SnapCon19 - Timetable

SUNDAY, SEPTEMBER 22ND	
15:00-17:00	Registration Transferzentrum, Bergheimer Str. 104
17:00-19:00	Welcome Science Caching
19:00 -	Dinner at Urban Kitchen (self-paying) with Cynthia Solomon - Logo Matters: Reflecting on Its 50-year History

MONDAY, SEPTEMBER 23RD

08:30-09:00 — Breakfast — Aula

09:00-9:15 — Official Welcome — Aula

09:15-10:15 — Keynote — Aula

Achieving CSforAll with the Beauty and Joy of Computing
– Dan Garcia

10:30-11:30 — Panel — Aula

What's new in Snap*!* 5
– Michael Ball, Brian Harvey, Jens Mönig, Bernat Romagosa

11:45-12:45

Talks 1 — R121

- **Mathland in the Age of Snap*!* – Reinventing Mathematics Education**
  - Gary Stager
- **Snap*!*'s Role in SAP's Digital Skills Initiative**
  - Nish Pangali, Christiane Bauer

Talks 2 — R122

- **Blocks-based Programming Education in Turkey**
  - Hakan Ataş, Resit Yalin Guckiran
- **Snap*!* in Primary Schools**
  - Jose García Yeste

12:45-14:00 — Lunch

14:00-15:30

Workshops 1 — R121 **Physical Computing Playground** - Bambi Brewer	Workshops 2 — R122 **Liquid Handling Robots** - Mark L. Miller	Workshops 3 — R123 **SnAIp*!* – Explore the magic with us** - Sven Jatzlau, Tilman Michaeli
Workshops 4 — R018 **Coding a Treasure Hunt** - Nathalie Carrié	Workshops 5 — Aula **Horses for Courses – How to compare jigsaw programming languages** - Richard Millwood	Workshops 6 — R021 **The Beauty and Joy of Computing as Middle School Curriculum** - Dan Garcia

15:30-16:30 — Transfer to Evening Reception

16:45-21:00 — Evening Reception at SAP

TUESDAY, SEPTEMBER 24TH

08:30-09:15 — Breakfast — Aula

09:15-10:15 — Keynote — Aula

The Story of Snap*!*
– Brian Harvey, Jens Mönig

10:30-11:30

Talks 3 — R121

- **Snap*!* – a powerful tool to study algorithms in school**
  - Nathalie Carrié
- **Agile School – methods for project-based teaching in computer science and beyond**
  - Ralf Romeike, Peter Brichzin, Petra Kastl

Talks 4 — R122

- **Paradigms of Visual Programming**
  - Sebastian Claus
- **ECDL Coding mit blockbasierten Programmiersprachen**
  - Ronald Bieber
- **Enlightening the Birth of Digital**
  - Uwe Geisler

11:45-12:45

Talks 5 — R121

- **Programming for All – a problem-based approach**
  - Joachim Wedekind
- **Secondary school exams with Snap*!* – an experience report from the past 7 years**
  - Fritz Hasselhorn

Talks 6 — R122

- **Snap*!* Arcade Machine**
  - Bernat Romagosa
- **Hands-on Cybersecurity Curriculum for Schools**
  - Akos Ledeczi
- **Snap*!* as prototyping help (especially for badly documented APIs)**
  - Michael Tobian

12:45-14:00 — Lunch

14:00-15:30

Workshops 7 — R121

Code Crafting – an alternative metaphor for teaching computing
- Ursula Wolz

Workshops 8 — R122

Robotics with Snap*!*
- Josep Ferrandíz, Joan Guillén, Toni Moreno

Workshops 9 — Aula

Object-orientation Concepts in Snap*!*
- Matthias Kim

Workshops 10 — R123

Block-based languages in the classroom – concepts, ideas and examples
- Sven Jatzlau

15:30-17:00 — Coffee Break — Poster Session

17:00-21:00 — Young Thinkers Night

WEDNESDAY, SEPTEMBER 25TH

08:30-09:15 — Breakfast — Aula

09:15-10:15 — Keynote — Aula

Computer Science for All
– Janice Cuny

10:30-11:30

Talks 7 — R121

- **Microblocks**
  - Bernat Romagosa
- **Liquid Handling Robots**
  - Mark L. Miller

Talks 8 — R122

- **Code Crafting – an alternative metaphor for teaching computing**
  - Ursula Wolz
- **Snap4Arduino**
  - Josep Ferrandíz, Joan Guillén
- **Snap4IoT – a Deep Learning and IoT library**
  - Fu-Chiung Cheng

11:45-12:30

Panel — R121

- **A constructionist model of teacher development**
  - Gary Stager, Silvia Martinez, Amy Dugré

Talks 9 — R122

- **Programming word-embeddings in Snap*!***
  - Ken Kahn
- **SnAIp! – explore the magic with us**
  - Sven Jatzlau, Tilman Michaeli

12:30-14:00 — Lunch

14:00-15:30

Workshops 11 — R121

Astronomical Image Processing
- Eckart Modrow

Workshops 12 — R122

Snap4IoT – a Deep Learning and IoT library
- Fu-Chiung Cheng

Workshops 13 — R123

Programming word embeddings in Snap*!*
- Ken Kahn

Workshops 14 — R018

Among Scarabs and Bugs – a live action debugging escape game
- Tilman Michaeli, Ralf Romeike

