# Android移动端开发设计文档

# APP功能模块

|  |  |
| --- | --- |
| **模块** | **实现功能** |
| 待办事项 | 1. 优化了单调的待办事项管理，可对待办事项设置分类（默认只有四种）、重要程度（4种）、事项备注，并根据不同的分类和重要程度显示不同的样式 2. 分类栏点击不同的分类可以筛选出对应分类的事项 3. 右上角选中隐藏已完成的按钮，可以隐藏标记已完成的事项 4. 点击右下角的添加按钮弹出添加事项的窗口，设置事项信息并添加 5. 点击某一条事项时，在弹出的对话框可以对其修改分类、修改名称、修改重要程度、修改备注、标记失败、删除事项 |
| 专注计时 | 1. 点击小三角按钮可以开始计时，开始计时后可以进行暂停或取消 2. 第三个完成的按钮，点击后将结束当前计时，并将本次计时的记录（计时时间和结束时的系统时间）显示在下方列表 |
| 音乐播放 | 1. 获取本地目录中的音乐文件，显示在列表，点击某一首音乐可以进行播放 2. 下方可以进行音乐的暂停和继续，以及取消音乐播放 |
| 每日先知 | 1. 实时查看本地天气 2. 显示当日头条新闻 |

# 关键功能的实现

**一、底部菜单栏切换页**

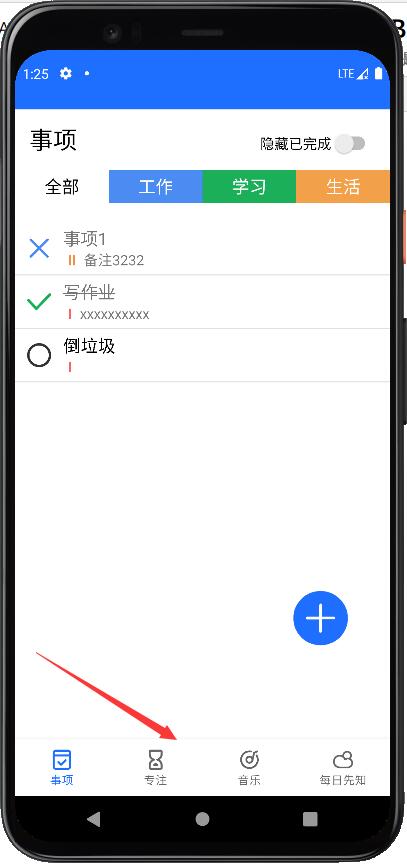
1.添加依赖

2.在res资源文件夹下新建一个menu文件夹，创建底部导航的菜单布局文件

* 创建对应数量的item，为每个菜单栏选项
* 给每个item定义title（标题），icon（图标）

3.在activity\_main布局页面引入 com.google.android.material.bottomnavigation.BottomNavigationView 控件

