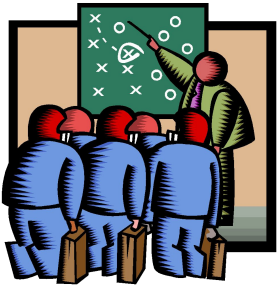


Урок 4





План на урок 4

- Разбор ДЗ
- Instance state vs Nonconfiguration instance state
- Hierarchy viewer и профилирование GUI
- Оптимизация UI - lint
- Layout xml's – include and merge
- Работа с диалоговыми окнами в Android
- Работа с меню

- Что нужно сделать к уроку 5

Instance state.

- Instance state это 'состояние экземпляра'
- **Сохраняется когда:** система удаляет activity(изменения конфигурации и т.п.)
- Под изменением конфигурации понимают например поворот экрана.
- **Не сохраняется когда:** вызывается метод finish(нажатие кнопки back)
- onSaveInstanceState(Bundle outState)
- onRestoreInstanceState(Bundle savedInstanceState)



- AlexandraHajtlLesson2
- DealList
- FifteenActivity
- FilipovVladimirLesson2
- javaBaseCourseEclipse
- LifeCycleExplorer
 - Android 4.1
 - src
 - gen (Generated Java Files)
 - assets
 - bin
 - res
 - AndroidManifest.xml
 - project.properties
 - PetrovigorLesson3

```
@Override
protected void onResume() {
    super.onResume();
    numResumes.setText(String.valueOf(count));
    count++;
}

@Override
protected void onRestoreInstanceState(Bundle savedInstanceState) {
    if ((savedInstanceState != null)
        && savedInstanceState.containsKey(COUNT_KEY)) {
        count = savedInstanceState.getInt(COUNT_KEY);
    }
    Toast.makeText(this, "onRestoreInstanceState called", Toast.LENGTH_LONG).show();
    super.onRestoreInstanceState(savedInstanceState);
}

@Override
protected void onSaveInstanceState(Bundle outState) {
    outState.putInt(COUNT_KEY, count);
    Toast.makeText(this, "onSaveInstanceState called", Toast.LENGTH_LONG).show();
    super.onSaveInstanceState(outState);
}
}
```

- ru.progschool.android.les
- import declarations
- Activity3
 - COUNT_KEY : String
 - numResumes : TextV
 - count : int
 - onCreate(Bundle) : v
 - onResume() : void
 - onRestoreInstanceSt
 - onSaveInstanceState

Android

```
[2013-03-06 21:34:38 - LifeCycleExplorer] Success!
[2013-03-06 21:34:38 - LifeCycleExplorer] Starting activity ru.progschool.android.lesson3.lifecycle.MainActivity on
[2013-03-06 21:34:38 - LifeCycleExplorer] ActivityManager: Starting: Intent { act=android.intent.action.MAIN cat=[a
```

Nonconfig Instance state.

- Nonconfig Instance state это 'состояние экземпляра'. Применяется только к текущему экземпляру и тому, который создается сразу после уничтожения первого.
- Можно передавать **plain java objects**
- `onRetainNonConfigurationInstance()`
- `getLastNonConfigurationInstance()`

```
@Override  
public Object onRetainNonConfigurationInstance() {  
    return new Date();  
}
```

File Edit Run Source Refactor Navigate Search Project Window Help

Package Explorer

- AlexandraHajtlLesson2
- DealList
- FifteenActivity
- FilipovVladimirLesson2
- javaBaseCourseEclipse
- LifeCycleExplorer
 - Android 4.1
 - src
 - gen [Generated Java Files]
 - assets
 - bin
 - res
 - AndroidManifest.xml
 - project.properties
 - PetrovigorLesson3

```
private TextView numResumes;
private int count;

@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity3);
    numResumes = (TextView) findViewById(R.id.numResumes);

    Date date = (Date) this.getLastNonConfigurationInstance();
    if(date != null) {
        Toast.makeText(this, date.toString(), Toast.LENGTH_LONG).show();
    }

    @Override
    protected void onResume() {
        super.onResume();
        numResumes.setText(String.valueOf(count));
        count++;
    }

    @Override
    protected void onRestoreInstanceState(Bundle savedInstanceState) {
```

Outline

- ru.progschool.android.les
- import declarations
- Activity3
 - COUNT_KEY : String
 - numResumes : TextV
 - count : int
 - onCreate(Bundle) : v
 - onResume() : void
 - onRestoreInstanceSt
 - onSaveInstanceState
 - onRetainNonConfig

