

横向滑动列表——未来天气

效果图



步骤

- 1.自定义横向List View：新建HorizontalListView.java
- 2.修改weather_info.xml
- 3.新建未来天气类——FutureWeather.java
- 4.新建适配器——HorizontalAdapter.java
- 5.修改parseXML函数与TodayWeather类
- 6.修改MainActivity.java

一·自定义横向ListView：新建HorizontalListView.java

代码：

```
1 import android.content.Context;
2 import android.database.DataSetObserver;
3 import android.graphics.Rect;
4 import android.util.AttributeSet;
```

```

5 import android.view.GestureDetector;
6 import android.view.GestureDetector.OnGestureListener;
7 import android.view.MotionEvent;
8 import android.view.View;
9 import android.widget.AdapterView;
10 import android.widget.ListAdapter;
11 import android.widget.Scroller;
12
13 import java.util.LinkedList;
14 import java.util.Queue;
15
16 public class HorizontalListView extends AdapterView<ListAdapter> {
17
18     public boolean mAlwaysOverrideTouch = true;
19     protected ListAdapter mAdapter;
20     private int mLeftViewIndex = -1;
21     private int mRightViewIndex = 0;
22     protected int mCurrentX;
23     protected int mNextX;
24     private int mMaxX = Integer.MAX_VALUE;
25     private int mDisplayOffset = 0;
26     protected Scroller mScroller;
27     private GestureDetector mGesture;
28     private Queue<View> mRemovedViewQueue = new LinkedList<View>();
29     private OnItemSelectedListener mOnItemSelected;
30     private OnItemClickListener mOnItemClicked;
31     private OnItemLongClickListener mOnItemLongClicked;
32     private boolean mDataChanged = false;
33
34
35     public HorizontalListView(Context context, AttributeSet attrs) {
36         super(context, attrs);
37         initView();
38     }
39
40     private synchronized void initView() {
41         mLeftViewIndex = -1;
42         mRightViewIndex = 0;
43         mDisplayOffset = 0;
44         mCurrentX = 0;
45         mNextX = 0;
46         mMaxX = Integer.MAX_VALUE;
47         mScroller = new Scroller(getContext());
48         mGesture = new GestureDetector(getContext(), mOnGesture);
49     }
50
51     @Override
52     public void setOnItemSelectedListener(OnItemSelectedListener listener)
53     {
54         mOnItemSelected = listener;
55     }

```

```

55
56     @Override
57     public void setOnItemClickListener(OnItemClickListener listener){
58         mOnItemClicked = listener;
59     }
60
61     @Override
62     public void setOnItemLongClickListener(OnItemLongClickListener
63 listener) {
64         mOnItemLongClicked = listener;
65     }
66
67     private DataSetObserver mDataObserver = new DataSetObserver() {
68
69         @Override
70         public void onChanged() {
71             synchronized(HorizontalListView.this){
72                 mDataChanged = true;
73             }
74             invalidate();
75             requestLayout();
76         }
77
78         @Override
79         public void onInvalidated() {
80             reset();
81             invalidate();
82             requestLayout();
83         }
84     };
85
86     @Override
87     public ListAdapter getAdapter() {
88         return mAdapter;
89     }
90
91     @Override
92     public View getSelectedView() {
93         //TODO: implement
94         return null;
95     }
96
97     @Override
98     public void setAdapter(ListAdapter adapter) {
99         if(mAdapter != null) {
100             mAdapter.unregisterDataSetObserver(mDataObserver);
101         }
102         mAdapter = adapter;
103         mAdapter.registerDataSetObserver(mDataObserver);
104         reset();

```

```

105     }
106
107     private synchronized void reset(){
108         initView();
109         removeAllViewsInLayout();
110         requestLayout();
111     }
112
113     @Override
114     public void setSelection(int position) {
115         //TODO: implement
116     }
117
118     private void addAndMeasureChild(final View child, int viewPos) {
119         LayoutParams params = child.getLayoutParams();
120         if(params == null) {
121             params = new LayoutParams(LayoutParams.FILL_PARENT,
122                                     LayoutParams.FILL_PARENT);
123         }
124
125         addViewInLayout(child, viewPos, params, true);
126         child.measure(MeasureSpec.makeMeasureSpec(getWidth(),
127                                                     MeasureSpec.AT_MOST),
128                     MeasureSpec.makeMeasureSpec(getHeight(),
129                                                     MeasureSpec.AT_MOST));
129     }
130
131     @Override
132     protected synchronized void onLayout(boolean changed, int left, int
133     top, int right, int bottom) {
134         super.onLayout(changed, left, top, right, bottom);
135
136         if(mAdapter == null){
137             return;
138         }
139
140         if(mDataChanged){
141             int oldCurrentX = mCurrentX;
142             initView();
143             removeAllViewsInLayout();
144             mNextX = oldCurrentX;
145             mDataChanged = false;
146         }
147
148         if(mScroller.computeScrollOffset()){
149             int scrollx = mScroller.getCurrX();
150             mNextX = scrollx;
151         }

