Not Just for ~~Kids~~ Novices: Blocks Programming Language Design and Implementation in MIT App Inventor

Franklyn Turbak
Wellesley College Computer Science Dept.

Williams College CS Colloquium Talk
October 24, 2014

App Inventor Development Team

Hal
Abelson
MIT

Andrew
McKinney
MIT

Jeff
Schiller
MIT

Paul
Medlock-Walton
MIT

Jose
Dominguez
MIT

Mark
Friedman
Google

Sharon
Perl
Google

Liz
Looney
Google

Neil
Fraser
Google (Blockly)

Franklyn
Turbak
Wellesley College

Computational Thinking Through Mobile Computing NSF Grant Team

Franklyn Turbak Eni Mustafaraj
Wellesley College

Ralph Morelli
Trinity College

Dave Wolber
U. of San Francisco

Larry Baldwin
BIRC

Hal Abelson Shay Pokress Josh Sheldon
MIT

Fred Martin Mark Sherman Karen Roehr
University of Massachusetts Lowell

Acknowledgment: This work was supported by the National Science Foundation under Grants 1225680, 1225719, 1225745, 1225976, and 1226216.

Talk Road Map

- MIT App Inventor (AI) demo
- Some notes on blocks languages in general
- AI blocks language design and implementation
  - Syntax
  - Static Semantics
  - Dynamic Semantics
  - Pragmatics
- Future work

Talk Road Map

- **MIT App Inventor (AI) demo**
- Some notes on blocks languages in general
- AI blocks language design and implementation
  - Syntax
  - Static Semantics
  - Dynamic Semantics
  - Pragmatics
- Future work

Talk Road Map

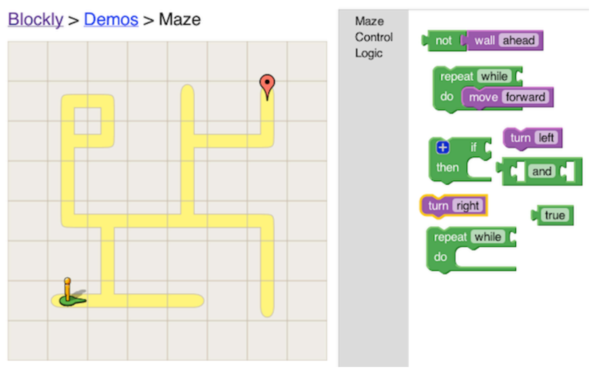
- MIT App Inventor (AI) demo
- **Some notes on blocks languages in general**
- AI blocks language design and implementation
  - Syntax
  - Static Semantics
  - Dynamic Semantics
  - Pragmatics
- Future work

Blocks Languages are Growing in Popularity

Scratch: multi-media programs, animations, and games

MIT App Inventor: apps for Android smartphones

StarLogo Nova: multi-agent simulations



Blockly: Many blocks-based activities; Basis for MIT AI, main code.org challenges.

code.org's Hour of Code: >20M participants, >75% blocks PLs

Negative Responses to Blocks Languages

I have never met a student who cut their teeth in any of these languages and did not come away profoundly damaged and unable to cope.

I mean this reads to me very similarly to teaching someone to be a carpenter by starting them off with plastic toy tools and telling them to go sculpt sand on the beach.

Not one thing they learn will bear any piece of resemblance to real work. All you're doing is teaching them misimpressions of what the job is, and tricking them out of having meaningful formative experiences.

http://blog.acthompson.net/2012/12/programming-with-blocks.html

These are not proper programming languages, anyone with half a brain knows that, but why deny those who can't or don't want to 'code' the opportunity of being creative with these tools and learning some logic skills along the way.

http://blog.acthompson.net/2012/12/programming-with-blocks.html

Working with actual code writing instead of a drag & drop interface prepares children better for the real world.

http://www.playcodemonkey.com/

Mark Sherman’s Response

Mark Sherman
UMass Lowell

So they currently see this:

when it is really this:

Yes, it is colorful and newfangled, but it still gets jobs done. Not all of them, but a bunch of them.

Why do they see it this way? Because they grew up on this:

More Positive Feedback

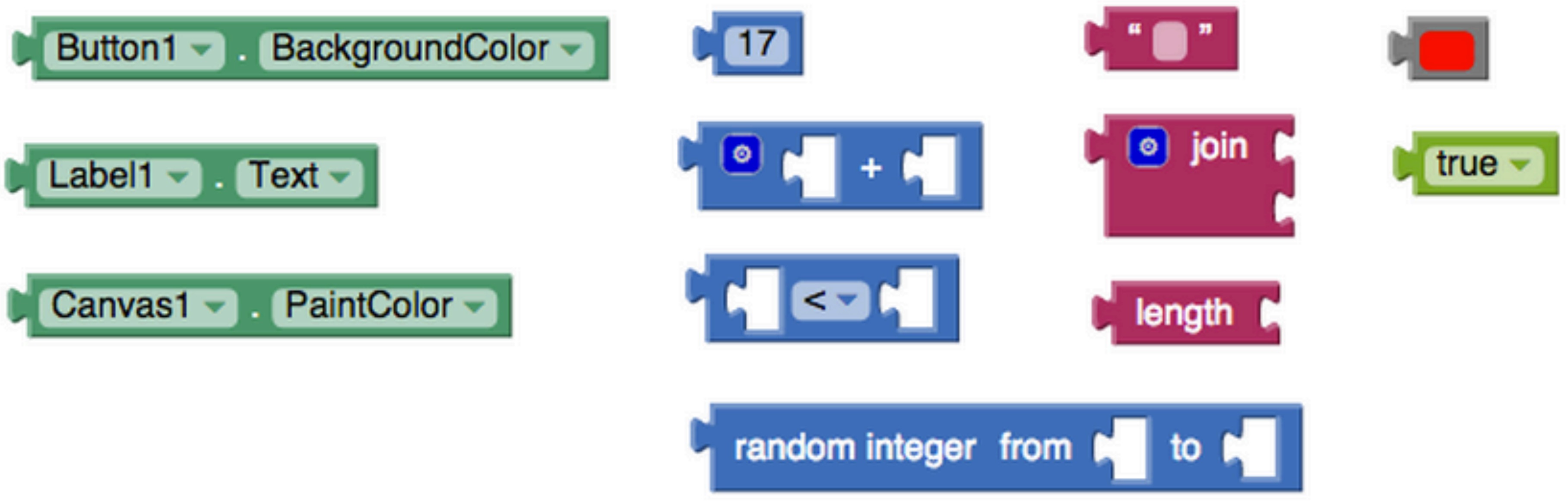
I would like to express my utmost appreciation for your product. I'm teaching several pre-CS courses for gifted youth at Junior-high school level (7th-9th grades) as well as CS and software engineering at high school (10th – 12th grades) including Android development in Java. **It is really amazing that in AppInventor, 7th grade students (with about 50 hours prior experience in Scratch) can do in 6 hours what 12th grade students take about 200-300 hours to achieve in Java (and this is after studying CS and Android development for about 700 hours).** AppInventor goes way beyond the 80:20 principle (80% of the utility in 20% of the effort) – it is more like 60:5 (60% of the functionality, for less than 5% of the effort) which makes it much more fun, and opens up a lot of space for creativity.

Yossi Yaron, Israeli teacher

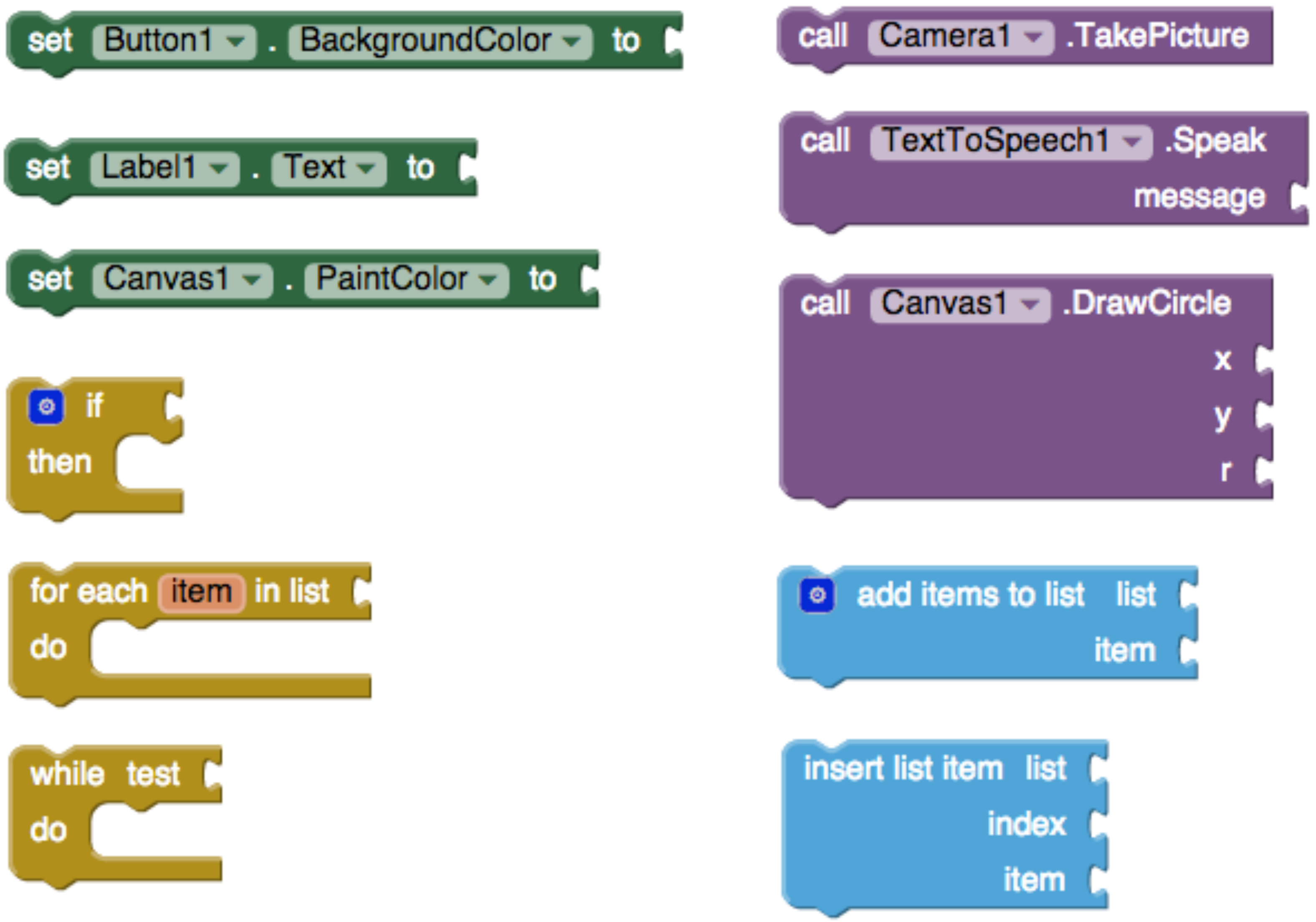
Talk Road Map

- MIT App Inventor (AI) demo
- Some notes on blocks languages in general
- **AI blocks language design and implementation**
  - **Syntax**
  - Static Semantics
  - Dynamic Semantics
  - Pragmatics
- Future work

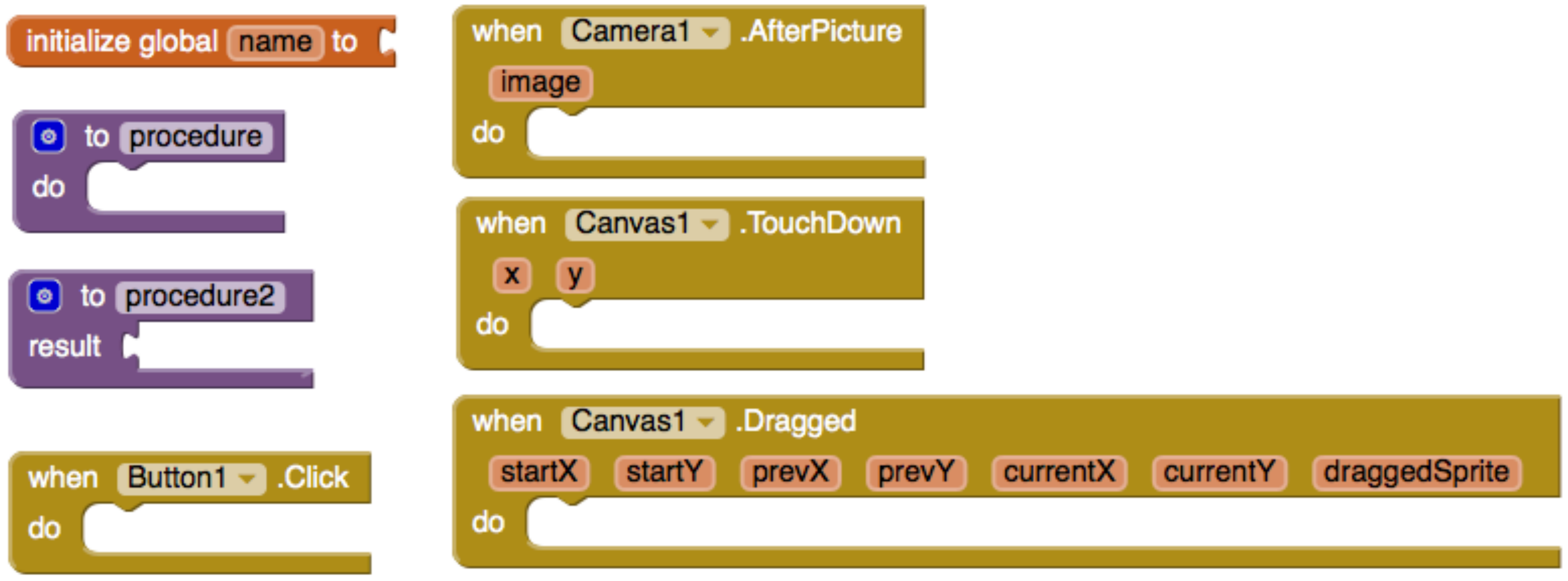
AI Blocks Syntax: Expressions



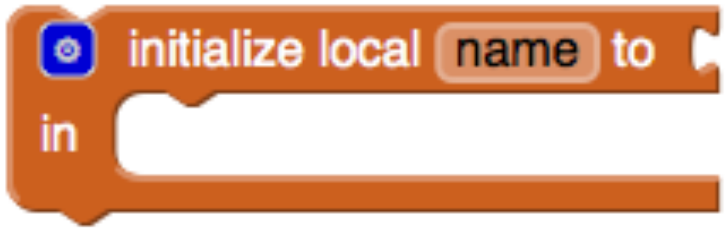
AI Blocks Syntax: Statements

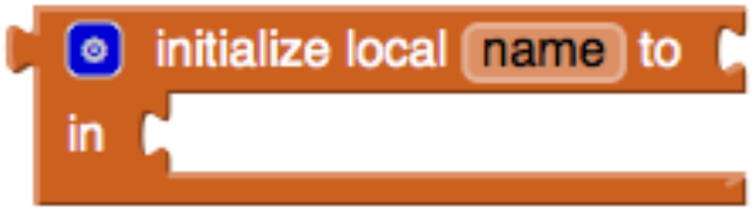


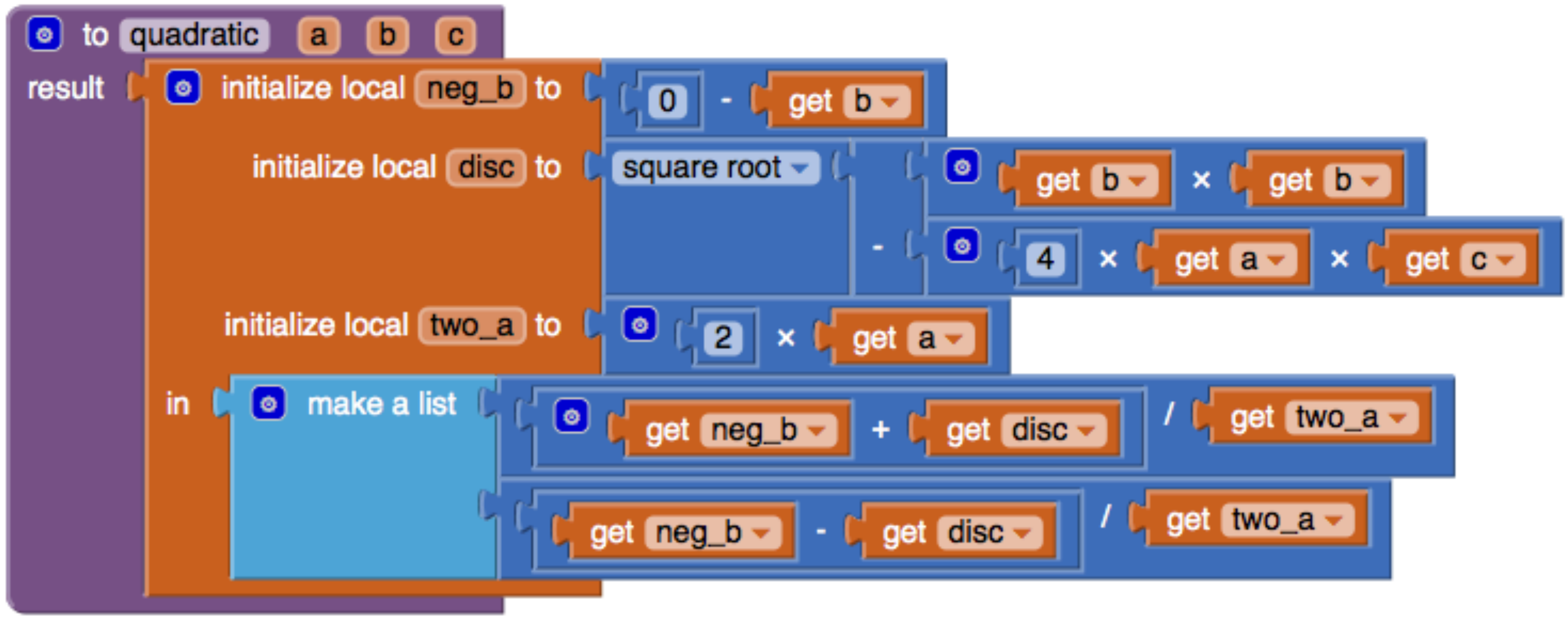
AI Blocks Syntax: Top Level Declarations



