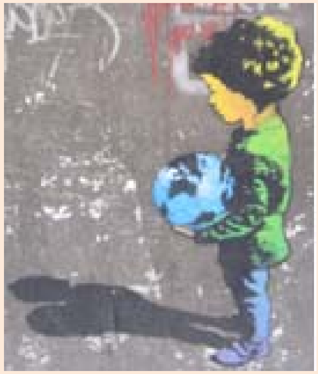
Connecting Worlds Barcelona

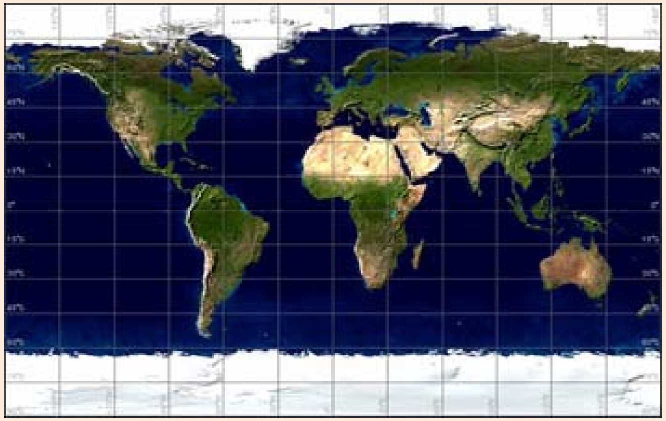


By Steve Holmes

Connecting Worlds – In More Than One Way

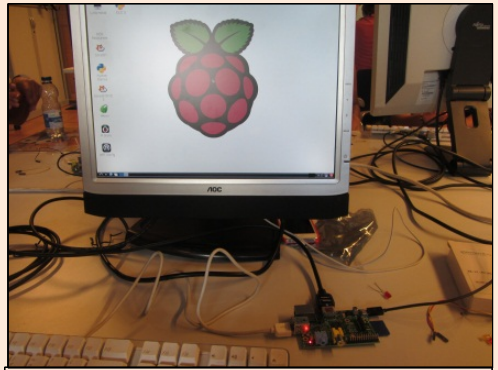
The conference theme "Connecting Worlds" has many meanings.

This was a worldwide event with attendees coming from North America, Europe, Norway, Mexico, Serbia, Tunisia and other places.



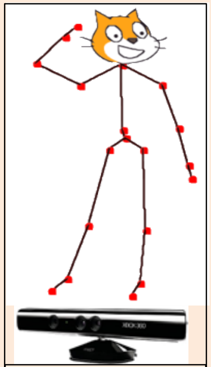
Connecting Worlds – Hardware to Scratch

There was an emphasis of connecting Scratch to the outside world via devices such as:

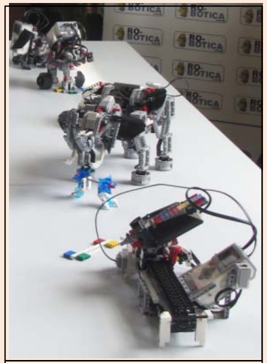


Raspberry PI

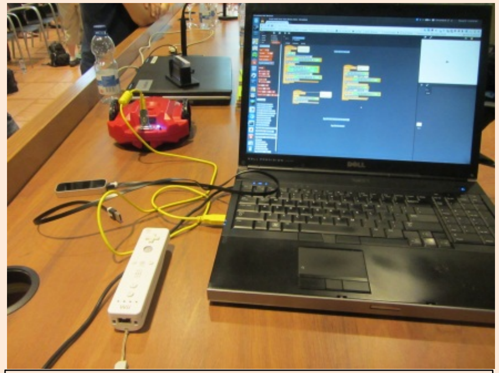
Leap Motion Ctlr



Kinect

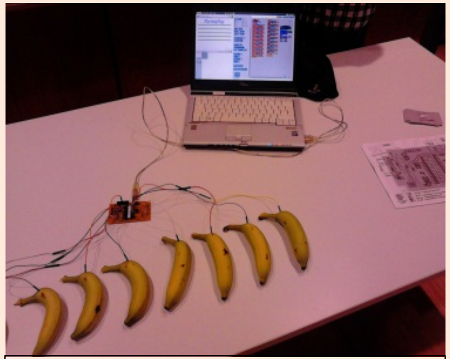


Robots



Wii / Lego NXT

Arduino



Bananas

Connecting Worlds – Different Interests

The conference was attended by people of diverse interests:

Educators

Computer Scientists

Artists

Musicians

Technologists

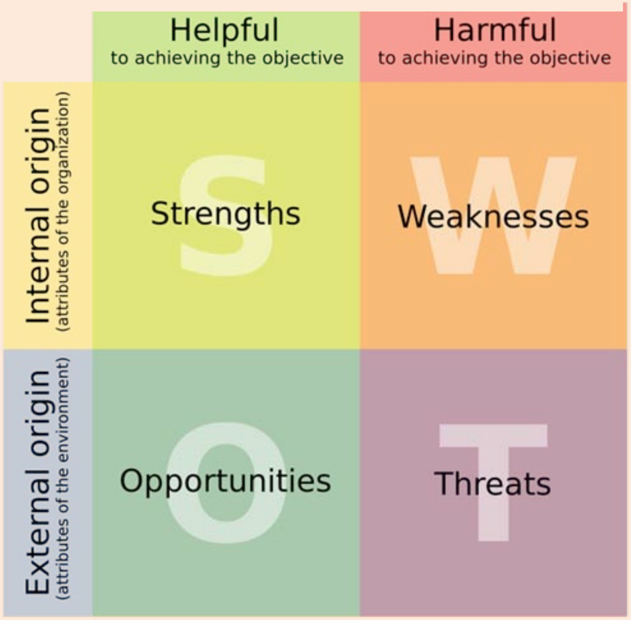
Community Activists

Mathematicians

Keynote Day 1: Scratch

Mitch Resnick and Karen Brennan of MIT Media Labs

- They presented the current state of Scratch in SWOT analysis format (Strengths, Weaknesses, Opportunities, Threats) :



	Helpful to achieving the objective	Harmful to achieving the objective
Internal origin (attributes of the organization)	Strengths	Weaknesses
External origin (attributes of the environment)	Opportunities	Threats

Keynote Day 1: Scratch

Mitch Resnick and Karen Brennan of MIT Media Labs

Scratch Strengths:

- Not just a programming language but a beginning to learning about digital creativity.
- Sharing and Community – Over 3 million projects uploaded from around the world.
- Diversity of learning entry paths. Not only “low floor – high ceiling” but also “wide walls”.

Scratch Weaknesses:

- Such diversity of use is hard to support.
- Standalone Scratch 2.0 out by summer end 2013. Tablet version coming available.
- Device driver interface delayed in 2.0 due to web service architecture challenges.

Keynote Day 1: Scratch

Mitch Resnick and Karen Brennan of MIT Media Labs

Scratch Opportunities:

- The importance of learning to code is increasing i.e. code.org.
- Schools are starting to adopt IT curriculum.
- Vibrant Scratch community has many diverse contributors.
- "Computational Creation" is total learning, not just computers.

Scratch Threats:

- "Workforce" is not the only path. People learn to write and use this skill in their lives, but not all become professional writers.
- Placing learning how to use the tool before the learning process itself. Known as "techno centrism". - Seymour Papert
- Controlling the computer is not the end goal. The learning process and transferring of thinking skills to other domains is most important.

Keynote Day 1: Q&A

Mitch Resnick and Karen Brennan of MIT Media Labs

Q. How does Scratch keep current?

A. Learn from kids. Listen to the community.

Q. What is Scratch doing to embrace the internet?

- Implementing chat rooms. This is proceeding slowly and cautiously to protect kids.
- Scaling up the data center. Cloud support is going to be increased.
- Exploring links to Facebook
- Integrating with Google Maps to explore geography: http://test.scratch.mit.edu/

Keynote Day 1: Q&A

Mitch Resnick and Karen Brennan of MIT Media Labs

Q. What comes after Scratch?

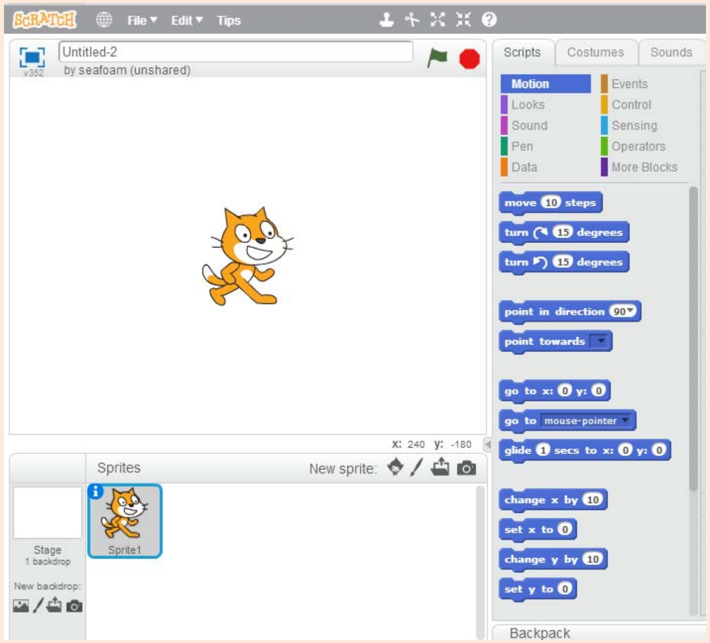
A. For advanced computer science features go to "Snap!".

