ISSEP 2014 – Istanbul University

The Many Facets of Scratch

Barbara Demo
Informatics Department
University of Torino – Italy

barbara@di.unito.it

Summary:

– Preliminary remark
– need to agree on terminology: mandatory to define what we are talking about
  • ACM&IE Report , Académie des Sciences
– what do we have to teach, where, how ?
– HOW: there is a danger here
– let's analyse what is going on in schools:
  • in my area T4T is teachers for teachers
– short time to train teachers
– previous experiences
  • small robots, csunplugged
– several types of activities using Scratch
– what next?

- preliminary remark:

We are in a *transition time* as for Informatics in schools
→ in all countries big changes:
- mandatory, new curricula in UK beginning this September → see CAS initiatives
- ……..
- optional, an example in Italy

Transition → means
- still under definition what a good curriculum is for the different levels and types of schools
- a (fast) evolving situation thus we must face the need of several and possibly also fast changes of approaches

- Need to agree on terminology

Informatics education: Europe cannot afford to miss the boat

Joint Report Informatics Europe & ACM Europe on Informatics Education - **April 2013**

Informatics Europe:
Walter Gander (chair), ETH Zurich, CH
Antoine Petit, Inria & ENS Cachan, F
Gérard Berry, Collège de France
Barbara Demo, University of Turin, Italy
Jan Vahrenhold, University of Munster, Germany

ACM Europe:
Andrew McGettrick, University of Strathclyde, Scotland
Roger Boyle, University of Aberystwyth, Wales
Michèle Drechsler, INRP, Lyon, France
Avi Mendelson, Microsoft, Israel
Chris Stephenson, CSTA , USA

- Need to agree on terminology

Activities boosted by :

For Informatics Europe: **Walter Gander (chair), ETH Zurich, CH**

ACM Europe and Informatics Europe liaison:

- **Carlo Ghezzi, Politecnico di Milano, Switzerland**
- **Bertrand Meyer, ETH Zurich-CH, ITMO-Russia, and Eiffel Sw - USA**

Published April 2013 **on:**
http://www.informatics-europe.org/

- Need to agree on terminology

- **All of Europe's citizens have to be educated in both**
  - ***digital literacy*** **and**
  - ***informatics***
- ***Digital literacy covers fluency with computer tools and the Internet.***
- ***Informatics*** **covers the science behind information technology having its own concepts, methods, body of knowledge and open issues.**
  It has emerged, in a role similar to that of mathematics, as a cross-discipline field

From Joint Report, page 3

- Need to agree on terminology

Informatics education:
Europe cannot afford to miss the boat

- Informatics education, unlike digital literacy education, is **sorely lacking in most European countries**
- Unless Europe takes resolute steps to change that situation, it will turn into a **mere consumer of information technology and miss its goal of being a major player.**

Recommendations :

1. ***All students should benefit from education in digital literacy, starting from an early age and mastering the basic concepts by age 12.***
2. ***All students should benefit from education in informatics as an independent scientific subject, studied both for its intrinsic intellectual and educational value and for its applications to other disciplines.***
3. ***A large-scale teacher training program should urgently be started. To bootstrap the process in the short term, creative solutions should be developed involving school teachers paired with experts from academia and industry.***
4. ***The definition of informatics curricula should rely on the considerable body of existing work on the topic and the specific recommendations of the present report (section 4).***

- Need to agree on terminology

Informatics in the curriculum

- *fosters **creativity***, **by illustrating the variety of ways to approach and solve a problem**
- *is constructive*: **designing algorithms is engineering work, producing visible (if virtual) artifacts.**
- *helps master complexity*: **learning to solve informatics problems helps solve complex problems in other areas.**
- *enhances accuracy and precise reasoning*: **writing successful programs requires exactness in every detail.**

- Need to agree on terminology

Report Académie des Sciences, "Teaching computer science in France: Tomorrow can't wait", Paris, May 2013

<<The essential decision consists in implementing a programme in computer science from the primary to the secondary-school level, … going far beyond simple use of hardware and software. This implementation can no longer be delayed.
………………………
Teacher training is a top priority. The government proposes massive training of teachers in the uses of digital technologies, but it fails to specify anything in terms of training in computer science.>>

- What do we have to teach, where, how?

