Easy Prototyping of Multimedia Interactive Educational Tools for Language Learning based on Block Programming

Stefano Federici, Elisabetta Sergi and Elisabetta Gola
Dept. of Education, Psychology and Philosophy, University of Cagliari, Italy

Keywords: Programming-based Learning, Foreign Languages, Block Languages, Scratch, Computational Thinking.

Abstract: Teaching in the new Digital Era is getting more and more difficult due to the expanding gap between what students and their teachers see as their media of election. Students would like to be engaged with multimedia educational tools and they need, for their future, to learn the basics of coding. In order to overcome the teachers' difficulties in creating multimedia educational interactive tools for their students, and to start using coding even in non-scientific topics, an interactive environment based on the metaphor of building blocks, BlockLang, has been built. BlockLang is very similar to block-based programming languages such as Scratch but it has been designed to teach to elementary students English phrases and sentences from the food domain that, if correctly "coded", will generate a corresponding picture on the tool's "stage". The tool, built by a student of a non-technical degree in just a few weeks, has shown to be effective when tested on 2nd grade students. It has been very well accepted by the students and, as a further bonus, it can be easily updated even by people that have no previous knowledge of computer programming.

1 INTRODUCTION

Can the usage of a block programming language, for example a tool similar to the Scratch programming language, help students to learn a foreign language?

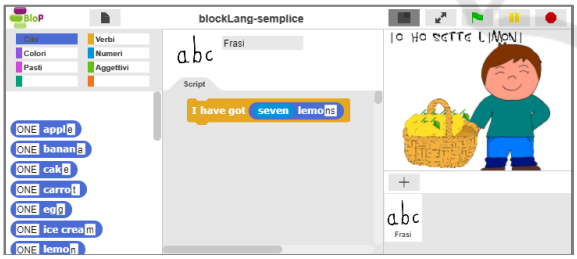


Figure 1: Learning English by Block Programming.

In previous studies, the usage of block programming has shown effective in improving the students' performances in scientific tasks such as learning physics (Lopez and Hernandez, 2015) but also in humanistic tasks such as learning history (Gresse et al., 2017) or English as a foreign language (Costa et al., 2016). In those studies, the programming language is used in order to write mathematical equations applied to moving images or to build interactive "stories" that illustrate a given historical fact or English sentence.

In more recent studies, a different approach has instead been followed, where a visual block-programming tool is used in order to "give life" to the components of the topic, for example the components of an exponentiation operation (figure 2) or the components of a work of art (figure 3) by programming their interactive behaviour. The students, in this approach, learn through a programming-based learning paradigm (Federici et al., 2018).

Figure 2: Expressing the exponentiation operation 2^3 by assembling visual blocks.

In those approaches the students and the teacher must learn a reasonable amount of block-based

Federici, S., Sergi, E. and Gola, E.
Easy Prototyping of Multimedia Interactive Educational Tools for Language Learning based on Block Programming.
DOI: 10.5220/0007766201400153
In *Proceedings of the 11th International Conference on Computer Supported Education (CSEDU 2019)*, pages 140-153
ISBN: 978-989-758-367-4

programming in order to fruitfully use the strategy. So, if teachers of scientific topics can be happy to find a new way of engaging their students and of improving their learning strategies, teachers of non-scientific topics are less likely willing to test it with their classes.

Figure 3: Programming the behaviour of the components of a work of art.

In a previous work (Federici et al., 2015) we have proposed a way to introduce teachers to coding by means of a reverse process coming from students. In order to overcome possible reluctance, we have now designed a new strategy that can be taught of as an introduction to the programming-based paradigm by means of an environment modelled on the teacher's interests. A new simplified programming language, reminiscent of the structure of the topic to be taught, is designed in order to make students and teachers aware of the mechanism of coding, but without the necessary effort of fully understanding all the basic -but, anyway, somewhat complex- concepts of computer programming (figure 1).

By using a special-purpose tool, the design of the new language is not extremely complex by itself, as it can be done by someone that has learnt how to use a block-programming environment just for a few weeks. But the final new programming language is extremely simpler to use and to be modified by someone that does not know much about computer programming.

Whereas in the previous approach a full programming language is used to describe the behaviour of the components of the topic, this new approach hinges on the compositional properties of a programming language that is now used in order to describe the components itself, leaving the description of their internal behaviour to mechanisms that are hidden from the students and the teacher.

It is important to note that the English learning tool we have devised here, by applying block programming to a very different topic such as English as a second language learning, is not intended to become a full-fledged tool to learn English. The final purpose indeed is just having a simple tool to introduce students to the basics of the English language and to bridge teachers to the open-ended possibilities of coding (Federici et al., 2015). This approach is similar in some sense to simple block programming tools (such as block; Federici, 2001) to introduce students to the basics of computer programming by overcoming their usual initial difficulties.

2 USAGE OF INTERACTIVE TOOLS FOR LANGUAGE LEARNING

The study of the usage of interactive tools in foreign language learning, and how these tools can improve student's performances, is something that has been analysed many times (Atkinson, 1972; Levy, 1997; Warschauer, 1998; Beatty, 2013). Specific studies concentrated on foreign language acquisition at the level of primary school (Neri et al, 2008; Chang et al, 2010; Han, 2012; Pathan and Aldersi, 2014; Moreno-León and Robles, 2015).

Learning a foreign language is something that hinges on several different abilities: retaining the correct translation of words, such as nouns, adjectives, verbs; remembering how nouns and adjectives are organized in noun groups; remembering how noun groups are organized in sentences by using verbs. Several studies have suggested different ways of facilitating the acquisition of a foreign language at the primary school level, from robots (Chang, 2010; Han, 2012) to games (Pathan and Aldersi et al., 2014) to stories (Moreno-León and Robles, 2015). All this approaches hinges on the *active learning* paradigm (Prince, 2004; Bachelor et al, 2012) where users must actively take part in the learning process.

What we propose in this paper is along the lines of active learning teaching and programming-based learning (Federici, 2018). Indeed students, by using a simplified visual programming language, build a syntactic structure that creates a drawing of the described situation. By using this strategy, students not only have the chance to test the meaning of the individual foreign language "components", but they can also *see* if the final English sentence corresponds to the phrase or sentence that they are translating (figure 4).