```

```

152         if(mNextX <= 0){
153             mNextX = 0;
154             mScroller.forceFinished(true);
155         }
156         if(mNextX >= mMaxX) {
157             mNextX = mMaxX;
158             mScroller.forceFinished(true);
159         }
160
161         int dx = mCurrentX - mNextX;
162
163         removeNonVisibleItems(dx);
164         fillList(dx);
165         positionItems(dx);
166
167         mCurrentX = mNextX;
168
169         if(!mScroller.isFinished()){
170             post(new Runnable(){
171                 @Override
172                 public void run() {
173                     requestLayout();
174                 }
175             });
176
177         }
178     }
179
180     private void fillList(final int dx) {
181         int edge = 0;
182         View child = getChildAt(getChildCount()-1);
183         if(child != null) {
184             edge = child.getRight();
185         }
186         fillListRight(edge, dx);
187
188         edge = 0;
189         child = getChildAt(0);
190         if(child != null) {
191             edge = child.getLeft();
192         }
193         fillListLeft(edge, dx);
194
195
196     }
197
198     private void fillListRight(int rightEdge, final int dx) {
199         while(rightEdge + dx < getWidth() && mRightViewIndex <
200         mAdapter.getCount()) {

```

```

201         View child = mAdapter.getView(mRightViewIndex,
mRemovedViewQueue.poll(), this);
202         addAndMeasureChild(child, -1);
203         rightEdge += child.getMeasuredWidth();
204
205         if(mRightViewIndex == mAdapter.getCount()-1) {
206             mMaxX = mCurrentX + rightEdge - getWidth();
207         }
208
209         if (mMaxX < 0) {
210             mMaxX = 0;
211         }
212         mRightViewIndex++;
213     }
214
215 }
216
217 private void fillListLeft(int leftEdge, final int dx) {
218     while(leftEdge + dx > 0 && mLeftViewIndex >= 0) {
219         View child = mAdapter.getView(mLeftViewIndex,
mRemovedViewQueue.poll(), this);
220         addAndMeasureChild(child, 0);
221         leftEdge -= child.getMeasuredWidth();
222         mLeftViewIndex--;
223         mDisplayOffset -= child.getMeasuredWidth();
224     }
225 }
226
227 private void removeNonVisibleItems(final int dx) {
228     View child = getChildAt(0);
229     while(child != null && child.getRight() + dx <= 0) {
230         mDisplayOffset += child.getMeasuredWidth();
231         mRemovedViewQueue.offer(child);
232         removeViewInLayout(child);
233         mLeftViewIndex++;
234         child = getChildAt(0);
235     }
236
237
238     child = getChildAt(getChildCount()-1);
239     while(child != null && child.getLeft() + dx >= getWidth()) {
240         mRemovedViewQueue.offer(child);
241         removeViewInLayout(child);
242         mRightViewIndex--;
243         child = getChildAt(getChildCount()-1);
244     }
245 }
246
247 private void positionItems(final int dx) {
248     if(getChildCount() > 0){
249         mDisplayOffset += dx;

```

```

250         int left = mDisplayOffset;
251         for(int i=0;i<getChildCount();i++){
252             View child = getChildAt(i);
253             int childWidth = child.getMeasuredWidth();
254             child.layout(left, 0, left + childWidth,
child.getMeasuredHeight());
255             left += childWidth + child.getPaddingRight();
256         }
257     }
258 }
259
260 public synchronized void scrollTo(int x) {
261     mScroller.startScroll(mNextX, 0, x - mNextX, 0);
262     requestLayout();
263 }
264
265 @Override
266 public boolean dispatchTouchEvent(MotionEvent ev) {
267     boolean handled = super.dispatchTouchEvent(ev);
268     handled |= mGesture.onTouchEvent(ev);
269     return handled;
270 }
271
272 protected boolean onFling(MotionEvent e1, MotionEvent e2, float
velocityX,
273                             float velocityY) {
274     synchronized(HorizontalListView.this){
275         mScroller.fling(mNextX, 0, (int)-velocityX, 0, 0, mMaxX, 0,
0);
276     }
277     requestLayout();
278
279     return true;
280 }
281
282 protected boolean onDown(MotionEvent e) {
283     mScroller.forceFinished(true);
284     return true;
285 }
286
287 private OnGestureListener mOnGesture = new
GestureDetector.SimpleOnGestureListener() {
288
289     @Override
290     public boolean onDown(MotionEvent e) {
291         return HorizontalListView.this.onDown(e);
292     }
293
294     @Override
295     public boolean onFling(MotionEvent e1, MotionEvent e2, float
velocityX,

