因为Android软件开发分工目前还没有细化，程序员往往需要负责软件界面的开发，尽管软件的界面图片已经由美工设计好了，但如果使用layout技术把软件做成如图片所示的界面确实很困难，而且也比较耗时。Android通过WebView实现了JavaScript代码与Java代码互相通信的功能，使得Android软件的界面开发也可以采用HTML网页技术，这样，广大网页美工可以参与进Android软件的界面开发工作，从而把程序员从中解脱出来。目前在一些场合已经可以取代Layout。下面是一个使用Html作为UI实现的的电话拨号器的实例。

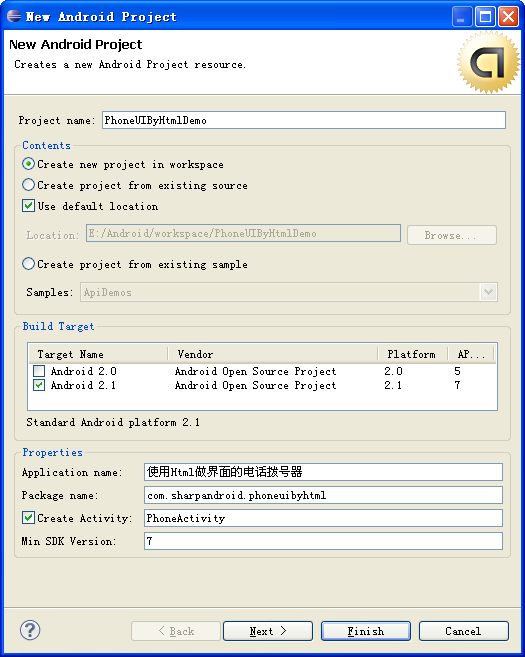
**界面设计**



其中数据可以来自于SQLite数据库，也可以来自于内容提供者，或者来自于网络，点击电话号码下的链接能够拨打电话，界面用Html实现。

访问数据库需要用Java语言，并且数据需要在Html界面上显示，但Html能运行JavaScript脚本语言，而不支持Java语言。值得大家高兴的是，Android实现了JavaScript与Java语言互访的功能，这样从Html🡪JavaScript—>Java，整个通道就被打通，Html可以使用SQLite数据库及内容提供者中的数据了。其中JavaScript和Java之间可以传递Json格式的数据，因此在传递数据时需要进行数据格式的转换。

**创建名为“PhoneUIByHtmlDemo”的工程**，图片如下



点击【Finish】。

**编写网页文件：**

在assets目录下新建一个Html文件phoneui.html。内容如下：

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>Insert title here</title>

<script type="text/javascript">

//该部分用到了JavaScript的一些知识。不熟练的读者可以查阅相关书籍。

//或者可由美工做出来之后，进行必要的修改！

function show(jsondata){

//将传递过来的json数据转换为对象

var jsonobjs = eval(jsondata);

//获取下面定义的的表格

var table = document.getElementById("personTable");

//遍历上面创建的json对象，将每个对象添加为表格中的一行，其每个属性为一列

for(var i=0; i<jsonobjs.length; i++){

//添加一行

var tr = table.insertRow(table.rows.length);

//添加三个单元格

var td1 = tr.insertCell(0);

var td2 = tr.insertCell(1);

td2.align = "center";

var td3 = tr.insertCell(2);

//设置么个单元格的内容和属性，

//其中innerHTML为设置或获取位于对象起始和结束标签内的 HTML

//jsonobjs[i]为对象数组中的第i个对象

td1.innerHTML = jsonobjs[i].id;

td2.innerHTML = jsonobjs[i].name

//为显示的内容添加超链接。

//超链接会调用Java代码中的call（）方法并且把内容作为参数传递过去

td3.innerHTML = "<a href='javascript:sharp.call(\""+ jsonobjs[i].phone +"\")'>"+jsonobjs[i].phone+ "</a>"; ;

}

}

</script>

</head>

<!--onload指定了当该页面被加载时调用的方法，本例调用了Java代码中的contactlist方法-->

<body bgcolor="#000000" text="#FFFFFF" style="margin:0000" onload="javascript:sharp.contactlist()">

<!-- 定义一个表格，显示数据-->

<table border="0" width="100%" id="personTable" cellspacing="0">

<tr>

<td width="15%">编号</td><td align="center">姓名</td><td width="15%">电话</td>

</tr>

</table>

</body>

</html>

说明：

* **function** show(jsondata)

**该方法将传入的json数据进行分割，每条数据作为一行显示在下面的表格当中。**

* td3.innerHTML = "<a href='javascript:sharp.call(\""+ jsonobjs[y].phone +"\")'>"+jsonobjs[y].phone+ "</a>";

