

Character Behaviors That Depend on Seeing the Player

Prof. Jim Whitehead
CMPS 80K, Winter 2006
February 27, 2006

“Seeing” the Player

- Often you want a computer character to take some action only after it sees the player.
 - The computer character may be in a maze, or in a building
 - Makes the computer characters seem more realistic, since it seems that they can see
- Examples:
 - Character only fires at the player when it can see the player
 - Character changes its movement once it sees the player

Seeing in Game Maker

- Built-in function: `collision_line`
 - You provide:
 - Two endpoints of a line
 - The name of an object
 - It returns:
 - True: if it finds any instances of the named object along the line
 - False: it finds no instances of the named object along the line

Example of collision_line

- collision_line(self.x,self.y, pacman.x,pacman.y, wall_obj, false, false)
 - self.x and self.y give the position of the computer character
 - Within an object, “self” refers to the object (i.e., “me”, “myself”)
 - When using this function within the actions associated with an event defined on a given object, “self” refers to the object
 - pacman.x and pacman.y are the coordinates of the player character
 - “pacman” is the name of the player’s object – substitute with the name of your player object in your game
 - wall_obj – the function will look for any instances of this object along the line.
 - “wall_obj” is the name of the object that represents walls
 - false, false
 - See next slide

Example of collision_line (cont'd)

- False, false
 - The first “false” indicates that the collision detection should not be precise, and should use the bounding box for the object
 - Use “true” for precise checking
 - The second “false” indicates that the calling object instance will be included in collision checking
 - This doesn't do anything, since we're looking for another kind of object
 - Would make sense in situations where you were looking for collisions with other objects of the same kind.
 - For example, perhaps a computer character will only fire a laser at the player if it can see the player, and then only if it has a “clear shot” (i.e., there are no other computer characters between it and the player).
 - Would do two checks, one collision_line to see if there are any collisions with walls, and a second to check for collisions with other computer characters

Where to use collision_line in Game Maker

- In the step event for a computer character
- Action: if expression is true
 - Octagon with a question mark on “Control” panel
 - In the dialog box that pops up
 - Enter the collision_line function call in the “expression” box
 - When checking for walls, will need to check the “NOT” box
 - collision_line will be true if there are any collisions with walls between the computer character and the player
 - When a character “sees” the player, there are no walls between the character and the player
 - Hence collision_line will be “false”
 - The NOT makes it so that the condition is true when you do not have any collisions
 - That is, the NOT reverses the outcome, making it so that the overall condition is true when there are no walls, and the computer character “sees” the player

What to do when the computer character “sees” the player

- Many actions can be taken when the computer character sees the player
 - It can fire at the player
 - “Create” a bullet object
 - It can change its movement
 - Start moving towards the player
 - It could even “talk” with other computer characters
 - Create a variable on the player, “been_seen”
 - Set it to 1 when any character has seen the player
 - Character behavior can depend on the value of this variable