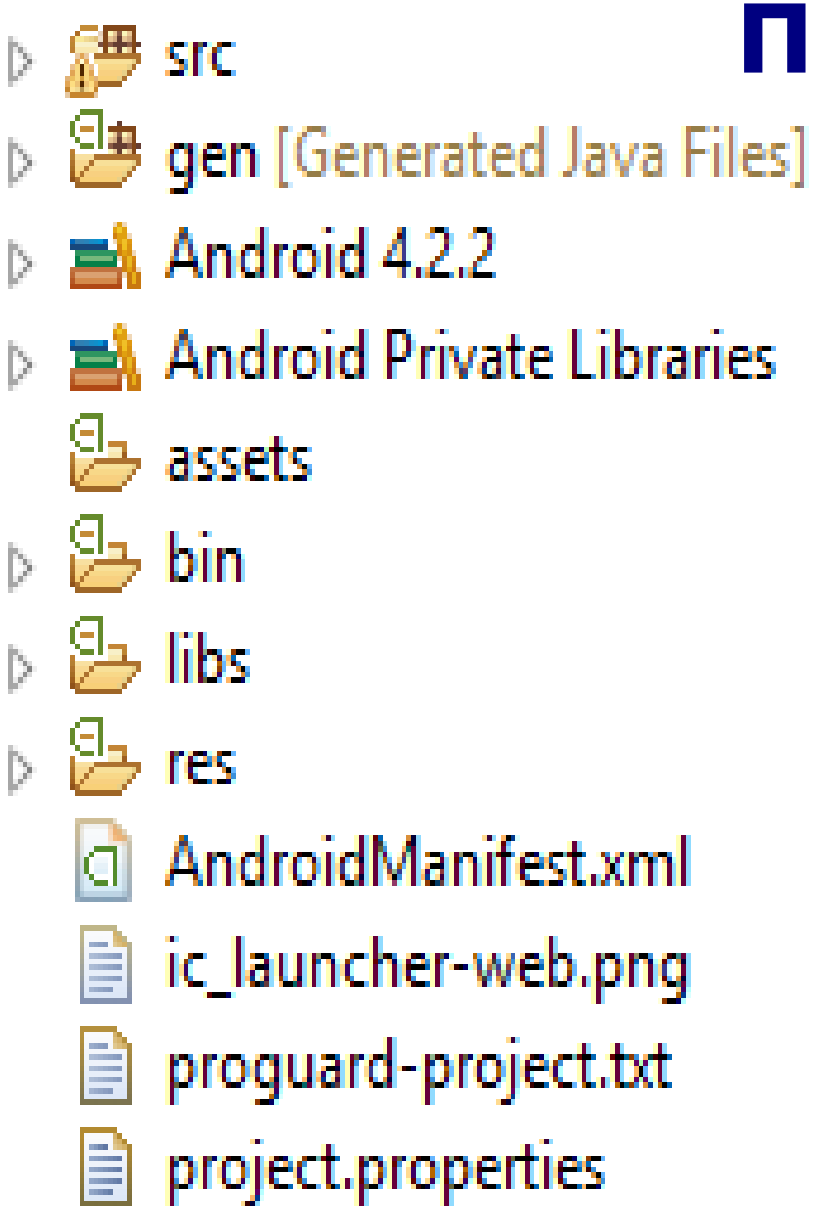
Давайте напишем первое приложение под Android. Оно будет представлять из себя просто экран с надписью Hello World!

Важные компоненты проекта



- **src** — папка, где хранятся все наши исходные коды (например, MainActivity.java)
- **gen** — файлы, генерируемые средой. Необходимы для работы приложения, лучше не трогать!

Важные компоненты проекта



- **Android 4.2.2** – стандартные библиотеки
- **assets** и **res** – папки для файлов-ресурсов
- **AndroidManifest.xml** – конфигурационный файл приложения

Код Hello World

1) package com.example.HelloWorld –

просто название проекта

2)import android.os.Bundle;

import android.app.Activity;

import android.view.Menu;

Подключаемые компоненты(для обновления нажать Ctrl+Shift+O)

3) public class MainActivity extends Activity {

...