AI Blocks Syntax: Local Variable Declarations

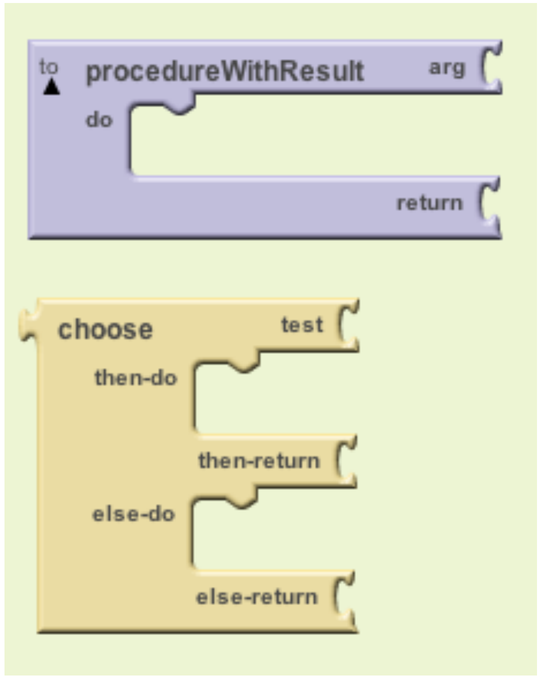




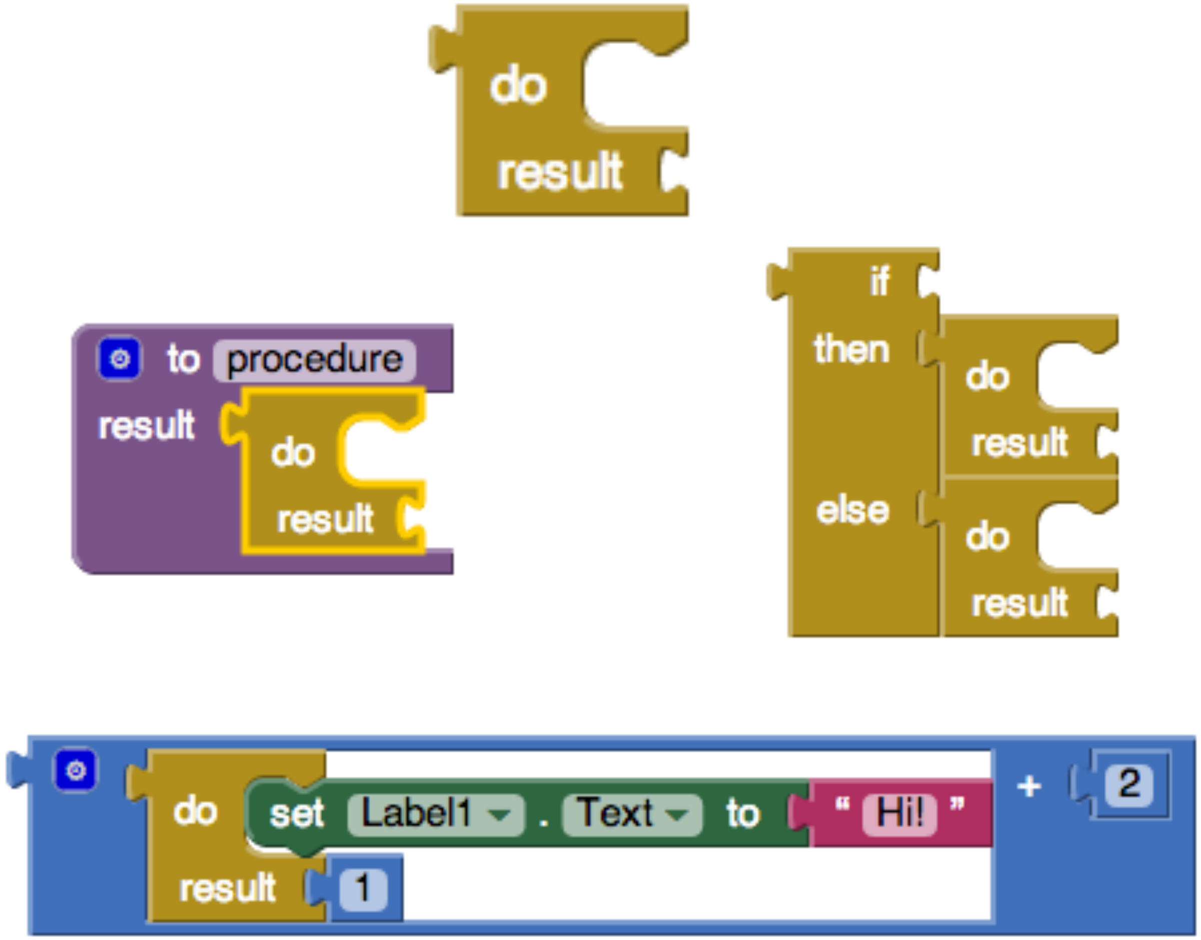


Performing actions before returning value

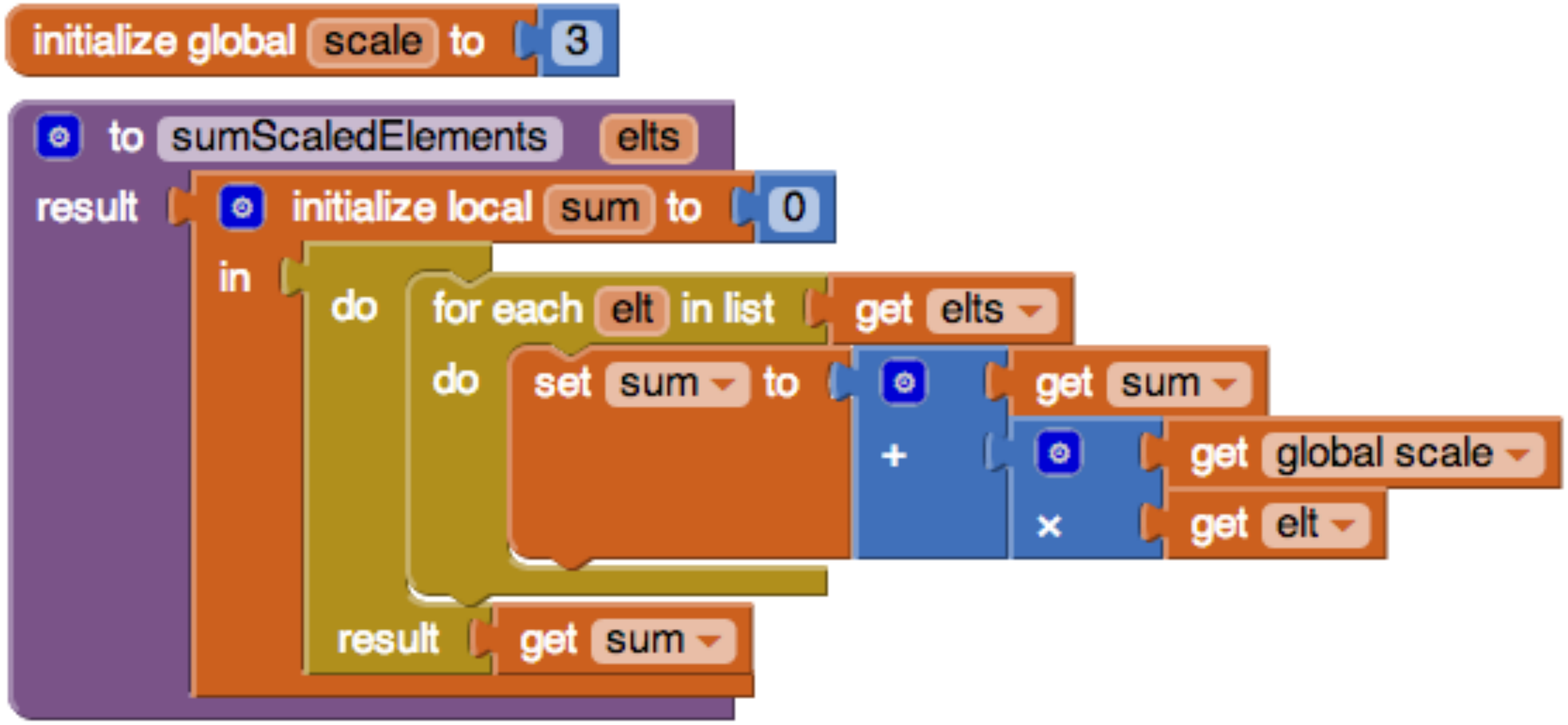
AI Classic



AI2

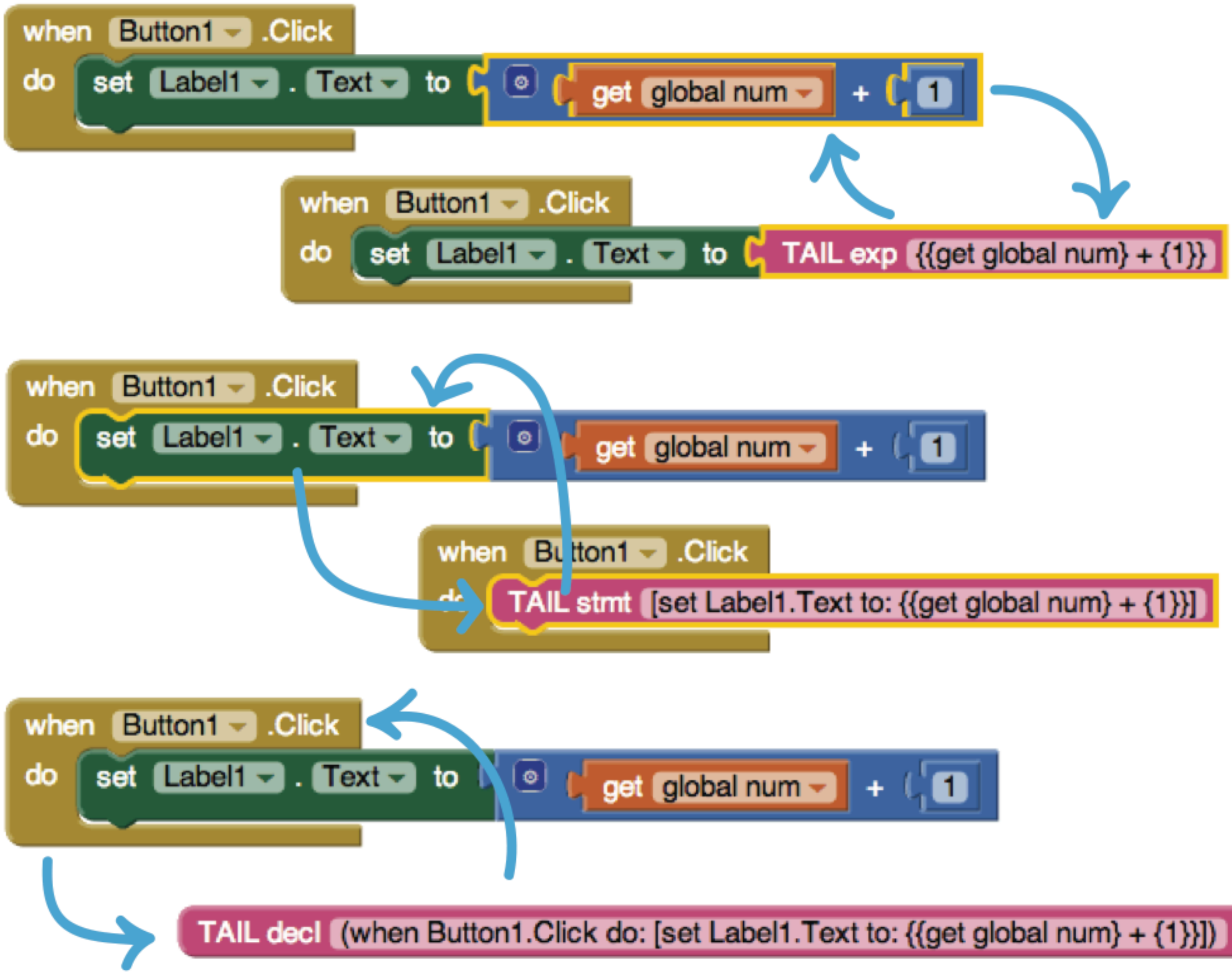


All together now



```
initialize global scale to 3

to sumScaledElements elts
result  initialize local sum to 0
        in  do  for each elt in list  get elts
                do  set sum to  +  get sum
                                   ×  get global scale
                                      get elt
            result  get sum
```

Conversion Between Blocks and Text


when Button1 .Click
do set Label1 . Text to get global num + 1
when Button1 .Click
do set Label1 . Text to TAIL exp {{get global num} + {1}}
when Button1 .Click
do set Label1 . Text to get global num + 1
when Button1 .Click
do TAIL stmt [set Label1.Text to: {{get global num} + {1}}]
when Button1 .Click
do set Label1 . Text to get global num + 1
TAIL decl (when Button1.Click do: [set Label1.Text to: {{get global num} + {1}}])

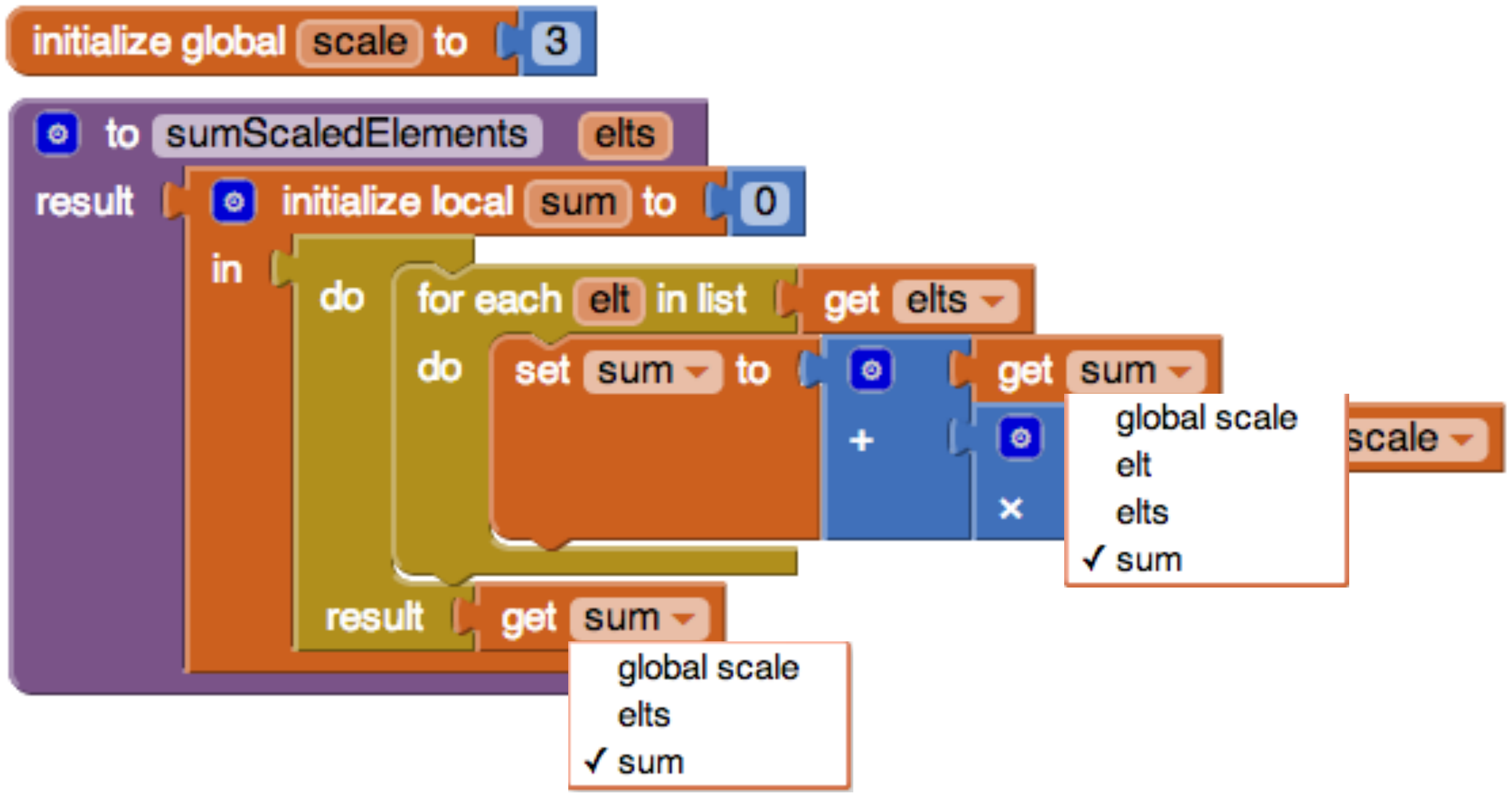
Karishma Chadha '14 Wellesley

Talk Road Map

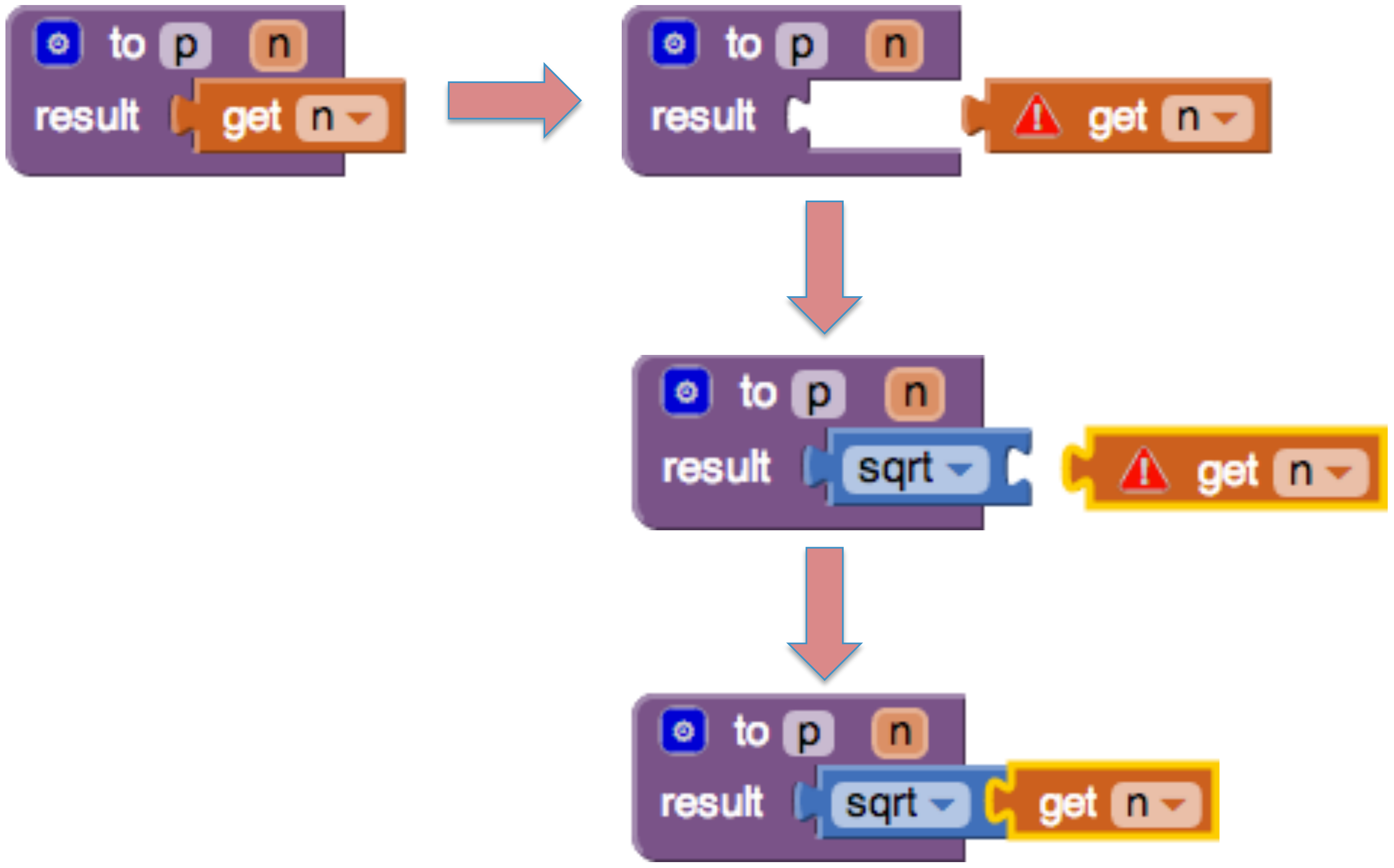
- MIT App Inventor (AI) demo
- Some notes on blocks languages in general
- **AI blocks language design and implementation**
  - Syntax
  - **Static Semantics**
  - Dynamic Semantics
  - Pragmatics
- Future work

Name Scoping in AI

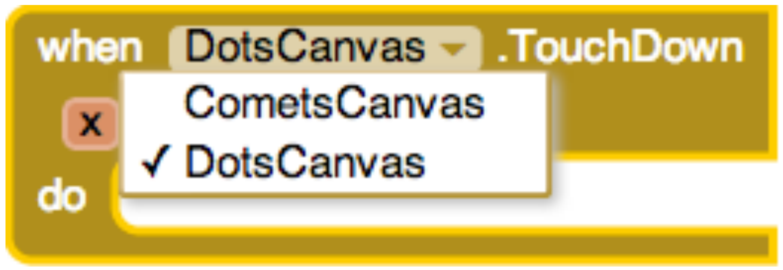
- Globals are in a separate namespace
- Indentation visually highlights area of name scope
- Drop-downs list only names in scope.
- Inner names can shadow outer ones
- Changing declared names automatically consistently changes all

