|  |  |  |
| --- | --- | --- |
| Controller | pawn |  |
| Provides control  Interprets input  No collision | Has collision  Has model  Has pawn state |  |
|  |  |  |

PlayerControllerClass=class'UTGame.UTPlayerController'

controller

pawn

controller c

pawn p

p = c.pawn;

if (p!= none && p.health >=1)

Contorller.possess(pawn)

controller.unpossess()

set !bUseClassicHUD to true

if use own HUD type

playcontroller has myHUD

controller -> pawn: controller.pawn

pawn -> controller pawn.controller

contonroller -> hud: controller.hud

Exec functions

member func wqualfier exec

can be called from console

can bind input to it

keyword out

has lifetime more than function

multiple returns

udk make && L6-Map.udk –log

udkgames script

roll=y -> pitch=x -> yaw=z

UDK: 360 degree = 65536 = 2 power 16

Short 2 bytes

Range = -32768 to 32767

Positive axis pointing at u, positive rotator is clockwise

When cast vector into rotator, the roll of rotator is lost

R rotator(v);

November 3, 2015

## Interface

Declare a set of public functions, eg functions

Useful for

* Gurantee implementing class declared functions
* Interface tagging

**Myinterface.uc**

Interface myInterface;

Vector function GetVect();

Float function add(float f1, float f2);

**Myactor.uc**

Class myActor extends actor implements myinterface;

**Object**

Actors can be put in a map

Object is a non placeable unreal script class

W8object.uc

Class w8object extends object;

Var w8object v;

Function creator() {

Local w8Object;

v = new class’W80Object’;

}

Function remove() {

V = none;

}

Object outer;

Playerinput

Input bindings macros

Command can be :

Exec functions

Piped exec functions

Piped other macros

Modifiers + exec function or variables

**Class literal:**

Class’<package>.<classname>’

local Texture2D myTex;

myTex.SizeX;

in

x,y,z

out x, y coordinate depth z

(

<0 cord in front

0 coord is on plane of cam

>0 coord behind

)

if want to know cor visibility

z>0

x>0 , y>0

x<hud.sizex

y<hud.sizey

deproject

world pos

world direction