The Many Facets of Scratch

G. Barbara Demo[1], Lawrence Williams[2]

[1] Informatics Department, University of Torino
C.so Svizzera 185, 10149 Torino, Italy
barbara@di.unito.it
[2] School of Sport and Education, Brunel University
Uxbridge, Middlesex, UB8 3PH, UK
lawrence.willams@brunel.ac.uk

Abstract. This paper describes different Scratch activities proposed in different learning situations: for primary school-children aged about 9-10 years old; for in-service teachers active in various levels and types of schools, but with very low or without computing competences; for teachers and pupils in middle schools; for students in their beginning two years of technical secondary school (not necessarily specialized in informatics); and for future informatics teachers having a good knowledge of computer science. In all of these situations, Scratch has been introduced as a system for producing original stories, rather than as a programming system. For beginners, this choice proves to be successful in inspiring and motivating their intent to carry on with new activities. This success is mostly due to the possibility of producing, in a relatively short time, artifacts that teachers can immediately use in school; and the students feel they have created a complete product to show to other students (or to the family and to friends). Informatics experts, in particular informatics teachers, discover a pedagogical methodology they are not used to for introducing to programming. All the while, by using Scratch, we can introduce a number of fundamental elements of computer science. Also, students go through an introductory programming experience, making a faster and smoother change to other programming languages and other environments. In general, the use of Scratch supports and facilitates a process toward achieving competences that many consider necessary to our future young people.

1 Introduction

Among the fundamental informatics competences that everybody should acquire, programming is the most concrete, and probably the most particular, aspect of computing for people having little or no acquaintance with computer science. Besides, it can be a pleasant way of acquiring the basics of informatics, because one may produce artifacts which are useful and enjoyable to interact with, provided that suitable software is used. Scratch is one of the systems best suited to this purpose.

Scratch has been developed at the MIT Media Lab by the Lifelong Kindergarten group directed by Resnick. The aim focused on building a system for beginners where they could express themselves, while being introduced to informatics [10]. It is not

adfa, p. 1, 2011.

only a programming language: it also provides an environment where a user finds several integrated tools (for drawing and recording, for example). More generally, Scratch is a system for producing stories that can have one or more characters (Sprites) acting on a Stage, with one or more Backgrounds, and sounds of different types (for example voices, music, noises). Characters behave as specified by means of code sequences, called Scripts. Here the word script is to be intended mostly as referring to roles in the theatre, or cinema. Of course, characters and their behaviours may be of quite diverse types: they can describe a journey; tell a joke; outline a narrative episode which happened to the storyteller; can be the instructions for using/doing something; or a story can be an original creative story [12]. Besides this, in Scratch, we can specify solutions to different problems, which is what programmers actually do while working in more sophisticated programming environments [1]. Unfortunately, often teachers who are expert in informatics propose traditional algorithms developed in a traditional way, as well, during the first activities with Scratch. Therefore, the students lose the opportunities for engagement in original activities, nearer to the spirit of the system.

This paper describes a pedagogical methodology emphasizing these several facets of Scratch. More specifically, section 2 concerns the use of Scratch for introducing to computing students and teachers from all types and levels of education, having no, or very little, informatics knowledge. In section 3, the focus is on how beginners are asked to develop some personal story which helps them understand that some programming components are present, also, in our everyday life. That means they are already in our way of thinking. Section 4 concerns the use of Scratch in middle school where interdisciplinary activities are particularly important to teachers and pupils.

All the activities described have been conceived, designed and developed:

- together with teachers, i.e. those who work in schools. They have also developed these activities with their students. Indeed, we consider it vital to stimulate and collect suggestions from teachers, and take into account their activities in schools
- in compliance with the European guidelines, to avoid juxtaposing initiatives which often cause lack of depth and confusion in schools
- with a practical mind, considering the several difficulties normally present in schools, such as disciplinary problems, poverty of means, and poverty of school time, because often we can count on just one hour of continuity for an activity. As an example: one of the scheduled projects could not be carried out because: “the informatics laboratory does not exist any more: computers have been moved one per classroom to maintain the digital class-register”, as one of the teachers said.

In all these situations, Scratch has proved to be a program development environment that has been successful in inspiring and motivating pupils, with the intent to carry on with new activities. These are suitable for producing, in a relatively short time, material to be immediately used in school, or felt as a complete product to be shown to other students (or even to the family and to friends). In the environments where we carried out our experiences, Scratch has been shown to stimulate the pupils’ dedication to these activities, and the will to continue. We could not have the same

working atmosphere during introductory activities when we used different programming languages.
In the concluding section, we point out that other programming environments using different languages are suggested to continue the experiences for enhancing the informatics competences of students and teachers.

