Пишем своё первое приложение на Android[из песочницы](https://habrahabr.ru/sandbox/)

[Разработка под Android](https://habrahabr.ru/hub/android_dev/)\*, [Eclipse](https://habrahabr.ru/hub/eclipse/" \o "Вы не подписаны на этот хаб)\*



Предисловие

Цель данного поста — с одной стороны поделиться своим успешным опытом старта разработки приложений на платформе Android и с другой стороны поспособствовать развитию рынка софта для этой замечательной и бурно растущей платформы за счёт (без ложной скромности скажу) возможно Вас, прочитавших данный пост. В сети, конечно, можно найти материалы на тему разработки приложения «чуть сложнее, чем helloworld», но как правило они разрозненные и в них не описываются различные мелкие подводные камешки. В данном посте мы рассмотрим полный цикл разработки приложения, начиная с чистого компьютера до готового apk-файла. Под катом скрины. 

Подготовка к разработке

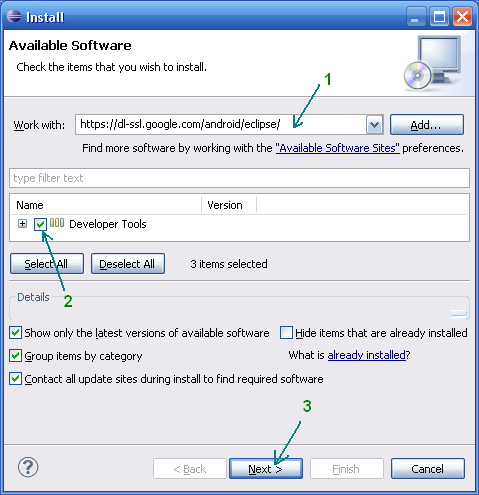
Для того, чтобы подготовиться к разработке не требуется каких-то особых действий и манипуляций. Буду описывать пошагово, что нужно сделать. У меня стоит Windows XP, поэтому и всё нижесказанное следует относить именно к данной ОС, хотя действия без каких-то сильных изменений можно применить и к другим ОС семейства Windows и даже Linux.

* Установка среды разработки Eclipse
* Установка ADT плагина к Eclipse
* Установка JDK и JRE
* Установка Android SDK

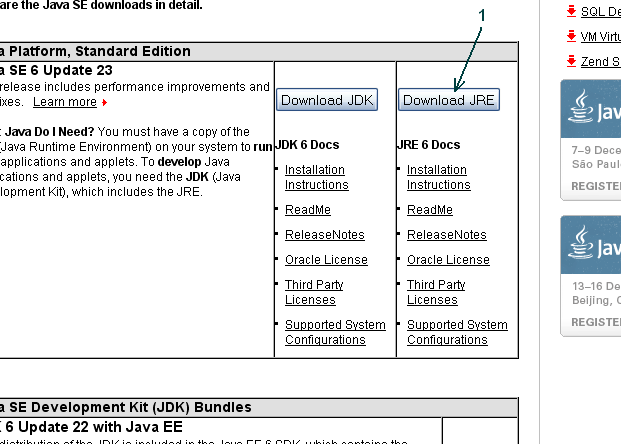
Установка среды разработки Eclipse

Тут всё просто. Идём в раздел Downloads на [официальном сайте](http://www.eclipse.org/downloads/) и качаем версию Classic. Далее просто распакуйте архив, куда вам хочется, я же решил это сделать в почетную директорию *C:\Eclipse*

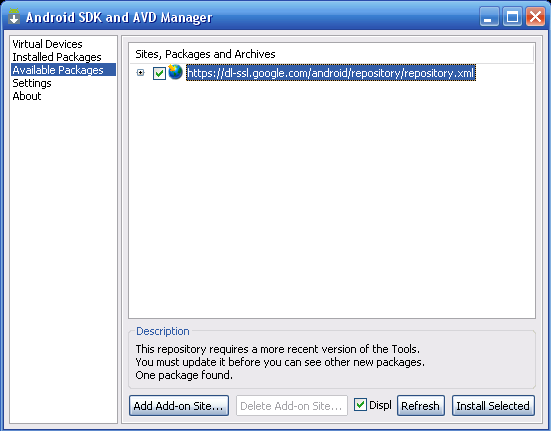
Установка ADT плагина к Eclipse

Запускаем Eclipse, и открываем диалог установки плагина через меню **Help → Install new software**. В поле Work with вписываем адрес загрузки плагина [*dl-ssl.google.com/android/eclipse*](https://dl-ssl.google.com/android/eclipse/) (если не получилось, то плагин можно скачать и установить вручную по ссылке <http://developer.android.com/sdk/eclipse-adt.html>), далее в таблице ниже появится Developer Tools, отмечаем его и идем далее.  
  
