TITLE: COLLECTOR

Logline: It’s not stealing

Joseph Swetz
Login: jsswetz

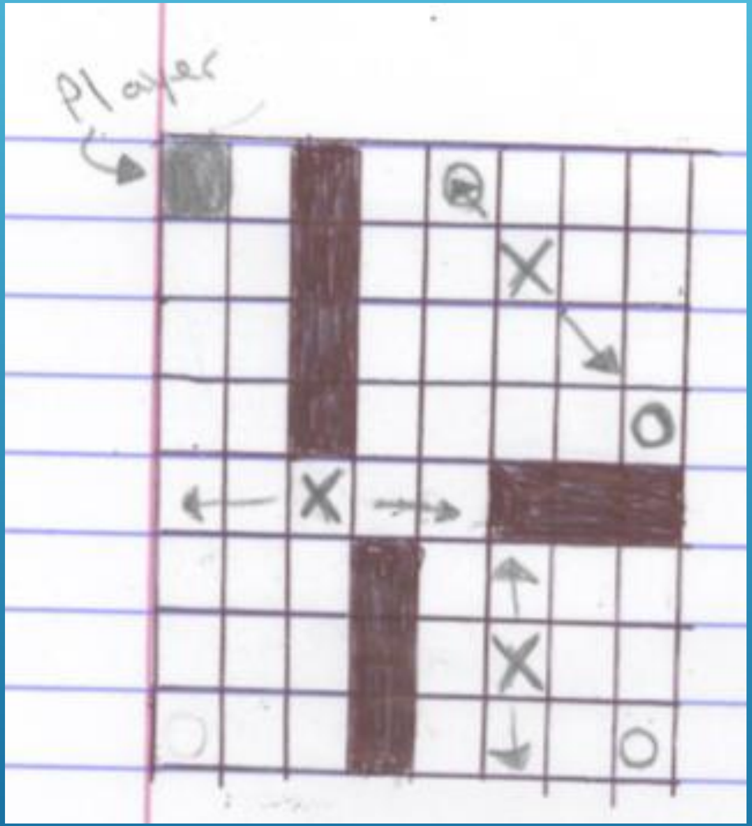
Play Description

The player moves a square around the board. The color and glyph of the player's square changes to indicate which square (each of which has its own color and glyph) they must collect next. The goal of each level is to collect all of these squares on the board in the correct order without getting hit by the moving gray squares (with an "X" glyph). Some of these gray squares may be fake, and the player can only find out by colliding with it. Upon collision with an actual moving gray square, the player must restart the level. If the gray square is fake, then it fades away and the player continues the level.

