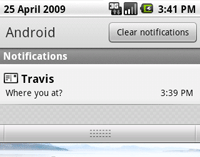
Работа со строкой состояния в Android



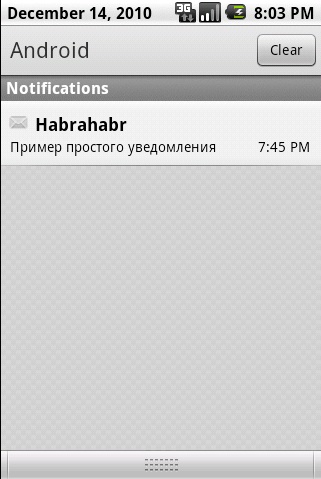
Кратко о строке уведомления

На первом скриншоте представлена раскрытая строка состояния в стандартной оболочке Android (некоторые производители мобильных устройств в своих оболочках могут изменить её внешний вид, хотя суть остаётся той же). Строка состояния в Android по большей части используется для уведомления пользователя о каких либо событиях произошедших в системе, а также о результатах работы каких либо приложений. Существенным отличием от всплывающих уведомлений Toast является то что уведомления в строке состояния не пропадают спустя время, и «висят» там до тех пор пока пользователь как-то на них отреагирует. Строку состояния удобно использовать для получения уведомлений от приложений запущенных в фоновом режиме, а также с появлением уведомления можно проиграть какой либо звук, вибрацию, или же воспользоваться мигающими индикаторами на устройстве (если они имеются). Уведомление представленное на скриншоте — именно то чего сегодня мы и будем добиваться.

Создание простого уведомления в строке состояния

Для начала попробуем создать стандартное уведомление для командной строки так как это рекомендуют в Google. Разметку интерфейса приложения оставим без изменений (уведомление будет появляться сразу после его запуска). И так, пример кода (с комментарием того что возможно может быть не ясным):

public class NotificationBar extends Activity {  
  /\*\* Called when the activity is first created. \*/  
     
  private static final int NOTIFY\_ID = 1; // Уникальный индификатор вашего уведомления в пределах класса  
  @Override  
    
    public void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.main);  
    NotificationManager mNotificationManager = (NotificationManager) getSystemService(Context.NOTIFICATION\_SERVICE); // Создаем экземпляр менеджера уведомлений  
    int icon = android.R.drawable.sym\_action\_email; // Иконка для уведомления, я решил воспользоваться стандартной иконкой для Email  
    CharSequence tickerText = "Hello Habrahabr"; // Подробнее под кодом  
    long when = System.currentTimeMillis(); // Выясним системное время  
    Notification notification = new Notification(icon, tickerText, when); // Создаем экземпляр уведомления, и передаем ему наши параметры  
    Context context = getApplicationContext();   
    CharSequence contentTitle = "Habrahabr"; // Текст заголовка уведомления при развернутой строке статуса  
    CharSequence contentText = "Пример простого уведомления"; //Текст под заголовком уведомления при развернутой строке статуса  
    Intent notificationIntent = new Intent(this, NotificationBar.class); // Создаем экземпляр Intent  
    PendingIntent contentIntent = PendingIntent.getActivity(this, 0, notificationIntent, 0);class); // Подробное описание в UPD к статье  
    notification.setLatestEventInfo(context, contentTitle, contentText, contentIntent); // Передаем в наше уведомление параметры вида при развернутой строке состояния  
    mNotificationManager.notify(NOTIFY\_ID, notification); // И наконец показываем наше уведомление через менеджер передав его ID   
  }  
}  
\* This source code was highlighted with [Source Code Highlighter](http://virtser.net/blog/post/source-code-highlighter.aspx).

**CharSequence tickerText = «Hello Habrahabr»;**— в этой строке мы указываем текст который будет показан в свёрнутой строке состояния на несколько секунд при появлении уведомления. Спустя несколько секунд он исчезнет, а в строке останется лишь иконка.   
  
  
  
Вот пожалуй и всё. Можно открывать шампанское, и наблюдать то что у нас получилось.  
  


Создание расширенного уведомления в строке состояния

Теперь несколько усложним задачу — мы будем создавать уведомление не по шаблону что предлагает Google, а по собственной разметке (благо такая возможность имеется). И так создадим новый файл разметки в папке layout, у вас он должен получиться таким:

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
       android:orientation="horizontal"  
       android:layout\_width="fill\_parent"  
       android:layout\_height="fill\_parent"  
       android:padding="3dp"  
       >  
  <ImageView android:id="@+id/image"  
       android:layout\_width="wrap\_content"  
       android:layout\_height="fill\_parent"  
       android:layout\_marginRight="10dp"  
       />  
  <TextView android:id="@+id/text"  
       android:layout\_width="wrap\_content"  
       android:layout\_height="fill\_parent"  
       android:textColor="#000"  
       />  
</LinearLayout>  
  
\* This source code was highlighted with Source Code Highlighter.

Также добавим какую либо картинку в папку drawable дабы потом установить её в ImageView.  
  
