



大话企业级 Android 开发 · 第八部分

本教程说明及版权声明

- 《大话企业级 Android 开发》是国土工作室为了方便中国 Android 开发者，推动 Android 企业级应用开发，特投入大量心血撰写的书籍，并在网络上免费发布，希望为移动互联网和智能手机时代贡献绵薄之力！所有相关文档版权均属国土工作室所有。
- 本教程是由国土工作室参考官方文档，综合市面相关书籍，经过充分的吸收消化，结合开发实践的一部原创作品，为了本教程及早与广大读者同仁见面、分享，特采用定稿一部分就发布一部分的连载方式发布。读者可以在本博客获取最新内容。
- 未经国土工作室授权，禁止将此文档及其衍生作品以标准（纸质）书籍形式发行。
- 本文档受有关法律的版权保护，对本文档内容的任何未经同意的复制和抄袭行为，将导致相应的法律责任。未经国土工作室同意，任何团体及个人不能用此教程牟利，违者必究。但是：在不收取其他人费用的前提下，您可以自由传播此文档，但必须保证版权信息、文档及其自带标示的完整性。
- 如果对该文档有任何疑问或者建议，请进入官方微博客
<http://www.cnblogs.com/guoshiandroid/>留言或者直接与国土工作室联系（后附联系方式），我们会慎重参考您的建议并根据需要对本文档进行修改，以造福更多开发者！
- 《大话企业级 Android 开发》的最新及完整内容会在国土工作室官方微博客定期更新，请访问国土工作室博客
<http://www.cnblogs.com/guoshiandroid/>获取更多更新内容。





关于国土工作室

我们(国土工作室)是一支专注于 Android 平台企业级应用开发的技术团队,对娱乐多媒体应用有着深刻的理解及研发能力,致力服务于企业用户。为音视频等娱乐多媒体网站、门户网站、SNS、论坛、电子商务等传统网络应用向移动互联网发展提供解决方案和技术支持,为企业提供 Android 培训服务等多种业务。

我们尤其擅长于提供从 Android 客户端到服务端的一站式解决方案和技术支持,服务端可以采用 Java EE,也可以采用轻量级流行的 LAMP 技术体系。目前,研发出了比 KU6、优酷更加强大和完善的 Android 视频网站娱乐多媒体客户端软件,并在持续升级中。

目前,我们正在务实而卓有成效的与音视频等娱乐多媒体网站、门户网站、SNS、论坛、电子商务等传统网络服务商合作,发展迅速,渴望有志之士的加入,和我们一起为成为世界最好的 Android 软件开发和咨询、培训公司而奋斗,为移动互联网和智能手机时代贡献力量!

联系我们

电话:15711060468

Email:guoshiandroid@gmail.com

博客: <http://www.cnblogs.com/guoshiandroid/>





1.1 常用的几种通知

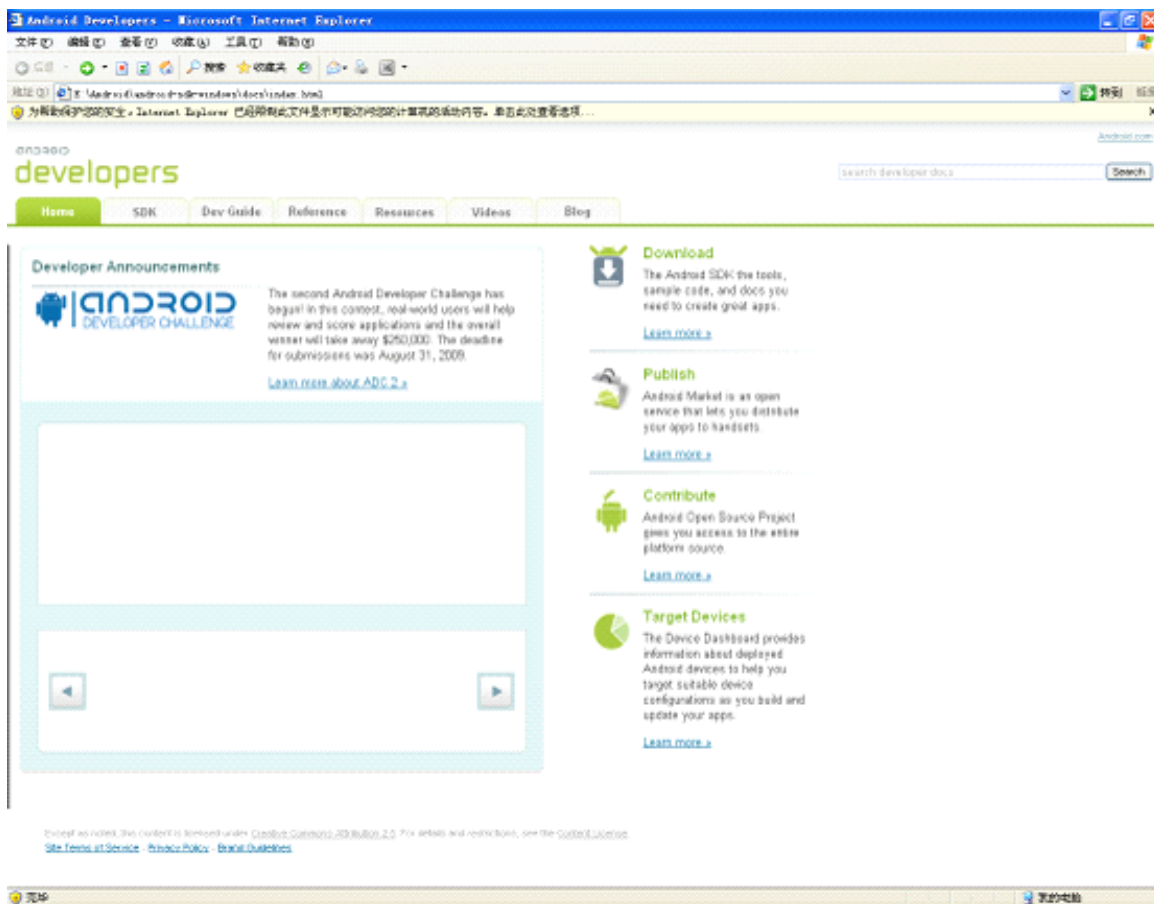
Android 中的通知有哪些种呢？例如以下几种可能的情况出现，需要你来通知应用程序的用户。

- 当一个文件事件，如保存完成后，应该会出现一个小的消息，以确认保存成功。
- 如果应用程序在后台运行，需要用户的注意，应用程序应该创建一个 notification，允许用户在他或她方便时回应。
- 如果应用程序正在执行的工作，用户必须等待文件（如加载），应用程序应该显示进度条。