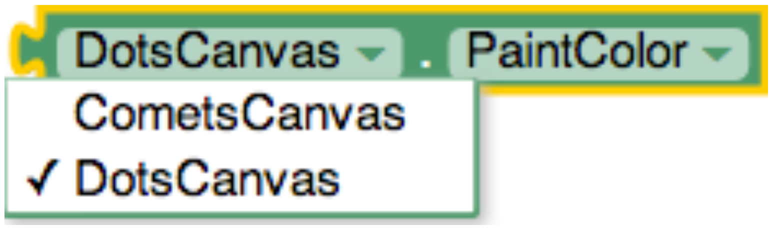


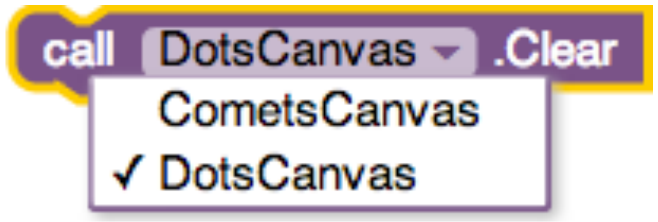
Handling Unbound Names

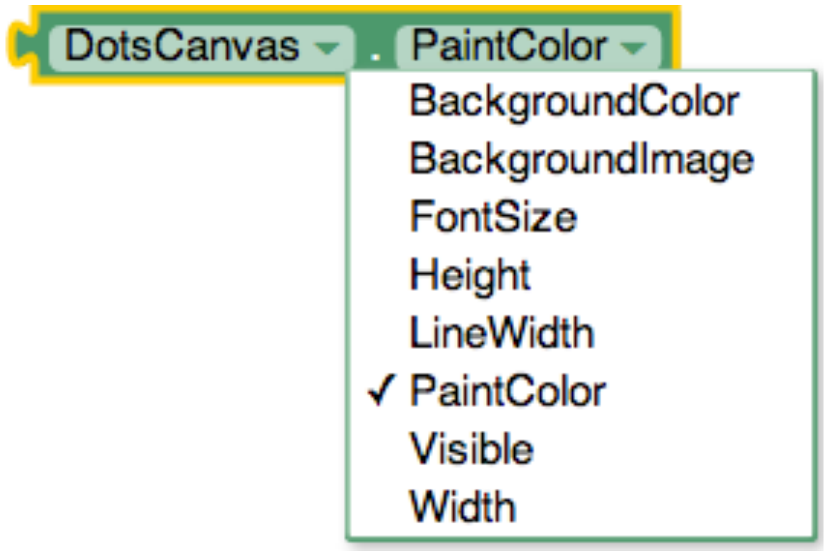


Other Drop-Downs Reduce Errors & Viscosity

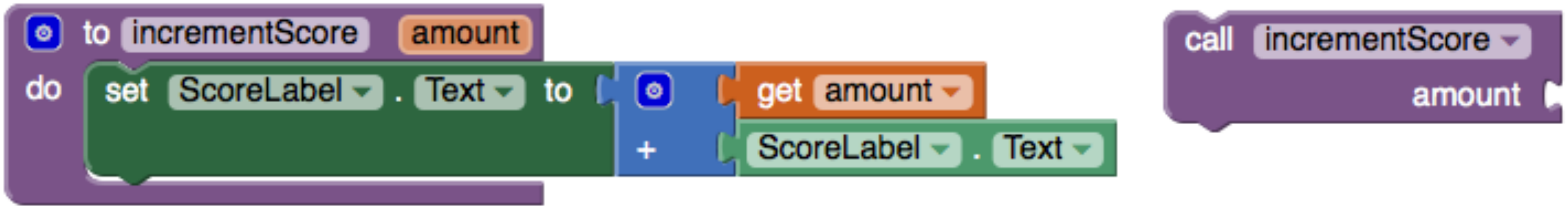


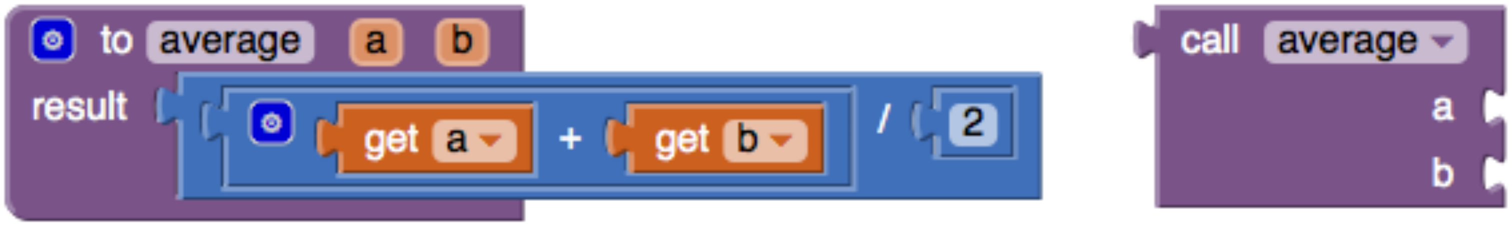






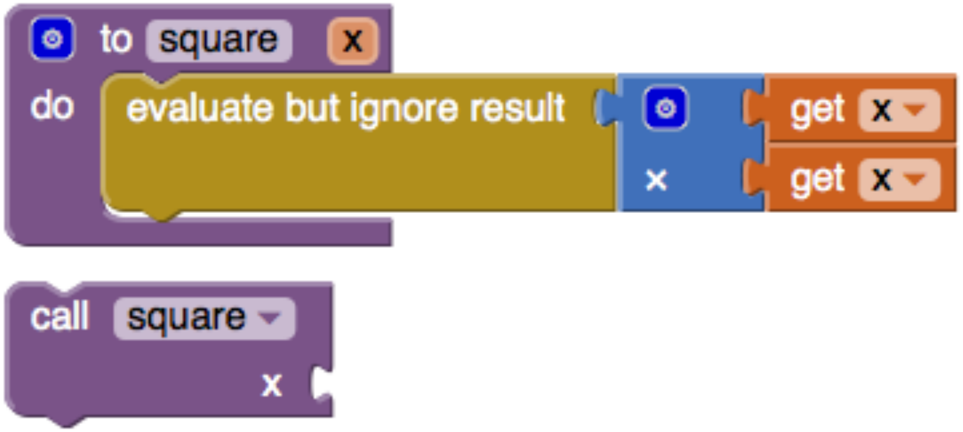
Distinguishing Void and Fruitful Procedures





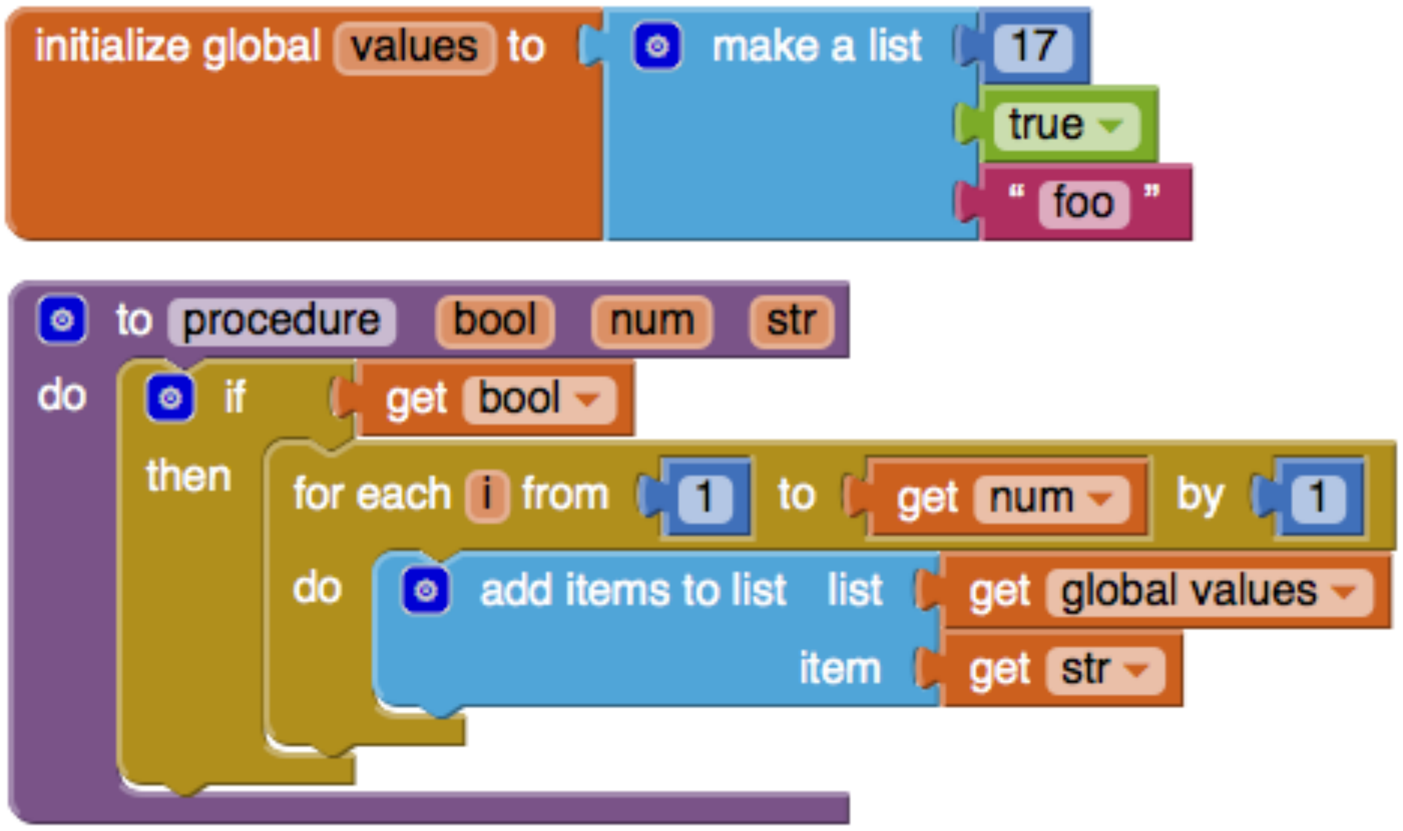
Python function gotcha

```
>>> def square (x):
...   x * x
...
>>> square(5)
>>>
```



What About Types?

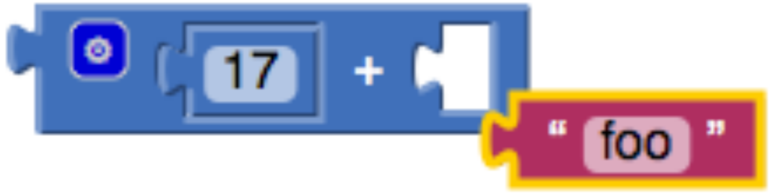
App Inventor is dynamically typed, so there's only one plug shape:



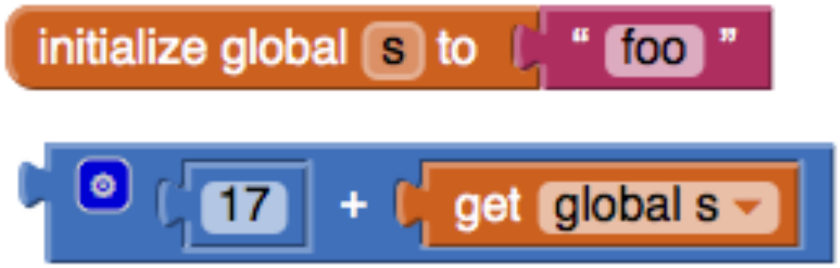
```
initialize global values to  make a list  17
                                          true
                                          " foo "

to procedure bool num str
do  if   get bool
    then for each i from 1 to get num by 1
         do  add items to list  list  get global values
                                item  get str
```

Simple "Soft" Static Type Checking

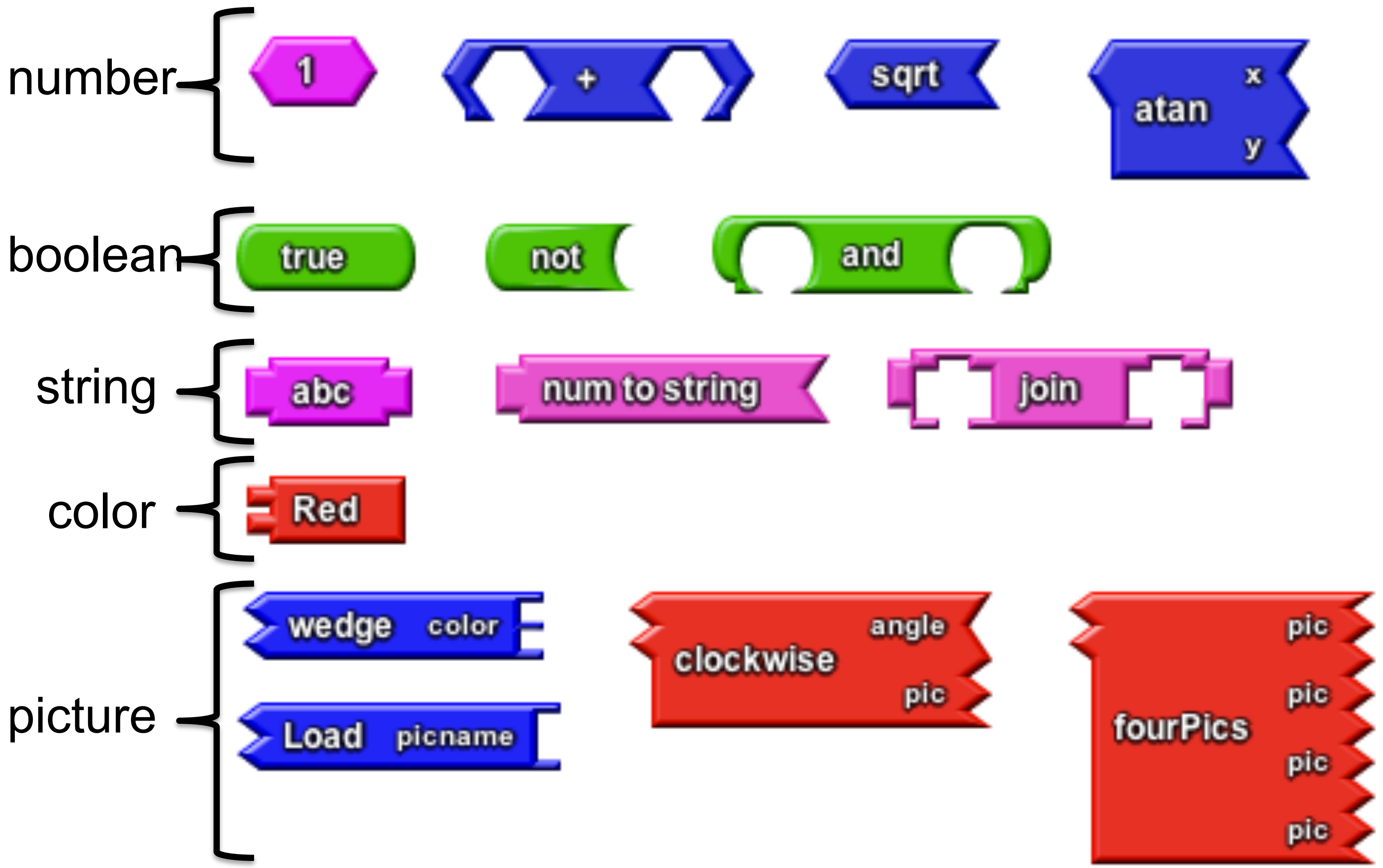
Type errors at block connection time are prohibited by "repulsion"



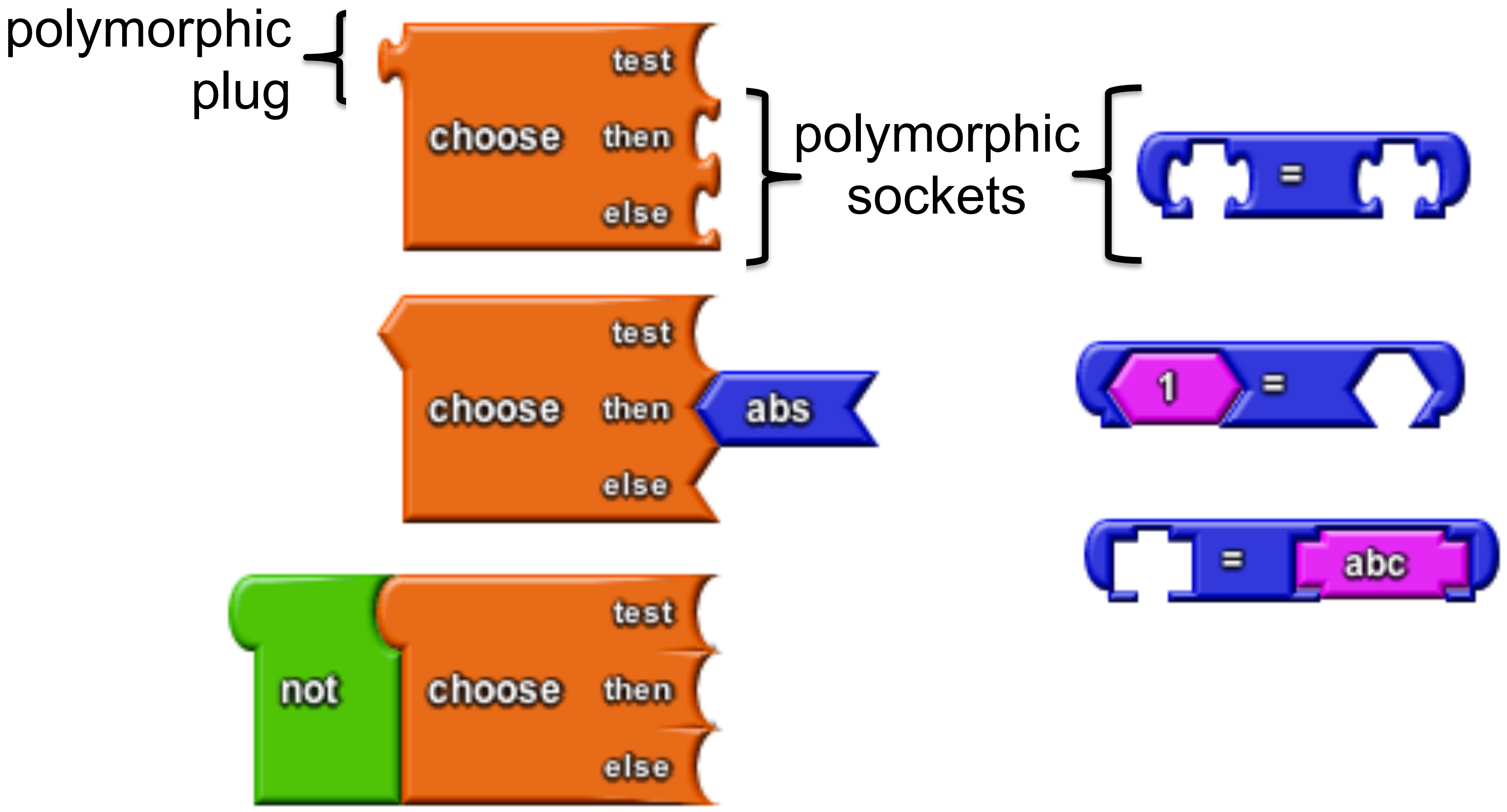
Dynamic type errors can be hidden by variables:



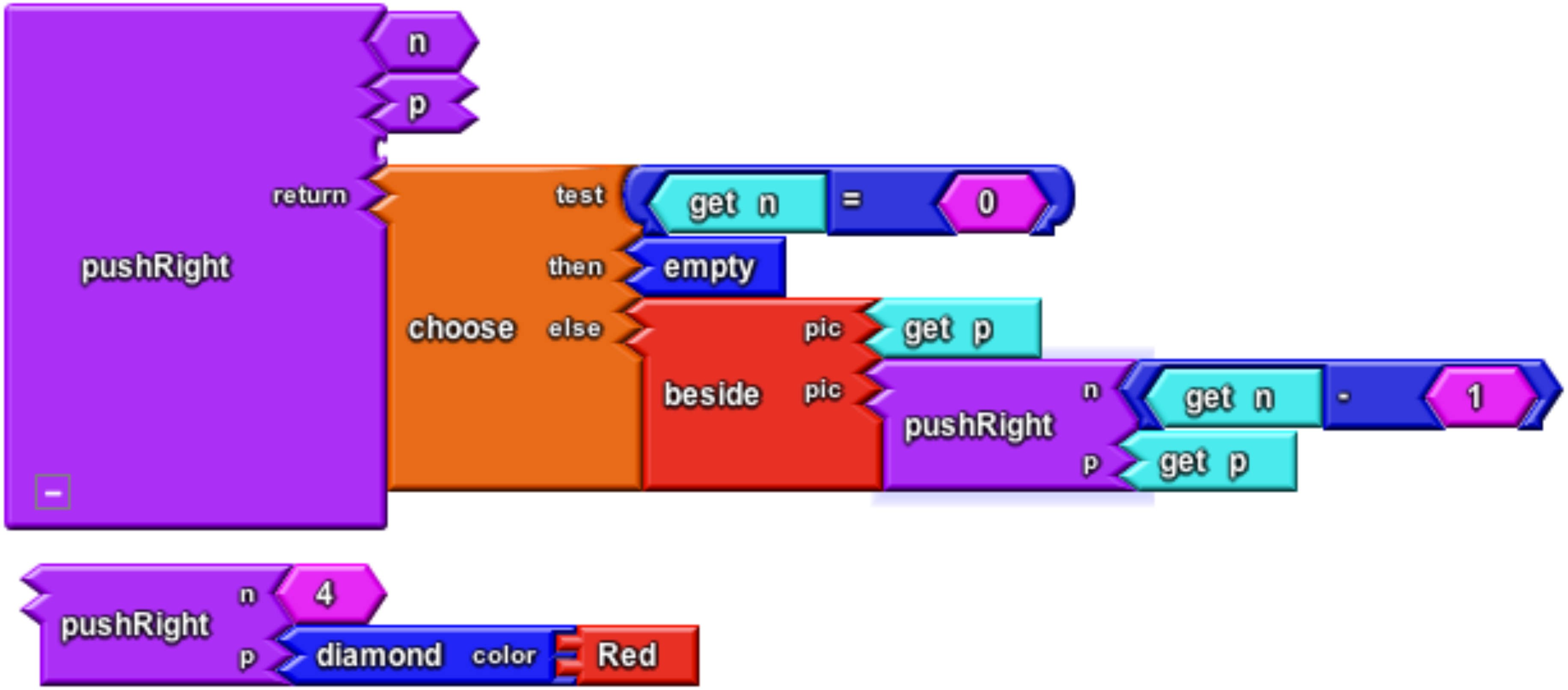
Digression: Connector Shapes in PictureBlocks



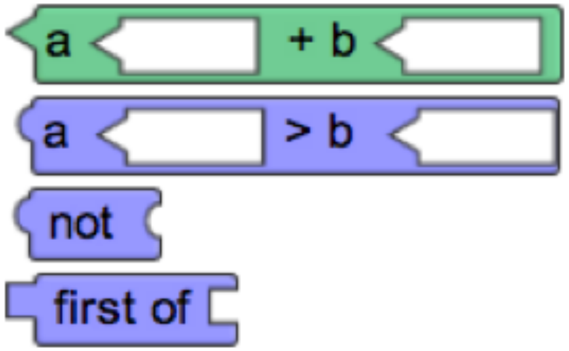
Polymorphism in PictureBlocks

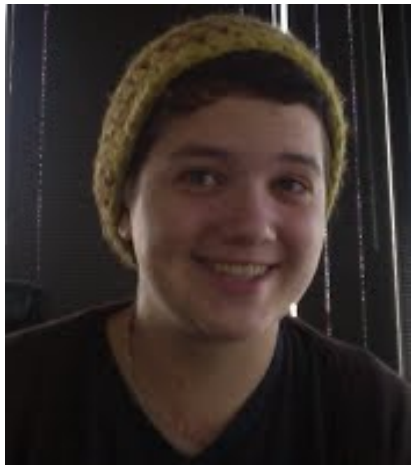


pushRight: Complete Declaration and Call

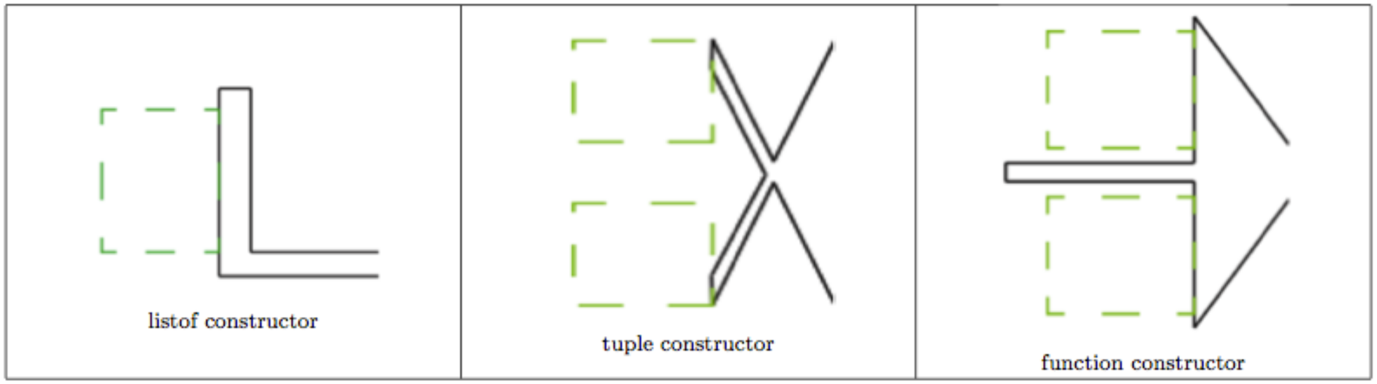


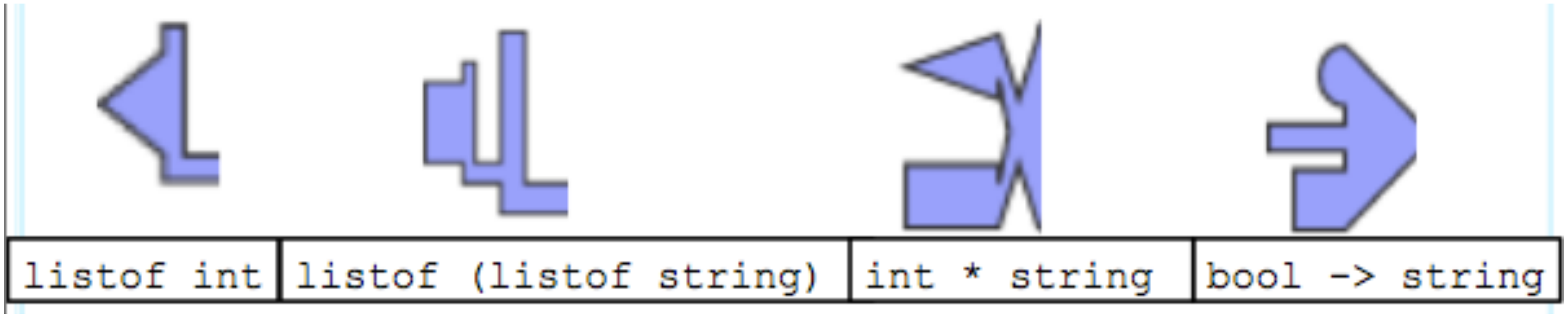
Continued Digression:Type Blocks



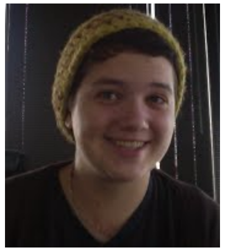


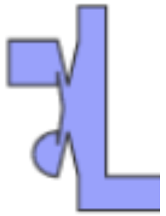
Marie Vasek '12
Wellesley



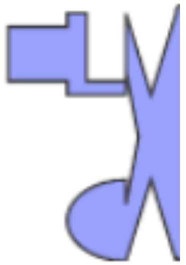


Type Blocks: More Examples

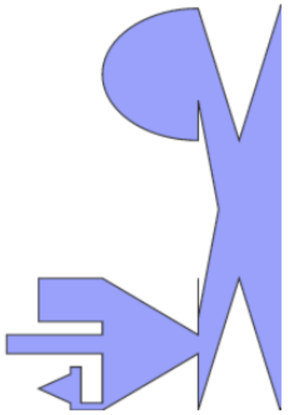




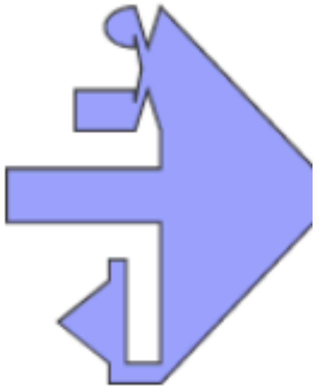
listof (string * boolean)



(listof string) * boolean

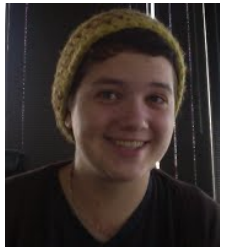


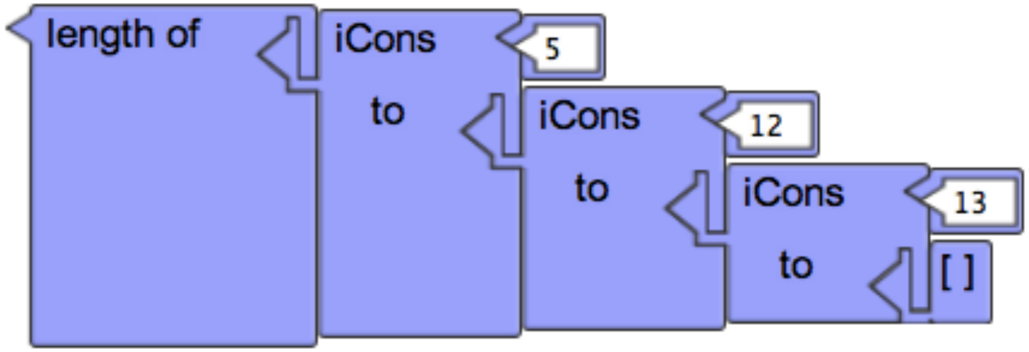
boolean * (string -> listof number)

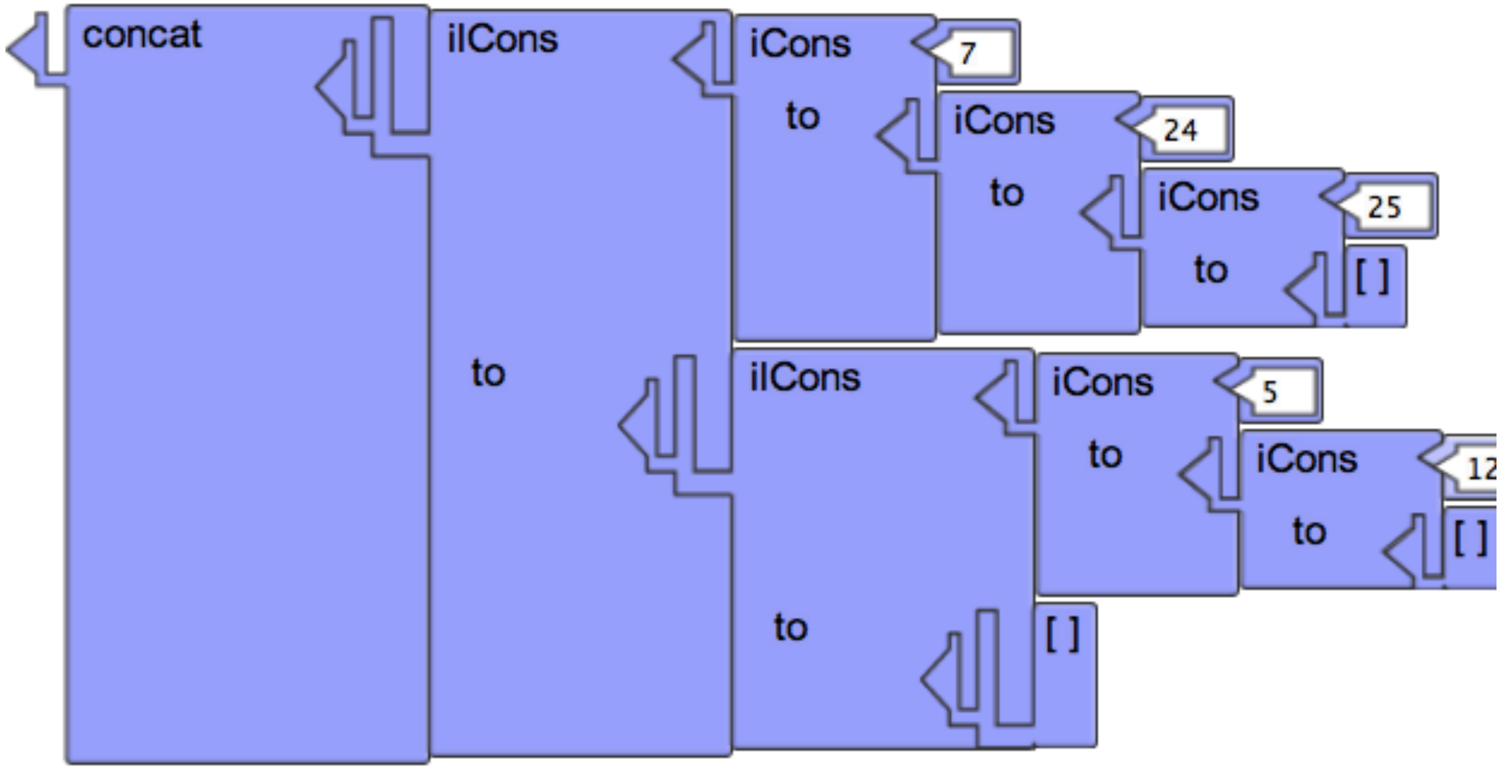


(boolean * string) -> (listof number)

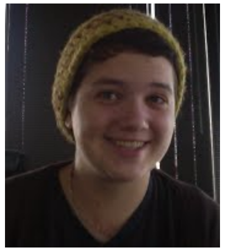
Type Blocks: Lists

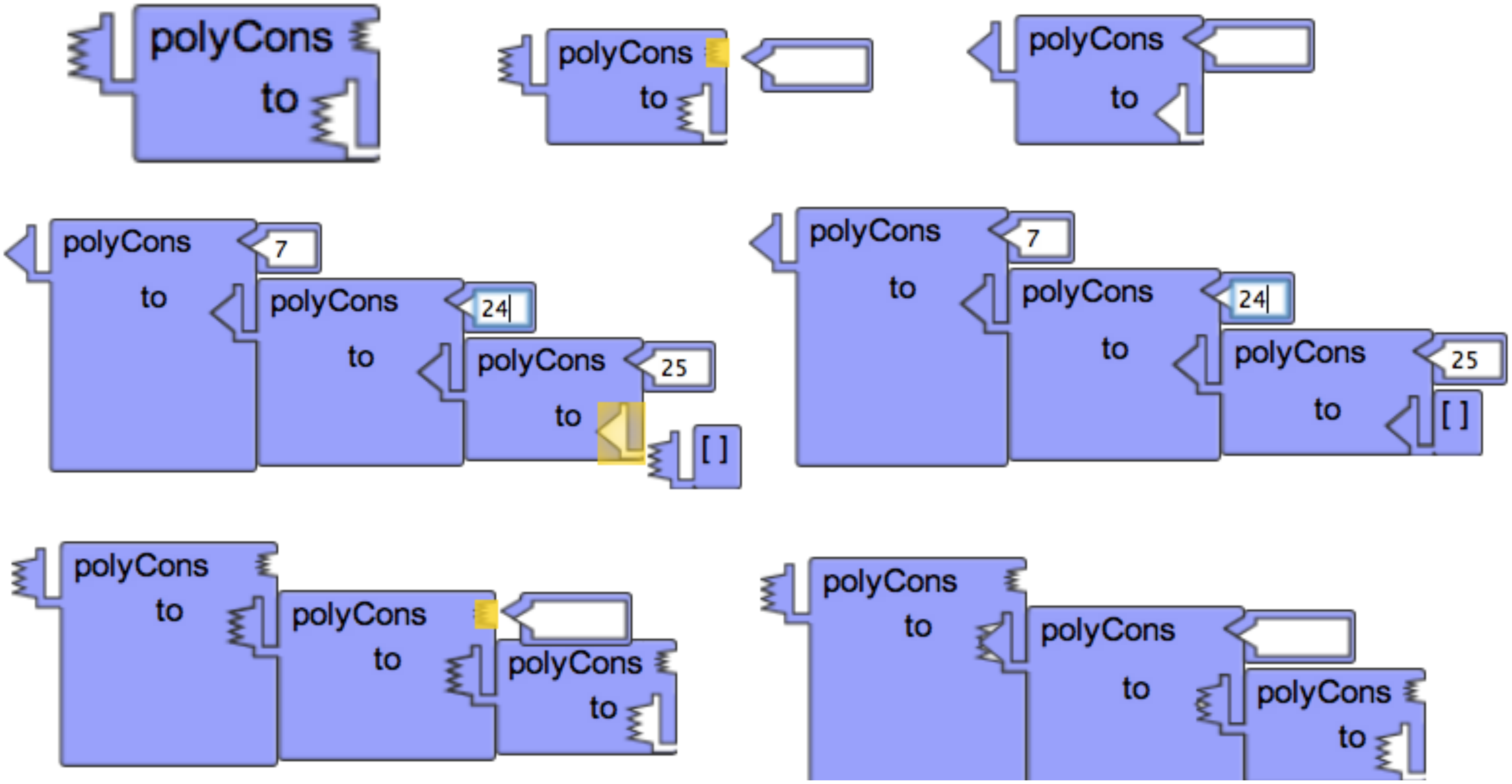






Type Blocks: ML Style Universal Polymorphism



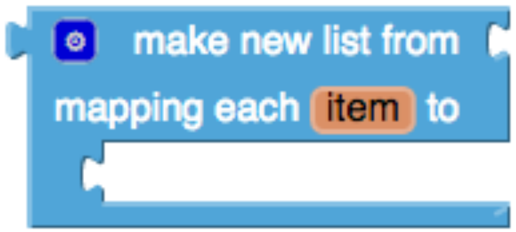


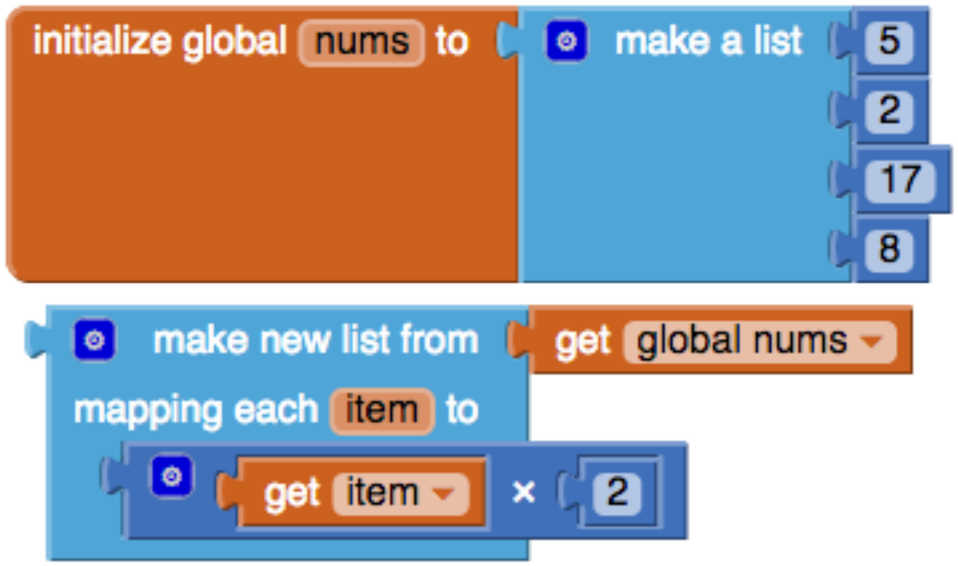
Back to AI: List Mapping

Python:

```
>>> nums = [5, 2, 17, 8]

>>> map(lambda x: x*2, nums)
[10, 4, 34, 16]
```