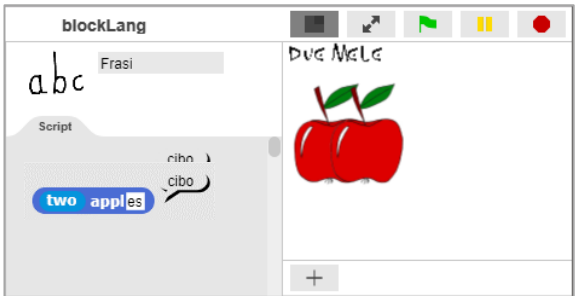


Figure 4: Learning the meaning of *two apples.*

Students must fully understand how all the parts of the English phrase or sentence fit together, in order to correctly create them by means of a simplified programming language.

In order to test this hypothesis, we have run an experiment on the students of two 2nd grade classes. The specific domain selected for the experiment was one that the teachers felt as particularly meaningful for their young students, namely the domain of food.

Our working hypothesis was that, by assembling language elements in an interactive way, students will remember better their relative position, by putting at work different learning strategies at the same time (Seemüller et al., 2012; Udomon et al., 2013).

Furthermore, in this tool the programming elements of the same category were all based on a common underlying structure that even a foreign language teacher, with no prior knowledge of computer programming, can easily update and reuse, increasing the number of linguistic components proposed by the tool and enlarging the student's vocabulary.

3 LEARNING PHRASES AND SENTENCES FROM THE FOOD DOMAIN IN A PRIMARY SCHOOL

Young Italian learners of 2nd grade use mostly oral explanations. They learn very common words, such as apple, orange, bread and very simple sentences such as "I have one apple" or "I like apples".

The most common mistakes made by young Italian 2nd grade learners are of a different nature when they must cope with open vs closed questions. In Italy, 2nd grade students use mostly oral tests or multi-choice questions and their written tests, if any, are usually fill-the-gap exercises with only one or -at most- two words to be filled-in for each phrase or sentence.

As to short *open* questions, the errors are mostly due to:

- spelling
- forgotten words

Errors in plurals or in using the wrong word tend to be rare.

Spelling errors are likely due to the oral nature of their learning, so that they tend to Italianise the spelling, by using the Italian transliteration of the pronunciation of the word, e.g. writing

- *carot* instead of carrot
- *fisC* instead of fish
- *milC* instead of milk
- *KEIK* instead of cake

As to forgotten words, they tend to forget words that have no Latin root, that is words that are very dissimilar from the corresponding Italian word, e.g.

- like (Italian *piacere*)
- milk (*latte*)
- breakfast (*colazione*)
- lunch (*pranzo*)
- dinner (*cena*).

When it comes to *closed* question instead, they

- very rarely choose the wrong translation of single words even when they are very different from the corresponding Italian ones. Likely, the suggestion they get is enough to them
- they tend to choose the wrong verb in sentences where the correct verb is not the literal translation of the verb used in the Italian sentence.

So, for example, in Italian people *do* a meal (instead of *have*), they *have* something (instead of *have got*) and they *not do* something (instead of *do not do*).

4 THE EXPERIMENT: "PROGRAMMING" ENGLISH SENTENCES

The experiment involved a total of 36 students from two 2nd grade classes from a local elementary school. The experiment started about at the end of the school year, when the students were studying specific words and sentences from the food domain, and was run during school hours already devoted to English. We followed what in the previous experiment has demonstrated a good protocol in preparing the classes (Federici, 2018):

- the teachers of the school selected two 2nd grade classes that they thought were roughly similar, basing on their general skills
- one of the two classes was the *test* class where students would have been building English phrases and sentences from the food domain by means of the simplified programming language. The other one was the *control* class, where students would have been learning phrases and sentences from the food domain by following standard explanations (see Table 1)
- not to interfere too much with the completion of the explanation of all English topics, the time spent in studying food domain words and sentences was limited to 3 sessions of 2 hours each (that is about 1 hour of real work for each session)
- a preliminary session of 2 hours was devoted to introduce the students to block programming principles
- the test was done after the 4th session

Table 1: Learning in the different groups.

Group	Learning
Test	Introduction to block-programming Creation of interactive phrases and sentences
Control	Standard lessons

From the test group we wanted to clearly understand if students could get good performances in translating English by allowing them to build interactive phrases and sentences in a fun way that, when "run", will automatically create visual representations of the phrase or sentence. At the same time the students were acquiring the basics of coding.

It is worth noting that students in the test group, in the end, spent significantly less time in exercising on phrases and sentences of the food domain than their peer in the control group, for two reasons: the first one is that their peers started studying the English food domain while they were introduced to block programming; the second one is that each session in the computer lab required a significant amount of the lesson time to be spent in technical operations.

As already said, an important part of the experiment was to make sure that the internal working of the programming blocks -created to represent English phrases and sentences- was programmed in such a way that even an English teacher, with no prior knowledge of computer programming, could easily update the set of phrases and sentences of the food domain in the simplest and more intuitive way.

4.1 Teaching 2nd Grade Students How to Manage Visual Programming Blocks

In order to allow the students to learn how to build interactive English phrases and sentences they were first exposed to projects created by using a programming language and then they were taught the basics of computer programming. As mentioned above, we choose a programming language based on the block metaphor, Snap (Harvey and Moenig, 2010, very similar to Scratch) specifically designed to easily teach computer programming to children 8-11, and that allowed for the easy creation of colourful interactive objects.

The structure of a programming environment for a visual block language is very easy and quick to grasp (figure 5).

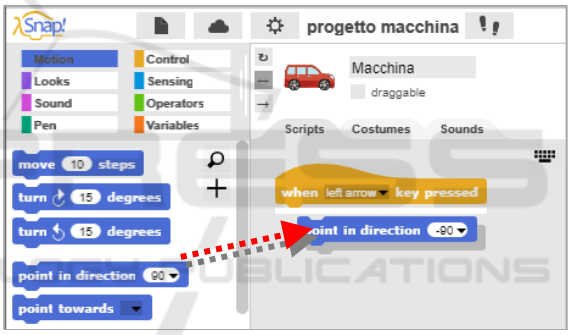


Figure 5: Creating block sequences in Snap by dragging-and-dropping blocks from the *block area* (to the left) to the *script area* (to the right).