Several experiences are well known:

- … **csunplugged** from 98, still popular
- Olimpic Games of Problem Solving, Bebras or Castor, Kangourou,…..
- others where basic programming concepts are introduced already in primary schools with different approaches:
  - EasyLogo, by L. Salanci, Bratislava University
  - educational robotics
  - ….
- what is "computational thinking" in first years of education?

- What do we have to teach, where, how?

Why chosing PROGRAMMING for PRIMARY?

It is the practical, almost tangible, aspect of computer science: you learn how to do something generally felt as "real" computer science

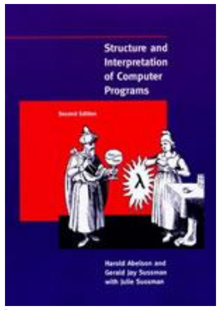
N. Wirth, Program development by stepwise refinement, CACM, April 1971

"The creative activity of programming – to be distinguished from coding- … is here considered as a sequence of design decisions concerning the decomposition of tasks into subtasks and of data into data structures".

programming ⟷ *problem solving*

- What do we have to teach, where, how?

Abelson often remembers:
"Programs are essential to make human beings used to reason in a systematic way on the solution of a problem and to write it in a formal way rather than to make a computer execute a job"



Structure and Interpretation of Computer Programs

V. Barr and C. Stephenson: *Bringing computational thinking to K-12: what is Involved …? **

Developing a definition of, or approach to, computational thinking that is suitable for K-12 is especially challenging in light of the fact that there is, yet, no widely agreed upon definition of computational thinking.

K-12 students already learn how to think and to problem solve, but computer scientists can help teachers understand these processes as algorithmic

K-12 education today is
- a highly complex and politicized environment where many competing priorities, ideologies, pedagogies vie for dominance
- simultaneously subject to wildly diverse expectations, intense scrutiny, and diminishing resources.

** in ACM Inroads archive, vol. 2 no. 1, March 2011, pages 48-54*

V. Barr and C. Stephenson: CREATING A DEFINITION FOR COMPUTATIONAL THINKING IN K-12 *

Embedding computational thinking in K-12 requires a practical approach, grounded in an operational definition, i.e. a definition coupled with examples that demonstrate how computational thinking can be incorporated in the classroom.

We must answer a set of questions focused specifically on K-12 implementation:
- What would computational thinking look like in the classroom?
- What are teachers already doing that could be modified and extended?
- What are the skills that students would demonstrate?
- What would a teacher need in order to put computational thinking into practice?

** in ACM Inroads Archive, vol. 2 no. 1, March 2011, pp 48-54*

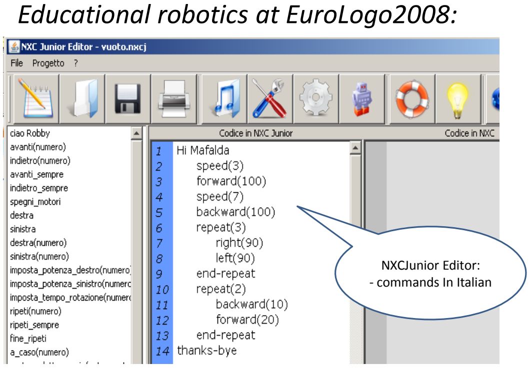
- What do we have to teach, where, how?

CREATING A DEFINITION FOR COMPUTATIONAL THINKING IN K-12

requires we answer a set of questions focused specifically on K-12 implementation:
- What would computational thinking look like in the classroom?
- What are the skills that students would demonstrate?
- What would a teacher need in order to put computational thinking into practice?
- What are teachers already doing that could be modified and extended?
- To be useful, a definition must ultimately be coupled with examples that demonstrate how computational thinking can be incorporated in the classroom.