Workshops 15 — R021

Media Computation with Snap*!*
- Jens Mönig, Jadga Hügle

Workshops 16 — Aula

Fractal Music with Snap*!*
- Markus Gälli

15:30-16:00 — Coffee Break

16:00-16:45	**Talks 10** R121 • **Should Users Write Their Own Control Structures?** - Brian Harvey • **Learning in a digital world – openSAP** - Katrin Elk	**Talks 11** R122 • **BloP – a Snap*!* Meta-Mod to Build Block-based Interactive Environments** - Stefano Federici • **Mandelbrot-Set** - Dan Garcia
16:45-17:15	**Coffee Break**	
17:15-18:00	**Panel** Aula **The Future of Snap*!*** – Michael Ball, Brian Harvey, Jens Mönig, Bernat Romagosa	
19:00-	**Conference Dinner and Closing** Aula	

Monday, September 23rd, 9:15-10:15

Achieving CSforAll with the Beauty and Joy of Computing
- Dan Garcia

At a time when computing is so much a part of all of our lives, has incredible job opportunities, and is so empowering, most students graduate high school without having had any introduction to computer science. A decade ago in the United States, the CSforALL movement was launched to broaden participation in computing to those traditionally underrepresented. This talk will reflect on the current state of that initiative and introduce the "Beauty and Joy of Computing (BJC)" course (that proudly uses Snap*!*), which has received worldwide attention and recently had 65% female enrolment at UC Berkeley, among the highest in the nation.

Dan Garcia is a Teaching Professor at UC Berkeley. Selected as an ACM Distinguished Educator, SAP Visionary Member, and ACM Distinguished Speaker, he is a national leader in the "CSforALL" movement. He is the SIGCSE Board Vice-Chair and was recently honored as the most frequent SIGCSE author in their 50-year history.

Tuesday, September 24th, 9:15-10:15

The story of Snap*!*
- Brian Harvey, Jens Mönig

Snap*!*/BYOB celebrates its 11th birthday this fall. On this occasion, Jens and Brian, Snap*!*'s authors, look back on the very first experimental versions, at ideas that didn't make it and on how Snap*!* became to be, what it is now.

Brian Harvey is the author of the three-volume *Computer Science Logo Style*; the lead developer of Berkeley Logo; a co-developer, with Jens Mönig, of Snap*!*; and a lead developer of *The Beauty and Joy of Computing*, a high school curriculum using Snap*!*.

Jens Mönig is the Snap*!* lead developer and makes interactive programming languages at SAP. Ever since he helped make Scratch Jens has fallen in love with live, blocks-based computing. So much, that he quit his career as a lawyer to become a researcher and designer of such software environments. Jens wants everyone to get a chance to discover the beauty and joy of computing.

Wednesday, September 25th, 9:15-10:15

Computer Science for All
- Janice Cuny

This program aims to provide *all* U.S. students the opportunity to participate in computer science (CS) and computational thinking (CT) education in their schools. With this solicitation, the National Science Foundation (NSF) focuses on fostering the research and development needed to bring CS and CT to all schools. Specifically, this solicitation aims to provide high school teachers with the preparation, professional development (PD) and ongoing support that they need to teach rigorous computer science courses.

Janice Cuny is the Program Director for Computing Education at the National Science Foundation and is responsible for the *Broadening Participation in Computing* and *Computer Science for All* programs.

Monday, September 21st, 10:30 – 11:15

- **What's new in Snap*!* 5**
  – Michael Ball, Brian Harvey, Jens Mönig, Bernat Romagosa

  In this Panel the Snap*!* developers will share their favorite functionality of the new Snap*!* version and discuss new features of Snap*!* 5.

Wednesday, September 25th, 11:45 – 12:30

- **A constructionist model of teacher development**
  - Gary Stager, Silvia Martinez, Amy Dugré

  For twelve years, the Constructing Modern Knowledge institute for educators has represented an uncompromising implementation of constructionism as a means for teacher development. This non-coercive and progressive approach to project-based learning continues to demonstrate the efficacy of constructionism as a means for educators to not only develop exceptional computing and engineering skills, but as a model for what is possible in any classroom. This panel discussion features the founder of Constructing Modern Knowledge, along with two long-time faculty members, including a school educator who used CMK as a vehicle for continuous growth in her school over many years. Join Gary Stager, Sylvia Martinez, and Amy Dugré of the Dusseldorf International School as they share project examples and lessons learned at Constructing Modern Knowledge.

Wednesday, September 25th, 17:15 – 18:00

- **The Future of Snap*!***
  – Michael Ball, Brian Harvey, Jens Mönig, Bernat Romagosa

  Meet with Snap*!* developers Jens Mönig, Brian Harvey, Bernat Romagosa, and Michael Ball for a wide-open discussion of what happens next in Snap*!*.

Talks 1 Monday, September 23rd, 11:45-12:45

- **Mathland in the Age of *Snap!* – Reinventing Mathematics Education**
  - Gary Stager

  Mathland is one of Papert's enduring metaphors for learning-by-computing. This presentation will explore the possibilities for reinventing mathematics education and creating new mathlands using Snap*!* and other modern constructionist computation environments. The case will be made for the urgency of creating what Papert called, a new diet of mathematics for children.

