Пишем функцию сохранения картинок на SD-карту

В процессе написания приложения для андроида у меня возникла задача сохранять произвольное изображение в файл на флешке. В этой статье я опишу, как я решил эту проблему, какие трудности встретились мне в процессе, и как они были решены. Хотелось бы отдельно отметить, что я .NET программист, судьба занесла меня в Java-мир только из-за необходимости создания небольших по размеру (поэтому monodroid сразу нет) и довольно простых с точки зрения интерфейса андроид-приложений. Это означает, что я тоже только учусь, а значит буду рад любым советом и замечаниям профессионалов.   
  
  
Итак, предположим, что у нас есть ImageView, в котором содержится картинка, необходимая нам в виде файла. Первый же вопрос — куда сохранять эту картинку?  
  
Самый правильный и гуглом одобренный подход будет сохранять картинку в папку кэша специально выделенную для вашей аппликации.  
Получить абсолютный путь к ней можно вызовом функции:   
  
context.getCacheDir();

Данная папка очищается при деинсталляции приложения, кроме того ее размер отслеживается системой, и при нехватке места внутренней памяти, файлы оттуда будут автоматически удалены андроидом.  
  
Есть у данного решения и минусы. Самый главный — папка кэша приложения содержится во внутренней памяти устройства, которой вечно не хватает. Рекомендуется сохранять там лишь временные и не очень большие файлы, сохранение большого количества тяжелых картинок здесь нежелательно.  
  
Лучше воспользоваться папкой кэша приложения на SD-карте, путь к которой можно получить функцией:   
  
context.getExternalCacheDir();

Данная папка тоже будет очищена при деинсталляции приложения, но ее размер не отслеживается системой, поэтому перед сохранением туда файлов, ее состояние желательно проверять командой:

Environment.getExternalStorageState();

В моем случае по совокупности разных факторов оба стандартных решения показались мне неоптимальными, поэтому я просто сохраняю файлы в корне SD-карты, путь туда можно получить функцией:

Environment.getExternalStorageDirectory();

Теперь о самой процедуре сохранения, ее код приведен ниже.

ImageView iv; // ImageView, содержащий изображение, которое нужно сохранить

String folderToSave = Environment.getExternalStorageDirectory().toString(); // папка куда сохранять, в данном случае - корень SD-карты

private String SavePicture(ImageView iv, String folderToSave)

{

OutputStream fOut = null;

Time time = new Time();

time.setToNow();

try {

File file = new File(folderToSave, Integer.toString(time.year) + Integer.toString(time.month) + Integer.toString(time.monthDay) + Integer.toString(time.hour) + Integer.toString(time.minute) + Integer.toString(time.second) +".jpg"); // создать уникальное имя для файла основываясь на дате сохранения

fOut = new FileOutputStream(file);

Bitmap bitmap = (BitmapDrawable) iv.getDrawable().getBitmap();

bitmap.compress(Bitmap.CompressFormat.JPEG, 85, fOut); // сохранять картинку в jpeg-формате с 85% сжатия.

fOut.flush();

fOut.close();

MediaStore.Images.Media.insertImage(getContentResolver(),

file.getAbsolutePath(), file.getName(), file.getName()); // регистрация в фотоальбоме

}

catch (Exception e) // здесь необходим блок отслеживания реальных ошибок и исключений, общий Exception приведен в качестве примера

{

return e.getMessage();

}

return "";

}

Несколько пояснений по коду.  
  
File file = new File(folderToSave, Integer.toString(time.year) + Integer.toString(time.month+1) + Integer.toString(time.monthDay) + Integer.toString(time.hour) + Integer.toString(time.minute) + Integer.toString(time.second) +".jpg");

Не очень изящное, но простое как карандаш и реально работающее решение получить уникальное имя для файла в формате удобном для сортировки — «2011 11 17 20 31 49.jpg»  
Здесь всего одна маленькая хитрость. Обратили внимание на *time.month+1*?  
Дело в том, что Java считает месяцы с нуля, и ноябрь получается 10м, а не 11м, как все привыкли, месяцем. Непорядок.

bitmap.compress(Bitmap.CompressFormat.JPEG, 85, fOut);

Тут тоже все самоочевидно, споры могут возникнуть лишь о процентах сжатия. Мне кажется, что 85% оптимальный вариант, но у каждого может быть свое мнение и свои условия задачи.

MediaStore.Images.Media.insertImage(getContentResolver(),   
 file.getAbsolutePath(), file.getName(), file.getName());

Интересный финт, о котором многие забывают. Не все андроид-пользователи продвинуты настолько, чтобы отыскать сохраненные вами файлы, даже если вы им показали путь файловой системы. Но вот про приложение «Фотоальбом» знают точно все (это там, где только что снятые фоточки можно посмотреть). Добавьте вышеупомянутую строчку в ваш код, и картинка не только сохранится в нужную директорию, но еще и зарегистрируется в фотоальбоме, и пользователь всегда ее отыщет ее без всяких проблем.  
  
Кстати, приведенная функция взята из реального работающего приложения Random Pictures, которое показывает случайные картинки из Интернета, а в платной версии программы — имеет возможность сохранения любой картинки к себе на флешку.  
  
Бесплатная версия Random Picture Free на андроид маркете https://habrastorage.org/getpro/habr/post_images/a6e/c50/7ff/a6ec507ff0733641ca440d8f3d942f39.png [Random Pictures Free](http://market.android.com/details?id=com.sly2soft.RandomPicturesFree)  
  
  
Описанная функция сохранения файла работает только в платной версии. Ее цена — 45 рублей, но для хабражителей — конечно же выкладывается бесплатно. (Хорошая традиция, надо сказать!) Ссылка от автора в комментариях.