

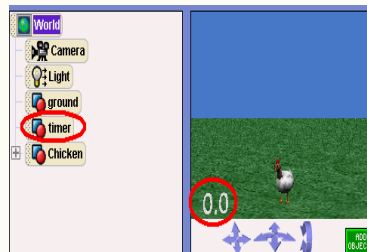
## Timers in Alice 2

Alice



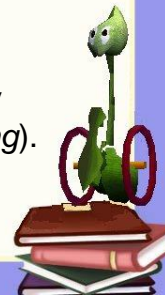
## Example

- Game programs often make use of a timer – some mechanism to keep track of the time remaining in the game.
- We will construct a timer.
- First, add a 3D text object to the world.
  - Any string of digits could be used. We arbitrarily chose "0.0"



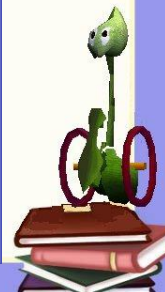
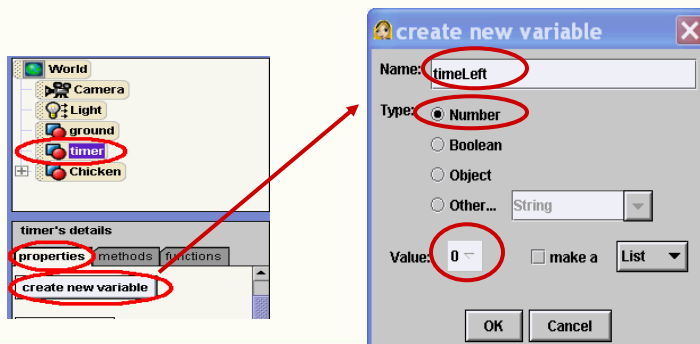
## Demo...things to think of...

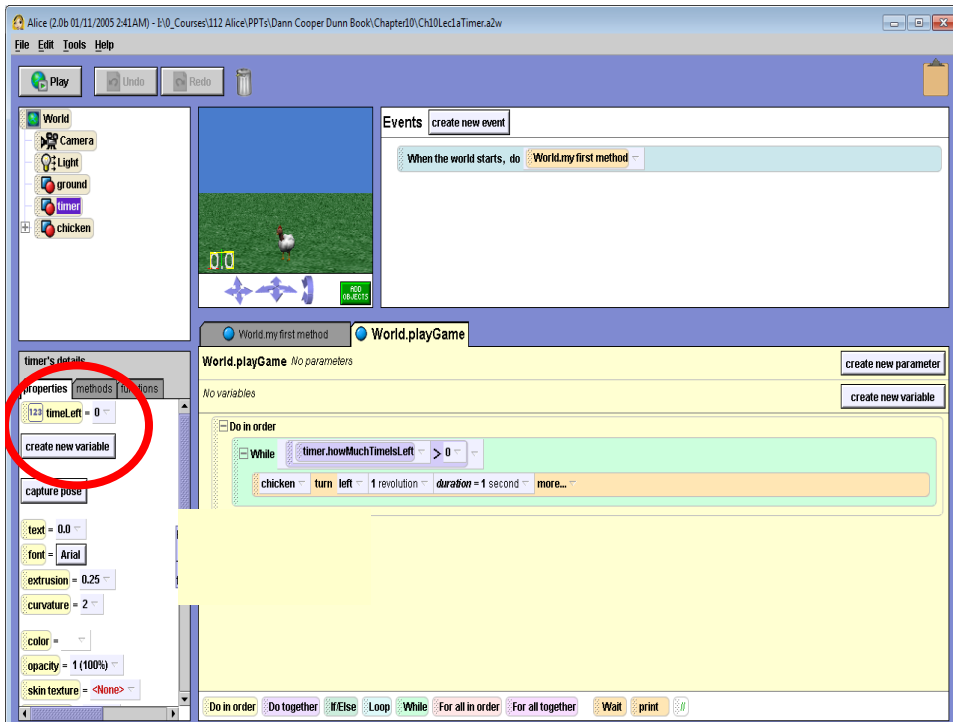
- 🌐 Concepts illustrated in this example
  - The value in a variable can be used in a conditional expression to control a loop.
    - 🔗 The value of *timeLeft* is compared to 0.
  - The value in a variable can be changed at runtime.
    - 🔗 The value of *timeLeft* is decreased by 1 each time through the *While* loop.
  - A number can be displayed as a text string by calling a built-in conversion function (*as a string*).