  
После успешной установки, можно перезапустить среду Eclipse.

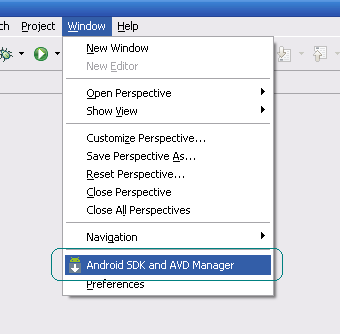
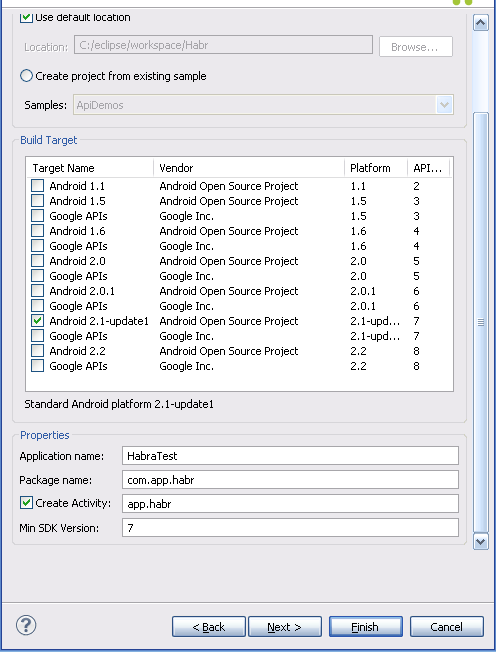
Установка среды для Java: JDK и JRE

Если у вас еще не установлены Java Development Kit (JDK) и Java Runtime Environment (JRE), то их необходимо установить. Сделать это можно на [сайте Oracle](http://www.oracle.com/technetwork/java/javase/downloads/index.html). Скачайте и установите JDK и JRE.  
  


Установка Android SDK

Дело осталось за малым — это скачать и установить свежий Android SDK. Это делается на [сайте для Android разработчиков](http://developer.android.com/sdk/index.html). Я лично установил SDK опять-таки в почётную директорию *C:\Android*. После этого необходимо добавить платформы и прочие дополнительные элементы SDK. Я добавил все доступные версии, так как планирую делать приложения и для ранних платформ, а также USB-драйвер и примеры приложений.  
  
  
  
Подготовка к разработке завершена. Теперь следующий этап — создание приложения.

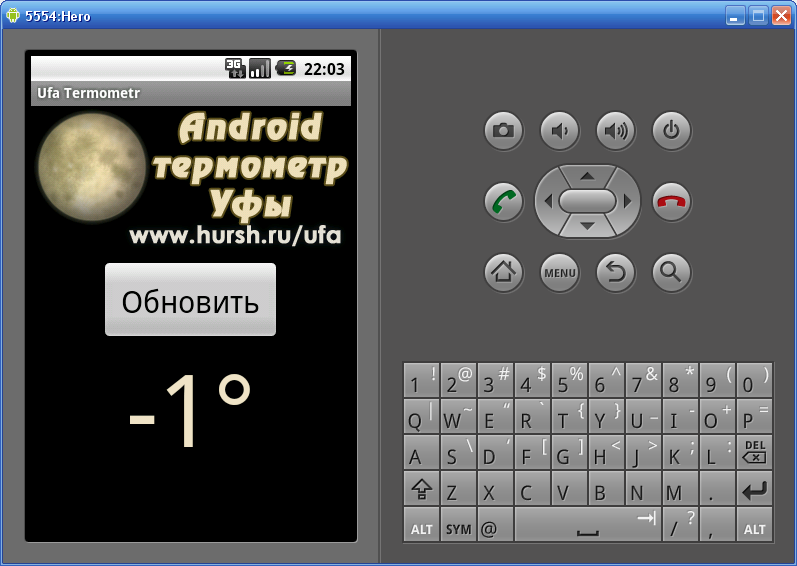
Создание Android-приложения

Перед тем как создать своё первое приложение, можно создать виртуальное Android-устройство, чтобы быстро тестировать на нём свой свеженаписанный софт. Вначале я хочу сказать вам несколько слов по поводу Android Virtual Device (AVD). Это виртуальный смартфон с ОС Android, на котором вы с легкостью сможете запускать созданную вами программу. Как можно заметить, преимущество Android Virtual Device состоит в том, что наглядно можно увидеть как будет работать Ваша программа на разных смартфонах с Android, а не покупать весь модельный ряд и тестировать приложение на каждом из них.  
  
Приступим к созданию виртуального устройства Android. Следуем в меню по пути **Window → Android SDK and AVD Manager**.  
  
  
  
Выбираем **New** в правой части окна, в появившемся окне вписываем имя виртуального устройства, платформу (допустим Android 2.1), размер карты памяти (допустим 512 Mb), тип дисплея (допустим HVGA). Затем жмем внизу на кнопку Create AVD.  
  
