**GameMaker – Coding Sheet # 3 – Collision Commands**

|  |  |  |
| --- | --- | --- |
| **Function** | **Purpose** | **Example** |
| place\_empty | checks to see if a position in the room is free from objects. | if place\_empty(100,100) then …. |
| place\_free | check to see if a position in the room is free from objects marked as solid | if place\_free(200,200) then |
| place\_meeting | checks to see if a position in the room is occupied by a type of object | if place\_meeting(x, y, objTree) then … |
| position\_empty | checks to see if a position in the room is free from any object’s collision mask | if position\_empty(x, y) then |
| position\_meeting | checks to see if a position in the room collides with an object’s collision mask | if position\_meeting(x, y, objBullet) then |
| position\_change | will change any object at that position into a new object type. You can also control whether or not to call the new object’s create event with true or false. | position\_change(mouse\_x, mouse\_y, objExplosion, true); |
| position\_destroy | will destroy any object at that location in the room |  |
| instance\_place | like place\_meeting but will return the actual object that it collides with | var inst;  inst = instance\_place(x, y, objEnemy); |
| collision\_circle | checks in a circle area around a given point, with a given radius for a specific object type | if collision\_circle(x, y, 50, objHero,false, false) then  (checks within a 50 pixel radius of (x, y) for a objHero object) |
| collision\_line | checks in a line (between 2 points) for a collision with a given object | if collision\_line(x, y, 200, 200, objEnemy, false, false) then |
| collision\_point | check a single pixel for a collision with a given object | if collision\_point(300,300, objWall, false, false) then… |
| collision\_rectangle | check for a collision in a rectangular region of the room with an object type. The four values are the X,Y of the top left of the rectangle and the X, Y of the bottom right corner of the rectangle | if collision\_rectange(100, 100, 250,250, objEnemy, false, false)… |
| move\_wrap | will allow an object to wrap vertically and/or horizontally in the room | move\_wrap(true, true, 32)  (horizontal, vertical, sprite\_size) |