## Creating a new variable

- 🌐 Select the timer object and then create a new variable named *timeLeft*.
  - The *isOn* variable is **declared** to be a Number type.
  - The value is 0 (the game hasn't begun, as yet).





## Start the timer

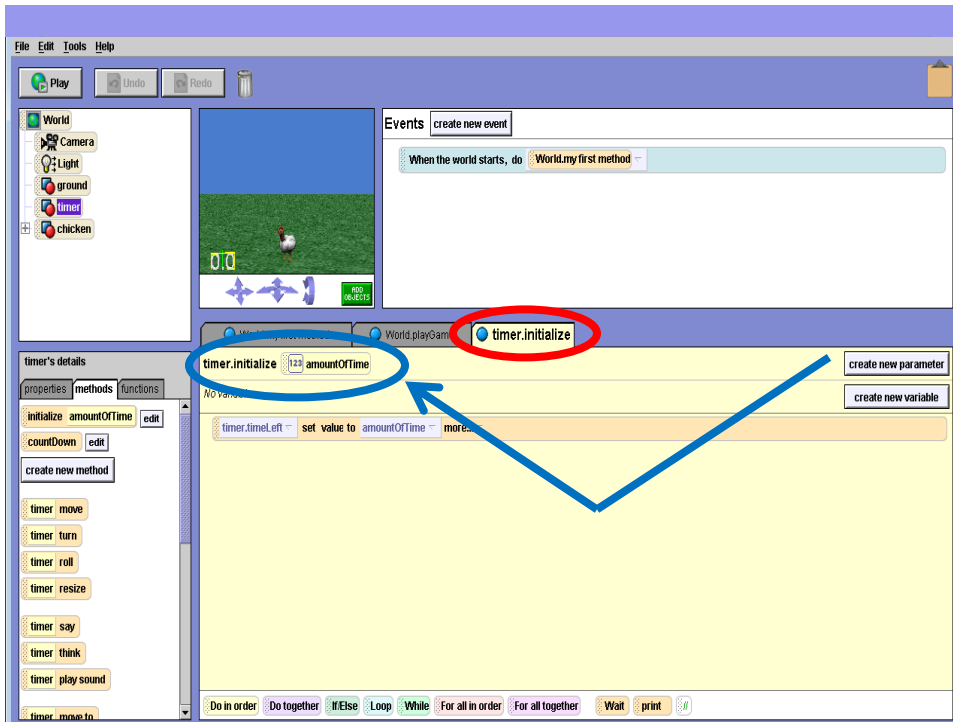
- At the start of the game, the timer's *timeLeft* variable must be **initialized** to the amount of time allowed for playing the game. (This example uses seconds, but minutes would work just as well.)
- Storyboard for a method to initialize the *timeLeft* variable:

*initialize*

**parameter:** *amountOfTime*

*timeLeft* is set to a value specified by *amountOfTime*





## Counting down

- 🌐 In this example, the timer will count down to 0 (no seconds left on the clock).
- 🌐 Storyboard for a **countDown** method:

*countDown*

*Do in order*

While timeLeft is greater than 0

*Do in order*

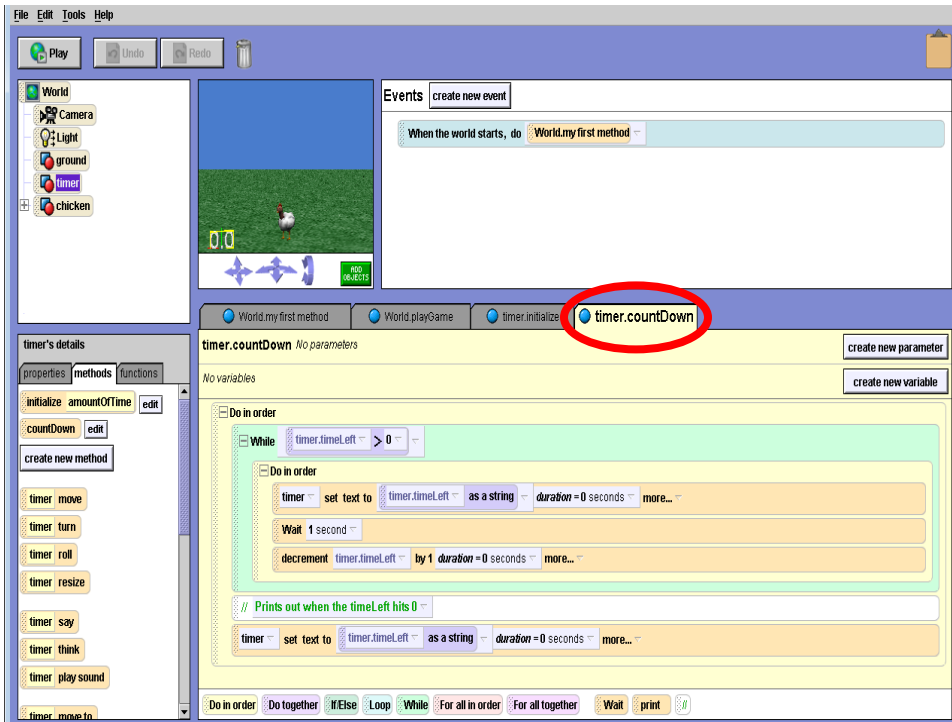
update the 3D text to show the remaining number of seconds

decrease timeLeft by 1

update the 3D text to display 0 seconds

**Note:** The last instruction updates the 3D text display after the **While** loop ends .





## class-level function

- To coordinate the game with the timer's countdown, write a **function** that returns the value of the *timeLeft* variable.

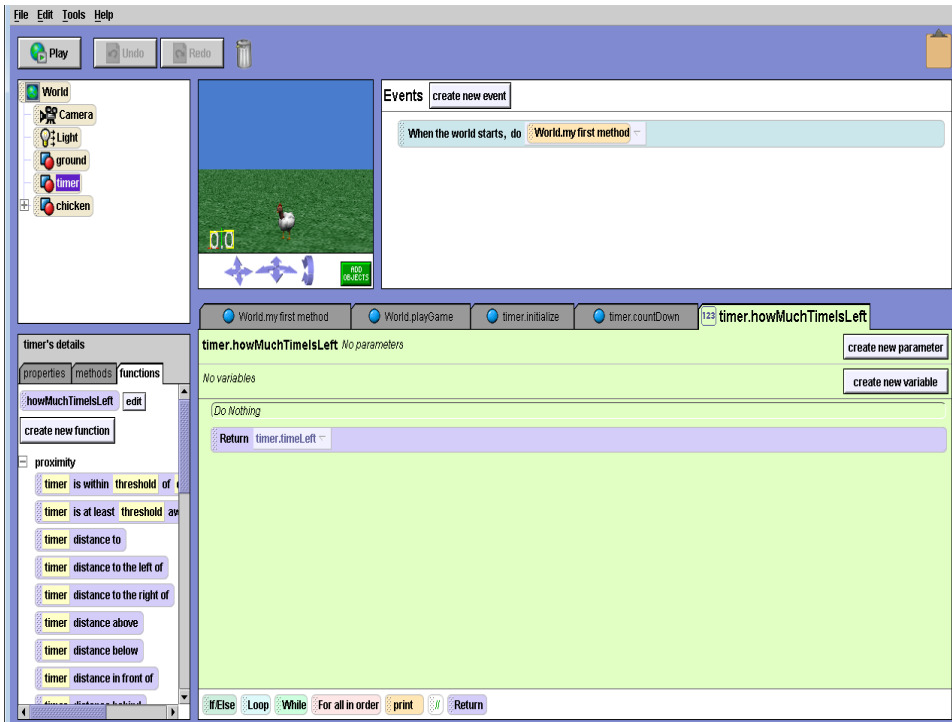
**timer.howMuchTimeIsLeft** *No parameters*

*No variables*

*Do Nothing*

**Return** timer.timeLeft





## Timer Object

- 🌐 We needed to use a variable to keep track of the time left – a numeric value.
- 🌐 The timer object is from the 3-D text class. It does not store a numeric value such as we needed to implement a timer.
- 🌐 We used the timer object to display the current value of the numeric variable.
- 🌐 How?