- **Snap*!*'s Role in SAP's Digital Skills Initiative**
  - Nish Pangali, Christiane Bauer

  Come learn about how SAP's outreach programs like the Corporate Social Responsibility or Young Thinkers team are featuring Snap*!* in various digital skill programs around the world. This gives young people an opportunity to learn coding in an easy and fun way. Learn why we think digital skills are so important for this generation.

Talks 2 Monday, September 23rd, 11:45-12:45

- **Blocks-based Programming Education in Turkey**
  - Hakan Ataş, Resit Yalin Guckiran

  Block-based programming and teaching computational thinking are the areas that educators from all over the world draw attention to recently. For the last half decade, the number of studies about these topics increased in Turkey as well. This interest has not shown itself only among educators; all stakeholders such as parents, policy makers and non-profit organization members are involved in the education of new generations. Many students and volunteers from different levels of education and backgrounds were trained in block-based programming and computational thinking thus far. One of these trainings was held in Habitat -a non-profit organization-. This project's aim was to reach 3000 students who live in disadvantaged areas and make them familiar with Scratch. Volunteers from different backgrounds are trained for three days. After curriculum, development and trainings were established with more than 70 volunteers at the end of 3 sessions. A questionnaire was administered to the volunteers to reveal their attitude and perspectives against block-based programming. Opinions of the stakeholders, results of the project, implementation process and curriculum content will be shared in the conference. These findings will help to raise awareness of the teachers, school administrators, policy makers and parents in terms of programs in place.

- **Snap*!* in Primary Schools**
  - Jose García Yeste

  Citilab is a citizen laboratory located in Conellà, a town near Barcelona, which since its creation in 2007, has promoted the use of programming for everyone. It created S4A Scratch for Arduino, and the first versions of Snap4Arduino and BeetleBlocks. In recent years we have concentrated our efforts on teacher training and in the earliest stages of education, with the Edulab project. In this project didactic activities are created with the support of programming and robotics, with the help of teachers, using various tools, including Snap*!*. We want to present some of these experiences where Snap*!* helped with his very useful features. As a result of the Edulab project, we will also present SnapJr*!*, which allows students from 5 to 8 years to work with programming by blocks and robotics.

Talks 3 Tuesday, September 24th, 10:30 – 11:30

- **Snap*!* – a powerful tool to study algorithms in school**
  - Nathalie Carrié

  Years of mathematics education in high school led me to approach the concepts of curriculum in terms of algorithms and to bring out the essential idea of a function in mathematics.
  Snap*!* allowed me to instill Computational Thinking in my students: analyze a problem, break it down into many small functions with very specific and very small tasks.
  Snap*!* led students to think in terms of algorithms purged of input-output problems, to finally reason in terms of functions. In fact, any algorithm can be seen as a function: input, box, result. We enter objects of all kinds, the box processes these objects and returns a result. This representation forces students to think mathematics in terms of functions. The regular use of Snap*!* in class allowed each concept to be approached as an algorithm and then to show students to what extent everything is a function in the mathematics curriculum taught in high schools.

- **Agile School – methods for project-based teaching in computer science and beyond**
  - Ralf Romeike, Peter Brichzin, Petra Kastl

  In start-ups today almost nothing works without Agile software development. At school, too, project teaching becomes more profitable thanks to agile methods and can lead pupils to self-organization. The lecture describes the background and philosophy of agile methods from a practical school perspective, illustrates these using concrete practical examples (with and without block languages) and provides implementation tips in the form of a tried and tested set of methods. A look at the corresponding research and experiences shows that pupils in agile projects also successfully apply self-organization competences acquired in other subjects.

Talks 4 Tuesday, September 24th, 10:30 – 11:30

- **Paradigms of Visual Programming**
  - Sebastian Claus

  The Berlin curriculum demands schools to introduce object-oriented programming. Advanced courses shall even introduce a declarative paradigm (logical or functional). In computer science didactics there is a trend to familiarize students with programming using visual programming languages (VPL). The question arises whether both can also be combined? Therefore, we try to answer the question: Which programming paradigms are realized by the respective VPL? We base Van Roy's and Haridi's broad interpretation of programming as "a general human activity, [...] extending or changing a system's functionality." This also permits the investigation of many interesting, not Turing complete visual domain-specific languages. Overall, Van Roy's programming concepts and their relation to the programming paradigms provide the theoretical framework for this work. First, we created a pool of VPL (N=140) from popular sources (Wikipedia, a blog post, Curly Directory Listing). These VPL were classified according to their program representation form (block-, flowchart-, form-, grid-, graph-, petri-net-, plex-based and programming-by-example). These classes were sorted by relevance, based on a combination of Google hits and Google Scholar hits. Finally, the eight first-placed VPL were examined according to their implemented programming concepts, thus drawing conclusions about the paradigms. Three new paradigms unique to VPL were identified.

- **ECDL Coding mit blockbasierten Programmiersprachen**
  - Ronald Bieber

- **Enlightening the Birth of Digital**
  - Uwe Geisler

  The word 'digital' became omnipresent today as is digital technology. But knowledge and understanding of core concepts of computing, such as digital, have not grown at the same pace. The didactical light art installations invented by Uwe Geisler have the potential to change this. Day-to-day products (Sensor-Night-Lights) are transformed into working digital technology (NOR-Gates). This interactive fun-approach gives users aged 8 thru 99 a deeper understanding of the world around them.
  That is true digital enlightenment.

