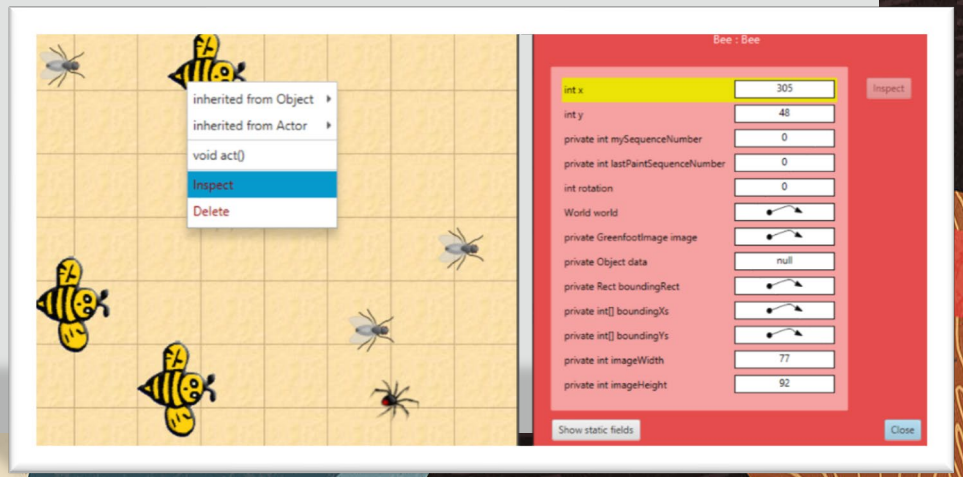

ORACLE
Academy

Creación de programas Java con Greenfoot

Lección 3

Uso de métodos, variables y parámetros



ORACLE
Academy

Visión general

- En esta lección se abordan los siguientes temas:
  - Definición de parámetros y su uso en los métodos
  - Comprensión de la herencia
  - Descripción de las propiedades de un objeto
  - Análisis del objetivo de una variable
  - Descripción de conceptos de programación y definición de la terminología

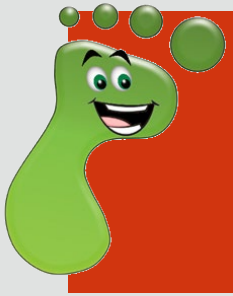
Ejemplo de métodos

- Para terminar una tarea, por ejemplo, de matemáticas, existen varias subtareas:
  - El estudiante termina la tarea de matemáticas
  - El estudiante va al colegio
  - El estudiante entrega la tarea a su profesor

- Gracias a las experiencias aprendidas en el colegio, así como a capacidades preprogramadas (como el pensamiento), el estudiante es capaz de completar esta tarea

Métodos

- En programación, cada objeto cuenta con una serie de operaciones (o tareas) que se puede llevar a cabo
- Los programadores escriben un programa para indicarle a un objeto cómo y cuándo realizar tareas, por ejemplo:
  - Ordenarle a un objeto que lleve a cabo una acción
  - Hacerle una pregunta a un objeto para obtener más información sobre lo que hace

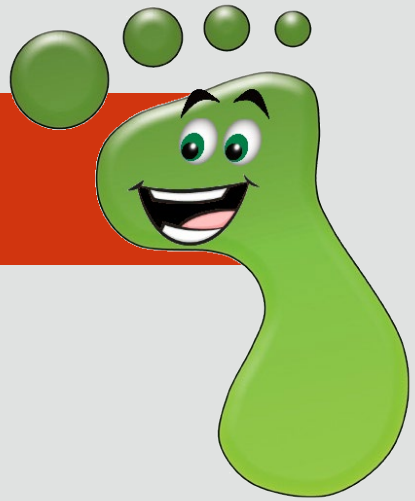


Los métodos son un conjunto de operaciones o tareas que pueden llevar a cabo las instancias de una clase. Cuando se llama a un método, este llevará a cabo la operación o la tarea especificada en el código fuente.

Herencia

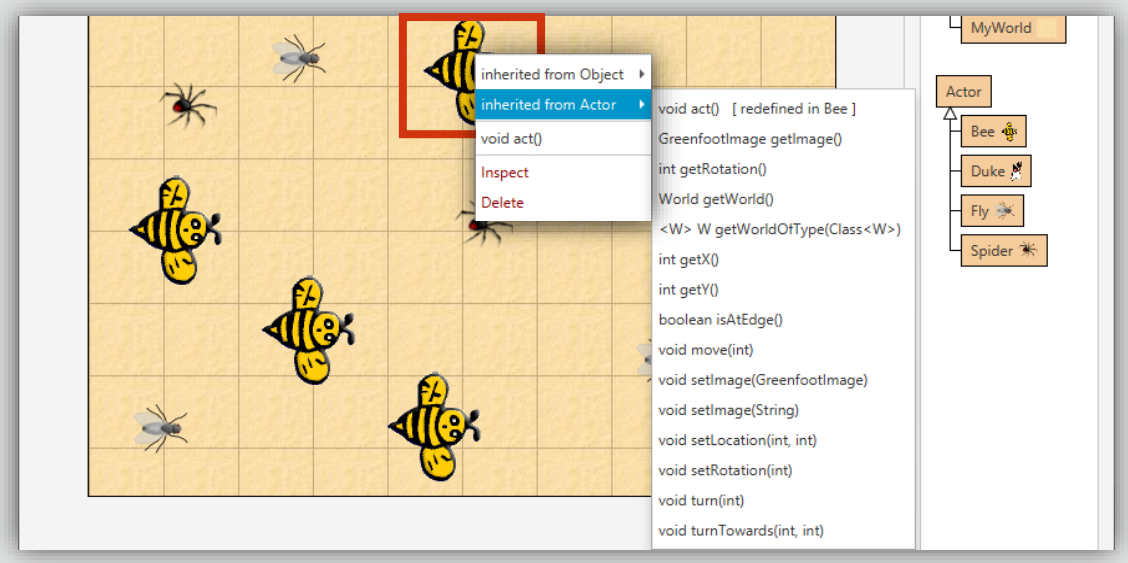
- Los objetos de Greenfoot heredan los métodos y propiedades de su clase y superclase
- Por ejemplo, una instancia Alligator (Caimán) heredaría los métodos de la superclase Actor y de la clase Alligator

