# Taller 2
# Creando un  nuevo Modelo en NetLogo

En el taller anterior aprendimos como usar el centro de comandos  y los monitores de agentes para inspeccionar y modificar agentes y hacer que realizaran acciones. Ahora estamos preparados para aprender el verdadero corazón de NetLogo: La pestaña de código.

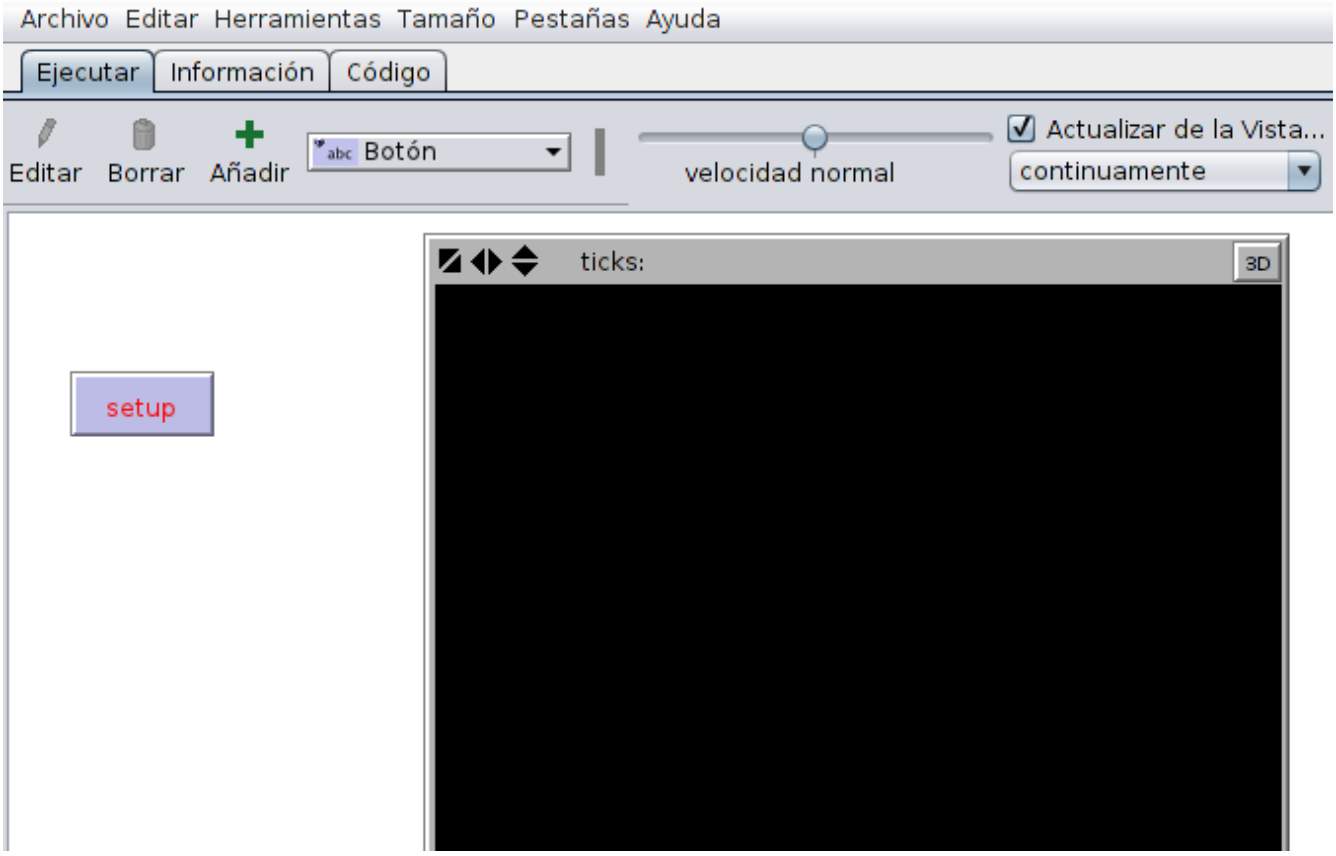
Hemos visto que los agentes en NetLogo se dividen en : parcelas, tortugas, enlaces y el observador. Las parcelas son estáticas y están organizadas en una grilla rectangular, losenlaces conectan a dos tortugas, el observador mira desde afuera lo que está pasando.
Los cuatro tipos de agentes pueden ejecutar comandos de netLogo. Los cuatro pueden realizar lo que se llaman “procedimientos”, Un procedimiento combina una serie de comandos de netLogo en u n solo comando al que se le puede dar un nombre único, aprenderemos entonces a escribir procedimientos que hagan mover, comer , reproducirse y morir  a las tortugas. También aprenderemos como construir deslizadores, monitores y gráficos. El modelo que construiremos es un ecosistema simple.

## Construyendo el botón de setup  (Inicialización)

Para construir un nuevo modelo en NetLogo, seleccione “Nuevo” del menú archivo, los pasos para crear el botón setup son los siguientes:

- Haga clic en el ícono añadir en la parte superior izquierda.
- En el menú a la derecha de añadir seleccione botón (si no está ya selecionado)
- Haga clic donde quiera que el botón aparezca en el área blanca vacia
- Una caja de diálogo aparece, coloque setup en la caja que dice instrucciones.
- Presione el botón ok, la caja de diálogo se cierra.

Tenemos ya el botón setup, al presionar el botón se corre un procedimiento llamado setup, un procedimiento es una secuencis de comandos de NetLogo al cual le asignamos un nombre. Este procedimiento lo definiremos pronto, todavía no lo hemos definido. En este m omneto esta botón se refiere a un procedimiento que no se ha definido, y entonces el botón aparece de color rojo.




Si quiere ver el mensaje de error, haga clic en el botón. Vamos entonces a crear el procedimiento setup.

- Vaya a la pestaña de código
- Teclee lo siguiente:
-

```
to setup
  clear-all
  create-turtles 100 [ setxy random-xcor random-ycor ]
  reset-ticks
end
```

Cuando haya terminado la pestaña de código debe verse así:


Observe que algunas líneas aparecen indentadas, para muchos esto hace más fácil la lectura y el entendimiento de los procedimientos.
El procedimiento comienza con la palabra `to` y finaliza con la palabra `end` todo procedimiento comienza y termina con estas palabras, miremos que hace cada una de las lineas del procedimiento que acabamos de definir:

to setup : define un procedimiento llamado setup
clear-all : limpia el mundo, todas las tortugas y parcelas que pudieran existir en el mundo desaparecen, basicamente prepara todo para un nuevo modelo.
Create-turtles 100 : crea cien tortugas
setxy random- xcor random-ycor : crea las tortugas en posiciones aleatorias (random) en la pantalla.
reset- ticks : inicializa el contador de ticks
end : completa la definición del procedimiento

Cuando termine de teclear el procedimiento ( o copiar y pegarlo de este documento!!!), seleccione la pestaña ejecutar y haga clic en el botón de setup, en este momento verá a cien tortugas de diferentes colores esparcidas en el mundo.