2 Introducing computing

The first reason why story-telling fits introductory activities well is that it starts with a simple coding exercise: a story is a sequence of actions, performed by the different characters, one after the other (or in parallel, but we do not consider this possibility with beginners). Through this simplicity, the results can be very stimulating for children and adults: an example is the Solar System Discovery developed by Lawrence Williams and his students.
See: http://www.literacyfromscratch.org.uk/index.htm
On Williams' web pages one can find many other stories and interesting materials.

The above two reasons suggest we should centre on a short story as the first activity, especially for people knowing almost nothing about informatics. To be as simple as possible, our beginning stories have characters' actions synchronized simply on a time base, i.e. "after n seconds from the start" a given script of character X begins.

The entire project methodology is based on an active learning paradigm defined by Lawrence Williams, and already experimented with different groups of attendees in Italy and in UK [9]. Notice how, our working group being English-Italian, we began with stories in these two languages (more languages were developed after attendees from different countries came to our workshops).

The steps of the first activity are the following:

- a story is shown full screen (dialogues are in English for activities in Italy, vice versa for activities in UK),
- the components of the Scratch system are then swiftly introduced
- attendees are then asked to translate the first few sentences of the story in the language they like better. This allows them to look into (and through) the code to choose the sentences to be translated. During this step, usually attendees "discover" different aspects of the system presented in the previous step, for example that one script belongs to one character, that one character may be associated more than one script,
- the translation step can result in different, though scaffolded, changes of the original story regarding images, sounds and dialogue
- results are uploaded in a shared environment, and discussed together.

During these steps, attendees work and produce something they immediately feel is their own, something they can show and, for teachers, something they can propose to their students. This approach is warmly appreciated by the participants.

If new dialogues are introduced, they can cause problems in synchronizing the characters' actions. Comments on this aspect often lead to consideration of different

possibilities for the characters entering the scene, or performing some actions, thus broadening the knowledge of the system. Also, discussing all together the story already implemented, and the changes made, gives a chance to share new aspects of the system among participants.

Story-telling is particularly important in primary and middle schools, but not exclusively. As already pointed out, an engaging story can be implemented with very few commands used in a repeated pattern to produce the whole story. This aspect is important for anyone new to programming. It is more important if we think to a story-telling activity in a primary and middle school, where the focus of the learning, for the pupil, is not on the actual coding. Rather, the pupil is developing a narrative, with characters and dialogue, and the coding is merely the tool for presenting the story in an entertaining way. This means that when difficulties arise, the pupil wants to overcome them in order to complete the story, rather than simply getting some abstract coding correct. Besides, story-telling can be a common use of Scratch in all disciplines of middle schools.

3 Personal stories and concrete programming

There are many projects proposing computing science activities without a computer beginning, with the well known CS-unplugged project by Tim Bell, Ian H. Witten and Mike Fellows [3], several contests for primary and middle schools such as the Bebras contest started in Lithuania by Valentina Dagiene in 2004 [2, 6, 11], Olympics of problem-solving started in Italy by Giorgio Casadei in 2006 with problems to solve on paper [5].

Algorithms are not exclusive to the digital world: they have to do with our lives. In 2006, for the master course of the Italian MIUR "Enhancing teaching of scientific disciplines in primary school", attendees were presented with informatics activities introducing them to programming mostly "unplugged", based on discussing algorithms present in our everyday normal life. Often educational robotics is seen as a proper introduction to programming, because it gives the opportunity of "concrete programming" i.e. making physical artifacts like small robots move around, avoiding obstacles, as pupils would do if they were asked to [8].

The procedures for performing arithmetic operations that everybody learns in primary school are examples of algorithms which we normally use, without knowing the name. Bringing everyday experiences into programming points out this relationship, and makes first programming experiences easier. As an example, we have asked students to tell their personal stories about going back home from school. One story was: "On Tuesday, I go to my grandparents' home, while the other days I go to my place." Another one was: "Going back home from school (if we need bread) I go to the bakery, otherwise I go directly home for lunch." This latter story is specified in the Scratch activity shown in figure 1.

In Scratch, we can explicitly transmit the relationship between some students' behaviour and its specification in a programming language, beginning with pupils' personal experiences, or personal stories such as holiday stories, for example. This

makes programming the concrete experience that is called concrete programming dealing with educational robotics [8].

