

实验十 传感器应用及位置服务

【实验目的】

1. 学会使用 android 手机不同的传感器，包括磁强计、加速度传感器等。
2. 学会使用 android 位置服务获得经纬度信息。

【实验内容】



1. 初始界面提示“摇一摇跳转页面”，手机摇一摇跳转页面，并有震动反馈（有兴趣同学可以效仿微信摇一摇实现动画效果和声音效果）；
2. 跳转之后的页面分两部分，上方箭头实现指南针效果，方向始终指向正北方向，箭头下方获取并显示旋转角度（原始正上方向为 0 度）；
3. 下方使用 GPS 或网络进行定位获取到当前的经纬度信息。

【参考内容】

1. 摇一摇有多种实现方法，这里使用加速度传感器实现，通过获取加速度传感器三个方向的重力加速度，通过判断是否超过设定的阈值来识别是否进行了摇一摇操作

```
@Override
public void onSensorChanged(SensorEvent event) {
    float[] values = event.values;
    float x = values[0];
    float y = values[1];
    float z = values[2];

    if ((Math.abs(x) > 17 || Math.abs(y) > 17 || Math
        .abs(z) > 17)) {
        mVibrator.vibrate(300);
        Intent intent = new Intent(MainActivity.this, LocationAndOrientation.class);
        startActivity(intent);
    }
}
```

2. 调用传感器首先要获取传感器的管理器，再通过管理器获取需要类型的传感器

```
mSensorManager = (SensorManager) getSystemService(SENSOR_SERVICE);
mAccelerometerSensor = mSensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
```

3. 传感器要注册才能正常使用，在 Activity 在前台运行（onResume）的时候注册传感器，在离开前台（onPause）的时候取消注册。注意，正常情况下不取消注册传感器，传感器会在后台一直传输数据耗费电量，本次实验由于后续指南针的实现也用到了加速度传感器，为避免冲突记得取消注册。同时在注册传感器的时候，需要指定传感器的监听器，当传感器数据更新的时候，系统会回调监听器里的 onSensorChange 函数，我们便可以在这里对传感器数据进行相应处理

```
@Override
protected void onResume(){
    checkPermission(this, "android.permission.ACCESS_FINE_LOCATION");
    super.onResume();
    mSensorManager.registerListener(mSensorEventListener, mAccelerometerSensor,
        SensorManager.SENSOR_DELAY_UI);
}

@Override
protected void onPause(){
    super.onPause();
    mSensorManager.unregisterListener(mSensorEventListener);
}
```

```
private SensorEventListener mSensorEventListener = new SensorEventListener() {
    @Override
    public void onSensorChanged(SensorEvent event) {
        float[] values = event.values;
        float x = values[0];
        float y = values[1];
        float z = values[2];

        if ((Math.abs(x) > 17 || Math.abs(y) > 17 || Math
            .abs(z) > 17)) {
            mVibrator.vibrate(300);
            Intent intent = new Intent(MainActivity.this, LocationAndOrientation.class);
            startActivity(intent);
        }
    }
    @Override
    public void onAccuracyChanged(Sensor sensor, int accuracy) {}
};
```

4. 实现震动反馈是通过 Vibrator 来获取震动服务，使用前要先获取权限

```
<uses-permission android:name="android.permission.VIBRATE"/>
```

```
mVibrator = (Vibrator) getSystemService(VIBRATOR_SERVICE);
```

5. 指南针的实现首先要获取手机的朝向，Android 在废弃了方向传感器后提供了一个新的方法：通过地磁传感器和加速度传感器获取到的数据来计算，并提供了计算这个数值的 API：首先通过 `getRotationMatrix` 得到一个旋转矩阵，然后使用 `getOrientation` 得到手机的朝向。当然使用传感器还需要按上述步骤获取管理器和注册监听器。指南针旋转可以通过动画效果实现

```
float[] R = new float[9];
float[] values = new float[3];
SensorManager.getRotationMatrix(R, null, accValues, magVaules);
SensorManager.getOrientation(R, values);

newRotation = values[0];
RotateAnimation animation = new RotateAnimation(curRotation, newRotation
    , Animation.RELATIVE_TO_SELF, 0.5f, Animation.RELATIVE_TO_SELF, 0.5f);
animation.setDuration(200);
animation.setFillAfter(true);
arrow.setAnimation(animation);
arrow.startAnimation(animation);
```

