



手动更新天气和切换城市

虽然酷欧天气主体功能完成了，但是还存在一个比较严重的 bug，当选中某个城市后，就没法再去查看其它城市的天气了，即使退出程序下次进来时还是直接跳转到 WeatherActivity。

1. 手动更新天气

前面我们实现了对天气信息进行缓存，每次展示的是缓存中的数据，因此需要能够让用户手动更新天气信息。采用下拉刷新的方式来实现手动更新天气信息。

修改 activity_weather.xml 中的代码，如下所示：

```
<FrameLayout
xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:background="@color/colorPrimary">
.....
<android.support.v4.widget.DrawerLayout
    android:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <android.support.v4.widget.SwipeRefreshLayout
        android:id="@+id/swipe_refresh"
        android:layout_width="match_parent"
        android:layout_height="match_parent">
.....
```

在 ScrollView 的外层嵌套一层 SwipeRefreshLayout，这样 ScrollView 就可以自动拥有下拉刷新功能。

然后修改 WeatherActivity 中的代码，加入更新天气逻辑，如下所示：

```
public class WeatherActivity extends AppCompatActivity {
.....
```



```
public SwipeRefreshLayout swipeRefresh;
.....

@Override
protected void onCreate(Bundle savedInstanceState) {
    swipeRefresh = (SwipeRefreshLayout)
    findViewById(R.id. swipe_refresh);
    swipeRefresh.setColorSchemeResources(R.color. colorPrimary);
    drawerLayout = (DrawerLayout) findViewById(R.id. drawer_layout);
    navButton = (Button) findViewById(R.id. nav_button);
    SharedPreferences prefs =
    PreferenceManager.getDefaultSharedPreferences(this);
    String weatherString = prefs.getString("weather", null);
    if (weatherString != null) {
        // 有缓存时直接解析天气数据
        Weather weather = Utility.handleWeatherResponse(weatherString);
        mWeatherId = weather.basic.weatherId;
        showWeatherInfo(weather);
    } else {
        // 无缓存时去服务器查询天气
        mWeatherId = getIntent().getStringExtra("weather_id");
        weatherLayout.setVisibility(View. INVISIBLE);
        requestWeather(mWeatherId);
    }
    swipeRefresh.setOnRefreshListener(new
    SwipeRefreshLayout.OnRefreshListener() {
        @Override
        public void onRefresh() {
            requestWeather(mWeatherId);
        }
    }
}
```



```
});  
}  
/**  
 * 根据天气 id 请求城市天气信息。  
 */  
public void requestWeather(final String weatherId) {  
    String weatherUrl = "http://guolin.tech/api/weather?cityid=" +  
weatherId + "&key=bc0418b57b2d4918819d3974ac1285d9";  
    HttpUtil.sendOkHttpRequest(weatherUrl, new Callback() {  
        @Override  
        public void onResponse(Call call, Response response) throws  
IOException {  
            final String responseText = response.body().string();  
            final Weather weather =  
Utility.handleWeatherResponse(responseText);  
            runOnUiThread(new Runnable() {  
                @Override  
                public void run() {  
                    if (weather != null && "ok".equals(weather.status))  
{  
                        SharedPreferences.Editor editor =  
PreferenceManager.getDefaultSharedPreferences(WeatherActivity.this).e  
dit();  
                        editor.putString("weather", responseText);  
                        editor.apply();  
                        mWeatherId = weather.basic.weatherId;  
                        showWeatherInfo(weather);  
                    } else {  
                        Toast.makeText(WeatherActivity.this, "获取天
```



```
气信息失败", Toast.LENGTH_SHORT).show();  
    }  
    swipeRefresh.setRefreshing(false);  
    }  
});  
}
```

在 `onCreate()` 方法中获取到 `SwipeRefreshLayout` 的实例，然后调用 `setColorSchemeResources()` 方法来设置下拉刷新进度条的颜色，这里使用主题中的 `colorPrimary` 作为进度条的颜色。接着定义一个 `weatherId` 变量，用于记录城市的天气 id，然后调用 `setOnRefreshListener()` 方法来设置一个下拉刷新的监听器，当触发了下拉刷新操作的时候，就会回调这个监听器的 `onRefresh()` 方法，在这里调用 `requestWeather()` 方法请求天气信息就可以了。

当请求结束后，需要调用 `SwipeRefreshLayout` 的 `setRefreshing()` 方法并传入 `false`，用于表示刷新事件结束，并隐藏刷新进度条。

