



## Table of Contents

### Overview

[What is Life?](#)

[Release history](#)

[Bugs and things to do](#)

### Available menus and commands

[File menu](#)

[Options menu](#)

[Shape menu](#)

[Step and Run](#)

[Zoom](#)

[Adding cells and shapes](#)

[Selecting an area](#)

### Technical information

[Sparse population representation](#)

[Shape libraries](#)

## What is Life

The above title is actually the title of chapter 25 of the excellent book by Berlekamp, Conway and Guy, *Winning Ways* (Academic Press, 1982), which will give you much more information than I could give you. The game of **Life** (a no-player game, as Conway says) was invented around 1970 by the famous mathematician Conway, for fun, and turned out much later to be the simplest cellular automaton which can simulate a universal Turing machine.

Much closer to earth, the rules are as follows:


- You start by disposing some cells on a rectangular grid (the cells are represented in the program by some red squares each filling one of the grid squares), and then compute what happens to them at each generation.
- The rules for going from one generation to the next depend just on the immediate neighbourhood of each cell: a cell is adjacent to 8 squares in the grid, which are empty or occupied. Its fate for the next generation depends just on how many of these adjacent squares are occupied (how many *neighbours* the cell has):
  - If a cell has no neighbour or one neighbour, it dies (of loneliness we may presume).
  - If a cell has four neighbours or more, it dies (from overcrowding).
  - A cell survives if it has just 2 or 3 neighbours.
- To make things interesting, if an empty square is surrounded by just three cells, a new cell will be born on that square.

All the above changes are applied simultaneously to compute the next generation. Some simple configurations whose fate you should think about are: 3 in a line, 4 in a square, five in a line.

When you look at the fate of various initial configurations, you will notice that most of the time the configuration degenerates to some collection of *still life*, after having generated a certain number of *gliders*.

Still life is stable configurations. A sample of them are:



A glider is a five-cell configuration  which moves diagonally on square each four generations (its speed is  $c/4$ , on quarter of the speed of light, the fastest possible speed of any interaction in the game).

You should now look how to [add cells and shapes](#), and at the [Run and Step menus](#), to start playing.

## Program history

This program is based on code I had written aeons ago (about 20 years ago, when I was in school -- we had then access to a HP computer with 5K of RAM!) where I invented the technique for computing successive generations of Life with a sparse-array representation (see [sparse population representation](#)). I did that in order to follow what happened to populations occupying very large areas (the technique allows to consider areas up to 32768x32768 cells). I lost interest soon after and all but forgot about it, but recently I decided to make it into a proper windows program for three reasons:

- I wanted to learn to program for windows, and it was an ideal pretext to get started (using Borland's objectwindows, actually, because that seemed easier).
- I discovered the News group comp.theory.cell.automata and noticed that people were still interested in Life, even discovering new things.
- I looked around for any decent DOS or Windows Life program, and did not find any (let me know if I missed something).

I intend to learn as much as possible from this experience, both about Life and programming for Windows. In order to do that, I make the code for this program freely available, so people can make any improvements they like, with one restriction: any modification should be sent to me before being redistributed, so I can decide if I want to incorporate it or not in the next distribution of the program (with a proper acknowledgement for the help, of course). See the **about** menu item in the help menu for my addresses (E-mail and postal).

## Release History

**June 1993:**

version 1.0

## **Bugs and things to do**

Currently the range of the scroll bars is the size of the population plus half the window size. It is only recomputed when doing a zoom out or in. There should be a better way to do it.

When preparing to put a shape down somewhere, the following should occur: when you press the mouse button, the shape appears where it will be, shimmering, and moves with the mouse. When you release the mouse button it is then laid down where indicated (I would do it if I knew how).

## File Menu

The File menu lets you execute the following actions:

### **Clear**

All cells will be erased, and the generation counter reset to 0

### **Erase area**

All cells in selected area will be erased. The generation counter is not reset. A frequent use for this facility is removing errand gliders or other moving objects which threaten to get out of the picture

### **Save as**

Prompts for a file name to which to save all cells. This is equivalent to saving an area exactly large enough to save all living cells.

### **Save area as**

Prompt for a file name to which to save all cells in selected area

### **Exit**

Exit the program.

## Options menu

This lets you set various features.

### **Set color for Set**

Lets you select the color for a cell which has just been born or added.