- What do we have to teach, where, how?

Educational robotics at EuroLogo2008:



- What do we have to teach, where, how?

Concrete programming

- An activity for an NXT robot described the holidays some schoolchildren spent in their summertime:
  - grandmother is not at home?
  - going to somewhere else and then back until she arrives at home.
  - expressed using the repeat-until-grandmother-arrives pattern that becomes the repeat-until-condition-satisfied pattern in the students mind.

HOW to teach programming: there is a danger here

We are used to teach to specialising students

- either to vocational or technical
- or at the university

the same activities though made easier are NOT suitable for all levels and types of schools:

- what kind of problems to propose?
- What do we have to begin with?

What about stories …..

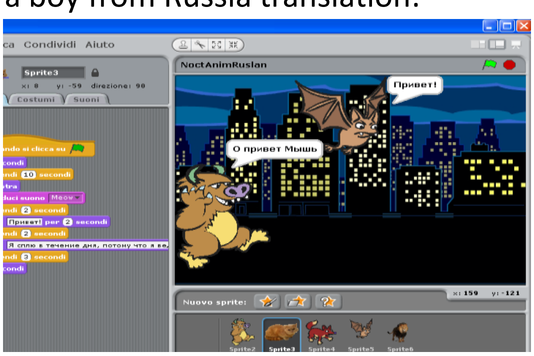
The story named “Nocturnal animals”, has been conceived and developed by an English student 11 years old

In it, a “monster”, out for a walk in a modern city, meets different animals learning that they are going around by night because they are nocturnal animals, but a lion, when challenged, says, “Of course not (nocturnal animal), silly! I am lost!”

HOW: → methodology

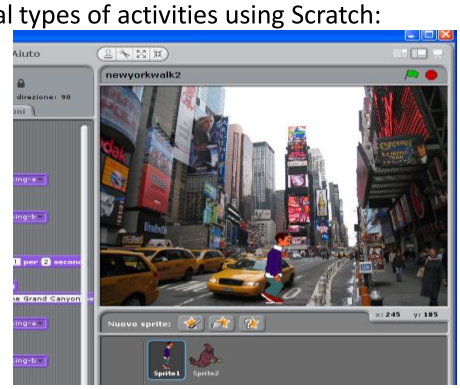
- Story-telling is the way we currently propose for introducing programming
- has proved to be quite appealing for both students and teachers having little cs knowledge
- Begin with a look-seeInside-modify methodology where teachers look at an already running activity with their students, explore the inside of what they have seen and then change it developing a story of their own.

a boy from Russia translation:



Several types of activities using Scratch:

here a first activity in a middle school



- several types of activities using Scratch

Backgrounds for scholchildren performances

An evolution of the stage described in the previous section concerning using if-then and if-then-else commands learnt developing our-on-stories into more sophisticated general stories and producing “stories-with-crossroads” [..], i.e. stories having not only sequential actions yet having a development that depends on user interactions or on other characters interactions

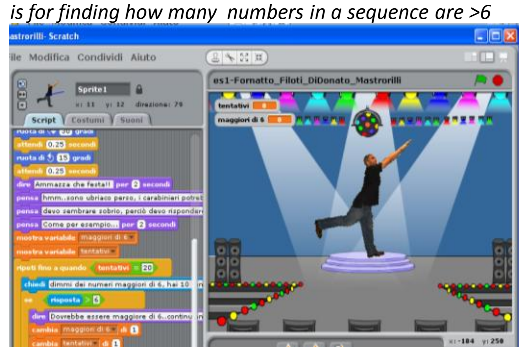
From Story-telling to Telling-my-story

Middle school:

a “variable” is a sprite changing costumes!?

where not yet introduced variables

Students like stories also for traditional algorithms: this one is for finding how many numbers in a sequence are >6



Evaluation of stories:

Name of the Story ☐ Your Name ☐

Literacy from Scratch - Peer Assessment Sheet

Please watch another student’s story and then fill in the rest of this assessment sheet.

