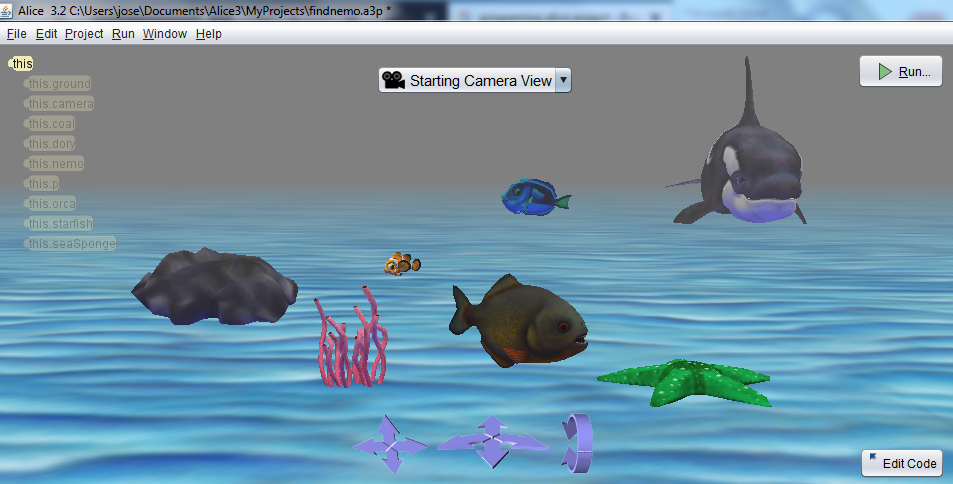
**8. Aprendiendo POO con Alice**

**8.1 Alice**



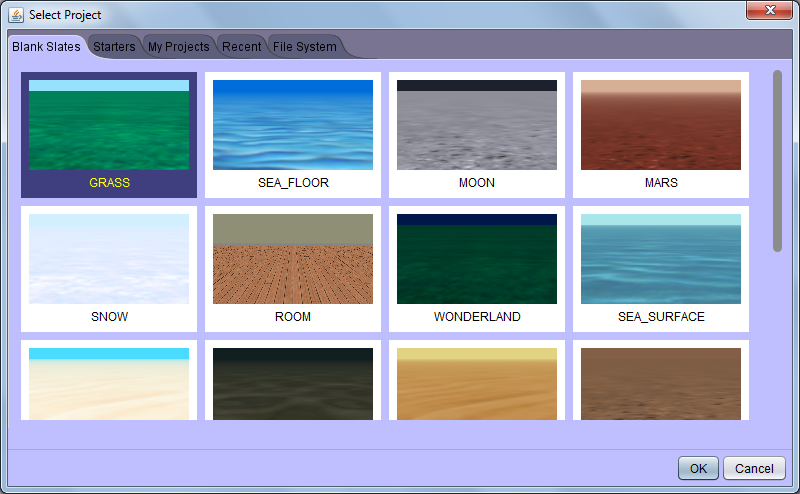
Alice es un innovador ambiente de desarrollo basado en bloques 3D, creado en el lenguaje de programación Java en 1999 por la universidad Carneige Mellon, con la intención de usarse como herramienta didáctica para facilitar el aprendizaje del paradigma de la programación orientada a objetos de una forma divertida y atractiva para los estudiantes. Es una aplicación de escritorio que se puede descargar gratuitamente desde el sitio oficial https://www.alice.org/ y está disponible para Windows, Linux y Mac. Al momento de escribir este material, la versión más reciente es la 3. También existe un plugin para Netbeans. Como una función especial, el entorno de Alice permite exportar como un video a Youtube parte de la ejecución del juego realizado.

Básicamente Alice permite crear un juego 3D en los que puede usar una galería de objetos que trae disponible, partiendo de la librería de objetos que incorpora la herramienta. En el juego, cada elemento es un objeto que puede interactuar con los demás. Importante: para instalar Alice necesita tener instalado previamente el JDK de Java.

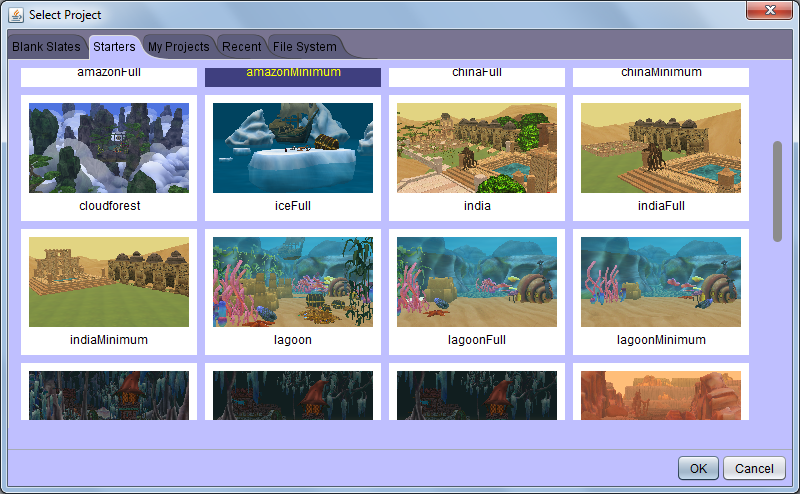


**8.2 Crear un proyecto**

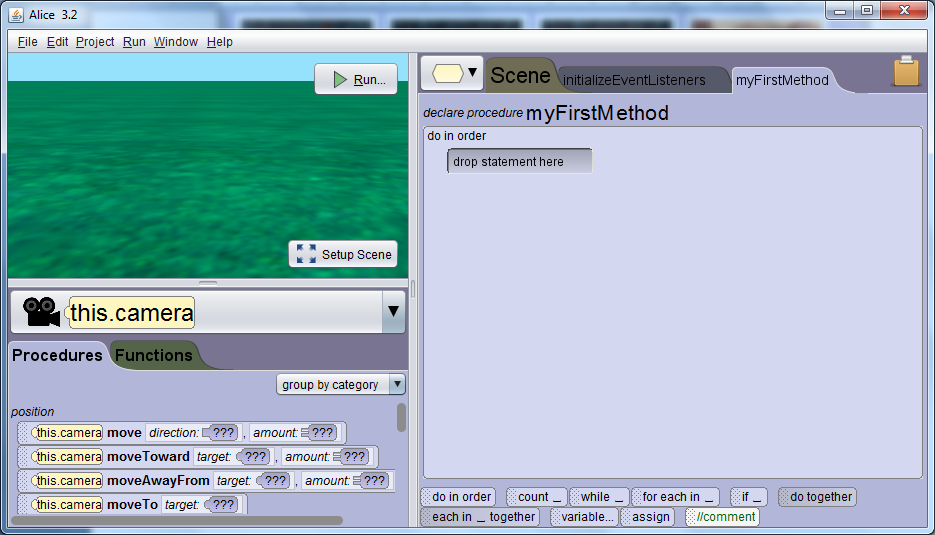
Al abrir Alice, la primera ventana que aparece permite crear un proyecto nuevo en blanco:



