An Active Essay on Evolution, "The Weasel Essay"

Ted Kaehler

Created at Apple Computer's
Advanced Technology Group in 1994.
Reprinted with permission of the author.

VPRI Paper for Historical Context

Viewpoints Research Institute, 1025 Westwood Blvd 2nd flr, Los Angeles, CA 90024   t: (310) 208-0524

Introduction

An Active Essay is an essay in a live medium with some fairly difficult requirements. Besides a textual explanation and images, it contains running simulations. The program for each simulation is present on the page, and can be modified and run by the reader. An Active Essay is a place for experimentation, where a student can debug his understanding by trying different variations of the simulation.

As we tried to create Active Essays in the early 1990's, we struggled to find a system that could handle text, images, and a running simulation. We also needed the pages to contain the code for the simulation, and a way to modify and run the code. It needed to have small file size, so that schools and students could use it, as opposed to an entire Squeak image and virtual machine.

HyperCard(™) from 1987 met most of these criteria. The problem was that scripts are edited in a separate window, and could not be on a card. However, HyperCard has an evaluate command. I wrote a script for the 'Accept' button that took the contents of a text field and installed it as a script. I also made a versioning system that allowed the user to go back to previous version of a script. The result is the An Active Essay on Evolution created in 1994 and shown here.

The essay brings to life Richard Dawkins' challenge to evolve 'Methinks it is like a weasel' with the only guidance being the number of positions that have a correct letter. Here is the passage that inspired this essay.

The Blind Watchmaker
by Richard Dawkins
pp46-48 in the chapter 'Accumulating small change'

...
Hamlet. Do you see yonder cloud that's almost in shape of a camel?
Polonius. By the mass, and 'tis like a camel, indeed.
Hamlet. Methinks it is like a weasel.
Polonius. It is backed like a weasel.
Hamlet. Or like a whale? Very like a whale.

I don't know who it was first pointed out that, given enough time, a monkey bashing away at random on a typewriter could produce all the works of Shakespeare. The operative phrase is, of course, given enough time. Let us limit the task facing our monkey somewhat. Suppose that he has to produce, not the complete works of Shakespeare but just the short sentence 'Methinks it is like a weasel', and we shall make it relatively easy by giving him a typewriter with a restricted keyboard, one with just the 26 (capital) letters, and a space bar. How long will he take to write this one little sentence?

The sentence has 28 characters in it, so let us assume that the monkey has a series of discrete 'tries', each consisting of 28 bashes at the keyboard. If he types the phrase correctly, that is the end of the experiment. If not, we allow him another 'try' of 28 characters. I don't know any monkeys, but fortunately my 11-month old daughter is an experienced randomizing device, and she proved only too eager to step into the role of monkey typist. Here is what she typed **on** the computer:

UMMK JK CDZZ F ZD DSDSKSM
S SS FMCV PU I DDRGLKDXRRDO
RDTE QDWFDVIOY UDSKZWDCCVYT
H CHVY NMGNBAYTDFCCVD D
RCDFYYYRM N DFSKD LD K WDWK
HKAUIZMZI UXDKIDISFUMDKUDXI

She has other important calls on her time, so I was obliged to program the computer to simulate a randomly typing baby or monkey:

WDLDMNLT DTJBKWIRZREZLMQCO P
Y YVMQKZPGJXWVHGLAWFVCHQYOPY
MWR SWTNUXMLCDLEUBXTQHNZVIQF
FU OVAODVYKDGXDEKYVMOGGS VT
HZQZDSFZIHIVPHZPETPWVOVPMZGF
GEWRGZRPBCTPGQMCKHFDBGW ZCCF

And so on and on. It isn't difficult to calculate how long we should reasonably expect to wait for the random computer (or baby or monkey) to type METHINKS IT IS LIKE A WEASEL. Think about the total number of possible phrases of the right length that the monkey or baby or random computer could type. It is the same kind of calculation as we did for haemoglobin, and it produces a similarly large result. There are 27 possible letters (counting 'space' as one letter) in the first position. The chance of the monkey happening to get the first letter-M -right is therefore 1 in 27. The chance of it getting the first two letters — ME - right is the chance of it getting the second letter - E - right (1 in 27) given that it has also got the first letter - M - right, therefore 1/27 x 1/27, which equals 1/729. The chance of it getting the first word - METHINKS - right is 1/27 for each of the 8 letters, therefore (1/27) X (1/27) x (1/27) x (1/27). .., etc. 8 times, or (1/27) to the power 8. The chance of it getting the entire phrase of 28 characters right is (1/27) to the power 28, i.e. (1/27) multiplied by itself 28 times. These are very small odds, about 1 in 10,000 million million million million million million. To put it mildly, the phrase we seek would be a long time coming, to say nothing of the complete works of Shakespeare.

So much for single-step selection of random variation. What about cumulative selection; how much more effective should this be? Very very much more effective, perhaps more so than we at first realize, although it is almost obvious when we reflect further. We again use our computer monkey, but with a crucial difference in its program. It again begins by choosing a random sequence of 28 letters, just as before:

WDLMNLTDTJBKWIRZREZLMQCOP

It now 'breeds from' this random phrase. It duplicates it repeatedly, but with a certain chance of random error - 'mutation' - in the copying. The computer examines the mutant nonsense phrases, the 'progeny' of the original phrase, and chooses the one which, *however slightly,* most resembles the target phrase, METHINKS IT IS LIKE A WEASEL. In this instance the winning phrase of the next 'generation' happened to be:

WDLTMNLTDTJBSWIRZREZLMQCOP

Not an obvious improvement! But the procedure is repeated, again mutant 'progeny' are 'bred from' the phrase, and a new 'winner' is chosen. This goes on, generation after generation. After 10 generations, the phrase chosen for 'breeding' was:

MDLDMNLS ITpSWHRZREZ MECS P

After 20 generations it was:

MELDINLS IT ISWPRKE Z WECSEL

By now, the eye of faith fancies that it **can** see **a resemblance to the** target phrase. By 30 generations there can be **no doubt:**

METHINGS IT ISWLIKE B WECSEL

Generation 40 takes us to within one letter of the target:

METHINKS IT IS LIKE I WEASEL

And the target was finally reached in generation 43. A second run of the computer began with the phrase:

Y YVMQKZPFfXWVHGLAWFVCHQXYOPY,

passed through (again reporting only every tenth generation):

Y YVMQKSPFTXWSHLIKEFV HQYSPY

YETHINKSPITXISHLIKEFA WQYSEY
METHINKS IT ISSLIKE A WEFSEY
METHINKS IT ISBLIKE A WEASES
METHINKS IT ISJLIKE A WEASEO
METHINKS IT IS LIKE A WEASEP

and reached the target phrase in generation 64. In a third run the computer started with:

GEWRGZRPBCTPGQMCKHFDBGW ZCCF

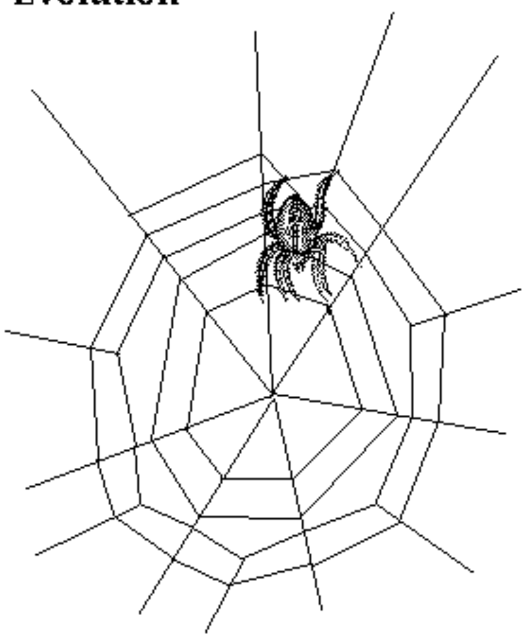
and reached METHINKS IT IS LIKE A WEASEL in 41 generations of selective 'breeding'.

The exact time taken by the computer to reach the target doesn't matter. […] What matters is the difference between the time taken by *cumulative* selection, and the time which the same computer, working flat out at the same rate, would take to reach the target phrase if it were forced to use the other procedure of *single-step selection:* about a million million million million million years. This is more than a million million million times as long as the universe has so far existed. Actually it would be fairer just to say that, in comparison with the time it would take either a monkey or a randomly programmed computer to type our target phrase, the total age of the universe so far is a negligibly small quantity, so small as to be well within the margin of error for this sort of back-of-an-envelope calculation. Whereas the time taken for a computer working randomly but with the constraint of *cumulative selection* to perform the same task is of the same order as humans ordinarily can understand, between 11 seconds and the time it takes to have lunch.

There is a big difference, then, between cumulative selection (in which each improvement, however slight, is used as a basis for future building), and single-step selection (in which each new 'try' is a fresh one). If evolutionary progress had had to rely on single-step selection, it would never have got anywhere.

Evolution Essay 4.0

Evolution



The world is an amazing place. We humans love to feel powerful by building machines that do things. A beautiful jet airliner is evidence of how smart we are and how well we can design things. In spite of our boasting, we really can't do much compared to the plants and animals that are already here.

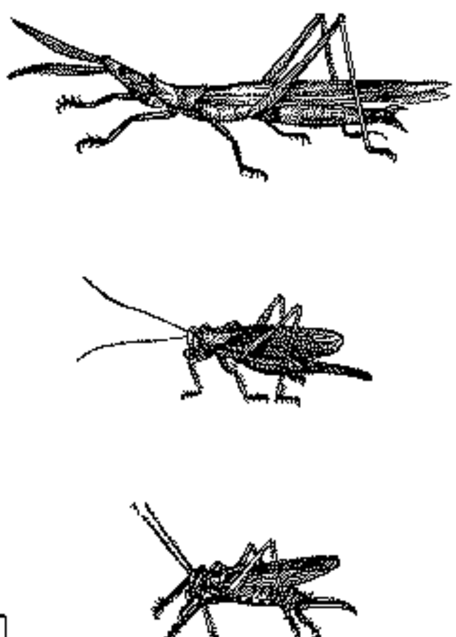
Every day we see plants and animals doing things we cannot do. No person can design and build a spider. It is smaller and more complex than any robot we can build. It can crawl around and build a web. Spider web silk is stronger than steel and can stretch 100 times as much as steel without breaking. Who was the designer who knew enough to create a spider and its silk?