Closing Remarks

- Try, fail, try again, fail again… Be comfortable being uncomfortable. Do not harshly judge failure.
- This is more than a Scratch conference. There is a broader set of ideas operating here:
  - Goals for supporting learning and determining the future of learning.
  - Preparing young people for a rapidly changing world. This requires creative thinking.
  - "Learner Agency" – pursuing a learning path of interest which resonates with the individual.

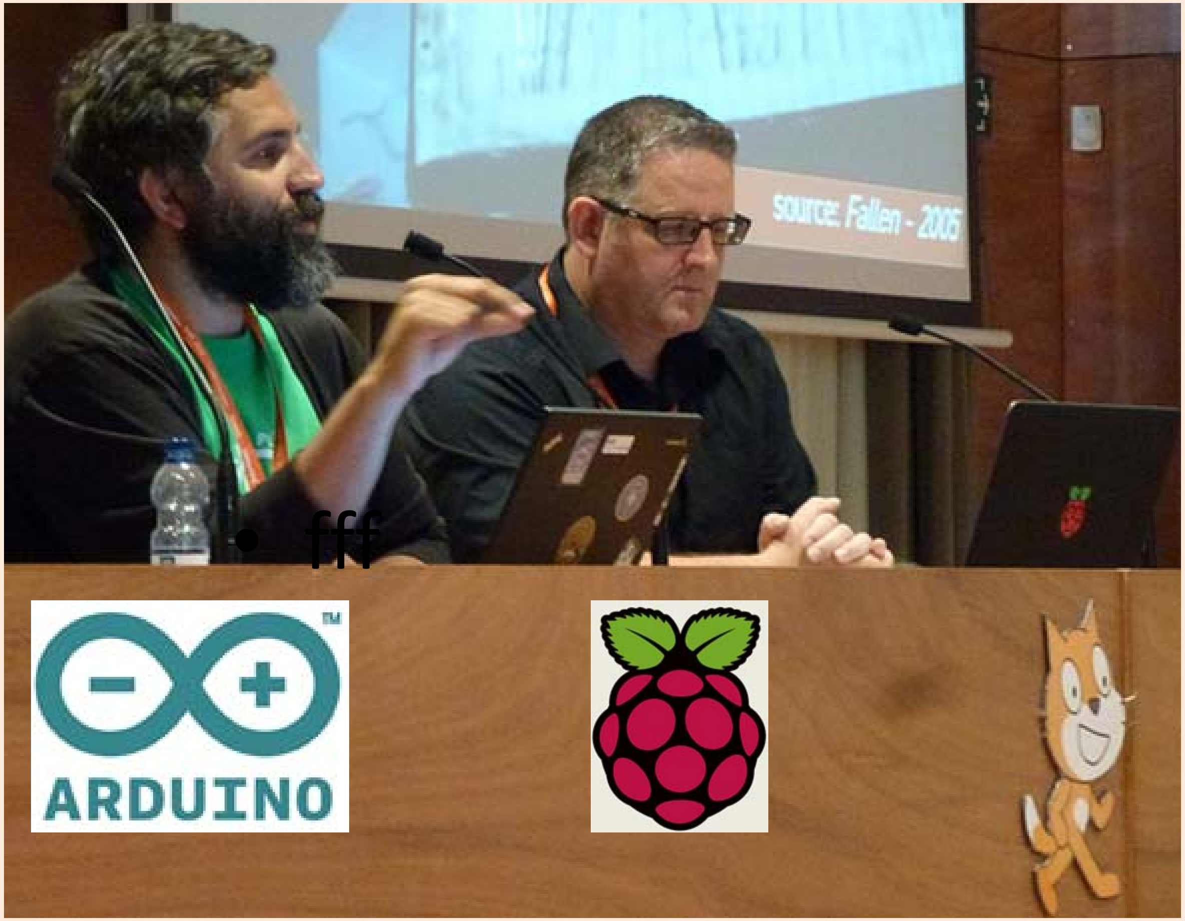
What’s New in Scratch 2.0

- Runs in the browser
- Improved (vector) graphics
- Sprite cloning
- Cloud variables
- Custom block creation
- Video motion detection
- Sharing made much easier:
  - – Easy viewing of source code for any project
  - – Backpack for holding objects from other projects
  - – Easy copying of any existing project



Keynote Day 2: Arduino & Raspberry Pi

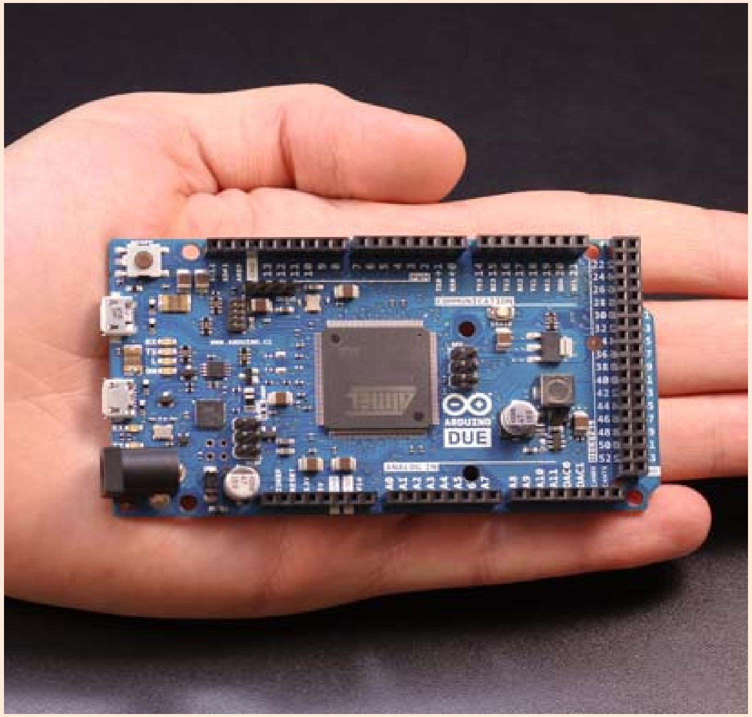
David Cuartielles of [Arduino]() (left) and Clive Beale of [Rasberry PI]() (right)



- fff

David Cuartielles: Arduino

Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists and anyone interested in creating interactive objects or environments.



ARDUINO

David Cuartielles Key Points:

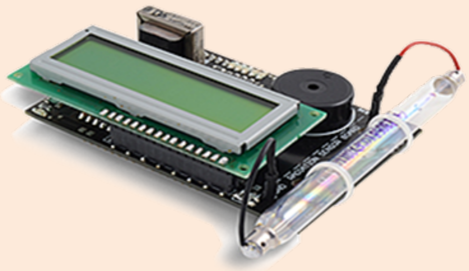
- Bio:
  - Phd researcher and lecturer at Malmo University.
  - Extensive experience in teaching with robots.
  - Co - founder of Arduino.
- Wired Magazine article boosted sales in 2008.
- Robots – kids prefer them over virtual software environments.
- Runs the robotics summer camp held in Madrid
- Teachers should be experienced in what they teach. IT teachers should have IT experience.
- Replace exams with critique sessions and presentations.