Please put a cross in the box below if the student has included this in their story.

☐ Speech ☐ Sounds
☐ Changing Backgrounds ☐ Hide/Show Sprites
☐ Movement of Sprites ☐ A Happy Ending

Was the storyline clear throughout? ☐Yes ☐No

Briefly explain the story in the box below

- is there any spelling mistakes or grammatical errors you noticed here
- how could the story telling be improved?
- how could the scripts (programming) be improved?

Easy examples → Think a number game

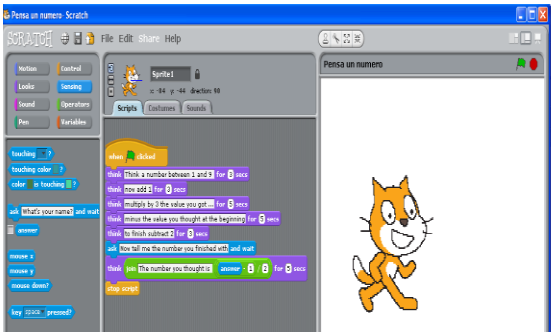
- Think a number
- Add 1
- Multiply by 3
- Subract the number you started with
- Add 5
- Tell me the number you finished with ← a
- Now I can tell you the number you thought at the beginning

$$(x + 1) * 3 – x + 5 = a$$

$$x = (a – 8) / 2$$

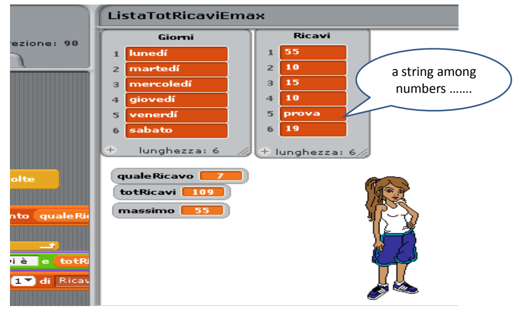
!! an activity with maths teachers

Think a number game



Scratch is a good choice to begin with

but has limits: problems found during activities



Some examples →

High-low-guess-a-number

- Think a number between 1 and 9999
- I will guess your number (with less questions than many humans)

- *Introducing time complexity*
- …………
- *In general: introducing properties of algorithms*

High-low-guess-a-number

- what next?

- *for future teachers in the Technology discipline in our region the department has obtained the course INFORMATICS whose contents are mostly programming in Scratch*

- *A way to solve the ambiguity that in the Italian "SCHOOL INDICATIONS" there is optional programming but teachers are not prepared to teach it*

- what next?

Manifesto for Informatics in schools

What are the different perceptions of Informatics?

- *Operational or Pragmatic*: Hardware & software tools to solve everyday problems

  → *common perception*, you must know how to use some/most popular hw devices and sw tools

  → shared by people who say that Informatics is the set of tools to better understand disciplines in school and thus they (proudly) want computers inside classrooms
- *Technological*: knowing systems and languages in order to implement tools → technical schools perception
- *Scientific* aspect: Informatics is also the (old) science on which sw and hw tools (from the middle of the XXieth century) are funded

What Informatics in secondary schools?

– Programming and problem solving not only from a technical point of view (such as learn coding in a given language) also algorithms properties and systems organisation (allowing students to acquire skills for some certification often offered by schools)
– Interactions with philosophy and history of science particularly in school types where these disciplines have a strong presence, for example in "Licei", but also, in a softer form, in the other school types

What Informatics in secondary schools?

– Interdisciplinarity: many helps from CS in proposing original ways for thinking to traditional concepts from other disciplines
  - Informatics to look at several mathematical concepts from different perspectives
  - Informatics and natural language analysis
  - …….

This is the end of my talk on " The Many Facets of Scratch"

Please write your comments to

barbara@di.unito.it

Many thanks to you for your attention

and many thanks to the teachers and the students working with us

→ they are the many faces of Scratch (for us)