The spider and its silk were not designed by any being. The process of **evolution** created the animals, both their bodies and their habits. Evolution is a very powerful process. Think of the great variety of wild plants and animals on the earth. All of them were created by evolution.

version 4.0, 20 Aug 1994

Evolution Essay 4.0

Variation



Evolution has two parts: **variation** and **selection**. The offspring of an animal is very much like its parent. Cows have baby cows and redwood tree cones sprout into new redwood trees. But offspring are not exactly like their parents. Each individual is a little bit different. There is variation between individuals.

We are used to the outwards signs of variation. People are different heights and have different features in their faces. Every person, except an identical twin, looks a little different from other people. This is a good thing; otherwise you could not tell who you were talking to.

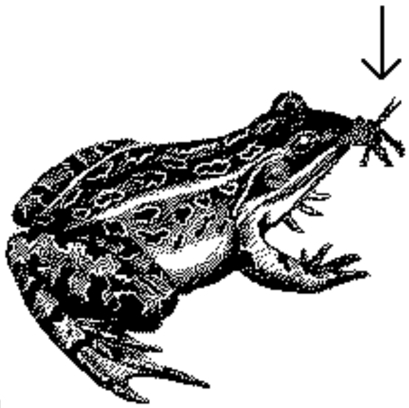
The insides of people vary also. Some people are allergic to cow's milk and some are not. Some people's hearts beat faster than others. Inside cells, on the microscopic level, there are also variations between individuals.

We are talking here about variation caused by an animal's genes. Some differences are caused by what has happened to an animal during its life. It might have a scar from an injury. It might have learned to dig for roots. Evolution works by using genetic variations that an animal is born with. Those variations can be passed on to its children.

Evolution Essay 4.0

Selection

The variation that made this cricket pause before it hopped was not good. The cricket did not live long enough to have offspring.



The second part of evolution is selection. Some animals live to have offspring and some do not. Life is full of dangers, from disease to predators to starvation to falling down a cliff. An animal has to avoid the dangers.

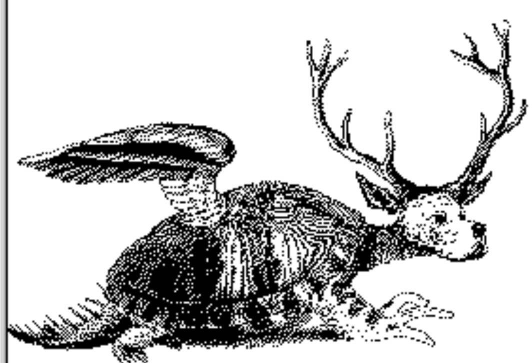
There is a strong element of chance in who survives. An animal may just be in the wrong place at the wrong time and get eaten. But this chance part of survival happens equally to all individuals. What does not happen equally to all is their variation. Over the long run, animals with certain variations will survive better than animals with other variations.

The animals with the successful variations will have more offspring that the others. Those offspring will inherit the variation that the parents had. Over time more of the population will inherit the variation that works.

Variation happens in all directions blindly. It is selection that determines which way evolution goes. Most variation has no effect on survival. It is just different in a harmless way. But some variations help survival and are selected for. Other variations hurt survival and are selected against.

Evolution Essay 4.0

Randomness



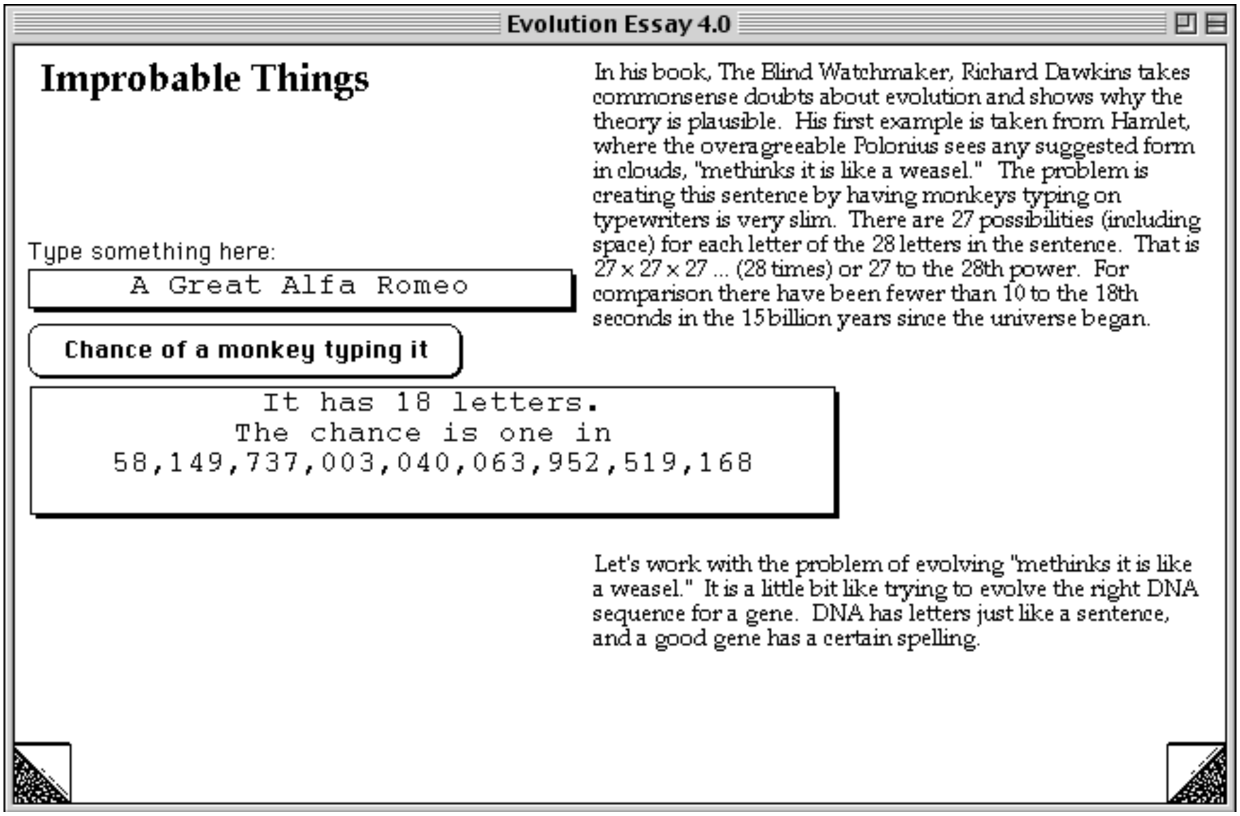
Only a small part of an animal varies in each generation. Most of the animal is the same as its parents. The part that does change, changes randomly.

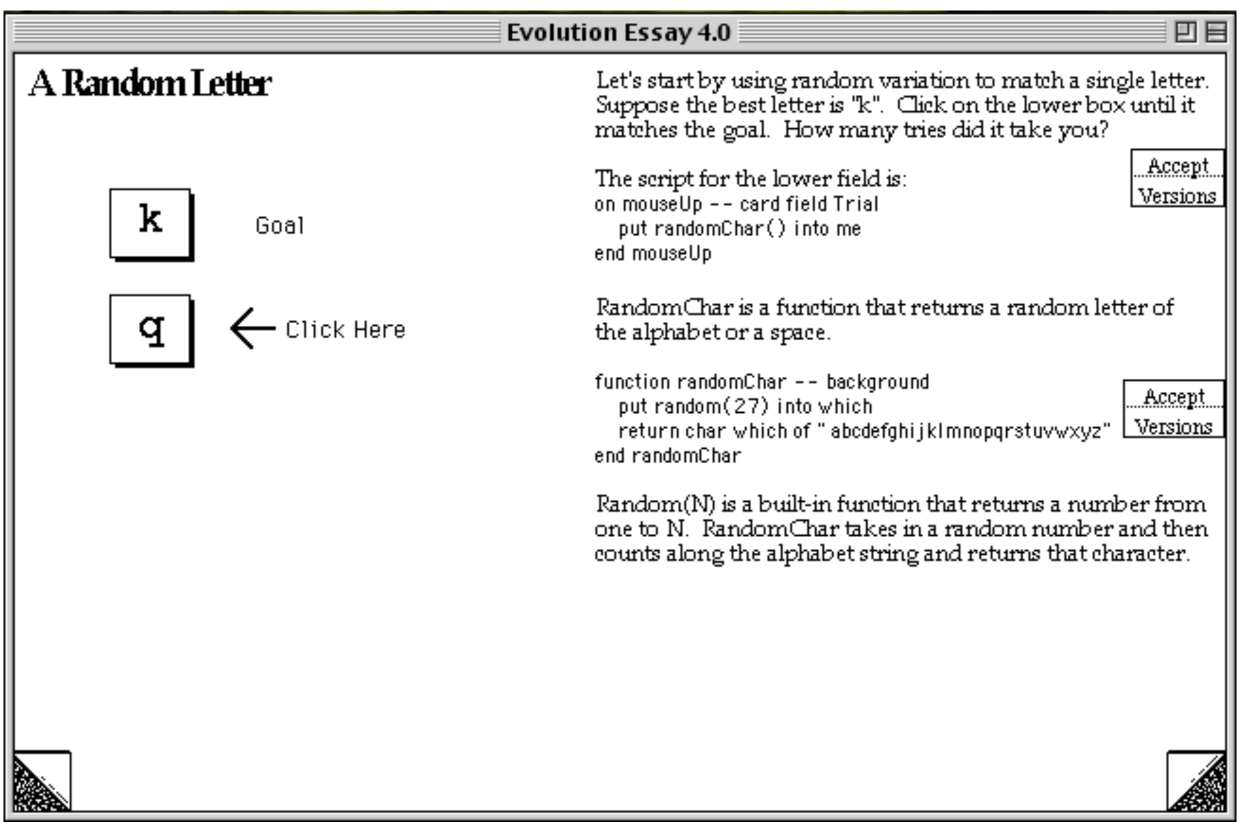
Every part of everything that has ever evolved came from a random event. Each variation began with a random change to a gene.