我们可以通过查看帮助文档来学习通知的信息。也可以通过这个过程再次熟悉如何查阅帮助文档。

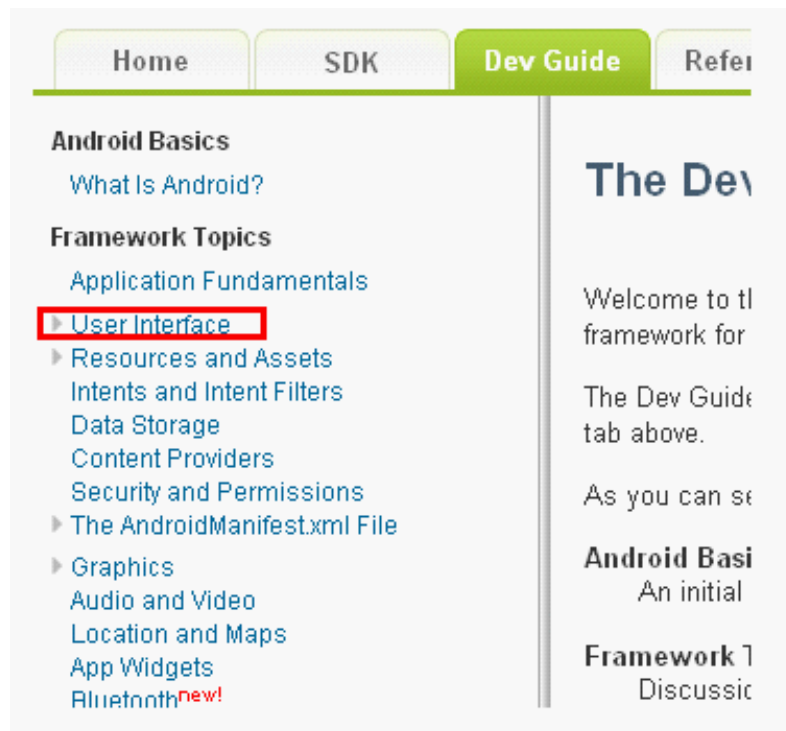
到 sdk 解压目录下找到 docs 子目录，目录是

“E:\Android\android-sdk-windows\docs”，打开其中的 index.html 文件，页面如下图所示：

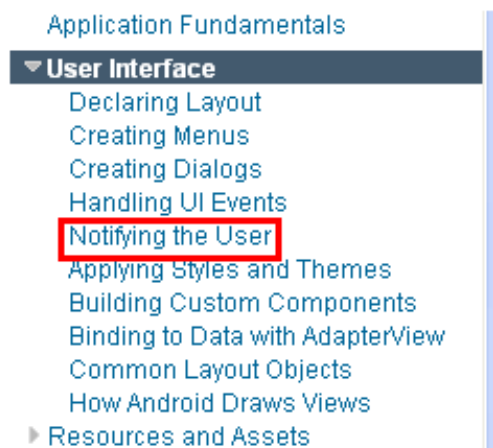


之后点击【Dev Guide】，界面如下。



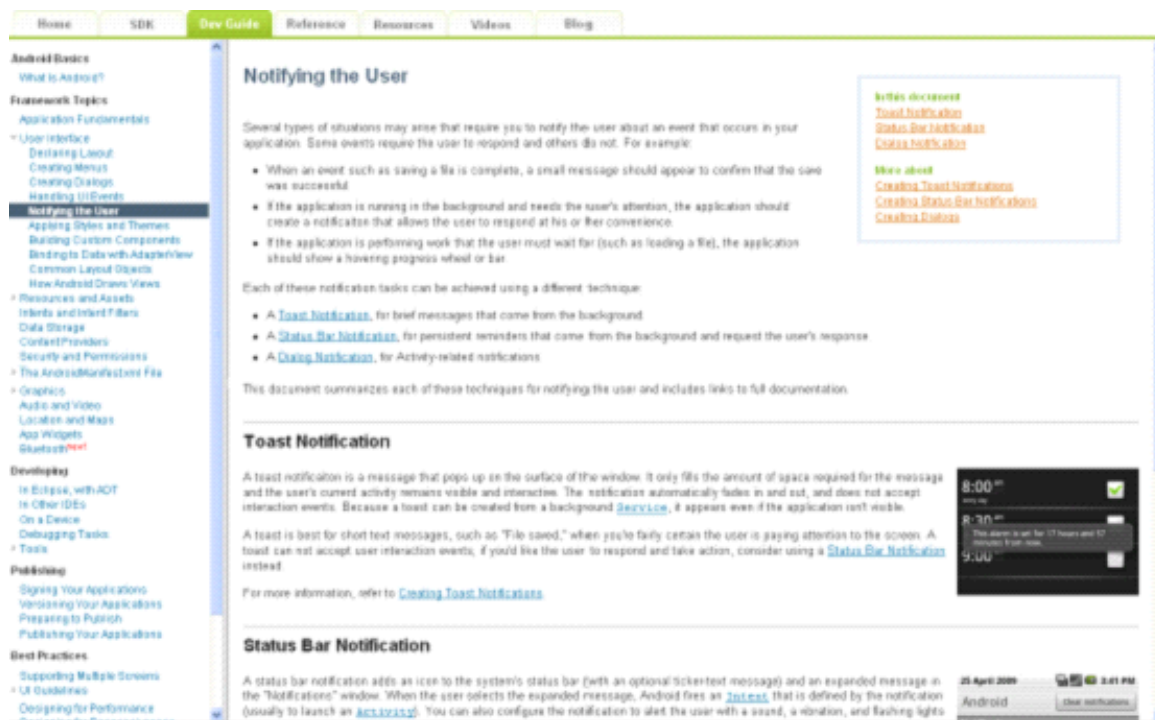


点击红框中的“User Interface”展开其子目录。结果如下图：

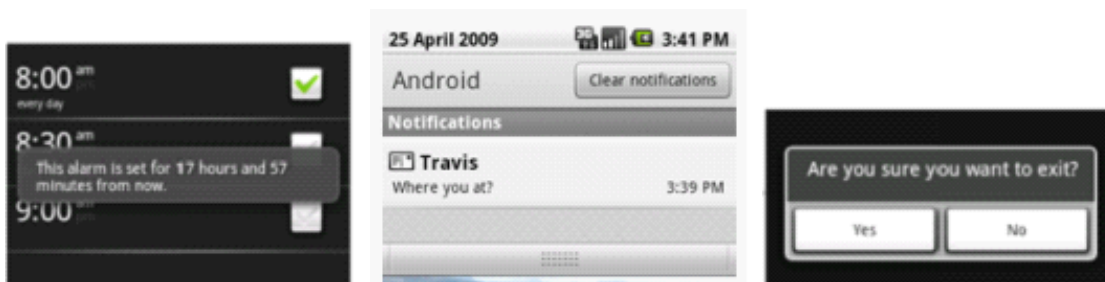


点击红框中的“Notifying the User”，界面如下图所示。





右侧的主区域列出了 Android 中的几种产生通知的方式。共三种, 分别是 [Toast Notification](#), [Status Bar Notification](#), [Dialog Notification](#)。即吐司通知、状态栏通知、对话框通知。其中对话框通知主要是当需要用户做出确定或其他某种选择时使用, 吐司通知只是简单的告知用户发生了什么事情, 状态栏通知最典型的一种就是当收到短信时会在状态栏上显示一个通知, 可以用鼠标点击状态栏并向下拖动, 以查看是否有新的状态栏通知。吐司通知、状态栏通知、对话框通知的风格依次如下图。下面将会介绍状态栏通知和对话框通知, 而吐司通知比较简单, 前面已经用到过, 因此不单独介绍。



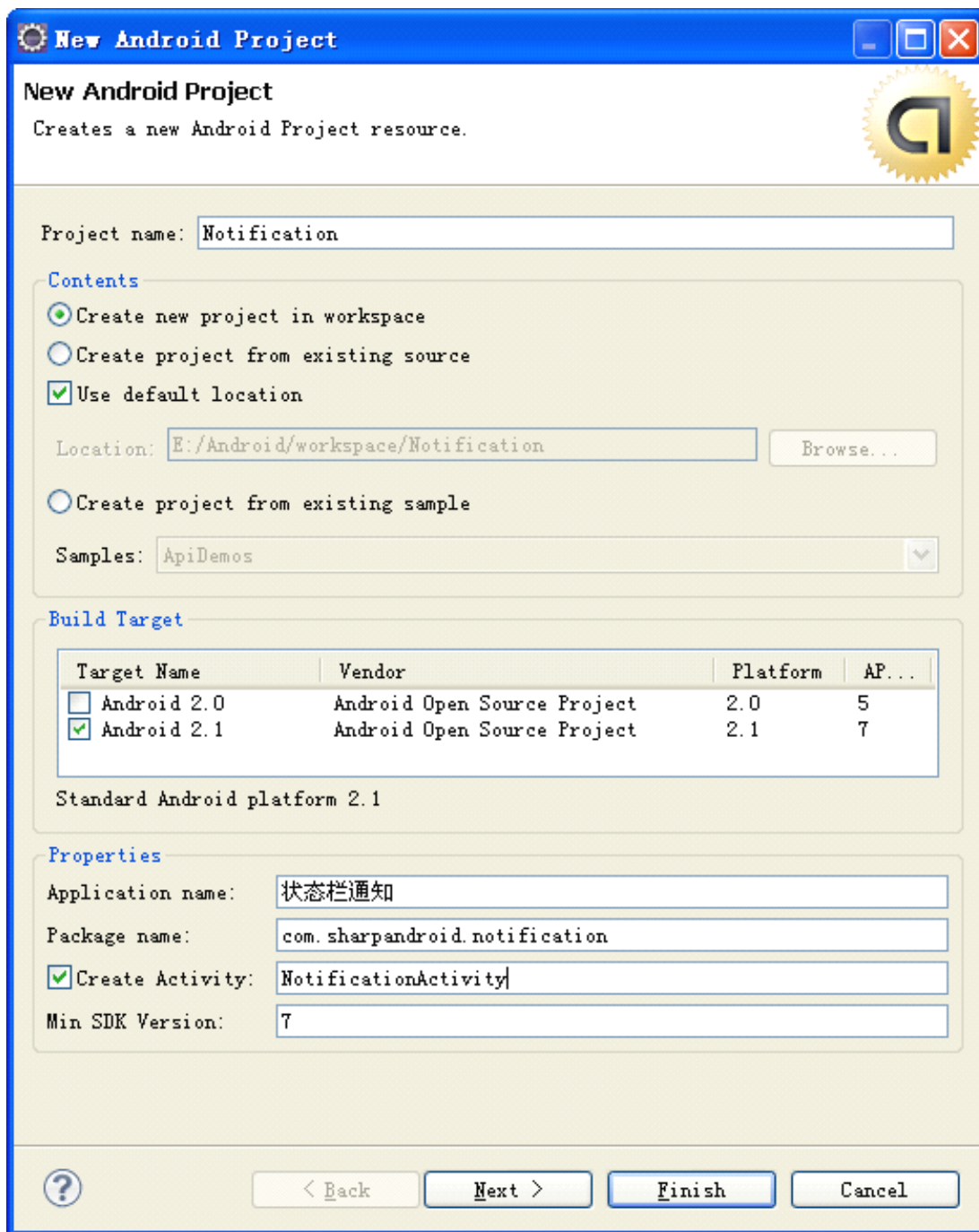
1.1.1 状态栏通知

下面是一个状态栏通知的开发示例。

创建项目

创建一个名为“Notification”的项目, 如下图:



**编写 strings.xml:**

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World, NotificationActivity!</string>
    <string name="app_name">状态栏通知</string>
    <string name="send">发送通知</string>
</resources>
```

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"
    >
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/send"
    android:id="@+id/button"
    />
</LinearLayout>
```

编写 NotificationActivity.java

```
package com.sharppandroid.notification;

import android.app.Activity;
import android.app.AlertDialog;
import android.app.Notification;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class NotificationActivity extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        Button button = (Button)findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                //获取通知管理器
                NotificationManager mNotificationManager = (NotificationManager)
getSystemService(Context.NOTIFICATION_SERVICE);

                int icon = android.R.drawable.sym_action_email;
                long when = System.currentTimeMillis();
```





音

```

//新建一个通知, 指定其图标和标题
//第一个参数为图标, 第二个参数为标题, 第三个为通知时间
Notification notification = new Notification(icon, null, when);
notification.defaults = Notification.DEFAULT_SOUND;//发出默认声

//当点击消息时就会向系统发送 openintent 意图

PendingIntent contentIntent =
PendingIntent.getActivity(NotificationActivity.this, 0, null, 0);

notification.setLatestEventInfo(NotificationActivity.this, "开会通知", "今天下午4点到会议室开会!", contentIntent);
mNotificationManager.notify(0, notification);//发送通知
    }
});
}
}
}