La herencia significa que cada subclase hereda los métodos de su superclase.



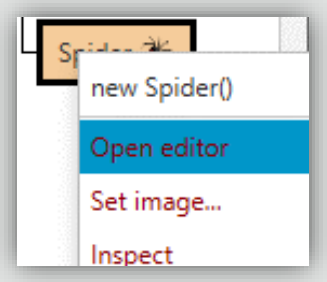
Visualización de métodos heredados en el menú Object

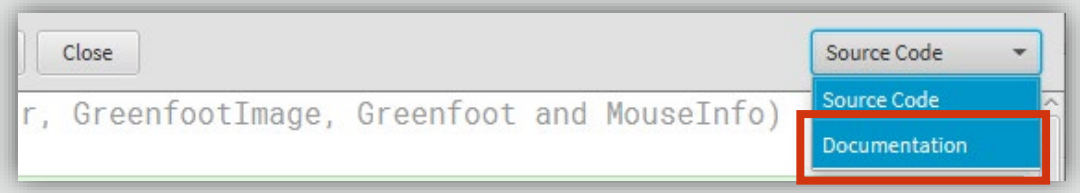
- El menú Object muestra todos los métodos que la instancia hereda de su clase y superclase
  - Haga clic con el botón derecho en la instancia para que aparezca el menú
  - Inherited From Actor muestra una lista de los métodos que la clase hereda de la superclase Actor



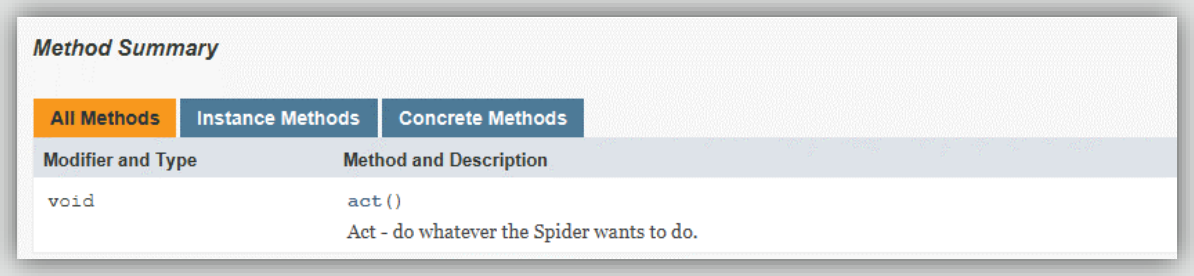
Pasos para visualizar los métodos heredados en el editor de códigos

- Haga clic con el botón derecho en una clase (en este ejemplo, en Spider)
- Haga clic en Open Editor
- En el editor de códigos, seleccione Documentation en el menú desplegable de la parte superior derecha



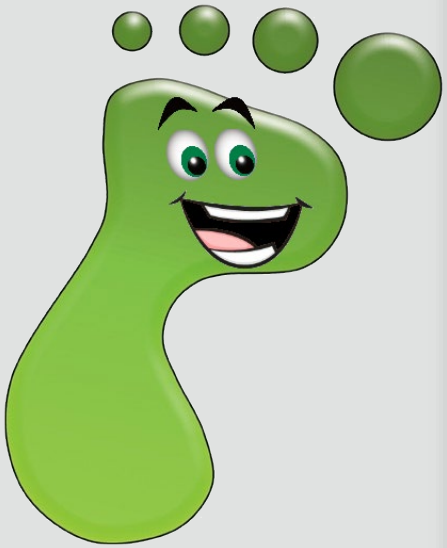


- Desplácese hacia abajo hasta Method Summary



Method Summary

- Method Summary muestra los métodos heredados de la clase



Spider - BeeWorld

Class Edit Tools Options

Spider

Compile Undo Cut Copy Paste Find... Close Documentation

Author:
(your name)

Constructor Summary

Constructors

Constructor and Description
`Spider()`

Method Summary

All Methods | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method and Description
void	`act()` Act - do whatever the Spider wants to do.

