Capítulo 12

Recibir entrada del usuario

Si consideramos *Interacción* como *comunicación*, es evidente que debe ir en las dos direcciones: mensajes de xLogo hacia el ususario y respuesta de este, bien por teclado o con el ratón.

12.1. Comunicación con el usuario

Ya conocemos varias primitivas que permiten escribir mensajes, bien en el Histórico de comandos, bien en pantalla o incluso con una ventana emergente:

Primitivas	Argumentos	Uso
`escribe`	`cualquiera`	Escribe en el Histórico de comandos el argumento.
`tipea`	`cualquiera`	Idéntico a `escribe`, pero el cursor queda en la linea donde se mostró el contenido del argumento.
`rotula`	`palabra o lista`	Dibuja la palabra o lista especificada, en la posición actual, y en la dirección que está mirando.
`largoetiqueta`	`lista`	Devuelve, en píxels, la longitud en pantalla de la lista.
`mensaje`, `msj`	`lista`	Muestra una caja de diálogo con el mensaje que está en la lista. El programa se detiene hasta que el usuario hace un *clic* en el botón "Aceptar"

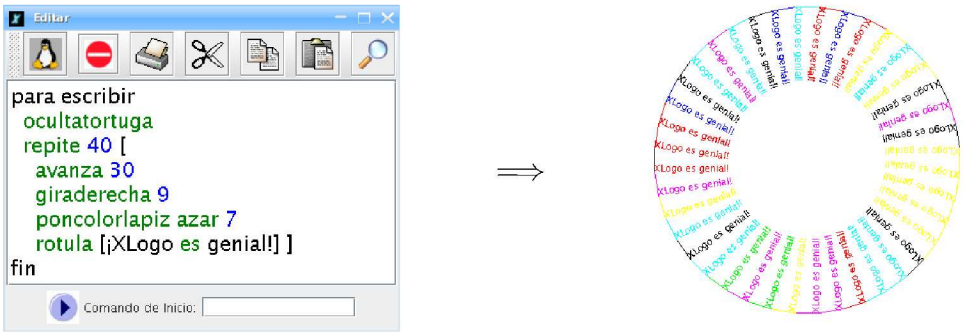
Llevamos ya tiempo trabajando con `escribe`, y utilizamos `tipea` al hablar de las operaciones (sección 7.1). Por su parte, `mensaje` apareció en el capítulo 3, en el programa `juego.lgo` para indicarnos si habíamos chocado con una "piedra" o llegado al "lago".

La primitiva `largoetiqueta` permite saber, entre otras muchas posibilidades, si al escribir en pantalla con `rotula` tienes suficiente espacio.

Ejemplo:

`largoetiqueta [Hola, ¿cómo estás?]` devuelve, en píxels la longitud en pantalla de la frase *Hola, ¿cómo estás?*

Ejemplo:



12.1.1. Ejercicios

1. Escribe tu nombre centrado en la pantalla
2. Plantea un procedimiento que recibe una lista como argumento, siendo esa lista el nombre y primer apellido de una persona. Con ello:
   a) Determina el número de letras del nombre, n
   b) Dibuja un polígono de n lados, siendo el lado el nombre y apellido antes dado.
   c) Puedes intentar que cada lado sea de un color

   En mi caso, sería: `nombre.poligono [Álvaro Valdés]`, y el resultado:

o

3. Escribe un procedimiento que escriba en el Histórico de Comandos los números del 1 al 100, con diez en cada línea, es decir:

```
1 2 3 4 5 6 7 8 9 10
11 12 13 14 15 16 17 18 19 20
      ...
91 92 93 94 95 96 97 98 99 100
```

4. Recupera el procedimiento `raiz_con_prueba` de la sección 9.2, y haz que cuando el radicando sea negativo, muestre un mensaje avisando de ello en una ventana, en vez de en el Histórico de Comandos.