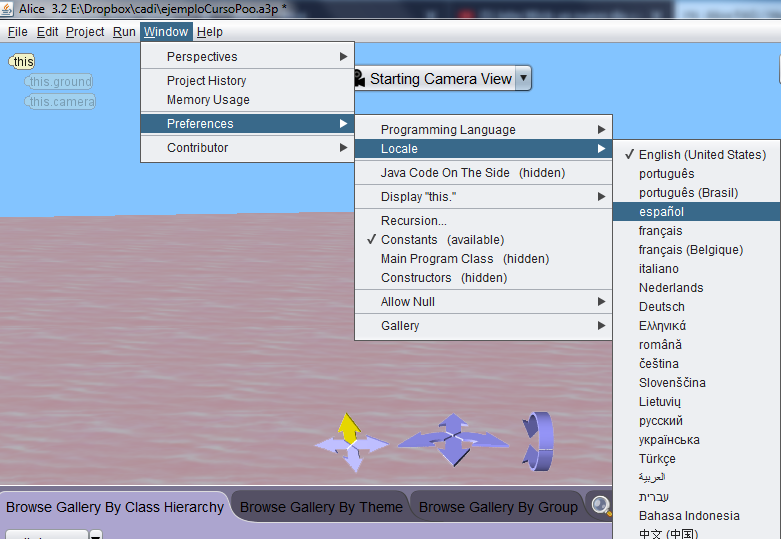
O crear un proyecto con el escenario ya teniendo incorporado varios objetos:



Al seleccionar un escenario (ya sea en blanco o con elementos), al hacer click en el botón Ok se abre el entorno de Alice para empezar a programar.



Por defecto, el idioma del entorno es el ingles, pero si el usuario desea cambiar el idioma, puede hacer en la opción Preferences - > Locale:



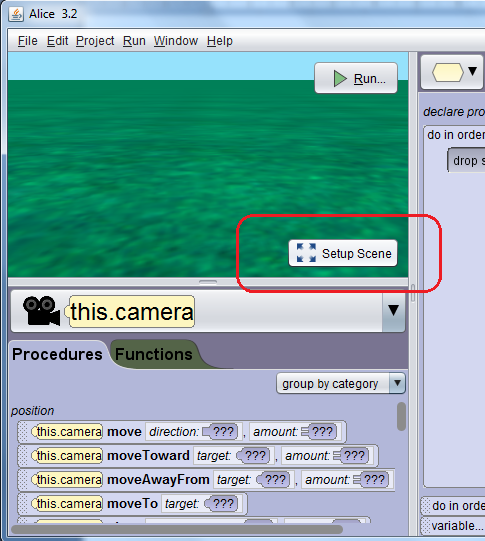
**8.3 Conociendo el entorno**

El entorno de Alice tiene 2 perspectivas:

- edición de código: se utiliza para programar el juego.

- configuración del escenario: se utiliza para diseñar el escenario, agregar objetos en tiempo de diseño, modificar sus propiedades, etc.

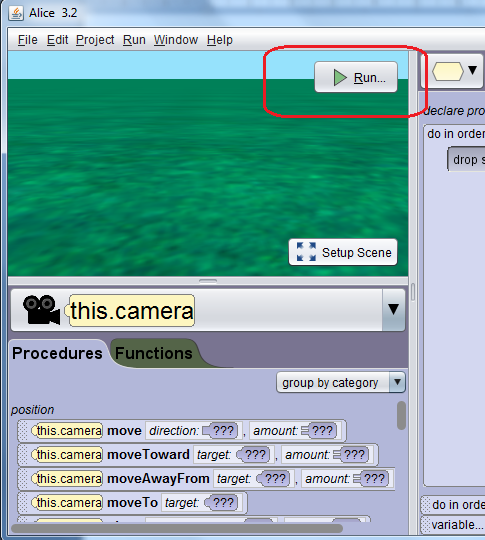
La perspectiva en la que por defecto de abre Alice es la perspectiva de código. Para pasar de la perspectiva de código a la de diseño, debe hacer click en el botón “Setup Scene”, como se muestra en la imagen:



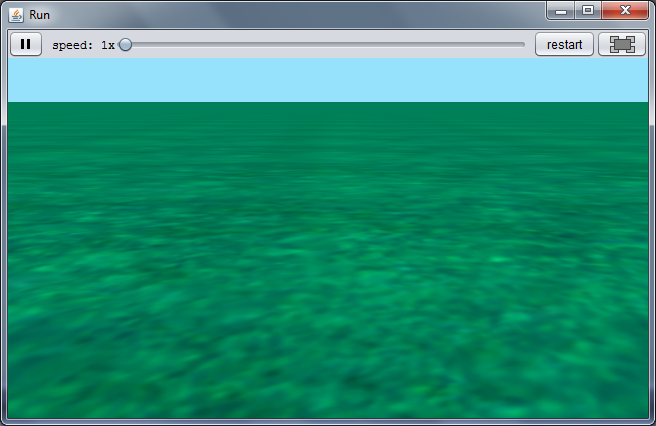
Estando en la perspectiva de configuración del escenario puede devolverse a la perspectiva de código haciendo click en el botón “Edit Code”:



En cualquier momento puede ejecutar el proyecto haciendo click en el botón “Run” o puede presionar la combinación de teclas CTRL+F5:



Al ejecutar el proyecto, aparece una ventana donde se reproducirá lo que se haya programado. Mientras esta en ejecución el proyecto no se puede editar ningún elemento del mismo. Al cerrar esa ventana, se regresa al entorno y puede continuar con la edición del proyecto:

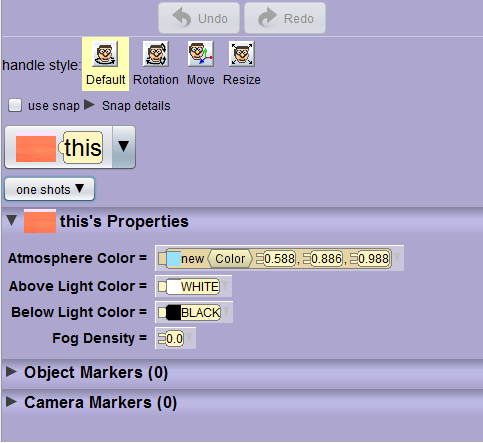


Al guardar un proyecto, este se guarda con la extensión “a3p”.

