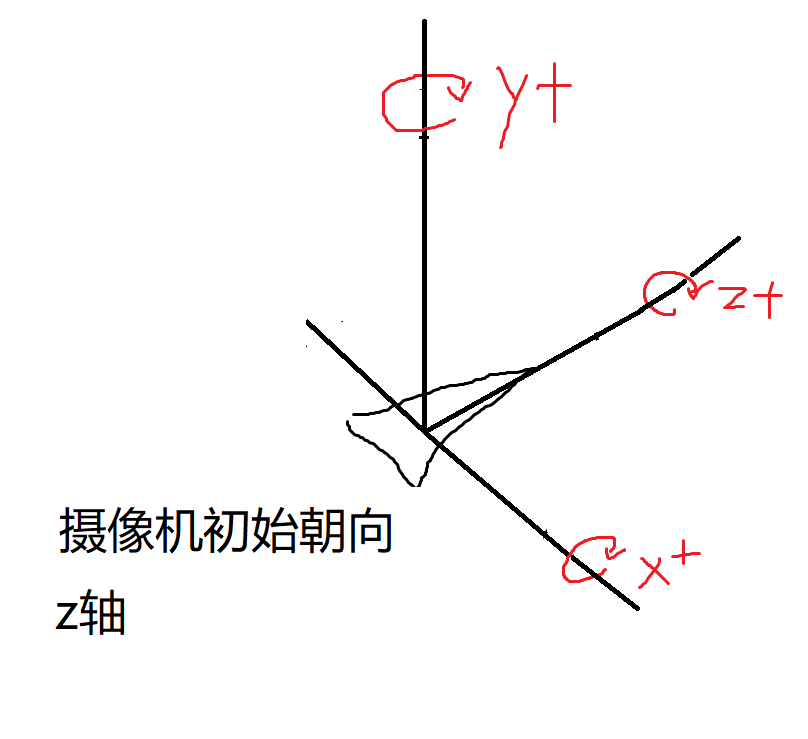
关于摄像机Transform的说明:



Canvas\_CameraRotate(canvas, 0, 1, 0)为绕y轴，沿y+方向旋转,其他类推

**Struct ArrayList**

#define TYPENAME int

struct ArrayList a;

//Initilization

ArrayList\_Init(&a, sizeof(TYPENAME));

TYPENAME tmp = 3;

//The element pushed in must be a pointer

    ArrayList\_PushBack(&a, &tmp);

    tmp = 4;

    ArrayList\_PushBack(&a, &tmp);

    tmp = 5;

ArrayList\_PushBack(&a, &tmp);

//Delete the last element in a

    ArrayList\_PopBack(&a);

    tmp = 6;

ArrayList\_PushBack(&a, &tmp);

//Assign the third element in a to variable b

TYPENAME b = ((TYPENAME \*)a.data)[2];

//b should equal to 6

    Del\_ArrayList(&a);

**Struct Transform**

The child-father relationships of objects are specified in transform.

If I want b be a child of a:

struct C{

    struct Vector3 point;

    struct Transform transform;

};

struct C a, b;

Transform\_Init(&a.transform, NULL);

Transform\_Init(&b.transform, &a.transform);

Get the local information of b:

struct Vector3 vec;

vec = b.transform.position;

vec = b.transform.rotation;

vec = b.transform.scale;

Get the global information of b:

//This function should be called before using the global information

Transform\_UpdateGlobal(&b.transform);

vec = b.transform.globalPosition;

vec = b.transform.globalRotation;

vec = b.transform.globalScale;

struct Vector3 point is a point within b. In other words, the position of this point is respective to the local coordinate system of b. This function could get its global position:

struct Vector3 globalPoint;

globalPoint = b.point;

Transform\_UpdateGlobal(&b.transform);

Transform\_ToGlobal(&b.transform, &globalPoint);

To move a:

struct Vector3 position;

Vector3\_Set(&position, 1, 1, 1);

Transform\_AddPosition(&a.transform, &position);

To rotate a:

struct Vector3 eularAngle;

Vector3\_Set(&eularAngle, 3.14, 0, 0);

Transform\_AddPosition(&a.transform, &position);