Talks 5 Tuesday, September 24th, 11:45-12:45

- **Programming for All – a problem-based approach**
  - Joachim Wedekind

  If the motto „Programming for All!" is taken seriously, it is not only children and adolescents who should become digitally competent. Especially adults whose school days did not include computers and the internet should acquire these competences, not only in order to understand their children's school material, but also in order to be able themselves to act competently even in a digitally influenced society. These addressees are usually not very interested in the typical (mathematically and informatically influenced) introductory examples of school teaching. For these addressees, I present a curriculum that covers important basic concepts of informatics and programming - orientated towards working on motivating problems. Some examples will be presented in more detail: Coded Art, Animated Optical Illusions, and Gothic Architecture. The transfer to other motivating topics will be addressed and discussed.

- **Secondary school exams with Snap*!* – an experience report from the past 7 years**
  - Fritz Hasselhorn

  Starting 2013 as the first school in Germany, we continuously use BYOB/Snap*!* right up to the A levels. In this presentation the key points of our concept beginning with monoeducative courses at the entry level, spiral curriculum, teaching computer science with lists as fundamental data structure are examined. Experiences of the past 7 years with that curriculum will be shared.

Talks 6 Tuesday, September 24th, 11:45-12:45

- **Snap*!* Arcade Machine**
  - Bernat Romagosa

  We have designed and built a Snap*!*-powered full-size arcade cabinet. Design your own arcade classics and get to play them as they are meant to: standing up and shoulder to shoulder with your friends!

- **Hands-on Cybersecurity Curriculum for Schools**
  - Akos Ledeczi

  The talk will present RoboScape, a collaborative, networked educational robotics environment. RoboScape is built on top of NetsBlox, an open-source, networked, visual programming environment based on Snap*!* that is specifically designed to introduce students to distributed computation and computer networking. With RoboScape, a user's program controlling the robot runs in the browser and not on the robot. This makes programming a robot just as easy as programming a sprite. Furthermore, the wireless communication between a student's program and the robot can be overheard by the programs of the other students. This makes cybersecurity an immediate need that students realize and can work to address. We have designed and delivered a cybersecurity summer camp to 70 students in grades between 7 and 12 as well as a week-long professional development workshop to 12 teachers. The talk will summarize the technology behind RoboScape, the hands-on curriculum of the camp and the lessons learned.

- **Snap*!* as prototyping help (especially for badly documented APIs)**
  - Michael Tobian

  The web holds tons of information, some are consumable via API, but service providers are lacking a good documentation. The demo syncs two APIs (openweathermap.org and telegram messenger) as an example, how easy you can re-engineer the structure of a JSON, no matter how many nests are in a nest.

Talks 7 Wednesday, September 25th, 10:30-11:30

- **Microblocks**
  - Bernat Romagosa

  MicroBlocks is a new programming language inspired by *Scratch* that runs right inside microcontroller boards such as the *micro:bit*, the *NodeMCU* and many *Arduino* boards
  The MicroBlocks system allows for dynamic, parallel and interactive programming, just like in Scratch, but with the twist of letting your projects run autonomously inside the board without being tethered to a computer. Thus, MicroBlocks provides the immediacy and liveness of tethered blocks programming, while supporting real-world applications that require precision timing, autonomous operation, or physically embedding the processor into projects.

- **Liquid Handling Robots**
  - Mark L. Miller

  Liquid handling robots, based on Snap*!* as the vehicle for student programming, have been under development over the last three years by a team of contributors from Learningtech.org and Stanford University. These inexpensive, prototype robots are capable of pipetting liquids, to perform experiments, under student program control. They support sixteen cuvettes and a 96 well plate. Students learn the basics of Snap*!* programming, augmented by nine domain-specific Snap*!* blocks, such as a block to dispense multiples of 0.1 mL of fluid. An Arduino Mega controlled by Snap4Arduino controls 4 motors and sensors. The robots are made using laser cut plastic from downloadable designs. The LHR project is an exemplar for an important trend in K-12 education in the United States: augmenting traditional science education through Computer Science (and conversely). Informal data from pilot studies suggests that students tend to become more engaged, and possibly develop a deeper understanding, both for the CS and for the underlying science topics. Future work will examine how to make the robots more accurate, more robust, and easier for students to assemble themselves, while we continue enriching the curriculum materials and conducting studies with additional schools.

Talks 8 Wednesday, September 25th, 10:30-11:30

- **Code Crafting – an alternative metaphor for teaching computing**
  - Ursula Wolz

  Despite many efforts, the dominant coding culture continues to focus on activities viewed as white male-centric, including games that involve collision, financial management, and sports simulation. Diversity initiatives tend to focus on how women and people from other cultures are invited into this culture. Code crafting is different, asserting that coding and computing originated in womens' craft, using the collaborative agenda of the American quilting bee. Through an adaptation of Snap*!* called 'TurtleStitch' we provide a radically different agenda and environment in which textile creativity provides the natural conceptual metaphors for coding. TurtleStitch supports the design of embroidery patterns ranging from simple geometric figures to complex graph traversal solutions. QuiltBlocks (still under development) supports the design of a collaborative quilt from embroidery patches. TurtleStitch is also used to create the quilting tessellations on the assembled blanket. The curriculum also ties in hand crocheting as an off computer coding experience and Arduino-based control of a punch card-based knitting machine. Taken as a whole, from crochet hook to JavaScript file handling for the knitting, the approach provides sufficient resources for activities ranging from an 'hour of code' to two semesters of foundations in computer science.