```

```

296         float velocityY) {
297             return HorizontalListView.this.onFling(e1, e2, velocityX,
velocityY);
298         }
299
300         @Override
301         public boolean onScroll(MotionEvent e1, MotionEvent e2,
float distanceX, float distanceY) {
302
303             synchronized(HorizontalListView.this){
304                 mNextX += (int)distanceX;
305             }
306             requestLayout();
307
308             return true;
309         }
310
311         @Override
312         public boolean onSingleTapConfirmed(MotionEvent e) {
313             for(int i=0;i<getChildCount();i++){
314                 View child = getChildAt(i);
315                 if (isEventWithinView(e, child)) {
316                     if(mOnItemClicked != null){
317                         mOnItemClicked.onItemClick(HorizontalListView.this, child, mLeftViewIndex
+ 1 + i, mAdapter.getItemId( mLeftViewIndex + 1 + i ));
318                     }
319                     if(mOnItemSelected != null){
320                         mOnItemSelected.onItemSelected(HorizontalListView.this, child,
mLeftViewIndex + 1 + i, mAdapter.getItemId( mLeftViewIndex + 1 + i ));
321                     }
322                     break;
323                 }
324             }
325
326             return true;
327         }
328
329         @Override
330         public void onLongPress(MotionEvent e) {
331             int childCount = getChildCount();
332             for (int i = 0; i < childCount; i++) {
333                 View child = getChildAt(i);
334                 if (isEventWithinView(e, child)) {
335                     if (mOnItemLongClicked != null) {
336                         mOnItemLongClicked.onItemLongClick(HorizontalListView.this, child,
mLeftViewIndex + 1 + i, mAdapter.getItemId(mLeftViewIndex + 1 + i));
337                     }
338                     break;
339                 }
340             }
341         }
342     }
343 }

```

```

340         }
341
342     }
343 }
344
345     private boolean isEventWithinView(MotionEvent e, View child) {
346         Rect viewRect = new Rect();
347         int[] childPosition = new int[2];
348         child.getLocationOnScreen(childPosition);
349         int left = childPosition[0];
350         int right = left + child.getWidth();
351         int top = childPosition[1];
352         int bottom = top + child.getHeight();
353         viewRect.set(left, top, right, bottom);
354         return viewRect.contains((int) e.getRawX(), (int)
355             e.getRawY());
356     }
357 }

```

二·修改weather_info.xml

展示图：



```

<!-- 今日天气 -->
<RelativeLayout...>
<!-- 未来天气 -->
<RelativeLayout
    android:id="@+id/future_weather"
    android:layout_width="match_parent"
    android:layout_height="140dp"
    android:layout_marginTop="100dp">
    <com.example.mwessj.mlayout.HorizontalListView
        android:id="@+id/future_weather_list"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_vertical"
        android:background="@drawable/biankuang"
        android:orientation="horizontal" />
</RelativeLayout>

```

代码：

```

1  <RelativeLayout
2      android:layout_width="match_parent"
3      android:layout_height="140dp"
4      android:layout_marginTop="100dp">
5      <com.example.mwessj.mlayout.HorizontalListView
6          android:id="@+id/future_weather_list"
7          android:layout_width="match_parent"
8          android:layout_height="wrap_content"
9          android:layout_gravity="center_vertical"
10 </RelativeLayout>

```

三·新建未来天气类——FutureWeather.java

代码：

```
1  public class FutureWeather {
2      private String fengli;
3      private String date;
4      private String high;
5      private String low;
6      private String type;
7      private int imageID;
8
9      public FutureWeather(String date, String high, String low, String
type, String fengli, int imageID){
10         this.fengli = fengli;
11         this.date = date;
12         this.high = high;
13         this.low = low;
14         this.type = type;
15         this.imageID = imageID;
16     }
17
18     public String getFengli() {
19         return fengli;
20     }
21
22     public String getDate() {
23         return date;
24     }
25
26     public String getHigh() {
27         return high;
28     }
29
30     public String getLow() {
31         return low;
32     }
33
34     public String getType() {
35         return type;
36     }
37
38     public int getImageID() {
39         return imageID;
40     }
41 }
```