Problems Javadoc Declaration Console LogCat

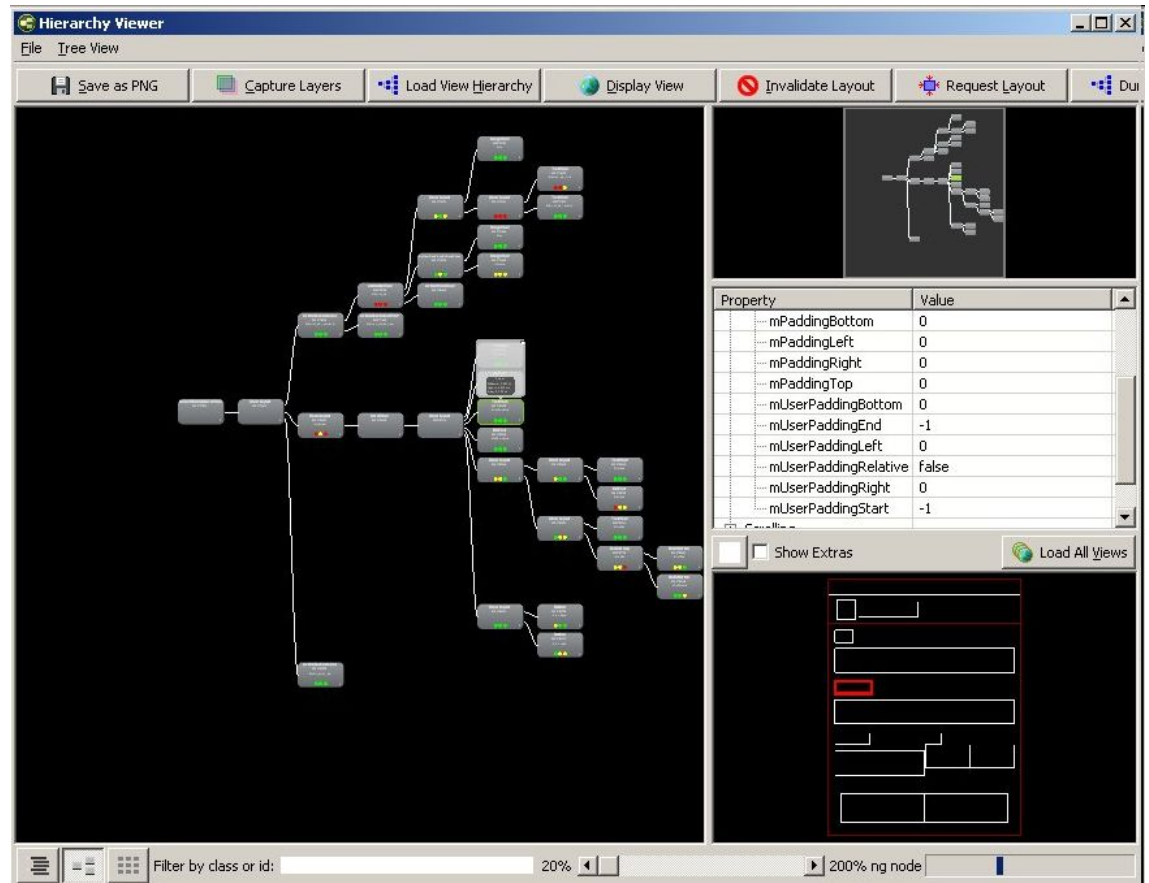
Android

```
[2013-03-06 21:34:38 - LifeCycleExplorer] Success!
[2013-03-06 21:34:38 - LifeCycleExplorer] Starting activity ru.progschool.android.lesson3.lifecycle.MainActivity on
[2013-03-06 21:34:38 - LifeCycleExplorer] ActivityManager: Starting: Intent { act=android.intent.action.MAIN cat=[a
```

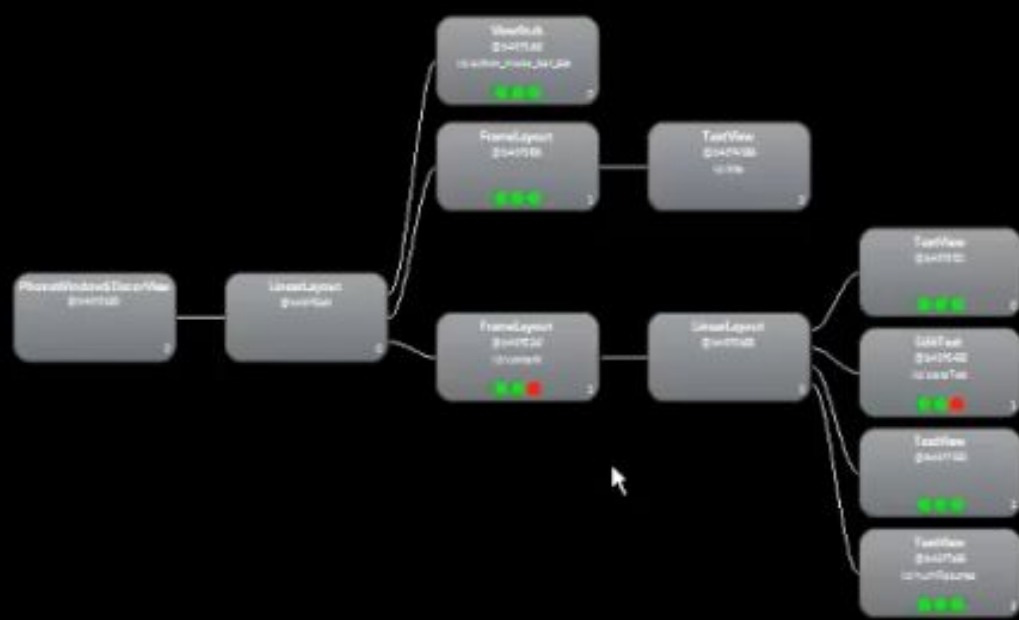
Writable Smart Insert 24 : 49 Sign in to Google

Hierarchy viewer

- Позволяет понять структуру GUI запущенного приложения и определить слабые места – дублирование, низкая производительность и т.д.



Save as PNG Capture Layers Load View Hierarchy Display View Invalidate Layout Request Layout Dump DisplayList