4.依次创建每个页面的Fragment类及布局文件，如Task页面

5.在MainActivity.java中进行设置BottomNavigation选择监听事件对fragment进行管理。

**二、每日先知界面**

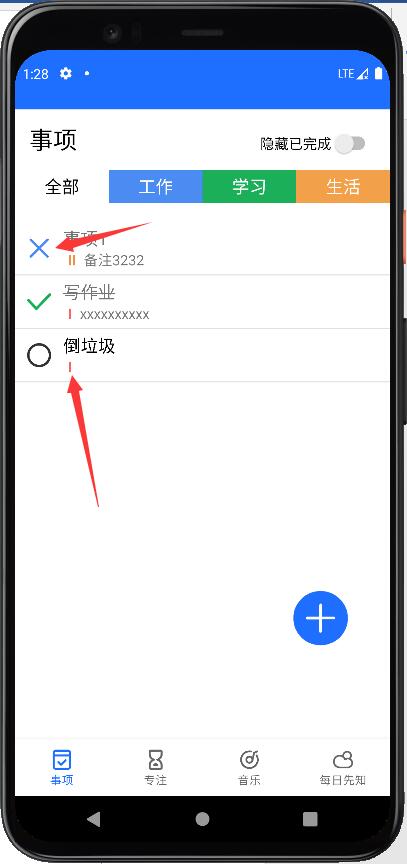
获取天气API接口，这里以临海市为例。使用OkHttp请求天气数据

**由于OkHttp的请求是在子线程中进行的，需要使用Handler消息队列机制将解析出来的Weather实例发送到主线程用以显示在界面上。**

****

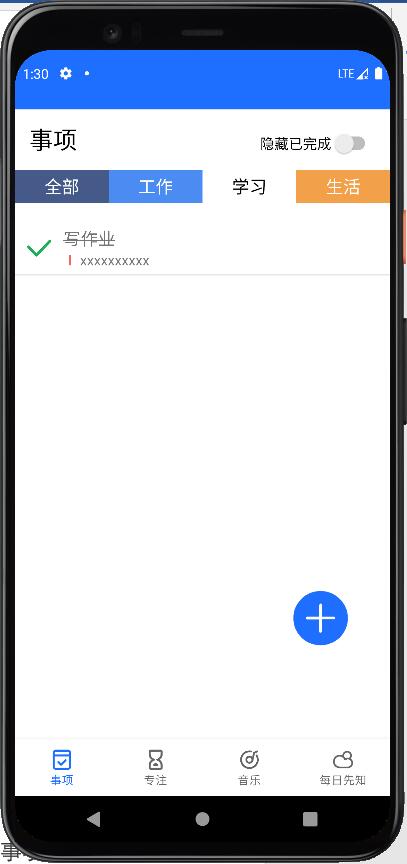
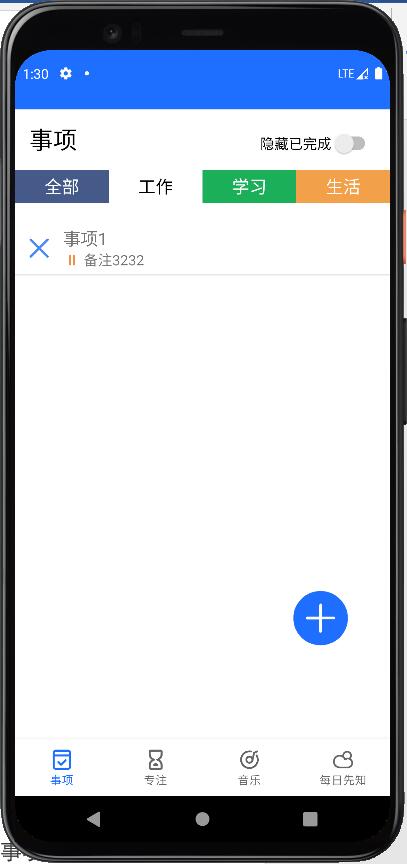
**三、待办事项界面**

**由于ListView是放在Fragment中的，所以直接在MainAcitivity.java中设置适配器可能会出现数据没法显示的bug。所以我直接把从数据库获取数据，Adapter的定义，ListView设置适配器的模块搬到了TaskFragment.java中。**

**建立ListView适配器。这里涉及到缓存convertView的使用**，使用convertView可以防止每创建一个item时就解析一个布局，这样效率肯定不高。convertView是Android提供的用于缓存的View，在第一次渲染item时，将将解析出来的View放入缓存convertView，在下一次渲染item的时候，判断convertView是否为空即可。

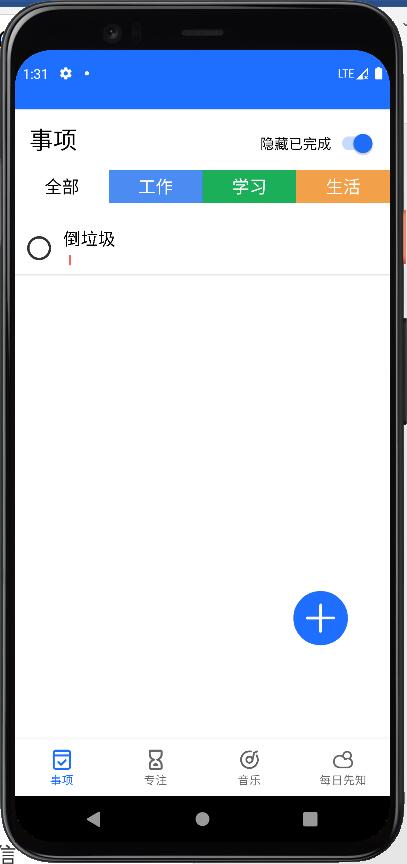
**分类栏的实现**

在顶部添加五个TextView作为分类查看事项菜单，点击某一分类即可查看该分类下的所有事项，并修改被点击TextView 的样式。



**隐藏已完成的实现**

task.xml布局右上角加入一个switch控件用以隐藏已完成事项。更改switch选中状态时根据要求在读取数据库数据时筛选即可

**添加事项对话框的实现**

这里使用的是在整个RelativeLayout布局中添加一个ImageView作为添加事项的按钮，并定义点击事件，点击时弹出对话框，在对话框中输入添加事项的信息；自定义对话框需要先设计一个layout布局文件add\_task\_dialog.xml；定义一个方法，实现弹出添加事项界面的对话框，并设置确认和取消按钮的点击事件，确认按钮即添加该事项到数据库并显示；然后在添加事项的点击事件中调用ShowAddTaskDialog()即可