All "instructions" are represented by coloured blocks that are visible in the *block area*, at the left-hand side of the window). By dragging the blocks from the block area to the central part of the tool (right area in figure 5), users can build sequences of blocks, called "scripts", for their characters to behave and interact as expected.

Blocks are organized in *categories*. Like building blocks, they are grouped in different bins with respect to their color/function. Blocks in each category can be accessed by clicking the desired button right above the block list, at the top left. Among the available categories in Snap we find Movement (to move the characters of the project), Looks (to change their appearance), Control (to make available basic programming structures such as repetition of a given behaviour), Sensing (to allow Snap characters to "sense" their environment), etc.

The organization of blocks in categories is important, so that users can quickly find the desired block by its function.

4.1.1 First Session: Arousing Enthusiasm

In the first session, as in the previous experiment, we allowed students to play with several projects created with Snap. Having only 4 sessions available, we had to arouse students' enthusiasm very quickly towards the possibility of building interactive objects that would behave as expected when executed in the block programming environment. The first projects were minigames that allowed them to move several characters on the Stage of the programming environment (see for example the seagull in figure 6, reminiscent of the *Angry Birds* character).

Figure 6: Moving-around in the *Wind Wings* minigame.

4.1.2 First Session: Introducing the Students to Block Programming

The main project shown in the first session aimed at showing to the students how they could move programming blocks around and how to assemble them in a "script", in order to get the correct result on the Stage once the script was executed.

Figure 7: Snap minigame to learn block programming.

For example, to build the "car" project (figure 7), the students had to drag blocks from the left *block area* to create the two scripts shown in the central *script area* so that the red car in the top right *stage area* moves left and right when the user presses the left/right arrow keys. The behaviour of each script can be also triggered by directly clicking the script.

Assembling scripts and activating them by click are the exact same mechanisms the students had then to use in order to assemble and "run" English phrases and sentences.

At the end of the first session they were eager to start using the tool to study English too.

4.1.3 Second Session: Learning How to Handle Words of the Food Domain by using Colored Blocks

In the second session, the students started using the tool specifically created for them in order to build English phrases and sentences by assembling colored programming blocks and by running them. The tool, that we called BlockLang, has been built by using BloP (*Block Programming* environment, Federici and Gola, 2014), a modification of the Snap environment that allows to easily build custom block-programming environments. Even if Snap already allows for some important customization features, such as creating new custom blocks, BloP is specifically designed so that the new tool created with BloP can be safely used even by young users:

- without the worry of impairing the environment by doing something inappropriate
- by organizing the new custom blocks in new meaningful categories
- by hiding "support sprites", that is sprite specifically created to handle the task at hand, but that must not be moved or be removed by the users

The BlockLang interface is very similar to the Snap environment (figure 8).

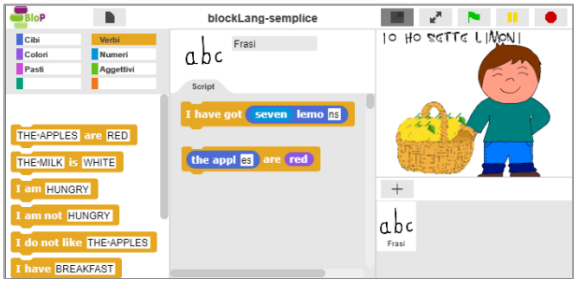


Figure 8: The *BlockLang* environment: the categories (at the top left), the blocks (at the bottom left), the scripts (in the center) and the Stage (at the top right).

The interface is fully in Italian (except, of course, for the English phrase/sentence blocks) so that young students are not overcharged by too many English items to learn in advance. It is worth noting that, thanks to the customization features offered by Snap, the interface can be easily translated to other languages so that the tool can be used to learn English food phrases/sentences by students of other foreign languages.

At the top left corner, we see the new block categories -specifically created for BlockLang thanks to the BloP features- for organizing the word/phrase blocks into meaningful groups (figure 9).

Figure 9: *BlockLang* block categories.

The BlockLang categories to handle food domain phrases and sentences have been enriched by adding a few words already known by the students, such as *color names* and *numbers*. The BlockLang categories are then:

- *Food*, that is both singular (both countable and uncountable) and plural names of food items, such as *apple*, *carrot*, *cheese*
- *Colors*, names of colors for food items
- *Meals*, names of main meals, that is *breakfast*, *lunch* and *dinner*
- *Verbs*, that is verbs related to the food domain such as *have* (for meals) and other base English verbs, such as *be*, *have*, *like*
- *Numbers*, numbers from 1 to 10
- *Adjectives*, that is adjectives related to the food domain, namely *hungry*

In this very first English session, the students learnt how food blocks (that is blocks from the *Food* category) create visual representations on the Stage. For example, the "the bread" block when dragged to the script area and "run" by clicking it, it draws a loaf of bread on the Stage together with its Italian translation "il pane" at the top of the Stage (figure 10).

Each food, when corresponding to a countable item, has a "companion" block with a fillable gap for an argument representing the *number* of items. For example, we have the *the apple* block but also the *ONE apple* block. Note that redundancy, in block languages, is a very well-accepted mechanism, largely used by Scratch and Snap. The capital letters are just a cue to the students that the gap must be filled by another block. The word *ONE*, for example,

Figure 10: Running a *Food* block.

suggests that the type of the blocks that correctly fills the gap is a number. If the students try to run the block without filling the gap, they see an error message on the Stage (figure 11).

Figure 11: Warning message "you must fill gap ONE with a block!".

Instead, when the gap is correctly filled by a block from the Number category (e.g., the block "two"), the final argument "e" is replaced by typing the correct singular/plural ending (in this case "es") inside the gap and the script is clicked, the correct result is shown on the Stage (figure 12).