```

说明:

- NotificationManager mNotificationManager = (NotificationManager) getSystemService(Context.[NOTIFICATION_SERVICE](#));
该行代码的作用是获取通知管理器。
getSystemService 方法是获取系统服务, 其参数 Context.[NOTIFICATION_SERVICE](#) 是 Context 类中的静态常量。返回的类型是 Object, 此行代码返回的是 NotificationManager 类型, 加以强转就得到一个通知管理器。
在 Eclipse 中, 按住 “Ctrl” 键, 同时用鼠标点击代码 [NOTIFICATION_SERVICE](#), 就可以查看它在 Context 类中的定义如下:
[public static final String NOTIFICATION_SERVICE = "notification";](#)
- int icon = android.R.drawable.sym_action_email;
获取一个图片的索引。该图片是系统自带的。当然也可以自己在 res 中添加图片, 然后再进行索引。Android 系统中带有很多的图片, 可以用上述方式进行引用。
- long when = System.currentTimeMillis();
获取系统时间
- Notification notification = new Notification(int icon, CharSequence tickerText, long when);
新建一个通知。参数介绍如下:
icon : 该通知将采用的图片
tickerText : 通知标题, 该通知的标题将在后面设置, 因此此处置空
when : 通知的时间
- notification.defaults = Notification.[DEFAULT_SOUND](#);
设置通知的声音。但在 AVD 上没有声音, 因此, 无法听到。
- PendingIntent contentIntent =
PendingIntent.getActivity(NotificationActivity.this, 0, null, 0);
创建一个 contentIntent。
- notification.setLatestEventInfo ([Context](#) context, [CharSequence](#) contentTitle, [CharSequence](#) contentText, [PendingIntent](#) contentIntent)





该行代码设置通知的相关信息。


context : 应用上下文
contentTitle : 通知的标题
contentText : 通知的内容
contentIntent : 当该通知被点击时，发出的意图。

- `mNotificationManager.notify(int id, Notification notification);`
调用通知管理器的 `notify` 方法发出该通知。其中：
id : 要发送的通知的标识码
notification: 要发送的通知

运行程序

运行后，点击【发送通知】按钮，效果如图：



可以看到状态栏上面有了通知 。向下拖动状态栏，看到通知的内容。

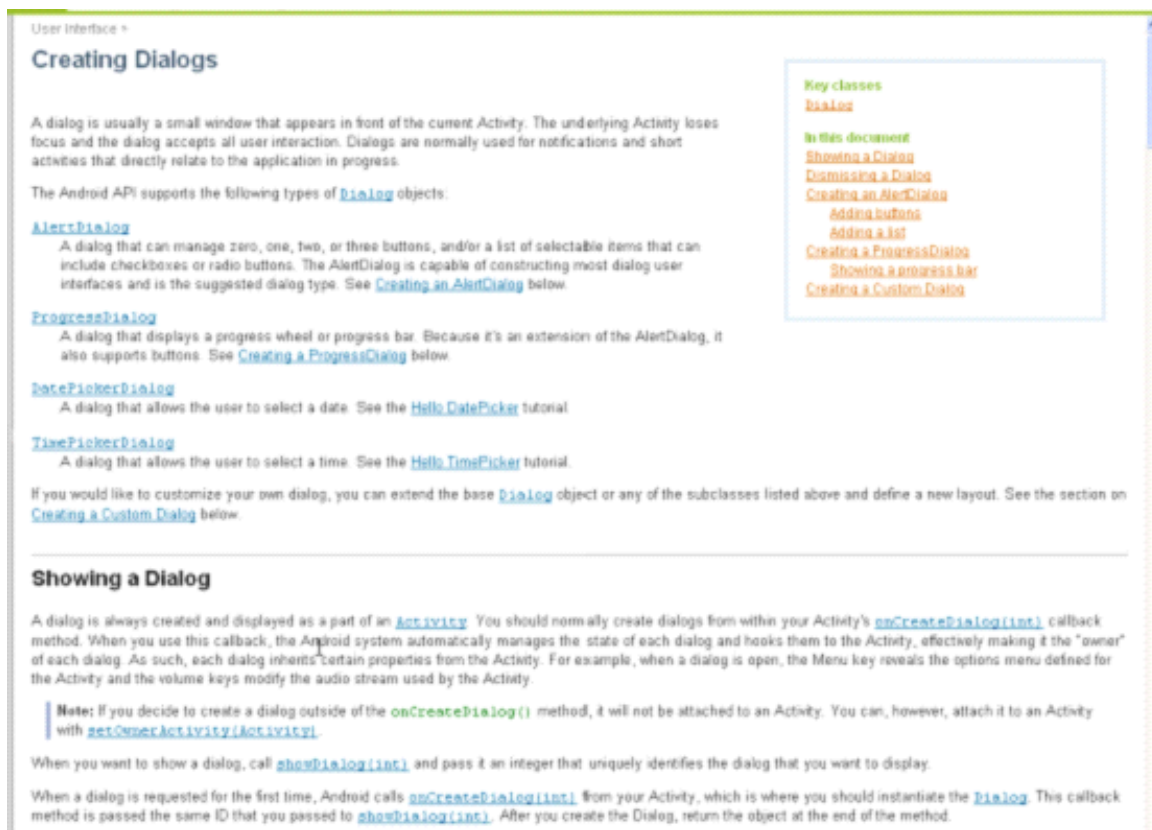


点击【清除】，可以将通知删除。

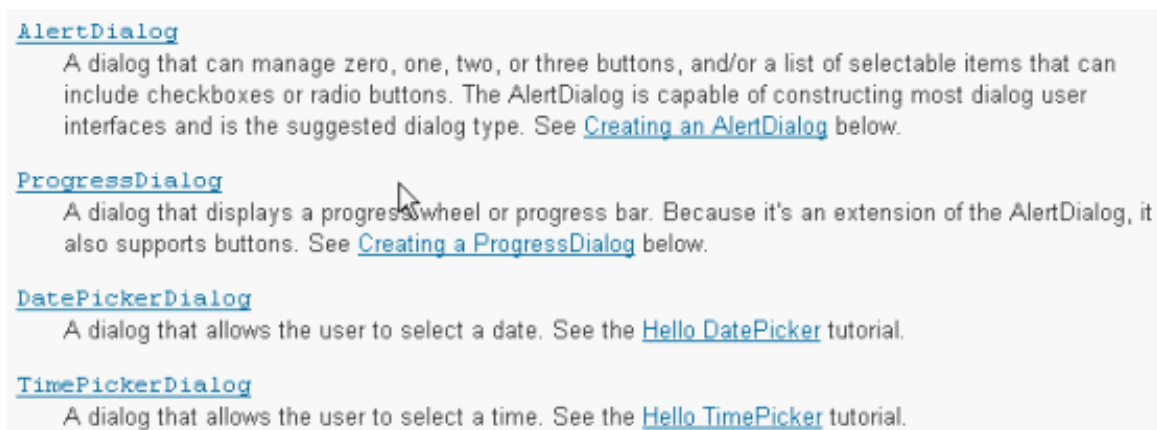
1.1.2 警告对话框

关于对话框我们在前面已经简单的介绍过，下面我们来开发一个对话框实例。我们进入帮助文档中的【Notifying the User】页面，滚动到【Dialog Notification】位置。点击【[Creating Dialogs.](#)】超链接，界面如下。