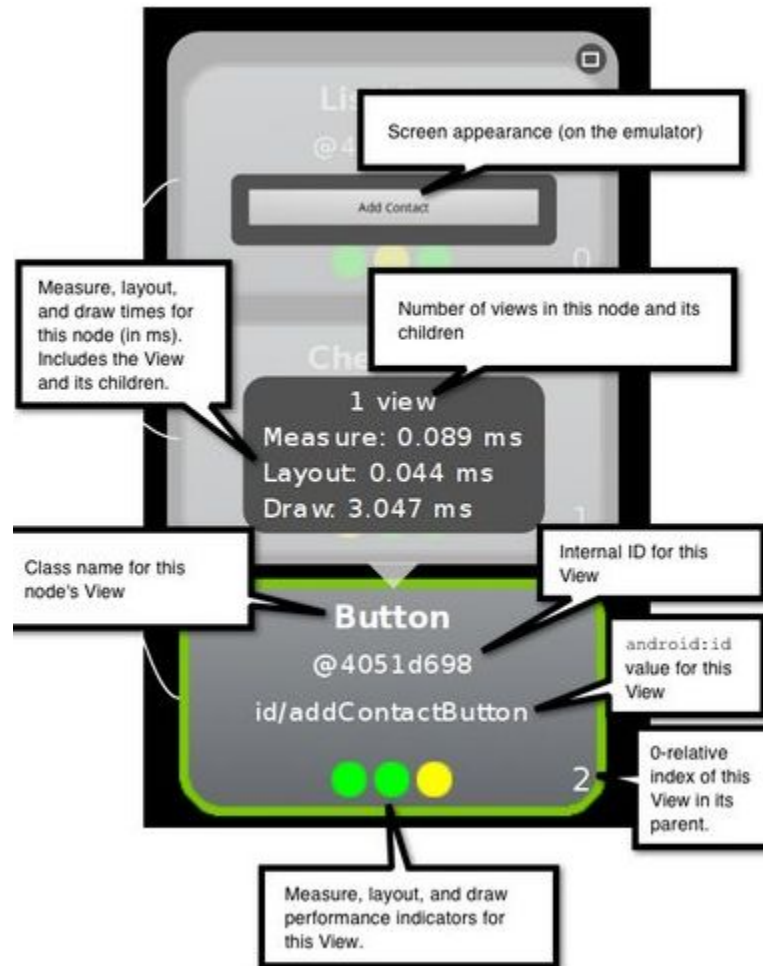
Property	Value

Show Extras [Load All Views](#)



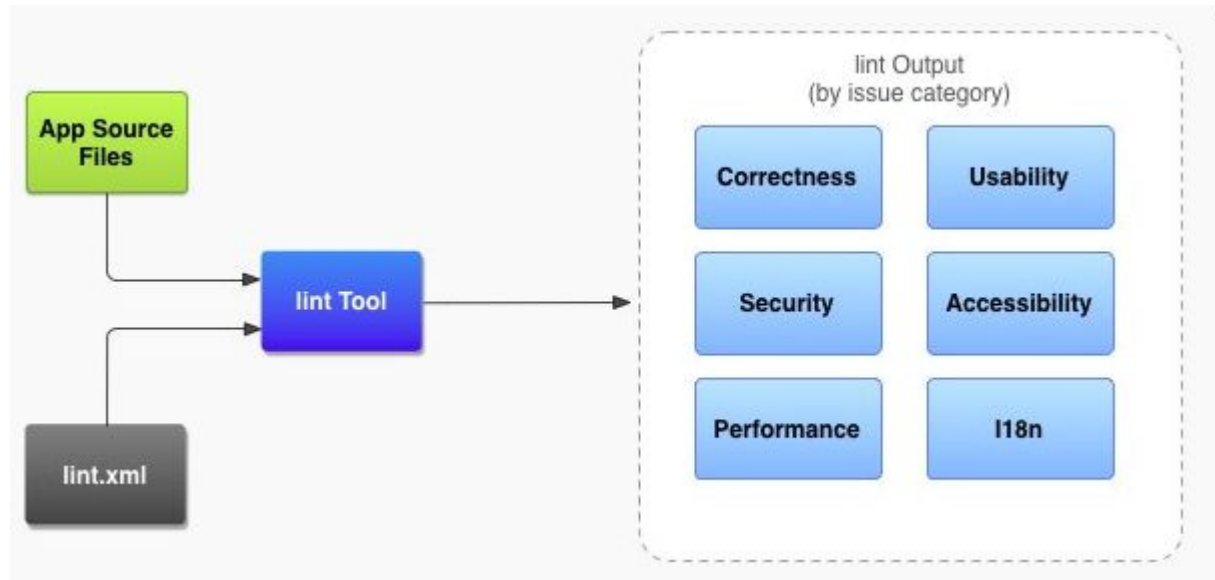
Hierarchy viewer

- <http://developer.android.com/tools/debugging/debugging-ui.html>



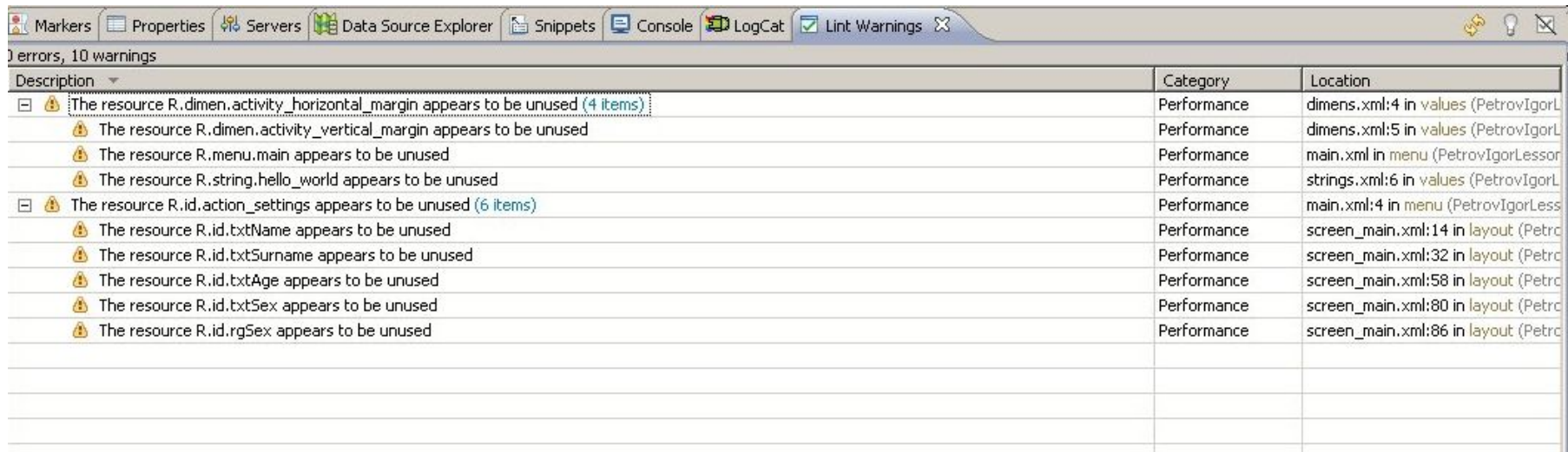
Optimize your UI – layoutopt and lint

- <http://developer.android.com/tools/debugging/debugging-ui.html>
- <http://developer.android.com/tools/debugging/improving-w-lint.html>