四·新建适配器——HorizontalAdapter.java

代码:

```
1  import android.content.Context;
2  import android.view.LayoutInflater;
3  import android.view.View;
4  import android.view.ViewGroup;
5  import android.widget.BaseAdapter;
6  import android.widget.ImageView;
7  import android.widget.TextView;
8
9  import com.example.mwessj.bean.FutureWeather;
10 import com.example.mwessj.mwessjweather.R;
11
12 import java.util.List;
13
14 public class HorizontalAdapter extends BaseAdapter {
15     private LayoutInflater mInflater;
16     private List<FutureWeather> mDatas;
17     public HorizontalAdapter(Context context, List<FutureWeather> datas){
18         mInflater = LayoutInflater.from(context);
19         mDatas = datas;
20     }
21
22     public void updateListView(List<FutureWeather> datas){
23         mDatas = datas;
24         notifyDataSetChanged();
25     }
26
27     @Override
28     public int getCount() {
29         return mDatas.size();
30     }
31
32     @Override
33     public Object getItem(int position) {
34         return position;
35     }
36
37     @Override
38     public long getItemId(int position) {
39         return position;
40     }
41
42     @Override
43     public View getView(int position, View convertView, ViewGroup parent)
44     {
45         ViewHolder holder = null;
46         if (convertView == null){
47             convertView = mInflater.inflate(R.layout.weather_today,
```

```

48         holder.weatherImgTv = (ImageView)
convertView.findViewById(R.id.weather_img);
49         holder.dataTv = (TextView)
convertView.findViewById(R.id.week_today);
50         holder.temperatureTv = (TextView)
convertView.findViewById(R.id.temperature);
51         holder.climateTv = (TextView)
convertView.findViewById(R.id.climate);
52         holder.windTv = (TextView)
convertView.findViewById(R.id.wind);
53         convertView.setTag(holder);
54     }else {
55         holder = (ViewHolder)convertView.getTag();
56     }
57
58     FutureWeather futureWeather = mDatas.get(position);
59     holder.weatherImgTv.setImageResource(futureWeather.getImageID());
60     holder.dataTv.setText(futureWeather.getDate());
61
62     holder.temperatureTv.setText(futureWeather.getHigh()+"~"+futureWeather.get
Low());
63     holder.climateTv.setText(futureWeather.getType());
64     holder.windTv.setText("风力: "+futureWeather.getFengli());
65     return convertView;
66
67     private class ViewHolder{
68         ImageView weatherImgTv;
69         TextView dataTv;
70         TextView temperatureTv;
71         TextView climateTv;
72         TextView windTv;
73     }
74 }

```

五.修改parseXML函数与TodayWeather类

首先，修改TodayWeather类，补充下列代码即可：

```

1  private String date1;
2  private String high1;
3  private String low1;
4  private String type1;
5  private String fengli1;
6
7  private String date2;
8  private String high2;
9  private String low2;
10 private String type2;
11 private String fengli2;