**点击某条事项弹出对话框**，显示事项信息，可以修改，删除，标记失败。和添加事项的对话框实现原理相同

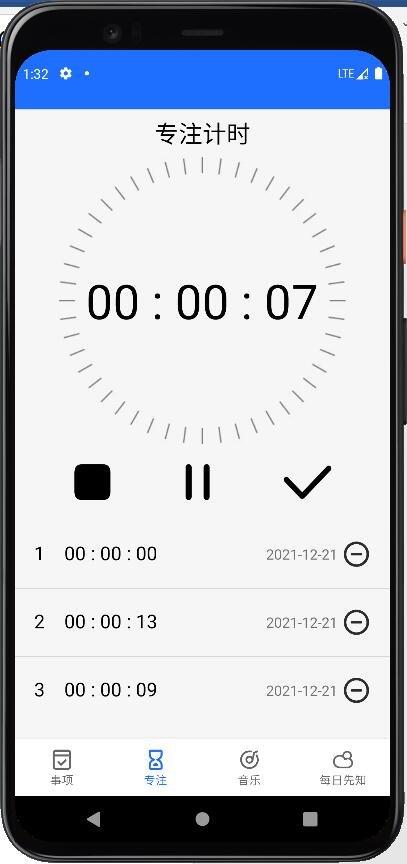
**四、专注计时界面**

**计时功能实现**

计时的原理是使用Android四大组件之一的Service开启计时线程，并每隔一秒钟发送一次本地广播通知主界面更新布局。

创建服务类TimeService，继承自Service。这里在Service类里面定义了一个TimeThread自定义线程类，用以方便线程的挂起和恢复；创建本地广播，用以接收TimeThread发送的秒数，并更新布局界面；在TimeThread的run方法中每一秒发送一次本地广播。

**显示专注计时的记录**

使用SQLite数据库实现，和待办事项界面一样，添加完成专注计时的按钮，点击事件为添加计时信息的字符串到数据库

**五、音乐界面**

实现原理，使用Service组件和MediaPlayer。点击音乐列表的某条音乐时，在服务中开启MediaPlayer播放音乐，并每隔一秒种发送一次本地广播（内容为当前已播放的秒数），设置界面中的进度条。并给进度条设置拖动的事件，将对应的播放进度传给MediaPlayer跳转至对应的进度。

**获取本地音乐文件**

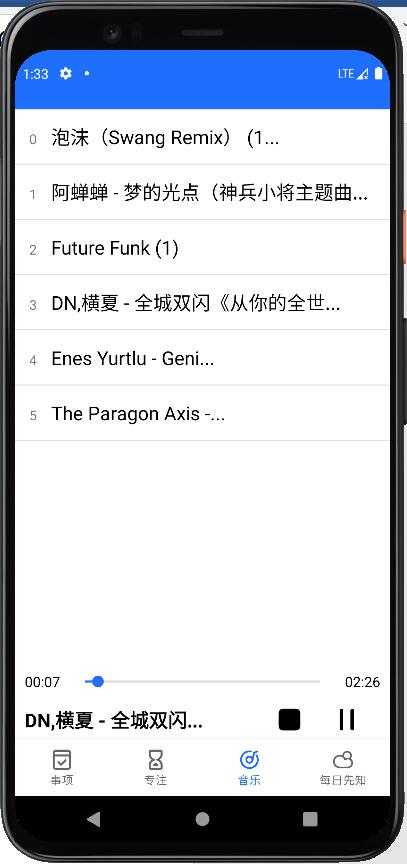
由于API 29以后getExternalStorageDirectory()方法被废弃，所以直接采用指定的路径获取MP3音乐文件。

**显示音乐播放进度**

在MyService中创建一个新的线程类，用于每隔一秒钟获取一次音乐的播放进度，原理和专注页面的计时线程相同；在MusicFragment中注册一个用于接收音乐播放进度和播放总时长的本地广播，在MusicTimeThread中每隔一秒发送一次播放进度和总时长；在MyService中写一个方法，用于发送当前播放进度和总时长的本地广播，并在run方法中每一秒钟发送一次。

**实现自动播放下一首**

将servicePlayMusic方法的参数改为音乐列表和第一首音乐的位置。MediaPlayer中有一个完成播放时的监听事件setOnCompletionListener，在该事件中调用传入的音乐列表的下一首就可以了，（注意对列表长度取余，否则会报超出范围的异常）。

另外，在每一首播放结束时，应该先暂停计时线程，在下一首播放时恢复计时线程。考虑到第一首播放时计时线程还未创建，应该做一个非空判断。

# 一些技术细节

**天气界面**：由于OkHttp的请求是在子线程中进行的，需要使用Handler消息队列机制将解析出来的Weather实例发送到主线程用以显示在界面上。

**待办事项界面**：建立ListView适配器。这里涉及到缓存convertView的使用，使用convertView可以防止每创建一个item时就解析一个布局，这样效率肯定不高。convertView是Android提供的用于缓存的View，在第一次渲染item时，将将解析出来的View放入缓存convertView，在下一次渲染item的时候，判断convertView是否为空即可。

**音乐播放界面**：MediaPlayer中有一个完成播放时的监听事件setOnCompletionListener，在该事件中调用传入的音乐列表的下一首就可以了，（注意对列表长度取余，否则会报超出范围的异常）。