- **Snap4Arduino**
  - Josep Ferrandíz, Joan Guillén

  Snap4Arduino as a Snap*!* desktop edition ready to connect our projects with our physical world and with other computational devices (smartphones, computers... and other Snap*!* projects!!)
  We will introduce Snap4Arduino (team, goals, features...) and present some examples to show its possibilities.

- **Snap4IoT – a Deep Learning and IoT library**
  - Fu-Chiung Cheng

  Artificial Intelligence (AI) and Internet-of-Things (IoT) are two of the most important innovations in 21st century. The combination of AI and IoT, namely "AIoT", has opened the possibility of various applications for the future. Many national leaders attach great importance to this trend and start teaching programming of AIoT for K12 students as early as possible. To help students understand computational thinking and programming, many visual programming tools such as Scratch, Snap*!* and Blockly, are developed to lower the learning curve. We develop an AIoT library for Snap*!*, called Snap!4AIoT. Together with other support tools (ESPlorer for IoT and Colab, Jupyter lab for AI), Snap!4AIoT extends Snap! to AI and/or IoT applications. For IoT, we develop IoT development boards and the code compiled by Snap!4AIoT can be downloaded to the boards directly and tested, which makes IoT teaching easier than ever before. We have successfully applied Snap!4AIoT to developing industrial IoT applications, including Taiwan's wireless charging system and Taipei city's smart health monitoring system. For AI, we support Keras, a high-level neural network (NN) framework and Python. Dense, convolutional, recurrent and hybrid NNs can be easily built, trained and tested. Snap!4AIoT simplifies AIoT development significantly and turns learning into fun.

Talks 9 Wednesday, September 25th, 11:45 – 12:45

- **Programming word-embeddings in Snap*!***
  - Ken Kahn

  Word embeddings is a technique in natural language processing whereby words are embedded in a high-dimensional space. They are used in sentiment analysis, entity detection, recommender systems, paraphrasing, text summarization, question answering, translation, and historical and geographic linguistics. We describe a Snap*!* Library that contains 20,000-word embeddings in 15 languages. Using a block that reports a list of 300 numbers for any of the known words, one can create programs that search for similar words, find words that are the average of other words, explore cultural biases, and solve word analogy problems. These programs can work in a single language or rely upon the alignment of the word embedding spaces of different languages to perform rough translations. To compute with word embeddings one needs perform vector arithmetic. This can be accomplished by providing vector arithmetic blocks. More advanced users can instead take advantage of Snap*!*'s support of higher-order functions to use list mapping blocks to perform the vector operations.

- **SnAIp! – explore the magic with us**
  - Sven Jatzlau, Tilman Michaeli

  AI and Machine Learning are the hot topics this decade, not only in industry, but also in computer science education, as they contain many exciting and interesting new concepts. However, many educators think these topics are "magical", too complex, and that they can't be taught in classrooms. But Machine Learning does not have to be crazy magic! Using the Snap*!*-environment, which is perfect for creating games with, we can use Machine Learning to teach an AI how to beat us at our favorite games. We use Reinforcement Learning implemented entirely in Snap*!* itself, without relying on external libraries, JavaScript-wrappers, or servers. Because of this, the learning algorithm is fast, intuitive, simple, and can easily be visualized for easier understanding. It also means that our approach can be transferred to many other game settings, simulations, problems, etc. In our workshop, we show several simple examples that can be taught in classroom settings or teacher workshops. The participants will get the chance to implement these examples and experiment with different configurations. Explore the magic with us!

Talks 10 Wednesday, September 25th, 16:00 – 16:45

- **Should Users Write Their Own Control Structures?**
  - Brian Harvey

  Since BYOB 3.0, one of the things about which we're most proud is that users can write blocks such as FOR, FOR EACH, MAP, and KEEP entirely in Snap*!*. Learning to do that is the main dividing line between beginners and experts. On the other hand, for several reasons, both pedagogic and technical, it's really best if those tools are available to beginners right away. In the past, we've used an awkward compromise: the "tools" library of blocks that are written in Snap*!* but are considered quasi-primitive. In Snap*!* 5.0, these blocks are true primitives, partly because of the speed demands of "big-ish data" projects. But this isn't the end of the story; we are considering various ideas to allow us to have our cake and eat it too.

- **Learning in a digital world – openSAP**
  - Katrin Elk

  openSAP provides free Massive Open Online Courses (MOOCs) to everyone interested in learning about SAP's latest innovations and how to survive in the digital economy. SAP is committed to supporting the 17 United Nations Sustainable Development Goals and as part of our contribution to goal 4, Quality Education, openSAP also delivers courses by thought leaders on digital transformation as well as courses by SAP's partners in the area of Corporate Social Responsibility. To support digital literacy, openSAP partners with a variety of organizations worldwide to deliver online access to training with programs, including Snap*!*

Talks 11 Wednesday, September 25th, 16:00 – 16:45

- **BloP – a Snap*!* Meta-Mod to Build Block-based Interactive Environments**
  - Stefano Federici