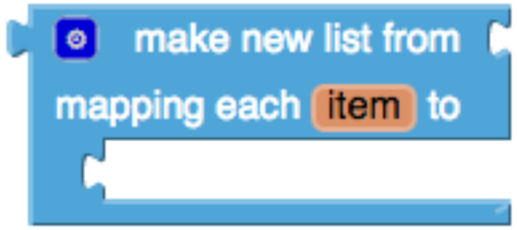
App Inventor doesn't have first-class functions, but can finesse mapping:

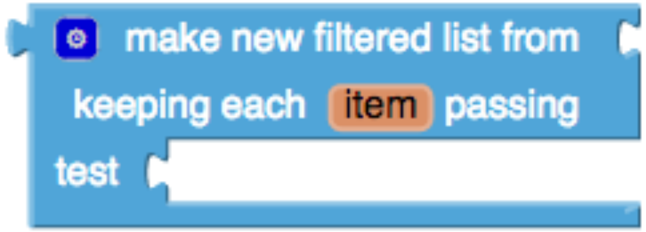


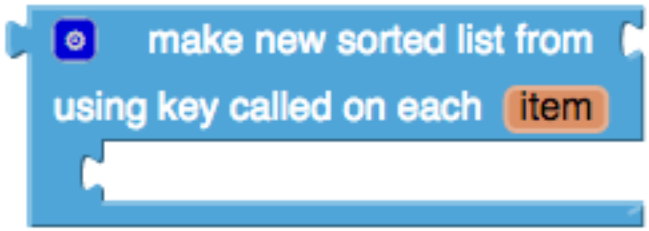


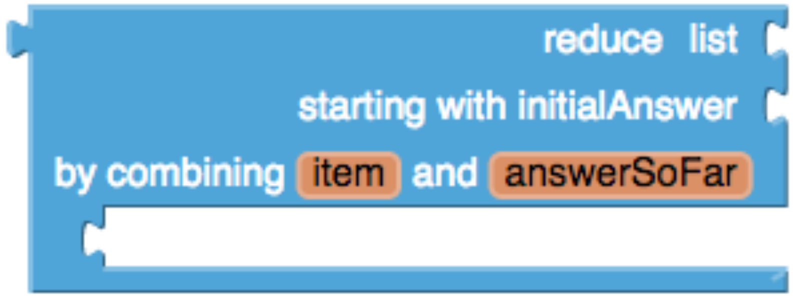
Experimental Higher-Order List Operators in AI

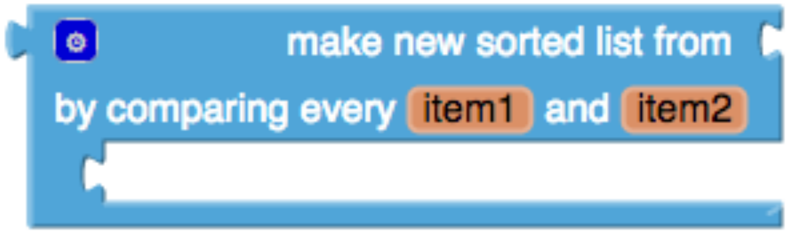
Soojin Kim '15
Wellesley



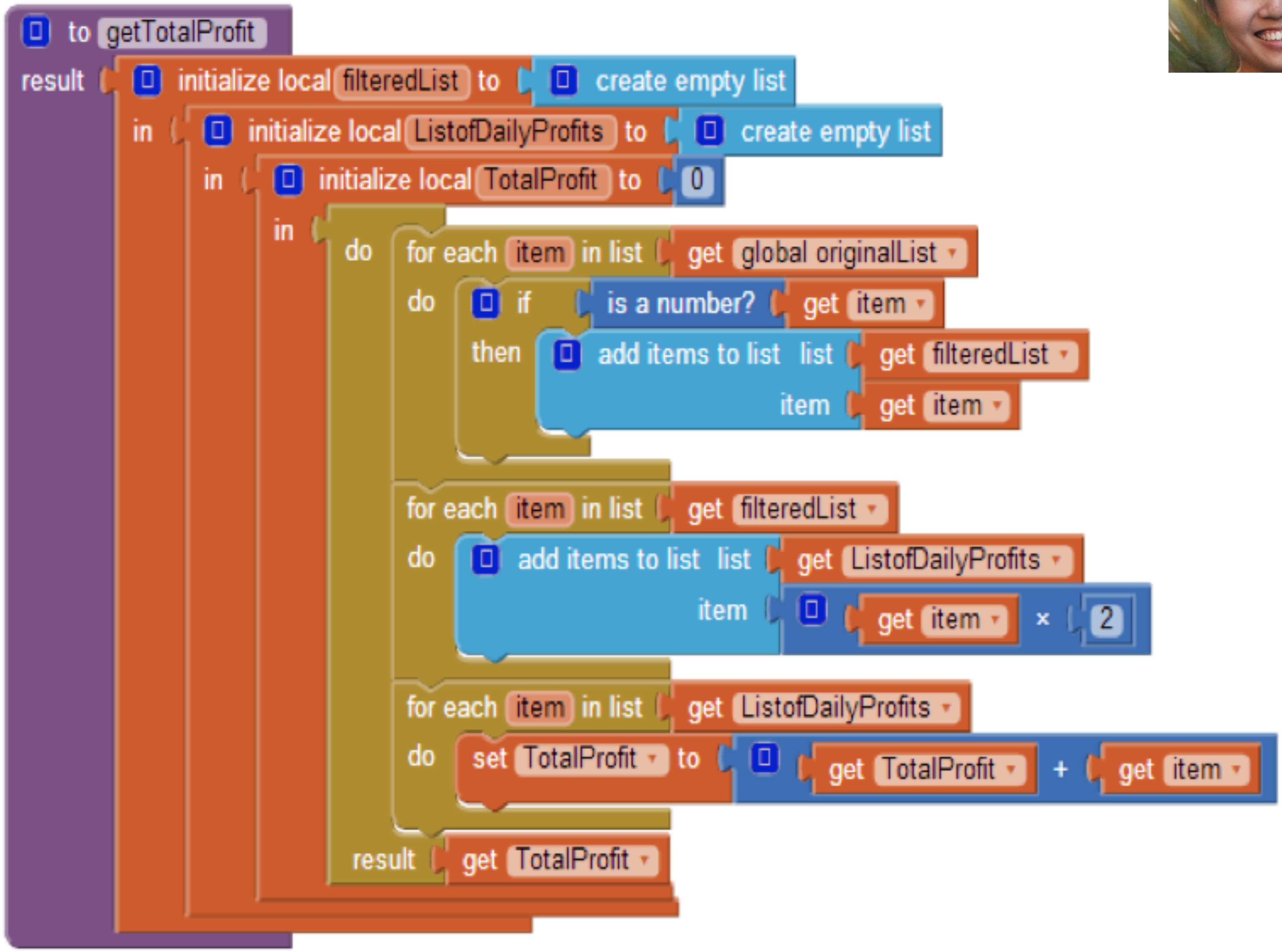





reduce list
starting with initialAnswer
by combining item and answerSoFar

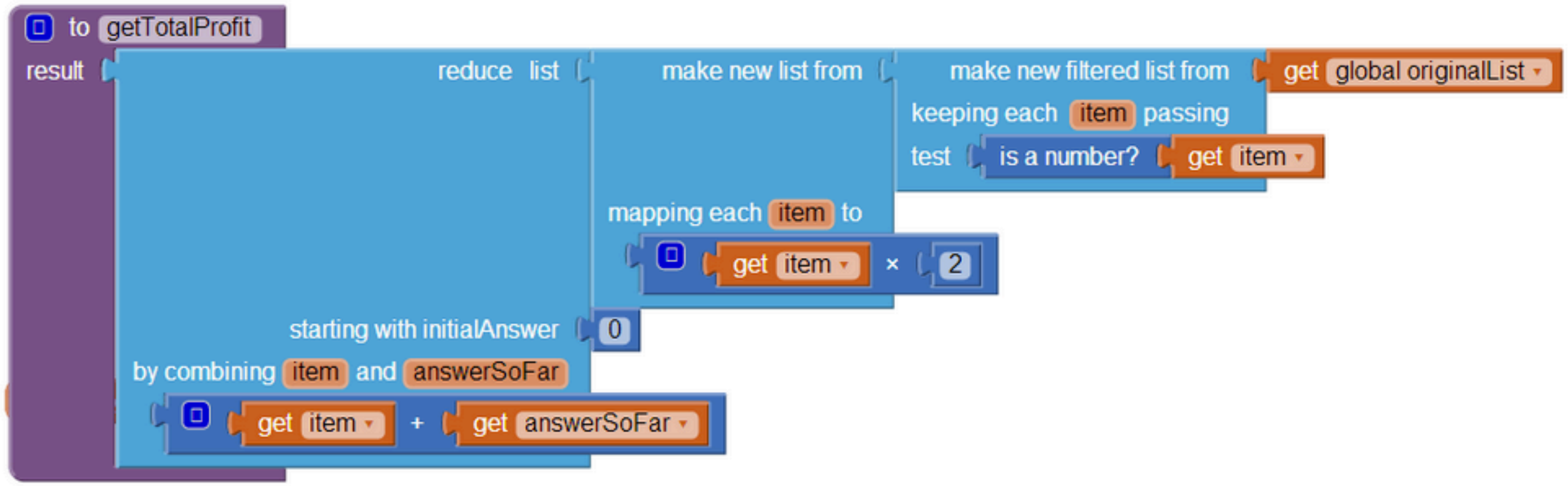


Loop-based List Processing



```
to getTotalProfit
result  initialize local filteredList to  create empty list
        in  initialize local ListofDailyProfits to  create empty list
            in  initialize local TotalProfit to  0
                in  do  for each item in list  get global originalList
                        do  if  is a number?  get item
                            then  add items to list  list  get filteredList
                                                     item  get item
                        for each item in list  get filteredList
                        do  add items to list  list  get ListofDailyProfits
                                               item  get item × 2
                        for each item in list  get ListofDailyProfits
                        do  set TotalProfit to  get TotalProfit + get item
                    result  get TotalProfit
```

List Processing With Higher-Order Operators

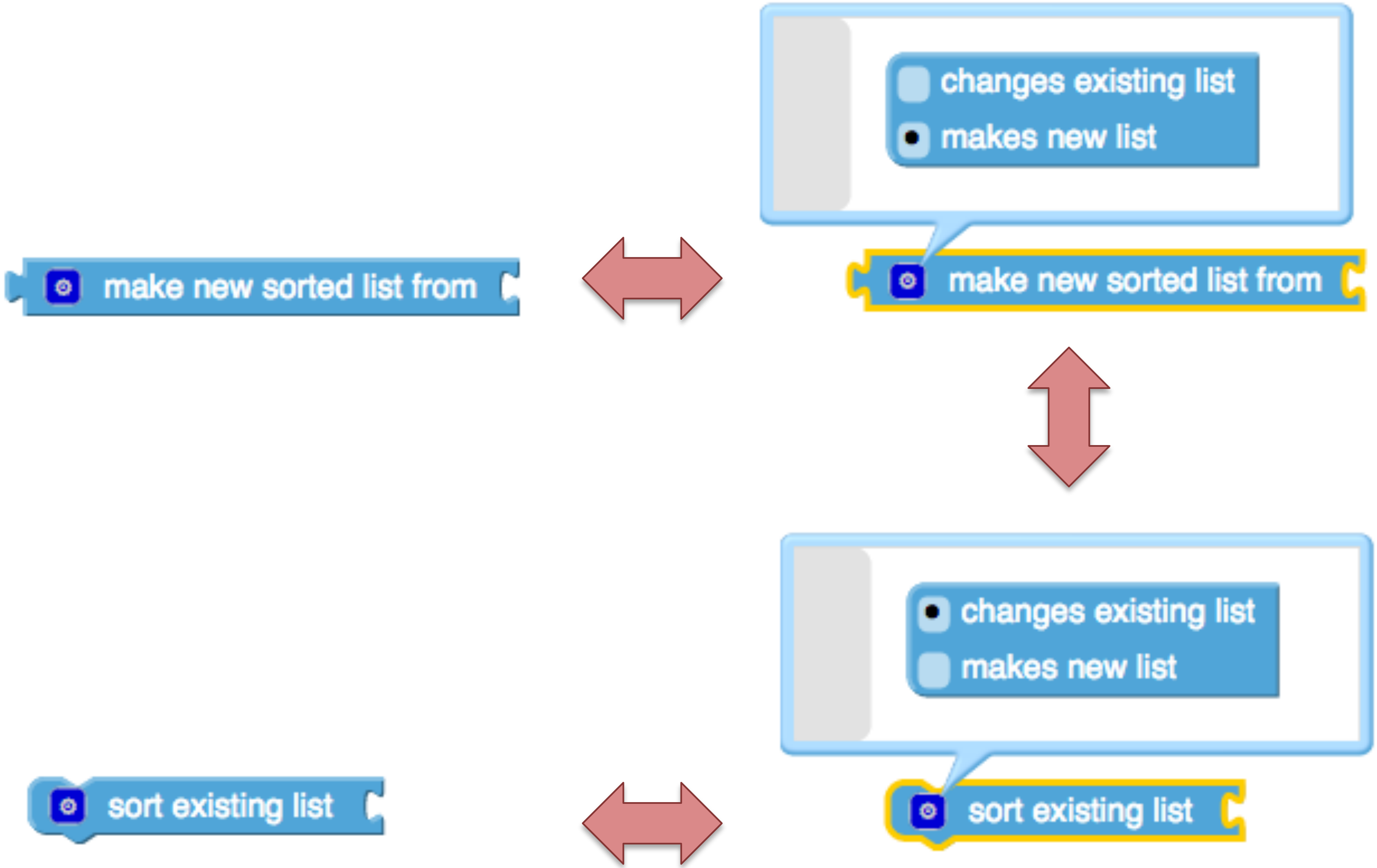


```
to getTotalProfit
result  reduce list  make new list from  make new filtered list from  get global originalList
                                         keeping each item passing
                                         test  is a number?  get item
                     mapping each item to
                        get item × 2
        starting with initialAnswer  0
        by combining item and answerSoFar
           get item + get answerSoFar
```

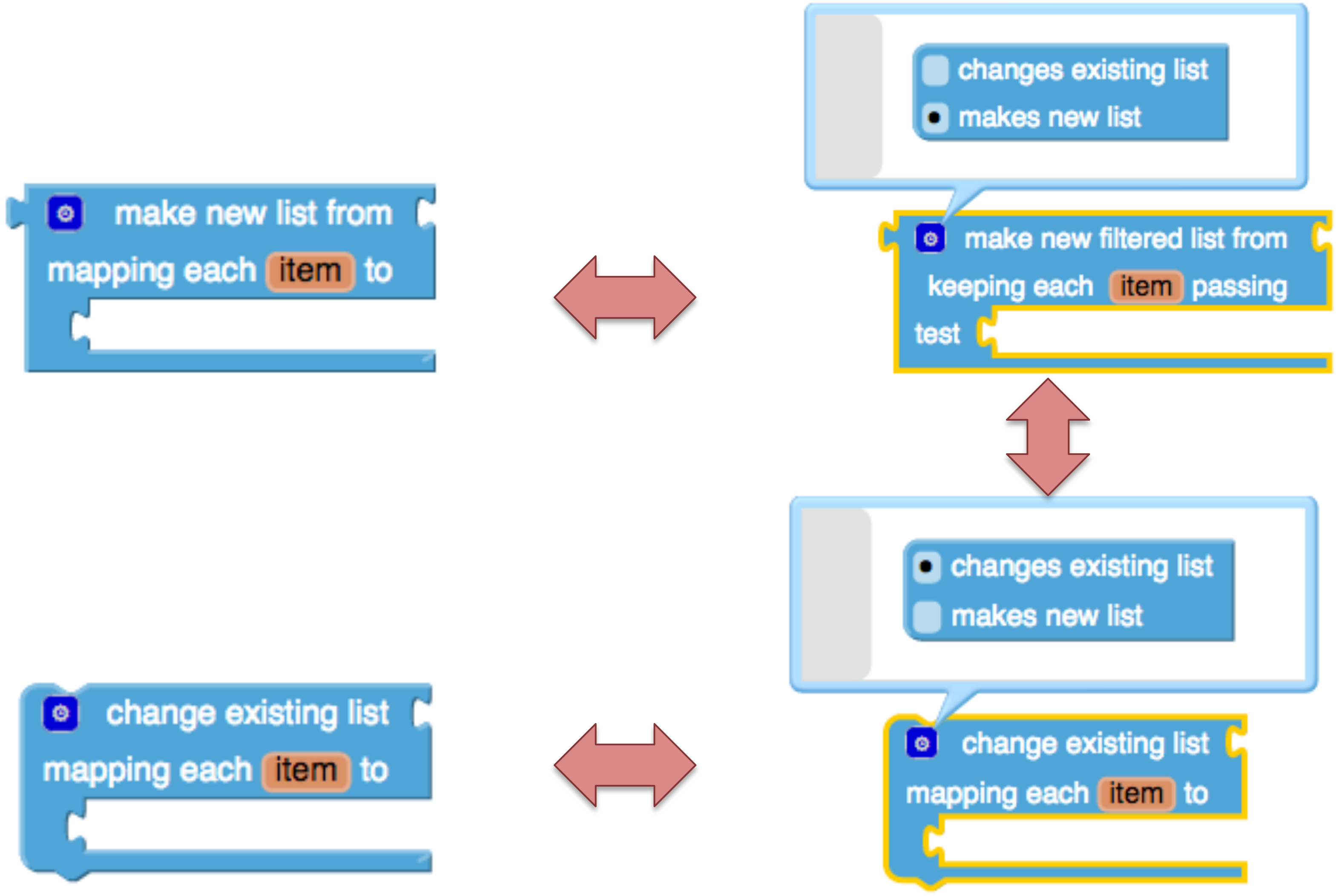
Nondestructive vs. Destructive List Ops In Python

```
>>> elts = [19, True, "foo", 23, "bar", 17, False]

>>> elts.sorted()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: 'list' object has no attribute
'sorted’

>>> sorted(elts)
[False, True, 17, 19, 23, 'bar', 'foo']

>>> elts
[19, True, 'foo', 23, 'bar', 17, False]

>>> elts.sort()

>>> elts
[False, True, 17, 19, 23, 'bar', 'foo']
```