为其文本加上超链接，点击该超链接会调用HtmlActivity的内部类SharpJavaScript类的实体对象的call方法来完成拨号，并将号码传入。

* onload=*"javascript:Sharp.contactlist()*

当页面加载时访问Sharp所代表的HtmlActivity的内部类SharpJavaScript类的实体对象的contactlist()方法，完成数据的获取，并将数据拆分成Json数据，之后该方法会调用JavaScript中的show方法来为表格添加数据。

**编写main.xml：**

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="vertical"

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"

>

<WebView

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"

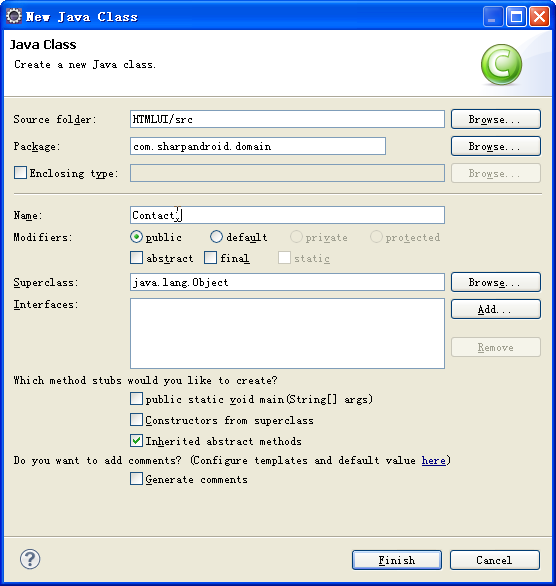
android:id="@+id/webView"

/>

</LinearLayout>

**编写Contact.java：**

创建一个实体类，封装联系人信息。如下图。



其代码如下：

**package** com.sharpandroid.domain;

**public** **class** Contact {

**private** Integer id;

**private** String name;

**private** String phone;

**public** Integer getId() {

**return** id;

}

**public** **void** setId(Integer id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getPhone() {

**return** phone;

}

**public** **void** setPhone(String phone) {

**this**.phone = phone;

}

**public** Contact(Integer id, String name, String phone) {

**this**.id = id;

**this**.name = name;

**this**.phone = phone;

}

**public** Contact(){}

}

**编写HtmlActivity.Java：**

HtmlActivity.java

package com.sharpandroid.html;

import java.util.ArrayList;

import java.util.List;

import org.json.JSONArray;

import org.json.JSONObject;

import android.app.Activity;

import android.content.Intent;

import android.net.Uri;

import android.os.Bundle;

import android.webkit.WebView;

import com.sharpandroid.domain.Contact;

public class HtmlActivity extends Activity {

private WebView webView;

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

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

webView = (WebView)this.findViewById(R.id.webView);//浏览器

webView.getSettings().setJavaScriptEnabled(true);//支持js

webView.getSettings().setSaveFormData(false);//不保存表单

webView.getSettings().setSavePassword(false);//不保存密码

webView.getSettings().setSupportZoom(false);//不支持放大功能

webView.addJavascriptInterface(new SharpJavaScript(), "sharp");//addJavascriptInterface方法中要绑定的Java对象

webView.loadUrl("file:///android\_asset/index.html");

}

public class SharpJavaScript{

public void contactlist(){

try {

String json = buildJson(getContacts());

webView.loadUrl("javascript:show('"+ json + "')");//调用js的show()方法

} catch (Exception e) {

e.printStackTrace();

}

}

public void call(String phone){

Intent intent = new Intent(Intent.ACTION\_CALL, Uri.parse("tel:"+ phone));

HtmlActivity.this.startActivity(intent);

}

}

//模拟从数据库或者从网络获取数据

public List<Contact> getContacts(){

List<Contact> contacts = new ArrayList<Contact>();

contacts.add(new Contact(1, "张三", "5554"));

contacts.add(new Contact(2, "李四", "5556"));

contacts.add(new Contact(3, "王五", "5557"));

return contacts;

}

public String buildJson(List<Contact> contacts) throws Exception{

JSONArray array = new JSONArray();

for(Contact contact : contacts){

JSONObject jsonObject = new JSONObject();

jsonObject.put("id", contact.getId());

jsonObject.put("name", contact.getName());

jsonObject.put("phone", contact.getPhone());

array.put(jsonObject);

}

return array.toString();

}

}

关键代码详解：

* webView = (WebView)this.findViewById(R.id.webView);//浏览器

获得webView对象。

* webView.getSettings().setJavaScriptEnabled(true);//支持js

webView.getSettings().setSaveFormData(false);//不保存表单

webView.getSettings().setSavePassword(false);//不保存密码

webView.getSettings().setSupportZoom(false);//不支持放大功能

完成对webView的一些属性设置。

* webView.addJavascriptInterface(new SharpJavaScript(), "sharp");

