

MagicWords – A Programmable Learning Toy

Mikael Kindborg

Computer Science Department
Linköping University
SE-58183 Linköping, Sweden
mikki@ida.liu.se

Robert Scholz

Computer Science Department
Linköping University,
SE-58183 Linköping, Sweden
robsc@ida.liu.se

ABSTRACT

This paper presents a design and technology demonstration that illustrates: (1) how high-level behaviours expressed as contextual words can be used for visual programming of graphical characters, and (2) the possible application of this programming technique to a language learning software toy.

Keywords

Visual programming, children, reading, language learning.

ACM Classification Keywords

D.1.7 Visual Programming, K.3 Computers and Education.

INTRODUCTION

There exists a wide variety of computer-based learning environments for children. Educational software range from pre-programmed drill & practice exercises to open-ended games and playful environments. However, it is unusual that educational software allow children to do programming activities. Creating dynamic and interactive worlds can be a creative and stimulating way to learn, but unfortunately, it is often hard to learn the programming required to do this. This paper demonstrates a way of programming using contextual words, that is meant to be suitable for open-ended creative learning environments, e.g. for language learning, where children work in an explorative way.

MAGIC WORDS

MagicWords is a system we have developed to illustrate a design idea for a software toy for language learning based on programmable objects. The intended application we have in mind is for children to discover the meaning of words and practice to read by learning to recognise words and short phrases. There are also several other potential applications for the programming technique we demonstrate, e.g. to make simple computer games, interactive simulations, and as a way to create interactive art.

Figure 1 shows a screenshot of the system. The system uses several modalities: text, voice, animation, interactivity and programmability. The interaction is based on using pads that contain words and phrases to create characters and to

animate and give them behaviours. In the user interface, children drag and drop different kinds of word pads on a play area, which produces different results. Words for characters and objects are in the menu on the left. Words for behaviours are on the right. Each word pad has a speaker icon, and when clicking on it, a recorded voice says the word. Dropping a character word on the play area creates a picture of the character, with the word visible beneath the character. Dropping a behaviour word on a character gives the character that behaviour, and the word pad is attached to the character and stays visible.

Examples of behaviour words we have implemented are: *Left*, *Right*, *Up*, *Down* (produces motion), *Bounce* (makes an object bounce against walls and other objects), *Steer* (use the keyboard to move an object), *Magnetic* (attracts other characters), and *Eat* (makes other characters disappear when touching them). There are also behaviour words that perform a one-time action and get deleted after they have been dropped. Examples of such words are: *Bigger*, *Smaller* (changes the size of the character), and *Remove* (deletes the character). Many more words and behaviours could be imagined, and it would be possible to have libraries of characters and behaviours for different scenarios and learning situations.

The pedagogical philosophy of the system is in the tradition of the “learning by doing” constructivist style, inspired by e.g. Jerome Bruner [1]. In Bruner's terminology, grabbing and dropping pads connects to the *enactive*, pictures to the *iconic*, and words to the *symbolic* mode of thinking. The word-based approach to reading is related to the “whole language” method [5]. (Without going into the debate of “whole language” versus “phonics”, we believe that a toy such as MagicWords could be useful as one among several educational tools available in the different approaches to learning how to read.)

PROGRAMMING MODEL

The style of programming used in the system is based on high-level behaviours, rather than an algorithmic or rule-based model. The interesting property of behaviours is that they abstract away algorithmic details like conditionals, which means that the user does not have to know about how to express computations. “Bounce” is an example of a behaviour that can be used without any knowledge of how a bouncing algorithm works.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

IDC '06, June 7-9, 2006 Tampere, Finland
Copyright 2006 ACM 1-59593-316-6/06/07... \$5.00