Nondestructive vs. Destructive Sorting In AI



Other Nondestructive vs. Destructive List Ops In AI



make new list from
mapping each item to

changes existing list
[✓ makes new list]

make new filtered list from
keeping each item passing
test

[✓ changes existing list]
makes new list

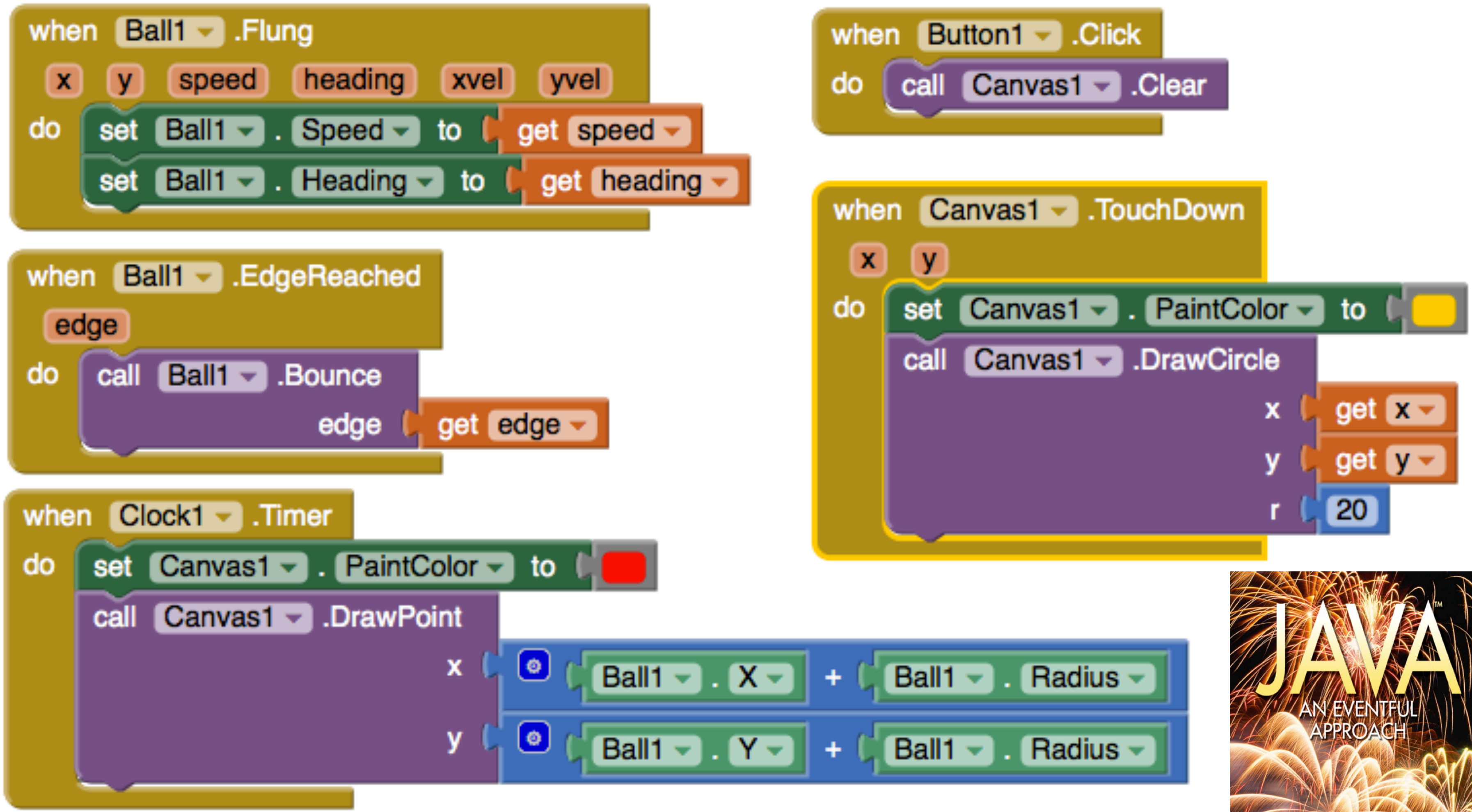
change existing list
mapping each item to

change existing list
mapping each item to

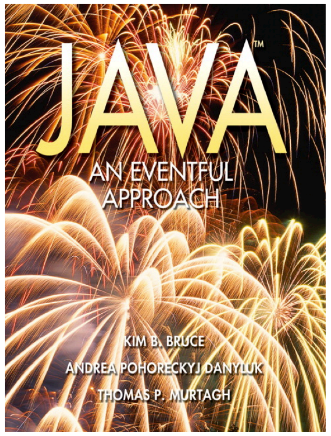
Talk Road Map

- MIT App Inventor (AI) demo
- Some notes on blocks languages in general
- **AI blocks language design and implementation**
  - Syntax
  - Static Semantics
  - **Dynamic Semantics**
  - Pragmatics
- Future work

The AI Event Model is Accessible for Simple Tasks …

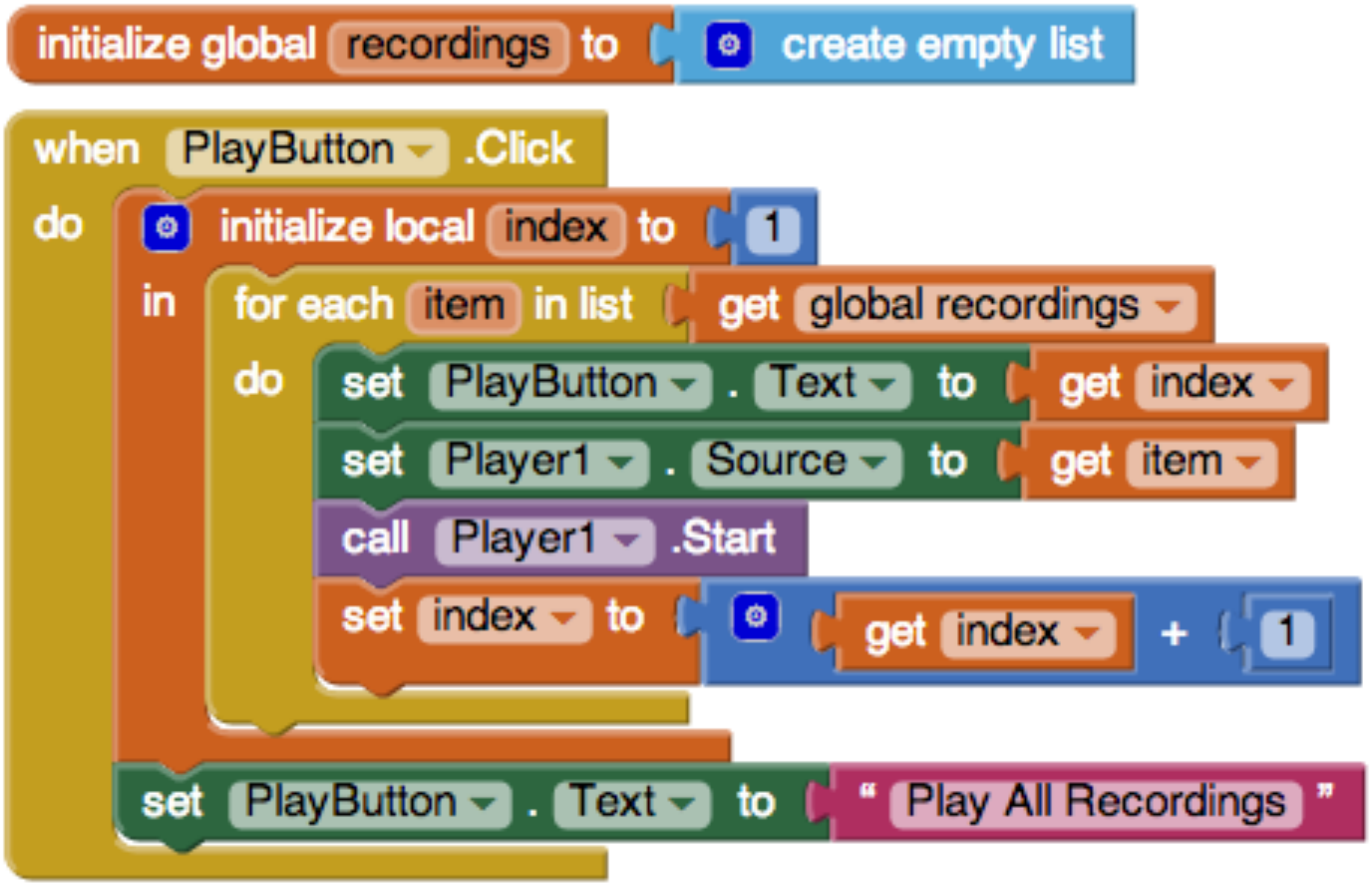


```
when Ball1 .Flung
  x  y  speed  heading  xvel  yvel
do  set Ball1 . Speed to  get speed
    set Ball1 . Heading to  get heading

when Ball1 .EdgeReached
  edge
do  call Ball1 .Bounce
            edge  get edge

when Clock1 .Timer
do  set Canvas1 . PaintColor to  🟥
    call Canvas1 .DrawPoint
            x   Ball1 . X  +  Ball1 . Radius
            y   Ball1 . Y  +  Ball1 . Radius

when Button1 .Click
do  call Canvas1 .Clear

when Canvas1 .TouchDown
  x  y
do  set Canvas1 . PaintColor to  🟨
    call Canvas1 .DrawCircle
            x  get x
            y  get y
            r  20
```



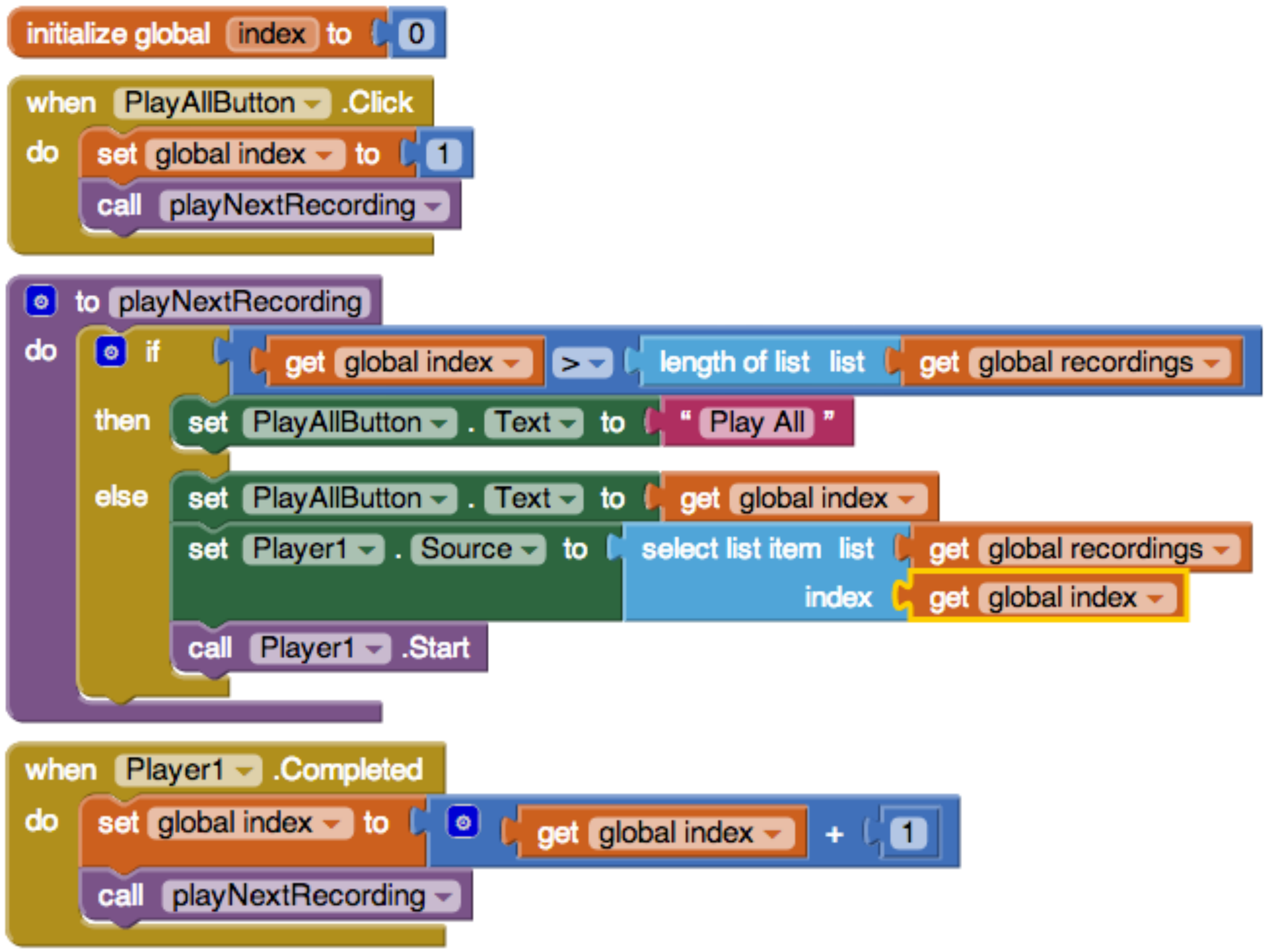
… but can be Confusing for more Complex Ones

This program for playing all recordings in a list does **not** work:



```
initialize global recordings to create empty list

when PlayButton .Click
do  initialize local index to 1
    in  for each item in list get global recordings
        do  set PlayButton . Text to get index
            set Player1 . Source to get item
            call Player1 .Start
            set index to get index + 1
    set PlayButton . Text to " Play All Recordings "
```

Correctly Playing all Recordings

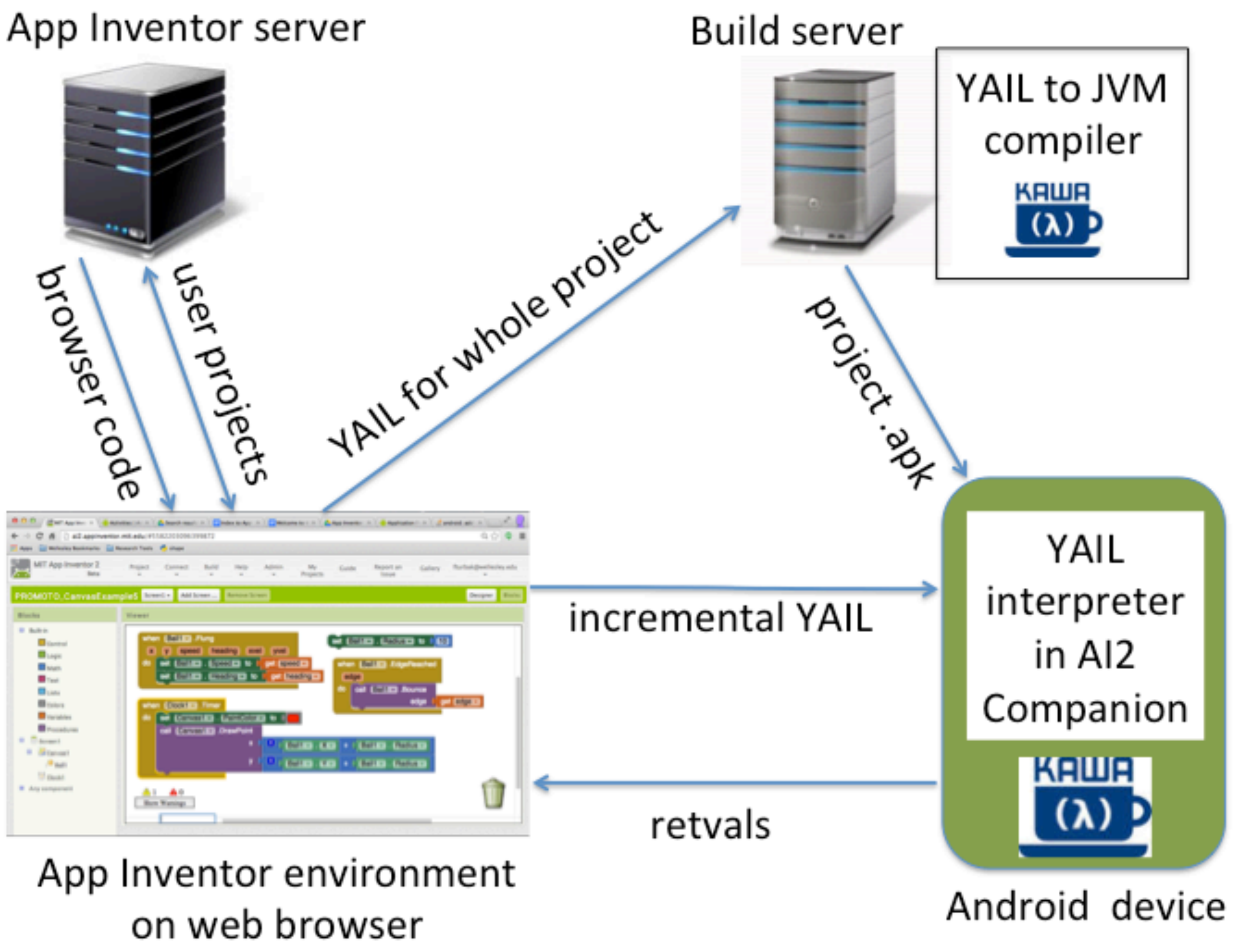


```
initialize global index to 0

when PlayAllButton.Click
do  set global index to 1
    call playNextRecording

to playNextRecording
do  if    get global index > length of list list get global recordings
    then  set PlayAllButton . Text to " Play All "
    else  set PlayAllButton . Text to get global index
          set Player1 . Source to select list item list get global recordings
                                                   index get global index
          call Player1 .Start

when Player1 .Completed
do  set global index to get global index + 1
    call playNextRecording
```

Talk Road Map

- MIT App Inventor (AI) demo
- Some notes on blocks languages in general
- **AI blocks language design and implementation**
  - Syntax
  - Static Semantics
  - Dynamic Semantics
  - **Pragmatics**
- Future work

AI Live Development Architecture

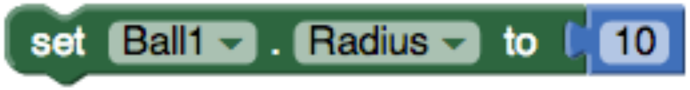


YAIL Example

```
;; Screen1
(do-after-form-creation
  (set-and-coerce-property! 'Screen1 'Title
                                        "Screen1" 'text))

;;; Canvas1
(add-component Screen1 Canvas Canvas1
  (set-and-coerce-property! 'Canvas1 'BackgroundColor
                                                #xFF00FFFF 'number)
  (set-and-coerce-property! 'Canvas1 'Width 200
'number)
  (set-and-coerce-property! 'Canvas1 'Height 300
'number))
```

```
;;; Ball1
(add-component Canvas1 Ball Ball1
  (set-and-coerce-property! 'Ball1 'X 46 'number)
  (set-and-coerce-property! 'Ball1 'Y 27 'number))

(define-event Ball1 Flung($x $y $speed $heading
$xvel $yvel)
  (set-this-form)
  (set-and-coerce-property! 'Ball1 'Speed
                                             (lexical-value
$speed)
                                              'number)
  (set-and-coerce-property! 'Ball1 'Heading
                                               (lexical-value
$heading)
                                               'number))
```

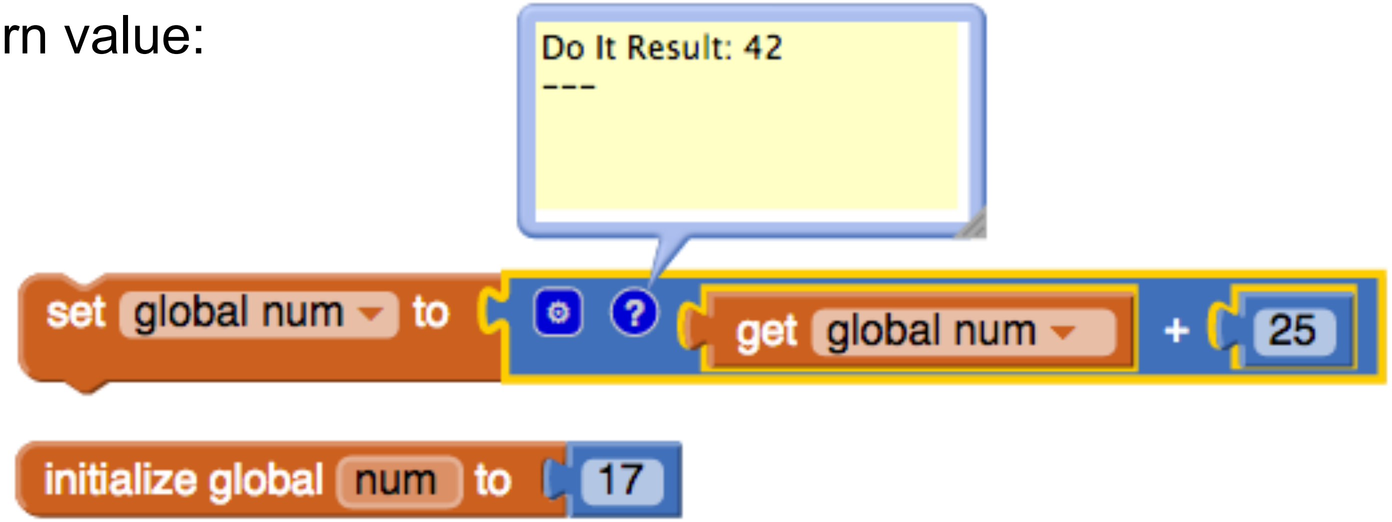
DoIt Examples



YAIL sent to companion:

```
(process-repl-input 186
   (set-and-coerce-property! 'Ball1 'Radius 10 'number))
```

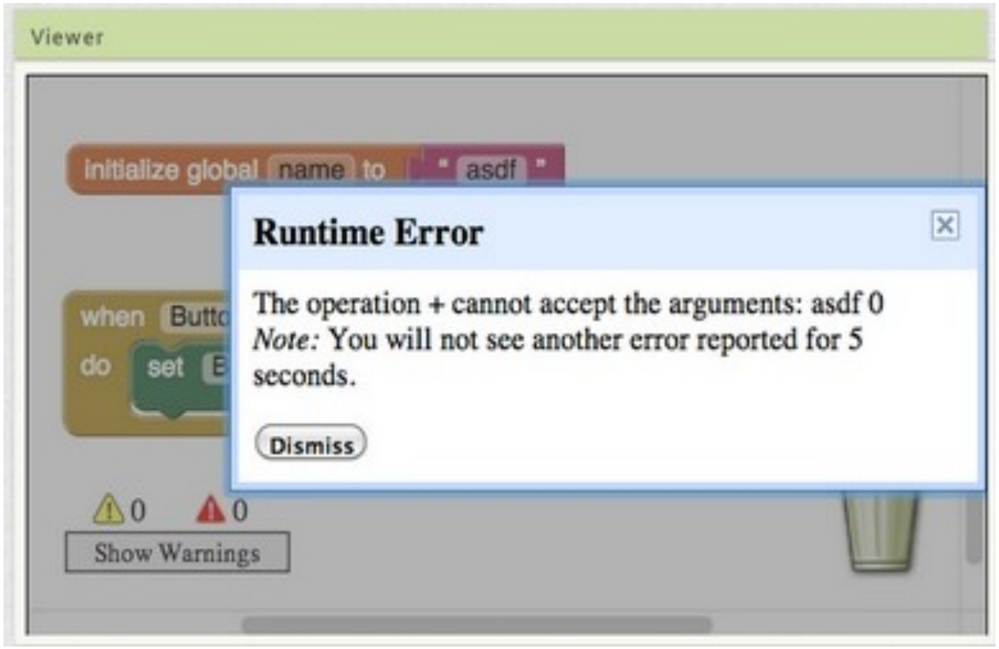
DoIt with return value:


set global num to get global num + 25
initialize global num to 17

```
Do It Result: 42
---
```

Better Error Handling

Currently, AI2 error window covers blocks and does not pinpoint block causing error:



Viewer

initialize global name to " asdf "

when Butto…

do set B…

Runtime Error

The operation + cannot accept the arguments: asdf 0
Note: You will not see another error reported for 5 seconds.