Presione setup un par de veces más y observe como la distribución de tortugas es diferente cada vez, observe que algunas tortugas pueden estar en la misma posición montadas una a la otra.
Piense por un momento en que hizo ud para que esto pasara, necesito hacer un botón en la interface y hacer un procedimiento que el botón usa. El botón solo funciona cuando ha completado estos dos pasos separados (crear el botón en un sitio y colocar el código en otro sitio). En lo que sigue de este taller, usted tendrá que hacer algo parecido para añadirle más funcionalidad al modelo.

## Cambiando el Visor de ticks

Los ticks son la unidad de tiempo de los modelos. El contador de ticks (el cual se inicializa con reset-ticks) es útil porque me va dando la percepción del tiempo del modelo, de hecho se puede hacer que el modelo se actualice cada vez que avance un tick. ( en lugar de que el modelo se actualice continuamente), para ello:

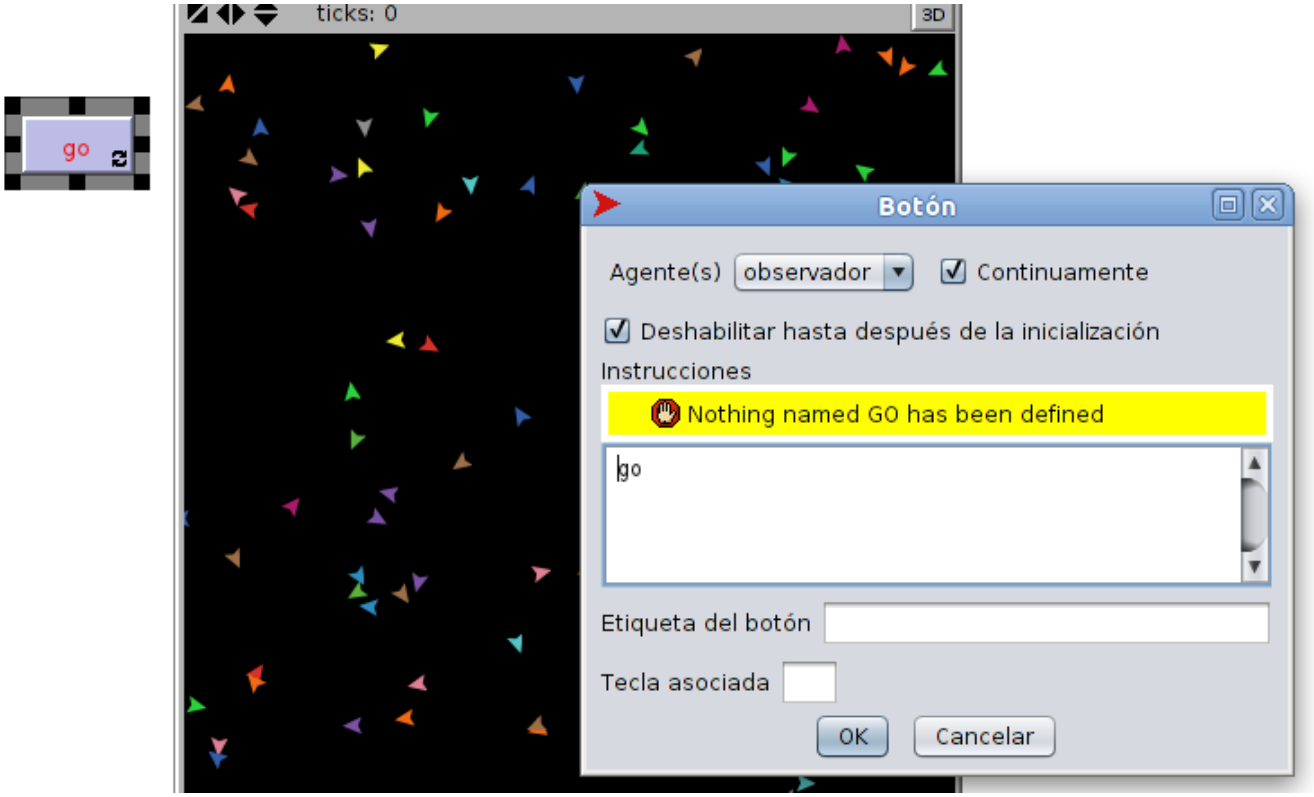
- Encuentre el menú Actualizar de la vista...(que por defecto está en la opción “continuamente”) .
- Seleccione la opción manualmente (ticks).

Esto hace que nuestro modelo se ejecute más rápido y tenga una apariencia más consistente (ya que las actualicaciones del mundo se realizan a tiempos consistentes, ticks).

## Construyendo el botón go

Ahora construiremos el botón go. Sigamos los mismos pasos para construir el botón go, excepto :

- en la caja Instrucciones coloque go
- Habilite  la opción continuamente
- Habilite  la opción “deshabilitar hasta después de la inicialización”




La opción continuamente hace que el botón  quede espichado  una vez oprimido, de tal manera que sus comandos **se ejecutan una y otra vez y no solo una vez.**
La opción deshabilitar hasta después de la inicialización previene que por accidente se oprima el botón go antes del botón setup.
Coloquemos ahora el código al botón go

Vaya a la pestaña código y debajo del código de setup coloque lo siguiente:

```
to go
  move-turtles
  tick
end
```

tick avanza una unidad de tiempo el mundo.

Pero que es move-turtles? Es una función predefinida de NetLogo? No es un  procedimiento que en

este momento vamos a definir. Vamos a definirle a cada tortuga como debe moverse cada vez que avanza una unidad de tiempo (tick), añada los siguientes procedimientos después del procedimiento go:

to go

```
  move-turtles
  tick
end

to move-turtles
  ask turtles [
    right random 360
    forward 1
  ]
end
```

Observe que alrededor del guión de `move-turtles` no hay espacios, en el taller anterior usamos red – 2, para restar dos números y loos espacios fueron necesarios, en este caso move-turtles es el nombre del procedimiento que mueva a las tortugas y com es un solo nombre no colocamos espacios.