Теперь создадим проект. Для этого идем по пути в меню **File → New → Other**, в появившемся списке**Android → Android Project**.  
  
  
  
В качестве примера расскажу о разработке одной моей простейшей программы UfaTermometr, показывающей текущую температуру с датчиков, расположенных на одном из объектов местной энерго-компании.  
  
После создания проекта, слева вы увидите дерево директорий. Перво-наперво загрузим свою иконку приложения, а точнее 3 иконки под разные варианты. В папку drawable-hdpi грузим png-картинку с прозрачным фоном размером в 72x72, в drawable-mdpi соответственно 48x48 и в drawable-ldpi самую маленькую размером 36x36. Делать это можно простым перетаскиванием файлов прямо на дерево. Следующим шагом будут элементы управления. В моей программе было необходимо всего три элемента интерфейса: ImageView (просто картинка логотип приложения), Button (обновить значение температуры) и TextView (вывод значения температуры). Все эти элементы управления нужно описать в специальном xml-файле. В дереве он расположен по адресу **res → layout → main.xml**. Во-многом разметка элементов управления сходна с вёрсткой веб-страниц, тут есть и padding, и margin, и аналог align. Код main.xml моего приложения:  
<?xml version=«1.0» encoding=«utf-8»?>  
<LinearLayout xmlns:android=«[schemas.android.com/apk/res/android](http://schemas.android.com/apk/res/android)»  
    android:orientation=«vertical»  
    android:layout\_width=«fill\_parent»  
    android:layout\_height=«fill\_parent»  
    >  
<ImageView android:layout\_width=«fill\_parent»  
    android:layout\_height=«wrap\_content»  
    android:src="@drawable/droid"  
    android:layout\_gravity=«center\_horizontal|center»  
    android:layout\_margin=«0px»  
    />  
<Button  
    android:id="@+id/refrbutton"   
    android:layout\_width=«wrap\_content»   
    android:layout\_height=«wrap\_content»  
    android:gravity=«center\_horizontal»  
    android:layout\_gravity=«center\_horizontal|center»  
    android:textSize=«30px»  
    android:padding=«20px»  
    android:layout\_marginTop=«10dip»  
    android:text=«Обновить»  
    />  
<TextView   
    android:layout\_width=«fill\_parent»   
    android:layout\_height=«wrap\_content»   
    android:text="..."  
    android:textSize=«100dip»  
    android:gravity=«center\_horizontal»  
    android:id="@+id/temper"  
    android:textColor="#EEE3C4"  
    />  
</LinearLayout>  
  
Подробно останавливаться на разметке элементов управления не буду, так как это всё достаточно подробно описано в мануале и вообще интуитивно понятно, особенно тому, кто имел дело с html/css-вёрсткой и/или разработкой в какой-либо визуальной среде (например, Delphi). Единственное, обратите внимание как я загрузил произвольное изображение в ImageView. В поле src указывается путь, начинающийся с символа @ и далее указываете адрес по дереву. В папке res я заблаговременно создал подпапку drawable и «закинул» туда логотип, а затем просто указал ImageView путь. Легко? По мне так очень.  
  
Непосредственно код приложения располагается по пути **src → «ваше название package» → «название приложения».java**.  
По-умолчанию создается «болванка» главного класса и от неё уже дальше можно плясать. Не буду останавливаться на тонкостях такого простого кода как у меня в примере, но поясню, что логика работы объясняется тремя простыми шагами:

* Загрузить HTML-страницу со значениями температуры
* С помощью регулярок «вытащить» температуру
* Показать в TextView значение температуры

Заметьте, что когда приложению требуются какие-то системные разрешения, нужно указывать их в манифесте **AndroidManifest.xml**. В данном конкретном случае мы используем подключение к Интернет, соответственно добавляем строчку <uses-permission android:name=«android.permission.INTERNET»/> в указанный выше xml-файл.  
Собственно сам код, реализующий указанную выше логику работы:  
package app.test.ufatermometr;  
  
import android.app.Activity;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.TextView;  
import android.widget.Button;  
import java.io.InputStreamReader;  
import java.net.URL;  
import java.net.URLConnection;  
import java.util.regex.\*;  
  
public class UfaTermometr extends Activity  
{  
  
    @Override  
    public void onCreate(Bundle savedInstanceState)  
    {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.main);  
        final Button button = (Button) findViewById(R.id.refrbutton);  
        button.setOnClickListener(new Button.OnClickListener() {  
            public void onClick(View v) // клик на кнопку  
            {  
                RefreshTemper();  
            }  
        });  
  