Methods inherited from class greenfoot.Actor

addedToWorld, getImage, getIntersectingObjects, getNeighbours, getObjectsAtOffset, getObjectsInRange, getOneIntersectingObject, getOneObjectAtOffset, getRotation, getWorld, getWorldOfType, getX, getY, intersects, isAtEdge, isTouching, move, removeTouching, setImage, setImage, setLocation, setRotation, turn, turnTowards

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

Loading class interface... Done. saved

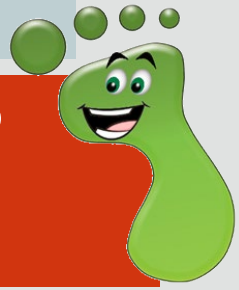
ORACLE Academy

Componentes de los métodos

- Un método cuenta con varios componentes que describen las operaciones o tareas que lleva a cabo
  - Tipo de retorno: Especifica el tipo de datos que devuelve el método
  - Nombre del método: Describe lo que hace el método
  - Lista de parámetros: Información que se incluye en la llamada al método
- Ejemplos de métodos:

```
void move(3)
int getX()
```

La llamada a un método ordena a la instancia que lleve a cabo una operación o tarea en Greenfoot. Lea el método para entender qué operación o tarea se llevará a cabo.



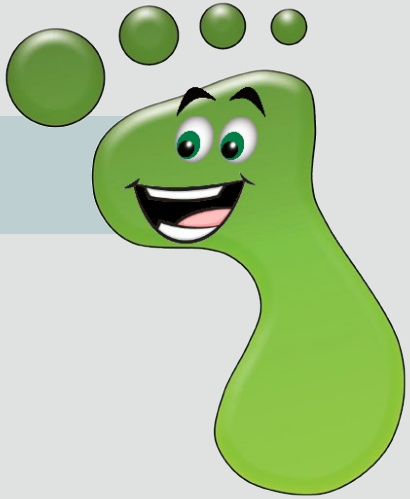
Firma de método

- La firma de método describe las intenciones del método
- Incluye los siguientes componentes:
  - Nombre del método
  - Lista de parámetros

```
void move(int)
```

Nombre del método

Lista de parámetros ()



Tipos de retorno

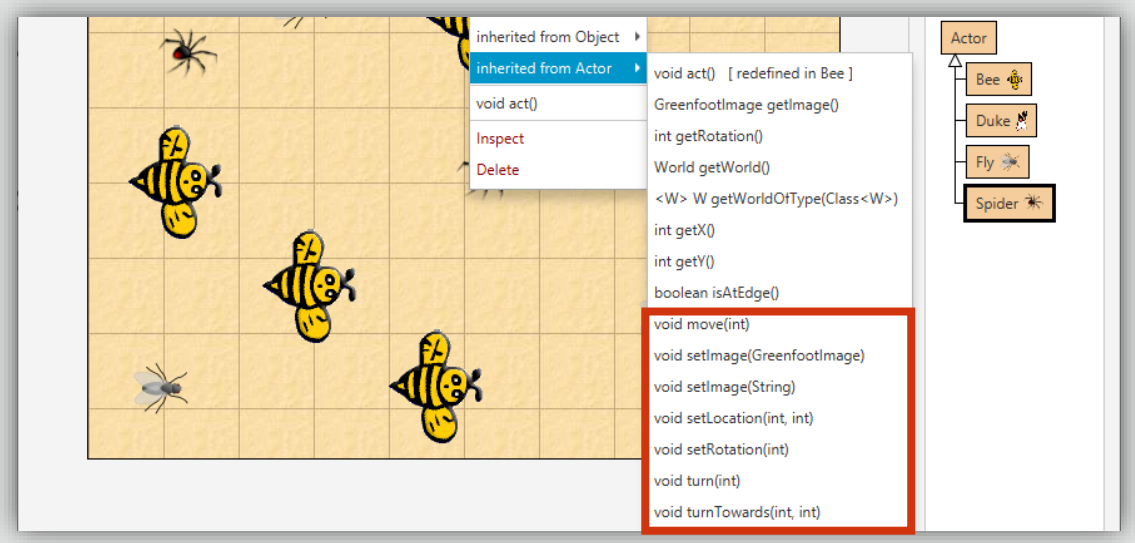
- El tipo de retorno es la palabra al principio del método que indica el tipo de información que devolverá la llamada al método
- Hay dos clases de tipos de retorno:
  - Void: No devuelve datos, emite un comando al objeto
  - Non-void: Devuelve datos, hace una pregunta al objeto

```
void move(int)
```

Tipo de retorno

Métodos con tipos de retorno void

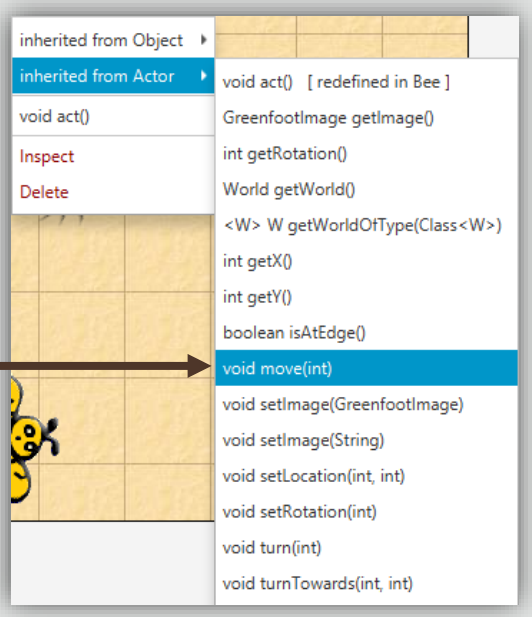
- Los métodos con tipos de retorno void emiten un comando que lleva a cabo una acción
  - Incluye la palabra "void"
  - No devuelve información sobre el objeto
  - Se utiliza para hacer que el objeto lleve a cabo una acción

