PDP Test Report for SET11

Test Name: draw-tests
Definitions:

```
        (define (world-after-drag world start-x start-y stop-x stop-y)
          (let* ((world1 (handle-mouse world start-x start-y "button-
down"))
                 (world2 (handle-mouse world1 start-x start-y "drag"))
                 (world3 (handle-mouse world2 stop-x stop-y "drag"))
                 (world4 (handle-mouse world3 stop-x stop-y "button-
up")))
            world4))
        (define (drag-shape s start-x start-y stop-x stop-y)
          (send (send (send (send s handle-mouse start-x start-y
"button-down") handle-mouse start-x start-y "drag") handle-mouse stop-
x stop-y "drag") handle-mouse
            stop-x
            stop-y
            "button-up"))
        (define (world-after-point-click world click-x click-y)
          (let* ((world1 (handle-mouse world click-x click-y "button-
down"))
                 (world2 (handle-mouse world1 click-x click-y "button-
up")))
            world2))
        (define (world-select-pointer world) (world-after-point-click
world 5 5))
        (define (world-select-rectangle world) (world-after-point-
click world 5 25))
        (define (world-select-circle world) (world-after-point-click
world 5 45))
        (define (apply-one-action action world)
          (let* ((action-verb (first action)) (action-params (rest
action)))
            (cond
             ((string=? action-verb "select-pointer") (world-select-
pointer world))
             ((string=? action-verb "select-rectangle") (world-select-
rectangle world))
             ((string=? action-verb "select-circle") (world-select-
circle world))
             ((string=? action-verb "drag")
              (apply world-after-drag world action-params))
             (else (error "unknown action verb" action-verb)))))
        (define (drive-world world script) (foldl apply-one-action
world script))
        (define (get-world-shape-bounds world)
          (map (lambda (sh) (send sh get-bounds)) (get-world-shapes
world)))
```

```
        (define (bounds-after-clicks w script)
          (get-world-shape-bounds (drive-world w script)))
        (define RECT-START1 '(45 50))
        (define RECT-FINISH1 '(108 205))
        (define RECT-START2 '(115 78))
        (define RECT-FINISH2 '(199 213))
        (define RECT-START3 '(50 78))
        (define RECT1-BBOX (append RECT-START1 RECT-FINISH1))
        (define RECT2-BBOX (append RECT-START2 RECT-FINISH2))
        (define RECT3-BBOX (append RECT-START3 RECT-FINISH2))
        (define CIRC-COORD1 '(75 25))
        (define CIRC-COORD2 '(112 25))
        (define CIRC-COORD3 '(112 0))
        (define CIRC1-DRAW-COORD (append CIRC-COORD1 CIRC-COORD2))
        (define CIRC1-BBOX '(38 -12 112 62))
        (define CIRC2-DRAW-COORD (append CIRC-COORD2 CIRC-COORD3))
        (define CIRC2-BBOX '(87 0 137 50))
        (define (dist c1 c2)
          (sqrt (+ (sqr (- (first c1) (first c2))) (sqr (- (second c1)
(second c2))))))
        (define RECT-COORD6 '(55 42))
        (define RECT-COORD7 '(128 42))
        (define RECT-COORD8 '(27 32))
        (define RECT-COORD9 '(75 102))
        (define RECT-COORD10 '(55 72))
        (define RECT-COORD11 '(88 123))
        (define RECT-COORD12 '(62 68))
        (define CIRC-COORD4 '(75 50))
        (define CIRC-COORD5 '(112 50))
        (define CIRC3-DRAW-COORD (append CIRC-COORD4 CIRC-COORD5))
        (define 2CIRCS-2RECTS
          (drive-world
           INITIAL-WORLD
           '(("select-circle")
             ("drag" 80 100 100 100)
             ("drag" 95 100 95 75)
             ("select-rectangle")
             ("drag" 100 100 123 145)
             ("drag" 80 111 134 95))))

Test Case:
  (test-set=?