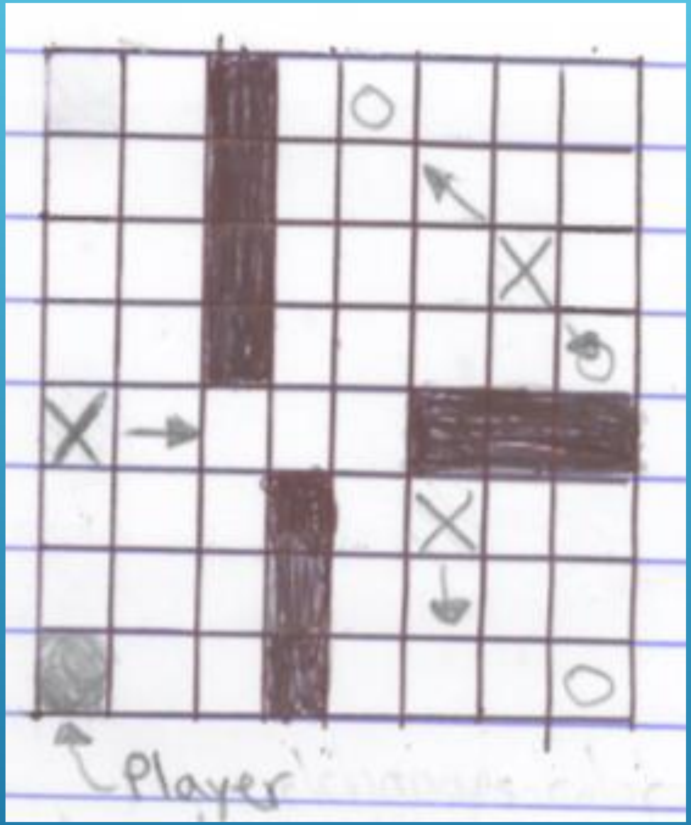
Mockups

X = Gray squares

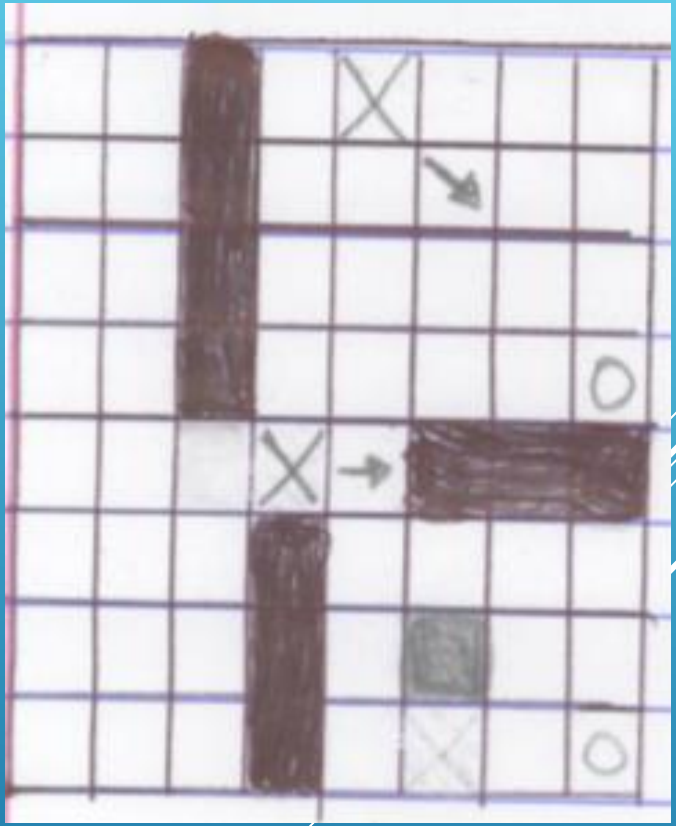
O = Colors to be collected



Start as the color you must collect next



Once collected, the player changes color to the next one to collect.