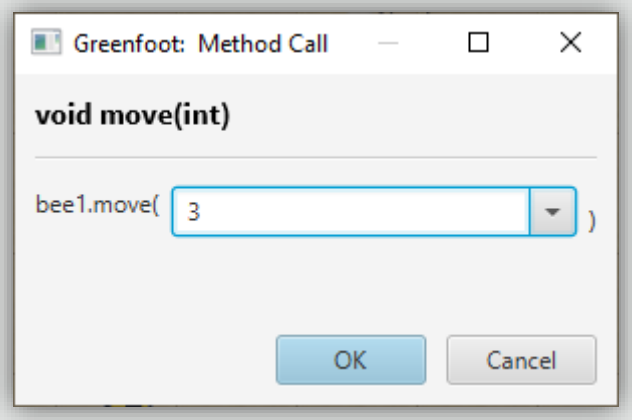


Llamada a métodos con tipo de retorno void

- Llamará a métodos con tipo de retorno void:
  - Para colocar con exactitud los objetos en su escenario inicial (el punto de inicio del juego)
  - Para ordenar a los objetos que lleven a cabo acciones en el juego

Ejemplo de selección
El método move



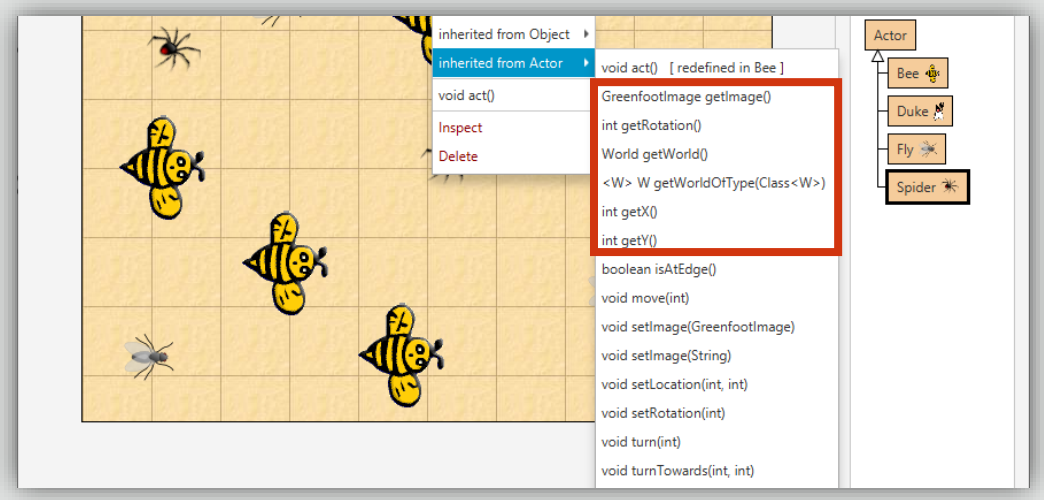


Hacer preguntas a los objetos

- Como programador, puede hacer preguntas a los objetos mediante métodos con tipos de retorno non-void para saber lo que un objeto puede hacer o lo que ha hecho en el pasado
- Por ejemplo, en el colegio, los profesores hacen preguntas a los estudiantes para comprobar si entienden lo que han aprendido en clase ese día
- Los estudiantes ofrecen respuestas a los profesores para que estos puedan saber lo que han aprendido

Métodos con tipo de retorno non-void

- Los métodos con tipo de retorno non-void hacen una pregunta al objeto
  - La firma de método no incluye la palabra "void"
  - El método devuelve información sobre el objeto, pero no lo modifica ni lo mueve



Ejemplos de tipos de retorno non-void

- Número entero (aparece como int)
  - Hace referencia a números enteros
  - Pregunta al objeto: ¿Cuántos?
- Expresión booleana
  - Devuelve un valor true o false
  - Tipos de preguntas que puede hacer a un objeto:
    - ¿Está tocando a otro objeto?
    - ¿Está en el extremo del mundo?

Parámetros del método

- Los parámetros proporcionan a los métodos datos adicionales para hacer que un objeto lleve a cabo una tarea cuando se necesita información para llamar al método
- Los parámetros están definidos por dos componentes:
  - Tipo de parámetro
  - Nombre del parámetro

Los parámetros se utilizan para indicar a los objetos que se muevan o para indicar a los objetos qué tipo de respuesta se espera cuando les hacemos una pregunta.

Ejemplos de parámetros del método

- Número entero (int): Introduce o muestra valores numéricos

```
private void updateLives(int change)
{
    lives +=change;
    updateImage(lives);
}//end of method
```

- Expresión booleana: Muestra valores true o false

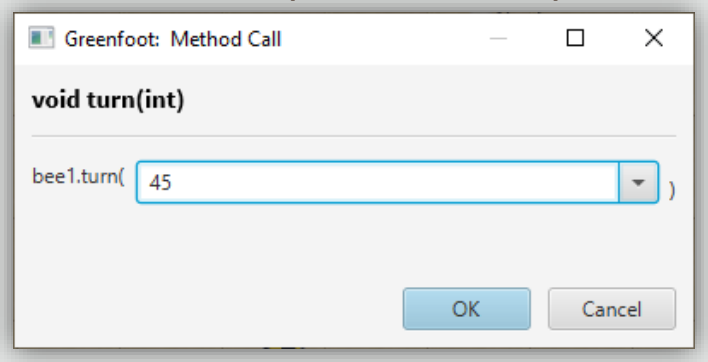
```
private void storeCurrentPosition(boolean collided)
{
    if(collided){
       prevX = getX();
       prevY = getY();
     }//endif
}//end of method store current position
```