}

Объявление класса MainActivity, который наследует класс Activity. Это значит, что 1) принимается функционал Activity, 2) мы сможем расширить/улучшить этот функционал

Код Hello World

4)@Override

```
protected void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
}
```

Метод onCreate класса MainActivity, вызывающийся при создании. Запускает аналогичный метод для класса, от которого наследовались(в нашем случае Activity). SetContentView устанавливает нужный layout в качестве содержимого. У нас выбирается единственный layout под названием activity_main. О layout-файлах будет рассказано ниже

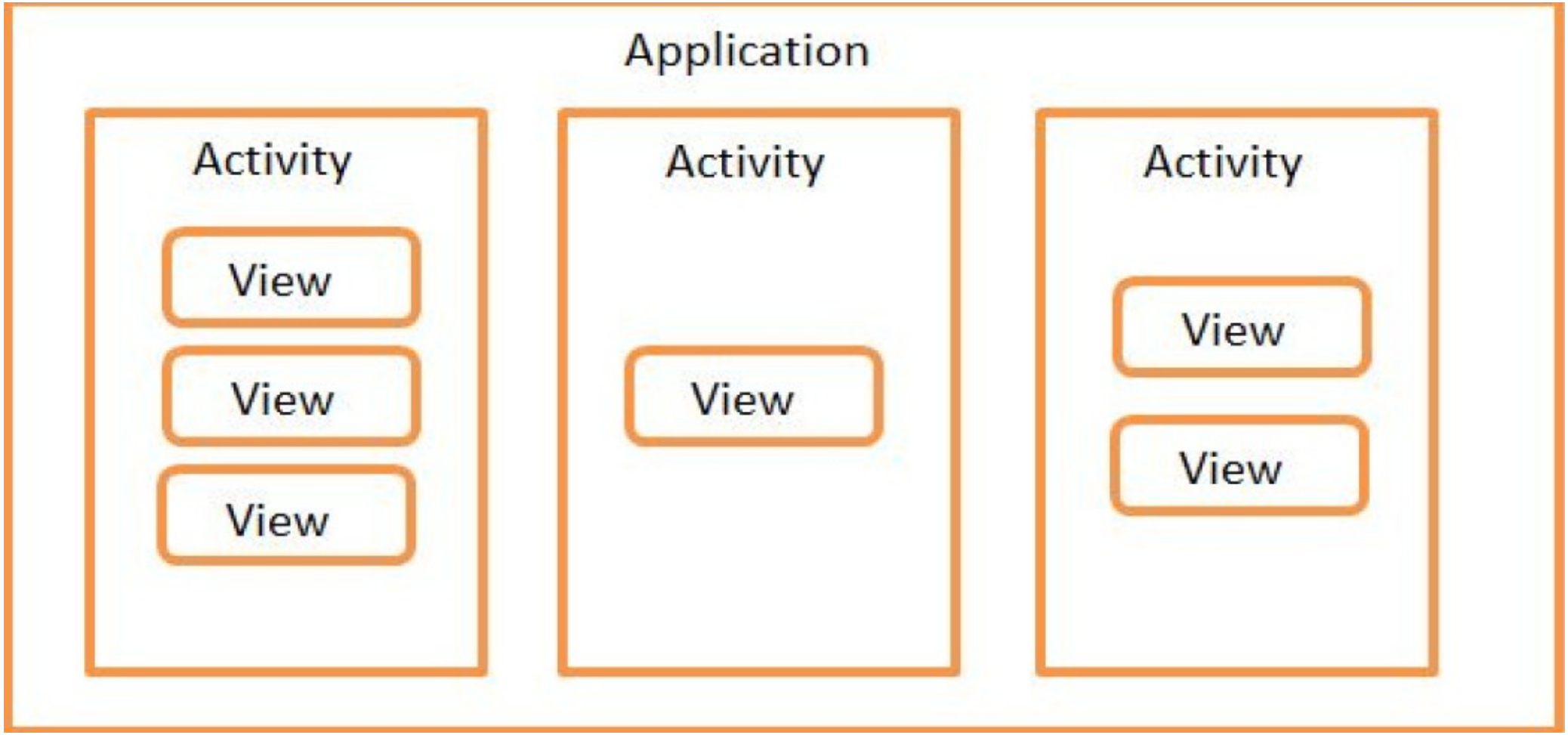
Код Hello World

5)@Override

```
public boolean onCreateOptionsMenu(Menu menu) {

getMenuInflater().inflate(R.menu.main, menu);

return true;

}
```