把Java对象绑定到JavaScript中。在JavaScript中通过“Sharp”可以引用到该对象。

参数一 要绑定的Java对象

参数二 在JavaScript中访问该对象时使用的名字。

* webView.loadUrl("file:///android\_asset/index.html");

加载指定Url路径对应的文件到webView上。

“[file:///android\_asset/index.html](android_assetindex.html)”对应于assets下的index.html文件。

file表示在文件系统中。

/android\_asset对应项目中的assets目录。

* public List<Contact> getContacts()

该方法是为了简化本程序的操作，模拟从数据库或者网络获取数据。我们在实际开发时可以在该方法中实现从数据库或网络获取数据。

* public String buildJson(List<Contact> contacts) throws Exception

把传入的对象数组构建成Json数据返回。

JSONArray是Android提供的一个将数据转化成Json数据的一个工具类。我们可以调用它的相关方法生成Json数据。这与我们自己拼凑Json数据方便了很多。

* public class SharpJavaScript{

public void contactlist(){

try {

String json = buildJson(getContacts());

webView.loadUrl("javascript:show('"+ json + "')");//调用js的show()方法

} catch (Exception e) {

e.printStackTrace();

}

}

该内部类封装了一些操作，供JavaScript调用。

String json = buildJson(getContacts());

获得Json数据。

webView.loadUrl("javascript:show('"+ json + "')");

调用index.html中JavaScript的show()方法，并将json数据传入。

* public void call(String phone){

Intent intent = new Intent(Intent.ACTION\_CALL, Uri.parse("tel:"+ phone));

HtmlActivity.this.startActivity(intent);

}

该方法创建一个拨打电话的意图。

**添加拨号权限**

功能清单文件如下：

AndroidManifest.xml

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<manifest xmlns:android=*"http://schemas.android.com/apk/res/android"*

package=*"com.sharpandroid.phoneuibyhtml"*

android:versionCode=*"1"*

android:versionName=*"1.0"*>

<application android:icon=*"@drawable/icon"* android:label=*"@string/app\_name"*>

<activity android:name=*".PhoneActivity"*

android:label=*"@string/app\_name"*>

<intent-filter>

<action android:name=*"android.intent.action.MAIN"* />

<category android:name=*"android.intent.category.LAUNCHER"* />

</intent-filter>

</activity>

</application>

<uses-sdk android:minSdkVersion=*"7"* />

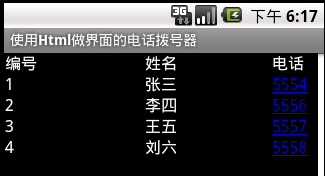
<uses-permission android:name=*"android.permission.CALL\_PHONE"*/>

</manifest>

**运行程序**

将程序发布到sharp模拟器上面。

结果如下图。



为了测试拨号效果，再启动一个AVD，启动listen。

在sharp中点击【5556】号码。

之后sharp中界面如下图。



listen中界面如下。



**项目总结**

整体的流程是，当该应用开始执行时，首先执行PhoneActivity中的onCreate（）方法，在OnCreate（）方法内部首先获取了在main.xml中定义的webView对象，之后通过其关联的webSettings对象对webView进行一系列的初始化设置如是否支持JavaScript，之后将PhoneActivity的一个内部类SharpJavaScript的一个对象通过AddJavaScriptInterface方法绑定到JavaScript中去，以使得JavaScript可以访问该SharpJavaScript类中封装的方法，接着通过loadUrl（）方法将assets/目录下的phoneui.html文件加载到webView中，当index.html页面被加载时，会发生onload事件时，而onload会调用java的contactlist()方法，contactlist()方法获取Json格式的数据，然后调用JavaScript的show（）方法，把Json数据传进去。show（）方法把从java中得到的json数据显示到表格中。

当点击电话号码上的超链接时，会调用java中的call()方法，并把号码传递给call（），call（）方法调用拨号服务，拨打电话。

JavaScript可以调用Java中的方法并传递数据，可以在Java代码中完成对本机资源的访问以及对手机服务的调用。并且在例子中我们看到将Java中获取的数据转化成Json数据传递给JavaScript中。Java中的数据可以来自于SQLite数据库，也可以来自于网络，还可以来自于内容提供者等等。

我们在今后的实际中可以按照本例的方法要求美工做好一个Html界面，进行必要的修改之后，放入assets/目录下，按照示例代码中的做法就可以利用该Html界面了。这样不仅加快开发进度，又能增加界面的美观，为你的应用程序界面打造一幅“美丽的脸庞”！