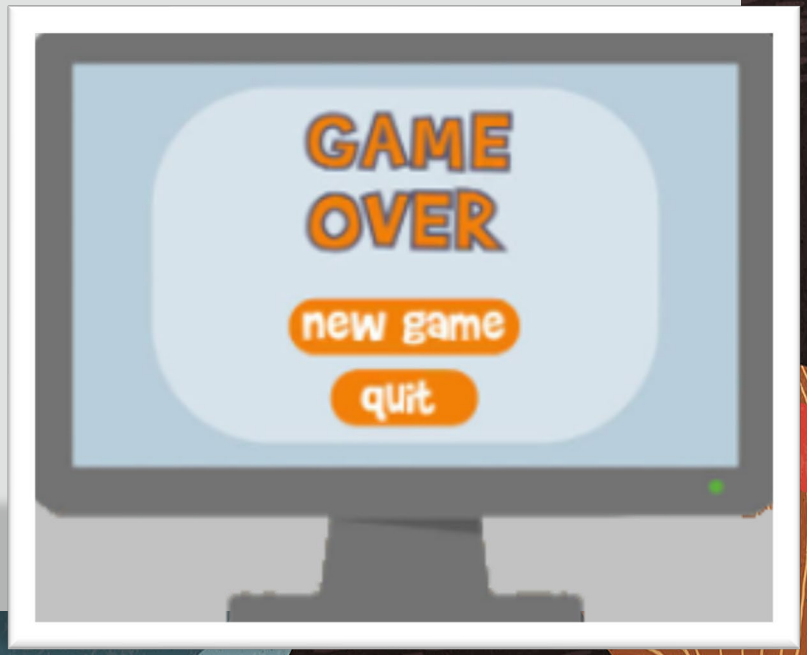

ORACLE
Academy

Creación de programas Java con Greenfoot

Lección 10

Final del juego



ORACLE
Academy

Objetivos

- En esta lección se abordan los siguientes objetivos:
  - Escribir sentencias de programación para terminar un juego

Final del juego

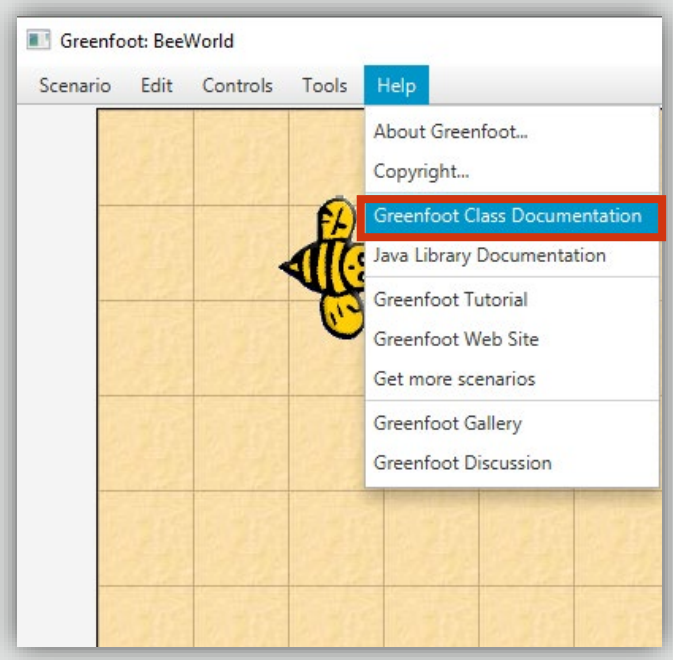
- La clase Greenfoot tiene un método stop() que puede utilizar para finalizar su partida en el punto que designe
- Es posible que desee finalizar la partida cuando:
  - El jugador alcance un objetivo
  - Se agota el tiempo en el reloj
  - La instancia toca una determinada coordenada u objeto

