cisc3660
game programming
fall 2012
lecture # II.2

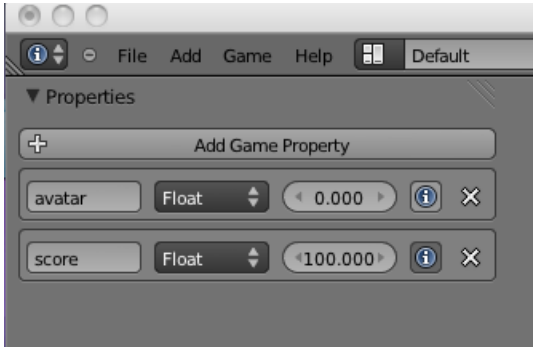
topics:

- game object properties
- collision detection
- more on actuators, including animation
- more on controllers

references:

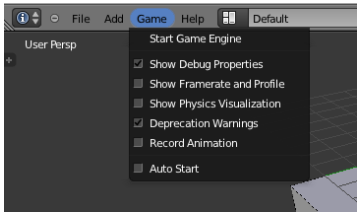
- `https://sites.google.com/site/blendergameprojects/`, by Prof Tim Hickey, Brandeis University (`http://www.cs.brandeis.edu/~tim`)
- Blender Game Engine Overview, User Manual version 2.6 `http://wiki.blender.org/index.php/Doc:2.6/Manual/Game_Engine`

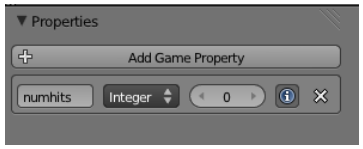
game properties



- game properties are like variables
- they can be of the following data types: *Timer*, *String*, *Float*, *Integer*, *Boolean*

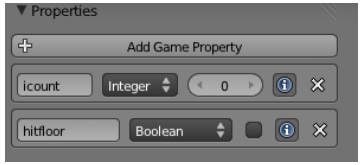
how to show game properties



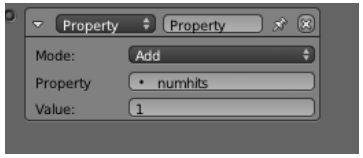


- select "Show Debug Properties" in the Game menu
- click on the "i" box in the properties panel for the property whose value you would like to display in the game window

collision detection



floor object



cube object

- define a property in one object—this essentially gives your object a name
- in another object, create a collision sensor and give it the name (property) of the first object

more types of actuators

- sound
- image filters
- object edit
- animation

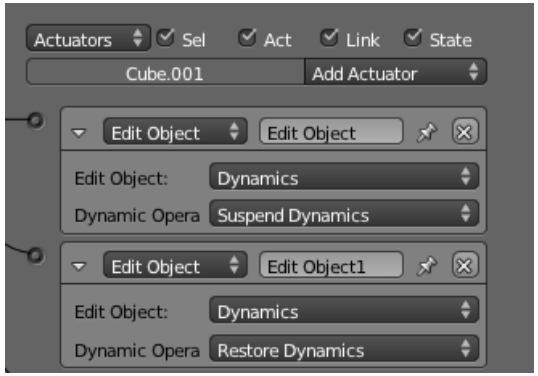
sound actuator

- you can link your game to external sound files and use an actuator to play them
- this is how you can create sound effects tied to certain actions that occur in your game

filter actuator

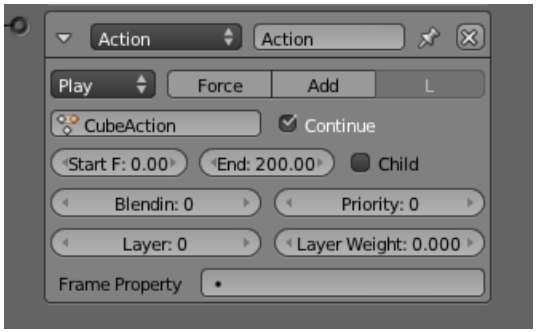
- some of an object's visual properties can be changed by the *Filter2D* actuator

object editing actuator



- some of an object's behavioral properties can be changed by the *Edit Object* actuator

animation actuator



- the **Start** and **End** refer to frames (i.e., frame numbers) that you have created using the *Timeline* in Blender Render mode

more on controllers

- an "expression" controller lets your game respond based on values of properties

  (from online manual `http://wiki.blender.org/index.php/Doc:2.6/Manual/Game_Engine/Logic/Controllers/Expression`)
- Evaluates a user written expression, and gives a positive (TRUE) output when the result of a controller expression is TRUE, and the object is in the designated State. For all other conditions the controller gives a negative (FALSE) output.
- expression can consist of variables, constants and operators
  - variables: sensor names, game properties
  - operations: mathematical operators (`+*, /, +, -`), logical operators (`<, >, >=, <=, ==, !=`), booleans operators (AND, OR, NOT)
  - conditional statements (if):
    `if( expression, pulse_if_expression_is_true, pulse_if_expression_is_false )`

expression controller

- top controller sends TRUE pulse if value of `icount` property is `0`
- bottom controller sends TRUE pulse if value of `icount` property is `1`

example to try

- from `https://sites.google.com/site/blendergameprojects/`, by Prof Tim Hickey, Brandeis University (`http://www.cs.brandeis.edu/~tim`)
- Shooting 2.61
  Demo of a "skeet shooting" game, where a cone slowly advances toward an avatar. The avatar (user) can shoot balls at the cone (using spacebar) and can change the shooting angle using the arrow keys. When the cone is hit, it jumps back 100 meters and the continues to move forward (and left and right).