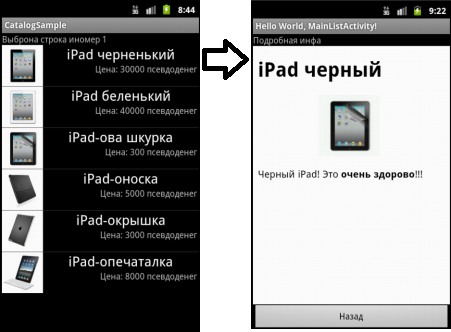
**Программирование Android справочника на основе ListView и WebView.**

Не корысти для, а стёба ради напишем приложение, которое будет представлять собой справочник-обертку для интернет магазина: этакий[каталог планшетов Apple iPad 2](http://www.hit-tovar.ru/apple/apple-ipad-2.html). При запуске нашего каталога будет отображаться список, каждая строка которого содержит картинку товара, его название и цену. При нажатии на строку в списке будет показана страница с подробным описанием товара. Для простоты, мы не будем рассматривать доставку данных из интернета (хотя при желании эта функциональность очень просто реализуется), а сосредоточимся на вопросе отображения уже полученной каким-то образом информации. Описание товара будем представлять в виде HTML кода, для отображения которого используем компонент WebView.

**Использование ListView в Android приложениях**

Создадим новый проект. Назовем его CatalogSample. В качестве целевой платформы выберем Android 2.3.3, Package Name: ru.mobilab.catalog, Create Activity: MainListActivity, MinimumSDK: 10.  
  
Откроем в дереве проекта файл res/layout/main.xml, который описывает шаблон нашей деятельности (экрана, формы) и добавим компонент ListView. Если Вы пользуетесь графическим редактором, этот компонент находится во вкладке Composite. Измените значение поля android:id на @android:id/list. Шаблон главной деятельности main.xml должен выглядеть следующим образом:

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent"  
 android:orientation="vertical">  
<TextView  
 android:id="@+id/titleMain"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="@string/hello">  
</TextView>  
<ListView  
 android:id="@android:id/list"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content">  
</ListView>  
</LinearLayout>

Перейдем к файлу MainListActivity.java в  папке src проекта. В настоящее время в качестве предка этого класса указан класс Activity. Заменим его на ListActivity, не забыв при этом подключить необходимую библиотеку android.app.ListActivity. Этот класс специально создан для деятельностей, в основе которых лежат списки. Он упрощает обработку событий. Нам ведь понадобится отслеживать события щелчка по строке списка. Чтобы показать, что у нас в приложении есть главный список, мы чуть ранее поменяли параметр android:id компонента ListView на "@android:id/list". Теперь внутри класса MainListActivity можно переопределить метод onListItemClick, который будет вызываться при касании пользователем строки списка. Пока для примера будем просто выводить в TextView сообщение "Выброна строка иномер N".

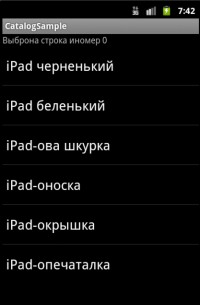
**public** void onListItemClick(ListView parent, View v, int position, long id){

myText.setText("Выброна строка иномер "+position);

}

Здесь myText - объект, связанный с TextView.  
  
Для отображения массивов данных в виджетах применяются, так называемые, адаптеры данных. Адаптер связывает данные с отображающим его виджетом. В простейшем случае, когда нам нужно отобразить массив строк в виде списка можно использовать ArrayAdapter<T>. В качестве параметров конструктору нужно передать:

1. экземпляр класса, реализующий деятельность;
2. идентификатор ресурса шаблона строки списка. Здесь можно указать самостоятельно созданный res/layout/\*.xml файл, либо воспользоваться одним из стандартных шаблонов, например android.R.layout.simple\_list\_item\_1 - просто отображает каждую строку, как TextView.
3. массив или список с данными.

Для начала мы реализуем простейший список, в котором содержатся только строки.

package ru.mobilab.catalog;

import ru.mobilab.catalog.R;

import android.app.ListActivity;

import android.os.Bundle;

import android.widget.ListView;

import android.widget.TextView;

import android.widget.ArrayAdapter;

import android.view.View;

**publicclass** MainListActivity**extends** ListActivity{

**private** TextView myText;

String[] names={"iPad черненький","iPad беленький",

"iPad-ова шкурка","iPad-оноска","iPad-окрышка",

"iPad-опечаталка"};

@Override

**public** void onCreate(Bundle savedInstanceState){

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

*//Создаем ArrayAdapter*

ArrayAdapter<String> myArrAd=**new** ArrayAdapter<String>

(this, android.R.layout.simple\_list\_item\_1,names);

setListAdapter(myArrAd);

myText=(TextView)findViewById(R.id.titleMain);

}

**public** void onListItemClick(ListView parent, View v, int position, long id){

myText.setText("Выброна строка иномер "+position);

}

}

**Добавляем собственный шаблон строки списка**

Щелкнем правой кнопкой мыши по res/layout и выберем New>Other>Android XML file. В появившемся диалоге выберем: Resource Type: Layout, File: my\_item, Root Element: Linear Layout и нажмем Finish. Приведем этот файл к виду