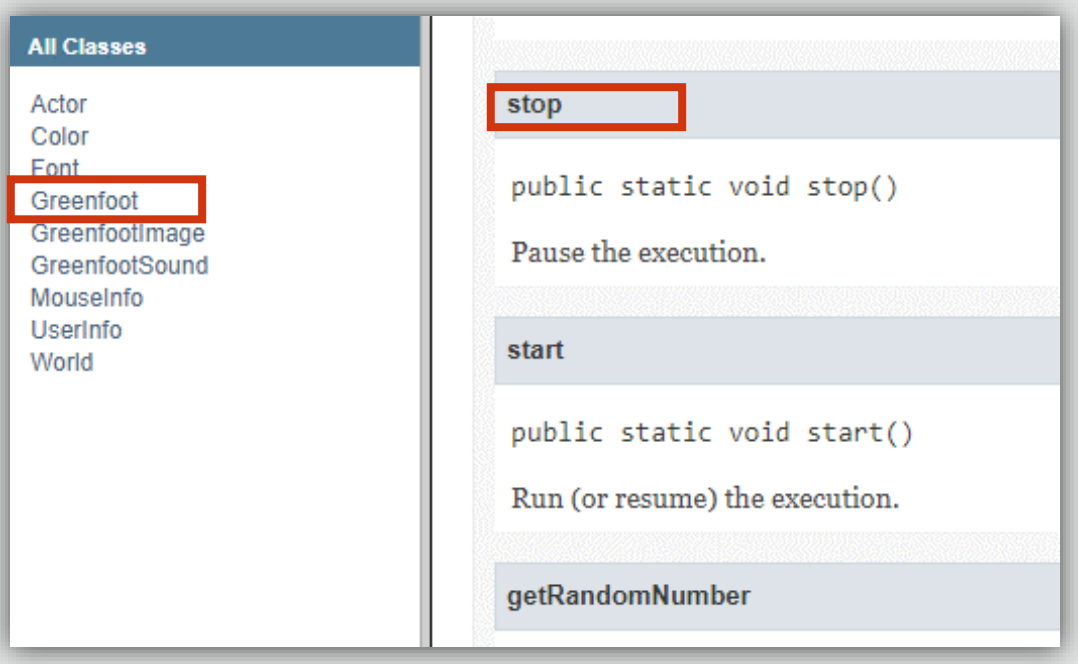
Ejemplo de juego Bee

- Ejemplo de juego:
  - El jugador decide cuántas veces el objeto araña atrapa a la abeja al final del juego
  - Cuando termina el juego, suena una voz indicando que ha finalizado el juego
- Especificaciones del juego:
  - Cree e inicialice variables para almacenar vidas y la puntuación
  - Proporcione un recuento del total de moscas atrapadas (puntuación)
  - Introduzca el método stop() para detener el juego cuando al jugador le queden 0 vidas

Buscar el método stop() en la API de Greenfoot

- Vaya al menú Help y seleccione Greenfoot Class Documentation
- Busque el método stop() en la API de Greenfoot





Escribir el método stop() en el código fuente

- En el momento en el que el juego deba terminar, escriba el método en el código fuente del siguiente modo
- La notación de puntos se utiliza para llamar al método

```
private void endGame(){
    Greenfoot.stop();
}//end method endGame
```

Ejemplo de asignación de variables a instancias

- La abeja debe atrapar un número de objetos mosca para aumentar la puntuación
- La abeja también perderá una vida si la araña la atrapa
- Las variables se definen antes de los constructores y los métodos
- El constructor Bee asigna las variables a las instancias que produce

```
public class Bee extends Actor

{
    private GreenfootImage image1;
    private GreenfootImage image2;
    private int score;
    private int lives;

    /**
     * Bee - sets the initial values of the bee
     */
    public Bee()
    {
        image1 = new GreenfootImage("bee.png");
        image2 = new GreenfootImage("bee2.png");
        setImage(image1);
        score = 0;
        lives = 3;
    }//end constructor
```