Miremos que hace cada comando de move-turtles:

`ask turtles […]` : le dice a cada tortuga que realice los comandos que están dentro de los paréntesis.

`right random 360`: cada tortuga selecciona un número al azar entre 0 y 360 y luego gira ese número de grados en su puesto antes de moverse.

`forward 1` : avanza un paso en la dirección en que está apuntando.

Por qué no escribimos estos comandos todos en un mismo procedimiento? Lo pudimos haber hecho, pero en el proceso de construir nuestro modelo, es factible que debamos colocar más partes al procedimiento go, y entonces necesitamos que el procedimiento `go` sea lo más sencillo posible, para que sea más entendible.Eventualmente, va a incluir más cosas que queremos que pase cuando corremaos nuestro modelo, como hacer unos cáclculos o dibujar unas gráficas, cada uno de estos procesos tendrá un procedimiento asociado y cada procedimiento tendrá un nombre único.

El botón `go` es un botón “eterno”, esto quiere decir que al oprimirlo ejecutará sus comandos para siempre una y otra vez, avanzando el número de ticks , por lo menos hasta que oprimamos el botón de nuevo para detenerlo.

La idea para ejecutar un modelo (esto ya lo vimos en el taller de observación de modelos) es oprimir el botón `setup` que inicializa el mundo y luego el botón `go` para observar en el tiempo el modelo.

Si observa con dertenimiento el modelo puede obseervar que si una tortuga llega al extremo izquierdo del mundo y se mueve a la izquierda en lugar de desaparecer aparece en el lado derecho del mundo, de igual manera si sube por encima de  la parte superior, reaparece en la parte inferior de nuevo, esto es la manera estandar de comportamiento de los bordes del mundo, pero puede ser cambiada y que poe ejemplo estas tortugas desaparezacan del modelo y no vuelvan a aparecer.

## Experimentando con Comandos

Le sugerimos en este momento, experimentar con algunos comandos . Vaya a la terminal de Instrucciones y teclle comandos como:

tortugas > `set color red`

ó

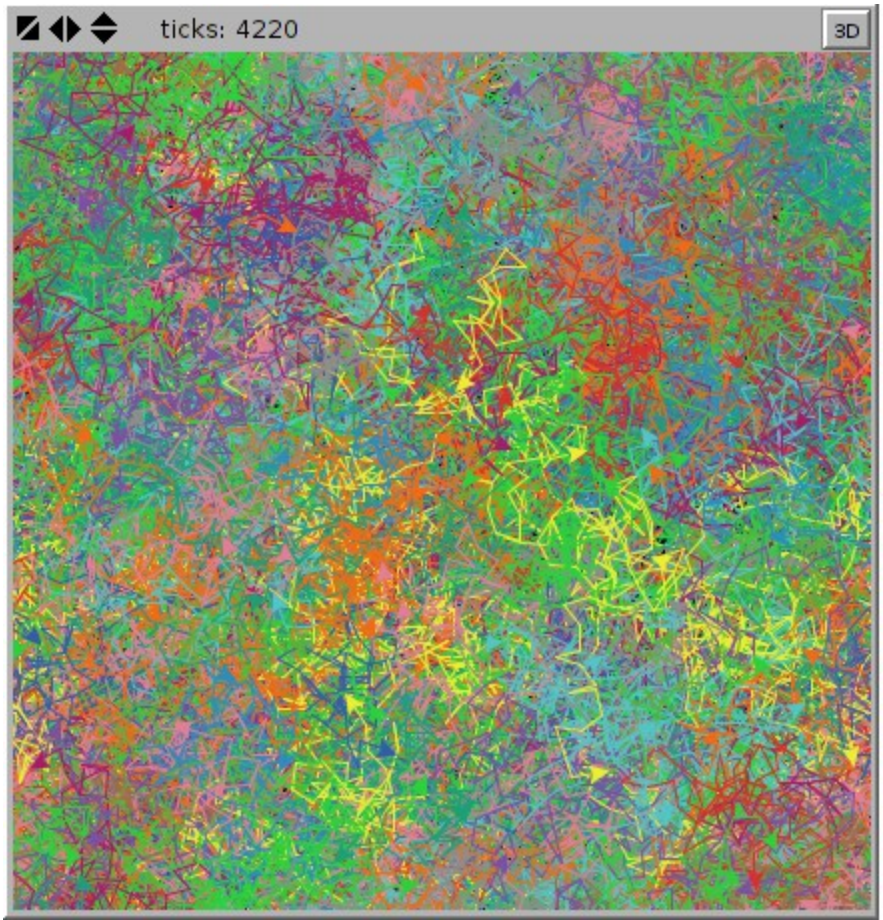
observador > `move-turtles`

observador >  `setup`

observador > `go`

Observe que cuando teclea comandos debe seleccionar el agente al cual quiere dirigir los comandos (ene este caso observador o tortugas).

Teclee:




tortugas > `pendown`

y luego oprima el botón go y observe lo que pasa.

El mundo se llena de tinta de diferentes colores!!!!!!!!!!

Ensaye ahora cambiar dentro de la instrucción move-turtles:

`right random 360` por `right random 45` y observe que pasa.

Ensaye y juegue, es fácil y l**os resultados son inmediatemente visibles**. Esta es una de las grandes fortaslezas de NetLogo. Cuando haya jugado lo suficiente puede continuar a mejorar el modelo que estamos construyendo.

## Parcelas y variables

En este momento tenemos 100 tortugas moviendose al azar en su mundo, totlamente inconscientes de cualquier cosa que exista alreddedor de ellas.Hagamos las cosas un poco mpas interesantes y demosle a las tortugas un fondo ( o piso) en el cual moverse.