12.1.2. Propiedades del Histórico de Comandos

Esta tabla define las primitivas que permiten ajustar y preguntar las propiedades de texto del área del Histórico de Comandos, es decir, las primitivas que controlan el color y tamaño del texto en este área.

Sólo afectan a las primitivas `escribe` y `tipea`. Para `rotula` se describen en la sección siguiente.

Primitivas	Argumentos	Uso
`borratexto`, `bt`	`no`	Borra el **Área de comandos**, y el área del **Histórico de comandos**.
`ponfuentetexto`, `pft`	`número`	Define el tamaño de la tipografía del área del **Histórico de comandos**. Sólo disponible para ser usada por la primitiva `escribe`.
`poncolortexto`, `pctexto`	`número o lista`	Define el color de la tipografía del área del **Histórico de comandos**. Sólo disponible para ser usada por la primitiva `escribe`.
`ponnombrefuentetexto`, `pnft`	`número`	Selecciona la tipografía número `n` para escribir en el área del **Histórico de comandos** con la primitiva `escribe`. Puedes encontrar la relación entre fuente y número en el menú **Herramientas → Preferencias → Fuente.**
`ponestilo`, `pest`	`lista o palabra`	Define los efectos de fuente para los comandos en el **Histórico de comandos**.
`fuentetexto`, `ftexto`	`no`	Devuelve el tamaño de la tipografía usada por la primitiva `escribe`.
`colortexto`	`no`	Devuelve el color de la tipografía usada por la primitiva `escribe` en el área del **Histórico de comandos**.
`nombrefuentetexto`, `nft`	`no`	Devuelve una lista con dos elementos. El primero es un número correspondiente a la fuente utilizada para escribir en el área del **Histórico de comandos** con la primitiva `escribe`. El segundo elemento es una lista que contiene el nombre de la fuente.

Primitivas	Argumentos	Uso
`estilo`,	`no`	Devuelve una lista que contiene todos los efectos de fuente utilizados por las primitivas `escribe` y `tipea`.

Puedes elegir entre siete estilos:

- `ninguno`, utiliza la fuente sin ningún cambio
- `negrita`
- `cursiva`
- `tachado`
- `subrayado`
- `superindice`
- `subindice`

Si se quiere aplicar varios estilos a la vez, deben escribirse en una lista.

Ejemplos de estilos de fuente:

```
ponestilo [negrita subrayado] escribe "Hola
```

Devuelve:

Hola

```
pest "tachado tipea [Tachado] pest "cursiva tipea "\ x
             pest "superindice escribe 2
```

Devuelve:

~~Tachado~~ x^2

Las primitivas `caracter`, (su forma corta es `car` y cuyo argumento es `n`: un número) y `unicode "a`, devuelven, respectivamente, el carácter unicode que corresponde al número `n` y el número unicode que corresponde al carácter `a`.

Ejemplo:
`unicode "A` devuelve `65`
`caracter 125` devuelve `}`

12.1.3. Ejercicios

1. Escribe programas que, dado un valor r, calculen:

   a) el perímertro de una circunferencia

   b) el área de un círculo

   c) el área de una esfera

   d) el volumen de una esfera

en todos los casos de radio r, indicando claramente sus unidades y evitando que lea valores negativos:

`perim.circ 5`

`El perímetro de una circunferencia de radio 5 es 3.141592 m`^{2}

2. Escribe un procedimiento con dos argumentos: `peso` y `altura`. y que calcule su índice de masa corporal (`i.m.c.`) mostrando cómo se calculó. El i.m.c. se calcula con la fórmula

$$\text{i.m.c.} = \frac{\text{peso}}{\text{altura}^2}$$

`i.m.c. 78 173`

`Tu i.m.c. es 78 / 173`^{2} `= 26.061679307694877`

Un i.m.c. muy alto indica obesidad. Los valores "normales" de i.m.c. están entre 20 y 25, pero esos límites dependen de la edad, del sexo, de la constitución física, etc.