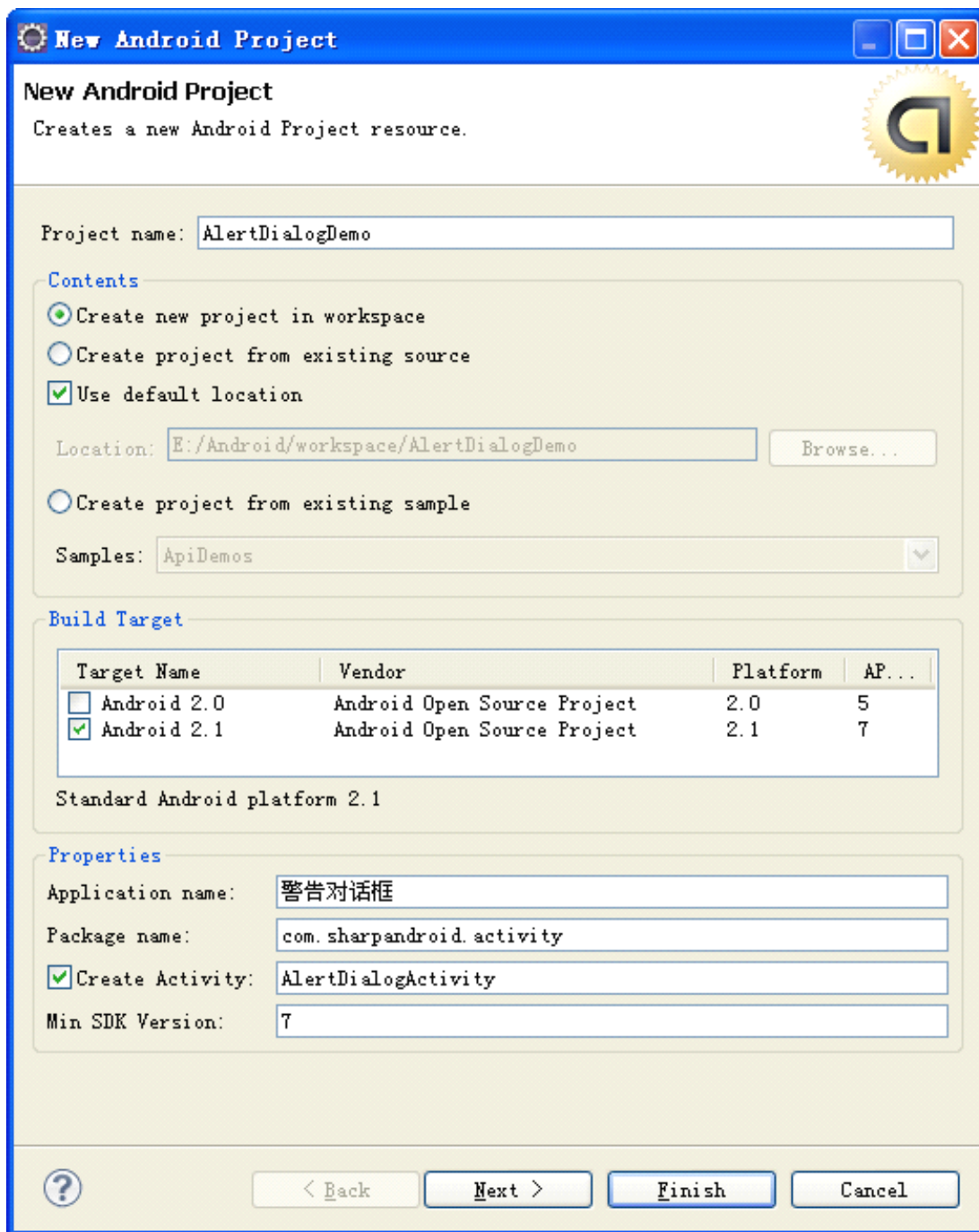
建议阅读该页面上的内容。常用的对话框通知有如下几种：



AlertDialog（警告对话框）、ProgressDialog（进度对话框）、DatePickerDialog（日期选择对话框）、TimePackerDialog（时间选择对话框）。在本节创建一个警告对话框。其他几种对话框在后面遇到时再讲解。

创建一个名为“AlertDialogDemo”的项目如下图：





编写 strings.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World, AlertDialogActivity!</string>
    <string name="app_name">警告对话框</string>
    <string name="button">弹出对话框</string>
</resources>
```

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"
    >
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/button"
        android:id="@+id/button"
    />
</LinearLayout>
```

编写 AlertDialogActivity.java

创建警告对话框的代码比较简单, 在文档中就有现成的例子, 实际开发中我们可以将这些示例代码拷贝到我们的项目中, 进行必要的修改就可以让这些代码为我所用了。

文档中创建 AlertDialog 的代码如下, 首先将它拷贝到我们项目中创建 AlertDialog 的地方。

```
AlertDialog.Builder builder = new AlertDialog.Builder(this);
builder.setMessage("Are you sure you want to exit?")
    .setCancelable(false)
    .setPositiveButton("Yes", new DialogInterface.OnClickListener() {
        public void onClick(DialogInterface dialog, int id) {
            MainActivity.this.finish();
        }
    })
    .setNegativeButton("No", new DialogInterface.OnClickListener() {
        public void onClick(DialogInterface dialog, int id) {
            dialog.cancel();
        }
    });
AlertDialog alert = builder.create();
```