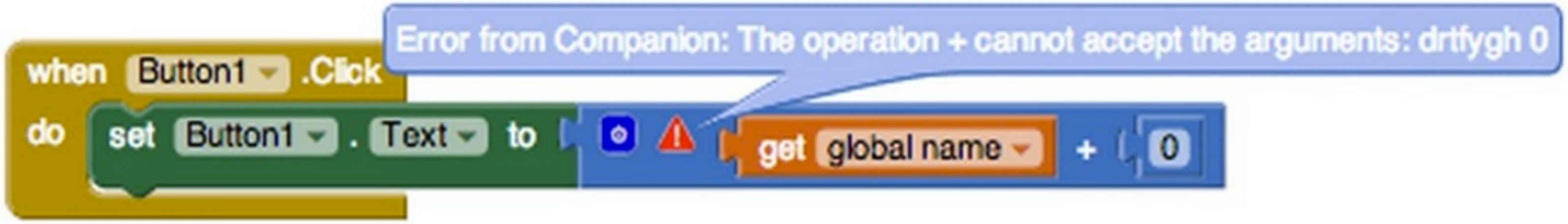
Dismiss

⚠ 0 ⚠ 0

Show Warnings

Johanna Okerlund '14 Wellesley

Soon, the error will appear on the block causing the error:

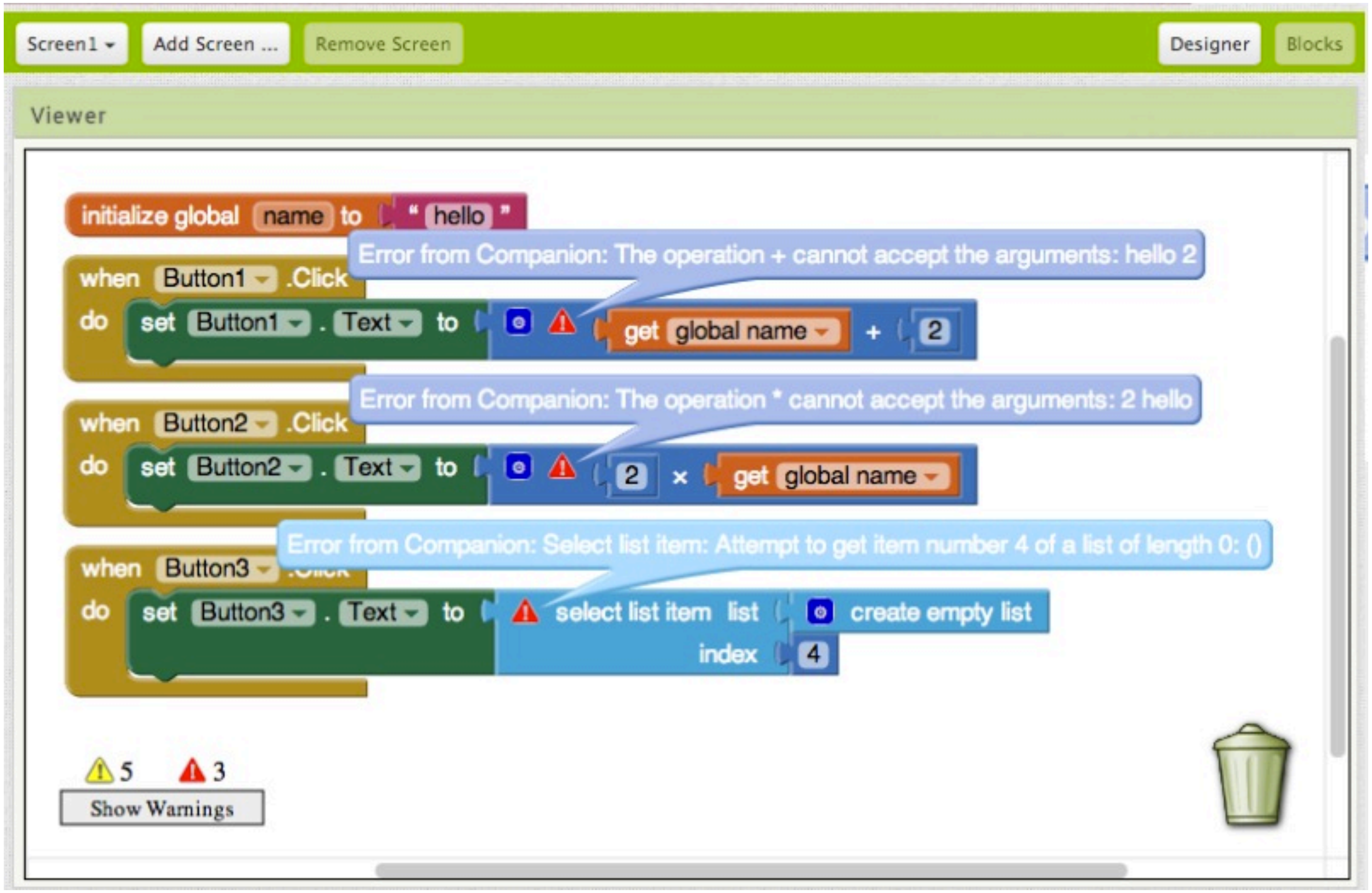


Error from Companion: The operation + cannot accept the arguments: drtfygh 0

when Button1 .Click
do set Button1 . Text to get global name + 0

Better Error Handling

Error messages can appear on multiple blocks until the errors are fixed:



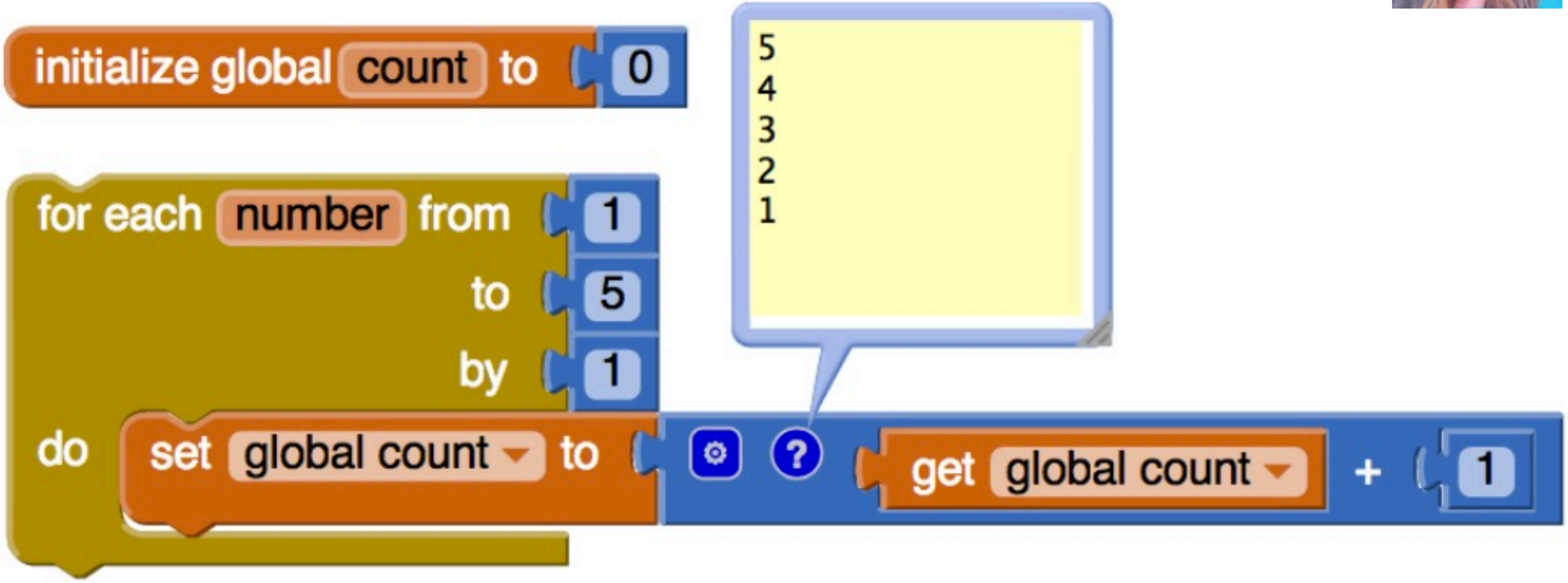
```
initialize global name to " hello "

when Button1 .Click
do  set Button1 . Text to  get global name + 2

when Button2 .Click
do  set Button2 . Text to  2 × get global name

when Button3 .Click
do  set Button3 . Text to  select list item list  create empty list
                                            index 4
```

Error from Companion: The operation + cannot accept the arguments: hello 2

Error from Companion: The operation * cannot accept the arguments: 2 hello

Error from Companion: Select list item: Attempt to get item number 4 of a list of length 0: ()

Better Debugging: Watch



```
initialize global count to 0

for each number from 1
                   to 5
                   by 1
do  set global count to  get global count + 1
```

```
5
4
3
2
1
```

Talk Road Map

- MIT App Inventor (AI) demo
- Some notes on blocks languages in general
- AI blocks language design and implementation
  - Syntax
  - Static Semantics
  - Dynamic Semantics
  - Pragmatics
- **Future work**

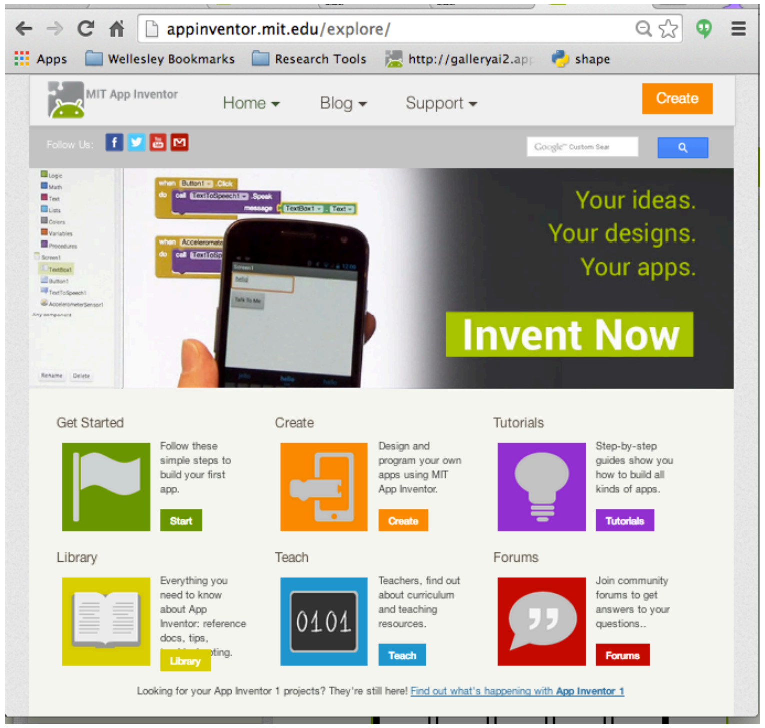
Future Work

- More flexible event handling
- Non-local returns
- More faithful live development
- User studies: what works and what doesn't?
- Blocks, text, and in-between
- Component development kit
- Blocks-based version of ML or Haskell

Want to join me? Email fturbak@wellesley.edu

Thank You! Questions?

appinventor.mit.edu



Get Started

Follow these simple steps to build your first app.

Start

Create

Design and program your own apps using MIT App Inventor.

Create

Tutorials

Step-by-step guides show you how to build all kinds of apps.

Tutorials

Library

Everything you need to know about App Inventor: reference docs, tips, ...ting.

Library

Teach

Teachers, find out about curriculum and teaching resources.

Teach

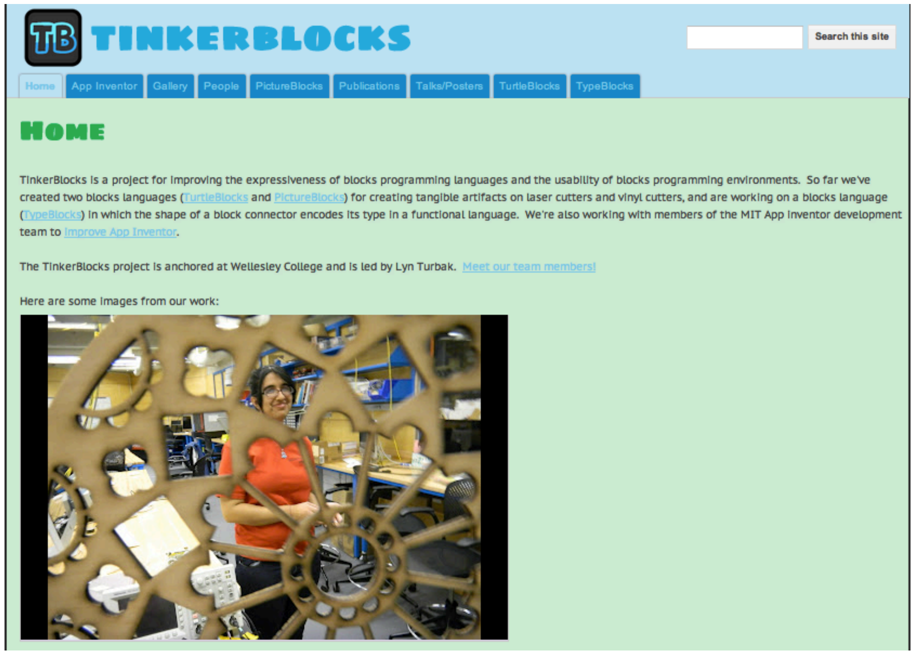
Forums

Join community forums to get answers to your questions..

Forums

Looking for your App Inventor 1 projects? They're still here! Find out what's happening with App Inventor 1

www.tinkerblocks.org



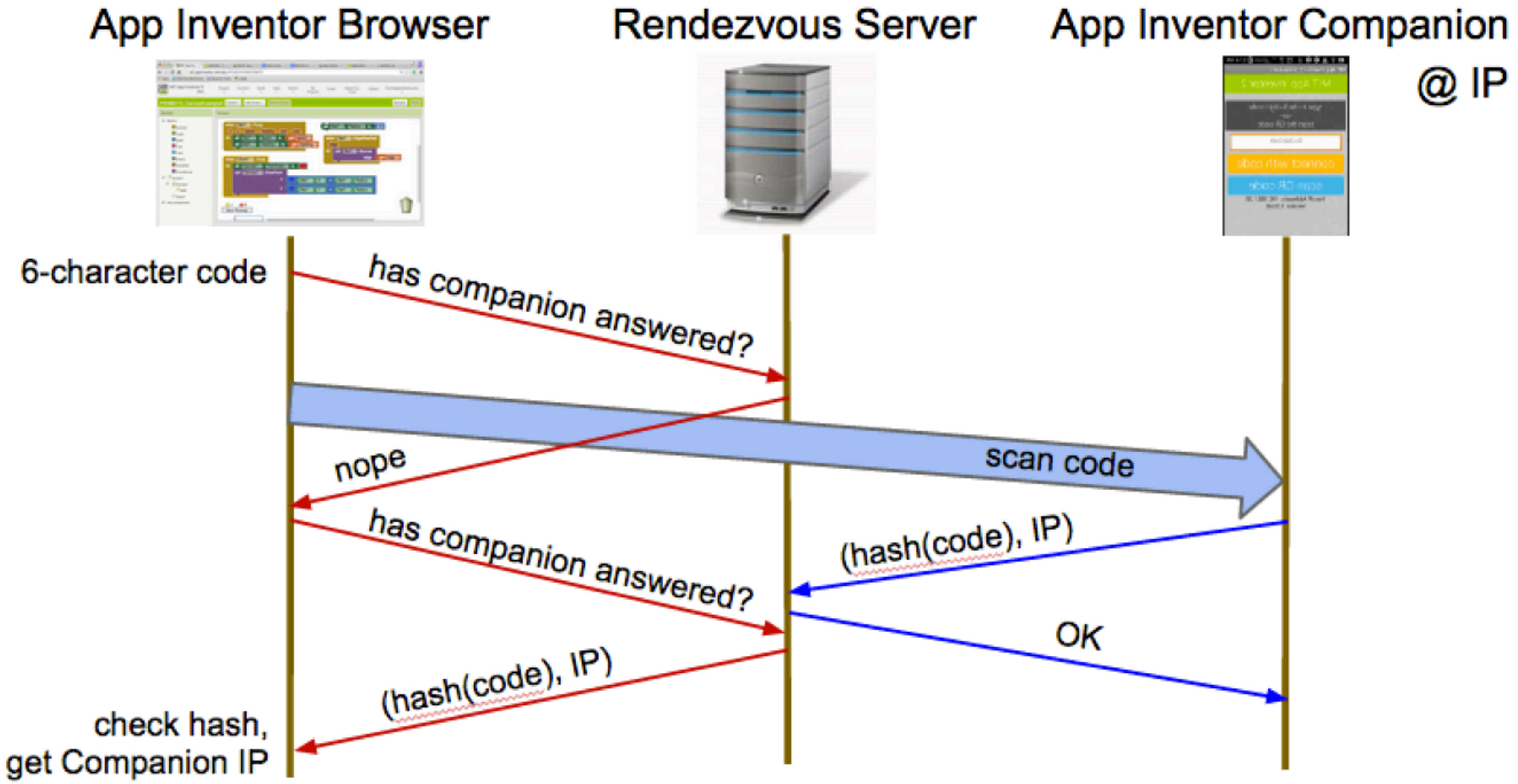
Home

TinkerBlocks is a project for improving the expressiveness of blocks programming languages and the usability of blocks programming environments. So far we've created two blocks languages (TurtleBlocks and PictureBlocks) for creating tangible artifacts on laser cutters and vinyl cutters, and are working on a blocks language (TypeBlocks) in which the shape of a block connector encodes its type in a functional language. We're also working with members of the MIT App Inventor development team to improve App Inventor.