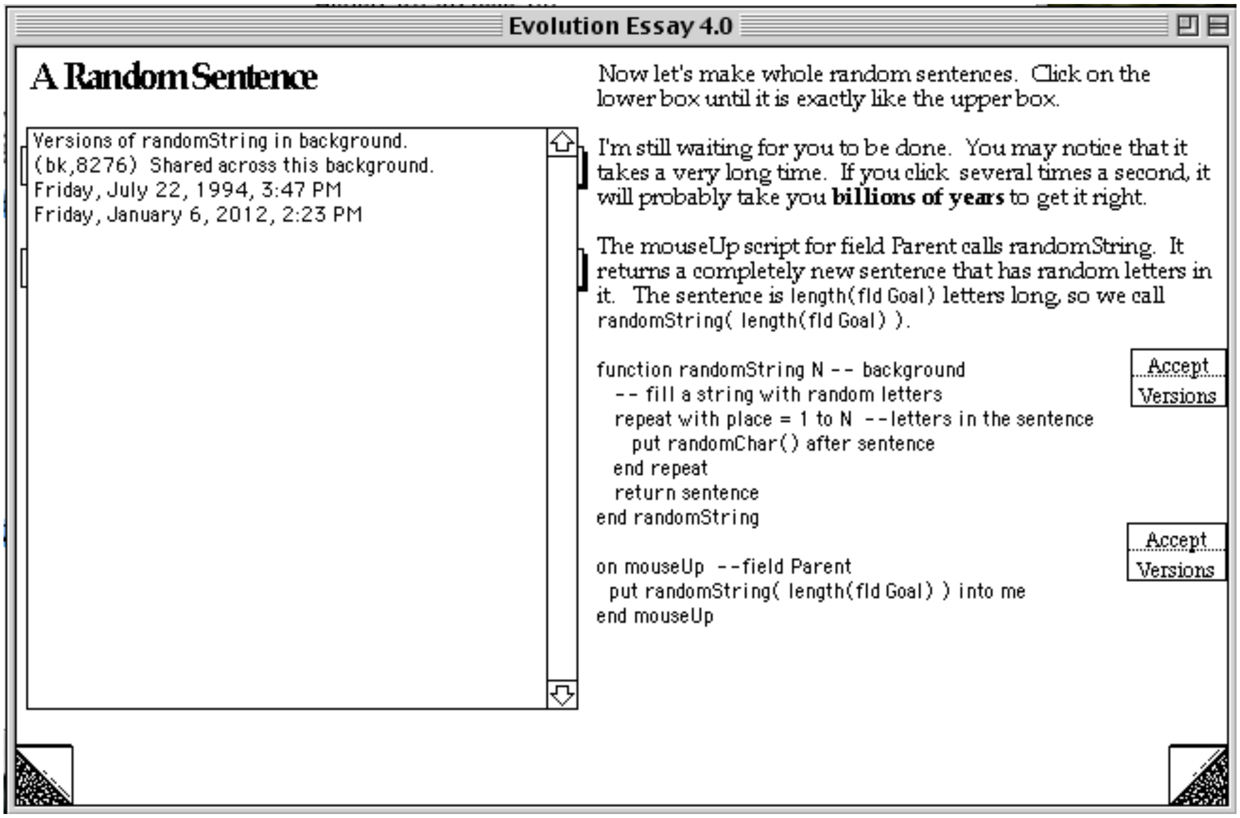
Animals are so well designed -- how could they have been built from random events? How could chance have created such wondrous things as eyes and fine fur?

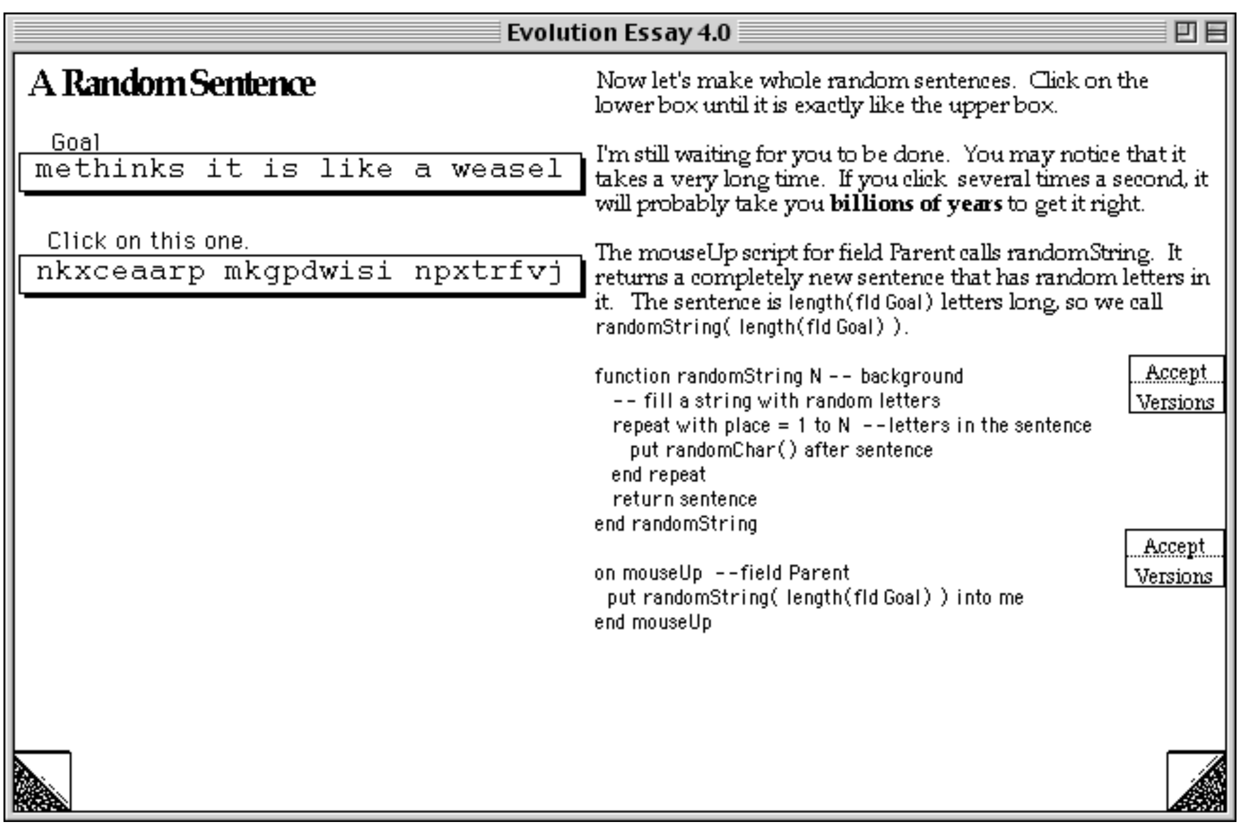
We know that animals are not simply random combinations of parts. The process of selection has made the difference. It saves the things that work and throws out the things that do not work. Selection does this over and over again, each generation, for a long time.

It is true that an eye is a very improbable thing. The chance of it appearing through variation in one generation is extremely small. Let's look at some other improbable things and see how repeated variation and selection can create them.


Evolution Essay 4.0
Improbable Things
Type something here:
A Great Alfa Romeo
Chance of a monkey typing it
It has 18 letters.
The chance is one in
58,149,737,003,040,063,952,519,168
In his book, The Blind Watchmaker, Richard Dawkins takes commonsense doubts about evolution and shows why the theory is plausible. His first example is taken from Hamlet, where the overagreeable Polonius sees any suggested form in clouds, "methinks it is like a weasel." The problem is creating this sentence by having monkeys typing on typewriters is very slim. There are 27 possibilities (including space) for each letter of the 28 letters in the sentence. That is 27 x 27 x 27 ... (28 times) or 27 to the 28th power. For comparison there have been fewer than 10 to the 18th seconds in the 15billion years since the universe began.
Let's work with the problem of evolving "methinks it is like a weasel." It is a little bit like trying to evolve the right DNA sequence for a gene. DNA has letters just like a sentence, and a good gene has a certain spelling.

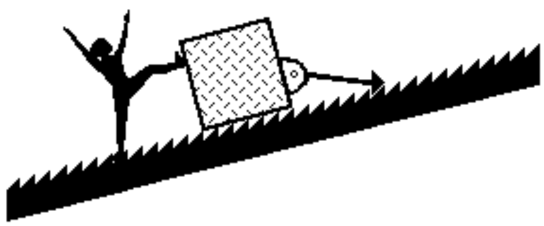

Evolution Essay 4.0
A Random Letter
k
Goal
q
Click Here
Let's start by using random variation to match a single letter. Suppose the best letter is "k". Click on the lower box until it matches the goal. How many tries did it take you?
The script for the lower field is:
on mouseUp -- card field Trial
put randomChar() into me
end mouseUp
Accept
Versions
RandomChar is a function that returns a random letter of the alphabet or a space.
function randomChar -- background
put random(27) into which
return char which of " abcdefghijklmnopqrstuvwxyz"
end randomChar
Accept
Versions
Random(N) is a built-in function that returns a number from one to N. RandomChar takes in a random number and then counts along the alphabet string and returns that character.


Evolution Essay 4.0
A Random Sentence
Versions of randomString in background.
(bk,8276) Shared across this background.
Friday, July 22, 1994, 3:47 PM
Friday, January 6, 2012, 2:23 PM
Now let's make whole random sentences. Click on the lower box until it is exactly like the upper box.
I'm still waiting for you to be done. You may notice that it takes a very long time. If you click several times a second, it will probably take you billions of years to get it right.
The mouseUp script for field Parent calls randomString. It returns a completely new sentence that has random letters in it. The sentence is length(fld Goal) letters long, so we call randomString(length(fld Goal)).
function randomString N -- background
-- fill a string with random letters
repeat with place = 1 to N --letters in the sentence
put randomChar() after sentence
end repeat
return sentence
end randomString
on mouseUp --field Parent
put randomString(length(fld Goal)) into me
end mouseUp
Accept
Versions
Accept
Versions


Evolution Essay 4.0
A Random Sentence
Goal
methinks it is like a weasel
Click on this one.
nkxceaarp mkgpdwisi npxtrfvj
Now let's make whole random sentences. Click on the lower box until it is exactly like the upper box.
I'm still waiting for you to be done. You may notice that it takes a very long time. If you click several times a second, it will probably take you billions of years to get it right.
The mouseUp script for field Parent calls randomString. It returns a completely new sentence that has random letters in it. The sentence is length(fld Goal) letters long, so we call randomString(length(fld Goal)).
function randomString N -- background
-- fill a string with random letters
repeat with place = 1 to N --letters in the sentence
put randomChar() after sentence
end repeat
return sentence
end randomString
on mouseUp --field Parent
put randomString(length(fld Goal)) into me
end mouseUp
Accept
Versions
Accept
Versions

Evolution Essay 4.0

Selection, a Little at a Time



Using random variation to get the whole sentence is not working. The problem is **not** that **some** letters don't come out right some of the time. In almost every sentence, at least one letter is correct somewhere in it. The problem is that we are throwing the correct letters away afterwards!