**9. Conociendo Alice**

**9.1 Configurando el Escenario**

En un proyecto Alice, el escenario lo conforman el mundo y todos los objetos que están dentro de este. Al crear un proyecto en blanco, por defecto Alice incorpora 3 objetos en el escenario: el mundo (this), el suelo (ground) y la cámara (camera). El suelo y la cámara son 2 objetos que son parte del objeto mundo. Los elementos del escenario se pueden modificar en tiempo de diseño o en tiempo de ejecución. Primero se aprenderá a modificarlo en tiempo de diseño (perspectiva de configuración del escenario). Estando en la perspectiva de configuración del escenario, Alice provee un panel para modificar las propiedades de los objetos:



**9.2 Modificando propiedades**

El mundo (this), el suelo (ground) y la cámara (camera) son objetos y como tales, tienen propiedades. En el panel de propiedades hay una lista que permite seleccionar el objeto que se necesita modificar. Según el elemento que este seleccionado en la lista, aparecerá una lista de propiedades que se pueden modificar de éste. Por defecto, esta seleccionado “this”, que se refiere al mundo. El mundo es el contenedor de todos los elementos que tendrá el proyecto.



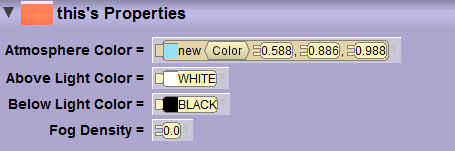
El objeto mundo tiene 4 propiedades:

-Atmosphere color: color de la atmosfera o del cielo. Puede ser un color absoluto (rojo, negro, blanco, etc) o un color personalizado.

- Above ligth color: color de la luz de arriba. Igual que el anterior.

- below ligth color: color de la luz de abajo

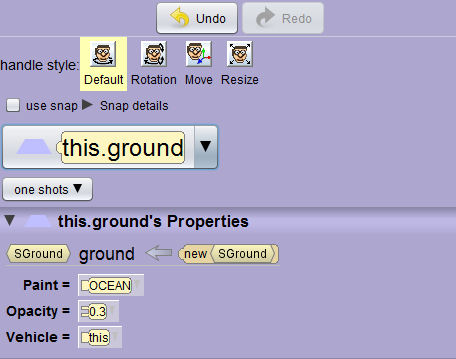
- fog density: densidad de la neblina. Oscila entre 0 y 1.



Inicialmente, todos estos atributos poseen unos valores por defecto, que dependerán del escenario que se haya seleccionado al crear el proyecto. Independientemente de eso, estos valores se pueden cambiar al gusto.

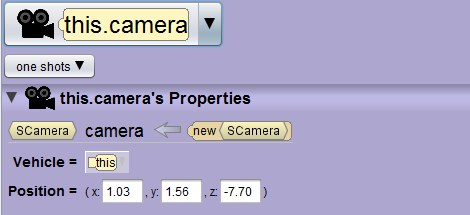
El objeto ground (sueldo) tiene 3 propiedades:

* Paint: tipo de suelo. Puede ser: nieve, agua, hierba, entre otros.
* Opacity: indica la transparencia del tipo de suelo. Oscila entre 0 y 1.
* Vehicle: es el objeto contendor. Por defecto es this (el mundo)



**9.3 La cámara**

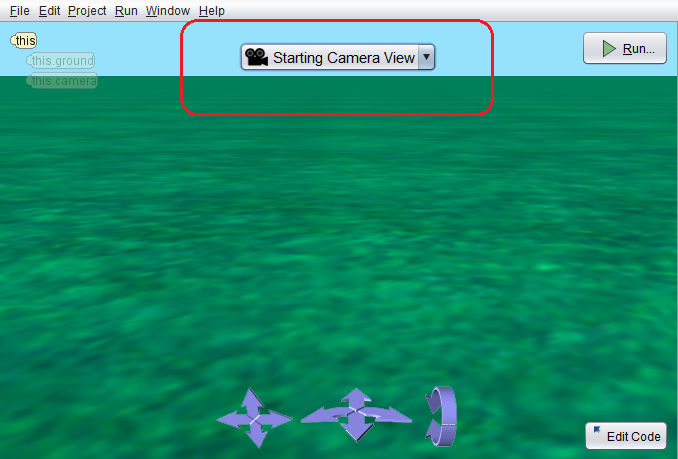
La cámara es la encargada de captar todo el escenario para dar la impresión de la filmación de algo que está ocurriendo en algún lugar. Es la que la perspectiva de los objetos y con la que se especifica qué parte de todo el mundo se desea captar. El objeto cámara solo posición determinada por 3 valores: X,Y y Z, que indica las coordenadas donde estará ubicada. Los tres valores son números reales.



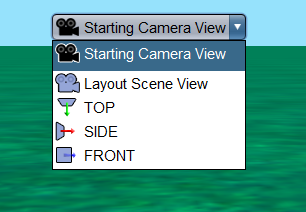
Hay otra forma de modificar la ubicación de la cámara, usando las flechas ubicadas en el escenario;



Además de cambiar la ubicación, también se puede cambiar hacia donde está mirando la cámara. También se puede cambiar el punto de vista del programador utilizando la lista de selección que se encuentra en el centro del escenario:



Los posibles valores a seleccionar de las vistas son:



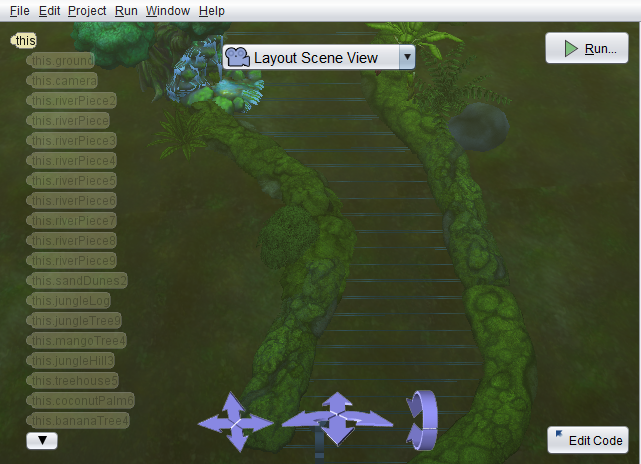
* Starting camera view: es el valor por defecto. Indica que la vista del usuario será la misma vista de la cámara.
* Layout Scene: es una vista ligeramente superior y ligeramente atrás al de la cámara
* Top: es una vista totalmente desde arriba
* Side: es una vista totalmente lateral
* Front: es una vista de frente

En un escenario vacío tal vez no se note la diferencia de cualquiera de estas vistas, pero cuando el escenario contiene objetos es fácil de entender su uso.

