```
1  using UnityEngine;
2  using UnityEngine.UI;
3
4  public class GyroControl : MonoBehaviour
5  {
6      #region 属性
7      Gyroscope myGyro;
8      float slerpFactor = 0.5f;
9      public Toggle tgSlerp;
10     public Slider sdUpdateInterval;
11     #endregion
12
13     // 初始化
14     private void Start()
15     {
16         if (!SystemInfo.supportsGyroscope)
17         {
18             Debug.LogWarning("找不到陀螺仪");
19             return;
20         }
21         myGyro = Input.gyro;
22         myGyro.enabled = true;
23     }
24
25     // 每帧获取陀螺仪方向数据，并改变相机方位
26     private void Update()
27     {
28         if (myGyro.enabled)
29         {
30             if(!tgSlerp.isOn)
31                 transform.rotation = GyroToUnity(myGyro.attitude);
32             else
33             {
34                 transform.rotation = Quaternion.Slerp(transform.rotation,
35                 GyroToUnity(myGyro.attitude), slerpFactor);
36             }
37         }
38     }
39
40     // 坐标转换
41     private Quaternion GyroToUnity(Quaternion q)
42     {
43         return Quaternion.Euler(90, 0, 0) * (new Quaternion(q.x, q.y, -q.z, -
           q.w));
44     }
45
46     // 滑动条消息响应函数
47     public void onChangeUpdateInterval()
48     {
49         myGyro.updateInterval = sdUpdateInterval.value;
50     }
51
52     // 输出辅助信息
53     void OnGUI()
54     {
55         //Output the rotation rate, attitude and
```

```
56          //the enabled state of the gyroscope as a Label
57          GUI.Label(new Rect(10, 30, 600, 40),
58              "Gyro rotation rate:" + myGyro.rotationRate);
59          GUI.Label(new Rect(10, 80, 600, 40),
60              "Gyro attitude:" + myGyro.attitude);
61          GUI.Label(new Rect(10, 130, 600, 40),
62              "Gyro enabled: " + myGyro.enabled);
63          GUI.Label(new Rect(10, 180, 600, 40),
64              "Gyro update Interval: " + myGyro.updateInterval);
65      }
66  }
67
68
69
```