```

```
12
13 private String date3;
14 private String high3;
15 private String low3;
16 private String type3;
17 private String fengli3;
18
19 private String date4;
20 private String high4;
21 private String low4;
22 private String type4;
23 private String fengli4;
24
25 public String getDate1() {
26     return date1;
27 }
28
29 public String getDate2() {
30     return date2;
31 }
32
33 public String getDate3() {
34     return date3;
35 }
36
37 public String getDate4() {
38     return date4;
39 }
40
41 public String getHigh1() {
42     return high1;
43 }
44
45 public String getHigh2() {
46     return high2;
47 }
48
49 public String getHigh3() {
50     return high3;
51 }
52
53 public String getHigh4() {
54     return high4;
55 }
56
57 public String getLow1() {
58     return low1;
59 }
60
61 public String getLow2() {
62     return low2;
```

```
63 }
64
65 public String getLow3() {
66     return low3;
67 }
68
69 public String getLow4() {
70     return low4;
71 }
72
73 public String getType1() {
74     return type1;
75 }
76
77 public String getType2() {
78     return type2;
79 }
80
81 public String getType3() {
82     return type3;
83 }
84
85 public String getType4() {
86     return type4;
87 }
88
89 public String getFengli1() {
90     return fengli1;
91 }
92
93 public String getFengli2() {
94     return fengli2;
95 }
96
97 public String getFengli3() {
98     return fengli3;
99 }
100
101 public String getFengli4() {
102     return fengli4;
103 }
104
105 public void setDate1(String date1) {
106     this.date1 = date1;
107 }
108
109 public void setDate2(String date2) {
110     this.date2 = date2;
111 }
112
113 public void setDate3(String date3) {
```

```
114     this.date3 = date3;
115 }
116
117 public void setDate4(String date4) {
118     this.date4 = date4;
119 }
120
121 public void setHigh1(String high1) {
122     this.high1 = high1;
123 }
124
125 public void setHigh2(String high2) {
126     this.high2 = high2;
127 }
128
129 public void setHigh3(String high3) {
130     this.high3 = high3;
131 }
132
133 public void setHigh4(String high4) {
134     this.high4 = high4;
135 }
136
137 public void setLow1(String low1) {
138     this.low1 = low1;
139 }
140
141 public void setLow2(String low2) {
142     this.low2 = low2;
143 }
144
145 public void setLow3(String low3) {
146     this.low3 = low3;
147 }
148
149 public void setLow4(String low4) {
150     this.low4 = low4;
151 }
152
153 public void setType1(String type1) {
154     this.type1 = type1;
155 }
156
157 public void setType2(String type2) {
158     this.type2 = type2;
159 }
160
161 public void setType3(String type3) {
162     this.type3 = type3;
163 }
164
```

```

165 public void setType4(String type4) {
166     this.type4 = type4;
167 }
168
169 public void setFengli1(String fengli1) {
170     this.fengli1 = fengli1;
171 }
172
173 public void setFengli2(String fengli2) {
174     this.fengli2 = fengli2;
175 }
176
177 public void setFengli3(String fengli3) {
178     this.fengli3 = fengli3;
179 }
180
181 public void setFengli4(String fengli4) {
182     this.fengli4 = fengli4;
183 }

```

然后，修改parseXML函数，补充下列代码即可：

```

1  else if (xmlPullParser.getName().equals("fengli") && fengliCount == 1) {
2      eventType = xmlPullParser.next();
3      todayWeather.setFengli1(xmlPullParser.getText());
4      fengliCount++;
5  } else if (xmlPullParser.getName().equals("fengli") && fengliCount == 2) {
6      eventType = xmlPullParser.next();
7      todayWeather.setFengli2(xmlPullParser.getText());
8      fengliCount++;
9  } else if (xmlPullParser.getName().equals("fengli") && fengliCount == 3) {
10     eventType = xmlPullParser.next();
11     todayWeather.setFengli3(xmlPullParser.getText());
12     fengliCount++;
13 } else if (xmlPullParser.getName().equals("fengli") && fengliCount == 4) {
14     eventType = xmlPullParser.next();
15     todayWeather.setFengli4(xmlPullParser.getText());
16     fengliCount++;
17 } else if (xmlPullParser.getName().equals("date") && dateCount == 1) {
18     eventType = xmlPullParser.next();
19     todayWeather.setDate1(xmlPullParser.getText());
20     dateCount++;
21 } else if (xmlPullParser.getName().equals("date") && dateCount == 2) {
22     eventType = xmlPullParser.next();
23     todayWeather.setDate2(xmlPullParser.getText());
24     dateCount++;
25 } else if (xmlPullParser.getName().equals("date") && dateCount == 3) {
26     eventType = xmlPullParser.next();
27     todayWeather.setDate3(xmlPullParser.getText());
28     dateCount++;
29 } else if (xmlPullParser.getName().equals("date") && dateCount == 4) {
30     eventType = xmlPullParser.next();