Метод, отвечающий за создание пунктов меню. Пока не очень нам важен, поскольку с меню мы пока не работаем.

Приложение состоит из нескольких Activity, а Activity - из различных View



Layout-файлы

В Layout-файлах задается размещение View на экране, их взаимное расположение. При запуске приложения, Activity читает эти файлы и отображает нам то, что мы настроили. Layout имеют расширение xml(eXtensible Markup Language), хотя Eclipse позволяет редактировать и просматривать их более удобным способом.

Создание другого layout-файла для использования

Давайте создадим другой layout-файл (назовем его new_screen.xml), и в него добавим TextView с нашим текстом, чтобы отличить его от activity_main. Заметим, что соответствующий файл появился в res/layout, а также в R.java, специальном файле, который хранит идентификаторы для разных частей приложения

Что такое обработчик нажатия и зачем он нужен?

Обработчик нажатия вызывается при нажатии на компонент, и позволяет на это нажатие среагировать. Как именно — решаем мы. Что нужно сделать:

1) Нам нужно завести переменные для TextView и двух кнопок(Button):

```
TextView tvOut;
Button btnOk;
Button btnCancel;
```

Использование обработчиков нажатий

2) в onCreate мы должны инициализировать эти переменные с помощью функции findViewById, которая возвращает компонент по его id.

```
tvOut = (TextView)
findViewById(R.id.tvOut);
btnOk = (Button)
findViewById(R.id.btnOk);
btnCancel = (Button)
findViewById(R.id.btnCancel);
```

Использование обработчиков нажатий

3) Заставим наш класс MainActivity выполнять обработку нажатия:

```
public class MainActivity extends Activity implements OnClickListener
```

4) Теперь в onCreate «сообщим» кнопкам, что MainActivity обрабатывает нажатия на них:

```
btnOk.setOnClickListener(this);
btnCancel.setOnClickListener(this);
```

Использование обработчиков нажатий

5) Напишем пока пустой метод onClick, который будет вызываться при нажатии.

```
@Override
public void onClick(View v) {

}
```

5) Определим, что именно будет происходить при нажатии на кнопки. Если была нажата кнопка OK, то в TextView выведем «Нажата кнопка OK», иначе «Нажата кнопка Cancel». Чтобы проверить это возьмем id кнопки.

Использование обработчиков нажатий

Чтобы установить текст в TextView, мы используем его метод setText:

```
if (v.getId() == R.id.btnOk)
    tvOut.setText("Нажата кнопка OK");
else tvOut.setText("Нажата кнопка Cancel");
```

Логи

С помощью логов мы сможем отслеживать, что происходит в написанном нами приложении, т. е. по сути они используются для дебага.
Работу с логами осуществляет класс Log. Его методы Log.v(), Log.d(), Log.i(), Log.w(), Log.e() соответствуют уровням важности выводимых логов Verbose, Debug, Info, Warn, Error соответственно. Каждый из этих методов принимает 2 параметра — тэг и текст самого сообщения. Тэг — это метка, которая будет служить для отделения наших логов от всех других(например, системных)

Использование логов

1) заведем строчку, в которой будет написано название для нашего тэга

```
private static final String TAG = "myLogs";
```

2) теперь просто в нужных местах будем писать что-то вроде

```
Log.d(TAG, "присваиваем обработчик кнопкам");
```

