Snap! (by Berkeley)

Andrea Sterbini – sterbini@di.uniroma1.it

Snap! (by Berkeley)

"Scratch for the Computer Scientist"

Object orientation

Many extensions/libraries

Support for code documentation

Support for debugging

Concurrency

Coroutines

Functional programming (APL)

...

Scalar programming (APL)

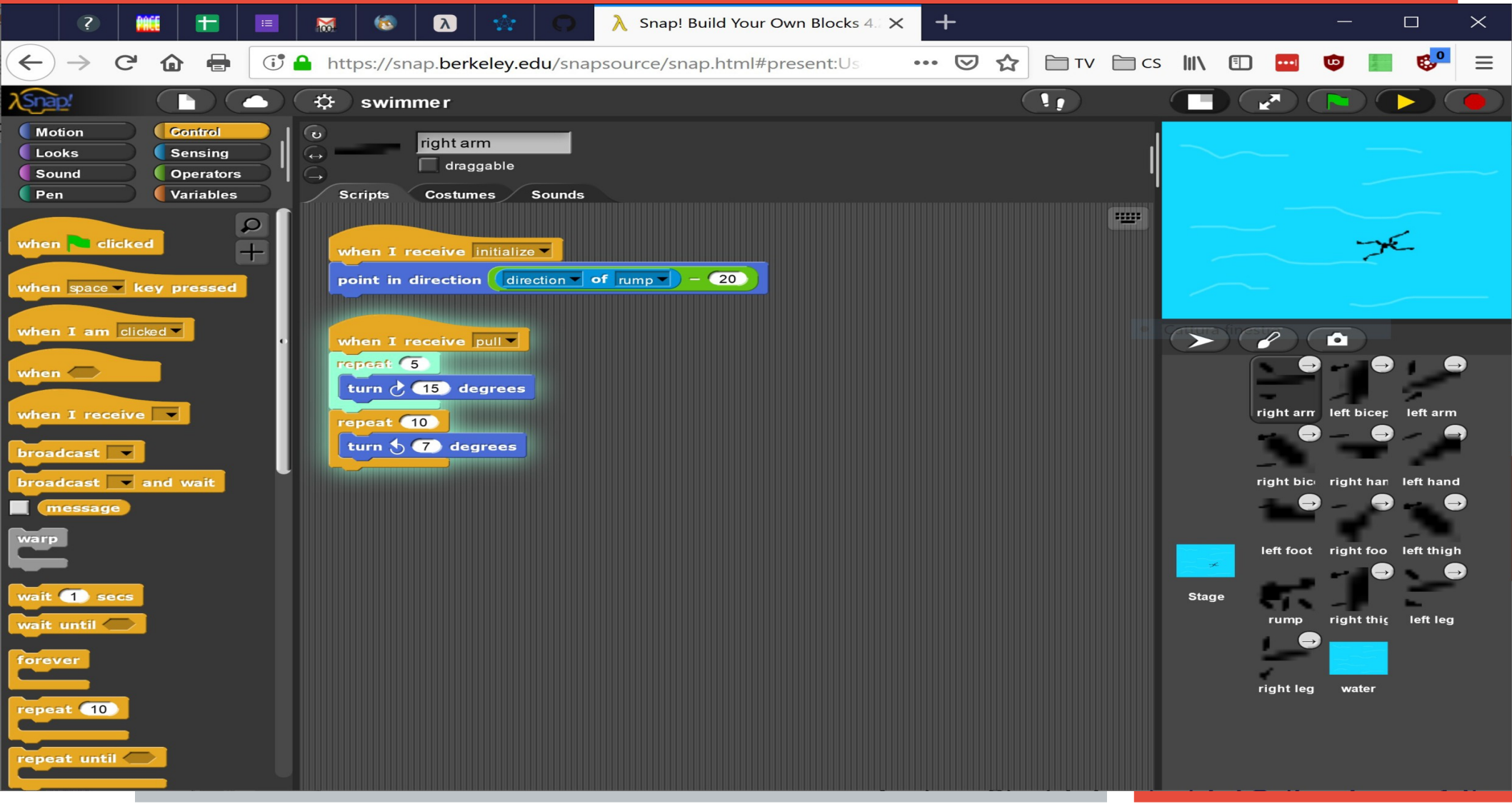
Music

Relative motion of sprites

HTML5 web app

Easy local install (just unzip)

See the Snap! manual for more info


Snap! Build Your Own Blocks 4
https://snap.berkeley.edu/snapsource/snap.html#present:Us
swimmer
Motion
Looks
Sound
Pen
Control
Sensing
Operators
Variables
when clicked
when space key pressed
when I am clicked
when
when I receive
broadcast
broadcast and wait
message
warp
wait 1 secs
wait until
forever
repeat 10
repeat until
right arm
draggable
Scripts
Costumes
Sounds
when I receive initialize
point in direction direction of rump - 20
when I receive pull
repeat 5
turn 15 degrees
repeat 10
turn 7 degrees
Stage
right arm
left bicep
left arm
right bic
right han
left hand
left foot
right foo
left thigh
rump
right thig
left leg
right leg
water

Snap! improves many Scratch language constructs

Scratch

- NO complex data
- NO functions (only procedures)
- NO local variables
- NO references to clones
- NO call methods
- NO libraries

Snap!

