Snap! (by Berkeley)

Andrea Sterbini – sterbini@di.uniroma1.it

Snap! (by Berkeley)

Evolution of Scratch

“Scratch for the Computer Scientist”

Object orientation

Many extensions/libraries

Support for code documentation

Support for debugging

Concurrency

Coroutines

Functional programming (APL)

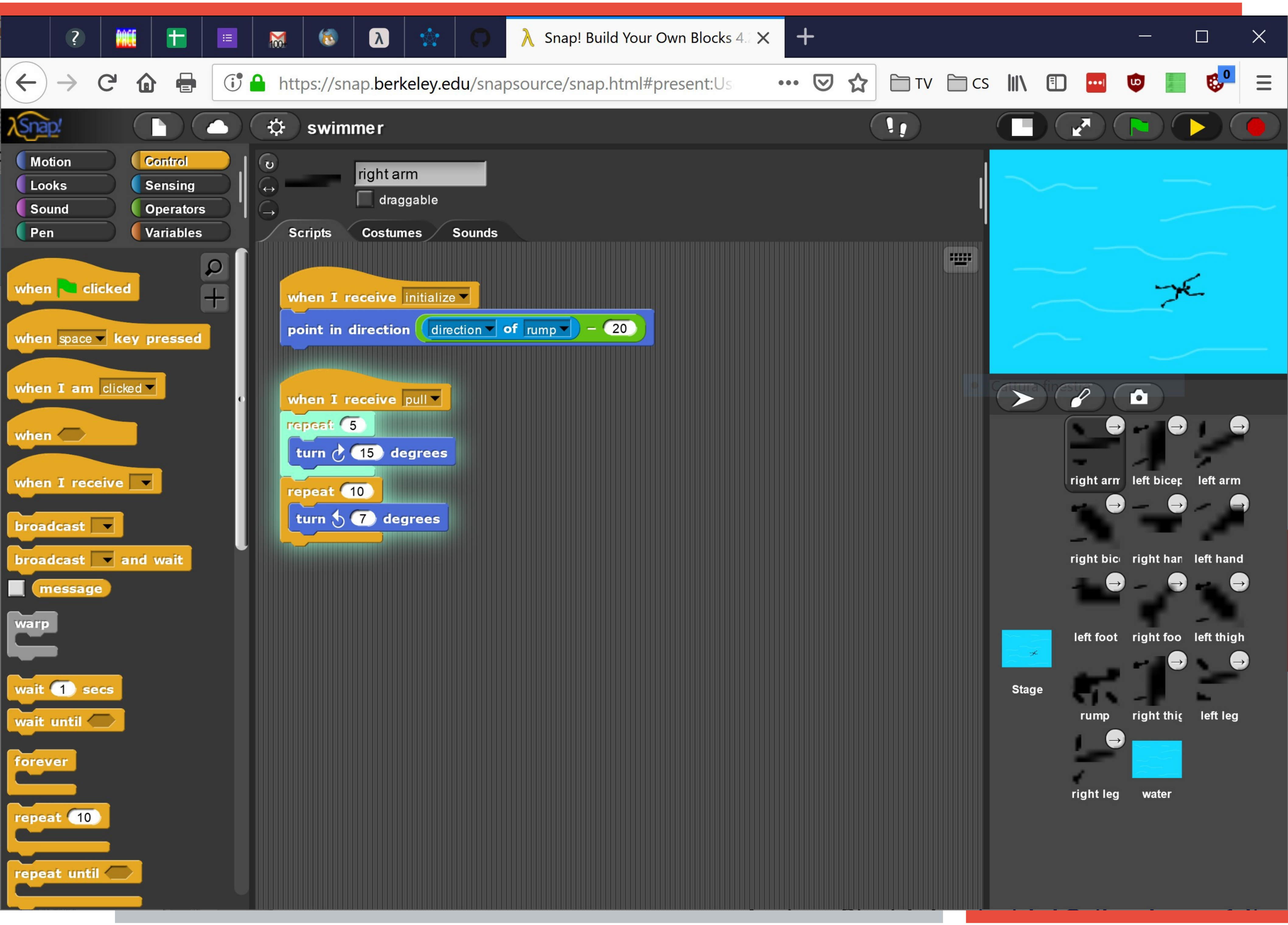
...