3. Escribe dos procedimentos que lean:

   a) una temperatura en grados Celsius y la devuelvan en Fahrenheit

   b) una temperatura en grados Fahrenheit y la devuelvan en Celsius

   La relación entre grados Celsius (°C) y grados Fahrenheit (°F) es:

$$°\text{F} - 32 = \frac{9}{5} \cdot (°\text{C})$$

   `C.a.F 35`

   `35.0 °C son 95.0 °F`

   OJO: se trata de obtener el "circulito" de "grados": °, no vale usar el símbolo "primero": º

4. Escribe un procedimiento que lea una fórmula química escrita directamente y la ponga en la forma habitual. Es decir, que convierta a los números en subíndices y deje las letras "normales":

   `fórmula.química H2SO4` $\longrightarrow$ $\mathrm{H_2SO_4}$

   Pista: Utiliza el valor unicode asociado a los números

5. ¿Cómo ampliarías el procedimiento anterior para que, además, calculara la masa molecular del compuesto escrito?

6. Escribe un procedimiento que lea una lista de varias palabras y que la reescriba en el Histórico de Comandos en Mayúsculas, distinguiendo las que ya empezaban por mayúsculas:

   `a.mayúsculas [Esto es un ejemplo]`

   `ESTO ES UN EJEMPLO`

12.1.4. Escritura en Pantalla

En la sección anterior controlamos la escritura en el Histórico de Comandos. Modifiquemos parámetros al escribir en el Área de Dibujo.

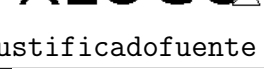
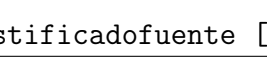
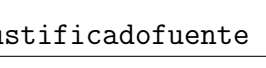
Primitiva	Forma larga	Forma corta
`ponnombrefuente`, `pnf`	`número`	Modifica la tipografía con la que se escribe. Para ver la lista completa de fuentes disponibles, Menú Herramientas ⟶ Opciones ⟶ Pestaña Fuente
`nombrefuente`, `nf`	`número`	Devuelve el tipo de la letra con que se rotula.
`ponfuente`, `pf`	`número`	Modifica el tamaño de la tipografía con que se rotula. Por defecto, el tamaño es 12.
`fuente`	`no`	Devuelve el tamaño de la letra con que se rotula.
`ponjustificadofuente`	`lista`	Indica cómo alinea el texto alrededor de la tortuga. La lista debe contener dos números: el alineamiento horizontal y el vertical.
`justificadofuente`	`no`	Devuelve la alineación del texto en pantalla.

Si queremos cambiar el color de las letras, debemos cambiar el color del lápiz.

Respecto a la alineación del texto, las opciones de la primitiva `ponjustificadofuente [a b]` proporcionan:

- Siendo `a` la alineación horizontal:
  - 0: A la izquierda.
  - 1: Centrado horizontalmente.
  - 2: A la derecha.
- Mientras, que `b`, la alineación vertical:
  - 0: En la zona inferior.
  - 1: Centrado verticalmente.
  - 2: En la zona superior.

Gráficamente, al teclear: `ponfuente 50 rotula "XLOGO`

 `ponjustificadofuente [2 0]`	 `ponjustificadofuente [1 0]`	 `ponjustificadofuente [0 0]`
 `ponjustificadofuente [2 1]`	 `ponjustificadofuente [1 1]`	 `ponjustificadofuente [0 1]`
 `ponjustificadofuente [2 2]`	 `ponjustificadofuente [1 2]`	 `ponjustificadofuente [0 2]`

12.1.5. Ejercicios

1. Observa la portada de este "libro". ¿Cómo harías para conseguir la "sombra" azulada de las letras que forman la palabra xLogo? (no hace falta que busques la misma fuente).

2. En la sección 10.2, creamos tres programas distintos para conjugar el futuro simple de un verbo regular. Modifica uno de ellos para que el resultado se muestre en el Área de Dibujo dispuesto. verticalmente

3. Haz lo mismo que en el ejercicio anterior, pero con el diseñado en 10.3 que conjuga presente y pasado, y añade una "etiqueta" encima de cada serie que indique el tiempo verbal.