- Cadena: Introduce o muestra valores de texto

```
private void play(String sound)
{//will play a sound file
  Greenfoot.playSound(sound);
}//end method Play
```

Listas de parámetros del método

- Las listas de parámetros del método indican si el método necesita información adicional para llamarlo y qué tipo de información
  - Las listas de parámetros aparecen como datos entre paréntesis
  - Suelen tener dos estados:
    - Vacío: No se espera ningún dato para llamar al método (método getRotation())
    - Con datos: Tienen datos y se espera uno o más parámetros para llamar al método
      - Por ejemplo, el método turn(int)



Propiedades del objeto

- Las propiedades del objeto describen la apariencia y habilidades de la instancia, por ejemplo:
  - Size
  - Color
  - Rango de movimientos

- Las propiedades pueden verse y modificarse en el código fuente de la clase
- Cada vez que se crea una nueva instancia de un Actor como Bee, esta tiene sus propias propiedades

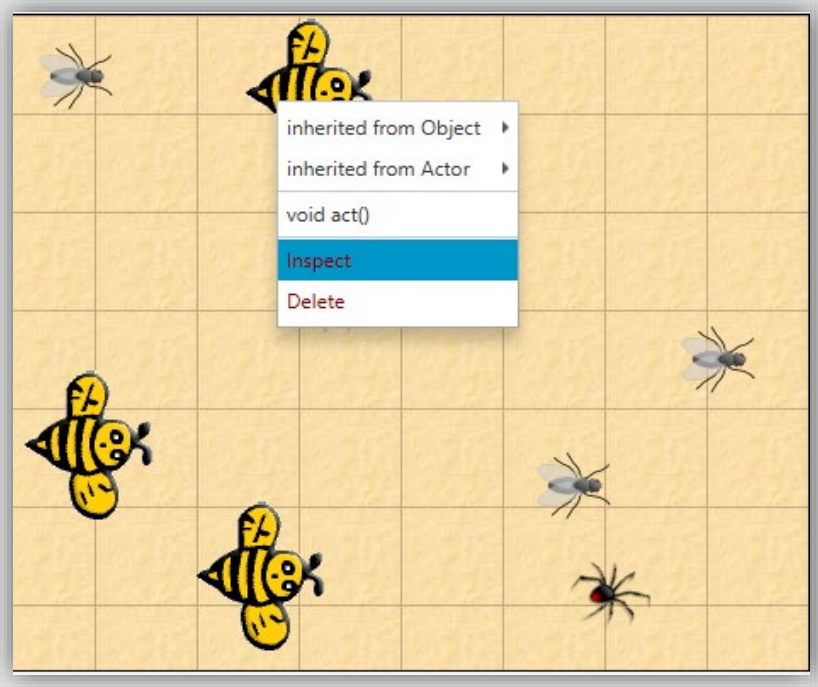
Variables

- Una variable, o campo, permite a la instancia almacenar información que puede utilizar de forma inmediata o posteriormente
- Por ejemplo, las propiedades del objeto son variables que almacenan información sobre la instancia, como su posición en el mundo

Las variables de una instancia son la memoria que pertenece a la instancia de la clase. Dicha memoria se puede guardar y acceder a ella posteriormente siempre que exista la instancia.

Ver las variables de las instancias

- Haga clic con el botón derecho en la instancia de un actor y, a continuación, haga clic en Inspect para ver las variables de la instancia en el inspector de objetos



Sintaxis de programación

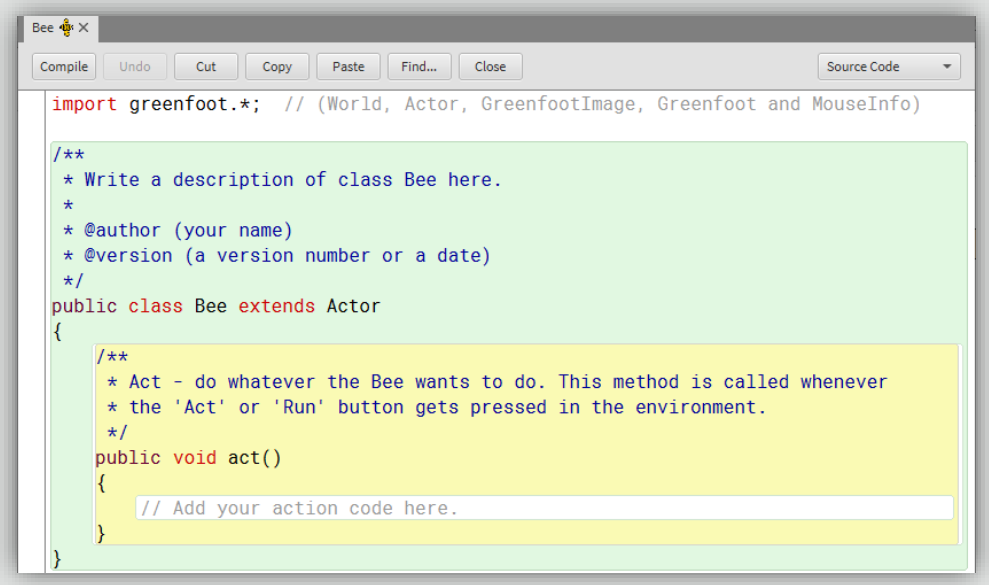
- El código fuente especifica todas las propiedades y características de una clase y sus objetos
- Escriba el código fuente (también conocido como sintaxis) en el editor de códigos de la clase para ordenar a los objetos que lleven a cabo una acción

```
/**
 * Prepare the world for the start of the program.
 * That is: create the initial objects and add them to the world.
 */
private void prepare()
{
    Bee bee = new Bee();
    addObject(bee,298,184);
    Fly fly = new Fly();
    addObject(fly,466,289);
    Spider spider = new Spider();
    addObject(spider,79,79);
    fly.setLocation(462,287);
    bee.setLocation(305,48);
```