Ejemplo del método definido catchFly()

- El método definido catchFly() se escribe a continuación del método act() para indicar a la abeja que atrape a las moscas
- Agregaremos uno a la variable de puntuación para cada mosca que se coma la abeja

```
/**
 * catchFly - if the Bee touches a fly the fly is removed
 * A sound is played and a new fly is added to the game
 */
private void catchFly(){
        if(isTouching(Fly.class)){
            removeTouching(Fly.class);
            Greenfoot.playSound("slurp.wav");
            score++;
            getWorld().addObject(new Fly(), Greenfoot.getRandomNumber(getWorld().getWidth()),
                                            Greenfoot.getRandomNumber(getWorld().getHeight()));
         }//endif
}
```

Ejemplo 2 de asignación de variables a instancias

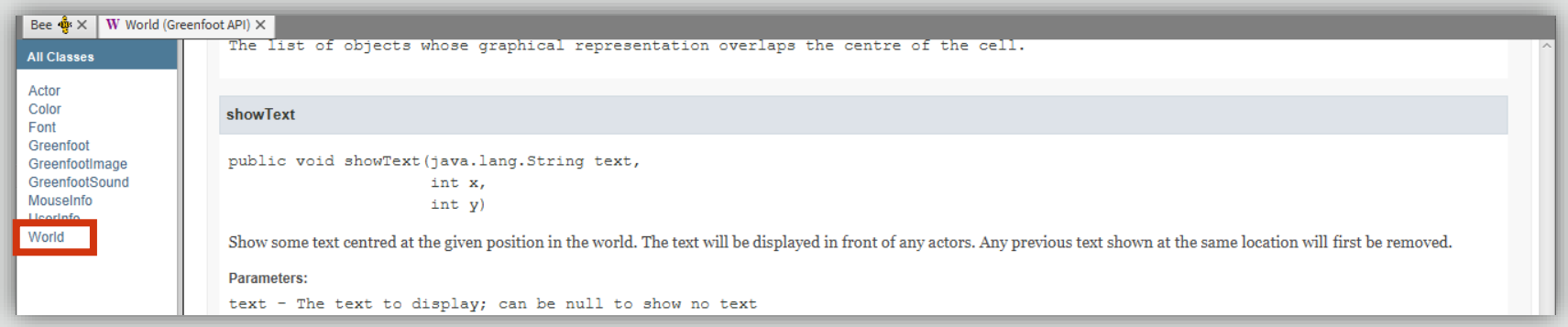
- Si la abeja entra en contacto con la araña, perderá una vida
- También volveremos a colocar la abeja en la parte superior izquierda
- Ampliaremos la clase Bee agregando un nuevo método (caughtBySpider()) y agregándole una llamada en el método act()
- A continuación, probaremos si el usuario se ha quedado sin vidas y se detendrá el juego

```
private void caughtBySpider(){
    if(isTouching(Spider.class)){
        setLocation(20,20);
        lives--;
        if(lives<0){
            endGame();
        }//endif
    }//endif
}//end method caughtBySpider

private void endGame(){
    Greenfoot.stop();
}//end method endGame
```

Mostrar texto

- A veces deseamos mantener informado al usuario de una aplicación sobre aspectos particulares de su interacción, como las vidas que posee, su puntuación o las cartas que le quedan
- De nuevo, Greenfoot tiene diversas formas de lograrlo
- La forma más sencilla consiste en utilizar el método World: showText()



Actualizar método catchfly()

- Vamos a agregar código al método catchFly() para incrementar el campo de puntuación y, a continuación, se mostrará el resultado en la pantalla
- También hemos movido la puntuación de actualización a su propio método y lo hemos llamado en catchFly()

```
private void catchFly(){
        if(isTouching(Fly.class)){
            removeTouching(Fly.class);
            Greenfoot.playSound("slurp.wav");
            updateScore();
            getWorld().addObject(new Fly(), Greenfoot.getRandomNumber(getWorld().getWidth()),
                                            Greenfoot.getRandomNumber(getWorld().getHeight()));
         }//endif
}

private void updateScore(){
    score++;
    getWorld().showText("Score : " + score, 60, 390);
}//end method updateScore
```

Inténtelo.

