Los siguientes ejercicios muestran cómo utilizar un objeto Timer
para efectuar animaciones sencillas.

Ejercicios con Timer

1. Descarga e importa el proyecto Timer Contador AL. Para ver el resultado del programa puedes descargar del aula virtual (en el apartado *Ver resultados con Timer*) y ejecutar (haciendo doble *click)* el fichero *contador.jar.*

Completa el panel.

Cada vez que se dibuja el panel (se invoca indirectamente a paintComponent()) hay que:

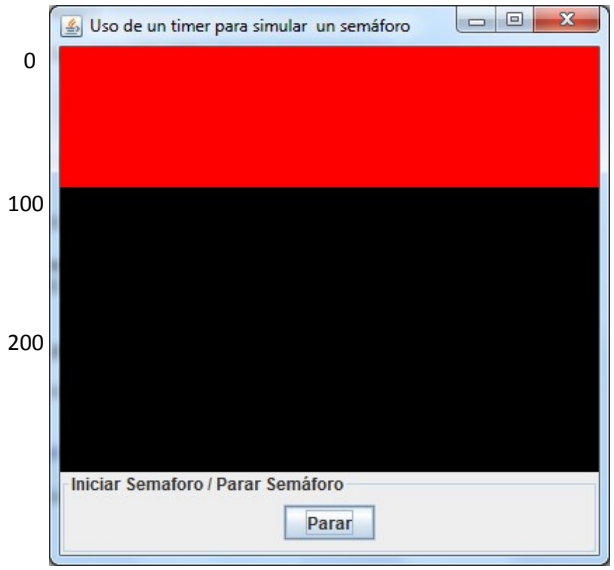
- mostrar el texto “Click para iniciar/continuar” o “Click para parar” dependiendo de si el *timer* está o no funcionando en la posición (20, 10)
- mostrar el valor del contador (elige tipo de letra y color)

Hay que utilizar un *timer* con el retardo indicado por la constante DELAY. El oyente de eventos del *timer* será el propio panel.
El *timer* se inicia/para haciendo *click* con el ratón sobre el panel. El oyente de eventos de ratón será el panel.
Una vez que el *timer* se ha iniciado el contador va incrementando su valor hasta llegar a 10. En este momento empieza a decrementarse hasta llegar a 0 y así sucesivamente.

Cuando termines:

- crea un *jar* para este proyecto con el nombre *ej1contador.jar*
- comprueba que funciona haciendo doble click
- ejecútalo también desde línea de comandos

2. Importa a Eclipse el proyecto Timer Semáforo AL y complétalo.
semaforo.jar muestra el resultado que has de obtener.



Completa en la clase Semaforo el método avanzar() que cambia el color del semáforo.

Completa la clase GuiSemaforo. Esta clase incluye una clase interna que representa el panel sobre el que se dibujará el semáforo. Un panel interno tiene como ventaja poder acceder directamente a los atributos de la clase GuiSemaforo (en nuestro caso al atributo *semaforo*). Dentro del panel tendrás que definir un *timer* con un retardo de 400 msg. El oyente de eventos del *timer* será el panel. El semáforo cambia de color cada vez que el *timer* genera un evento.
El *timer* se inicia/para con los botones. El oyente para estos botones también es el panel.

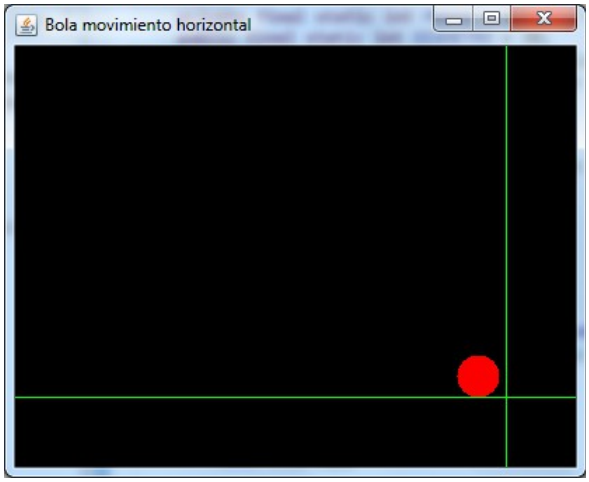
Cada vez que se dibuja el panel habrá que pintar un rectángulo del color que tenga el semáforo en la franja adecuada (rojo arriba, amarillo en medio, verde abajo).

Cuando termines el proyecto expórtalo como *ej2semaforo.jar.* Haz doble *click* para comprobar que funciona.

Nota

- para dibujar un rectángulo: g.fillRect(x, y, ancho, alto)

3. Importa a Eclipse el proyecto Timer Bola horizontal AL y complétalo. *bolahorizontal.jar* muestra el resultado que has de obtener.



El *timer* se inicia al crearse el panel.
La dirección inicial de la bola es hacia la derecha.
La primera vez que se dibuja el panel se muestran las líneas de color verde y la bola roja en sus posiciones iniciales determinadas por las constantes que hay definidas.

En cada evento del *timer* se actualizarán las coordenadas de la bola adecuadamente teniendo en cuenta que:

- si va hacia la derecha la coordenada *x* se incrementa en 5. El tope lo marca la línea vertical. Si la bola toca esta línea cambiará de dirección
- si va hacia la izquierda *x* se decrementa en 5. El tope lo marca el borde izquierdo del panel. Si la bola lo toca cambia otra vez de dirección
- **pista** – para saber la dirección de la bola utiliza una variable booleana *direccion*

4. Importa a Eclipse el proyecto Timer Animacion Personaje AL y complétalo. *personajeanimadotimer.jar* muestra el resultado que has de obtener.

Se trata de mostrar una secuencia de imagénes a intervalos regulares de tiempo al mismo tiempo que se va actualizando su posición en el panel. Esto dará la impresión de movimiento.

Completa el proyecto teniendo en cuenta:

- en el constructor del panel
  - crea el *timer*
  - crea el array que contendrá las imágenes, los *frames* (es un array de objetos de tipo ImageIcon)

- para asignar a cada elemento del array la imagen correspondiente crearemos un objeto ImageIcon como
  - `URL imagenUrl = this.getClass().getResource(nombre);`
  - `ImageIcon img = new ImageIcon(imagenUrl);`
    donde *nombre* es la ruta relativa del fichero que contiene la imagen
  - utilizamos `imagenUrl` de esta manera para poder crear un fichero *jar* con imágenes y que éstas se localicen correctamente dentro de él (se está solicitando al cargador de clases - *class loader* - la localización de los ficheros de imagen)

- el oyente de eventos es el panel tanto para los eventos del *timer* como para los de los botones
- cada vez que el *timer* genera un evento se actualiza la imagen a mostrar y la posición x en la que se muestra en el panel (el incremento de x será de 5)
  - cuando la imagen sale del panel por la derecha vuelve a aparecer por la izquierda
- en el método paintComponent() únicamente habrá que dibujar la imagen (ImageIcon) que toque
  - imagenes[imagenActual].paintIcon(this, g, x, y);
    donde *g* es el objeto de tipo Graphics, x, y la posición en el panel en la que se dibujará la imagen y *this* un componente que actúa como observador de la imagen
- click en el botón *iniciar* inicia la animación (si no está ya iniciada)
- click en el botón *parar* para la animación (si no está ya parada)