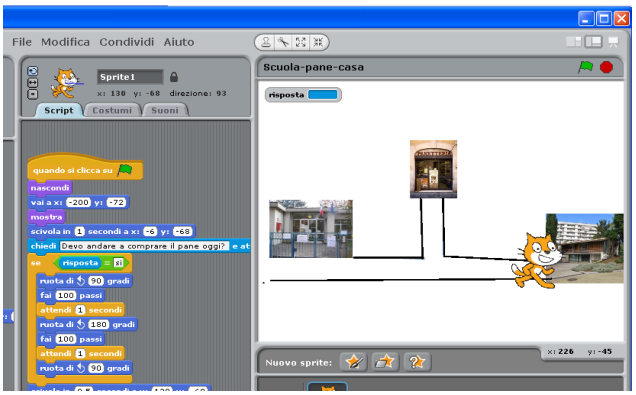

Figure 1. “Going home from school: do we need bread?”

Scratch allows personal activities to become the story of the scene, because pupils make a character (a sprite) behave like one of them would behave in several environments. In figure 1 the background shows a path going from school to a house with a street on the left reaching a bakery (“panetteria”). When the sprite arrives at the corner he asks, “Do we need bread today?” and goes to the bakery if the answer is positive, otherwise he goes directly home.

In this way, a pupil describes the background what would be his/her environment, draws his/her avatar/sprite, and specifies in Scratch the scripts for the sprite to have his/her behaviour. The answer yes to the question, “Do we need bread?”, asked to the user running the program, makes the sprite go to the bakery, then turn to go back to the crossroad, and finally go home, as the schoolchildren would have done in case the bread was not necessary.

These activities have been proposed to students from primary to first year of secondary school during lectures introducing programming, after story-telling activities.

Everyday life has plenty of stories like the School-bakery-home in figure 1. These stories are problems already solved, because adults and young people live them every day: they are our personal stories. Teaching characters to behave in the same way only or basically requires learning how to express, in a digital environment, the actions we perform in our normal life. For someone attending lectures introductory to computing, this is a going from story-telling to telling-my-story.

Other activities can be introduced similarly. In [8] the holidays that some schoolchildren spent in their summertime are “translated” into a journey for an NXT robot. Arriving in the town where the grandmother of one student is living, and not finding her at home, results in going to somewhere else, and then back until she arrives home. This is another example of a personal-story expressed using the repeat-until-grandmother-arrives pattern that doesn’t require much explanation, and becomes the repeat-until-condition-satisfied pattern in the student’s mind.

An evolution of the stage just described concerns developing "stories-with-crossroads" i.e. more sophisticated stories having not only sequential actions yet with a development that depends on the interactions of the sprites with the users who are looking at the story, or on the interactions with other characters.

4 Interdisciplinary activities

From middle school, different disciplines are taught by different teachers. In Italy, there is no discipline where teaching informatics is mandatory, nevertheless the National Indications of the Education Ministry wish for having some programming activity in the Technology discipline. The problem is that this discipline has only two hours per week, and also when several other technologies are taught. Moreover, technology professors normally are architects having very little, if any, knowledge of informatics. As a consequence, informatics risks having no place at all in primary schools, and very little in the middle ones. To ride over this problem, it is important to propose tools making easy-to-share activities among different disciplines, and to spread introductory computing competences to as many as possible teachers and students.

Among early interdisciplinary activities in a middle school, we had practical experiences with simple linear equations in one variable. It is easy to find examples of simple linear equations and lectures on how these can be solved. It is uncommon to find how to put them into practice. While working on educational robotics, we discussed with teachers why modeling a problem by an equation was not a frequent exercise. An example for measuring a path, with circuits covered by a small robot, is proposed in [7]. In our Scratch activities, we discussed with the mathematics professors the "guess a number" game that students sometimes play at school, already described in [1]. We had the students develop a program in which a cat asks the user sitting in front of the screen to do actions such as: "think of a number between 1 and 9" (call it x), "now add 1", "multiply the result by 2", "subtract the number you thought of at the beginning", "subtract 4". At this point the cat says "Tell me the number you finished with". Once the user says his/her answer, call it y, the cat computes x=y+1 and says the original number x. In the referred case the result comes from the equation 2(x+1)-x-4=y. The script with the commands that the cat gives during the game is simply a sequence of instructions, followed by the number of seconds before the next command is given. Input and output communication commands are used. This, and similar experiences, normally lead to discussion and work with Mathematics teachers, who analyze the game as an exercise on one variable equation.