ORACLE Academy

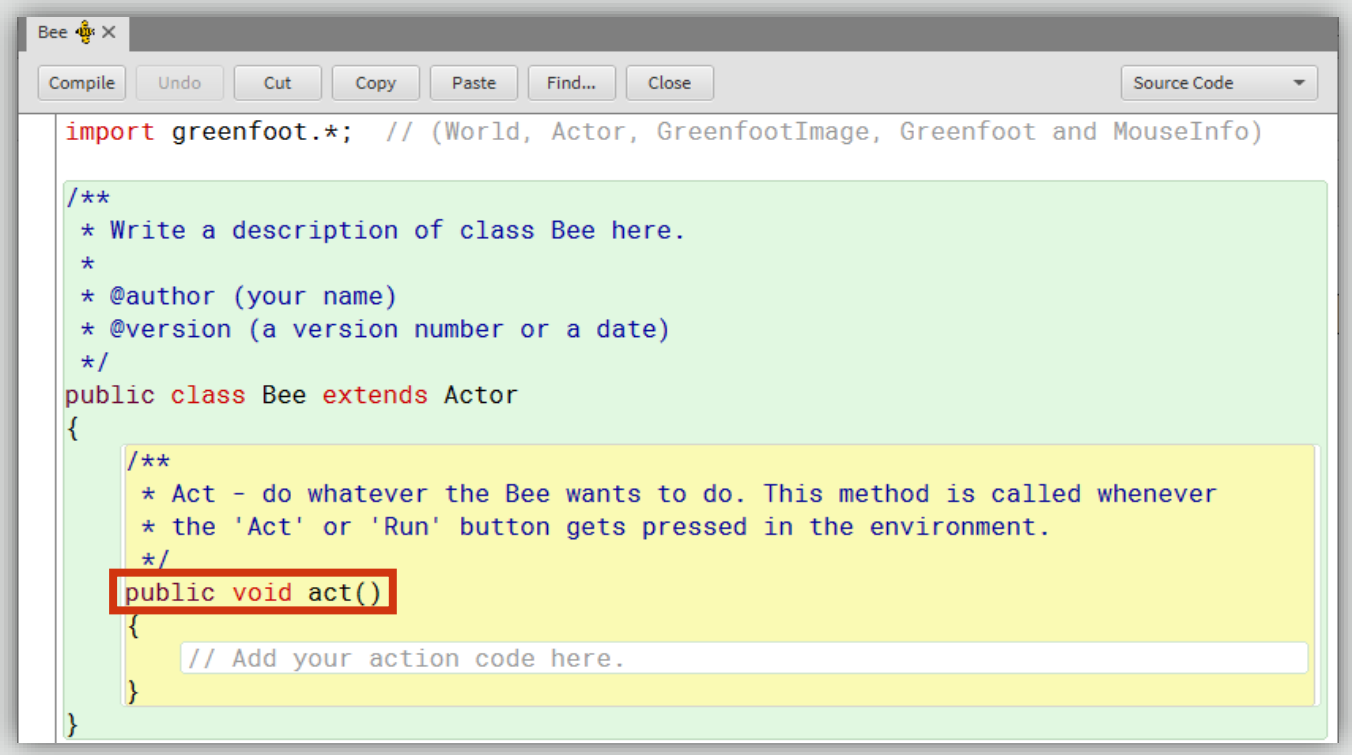
Mostrar el código fuente de la clase

- Desde el mundo, haga clic con el botón derecho en una clase y seleccione Open Editor para abrir el editor de códigos
- El código fuente que aparece define lo que pueden hacer los objetos de la clase