  Snap*!* has anticipated Scratch's move to JavaScript by providing a completely new, clean and well-organized code structure that allowed many developers to create their own mods. Contrary to Scratch mods -usually just sets of additional blocks-, Snap*!* mods are very often new environments that allow people to create real 3D objects (Beetleblocks), design patterns for embroidery machines (TurtleStitch), manipulate graphs via algorithms (Edgy). Building such environments requires a good knowledge of both JavaScript and Snap*!*'s code. But new environments can be created in Snap*!* even just using its current features. The only drawback is that users can impare the environment by using the standard features of Snap*!*'s UI. A mod, called BloP, has been built so that -once blocks and sprites of the new language have been defined - elements of the Snap's UI can be hidden and the environment can be locked so to make it safe for newbie users. By using BloP, new environments can be built to learn complex programming languages such as C (blockC) and HTML/PHP/MySQL (blockHtmlPhpMySql), simple languages such as Logo (blockLogo), but also search and sorting algoritms (ASSL), simple English sentences (blockLang) or to build simple stories (storyBlock).

- **Mandelbrot-Set**
  - Dan Garcia

Evening Talk

Sunday, September 22nd, 17:30, Transferzentrum Heidelberg

Logo Matters: Reflecting on Its 50-year History
- Cynthia Solomon

Cynthia Solomon created Logo, the first programming language for children, along with Danny Bobrow, Wally Feurzeig, and Seymour Papert at Bolt, Beranek and Newman in 1966. She and Papert continued Logo research at the MIT Artificial Intelligence Lab, where the Logo environment was extended to music, turtle graphics, and robotics with the collaboration of Marvin Minsky and other Lab members. Her seminal book *Computer Environments for Children* was the first comprehensive reflection on computers in education, and her paper with Seymour Papert, "Twenty Things to do with a Computer," is a classic in the field. Her new book, *Inventive Minds,* is a major contribution to thinking about computational thinking and children. She received an MA in Computer Science from Boston University (1976) and an EdD from Harvard University (1985). She serves on the program committee of Constructing Modern Knowledge and in 2016 was awarded both the National Center for Women & Information Technology Pioneer Award and the Constructionism Lifetime Achievement Award. She also received the 2019 FabLearn Lifetime Achievement Award.

Wednesday, September 25th, 19:00

Constructionism and the Limits of Instruction
-Gary Stager

This presentation builds upon the research legacy of Papert, Harel, Kafai, Cavallo, and Blikstein to hypothesize that not only should projects be an educator's smallest unit of concern, but a good project may replace instruction entirely. The speaker will propose a research agenda that explores non-coercive forms of schooling and asserts the role of constructionism as the primary means for knowledge construction.

Workshops 1 Monday, September 23rd, 14:00-15:30

Physical Computing Playground
- Bambi Brewer

Physical computing inspires and delights students learning computer science by providing them a tangible representation of their code. In this workshop, participants will use Snap*!* to explore three physical computing devices: the micro:bit, the Hummingbird Robotics Kit, and the Finch Robot. The micro:bit is a popular, low-cost way to introduce students to physical computing, while the Hummingbird and Finch are based on the micro:bit but extend its capabilities. The Hummingbird is comprised of lights, sensors, and motors which allow students to build a robot out of any materials, and the Finch is a robot that can move, light up, and respond to sensors. We have extended Snap*!* to enable students to write programs that use the lights, motors, and sensors of these robots. This broadens the low-floor, high-ceiling approach of Snap*!* to encompass the domain of physical computing. Absolute beginners can get started in minutes but continue learning more advanced computer science and engineering for years. Participants in this workshop will have the opportunity to explore all three robots at the level of complexity that is right for them!

Workshops 2 Monday, September 23rd, 14:00-15:30

Liquid Handling Robots
- Mark L. Miller

Liquid handling robots, based on Snap*!* as the vehicle for student programming, have been under development over the last three years by a team of contributors from Learningtech.org and Stanford University. These inexpensive, prototype robots are capable of pipetting liquids, to perform experiments, under student program control. They support sixteen cuvettes and a 96 well plate. Students learn the basics of Snap*!* programming, augmented by nine domain-specific Snap*!* blocks, such as a block to dispense multiples of 0.1 mL of fluid. An Arduino Mega controlled by Snap4Arduino controls 4 motors and sensors. The robots are made using laser cut plastic from downloadable designs. The LHR project is an exemplar for an important trend in K-12 education in the United States: augmenting traditional science education through Computer Science (and conversely). Informal data from pilot studies suggests that students tend to become more engaged, and possibly develop a deeper understanding, both for the CS and for the underlying science topics. Future work will examine how to make the robots more accurate, more robust, and easier for students to assemble themselves, while we continue enriching the curriculum materials and conducting studies with additional schools.

Workshops 3 Monday, September 23rd, 14:00-15:30

SnAIp! – Explore the magic with us
- Sven Jatzlau, Tilman Michaeli

AI and Machine Learning are the hot topics this decade, not only in industry, but also in computer science education, as they contain many exciting and interesting new concepts. However, many educators think these topics are “magical”, too complex, and that they can’t be taught in classrooms. But Machine Learning does not have to be crazy magic! Using the Snap*!*-environment, which is perfect for creating games with, we can use Machine Learning to teach an AI how to beat us at our favorite games. We use Reinforcement Learning implemented entirely in Snap*!* itself, without relying on external libraries, JavaScript-wrappers, or servers. Because of this, the learning algorithm is fast, intuitive, simple, and can easily be visualized for easier understanding. It also means that our approach can be transferred to many other game settings, simulations, problems, etc. In our workshop, we show several simple examples that can be taught in classroom settings or teacher workshops. The participants will get the chance to implement these examples and experiment with different configurations. Explore the magic with us!

