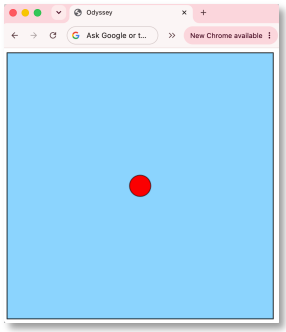
IG Problem 2A: The Odyssey

Written by Ben Yan




```
/* Constants (in pixels) */
const GWINDOW_HEIGHT = 500;
const GWINDOW_WIDTH = 500;

const SHIP_RADIUS = 20;
const SHIP_DIAMETER = 2 * SHIP_RADIUS;
const DRIFT = 20; // pixels moved in each time step

/* Constants (in milliseconds) */
const TIME_STEP = 500;

/* Main program to playfully imitate the story of King Odysseus's return */
function Odyssey() {
	let gw = GWindow(GWINDOW_WIDTH, GWINDOW_HEIGHT);

	// top-left coordinate of ship when it starts at the window's center
	let centerX = GWINDOW_WIDTH / 2 - BOAT_RADIUS;
	let centerY = GWINDOW_HEIGHT / 2 - BOAT_RADIUS;

	// introduce the ship as a small, centered red circle
	let ship = GOval(centerX, centerY, BOAT_DIAMETER, BOAT_DIAMETER);
	ship.setFillColor("red");
	ship.setFilled(true);
	gw.add(ship);

	// continually move ship in random directions until it returns to center
	let timer = setInterval(moveBoat, TIME_STEP);

	let moveBoat = function() {
		let angle = randomInteger(0, 3) * 90; // one of 4 cardinal angles
		ship.movePolar(DRIFT, angle);

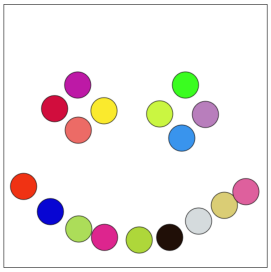
		if (hasReturnedHome()) clearInterval(timer);

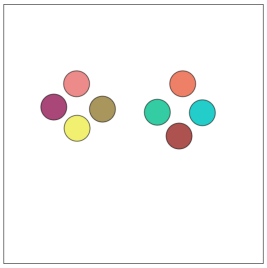
		console.log("Daddy is back"); // this is very optional, why Ben
	}

	let hasReturnedHome = function() {
		return (ship.getX() == centerX) && (ship.getY() == centerY);
	}
}
```

IG Problem 2B: Holiday Lights

Written by Ben Yan





```
/* Constants (in pixels) */
const GWINDOW_WIDTH = 500;
const GWINDOW_HEIGHT = 500;
const LIGHT_RADIUS = 25;
const LIGHT_DIAMETER = 2 * LIGHT_RADIUS;
s
/* Constants (in milliseconds) */
const COLOR_CHANGE_DELAY = 1000;

/* Main program, where you enable adding and removing holiday lights */
function HolidayLights() {
      let gw = GWindow(GWINDOW_WIDTH, GWINDOW_HEIGHT);

      let addLight = function(e) {
            /* create a randomly colored light, centered at the mouse click */
            let light = GOval(LIGHT_DIAMETER, LIGHT_DIAMETER);
            light.setFilled(true);
            light.setFillColor(randomColor());
            gw.add(light, e.getX() - LIGHT_RADIUS, e.getY() - LIGHT_RADIUS);

            /* timer animation to change the color every 1000 milliseconds */
            let changeColor = function() {
                  light.setFillColor(randomColor());
            }
            let timer = setInterval(changeColor, COLOR_CHANGE_DELAY);
      }

      let clickAction = function(e) {
            let obj = gw.getElementAt(e.getX(), e.getY());

            if (obj === null) {
                  // add a new light at the empty location we clicked
                  addLight(e);

            } else {
                  // remove the existing light that was clicked on
                  gw.remove(obj);
            }
      }

      gw.addEventListener("click", clickAction);
}
```

Bonus: If you're curious how to terminate the color animation of a removed light—which we in practice should do, so that we don't overwhelm the graphics window in endlessly ongoing animations—**we need some way of storing and remembering each light's `timer`**, then calling **`clearInteval`** on only that timer.

The following is a neat technique you'll learn in Assign4, so you're not expected to know it right now ☺. Because **`GOval`**s, and **`GObject`**s more broadly, are JavaScript objects, you can assign a property to them beyond the ones that exist by default, e.g., **`width, height, x, y`**. For instance,



```
light.name = "Yagami";  // not sure why you'd do this, but it works
console.log(light.name);  // prints the string "Yagami"
```

As such, we store each timer as a property of its corresponding light, e.g., **`light.timer = timer`**. Here's a refurbished implementation that supports exactly this. The added lines are enclosed in boxes yay.

```
function HolidayLights() {
      let gw = GWindow(GWINDOW_WIDTH, GWINDOW_HEIGHT);

      let addLight = function(e) {
            /* create a randomly colored light, centered at the mouse click */
            let light = GOval(LIGHT_DIAMETER, LIGHT_DIAMETER);
            light.setFilled(true);
            light.setFillColor(randomColor());
            gw.add(light, e.getX() - LIGHT_RADIUS, e.getY() - LIGHT_RADIUS);

            /* timer animation to change the color every 1000 milliseconds */
            let changeColor = function() {
                  light.setFillColor(randomColor());
            }
            let timer = setInterval(changeColor, COLOR_CHANGE_DELAY);
            light.timer = timer;
      }

      let clickAction = function(e) {
            let obj = gw.getElementAt(e.getX(), e.getY());

            if (obj === null) {
                  // add a new light at the empty location we clicked
                  addLight(e);

            } else {
                  // remove the clicked-on light, and clear its animation
                  gw.remove(obj);
                  clearInterval(obj.timer);
            }
      }

      gw.addEventListener("click", clickAction);
}
```