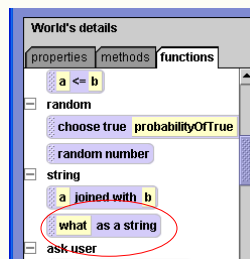


## Using 3D Text to Display Values

- A 3D text object displays a string
- We can change what it displays by using the timer's **text** property as follows:
- Drag the Text tile to the program pane, and choose Default String as a placeholder..



- Then we use the World function *what as a string* to convert the numeric value into text so that it can be displayed by the 3D text object:



The screenshot shows the Scratch IDE with the 'timer.countDown' script selected. On the left, the 'timer's details' panel is open, showing the 'properties' tab. The 'text' property is set to '0.0'. A red circle highlights this value, with a line pointing to the 'set text to default string' block in the script. The script itself is as follows:

```

timer.countDown No parameters
No variables
Do in order
  While timer.timeLeft > 0
    Do in order
      timer set text to default string more...
      Wait 1 second
      decrement timer.timeLeft by 1 duration = 0 seconds more...
    // Prints out when the timeLeft hits 0
  timer set text to timer.timeLeft as a string duration = 0 seconds more...

```

At the bottom right, there is a small illustration of a green alien-like character sitting on a stack of books.

## Resulting in:

The screenshot shows the Scratch IDE with the 'timer.countDown' script selected. On the left, the 'World's details' panel is open, showing the 'properties' tab. The 'what as a string' block is highlighted with a red circle. A red circle also highlights the 'timer' block in the script, with a context menu open over it. The context menu shows the following options:

- the entire World
- my first method
- Camera
- Light
- behavior0
- ground
- timer
- chicken
- playGame
- import sound file...
- expressions
- timer.timeLeft
- timer.howMuchTimeLeft
- timer.timeLeft
- timer.howMuchTimeLeft

The script in the background is the same as in the first screenshot.

At the bottom right, there is a small illustration of a green alien-like character sitting on a stack of books.

# And finally:

The screenshot shows the Alice software interface with the 'timer.countDown' method selected. The 'World's details' panel on the left lists various objects and methods. The main workspace displays the 'timer.countDown' method's code blocks:

- Do in order**
  - While** `timer.timeLeft > 0`
    - Do in order**
      - timer** `set text to timer.timeLeft as a string` (highlighted with a red oval)
      - Wait** `1 second`
      - decrement** `timer.timeLeft by 1 duration = 0 seconds`
  - // Prints out when the timeLeft hits 0**
  - timer** `set text to timer.timeLeft as a string duration = 0 seconds`

The bottom of the workspace shows navigation buttons: `Do in order`, `Do together`, `If/Else`, `Loop`, `While`, `For all in order`, `For all together`, `Wait`, `print`, and `//`.

The screenshot shows the Alice software interface with the 'World.playGame' method selected. The 'World' panel on the left lists objects: Camera, Light, ground, timer, and chicken. The main workspace displays the 'World.playGame' method's code blocks:

- Do in order**
  - While** `timer.howMuchTimesLeft > 0`
    - chicken** `turn left 1 revolution duration = 1 second`

The 'Events' panel on the right shows: `When the world starts, do World.my first method`. The 'World's details' panel on the left shows the 'my first method' and 'playGame' methods with 'edit' buttons. A tooltip for 'create new method' is visible, stating 'Open this method for editing.'

Alice (2.0b 01/11/2005 2:41AM) - E:\0\_Courses\112 Alice\PPTs\Dann Cooper Dunn Book\Chapter10\Ch10Lec1aTimer.a2w

File Edit Tools Help

Play Undo Redo

**World**

- Camera
- Light
- ground
- timer
- chicken

World's details

properties methods functions

my first method edit

playGame edit

create new method

0.0

NO OBJECTS

**Events** create new event

When the world starts, do World.my first method

**World.my first method**

World.my first method No parameters

No variables

timer.initialize amountOfTime = 10

Do together

- World.playGame
- timer.countDown