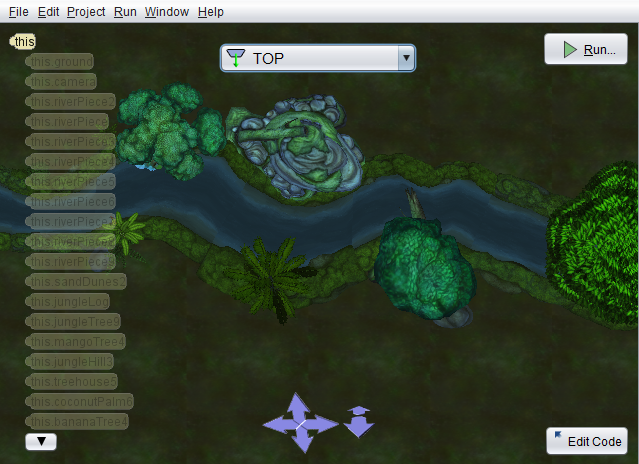
El siguiente ejemplo de un escenario muestra el uso de las distintas vistas, comenzando por la vista desde la cámara:



El siguiente muestra la perspectiva “Layout Scene”:



La vista desde arriba:



La vista de lado



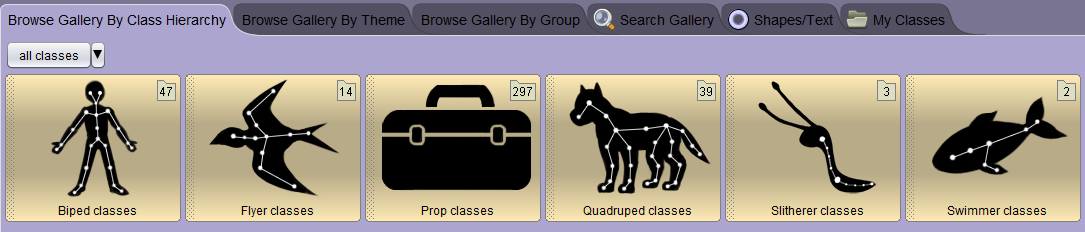
La vista de frente:



**10. Trabajando con Objetos**

**10.1 Agregar objetos al escenario**

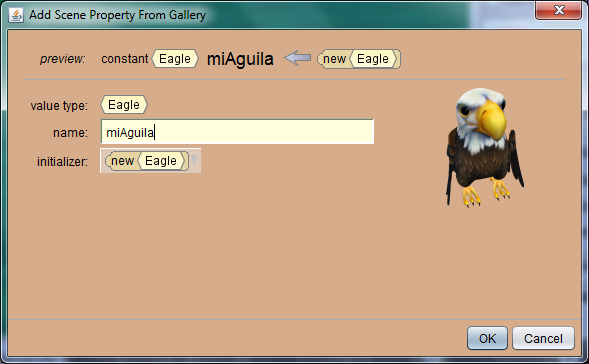
Como se ha comentado previamente, Alice es una herramienta didáctica para ayudar a entender los conceptos de la programación orientada a objetos. Por lo tanto, en Alice todo es un objeto. Un nuevo proyecto vacio tiene un escenario con 3 objetos por defecto, pero se pueden agregar más objetos. Para agregar objetos, primero se debe seleccionar la clase a la cual pertenece el objeto que se desea agregar. Alice provee una librería de clases que se pueden usar, agrupando las clases en súper clases:



Por ejemplo, si se selecciona la súper clase de Voladores (Flyer), se mostraran todas las clases de aves que están definidas en la galería:



Luego, se selecciona la sub clase específica. Si no hay otra sub clase, Alice muestra una ventana donde solicita al usuario indicar un nombre al objeto que esta por crearse:



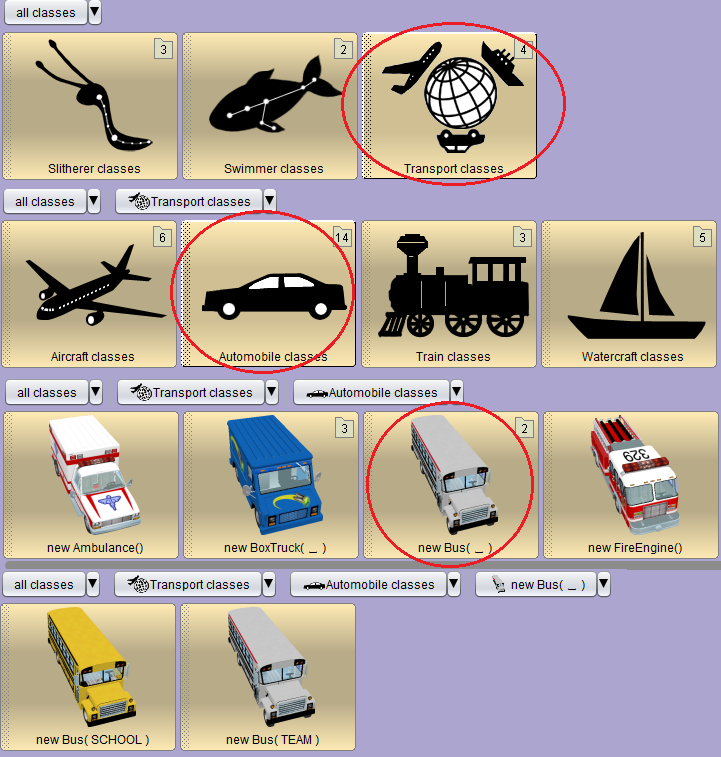
El nombre que por defecto le asigna Alice es el mismo nombre de la clase pero en minúsculas. Este nombre será con el que se hará referencia posteriormente a este objeto en la programación de los métodos. El nuevo objeto es añadido al escenario:



Aunque se pueden agregar varios objetos de la misma clase al escenario, no puede haber 2 objetos con el mismo nombre. En la siguiente imagen se muestra el error que se obtendría al intentar hacerlo:

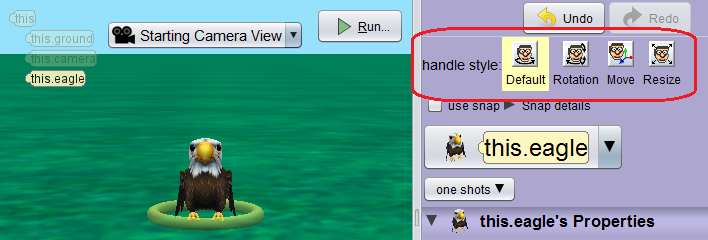
****

Algunas clases tienen varias sub clases y para crear el objeto se debe navegar entre ellas hasta llegar a la clase específica:

****

**10.2 Modificando los objetos**

Cuando un objeto es agregado al escenario, es agregado con valores por defecto en sus atributos. Si se agrega un segundo objeto, es probable que no se distinga porque está montado sobre el primero. Una forma de modificar la ubicación y tamaño del objeto es usando los “handle style” que aparecen en el panel de propiedades, para mover, rotar o redimensionar el objeto:

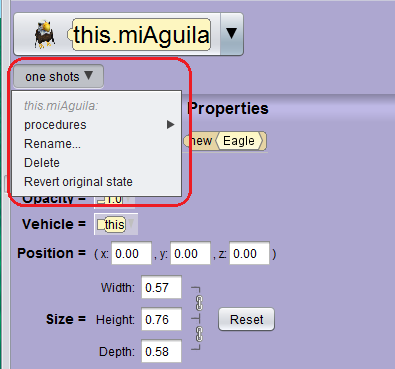


Los valores de los atributos de un objeto pueden modificarse en el mismo panel donde se modificaron previamente las propiedades de los objetos. Para modificar un objeto, primero debe seleccionarlo en la lista o en el escenario. Todos los objetos tienen los atributos:

* Position: posición donde está ubicado (X,Y,Z)
* Size: es el tamaño del objeto (width, height y depth)



También se pueden realizar algunas actividades sobre el objeto, como por ejemplo, renombrarlo, eliminarlo, volverlo al estado original:



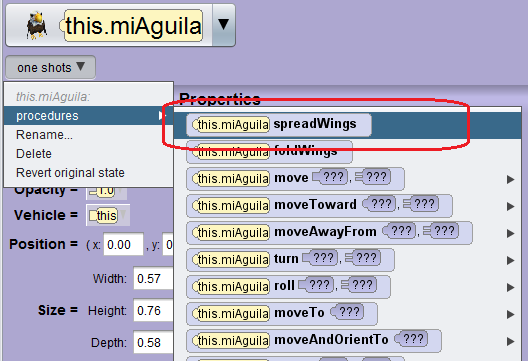
**10.3 Modificando objetos contenidos**

Los objetos normalmente están compuestos por otros objetos, por lo que en ocasiones es necesario modificar una propiedad específica de alguno de los objetos que están contenidos en el objeto en cuestión. Para tal fin, se debe seleccionar el objeto contenedor y luego ubicar el objeto contenido para luego modificar sus propiedades:

**11. Los métodos**

**11.1 Ejecutando métodos**

Repasando lo visto en capítulos anteriores, un método es una función que realiza un objeto. Los métodos afectan o modifican al objeto que los ejecuta. En Alice, los métodos son llamados “procedures” y permiten que en tiempo de diseño un objeto pueda ser modificado ejecutando alguno de los métodos que éste posea según su clase, por ejemplo, la clase águila tiene un método para hacer que el objeto despliegue las alas:



El resultado es que el objeto “miAguila” desplegara las alas en el escenario:



Asimismo, puede ejecutarse el método que recoge las alas para volver la objeto a su estado original:



**10.2 tipos de Métodos más usados**

La mayoría de las clases en Alice contienen métodos comunes que permiten modificar el estado de los objetos, cambiando con estos métodos propiedades como: ubicación, dirección y sentido. Los métodos más comunes son:

* Move
* Turn
* Roll

**10.2 Métodos de objetos contenidos**

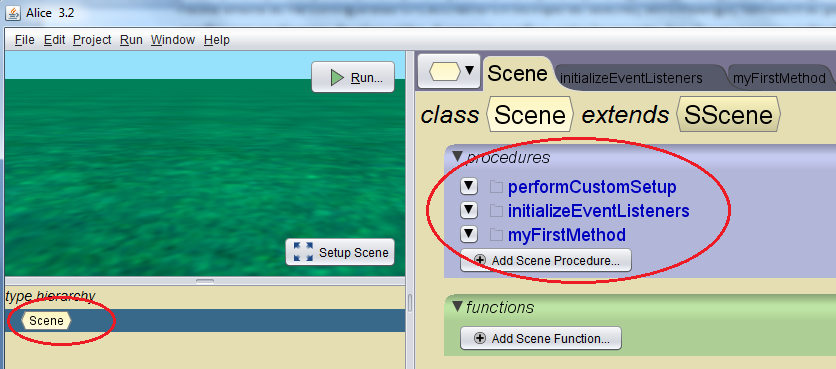
Para ejecutar un método de un objeto contenido adentro de otro se debe acceder al objeto contenido, para luego hacer referencia al método particular.

**12. Programar orientado a objetos en Alice**

**12.1 Configuración el escenario**

Hasta ahora se ha configurado el escenario en tiempo de diseño; sin embargo, también se puede configurar en tiempo de ejecución. Como se explico anteriormente, las clases poseen métodos predefinidos, pero también se pueden crear métodos que ejecuten acciones particulares sobre un objeto de cierta clase. En primera instancia, se crearan métodos para el escenario.

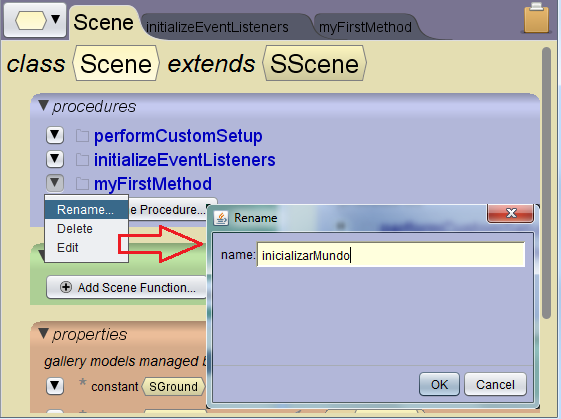
Los métodos se programan en la perspectiva “Edición de código”. Al seleccionar una de las clases de los objetos que han sido agregados al escenario, se pueden visualizar los métodos que éste tiene programados. Inicialmente cuando el escenario está vacío, la única clase que se muestra es “Scene” que es la clase que agrupa a los escenarios:



Alice agrega por defecto al escenario un método con el nombre “myFirstMethod”, que se ejecuta automáticamente al iniciar el juego. Este método se utiliza para manipular el escenario en tiempo de ejecución al iniciar el mundo. Al seleccionar la pestaña con el nombre del método, se activa un espacio donde se pueden agregar las instrucciones que ejecutara el método en el área con el texto “drop statement here”:

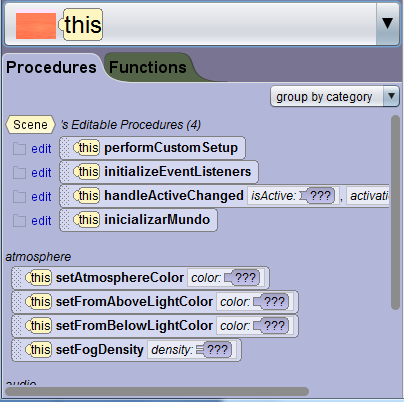


Puede cambiar el nombre al método ubicándolo en la pestaña “Scene”

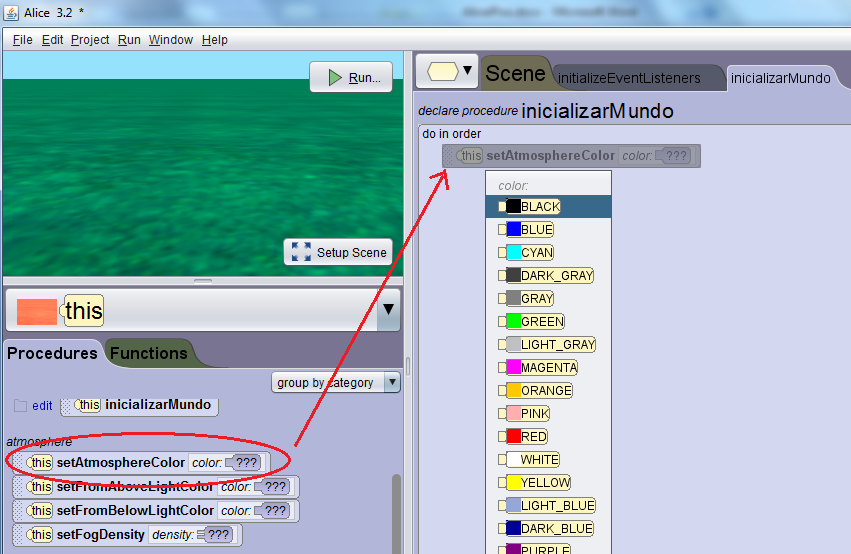


**12.2 ejecutar métodos**

Al inicializar el mundo, se puede cambiar la configuración del escenario en tiempo de ejecución. Esto se logra ejecutando métodos de la clase “Scene” que permitan cambiar una de sus propiedades, que como se estudio en capítulos anteriores son: el color de la atmosfera, el nivel de neblina, el color de la luz, etc. Para lograr este objetivo, se deben ubicar en el panel inferior izquierdo los métodos que contiene la clase “Scene”:



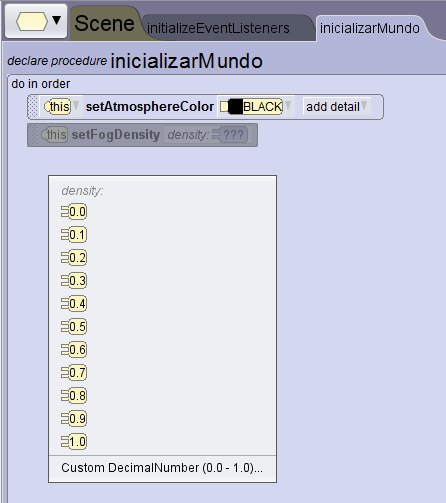
Al seleccionar el método que se desea usar, se arrastra hacia el espacio identificado con el texto “drop statement here”. Por ejemplo, si se desea cambiar el color de la atmosfera al iniciar el juego, se arrastra el método “setAtmosphereColor”:



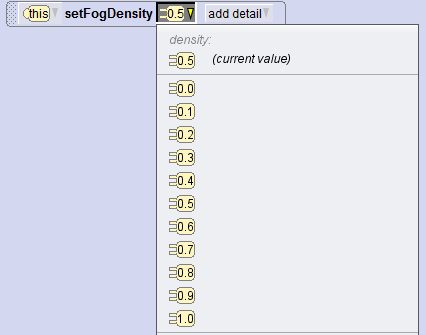
Se pueden agregar varias instrucciones en el método para que se ejecuten una después de otra. Por ejemplo, si además de cambiar el color de la atmosfera se necesita cambiar el nivel de neblina:



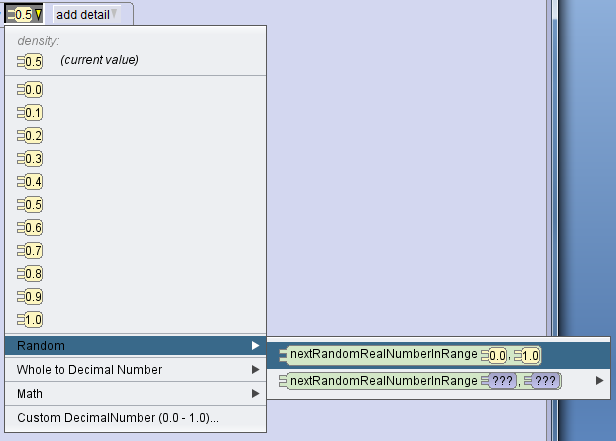
Cuando se utilizan valores numéricos, Alice provee valores predeterminados para seleccionar uno de estos, por ejemplo, al establecer el valor de la densidad de la neblina se puede seleccionar un número entre 0 y 1:



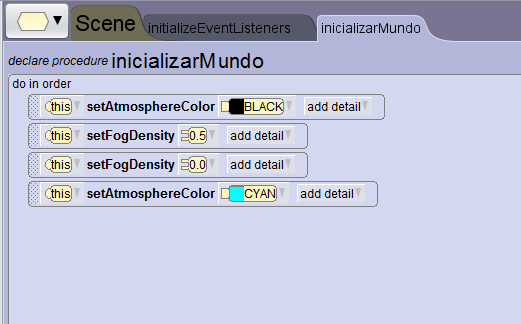
Este valor asignado se puede cambiar por otro valor de los que provee por defecto Alice:



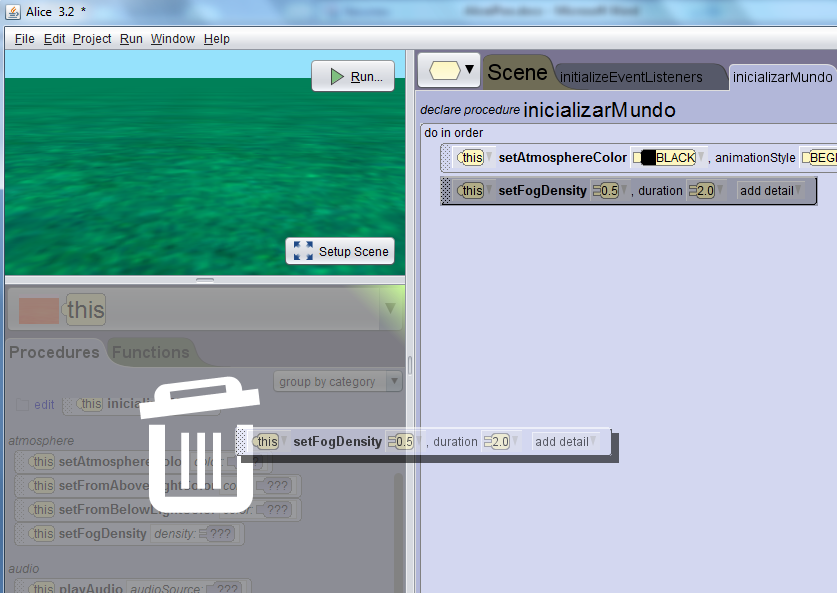
O se puede seleccionar un número aleatorio:



Se pueden agregar tantas instrucciones como sea necesario:



Si se desea eliminar alguna instrucción, simplemente se arrastra nuevamente hacia el panel izquierdo y desaparecerá del método:

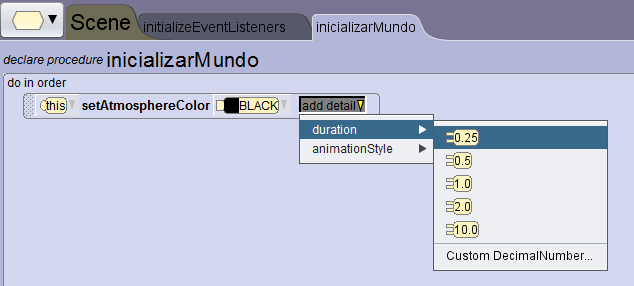


**12.3 configuración de la Animación**

Cuando se ejecutan instrucciones que modifican el estado de un objeto, Alice por defecto hace una animación. A la mayoría de estas instrucciones se le pueden agregar detalles de la animación, tales como:

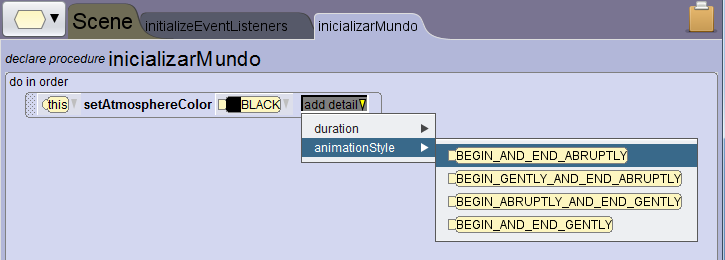
* estilo de animación
* duración de la animación del efecto

Para establecer algún detalle, se hace click en la opción “Add detail” y se selecciona el detalle que se desea agregar. Para la duración de la animación se puede seleccionar alguno de los valores predeterminados o un valor especifico:



También se puede establecer el estilo de animación que se desea usar al hacer el cambio de color. Los posibles valores son:

* BEGIN\_AND\_END\_ABRUPTLY: se utiliza para iniciar y terminar la animación del cambio de valor abruptamente.
* BEGIN\_GENTLY\_AND\_END\_ABRUPTLY: se utiliza para hacer la animación del cambio de valor suavemente al inicio pero abruptamente al final.
* BEGIN\_ABRUPTLY\_AND\_END\_GENTLY: se utiliza para iniciar la animación del cambio de valor abruptamente y terminarla suavemente.
* BEGIN\_AND\_END\_GENTLY: se utiliza para iniciar y terminar abruptamente la animación del cambio de valor.



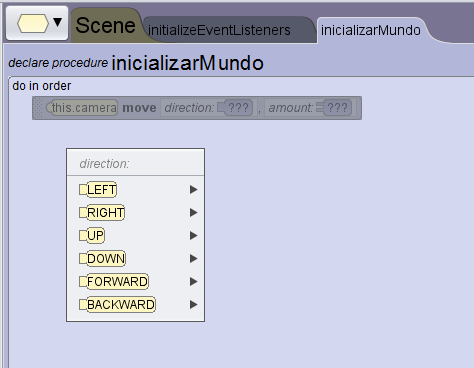
Para eliminar algún detalle agregado a la instrucción, se hace click con el botón derecho sobre esta y se selecciona la opción “Remove animation Argument”:



**13 Manipular objetos**

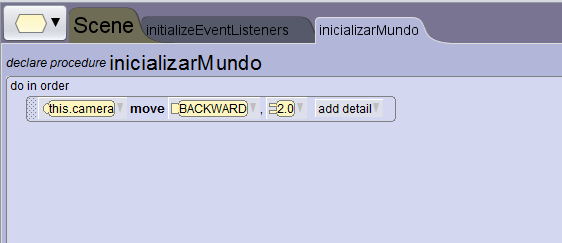
**13.1 métodos de la cámara**

La cámara como cualquier otro objeto puede ejecutar métodos para cambiar su estado. La cámara tiene una gran cantidad de posibles movimientos que lograran cambiar el escenario que se esté captando en un momento dado. El método más básico es “move”, que tiene 6 posibles movimientos:



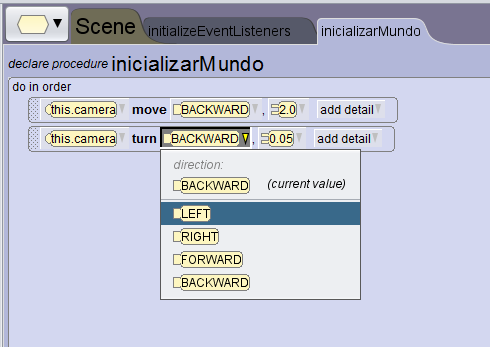
Y luego se especifica una cantidad de unidades hacia donde se desea mover la cámara. La cámara se puede mover:

* Left: hacia la izquierda.
* Rigth: hacia la derecha.
* Up: hacia arriba.
* Down: hacia abajo.
* Forward: hacia adelante.
* Backward: hacia atrás.