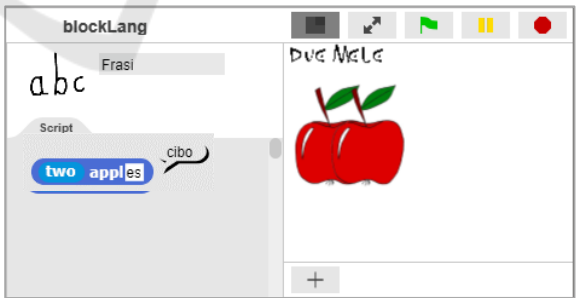


Figure 12: Running the "*two* apple*s*" script after filling the gap "ONE" of the "ONE apple" block with a "two" block from the Number category and the "e" gap with "es".

The correct translation "due mele" shows up at the top of the Stage and two apples are drawn on the Stage.

Every script that does not contain unfilled gaps can be also run, so that its visual meaning is shown on the Stage. So, even running the "two" block will

show the number 2 on the Stage (figure 13) and running, e.g., the "dinner" block from the Meals category will draw a classic dinner scene on the Stage (figure 14).

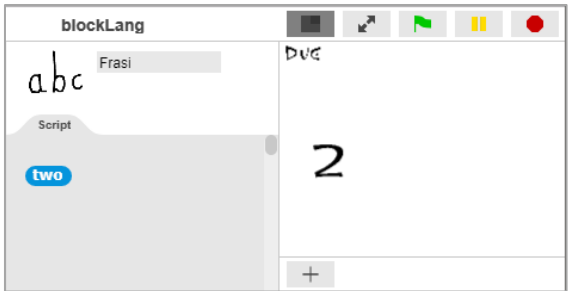


Figure 13: Running the "two" script.

Figure 14: Running the "dinner" script.

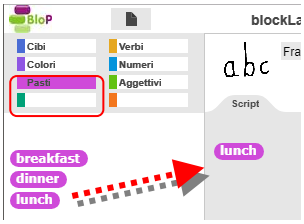
At the end of this session, many students were able to correctly anticipate the category where to find the correct block corresponding to a given Italian word. They were also able to assemble simple nominal constituents such as "three lemons" or "four ice creams" and to show the outcome on the Stage by running the corresponding scripts. Every time a phrase/sentence was shown on the overhead projector the correct pronunciation of the item was practised.

4.1.4 Third Session: Learning All the Words from the Experiment Vocabulary

In the third session the students revised what they had learned in the first session and learned (or revised) all the words in the Food, Colors, Meals and Numbers categories.

In this session we were showing on the overhead projector of the classroom the Stage of our tool in which they could read the Italian sentence and look at the corresponding drawing. For example, looking at the noun phrase "pranzo" they were guided to go to the *Meals* category and drag and run the *lunch* block on the script area (figure 15).

Stage for *lunch*



The *lunch* block of the *Meals* category

Figure 15: Running the *lunch* block.

At the end of this session, many students were able to correctly anticipate the category and the block/script that would create on the Stage the objects/scene that was shown to them on the overhead projector. Every time a new phrase/sentence was shown on the overhead projector the correct pronunciation of the phrase/sentence was practised.

4.1.5 Third Session: Learning Phrases and Sentences based on Words from the Food Domain

Due to the limited vocabulary usually learned by 2nd grade students, the phrases and sentences in this experiment on the food vocabulary could not be very different from the ones they had learned so far. The experiment has then been based on the following phrases/sentences:

- I like/don't like *Food*
- I have got *Food*
- I am/am not hungry
- *Food* is/are *Color*
- I have *Meal*

The students then practised how to build sentences like *I like the bread*, *I have got seven lemons*, *I am hungry*, *the chocolate is brown*, *I have breakfast*. Note that, due to the stepwise learning of the English language in the Italian primary school, the students were thaught to always use the article "the" in front of names -as it happens in Italian- when there was no cardinal number. So, they learned to say *I like the lemon* instead of *I like lemon*, as if they were always referring to a specific item of food. That is why, in the BlockLang tool, there is no single *lemon* block, but only the *the lemon* and *ONE lemon* blocks.

All the new blocks in this session were from the Verbs and Adjectives categories. So, for example, by dragging, assembling and running the *I have BREAKFAST* block from the Verb category and the *dinner* block from the Meal category they could

build the sentence *I have dinner* (figure 16), that is the correct translation of "Io ceno".

Figure 16: Running the *I have dinner* script.

In the same way, by using the *THE MILK is WHITE* block from the Verb category, the *the lemon* block from the Food category, and the *yellow* block from the Color category, they could build the sentence *the lemon is yellow* (figure 17), that is the correct translation of "il limone è giallo".

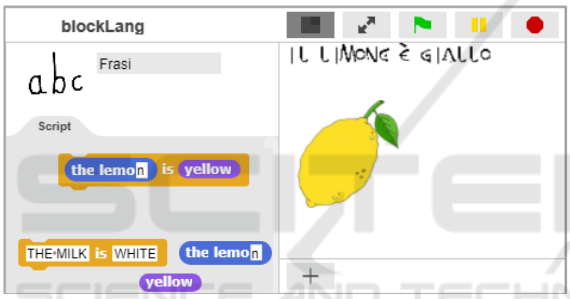


Figure 17: Running the *the lemon is yellow* script.

4.1.6 Fourth Session: Reinforcing the Food Domain

In the fourth and final session, the students reinforced the knowledge of phrases and sentences from the food domain by practising how to build phrases and sentences using the BlockLang tool just by looking just at the objects/scene and at their Italian translation shown on the teacher's Stage and without any suggestion by the teacher. They were serious and followed all the teacher's instructions very carefully as they knew that, in the next session, they would have had a test.

In order to get the correct translation of a full sentence they had to:

- identify the correct category for each word in the sentence
- drag the correct blocks from the correct category into the script area
- assemble the blocks by filling all the gaps in order to build the correct script

So, for example, in order to build the correct translation of the sentence "il limone è giallo" (*the lemon is yellow*) they had to drag the *the lemon*, *yellow* and *THE MILK is WHITE* blocks from the Food, Colors and Verbs categories respectively. The they had to assemble them in the script area in the correct order, suggested by the uppercase fillers, that is *THE MILK* and *WHITE*, of the *is* block. Finally, they had to run the script in order to check, on the Stage, if the correct sentence and the correct drawing were showing up on the Stage (figure 17).