12.2. Interactuar con el teclado

Durante la ejecución del programa, se puede recibir texto ingresado por el usuario a través de 3 primitivas: `tecla?`, `leecar` y `leelista`.

- `tecla?`: Da `cierto` o `falso` según se haya pulsado o no alguna tecla desde que se inició la ejecución del programa.

- `leecar` o `leetecla`:

  - Si `tecla?` es `falso`, el programa hace una pausa hasta que el usuario pulse alguna tecla.
  - Si `tecla?` es `cierto`, devuelve el valor unicode de la última tecla que haya sido pulsada.

Estos son los valores que dan ciertas teclas:

A → 65	B → 66	C → 67	...	Z → 90
◁ → -37 ó -226	△ → -38 ó -224	▷ → -39 ó -227	▽ → -40 ó -225	
`Esc` → 27	`F1` → -112	`F2` → -113	...	`F12` → -123
`SHIFT` → -16	`ESPACIO` → 32	`CTRL` → -17	`ENTER` → 10	

En Mac, las teclas `F1` a `F12` no devuelven valor alguno, ya que el sistema las reserva para tareas "propias".

Si tienes dudas acerca del valor que da alguna tecla, puedes probar con: `es leecar`. El intérprete esperará hasta que pulses una tecla, y escribirá su valor.

- `leelista [titulo] "palabra` o `leeteclado [titulo] "palabra`: Presenta una caja de diálogo titulada `titulo`. El usuario puede escribir un texto en el área de entrada, y esta respuesta se guardará seleccionando automáticamente si en forma de número o de lista en la variable `:palabra` cuando se haga *clic* en el botón `OK`.

  Ejemplos:

```
para edades
  leelista [¿Qué edad tienes?] "edad
  si :edad < 18 [escribe [Eres menor]]
  si :edad > 17 [escribe [Eres adulto]]
  si :edad > 69 [escribe [Con todo respeto!!]]
fin

para dibujar
# La tortuga es controlada con las flechas del teclado.
# Se termina con Esc.
si tecla?
  [ haz "valor leecar
    si :valor=-37 [giraizquierda 90]
    si :valor=-39 [giraderecha 90]
    si :valor=-38 [avanza 10]
    si :valor=-40 [retrocede 10]
    si :valor=27 [alto] ]
 dibujar
fin
```

12.3. Ejercicios

1. Escribe un procedimiento que haga que la tortuga:

- dibuje en pantalla el número que se pulsa en el teclado
- rotule en pantalla las letras que se pulsen en el teclado

quedando en espera para dibujar o rotular más números o letras a la derecha de los ya introducidos. Necesitarás:

a) Hacerlo recursivo, de modo que termine al pulsar `Esc`

b) Ir desplazando a la tortuga hacia la derecha a medida que son dibujados los números o rotuladas las letras.

c) OJO: Dibujados, no rotulados. Esto implica crear 10 procedimientos, uno por cada cifra.

No obstante, para ver si la parte que lee y desplaza a la tortuga está bien diseñada, puedes usar `rotula` para mostrar el resultado en el Área de Dibujo como con las letras.

d) Para cambiar el tamaño de las letras, usa `ponfuente` (sección 12.1.4)

2. Escribe un procedimiento que lea una lista de varias palabras y que la reescriba en el Histórico de Comandos del siguiente modo:

a) Si la palabra empieza por vocal, en cursiva (itálica)

b) Si es un número:

1) Como subíndice si es par
2) Como superíndice si es impar

c) Si la palabra empieza por consonante:

1) Si está entre `b` y `l`, tachada
2) Si está entre `m` y `z`, subrayada

d) Si, además, empieza por mayúscula, en negrita

Pista: Observa que usando `unicode`, las letras mayúsculas `A - Z` devuelven valores consecutivos entre 65 y 90, mientras que las minúsculas `a - z` lo hacen entre 97 y 122. Debes tener cuidado, eso sí, con los caracteres especiales del castellano: vocales acentuadas y "eñes".

