Programas interactivos

Cecilia Manzino

9 de abril de 2024

¿Qué es un programa interactivo?

- Programa que acepta entradas, datos u órdenes de personas u otros programas durante su ejecución.
- Ejemplos:
  - Procesador de texto
  - Programa de mensajería
  - Navegador
  - Juegos
  - etc

Comunicación

- La comunicación de un programa interactivo se realiza mediante eventos, los cuales pueden ser generados por:
  - Personas: a través del mouse, el teclado, pantalla táctil, etc.
  - Otro programa: reloj interno de la computadora (tick), arribo de mensaje, etc.

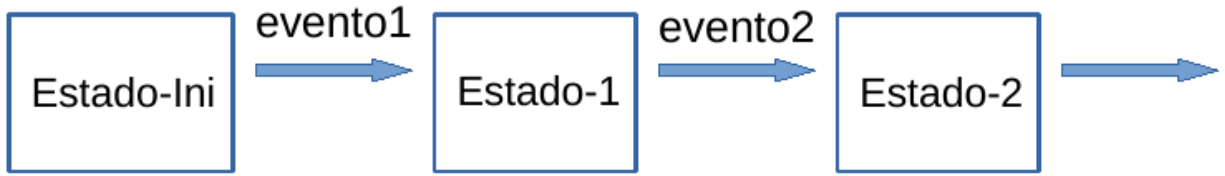
Ejemplo 1

Escribiremos un programa que reaccione ante paso del tiempo.

"El programa mostrará en una escena vacía un círculo de diámetro 50 pixeles de color rojo. El círculo cambiará de color con cada 3 segundos, pasando por los colores amarillo, verde y de nuevo rojo."

Estado

- Para diseñar un programa interactivo el primer paso es identificar el estado del programa.
- El estado de un programa es el conjunto de propiedades o valores que cambian cuando ocurre un evento.



- Luego, tendremos que pensar cómo representar el estado, dar el diseño de datos.
- Y definir el estado el inicial.

Diseño de datos

```
; El estado del sistema será una cadena
; que representa el color del círculo.
; Posibles valores: "red", "green" y "yellow"

; Estado inicial
(define INICIAL "red")
```

Función que interpreta el estado

```
; interpretar: Estado -> Image
; dado un estado, devuelve la imagen a mostrar

(define (interpretar e)
        (place-image (circle 50 "solid" s)
                     150
                     150
                     (empty-scene 300 300)))
```

Expresión big-bang

```
(big-bang   INICIAL ; estado inicial
            [to-draw interpretar] ; evento obligatorio
            [on-tick cambiar-color 3])
```

Manejador de on-tick

```
; cambiar-color: Estado -> Estado
; dado un color, devuelve el siguiente color del círculo

(check-expect (cambiar-color "red") "yellow")
(check-expect (cambiar-color "yellow") "green")
(check-expect (cambiar-color "green") "red")

(define (cambiar-color c)
  (cond [(string=? c "red") "yellow"]
        [(string=? c "yellow") "green"]
        [(string=? c "green") "red"]))
```

Ejercicio 1

Modificar el ejemplo anterior para que el círculo sea siempre de color azul y cambie su tamaño a medida que pase el tiempo. El radio del círculo será inicialmente de 20 píxeles, aumentará 5 píxeles cada medio segundo y volverá al estado inicial cuando su radio llegue a 150 píxeles.

Más eventos

- Cuando ocurre un evento el programa interactivo responde al mismo evaluando la función que maneja el evento.
- Un manejador de evento es un transformador de estado.

Evento	Manejador de evento
to-draw	interpretar
on-tick	manejador-on-tick
on-key	manejador-on-key
on-mouse	manejador-on-mouse

- Los manejadores de los eventos on-key y on-mouse reciben otra información además del estado.

La expresión big-bang

Para asociar un evento con su manejador usamos expresiones big-bang:

```
(big-bang   INICIAL ; estado inicial
            [to-draw interpretar] ; evento obligatorio
            [evento1 manjedor-e1]
            ...
            [eventoN manjedor-eN])
```

Ejemplo: Agregamos el evento on-key

Podremos modificar el color del círculo a través de algunas teclas del teclado:

- "r" $\rightarrow$ "red"
- "y" $\rightarrow$ "yellow"
- "g" $\rightarrow$ "green"
- "left" (flecha izq) $\rightarrow$ color anterior
- "right" (flecha derecha) $\rightarrow$ color siguiente

Manejador de on-key

```
; manejador-on-key: Estado String → Estado
; controla los eventos del teclado de acuerdo a :
; "r" → "red" , "y" → "yellow" , "g" → "green"
; "left" → color anterior, "right" → color siguiente

(check-expect (manejador-on-key "red" "g") "green")
(check-expect (manejador-on-key INICIAL "r") "red")
(check-expect (manejador-on-key "left" "yellow") "red")
```

Manejador de on-key

```
(define (manejador-teclado e k)
   (cond  [(string=? k "y") "yellow"]
          [(string=? k "g") "green"]
          [(string=? k "r") "red"]
          [(string=? k "left") (anterior e)]
          [(string=? k "right") (siguiente e)]
          [else e]
```

Modificamos expresión big-bang

```
(big-bang   INICIAL ; estado inicial
            [to-draw interpretar] ; evento obligatorio
            [on-tick cambiar-color 3])
            [on-key manejador-on-key]
```

Ejercicio 2

Modificar el ejemplo del ejercicio 1 para que el programa reaccione al apretar las teclas "up" y "down", aumentando o disminuyendo respectivamente el radio del círculo 10 píxeles.

Agregamos el evento on-mouse

Podremos modificar el radio del círculo del Ejercicio 2 usando el mouse de la siguiente manera: cuando se precione el botón del mouse el nuevo radio del círculo será la distancia del punto donde ocurrió el evento y el origen del círculo.

Ejemplo: Manejador de on-mouse

```
; manejador-on-mouse: Estado Number Number String → Estado

; controla los eventos del mouse de acuerdo a :
; si se precionó el botón del mouse el estado devuelto es la
; distancia del punto donde ocurrió el evento y el origen del círculo

(check-expect (manejador-on-mouse 100 280 250 "button-down") 30)
(check-expect (manejador-on-mouse 100 280 250 "drag") 100)
```

Ejemplo: Manejador de on-mouse

```
(define (manejador-on-mouse e x y b)
  (if   (string=? b "button-down")
        (dist x y 250 250)
        e))
```