拷贝完成之后, 根据我们的需要修改一部分内容之后的 AlertDialogActivity.java 类的代码如下。

```
package com.sharpandroid.activity;

import android.app.Activity;
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
```





```
public class AlertDialogActivity extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        Button button = (Button)findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                AlertDialog.Builder builder =
                    new
AlertDialog.Builder(AlertDialogActivity.this);
                builder.setTitle("sharpandroid")
                    .setMessage("你确定要访问 Android 网站吗? ")
                    .setCancelable(false)
                    .setPositiveButton("确定",
new DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface dialog, int id)
{
                            //创建一个访问 "http://www.android.com" 网站
的意图,

                            //该意图会告知系统打开浏览器, 并访问该网址。
                            Intent intent =
new Intent(Intent.ACTION_VIEW, Uri.parse("http://www.android.com"));
                            startActivity(intent);
                        }
                    })
                    .setNegativeButton("取消",
new DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface dialog, int id)
{
                            dialog.cancel(); //删除对话框
                        }
                    });
                AlertDialog alert = builder.create();//创建对话框
                alert.show();//显示对话框
            }
        });
    }
}
```

说明:

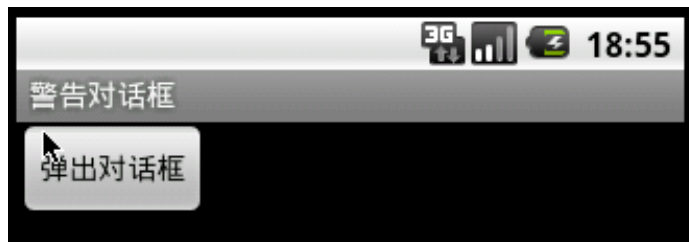




- AlertDialog.Builder builder = new AlertDialog.Builder(AlertDialogActivity.this);
创建一个 AlertDialog 类的内部类 Builder 的对象。
- builder.setTitle("sharpandroid")
为 builder 设置标题
- setMessage("你确定要访问 Android 网站吗? ")
为 builder 设置消息内容
- setCancelable(false)
设置为 false 后，键盘上的后退键失效。
- setPositiveButton("确定",
new DialogInterface.OnClickListener() {
 public void onClick(DialogInterface dialog, int id) {
 //创建一个访问 "http://www.android.com" 网站的意图，
 //该意图会告知系统打开浏览器，并访问该网址。
 Intent intent = new Intent(Intent.ACTION_VIEW,
 Uri.parse("http://www.android.com"))
 ;
 startActivity(intent);
 }
})
添加确定按钮，并添加对该按钮的处理事件。本例中的处理事件是访问 <http://www.android.com> 网站。
- .setNegativeButton("取消", new DialogInterface.OnClickListener() {
 public void onClick(DialogInterface dialog, int id)
 {
 dialog.cancel(); //删除对话框
 }
});
添加拒绝按钮，并添加处理事件，处理结果是删除该对话框。

运行程序


运行程序到 sharp 模拟器上。结果图如下：



点击【弹出对话框】按钮。界面如下：





首先，点击键盘上的  按钮，观察界面是否后退，结果没有反应。这是由于设置了 `setCancelable(false)`。之后点击【取消】，回到图??的界面，再此点击【弹出对话框】按钮，界面如下：



访问 `www.android.com` 成功。

优化代码

将创建 `AlertDialog` 的代码优化为如下所示，





```
new AlertDialog.Builder(AlertDialogActivity.this)
    .setTitle("sharpandroid")
    .setMessage("你确定要访问 Android 网站吗? ")
    .setCancelable(false)
    .setPositiveButton("确定", new DialogInterface.OnClickListener() {
        public void onClick(DialogInterface dialog, int id) {
            //创建一个访问 "http://www.android.com" 网站的意图,
            //该意图会告知系统打开浏览器, 并访问该网址。
            Intent intent = new Intent(Intent.ACTION_VIEW,
                                      Uri.parse("http://www.android.com"));
            startActivity(intent);
        }
    })
    .setNegativeButton("取消", new DialogInterface.OnClickListener() {
        public void onClick(DialogInterface dialog, int id) {
            dialog.cancel(); //删除对话框
        }
    }).show(); //显示对话框
```

说明:

优化后的代码直接采用匿名内部类, 并且省去了 create() 方法, 直接调用 show() 方法, 这样看起来更加简洁。上面代码采用的是一个链式调用, 像 setTitle()、setMessage() 这些方法, 他们的返回值都是当前对话框对象。

在文档中有关于其他几种对话框的例子, 可以参照本节的做法自行学习。自学能力和举一反三的模仿能力是每个程序员必须具备的良好素质。

再次执行, 结果与上次相同, 不再赘述。

1.2 UI 综合应用——用户注册

前面我们介绍了关于用户界面的常用基本知识, 接下来以网络中最常用的登陆功能功能为例, 做一个应用示例, 以对 UI 的知识做一个应用整合, 帮助理解和掌握 UI 知识。

用户界面构思:

本例主要在于为用户介绍各种布局相互搭配使用, 与各种控件的应用。一个注册的页面, 有相应的信息输入框, 还有相应的拼写检查。

1. 新建项目

创建一个名为 "UserLogin" 的项目。





Project name:

Contents

☒ Create new project in workspace
☐ Create project from existing source
☒ Use default location

Location:

☐ Create project from existing sample

Samples:

Build Target

Target Name	Vendor	Platform	API ...
<input type="checkbox"/> Android 2.0	Android Open Source Project	2.0	5
<input type="checkbox"/> Android 2.0.1	Android Open Source Project	2.0.1	6
<input checked="" type="checkbox"/> Android 2.1	Android Open Source Project	2.1	7

Standard Android platform 2.1

Properties

Application name:

Package name:

☒ Create Activity:

Min SDK Version:

2. UI 设计

2.1. 构思: 在设计 UI 初期, 可以先使用画图工具将大致草图画出来。





根据草图，仅仅使用 `LinearLayout` 布局是不够的，这里我们需要与 `RelativeLayout` 布局嵌套使用。我们可以将用户名和用户的输入框划分为一个 `RelativeLayout` 区域，依次类推。这样基本的布局框架就搭建起来了。

条件：用户名、密码、年龄非空。密码设为不可见。

编写 `strings.xml`:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<resources>
    <string name="hello">Hello World, LoginActivity!</string>
    <string name="app_name">用户注册界面</string>
    <string name="nameString">用户名: </string>
    <string name="ageString">年龄: </string>
    <string name="registerButtonText">注册</string>
    <string name="sexString">性别: </string>
    <string name="favoriteString">喜好: </string>
    <string name="cityString">城市: </string>
    <string name="passString">密码: </string>
    <string name="pingpang">乒乓球</string>
    <string name="basketball">篮球</string>
    <string name="football">足球</string>
    <string name="tennis">网球</string>
</resources>
```

编写 `parameters.xml`:

新建了一个资源文件 `parameters.xml`，存放了一些属性信息，如字体大小、`TextView` 以及 `EditText` 的宽度。代码如下：

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