When they were doing mistakes, they were getting a warning and/or a non-matching drawing or a non-matching sentence on the Stage. For example, if they were choosing the verb *are* instead of the verb *is*, they were getting both the warning "the verb ARE needs a PLURAL food!" and an incomplete "il limone" (that is the lemon) instead of the "il limone è giallo" (the lemon is yellow) on top of the Stage (figure 18).

Figure 18: Results of running the wrong sentence *the lemon are yellow*.

Instead, if they were using the *the lemon* and *yellow* blocks in the wrong order, they were still getting and incomplete sentence at the top of the Stage and the warning "fill the THE MILK gap with a FOOD block!" (figure 19).

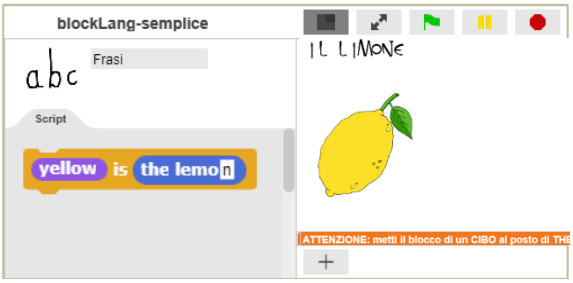


Figure 19: Results of running the wrong sentence *yellow is the lemon*.

4.2 Strengths and Weaknesses of using the Blocklang Tool

The effective support of an interactive tool in learning a given topic depends on the "freedom" you have in creating what you think is a correct item for the topic at stake. Of course, the more freedom you have, the higher the complexity, the more time you need to fully learn the task. Conversely, less freedom -that is less possibility of doing mistakes- corresponds to a lower number of features of the topic that you can learn by using the tool, but it also requires a shorter time needed to learn those features. In the BlockLang tool the learnable features related to phrases and sentences from the food domain have been limited, on purpose, to the following list:

- recognition of the correct word (that is correct noun, verb, adjective)
- usage of the correct ending for singular/plural words
- usage of the correct construction for noun phrases containing numerals
- correct agreement in number between noun group and verb
- correct construction for sentences

So, just to give a few examples, the student had to:

- choose *nut* instead of, e.g., *cake* when translating "noce"
- use *nuts* instead of *nut* when translating the plural form
- use *ten* instead of *the* when building a noun group containing a numeral, like *ten bananas*
- use *are* instead of *is* when building a plural sentence like *the carrots are orange*
- use *I have lunch* instead of *I lunch* when translating a sentence indicating that I'm having a meal

So, by using the available building blocks, they could not practice the correctly *spelling* of a given word, nor they were forced to think of the correct word to use as a correct translation. They could just test the available blocks, until they saw the correct translation and drawing on the Stage. So, they learnt how to *select* the correct word among the (little amount) of available ones, *not* how to immediately *associate* each Italian word to the corresponding English word.

When they were applying the wrong feature (that is word, singular/plural, agreement, construction, etc) they got a double feedback: an orange warning message at the top of the Stage and/or the wrong drawing on the Stage. By quick trial and error, they could build the correct translation, so they didn't get frustrated and they could -hopefully- get the correct translation in a shorter time for each new phrase/sentence to translate.

4.3 Blind-testing Knowledge about English Phrases and Sentences

After the fourth session had ended, the knowledge acquired by the students of the two groups about words, phrases and sentences of the food domain has been tested by giving them a test with 12 closed questions and 11 open questions. The test was done completely on paper, so that the students of the control group, used to paper and pencil, could be on a par with the students of the test group. On the other side, students of the test group were in a slight difficult position as they were used to complete this kind of test by using the tool.

The test was prepared in collaboration with the teachers and administered by the teachers of the two classes. The results of the test were anonymized (both for privacy reasons and for blind-testing purposes) and then sent to us. The teachers, due do the young age of their students, insisted on having a test made of closed questions as they didn't think their students could be able to correctly answer to open questions, being them used at most to fill-the-gap assignments. But we wanted to see if using a block programming-based approach could have a major impact on learning the "structure" of the English phrases. So, we asked to add a further test based on open questions, even if this was something that the students were not used to.

The *closed* questions were in the majority two-words noun phrases (7 questions) where the students had to select the correct food domain word, and short sentences (5 questions) where the students had to select the correct verb or the correct word order. The *open* questions instead were mostly two-words noun phrases (9 questions) and a few short sentences (2 questions) where the students had to write down the correct translation.

Taking into account the features of the tool, we were expecting the test class doing well in those tasks that were better supported by the tool, namely using the correct singular/plural form, the correct verb and the correct construction. Instead we were expecting them to do less (hopefully, slightly) well in the tasks less or not supported by the tool, like writing down a word by using the correct spelling or even remembering the correct translation of each word in the phrase/sentence.

Taking into account the common mistakes made by young Italian students of 2nd grade, the errors made by the students of the two classes were due to:

- spelling
- wrong singular/plural
- wrong word
- missing word
- wrong word order/verb structure

Due to the different nature of the two tests, the distribution of errors was different in the two tests, as shown in Table 2.

Table 2: Distribution of the different sources of errors.

Closed questions		**Open Questions**	
wrong spelling	nd	wrong spelling	30%
wrong sing/plur	nd	wrong sing/plur	11%
wrong word	48%	wrong word	8%
missing word	nd	missing word	50%
wrong word order	52%	wrong word order	1%

4.4 Test Results

The results followed the patterns foreseen when taking into consideration the features that the students could learn by using the BlockLang tool, but the differences were less than expected. Indeed, students in the test group found slightly more difficult to remember the correct translation of food domain words with respect to the students in the control group even if they were used to search for those words by trial and error and hadn't had time to practice the recall of the correct translation of a given word without using the tool. Instead, as expected, they did much better than the control group when it came to choose the correct word order or the correct structure of the sentence.