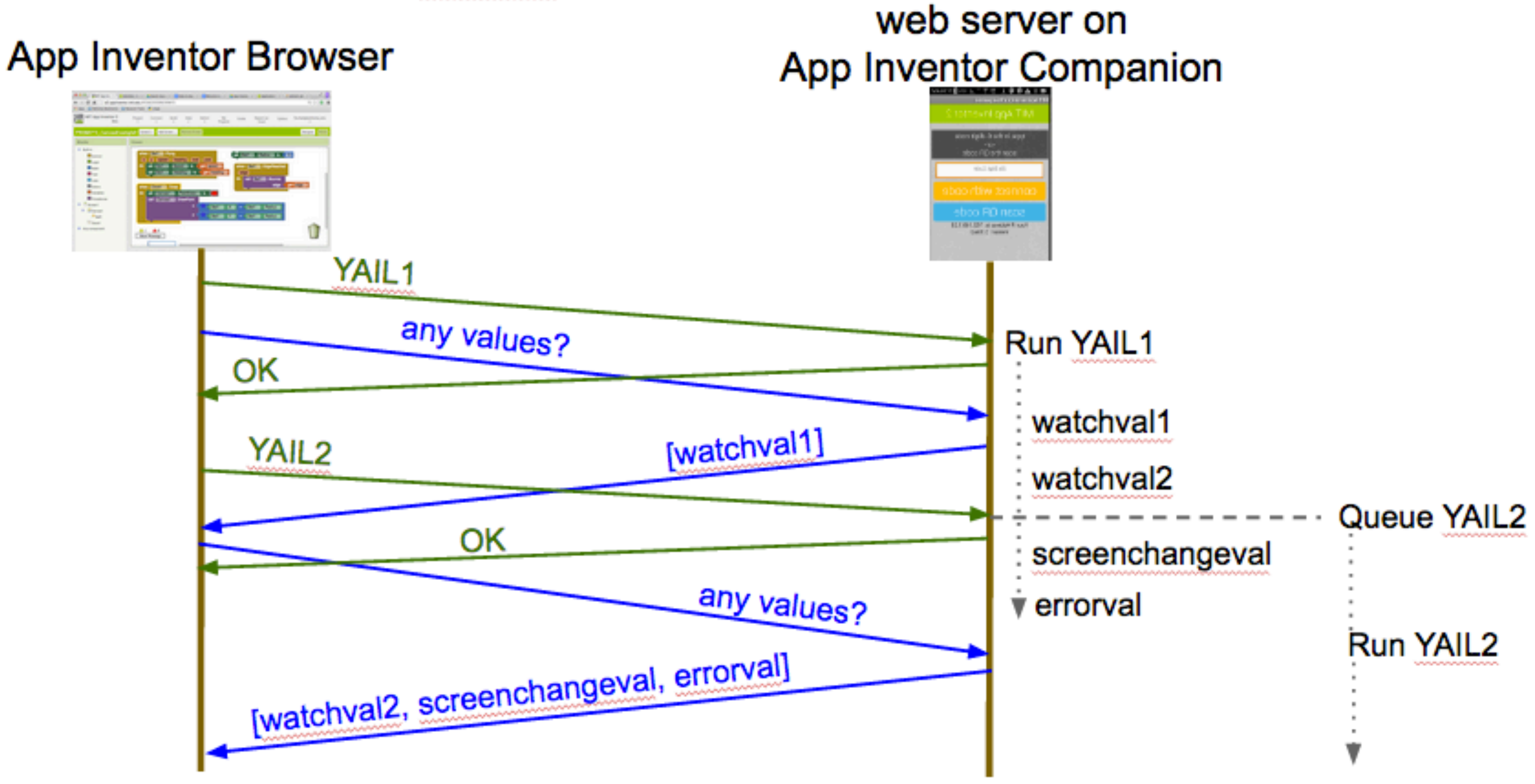
The TinkerBlocks project is anchored at Wellesley College and is led by Lyn Turbak. Meet our team members!

Here are some images from our work:

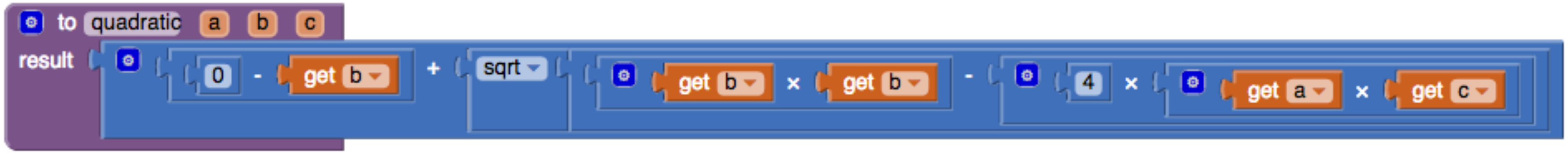
Establishing WiFi communication

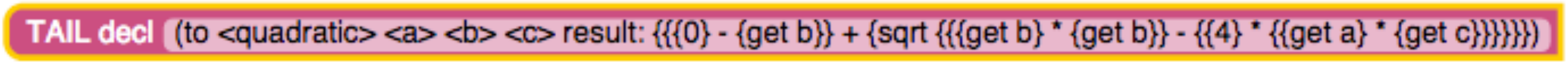


Two-way WiFi communication via HTTP



Another Conversion Example

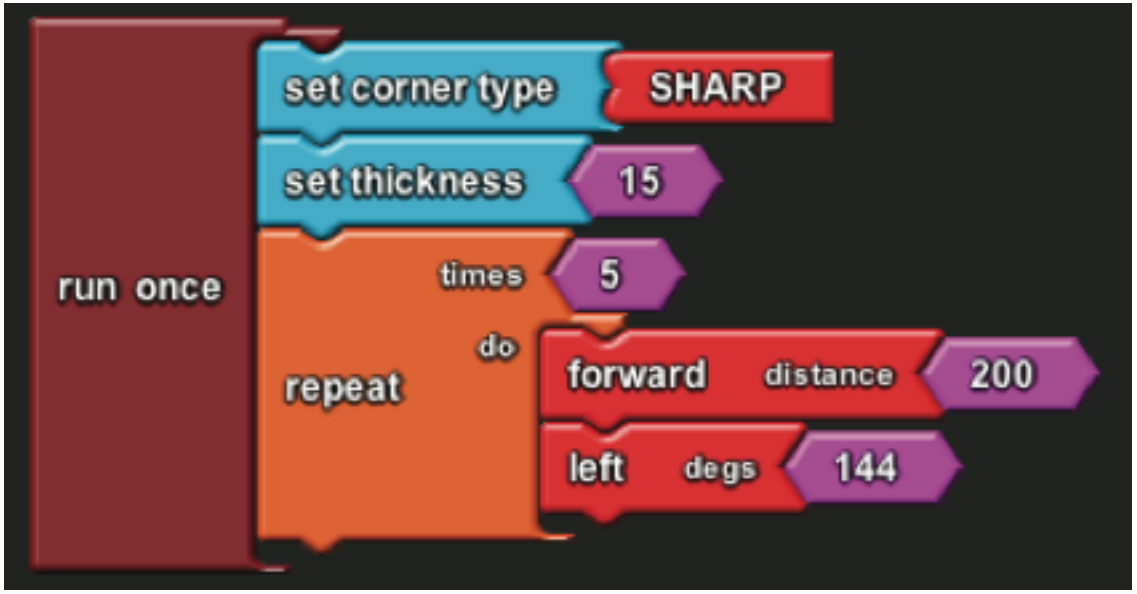




TAIL decl (to <quadratic> <a> <b> <c> result: {{{0} - {get b}} + {sqrt {{{get b} * {get b}} - {{4} * {{get a} * {get c}}}}}})

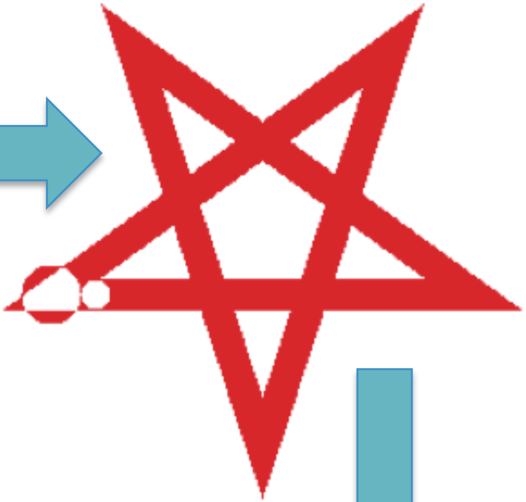
TurtleBlocks

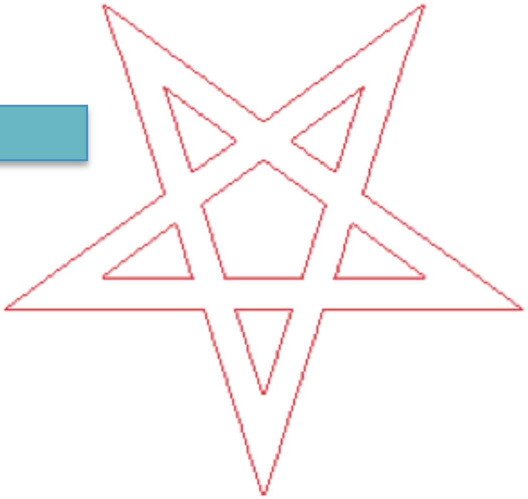
TurtleBlocks program



```
run once
  set corner type  SHARP
  set thickness  15
  repeat  times  5
          do
            forward  distance  200
            left  degs  144
```

turtle drawing



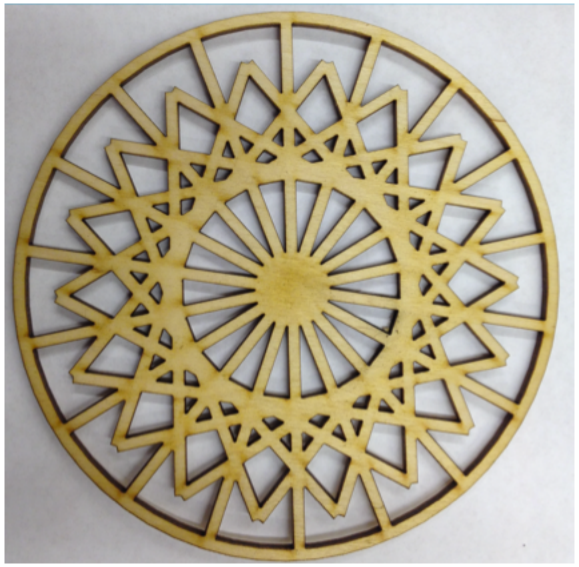
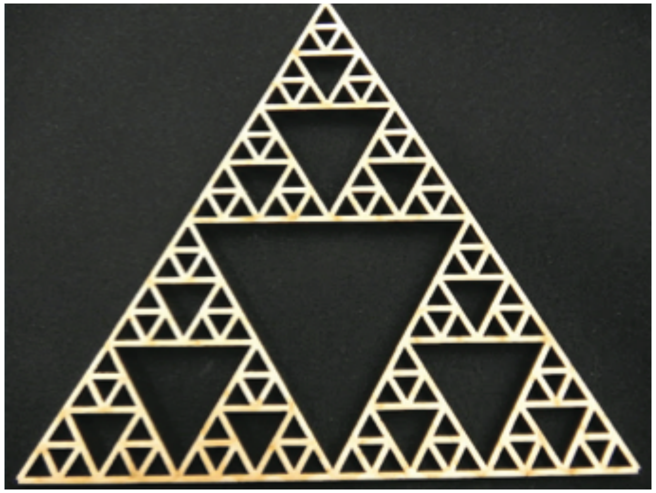
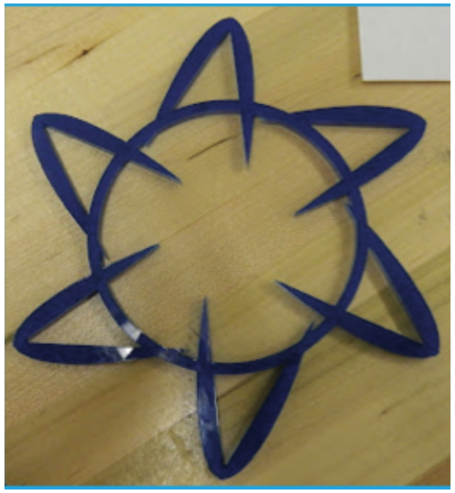
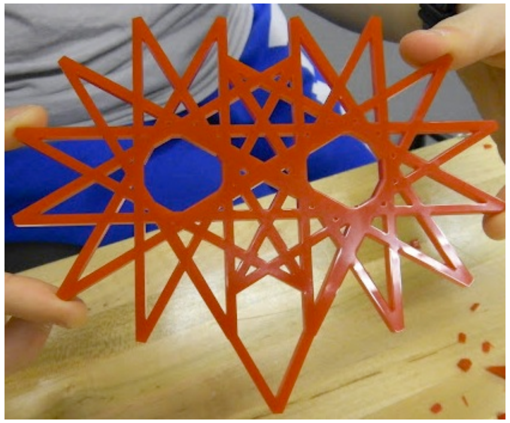
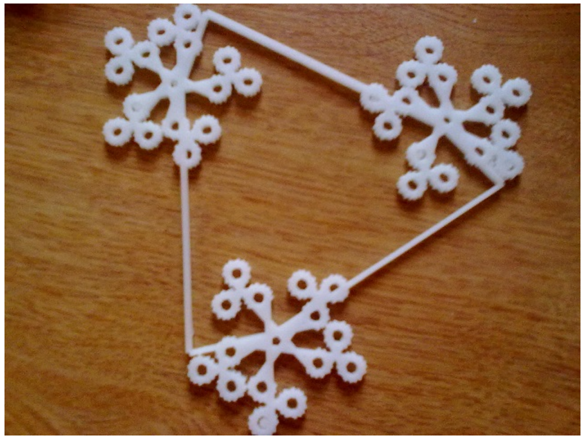


drawing boundary

cardstock

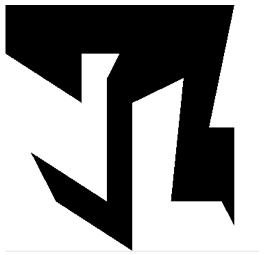
acrylic

TurtleBlocks Artifacts

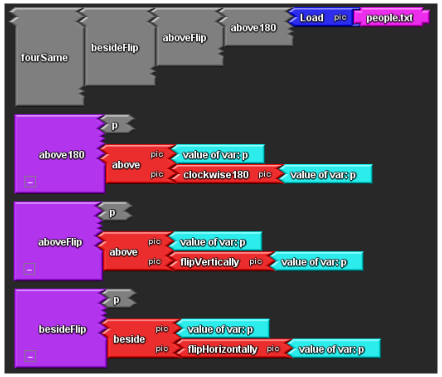






PictureBlocks: Sketching & Engraving

user sketch

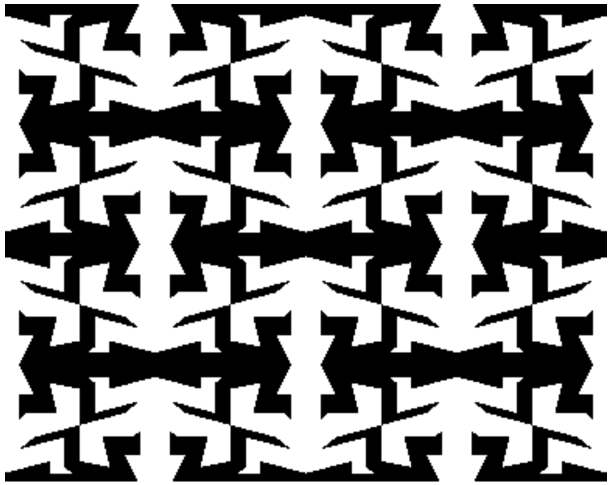


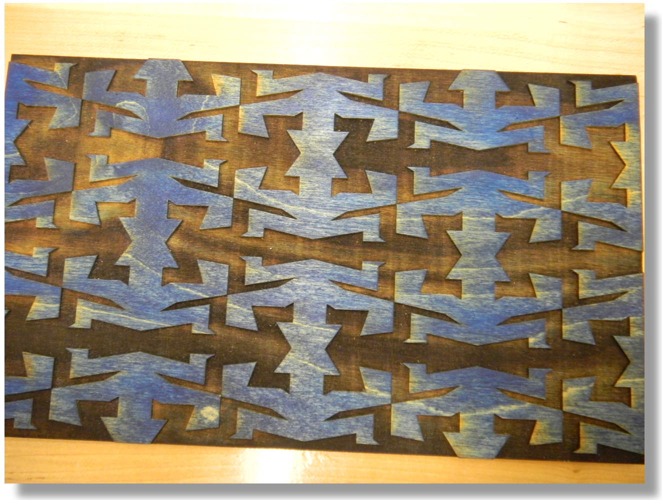
PictureBlocks program



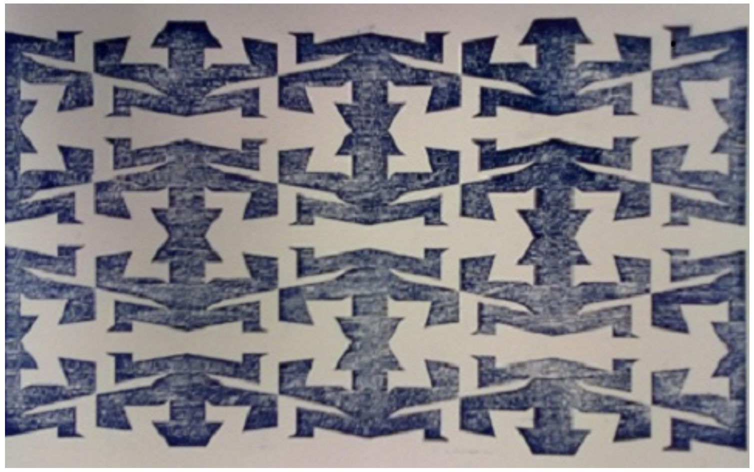
```
fourSame besideFlip aboveFlip above180 Load pic people.txt

above180 p
  above pic value of var: p
        pic clockwise180 pic value of var: p

aboveFlip p
  above pic value of var: p
        pic flipVertically pic value of var: p

besideFlip p
  beside pic value of var: p
         pic flipHorizontally pic value of var: p
```

resulting picture



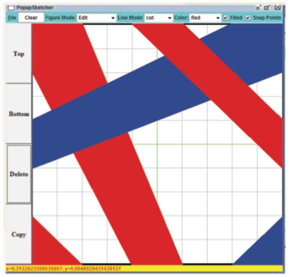


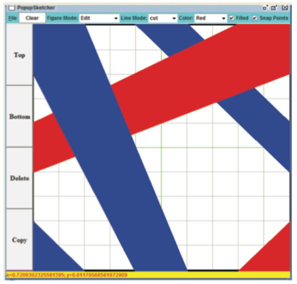
wood engraving

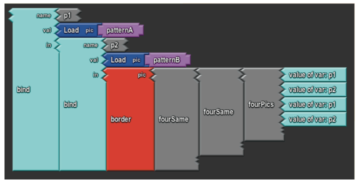


print from engraving

PictureBlocks: Engraving + Cutting


PopupSketcher
File
Clear
Figure Mode:
Edit
Line Mode:
cut
Color:
Red
Filled
Snap Points
Top
Bottom
Delete
Copy


PopupSketcher
File
Clear
Figure Mode:
Edit
Line Mode:
cut
Color:
Red
Filled
Snap Points
Top
Bottom
Delete
Copy


bind
name
p1
val
Load
pic
patternA
in
bind
name
p2
val
Load
pic
patternB
in
border
pic
fourSame
fourSame
fourPics
value of var: p1
value of var: p2
value of var: p1
value of var: p2