```
<dimen name="fontSize">22px</dimen>
<dimen name="TextViewWidth">90px</dimen>
<dimen name="EditTextWidth">160px</dimen>
</resources>
```

编写 main.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">
    <RelativeLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content">
        <TextView
            android:layout_width="@dimen/TextViewWidth"
            android:layout_height="wrap_content"
            android:text="@string/nameString"
            android:textSize="@dimen/fontSize"
            android:id="@+id/name"/>
        <EditText
            android:layout_width="@dimen/EditTextWidth"
            android:layout_height="wrap_content"
            android:layout_toRightOf="@id/name"
            android:layout_alignTop="@id/name"
            android:id="@+id/nameValue"/>
    </RelativeLayout>
    <RelativeLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content">
        <TextView
            android:layout_width="@dimen/TextViewWidth"
            android:layout_height="wrap_content"
            android:text="@string/passString"
            android:textSize="@dimen/fontSize"
            android:id="@+id/pass"/>
        <EditText
            android:password="true"
            android:layout_width="@dimen/EditTextWidth"
            android:layout_height="wrap_content"
            android:layout_toRightOf="@id/pass"
            android:layout_alignTop="@id/pass"
            android:id="@+id/passValue"/>
    </RelativeLayout>
    <RelativeLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content">
        <TextView
            android:layout_width="@dimen/TextViewWidth"
            android:layout_height="wrap_content"
            android:text="@string/ageString"
            android:textSize="@dimen/fontSize"
            android:id="@+id/age"/>
        <EditText
            android:numeric="integer"
```





```
        android:layout_width="@dimen/EditTextWidth"
        android:layout_height="wrap_content"
        android:layout_toRightOf="@id/age"
        android:layout_alignTop="@id/age"
        android:id="@+id/ageValue"/>
    </RelativeLayout>
    <RelativeLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content">
        <TextView
            android:layout_width="@dimen/TextViewWidth"
            android:layout_height="wrap_content"
            android:text="@string/sexString"
            android:textSize="@dimen/fontSize"
            android:id="@+id/sex"/>
        <RadioGroup
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_toRightOf="@id/sex"
            android:checkedButton="@+id/radioMan"
            android:orientation="horizontal"
            android:id="@+id/sexMenu">
            <RadioButton android:text="男" android:id="@id/radioMan"/>
            <RadioButton android:text="女"
android:id="@+id/radioWoman"/>
        </RadioGroup>
    </RelativeLayout>
    <RelativeLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content">
        <TextView
            android:layout_width="@dimen/TextViewWidth"
            android:layout_height="wrap_content"
            android:text="@string/favoriteString"
            android:textSize="@dimen/fontSize"
            android:id="@+id/favorite"/>
        <CheckBox
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_toRightOf="@id/favorite"
            android:text="@string/pingpang"
            android:id="@+id/checkboxpingpang"/>
        <CheckBox
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_toRightOf="@id/checkboxpingpang"
            android:text="@string/football"
            android:id="@+id/checkboxfootball"/>
        <CheckBox
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_toRightOf="@id/favorite"
            android:layout_below="@id/checkboxfootball"
            android:text="@string/basketball"
            android:id="@+id/checkboxbasketball"/>
        <CheckBox
            android:layout_width="wrap_content"
```





```
        android:layout_height="wrap_content"
        android:layout_toRightOf="@id/checkboxbasketball"
        android:layout_alignTop="@id/checkboxbasketball"
        android:text="@string/tennis"
        android:id="@+id/checkboxtennis"/>
    </RelativeLayout>
    <RelativeLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content">
        <TextView
            android:layout_width="@dimen/TextViewWidth"
            android:layout_height="wrap_content"
            android:text="@string/cityString"
            android:textSize="@dimen/fontSize"
            android:id="@+id/city"/>
        <Spinner
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_toRightOf="@id/city"
            android:id="@+id/cityItems">
        </Spinner>
    </RelativeLayout>
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/registerButtonText"
        android:id="@+id/registerButton"/>
</LinearLayout>
```

说明:

- 参数设置: 这里我们要做的是个 UI, 所以要考虑到应用的美观, 所以, 推荐相应的控件不要再使用 `wrap_content` 来设置控件的宽, 使用数值可以使你的应用看起来更规矩。而且希望在自己编写的时候, 要遵从 MVC 的设计模式, 在 `/res/values/` 下建立一个专门存放页面参数的 xml。这里的作用, 做过网页开发的, 联想一下 CSS 的作用。

```
<dimen name="fontSize">22px</dimen>
```

存放 px (pixels), in (inches), mm (millimeters), pt (points at 72 DPI) 类型的数据。

应用:

```
android:textSize="@dimen/fontSize"/>
```

- 布局的特别属性:

在这里我们需要用到 `RelativeLayout` 的个别属性, 例如:

```
android:layout_toRightOf="@id/age"
```

这是与控件 age 向右对齐。

- 输入框的输入限制:

显然我们在填写年龄的时候, 不能填写入非数字的文本所以, 在 UI 中我们需要对才做限制: `android:numeric="integer"`, 将用户的输入限制在整形数字。其他输入无效。

具体的 layout 设置完以后, 我们通过 Activity 来进行显示与控制:

编写 LoginActivity.java





```
package com.sharpandroid.UserLogin;

import java.util.ArrayList;
import java.util.List;

import android.app.Activity;
import android.app.AlertDialog;
import android.app.ProgressDialog;
import android.content.DialogInterface;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.Spinner;

public class LoginActivity extends Activity {

    private static final String[] cities = { "北京", "上海", "广州", "郑州" };
    private EditText name, age, pass;
    private Button regButton;
    private RadioGroup sexRadioGroup;
    private CheckBox basketball, football, pingpang, tennis;
    private Spinner cityItems;

    private boolean flag = true;
    private List<CheckBox> favorites;

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        //定义一个 ArrayList, 用来存放所有的 CheckBox
        favorites = new ArrayList<CheckBox>();

        //得到相应的显示控件的对象
        name = (EditText) findViewById(R.id.nameValue);
        age = (EditText) findViewById(R.id.ageValue);
        pass = (EditText) findViewById(R.id.passValue);
        regButton = (Button) findViewById(R.id.registerButton);
        cityItems = (Spinner) findViewById(R.id.cityItems);
        sexRadioGroup = (RadioGroup) findViewById(R.id.sexMenu);

        basketball = (CheckBox) findViewById(R.id.checkboxbasketball);
        //将 basketball 对象添加到 favorites 中
        favorites.add(basketball);

        football = (CheckBox) findViewById(R.id.checkboxfootball);
        favorites.add(football);

        pingpang = (CheckBox) findViewById(R.id.checkboxpingpang);
        favorites.add(pingpang);

        tennis = (CheckBox) findViewById(R.id.checkboxtennis);
```





```
        favorites.add(tennis);

        //创建一个数组型适配器, 并将 cities 中的数据
        ArrayAdapter<String> adapter = new ArrayAdapter<String>(
            LoginActivity.this,
            android.R.layout.simple_spinner_item,
            cities);

        adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
        cityItems.setAdapter(adapter);
        //为 regButton 注册一个点击事件监听器
        regButton.setOnClickListener(new View.OnClickListener() {
            //当按钮被点击时调用该方法
            @Override
            public void onClick(View v) {

                flag = addUser();
                if (flag)
                    //创建一个 AlertDialog 对话框显示登陆的信息。
                    new
AlertDialog.Builder(LoginActivity.this).setTitle(
                        "请确认信息").setMessage(
                        "您的信息如下: " + "\n" + "姓名: "
                        + name.getText().toString() + "\n" + "年龄: "
                        + age.getText().toString() + "\n" + "性别: "
                        + getSex() + "\n" + "爱好: " + getFavorite()
                        + "\n" + "城市: " + getCity() + "\n")
                    .setCancelable(false).setPositiveButton("确定",
                        new DialogInterface.OnClickListener() {
                            public void onClick(
                                DialogInterface dialog, int id) {
                                    ProgressDialog.show(
                                        LoginActivity.this,
                                        "用户信息注册中", "请耐心等待.....")
                                        .setCancelable(true);
                                }
                            }).setNegativeButton("修改",
                                new DialogInterface.OnClickListener() {
                                    public void onClick(
                                        DialogInterface dialog, int id) {
                                            dialog.cancel(); // 删除对话框
                                        }
                                    }).show(); // 显示对话框
                    });
            }
        });

        //获取 Spinner 中的值
        private String getCity() {
            return cities[cityItems.getSelectedItemPosition()];
        }

        //获取 CheckBox 的值
        private String getFavorite() {
            String favString = "";
            for (CheckBox cb : favorites) {
```





```
        if (cb.isChecked()) {
            favString += cb.getText().toString();
            favString += ",";
        }
    }
    if (favString != "") {
        favString = favString.substring(0, favString.length() - 1);
    } else {
        favString = "您没有选择爱好! ";
    }
    return favString;
}
//获取一组 RadioGroup 中被选中的 RadioButton 的值
private String getSex() {
    RadioButton mRadio = (RadioButton) findViewById(sexRadioGroup
        .getCheckedRadioButtonId());
    return mRadio.getText().toString();
}

/**
 * 拼写检测, 检测输入内容是否合乎要求
 */
public boolean addUser() {
    if (name.getText().toString().length() == 0) {
        name.setError("用户名不能为空");
        return false;
    }
    if (age.getText().toString().length() == 0) {
        age.setError("年龄不能为空");
        return false;
    }
    if (pass.getText().toString().length() == 0) {
        pass.setError("密码不能为空 ");
        return false;
    }
    return true;
}
}
```

说明:

- `age.setError("年龄不能为空");`
添加页面检测功能, 当用户点击提交信息, 如果用户输入的空, 出现错误提示信息。
- `new AlertDialog.Builder(LoginActivity.this).……. show();`
创建一个对话框, 并设置对话框中弹出的内容。
- `ProgressDialog.show(LoginActivity.this, "用户信息注册中", "waiting...");`
对话框的显示。模拟一下用户注册的页面, 当用户点击注册按钮, 首先对输入框进行检测, 如果通过则弹出对话框。

执行应用:





错误提示，没有输入用户名，点击注册，出现错误提示，如下图：





按如下界面填写内容，





用户注册界面

用户名： androidabc08

密码：

年龄： 29

性别： ☒ 男 ☐ 女

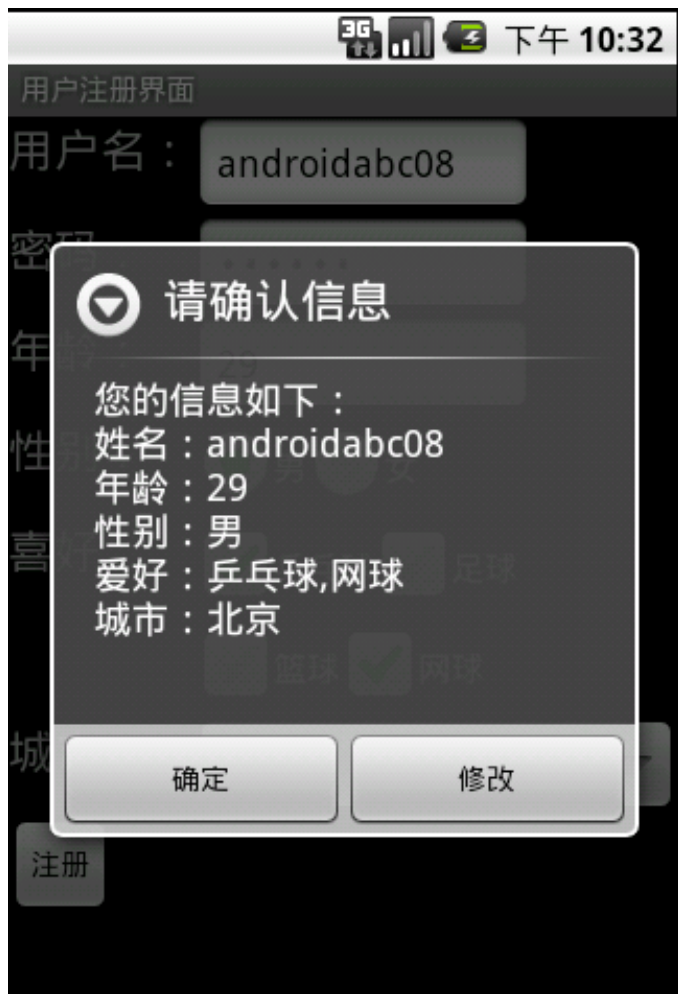
喜好： ☒ 乒乓球 ☒ 足球
☒ 篮球 ☒ 网球

城市： 北京

注册

点击 【注册】，





点击【修改】返回前一个界面，点击【确定】，效果如下，由于并没有连接后台数据库等，因此仅在此以进度对话框提示，表示正在注册。