Optimize your UI – layoutopt and lint

- <http://developer.android.com/tools/debugging/debugging-ui.html>
- <http://developer.android.com/tools/debugging/improving-w-lint.html>



The screenshot shows the 'Lint Warnings' window in an IDE. The window title is '0 errors, 10 warnings'. The table below lists the warnings, their categories, and their locations in the code.

Description	Category	Location
 The resource R.dimen.activity_horizontal_margin appears to be unused (4 items)	Performance	dimens.xml:4 in values (PetrovIgorL
 The resource R.dimen.activity_vertical_margin appears to be unused	Performance	dimens.xml:5 in values (PetrovIgorL
 The resource R.menu.main appears to be unused	Performance	main.xml in menu (PetrovIgorLessor
 The resource R.string.hello_world appears to be unused	Performance	strings.xml:6 in values (PetrovIgorL
 The resource R.id.action_settings appears to be unused (6 items)	Performance	main.xml:4 in menu (PetrovIgorLess
 The resource R.id.txtName appears to be unused	Performance	screen_main.xml:14 in layout (Petro
 The resource R.id.txtSurname appears to be unused	Performance	screen_main.xml:32 in layout (Petro
 The resource R.id.txtAge appears to be unused	Performance	screen_main.xml:58 in layout (Petro
 The resource R.id.txtSex appears to be unused	Performance	screen_main.xml:80 in layout (Petro
 The resource R.id.rgSex appears to be unused	Performance	screen_main.xml:86 in layout (Petro

Code **Analyze** Refactor Build Run Tools VCS Window

Inspect Code...

Run Inspection by Name... ⌘⇧⌘I

Configure Current File Analysis... ⌘⇧⌘H

View Offline Inspection Results...

Infer Nullity...

Analyze Dependencies...

Analyze Backward Dependencies...

Analyze Module Dependencies...

Analyze Cyclic Dependencies...

Analyze Data Flow to Here

Analyze Data Flow from Here

Analyze Stacktrace...

d > c

eaProje

enActivity.java

package ru.myd

ort ...

public class F

private fi

private Bu

private Bu

private Bu

private Bu

private Bu



Inspection Results for Inspection Profile 'Project Default'



▼ IgorPetrovFifteenPuzzle (63 items)

- ▼ **Android Lint** (3 items)
 - ▶ 🧑 Image defined in density-independent drawable folder (2 items)
 - ▶ 🧑 Target SDK attribute is not targeting latest version (1 item)
- ▶ **Class structure** (16 items)
- ▶ **Control flow issues** (1 item)
- ▶ **Declaration redundancy** (8 items)
- ▶ **Error handling** (2 items)
- ▶ **Performance issues** (2 items)
- ▶ **Probable bugs** (10 items)
- ▶ **Properties Files** (1 item)
- ▶ **Spelling** (15 items)
- ▶ **XML** (5 items)

Layouts duplication – include and merge.

- Принцип DRY – Don't Repeate Yourself
- Если вы хотите избавиться от дублирующего кода в layouts используйте include
- Include – позволяет выделить часто используемый кусок layout в отдельный layout файл.
- `<include layout="@layout/button_bar" />`

Layouts duplication – include and merge.

- Merge используется в тех случаях когда вам нужно включить ваш повторяющийся код без использования родителя.



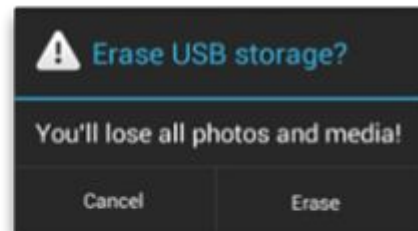
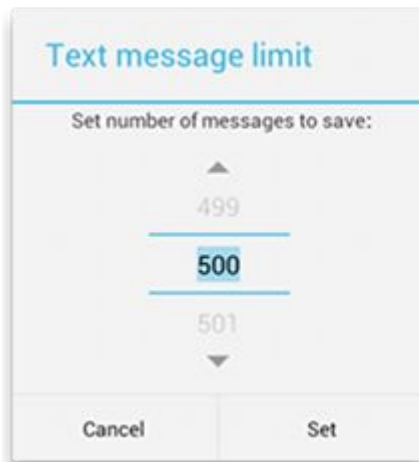
```
screen_main.xml  *button_bar.xml X
<?xml version="1.0" encoding="utf-8"?>

<merge xmlns:android="http://schemas.android.com/apk/res/android" >
  <Button
    android:id="@+id/btnSave"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:textSize="@dimen/font_size"
    android:textStyle="bold"
    android:text="@string/lblSave" />

  <Button
    android:id="@+id/btnLoad"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:textSize="@dimen/font_size"
    android:textStyle="bold"
    android:text="@string/lblLoad" />
</merge>
```


Создание диалогов. Создание меню.

- В этом уроке вы научитесь:
- Создавать различные виды стандартных диалогов
- Создавать диалоги с произвольным содержимым
- Получать данные из диалогов, введенные пользователем
- Выводить диалоги двумя способами
- Создавать главное меню окна и контекстного меню



Диалоги в Android.