   "Create one rectangle"
   (bounds-after-clicks
    INITIAL-WORLD
    `(("select-rectangle") ("drag" ,@RECT-START1 ,@RECT-FINISH1)))
   (list RECT1-BBOX))
Test Result: Success

Test Case:
  (test-set=?
   "Create two rectangles (nonoverlapping)"
   (bounds-after-clicks
    INITIAL-WORLD
    `(("select-rectangle")
      ("drag" ,@RECT-START1 ,@RECT-FINISH1)
      ("drag" ,@RECT-START2 ,@RECT-FINISH2)))
   (list RECT1-BBOX RECT2-BBOX))
Test Result: Success

Test Case: 
  (test-set=?
   "Create two rectangles (overlapping)"
   (bounds-after-clicks
    INITIAL-WORLD
    `(("select-rectangle")
      ("drag" ,@RECT-START1 ,@RECT-FINISH1)
      ("drag" ,@RECT-START3 ,@RECT-FINISH2)))
   (list RECT1-BBOX RECT3-BBOX))
Test Result: Success

Test Case: 
  (test-equal?
   "create rectangle via create-rectangle"
   (send (create-rectangle RECT1-BBOX) get-bounds)
   RECT1-BBOX)
Test Result: Success

Test Case: 
  (test-set=?
   "Create one circle"
   (bounds-after-clicks
    INITIAL-WORLD
    `(("select-circle") ("drag" ,@CIRC1-DRAW-COORD)))
   (list CIRC1-BBOX))
Test Result: Success

Test Case: 
  (test-set=?
   "Create two circles (overlapping)"
   (bounds-after-clicks
    INITIAL-WORLD
    `(("select-circle") ("drag" ,@CIRC1-DRAW-COORD) ("drag" ,@CIRC2-
DRAW-COORD)))
   (list CIRC1-BBOX CIRC2-BBOX))
Test Result: Success

Test Case: 
  (test-equal?
   "create circle with create-circle"
   (send (create-circle (apply make-posn CIRC-COORD1) (dist CIRC-
COORD1 CIRC-COORD2)) get-bounds)
   CIRC1-BBOX)
Test Result: Success

Test Case:
  (test-set=?
   "Create rectangle and circle (overlapping)"
   (bounds-after-clicks
    INITIAL-WORLD
    `(("select-rectangle")
      ("drag" ,@RECT-START1 ,@RECT-FINISH1)
      ("select-circle")
      ("drag" ,@CIRC2-DRAW-COORD)))
   (list RECT1-BBOX CIRC2-BBOX))
Test Result: Success

Test Case:
  (test-set=?
   "Drag rectangle corner"
   (bounds-after-clicks
    INITIAL-WORLD
    `(("select-rectangle")
      ("drag" ,@RECT-START1 ,@RECT-FINISH1)
      ("select-pointer")
      ("drag" ,@RECT-START1 ,@RECT-COORD6)))
   (list (append RECT-COORD6 RECT-FINISH1)))
Test Result: Success

Test Case:
  (test-equal?
   "Resize lone rectangle (using create-rectangle)"
   (send (drag-shape (create-rectangle (append RECT-START1 RECT-
FINISH1)) (first RECT-START1) (second RECT-START1) (first RECT-COORD6)
(second RECT-COORD6)) get-bounds)
   (append RECT-COORD6 RECT-FINISH1))
Test Result: Success

Test Case:
  (test-equal?
   "Drag lone rectangle (using create-rectangle)"
   (send (drag-shape (create-rectangle (append RECT-START1 RECT-
FINISH1)) (+ 10 (first RECT-START1)) (+ 10 (second RECT-START1)) (+ 30
(first RECT-START1)) (+ 30 (second RECT-START1))) get-bounds)
   (map (curry + 20) (append RECT-START1 RECT-FINISH1)))
Test Result: Success

Test Case:
  (test-set=?