2. 切换城市

将碎片放入滑动菜单，正常情况下它不占据主界面的任何空间，想要切换城市的时候只需要通过滑动的方式将菜单显示出来就可以了。

按照 `Material Design` 的建议，在头布局中加入一个切换城市的按钮，不然的话用户可能根本不知道屏幕左侧边缘是可以拖动的。修改 `title.xml` 中的代码，如下所示：

```
<RelativeLayout  
    xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout_width="match_parent"  
    android:layout_height="?attr/actionBarSize">  
    <Button  
        android:id="@+id/nav_button"  
        android:layout_width="30dp"  
        android:layout_height="30dp"
```



```
android:layout_marginLeft="10dp"  
android:layout_alignParentLeft="true"  
android:layout_centerVertical="true"  
android:background="@drawable/ic_home" />
```

.....

```
</RelativeLayout>
```

添加了一个 Button 作为切换城市的按钮，并且让它居左显示，提供一张图片作为按钮的背景图。

修改 activity_weather.xml 布局，加入滑动菜单功能，如下所示：

```
<FrameLayout  
xmlns:android="http://schemas.android.com/apk/res/android"  
android:layout_width="match_parent"  
android:layout_height="match_parent"  
android:background="@color/colorPrimary">
```

.....

```
<android.support.v4.widget.DrawerLayout  
    android:id="@+id/drawer_layout"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent">
```

.....

```
<fragment  
    android:id="@+id/choose_area_fragment"  
    android:name="com.coolweather.android.ChooseAreaFragment"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:layout_gravity="start"  
    />
```

```
</android.support.v4.widget.DrawerLayout>
```

```
</FrameLayout>
```



在 `SwipeRefreshLayout` 的外面嵌套了一层 `DrawerLayout`。`DrawerLayout` 中的第一个子控件用于作为主屏幕中显示的内容，第二个子控件用于作为滑动菜单中显示的内容，因此这里我们在第二个子控件的位置添加了用于遍历省市县数据的碎片。

下面修改 `WeatherActivity` 中加入滑动菜单的逻辑处理，修改代码如下：

```
public class WeatherActivity extends AppCompatActivity {
    public DrawerLayout drawerLayout;
    private Button navButton;
    ...
    drawerLayout = (DrawerLayout) findViewById(R.id.drawer_layout);
    navButton = (Button) findViewById(R.id.nav_button);
    .....
    navButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            drawerLayout.openDrawer(GravityCompat.START);
        }
    });
    .....
}
```

在 `onCreate()` 方法中获取新增的 `DrawerLayout` 和 `Button` 的实例，然后在 `Button` 的点击事件中调用 `DrawerLayout` 的 `openDrawer()` 方法来打开滑动菜单。

在 `ChooseAreaFragment` 中完成处理切换城市的逻辑。之前选中了某个城市后跳转到 `WeatherActivity`，现在就是在 `WeatherActivity` 当中，因此不需要跳转，只是去请求新选择城市的天气信息就可以了。

需要根据 `ChooseAreaFragment` 的不同状态来进行不同的逻辑处理，修改代码如下：

```
public class ChooseAreaFragment extends Fragment {
```



```
...  
  
@Override  
public void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    listView.setOnItemClickListener(new  
AdapterView.OnItemClickListener() {  
        @Override  
        public void onItemClick(AdapterView<?> parent, View view, int  
position, long id) {  
            if (currentLevel == LEVEL_PROVINCE) {  
                selectedProvince = provinceList.get(position);  
                queryCities();  
            } else if (currentLevel == LEVEL_CITY) {  
                selectedCity = cityList.get(position);  
                queryCounties();  
            } else if (currentLevel == LEVEL_COUNTY) {  
                String          weatherId          =  
countyList.get(position).getWeatherId();  
                if (getActivity() instanceof MainActivity) {  
                    Intent intent = new Intent(getActivity(),  
WeatherActivity.class);  
                    intent.putExtra("weather_id", weatherId);  
                    startActivity(intent);  
                    getActivity().finish();  
                } else if (getActivity() instanceof WeatherActivity)  
{  
                    WeatherActivity activity = (WeatherActivity)  
getActivity();  
                    activity.drawerLayout.closeDrawers();  
                }  
            }  
        }  
    }  
}
```



```
        activity.swipeRefresh.setRefreshing(true);
        activity.requestWeather(weatherId);
    }
}
});
backButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (currentLevel == LEVEL_COUNTY) {
            queryCities();
        } else if (currentLevel == LEVEL_CITY) {
            queryProvinces();
        }
    }
});
queryProvinces();
}
.....
}
```

在碎片中调用 `getActivity()` 方法，配合 `instanceof` 关键字，可以判断出该碎片是在 `MainActivity` 中，还是在 `WeatherActivity` 中，如果在 `MainActivity` 中，处理逻辑不变；如果在 `WeatherActivity` 中，就关闭滑动菜单，显示下拉刷新进度条，然后请求新城市的天气信息。

可以看到标题栏多了一个用于切换城市的按钮，点击该按钮或者在屏幕左侧边缘进行拖动，就能让滑动菜单界面显示出来。