- Vaya a la pestaña de código al procedimiento setup y reescribalo de la siguiente manera (puede usar también copiar y pegar si no quiere escribir).

```
to setup
  clear-all
  setup-patches
  setup-turtles
  reset-ticks
end
```

- La nueva definición de setup define dos nuevos procedimientos, para definir set-patches coloque el siguiente código:

```
to setup-patches
  ask patches [ set pcolor green ]
end
```

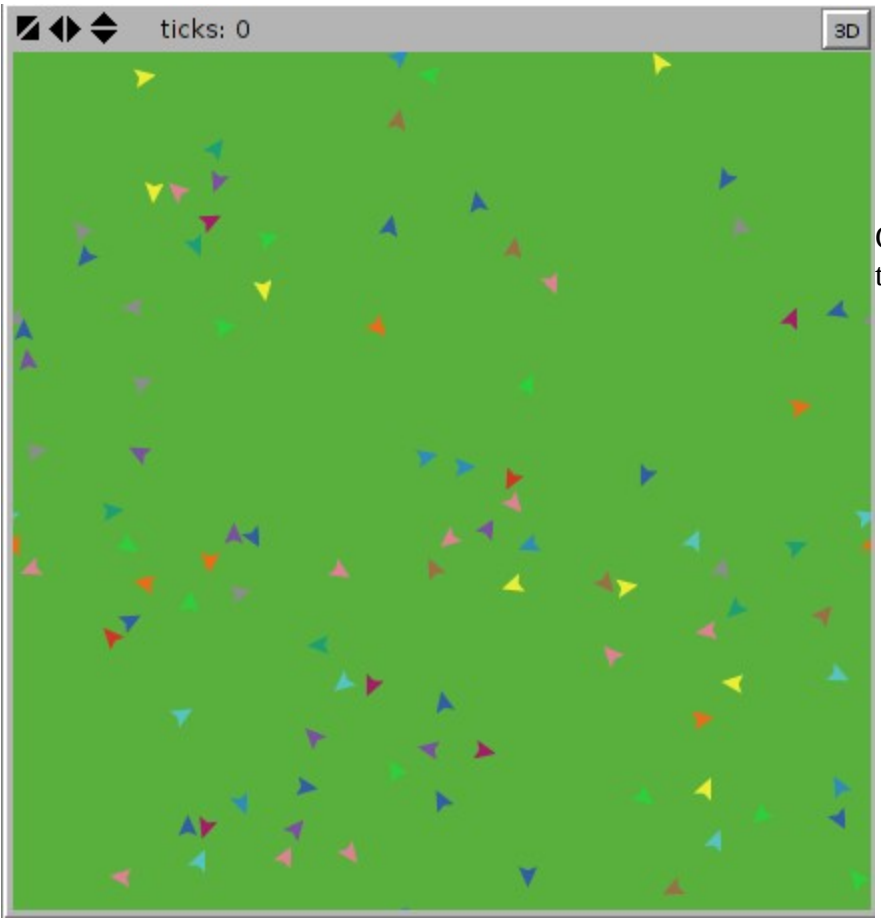
Este procedimiento da el color verde a cada parcela del mundo (recuerde que el color de la tortugas es `color` y el del las parcelas es `pcolor`)

Lo único que falta de nuestro nuevo procedimiento `setup` es el nuevo `setup-turtles:`

```
to setup-turtles

  create-turtles 100
  ask turtles [ setxy random-xcor random-ycor ]
end
```

observe que el nuevo procedimiento para las tortugas es el mismo anterior solo que lo estamos colocando aparte de `setup` para mayor claridad.




- Vaya de nuevo a la pestaña ejecutar
- oprima el botón `setup`

Guauuu!!! un nuevo mundo para nuestras tortugas, ya tienen parcelas verdes.

En este momento puede oprimir el botón setup varias veces y por favor **revise con cuidado los procedimientos definidos hasta entenderlos completamente.**

## Variables de Tortuga

Tenemos entonces algunas tortugas caminando sobre parcelas verdes, pero no están interactuando con las parcelas, añadamos interaccipon entre las tortugas y las parcelas.
Haremos que las tortugas coman pasto (“parcelas verdes”), se reproduzcan y mueran. El pasto crecerá gradualmente a medida que es consumido.
Necesitamos una manera de controlar cuando una tortuga se reproduce y muere, determinaremos esto sabiendo cuanta “energía” tiene una tortuga en cada momento (tick), para ello definiremos una variable de tortuga llamada energía. Observe que ya hemos visto variables de tortuga (como por ejem plo la variable color) la diferencia es que color es una variable predefinida por netLogo y energía va a ser una variable definida por nosotros.
Para crear una variable definida por nosotros se debe colocar en la primera línea del código el comndo: `turtles-own,` llamemos a la variable **energy:**

```
turtles-own [energy]
```

Usemos esta nueva variable para permitir a las tortugas come

- Vaya a la pestaña código.
- Reescriba el procedimiento go:

```
to go
  move-turtles
  eat-grass
  tick
end
```

- Añada un nuevo procedimiento `eat-grass`:

```
to eat-grass

  ask turtles [
    if pcolor = green [
      set pcolor black
      set energy energy + 10
    ]
  ]
end
```

Estamos usando el comando `if` de NetLogo por primera vez. Observe el procedimiento `eat-grass` con atención. Cada tortuga cuando vaya a correr este comando, compara el color de la parcela donde ella esta parada con el color verde ( una tortuga puede examinar las variables de la parcela donde está parada). Si el color de la parcela es verde , la comparación es verdadera y en ese caso realiza los comandos que están dentro del paréntesis ( de otra manera salta estos comandos). Estos comandos entre paréntesis cambian el color de la parcela a negro y aumentan en 10 la energía de la tortuga. Que la parcela quede en negro significa que el pasto de la parcela fue consumido y entonces debido a eso se le aumenta la energía a la tortuga.

Ahora hagamos que el movimiento de la tortuga le reduzca la energía

- Reescriba `move-turtles de` la siguiente manera:

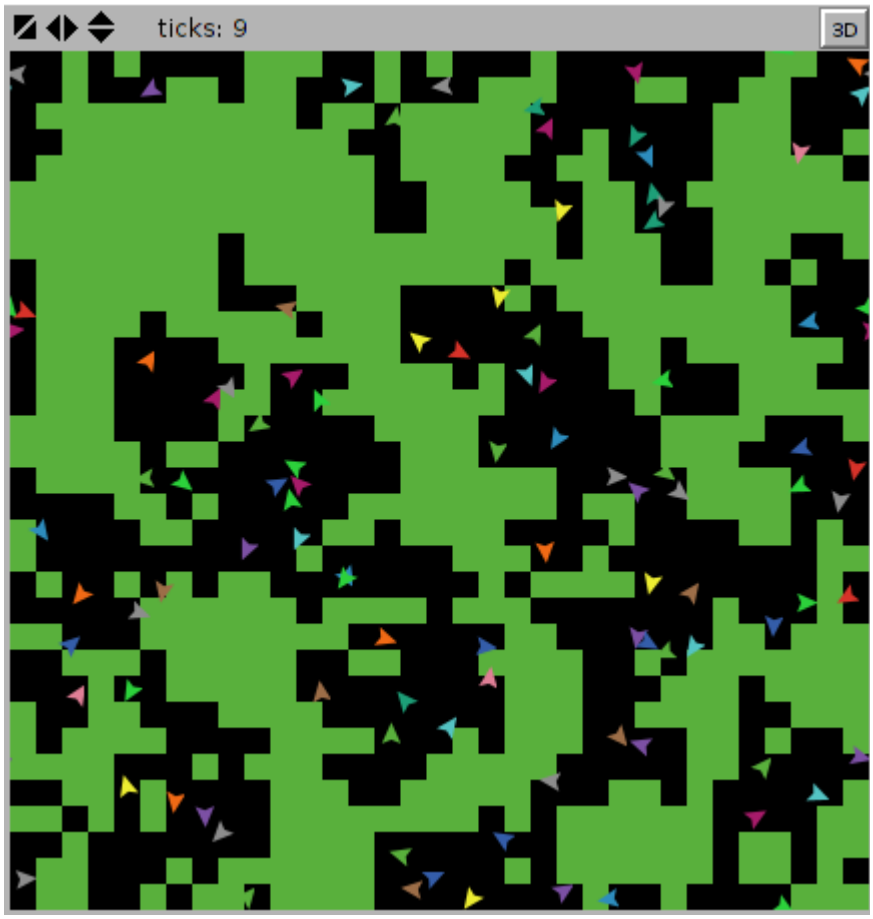
```
to move-turtles

  ask turtles [
    right random 360
    forward 1
    set energy energy - 1
  ]
end
```

A medida que la tortuga avanza un paso pierde una unidad de energía, coloque estos procedimientos en la pestaña de código si no lo ha hecho (algunos tendrá que reescribirlos otros son nuevos).

- Vaya a la pestaña ejecutar y oprima los botones `setup` y `go.`

Verá las parcelas convertirse en negro a medida que las tortugas las recorren .




## Monitores

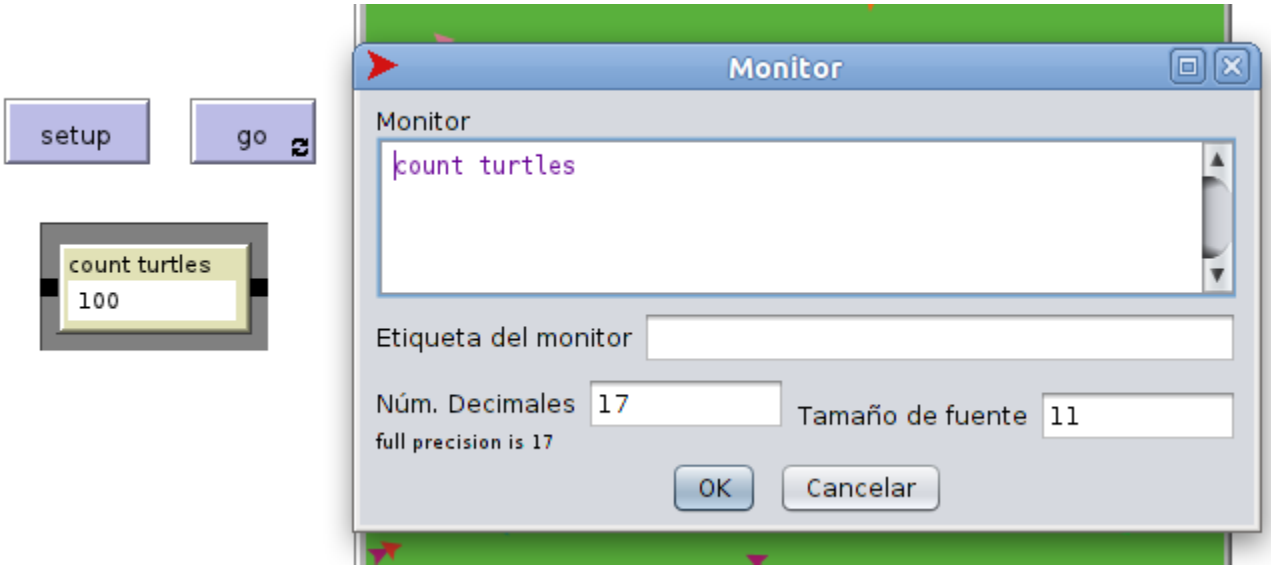
Vamos a crear ahora dos monitores para observar variables dinámicamente, estos monitores se colocan en la pestaña ejecutar, puede ser debajo de los botones. Los monitores se crean de manera parecida a los botones.

- Cree un monitor haciendo clic en el botón añadir , seleccionando del menu de despliegue la opción monitor y haciendo clic en el lugar donde quiere colocarlo en la aprte blanca.

**Aparece un cuadro de diálogo**

- En la caja Monitor coloque `count turtles` (Vea la figura)
- Oprima el botón Ok para cerrar el diálogo

turtles es un conjunto de agentes, es el conjunto de todas las tortugas, el comando count cuenta el número de tortugas que hay en el conjunto.