Диалог – это небольшое окно, всплывающее поверх остальных окон приложения.

Диалог используется обычно для следующих целей:

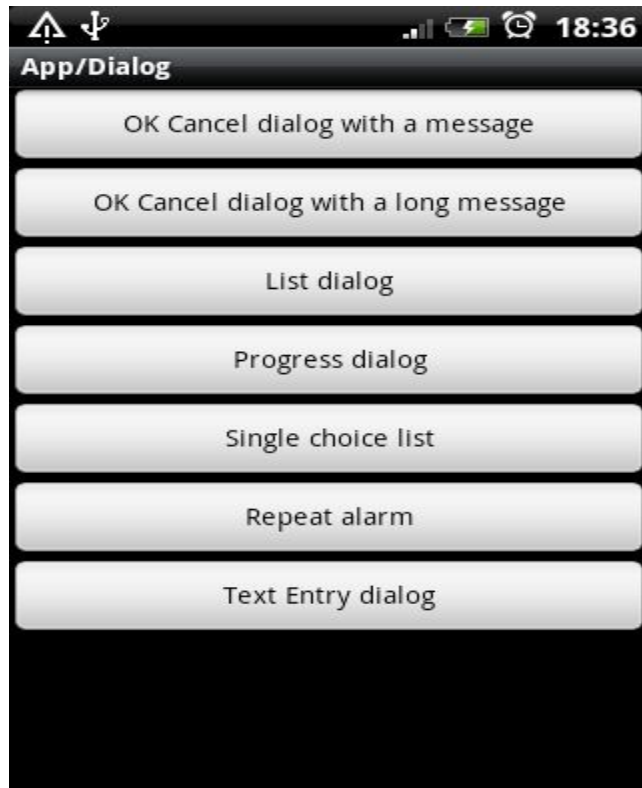
- Сообщить пользователю какую-либо информацию, например, о недоступности сети, внутренней ошибке. Обычно, в этом случае на диалоге появляется единственная кнопка «ОК», закрывающая диалог.
- Запросить у пользователя подтверждение какого-либо действия, в этом случае на диалоге имеется 2-3 кнопки: «ОК», «Отмена», иногда «Справка».
- Запросить у пользователя какую-либо информацию, например, ввод логина и пароля.

Вывод диалога на экран – действие, прерывающее жизненный цикл окна приложения, поверх которого он выводится – вызывается метод `onPause()` класса окна.

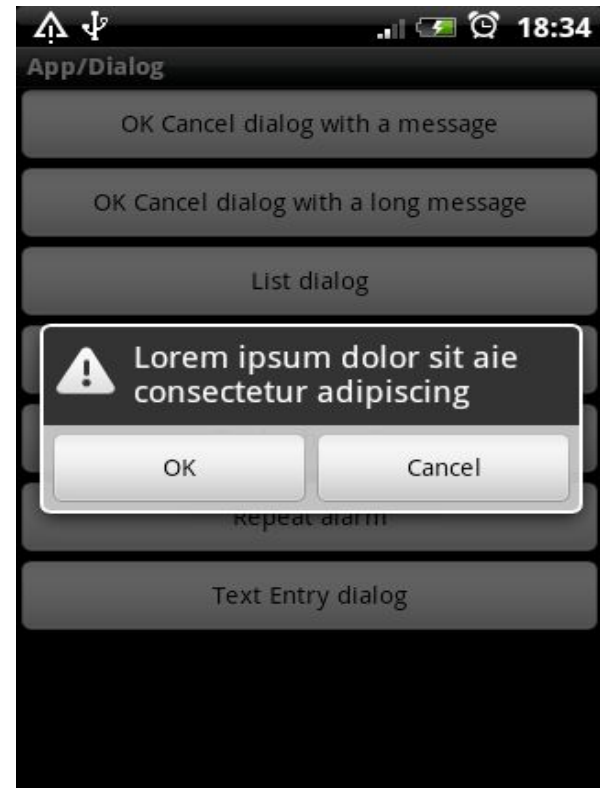
При этом это окно недоступно для пользователя, пока диалог не будет закрыт и фокус не будет возвращен вызывающему окну.

Диалоги в Android.

Внешний вид диалога в Android может быть абсолютно любым – не существует никакого минимального или максимального набора составляющих элементов окна диалога. В общем случае, можно в качестве содержимого диалога передать произвольный layout.