6. 位置服务首先也是获取位置管理器，并提供数据来源 provider。Provider 分为 GPS 和网络两种，分别对应 `NETWORK_PROVIDER` 和 `GPS_PROVIDER` 字段。其中 `NETWORK_PROVIDER` 使用基站和 Wi-Fi 信号来决定用户的位置，更新速度更快耗电量更少，但可能精确度较低，而 `GPS_PROVIDER` 使用 GPS 卫星进行定位，精度高但只能在室外使用，耗电量也更高。在使用过程中可以选择使用其中一个 provider，也可以两个 provider 同时使用

```
locationManager = (LocationManager) getSystemService(LOCATION_SERVICE); //获取系统服务
```

```
final String provider = LocationManager.GPS_PROVIDER; //定义定位方法为GPS
```

注意使用位置服务前要先声明权限

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<!-- Needed only if your app targets Android 5.0 (API Level 21) or higher. -->
<uses-feature android:name="android.hardware.location.gps" />
<uses-feature android:name="android.hardware.location.network" />
```

Android6.0 以上实现动态获取权限

```
private static void checkPermission(Context context, String permission){
    int perm = context.checkCallingOrSelfPermission(permission);
    String[] permissions = { permission };
    if(perm != PackageManager.PERMISSION_GRANTED){
        ActivityCompat.requestPermissions((Activity)context, permissions,1);
    }
}

@Override
public void onRequestPermissionsResult(int requestCode, String permissions[], int[] grantResults) {
    if (grantResults.length > 0
        && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
        Toast.makeText(this, "权限获取成功", Toast.LENGTH_SHORT).show();
    } else {
        // 权限被用户拒绝了关闭界面。
        System.exit(0);
    }
}
```

7. 与传感器数据获取方式类似，可以注册 `LocationListener` 监听位置信息的更新 并作出相应的处理，特别地，使用 `getLastKnownLocation` 可以立刻获得一个位置坐标，如果之前有进行过定位操作，那么这个位置就是之前获取到的位置缓存，如果没有获得或位置，那么返回一个空对象

```
//获取缓存位置信息
Location lastKnownLocation = locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER)
```

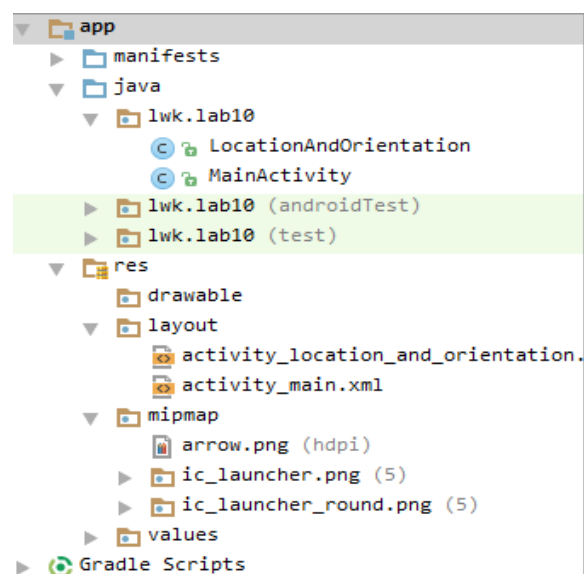
```
locationListener = new LocationListener() {
    @Override
    public void onLocationChanged(Location location) {
        lat = location.getLatitude();
        lon = location.getLongitude();
        latitude.setText(Double.toString(lat));
        longitude.setText(Double.toString(lon));
    }

    @Override
    public void onStatusChanged(String provider, int status, Bundle extras) {}
    @Override
    public void onProviderEnabled(String provider) {}
    @Override
    public void onProviderDisabled(String provider) {}
};
locationManager.requestLocationUpdates(provider, 2000, 10, locationListener);
```

与传感器一样，位置监听器的注册写在 `onResume` 里，取消注册写在 `onPause` 里

```
@Override
protected void onPause(){
    super.onPause();
    mSensorManager.unregisterListener(mSensorEventListener);
    locationManager.removeUpdates(locationListener);
}
```

【参考目录】



【参考资料】

1. Android 官方开发文档——Sensor Overview
https://developer.android.com/guide/topics/sensors/sensors_overview.html
2. Android 官方开发文档——Location Strategies
<https://developer.android.com/guide/topics/location/strategies.html>

【提交说明】

*** 不验收不评分，自由完成