Music

Relative motion of sprites

HTML5 web app

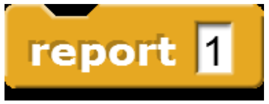
Easy local install (just unzip)

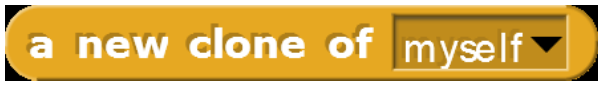


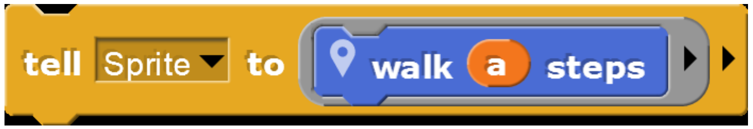
```
when I receive initialize
point in direction ((direction of rump) − 20)

when I receive pull
repeat 5
    turn ↻ 15 degrees
repeat 10
    turn ↺ 7 degrees
```

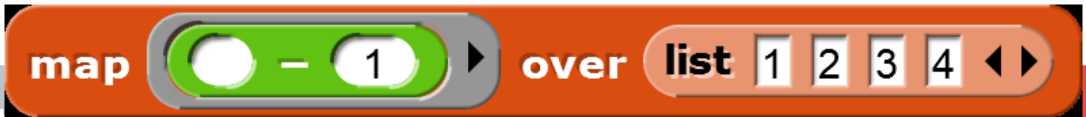
Snap! improves many Scratch language constructs

Scratch	Snap!
NO complex data	Objects, Lists, Lists of Objects
NO functions (only procedures)	Functions (return)
NO local variables	Local variables (easy recursion)
NO references to clones	References to clones
NO call methods	Call methods
NO libraries	Global blocks (library of functions)
	Inheritance of clone properties
	“Lambda” functions



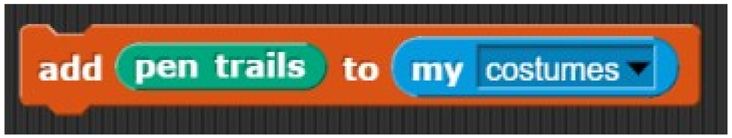






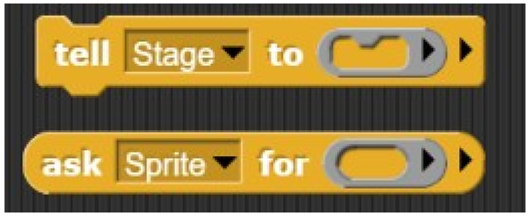
Other functions

Can create a “costume” by drawing



```
add (pen trails) to (my costumes)
```

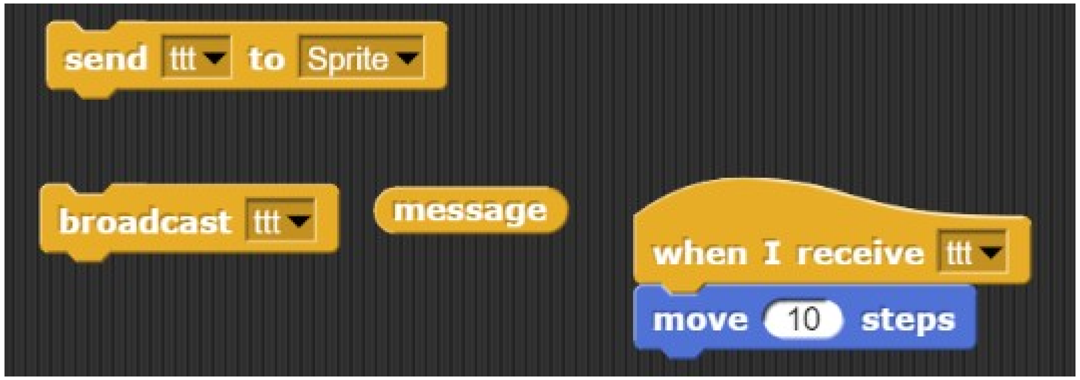
Objects can ask each other to do/compute something



```
tell Stage to ( )
ask Sprite for ( )
```