We are throwing away any partial progress we make. If it is not totally right, we throw it all out and start over.

Suppose we don't start over each time. Each time we'll start with the best sentence we've seen so far, and make variations on it. If the sentence we create is closer to the goal sentence, we'll keep it. By keeping each improvement as it comes, we can crawl towards the goal.

Not starting over each time is exactly what animals do. Animal offspring are mostly like their parents. The parent is what the process of evolution has produced so far. The offspring are that plus a little change. Let's model our sentence evolving project on parents and offspring.

Evolution Essay 4.0

Keeping Improvements

Goal

methinks it is like a weasel

Parent

pksoodyhihapjnudrhoabobesgkv

Current **Copy**

pksoodyhihapjnudrhoabobesgkv

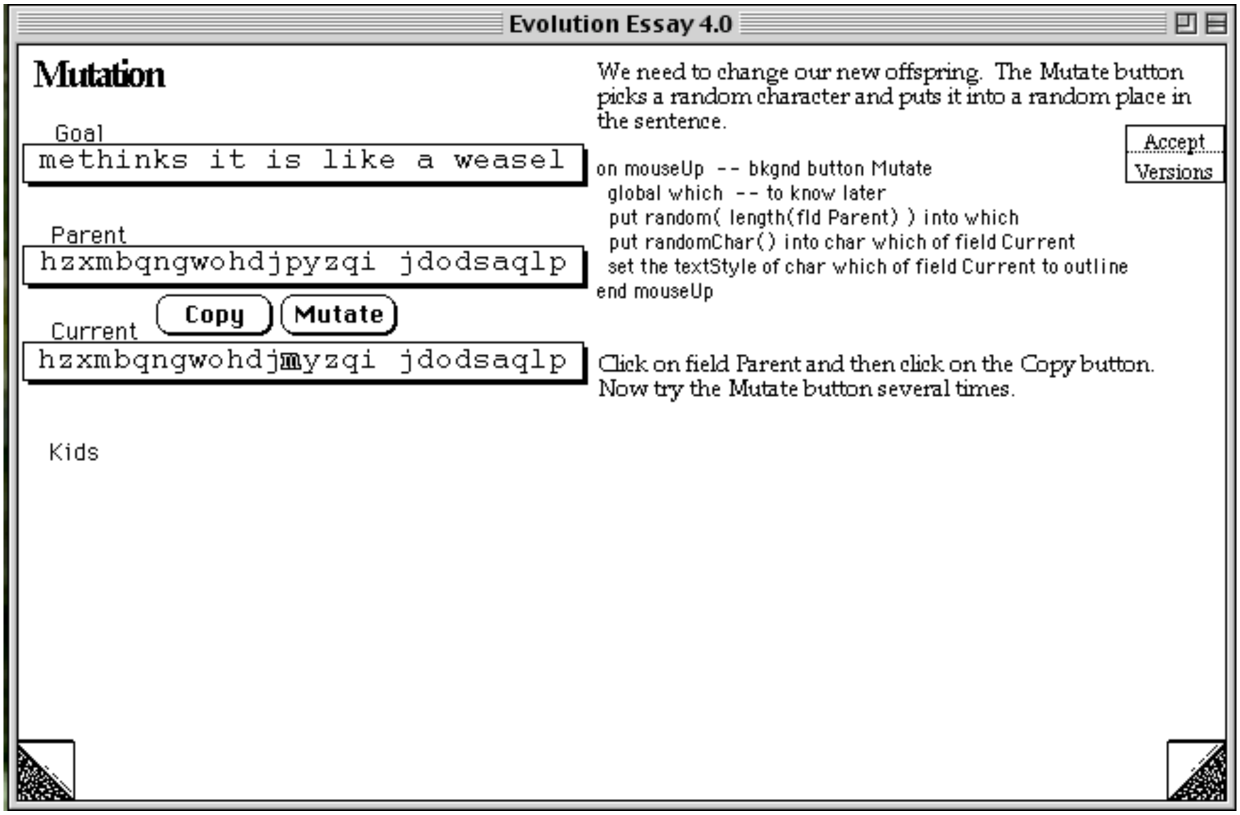
Kids

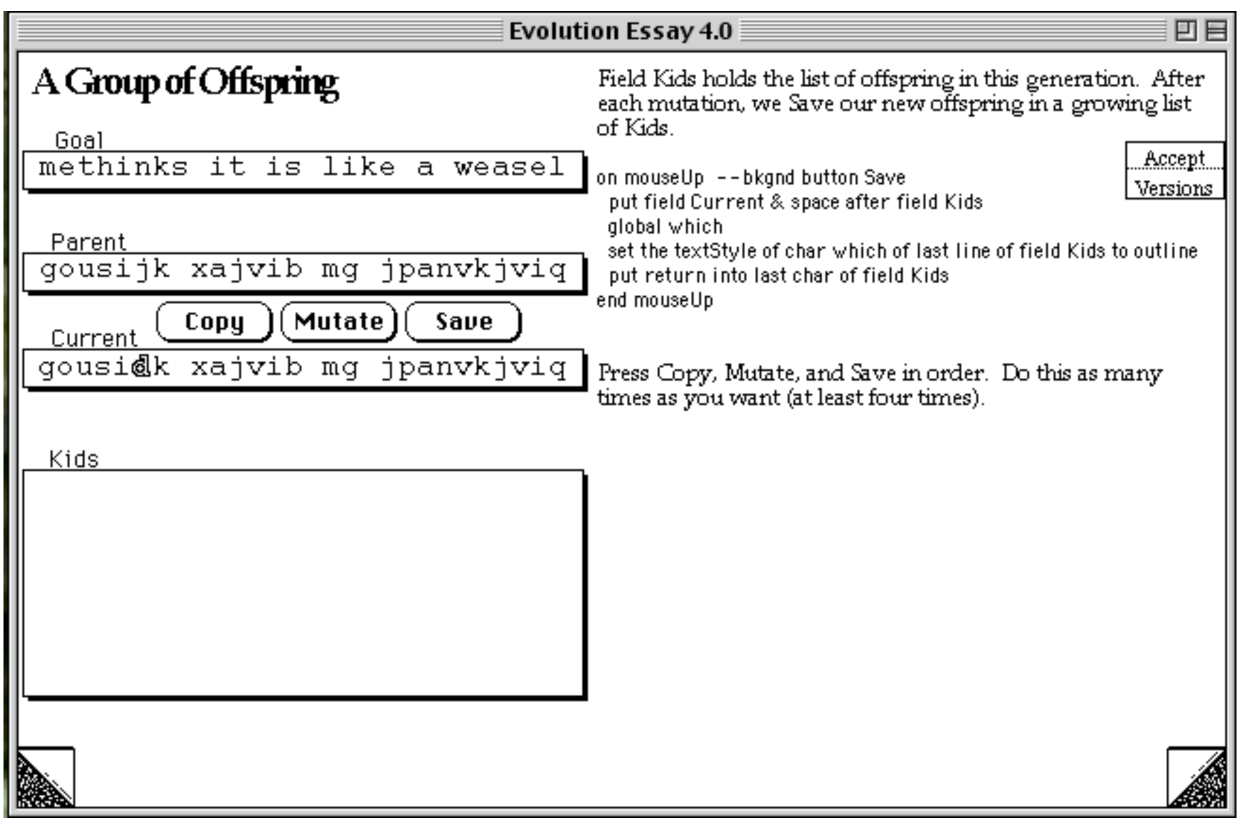
Field Parent will be the best sentence we have made so far. Remember that clicking on field Parent puts a random sentence into it. Only do that once at the beginning.

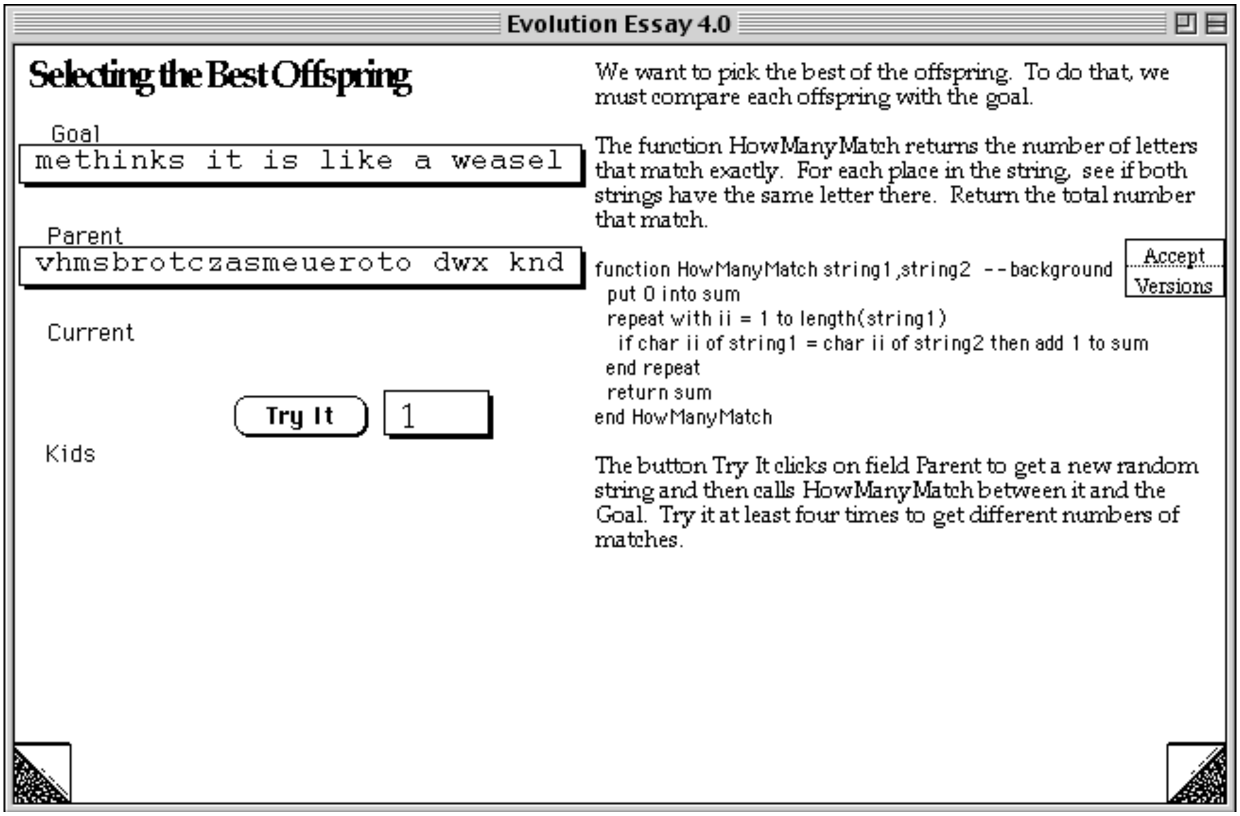
In field Current, we will create a new offspring. We do this by copying field Parent into field Current. Here is the script of the Copy button.