### **Set color for Reset**

Lets you select the color for a cell which has survived from a previous generation.

### **Set color for Unset**

Lets you select the color given to a square where a cell has died. No track is kept of these squares, so this color vanishes whenever the part of the window where it is repainted.

### **Reset Generation Counter**

Resets to 0 the generation counter (displayed in the title bar).

## Shapes menu item

This menu item pops a window which lets you select shapes to be put on the screen. This window has the following parts:

### **Load library** button.

By default the library **life.lib** in the starting directory is loaded. You can change the repertoire of loaded shapes by loading another library, like **spaceship.lib**.

### **Library scroll list**

This list contains the names of the shapes in the currently loaded shape library, and lets you select one of them.

Below the library scroll list are the following buttons:

### **Load from file**

This loads just one nameless shape from a file. It can be specially useful with the **save as** and **save area as** items in the file menu to do some complicated cutting and pasting.

### **Line**

This selects as the current shape a line of cells whose length you are prompted to enter.

### **Preview area.**

This shows you the currently selected shape. A cross-hair cursor shows where exactly the shape will be laid down in relation to the cursor. It is possible to select a mirror image or a rotated copy of any shape, by using the two buttons **mirror** and **rotate** below the preview area.

Finally, there is an **OK** button. This is to let you select again the same shape without doing any transformation (when you select a shape, it is laid down for just the next mouse click on the life window, and then the selected shape reverses to a single cell; to select again another shape you must do some operation in the shapes window).



## **Step and Run menu items**

These items control going from one generation to the next.

### **Step**

goes forward one generation.

### **Run**

advances generations continuously until you click the mouse or strike a key.

## **Zoom menu items**

These items control at which magnification the cells are displayed, that is how many pixel each cell occupies. The minimum is one pixel per cell, the maximum is a square 25x25 pixels. When cells occupy more than 3x3 pixels, a one-pixel wide border is left blank so you can easily distinguish individual cells.

### **Zoom in**

go to next higher magnification.

### **Zoom out**

go to next lower magnification.

The available magnifications are: 1, 2, 3, 5, 7, 10, 15 and 25.

## **Adding cells and shapes**

To add the currently selected shape (it is a single cell unless you just selected another shape via the Shape menu), just click with the left mouse button where you want to put it. The shape you put is *XORed* with what is already there: that is, if there is a cell already there and a cell in the current shape would be put on the same spot, no cell will be left. This makes it easy to undo the addition of a shape: just click again on the same spot

## Selecting an area.

To select an area, click with the right mouse button, and then move the mouse while keeping this button pressed. You will see a rectangle outlined between the place you clicked and the current position. When you release the right button, this rectangle will remain and represents the *selected area*. This area will remain there until it is used by the Erase area or Save area menu items, or until you select a new area.

## Sparse population representation

One of the reasons I release this program is that I think the algorithm used to compute the next generation has some merit, and I don't know if it is already known. I found it 20 years ago, when I was trying to follow the fate of populations occupying large areas, on a very small computer (an HP computer with 5K of RAM). I found a way to compute quickly the next generation when storing just the co-ordinates of living cells: thus, if I want to follow what happens to 1000 cells on a 30000x30000 grid, I don't need 30000x30000 bits (about 110 Megabytes) but just 1000x2 integers (4K bytes). The computing time for next generation is also just proportional to the number of cells, so it also is very quick for populations occupying large areas. The idea is to sort lexicographically all cells, and to keep a sorted list of neighbours of the current cell while scanning the population. See **cellpop::nextgen()** in **cell.cpp** for details.

The program gives you access to an area of 32768x32768 where to put cells.

## Shape libraries

To explore various configuration, you can store and load them from shape libraries. These are files containing life configurations, with a name in brackets before each of them. When you load such a file, the shapes in it replace those in the library scroll list in the shape window. The default library is life.lib. Another provided library is spacship.lib, which is just the series of excellent articles about spaceships in life by David I. Bell which appeared recently in the news, edited to fit in the library format.

As you can see from these files, the library format is as follows:

- a line beginning with [ should just contain a name between brackets, and appear just before a shape.
- in lines describing a shape, o or O represent an occupied cell, a blank or . an empty space, and \$ ten empty spaces.
- a line beginning by | is a comment line.