3. Modifica el procedimiento `dibujar` para que suba o baje el lápiz al pulsar la barra espaciadora, borre al pulsar "borrar", ... y/o cualquier otra opción que se te ocurra (cambiar el color al pulsar la inicial del mismo, ...)

4. **Un pequeño juego**: Diseña un juego tal que el programa elija un número entre 0 y 32 aleatoriamente (recuerda la primitiva `azar` (página 61). A continuación, abra un cuadro de diálogo que pida al usuario que introduzca un número.

Si este número entero es igual al guardado, muestra "GANASTE"en la zona de texto. En caso contrario, el programa indica si el númer guardado en mayor o menor que el introducido por el usuario y vuelve a abrir el cuadro de diálogo.

El programa terminará cuando el usuario haya dado el número correcto.

UNA AYUDA:

- El número elegido por XLOGO se almacena en una variable llamada `numero`.
- El cuadro de diálogo se llamará "Dame un número, por favor"
- El número elegido por el usuario se almacena en una variable llamada `intento`.
- El procedimiento principal se llamará `juego`.

Algunas posibles mejoras:

- Mostrar el número de intentos.
- Que XLOGO elija un número entre 0 y 2000.
- Comprobar que el usuario introduce un número válido. (Recuerda la primitiva `número?`, página 75).

12.4. Interactuar con el ratón

Durante la ejecución del programa, se pueden recibir eventos del ratón a través de tres primitivas: `leeraton`, `raton?` y `posraton`.

- `leeraton`: el programa hace una pausa hasta que el usuario hace un *clic* o un movimiento. Entonces, da un número que representa ese evento. Los diferentes valores son:
  - 0 → El ratón se movió.
  - 1 → Se hizo un *clic* izquierdo.
  - 2 → Se hizo un *clic* central (Linux).
  - 3 → Se hizo un *clic* derecho.

  En Mac, sólo se reciben 0 y 1, ya que el ratón no tiene botón derecho, y la opción de mantener pulsado `Ctrl` no es interpretada por XLOGO como "botón derecho".
- `posraton`: Da una lista que contiene la posición actual del ratón.
- `raton?`: Devuelve `cierto` o `falso` según toquemos o no el ratón desde que comienza la ejecución del programa

Ejemplos:

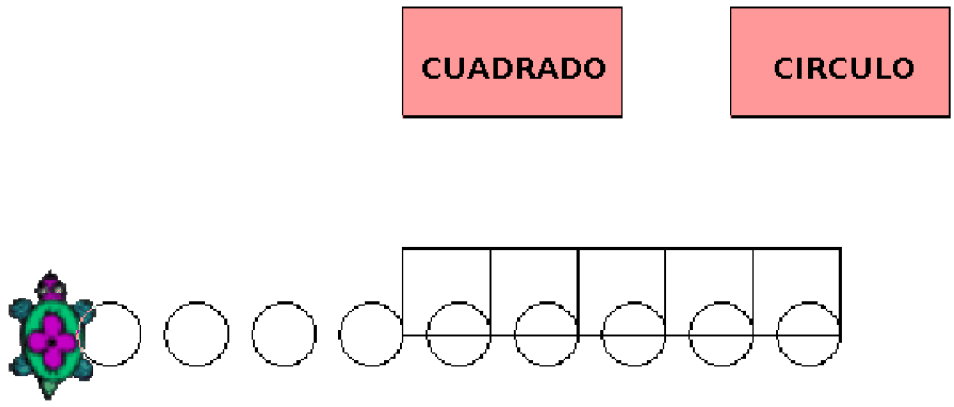
En este primer procedimiento, la tortuga sigue los movimientos del ratón por la pantalla.

```
para seguir
# cuando el raton se mueve, la tortuga cambia de posicion.
  si leeraton=0 [ponpos posraton]
  seguir
fin
```