Another example is the photosynthesis story: after a lesson about the photosynthesis process a story was developed by assembling together ideas and drawings from different Scratch activities that pupils had implemented.

Other pupils have developed stories describing monuments and curiosities of their country or where they tell jokes or outline narrative episodes that happened to them.

In a secondary school, students about 16 years old developed a Scratch video-game on the "Divina Commedia" by Dante Alighieri. In it, the scenes include many distinctive traits of the poem, and the player increases her/his score answering to questions verifying his/her comprehension. In a technical secondary school a group of students implemented the simulation of a token ring net: by using Scratch this activity was developed in a short time and students were allowed to focus on the peculiar aspects of the simulated net type.

It is worth noticing that, in a secondary school, when the informatics professor assigned typical algorithmic exercises, the students implemented their solutions with sprites asking and answering in a story-telling fashion, with coloured environments and jokes.

5 Concluding remarks

The experiences here described have been developed over several years, ever-increasing our appreciation of the Scratch environment for introducing computing science both to students and teachers. After all, Scratch was not initiated as a programming environment for a general use, and it is unreasonable to think of, or introduce it as such. Scratch authors aimed at building an environment that was easy to use for creating digital stories. Let's remember that procedures were not available for a long time, and have been introduced only through some extension project, for example the Berkeley ByOB project [4], or the Scratch 2.0 version available from summer 2013. For students choosing to continue, and wanting to develop a deeper programming competence, we suggest, for example, Python as the next language. It is a textual language, interesting because it maintains some simplicity, but also it is offered in programming environments that are nearer to those of languages more used in computing, such as C or Java.

As we said above, the activities here described were offered in training courses for both in-service teachers and future teachers, or directly to students and their teachers in schools. Teachers with very low or no informatics knowledge have been guided to meet many different Scratch facets here described. The aim was that after our workshops they could, by themselves, adjust in collaboration with their students, the activities developed together. The informatics teachers have learnt that informatics can be developed by telling stories, and that, for example, the binary search can be developed as a riddle. Now, using Scratch in the first two years of secondary schools, they have students developing coding as a story, even finding the maximum and minimum of a set of numbers. Mathematics teachers have built, together with their students, activities where they put in practice other parts of the curriculum. We mentioned here the idea of a game, where linear equations in one variable are used; or simply they developed a memory riddle for training in making calculations. Similar ideas have been developed with other disciplines. In all cases, it turns out that Scratch

allows us to see the computer, and programming, also as tools to express everyone's creativity.

References

1. Barbero, A., Demo, G. B., The Art of Programming in a Technical Institute after the Italian Secondary School Reform, in Proc. ISSEP 2011, Bratislava (2011)
2. Bebras Contest, International Contest on Informatics and Computer Fluency http://bebras.org/
3. Bell, T., Witten, I.H. and Fellows, M., CS Unplugged, 1998, http://csunplugged.org
4. ByOB, Build Your Own Blocks 4.0, http://snap.berkeley.edu/, last visited March, 1st 2014.
5. Casadei, G., Teolis, A., "k-12 Problem Solving Olympic Games", Proc. Didamatica 2009, Trento (in Italian).
6. Dagiene, V., Informatics Education for New Millennium Learners, Proc. 5th International Conference on Informatics in Schools: Situation, Evolution and Perspectives, ISSEP 2011, Bratislava, Slovakia, October 26-29, 2011.
7. Demo, G.B., From Mini Rover Programs to Algebraic Expressions, Proceedings 10th IEEE International Conference on Advanced Learning Technologies - ICALT, (2010) 336-340
8. Demo,G.B., Marcianò, G., Siega,S., Concrete Programming using Small Robots in Primary Schools, Proc. 7th IEEE International Conference on Advanced Learning Technologies - ICALT, (2007).
9. Demo, G.B., Williams, L., Scratch Story-Telling for Introducing Computing to In-service Teachers, Internal Report, Informatics Department of the University of Torino, Italy, September 2013.
10. Resnick, M. et all., Scratch: Programming for All, ACM Communications, 52, 11 (Nov. 2009), 60-67.
11. Syslo, M.M.,: Outreach to Prospective Informatics Students. ISSEP 2011: 56-70
12. Williams, L. and Cernochova, M. Literacy from Scratch. In Proceedings of the10th IFIP World Conference on Computers in Education (Torun, Poland, July 2-5, 2013) WCCE 2013. Copernicus University Publ, Torun, PL, 17-27.