So, in the first test based on closed questions the students in the control group got a better overall score (Table 3) as they did slightly worse at remembering the correct words (22% of error vs 21% of errors in the control group) when they had to choose, e.g., between *nut* and *cake* as the correct translation of "noce", but they only made 27% of errors when they had to choose for the correct word order choosing, e.g., between *orange are the carrots* and *the carrots are orange* or when they had to choose the correct structure, e.g., between *I not like the cakes* and *I don't like the cakes* . Instead, in the control group the error rate for these sentences was 40%.

Table 3: Errors on *closed* questions.

Test group		**Control Group**	
wrong word	22%	wrong word	21%
wrong word order	27%	wrong word order	40%
GLOBAL	*24%*	*GLOBAL*	*29%*

Instead, in the second test, based on open questions, the students in the control group got a better overall score (Table 4).

Table 4: Errors on open questions.

Test group		**Control Group**	
wrong spelling	44%	wrong spelling	37%
wrong sing/plur	15%	wrong sing/plur	15%
wrong word	8%	wrong word	12%
missing word	81%	missing word	55%
wrong word order	0%	wrong word order	1%
GLOBAL	24%	GLOBAL	21%

There are two interesting things to note. The first one is that even if the test group, as expected, does not remember a lot of translations (81% of phrase/sentences have at least one *missing word*), nonetheless when they use a word, they are more accurate than the control group (only 8% of wrong words instead of 12%). Second, if we don't take into account the features whose training is not specifically supported by BlockLang (that is spelling and rote learning of single word translations) the difference between the two groups decreases sensibly, with even a slight superiority of the test group, that is 8% vs 9% (Table 5).

Table 5: Errors on *open* questions (revised).

Test group		**Control Group**	
wrong sing/plur	15%	wrong sing/plur	15%
wrong word	8%	wrong word	12%
wrong word order	0%	wrong word order	1%
GLOBAL	8%	GLOBAL	9%

Even if the students in the test group don't remember well the translation of each word, we are confident that this can improve when using the tool for a longer time. In any case, the rote memorization of a word is something that cannot be "seen", so it is beyond the purposes of the BlockLang tool. Memorization is something that can be reached by repetition. The students in the test group could exercise only for less than half of the time of the other class (namely 3 hours instead of 8).

5 EASILY INCREASING THE TOOL VOCABULARY

Students have shown that they can learn with a lower error rate the features of an English language topic when it is taught by means of an interactive tool based on the block programming metaphor. But today interactive tools for teaching English are a widespread strategy. If the tool would need an expert programmer every time the teacher wants to extend the tool vocabulary by adding a new word, the usage of the tool would be likely very limited. The strategy of creating a full-blown tool for building English sentences, with all the necessary words already plugged in, would not be a good solution as it would make finding the correct block very difficult, especially for young students.

Even if the Snap tool is a general programming tool, its block-programming metaphor allows just after a few weeks to build very complex projects even to people that didn't know how about computer programming in advance. However, this does not mean that an English teacher would want to invest even a few weeks to learn block-programming. So, in order to make the BlockLang tool more user-friendly, we set up the BlockLang tool so to encapsulate the scripts' complexity inside several general blocks (e.g. the *ARTICLE FOOD* block) that allow the English teachers to start learning the basics of block-programming very gradually by quickly updating the tool.

In order to add a new noun phrase, the steps needed are just the following:

1. unlock the BlockLang tool
2. duplicate a block of the desired category
3. change the block features and add the necessary visual elements
4. lock the BlockLang tool

So, for example, if we want to add to the tool the *cookie* food, after unlocking the tool by shift-clicking the BloP icon at the top left corner and selecting "Unlock GUI" (figure 20), we can just duplicate a similar countable food block, e.g. *apple* (figure 21) by right-clicking it and selecting "duplicate block definition…" in the contextual menu.

Then we can change each element of the block name (e.g. the *appl* root, figure 22) and each element of the block definition (e.g. the "mela" translation of the singular form, figure 22) by clicking it and updating its value.

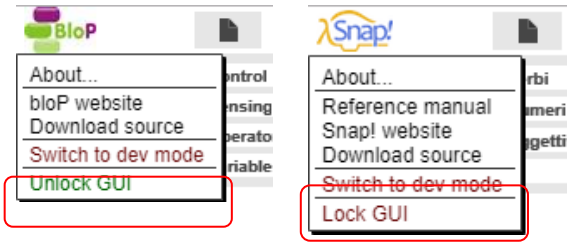


Figure 20: Locking and Unlocking the BlockLang tool – steps 1 and 4.

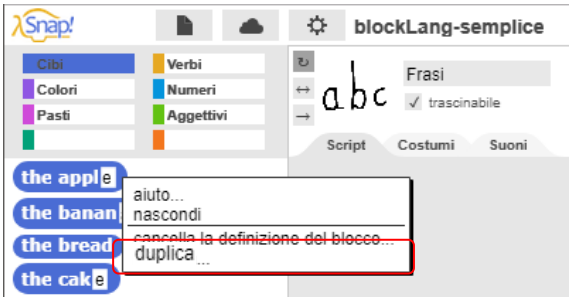


Figure 21: Adding new Food blocks – step 2.

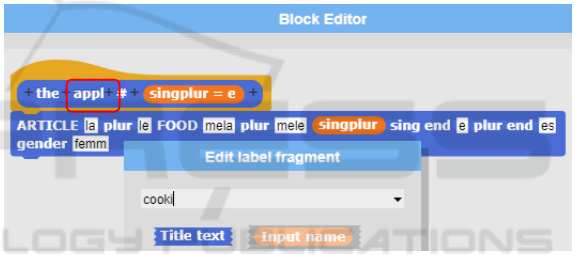


Figure 22: Adding new Food blocks – step 3a. Changing root (from *appl* to *cooki*).