David Cuartielles Key Points:

- Arduino is an open ended tool. There are infinite possibilities for usage:

Sony Programmable Watch

Earthquake Sensor



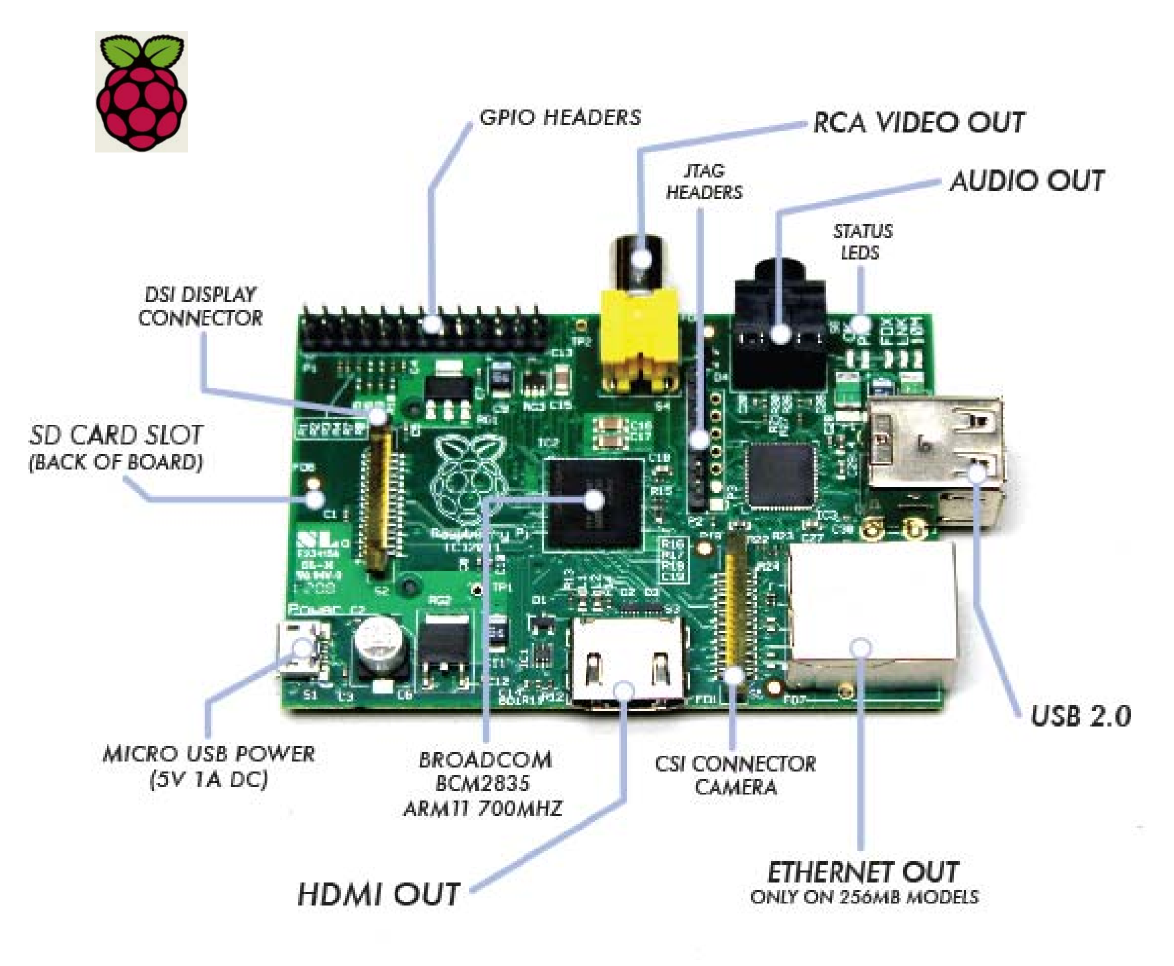
Radiation Detector

David Cuartielles Speaking Videos:

- TEDx Talk: http://www.tedxbrussels.eu/2011/speakers/david_cuartielles.html
- Maker Faire: Hands on with the Arduino Robot:
- http://www.tested.com/tech/robots/455850-maker-faire-2013-hands-arduino-robot/

Clive Beale: The Raspberry PI

The Raspberry Pi is a credit-card sized computer that plugs into your TV and a keyboard. It can be used for many of the things that your desktop PC does, like spreadsheets, word-processing and games. It also plays high-definition video. We want to see it being used by kids all over the world to learn programming.


GPIO HEADERS
RCA VIDEO OUT
JTAG HEADERS
AUDIO OUT
STATUS LEDS
DSI DISPLAY CONNECTOR
SD CARD SLOT (BACK OF BOARD)
USB 2.0
MICRO USB POWER (5V 1A DC)
BROADCOM BCM2835 ARM11 700MHZ
CSI CONNECTOR CAMERA
ETHERNET OUT ONLY ON 256MB MODELS
HDMI OUT

The Raspberry Pi's Interfaces

Clive Beale Key Points:

- Bio:
  - Director of educational development, Raspberry Pi Foundation.
  - His strong teaching background was evident during the workshops.
- Raspberry Pi created because there were no home computers to hack as there were in the ‘80s.
- Decreasing quality of graduates arriving at job interviews – Eben Upton, creator.
- Highest usage in Europe. N. America in close second.
- Now that Raspberry Pi is selling and shipping, funds can be directed towards education.

Clive Beale Key Points:

- Highlighted uses for Raspberry Pi:
  - Can be used as a media center, however, it is “hands on” and requires programming.
  - A 13 year old sent one up in a weather balloon and filmed marshmallows expanding.
  - Processor used in Mars Rover model created in a school.
  - Interface to sensor boards can sense weather data such as humidity and temperature.
  - Hook into Minecraft to create a physical plunger.
  - Realtime clock teaches math by stealth.

Scratch with Raspberry PI Workshop:

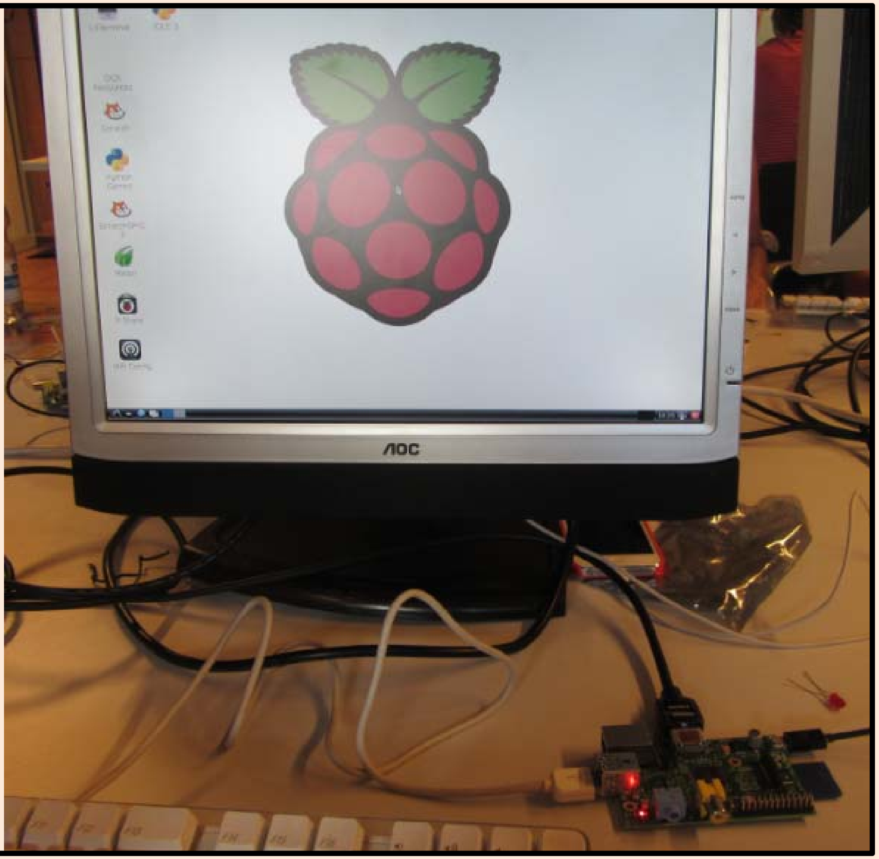
We ran the new Pi distribution (named GPIO2) which contains a version of Scratch which interfaces to the IO pins.

Clive provided the Pis and guidance.