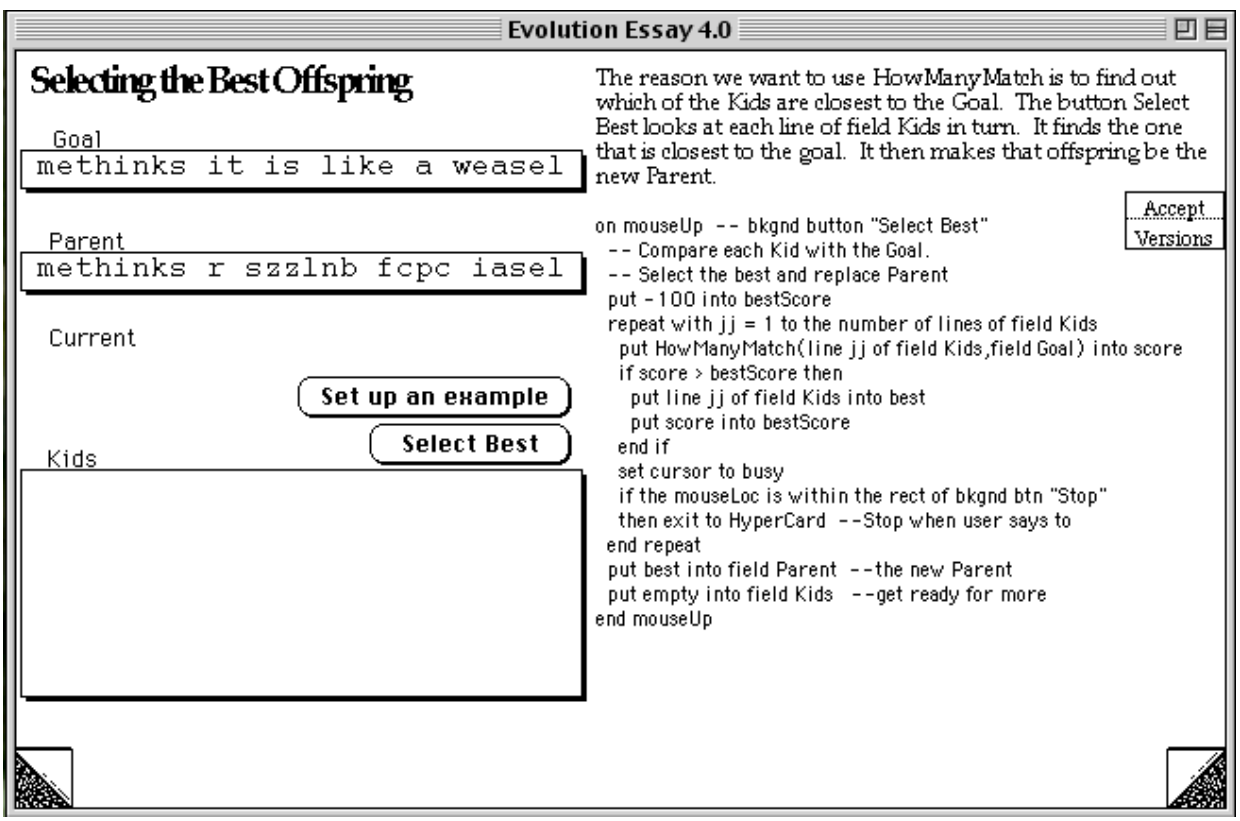
```
on mouseUp  -- bkgnd button Copy
  put field Parent into field Current
end mouseUp
```