Workshops 4 Monday, September 23rd, 14:00-15:30

Coding a Treasure Hunt
- Nathalie Carrié

In Mathematics, kids love puzzle games. They invented the following game: "the treasure hunt". A pupil known as "the pirate", hides a treasure in the rectangular courtyard of the school. Then, the pirate gives to Oblazo, the player, a special treasure map. This map indicates how Oblazo's distance to the treasure changes when he walks around the courtyard. He is only allowed to walk on the courtyard's edge. Oblazo must use this map and his thinking abilities to find the treasure. In Snap*!* the courtyard is represented by the stage. The inner courtyard is a rectangle at x.px on the left side of the stage, and y.px on the top of the stage. The treasure has been hidden in the inner courtyard and Oblazo has to find it by walking along the edge of the inner courtyard. In this workshop, I will help you to simulate this treasure hunt. You will be able to play the game and guess where the treasure is. Finally, you'll go further with the math behind this treasure hunt: you'll have to compute the distance between Oblazo and the treasure and show its graphic representation on the stage, when Oblazo moves along the schoolyard.

Workshops 5 Monday, September 23rd, 14:00-15:30

Horses for Courses – How to compare jigsaw programming languages
- Richard Millwood

This conversation will have started at the Scratch Conference in Cambridge, August 2019 in a half-hour discussion about different blocks-based programming languages and their capabilities, advantages and limits. This workshop is intended to extend and consolidate that initial discussion and be the basis for a publishable outcome.

Workshops 6 Monday, September 23rd, 14:00-15:30

The Beauty and Joy of Computing as Middle School Curriculum
- Dan Garcia

The Beauty and Joy of Computing is an introductory computer science curriculum developed at the University of California, Berkeley, intended for non-CS majors at the high school junior through undergraduate freshman level. BJC middle school is an attempt to slow that material down, and provide more interesting examples for younger students.

Workshops 7 Tuesday, September 24th, 14:00-15:30

Code Crafting – an alternative metaphor for teaching computing
- Ursula Wolz

Despite many efforts, the dominant coding culture continues to focus on activities viewed as white male-centric, including games that involve collision, financial management, and sports simulation. Diversity initiatives tend to focus on how women and people from other cultures are invited into this culture. Code crafting is different, asserting that coding and computing originated in women's' craft, using the collaborative agenda of the American quilting bee. Through an adaptation of Snap*!* called 'TurtleStitch' we provide a radically different agenda and environment in which textile creativity provides the natural conceptual metaphors for coding. TurtleStitch supports the design of embroidery patterns ranging from simple geometric figures to complex graph traversal solutions. QuiltBlocks (still under development) supports the design of a collaborative quilt from embroidery patches. TurtleStitch is also used to create the quilting tessellations on the assembled blanket. The curriculum also ties in hand crocheting as an off-computer coding experience and Arduino-based control of a punch card-based knitting machine. Taken as a whole, from crochet hook to JavaScript file handling for the knitting, the approach provides sufficient resources for activities ranging from an 'hour of code' to two semesters of foundations in computer science.

Workshops 8 Tuesday, September 24th, 14:00-15:30

Robotics with Snap
- Josep Ferrandíz, Joan Guillén, Toni Moreno

Snap*!* connection with our physical world and the role of abstraction in Robotics. Starting with a global vision of the different ways to connect Snap*!* with our physical world, we will explore the possibilities of Snap4Arduino and the creation of our educational starting points (libraries, project templates...).
We will build a real project using a sensor shield over Arduino and connecting it to the virtual world inside a Snap*!* project.

Workshops 9 — Tuesday, September 24th, 14:00-15:30

Object-orientation Concepts in Snap*!*
- Matthias Kim

Scratch is a perfect tool for teaching kids the first ideas of programming (under 12 years). But Scratch is not the right tool to teach real CS (Computer Science) and OO (Object oriented programming). OO is usually taught with languages like C++, Java, Python, … But the barriers and walls to jump from Scratch to a real OO programming language are high: 1. There is Typing including characters like ";" "{", which stops kids from having fun 2. OO Concepts are very abstract and difficult at first sight. I want to discuss how we utilize SNAP as the perfect tool for undergraduate CS Education, esp. OO. We value SNAP as the missing link between Scratch and a real text language. My presentation will cover how we use SNAP to teach OO and what OO games we us as examples for OO concepts. Types, Parameter/Type of Parameter Global /Local Scope Class/Object, sprite as class/clone as object, constructor, Instantiation, Methods/Attributes of a class, Private/Public in SNAP Methods cross Sprites. Focus will not be on subjects as such, but on the logic of the teaching sequence.