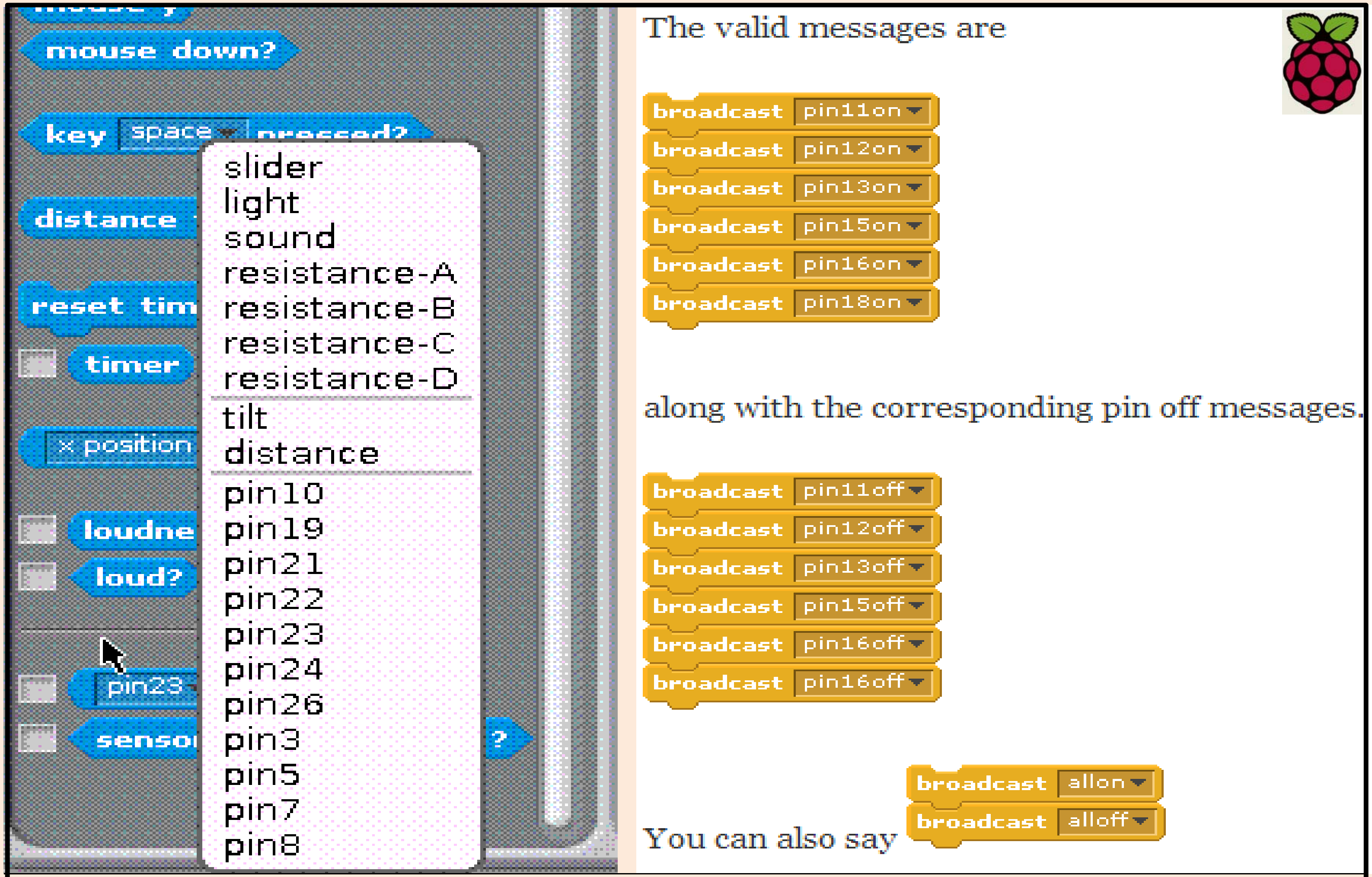
Workshop underway. Mitch Resnick observing in the background.

Scratch with Raspberry PI Workshop:

We wrote a Scratch program that would turn lights attached to the Pi's gpio pins on and off. Each of the teams had created their own original program.



Scratch with Raspberry PI Workshop:



The valid messages are

broadcast pin11on
broadcast pin12on
broadcast pin13on
broadcast pin15on
broadcast pin16on
broadcast pin18on

along with the corresponding pin off messages.

broadcast pin11off
broadcast pin12off
broadcast pin13off
broadcast pin15off
broadcast pin16off
broadcast pin16off

You can also say

broadcast allon
broadcast alloff

Scratch GPIO2: Raspberry Pi hardware resources are seen graphically.

The Beauty and Joy of Computing

A computer science course for high school or undergraduates

- BJC Facts:
  - A college course for non traditional computer science students.
  - Uses Snap! Programming environment.
  - Course materials are free online
  - Social implications of computing.
  - Limits of computing.
  - How stuff works:
    - 3D Graphics
    - Video Games
    - Computational Game Theory

Presenter Dan Garcia, UC Berkeley

bjc The Beauty and Joy of Computing

A computer science course for high school or undergraduates

- BJC was created in response to low interest in computing in schools.
- There are more jobs than skilled people. As a result, scientists such as physicists and mathematicians are being recruited to fill IT roles.
- A U.S poll has revealed that IT jobs rank highest amongst the best jobs to have.
- Dot com bubble burst dropped the numbers of IT people to an all-time low. This remains so today.
- Gender bias is 80% male 20% female.
- Low secondary school participation in computing leads to low numbers in college participation.

- President Obama has called for teaching computer programming in schools in a Google+ interview:

http://www.youtube.com/watch?v=PClfyIbIr5Q

bjc The Beauty and Joy of Computing

A computer science course for high school or undergraduates

BJC is an implementation of csprinciples.org **-** a program which seeks to broaden participation in computing with the following:

6 Computational Thinking Practices:

- **Connecting** computing
- **Developing** computational artifacts
- **Abstracting**
- **Analyzing** problems & artifacts
- **Communicating**
- **Collaborating**

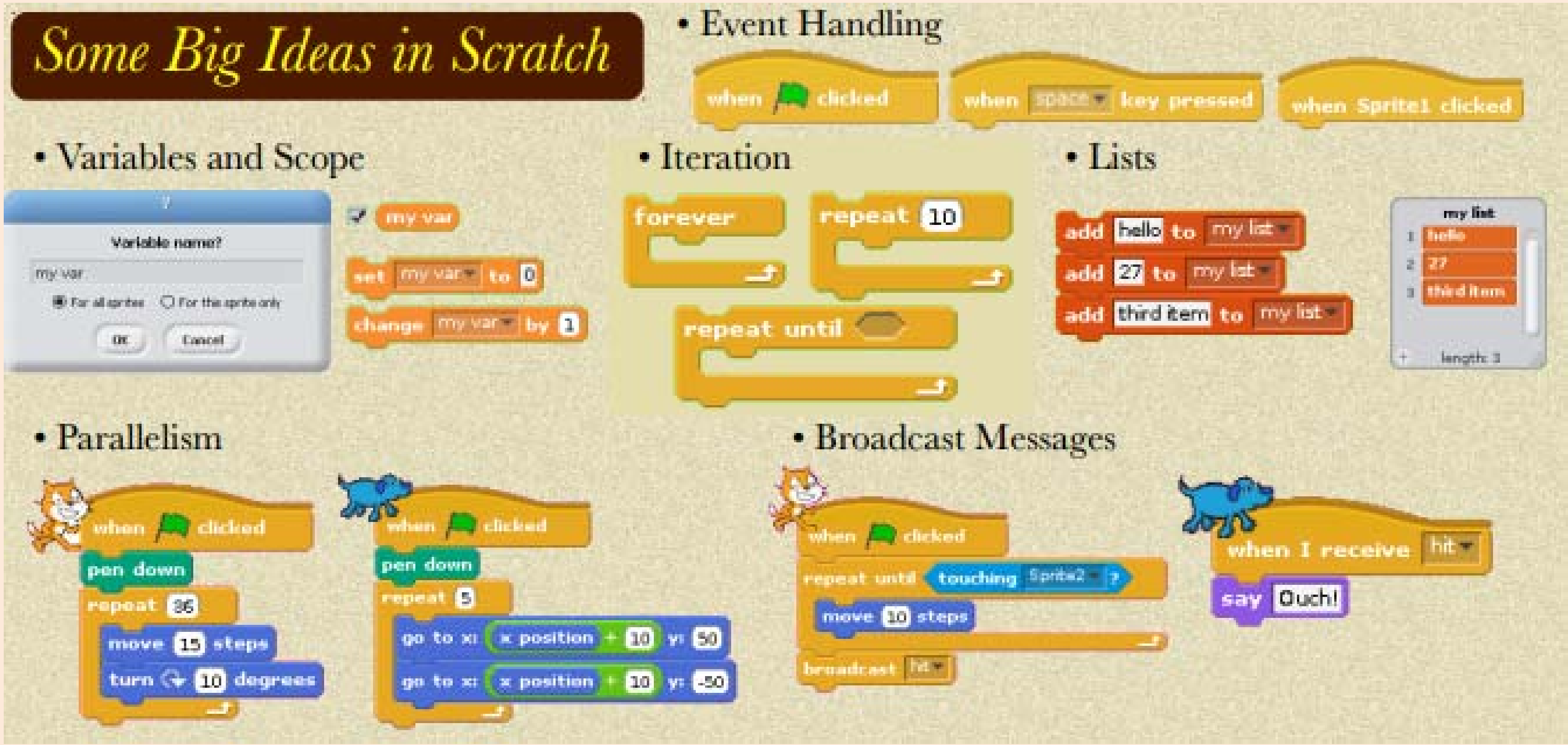
7 Big Ideas:

- computing is a **Creative** activity.
- **Abstraction** reduces information and detail to focus on relevant concepts.
- **Data** and information facilitate the creation of knowledge
- **Algorithms** are used to develop and express solutions to computational problems
- **Programming** enables problem solving, human expression, and creation of knowledge.
- the **Internet** pervades modern computing.
- computation has global **Impacts.**

- Other points:
  - – Computing today is highly collaborative.
  - – Gender balance climbs towards parity when a broader computing curriculum is taught.
  - – CS10k.org: Project goal is to have rigorous, academic curricula incorporated into computing courses in 10,000 high schools, taught by 10,000 well-trained teachers.

Snap! = Scratch Extended

Snap! Is a visual grammar programming environment inspired by Scratch which has been extended to teach higher level computer science concepts.



Some Big Ideas in Scratch

- Event Handling

```
when clicked
```

```
when space key pressed
```

```
when Sprite1 clicked
```

- Variables and Scope

Variable name?

my var

● For all sprites ○ For the sprite only

OK Cancel

```
☑ my var
set my var to 0
change my var by 1
```

- Iteration

```
forever
```

```
repeat 10
```

```
repeat until
```

- Lists

```
add hello to my list
add 27 to my list
add third item to my list
```

my list	
1	hello
2	27
3	third item
length: 3	

- Parallelism

```
when clicked
pen down
repeat 36
    move 15 steps
    turn 10 degrees
```

```
when clicked
pen down
repeat 5
    go to x: (x position + 10) y: 50
    go to x: (x position + 10) y: -50
```

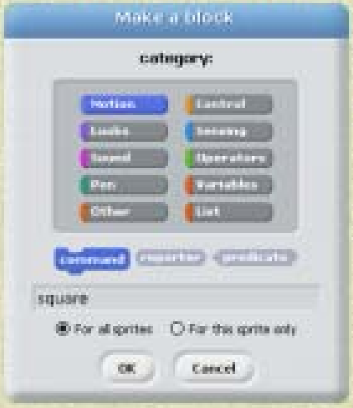
- Broadcast Messages

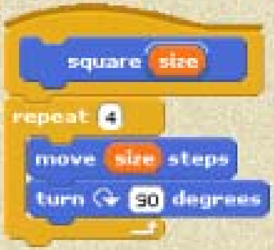
```
when clicked
repeat until touching Sprite2 ?
    move 10 steps
broadcast hit
```

```
when I receive hit
say Ouch!
```

Big Ideas Added in λSnap! (http://snap.berkeley.edu)

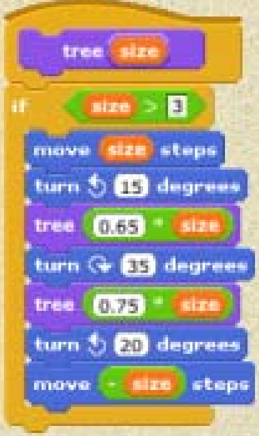
- Procedures (Build Your Own Blocks)



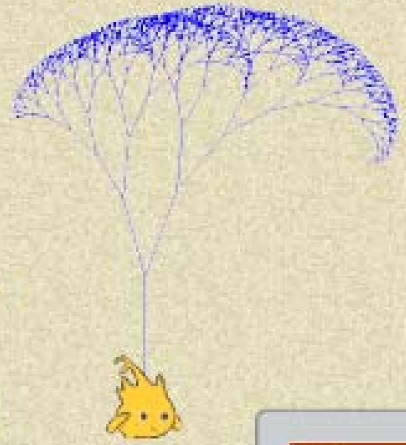


```
square (size)
repeat 4
    move (size) steps
    turn ↻ 90 degrees
```

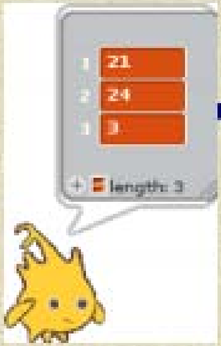
- Recursion

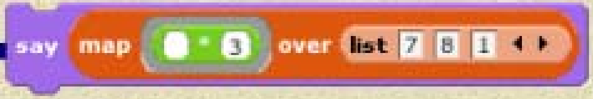


```
tree (size)
if < (size) > 3 >
    move (size) steps
    turn ↺ 15 degrees
    tree (0.65 * (size))
    turn ↻ 35 degrees
    tree (0.75 * (size))
    turn ↺ 20 degrees
    move (- (size)) steps
```

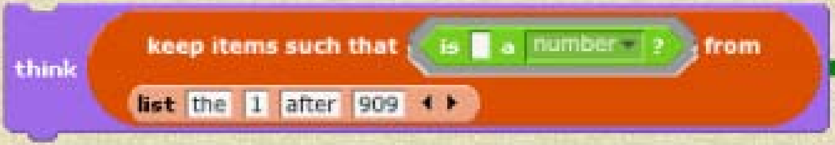


- Functions and Lists as First Class Data

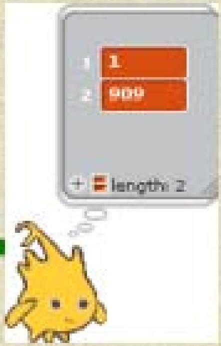


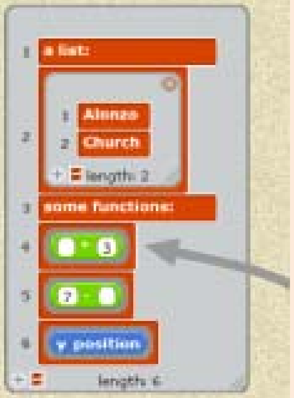


```
say (map (( ) * 3) over (list 7 8 1 ◂ ▸))
```

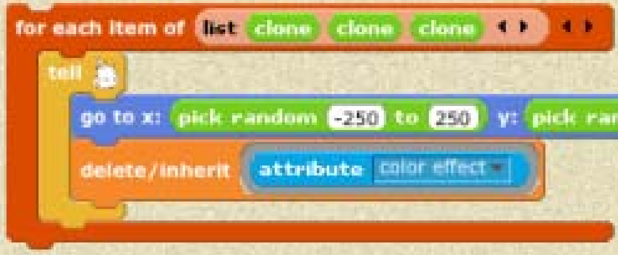


```
think (keep items such that <is [ ] a [number] ?> from (list the 1 after 909 ◂ ▸))
```

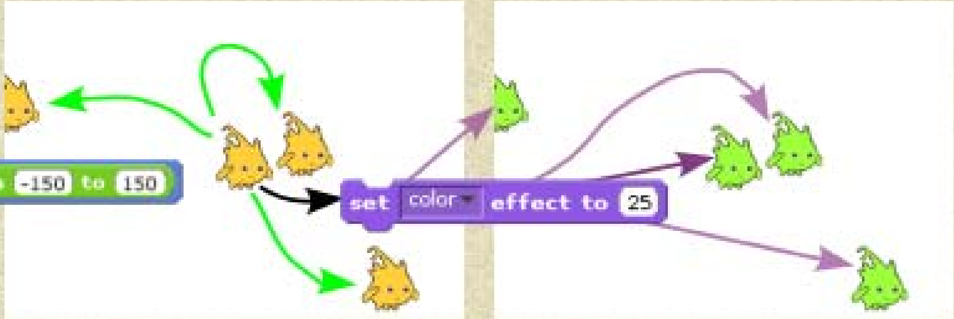




- Object Oriented First Class Sprites with Inheritance



```
for each (item) of (list (clone) (clone) (clone) ◂ ▸) ◂ ▸
    tell (item) [
        go to x: (pick random -250 to 250) y: (pick random -150 to 150)
        delete/inherit (attribute [color effect])
    ]
```



```
set [color] effect to 25
```

