Step 7: 添加城市列表类，重构城市列表处理的相关功能

目录

[目标 1](#_Toc448482725)

[步骤 1](#_Toc448482726)

[1、新建城市列表维护Activity 1](#_Toc448482727)

[2、在主界面上添加跳转到城市列表Activity的按钮 3](#_Toc448482728)

[3、修改CityListActivity的UI 4](#_Toc448482729)

[4、CityListActivity保存城市列表的方法 7](#_Toc448482730)

[5、CityListActivity添加城市方法 7](#_Toc448482731)

[6、CityListActivity删除城市方法 8](#_Toc448482732)

[7、在CityListActivity的Toolbar上添加按钮 9](#_Toc448482733)

## 目标

1）新建CityList城市列表类；

2）重构和城市列表相关的功能。

注意：本次修改不修改UI

## 步骤

### 1、新建CityList城市列表类

知识点：观察者设计模式。

功能：把城市列表集中、统一管理。

代码如下：

|  |
| --- |
| **package** cn.edu.uoh.cs.weatherforecast;  **import** android.app.Activity; **import** android.content.Context; **import** android.content.SharedPreferences;  **import** com.google.gson.Gson;  **import** java.util.ArrayList; **import** java.util.Collections; **import** java.util.List;  */\*\*  \* 城市列表，提供信息保存，获取功能。  \* Created by yan on 2016/4/15.  \*/* **public class** CityList {  */\*\*  \* 城市列表的修改监听器接口，当城市列表发生改变时，通知监听器。  \* 参考：观察者设计模式  \*/* **public interface** ChangeListener {  **void** onChange();  }   */\*\*  \* 城市列表存储key  \*/* **public final static** String ***StoreListKey*** = **"cn.edu.uoh.cs.weatherforecast.citynameList"**;   **private** ArrayList<String> **cityList** = **new** ArrayList<>();  **private** ChangeListener **listener**;   **private** CityList() {  *// 阻止New* }   */\*\*  \* 从保存的数据中获取城市列表  \** ***@param activity*** *activity  \** ***@return*** *CityList  \*/* **public static** CityList load(Activity activity) {  SharedPreferences sharedPref = activity.getPreferences(Context.***MODE\_PRIVATE***);  String cityNames = sharedPref.getString(***StoreListKey***, **"[蒙自]"**);  *// 转换字符串到数组* Gson gson = **new** Gson();  String[] cityArray = gson.fromJson(cityNames, String[].**class**);  *// 建立CityList对象并赋值* CityList list = **new** CityList();  Collections.*addAll*(list.**cityList**, cityArray);  **return** list;  }   */\*\*  \* 保存城市列表  \** ***@param activity*** *\*/* **public void** save(Activity activity) {  *// 转换成数组* String[] cityArray = **cityList**.toArray(**new** String[**cityList**.size()]);  *// 转换成Json字符串* Gson gson = **new** Gson();  String cityJson = gson.toJson(cityArray);  *// 保存* SharedPreferences sharedPref =activity.getPreferences(Context.***MODE\_PRIVATE***);  SharedPreferences.Editor editor = sharedPref.edit();  editor.putString(MainActivity.***StoreListKey***, cityJson);  editor.apply();  }   */\*\*  \* 城市列表  \** ***@return*** *\*/* List<String> getCityList() {  **return cityList**;  }   */\*\*  \* 列表中的城市数量  \** ***@return*** *\*/* **public int** size() {  **return cityList**.size();  }   */\*\*  \* 注册列表改变监听器。  \** ***@param listener*** *监听器  \*/* **public void** setChangeListener(ChangeListener listener) {  **this**.**listener** = listener;  }   */\*\*  \* 当城市列表改变时，调用此方法，在这个方法中通知已注册监听器  \*/* **private void** onChange() {  **if** (**listener** != **null**) {  **this**.**listener**.onChange();  }  }   */\*\*  \* 添加城市  \** ***@param city*** *城市名称  \*/* **public void** add(String city) {  **cityList**.add(city);  onChange();  }   */\*\*  \* 删除指定位置的城市  \** ***@param position*** *指定城市位置  \*/* **public void** remove(**int** position) {  **cityList**.remove(position);  onChange();  } } |

### 2、重构CityListActivity

重构CityListActivity，把直接对adapter的修改调用，都转移到cityList上。

代码如下，加粗的表示修改过的代码

|  |
| --- |
| **public class** CityListActivity **extends** AppCompatActivity {  **private CityList cityList;**   @Override  **protected void** onCreate(Bundle savedInstanceState) {  **super**.onCreate(savedInstanceState);  setContentView(R.layout.***activity\_city\_list***);  Toolbar toolbar = (Toolbar) findViewById(R.id.***toolbar***);  setSupportActionBar(toolbar);  getSupportActionBar().setDisplayHomeAsUpEnabled(**true**);  InitListView();  }   */\*\*  \* 初始化ListView，显示已保存的城市  \*/* **private void** InitListView() {  ***// 获取保存的城市列表* cityList = CityList.*load*(this);  *// 建立ListView的数据adapter* final ArrayAdapter<String> adapter = new ArrayAdapter<>(this, android.R.layout.*simple\_list\_item\_1*, cityList.getCityList());  *// 注册城市列表改变事件, 当城市列表改变时，通知adapter，* cityList.setChangeListener(new CityList.ChangeListener() {  @Override  public void onChange() {  adapter.notifyDataSetChanged();  }  });**  *// 设置ListView* ListView listView = (ListView) findViewById(R.id.***listView***);  listView.setAdapter(adapter);  }   @Override  **public boolean** onCreateOptionsMenu(Menu menu) {  getMenuInflater().inflate(R.menu.***menu\_city\_list***, menu);  **return true**;  }   @Override  **public boolean** onOptionsItemSelected(MenuItem item) {  **int** id = item.getItemId();  *// 单击删除按钮时删除选中的城市* **if** (id == R.id.***remove\_city***) {  removeCity();  **return true**;  }  *// 单击添加按钮是显示添加UI* **if** (id == R.id.***add\_city***) {  View view = findViewById(R.id.***addLayout***);  view.setVisibility(View.***VISIBLE***);  **return true**;  }  **return super**.onOptionsItemSelected(item);  }   */\*\*  \* 删除选中的城市  \*/* **private void** removeCity() {  ListView listView = (ListView) findViewById(R.id.***listView***);  **int** position = listView.getCheckedItemPosition();  **if** (position < 0 || position >= **cityList**.size()) {  **return**;  }  **cityList.remove(position);  cityList.save(this);**  }   */\*\*  \* 添加城市  \** ***@param view*** *view  \*/* **public void** addCity(View view) {  *// 取得输入框中的城市名称* EditText edtCity = (EditText) findViewById(R.id.***edtCity***);  String cityName = edtCity.getText().toString();  *// 如果输入不是空，添加到列表* **if** (cityName.trim().length() > 0) {  **cityList.add(cityName);  cityList.save(this);**  }  *// 清空输入* edtCity.setText(**""**);  *// 隐藏输入框* View addLayout = findViewById(R.id.***addLayout***);  addLayout.setVisibility(View.***GONE***);  }  } |