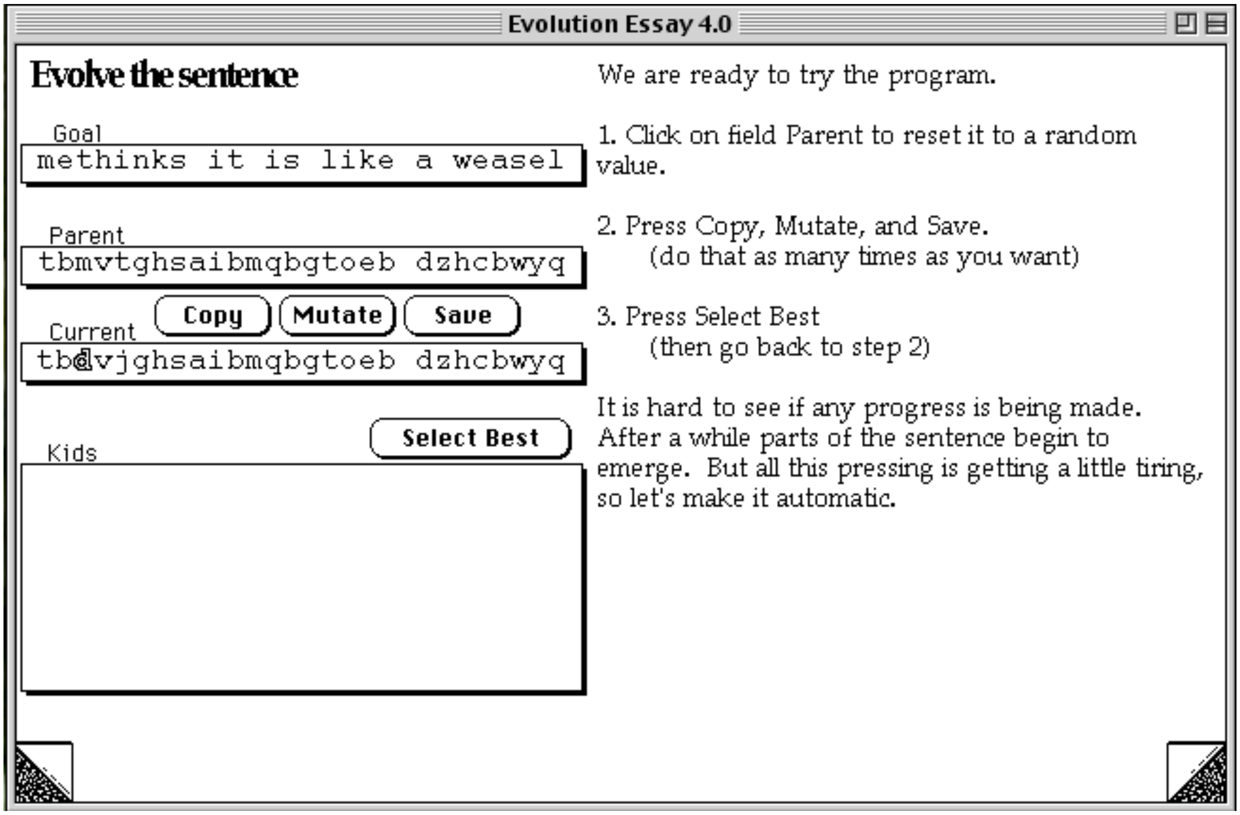
Accept
Versions

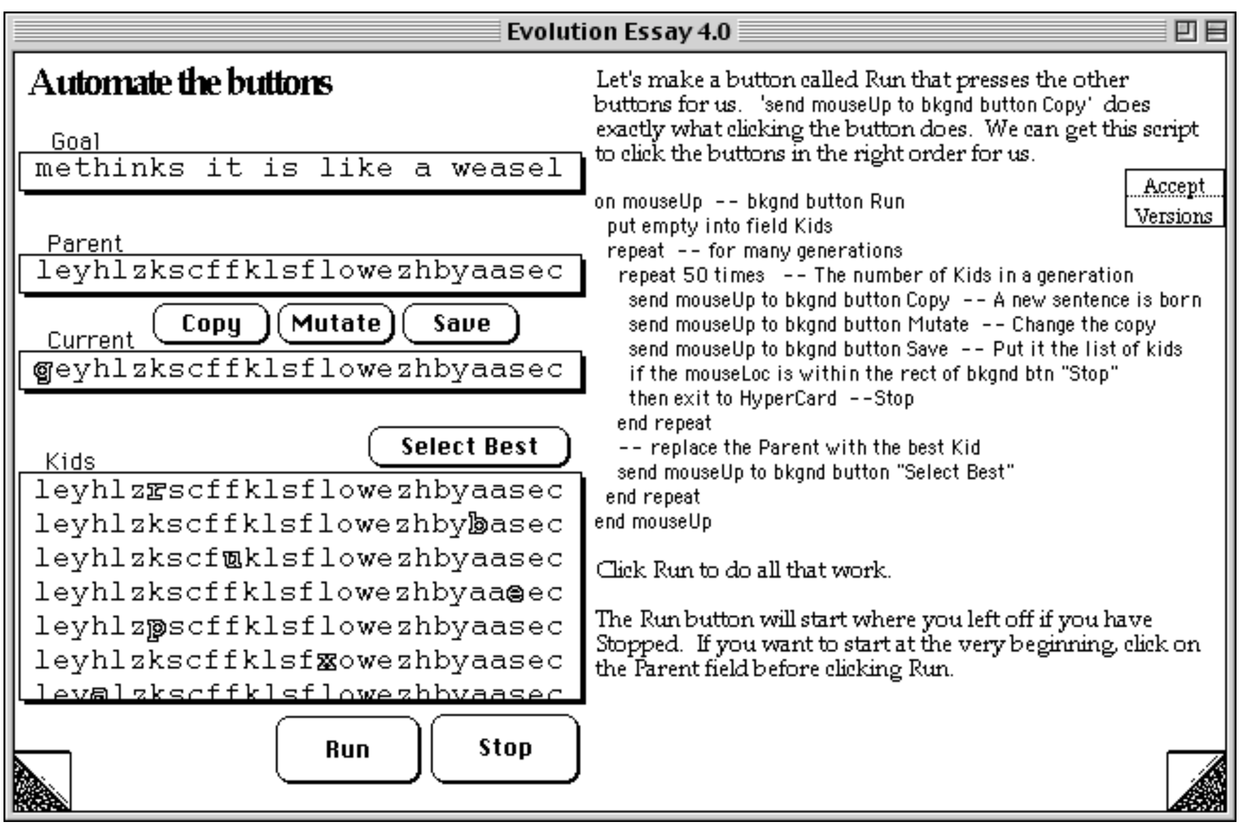

Evolution Essay 4.0
Mutation
Goal
methinks it is like a weasel
Parent
hzxmbqngwohdjpyzqi jdodsaqlp
Copy
Mutate
Current
hzxmbqngwohdjmyzqi jdodsaqlp
Kids
We need to change our new offspring. The Mutate button picks a random character and puts it into a random place in the sentence.
Accept
Versions
on mouseUp -- bkgnd button Mutate
global which -- to know later
put random(length(fld Parent)) into which
put randomChar() into char which of field Current
set the textStyle of char which of field Current to outline
end mouseUp
Click on field Parent and then click on the Copy button. Now try the Mutate button several times.

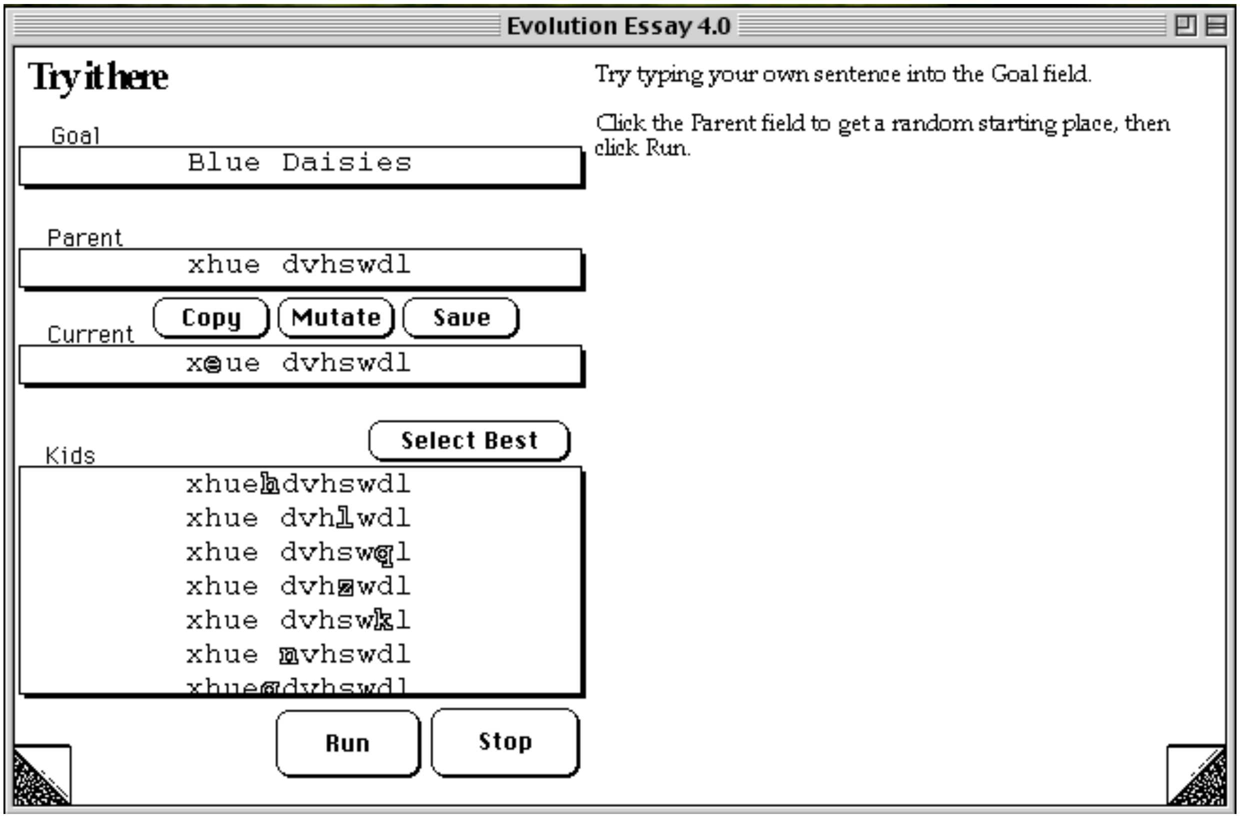

Evolution Essay 4.0
A Group of Offspring
Goal
methinks it is like a weasel
Parent
gousijk xajvib mg jpanvkjviq
Copy
Mutate
Save
Current
gousick xajvib mg jpanvkjviq
Kids
Field Kids holds the list of offspring in this generation. After each mutation, we Save our new offspring in a growing list of Kids.
Accept
Versions
on mouseUp --bkgnd button Save
put field Current & space after field Kids
global which
set the textStyle of char which of last line of field Kids to outline
put return into last char of field Kids
end mouseUp
Press Copy, Mutate, and Save in order. Do this as many times as you want (at least four times).

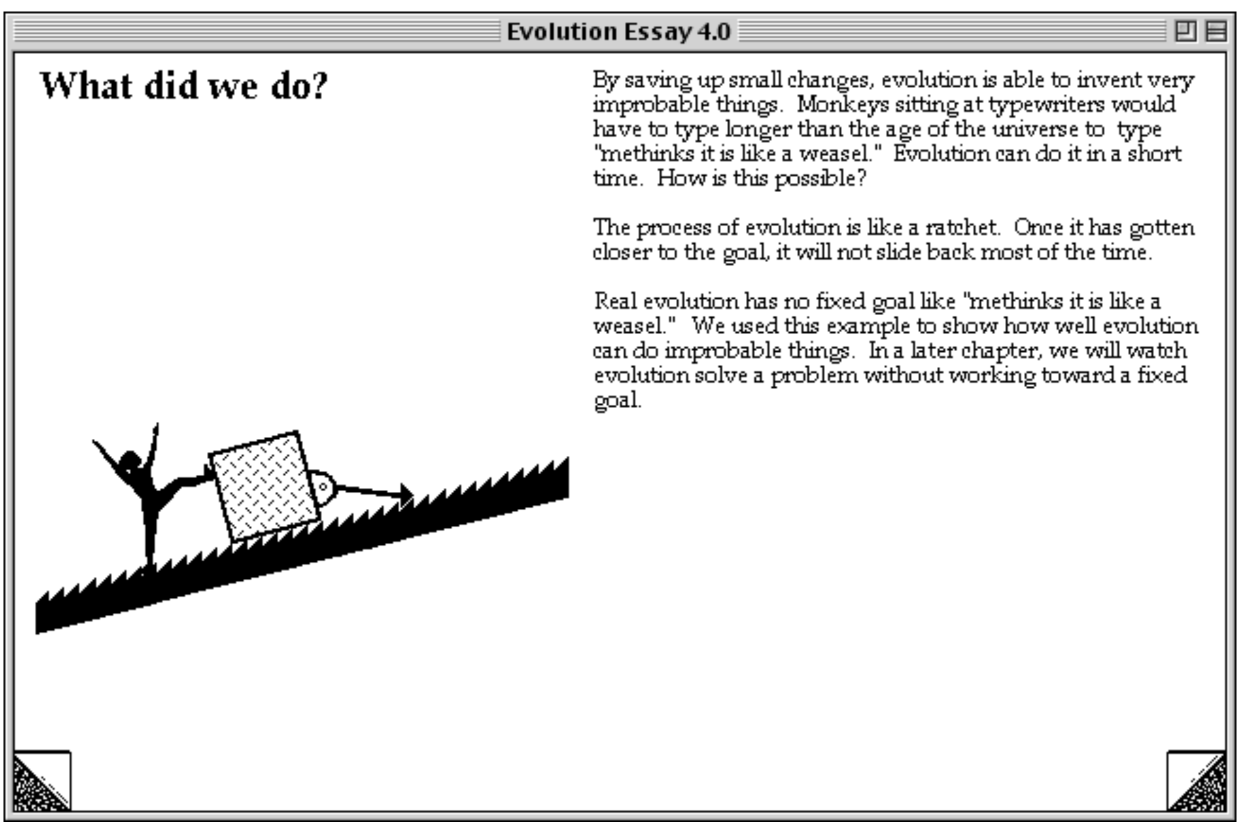

Evolution Essay 4.0
Selecting the Best Offspring
Goal
methinks it is like a weasel
Parent
vhmsbrotczasmeueroto dwx knd
Current
Try It
1
Kids
We want to pick the best of the offspring. To do that, we must compare each offspring with the goal.
The function HowManyMatch returns the number of letters that match exactly. For each place in the string, see if both strings have the same letter there. Return the total number that match.
Accept
Versions
function HowManyMatch string1,string2 --background
put 0 into sum
repeat with ii = 1 to length(string1)
if char ii of string1 = char ii of string2 then add 1 to sum
end repeat
return sum
end HowManyMatch
The button Try It clicks on field Parent to get a new random string and then calls HowManyMatch between it and the Goal. Try it at least four times to get different numbers of matches.