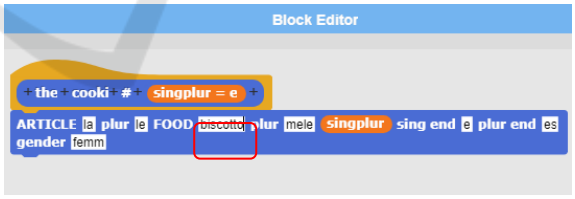


Figure 23: Adding new Food blocks – step 3b. Changing singular form (from *mela* to *biscotto*).

The block definition for a countable noun (Figure 23) contains the following information: article for the singular form, article for the plural form, singular form, plural form, ending of the singular form, ending of the plural form, gender.

When adding a new Food item, we need a "costume" for it, that is an image that will show up on the Stage when we run the corresponding block. Adding an image to BlockLang is straightforward:

just drag the image to the “Costume” tab of the tool (Figure 24).

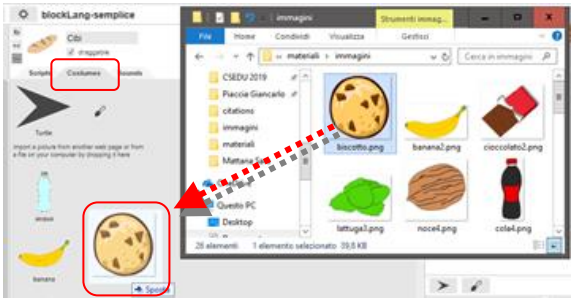


Figure 24: Adding new Food blocks – step 3c. Adding an image for the new food.

Now we have just to lock the tool back up, by shift-clicking the tool logo again (figure 21).

There are other possibilities of customization. For example, if we want to add/create several new words, but don’t want to make them all visible to the students (e.g. to allow them to learn new words in a stepwise process) we can individually or globally hide/show the blocks in the categories by using the right-click menu (figure 25).

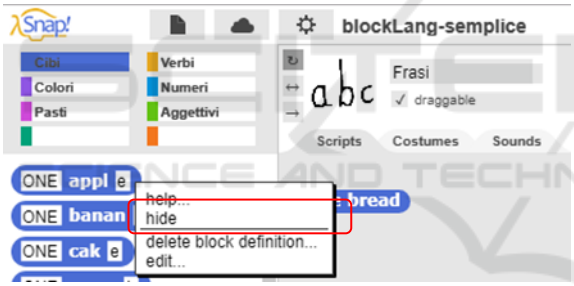


Figure 25: Showing and hiding custom blocks.

6 PROGRAMMING-BASED LEARNING: ANALYSIS OF THE RESULTS

What follows from the results of the two tests is that even adding a different kind of interactivity, namely operations reminiscent of block-programming, with respect to the classic interactive educational tools for learning English, improves the performance on the features specifically designed in the interactive tool. And this even if, to learn how to use this tool, students devote less time to exercising.

We explore the possible reasons in the following sections. We want just to notice here that programming-based learning does not require, for every new topic, 3 or 4 more two-hour sessions than the standard classroom learning. Indeed, we must remember that the first two-hour session was devoted to introduce computer programming in a block-based environment, and that the first part of the second session was devoted to learning how to use BlockLang. When this computer-supported educational methodology is acquired by the class, part of the time spent in teaching and exercising can be fruitfully replaced by self-exploration of the items made available in the tool.

6.1 Explanation of the Results

The results of this experiment show that even less exercise is not a drawback if it is replaced by other kinds of meaningful activities that gives the students further insight in what is behind the specific topic they are studying. A lot of exercise (more than 8 hours spent in just translating food domain phrases/sentences) proves certainly effective for a short-term evaluation. But, as shown in previous experiments, we think that when time passes by, students tend to forget the constructions they have learnt because they have just memorized them, but they have not deeply internalized how a noun phrase, a number, an agreement or a verb really works in an English sentence. Instead, by assembling the basic items to form a sentence and by getting an immediate visual feedback, the students are forced to *see* the elements that corresponds to the English translation in a dynamic way.

6.1.1 Handling Interactive Objects to Understand and Remember a given Topic

To allow the student to test their knowledge about English phrases and sentences we could build an interactive project that will ask them just to click each word in the translation in the correct order.

However, this is not much different from writing down the translation on a piece of paper. On the contrary, this tool, even if likely more fun, would be an even weaker training than writing down the translation as the user would not train the spelling of the word.

A good way of using a visual programming language such as Snap is instead to create an interactive model of the problem by creating interactive objects for each single component of the problem. Those objects must be assembled in the correct way so adding memorization of the dynamic process. To create the correct phrase/sentence the student will have then to know which kind of

component must be used to fill a gap and move the component to the right gap.

All these elements are clearly visible in the list of blocks of BlockLang (figure 26) so for the student they are tangible objects whose corresponding "real object" can be seen in the drawings shown on the Stage.

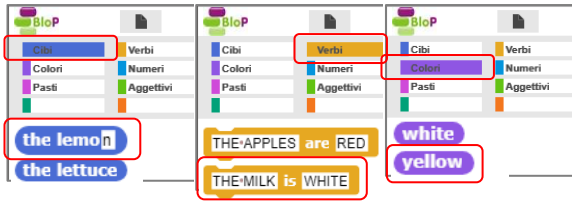


Figure 26: Interactive objects clearly visible in the block categories of BlockLang.

Several different learning strategies are working in this case together (Udomon et al., 2013, Seemüller et al., 2012) to build an interactive virtual model that will help the student to improve the recall of the topic. Indeed, each element of the correct answer (each noun, each verb, etc) is "physically" represented in the project by an interactive object that can be *seen*. Furthermore, each element must be "physically" *manipulated* by the student (for example by selecting it and by dragging and dropping it) in order to correctly place it in the script area.

6.2 Applying Programming-based Learning to Other Disciplines

The strategy discussed in this paper, that allows students to acquire a deeper understanding of English words, phrases and sentences, in our view is not limited to foreign languages. We think that every task that has a "linguistic" structure can be mimicked by building blocks that are reminiscent of visual programming languages.

Just to give an example, a deeper text comprehension (a topic that is currently under continuous investigation at all school levels) could be reached by using the common tools of computer programming to make students understand the underlying meaning of constructs such as *every time that* (cycle) or what does it mean use a word like *eat* to mean a complex list of steps (that is opening the mouth, inserting food inside, etc.