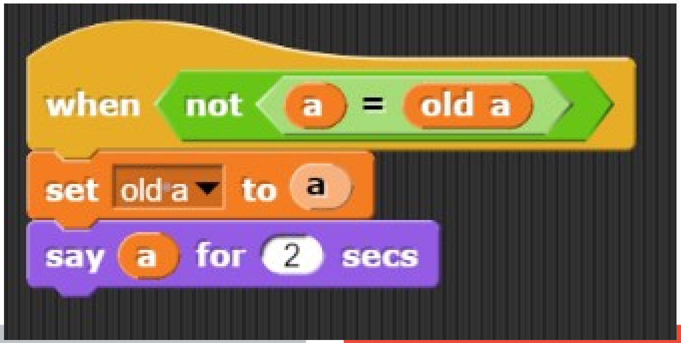
Can use individual messages

Or broadcast messages to all



```
send ttt to Sprite

broadcast ttt     (message)

when I receive ttt
move 10 steps
```

Generic events
(e.g. variable observer)



```
when <not <(a) = (old a)>>
set old a to (a)
say (a) for 2 secs
```

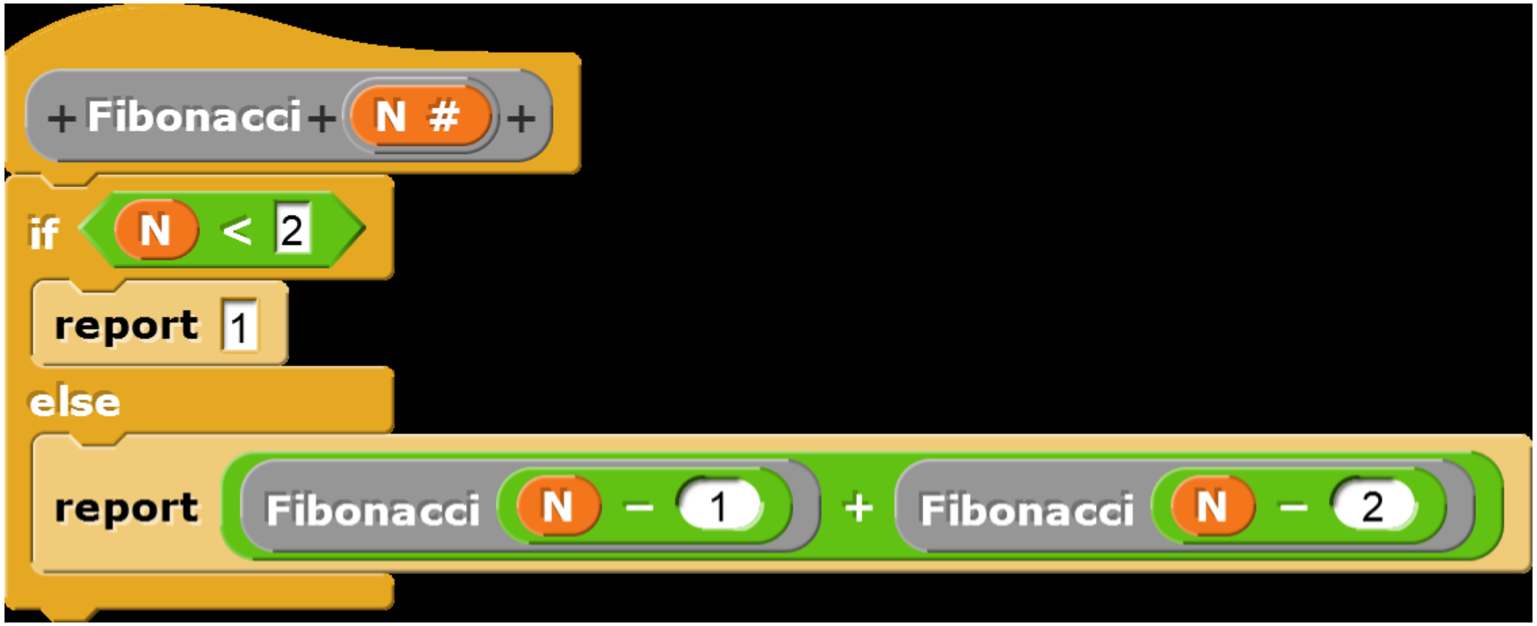
Relative motion of Sprites/Agents

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

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



```
+ Fibonacci + (N #) +
if <(N) < 2>
    report 1
else
    report ((Fibonacci ((N) − 1)) + (Fibonacci ((N) − 2)))
```

Standard Libraries/Extensions

Loops and compositions

List operations

Generators (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

Procedural

Concurrent

- Concurrent execution
- Message events

Object-oriented/Agent based

- Agent properties, Agent methods
- Clones: references to created clones, inherited properties