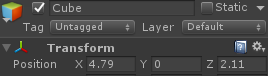
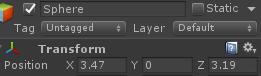
**验证：**

1. **Unity的矩阵是以列为主序**
2. **除了在眼坐标为右手坐标系，其他的都为左手坐标系**



{

Debug.Log("parent row1 = " + obj.transform.localToWorldMatrix.GetRow(0));

Debug.Log("parent row2 = " + obj.transform.localToWorldMatrix.GetRow(1));

Debug.Log("parent row3 = " + obj.transform.localToWorldMatrix.GetRow(2));

Debug.Log("parent row4 = " + obj.transform.localToWorldMatrix.GetRow(3));

Vector3 localPosition = obj.transform.localPosition;

Debug.Log("parent localPosition = " + localPosition);

Vector3 worldPos1 = obj.transform.localToWorldMatrix.MultiplyPoint(**new Vector3(0, 0, 0)**);

Vector3 worldPosition = obj.transform.position;

Debug.Log("parent 1 x = " + worldPos1.x + ", y = " + worldPos1.y + ", z = " + worldPos1.z);

Debug.Log("parent 2 x = " + worldPosition.x + ", y = " + worldPosition.y + ", z = " + worldPosition.z);

}

{

Debug.Log("child row1 = " + subObj.transform.localToWorldMatrix.GetRow(0));

Debug.Log("child row2 = " + subObj.transform.localToWorldMatrix.GetRow(1));

Debug.Log("child row3 = " + subObj.transform.localToWorldMatrix.GetRow(2));

Debug.Log("child row4 = " + subObj.transform.localToWorldMatrix.GetRow(3));

Vector3 localPosition = subObj.transform.localPosition;

Debug.Log("child localPosition = " + localPosition);

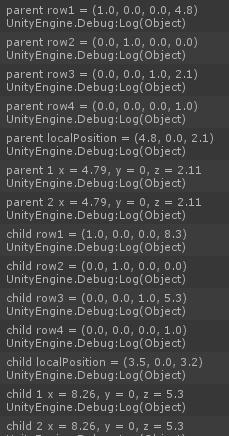
Vector3 worldPos1 = subObj.transform.localToWorldMatrix.MultiplyPoint(**new Vector3(0, 0, 0)**);

Vector3 worldPosition = subObj.transform.position;

Debug.Log("child 1 x = " + worldPos1.x + ", y = " + worldPos1.y + ", z = " + worldPos1.z);

Debug.Log("child 2 x = " + worldPosition.x + ", y = " + worldPosition.y + ", z = " + worldPosition.z);

}



实际上我们通过矩阵的元素可以发现unity的矩阵是以列为主序排列的，与opengl一样

同时我们也会发现localToWorldMatrix是模型坐标转换为世界坐标矩阵，而localPosition仅仅是相对于父坐标系的坐标，并不是模型坐标

关于眼坐标系方向：

{

Debug.Log("obj.transform.position = " + obj.transform.position);

Vector3 eyePosition = camera.worldToCameraMatrix.MultiplyPoint(obj.transform.position);

Debug.Log("eyePosition = " + eyePosition);

}



从上面的结果看出，在世界坐标系（左手坐标系）中的z值为2.1，在眼坐标系中的z值为-2.1，可见眼坐标系为右手坐标系