Assets (Активы)

* [Чтение файлов](http://developer.alexanderklimov.ru/android/theory/assets.php#readfile)
* [Используем собственные шрифты](http://developer.alexanderklimov.ru/android/theory/assets.php#customfont)
* [Загрузка локальных файлов из активов в WebView](http://developer.alexanderklimov.ru/android/theory/assets.php#webview)
* [Получаем список файлов в папке assets](http://developer.alexanderklimov.ru/android/theory/assets.php#listfiles)

В Android имеется еще один каталог, в котором могут храниться файлы, предназначенные для включения в пакет - **assets**. Этот каталог находится на том же уровне, что и **res**. Для файлов, располагающихся в **assets**, в **R.java** не генерируются идентификаторы ресурсов. Для их считывания необходимо указать путь к файлу. Путь к файлу является относительным и начинается с **/assets**. Этот каталог, в отличие от подкаталога **res**, позволяет задавать произвольную глубину подкаталогов и произвольные имена файлов и подкаталогов.

По умолчанию проект в студии не содержит данную папку. Чтобы её создать, выберите меню **File | New | Folder | Assets Folder**.

Чтение файлов

Для доступа к файлам Assets используется класс **AssetManager**. Пример для чтения текстового файла:

String getStringFromAssetFile(Activity activity)

{

AssetManager am = activity.getAssets();

InputStream is = am.open("test.txt");

String s = convertStreamToString(is);

is.close();

return s;

}

Другой способ:

String text = filename;

byte[] buffer = null;

InputStream is;

try {

is = getAssets().open(text);

int size = is.available();

buffer = new byte[size];

is.read(buffer);

is.close();

} catch (IOException e) {

e.printStackTrace();

}

String str\_data = new String(buffer);

return str\_data;

Для доступа к графическому файлу из актива можно использовать следующий код:

public void loadImageFromAsset() {

try {

// получаем входной поток

InputStream ims = getAssets().open("cat.jpg");

// загружаем как Drawable

Drawable d = Drawable.createFromStream(ims, null);

// выводим картинку в ImageView

mImage.setImageDrawable(d);

}

catch(IOException ex) {

return;

}

}

Вы также можете загрузить изображение в **Bitmap**, используя **BitmapFactory.decodeStream()**, вместо **Drawable**.

Используем собственные шрифты

Напишем практический пример создания приложения, в котором будут использоваться собственные шрифты, не входящие в стандартную библиотеку шрифтов Android. Для этого мы упакуем нужные шрифты вместе с приложением. Поместим в каталог **assets/fonts** файлы шрифтов (можно скачать бесплатные шрифты с сайтов [1001 Free Fonts](http://1001freefonts.com/) или [Urban Fonts](http://www.urbanfonts.com/free-fonts.htm)).

В файл разметки добавим пару текстовых полей с заготовленным текстом для вывода этого текста с нашим шрифтом.

В классе активности загрузим объект **EditText** из ресурсов, а затем создадим объект **Typeface**, используя вызов статического метода **Typeface.createFromAsset()**. Метод **createFromAsset()** принимает два параметра:

* объект **AssetManager**, который можно получить вызовом метода **getAssets()**
* путь к файлу актива.

Например, загрузить шрифт для текстового поля **EditText** можно следующим способом:

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

final EditText firstEditText = (EditText)findViewById(R.id.editText1);

firstEditText.setTypeface(Typeface.createFromAsset(

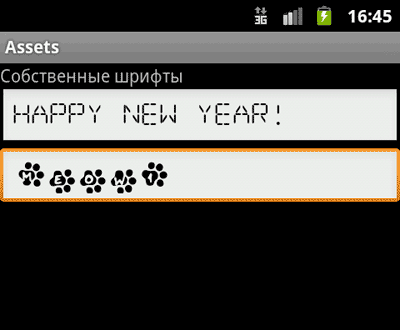
getAssets(), "fonts/DigitalDream.ttf"));

final EditText secondEditText = (TextView)findViewById(R.id.editText2);

secondEditText.setTypeface(Typeface.createFromAsset(getAssets(), "fonts/Catwalk.ttf"));

}

Запустив проект, мы увидим в текстовых полях надписи **Happy New Year!** и **Meow!**, выводимые нашими собственными шрифтами.



Пример для фрагмента.

@Override

public void onAttach(Activity activity) {

super.onAttach(activity);

// Какой-то шрифт

courgette = Typeface.createFromAsset(getActivity().getAssets(), "Courgette-Regular.ttf");

}

Загрузка локальных файлов из активов в WebView

Если нужно загрузить локальные страницы и изображения из активов в WebView, то можно использовать префикс **file://android\_asset**. Подробнее смотрите в статье про WebView.

myBrowser = (WebView)findViewById(R.id.mybrowser);

myBrowser.loadUrl("file:///android\_asset/mypage.html");

Получаем список файлов в папке assets

Можно получить список файлов, которые находятся в папке assets. Для быстрой проверки кода я вручную скопировал в папку два файла:

AssetManager myAssetManager = getApplicationContext().getAssets();

try {

String[] Files = myAssetManager.list(""); // массив имен файлов

Toast.makeText(getApplicationContext(), Files[0] + ", " + Files[1],

Toast.LENGTH\_LONG).show();

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

Кроме ваших файлов, также возвращаются странные папки **/images**, **/sounds**, **/webkit**. Учитывайте это в своих проектах. Так как в папке можно создавать собственные подпапки, то можно воспользоваться вспомогательным методом:

void displayFiles(AssetManager mgr, String path) {

try {

String list[] = mgr.list(path);

if (list != null)

for (int i = 0; i < list.length; ++i) {

Log.v("Assets:", path + "/" + list[i]);

displayFiles(mgr, path + "/" + list[i]);

}

} catch (IOException e) {

Log.v("List error:", "can't list" + path);

}

}

// вызываем метод

public void onClick(View view) {

final AssetManager mgr = getAssets();

displayFiles(mgr, ""); // содержимое папки /assets, включая странные папки

displayFiles(mgr, "cats"); // содержимое подпапки /assets/cats

}

Ограничение на размер файлов

По сети гуляет информация, что существует ограничение в 1 Мб на размер файлов в папке **assets**. При превышении размера у вас может появиться ошибка:

Data exceeds UNCOMPRESS\_DATA\_MAX

Я не сталкивался, поэтому рецепт решения проблемы не предлагаю.