Este segundo procedimiento es similar, pero hay que hacer *clic* izquierdo para que la tortuga se mueva.

```
para seguir2
  si leeraton = 1 [ponpos posraton]
  seguir2
fin
```

En este tercer ejemplo, hemos creado dos botones rosa. Si hacemos *clic* izquierdo, la tortuga dibuja un cuadrado de lado 40. Si hacemos *clic* derecho, la tortuga dibuja un pequeño círculo. Por último si hacemos *clic* derecho en el botón derecho, se detiene el programa.




```
para boton
# crea un boton rectangular color rosa, de 50 x 100.
  repite 2 [
   avanza 50 giraderecha 90 avanza 100 giraderecha 90 ]
  giraderecha 45 subelapiz avanza 10
```

```
   bajalapiz poncolorlapiz [255 153 153]
   rellena retrocede 10 giraizquierda 45 bajalapiz poncolorlapiz 0
fin

para empieza
  borrapantalla boton subelapiz ponpos [150 0] bajalapiz boton
  subelapiz ponpos [30 20] bajalapiz rotula "Cuadrado
  subelapiz ponpos [180 20] bajalapiz rotula "Circulo
  subelapiz ponpos [0 -100] bajalapiz
  raton
fin

para raton
# ponemos el valor de leeraton en la variable ev
# ponemos la primer coordenada en la variable x
# ponemos la segunda coordenada en la variable y
  haz "ev leeraton
  haz "x elemento 1 posraton
  haz "y elemento 2 posraton
# si hay clic izquierdo
  si :ev=1 & :x>0 & :x<100 & :y>0 & :y<50 [cuadrado]
# si hay clic derecho
  si :x>150 & :x<250 & :y>0 & :y<50 [
  si :ev=1 [circulo]
  si :ev=3 [alto] ]
  raton
fin

para circulo
  repite 90 [av 1 gi 4]
  giraizquierda 90 subelapiz avanza 40 giraderecha 90 bajalapiz
fin

para cuadrado
  repite 4 [avanza 40 giraderecha 90]
  giraderecha 90 avanza 40 giraizquierda 90
fin
```

12.5. Ejercicios

1. Diseña un procedimiento que dibuje un tablero de ajedrez, y determine el color de una casilla al hacer "*clic*" sobre ella.

2. Observa el funcionamiento del ejemplo anterior, y piensa cómo podrías utilizarlo para diseñar un procedimiento que cargue en el área de dibujo un mapa "mudo" (Deberás utilizar la primitiva `cargaimagen`, sección 13.6.2) y el alumno tenga que ir haciendo "*clic*" en un área según se le vaya mostrando el nombre de la misma (por ejemplo, Países, Comunidades Autónomas, Concejos, ...).

   Simplifica lo más posible el planteamiento, ya veremos más adelante cómo "pulirlo" para tener en cuenta la forma de las fronteras.

3. Plantea un procedimiento que haga que la tortuga "persiga al ratón". Para ello, la tortuga debe quedarse en un punto hasta que se hace *clic* en otro, hacia donde se dirigirá despacio después de haberse orientado hacia allí. Puedes ralentizar el movimiento con un procedimiento del tipo:

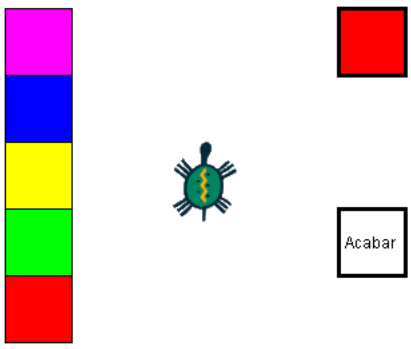
```
para avance.lento :distancia
  repite :distancia
    [ avanza 1 espera 10 ]
fin
```

   Explicaremos `espera` en el capítulo 18.