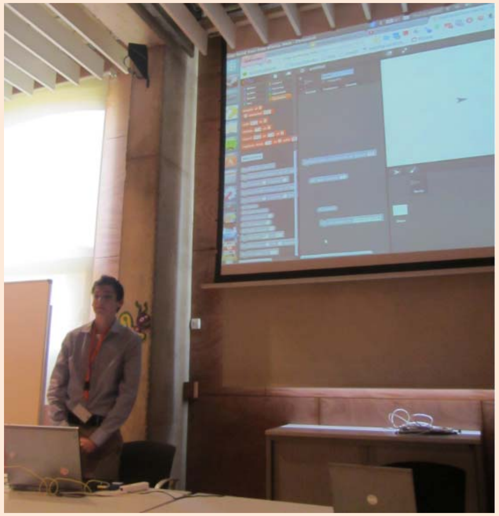
Facts and Features

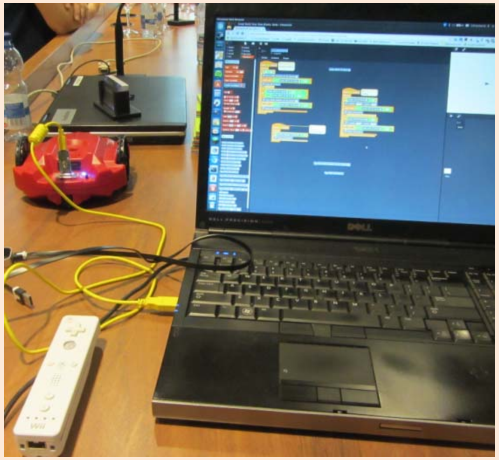
- Written in HTML5 and Javascript
  - Runs on tablets and phones
- HTTP blocks for web connectivity
- Device support
- Created by Brian Harvey (UC Berkeley) & Jens Mönig (Mio Soft)

- Snap! Website: http://byob.berkeley.edu/

Conor Hudson

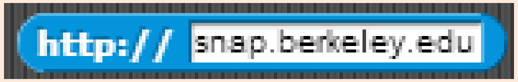
- Presented device interfaces he developed for Snap! in Python:
  - Lego NXT
  - Nintendo Wiimote
  - Parallax S2 robot
  - Leap Motion
  - Speech synthesis
  - His device interface architecture caught the attention of the Scratch development team.
  - He’s 15 years old. See Snap! website for links to his drivers (and others).



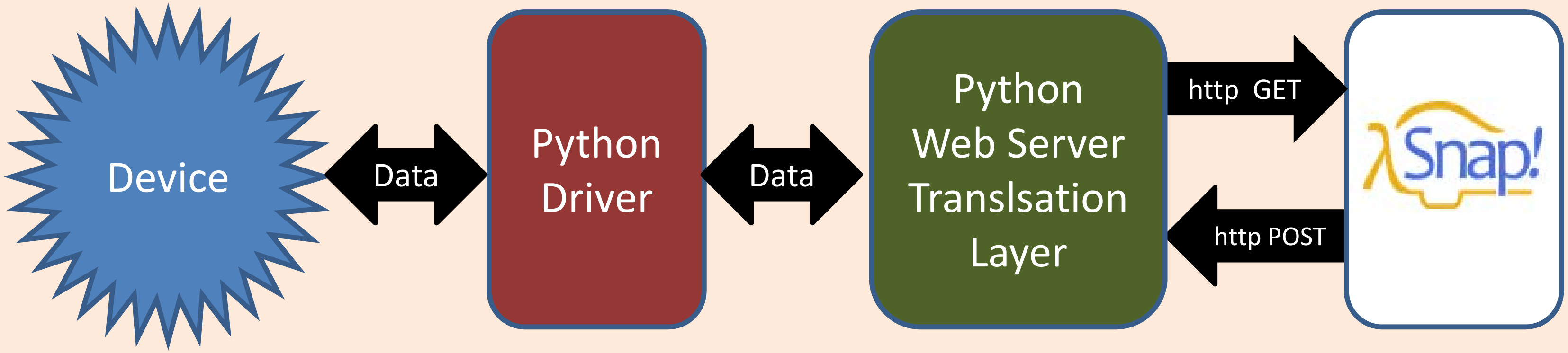


Python <-> Snap! Device Interface Architecture

- Python drivers can be found for most devices.
- Developer must create a Python web server which acts as translation layer between the driver and Snap!.
- Snap! data IO via the http block:



http:// snap.berkeley.edu



Device

Data

Python Driver

Data

Python Web Server Translsation Layer

http GET

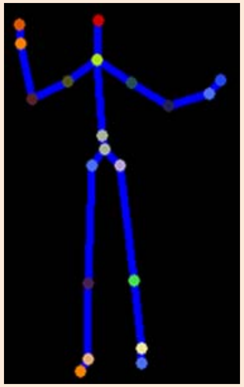
http POST

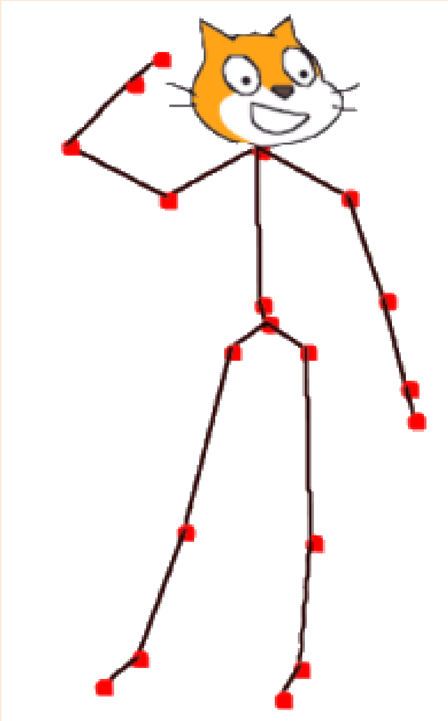
Snap!

Kinect2Scratch

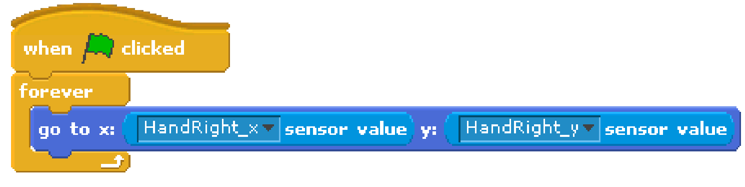
http://scratch.saorog.com/

Stephen Howell of IT Tallaght demonstrated how the Kinect can be connected to Scratch.





24 Kinect points are mapped to Scratch sprites.



```
when ⚑ clicked
forever
    go to x: (HandRight_x ▾ sensor value) y: (HandRight_y ▾ sensor value)
```

Each sprite can be easily programmed to follow the position of a Kinect point. The possibilities for game creation are unlimited.

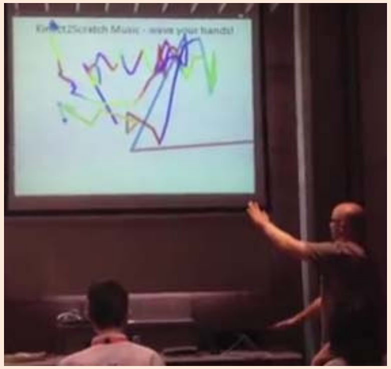
Playing Space Invaders with Kinect using body motion.

Leap2Scratch

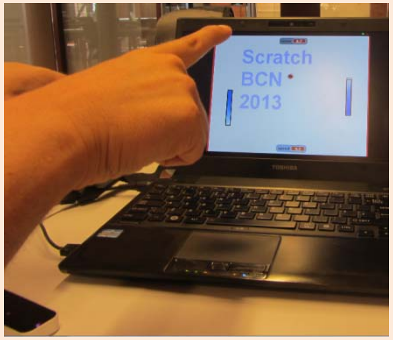
www.leapmotion.com/

- The Leap Motion controller tracks the motion and position of each finger in each hand in 3 dimensional space.

- Stephen Howell demonstrating fingertip control of a Scratch program via the Leap Motion controller.