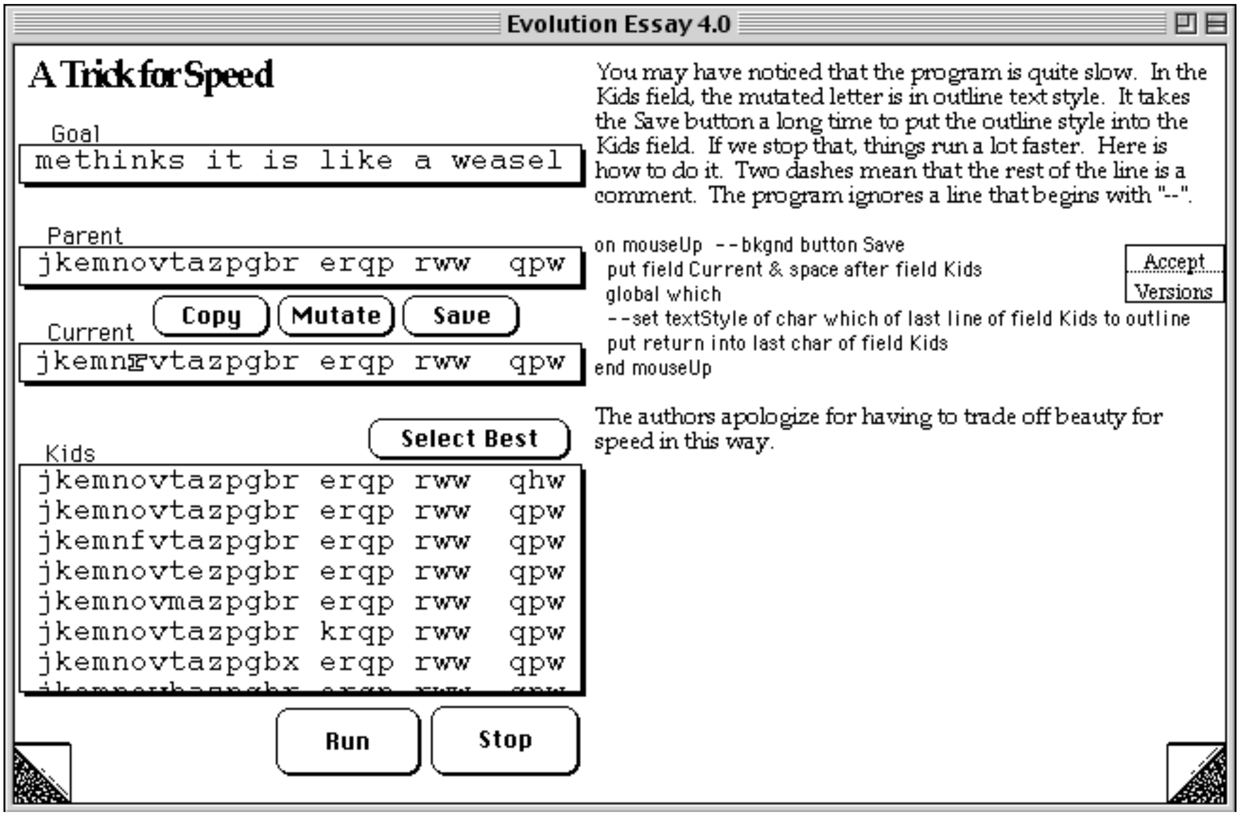

Evolution Essay 4.0
Selecting the Best Offspring
Goal
methinks it is like a weasel
Parent
methinks r szzlnb fcpc iasel
Current
Set up an example
Select Best
Kids
The reason we want to use HowManyMatch is to find out which of the Kids are closest to the Goal. The button Select Best looks at each line of field Kids in turn. It finds the one that is closest to the goal. It then makes that offspring be the new Parent.
Accept
Versions
on mouseUp -- bkgnd button "Select Best"
-- Compare each Kid with the Goal.
-- Select the best and replace Parent
put -100 into bestScore
repeat with jj = 1 to the number of lines of field Kids
put HowManyMatch(line jj of field Kids,field Goal) into score
if score > bestScore then
put line jj of field Kids into best
put score into bestScore
end if
set cursor to busy
if the mouseLoc is within the rect of bkgnd btn "Stop"
then exit to HyperCard --Stop when user says to
end repeat
put best into field Parent --the new Parent
put empty into field Kids --get ready for more
end mouseUp

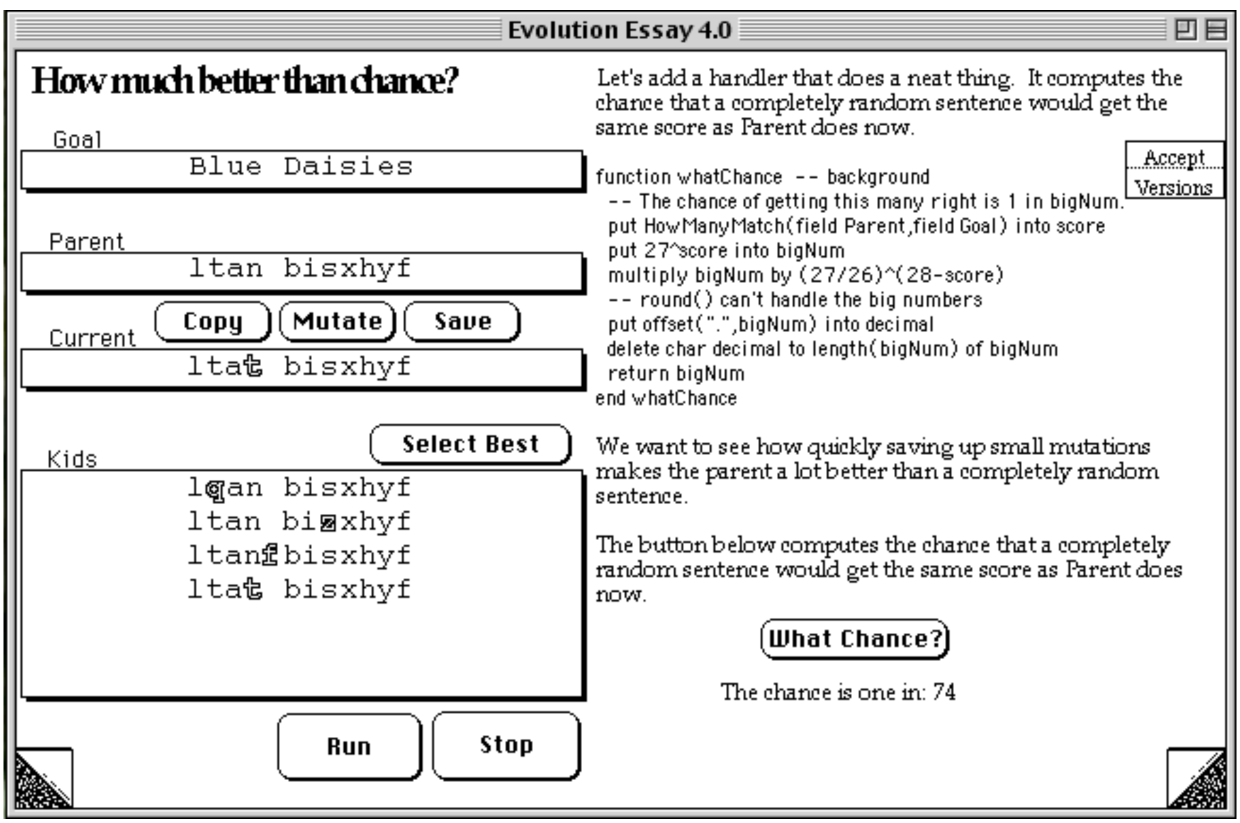

Evolution Essay 4.0
Evolve the sentence
Goal
methinks it is like a weasel
Parent
tbmvtghsaibmqbgtoeb dzhcbwyq
Copy
Mutate
Save
Current
tbdvjghsaibmqbgtoeb dzhcbwyq
Select Best
Kids
We are ready to try the program.
1. Click on field Parent to reset it to a random value.
2. Press Copy, Mutate, and Save.
(do that as many times as you want)
3. Press Select Best
(then go back to step 2)
It is hard to see if any progress is being made. After a while parts of the sentence begin to emerge. But all this pressing is getting a little tiring, so let's make it automatic.