        RefreshTemper(); // при запуске грузим температуру сразу  
    };  
    //----------------------------------------------------------------  
    public String GetTemper(String urlsite) // фукция загрузки температуры  
    {  
        String matchtemper = "";  
        try  
        {  
                // загрузка страницы  
            URL url = new URL(urlsite);  
            URLConnection conn = url.openConnection();  
            InputStreamReader rd = new InputStreamReader(conn.getInputStream());  
            StringBuilder allpage = new StringBuilder();  
            int n = 0;  
            char[] buffer = new char[40000];  
            while (n >= 0)  
            {  
                n = rd.read(buffer, 0, buffer.length);  
                if (n > 0)  
                {  
                    allpage.append(buffer, 0, n);                      
                }  
            }  
            // работаем с регулярками  
            final Pattern pattern = Pattern.compile  
            ("<span style=\«color:#[a-zA-Z0-9]+\»>[^-+0]+([-+0-9]+)[^<]+</span>[^(а-яА-ЯёЁa-zA-Z0-9)]+([а-яА-ЯёЁa-zA-Z ]+)");  
            Matcher matcher = pattern.matcher(allpage.toString());  
            if (matcher.find())  
            {      
                matchtemper = matcher.group(1);              
            }          
            return matchtemper;  
        }  
        catch (Exception e)  
        {  
              
        }  
        return matchtemper;    
    };  
    //----------------------------------------------------------------  
    public void RefreshTemper()  
    {   
        final TextView tTemper = (TextView) findViewById(R.id.temper);  
         String bashtemp = "";  
          bashtemp = GetTemper(«[be.bashkirenergo.ru/weather/ufa](http://be.bashkirenergo.ru/weather/ufa/)»);  
        tTemper.setText(bashtemp.concat("°")); // отображение температуры  
    };  
}  
После того как приложение написано, можно уже его отлаживать и тестировать.

Запуск и тестирование приложения

Вспоминаем про нашу виртуальную машину и запускаем привычной кнопкой или менюшкой **Run → Run**  
  
Вот такую картину мы можем затем созерцать:  
  
  
  
Теперь, если вы хотите поделиться приложением, то нужно собрать apk-файл. Для этого воспользуемся меню **File → Export** и в списке **Android → Export Android application**. Далее выберите проект, затем создайте хранилище ключей keystore и ключ key, для этого нужно будет заполнить несколько полей с всякого рода справочной информацией. Получившийся apk-файл можно распространять и даже выложить в Android Market, но для этого придется зарегистрироваться и уплатить $25, что в общем-то немного, особенно для стоящего проекта. Но регистрация в Маркете, пожалуй, тема отдельной статьи.

Заключение

В заключении хотелось бы сказать, что безусловно разработка приложений для Android приятна, достаточно проста и интересна. Конечно, мы рассмотрели лишь верхушку айсберга, но я надеюсь, что в умах тех из вас, кто раньше не пробовал делать ничего подобного «загорелась лампочка» и вполне возможно, что когда-то вашим приложением будут пользоваться миллионы.

Источники, ссылки

В посте использованы материалы:  
[www.ibm.com](http://ibm.com/)  
[www.itblog.name](http://itblog.name/)  
[Википедия](http://ru.wikipedia.org/)  
  
[APK-файл приложения](http://rghost.ru/3561566)

UPD: Устранение возможных ошибок

[Defite](http://defite.habrahabr.ru/):  
1. Ошибка

ERROR: Unable to open class file C:\workspace\Test\gen\com\example\test\R.java: No such file or directory

решается очисткой проекта через меню **Project → Clean** или перезапуском Eclipse.  
2. Когда возникает ошибка

emulator: ERROR: no search paths found in this AVD's configuration. Weird, the AVD's config.ini file is malformed. Try re-creating it

— это следствие того, что у Вас кириллица в имени пользователя. Решается: идем в **Компьютер → Свойства системы → Дополнительные параметры системы → Переменные среды**. Создаем новую переменную с именем ANDROID\_SDK\_HOME и значением – путем, где находится папка AVD (например, C:\Android\). Создаем, далее ищем там же переменную Path, открываем и в поле значений через точку с запятой добавляем путь до папки tools Android SDK (например, C:\Android\tools). Сохраняем. Запускаем Eclipse, запускаем программу.  
  
UPD 05.10.2011, во что превратилось данное приложение можно лицезреть на следующих скринах:  
  
  
Это результат развития той самой болванки, которая выше дана в качестве примера.  
[Разработка и создание Android-приложений](http://orpis.ru/)  
[Создание сайтов](http://loftylab.ru/)

* [android](https://habrahabr.ru/search/?q=%5Bandroid%5D&target_type=posts),
* [sdk](https://habrahabr.ru/search/?q=%5Bsdk%5D&target_type=posts),
* [программ](https://habrahabr.ru/search/?q=%5B%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%5D&target_type=posts)