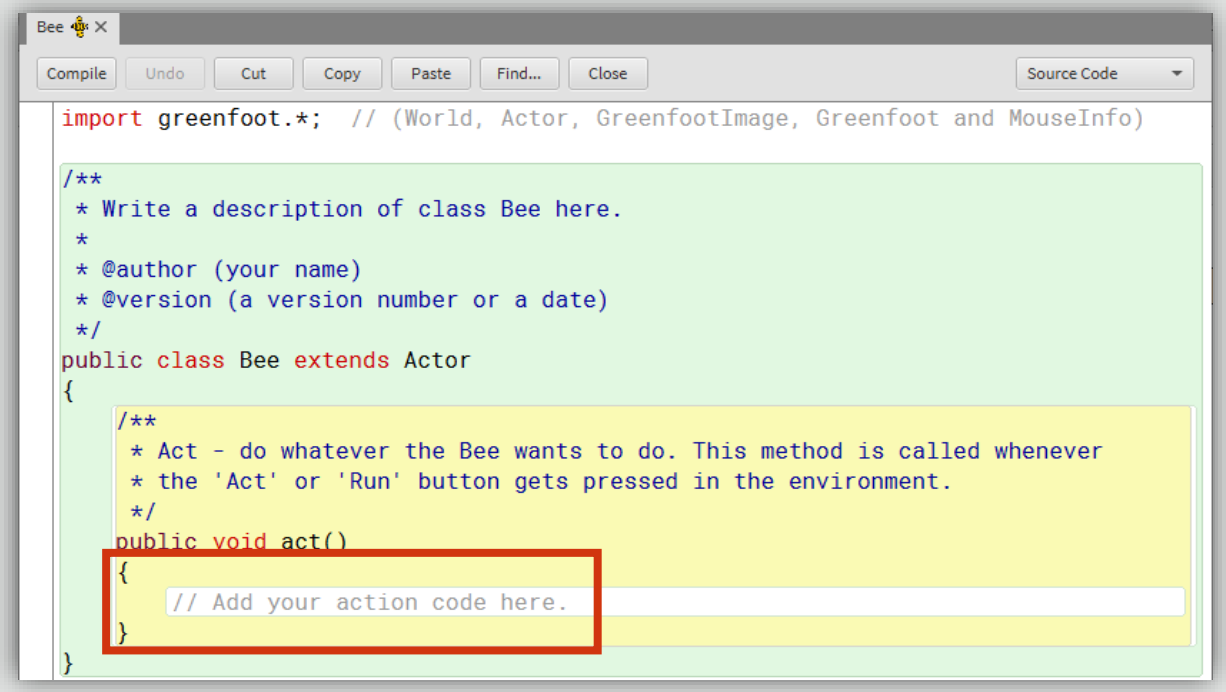
Método act()

- Siempre que haga clic en los controles de ejecución de Act o Run en el entorno, el objeto hará repetidamente lo que esté programado en el método act



Cuerpo del método act()

- Los corchetes y su contenido son el cuerpo del método
- Aquí puede escribir código para indicar a las instancias de la clase que actúen cuando se haga clic en los botones Act o Run



Ejemplo del método act()

- Llame a los métodos move() y turn() desde el método act() para que las instancias de la clase se muevan y giren
- Los métodos se deben escribir correctamente, sin errores tipográficos, con todos los caracteres y utilizando correctamente las mayúsculas, o el código fuente no se compilará

```
public class Bee extends Actor
{
    /**
     * Act - do whatever the Bee wants to do. This method is called whenever
     * the 'Act' or 'Run' button gets pressed in the environment.
     */
    public void act()
    {
        move(1);
        turn(15);
    }
}
```

Class compiled - no syntax errors

saved

Llamada a métodos desde el método act()

- Para llamar a métodos desde el método act(), escríbalos en secuencia de la siguiente forma:
  - Nombre del método en minúscula
  - Paréntesis, con la lista de parámetros si es necesario
  - Punto y coma, al final de la sentencia

```
public class Bee extends Actor
{
    /**
     * Act - do whatever the Bee wants to do. This method is called whenever
     * the 'Act' or 'Run' button gets pressed in the environment.
     */
    public void act()
    {
        move(1);
        turn(15);
    }
}
```

Proceso de depuración en Greenfoot

- Los caracteres escritos incorrectamente o que falten en su código fuente darán lugar a mensajes de error
- Si se encuentra un error, se mostrará un mensaje de error y este deberá ser corregido por el programador para que el programa funcione
  - Greenfoot proporciona estos mensajes de error para facilitar la corrección de errores y aprender de ellos

La depuración es el proceso de búsqueda y eliminación de bugs (o errores) en un programa informático.

Ejemplo de errores de sintaxis

- Al método move() le falta un punto y coma

Bee - BeeWorld

Class Edit Tools Options

Bee

Compile Undo Cut Copy Paste Find... Close Source Code

```
import greenfoot.*;  // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

/**
 * Write a description of class Bee here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class Bee extends Actor
{
    /**
     * Act - do whatever the Bee wants to do. This method is called whenever
     * the 'Act' or 'Run' button gets pressed in the environment.
     */
    public void act()
    {
        move(1)
        turn(15);
    }
}
```