7 CONCLUSIONS

In this paper we illustrated the positive outcomes of a recent experiment in two 2nd grades classes proving that the metaphor of block programming can be introduced as an effective strategy to improve the performances and the interest of the students even for topics apparently so far from computer programming like learning a foreign language.

The devised strategy is not limited to language learning but can be fruitfully applied to further linguistic disciplines.

The strategy is simple enough so that even teachers of non-scientific disciplines can use a simplified version of computer programming in order to build their own multimedia and interactive tools.

ACKNOWLEDGEMENTS

Stefano Federici and Elisabetta Gola express their gratitude for the support of Fondazione Banco di Sardegna within the project ``Science and its Logics: The Representation's Dilemma", Cagliari, number: F72F16003220002.

All authors would like to thank the *Istituto Comprensivo n°2* school in Sinnai and the teachers Debora e Donatella for their help in designing and administering the experiment.

REFERENCES

Atkinson, R.C., 1972, Optimizing the learning of a second-language vocabulary, in *Journal of Experimental Psychology*, 96(1), 124-129.

Bachelor, R.L., Vaughan, P.M., Wall, C.M., 2012, *Exploring the effects of active learning on retaining essential concepts in secondary and junior high classrooms*, dissertation thesis, School of Education, Saint Xavier University, Chicago, Illinois.

Beatty, K, 2013, *Teaching & researching: Computer-assisted language learning*, Routledge, Taylor and Francis Group, New York

Boulton, K., 2013, Why is it that students always seem to understand, but then never remember?, in *toTheReal*, https://tothereal.wordpress.com/2013/05/06/why-is-it-that-students-always-seem-to-understand-but-then-never-remember/ (last retrieved on 29/10/2017).

Chang, C.W., Lee, J.H., Chao, P.Y., Wang, C.Y., and Chen, G.D., 2010, Exploring the Possibility of Using Humanoid Robots as Instructional Tools for Teaching a Second Language in Primary School, in *Journal of*

Educational Technology & Society, Vol. 13, No. 2, pp. 13-24

Costa, S., Gomes, A., Pessoa, T., 2016, Using Scratch to Teach and Learn English as a Foreign Language in Elementary School, in *International Journal of Education and Learning Systems*, Vol. 1.

Federici, S., 2011, A Minimal, Extensible, Drag-and-Drop Implementation of the C Programming Language, in proceedings of *SIGITE'11*, October 20–22, 2011, West Point, New York, USA.

Federici, S., Gola, E., 2014, BloP: easy creation of Online Integrated Environments to learn custom and standard Programming Languages, in proceedings of *SIREM-SIEL 2014, 1st Joint SIREM-SIel conference*. The Innovative LEDI publishing Company.

Federici, S., Gola, E., Brau, D., Zuncheddu, A., 2015, Are educators ready for coding? From students back to teacher: introducing the class to coding the other way round, in *Proceedings of CSEDU 2018 - Proceedings of the 10th International Conference on Computer Supported Education*, Vol. 2, 2018, pp. 124-133, Funchal, Madeira; Portugal

Federici, S., Medas, C., Gola, E., 2018, Who learns better: Achieving long-term knowledge retention by programming-based learning, in *Proceedings of CSEDU 2015 - Proceedings of the 10th International Conference on Computer Supported Education*, Vol. 2, 2018, pp. 124-133, Funchal, Madeira; Portugal

Gresse von Wangenheim, C., Cruz Alves, N., Rodrigues, P. E., Hauck, J. C., 2017, Teaching Computing in a Multidisciplinary Way in Social Studies Classes in School – A Case Study, in *International Journal of Computer Science Education in Schools*, Vol. 1, No. 2, DOI: 10.21585/ijcses.v1i2.9

Han, J, 2012, Emerging Technologies, Robot Assisted Language Learning in *Language Learning & Technology*, Vol. 16, No. 3 pp. 1-9

Harvey, B, Monig, J, 2010, Bringing "no ceiling" to scratch: Can one language serve kids and computer scientists, in *Proceedings of Constructionism 2010*, Paris

Levy, M, 1997, *Computer-assisted language learning: Context and conceptualization*, Oxford : Oxford University Press

Lopez, V., and Hernandez, M. I., 2015, Scratch as a computational modelling tool for teaching physics, in *Physics Education*, Vol. 50, No. 3, IOP Publishing Ltd,

Maloney, J., Resnick, M., Rusk, N., Silverman, B., and Eastmond, E., 2010, The scratch programming language and environment, in *ACM Transactions on Computing Education*. volume 10, n. 4, doi=10.1145/1868358.1868363.

Moreno-León, J., Robles, G., 2015, Computer programming as an educational tool in the English classroom, in *Proceedings of IEEE Global Engineering Education Conference (EDUCON 2015)*, p. 962.

Pathan, M. M., Aldersi, Z. E. M., 2014, Using Games in Primary Schools for Effective Grammar Teaching: a Case Study from Sebha, in *International Journal of English Language & Translation Studies*, 2(2), 211-227.

Neri, A., Mich, O., Gerosa, M, Giuliani, D., 2008, The effectiveness of computer assisted pronunciation training for foreign language learning by children, in *Journal of Computer Assisted Language Learning*, Vol. 21, 2008 - Issue 5

Prince, M., 2004, Does Active Learning Work? A Review of the Research, in *The Research Journal of Engineering Education*, volume 93, n. 3, p. 223-231, John Wiley & Sons, Inc.

Seemüller, A., Müller, E.M., Rösler, F., 2012, EEG-power and -coherence changes in a unimodal and a crossmodal working memory task with visual and kinesthetic stimuli, in *International Journal of Psychophysiology*, issue 83, p. 87-95.

Udomon, I., Xiong, C., Berns, R., Best, K., Vike, N., 2013, Visual, Audio, and Kinesthetic Effects on Memory Retention and Recall, in *Journal of Advanced Student Science (JASS)*, issue 1

Warschauer, M., Healey, D., 1998, Computers and language learning: An overview, in *Language Teaching*, 31, 57-71.