Активное окно



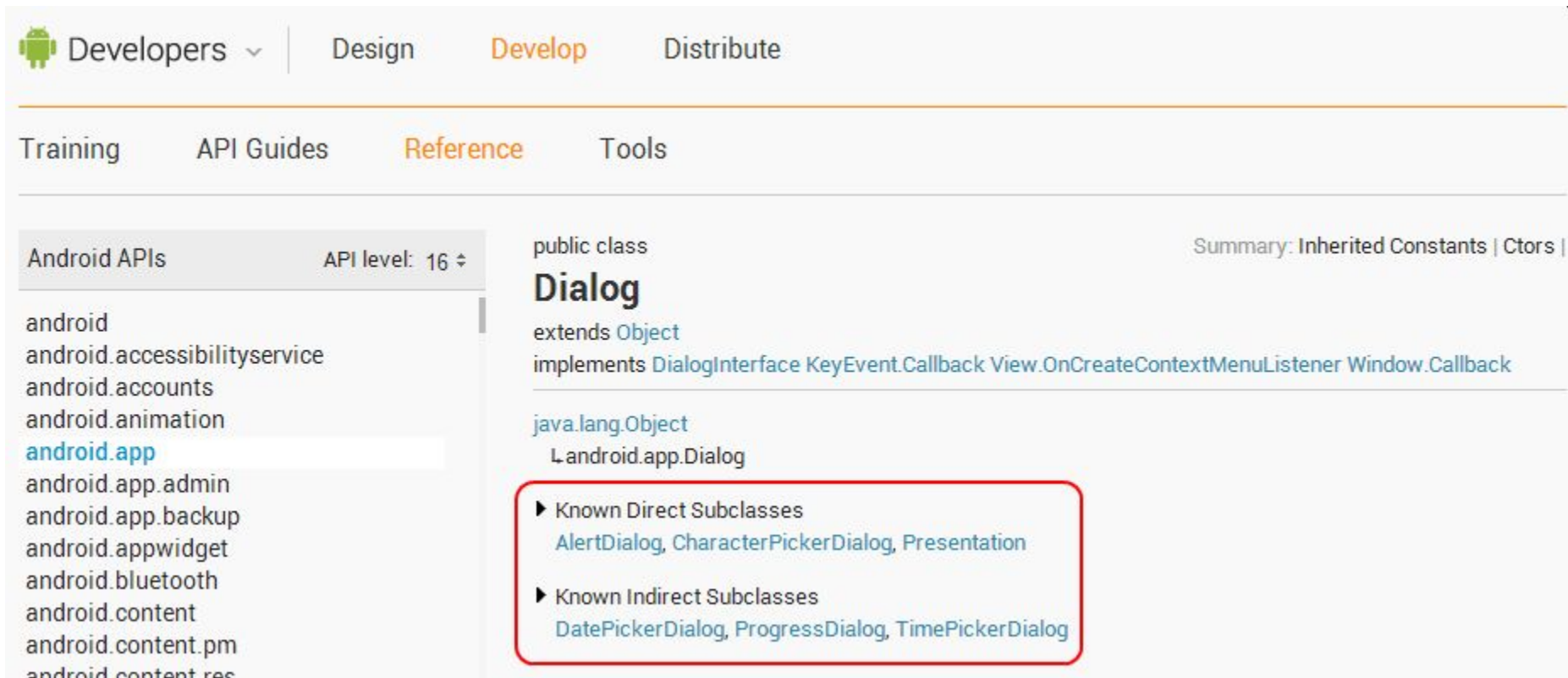
фокус
ввода передан диалогу

Диалоги в Android.

Все диалоги в Android являются наследниками класса Dialog.

Непосредственно создавать объект класса Dialog нам не придется, мы всегда будем пользоваться его известными наследниками

<http://developer.android.com/reference/android/app/Dialog.html>

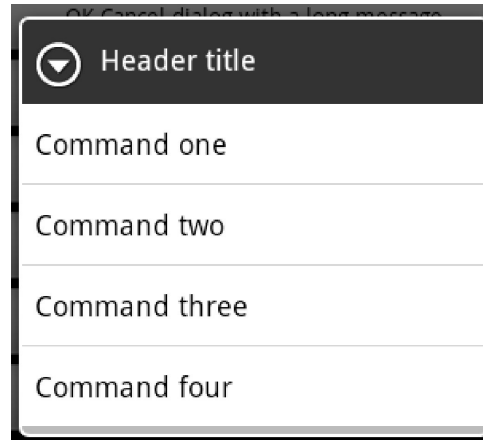


The screenshot shows the Android Developer website interface. At the top, there are navigation tabs: "Design", "Develop" (highlighted in orange), and "Distribute". Below that, there are sub-tabs: "Training", "API Guides", "Reference" (highlighted in orange), and "Tools". On the left side, there is a sidebar with "Android APIs" and "API level: 16". A list of packages is shown, with "android.app" highlighted. The main content area displays the "Dialog" class page. It shows "public class Dialog" which "extends Object" and "implements DialogInterface, KeyEvent.Callback, View.OnCreateContextMenuListener, Window.Callback". Below this, it indicates "java.lang.Object" and "Landroid.app.Dialog". A red box highlights two sections: "Known Direct Subclasses" (AlertDialog, CharacterPickerDialog, Presentation) and "Known Indirect Subclasses" (DatePickerDialog, ProgressDialog, TimePickerDialog). In the top right corner, there is a link for "Summary: Inherited Constants | Ctors |".

Alert Dialog.

Назван диалогом «предупреждения», хотя это необязательно так. Это может быть любой диалог, содержащий любую информацию, какие-то кнопки. Возможно, AlertDialog предложит пользователю выбрать какой-то вариант из предложенных, например, день недели, на который установить событие.

<http://developer.android.com/reference/android/app/AlertDialog.html>



Date Picker Dialog.

Стандартный диалог выбора даты, встроенный в ОС Android. Обычно, содержит барабан и сформированную дату.

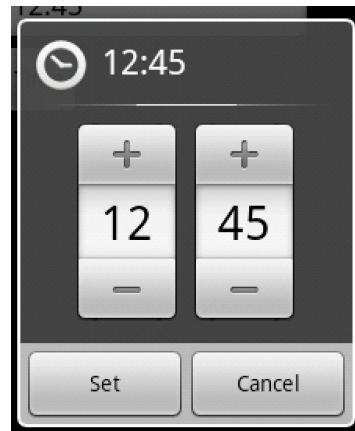
<http://developer.android.com/reference/android/app/DatePickerDialog.html>



TimePickerDialog.

Стандартный диалог выбора времени, встроенный в ОС Android.
Обычно, содержит барабан и сформированное время.