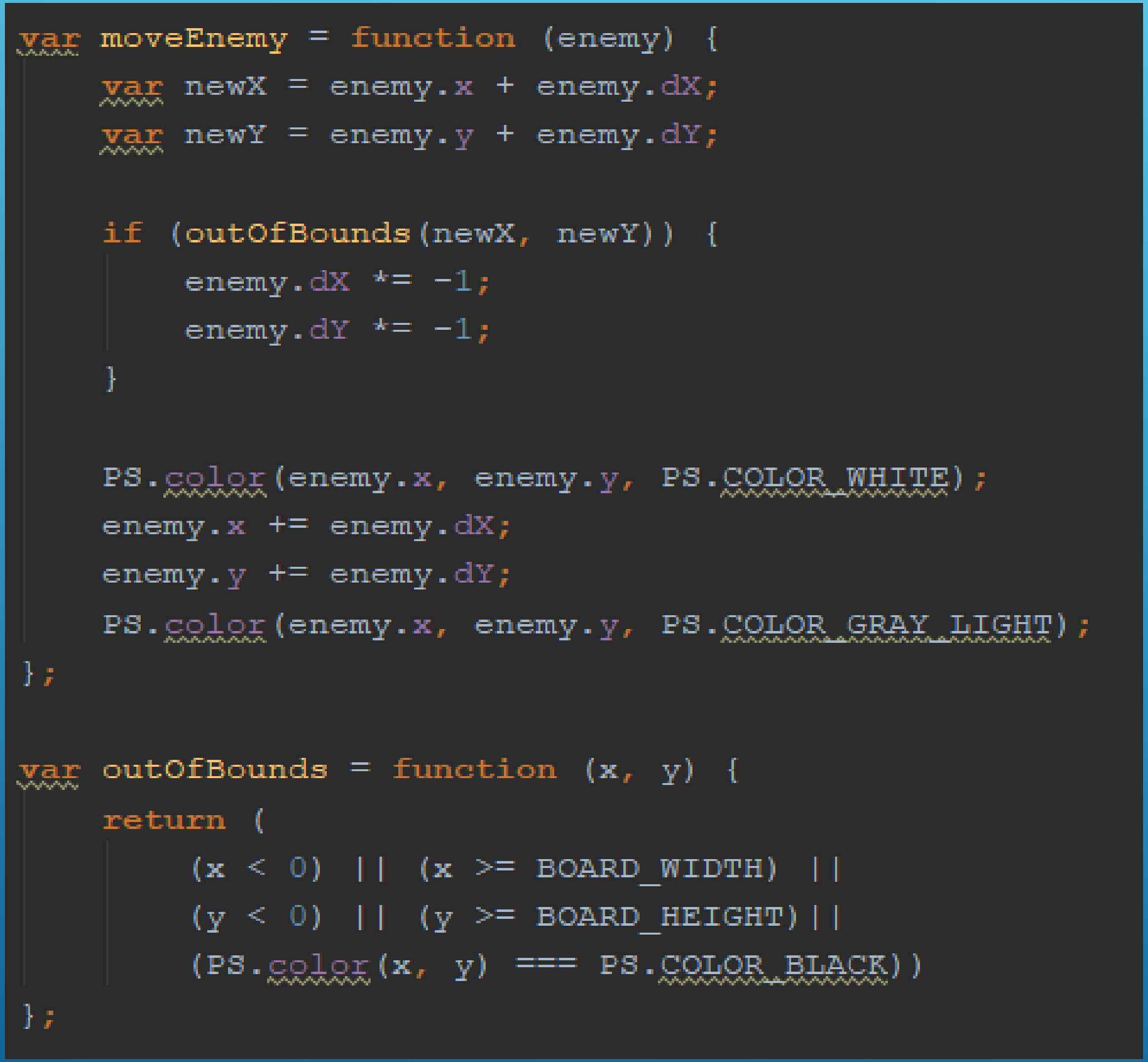


Upon colliding with a fake enemy, It fades away.

Features

- Design restrictions
  - Input only from the keyboard
  - Max grid size of 16x16 beads
  - No words, initials, numbers or labels in any language except for title in status line.
- Emotional Goal: Surprise
  - Random order of collection in each level.
  - Gray squares may or may not be real.
    - If a gray square is real, collision results in a loss.
    - If a gray square is not real, it will fade way upon collision, leaving the player unharmed.
- Multiple Levels
  - Start out with simple levels to show mechanics of the game.
  - To increase the difficulty:
    - Increase the number of collectible squares and gray squares.
    - Set the paths of the gray squares so player must time their movements just right.
- Audio
  - Sound effects for collision, picking up the right square, and trying to pick up the wrong square.
- Various Visual Cues
  - Effects for collision and collection so audio is not required to play.

Demo Code: Movement



```
var moveEnemy = function (enemy) {
    var newX = enemy.x + enemy.dX;
    var newY = enemy.y + enemy.dY;

    if (outOfBounds(newX, newY)) {
        enemy.dX *= -1;
        enemy.dY *= -1;
    }

    PS.color(enemy.x, enemy.y, PS.COLOR_WHITE);
    enemy.x += enemy.dX;
    enemy.y += enemy.dY;
    PS.color(enemy.x, enemy.y, PS.COLOR_GRAY_LIGHT);
};

var outOfBounds = function (x, y) {
    return (
        (x < 0) || (x >= BOARD_WIDTH) ||
        (y < 0) || (y >= BOARD_HEIGHT)||
        (PS.color(x, y) === PS.COLOR_BLACK))
};
```

- Keep track of the change in X and Y that the gray square must make with each call to its timer.
- Prevent gray squares from going through walls.