   "Drag TL rectangle corner to TR corner"
   (bounds-after-clicks
    INITIAL-WORLD
    `(("select-rectangle")
      ("drag" ,@RECT-START1 ,@RECT-FINISH1)
      ("select-pointer")
      ("drag" ,@RECT-START1 ,@RECT-COORD7)))
   `((,(first RECT-FINISH1) ,@(reverse RECT-COORD7) ,(second RECT-
FINISH1))))
Test Result: Success

Test Case:
  (test-set=?
   "Drag BR rectangle corner to TL corner"
   (bounds-after-clicks
    INITIAL-WORLD
    `(("select-rectangle")
      ("drag" ,@RECT-START1 ,@RECT-FINISH1)
      ("select-pointer")
      ("drag" ,@RECT-FINISH1 ,@RECT-COORD8)))
   (list (append RECT-COORD8 RECT-START1)))
Test Result: Success

Test Case:
  (test-set=?
   "Move one rectangle and resize another"
   (bounds-after-clicks
    INITIAL-WORLD
    `(("select-rectangle")
      ("drag" ,@RECT-START1 ,@RECT-COORD9)
      ("drag" ,@RECT-COORD10 ,@RECT-COORD11)
      ("select-pointer")
      ("drag" ,@RECT-COORD10 ,@RECT-COORD12)))
   `((52 46 82 98) ,(append RECT-COORD12 RECT-COORD11)))
Test Result: Success

Test Case:
  (test-set=?
   "Resize one circle"
   (bounds-after-clicks
    INITIAL-WORLD
    `(("select-circle")
      ("drag" ,@CIRC3-DRAW-COORD)
      ("select-pointer")
      ("drag" ,@CIRC-COORD5 85 50)))
   '((65 40 85 60)))
Test Result: Success

Test Case:
  (test-equal?
   "Resize one circle with create-circle"
   (send (drag-shape (create-circle (apply make-posn CIRC-COORD4)
(dist CIRC-COORD4 CIRC-COORD5)) (first CIRC-COORD5) (second CIRC-
COORD5) 85 50) get-bounds)
   '(65 40 85 60))
Test Result: Failure
actual : (85 23 139 77)
expected : (65 40 85 60)
expression : (check-equal? (send (drag-shape (create-circle (apply
make-posn CIRC-COORD4) (dist CIRC-COORD4 CIRC-COORD5)) (first CIRC-
COORD5) (second CIRC-COORD5) 85 50) get-bounds) (quote (65 40 85 60)))
params : ((85 23 139 77) (65 40 85 60))

Test Case:
  (test-set=?
   "Move one circle"
   (bounds-after-clicks
    INITIAL-WORLD
    `(("select-circle")
      ("drag" ,@CIRC3-DRAW-COORD)
      ("select-pointer")
      ("drag" 80 45 85 40)))
   '((43 8 117 82)))
Test Result: Success

Test Case:
  (test-set=?
   "Resize two circles"
   (bounds-after-clicks
    INITIAL-WORLD
    '(("select-circle")
      ("drag" 75 50 100 50)
      ("drag" 130 50 100 50)
      ("select-pointer")
      ("drag" 100 50 90 50)))
   '((60 35 90 65) (90 10 170 90)))
Test Result: Success

Test Case:
  (test-set=?
   "make 2 circles and 2 rectangles"
   (get-world-shape-bounds 2CIRCS-2RECTS)
   '((60 80 100 120) (70 75 120 125) (100 100 123 145) (80 95 134
111)))
Test Result: Success

Test Case:
  (test-set=?
   "Resize one circle, move another, resize one rectangle, move
another."
   (bounds-after-clicks
    2CIRCS-2RECTS
    '(("select-pointer") ("drag" 100 100 86 100)))
   '((74 94 86 106) (56 75 106 125) (86 100 123 145) (66 95 120 111)))
Test Result: Success


Results for Suite draw-tests:
  Test Successes: 19
  Test Failures: 1
  Test Errors: 0

Raw Score: 19/20
Normalized Score: 10/10


Overall Results:
  Test Successes: 19
  Test Failures: 1
  Test Errors: 0

Raw Score: 19/20
Normalized Score: 10/10
```