4. Plantea un procedimiento que:

   a) Dibuje varios cuadrados apilados y los rellene con distintos colores
   b) Dibuje otros dos, dejando uno en blanco y en el otro rotula "Acabar"
   c) Al hacer *clic* en uno de los cuadrados coloreados, el que está en blanco adquirirá el color de aquél.

      Necesitarás la primitiva `encuentracolor`, que se explica en el capítulo siguiente.
   d) Al hacer *clic* en el que dice "Acabar", el programa terminará



5. **Este problema implica conocimientos de Física**: Plantea un procedimiento que calcule el potencial electrostático y/o gravitatorio en el origen de coordenadas de un sistema de cargas o masas ubicadas haciendo *clic* con el ratón.

   Con cada "*clic*" del ratón, se debe dibujar un círculo centrado en ese punto.

12.6. Componentes Gráficos

Desde la versión 0.9.90, xLogo permite añadir componentes gráficos en el Área de dibujo (botones, menús, ...)

Las primitivas que permiten crear y modificar estos componentes terminan con el sufijo `igu` (Interfaz Gráfica de Usuario – *Graphical User Interface, gui* son sus siglas inglesas).

12.6.1. Crear un componente gráfico

La secuencia de pasos que debes seguir es: **Crear** → **Modificar** sus propiedades o características → **Mostrar**lo en el Área de dibujo.

Crear un Botón

Usaremos al primitiva `botonigu`, cuya sintaxis es:

```
# Esta primitiva crea un boton llamado b
# y cuya leyenda es: Clic
botonigu "b "Clic
```

Crear un Menú

Disponemos de la primitiva `menuigu`, cuya sintaxis es:

```
# Esta primitiva crea un menu llamado m
# y que contiene 3 opciones: opcion1, opcion2 y opcion3
menuigu "m [opcion1 opcion2 opcion3]
```

Modificar las propiedades del componente gráfico

`posicionigu` determina las coordenadas donde se colocará el elemento gráfico. Por ejemplo, para colocar el botón definido antes en el punto de coordenadas `(20 , 100)`, escribiremos:

```
posicionigu "b [20 100]
```

Si no se especifica la posición, el objeto será colocado por defecto en la esquina superior izquierda del Área de dibujo.

Eliminar un componente gráfico

La primitiva `eliminaigu` elimina un componente gráfico. Para eliminar el botón anterior

```
eliminaigu "b
```

Definir acciones asociadas a un componente gráfico

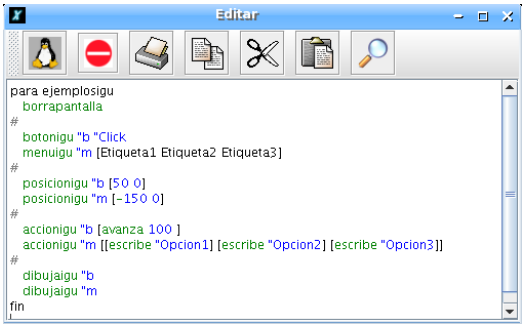
La primitiva `accionigu`, define una acción asociada al componente, y que se realizará cuando el usuario interactúa con él.

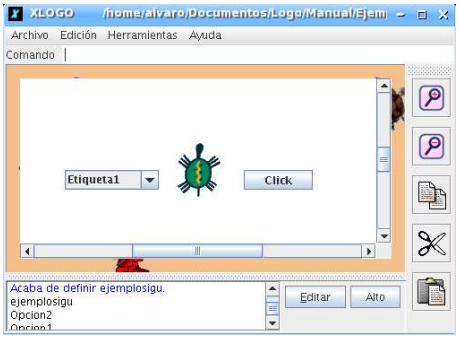
```
# Que la tortuga avance 100 al pulsar el boton "b
accionigu "b [avanza 100 ]
# En el menu, cada opcion indica su accion
accionigu "m [[escribe "Opcion1] [escribe "Opcion2] [escribe "Opcion3]]
```

Dibujar el componente gráfico

La primitiva `dibujaigu`, muestra el componente gráfico en el Área de dibujo. Para mostrar el botón que estamos usando como ejemplo:

```
dibujaigu "b
```