<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout\_width="match\_parent"  
    android:layout\_height="match\_parent"  
    android:orientation="vertical"  
    >  
    <LinearLayout  
        android:id="@+id/linearLayout1"  
        android:layout\_width="match\_parent"  
        android:layout\_height="wrap\_content" >  
        <ImageView  
            android:id="@+id/imageView1"  
            android:layout\_width="wrap\_content"  
            android:layout\_height="wrap\_content"  
            android:src="/@drawable/ic\_launcher" />  
        <LinearLayout  
            android:id="@+id/linearLayout2"  
            android:layout\_width="match\_parent"  
            android:layout\_height="match\_parent"  
            android:orientation="vertical" >  
            <TextView  
                android:id="@+id/textView1"  
                android:layout\_width="wrap\_content"  
                android:layout\_height="wrap\_content"  
                android:layout\_gravity="center"  
                android:text="Large Text"  
                android:textAppearance="?android:attr/textAppearanceLarge"  
                />  
            <TextView  
                android:id="@+id/textView2"  
                android:layout\_width="wrap\_content"  
                android:layout\_height="match\_parent"  
                android:layout\_gravity="right"  
                android:text="TextView"  
                 />  
        </LinearLayout>  
    </LinearLayout>  
</LinearLayout>

Если попробовать использовать этот шаблон в конструкторе ArrayAdapter, при запуске приложения будет возникать ошибка. Нам необходимо написать собственный**адаптер данных**, который сможет корректно взаимодействовать с новым шаблоном.  
  
Прежде чем приступать к его созданию,  напишем класс для хранения информации о товаре в нашем каталоге.  Щелкнем правой кнопкой по src/ru.mobilab.catalog и выберем New Java Class. В качестве Name укажем List Data и нажмем Finish.

package ru.mobilab.catalog;

**publicclass** ListData{

String title;*//Название товара*

int price;*//Цена товара*

int image;*//Ссылка на изображение*

String discribe;*// HTML описание товара*

ListData(String \_title, int \_price, int \_image, String \_discribe){

title= \_title;

price= \_price;

image= \_image;

discribe=\_discribe;

}

}

Перейдем к созданию адаптера данных. Создадим еще один Java Class с именем CatalogAdapter. В качестве класса предка укажем BaseAdapter - это базовый класс общей реализации адаптеров данных. Внутри этого класса нужно переопределить несколько методов:

* **int getCount()** - возвращает количество строк в полученном массиве данных.
* **Object getItem(int position)** - возвращает объект из массива данных, находящийся на позиции position.
* **long getItemId(int position)** - возвращает идентификатор элемента массива данных, находящегося на позиции position.
* **View getView(int position, View convertView, ViewGroup parent)** - формирует для позиции position объект View, отвечающий за отображение строки в списке. Именно в этом методе элементам шаблона присваиваются конкретные значения из массива данных.

package ru.mobilab.catalog;

import java.util.ArrayList;

import android.content.Context;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.BaseAdapter;

import android.widget.ImageView;

import android.widget.TextView;

**publicclass** CatalogAdapter**extends** BaseAdapter{

Context cont;

LayoutInflater lInflater;

ArrayList<ListData> objects;

CatalogAdapter(Context context, ArrayList<ListData> mylist){

cont= context;

objects= mylist;

lInflater=(LayoutInflater) cont.getSystemService(Context.LAYOUT\_INFLATER\_SERVICE);

}

**public** int getCount(){

return objects.size();

}

**public** Object getItem(int position){

return objects.get(position);

}

**public** long getItemId(int position){

return position;

}

**public** View getView(int position, View convertView, ViewGroup parent){

*// используем созданные, но не используемые view*

View view= convertView;

if(view==**null**){

*//получаем LayoutInflater для работы с layout-ресурсами*

view= lInflater.inflate(R.layout.my\_item, parent,**false**);

}

ListData p=((ListData) getItem(position));

*// заполняем View в пункте списка данными*

((TextView) view.findViewById(R.id.textView1)).setText(p.title);

((TextView) view.findViewById(R.id.textView2)).setText("Цена: "+p.price+" псевдоденег");

((ImageView) view.findViewById(R.id.imageView1)).setImageResource(p.image);

return view;

}

}

Скопируем картинки[pic01.png,...,pic09.png](http://www.mobilab.ru/files/drawable-listview.zip) в папки res/draweble-hdpi/ res/draweble-ldpi/ res/draweble-mdpi/. Внесем изменения в класс MainListActivity.

package ru.mobilab.catalog;

import java.util.ArrayList;

import ru.mobilab.catalog.R;

import ru.mobilab.catalog.CatalogAdapter;

import ru.mobilab.catalog.ListData;

import android.app.ListActivity;

import android.os.Bundle;

import android.widget.ListView;

import android.widget.TextView;

import android.view.View;

**publicclass** MainListActivity**extends** ListActivity{

*/\*\* Called when the activity is first created. \*/*

**private** TextView myText;

**private** ArrayList<ListData> catalog;

String[] names={"iPad черненький","iPad беленький","iPad-ова шкурка",

"iPad-оноска","iPad-окрышка","iPad-опечаталка"};