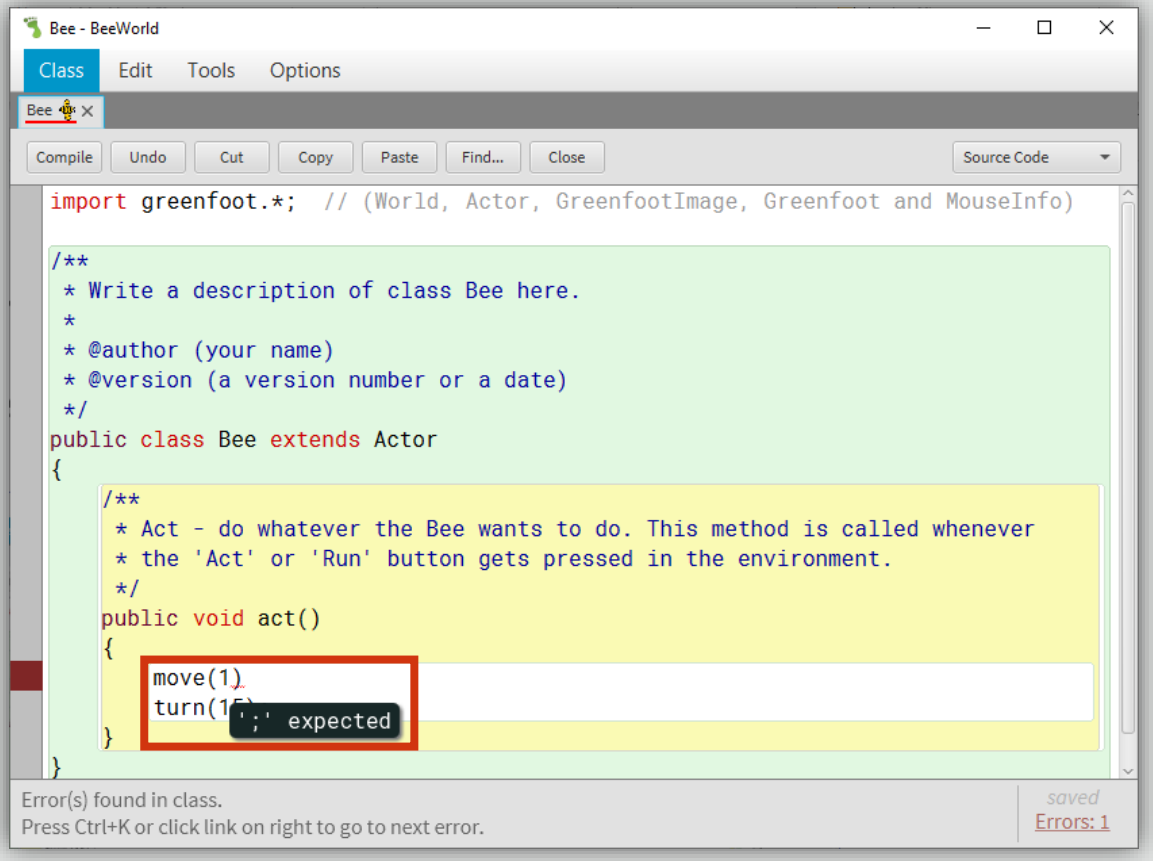
Error(s) found in class.
Press Ctrl+K or click link on right to go to next error.

saved
Errors: 1

ORACLE Academy

Ejemplo de mensaje de error

- Aparece un mensaje de error en la parte inferior de la pantalla y se resalta el código incorrecto



Terminología

- Entre los términos clave utilizados en esta lección se incluyen:
  - Depuración
  - Herencia
  - Variable de instancia
  - Método
  - Llamada al método
  - Parámetro
  - Tipo de retorno
  - Firma de método
  - Variable

Inténtelo.

Colocación de instancias

Para llevar a cabo esta actividad debe comenzar con el archivo de proyecto que se guardó en el tema anterior FrogFly_L2T2.

FrogFly_L2T2.zip

Instrucciones:

Agregue 2 rocas, 2 hormigas, 1 rana, 10 moscas y 1 instancia de instrucciones al escenario.

Coloque las moscas con el método setRotation(), de forma que todas miren en diferentes direcciones.

Coloque la rana en x = 50 y = 50 mediante el método setLocation().

Coloque las hormigas y las rocas en el lugar que desee en el mundo.

Guarde el escenario.

Inténtelo.

Codificación de una instrucción de programación y ejecución del escenario

Para llevar a cabo esta actividad debe comenzar con el archivo de proyecto que se guardó en el tema anterior FrogFly_L2T2.

FrogFly_L2T2.zip

Instrucciones:

Abra el editor de códigos de la clase Fly. Desde el método act(), programe la mosca para que se mueva 4 pasos y, a continuación, gire 5 grados.

Abra el editor de códigos de la clase Ant. Desde el método act(), programe la hormiga para que se mueva 3 pasos y, a continuación, gire 5 grados.

Guarde el escenario como FrogFly_L3T3, compílelo y ejecútelo.

Inténtelo.

Codificación de una instrucción de programación y ejecución del escenario

Para llevar a cabo esta actividad debe comenzar con el archivo de proyecto que se guardó en el tema anterior Jungle_L1T1.

Jungle_L1T1.zip

Instrucciones:

Abra el escenario Jungle (Selva) que guardó en el tema anterior Jungle_L1T1 y agregue un hipopótamo, un elefante, y un lémur al mundo.

Abra el editor de códigos de la clase de Elephant (Elefante). En el método act(), programe el elefante para que se mueva 8 pasos y, a continuación, gire 2 grados.

Abra el editor de códigos de la clase Hippo (Hipopótamo). En el método act(), programe el hipopótamo para que se mueva 3 pasos y, a continuación, gire 5 grados.

Guarde el escenario como Jungle_L3T2, compílelo y ejecútelo. Observe cómo se mueven el elefante y el hipopótamo.

Summary

- En esta lección, debe haber aprendido lo siguiente:
  - Definición de parámetros y cómo se utilizan en los métodos
  - Comprensión de la herencia
  - Descripción de las propiedades de un objeto
  - Análisis del objetivo de una variable
  - Descripción de conceptos de programación y definición de la terminología


ORACLE
Academy