- Objects, Lists, Lists of lists, Lists of Objects
- Functions (return)

- Local variables (easy recursion)
- References to clones

- Call methods

- Global blocks (library of functions)
- Inheritance of clone properties
- Anonymous “Lambda” functions

Other functions

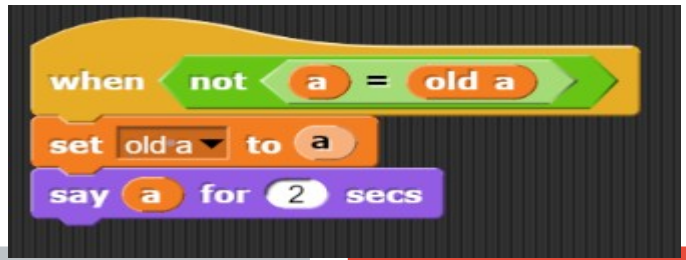
Can create a “costume” by drawing

Objects can ask each other to do / report something

Can use individual messages

Or broadcast messages to all

Generic events
(e.g. variable observer)



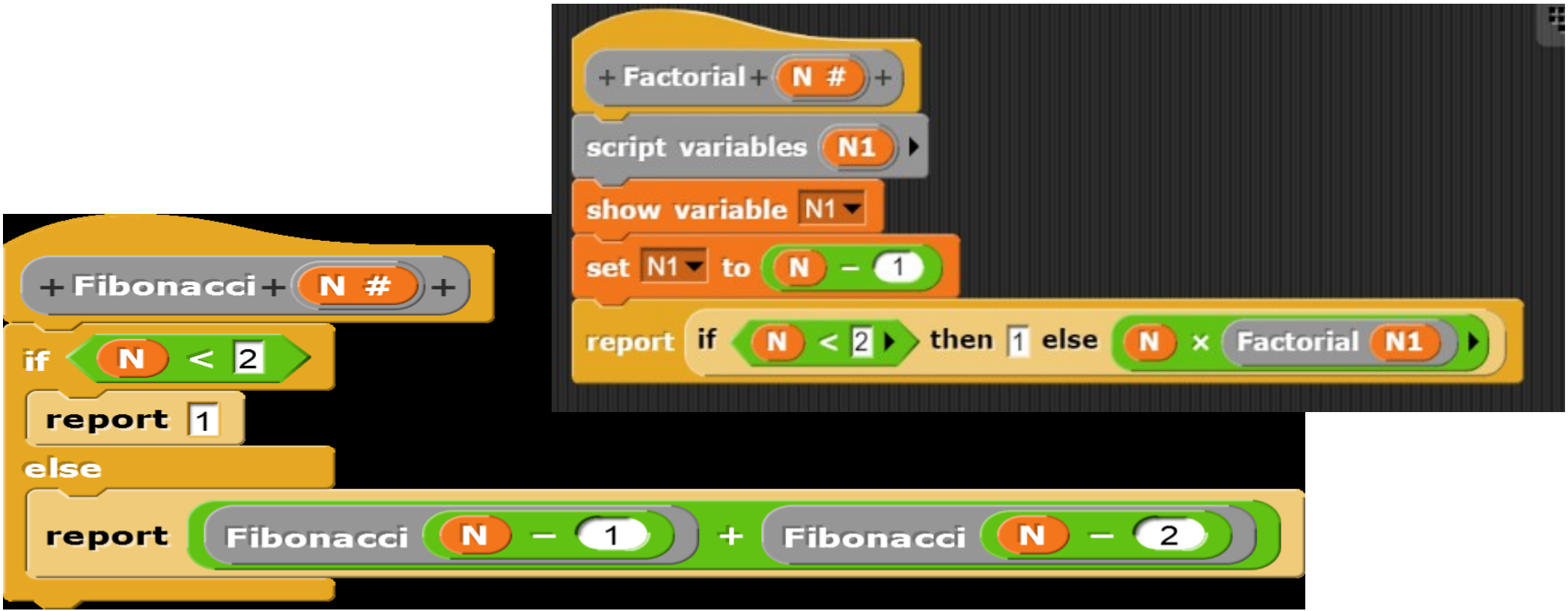
Relative motion of Sprites/Agents wrt other agents

It makes easy building:
collective motion of many clones (fireworks, snow, birds, …)
coordinated motion of an agent with many parts (man walking, multipropeller drone)

Example: Swimmer

- Main motion: body trunk and head (straight motion **bumping** to the walls)
- Attached to body: thighs and biceps (**rotating** w.r.t. the body)
- Attached to thighs and biceps: arms and legs (just kept in the body direction)
- Attached to arms and legs: hands and feet (**rotating** w.r.t. the arm and leg)

Easy recursion (with local variables)



Standard Libraries/Extensions

Loops and compositions

List operations

Streams (lazy lists)

Multiple args operators

Web access

Words manipulation

Switch/case

RGB/HSV colors

Handle big lists

Frequency distribution analysis

Try/catch

Multiline input

GUI settings

Bignum, rational, complex

Text to speech

Animations

Image manipulation

Audio generation

Json

Parallelization

and more ...

Other extensions

SOFTWARE:

Cellular automata (Cellular)

Graphs (Edgy)

NLP (NLTK wrapper)

HARDWARE:

Orbotix Sphero

Lego NXT (but not EV3 yet)

Wiimote

Arduino

Raspberry Pi

Speech synthesis

LEAP

Finch, Hummingbird

Many programming styles!

Functional

Lists, filters, map, coroutines, continuations, generators

Procedural

Concurrent

Concurrent execution

Message events

Object-oriented/Agent based

Agent properties, Agent methods

Clones: references to created clones, inherited properties

Snap! for C.T. applied to other Subjects

Pro:

- Rich language with all CS constructs and more!
- Rich data structures (including objects, Json and CSV tables)
- Easy animation of multi-agent groups with relative motion
- Many extension libraries

Con?:

- Sophisticated constructs for more experienced programmers
- Good for older students and more complex projects