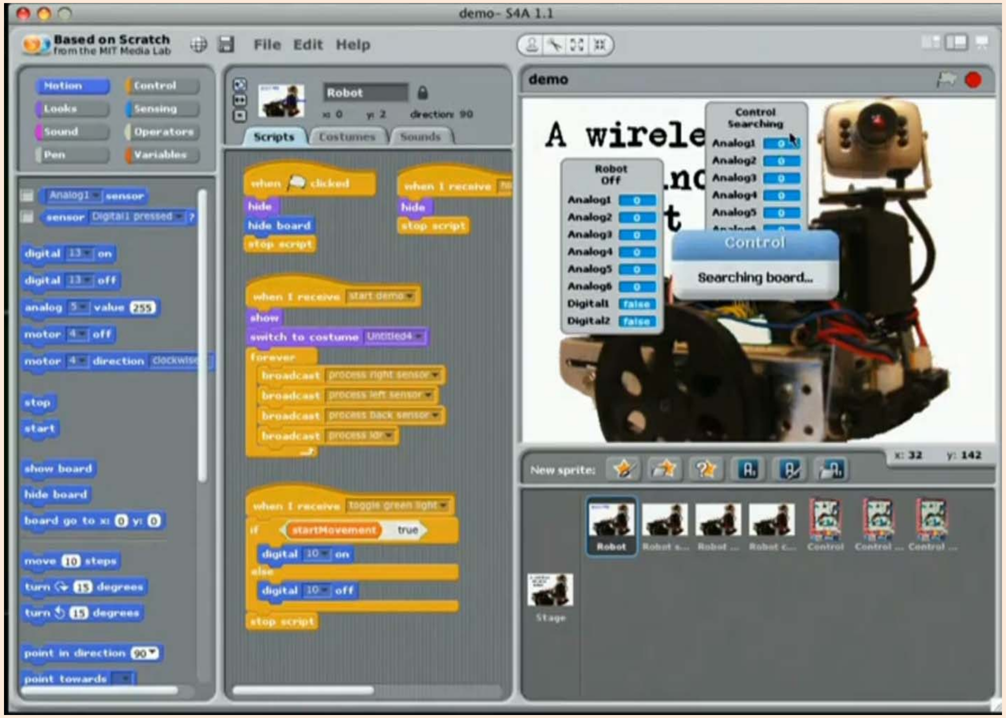
- Scratch programs can be created to respond to hand and finger movements. This a 2 player pong game written in Scratch and controlled by fingers.



S4A (Scratch for Arduino)

http://seaside.citilab.eu/scratch/arduino

S4A is a Scratch variant which has a built-in Arduino device interface which offers the opportunity to write programs to control the Arduino in Scratch.

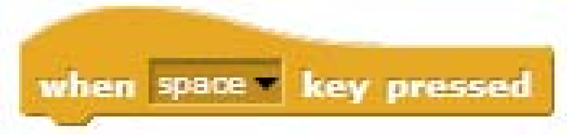


```
when [flag] clicked
hide
hide board
stop script

when I receive [...]
hide
stop script

when I receive [start demo]
show
switch to costume [Untitled4]
forever
  broadcast [process right sensor]
  broadcast [process left sensor]
  broadcast [process back sensor]
  broadcast [process ldr]

when I receive [toggle green light]
if <(startMovement) = true>
  digital [10] on
else
  digital [10] off
stop script
```

Makey Makey

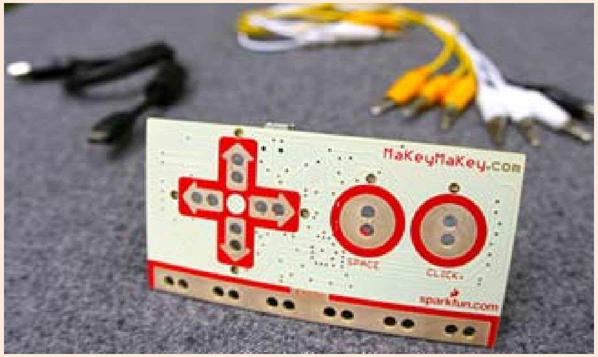
www.makeymakey.com/

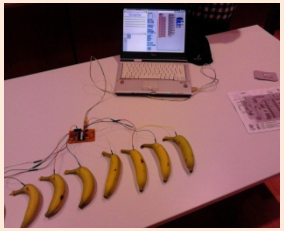
The Makey Makey is a keyboard emulator circuit board which has pins which correspond to the each key on the keyboard. When the circuit is closed a keypress or mouse move message is sent to the computer.



```
when space key pressed
```

Switches, buttons, play dough, pieces of fruit can be hooked up to Scratch. Programs can then be written to handle the key press events.



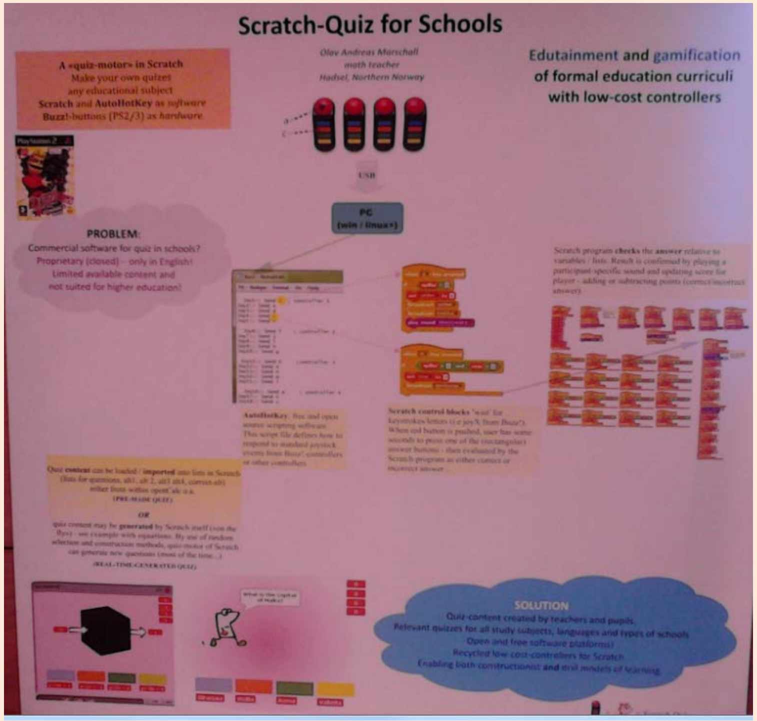


Classroom Quiz Game

PS2 controllers are connected to Scratch via Autohotkey.

The Scratch program would ask a question:

- The first person to hit the big button must choose the correct answer via the corresponding small button.



Scratch-Quiz for Schools

Olav Andreas Marschall
math teacher
Hadsel, Northern Norway

A «quiz-motor» in Scratch
Make your own quizes
any educational subject
Scratch and AutoHotKey as *software*
Buzz!-buttons (PS2/3) as *hardware*

Edutainment and gamification
of formal education curriculi
with low-cost controllers

PROBLEM:
Commercial software for quiz in schools?
Proprietary (closed) – only in English!
Limited available content and
not suited for higher education!

Scratch program **checks** the **answer** relative to variables / lists. Result is confirmed by playing a participant-specific sound and updating score for player - adding or subtracting points (correct/incorrect answer).

AutoHotKey: free and open source scripting software. This script file defines how to respond to standard joystick events from Buzz! controllers or other controllers.

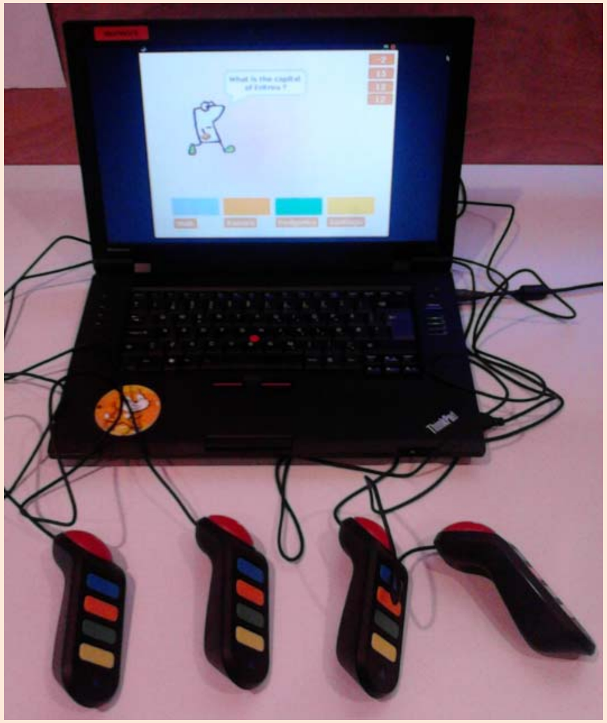
Scratch control blocks 'wait' for keystrokes letters (i.e joyN from Buzz!). When red button is pushed, user has some seconds to press one of the (rectangular) answer buttons - then evaluated by the Scratch program as either correct or incorrect answer.