Теперь код. Код не сильно отличается от кода простого уведомления, но тем не менее требует комментария

public class NotificationBar extends Activity {  
  /\*\* Called when the activity is first created. \*/  
     
  private static final int NOTIFY\_ID = 1; // Уникальный индификатор вашего уведомления в пределах класса  
  @Override  
    
    public void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    NotificationManager mNotificationManager = (NotificationManager) getSystemService(Context.NOTIFICATION\_SERVICE); // Создаем экземпляр менеджера уведомлений  
    int icon = android.R.drawable.sym\_action\_email; // Иконка для уведомления, я решил воспользоваться стандартной иконкой для Email  
    CharSequence tickerText = "Hello Habrahabr"; // Подробнее под кодом  
    long when = System.currentTimeMillis(); // Выясним системное время  
    Intent notificationIntent = new Intent(this, NotificationBar.class); // Создаем экземпляр Intent  
    Notification notification = new Notification(icon, tickerText, when); // Создаем экземпляр уведомления, и передаем ему наши параметры  
    PendingIntent contentIntent = PendingIntent.getActivity(this, 0, notificationIntent, 0); // Подробное описание смотреть в UPD к статье  
    RemoteViews contentView = new RemoteViews(getPackageName(), R.layout.notlayout); // Создаем экземпляр RemoteViews указывая использовать разметку нашего уведомления  
    contentView.setImageViewResource(R.id.image, R.drawable.habr); // Привязываем нашу картинку к ImageView в разметке уведомления  
    contentView.setTextViewText(R.id.text,"Привет Habrahabr! А мы тут, плюшками балуемся..."); // Привязываем текст к TextView в нашей разметке  
    notification.contentIntent = contentIntent; // Присваиваем contentIntent нашему уведомлению  
    notification.contentView = contentView; // Присваиваем contentView нашему уведомлению  
    mNotificationManager.notify(NOTIFY\_ID, notification); // Выводим уведомление в строку  
    }  
}  
  
\* This source code was highlighted with Source Code Highlighter.

В итоге можем наблюдать примерно такую картину:  
  


Добавляем звук и вибрацию, мигаем индикаторами.

Для пущей важности добавим звук при выводе уведомления и вибрацию. Сделать это совсем не сложно.  
  
**notification.defaults |= Notification.DEFAULT\_SOUND;** — данная строка присваивает уведомлению звук что используется в системе по умолчанию.  
**notification.sound = Uri.parse(«file:///sdcard/notification/ringer.mp3»);** — таким способом можно установить звук из файла на SD карте.  
  
**notification.defaults |= Notification.DEFAULT\_VIBRATE;** — данная строка добавляет вибрацию в колличестве времени по умолчанию.  
  
Кроме того имеется возможность задать время вибрации самостоятельно. Делается это двумя строчками  
**long[] vibrate = {0,100,200,300}**; — создаем массив, в котором 1-ое число — время которое следует подождать до того как запустить вибрацию. Второе значение — время первой вибрации в миллисекундах (аналогично и 3, и 4 значение). Количество вибраций может быть бесконечно большим по усмотрению программиста.  
**notification.vibrate = vibrate;** — присваиваем массив нашему уведомлению.  
  
**notification.defaults |= Notification.DEFAULT\_LIGHTS;** — данной строкой мы можем по мигать индикаторами с параметрами по умолчанию.  
  
Конечно можно настроить параметры и в ручную.Для этого нам потребуется 4 строки:  
**notification.ledARGB = 0xff00ff00;** — задаем цвет для RGB индикатора.  
**notification.ledOnMS = 300;** — задаем время между миганиями  
**notification.ledOffMS = 1000;** — задаем время спустя которое горящий индикатор потухнет  
**notification.flags |= Notification.FLAG\_SHOW\_LIGHTS;** — разрешаем мигать  
Тут следует заметить что далеко не на всех девайсах вообще имеются какие либо индикаторы.  
  
Естественно все эти установки нужно производить до строчки вызова уведомления.

Заключение

Вот пожалуй и всё. Пост никак не претендует на полноту, и всё же данной информации должно быть вполне достаточно новичкам. В качестве источника, а также более полного описания работы с NotificationBar могу привести [оригинальную](http://developer.android.com/guide/topics/ui/notifiers/notifications.html) статью на [developer.android.com](http://developer.android.com/). Надеюсь статья поможет кому либо в изучении возможностей платформы. Спасибо за внимание.  
  
UPD: при внимательном рассмотрении выяснилось что комментарий к одной из строчек кода отсутствует (обещается объяснить под кодом, но самого объяснения нет). Досадную ошибку исправляет, а также ряд ф-и строки состояния о которых я сообщить забыл восполняет пользователь [djvu](https://habrahabr.ru/users/djvu/), взглянуть можно в комментариях: [1](http://habrahabr.ru/blogs/android/111238/#comment_3546742), [2](http://habrahabr.ru/blogs/android/111238/#comment_3546811). О существовании серьезных подводных камней сообщает [sdmitry](https://habrahabr.ru/users/sdmitry/) вот [тут](http://habrahabr.ru/blogs/android/111238/#comment_3546874).