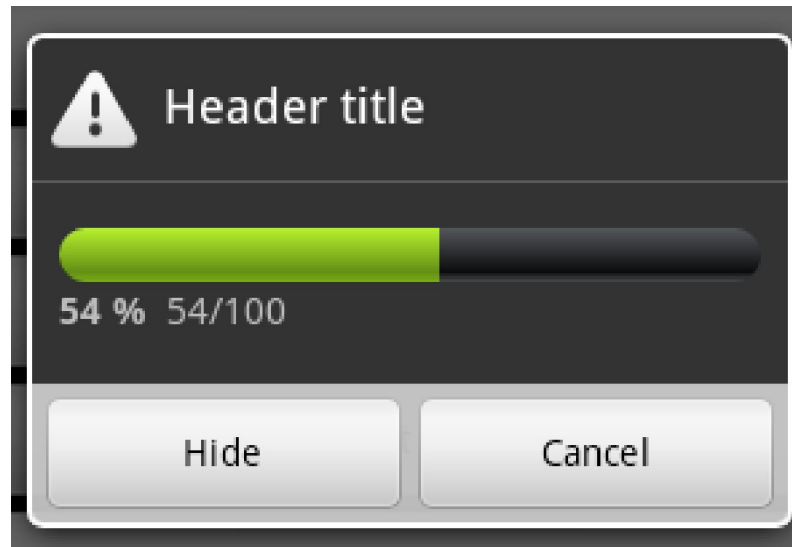
<http://developer.android.com/reference/android/app/TimePickerDialog.html>



ProgressDialog.

Диалог, содержащий полосу прогресса и некоторую информацию о текущем прогрессе выполнения какой-либо асинхронной длительной операции (скачивание файлов, вычисления, обращение к базе данных, т.д.)

<http://developer.android.com/reference/android/app/ProgressDialog.html>



Alert Dialog.

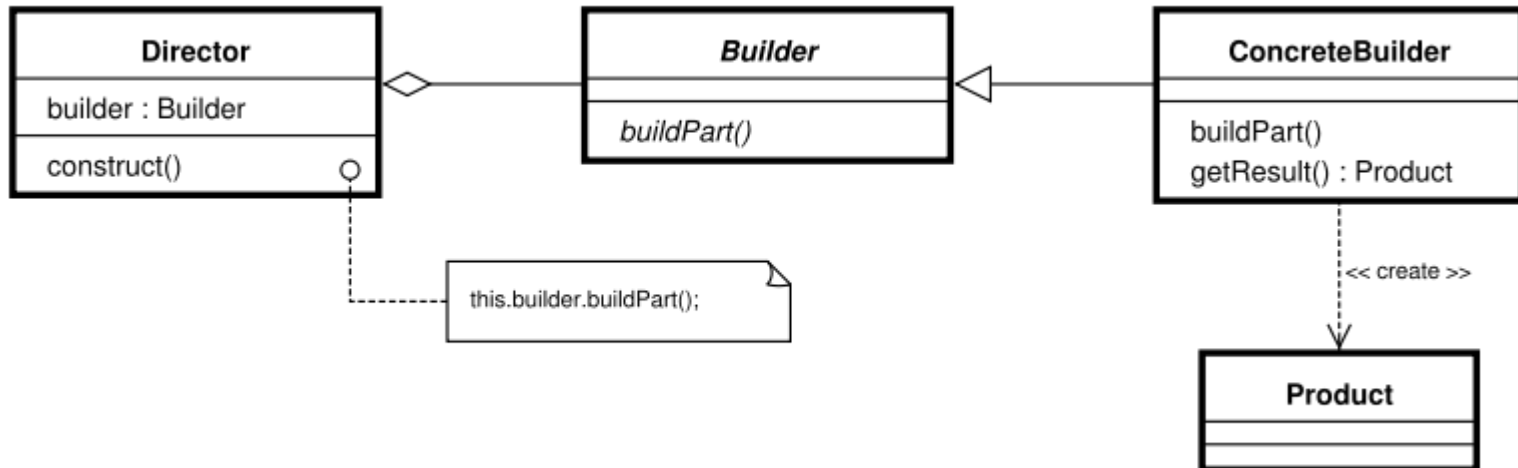
Как уже говорилось, класс `AlertDialog` используется для вывода любых диалогов. Если диалог, который мы хотим вывести, явно не подходит ни к одному из описанных выше типов (`ProgressDialog`, `DatePickerDialog`, т.д.), мы принимаем решение пользоваться классом `AlertDialog`.

Для конструирования диалогов типа `AlertDialog` используется специальный класс `AlertDialog.Builder`. Методами этого класса мы создаем внешний вид диалога, задаем заголовок, подготавливаем внешний вид компонентов, т.д. Последующий вызов метода `create()` этого класса возвращает нам объект класса `Dialog`, готовый к выводу.



Шаблон проектирования Builder

- http://en.wikipedia.org/wiki/Builder_pattern
- Порождающий шаблон проектирования Строитель служит для создания сложных объектов шаг за шагом.



Alert Dialog.

<http://developer.android.com/reference/android/app/AlertDialog.html>

<http://developer.android.com/guide/topics/ui/dialogs.html>

```
// 1. Instantiate an AlertDialog.Builder with its constructor
AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());

// 2. Chain together various setter methods to set the dialog characteristics
builder.setMessage(R.string.dialog_message)
    .setTitle(R.string.dialog_title);

// 3. Get the AlertDialog from create()
AlertDialog dialog = builder.create();
```



```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is present
    getMenuInflater().inflate(R.menu.activity_main, menu);
    return true;
}

public void showAlertDialog(View view) {
    AlertDialog.Builder builder = new AlertDialog.Builder(this);

    builder.setMessage("this is our alert dialog");
    builder.setCancelable(false);

    Dialog dialog = builder.create();
    dialog.show();
}
```

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is present
    getMenuInflater().inflate(R.menu.activity_main, menu);
    return true;
}

public void showAlertDialog(View view) {
    AlertDialog.Builder builder = new AlertDialog.Builder(this);

    builder.setTitle("Alert");
    builder.setIcon(R.drawable.ic_launcher);
    builder.setMessage("this is our alert dialog");
    // builder.setCancelable(false);