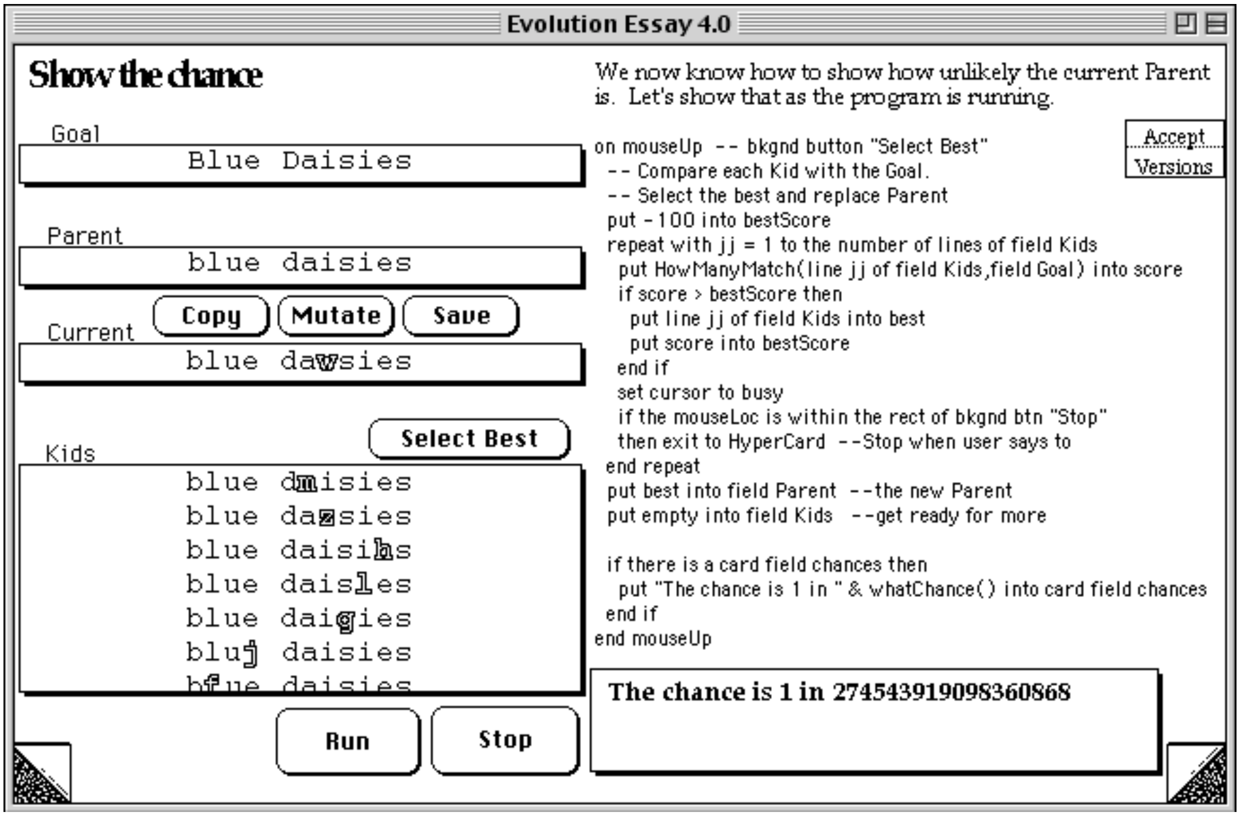

Evolution Essay 4.0
Automate the buttons
Goal
methinks it is like a weasel
Parent
leyhlzkscffklsflowezhbyaasec
Copy
Mutate
Save
Current
geyhlzkscffklsflowezhbyaasec
Select Best
Kids
leyhlzrscffklsflowezhbyaasec
leyhlzkscffklsflowezhbybasec
leyhlzkscfuklsflowezhbyaasec
leyhlzkscffklsflowezhbyaaeec
leyhlzpscffklsflowezhbyaasec
leyhlzkscffklsfxowezhbyaasec
Run
Stop
Let's make a button called Run that presses the other buttons for us. 'send mouseUp to bkgnd button Copy' does exactly what clicking the button does. We can get this script to click the buttons in the right order for us.
Accept
Versions
on mouseUp -- bkgnd button Run
put empty into field Kids
repeat -- for many generations
repeat 50 times -- The number of Kids in a generation
send mouseUp to bkgnd button Copy -- A new sentence is born
send mouseUp to bkgnd button Mutate -- Change the copy
send mouseUp to bkgnd button Save -- Put it the list of kids
if the mouseLoc is within the rect of bkgnd btn "Stop"
then exit to HyperCard --Stop
end repeat
-- replace the Parent with the best Kid
send mouseUp to bkgnd button "Select Best"
end repeat
end mouseUp
Click Run to do all that work.
The Run button will start where you left off if you have Stopped. If you want to start at the very beginning, click on the Parent field before clicking Run.

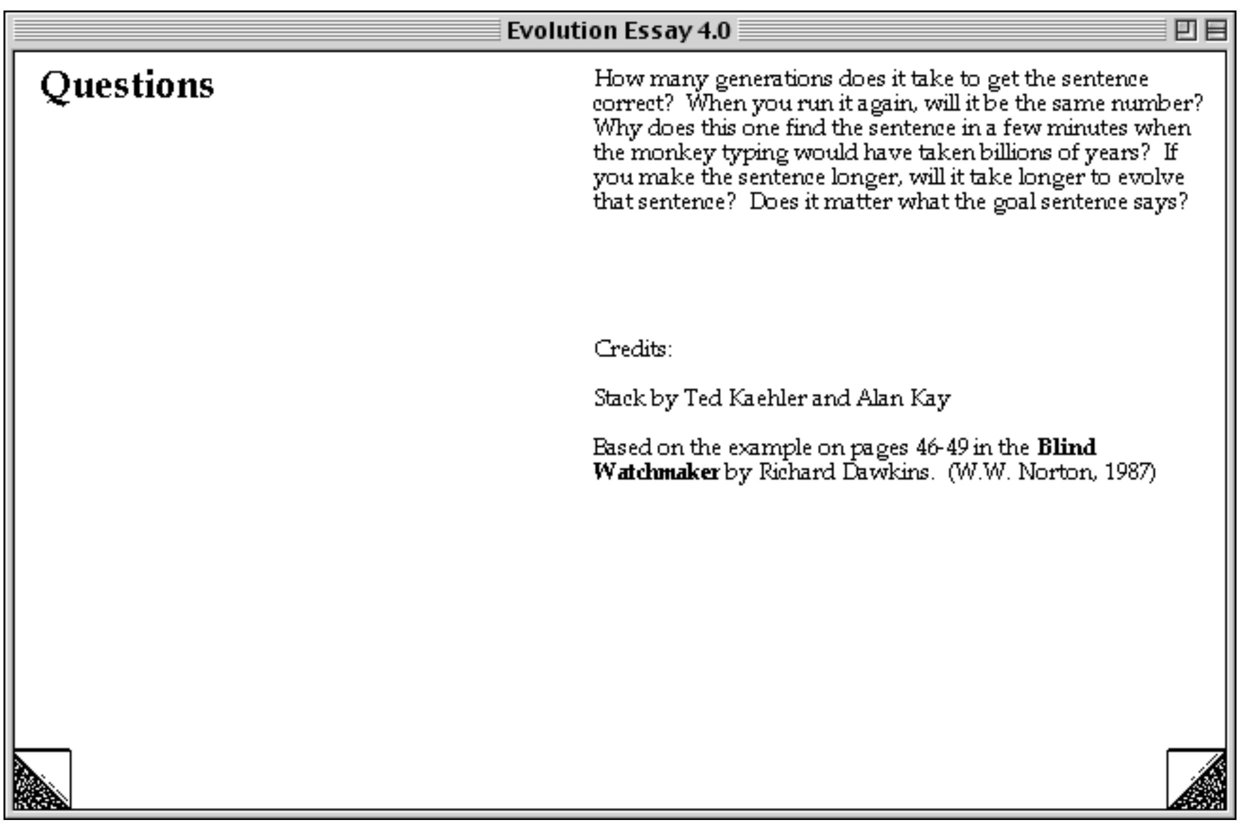

Evolution Essay 4.0
Try it here
Goal
Blue Daisies
Parent
xhue dvhswdl
Copy
Mutate
Save
Current
xeue dvhswdl
Select Best
Kids
xuehdvhswdl
xhue dvhlwdl
xhue dvhswql
xhue dvhzwdl
xhue dvhswkl
xhue nvhswdl
Run
Stop
Try typing your own sentence into the Goal field.
Click the Parent field to get a random starting place, then click Run.


Evolution Essay 4.0
What did we do?
By saving up small changes, evolution is able to invent very improbable things. Monkeys sitting at typewriters would have to type longer than the age of the universe to type "methinks it is like a weasel." Evolution can do it in a short time. How is this possible?
The process of evolution is like a ratchet. Once it has gotten closer to the goal, it will not slide back most of the time.
Real evolution has no fixed goal like "methinks it is like a weasel." We used this example to show how well evolution can do improbable things. In a later chapter, we will watch evolution solve a problem without working toward a fixed goal.


Evolution Essay 4.0
A Trick for Speed
Goal
methinks it is like a weasel
Parent
jkemnovtazpgbr erqp rww qpw
Copy
Mutate
Save
Current
jkemnovtazpgbr erqp rww qpw
Select Best
Kids
jkemnovtazpgbr erqp rww qhw
jkemnovtazpgbr erqp rww qpw
jkemnfvtazpgbr erqp rww qpw
jkemnovtezpgbr erqp rww qpw
jkemnovmazpgbr erqp rww qpw
jkemnovtazpgbr krqp rww qpw
jkemnovtazpgbx erqp rww qpw
Run
Stop
You may have noticed that the program is quite slow. In the Kids field, the mutated letter is in outline text style. It takes the Save button a long time to put the outline style into the Kids field. If we stop that, things run a lot faster. Here is how to do it. Two dashes mean that the rest of the line is a comment. The program ignores a line that begins with "--".
on mouseUp --bkgnd button Save
put field Current & space after field Kids
global which
--set textStyle of char which of last line of field Kids to outline
put return into last char of field Kids
end mouseUp
Accept
Versions
The authors apologize for having to trade off beauty for speed in this way.


Evolution Essay 4.0
How much better than chance?
Goal
Blue Daisies
Parent
ltan bisxhyf
Copy
Mutate
Save
Current
ltat bisxhyf
Select Best
Kids
lqan bisxhyf
ltan bizxhyf
ltanfbisxhyf
ltat bisxhyf
Run
Stop
Let's add a handler that does a neat thing. It computes the chance that a completely random sentence would get the same score as Parent does now.
function whatChance -- background
-- The chance of getting this many right is 1 in bigNum.
put HowManyMatch(field Parent,field Goal) into score
put 27^score into bigNum
multiply bigNum by (27/26)^(28-score)
-- round() can't handle the big numbers
put offset(".",bigNum) into decimal
delete char decimal to length(bigNum) of bigNum
return bigNum
end whatChance
Accept
Versions
We want to see how quickly saving up small mutations makes the parent a lot better than a completely random sentence.
The button below computes the chance that a completely random sentence would get the same score as Parent does now.
What Chance?
The chance is one in: 74


Evolution Essay 4.0
Show the chance
Goal
Blue Daisies
Parent
blue daisies
Current
blue dawsies
Copy
Mutate
Save
Kids
Select Best
blue dmisies
blue dazsies
blue daisihs
blue daislies
blue daigies
bluj daisies
bfue daisies
Run
Stop
We now know how to show how unlikely the current Parent is. Let's show that as the program is running.
Accept
Versions
on mouseUp -- bkgnd button "Select Best"
-- Compare each Kid with the Goal.
-- Select the best and replace Parent
put -100 into bestScore
repeat with jj = 1 to the number of lines of field Kids
put HowManyMatch(line jj of field Kids,field Goal) into score
if score > bestScore then
put line jj of field Kids into best
put score into bestScore
end if
set cursor to busy
if the mouseLoc is within the rect of bkgnd btn "Stop"
then exit to HyperCard --Stop when user says to
end repeat
put best into field Parent --the new Parent
put empty into field Kids --get ready for more
if there is a card field chances then
put "The chance is 1 in " & whatChance() into card field chances
end if
end mouseUp
The chance is 1 in 274543919098360868


Evolution Essay 4.0
Questions
How many generations does it take to get the sentence correct? When you run it again, will it be the same number? Why does this one find the sentence in a few minutes when the monkey typing would have taken billions of years? If you make the sentence longer, will it take longer to evolve that sentence? Does it matter what the goal sentence says?
Credits:
Stack by Ted Kaehler and Alan Kay
Based on the example on pages 46-49 in the Blind Watchmaker by Richard Dawkins. (W.W. Norton, 1987)