Step 9: 左右滑动切换城市

目录

[目标 1](#_Toc448520070)

[步骤 1](#_Toc448520071)

[1、重构NetTask 1](#_Toc448520072)

[2、新建手势处理类 3](#_Toc448520073)

[3、在MainActivity中加入手势识别 4](#_Toc448520074)

[4、实现城市切换（CityList） 5](#_Toc448520075)

[5、实现城市切换（MainActivity） 6](#_Toc448520076)

[6、在滑动时切换城市 7](#_Toc448520077)

## 目标

使用左右滑动切换各个城市的天气。

## 步骤

### 1、重构NetTask

1）重构网络异步任务类NetTask，把它从MainActivity中移除，放在单独的文件中。

|  |
| --- |
| */\*\*  \* 定义异步任务，用于通过网络获取数据（Android不能在主线程中访问网络）  \* 重构后定义了一个独立的类，原来是MainActivity的内部类  \* Created by yan on 2016/4/15.  \*/* **class** NetTask **extends** AsyncTask<String, Void, String> {  MainActivity **activity**;  IOException **exception** = **null**;   NetTask(MainActivity activity) {  **this**.**activity** = activity;  }   @Override  **protected** String doInBackground(String... params) {  **try** {  *// 获取调用execute方法时传入的参数：城市名* String cityName = params[0];  Log.*i*(MainActivity.***tag***, **"Get weather： "** + cityName);  *// 获取天气信息，得到的是Json字符串* String json = HttpTools.*getWeatherInfo*(cityName);  Log.*i*(MainActivity.***tag***, json);  **return** json;  } **catch** (IOException e) {  *// 如果有异常，记录* **exception** = e;  **return null**;  }  }   @Override  **protected void** onPostExecute(String result) {  **if** (**exception** == **null**) {  *// 显示获取的信息  // TextView txtInfo = (TextView) findViewById(R.id.txtInfo);  // txtInfo.setText(result);* **activity**.showForecastList(result);  } **else** {  *// 如果有异常，显示错误消息* **activity**.showError(**"获取天气信息失败："** + **exception**.getMessage());  }  } } |

2）修改MainActivity中使用NetTask的方法

|  |
| --- |
| **private void** refreshWeatherInfo() {  *// 城市名称* String cityName = **cityList**.getCurrentCityName();  Log.*i*(***tag***, **"refresh weather: "** + cityName);  *// 修改工具栏标题* **if** (cityName == **null**) {  Toolbar toolbar = (Toolbar) findViewById(R.id.***toolbar***);  toolbar.setTitle(**"请添加城市"**);  **return**;  }  *// 调用异步任务类，访问网络，获取数据* **NetTask task = new NetTask(this);**  task.execute(cityName); } |

### 2、新建手势处理类

代码如下，目前各个滑动的处理内容空

|  |
| --- |
| */\*\*  \* 手势识别，处理类  \* Created by yan on 2016/4/15.  \*/* **class** GestureDetectorListener **implements** GestureDetector.OnGestureListener {  MainActivity **activity**;   GestureDetectorListener(MainActivity activity) {  **this**.**activity** = activity;  }   @Override  **public boolean** onDown(MotionEvent e) {  **return false**;  }   @Override  **public void** onShowPress(MotionEvent e) {  }   @Override  **public boolean** onSingleTapUp(MotionEvent e) {  **return false**;  }   @Override  **public boolean** onScroll(MotionEvent e1, MotionEvent e2, **float** distanceX, **float** distanceY) {  **return false**;  }   @Override  **public void** onLongPress(MotionEvent e) {  }   @Override  **public boolean** onFling(MotionEvent e1, MotionEvent e2, **float** velocityX, **float** velocityY) {  **float** minMove = 120; *//最小滑动距离* **float** minVelocity = 0; *//最小滑动速度* **float** beginX = e1.getX();  **float** endX = e2.getX();  **float** beginY = e1.getY();  **float** endY = e2.getY();   **if** (beginX - endX > minMove && Math.*abs*(velocityX) > minVelocity) {  *//左滑* Toast.*makeText*(**activity**, velocityX + **"左滑"**, Toast.***LENGTH\_SHORT***).show();  } **else if** (endX - beginX > minMove && Math.*abs*(velocityX) > minVelocity) {  *//右滑* Toast.*makeText*(**activity**, velocityX + **"右滑"**, Toast.***LENGTH\_SHORT***).show();  } **else if** (beginY - endY > minMove && Math.*abs*(velocityY) > minVelocity) {  *//上滑* Toast.*makeText*(**activity**, velocityX + **"上滑"**, Toast.***LENGTH\_SHORT***).show();  } **else if** (endY - beginY > minMove && Math.*abs*(velocityY) > minVelocity) {  *//下滑* Toast.*makeText*(**activity**, velocityX + **"下滑"**, Toast.***LENGTH\_SHORT***).show();  }  **return false**;  } } |

