

BOXER Introduces Itself

Author(s): Declan O'Reilly

Source: *Mathematics in School*, Vol. 25, No. 3 (May, 1996), pp. 42-46

Published by: [The Mathematical Association](#)

Stable URL: <http://www.jstor.org/stable/30211770>

Accessed: 07/04/2014 15:30

Your use of the JSTOR archive indicates your acceptance of the Terms & Conditions of Use, available at
<http://www.jstor.org/page/info/about/policies/terms.jsp>

JSTOR is a not-for-profit service that helps scholars, researchers, and students discover, use, and build upon a wide range of content in a trusted digital archive. We use information technology and tools to increase productivity and facilitate new forms of scholarship. For more information about JSTOR, please contact support@jstor.org.



The Mathematical Association is collaborating with JSTOR to digitize, preserve and extend access to *Mathematics in School*.

<http://www.jstor.org>

BOXER

introduces itself

by Declan O'Reilly, University of Sheffield

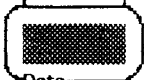
super-demo

Introduction to Boxer

Boxer is a new type of software being developed by Andrea diSessa and his colleagues at the University of California in Berkeley. It is both a continuation of and an advance on, Logo. It is a continuation because the programming language at its core is Logo. It is an advance because Boxer exploits visual techniques for simplifying the language, and because it integrates several facilities, such as text processing, data handling, and dynamic graphics within the same environment. In this short paper, I shall be using Boxer to introduce itself.

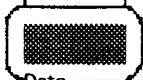
Data

demo1



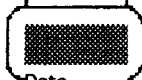
Data

demo2



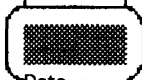
Data

demo3



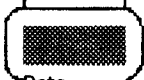
Data

demo4



Data

demo5



Data

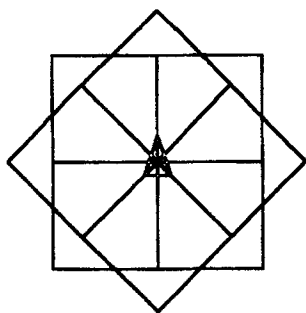
Final comment



Data

Data

demo1



menu

```
fd 30
cs
square 10
cs rotate-square 40
```

Data

square

```
input side
repeat 4
```

```
fd side rt 90
```

Doit

Doit

rotate-square

```
input x
repeat 8
```

```
square x
rt 45
```

Doit

Doit

Some more details

It is called Boxer because everything is organised around the metaphor of a box. Programs are simply rectangular boxes, while data appears in boxes with rounded corners. These can be seen above along with a graphics box and name-tag boxes. Boxes can be open or closed and their sizes varied. There is no distinction between editor and workplace as in most versions of Logo: you can edit what you see on the screen.

Data

demo2



menu

```
fd 30
cs
square 10|
cs rotate-square 40
```

Error: Can't find a box named SQUARE

Data

Data

rotate-square

```
input x
repeat 8
```

```
square x
rt 45
```

Doit

square



Doit

Doit

My-commentary

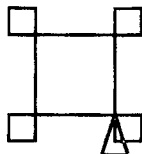
This time, I have placed the doit box 'square' inside the 'rotate-square' box. 'rotate-square' can be run from the menu as before, but look what happens when I try to run 'square'. I get an error message because 'square' is local to 'rotate-square'.

Note also that I've closed the 'square' box getting rid of unnecessary details.

Data

Data

demo3



menu

```
fd 30
cs
square 10
cs rotate-square 40
cs square-on-square 40 10
```

Data

Data

rotate-square

input x
repeat 8

square x
rt 45

Doit

square

input side
repeat 4

fd side rt

Doit

Doit

square-on-square

input y x
repeat 4

fd y square x

Doit

square

input bit
repeat 3

fd bit rt 90

Doit

Doit

My-commentary

Being lazy, I didn't want to type out square again, so I used the same name etc. inside square-on-square. Ok, it's not a square, but since, it is local, having the same name doesn't matter.

PS.
Could you do a hexagons on hexagon pattern?