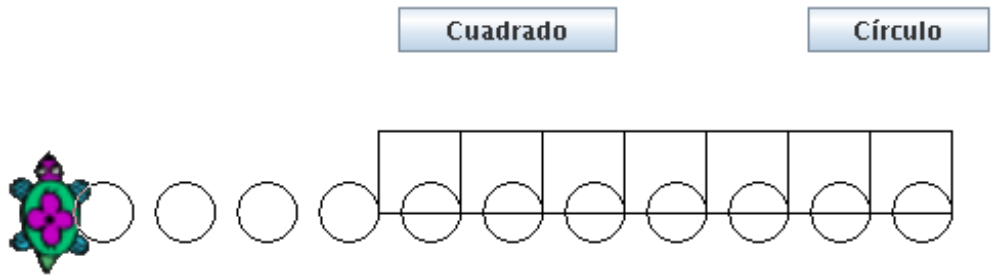


Cambiemos el ejemplo anterior utilizando las nuevas primitivas:

```
para empieza
  botonigu "Boton_Circ "Circulo
  botonigu "Boton_Cuad "Cuadrado
  posicionigu "Boton_Circ   [50 100]
  posicionigu "Boton_Cuad [-150 100]
  accionigu "Boton_Circ [ circunferencia ]
  accionigu "Boton_Cuad [ cuadrados ]
  dibujaigu "Boton_Circ
  dibujaigu "Boton_Cuad
fin

para circunferencia
  repite 90 [av 1 gi 4]
  giraizquierda 90 subelapiz avanza 40 giraderecha 90 bajalapiz
fin
```

```
para cuadrado
  repite 4 [avanza 40 giraderecha 90]
  giraderecha 90 avanza 40 giraizquierda 90
fin
```



12.7. Ejercicios

1. a) Crea cuatro botones: `Av`, `Re`, `Gd` y `Gi`, cuyas etiquetas sean, respectivamente: `Avanzar`, `Retroceder`, `Girar Derecha` y `Girar Izquierda`

   b) Ubica los botones anteriores en las posiciones `[-350 25]`, `[-360 -25]`, `[-300 0]` y `[-440 0]`

   c) Asígnales las acciones: `avanza 10`, `retrocede 10`, `giraderecha 90` y `giraizquierda 90`

   d) Dibuja los cuatro botones en pantalla y comprueba que "funcionan" correctamente.

   e) Crea dos menús `Cl` y `Gl`, cuyas opciones sean, respectivamente: `[Color Lápiz Azul Rojo Amarillo Verde Negro Blanco]` y `[Grosor Lápiz 1 2 3 4 5]`

   f) Asígnales las acciones:

      1) `[ ] [poncolorlapiz azul] [poncolorlapiz rojo] [poncolorlapiz amarillo] [poncolorlapiz verde] [poncolorlapiz negro] [poncolorlapiz blanco]`
      2) `[ ] [pongrosor 1] [pongrosor 2] [pongrosor 3] [pongrosor 4] [pongrosor 5]`

      Observa que, en ambos casos, la primera opción NO hace nada

   g) Ubícalos en las posiciones `[-350 100]` y `[-350 70]`

   h) Dibuja los dos menús en pantalla y comprueba que "funcionan" correctamente.

   Compara el resultado de los ejercicios con el xLogo con que iniciamos el curso. ¿Serías capaz de diseñar, asignar y dibujar los elementos que faltan?

2. Crea un programa sobre conjugación de verbos que:

   a) Lea un verbo con `leelista`, analizando previamente si ya ha sido definido uno o no

   b) Conjugue correctamente en presente, pasado y futuro simple (sólo verbos regulares)

   c) El tiempo verbal se elija con un menú contextual

   d) Rotule en pantalla el tiempo verbal elegido

   e) Rotule `¿Acertaste?` tras mostrar el timepo verbal

   f) Se pueda cambiar el verbo con un botón