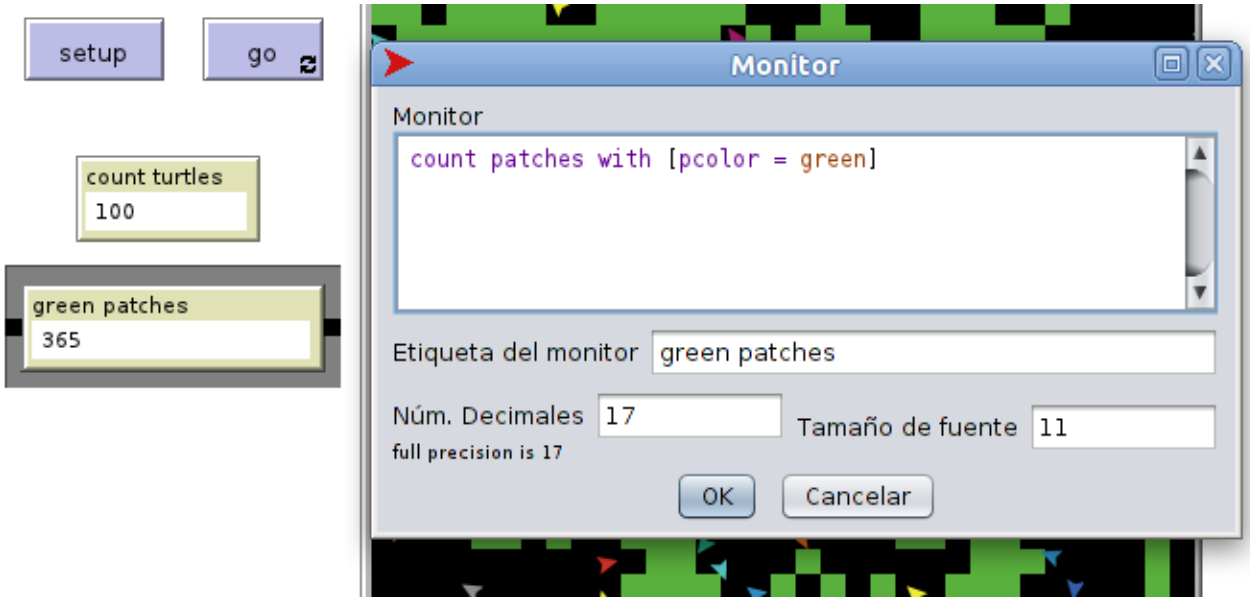



Construyamos ahora el segundo monitor:

- Cree un monitor haciendo clic en el botón añadir selección ando en el menú monitor y luego haciendo clic en un lugar libre de la interfaz (puede ser al lado del monitor que acaba de construir).

**Aparece un cuadro de diálogo**

- En la caja Monitor coloque `count patches with [pcolor = green]` (Vea la figura)
- En la caja Etiqueta del monitor coloque: `green patches`
- Oprima el botón Ok para cerrar el diálogo.



Acá estamos usando `count` otra vez para ver cuantos agentes en el conjunto de agentes parcelas tienen color verde.No queremos las parcelas totales sino sololas verdes por eso usamos el comando `with`, para seleccionar las parcelas que cumplen la condición `pcolor=green`.

Ahora tenemos dos monitores que nos reportan cuantas tortugas y cuantas parcelas hay en todo momento.

- Use el botón `setup` y el `go` y observe como cambian de valor los monitores. (observe que el monitor de tortugas no cambia y el de parcelas si).

## Interruptores y avisos

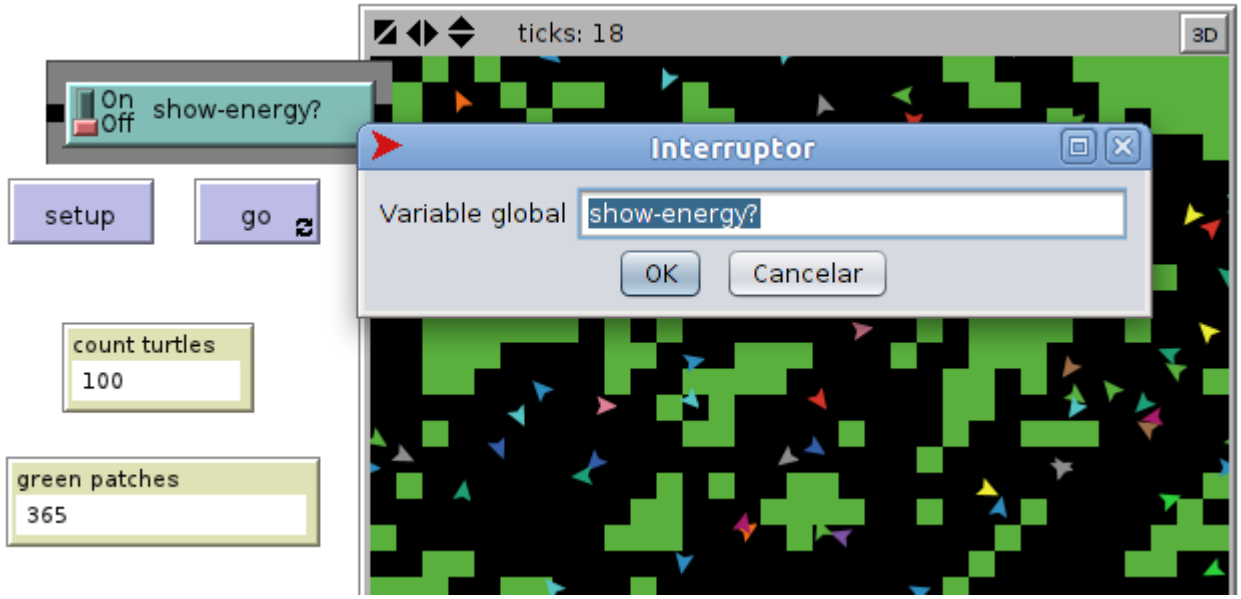
Las tortugas no solo comsn pasto, también ganan y pierden energía.A media que el modelo avanza en el tiempo se podría tener un monitor para observar el cambio de energía.

Pero sería mejor si podemos ver los cambios individuales de energía de cada tortuga todo el tiempo, eso es lo que vamos a hacer, añadiremos un interruptor para que nos permita mirar o no la energía de cada tortuga cuando queramos.

- Haga clic en el botón añadir.
- Seleccione interruptor del menú de despliegue
- Haga clic en un espacio libre de la interfaz.

**Aparece un cuadro de diálogo**

- En el campo variable global, teclee `show-energy?`. No olvide colocar el signo de interrogación al final (vea al figura)



- Vayamos ahora  a la pestaña de código , al procedimiento `eat-grass` y  reescribámoslo:

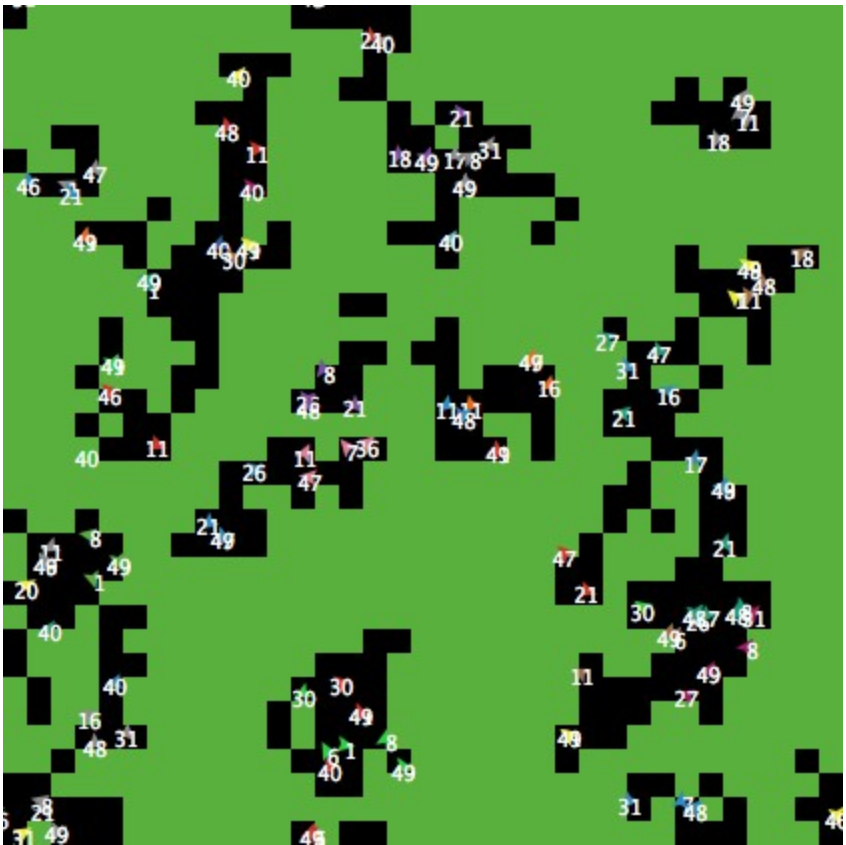
```
to eat-grass

  ask turtles [
    if pcolor = green [
      set pcolor black
      set energy energy + 10
    ]
  ifelse show-energy?
    [ set label energy ]
    [ set label "" ]
  ]
end
```

El procedimiento `eat-grass` usa el comando `ifelse` . Mire con atención el código. Cada tortuga cuando ejecuta esta comando revisa el valor de `show-energy?` .  Si el interruptor está prendido la tortuga ejecuta los comandos dentro del paréntesis, en este caso activa su aviso (label) de energía. Si el interruptor está apagado la tortuga realiza los comandos dentro del segundo parén tesis en este caso coloca como aviso “” que es un aviso vacio.

- Pruebe los avisois de energía de cada tortuga corriendo el modelo y colocando el interruptor de energía en on y en off.

Cuando el interruptor está en on, ud verá al lado de cada tortuga su nivel de energía, observe que cada vez que una tortuga come pasto aumenta su energía y cada vez que se mueve su energía disminuye.



## Más Procedimientos

Nuestras tortugas ya están comiendo, ahora hagamos que se reproduzcan y mueran. Y también hagamos que el pasto vuelva a crecer. Añadiremos cada uno de estos comportamientos en tres procedimientos separados:

- Vaya a la pestaña de código.
- Reescriba el procedimiento go de la siguiente manera:

to go

```
  move-turtles
  eat-grass
  reproduce
  check-death
  regrow-grass
  tick
end
```

- Añada los procedimientos `reproduce`, `check-death y regrow-grass:`

```
to reproduce

  ask turtles [
    if energy > 50 [
      set energy energy - 50
      hatch 1 [ set energy 50 ]
    ]
  ]
end

to check-death
  ask turtles [
```

```
    if energy <= 0 [ die ]
  ]
end

to regrow-grass
  ask patches [
    if random 100 < 3 [ set pcolor green ]
  ]
end
```

Cada uno de estos procedimientos usa el comando if . Cada tortuga cuando ejecuta `check-death` revisa si su energía es menor o igual a 0. Si esto es verdadero muere (die).

Cuando cada tortuga corre el procedimiento `reproduce`, revisa el valor de la variable energy, si la energía es mayor que 50 , entonces crea (`hatch`) una nueva tortuga con una energía inicial de 50

Cunado cada parcela ejecuta el comando `regrow-grass` mira si un número aleatorio entre 0 y 99 es menor que 3, esto en cristiano es que el comando `set patch green` se ejecuta 3 de cada 100 veces.

- Vaya a la pestaña ejecutar y presione el botón de setup

Verá ahora un interesante comportamiento, las tortugas mueren y nacen y el pasto vuelve a crecer. Si observa los monitores verá que ambos fluctuan. Es el patrón de estos monitores predecible? Existe una relación entre estas variables?

Sería muy chévere si tuvieramos una manera más sencilla de observar los cambios en nuestro modelo a lo largo del tiempo. Si la tenemos, podemos construir sin el menor esfuerzo gráficas de variables y observar su comportamiento a lo largo el tiempo, este será nuestro siguiente paso.

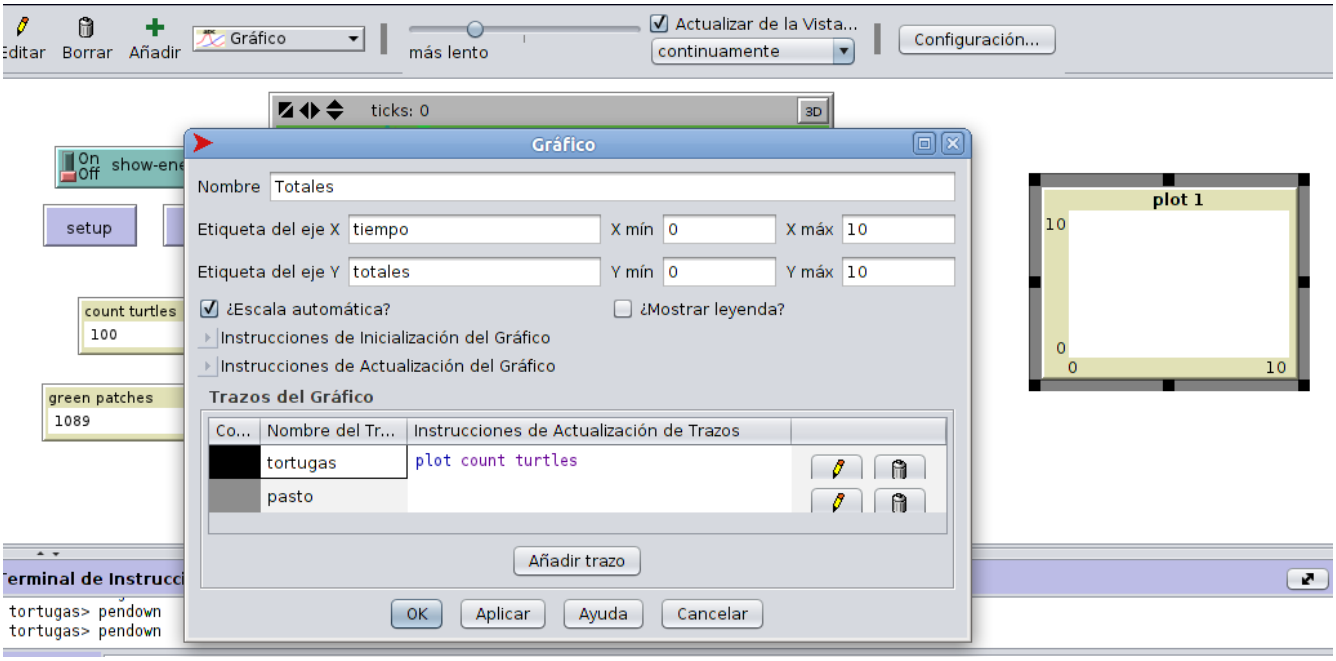
## Gráficas

Para poder definir gráficas en NetLogo, necesitamos crear una gráfica en la interfaz de nuestro modelo.

