Workshop: A Creative Introduction to Programming with Scratch

Ralf Romeike

University of Potsdam
Department of Computer Science
A.-Bebel-Str. 89
14482 Potsdam, Germany
romeike@cs.uni-potsdam.de

Abstract. This workshop introduces a creative way of teaching computer science. Creativity will be regarded as essential for CS and for motivation and interests of students. The relevance of creativity for CS education will be discussed; criteria for creative CS lessons and a creativity teaching framework are presented. Together we will run through an introduction to programming while exploring and being creative with the programming environment. Scratch offers an intuitive way into programming and leaves lots of space for creativity. Participants will leave the workshop with ideas how to design creative lessons and familiarity in the use of Scratch.

Intended audience

Secondary and early post-secondary CS educators who are interested in challenging their students by applying creativity in the CS classroom and/or want to get to know Scratch.

Presenter Biography

Ralf Romeike is a research associate at the University of Potsdam, Germany as well as a teacher for computer science at a high-school in Potsdam. He studied computer science and music and focuses his research interest on the use and encouragement of creativity in computer science education.

The teaching example used in the workshop has been conducted and evaluated by the presenter. Versions of this workshop have been hold previously by the presenter at two major German conferences for computer science teachers and received good feedback.

Materials provided

Each participant receives both a paper and electronic copy of (1) detailed handouts to be used during the lecture portions of the workshop, (2) descriptions of the creative teaching technique, creativity criteria and the underlying theory (3) A lesson outline for the lesson unit that can be applied in class-room right away, including working sheets (4) an installation routine for Scratch plus material from the Scratch developers (5) ideas, ideas, ideas.

Audio/Visual and Computer requirements

Ideally, participants will have wireless internet access and laptop power at each seat, but the workshop could proceed without these as internet use will not be central to the workshop. The practical part of the workshop will contain exercises and challenges to be solved in Scratch. We will also need a digital projector (for presenters) and a flipchart with pens or blackboard (for collecting ideas). Windows and Mac laptops will be supported.

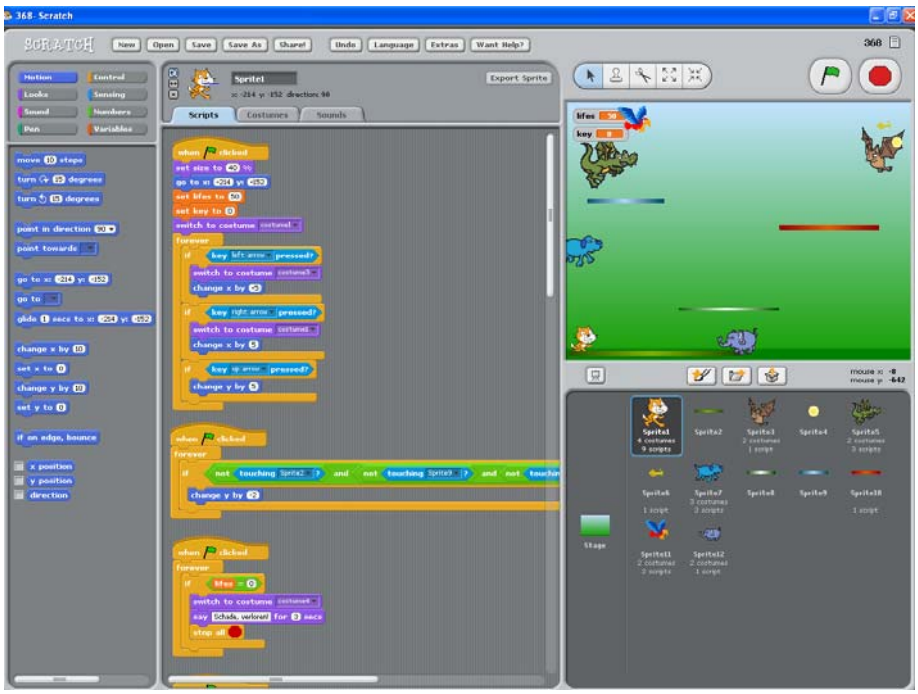

Fig. 1: Scratch programming environment

Laptop

Recommended
Not everybody will need to have a laptop.

Maximum number of participants

Up to 40 participants should be fine. Depending on the facilities even more than that should be possible, just not as cozy.

Other critical information

The teaching unit central to the workshop was awarded the 1st prize of the teachers instruction competition of the German Computer Science Association (GI) 2007.

Brief description of what will be done in the workshop

This workshop introduces a creative way of teaching computer science. Creativity will be regarded as essential for CS and for motivation and interests of students. The relevance of creativity for CS education will be discussed; criteria for creative CS lessons [1] and a creativity teaching framework [2] are presented. Together we will run through an introduction to programming while exploring and being creative with the programming environment. Scratch offers an intuitive way into programming and leaves lots of space for creativity. Participants will leave the workshop with ideas how to design creative lessons and familiarity in the use of Scratch.

Creativity

How shall an educator expect new and original achievements from his students? Boden [3] describes two aspects of creative achievements. Historical creativity (H-creativity) describes ideas which are novel and original in the sense that nobody has had them before. Something that is fundamentally novel to the individual Boden describes as psychologically creative (P-creativity). In an educational context the latter is more interesting and can be aimed for in the classroom. We want to call something creative if it leads to personal new, unique and useful ideas, solutions or insights (cp. [4], [5]). As summarized by [6], in the classroom creativity can enhance learning through improved motivation, alertness, curiosity, concentration and achievement.

Scratch

The visual programming (mini) language Scratch was originally designed for young students to develop 21st century skills [7]. It allows creating animations, games and other programs by “clicking together” programming constructs represented as building blocks. Nevertheless, due to its intuitive appearance and usability it is used in computer club houses, high schools and even in introductory programming college courses. We chose Scratch because it emphasizes practical learning of fundamental CS concepts and at the same time supports the idea of fostering creativity in CS classes. Mini languages are said to provide an insight into programming and teach algorithmic thinking for general computer science in an intuitive, simple but powerful way [8]. Thus Scratch meets the needs for the intended purpose.

References

1. Romeike, R. *Kriterien kreativen Informatikunterrichts*. in *12. GI-Fachtagung "Informatik und Schule - INFOS 2007"*. 2007. Siegen, Germany: Köllen.
2. Romeike, R. *Applying Creativity in CS High School Education - Criteria, Teaching Example and Evaluation*. in *the 7th Baltic Sea Conference on Computing Education Research, Koli Calling*. 2007.
3. Boden, M.A., *The creative mind: myths & mechanisms*. 1990, London: Basic Books.
4. Runco, M.A. and I. Chand, *Cognition and Creativity.* Educational Psychology Review, 1995. **7**(3): p. 243-267.
5. Kaufman, J.C. and R.J. Sternberg, *Creativity.* Change: The Magazine of Higher Learning, 2007. **39**(4): p. 55-60.
6. Fasko, D., *Education and creativity.* Creativity Research Journal, 2000. **13**(3-4): p. 317-327.
7. Maloney, B., Kafai, Rusk, Silverman, Resnick (2004): Scratch: A Sneak Preview. IEEE *Computer Society***,** 104 - 109.
8. Brusilovsky, P., Calabrese, E., Hvorecky, J., Kouchnirenko, A. and Miller, P. (1997): Mini-languages: a way to learn programming principles. *Education and Information Technologies*, **2,** 65-83.