### 3、在MainActivity中加入手势识别

修改MainActivity:

1）加实例变量GestureDetector detector;

2）修改onCreate()方法

3）添加方法dispatchTouchEvent

|  |
| --- |
| **public class** MainActivity **extends** AppCompatActivity {  *//定义手势检测器实例* GestureDetector **detector**;   @Override  **protected void** onCreate(Bundle savedInstanceState) {  **super**.onCreate(savedInstanceState);  setContentView(R.layout.***activity\_main***);  Toolbar toolbar = (Toolbar) findViewById(R.id.***toolbar***);  setSupportActionBar(toolbar);  *// 恢复保存的城市名称* reloadCityList();  *// 刷新天气* refreshWeatherInfo();  ***//创建手势检测器* detector = new GestureDetector(this, new GestureDetectorListener(this));**  }   ***/\*\*  \* 在事件dispatch时处理触摸手势。  \* @param ev event  \* @return super.dispatchTouchEvent(ev)  \*/* @Override  public boolean dispatchTouchEvent(MotionEvent ev) {  this.detector.onTouchEvent(ev);  return super.dispatchTouchEvent(ev);  }**  } |

这个步骤完成后，在屏幕滑动，可以看到是什么方向的滑动提示。

### 4、实现城市切换（CityList）

添加方法previous，next，移动当前城市到前一个、后一个

|  |
| --- |
| */\*\*  \* 在城市列表中，移动到前一个城市  \** ***@return*** *true 如果有前一个城市  \*/* **public boolean** previous() {  **if** (**currentCityIndex** > 0) {  **currentCityIndex**--;  **currentCityName** = **cityList**.get(**currentCityIndex**);  **return true**;  }  **return false**; }  */\*\*  \* 在城市列表中，移动到后一个城市  \** ***@return*** *true 如果有后一个城市  \*/* **public boolean** next() {  **if** (**currentCityIndex** < **cityList**.size()-1) {  **currentCityIndex**++;  **currentCityName** = **cityList**.get(**currentCityIndex**);  **return true**;  }  **return false**; } |

### 5、实现城市切换（MainActivity）

实现城市切换代码，可以切换到前一个城市（向右滑动），后一个城市（向左滑动）。

添加方法previous，next，移动当前城市到前一个、后一个，获取数据并刷新UI

|  |
| --- |
| */\*\*  \* 切换到前一个城市，如果成功，刷新天气  \*/* **public void** previous() {  **if** (**cityList**.previous()) {  *// 如果有前一个城市，刷新天气情况* refreshWeatherInfo();  } }  */\*\*  \* 切换到后一个城市，如果成功，刷新天气  \*/* **public void** next() {  **if** (**cityList**.next()) {  *// 如果有后一个城市，刷新天气情况* refreshWeatherInfo();  } } |

### 6、在滑动时切换城市

修改GestureDetectorListener类的onFling方法，在检测到滑动手势时调用MainAcivity中的相应方法

|  |
| --- |
| **public boolean** onFling(MotionEvent e1, MotionEvent e2, **float** velocityX, **float** velocityY) {  **float** minMove = 120; *//最小滑动距离* **float** minVelocity = 0; *//最小滑动速度* **float** beginX = e1.getX();  **float** endX = e2.getX();  **float** beginY = e1.getY();  **float** endY = e2.getY();   **if** (beginX - endX > minMove && Math.*abs*(velocityX) > minVelocity) {  *//左滑* **activity**.next();  } **else if** (endX - beginX > minMove && Math.*abs*(velocityX) > minVelocity) {  *//右滑* **activity**.previous();  } **else if** (beginY - endY > minMove && Math.*abs*(velocityY) > minVelocity) {  *//上滑* } **else if** (endY - beginY > minMove && Math.*abs*(velocityY) > minVelocity) {  *//下滑* }  **return false**; } |