или

```
Log.d(TAG, "по id определеяем кнопку, вызвавшую этот обработчик");
```

Всплывающие сообщения

Всплывающие сообщения — это **инструмент для показа пользователю** какой-то **информации** или **уведомлений**. Также они могут использоваться для отладки приложения, хоть и не так удобны, как логи. За такие сообщения отвечает класс **Toast** и его метод **makeText**, который принимает:

1) **контекст** — пока нам не важно, что это такое, передаем this
2) **сам текст** для вывода
3) **продолжительность показа** (2 варианта — Toast.LENGTH_LONG и Toast.LENGTH_SHORT)

Всплывающие сообщения

Пишем так:

```
Toast.makeText(this, "Нажата
кнопка Cancel",
Toast.LENGTH_LONG).show();
```

Здесь this – контекст, «Нажата кнопка Cancel» - текст, Toast.LENGTH_LONG – продолжительность, и show() - метод, который заставляет сообщение показаться

Контекстное меню

Контекстное меню вызывается длительным нажатием на экранный компонент и служит тем же целям, что и контестное меню в других ОС. Итак, что мы делаем, чтобы его использовать:

1) Объявляем переменные для TextView и инициализируем их

```
TextView tvColor, tvSize;
tvColor = (TextView)
findViewById(R.id.tvColor);
tvSize = (TextView)
findViewById(R.id.tvSize);
```

Контекстное меню

2) Указываем, что нам нужно для них контекстное меню

```
registerForContextMenu(tvColor);
registerForContextMenu(tvSize);
```

3) Будем использовать **константы** для хранения **ID** пунктов меню

```
final int MENU_COLOR_RED = 1;
final int MENU_COLOR_BLUE = 2;
final int MENU_SIZE_22 = 3;
final int MENU_SIZE_30 = 4;
```

Контекстное меню

4) Пишем метод onCreateContextMenu, пока пустой

```
@Override
public void onCreateContextMenu(ContextMenu menu, View v, ContextMenuInfo menuInfo) {
}
```

5) В реализации этого метода мы сначала проверяем, с каким компонентом мы имеем дело. В зависимости от этого мы добавляем пункты меню либо относящиеся к цвету, либо к размеру. Для этого используем menu.add(), который принимает 4 параметра: ID группы, ID пункта, позиция пункта меню, текст, который будет показан

Контекстное меню

```
if (v.getId() == R.id.tvColor)
{
   menu.add(0, MENU_COLOR_RED, 0, "Red");
   menu.add(0, MENU_COLOR_BLUE, 0, "Blue");
}
else
{
   menu.add(0, MENU_SIZE_22, 0, "22");
   menu.add(0, MENU_SIZE_30, 0, "30");
}
```

6) Реализуем обработку нажатий на пункты меню. Для этого смотрим, пункт с каким ID был выбран, и меняем либо цвет текста в tvColor, либо размер в tvSize. Начнем писать реализацию метода onContextItemSelected:

```
@Override
public boolean onContextItemSelected(MenuItem item)
{
   ...
   return super.onContextItemSelected(item);
}
```

Контекстное меню

```
switch(item.getItemId())
{
   case MENU_COLOR_RED:
   {
      tvColor.setTextColor(Color.RED);
      tvColor.setText("color = red");
      break;
   }
   case MENU_COLOR_BLUE:
   {
      tvColor.setTextColor(Color.BLUE);
      tvColor.setText("text color = blue");
      break;
   }
```

Контекстное меню

```
case MENU_SIZE_22:
{
  tvSize.setTextSize(22);
  tvSize.setText("text size = 22");
  break;
}
case MENU_SIZE_30:
{
  tvSize.setTextSize(30);
  tvSize.setText("text size = 30");
  break;
}
```

Summary (Итог)

Итак, из этого спецкурса вы узнали:

1) Что такое Eclipse и как использовать Java для разработки под Android
2) Что такое android virtual device
3) Как написать Hello World для Android
4) Из каких основных компонентов состоят проекты для Android
5) Что такое Activity, View, Layout и зачем они нужны
6) Как реализовать обработчик нажатий
7) Как дебажить под Android
8) Как реализовать контестное меню

Спасибо за внимание,
продолжайте
изучать
программирование!