Data

Data

demo4



menu

```
fd 30
cs
square 10
cs rotate-square 40
cs square-on-square 40 10
```

Data

Data

rotate-square



Doit

square-on-square



Doit

My-commentary

You may have noticed how I have been using the screen to communicate with you. You can do the same with children or they can use it to communicate with each other, e.g., setting challenges.

For instance,

challenge



Data

Data

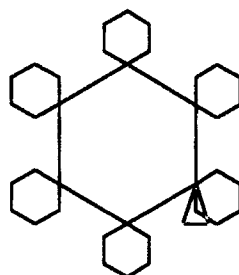
Data

My-commentary

You may have noticed how I have been using the screen to communicate with you. You can do the same with children or they can use it to communicate with each other, e.g., setting challenges.

For instance,

challenge



message for elaine

Hi Elaine,

I've learnt
how to do the hex-on-hex
challenge. I bet you can't
do an oct-on-oct?

Julia

Data

hex-on-hex



Doit

Data

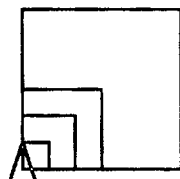
menu

cs hex-on-hex 40 10

Data

Data

demo5



menu

fd 30
cs
square 60

Data

square

input mutt
repeat 4

fd mutt
rt 90

Doit

mutt

30

Data

Doit

My-commentary

You may also have noticed how I have been using variables right from the start.

In Boxer, a variable is simply a named data box. I introduced variables to year 5 children by using internal data boxes like 'mutt'. To change the value of the variable, they had to go inside the box to change the numbers. There was little difficulty with this idea. One child said: "They're called variables 'cause you vary them."

In the above diagram, the small squares have been obtained by running the line 'repeat 4 ...' with the values 10, 20 and 30 in 'mutt' respectively from inside 'square'. The big square was obtained by running 'square 60' from the menu.

Data

Data

super-demo

Introduction to Boxer

Data

demo1

Data

demo2

Data

demo3

Data

demo4

Data

demo5

Data

Final comment

This is where I've got to after about 2 hours work. I am, in effect, producing my own 'computer worksheets' to tell you about Boxer. You can do something similar for children. In my research, we used this method, i.e., we used Boxer to introduce itself. This gave us a way of enabling all 30 children to gain access to the one computer with Boxer on without me having to be always present. This was more than a computer tutorial as both the teacher and I were able to change our computer worksheets in response to children's experiences, just as good teachers have always done with traditional media!

Data

Data

I have deliberately chosen to end this article by coming out of Boxer and using a conventional word-processing package. I have done this by way of warning. Boxer is not commercially marketed yet. The version that I have running on my Mac, for example, is entitled 'Experiment Mac Boxer', and its word-processing is not on a par with commercial products. It requires a lot of memory: (preferable 12 megs or more) and it costs \$200 for a school license.

I am ending on this sober note because I believe that the lessons of Logo suggest that exaggerated claims, in the long run, can do more harm than good. I personally believe that there is a place in school for programming—and not just in mathematics—and Boxer looks like the kind of language to fill that place.

References

- diSessa, A. A. (1986d). From Logo to Boxer, A New Computational Environment. *Australian Educational Computing*, July, 8-15.
- diSessa, A. A., Abelson, H. and Ploger, D. (1991). An overview of Boxer. *Journal of Mathematical Behaviour*, 10, 2.
- Hoyles, C. and Noss, R. (1992). Introducing Boxer. *Micromath*, 8, 3.

Phone: (510) 643-7201, Fax: (510) 642-4566,
E.mail: dahall@uclink4.berkeley.edu

Contact for Licensing

David A. Hall, Licensing Associate-Physical Sciences, Engineering, and Computer Software, University of California at Berkeley, Office of Technology Licensing, 2150 Shattuck Ave., Suite 510, Berkeley, CA 94720-1620, USA

Author

Declan O'Reilly, The Education Building, 388 Glossop Road, Sheffield, S10 2JA

E-mail

D. O'Reilly @ Sheffield.ac.uk