```

```
31     todayWeather.setDate4(xmlPullParser.getText());
32     dateCount++;
33 } else if (xmlPullParser.getName().equals("high") && highCount == 1) {
34     eventType = xmlPullParser.next();
35     todayWeather.setHigh1(xmlPullParser.getText().substring(2).trim());
36     highCount++;
37 } else if (xmlPullParser.getName().equals("high") && highCount == 2) {
38     eventType = xmlPullParser.next();
39     todayWeather.setHigh2(xmlPullParser.getText().substring(2).trim());
40     highCount++;
41 } else if (xmlPullParser.getName().equals("high") && highCount == 3) {
42     eventType = xmlPullParser.next();
43     todayWeather.setHigh3(xmlPullParser.getText().substring(2).trim());
44     highCount++;
45 } else if (xmlPullParser.getName().equals("high") && highCount == 4) {
46     eventType = xmlPullParser.next();
47     todayWeather.setHigh4(xmlPullParser.getText().substring(2).trim());
48     highCount++;
49 } else if (xmlPullParser.getName().equals("low") && lowCount == 1) {
50     eventType = xmlPullParser.next();
51     todayWeather.setLow1(xmlPullParser.getText().substring(2).trim());
52     lowCount++;
53 } else if (xmlPullParser.getName().equals("low") && lowCount == 2) {
54     eventType = xmlPullParser.next();
55     todayWeather.setLow2(xmlPullParser.getText().substring(2).trim());
56     lowCount++;
57 } else if (xmlPullParser.getName().equals("low") && lowCount == 3) {
58     eventType = xmlPullParser.next();
59     todayWeather.setLow3(xmlPullParser.getText().substring(2).trim());
60     lowCount++;
61 } else if (xmlPullParser.getName().equals("low") && lowCount == 4) {
62     eventType = xmlPullParser.next();
63     todayWeather.setLow4(xmlPullParser.getText().substring(2).trim());
64     lowCount++;
65 } else if (xmlPullParser.getName().equals("type") && typeCount == 1) {
66     eventType = xmlPullParser.next();
67     todayWeather.setType1(xmlPullParser.getText());
68     typeCount++;
69 } else if (xmlPullParser.getName().equals("type") && typeCount == 2) {
70     eventType = xmlPullParser.next();
71     todayWeather.setType2(xmlPullParser.getText());
72     typeCount++;
73 } else if (xmlPullParser.getName().equals("type") && typeCount == 3) {
74     eventType = xmlPullParser.next();
75     todayWeather.setType3(xmlPullParser.getText());
76     typeCount++;
77 } else if (xmlPullParser.getName().equals("type") && typeCount == 4) {
78     eventType = xmlPullParser.next();
79     todayWeather.setType4(xmlPullParser.getText());
80     typeCount++;
81 }
```

六.修改MainActivity.java

变量的定义：

```
1 private HorizontalListView fw_list;
2 private List<FutureWeather> fwList;
3 private HorizontalAdapter horizontalAdapter;
```

在initview函数中补充以下代码：

```
1 fw_list = (HorizontalListView) findViewById(R.id.future_weather_list);
2 fwList = new ArrayList<FutureWeather>();
3 horizontalAdapter = new HorizontalAdapter(MainActivity.this, fwList);
4 fw_list.setAdapter(horizontalAdapter);
```

添加更新未来天气的相关函数：

```
1 void updateFutureWeather(TodayWeather todayWeather){
2     fwList.clear();
3
4     addingData(todayWeather.getDate1(), todayWeather.getHigh1(), todayWeather.getLow1(), todayWeather.getType1(), todayWeather.getFengli1());
5
6     addingData(todayWeather.getDate2(), todayWeather.getHigh2(), todayWeather.getLow2(), todayWeather.getType2(), todayWeather.getFengli2());
7
8     addingData(todayWeather.getDate3(), todayWeather.getHigh3(), todayWeather.getLow3(), todayWeather.getType3(), todayWeather.getFengli3());
9
10    addingData(todayWeather.getDate4(), todayWeather.getHigh4(), todayWeather.getLow4(), todayWeather.getType4(), todayWeather.getFengli4());
11    horizontalAdapter.updateListView(fwList);
12    Toast.makeText(MainActivity.this, "更新成功! ", Toast.LENGTH_SHORT).show();
13 }
14
15 public void addingData(String data, String high, String low, String type, String fengli){
16     FutureWeather fw = new FutureWeather(data, high, low, type, fengli, getWeatherImgId(type));
17     fwList.add(fw);
18     Log.d("myWeather", data+" "+high+" "+low+" "+type+" "+fengli);
19 }
```

在线程中添加更新函数即可：

```
1 private Handler mHandler = new Handler(){
2     public void handleMessage(android.os.Message msg){
3         switch (msg.what){
4             case UPDATE_TODAY_WEATHER:
5                 updateTodayWeather((TodayWeather) msg.obj);
6                 updateFutureWeather((TodayWeather) msg.obj); //new add
7                 break;
```

```
8         default:
9             break;
10    }
11 }
12 };
```