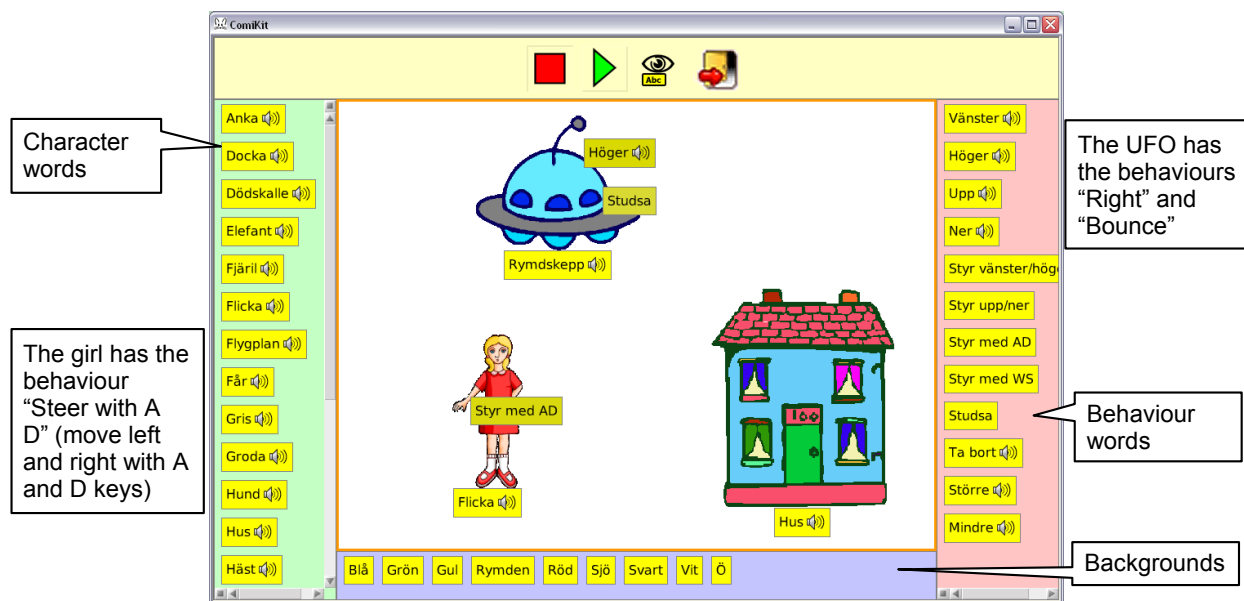


Figure 1. Screenshot from MagicWords (words are in Swedish).

In addition to using high-level behaviours, the system uses a contextual layout of the word pads, which is inspired by comics. In comics, there are several symbolic signs that are used to communicate dynamic and emotional attributes of characters [4]. Examples are voice bubbles, speed lines and sounding words like *SWOOOCH*. Because such signs are shown in the context of the picture of a character, a direct visual relation is created between the sign and the character.

We have used a similar layout, where behaviour words are placed directly on a character. We consider this presentation technique to have a certain directness that we find appealing – it is immediately visible which behaviours a character has. A problem is that having many behaviours makes the screen look cluttered; therefore the user interface has a button for hiding and showing the word pads. There is also a button for pausing the animation to make it easier to add and remove behaviours from moving characters.

The programming model could be made more advanced by introducing parameters to behaviours. For example, rather than just make the “Eat” word delete any character, we could have “Eat X” where “X” would be specified by the user (either as a name or a picture of a character). It would also be possible to use phrases, like “Move left” or “I move left”, and “Eat other characters” or “I eat other characters”. This could be selectable in the tool, and learning could progress from reading single words to reading phrases.

RELATED WORK ON PROGRAMMING TOOLS

Several programming systems for children use a concept of textual “pads” or “tiles”. Examples include Squeak eToys [2] and Scratch [3]. However, these systems are oriented towards algorithms rather than high-level behaviours, and the programming signs are visually separated from the graphical objects. ToonTalk has a concept of behaviours that can be placed on the back of pictures to make them

move and perform different actions [6]. However, since these behaviours are on the back of pictures, they are not visible when the program is running.

DISCUSSION

The system we present is a design and technology demonstration. So far, three children aged 2, 6, and 12 have been involved in the development, and we have focused on play and programming aspects. The language learning aspects of the system have not been tested yet. It would also be interesting to use the system with children who already can read, but are learning a foreign language, and to explore other application areas for programming with contextual words. You are welcome to contact us to get a copy of the system (the version available at the time of writing is in Swedish; additional languages are planned).

REFERENCES

1. Bruner, J. *På väg mot en undervisningsteori*. Lund: Bröderna Ekstads Tryckeri, Swedish edition, 1971. (English title: *Toward a Theory of Instruction*.)
2. Guzdial, M., Rose, K. *Squeak – Open Personal Computing and Multimedia*. Prentice Hall, 2002.
3. Maloney, J., Burd, L., Kafai, Y., Rusk, N., Silverman, B., Resnick, M. *Scratch: A Sneak Preview*. *Second International Conference on Creating, Connecting and Collaborating through Computing (C5'04)*, 2004.
4. McCloud, S. *Understanding Comics*. New York: HarperCollins Publishers, 1993.
5. The National Council of Teachers of English: The Whole Language Umbrella (WLU) Network. <http://www.ncte.org/groups/wlu> (March 2006)
6. Tholander, J., Kahn, K., Jansson, C-G. *Real Programming of an Adventure Game by an 8 year old*. *ICLS 2002*. Lawrence Erlbaum Associates, 2002.