    Dialog dialog = builder.create();
    dialog.show();
}
```

```
public void showAlertDialog(View view) {
    AlertDialog.Builder builder = new AlertDialog.Builder(this);

    builder.setTitle("Alert");
    builder.setIcon(R.drawable.ic_launcher);
    builder.setMessage("this is our alert dialog");
    // builder.setCancelable(false);

    builder.setPositiveButton("Ok", new DialogInterface.OnClickListener()

        @Override
        public void onClick(DialogInterface dialog, int which) {
            // TODO Auto-generated method stub

        }
    });

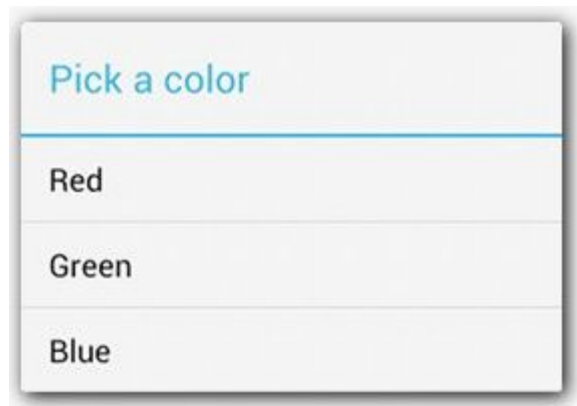
    Dialog dialog = builder.create();
    dialog.show();
}
```

Alert Dialog with single choice

<http://developer.android.com/reference/android/app/AlertDialog.html>

<http://developer.android.com/guide/topics/ui/dialogs.html>

```
public void openSingleChoiceDialog(View v) {  
    final CharSequence[] items = {"Red", "Green", "Blue"};  
  
    AlertDialog.Builder builder = new AlertDialog.Builder(this);  
    builder.setTitle("Pick color");  
    builder.setSingleChoiceItems(items, -1, new DialogInterface.OnClickListener() {  
  
        @Override  
        public void onClick(DialogInterface dialog, int which) {  
            Toast.makeText(Lesson4_dialogsActivity.this, items[which], Toast.LENGTH_LONG).show();  
        }  
    });  
  
    AlertDialog dialog = builder.create();  
    dialog.show();  
}
```



Dialog с произвольным содержимым

<http://developer.android.com/reference/android/app/AlertDialog.html>

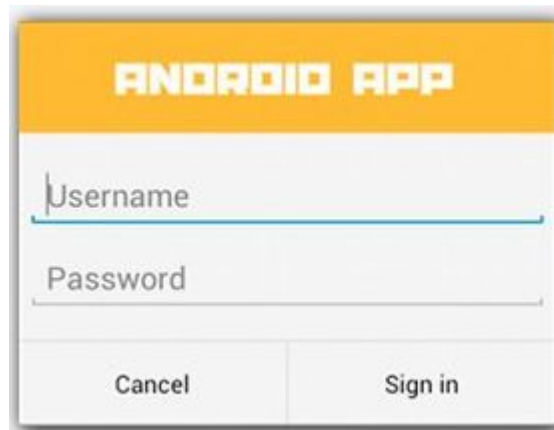
<http://developer.android.com/guide/topics/ui/dialogs.html>

```
public void openCustomView(View v) {
    AlertDialog.Builder builder = new AlertDialog.Builder(this);

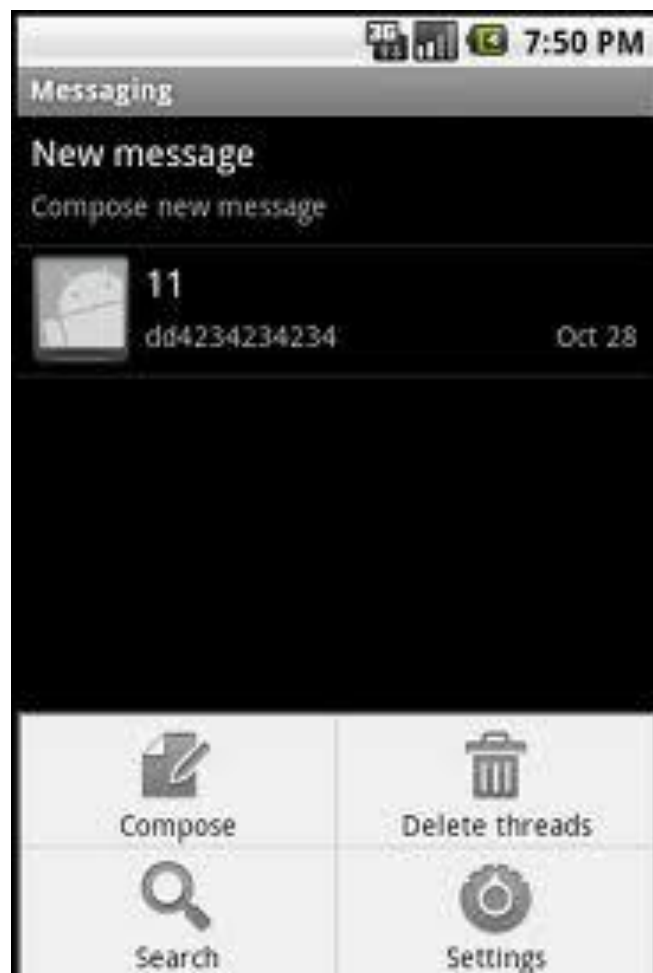
    // create view
    final View view = getLayoutInflater().inflate(R.layout.dialogs_login, null);
    builder.setView(view);

    builder.setPositiveButton("Ok", new DialogInterface.OnClickListener() {

        @Override
        public void onClick(DialogInterface dialog, int which) {
            Toast.makeText(Lesson4_dialogsActivity.this, "Ok button clicked", Toast.LENGTH_LONG).show();
        }
    });
}
```



Меню главное.



Меню контекстное.