Quiz **content** can be loaded / **imported** into lists in Scratch (lists for questions, alt1, alt 2, alt3 alt4, correct-alt) either from within openOffice a.o.
(PRE-MADE QUIZ)

OR

quiz content may be **generated** by Scratch itself (on the fly) - see example with equations. By use of random selection and construction methods, quiz-motor of Scratch can generate new questions (most of the time...)
(REAL-TIME-GENERATED QUIZ)

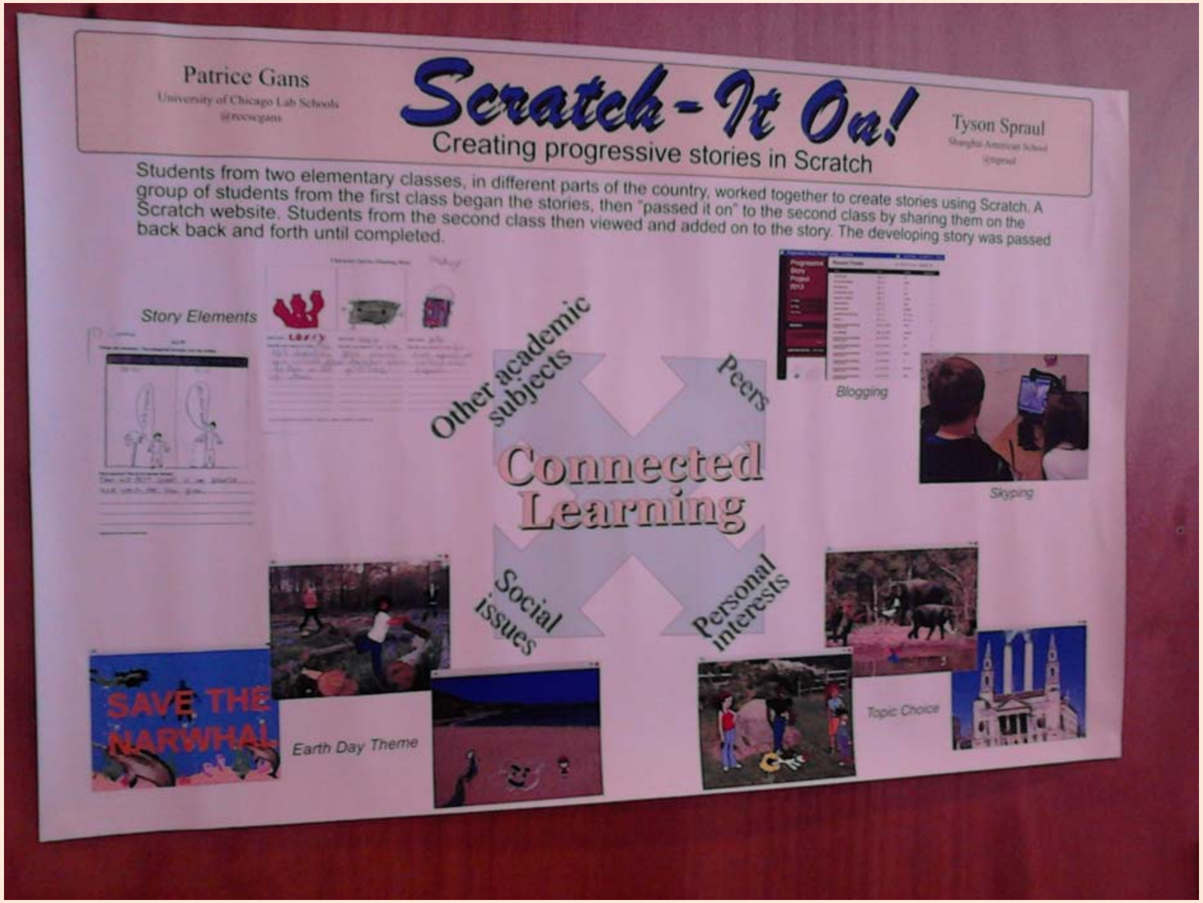
SOLUTION
Quiz-content created by teachers and pupils
Relevant quizzes for all study subjects, languages and types of schools
Open and free software platforms)
Recycled low cost-controllers for Scratch
Enabling both constructionist and drill models of learning



Connected Learning With Scratch

- Kids in separate locations collaborated using online tools to create a Scratch project.

- Tools used:
  - Skype
  - Email
  - Blogs
  - Scratch Website



Patrice Gans
University of Chicago Lab Schools
@reesegans

Scratch-It On!
Creating progressive stories in Scratch

Tyson Spraul
Shanghai American School
@tspraul

Students from two elementary classes, in different parts of the country, worked together to create stories using Scratch. A
group of students from the first class began the stories, then "passed it on" to the second class by sharing them on the
Scratch website. Students from the second class then viewed and added on to the story. The developing story was passed
back back and forth until completed.

Keynote Day 3: Formal & Informal Learning

James Whelton of Coder Dojo, Suchi Grover of Stanford University

- **James Whelton** – an enthusiastic coder, ran a computer club in his secondary school. He met up with businessman Bill Lao and they founded Coder Dojo in Ireland in early 2011. These are free clubs where kids help each other to program.

- **Shuchi Grover** - a computer scientist and educator by training, is currently a doctoral candidate at Stanford University. One of her primary research interests is concerned with bringing meaningful computing education to the secondary level.

James Whelton of Coder Dojo

- Coder Dojo is an example of informal learning.
- Pros
  - Low expectations
  - Easily react quickly to new curriculum
  - “Not school” environment appeals to kids
- Cons
  - Lack of assessment
  - Spotty coverage
- Symbiotic relationship with formal education
  - “Informal informs formal”

Coder Dojo Facts

- Recognized as an Irish export worldwide.
- 200 dojos in 23 countries. 10,000 kids.
- De-centralized movement: no HQ or bank account.
- Dojos run in Irish, EU, Scottish parliaments.
- Ethos: Kids helping kids. Be cool – don’t harm.
- helloworldfoundation.com has been created to achieve scale in the delivery of resources and support to dojos.
  - Mission statement: “Inspire Young Coders and Support Coding Clubs”

Coder Dojo Group in Barcelona 2013

Suchi Grover

- A balance must be struck between formal and informal learning:
  - Informal learning is by discovery and is incidental.
  - Formal learning is structured and goal oriented.
- The process is what matters, not the end result.
  - “But it works”! - is not good enough.
  - Must be able to understand the design process.
- Learning by experimentation engenders a bottom-up development processes, however, there are benefits:
  - Buggy code is a learning opportunity
  - Debugging boundary conditions leads to understanding.

Snap! And Scratch Developments

- **Kurt De/Compiler**: Allows Scratch projects to be converted to Python code where they can be modified via text editing, then re-converted back to Scratch projects. http://archive.scratch.mit.edu/forums/viewtopic.php?id=92843
- **Generating Code from Snap! Blocks**: Currently under development. Converting snap blocks to C, Javascript, Python, Squeak languages was demonstrated by Jens Mönig.

Robot Exhibits

- **Hummingbird & Finch** are robots which can be driven by Snap! via Java and Python interfaces. http://www.hummingbirdkit.com/learning/software/snap
- **Lego Mindstorms NXT & Enchanging**: Enchanting is a tool to allow children an easy way to program LEGO MINDSTORMS NXT. It is based on Scratch and BYOB/Snap!, and powered by leJOS NXJ(Java for the NXT). Enchanting is free and open-source. http://enchanting.robotclub.ab.ca/tiki-index.php

Links to Video Interviews at Scratch 2013 Barcelona

- Mitch Resnick: http://www.youtube.com/watch?v=yAgE_ek2K0o
- Karen Brennan: http://www.youtube.com/watch?v=adJsop3vfXY
- Sayamindu Dasgupta (Scratch SW dev team): http://www.youtube.com/watch?v=6aWsdwysneQ
- David Cuartielles : http://www.youtube.com/watch?v=-m-mOw8Q3QM
- Clive Beale: http://www.youtube.com/watch?v=fYos5JPtwAE
- James Whelton: http://www.youtube.com/watch?v=ThIKegE3OTo
- Suchi Grover: http://www.youtube.com/watch?v=7MVVUV5cVwM
- Brian Harvey and Jens Mönig: http://www.youtube.com/watch?v=rXN81Hsj_A4
- Frank Sabaté (Scratch teacher) : http://www.youtube.com/watch?v=-etPAFLOZiw

Thanks!

Thank you Mitch, the Scratch Team, and the Scratch Community for a interesting and inspirational experience at Scratch 2013 BCN.

- Steve