Workshops 10 — Tuesday, September 24th, 14:00-15:30

Block-based languages in the classroom – concepts, ideas and examples
- Sven Jatzlau

Visual block-based programming languages make it easier for novices to learn how to program. Studies show that intuitivity and contextualized examples make them well-suited for use in school, lead to more profound understanding, and ultimately make teaching easier. Block-based programming has become indispensable even in secondary education, being used successfully from the beginning until the end. The workshop is aimed both at beginners and experienced teachers of visual programming languages, shows motivating examples and explains new didactic concepts for programming with visual languages. Furthermore, we will outline sample tasks that are ready to be used in classrooms and discuss further applications.

Workshops 11 — Wednesday, September 25th, 14:00-15:30

Astronomical Image Processing
- Eckart Modrow

For some years now we use graphical programming languages in courses for up-coming physics-teachers in order to let the students construct the necessary methods for themselves. Combined with usage of robotic telescopes, e.g. in the "Monitoring Network of Telescopes: MONET", this results in lively and very motivating teaching. The programming language must provide commands which makes it possible to create algorithmically simple operations to perform the desired functionality - very similar to computer graphics, robot control or database access. For astronomical ap-plications it must be ensured that the actual data, the "image", remains separated from the current representation of the data, e.g. by false colors, and that the operations are sufficiently fast. Both can be easily implemented in the current version of Snap*!* Currently, a library has been developed to display data on a sprite's costume as graph, diagram, image, histogram, table, ... as well as operations on these data such as affine transformations, groupings, search and sort, convolution with kernel, ... With these operations machine learning methods or typical image processing methods of astrophysics can be realized. This will be explained and tested in the workshop using a few examples.

Workshops 12 — Wednesday, September 25th, 14:00-15:30

Snap4IoT – a Deep Learning and IoT library
- Fu-Chiung Cheng

Artificial Intelligence (AI) and Internet-of-Things (IoT) are two of the most important innovations in 21st century. The combination of AI and IoT, namely "AIoT", has opened the possibility of various applications for the future. Many national leaders attach great importance to this trend and start teaching programming of AIoT for K12 students as early as possible. To help students understand computational thinking and programming, many visual programming tools such as Scratch, Snap*!* and Blockly, are developed to lower the learning curve. We develop an AIoT library for Snap!, called Snap!4AIoT. Together with other support tools (ESPlorer for IoT and Colab, Jupyter lab for AI), Snap!4AIoT extends Snap! to AI and/or IoT applications. For IoT, we develop IoT development boards and the code compiled by Snap!4AIoT can be downloaded to the boards directly and tested, which makes IoT teaching easier than ever before. We have successfully applied Snap!4AIoT to developing industrial IoT applications, including Taiwan's wireless charging system and Taipei city's smart health monitoring system. For AI, we support Keras, a high-level neural network (NN) framework and Python. Dense, convolutional, recurrent and hybrid NNs can be easily built, trained and tested. Snap!4AIoT simplifies AIoT development significantly and turns learning into fun.

Workshops 13 Wednesday, September 25th, 14:00-15:30

Programming word embeddings in Snap*!*
- Ken Kahn

Word embeddings is a technique in natural language processing whereby words are embedded in a high-dimensional space. They are used in sentiment analysis, entity detection, recommender systems, paraphrasing, text summarization, question answering, translation, and historical and geographic linguistics. We describe a Snap*!* Library that contains 20,000 word embeddings in 15 languages. Using a block that reports a list of 300 numbers for any of the known words, one can create programs that search for similar words, find words that are the average of other words, explore cultural biases, and solve word analogy problems. These programs can work in a single language or rely upon the alignment of the word embedding spaces of different languages to perform rough translations. To compute with word embeddings, one needs perform vector arithmetic. This can be accomplished by providing vector arithmetic blocks. More advanced users can instead take advantage of Snap*!*'s support of higher-order functions to use list mapping blocks to perform the vector operations.

Workshops 14 Wednesday, September 25th, 14:00-15:30

Among Scarabs and Bugs – a live action debugging escape game
- Tilman Michaeli, Ralf Romeike

Prof. Heinrich W. Müller, an archaeological computer scientist, discovered a new burial chamber during excavations in the Valley of the Kings. Rumor has it he took away a particularly impressive object for his private collection...Since then there have been many strange deaths in Cairo, especially among expedition members and workers who helped with the excavations... It's up to you to end the Pharaoh's curse and recover the stolen object! Join our workshop to learn about the connection between debugging skills and live action escape room puzzles. Systemically checking programs for errors, finding and fixing them is an essential competence for everyone who gets involved into programming, but can also be successfully applied in our daily lives... or an Escape Room. Can you find bugs (and scarabs ;)) and solve all debugging riddles?

Workshops 15 Wednesday, September 25th, 14:00-15:30

Media Computation with Snap*!*
- Jens Mönig, Jadga Hügle

Oh, how we love the graphic effects in Scratch and Photoshop! But can you explain how they work? And if you do can you invent your own? In this workshop you'll learn both, and also find out how to apply the same techniques to sounds. For fun and science! We will explore the pixels and audio blocks of Snap*!* We'll have fun creating our own graphic and sound effects and applying them to webcam pics, live videos and sound recordings of ourselves.
Ever wanted to turn your own voice into a musical instrument? Ever wished for your sunset pics to be more awesome than your holidays? Ever wondered how to engage your students with "Big Data" and "Higher Order Functions"? Join us for the bright side of Computer Science.

Workshops 16 Wednesday, September 24th, 14:00-15:30

- **Fractal Music with Snap*!***
  - Markus Gälli

Porting Musinum to Snap*!*