Creación de un constructor de la subclase Actor

Esta actividad necesita que comience con el archivo de proyecto que se ha guardado en el tema anterior FrogFly_L9T14.

FrogFly_L9T14.zip

Instrucciones:

Abra el editor de códigos de la clase Frog (Rana) y cree un constructor Frog.

En el cuerpo del constructor Frog, asigne a cada variable de imagen para su respectiva imagen image1(izquierda), image2(derecha), image3(arriba), image4(abajo).

Escriba el código para definir la primera imagen que la rana mostrará como image2.

En el método act(), sustituya el código dentro de los parámetros setImage por el nombre de la variable que contiene la imagen que desea mostrar cuando se presione la tecla correspondiente.

Inténtelo.

Creación de un constructor de la subclase Actor

Instrucciones (continuación):

Compile el código. Ejecute el escenario y compruebe que funcione. Asegúrese de que la rana comience mirando hacia la derecha y que luego cambie la imagen al presionar las teclas de flecha izquierda, derecha, arriba y abajo.

Guarde el escenario con el nombre FrogFly_L10T15.

Inténtelo.

Final del juego

Esta actividad necesita que comience con el archivo de proyecto que se ha guardado en el tema anterior FrogFly_L10T15.

FrogFly_L10T15.zip

Instrucciones:

Abra el editor de códigos de la clase Frog.

Declare una variable llamada countFliesEaten para contener el número de objetos Fly que come el objeto Frog.

Declare una variable llamada countAntCollisions que contenga el número de veces que la rana se choca con el objeto Ant (Hormiga).

Declare una variable llamada countRockCollisions que contenga el número de veces que la rana se choca con el objeto Rock (Roca).

Inténtelo.

Final del juego

Instrucciones (continuación):

En el método eatFly() (el método que eliminará un mosca del mundo si choca con una rana), se incluye un contador para incrementar la variable countFliesEaten.

Defina un método antCollision() que hará que la rana pase a una posición estática en el mundo si se choca con la hormiga. El método debe incluir un contador para incrementar la variable countFliesEaten.

Defina un método rockCollision()) que hará que la rana pase a una posición estática en el mundo si se choca con una roca. El método debe incluir un contador para incrementar la variable countRockCollisions.

Inténtelo.

Final del juego

Instrucciones (continuación):

Defina un método scoreKeeper() para controlar el final del juego.

- Si la rana come 5 moscas, el jugador gana el juego. La imagen de la rana debe cambiar a una imagen que usted haya creado que debe decir "¡Ha ganado!" y el escenario debe terminar.
- Si la rana se choca con la hormiga 5 veces, el jugador pierde el juego. La imagen de la rana debe cambiar a una imagen que usted haya creado que debe decir "¡Ha perdido!" y el escenario debe terminar.
- Si la rana se choca con una roca 5 veces, el jugador pierde el juego. La imagen de la rana debe cambiar a una imagen que usted haya creado que debe decir "¡Ha perdido!" y el escenario debe terminar.

Inténtelo.

Final del juego

Instrucciones (continuación):

Nota: Deberá crear las imágenes correspondientes a ¡Ha ganado! y ¡Ha perdido! con el programa de dibujo o diseño gráfico de su equipo y luego guardar la imagen en la carpeta de imágenes de su escenario.

Compile el escenario. Ejecute el escenario y compruebe que funcione. Gane y pierda el juego para poder probar las diferentes funciones que deben activarse.

Guarde el escenario con el nombre FrogFly_L10T16.

Summary

- En esta lección, debe haber aprendido lo siguiente:
  - Escribir sentencias de programación para terminar un juego


ORACLE
Academy