String[] desc={"<h1>iPad черный</h1> <center><img src=**\"**pic03.png**\"**>

</center><p>Черный iPad! Это <b>очень здорово</b>!!!</p>",

"<h1>iPad беленький</h1>","<h1>iPad-ова шкурка</h1>",

"<h1>iPad-оноска</h1>","<h1>iPad-окрышка</h1>",

"<h1>iPad-опечаталка</h1>"};

int[] cost={30000,40000,300,5000,3000,8000};

int[] img={R.drawable.pic01,R.drawable.pic02,R.drawable.pic03,

R.drawable.pic04,R.drawable.pic05,R.drawable.pic06};

@Override

**public** void onCreate(Bundle savedInstanceState){

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

myText=(TextView)findViewById(R.id.titleMain);

*//Создаем массив объектов ListData и заполняем их данными*

catalog=**new** ArrayList<ListData>();

for(int i=1; i<=6; i++){

catalog.add(**new** ListData(names[i-1], cost[i-1], img[i-1],desc[i-1]));

}

*//Создаем адаптер данных*

CatalogAdapter catAdapter;

catAdapter=**new** CatalogAdapter(this, catalog);

setListAdapter(catAdapter);

}

**public** void onListItemClick(ListView parent, View v, int position, long id){

myText.setText("Выброна строка иномер "+position);

}

}

Мы ввели еще три массива данных: desc, cost и img и использовали их в конструкторе onCreate при создании списка ListData. После того как список создан, мы создали на его основе CatalogAdapter и вызвали setListAdapter(catAdapter). Запустите проект.



**Вывод подробной информации через WebView**

Как отмечалось ранее, описание товара храниться в виде HTML текста. Для его отображения нам понадобится новая деятельность. Для начала создадим новый xml шаблон в res/layout/about.xml

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent"  
 android:orientation="vertical">  
<TextView  
 android:layout\_width="fill\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Подробная инфа">  
</TextView>  
<WebView  
    android:id="@+id/webView1"  
    android:layout\_width="match\_parent"  
    android:layout\_height="420dp" />  
<Button  
    android:id="@+id/button1"  
    android:layout\_width="match\_parent"  
    android:layout\_height="match\_parent"  
    android:text="Назад" />  
</LinearLayout>

Создадим новую деятельность, для этого добавим в проект Java class AboutActivity. Для регистрации новой деятельности в AndroidManifest.xml перед  </application> добавим новую строку:

<activity android:name="AboutActivity" android:label="@string/hello"></activity>

В AboutActivity.java вставьте текст

package ru.mobilab.catalog;

import android.app.Activity;

import android.os.Bundle;

import android.view.View;

import android.webkit.WebView;

import ru.mobilab.catalog.R;

import android.widget.Button;

import android.view.View.OnClickListener;

import android.content.Intent;

**publicclass** AboutActivity**extends** Activity implements OnClickListener{

*/\*\* Called when the activity is first created. \*/*

**public** static final String EXT\_TextToShow="text";

**private** WebView web;

*/\*\* Called when the activity is first created. \*/*

**public** void onCreate(Bundle savedInstanceState){

super.onCreate(savedInstanceState);

setContentView(R.layout.about);

*//Получаем данне от деятельности MainListActivity*

Intent intent= getIntent();

String cells= intent.getStringExtra(EXT\_TextToShow);

*//Устанавливаем обработчик нажатия кнопки*

Button btnAct=(Button) findViewById(R.id.button1);

btnAct.setOnClickListener(this);

*//Получаем доступ к WebView и загружаем туда HTML страницу*

web=(WebView) findViewById(R.id.webView1);

web.loadDataWithBaseURL("file:///android\_res/drawable/",

"<!Doctype html><html><head><meta charset=utf-8></head>

<body>"+cells+"</body></html>","text/html","utf-8","");

}

**public** void onClick(View v){

finish();

}

}

Для загрузки страницы в WebView мы использовали метод web.loadDataWithBaseURL. В качестве первого параметра указывается базовая папка, которая используется для вычисления относительных ссылок и путей. Например, если нам нужно показать картинку, мы используем тег <img src="/pic01.png> при этом если в качестве первого параметра стоит "file:///android\_res/drawable/", то картинка ищется в папках drawable. Доступ к папкам drawable из WebView появился в Android 2.2. Если Вы пишете приложение для более старой платформы, можете использовать для размещения картинок папку кэша assets.  
  
Сам текст для отображения передается через Intent. Подробно об Intent-ах и механизме перехода между деятельностями рассказано[**тут**](http://www.mobilab.ru/articles/105/).  
  
Вернемся к файлу MainListActivity.java и изменим метод onListItemClick

**public** void onListItemClick(ListView parent, View v, int position, long id){

Intent intent=**new** Intent(this, AboutActivity.**class**);

intent.putExtra(AboutActivity.EXT\_TextToShow, catalog.get(position).discribe);

startActivity(intent);

}

Запустите проект. Теперь при касании по строке должна открываться страница с подробным описанием товара.