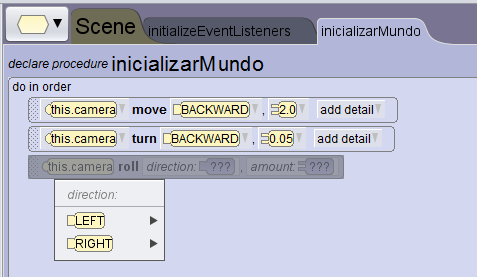


Así mismo se puede apuntar la cámara hacia una dirección en particular con el método “turn”. El valor en que se puede girar se mide en número de vueltas o fracciones de esta. Por ejemplo, si coloca el valor 1 es porque va a girar una vuelta completa en la dirección indicada. Se puede girar hacia:

* Left: hacia la izquierda.
* Rigth: hacia la derecha.
* Forward: gira hacia adelante (da la impresión de girar hacia abajo).
* Backward: gira hacia atrás (da la impresión de girar hacia arriba).



El método “roll” hace girar la cámara sobre su eje. Se puede girar a la izquierda o a la derecha:



El resto de los métodos trabajan en función de algún objeto objetivo. Por ejemplo:

* moveToward: acercarse hacia el objetivo
* moveAwayFrom: alejarse del objetivo
* moveTo: mover al objetivo
* turnToFace:
* orientTo:

**13.2 métodos de otros objetos**

Casi todas las clases poseen estos métodos para modificar la ubicación y la orientación del objeto, pero algunas clases tienen otros métodos particulares que también se pueden ejecutar. Por ejemplo, el águila tiene el método para adoptar la posición de vuelo. Para agregar ese método, debe seleccionar el objeto de la clase Águila y arrastrar el método:

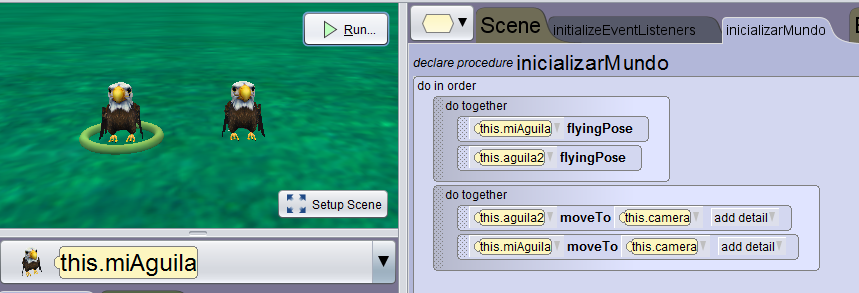


Se pueden llamar varios métodos del mismo objeto, por ejemplo, adoptar posición de vuelo y luego moverse hacia la cámara:



**13.3 ejecución en paralelo**

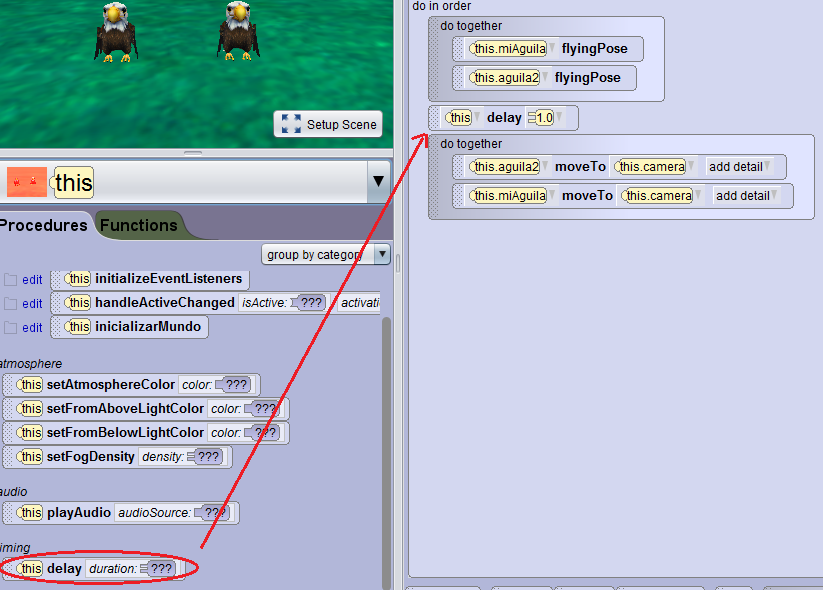
Por defecto, Alice coloca las instrucciones para que se ejecuten secuencialmente (una después de la otra). Esto es porque todo el método contiene un bloque denominado “do in order”(hacer en orden). Pero en ocasiones es necesario que se no ejecuten las instrucciones en orden, sino en paralelo (a la vez). Para esto, se debe agregar un bloque llamado “do together” (hacer juntos). Todas las instrucciones que se agreguen en ese bloque, se ejecutaran paralelamente. Por ejemplo: se necesita que 2 águilas adopten la posición de volar y se muevan hacia la cámara (para dar la impresión de que vuelan):



**14 instrucciones de control**

**14.1 delay**

Cuando las instrucciones se ejecutan secuencialmente es posible necesitar que haya un retraso entre la finalización de una y el inicio de la otra. A eso se le denomina “delay” (o retardo). El retardo se estable en valores de segundos o fracciones de segundos. Por ejemplo, después de adoptar la posición de vuelo las águilas esperan 1 segundo para moverse:

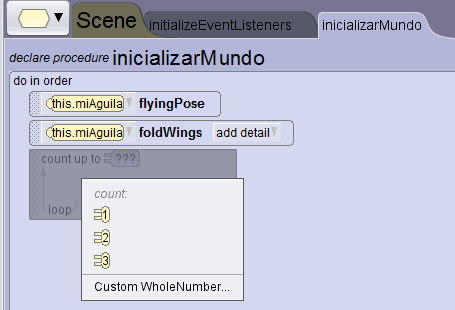


14.2 ciclos

Algunos métodos deben ejecutarse varias veces para lograr un efecto deseado. Por ejemplo, se necesita que un águila abra y cierre las alas varias veces:



Una solución sería repetir el par de métodos varias veces copian y pegando. La otra opción es usar instrucciones repetitivas o ciclos. Es posible conocer de antemano cuantas veces es necesario repetir los métodos. En ese caso, se utiliza una instrucción llamada “count”. Al agregar la instrucción, se debe indicar cuantas veces se debe repetir el ciclo. Se puede seleccionar alguno de los valores por defecto o colocar un valor personalizado:



Luego se agregan los métodos que se desea se repitan varias veces. Por ejemplo:

14.3 condiciones

15. trabajando con variables

15.1 declaración de variables

15.2 asignaciones

15.3 uso de las variables

16. Respondiendo a Eventos

16.1