Object: Code Snippets

OrxFrame

OrxFX

```
orxObject_AddFX(heroObject, "HitPowerPill");
```

```
[HitPowerPill]
SlotList = ColorFlash

[ColorFlash]
Type        = color
StartTime   = 0.0
EndTime     = 1.0
Curve       = sine
Absolute    = true
Period = 0.5
EndValue   = (0, 128, 255)
StartValue  = (255, 255, 255)
```

OrxFXPointer

OrxObject

OrxObject_Create

```
orxOBJECT *heroObject = orxObject_CreateFromConfig("HeroObject");
orxVECTOR pointA = {500,150,0};
orxObject_SetPosition(heroObject, &pointA);
```

OrxObject_SetPosition

```
orxVECTOR tilePos;
orxVector_Set(&tilePos, orx2F(80.0f) * x, orx2F(160.0f), orxFLOAT_0);
orxObject_SetPosition(tile, &tilePos);
```

```
orxVECTOR pos;
orxObject_GetPosition(player, &pos);
pos.fX = -pos.fX;
orxObject_SetPosition(player, &pos);
```

orxObject_SetRotation

Rotation is set in radians. Zero rad vector is equivalent to (1, 0) vector in screen coordinates. In other words it is a horizontal line pointing from left to right.

Positive rotation is set in clockwise direction. If vector origin was in the center of the screen, then 1 rad would point to the bottom right corner of the screen.

orxObject_SetAngularVelocity

Set angular velocity changes object rotation value over time. Setting positive value make object rotate clockwise.

TBD: What's the unit of measure?

By default object starts with zero rotation angle which points horizontally from left to right. As object rotates a full circle its rotation angle value will not reset to zero. Instead it will continue to grow in positive or negative direction according to angular velocity value.

Thus after one full circle the object rotation value will satisfy the condition:

```
orxMath_Abs(orxObject_GetRotation(obj)) >= orxMATH_KF_2_PI
```

orxObject_GetWorldRotation and orxObject_GetRotation

Returns current object rotation value in rad. The value returned can be any floating value.

See orxObject_SetRotation for coordinate system reference. See orxObject_SetAngularVelocity for discussion of continuous rotation.

orxObject_CreateNeighborList and orxObject_DeleteNeighborList

Use it to obtain objects within the specified bounding box.

```
    orxOBJECT *obj; // comes from mouse click event or in some other way
    orxVECTOR pos, size;
    orxFLOAT range = orx2F(250.f);
    orxOBOX box;

    orxObject_GetWorldPosition(obj, &pos);
    orxVector_Set(&size, range, range, orxFLOAT_0);
    orxOBox_2DSet(&box, &pos, &orxVECTOR_0, &size, orxFLOAT_0);

    orxBANK *neighbors = orxObject_CreateNeighborList(box);
    void* cell = orxNULL;
    while ((cell = orxBank_GetNext(neighbors, cell))) {
```

```
        orxOBJECT **n = cell;
        orxLOG("object name: %s.", orxObject_GetName(*n));
    }
    orxObject_DeleteNeighborList(neighbors);
```

object traversing

Object Traversing has its own page due to somewhat large discussion on the topic.

OrxOBox

Function to return an object within a boxed area:

```
  orxOBJECT* GetObjectInTheArea(){

      orxVECTOR objectPickVector;
      objectPickVector.fX = 878;
      objectPickVector.fY = 1185;
      objectPickVector.fZ = -1.0;
      orxOBOX objectBoxArea;
      orxVECTOR pivot = {0, 0, 0};
      orxVECTOR position;
      position.fX = 834;
      position.fY = 1150;
      position.fZ = -0.1;
      orxVECTOR size;
      size.fX = 21;
      size.fY = 160;
      size.fZ = 1;
      orxOBox_2DSet(&objectBoxArea, &position, &pivot, &size, 0);
      orxU32 objectGroupID = orxCamera_GetGroupID(pstCamera, 1);
      orxOBJECT *objectToFind = orxObject_BoxPick(&objectBoxArea,
objectGroupID);
      return objectToFind;
  }
```

OrxSpawner

```
orxSTRUCTURE *structure = orxStructure_GetFirst(orxSTRUCTURE_ID_SPAWNER );
while (structure != orxNULL){
    orxSPAWNER *spawn = orxSPAWNER(structure);
    if (orxString_Compare(orxSpawner_GetName(spawn), spawnerName) == 0 ){
        return spawn;
    } else {
        structure = orxStructure_GetNext(structure );
    }
```

```
}
```

OrxStructure

```
orxSTRUCTURE *structure = orxStructure_GetFirst(orxSTRUCTURE_ID_OBJECT);
while (structure != orxNULL){
    orxOBJECT *object = orxOBJECT(structure);
    const orxSTRING objectName = orxObject_GetName(object);
    orxSTRUCTURE *nextStructure = orxStructure_GetNext(structure);
    if (orxString_Compare(objectName , "SomeObjectName") == 0){
        orxObject_SetLifeTime(orxOBJECT(structure), orx2F(0.0));
    }
    structure = nextStructure;
}
```

OrxTimeLine

```
orxOBJECT *dummy =
orxObject_CreateFromConfig("BringUpGameOverTimelineObject");
orxObject_AddTimeLineTrack(dummy, "BringUpGameOverTimeline");
orxObject_EnableTimeLine(dummy, orxTRUE);
```

```
[BringUpGameOverTimelineObject]

[BringUpGameOverTimeline]
2.0 = "ShowGameOver"
```

OrxVector

Some ways to initialise an empty orxVECTOR.

Avoid this:

```
  orxVECTOR position;
```

Rather, do one of these:

```
  orxVECTOR position = {0, 0, 0};
```

```
  orxVECTOR position = orxVECTOR_0;
```

```
  orxVECTOR position;
  position.fX = 0;
  position.fY = 0;
```

```
  position.fZ = 0;
```

Uninitialised orxVECTORs can create unintended consequences in your game.

From:
https://www.orx-project.org/wiki/ - **Orx Learning**

Permanent link:
https://www.orx-project.org/wiki/en/orx/reference/object/snippets?rev=1417126310

Last update: **2025/09/30 17:26 (14 months ago)**