Los comandos que colocaremos en las gr´aficas arancarán automáticamente cuando oprimamos el botón setup y oprimamos el botón go.

- Haga clic en el botón añadir, seleccione gráfica en el menú de despliegue y haga clic en un espacio vacío de la interfaz.
- Colóquele como nombre `Totales` (ver la imagen)
- Cóloque `tiempo` en la etiqueta del eje X
- Coloque `totales` en la etiqueta del eje Y
- Cambie el nombre del trazo (default) por `tortugas`
- Coloque `plot count turtles` en Instrucciones de actualización de trazos.

- Oprima el botón añadir trazo
- Cambie el nombre del nuevo trazo a `pasto`
- Colque `plot count patches with [pcolor = green]` en Instrucciones de actualización de trazos.



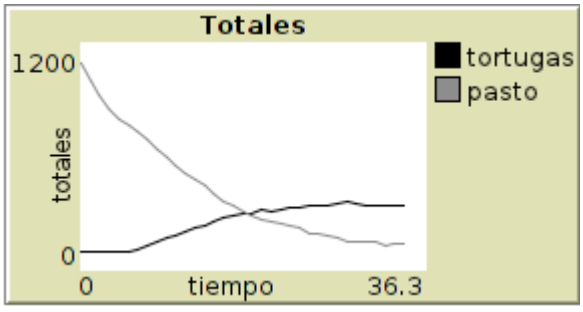

- Presione Ok en la caja de diálogo para terminar la edición.

Observe que cuando ud crea un gráfico puede definir también los valores mínimos y máximos de las variables en el eje X e Y. Si mantiene chequeado la opción de ¿Escala Automática? Cad avez que la graica exceda los valores mínimo y m áximo **los ejes crecen dinámicamente** para que se puedan ver todos los datos.

- Presione el botón `setup` y vuelva a correr el modelo.

Puede observar copmo se van dibujando las gráficas a medida que el modelo va avanzando en el tiempo, recuerde que si está chequeada la opción ¿Escala Automática? La gráfica se va ajustando dinámicamente a los datos.

Si se le olvido que color corresponde a cada variable, puede chequear la opción ¿Mostrar leyenda? dentro de la caja de diálogo.



Puede correr el sistema varias veces y observar como cambian las gráficas.

## Contador de Ticks

Para poder realizar comparaciones entre diferentes “corridas” de nuestro modelo, es muy útil realizar la comparación para la misma “longitud” del modelo. Aprender como parar o arrancar una acción en un tiempo específico puede ayudar a esto ya que podemos hacer que las diferentes “corridas” paren en el mismo lugar.

Saber cuantas veces se ejecuta el comando go es un camino hacia este objetivo, esto es lo que hace elo contador de ticks.

Ya estamos usando el contador de ticks en nuestro modelo , con los comandos `reset-ticks y tick`, los cuales también son los que se usan para dibujar los puntos de las gráficas (un punto por cada tick).

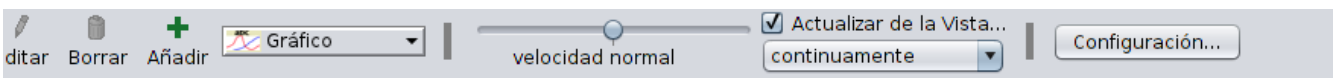
El contador de ticks se puede tambipén usar para limitar el número de corridas del modelo, miremos un ejemplo.

- Cambie el procedim iento go por el siguiente:

to go

```
  if ticks >= 500 [ stop ]
  move-turtles
  eat-grass
  check-death
  reproduce
  regrow-grass
  tick
end
```

- Ponga el modelo en velocidad normal.




- Ahora oprima el botón setup.

El modelo no corre para siempre, para automaticamente en el tick 500.

## Más detalles

En lugar de usar 100 tortugas, podemos manejar un número variable de tortugas

- Hagsa clic en el botón añadir, seleccione del menu la opción deslizador y luego haga clic en un espacio vacio de la interfaz.

**Aparece un cuadro de diálogo**

- En el campo variable global, teclee `número-de-tortugas`
- Deje el valor mínimo en cero y coloque 500 como valor máximo.
- Oprima Ok para cerrar la caja de diálogo
- Ahora vaya a la pestaña de código y cambie elo procedimiento `setup-turtles`

```
to setup-turtles

  create-turtles número-de-tortugas [ setxy random-xcor random-ycor ]
end
```

Pruebe este cambio en el modelo variando el número de tortugas con el deslizador

Ahora creemos otros dos deslizadores para poder variar la energía con la que nace una tortuga y la que gana al comer pasto.

Construya dos deslizadores llamados:

- `energy-from-grass  y birth-energy.`

Dentro de los procedimientos eat-grass y reproduce haga los siguientes cambios:

```
to eat-grass

  ask turtles [
```

```
    if pcolor = green [
      set pcolor black
      set energy (energy + energy-from-grass)
    ]
    ifelse show-energy?
      [ set label energy ]
      [ set label "" ]
  ]
end

to reproduce

  ask turtles [
    if energy > birth-energy [
      set energy energy - birth-energy
      hatch 1 [ set energy birth-energy ]
    ]
  ]
end
```

Pruebe estos dos deslizadores.

## Tarea

¿qué otras modificaciones puede hacerle al sistema?

- ¿Un deslizador para el nacimiento del pasto?
- ¿ Nuevas reglas para el movimiento de las tortugas?
- ¿Nuevas gráficas?

Piense en dos o tres modificaciones, **realizelas** y observe los resultados.